**TRƯỜNG ĐẠI HỌC KINH TẾ - ĐẠI HỌC ĐÀ NẴNG**

**KHOA THỐNG KÊ – TIN HỌC**

🖎🕮✍



**BÁO CÁO GIỮA KỲ**

**HỌC PHẦN: PHÂN TÍCH DỮ LIỆU BẰNG PYTHON**

**ĐỀ TÀI: SỬ DỤNG KỸ THUẬT K-MEANS VÀ PHÂN TÍCH RFM TRONG NGHIÊN CỨU PHÂN CỤM KHÁCH HÀNG**

Giảng viên hướng dẫn : TS. Lê Diên Tuấn

Sinh viên thực hiện : La Thị Thanh Tuyết

Nguyễn Lê Trân Châu

Nguyễn Hữu Toàn

Phùng Yến Vy

Lớp : 48K21.1

Nhóm : 12

***Đà Nẵng, 3/2025***

**ĐÁNH GIÁ ĐÓNG GÓP THÀNH VIÊN**

|  |  |  |
| --- | --- | --- |
| **STT** | **Họ và tên** | **Phần trăm đóng góp (%)** |
| 1 | La Thị Thanh Tuyết | 100% |
| 2 | Nguyễn Lê Trân Châu | 100% |
| 3 | Nguyễn Hữu Toàn | 100% |
| 4 | Phùng Yến Vy | 100% |

LỜI CẢM ƠN

Nhóm chúng em xin chân thành cảm ơn Trường Đại học Kinh tế – Đại học Đà Nẵng, đặc biệt là Khoa Thống kê Tin học đã tạo điều kiện thuận lợi về cơ sở vật chất và môi trường học tập để nhóm có thể thực hiện đề tài nghiên cứu một cách hiệu quả.

Nhóm cũng xin bày tỏ lòng biết ơn sâu sắc đến TS. Lê Diên Tuấn – giảng viên học phần Phân tích dữ liệu bằng Python – người đã tận tình hướng dẫn, truyền đạt kiến thức, và định hướng xuyên suốt quá trình học tập và thực hiện đề tài.

Đề tài nghiên cứu "Sử dụng kỹ thuật K-Means và phân tích RFM trong nghiên cứu phân cụm khách hàng" là kết quả của quá trình vận dụng kiến thức học được từ học phần, kết hợp với sự nỗ lực, phối hợp của các thành viên trong nhóm dưới sự hỗ trợ quý báu từ thầy.

Mặc dù nhóm đã cố gắng hoàn thành bài nghiên cứu một cách tốt nhất, nhưng chắc chắn vẫn không tránh khỏi những thiếu sót. Nhóm rất mong nhận được những ý kiến đóng góp từ quý thầy cô và các bạn để hoàn thiện hơn trong các nghiên cứu tiếp theo.

Nhóm xin chân thành cảm ơn!

LỜI CAM ĐOAN

Nhóm chúng em xin cam đoan rằng bài nghiên cứu "Sử dụng kỹ thuật K-Means và phân tích RFM trong nghiên cứu phân cụm khách hàng" là sản phẩm do chính nhóm thực hiện dưới sự hướng dẫn của TS. Lê Diên Tuấn, trong khuôn khổ học phần Phân tích dữ liệu bằng Python tại Trường Đại học Kinh tế – Đại học Đà Nẵng, Khoa Thống kê Tin học.

Toàn bộ nội dung, dữ liệu và kết quả trong bài nghiên cứu đều được nhóm thu thập, xử lý và phân tích một cách trung thực, khoa học. Nhóm không sao chép hoặc sử dụng bất kỳ nội dung nào từ các công trình nghiên cứu khác mà không trích dẫn rõ ràng nguồn tham khảo.

Nếu phát hiện có bất kỳ hành vi vi phạm nào liên quan đến đạo đức học thuật, nhóm xin hoàn toàn chịu trách nhiệm trước nhà trường và quy định hiện hành.

MỤC LỤC

[LỜI CẢM ƠN......... 1](#_Toc197083204)

[LỜI CAM ĐOAN 2](#_Toc197083205)

[DANH MỤC HÌNH ẢNH 5](#_Toc197083206)

[DANH MỤC CÁC TỪ VIẾT TẮT 7](#_Toc197083207)

[CHƯƠNG 1. TỔng quan về đề tài và cơ sở lý thuyết 8](#_Toc197083208)

[1.1. Giới thiệu 8](#_Toc197083209)

[1.2. Cơ sở lý thuyết 9](#_Toc197083210)

[1.2.1. CRM 9](#_Toc197083211)

[1.2.2. RFM Segmentation 9](#_Toc197083212)

[1.2.3. IQR 9](#_Toc197083213)

[1.2.4. Log Transformation 10](#_Toc197083214)

[1.2.5. Power Transformer với phương pháp Yeo-Jonhson 10](#_Toc197083215)

[1.2.6. StandardScaler 11](#_Toc197083216)

[1.2.7. Phân cụm K-Means 11](#_Toc197083217)

[1.2.8. Phương pháp Elbow 12](#_Toc197083218)

[1.2.9. Kiểm định Levene 12](#_Toc197083219)

[1.2.10. Welch’s ANOVA 12](#_Toc197083220)

[CHƯƠNG 2. Triển khai dự án Python 13](#_Toc197083221)

[2.1. Framework 13](#_Toc197083222)

[2.2. Dữ liệu 13](#_Toc197083223)

[2.2.1. Tổng quan dữ liệu 13](#_Toc197083224)

[2.2.2. Mô tả dữ liệu 13](#_Toc197083225)

[2.3. Tiền xử lí dữ liệu 15](#_Toc197083226)

[2.3.1. Thống kê mô tả 15](#_Toc197083227)

[2.3.2. RFM 21](#_Toc197083228)

[2.3.3. Biến đổi dữ liệu bằng Log Transformation 34](#_Toc197083229)

[2.3.4. Loại bỏ Outliers 37](#_Toc197083230)

[2.3.5. Power Transformer 38](#_Toc197083231)

[2.3.6. Chuẩn hóa dữ liệu bằng StandardScaler 40](#_Toc197083232)

[2.3.7. Phân cụm dữ liệu 41](#_Toc197083233)

[2.3.8. Kiểm đinh giả thuyết 44](#_Toc197083234)

[CHƯƠNG 3. Kết luận 48](#_Toc197083235)

[3.1. Kết quả nghiên cứu 48](#_Toc197083236)

[3.2. Hàm ý quản trị 49](#_Toc197083237)

[3.3. Hạn chế và hướng phát triển trong tương lại 51](#_Toc197083238)

[Tài liệu tham khảo 52](#_Toc197083239)

[PHỤ LỤC](#_Toc197083240)

DANH MỤC HÌNH ẢNH

[Hình 1. Framework 13](#_Toc196869564)

