

**ĐẠI HỌC BÁCH KHOA HÀ NỘI**  
**KHOA TOÁN - TIN**



**CUSTOMER SEGMENTATION**

**HỆ HỖ TRỢ RA QUYẾT ĐỊNH**

Chuyên ngành: Hệ thống thông tin quản lý

Nhóm 32

Giảng viên hướng dẫn: Ths. Lê Hải Hà

Sinh viên thực hiện: Lê Đức Việt 20216968

Nguyễn Phúc Hưng 20216934

Mã lớp học: 150330

HÀ NỘI – 2024

## LỜI CẢM ƠN

Lời đầu tiên, em xin gửi lời cảm ơn tới Ths. Lê Hải Hà, giảng viên Bộ môn Toán tin, Khoa Toán Tin, Đại học Bách khoa Hà Nội đã tận tình hướng dẫn và chỉ bảo trong quá trình thực hiện bài tập lớn môn học.

Do còn nhiều thiếu sót về kiến thức, kỹ năng cũng như kinh nghiệm thực tế, bài báo cáo của em không thể tránh khỏi những sai sót. Em rất mong nhận được những ý kiến đóng góp, phê bình của quý thầy để bài báo cáo này ngày càng hoàn thiện hơn. Quá trình thực hiện bài tập lớn lần này đã làm cho em nhận ra giá trị của sự tự giác và chủ động trong công việc. Em hiểu rằng chúng ta cần tự tìm hiểu, nghiên cứu và rèn luyện kỹ năng mình còn thiếu sót. Em rất hy vọng rằng trong các dự án nghiên cứu tiếp theo, em sẽ vẫn được hướng dẫn và đồng hành cùng thầy. Một lần nữa, em xin bày tỏ lòng biết ơn sâu sắc và cảm kích đến sự tận tâm và hỗ trợ đặc lực của thầy, đã giúp em vượt qua khó khăn và tiến bộ trong quá trình học tập.

Em xin chân thành cảm ơn!

# Mục lục

Danh sách bảng	4
Danh sách hình	5
Giới thiệu	8
<b>Chương 1: PHÁT BIỂU BÀI TOÁN</b>	<b>10</b>
1.1 Mô tả bài toán . . . . .	10
1.2 Đầu vào . . . . .	10
1.3 Yêu cầu xử lí . . . . .	11
1.4 Đầu ra . . . . .	13
<b>Chương 2: KHAI PHÁ DỮ LIỆU</b>	<b>14</b>
2.1 Thống kê dữ liệu mẫu . . . . .	14
2.2 Tiền xử lý dữ liệu . . . . .	16
2.2.1 Xử lí giá trị khuyết thiếu . . . . .	16
2.2.2 Xử lí giá trị trùng lặp . . . . .	17
2.2.3 Xử lí các đơn hàng bị hủy hoặc đổi trả . . . . .	18
2.2.4 Loại bỏ đơn giá rỗng . . . . .	20
2.2.5 Xử lí các giá trị thời gian . . . . .	21
2.2.6 Xử lí dữ liệu cột Mô tả ( Description ) . . . . .	22
2.2.7 Loại bỏ giá trị ngoại lai . . . . .	23
2.2.8 Chuẩn hóa dữ liệu . . . . .	24
2.3 Phân tích RFM . . . . .	25
2.3.1 Recency (R) . . . . .	26
2.3.2 Frequency (F) . . . . .	27
2.3.3 Monetary (M) . . . . .	27

2.4 Tạo và luyện và đánh giá mô hình . . . . .	28
2.4.1 Xây dựng mô hình phân cụm với thuật toán K-Means . . . . .	28
2.4.2 Xây dựng mô hình phân cụm với thuật toán DBSCAN . . . . .	34
2.5 Đánh giá và so sánh . . . . .	36
2.6 Ứng dụng xây dựng mô hình hệ thống đề xuất dựa trên các phân khúc khách hàng . . . . .	37
<b>Chương 3: KẾT QUẢ</b>	<b>40</b>
3.1 Diễn giải các kết quả . . . . .	40
3.2 Khả năng ứng dụng trong tương lai . . . . .	42

# Danh sách bảng

2.1 Evaluation Metrics . . . . .	34
2.2 So sánh . . . . .	36

# Danh sách hình vẽ

1.1 Bộ dữ liệu . . . . .	11
2.1 Mô tả sơ lược bộ dữ liệu . . . . .	14
2.2 Mô tả sơ lược bộ dữ liệu . . . . .	15
2.3 Kiểm tra giá trị khuyết thiếu . . . . .	17
2.4 Xử lý giá trị khuyết thiếu . . . . .	17
2.5 Giá trị trùng lặp . . . . .	18
2.6 Bộ dữ liệu sau khi loại bỏ giá trị trùng lặp . . . . .	18
2.7 Thống kê giao dịch bị hủy . . . . .	19
2.8 Xóa bỏ giá trị số lượng âm . . . . .	19
2.9 Biểu đồ hộp cho số lượng . . . . .	20
2.10 Biểu đồ hộp cho đơn giá . . . . .	20
2.11 Biểu đồ hộp cho đơn giá và số lượng sau khi xử lý . . . . .	21
2.12 Chuyển định dạng datetime . . . . .	21
2.13 Time Series Decomposition . . . . .	22
2.14 Các mô tả chứa các ký tự chữ thường . . . . .	23
2.15 Xử lý dữ liệu cột Mô tả . . . . .	23
2.16 Giá trị ngoại lai . . . . .	24
2.17 Loại bỏ giá trị ngoại lai . . . . .	24
2.18 Chuẩn hóa dữ liệu . . . . .	25
2.19 Ví dụ vài cột dữ liệu sau khi chuẩn hóa . . . . .	25
2.20 Tạo cột Recency . . . . .	26
2.21 Tạo cột Recency . . . . .	26
2.22 Tạo cột Frequency . . . . .	27
2.23 Tạo cột Frequency . . . . .	27

2.24Tạo cột Monetary . . . . .	28
2.25Tạo cột Monetary . . . . .	28
2.26Xác định tham số k . . . . .	28
2.27Biểu đồ độ chính xác với k . . . . .	29
2.28Xây dựng thuật toán xác định K với Silhouette . . . . .	30
2.29Biểu đồ độ chính xác với k . . . . .	30
2.30Xây dựng mô hình K-Means . . . . .	31
2.31Biểu đồ điểm 3D . . . . .	32
2.32Bộ dữ liệu được phân bổ theo các cụm . . . . .	32
2.33Hiệu chỉnh tham số . . . . .	33
2.34Hiệu chỉnh tham số . . . . .	33
2.35Xây dựng thuật toán DBSCAN . . . . .	34
2.36Biểu đồ điểm 3D biểu diễn phân cụm DBSCAN . . . . .	35
2.37Đánh giá thuật toán DBSCAN . . . . .	36
2.38Tiền xử lí và loại bỏ outlier . . . . .	38
2.39Xây dựng hệ thống đề xuất . . . . .	38
2.40Kết quả hệ thống đề xuất . . . . .	39
3.1 Bộ dữ liệu được phân bổ theo các cụm . . . . .	40

## Tóm tắt

Thấu hiểu khách hàng là mục tiêu hàng đầu của hầu hết các doanh nghiệp trong lĩnh vực kinh doanh sản phẩm, dịch vụ nói chung và lĩnh vực thương mại điện tử nói riêng. Muốn đạt được điều đó, nhà quản trị phải có khả năng phân chia khách hàng của mình vào từng nhóm riêng biệt và đưa ra chính sách chăm sóc phù hợp với nhu cầu của từng nhóm, cụ thể hơn là từng khách hàng. Việc làm này được gọi là phân khúc khách hàng (Customer Segmentation). Ngày nay, khoa học dữ liệu cùng các công cụ, kỹ thuật phân tích dữ liệu đã và đang phát triển rất nhanh chóng. Việc tận dụng nguồn dữ liệu khổng lồ từ hành vi mua hàng và nhân khẩu học của khách hàng, đồng thời ứng dụng các thuật toán, mô hình nhằm phân tích dữ liệu đó là điều hết sức quan trọng mà bất kỳ doanh nghiệp thương mại điện tử nào cũng phải nắm bắt nếu muốn đạt được thành công trong thời đại số.

Bài báo này đề xuất một mô hình kết hợp giữa phương pháp tính toán các giá trị RFM (Recency, Frequency, Monetary) và phân cụm bằng thuật toán học máy K-means (Machine Learning) để phân nhóm khách hàng trong lĩnh vực thương mại điện tử. Qua kiểm định chất lượng, nghiên cứu đã cho thấy tính hiệu quả và khả năng ứng dụng của phương pháp vào thực tiễn. Không chỉ đóng góp về mặt lý thuyết, mô hình này còn giúp các doanh nghiệp và nhà quản trị có thể đưa ra những quyết định chính xác hơn dựa trên dữ liệu, từ đó thiết lập các chiến dịch tiếp thị phù hợp cho từng phân khúc khách hàng, mang lại hiệu quả kinh tế và giữ chân được khách hàng.



# GIỚI THIỆU

Trong môi trường kinh doanh hiện đại, việc hiểu rõ khách hàng và đáp ứng nhu cầu của họ là yếu tố quyết định để doanh nghiệp tồn tại và phát triển. Một trong những phương pháp hiệu quả để đạt được điều này là sử dụng mô hình phân khúc khách hàng. Phân khúc khách hàng là quá trình chia thị trường thành các nhóm khách hàng khác nhau dựa trên các đặc điểm chung như hành vi tiêu dùng, nhu cầu và sở thích. Việc phân khúc giúp doanh nghiệp tối ưu hóa các chiến lược marketing, phát triển sản phẩm và dịch vụ phù hợp với từng nhóm khách hàng cụ thể, từ đó nâng cao sự hài lòng và trung thành của khách hàng.

Phân khúc khách hàng là nhóm các đối tượng khách hàng được phân chia theo từng đặc điểm cụ thể. Từng phân khúc khách hàng khác nhau sẽ có những đặc điểm, hành vi mua hàng khác nhau, ảnh hưởng đến chiến lược kinh doanh của doanh nghiệp. Do đó, tùy vào phân khúc khách hàng khác nhau, doanh nghiệp nên chuẩn bị nội dung, thông điệp phù hợp hơn trong việc tiếp cận khách hàng. Một chiến lược tiếp thị, bán hàng phù hợp với phân khúc, đối tượng khách hàng mục tiêu sẽ mang đến hiệu quả vượt trội trong hoạt động kinh doanh của doanh nghiệp.

Sự phát triển của công nghệ thông tin và khả năng thu thập, phân tích dữ liệu lớn đã mở ra nhiều cơ hội mới cho việc nghiên cứu và áp dụng mô hình phân khúc khách hàng. Dữ liệu khách hàng ngày càng phong phú và đa dạng, bao gồm các thông tin về hành vi mua sắm, tương tác trên mạng xã hội và phản hồi từ khách hàng. Việc sử dụng các phương pháp phân tích dữ liệu tiên tiến, như *machine learning* và *data mining*, cho phép doanh nghiệp xác định các đặc điểm và xu hướng tiềm ẩn trong dữ liệu, từ đó phân khúc khách hàng một cách chính xác và hiệu quả hơn.

Mục tiêu của đề tài là xây dựng một mô hình phân khúc khách hàng hiệu quả trong lĩnh vực thương mại điện tử, giúp doanh nghiệp nhận diện và hiểu rõ các nhóm khách hàng khác nhau. Mô hình này sẽ hỗ trợ doanh nghiệp trong

việc phát triển các chiến lược marketing, bán hàng và chăm sóc khách hàng phù hợp với từng phân khúc, từ đó tối ưu hóa hiệu quả kinh doanh và tăng cường sự hài lòng của khách hàng.

Đề tài này không chỉ đóng góp vào việc phát triển lý thuyết về phân khúc khách hàng mà còn mở ra những hướng nghiên cứu mới về ứng dụng các kỹ thuật phân tích dữ liệu trong quản lý và marketing. Việc kết hợp các phương pháp phân tích dữ liệu tiên tiến với lý thuyết marketing truyền thống sẽ tạo ra những mô hình phân khúc khách hàng hiệu quả và chính xác hơn.