[Hình 2. Các cột dữ liệu trong bộ dữ liệu bán hàng Tea-Sale. 14](#_Toc196869565)

[Hình 3. Kết quả thống kê mô tả dữ liệu Tea-Sale. 15](#_Toc196869566)

[Hình 4. Kiểm tra kiểu dữ liệu của các thuộc tính trong bảng dữ liệu. 18](#_Toc196869567)

[Hình 5. Cách chuyển đổi kiểu dữ liệu cột “thời\_gian\_tạo\_đơn”, tách thành 2 cột “ngày\_tạo\_đơn” và “giờ\_tạo\_đơn”. 19](#_Toc196869568)

[Hình 6. Kiểm tra kiểu dữ liệu sau khi thay đổi kiểu dữ liệu và tách cột. 19](#_Toc196869569)

[Hình 7. Kiểm tra giá trị trùng lặp trong tập dữ liệu. 19](#_Toc196869570)

[Hình 8. Kiểm tra giá trị duy nhất của các cột trong dữ liệu. 20](#_Toc196869571)

[Hình 9. Kết quả kiểm tra dữ liệu thiếu. 20](#_Toc196869572)

[Hình 10. Xử lý giá trị thiếu và kiểm tra lại giá trị thiếu ở cột “tên\_khách\_hàng”. 21](#_Toc196869573)

[Hình 11. Kiểm tra dữ liệu ở các cột có kiểu dữ liệu là số. 21](#_Toc196869574)

[Hình 12. Tiến hành tính điểm RFM đối với từng khách hàng. 22](#_Toc196869575)

[Hình 13. Định nghĩa hàm tính điểm Recency cho mỗi khách hàng. 24](#_Toc196869576)

[Hình 14. Biểu đồ phân phối điểm Recency của khách hàng. 24](#_Toc196869577)

[Hình 15. Định nghĩa hàm tính điểm Frequency cho mỗi khách hàng. 25](#_Toc196869578)

[Hình 16. Biểu đồ phân phối điểm Frequency của khách hàng. 26](#_Toc196869579)

[Hình 17. Định nghĩa hàm tính điểm Frequency cho mỗi khách hàng. 27](#_Toc196869580)

[Hình 18. Biểu đồ phân phối điểm Monetary của khách hàng. 27](#_Toc196869581)

[Hình 19. Định nghĩa hàm tính điểm RFM cho từng khách hàng. 29](#_Toc196869582)

[Hình 20. Biểu đồ phân phối các nhóm khách hàng theo RFM Score. 29](#_Toc196869583)

[Hình 21. Tạo DataFrame với các phân khúc khách hàng và các biểu thức RFM. 30](#_Toc196869584)

[Hình 22. Định nghĩa hàm phân loại khách hàng trên điểm số RFM. 32](#_Toc196869585)

[Hình 23. Áp dụng hàm Categorizer vào cột “Customer\_RFM\_Score” để phân loại khách hàng. 32](#_Toc196869586)

[Hình 24. Biểu đồ thể hiện tỷ lệ, số lượng của các nhóm khách hàng theo RFM. 33](#_Toc196869587)

[Hình 25. Biểu đồ Histogram của 3 biến RFM trước khi biến đổi dữ liệu. 35](#_Toc196869588)

[Hình 26. Thiết lập độ lệch để xác định các cột có độ lệch cao trong 3 biến RFM. 35](#_Toc196869589)

[Hình 27. Xử lý độ lệch của dữ liệu bằng Log transformation. 35](#_Toc196869590)

[Hình 28. Biểu đồ Histograms của 3 biến RFM sau khi biến đổi log. 36](#_Toc196869591)

[Hình 29. Biểu đồ Boxplot của 3 biến RFM trước khi xử lý outlier. 37](#_Toc196869592)

[Hình 30. Biểu đồ Boxplot của 3 biến RFM sau khi xử lý Outlier bằng IQR. 38](#_Toc196869593)

[Hình 31. Kết quả sử dụng Power Transformer. 38](#_Toc196869594)

[Hình 32. Diễn giải độ lệch sau khi sử dụng Power Transformer. 39](#_Toc196869595)

[Hình 33. Biểu đồ Histogram của 3 biến RFM sau khi sử dụng Power Transformer. 39](#_Toc196869596)

[Hình 34. Biểu đồ sau khi Chuẩn hóa dữ liệu bằng StandardScaler. 40](#_Toc196869597)

[Hình 35. Biểu đồ Elbow. 41](#_Toc196869598)

[Hình 36. Biểu đồ Elbow xác định số lượng cụm tối ưu. 42](#_Toc196869599)

[Hình 37. Biểu đồ phân tán kết quả phân cụm K-Means (k=4) theo Recency, Frequency và Monetary. 43](#_Toc196869600)

[Hình 38. Kết quả kiểm định Levene trên 3 biến RFM. 45](#_Toc196869601)

[Hình 39. Kết quả kiểm định Welch’s ANOVA đối với 3 biến RFM. 46](#_Toc196869602)

DANH MỤC CÁC TỪ VIẾT TẮT

**Mã PKKH** : Mã Phân khúc khách hàng

**S**L : Số lượng

**CRM** : Customer Relationship Management

**RFM** : Recency, Frequency, Monetary

**IQR** : Interquartile Range

**Q1** : First Quartile

**Q2** : Second Quartile

**Q3** : Third Quartile

**WSS** : Within-cluster Sum of Squares

**VIP** : Very important person

# TỔng quan về đề tài và cơ sở lý thuyết

## Giới thiệu

Phân tích dữ liệu là ngành cung cấp phương pháp xử lý dữ liệu thô bằng các công cụ nhằm tìm ra các quy luật của dữ liệu được gọi là bất thường hoặc chưa rõ ràng trong tập dữ liệu lớn [[1]](https://tuyensinh.uel.edu.vn/nganh-phan-tich-du-lieu-la-gi/). Ngành phân tích dữ liệu xuất hiện đầu tiên vào năm 1970, với công nghệ thô sơ việc xử lý và khai phá dữ liệu lớn còn gặp nhiều khó khăn và thách thức. Đến năm 2008, nhờ sự tiến bộ của khoa học công nghệ máy tính đã giúp việc phân tích dữ liệu trở nên phát triển bùng nổ hơn với hàng loạt những bài báo khoa học, ấn phẩm được xuất bản.

Với nhu cầu về dữ liệu lớn của các doanh nghiệp ngày càng tăng trong việc giúp tối ưu hóa quy trình, quản lý khách hàng, theo dõi doanh số… dẫn đến sự hình thành và phát triển mạnh mẽ của CRM. CRM đã trở thành xu hướng kỹ thuật số mới của các doanh nghiệp hiện đại. Là một công cụ giúp doanh nghiệp lưu trữ dữ liệu, phân tích hành vi khách hàng, tự động hóa quy trình bán hàng và cá nhân hóa trải nghiệm nhờ AI, Machine Learning, Big Data [[2]](https://www.beryl8.com/vn/newsroom/insights/42). Nhờ vào CRM, các nhà quản trị cấp cao có thể dựa vào số liệu đó để lên kế hoạch chính xác cho các chiến dịch nhằm đạt được lợi ích tối đa với chi phí thấp và còn có thể theo dõi, chăm sóc khách hàng mục tiêu một cách chặt chẽ hơn.

Bài báo cáo tập trung vào phân khúc khách hàng bằng phương pháp K-Means dựa trên RFM, nhằm chia khách hàng vào các nhóm có hành vi tương đồng. K-Means là thuật toán phân cụm dựa trên khoảng cách, giúp nhóm khách hàng có hành vi mua sắm tương đồng nhau vào cùng một cụm. Trong khi đó, RFM đánh giá khách hàng dựa trên Recency - Thời gian giao dịch gần nhất, Frequency - Tần suất mua hàng và Monetary - Giá trị chi tiêu. Việc phân cụm khách hàng dựa trên RFM giúp doanh nghiệp hiểu rõ hơn về hành vi tiêu dùng, xác định nhóm khách hàng tiềm năng và xây dựng chiến lược tiếp cận phù hợp, từ đó tối ưu hóa chiến dịch marketing và nâng cao trải nghiệm khách hàng.

## Cơ sở lý thuyết

### CRM

Quản lý quan hệ khách hàng (Customer Relationship Management - CRM) là một chiến lược kinh doanh tổng hợp, kết hợp giữa công nghệ, quy trình và con người nhằm xây dựng, phát triển và duy trì mối quan hệ lâu dài với khách hàng. CRM giúp doanh nghiệp thu thập, quản lý và phân tích thông tin về khách hàng từ nhiều nguồn khác nhau. Bằng cách sử dụng hệ thống CRM, các tổ chức có thể hiểu rõ hơn về nhu cầu mong muốn của khách hàng từ đó tối ưu hóa tương tác với khách hàng, nâng cao trải nghiệm, cải thiện chất lượng dịch vụ và thúc đẩy tăng trưởng doanh thu. [[7]](https://www.pace.edu.vn/tin-kho-tri-thuc/crm-la-gi)

### RFM Segmentation

RFM là một phương pháp phổ biến được sử dụng để phân tích giá trị khách hàng, đặc biệt trong lĩnh vực marketing cơ sở dữ liệu và marketing trực tiếp. Phương pháp này đã nhận được sự quan tâm đáng kể của các doanh nghiệp trong ngành bán lẻ và dịch vụ.

RFM định lượng giá trị của một khách hàng dựa trên ba thông tin chính:

* Recency: Đại diện cho khoảng thời gian mua hàng gần đây nhất của khách hàng.
* Frequency: Thể hiện tần suất mua hàng của khách hàng trong một khoảng thời gian nhất định.
* Monetary: Là tổng số tiền chi tiêu của khách hàng trong khoảng thời gian phân tích. [[3]](https://phamdinhkhanh.github.io/2019/11/08/RFMModel.html)

RFM Segmentation là kỹ thuật giúp doanh nghiệp phân loại khách hàng dựa trên ba yếu tố trên, qua đó hiểu rõ hơn về hành vi khách hàng và xây dựng các chiến lược tiếp thị, chăm sóc phù hợp cho từng nhóm khách hàng cụ thể.

### IQR

Khoảng tứ phân vị (Interquartile Range - IQR) là một đại lượng thống kê mô tả được sử dụng phổ biến để đo lường mức độ phân tán của tập dữ liệu, đặc biệt là trong các trường hợp dữ liệu không tuân theo phân phối chuẩn hoặc có chứa các giá trị ngoại lai. IQR giúp xác định phạm vi mà trong đó phần lớn các giá trị dữ liệu tập trung, đồng thời loại bỏ ảnh hưởng của các điểm dữ liệu ngoại lai. [[12]](https://thongke.cesti.gov.vn/dich-vu-thong-ke/tai-lieu-phan-tich-thong-ke/845-thong-ke-mo-ta-trong-nghien-cuu-dai-luong-do-phan-tan)

IQR được định nghĩa là khoảng cách giữa tứ phân vị thứ ba (Q3) và tứ phân vị thứ nhất (Q1), cụ thể như sau:

Trong đó:

* Q1: Giá trị tại vị trí 25% của dữ liệu.
* Q3: Giá trị tại vị trí 75% của dữ liệu.

### Log Transformation

Log transformation là một kỹ thuật biến đổi dữ liệu phổ biến trong phân tích thống kê và học máy, nhằm mục đích giảm thiểu sự lệch phân phối (skewness) và ổn định phương sai. Phương pháp này được áp dụng bằng cách lấy logarit của từng giá trị dữ liệu, theo công thức:

Trong đó:

* x là giá trị ban đầu.
* x’ là giá trị sau khi biến đổi.

Nhờ khả năng cải thiện tính phân phối của dữ liệu, log transformation thường được sử dụng như một bước tiền xử lý quan trọng trong các bài toán hồi quy, phân cụm và phân loại. [[13]](https://rdsic.edu.vn/blog/blog-4/log-transformation-vi-cb.html)

### Power Transformer với phương pháp Yeo-Jonhson

Power transformation là một kỹ thuật biến đổi dữ liệu nhằm giảm thiểu độ lệch (skewness), ổn định phương sai và giúp dữ liệu tiến gần hơn tới phân phối chuẩn. Hai phương pháp phổ biến nhất là Box-Cox transformation chỉ áp dụng cho các giá trị dương, và Yeo-Johnson transformation cho phép biến đổi dữ liệu với giá trị âm, 0 và dương. Trong đó, Yeo-Johnson transformation được thiết kế để mở rộng khả năng xử lý cho các tập dữ liệu chứa cả số âm lẫn số dương. Công thức tổng quát của phép biến đổi Yeo-Johnson như sau:

Trong đó:

* y là giá trị dữ liệu gốc
* λ là tham số điều chỉnh hình dạng của phân phối sau biến đổi.

Yeo-Johnson transformation cho phép xử lý linh hoạt các tập dữ liệu thực tế thường chứa giá trị âm, giúp nâng cao hiệu quả khi áp dụng các phương pháp phân tích và mô hình hóa yêu cầu dữ liệu phân phối chuẩn. [[11]](https://www.stat.umn.edu/arc/yjpower.pdf)

### StandardScaler

StandardScaler là một phương pháp chuẩn hóa phổ biến được sử dụng trong các kỹ thuật học máy, nhằm đảm bảo rằng các đặc trưng trong dữ liệu có giá trị trung bình bằng 0 và độ lệch chuẩn bằng 1. Phương pháp này được triển khai thông qua công cụ StandardScaler trong thư viện scikit-learn của Python. Quá trình chuẩn hóa dữ liệu với StandardScaler giúp giảm sự chênh lệch về quy mô giữa các đặc trưng, từ đó cải thiện hiệu suất của các mô hình học máy.