# Chương 1 PHÁT BIỂU BÀI TOÁN

## 1.1 Mô tả bài toán

Bài toán phân khúc khách hàng dựa trên bộ dữ liệu về thương mại điện tử bao gồm 541,910 dòng với 8 cột. Bài tập này đề xuất một mô hình kết hợp giữa phương pháp tính toán các giá trị RFM (Recency, Frequency, Monetary) và phân cụm bằng thuật toán học máy K-means (Machine Learning) để phân khúc khách hàng trong lĩnh vực thương mại điện tử. Qua đó, ta thực hiện đánh giá tính hiệu quả của nghiên cứu này và khả năng ứng dụng của nó vào thực tiễn. Không chỉ đóng góp về mặt lý thuyết, mô hình này sẽ còn hỗ trợ các doanh nghiệp và nhà quản trị có thể đưa ra những quyết định chính xác hơn dựa trên dữ liệu, từ đó thiết lập các chiến dịch tiếp thị phù hợp cho từng phân khúc khách hàng, mang lại hiệu quả kinh tế và giữ chân được khách hàng.

## 1.2 Đầu vào

Thông thường, các tập dữ liệu thương mại điện tử là tài sản độc quyền và do đó rất khó tìm trong số các dữ liệu công khai có sẵn. Tuy nhiên, Kho lưu trữ Học máy UCI đã cung cấp tập dữ liệu này chứa các giao dịch thực tế từ năm 2010 và 2011. Tập dữ liệu được duy trì trên trang web của họ, nơi có thể được tìm thấy với tiêu đề "Bán lẻ trực tuyến".

Đây là một tập dữ liệu giao dịch chứa tất cả các giao dịch xảy ra từ ngày 01/12/2010 đến ngày 09/12/2011 cho một công ty bán lẻ trực tuyến không có cửa hàng có trụ sở tại Vương quốc Anh và được đăng ký. Công ty chủ yếu bán các món quà độc đáo cho mọi dịp. Nhiều khách hàng của công ty là các nhà bán buôn.

Bộ dữ liệu bao gồm 541,910 dòng với 8 cột. Dưới đây là tổng quan chi tiết về từng cột:

- **InvoiceNo:** Kiểu dữ liệu: Chuỗi. Chứa số hóa đơn cho mỗi giao dịch,

trong đó mỗi số có thể đại diện cho nhiều mặt hàng được mua trong một giao dịch duy nhất.

- **StockCode:** Kiểu dữ liệu: Chuỗi. Đại diện cho mã sản phẩm của từng mặt hàng.
- **Description:** Kiểu dữ liệu: Chuỗi. Chứa mô tả sản phẩm. Một số mục bị thiếu, với 540,455 giá trị không rỗng (đầy đủ 99,73%).
- **Quantity:** Kiểu dữ liệu: Số nguyên. Chỉ ra số lượng sản phẩm được mua trong mỗi giao dịch.
- **InvoiceDate:** Kiểu dữ liệu: Ngày giờ. Ghi lại ngày và giờ của mỗi giao dịch.
- **UnitPrice:** Kiểu dữ liệu: Số thực. Đại diện cho giá mỗi đơn vị của từng mặt hàng.
- **CustomerID:** Kiểu dữ liệu: Số thực. Chứa mã khách hàng cho mỗi giao dịch. Cột này có nhiều giá trị bị thiếu, với chỉ 406,829 giá trị không rỗng (đầy đủ 75,07%).
- **Country:** Kiểu dữ liệu: Chuỗi. Ghi lại quốc gia nơi mỗi giao dịch diễn ra.

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
...	...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	12/9/2011 12:50	0.85	12680.0	France
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	12/9/2011 12:50	2.10	12680.0	France
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	12/9/2011 12:50	4.15	12680.0	France
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	12/9/2011 12:50	4.15	12680.0	France
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	12/9/2011 12:50	4.95	12680.0	France

541909 rows x 8 columns

Hình 1.1: Bộ dữ liệu

## 1.3 Yêu cầu xử lí

### Nhận định tiền xử lí:

- Để thực hiện quá trình huấn luyện, trước hết ta cần phải giải quyết các giá trị bị thiếu: Từ cái nhìn tổng quan sơ bộ, dường như có các giá trị bị thiếu trong các cột Description và CustomerID cần được giải quyết.

- Loại bỏ các giá trị trùng lặp.
- Chuyển đổi InvoiceDate sang định dạng ngày giờ.
- Cột InvoiceDate nên được chuyển đổi thành định dạng ngày giờ, điều này sẽ tạo điều kiện thuận lợi cho việc phân tích chuỗi thời gian tiếp theo.
- Nhiều giao dịch mỗi khách hàng: Chúng tôi cũng nhận thấy rằng một khách hàng có thể có nhiều giao dịch.
- Loại bỏ các giá trị ngoại lai có thể làm ảnh hưởng đến kết quả quá trình huấn luyện, các giá trị phi thực tế như Số lượng, giá bán lẻ âm...

## **Phân tích bộ dữ liệu theo phương pháp RFM**

Phân tích RFM là một mô hình phân tích khách hàng trong lĩnh vực marketing và quản lý quan hệ khách hàng. Mô hình RFM đánh giá các khía cạnh quan trọng của hành vi mua hàng của khách hàng dựa trên 3 yếu tố:

- Thời gian gần nhất khách mua hàng (Recency),
- Tần suất mua hàng của họ (Frequency)
- Số tiền khách hàng chi tiêu (Monetary).

Đây là một kỹ thuật phân khúc khách hàng cho phép các nhà tiếp thị, các doanh nghiệp nhắm đến các nhóm khách hàng cụ thể với các thông điệp phù hợp hơn với hành vi của họ, qua đó làm tăng đáng kể cơ hội bán hàng. Dựa trên ba yếu tố này, khách hàng được phân loại thành các nhóm khác nhau, như khách hàng quan trọng, khách hàng tiềm năng, khách hàng mất dần và khách hàng không hoạt động... Mô hình RFM giúp doanh nghiệp dự đoán tiềm năng của từng phân khúc khách hàng, sắp xếp các phân khúc khách hàng theo thứ tự ưu tiên để phân bổ nguồn lực một cách hiệu quả và tiết kiệm chi phí. Điều này giúp tối ưu hóa ngân sách chiến lược tiếp thị, cải thiện tương tác với khách hàng và tăng doanh thu. tác.

## **Phân cụm bằng các mô hình học máy**

- Bài tập này sẽ tiến hành phân cụm khách hàng dựa trên các phương pháp học máy khác nhau: K-Means, Cure, DBSCAN.
- Các nhóm (clusters) khách hàng được xác định bằng các thuật toán học máy trên dựa theo các giá trị RFM đã được chuẩn hóa.

- Mỗi khách hàng được gán vào một cụm (cluster) nhất định, biểu thị bằng nhãn cụm.
- Từ đó đưa ra đánh giá, so sánh rồi cuối cùng đưa ra kết luận.

## Kết luận

Đưa ra kết luận và đặt tên về các nhóm khách hàng đã tìm được từ các phương pháp trên.

## 1.4 Đầu ra

- Mô tả và phân tích các đặc điểm của từng cụm khách hàng, bao gồm số lượng khách hàng trong mỗi cụm, giá trị trung bình của các chỉ số RFM trong mỗi cụm.
- Các biểu đồ trực quan hóa như biểu đồ phân tán (scatter plots) để minh họa sự phân bố của các khách hàng trong không gian RFM, được phân màu theo cụm.
- Các phân khúc khách hàng cụ thể được xác định như:
  - Khách hàng lý tưởng (Ideal Customers): Khách hàng có điểm RFM cao nhất.
  - Khách hàng tiềm năng trung thành (Potential Loyalists): Khách hàng có tần suất và giá trị tiền tệ cao nhưng mức độ gần đây thấp.
  - Khách hàng có nguy cơ rời bỏ (At-risk Customers): Khách hàng có mức độ gần đây và tần suất thấp nhưng giá trị tiền tệ cao.
  - Khách hàng mới (New Customers): Khách hàng vừa mới mua hàng lần đầu.

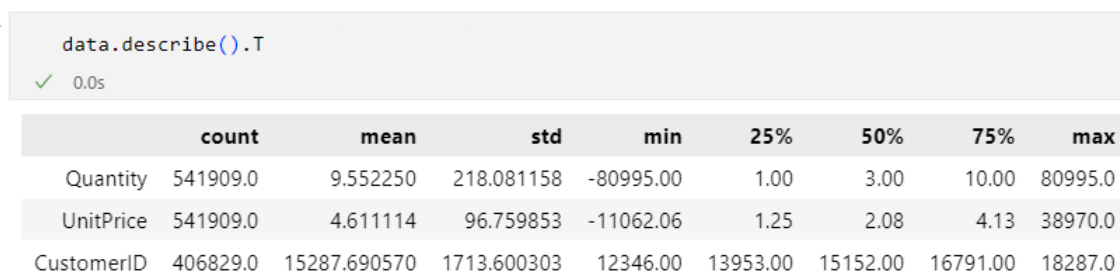
## Chương 2 KHAI PHÁ DỮ LIỆU

### 2.1 Thống kê dữ liệu mẫu

Thống kê từng cột dữ liệu:

- **Số lượng (Quantity):**

- **Số lượng trung bình** của sản phẩm trong một giao dịch khoảng 9.55.
- Số lượng có phạm vi rộng, với giá trị tối thiểu là -80995 và giá trị tối đa là 80995. Các giá trị **âm chỉ các đơn hàng bị trả lại hoặc hủy bỏ**, cần được xử lý một cách thích hợp.
- **Độ lệch chuẩn** khá lớn, cho thấy *sự phân tán lớn trong dữ liệu*. **Sự hiện diện của các giá trị ngoại lai** được chỉ ra bởi sự khác biệt lớn giữa giá trị tối đa và giá trị phần trăm thứ 75.
- **Số lượng** có độ lệch chuẩn rất cao (khoảng 218.08) so với giá trị trung bình, cho thấy một phạm vi rộng của các giá trị và có thể có các giá trị ngoại lai.
- 25% giá trị Số lượng nằm ở mức hoặc thấp hơn 1, và 75% ở mức hoặc thấp hơn 10, có nghĩa là hầu hết các giao dịch liên quan đến số lượng nhỏ.



```
data.describe().T
```

✓ 0.0s

	count	mean	std	min	25%	50%	75%	max
Quantity	541909.0	9.552250	218.081158	-80995.00	1.00	3.00	10.00	80995.0
UnitPrice	541909.0	4.611114	96.759853	-11062.06	1.25	2.08	4.13	38970.0
CustomerID	406829.0	15287.690570	1713.600303	12346.00	13953.00	15152.00	16791.00	18287.0

Hình 2.1: Mô tả sơ lược bộ dữ liệu

- **Giá bán lẻ (UnitPrice):**

- **Giá bán lẻ trung bình** của sản phẩm khoảng 4.61.
- Giá bán lẻ cũng có phạm vi rộng, từ -11062.06 đến 38970, điều này **cho thấy sự hiện diện của lỗi hoặc nhiễu trong dữ liệu**, vì giá trị âm là không hợp lý.
- Tương tự như cột Số lượng, sự **hiện diện của các giá trị ngoại lai** được chỉ ra bởi sự khác biệt lớn giữa giá trị tối đa và giá trị phần trăm thứ 75.
- **Giá bán lẻ** cũng có độ lệch chuẩn cao (khoảng 96.76), cho thấy sự biến động đáng kể trong giá sản phẩm.
- 25% giá trị Giá bán lẻ nằm ở mức hoặc thấp hơn 1.25, và 75% ở mức hoặc thấp hơn 4.13, điều này cho thấy hầu hết các sản phẩm tương đối rẻ.

- **Mã Khách hàng (CustomerID):**

- Có 406829 giá trị không null, chỉ ra rằng có giá trị bị thiếu trong bộ dữ liệu cần được xử lý.

	count	unique	top	freq
InvoiceNo	541909	25900	573585	1114
StockCode	541909	4070	85123A	2313
Description	540455	4223	WHITE HANGING HEART T-LIGHT HOLDER	2369
InvoiceDate	541909	23260	10/31/2011 14:41	1114
Country	541909	38	United Kingdom	495478

Hình 2.2: Mô tả sơ lược bộ dữ liệu

- **Số hóa đơn (InvoiceNo):**

- Có 25900 số hóa đơn duy nhất, chỉ ra 25900 giao dịch riêng biệt.
- Số hóa đơn phổ biến nhất là 573585, xuất hiện 1114 lần, có thể đại diện cho một giao dịch lớn hoặc một đơn hàng với nhiều mặt hàng.