Trong đó:

* x là giá trị gốc của đặc trưng.
* μ là giá trị trung bình của đặc trưng.
* σ là độ lệch chuẩn của đặc trưng.

Quá trình chuẩn hóa này không làm thay đổi phân phối của dữ liệu mà chỉ thay đổi quy mô của các đặc trưng, giúp các mô hình học máy học dễ dàng hơn và đạt được hiệu quả tối ưu. [[10]](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html)

### Phân cụm K-Means

Phân cụm K-means là một thuật toán học không giám sát được sử dụng để phân nhóm dữ liệu chưa được gắn nhãn. Thuật toán này có khả năng xác định các mẫu hoặc cấu trúc ẩn trong dữ liệu bằng cách chia dữ liệu thành các cụm dựa trên mức độ tương đồng giữa các điểm dữ liệu.

K-means hoạt động dựa trên việc đo khoảng cách giữa các điểm dữ liệu đến các tâm cụm. Mỗi điểm dữ liệu sẽ được gán vào cụm có tâm gần nó nhất. Quá trình này được lặp lại nhằm tối ưu vị trí các tâm cụm sao cho tổng khoảng cách từ các điểm đến tâm cụm của chúng là nhỏ nhất.

Tham số K đại diện cho số lượng cụm phân chia. K càng lớn thì dữ liệu sẽ được chia thành nhiều cụm nhỏ hơn với mức độ chi tiết hơn và ngược lại. [[6]](https://www.ibm.com/think/topics/k-means-clustering)

### Phương pháp Elbow

Phương pháp Elbow là một kỹ thuật phổ biến dùng để xác định số lượng cụm tối ưu trong thuật toán phân cụm K-Means. Bằng cách vẽ biểu đồ tổng sai số bình phương trong cụm (WSS - Within-Cluster Sum of Squares) theo số lượng cụm, phương pháp Elbow giúp chọn số cụm hợp lý bằng cách nhìn vào điểm mà biểu đồ WSS bắt đầu “gãy” lại hay nói cách khác là tại “khuỷu tay”, vì số lượng cụm tăng lên WSS sẽ giảm mạnh nhưng sau một điểm nhất định sự giảm dần này sẽ trở nên chậm hơn và ít đáng kể hơn, tạo ra một hình dạng khuỷu tay rõ rệt. Điểm này chính là số lượng cụm tối ưu mà ta nên chọn. [[9]](https://bigdatauni.com/tin-tuc/cac-phuong-phap-danh-gia-trong-thuat-toan-clustering.html)

### Kiểm định Levene

Kiểm định Levene được sử dụng để kiểm tra giả thuyết rằng phương sai của các nhóm là bằng nhau. Kiểm định này giúp xác định xem các nhóm trong dữ liệu có phương sai giống nhau hay không, điều này quan trọng trong việc kiểm định sự khác biệt giữa các nhóm. Kiểm định Levene được xây dựng dựa trên các giả thuyết:

* H₀: Phương sai của các biến Recency, Frequency và Monetary là bằng nhau giữa các cụm.
* H₁: Có ít nhất một trong ba biến Recency, Frequency hoặc Monetary có phương sai khác nhau giữa các cụm. [[8]](https://notebook.vn/kiem-dinh-levene/)

### Welch’s ANOVA

Welch’s ANOVA là một điều chỉnh của phương pháp ANOVA (Analysis of Variance) truyền thống, được phát triển để xử lý trường hợp phương sai của các nhóm không đồng nhất. Mục tiêu chính của Welch's ANOVA là kiểm tra xem có sự khác biệt có ý nghĩa thống kê nào giữa các trung bình của các nhóm hay không. Kiểm định Welch’s ANOVA được xây dựng dựa trên các giả thuyết:

* H₀: Không có sự khác biệt về giá trị trung bình giữa các nhóm (Cluster) đối với biến phụ thuộc (Recency, Frequency, hoặc Monetary).
* H₁: Có sự khác biệt về giá trị trung bình giữa các nhóm đối với biến phụ thuộc. [[7]](https://statisticsbyjim.com/anova/welchs-anova-compared-to-classic-one-way-anova/)

# Triển khai dự án Python

## Framework

A diagram of a company

AI-generated content may be incorrect.

Hình 1. Framework

## Dữ liệu

### Tổng quan dữ liệu

Là 1 tập dữ liệu chứa các giao dịch từ 01/01/2022 lúc 08:01:09 đến 31/12/2022 lúc 23:48:48 cho 1 cửa hàng bán lẻ trà khô ở miền Bắc, Việt Nam. Nguồn dữ liệu được lấy từ Môn trực quan hóa dữ liệu của thầy Trần Danh Nhân.

### Mô tả dữ liệu

Dữ liệu bao gồm 82.183 dòng và 13 cột. Cấu trúc các cột dữ liệu bao gồm:

* Thời gian tạo đơn: Ngày giờ cụ thể khi đơn hàng được tạo, định dạng YYYY-MM-DD HH:MM:SS.
* Mã đơn hàng: Chuỗi ký tự đại diện cho từng đơn hàng duy nhất gồm có 10 ký tự. Mã đơn hàng bắt đầu bằng “ORD” và theo sau là một dãy số có độ dài 7 chữ số.
* Mã khách hàng: Mã số duy nhất cho mỗi khách hàng, giúp nhận diện khách hàng trong hệ thống. Mã khách hàng bắt đầu bằng “CUZ” và theo sau là một dãy số có độ dài 5 chữ số.
* Tên khách hàng: Họ và tên đầy đủ của khách hàng thực hiện giao dịch.
* Mã PKKH: Mã phân khúc khách hàng, đại diện cho nhóm khách hàng theo tiêu chí doanh nghiệp đặt ra. Có cấu trúc bắt đầu bằng một trong các chữ cái “A”, “B”, hoặc “C”, sau đó là một chữ số từ 1 đến 3.
* Mô tả Phân Khúc Khách hàng: Thông tin chi tiết về nhóm khách hàng.
* Mã nhóm hàng: Mã đại diện cho từng nhóm sản phẩm riêng biệt.
* Tên nhóm hàng: Tên danh mục của nhóm sản phẩm.
* Mã mặt hàng: Mã số duy nhất đại diện cho từng mặt hàng cụ thể trong hệ thống. Có cấu trúc là Mã nhóm hàng theo sau là 2 chữ số.
* Tên mặt hàng: Tên sản phẩm được bán trong đơn hàng.
* SL: Số lượng của từng mặt hàng trong đơn hàng.
* Đơn giá: Giá của từng mặt hàng trên mỗi đơn vị tính.
* Thành tiền: Tổng số tiền của mặt hàng trong đơn hàng, được tính bằng SL nhân Đơn giá.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 2. Các cột dữ liệu trong bộ dữ liệu bán hàng Tea-Sale.