- **Mã hàng (StockCode):**

- Có 4070 mã hàng duy nhất đại diện cho các sản phẩm khác nhau.
- Mã hàng phổ biến nhất là 85123A, xuất hiện 2313 lần trong bộ dữ liệu.



- **Mô tả (Description):**

- Có 4223 mô tả sản phẩm duy nhất.
- Có 4223 mô tả sản phẩm duy nhất trong khi mã hàng duy nhất là 4070. Điều này có thể chỉ ra rằng chúng ta có nhiều mô tả cho cùng một Mã hàng.
- Mô tả sản phẩm phổ biến nhất là "WHITE HANGING HEART T-LIGHT HOLDER", xuất hiện 2369 lần.
- Có một số giá trị thiếu trong cột này cần được xử lý.

- **Quốc gia (Country):**

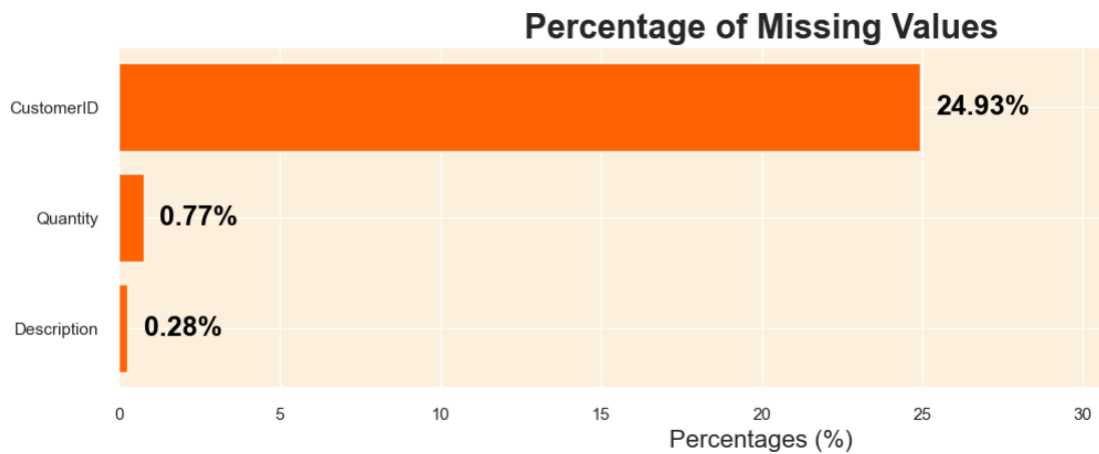
- Các giao dịch đến từ 38 quốc gia khác nhau, với phần lớn các giao dịch xuất phát từ Vương quốc Anh.

Từ phân tích này, rõ ràng có những giá trị cực đoan (cả cao và thấp) trong cả **Số lượng** và **Giá bán lẻ**, có thể là các giá trị ngoại lai hoặc lỗi. Những giá trị này có thể ảnh hưởng đến **trung bình** và **độ lệch chuẩn**. Chúng ta sẽ xem xét các trường hợp cực đoan này để xác định xem chúng có chính xác không và hiểu ảnh hưởng của chúng đến phân tích. Đối với bộ dữ liệu này, ta còn có thể xem xét bối cảnh của các giao dịch để quyết định cách xử lý chúng cho các phân tích trong tương lai.

## 2.2 Tiền xử lý dữ liệu

### 2.2.1 Xử lý giá trị khuyết thiếu

Kiểm tra giá trị khuyết thiếu:



Hình 2.3: Kiểm tra giá trị khuyết thiếu

Loại bỏ các dòng chứa giá trị khuyết thiếu trong cột CustomerID:

```
data = data.dropna(subset = ["CustomerID"])
```

Sau khi loại bỏ các giá trị khuyết thiếu ta thu được tập dữ liệu mới bao gồm 406829 dòng, 8 cột.

```
data.isnull().mean() * 100
```

InvoiceNo	0.0
StockCode	0.0
Description	0.0
Quantity	0.0
InvoiceDate	0.0
UnitPrice	0.0
CustomerID	0.0
Country	0.0

```
dtype: float64
```

```
data.shape
```

```
(406829, 8)
```

Hình 2.4: Xử lý giá trị khuyết thiếu

## 2.2.2 Xử lý giá trị trùng lặp

Thực hiện kiểm tra giá trị trùng lặp:

```
duplicates = data.duplicated()
```

*duplicates.sum()*

InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
517	536409	21866	UNION JACK FLAG LUGGAGE TAG	1	12/1/2010 11:45	1.25	17908.0 United Kingdom
527	536409	22866	HAND WARMER SCOTTY DOG DESIGN	1	12/1/2010 11:45	2.10	17908.0 United Kingdom
537	536409	22900	SET 2 TEA TOWELS I LOVE LONDON	1	12/1/2010 11:45	2.95	17908.0 United Kingdom
539	536409	22111	SCOTTIE DOG HOT WATER BOTTLE	1	12/1/2010 11:45	4.95	17908.0 United Kingdom
555	536412	22327	ROUND SNACK BOXES SET OF 4 SKULLS	1	12/1/2010 11:49	2.95	17920.0 United Kingdom
...	...	...	...	...	...	...	...
541675	581538	22068	BLACK PIRATE TREASURE CHEST	1	12/9/2011 11:34	0.39	14446.0 United Kingdom
541689	581538	23318	BOX OF 6 MINI VINTAGE CRACKERS	1	12/9/2011 11:34	2.49	14446.0 United Kingdom
541692	581538	22992	REVOLVER WOODEN RULER	1	12/9/2011 11:34	1.95	14446.0 United Kingdom
541699	581538	22694	WICKER STAR	1	12/9/2011 11:34	2.10	14446.0 United Kingdom
541701	581538	23343	JUMBO BAG VINTAGE CHRISTMAS	1	12/9/2011 11:34	2.08	14446.0 United Kingdom

5225 rows × 8 columns

Hình 2.5: Giá trị trùng lặp

# Loại bỏ giá trị trùng lặp

```
data = data.drop_duplicates()
```

# Đánh lại thứ tự

```
data.reset_index(drop=True, inplace=True)
```

InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID
0	536365	85123A WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850.0
1	536365	71053 WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850.0
2	536365	84406B CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850.0
3	536365	84029G KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850.0
4	536365	84029E RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850.0

**data.shape**  
 ✓ 0.0s  
 (401604, 8)

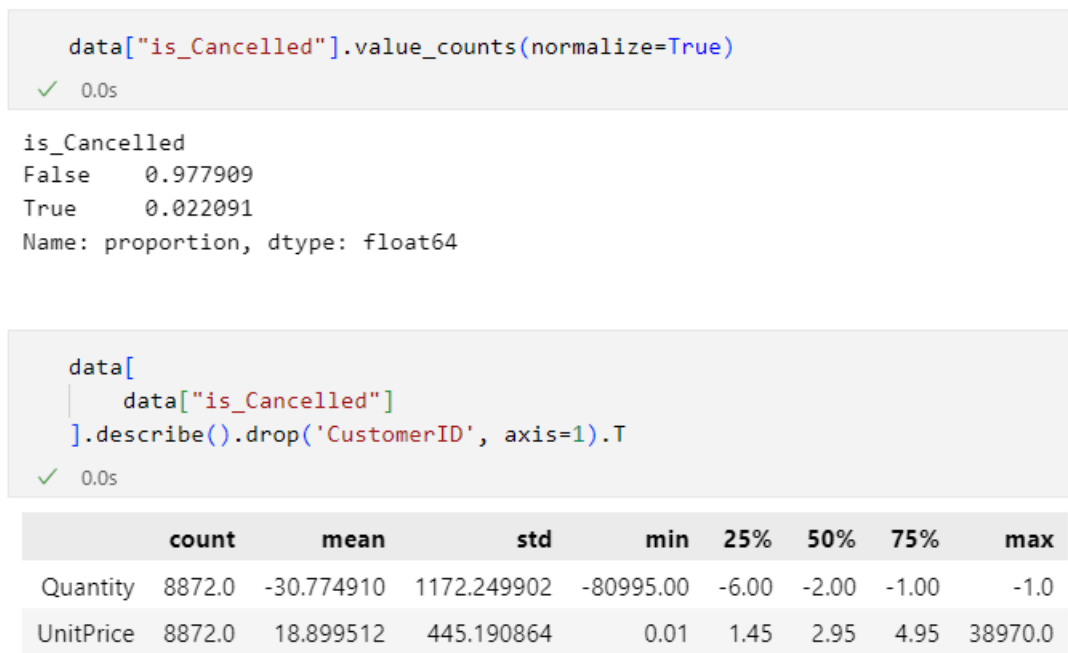
Hình 2.6: Bộ dữ liệu sau khi loại bỏ giá trị trùng lặp

-> Sau khi loại bỏ giá trị trùng lặp, ta thu được còn 401604 dòng dữ liệu.

### 2.2.3 Xử lý các đơn hàng bị hủy hoặc đổi trả

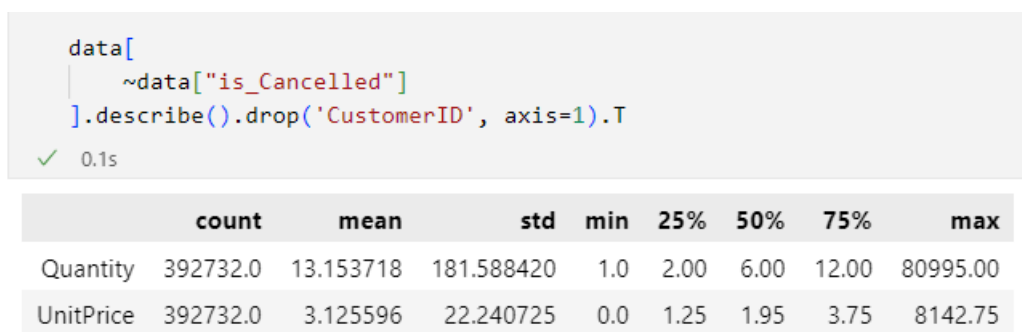
Trong quá trình xem xét ban đầu của tập dữ liệu, chúng tôi đã quan sát thấy rằng cột "Số lượng" chứa các giá trị âm. Những con số âm này có thể chỉ ra các mặt hàng bị trả lại hoặc đơn đặt hàng bị hủy. Sơ bộ, có vẻ như các mục này tương quan với các số hóa đơn bắt đầu bằng chữ 'C', có thể là một quy ước được sử dụng để biểu thị các đơn hàng bị hủy.

Tỷ lệ phần trăm của các giao dịch bị hủy trong tập dữ liệu là: 2.21

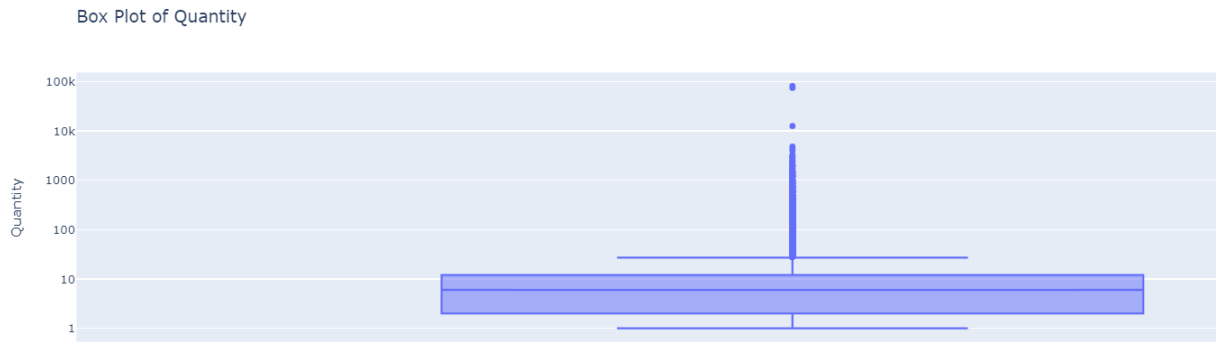


Hình 2.7: Thống kê giao dịch bị hủy

Thực hiện xóa bỏ giá trị âm:



Hình 2.8: Xóa bỏ giá trị số lượng âm

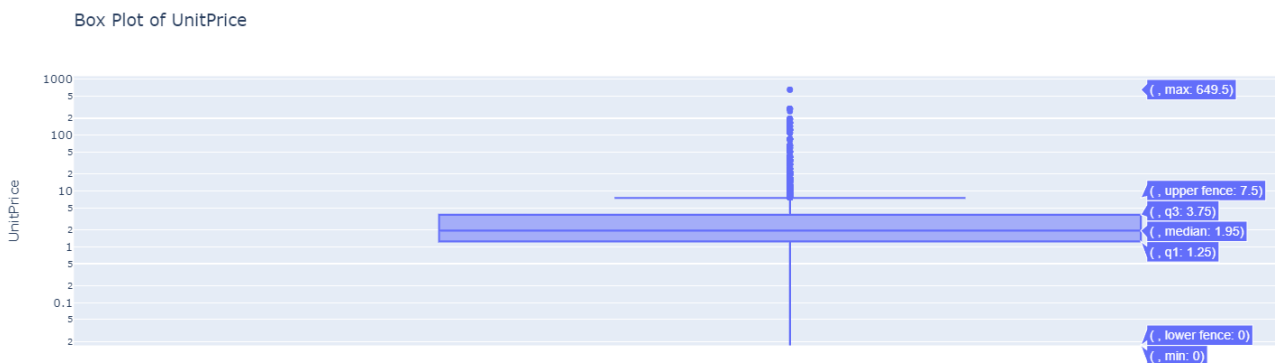


Hình 2.9: Biểu đồ hộp cho số lượng

#### 2.2.4 Loại bỏ đơn giá rỗng

Giá trị đơn giá tối thiểu bằng không. Điều này cho thấy có một số giao dịch mà đơn giá là không, có thể là do mặt hàng miễn phí hoặc lỗi nhập dữ liệu.

Do đó ta tiến hành loại bỏ các đơn giá có trị bằng không, để thuận lợi cho quá trình huấn luyện sau này.



Hình 2.10: Biểu đồ hộp cho đơn giá

- Các đơn giá được trải rộng trên nhiều mức độ lớn, với phần lớn dữ liệu tập trung ở đầu thấp của thang giá. Điều này là điển hình trong thương mại điện tử, nơi nhiều mặt hàng có giá tương đối rẻ, chỉ có một vài mặt hàng có giá cao.

- Giá trị ngoại lai: Có các giá trị ngoại lai ở đầu cao hơn, như được chỉ ra bởi các điểm phía trên râu trên. Trong bối cảnh của thang đo logarit, các giá trị ngoại lai này có giá đắt hơn theo cấp số nhân so với phần lớn các mặt hàng.

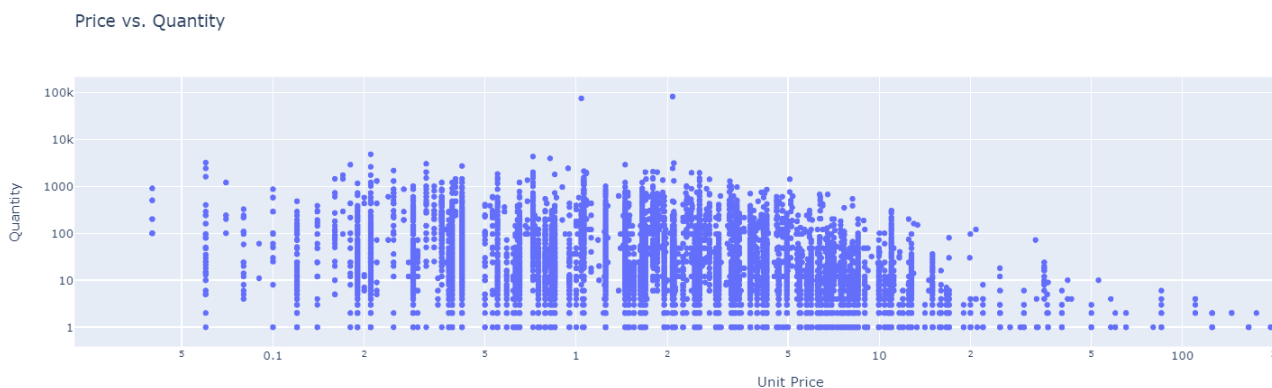
- Giá ở mức thấp: Ranh giới dưới nằm ở giá trị không tối thiểu, cho thấy không có các giá trị ngoại lai cực thấp. Tất cả các điểm dữ liệu đều nằm trong một phạm vi giá hợp lý trên mức không.

- Tính khả dụng: Biểu đồ hộp gợi ý rằng nền tảng thương mại điện tử có

khả năng phục vụ một đối tượng rộng lớn, cung cấp các sản phẩm chủ yếu là phải chăng, với một vài tùy chọn cao cấp.

Lấy đơn giá có giá trị lớn hơn 0:

```
data = data[data["UnitPrice"] > 0]
```



Hình 2.11: Biểu đồ hộp cho đơn giá và số lượng sau khi xử lý

### 2.2.5 Xử lý các giá trị thời gian

Chuyển đổi cột InvoiceDate thành định dạng datetime và đặt lại chỉ mục:

```
data['InvoiceDate'] = pd.to_datetime(data['InvoiceDate'])
viz_data = data.set_index('InvoiceDate')
viz_data['Sales'] = viz_data['Quantity'] * viz_data['UnitPrice']
daily_sales = viz_data['Sales'].resample('D').sum()
```

InvoiceDate	UnitPrice	CustomerID	Country	Date
12/1/2010 8:26	2.55	17850.0	United Kingdom	2010-12-01 08:26:00
12/1/2010 8:26	3.39	17850.0	United Kingdom	2010-12-01 08:26:00
12/1/2010 8:26	2.75	17850.0	United Kingdom	2010-12-01 08:26:00
12/1/2010 8:26	3.39	17850.0	United Kingdom	2010-12-01 08:26:00
12/1/2010 8:26	3.39	17850.0	United Kingdom	2010-12-01 08:26:00

Hình 2.12: Chuyển định dạng datetime



Hình 2.13: Time Series Decomposition

### 2.2.6 Xử lý dữ liệu cột Mô tả ( Description )

Tất cả các mô tả đều ở dạng chữ in hoa, có thể đây là một định dạng chuẩn để nhập mô tả sản phẩm vào cơ sở dữ liệu. Tuy nhiên, nếu ta xét đến các sự không nhất quán và bất thường đã gặp trong tập dữ liệu cho đến nay, sẽ hợp lý nếu kiểm tra xem có các mô tả nào được nhập bằng chữ thường hoặc hỗn hợp các kiểu chữ hay không.

# Tìm mô tả có chứa ký tự chữ thường

```
lowercase_descriptions = df['Description'].unique()
lowercase_descriptions = [desc for desc in lowercase_descriptions if
any(char.islower() for char in desc)]
```

# In các mô tả chứa các ký tự chữ thường

```
print("The unique descriptions containing lowercase characters are:")
print("-"*60)
for desc in lowercase_descriptions:
    print(desc)
```

```

The unique descriptions containing lowercase characters are:
-----
3 TRADITIONAL BISCUIT CUTTERS SET
BAG 125g SWIRLY MARBLES
NUMBER TILE VINTAGE FONT No
BAG 250g SWIRLY MARBLES
POLYESTER FILLER PAD 40x40cm
POLYESTER FILLER PAD 45x45cm
FOLK ART GREETING CARD,pack/12
BAG 500g SWIRLY MARBLES
ESSENTIAL BALM 3.5g TIN IN ENVELOPE
POLYESTER FILLER PAD 30CMx30CM
POLYESTER FILLER PAD 45x30cm
FRENCH BLUE METAL DOOR SIGN No
Next Day Carriage
High Resolution Image
THE KING GIFT BAG 25x24x12cm
POLYESTER FILLER PAD 65CMx65CM
NUMBER TILE COTTAGE GARDEN No
FLOWERS HANDBAG blue and orange
POLYESTER FILLER PAD 60x40cm

```

Hình 2.14: Các mô tả chứa các ký tự chữ thường

- Loại bỏ các hàng mà mô tả chứa thông tin liên quan đến dịch vụ như "Next Day Carriage" và "High Resolution Image"
- Đối với các mô tả còn lại có kiểu chữ hỗn hợp, chuẩn hóa văn bản thành chữ in hoa để duy trì tính đồng nhất trong toàn bộ tập dữ liệu.

```

service_related_descriptions = ["Next Day Carriage", "High Resolution Image"]

# Calculate the percentage of records with service-related descriptions
service_related_percentage = df[df['Description'].isin(service_related_descriptions)].shape[0] / df.shape[0] * 100

# Print the percentage of records with service-related descriptions
print(f"The percentage of records with service-related descriptions in the dataset is: {service_related_percentage:.2f}%")

# Remove rows with service-related information in the description
df = df[~df['Description'].isin(service_related_descriptions)]

# Standardize the text to uppercase to maintain uniformity across the dataset
df['Description'] = df['Description'].str.upper()

```

✓ 0.1s

The percentage of records with service-related descriptions in the dataset is: 0.02%

Hình 2.15: Xử lý dữ liệu cột Mô tả

## 2.2.7 Loại bỏ giá trị ngoại lai

```

# Khởi tạo mô hình Isolation Forest với contamination = 0.05
model = IsolationForest(contamination=0.05, random_state=0)

```

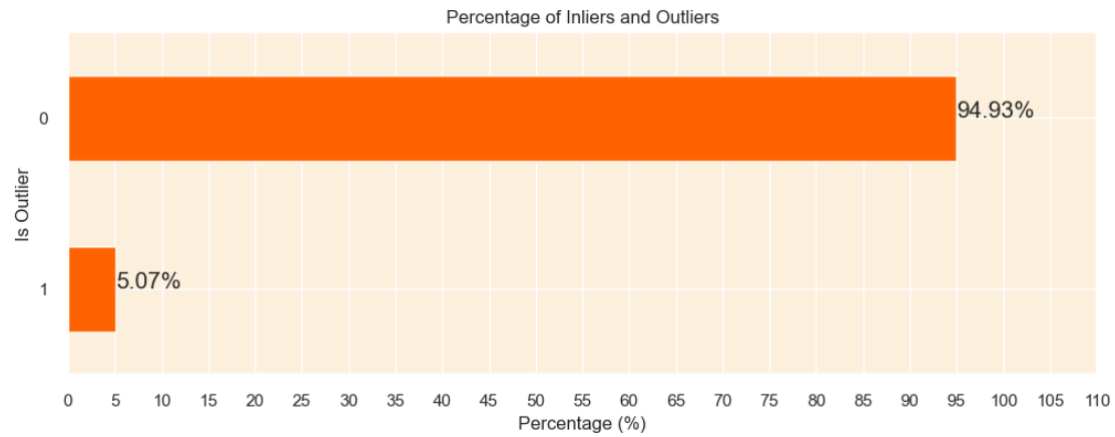
```

# Huấn luyện mô hình trên tập dữ liệu
customer_data['Outlier_Scores'] = model.fit_predict(customer_data.iloc[:,
1:].to_numpy())

```



```
# Tạo một cột mới để xác định ngoại lệ (1 cho inliers và -1 cho outliers)
customer_data['Is_Outlier'] = [1 if x == -1 else 0 for x in
customer_data['Outlier_Scores']]
```



Hình 2.16: Giá trị ngoại lai

```
# Tách các ngoại lệ để phân tích
outliers_data = customer_data[customer_data['Is_Outlier'] == 1]

# Loại bỏ các ngoại lệ khỏi tập dữ liệu chính
customer_data_cleaned = customer_data[customer_data['Is_Outlier'] == 0]

# Loại bỏ các cột 'Outlier_Scores' và 'Is_Outlier'
customer_data_cleaned = customer_data_cleaned.drop(columns=['Outlier_Scores', 'Is_Outlier'])

# Đặt lại chỉ số của dữ liệu đã dọn dẹp
customer_data_cleaned.reset_index(drop=True, inplace=True)
```