## Tiền xử lí dữ liệu

### Thống kê mô tả

Để hiểu rõ hơn về đặc điểm của dữ liệu trước khi tiến hành phân tích RFM và phân cụm khách hàng, nhóm đã thực hiện thống kê mô tả trên tập dữ liệu bán hàng. Quá trình này tập trung vào việc phân tích các khía cạnh chính như thời gian tạo đơn, mã đơn hàng, thông tin khách hàng, phân khúc khách hàng, nhóm hàng, mặt hàng, số lượng, đơn giá và thành tiền.

A red and white line

AI-generated content may be incorrect.

**A close up of a text

AI-generated content may be incorrect.**

Hình 3. Kết quả thống kê mô tả dữ liệu Tea-Sale.

#### *Kết quả thống kê mô tả*

Phân tích kết quả thống kê mô tả từng cột trong bộ dữ liệu bán hàng:

* Cột “mã\_đơn\_hàng” ghi nhận 9.657 dòng dữ liệu, phản ánh số lượng đơn hàng của khách hàng. Trung bình, mỗi khách hàng đến cửa hàng là 3,26 lần với độ lệch chuẩn đạt 3,64 cao hơn giá trị trung bình, cho thấy dữ liệu có sự phân bổ khá đồng đều. Đa số khách hàng chỉ mua hàng một lần, trong khi khách hàng mua nhiều nhất lên đến 22 lần. Giá trị nhỏ nhất và trung vị (Q2) đều bằng 1, đồng thời phân vị thứ nhất (Q1) cũng là 1, cho thấy 50% khách hàng chỉ đến mua hàng đúng một lần. Đến phân vị thứ ba (Q3), số lần mua hàng tăng lên 4 lần, tức là 75% khách hàng mua là 4 lần. Điều này phản ánh rằng phần lớn khách hàng là khách mới và tỷ lệ quay lại mua hàng còn thấp.
* Đối với cột “đơn\_giá”, giá sản phẩm dao động từ 18.000 VND đến 180.000 VND. Mức đơn giá trung bình đạt 52.372,51 VND, với độ lệch chuẩn 17.760,07 VND, cho thấy mức độ chênh lệch giá giữa các sản phẩm tương đối vừa phải. Phân tích các giá trị tứ phân vị cho thấy 25% sản phẩm có giá dưới 41.000 VND, trung vị đạt 48.928,57 VND và 75% sản phẩm có giá dưới 58.714,29 VND. Điều này chứng tỏ rằng phần lớn sản phẩm tập trung trong khoảng giá 41.000 VND đến 58.700 VND, hướng tới nhóm khách hàng phổ thông có mức thu nhập trung bình.
* Ở cột “thành\_tiền”, thành tiền của mỗi đơn hàng ghi nhận sự biến động rất lớn, giao động từ 18.000 VND đến 3.935.000 VND. Mức giá trị trung bình mỗi đơn hàng đạt 494.881,85 VND, trong khi độ lệch chuẩn lên tới 590.642,26 VND, phản ánh sự phân hóa mạnh giữa các đơn hàng nhỏ và các đơn hàng lớn. Về tứ phân vị, 25% đơn hàng có giá trị dưới 131.000 VND, trung vị là 249.000 VND và 75% đơn hàng có giá trị dưới 614.000 VND. Điều này cho thấy phần lớn đơn hàng thuộc nhóm giá trị nhỏ đến trung bình, tuy nhiên vẫn tồn tại một số lượng nhỏ các đơn hàng lớn bất thường, đóng góp đáng kể vào tổng doanh thu.
* Cột “thời\_gian\_tạo\_đơn” ghi nhận tổng cộng 82.137 dòng dữ liệu, trong đó có 31.403 giá trị thời gian khác nhau, cho thấy mức độ đa dạng và phân tán cao về thời điểm đặt hàng. Thời điểm được lặp lại nhiều nhất là “2022-07-27 19:52:39” với tần suất 9 lần.
* Đối với cột “mã\_đơn\_hàng”, có 82.137 dòng và 31.438 mã đơn hàng duy nhất, cho thấy rằng nhiều mã đơn hàng được lặp lại chứng tỏ rằng một đơn hàng bao gồm nhiều dòng dữ liệu chi tiết. Mã đơn hàng xuất hiện nhiều nhất là “ORD0022824” với tần suất 7 lần.
* Cột “mã\_khách\_hàng” có tổng cộng 82.137 dòng dữ liệu nhưng chỉ có 9.657 mã khách hàng duy nhất, có nghĩa rằng một khách hàng có thể đặt nhiều đơn hàng khác nhau. Mã khách hàng xuất hiện nhiều nhất là “CUZ00797” với 67 lần.
* Ở cột “tên\_khách\_hàng” ghi nhận 79.619 dòng, thiếu 2.518 dòng so với tổng số dòng trong dữ liệu (82.137 dòng), điều này cho thấy rằng dữ liệu bị thiếu hoặc có một số đơn chưa được cập nhật tên khách hàng. Có 7.671 tên khách hàng duy nhất, trong đó tên phổ biến nhất là “Thành viên mới” với 298 lần xuất hiện, phản ánh rằng nhiều khách hàng vẫn sử dụng tên mặc định khi mua hàng.
* Đối với cột “mã\_pkkh” có 9 phân khúc khách hàng được xác định trong 82.137 dòng dữ liệu. Phân khúc phổ biến nhất là “A1” với 18.124 lần xuất hiện. Điều này chỉ ra rằng nhóm khách hàng thuộc phân khúc “A1” chiếm tỷ lệ lớn trong cơ sở khách hàng.
* Cột “mô\_tả\_phân\_khúc\_khách\_hàng” tương ứng với “mã\_pkkh”, cũng có 9 mô tả khác nhau. Mô tả nổi bật nhất là nhóm “Nhân viên văn phòng, chủ doanh nghiệp (36-45 tuổi) có mức thu nhập cao, thường xuyên tìm hiểu và quan tâm đến sức khoẻ” xuất hiện 18.124 lần.
* Đối với cột “mã\_nhóm\_hàng” chỉ có 5 nhóm hàng hóa khác nhau, với mã nhóm phổ biến nhất là “THO” với 21.569 lần xuất hiện, cho thấy rằng đây là dòng sản phẩm chủ lực của doanh nghiệp.
* Đối với cột “tên\_nhóm\_hàng”, đồng nhất với mã nhóm hàng có 5 tên nhóm hàng khác nhau, trong đó tên nhóm hàng “Trà hoa” hay trong mã nhóm hàng là “THO” phổ biến nhất chiếm 21.569 dòng.
* Đối với cột “mã\_mặt\_hàng”, có 19 loại mã mặt hàng khác nhau. Mã “BOT01” là mã mặt hàng phổ biến nhất với 12.664 lần xuất hiện.
* Đối với cột “tên\_mặt\_hàng”, dữ liệu ghi nhận 19 loại sản phẩm khác nhau, với "Bột cần tây" có mã mặt hàng là “BOT01” là sản phẩm phổ biến nhất, xuất hiện đến 12.664 lần. Điều này cho thấy sản phẩm này có sự ưa chuộng cao và đóng góp mạnh mẽ vào tổng doanh thu.