Hình 2.17: Loại bỏ giá trị ngoại lai

## 2.2.8 Chuẩn hóa dữ liệu

```

# Khởi tạo StandardScaler
scaler = StandardScaler()

# Danh sách các cột không cần phải chuẩn hóa
columns_to_exclude = ['CustomerID', 'Is_UK', 'Day_Of_Week']

# Danh sách các cột cần phải chuẩn hóa
columns_to_scale = customer_data_cleaned.columns.difference(columns_to_exclude)

# Sao chép tập dữ liệu đã được làm sạch
customer_data_scaled = customer_data_cleaned.copy()

# Áp dụng scaler vào các cột cần thiết trong tập dữ liệu
customer_data_scaled[columns_to_scale] = scaler.fit_transform(customer_data_scaled[columns_to_scale])

# Hiển thị một số dòng đầu tiên của dữ liệu đã được chuẩn hóa
customer_data_scaled.head()

```

Hình 2.18: Chuẩn hóa dữ liệu

Days_Since_Last_Purchase	Total_Transactions_x	Total_Products_Purchased_x
0.690570	-0.619661	0.246006
-0.811048	-0.619661	2.511062
-0.977894	0.411474	1.818623
-1.478433	-0.619661	0.940883
-1.812126	-0.619661	1.299293

Hình 2.19: Ví dụ vài cột dữ liệu sau khi chuẩn hóa

## 2.3 Phân tích RFM

RFM là viết tắt của Recency (Gần đây), Frequency (Tần suất) và Monetary (Giá trị tiền tệ), mỗi yếu tố tương ứng với một đặc điểm quan trọng của khách hàng. Phân tích RFM là một công cụ phân tích marketing, đánh giá các khía cạnh quan trọng của hành vi mua hàng của khách hàng dựa trên 3 yếu tố:

- Thời gian gần nhất khách mua hàng (Recency),
- Tần suất mua hàng của họ (Frequency)
- Số tiền khách hàng chi tiêu (Monetary).

Phân tích từng thành phần

- Recency (R): Giá trị này đo lường thời gian gần đây nhất khách hàng đã thực hiện mua hàng. Thời gian mua hàng càng gần đây, điểm Recency càng cao.

- Frequency (F): Giá trị này đánh giá tần suất mua hàng của khách hàng trong một khung thời gian cụ thể. Tần suất cao hơn chỉ ra khách hàng có sự tham gia cao hơn và có khả năng trung thành hơn.
- Monetary (M): Giá trị này phản ánh số tiền mà khách hàng chi tiêu cho các lần mua hàng trong một khoảng thời gian. Khách hàng chi tiêu nhiều hơn thường được coi là có giá trị hơn.

Trong phân tích RFM, khách hàng được gán một số thứ hạng (thường là từ 1 đến 5, 1 là thấp nhất, 5 là cao nhất) cho mỗi tham số RFM. Ba điểm số này cùng nhau được sử dụng để xác định các phân khúc khách hàng, chẳng hạn như khách hàng trung thành tiềm năng, khách hàng mới, khách hàng có nguy cơ, v.v. Ví dụ, một khách hàng có điểm RFM là 5-5-5 được coi là lý tưởng, đã mua hàng gần đây, thường xuyên và chi tiêu nhiều.

### 2.3.1 Recency (R)

```
# Tìm ngày mua hàng gần nhất cho mỗi khách hàng
customer_data = df.groupby('CustomerID')['InvoiceDay'].max().reset_index()
# Loại bỏ tất cả các hàng mà 'InvoiceDay' bằng với giá trị lớn nhất của 'InvoiceDay'

# Tìm ngày gần nhất trong toàn bộ tập dữ liệu
most_recent_date = df['InvoiceDay'].max()

# Chuyển đổi InvoiceDay thành kiểu datetime trước khi trừ
customer_data['InvoiceDay'] = pd.to_datetime(customer_data['InvoiceDay'])
most_recent_date = pd.to_datetime(most_recent_date)

# Tính số ngày kể từ lần mua hàng cuối cùng cho mỗi khách hàng
customer_data['Days_Since_Last_Purchase'] = (most_recent_date - customer_data['InvoiceDay']).dt.days

# Loại bỏ cột InvoiceDay
customer_data.drop(columns=['InvoiceDay'], inplace=True)
```

Hình 2.20: Tạo cột Recency

Ta sẽ tạo 1 cột mới, trả về ngày mua hàng gần nhất của mỗi khách hàng.

	CustomerID	Days_Since_Last_Purchase
0	12347.0	16
1	12348.0	7
2	12370.0	6
3	12377.0	3
4	12383.0	1

Hình 2.21: Tạo cột Recency

### 2.3.2 Frequency (F)

Ta sẽ tạo ra 2 cột mới:

Tổng giao dịch: Đây là đặc trưng biểu thị tổng số giao dịch mà một khách hàng đã thực hiện.

Tổng sản phẩm đã mua: Đây là đặc trưng cho biết tổng số lượng sản phẩm mà khách hàng đã mua trong tất cả các giao dịch.

```
# Tính tổng số giao dịch của mỗi khách hàng
total_transactions = df.groupby('CustomerID')['InvoiceNo'].nunique().reset_index()
total_transactions.rename(columns={'InvoiceNo': 'Total_Transactions'}, inplace=True)

# Tính tổng số sản phẩm đã mua của mỗi khách hàng
total_products_purchased = df.groupby('CustomerID')['Quantity'].sum().reset_index()
total_products_purchased.rename(columns={'Quantity': 'Total_Products_Purchased'}, inplace=True)

# Nối các đặc trưng mới vào customer_data dataframe
customer_data = pd.merge(customer_data, total_transactions, on='CustomerID')
customer_data = pd.merge(customer_data, total_products_purchased, on='CustomerID')

# Hiển thị một số dòng đầu tiên của customer_data dataframe
customer_data.head()
```

Hình 2.22: Tạo cột Frequency

	CustomerID	Days_Since_Last_Purchase	Total_Transactions_x	Total_Products_Purchased_x	Total_Transactions_y	Total_Products_Purchased_y
0	12347.0	16	1	319.0	1	319.0
1	12348.0	7	1	1248.0	1	1248.0
2	12370.0	6	2	964.0	2	964.0
3	12377.0	3	1	604.0	1	604.0
4	12383.0	1	1	751.0	1	751.0

Hình 2.23: Tạo cột Frequency

### 2.3.3 Monetary (M)

Ta sẽ tạo ra hai cột để biểu thị số tiền giao dịch của khách hàng:

Tổng chi tiêu: Đặc trưng này biểu thị tổng số tiền mà mỗi khách hàng đã chi tiêu. Nó được tính bằng tổng của tích của UnitPrice và Quantity cho tất cả các giao dịch mà khách hàng đã thực hiện.

Giá Trị giao dịch trung bình: Đặc trưng này được tính bằng cách chia Tổng Chi Tiêu cho Tổng Giao Dịch của mỗi khách hàng. Nó chỉ ra giá trị trung bình của một giao dịch mà khách hàng thực hiện.

```
# Tính tổng chi tiêu của mỗi khách hàng
df['Total_Spend'] = df['UnitPrice'] * df['Quantity']
total_spend = df.groupby('CustomerID')['Total_Spend'].sum().reset_index()

# Tính giá trị giao dịch trung bình cho mỗi khách hàng
average_transaction_value = total_spend.merge(total_transactions, on='CustomerID')
average_transaction_value['Average_Transaction_Value'] = average_transaction_value['Total_Spend'] / average_transaction_value['Total_Transactions']

# Nối các đặc trưng mới vào customer_data dataframe
customer_data = pd.merge(customer_data, total_spend, on='CustomerID')
customer_data = pd.merge(customer_data, average_transaction_value[['CustomerID', 'Average_Transaction_Value']], on='CustomerID')

# Hiển thị một số dòng đầu tiên của customer_data dataframe
customer_data.head()
```

Hình 2.24: Tạo cột Monetary

	CustomerID	Days_Since_Last_Purchase	Total_Transactions_x	Total_Products_Purchased_x	Total_Transactions_y	Total_Products_Purchased_y	Total_Spend	Average_Transaction_Value
0	12347.0	16	1	319.0	1	319.0	711.79	711.790
1	12348.0	7	1	1248.0	1	1248.0	652.80	652.800
2	12370.0	6	2	964.0	2	964.0	1744.27	872.135
3	12377.0	3	1	604.0	1	604.0	1001.52	1001.520
4	12383.0	1	1	751.0	1	751.0	555.72	555.720

Hình 2.25: Tạo cột Monetary

## 2.4 Tạo và luyện và đánh giá mô hình

### 2.4.1 Xây dựng mô hình phân cụm với thuật toán K-Means

Ta sẽ xây dựng mô hình phân cụm K-Means dựa trên bộ dữ liệu đã được chuẩn hóa ở trên, giúp làm đồng nhất độ lớn của các biến phụ thuộc, giúp mô hình không bị ảnh hưởng bởi sự chênh lệch độ lớn.

Để sử dụng thuật toán K-Means một cách hiệu quả nhất, ta cần tìm giá trị  $k$  - số lượng cụm mà chúng ta muốn sử dụng để dự đoán nhãn của một điểm dữ liệu mới - tốt nhất để mang lại hiệu quả cao cho bài toán. Vì vậy, ta vẽ biểu đồ để xét độ chính xác đối với những giá trị  $k$  khác nhau.

```
# Thiết lập kiểu plot và màu nền
sns.set(style='darkgrid', rc={'axes.facecolor': '#fcf0dc'})

# Thiết lập bảng màu cho đồ thị
sns.set_palette(['#ff6200'])

# Khởi tạo mô hình phân cụm với các tham số đã chỉ định
km = KMeans(init='k-means++', n_init=10, max_iter=100, random_state=0)

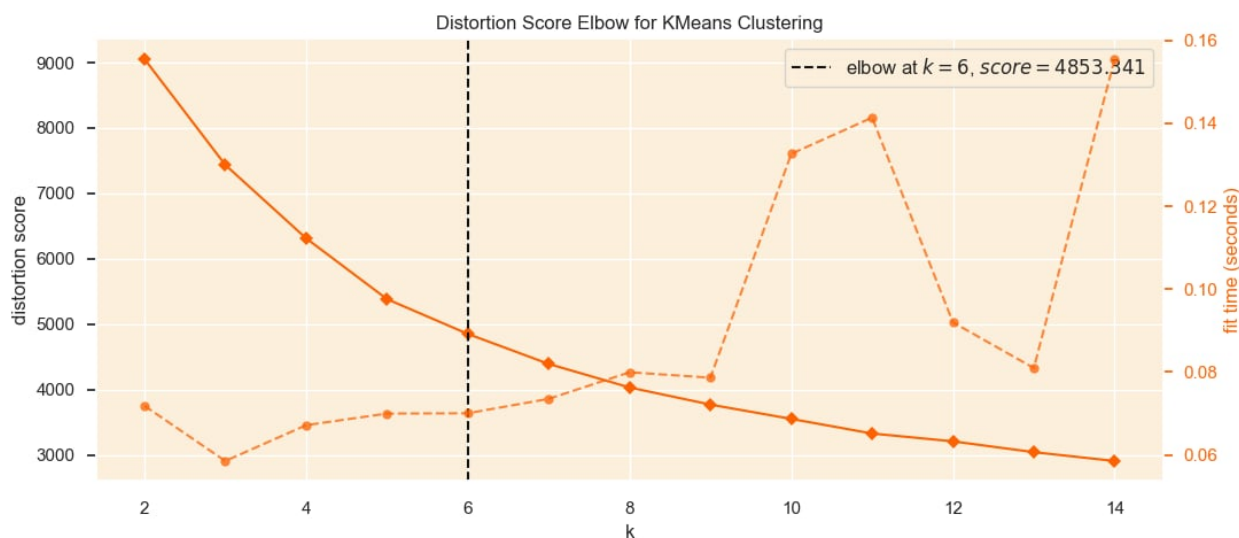
# Tạo figure và axis với kích thước mong muốn
fig, ax = plt.subplots(figsize=(12, 5))

# Khởi tạo KElbowVisualizer với mô hình và phạm vi giá trị k, và vô hiệu hóa biểu đồ thời gian
visualizer = KElbowVisualizer(km, k=(2, 15), timings=False, ax=ax)

# Fit dữ liệu vào visualizer
visualizer.fit(customer_data_pca)

# Hoàn thiện và hiển thị figure
visualizer.show();
```

Hình 2.26: Xác định tham số  $k$



Hình 2.27: Biểu đồ độ chính xác với k

Giá trị tối ưu của k cho thuật toán phân cụm KMeans có thể được tìm thấy tại điểm khuỷu tay. Tuy nhiên, trong trường hợp này chúng ta không có một điểm khuỷu tay rõ rệt, điều này thường xảy ra trong dữ liệu thực tế. Từ đồ thị, chúng ta thấy rằng độ lệch tiếp tục giảm đáng kể cho đến khi  $k=6$ , cho thấy rằng giá trị tối ưu của k có thể nằm giữa 3 và 7. Để chọn k tốt nhất trong khoảng này, chúng ta có thể sử dụng phân tích silhouette, một phương pháp đánh giá chất lượng phân cụm khác. Ngoài ra, việc kết hợp thông tin doanh nghiệp cũng có thể giúp xác định giá trị k thực tế hữu ích nhất.

### Phương pháp Silhouette:

Đầu tiên, tôi sẽ lựa chọn một phạm vi từ 2 đến 6 cho số lượng cụm (k) dựa trên phương pháp Elbow từ phần trước. Tiếp theo, tôi sẽ vẽ điểm số Silhouette cho mỗi giá trị k để xác định giá trị có điểm số cao nhất.

Tiếp theo, để điều chỉnh lựa chọn k phù hợp nhất, tôi sẽ tạo biểu đồ Silhouette để trực quan hóa hệ số silhouette cho mỗi điểm dữ liệu trong từng cụm khác nhau.

```
def silhouette_analysis(df, start_k, stop_k, figsize=(15, 16)):
    plt.figure(figsize=figsize)
    grid = gridspec.GridSpec(stop_k - start_k + 1, 2)
    first_plot = plt.subplot(grid[0, :])
    sns.set_palette(['darkorange'])

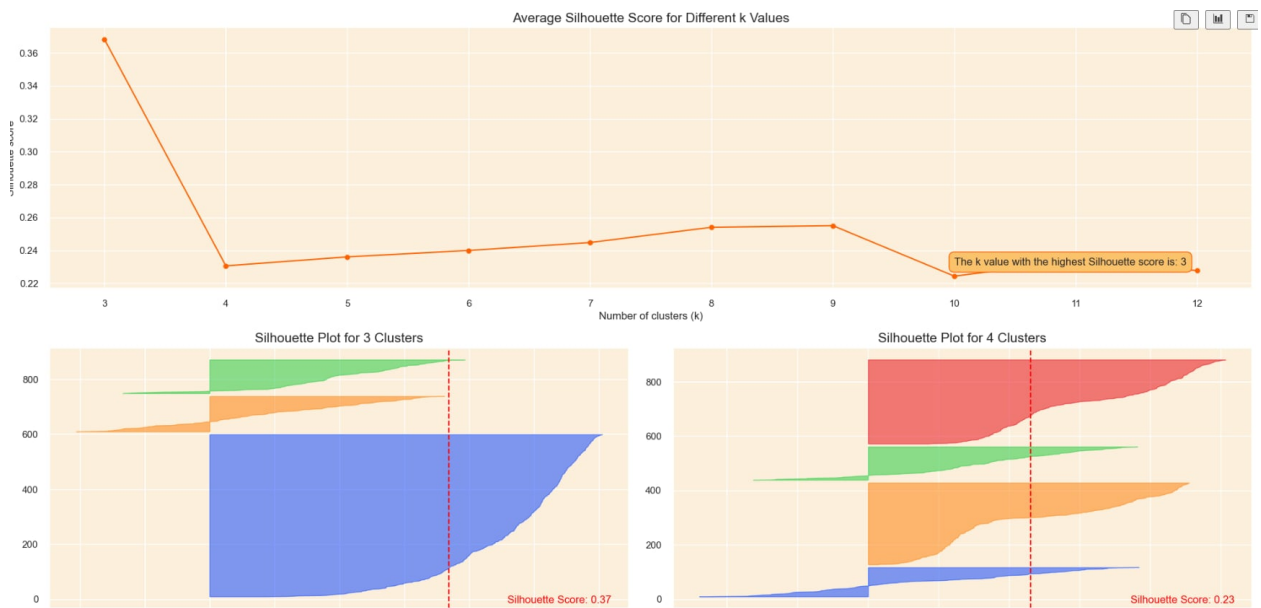
    silhouette_scores = []

    for k in range(start_k, stop_k + 1):
        km = KMeans(n_clusters=k, init='k-means++', n_init=10, max_iter=100, random_state=0)
        km.fit(df)
        labels = km.predict(df)
        score = silhouette_score(df, labels)
        silhouette_scores.append(score)

    best_k = start_k + silhouette_scores.index(max(silhouette_scores))

    plt.plot(range(start_k, stop_k + 1), silhouette_scores, marker='o')
    plt.xticks(range(start_k, stop_k + 1))
    plt.xlabel('Number of clusters (k)')
    plt.ylabel('Silhouette score')
    plt.title('Average Silhouette Score for Different k Values', fontsize=15)
```

Hình 2.28: Xây dựng thuật toán xác định K với Silhouette



Hình 2.29: Biểu đồ độ chính xác với k

Qua biểu đồ silhouette, rõ ràng rằng lựa chọn  $k=3$  là sự lựa chọn tốt hơn với điểm Silhouette là cao nhất. Lựa chọn này mang lại cho chúng ta các cụm có sự phân bố đồng đều hơn và rõ ràng hơn, làm cho giải pháp phân cụm mạnh mẽ và đáng tin cậy hơn.

Do K-Means là một thuật toán, một phương pháp học máy không giám sát nên ta sẽ không chia tập train, test và sẽ đánh giá qua điểm Silhouette.

Với  $k = 3$ , ta xây dựng mô hình phân cụm K-Means:



```

# Áp dụng phân cụm KMeans với k tối ưu
kmeans = KMeans(n_clusters=3, init='k-means++', n_init=10, max_iter=100, random_state=0)
kmeans.fit(customer_data_pca)

# Lấy tần suất của mỗi cụm
cluster_frequencies = Counter(kmeans.labels_)

# Tạo một ánh xạ từ nhãn cũ sang nhãn mới dựa trên tần suất
label_mapping = {label: new_label for new_label, (label, _) in
                  enumerate(cluster_frequencies.most_common())}

# Đảo ngược ánh xạ để gán nhãn theo tiêu chí của bạn
label_mapping = {v: k for k, v in {2: 1, 1: 0, 0: 2}.items()}

# Áp dụng ánh xạ để lấy nhãn mới
new_labels = np.array([label_mapping[label] for label in kmeans.labels_])

# Thêm nhãn cụm mới vào lại tập dữ liệu gốc
customer_data_cleaned['cluster'] = new_labels

# Thêm nhãn cụm mới vào phiên bản PCA của tập dữ liệu
customer_data_pca['cluster'] = new_labels

```

Hình 2.30: Xây dựng mô hình K-Means

### Điều kiện dừng của thuật toán:

Trong thuật toán K-Means, điều kiện dừng có thể dựa trên một hoặc nhiều tiêu chí sau:

**Số lần lặp tối đa:** Thuật toán sẽ dừng sau khi đạt đến số lần lặp tối đa được chỉ định, ngay cả khi chưa hội tụ. Trong đoạn mã bạn cung cấp, số lần lặp tối đa là 100 (`max_iter=100`).

**Sự hội tụ của centroid:** Thuật toán sẽ dừng nếu vị trí của các centroid không thay đổi đáng kể giữa các lần lặp liên tiếp, nghĩa là sự thay đổi nhỏ hơn một ngưỡng đã định trước. Điều này không được thiết lập rõ ràng trong đoạn mã của bạn, nhưng nó là một tiêu chí dừng mặc định trong nhiều triển khai K-Means.

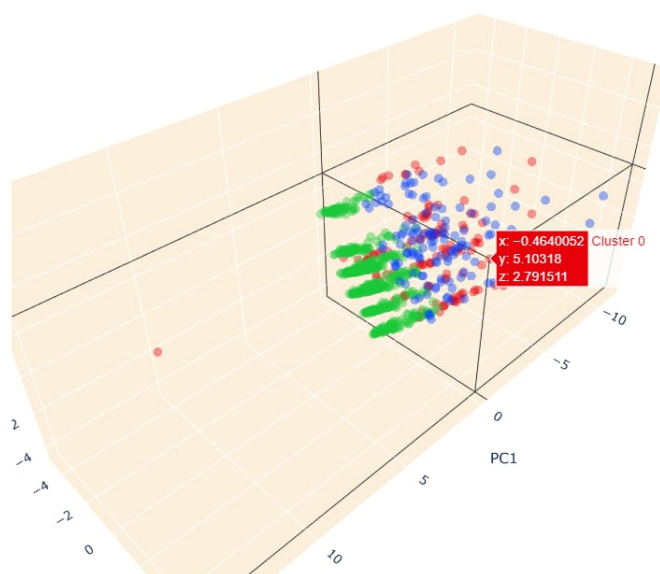
**Sự hội tụ của nhãn cụm:** Thuật toán sẽ dừng nếu các nhãn của dữ liệu không thay đổi giữa các lần lặp.

Ta sẽ sử dụng các điều kiện dừng mặc định của hàm `KMeans` từ thư viện `scikit-learn`, trong đó bao gồm:

- **Số lần lặp tối đa (`max_iter=100`):** Thuật toán sẽ dừng sau 100 lần lặp nếu không hội tụ trước đó.
- **Sự thay đổi nhỏ của centroid:** Thuật toán sẽ dừng nếu sự thay đổi của centroid giữa các lần lặp nhỏ hơn ngưỡng được xác định bởi tham số `tol`, với giá trị mặc định là  $1 \times 10^{-4}$ .



Ta có các thống kê biểu đồ sau:



Hình 2.31: Biểu đồ điểm 3D



Hình 2.32: Bộ dữ liệu được phân bổ theo các cụm

## Hiệu chỉnh siêu tham số

```
# Định nghĩa hàm tính chỉ số Silhouette
def silhouette_scorer(estimator, X):
    cluster_labels = estimator.fit_predict(X)
    return silhouette_score(X, cluster_labels)

# Tạo hàm tính điểm Silhouette
silhouette = make_scorer(silhouette_scorer)

# Thiết lập lưới tham số để hiệu chỉnh
param_grid = {
    'n_clusters': [2, 3, 4, 5],
    'init': ['k-means++', 'random'],
    'n_init': [10, 20, 30],
    'max_iter': [100, 200, 300],
    'random_state': [0]
}

# Tạo đối tượng GridSearchCV
grid_search = GridSearchCV(KMeans(), param_grid, cv=5, scoring=silhouette)

# Áp dụng GridSearchCV lên dữ liệu PCA
grid_search.fit(customer_data_pca)

# Lấy các siêu tham số tốt nhất
best_params = grid_search.best_params_

# In ra các siêu tham số tốt nhất
print("Các siêu tham số tốt nhất: ", best_params)

# Áp dụng phân cụm KMeans với các siêu tham số tốt nhất
best_kmeans = KMeans(n_clusters=best_params['n_clusters'],
                     init=best_params['init'],
                     n_init=best_params['n_init'],
                     max_iter=best_params['max_iter'],
                     random_state=best_params['random_state'])
```

Hình 2.33: Hiệu chỉnh tham số

```
random_state=best_params['random_state'])

# Huấn luyện mô hình
best_kmeans.fit(customer_data_pca)

# Lấy tần suất của mỗi cụm
cluster_frequencies = Counter(best_kmeans.labels_)

# Tạo ánh xạ từ nhãn cũ sang nhãn mới dựa trên tần suất
label_mapping = {label: new_label for new_label, (label, _) in
                 enumerate(cluster_frequencies.most_common())}

# Đảo ngược ánh xạ để gán nhãn theo tiêu chí của bạn
label_mapping = {v: k for k, v in {2: 1, 1: 0, 0: 2}.items()}

# Áp dụng ánh xạ để lấy nhãn mới
new_labels = np.array([label_mapping[label] for label in best_kmeans.labels_])

# Thêm nhãn cụm mới vào lại tập dữ liệu gốc
customer_data_cleaned['cluster'] = new_labels

# Thêm nhãn cụm mới vào phiên bản PCA của tập dữ liệu
customer_data_pca['cluster'] = new_labels

✓ 23.0s
```

Hình 2.34: Hiệu chỉnh tham số

Ta có kết quả bộ tham số tốt nhất là:

```
{'init': 'k-means++', 'max_iter': 100, 'n_clusters': 2, 'n_init': 10, 'random_state': 0}
```

=> Như vậy có thể thấy nếu ta chia thành 2 cụm, thì kết quả có thể sẽ tốt hơn. Nhưng trong bài tập này, tôi vẫn sẽ phân thành 3 cụm nhằm mục đích cuối cùng là phân khúc khách hàng được rõ ràng hơn.