#### *Xử lý dữ liệu sau quá trình thống kê mô tả*

Sau khi hoàn thành thống kê mô tả để hiểu về cấu trúc và chất lượng dữ liệu đồng thời nhóm cũng đã phát hiện một số vấn đề liên quan. Từ đó, nhóm tiến hành các bước tiền xử lý dữ liệu tiếp theo bắt đầu bằng kiểm tra và chuẩn hóa kiểu dữ liệu của các thuộc tính:

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4. Kiểm tra kiểu dữ liệu của các thuộc tính trong bảng dữ liệu.

Kết quả cho thấy có 10 thuộc tính trong bảng dữ liệu thuộc kiểu dữ liệu object và 3 thuộc tính gồm “sl”, “đơn\_giá” và “thành\_tiền” có kiểu dữ liệu là int64. Bước kiểm tra này giúp chúng ta xác định xem các cột nào cần chuyển đổi kiểu dữ liệu cho phù hợp, cụ thể ở đây sẽ là cột “thời\_gian\_tạo\_đơn” hiện đang ở dạng object cần được chuyển đổi sang định dạng datetime.

Từ kết quả trên, nhóm tiến hành chuyển đổi kiểu dữ liệu của cột “thời\_gian\_tạo\_đơn” sang kiểu Datetime và tiến hành tách cột “thời\_gian\_tạo\_đơn” thành hai cột riêng biệt là “ngày\_tạo\_đơn” có kiểu dữ liệu Datetime và “giờ\_tạo\_đơn” có kiểu dữ liệu Datetime.time, việc tách này phục vụ cho việc tính toán Recency trong mô hình RFM. Sau đó tiến hành xóa cột “thời\_gian\_tạo\_đơn” để tránh dư thừa dữ liệu.

A screenshot of a computer screen

AI-generated content may be incorrect.

Hình 5. Cách chuyển đổi kiểu dữ liệu cột “thời\_gian\_tạo\_đơn”, tách thành 2 cột “ngày\_tạo\_đơn” và “giờ\_tạo\_đơn”.

Ta sẽ tiến hành kiểm tra lại kiểu dữ liệu của các cột vừa được tạo và nhận thấy cột “ngày\_tạo\_đơn” và “giờ\_tạo\_đơn” đã có kiểu dữ liệu phù hợp với mục tiêu xử lý và phân tích.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 6. Kiểm tra kiểu dữ liệu sau khi thay đổi kiểu dữ liệu và tách cột.

Thứ ba, nhóm tiến hành kiểm tra các giá trị trùng lặp trong dữ liệu. Từ kết quả ta có thể thấy rằng dữ liệu này không có giá trị trùng lặp.

A white background with black text

AI-generated content may be incorrect.

Hình 7. Kiểm tra giá trị trùng lặp trong tập dữ liệu.

Trong bước thứ tư, nhóm thực hiện kiểm tra tính duy nhất (unique) các giá trị trong các cột dữ liệu. Mục đích của việc kiểm tra này nhằm hiểu rõ mức độ đa dạng và cấu trúc của dữ liệu.

Kết quả kiểm tra cho thấy: tập dữ liệu ghi nhận 31.403 thời điểm giao dịch với 31.438 mã đơn hàng. Có 9 phân khúc khách hàng kèm theo mô tả PKKH, 5 giá trị đối với mã nhóm hàng và tên nhóm hàng, cùng với 19 giá trị duy nhất cho mã mặt hàng và tên mặt hàng. Ngoài ra, cột số lượng có 5 giá trị, cột đơn giá có 16 giá trị và cột thành tiền có 41 giá trị duy nhất. Đáng chú ý hơn, khi nhìn vào dữ liệu chúng ta thấy có 9.657 mã khách hàng nhưng chỉ có 7.671 tên khách hàng điều này chứng tỏ rằng giữa các khách hàng có những khách hàng trùng tên với nhau, bị bỏ trống hoặc trường hợp khách hàng chưa đăng ký thành viên nên tên khách hàng được lưu mặc định là “Thành viên mới” điều này đã được phản ánh rõ qua thống kê mô tả ở phần trước.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 8. Kiểm tra giá trị duy nhất của các cột trong dữ liệu.

Thứ năm, nhóm đã tiến hành kiểm tra các cột để xác định các giá trị thiếu và phát hiện cột “tên\_khách\_hàng” có 2.518 giá trị bị thiếu và chiếm khoảng 3.07% trên tổng số dòng dữ liệu. Để đảm bảo tính đầy đủ và chính xác cho quá trình phân tích sau này, nhóm đã xử lý các giá trị thiếu này bằng cách thay thế chúng bằng cách điền giá trị xuất hiện nhiều nhất trong tên khách hàng đó chính là “Thành viên mới”.

A screenshot of a message

AI-generated content may be incorrect.

Hình 9. Kết quả kiểm tra dữ liệu thiếu.

Tiếp theo, nhóm tiến hành xử lý các giá trị thiếu trong cột “tên\_khách\_hàng” bằng cách điền giá trị phổ biến nhất. Sau khi xử lý, kết quả kiểm tra lại cho thấy cột “tên\_khách\_hàng” không còn giá trị thiếu nào.

*A screenshot of a computer

AI-generated content may be incorrect.*

Hình 10. Xử lý giá trị thiếu và kiểm tra lại giá trị thiếu ở cột “tên\_khách\_hàng”.

Cuối cùng, nhóm đã kiểm tra các cột có kiểu dữ liệu số (int64) và nhận thấy rằng không có giá trị âm, không có giá trị bằng 0, cũng như không có giá trị bị thiếu trong các cột này.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 11. Kiểm tra dữ liệu ở các cột có kiểu dữ liệu là số.