## Đánh giá thuật toán K-Means

Metric	Value
Number of Observations	842
Silhouette Score	0.36822446068998416
Calinski Harabasz Score	222.11448291480673
Davies Bouldin Score	1.2904097503425298

Bảng 2.1: Evaluation Metrics

Điểm Silhouette khoảng 0.36, mặc dù không gần bằng 1, nhưng vẫn cho thấy một mức độ tách biệt hợp lý giữa các cụm. Điều này gợi ý rằng các cụm có phần khác biệt, nhưng có thể có sự chồng chéo nhẹ giữa chúng. Thông thường, một điểm số gần 1 sẽ lý tưởng hơn, chỉ ra các cụm khác biệt và tách biệt rõ ràng hơn.

Điểm Calinski Harabasz là 222.11, khá cao, chỉ ra rằng các cụm được xác định rõ ràng. Điểm số cao hơn trong chỉ số này thường cho thấy định nghĩa cụm tốt hơn, do đó, ngụ ý rằng việc phân cụm của chúng ta đã tìm ra cấu trúc đáng kể trong dữ liệu.

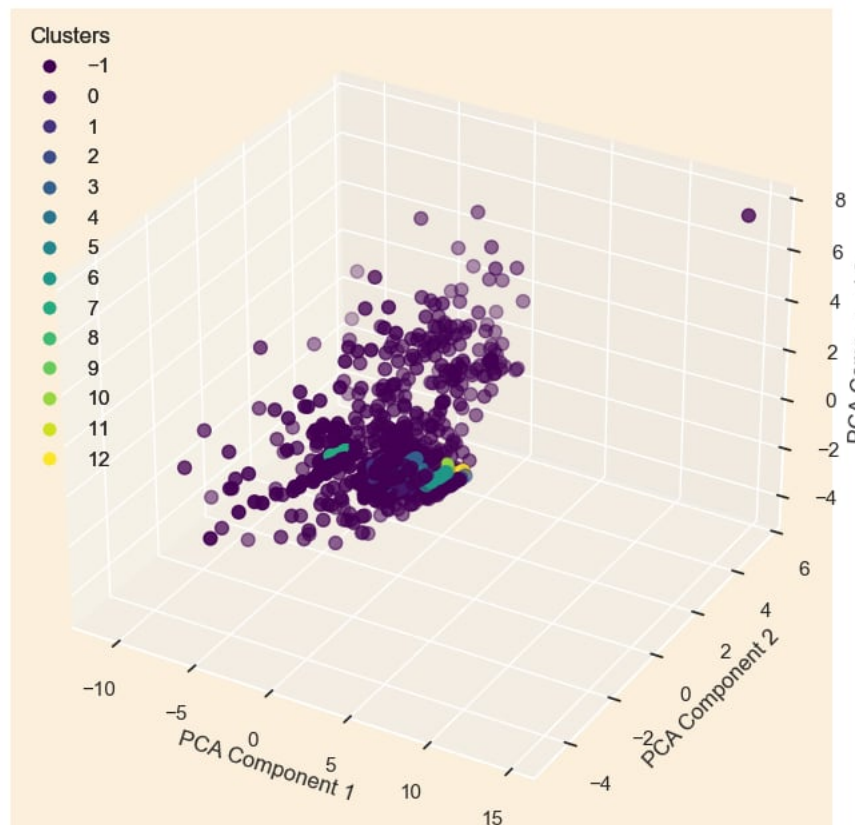
Điểm Davies Bouldin là 1.29 là một điểm số hợp lý, chỉ ra mức độ tương đồng vừa phải giữa mỗi cụm và cụm tương đồng nhất của nó. Điểm số thấp hơn thường tốt hơn vì nó chỉ ra ít sự tương đồng giữa các cụm, do đó, điểm số của chúng ta ở đây gợi ý sự tách biệt khá giữa các cụm.

### 2.4.2 Xây dựng mô hình phân cụm với thuật toán DBSCAN

```
from sklearn.cluster import DBSCAN
db11 = customer_data_pca
dbsc = DBSCAN(eps = .5, min_samples = 5).fit(db11)
clusters_scaled = customer_data_pca.copy()
clusters_scaled['cluster_pred'] = dbsc.fit_predict(db11)
clusters_scaled
ax = sns.countplot(x="cluster_pred", data=clusters_scaled)
clusters_scaled.groupby(['cluster_pred']).count()
```

✓ 0.2s

Hình 2.35: Xây dựng thuật toán DBSCAN



Hình 2.36: Biểu đồ điểm 3D biểu diễn phân cụm DBSCAN

### Điều kiện dừng thuật toán DBSCAN

Tất cả các điểm đã được gán cụm hoặc đã được xác định là điểm nhiễu:

- DBSCAN sẽ dừng lại khi tất cả các điểm dữ liệu đã được gán vào một cụm hoặc được xác định là điểm nhiễu (noise point).
- Một điểm được coi là điểm nhiễu nếu nó không có đủ điểm lân cận trong bán kính  $\text{eps}=0.5$  (bán kính lân cận) để tạo thành một cụm với điều kiện  $\text{min\_samples} = 5$ .

Không Còn Điểm Dữ Liệu Nào Để Xét:

Thuật toán bắt đầu với một điểm chưa được thăm và lân cận của nó. Nếu không còn điểm nào chưa được thăm, thuật toán sẽ dừng lại.

### Đánh giá thuật toán DBSCAN

Các chỉ số đánh giá của mô hình DBSCAN với các giá trị trên cho thấy một bức tranh tích cực về phân cụm:

- Silhouette Score: Mặc dù không gần 1, giá trị này vẫn cho thấy rằng các cụm có mức độ tách biệt tương đối. Một số chồng chéo có thể tồn tại, nhưng nhìn chung các cụm có xu hướng khá rõ ràng.

```
# Lấy dữ liệu không có nhiễu để đánh giá (các nhãn không phải -1)
mask = clusters != -1
filtered_data = customer_data_pca[mask]
filtered_labels = clusters[mask]

# Tính toán các chỉ số đánh giá
silhouette_avg = silhouette_score(filtered_data, filtered_labels)
calinski_harabasz_avg = calinski_harabasz_score(filtered_data, filtered_labels)
davies_bouldin_avg = davies_bouldin_score(filtered_data, filtered_labels)

print(f'Silhouette Score: {silhouette_avg}')
print(f'Calinski-Harabasz Score: {calinski_harabasz_avg}')
print(f'Davies-Bouldin Score: {davies_bouldin_avg}')
✓ 0.0s

Silhouette Score: 0.3198379557291562
Calinski-Harabasz Score: 87.62899458770734
Davies-Bouldin Score: 1.045085157762767
```

Hình 2.37: Đánh giá thuật toán DBSCAN

- Calinski-Harabasz Score: Giá trị cao này cho thấy rằng các cụm có sự phân tán tốt và được tách biệt rõ ràng. Điều này gợi ý rằng mô hình đã làm việc tốt trong việc phân cụm dữ liệu.
- Davies-Bouldin Score: Giá trị thấp này chỉ ra rằng các cụm không quá tương đồng với nhau, ngụ ý rằng các cụm được tách biệt rõ ràng và có sự khác biệt đáng kể.

### Đánh giá tổng thể:

- Silhouette Score: Giá trị trên 0.3 là khá tốt, cho thấy một mức độ tách biệt hợp lý giữa các cụm.
- Calinski-Harabasz Score: Giá trị cao cho thấy mô hình phân cụm của bạn tìm ra các cụm có độ phân tán tốt.
- Davies-Bouldin Score: Giá trị gần 1 là tốt, cho thấy các cụm không quá tương đồng và được tách biệt tốt.

## 2.5 Đánh giá và so sánh

Phương pháp	Silhouette Score	Calinski-Harabasz Score	Davies-Bouldin Score
K-Means	0.368	222.114	1.290
DBSCAN	0.320	87.629	1.045

Bảng 2.2: So sánh

Dựa trên các thông số kết quả của thuật toán K-Means và DBSCAN mà bạn cung cấp, chúng ta có thể đưa ra một số nhận xét so sánh:

**Silhouette Score:**

K-Means: 0.368

DBSCAN: 0.320

Silhouette Score đo độ tách biệt giữa các cụm và độ gắn kết của các điểm trong cùng một cụm. Giá trị càng gần 1 càng tốt, cho thấy các cụm tách biệt tốt và các điểm trong cụm gắn kết tốt. Trong trường hợp này, K-Means có Silhouette Score cao hơn so với DBSCAN, chỉ ra rằng K-Means có khả năng phân cụm tốt hơn.

**Calinski-Harabasz Score:**

K-Means: 222.114

DBSCAN: 87.629

Calinski-Harabasz Score cũng được sử dụng để đánh giá sự tách biệt giữa các cụm, với giá trị càng cao càng tốt. Kết quả của K-Means (222.114) cao hơn rất nhiều so với của DBSCAN (87.629), ngụ ý rằng K-Means cho kết quả phân cụm tách biệt và gắn kết tốt hơn so với DBSCAN trong trường hợp này.

**Davies-Bouldin Score:**

K-Means: 1.290

DBSCAN: 1.045

Davies-Bouldin Score đo mức độ tách biệt giữa các cụm, với giá trị càng thấp càng tốt. DBSCAN có Davies-Bouldin Score thấp hơn (1.045 so với 1.290 của K-Means), cho thấy DBSCAN có xu hướng phân cụm tốt hơn trong trường hợp này.

**Nhận xét chung:**

Sự lựa chọn giữa K-Means và DBSCAN phụ thuộc vào mục đích sử dụng và đặc tính của dữ liệu. Nếu ta quan tâm đến sự tách biệt giữa các cụm, DBSCAN có thể là lựa chọn hợp lý hơn. Nếu ta muốn cụm có kích thước tương đồng và đồng đều, có thể K-Means là phương pháp phù hợp hơn.

Tóm lại, việc lựa chọn giữa K-Means và DBSCAN phụ thuộc vào mục đích cụ thể của phân tích cụm và đặc tính của dữ liệu.

## 2.6 Ứng dụng xây dựng mô hình hệ thống đề xuất dựa trên các phân khúc khách hàng

Ta sẽ xây dựng hệ thống đề xuất (recommendation system) dựa trên dữ liệu giao dịch và phân khúc khách hàng đã được phân loại bằng thuật toán

## K-means hay DBSCAN ở trên.

```
# Bước 1: Loại bỏ giao dịch của các khách hàng bất thường từ dataframe chính
outlier_customer_ids = outliers_data['CustomerID'].astype('float').unique()
df_filtered = df[~df['CustomerID'].isin(outlier_customer_ids)]

# Bước 2: Đảm bảo kiểu dữ liệu nhất quán cho cột CustomerID trước khi merge
customer_data_cleaned['CustomerID'] = customer_data_cleaned['CustomerID'].astype('float')

# Bước 3: Merge dữ liệu giao dịch với dữ liệu khách hàng để lấy thông tin phân khúc cho từng giao dịch
merged_data = df_filtered.merge(customer_data_cleaned[['CustomerID', 'cluster']], on='CustomerID', how='inner')

# Bước 4: Xác định 10 sản phẩm bán chạy nhất trong mỗi phân khúc dựa trên tổng lượng bán
best_selling_products = merged_data.groupby(['cluster', 'StockCode', 'Description'])['Quantity'].sum().reset_index()
best_selling_products = best_selling_products.sort_values(by=['cluster', 'Quantity'], ascending=[True, False])
top_products_per_cluster = best_selling_products.groupby('cluster').head(10)

# Bước 5: Tạo bản ghi các sản phẩm được mua bởi mỗi khách hàng trong mỗi phân khúc
customer_purchases = merged_data.groupby(['CustomerID', 'cluster', 'StockCode'])['Quantity'].sum().reset_index()
```

Hình 2.38: Tiền xử lí và loại bỏ outlier

- `Outliers_data`: Dữ liệu về các khách hàng bất thường đã được phát hiện, được sử dụng để loại bỏ các giao dịch của các khách hàng này khỏi `df`.
- Chuyển đổi kiểu dữ liệu của cột `CustomerID` sang `float` để đảm bảo sự nhất quán khi merge với `df_filtered`.
- `merged_data`: DataFrame kết hợp giữa `df_filtered` (chứa giao dịch) và `customer_data_cleaned` (chứa thông tin khách hàng) dựa trên cột `CustomerID`. Kết quả là một DataFrame mới chứa thông tin về giao dịch cùng với phân khúc của từng khách hàng.
- `best_selling_products`: Tính toán tổng lượng sản phẩm bán được của mỗi sản phẩm trong mỗi phân khúc.
- `top_products_per_cluster`: Chọn ra 10 sản phẩm bán chạy nhất trong mỗi phân khúc dựa trên tổng lượng bán, sắp xếp theo thứ tự giảm dần.

```
# Bước 6: Tạo đề xuất sản phẩm cho mỗi khách hàng trong mỗi phân khúc
recommendations = []
for cluster in top_products_per_cluster['cluster'].unique():
    top_products = top_products_per_cluster[top_products_per_cluster['cluster'] == cluster]
    customers_in_cluster = customer_data_cleaned[customer_data_cleaned['cluster'] == cluster]['CustomerID']

    for customer in customers_in_cluster:
        # Xác định các sản phẩm đã mua bởi khách hàng
        customer_purchased_products = customer_purchases[(customer_purchases['CustomerID'] == customer) &
                                                         (customer_purchases['cluster'] == cluster)]['StockCode'].tolist()

        # Tìm 3 sản phẩm hàng đầu trong danh sách bán chạy mà khách hàng chưa mua
        top_products_not_purchased = top_products[~top_products['StockCode'].isin(customer_purchased_products)]
        top_3_products_not_purchased = top_products_not_purchased.head(3)

        # Thêm các đề xuất vào danh sách
        recommendations.append([customer, cluster] + top_3_products_not_purchased[['StockCode', 'Description']].values.flatten().tolist())

# Bước 7: Tạo dataframe từ danh sách đề xuất và merge nó với dữ liệu khách hàng gốc
recommendations_df = pd.DataFrame(recommendations, columns=['CustomerID', 'cluster', 'Rec1_StockCode', 'Rec1_Description', \
                                                            'Rec2_StockCode', 'Rec2_Description', 'Rec3_StockCode', 'Rec3_Description'])
customer_data_with_recommendations = customer_data_cleaned.merge(recommendations_df, on=['CustomerID', 'cluster'], how='right')
```

Hình 2.39: Xây dựng hệ thống đề xuất

Vòng lặp này duyệt qua mỗi phân khúc và cho mỗi khách hàng trong phân khúc đó, tìm ra 3 sản phẩm bán chạy nhất mà khách hàng chưa mua.

	Rec1_StockCode	Rec1_Description	Rec2_StockCode	Rec2_Description	Rec3_StockCode	Rec3_Description
CustomerID						
14813.0	17084R	ASSORTED INCENSE PACK	85123A	WHITE HANGING HEART T-LIGHT HOLDER	84879	ASSORTED COLOUR BIRD ORNAMENT
17041.0	22834	HAND WARMER BABUSHKA DESIGN	17084R	ASSORTED INCENSE PACK	85123A	WHITE HANGING HEART T-LIGHT HOLDER
16898.0	22834	HAND WARMER BABUSHKA DESIGN	17084R	ASSORTED INCENSE PACK	85123A	WHITE HANGING HEART T-LIGHT HOLDER
17700.0	22834	HAND WARMER BABUSHKA DESIGN	17084R	ASSORTED INCENSE PACK	85123A	WHITE HANGING HEART T-LIGHT HOLDER
17076.0	22834	HAND WARMER BABUSHKA DESIGN	17084R	ASSORTED INCENSE PACK	85123A	WHITE HANGING HEART T-LIGHT HOLDER
14913.0	22834	HAND WARMER BABUSHKA DESIGN	17084R	ASSORTED INCENSE PACK	85123A	WHITE HANGING HEART T-LIGHT HOLDER
16456.0	22834	HAND WARMER BABUSHKA DESIGN	17084R	ASSORTED INCENSE PACK	84879	ASSORTED COLOUR BIRD ORNAMENT
14030.0	22834	HAND WARMER BABUSHKA DESIGN	17084R	ASSORTED INCENSE PACK	85123A	WHITE HANGING HEART T-LIGHT HOLDER
15664.0	22834	HAND WARMER BABUSHKA DESIGN	17084R	ASSORTED INCENSE PACK	85123A	WHITE HANGING HEART T-LIGHT HOLDER
17019.0	22834	HAND WARMER BABUSHKA DESIGN	17084R	ASSORTED INCENSE PACK	85123A	WHITE HANGING HEART T-LIGHT HOLDER

Hình 2.40: Kết quả hệ thống đề xuất

Các cột Rec\_StockCode là các cột được lưu mã sản phẩm (StockCode) của sản phẩm đề xuất hàng cho từng khách hàng trong mỗi phân khúc. Ví dụ cụ thể:

- Rec1\_StockCode: Là mã sản phẩm (StockCode) của sản phẩm đề xuất hàng đầu cho mỗi khách hàng.
- Rec1\_Description: Là mô tả sản phẩm tương ứng với Rec1\_StockCode.
- Các Rec2, Rec3 tương tự



## Chương 3 KẾT QUẢ

### 3.1 Diễn giải các kết quả

Với các kết quả đánh giá ở trên của thuật toán K-Means và DBSCAN, ta sẽ chia nhóm khách hàng thành 3 phân khúc như sau:



Hình 3.1: Bộ dữ liệu được phân bổ theo các cụm

#### Cụm 0 - Những người mua sắm vào cuối tuần

- Khách hàng trong cụm này thường mua sắm ít thường xuyên và chi tiêu ít hơn so với các cụm khác.
- Họ thường có số lượng giao dịch nhỏ hơn và mua ít sản phẩm hơn.
- Những khách hàng này có xu hướng mua sắm vào cuối tuần, có thể tham gia vào việc xem hàng hoặc mua sắm không có kế hoạch.
- Thói quen chi tiêu của họ khá ổn định theo thời gian, cho thấy ít biến động trong chi tiêu hàng tháng của họ.

- Họ hiếm khi hủy giao dịch của mình, cho thấy hành vi mua sắm quyết định hơn.
- Khi họ mua sắm, chi tiêu mỗi giao dịch của họ thường thấp hơn so với các cụm khác.

## **Cụm 1 - Những người thỉnh thoảng chi tiêu lớn**

- Khách hàng trong cụm này không mua sắm thường xuyên nhưng có xu hướng chi tiêu đáng kể khi họ mua, mua nhiều loại sản phẩm.
- Chi tiêu của họ đang tăng lên, cho thấy sự quan tâm hoặc đầu tư ngày càng tăng trong các giao dịch mua của họ.
- Họ thích mua sắm vào cuối ngày, có thể sau giờ làm việc, và chủ yếu ở Vương quốc Anh.
- Họ có xu hướng hủy giao dịch ở mức độ trung bình, có thể do chi tiêu cao hơn; có lẽ họ thường xuyên xem xét lại các giao dịch mua của mình.
- Giao dịch mua của họ thường đáng kể, cho thấy sự ưa chuộng các sản phẩm chất lượng hoặc cao cấp.

## **Cụm 2 - Những người mua sắm nhiệt tình**

- Khách hàng trong cụm này được đặc trưng bởi thói quen chi tiêu cao. Họ có xu hướng mua nhiều loại sản phẩm độc đáo và tham gia vào nhiều giao dịch.
- Mặc dù chi tiêu cao, họ có xu hướng hủy một phần đáng kể các giao dịch của mình, có thể cho thấy hành vi mua sắm bốc đồng.
- Họ thường mua sắm vào các giờ đầu ngày, có lẽ tìm thời gian trước các cam kết hàng ngày hoặc tận dụng các giao dịch buổi sáng sớm.
- Mô hình chi tiêu của họ khá biến động, với sự dao động lớn trong chi tiêu hàng tháng, cho thấy một mô hình mua sắm ít dự đoán.
- Xu hướng chi tiêu của họ đang giảm nhẹ, có thể báo hiệu một sự thay đổi trong thói quen mua sắm của họ.

## 3.2 Khả năng ứng dụng trong tương lai

Nghiên cứu phân cụm sử dụng các thuật toán K-Means và DBSCAN có thể mở ra nhiều ứng dụng tiềm năng trong tương lai trong các lĩnh vực khác nhau. Dưới đây là một số khả năng ứng dụng chính:

- **Cá Nhân Hóa Trải Nghiệm Khách Hàng:**

- Với việc phân cụm khách hàng, doanh nghiệp có thể hiểu rõ hơn về hành vi mua sắm và nhu cầu của từng nhóm khách hàng.
- Điều này cho phép doanh nghiệp cá nhân hóa các chiến dịch tiếp thị và cung cấp các ưu đãi phù hợp với từng nhóm khách hàng, từ đó tăng cường sự hài lòng và trung thành của khách hàng.

- **Phát Triển Sản Phẩm:**

- Thông qua việc phân tích các cụm khách hàng, doanh nghiệp có thể xác định các nhu cầu và xu hướng tiêu dùng mới.
- Điều này giúp doanh nghiệp phát triển các sản phẩm và dịch vụ mới phù hợp hơn với thị trường, đáp ứng đúng nhu cầu của khách hàng.

- **Tối Ưu Hóa Quản Lý Hàng Tồn Kho:**

- Phân cụm khách hàng giúp doanh nghiệp dự đoán nhu cầu tiêu dùng của từng nhóm khách hàng cụ thể.
- Điều này cho phép doanh nghiệp tối ưu hóa quy trình quản lý hàng tồn kho, giảm thiểu lãng phí và đảm bảo hàng hóa luôn sẵn sàng phục vụ khách hàng.

- **Phân Tích Rủi Ro và Gian Lận:**

- Trong lĩnh vực tài chính, việc phân cụm có thể giúp nhận diện các mẫu hành vi bất thường, từ đó phát hiện các hoạt động gian lận.
- Điều này giúp doanh nghiệp giảm thiểu rủi ro và bảo vệ tài sản của mình.

- **Quản Lý Quan Hệ Khách Hàng (CRM):**

- Phân cụm khách hàng giúp các hệ thống CRM hoạt động hiệu quả hơn bằng cách cung cấp thông tin chi tiết về từng nhóm khách hàng.

- Điều này giúp cải thiện chất lượng dịch vụ khách hàng và tăng cường hiệu quả của các chiến dịch quản lý quan hệ khách hàng.

- **Phát Triển Chiến Lược Kinh Doanh:**

- Doanh nghiệp có thể sử dụng kết quả phân cụm để phát triển các chiến lược kinh doanh dài hạn, nhắm đến các phân khúc khách hàng tiềm năng.
- Điều này giúp doanh nghiệp cạnh tranh hiệu quả hơn trên thị trường và đạt được tăng trưởng bền vững.

- **Ứng Dụng Trong Các Lĩnh Vực Khác:**

- Ngoài ra, các thuật toán phân cụm còn có thể được áp dụng trong nhiều lĩnh vực khác như y tế, giáo dục, giao thông, và nhiều lĩnh vực khác để phân tích dữ liệu và đưa ra các quyết định hiệu quả hơn.