### RFM

#### *Tính toán điểm RFM*

Đầu tiên, nhóm đã tiến hành tính toán bộ chỉ số RFM (Recency - Frequency - Monetary) cho từng khách hàng được định danh bằng mã khách hàng trên dữ liệu giao dịch. Với chỉ số Recency, nhóm đã xác định ngày gần nhất xuất hiện trong tập dữ liệu làm mốc tham chiếu. Khoảng thời gian kể từ lần giao dịch gần nhất của khách hàng cho đến ngày tham chiếu sẽ được tính toán, từ đó phản ánh mức độ gần gũi của khách hàng với doanh nghiệp. Tiếp theo, chỉ số Frequency được tính bằng cách đếm số lần giao dịch của khách hàng trong khoảng thời gian nghiên cứu. Cuối cùng, chỉ số Monetary là tổng số tiền mà khách hàng đã chi tiêu trong một khoảng thời gian nhất định.

A screenshot of a computer program

AI-generated content may be incorrect.

Hình 12. Tiến hành tính điểm RFM đối với từng khách hàng.

Kết quả tính toán chỉ số RFM đối với từng mã khách hàng được trình bày rõ ràng trong hình trên. Đồng thời, nhóm xác định sử dụng kết quả tứ phân vị của các biến RFM để gán điểm cho từng khách hàng, qua đó phân tích và đánh giá hành vi mua sắm của họ. Cụ thể, đối với Recency, các phân vị cho thấy phần lớn khách hàng có xu hướng quay lại giao dịch sau khoảng thời gian khá dài, với phân vị thứ nhất (Q1) là 18 ngày, phân vị thứ hai (Q2) là 71 ngày và phân vị thứ ba (Q3) là dưới 200 ngày. Điều này cho thấy phần lớn khách hàng có sự gián đoạn đáng kể trong các giao dịch tiếp theo. Về Frequency, phân vị thứ nhất (Q1) và phân vị thứ hai (Q2) đều chỉ ra rằng 50% khách hàng thực hiện tối đa một giao dịch, và chỉ có 25% khách hàng có từ 4 giao dịch trở lên ở phân vị thứ ba (Q3), cho thấy tần suất giao dịch của khách hàng chủ yếu rất thấp. Về Monetary, phân vị thứ nhất (Q1), phân vị thứ hai (Q2) và phân vị thứ ba (Q3) lần lượt cho thấy mức chi tiêu của khách hàng là 131.000 VNĐ, 249.000 VNĐ và 614.000 VNĐ, phản ánh rằng phần lớn khách hàng có mức chi tiêu thấp, trong khi một bộ phận nhỏ khách hàng lại có giá trị chi tiêu đáng kể. Những kết quả này chỉ ra sự phân hóa rõ rệt trong hành vi của khách hàng, khi mà đa số chỉ thực hiện một lần giao dịch với giá trị chi tiêu thấp, trong khi một nhóm nhỏ lại có mức độ mua sắm cao và chi tiêu lớn. Điều này mở ra cơ hội phân nhóm chi tiết hơn, từ đó giúp tối ưu hóa các chiến lược chăm sóc và khai thác khách hàng một cách hiệu quả hơn.

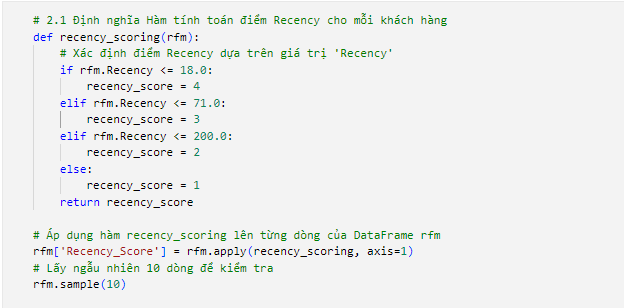
#### *Gán điểm cho các đặc chưng RFM*

Sau khi hoàn thiện bước tính toán các chỉ số RFM, nhóm tiếp tục thực hiện việc gán điểm cho các đặc trưng Recency, Frequency, Monetary cho từng khách hàng. Việc này giúp phân loại khách hàng thành các nhóm cụ thể, từ đó tạo cơ sở cho các chiến lược chăm sóc và tiếp thị nhắm đến các nhóm khách hàng có đặc điểm hành vi mua sắm tương tự.

Đầu tiên, đặc trưng Recency được sử dụng để phân loại mức độ gần đây của hành vi mua sắm của khách hàng. Nhằm mục đích này, nhóm đã xây dựng hàm recency\_scoring() để gán điểm từ 1 đến 4 cho từng khách hàng dựa trên khoảng giá trị của chỉ số Recency. Các ngưỡng phân chia được xác định từ các giá trị phân vị của dữ liệu, cụ thể như sau:

* Khách hàng có Recency lớn hơn 200 ngày sẽ được gán điểm 1, thể hiện những khách hàng đã lâu không thực hiện giao dịch.
* Khách hàng có Recency từ 72 đến 200 ngày được gán điểm 2.
* Khách hàng có Recency trong khoảng từ 19 đến 71 ngày được gán điểm là 3.
* Khách hàng có Recency nhỏ hơn hoặc bằng 18 ngày được gán điểm 4, biểu thị cho những khách hàng giao dịch rất gần đây.

Sau khi xác định các tiêu chí để gán điểm cho khách hàng theo biến Recency, nhóm sẽ tiến hành áp dụng các điểm này cho từng khách hàng trong tập dữ liệu.



Hình 13. Định nghĩa hàm tính điểm Recency cho mỗi khách hàng.

Sau khi có được kết quả gán điểm cho các khách hàng, nhóm tiến hành trực quan hóa dữ liệu bằng cách sử dụng biểu đồ phân phối để thể hiện số lượng khách hàng thuộc từng nhóm điểm (1, 2, 3, 4) của chỉ số Recency.

*A pie chart with numbers and a few different colored circles

AI-generated content may be incorrect.*

Hình 14. Biểu đồ phân phối điểm Recency của khách hàng.

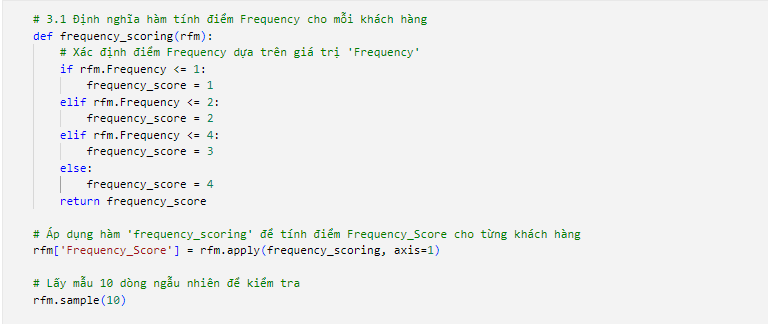
Kết quả thống kê cho thấy phân phối điểm Recency Score giữa các nhóm khách hàng khá đồng đều:

* Nhóm khách hàng có điểm 1 – những khách hàng đã lâu không thực hiện giao dịch chiếm khoảng 24,9% tổng số khách hàng.
* Nhóm có điểm 2 – khách hàng mua hàng trong khoảng thời gian không quá gần đây cũng chiếm tỷ lệ 24,9%.
* Nhóm điểm 3 – khách hàng có giao dịch tương đối gần đây chiếm 24,3%.
* Đặc biệt, nhóm điểm 4 – những khách hàng vừa thực hiện giao dịch gần đây nhất chiếm tỷ lệ cao nhất, lên đến 25,9%.

Sau khi hoàn thiện việc gán điểm cho đặc trưng Recency, nhóm tiếp tục thực hiện gán điểm cho đặc trưng Frequency nhằm phân loại mức độ thường xuyên giao dịch của khách hàng. Cụ thể, nhóm xây dựng hàm frequency\_scoring() để gán điểm từ 1 đến 4 dựa trên số lần giao dịch của khách hàng. Các ngưỡng phân chia điểm được xác định như sau:

* Khách hàng có số lần giao dịch nhỏ hơn hoặc bằng 1 lần sẽ được gán điểm 1, biểu thị cho những khách hàng ít giao dịch.
* Khách hàng có từ 2 đến 4 lần giao dịch sẽ được gán điểm 2 hoặc 3, tùy theo số lần giao dịch cụ thể.
* Khách hàng có hơn 4 lần giao dịch sẽ được gán điểm 4, cho thấy họ là những khách hàng rất thường xuyên giao dịch.

Sau khi xác định tiêu chí phân loại Frequency, nhóm áp dụng hàm frequency\_scoring() để tính điểm Frequency cho từng khách hàng trong tập dữ liệu RFM.



Hình 15. Định nghĩa hàm tính điểm Frequency cho mỗi khách hàng.

Dựa trên thang điểm đã gán cho đặc trưng Frequency, nhóm tiến hành phân loại toàn bộ khách hàng theo điểm số và trực quan hóa kết quả bằng biểu đồ phân phối.

A pie chart with numbers and a few different colored triangles

AI-generated content may be incorrect.

Hình 16. Biểu đồ phân phối điểm Frequency của khách hàng.

Kết quả thống kê cho thấy phân phối điểm Frequency Score giữa các nhóm khách hàng có sự chênh lệch rõ rệt:

* Nhóm khách hàng có điểm 1 – chỉ mua hàng 1 lần chiếm tỷ lệ cao nhất, khoảng 50,4% tổng số khách hàng. Điều này cho thấy phần lớn khách hàng chỉ phát sinh giao dịch duy nhất một lần.
* Nhóm có điểm 2 – khách mua 2 lần chiếm 12,4%, cho thấy số lượng khách hàng quay lại thêm một lần là tương đối thấp.
* Nhóm điểm 3 – khách hàng có tần suất mua từ 3 đến 4 lần chiếm 15,0%.
* Đặc biệt, nhóm điểm 4 – khách hàng mua nhiều hơn 4 lần, nhóm trung thành nhất chiếm 22,2%, cho thấy vẫn còn một lượng khách hàng trung thành tiềm năng đáng chú ý.

Sau khi hoàn thiện việc gán điểm cho đặc trưng Frequency, cuối cùng nhóm tiến hành gán điểm cho đặc trưng Monetary nhằm phân loại khách hàng theo mức độ giá trị giao dịch. Cụ thể, nhóm xây dựng hàm monetary\_scoring() để gán điểm từ 1 đến 4 dựa trên tổng giá trị giao dịch của từng khách hàng. Các ngưỡng phân chia điểm được xác định như sau:

* Khách hàng có tổng giá trị giao dịch nhỏ hơn 131.000 đồng được gán điểm 1, biểu thị cho nhóm khách hàng có giá trị thấp.
* Khách hàng có tổng giá trị giao dịch từ 131.000 đồng đến dưới 249.000 đồng được gán điểm 2.
* Khách hàng có tổng giá trị giao dịch từ 249.000 đồng đến dưới 614.000 đồng được gán điểm 3.
* Khách hàng có tổng giá trị giao dịch từ 614.000 đồng trở lên được gán điểm 4, thể hiện họ là nhóm khách hàng có giá trị cao nhất.

Sau khi xác định tiêu chí phân loại Monetary, nhóm áp dụng hàm monetary\_scoring() để tính điểm Monetary cho từng khách hàng trong tập dữ liệu RFM.

A computer screen shot of a computer code

AI-generated content may be incorrect.

Hình 17. Định nghĩa hàm tính điểm Frequency cho mỗi khách hàng.

Từ các điểm trên, ta có thể chia tất cả khách hàng theo điểm Monetary như sau:

A pie chart with numbers and a number on it

AI-generated content may be incorrect.

Hình 18. Biểu đồ phân phối điểm Monetary của khách hàng.

Phân phối điểm Monetary Score giữa các nhóm khách hàng khá đồng đều, với mỗi nhóm chiếm khoảng 25% tổng số khách hàng. Điều này cho thấy khách hàng được phân bổ khá đồng đều về giá trị giao dịch, tuy nhiên, vẫn có sự khác biệt đáng kể giữa các nhóm:

* Nhóm điểm 1 – Khách hàng có giá trị giao dịch thấp, chiếm khoảng 24,9% tổng số khách hàng. Điều này cho thấy đây là nhóm khách hàng có mức chi tiêu thấp nhất.
* Nhóm điểm 2 – Giá trị giao dịch trung bình thấp chiếm 25,1%, nhóm này có giá trị giao dịch tương đối thấp nhưng vẫn cao hơn nhóm 1.
* Nhóm điểm 3 – Giá trị giao dịch trung bình cao chiếm khoảng 25,0%, nhóm này có mức chi tiêu cao hơn và có thể là những khách hàng trung thành hơn với thương hiệu.
* Nhóm điểm 4 – Khách hàng có giá trị giao dịch cao nhất chiếm 25,0% trên tổng số khách hàng, đây là nhóm khách hàng có giá trị giao dịch cao nhất, đóng góp lớn vào doanh thu.

Sau khi gán các điểm cho các đặc trưng RFM, nhóm tiến hành tạo mã điểm RFM cho mỗi khách hàng. Cụ thể, nhóm đã tạo thêm hàm rfm\_scoring() để kết hợp ba điểm số Recency\_Score, Frequency\_Score và Monetary\_Score thành một mã điểm duy nhất cho từng khách hàng. Hàm này thực hiện việc nối ba điểm số lại với nhau theo thứ tự Recency – Frequency – Monetary, tạo thành một chuỗi số giúp phản ánh tổng quan hành vi của khách hàng.

Sau khi gán điểm cho từng đặc trưng Recency, Frequency và Monetary, nhóm tạo mã RFM cho mỗi khách hàng bằng cách kết hợp ba điểm này thành một chuỗi số. Việc sử dụng mã tổ hợp này giúp dễ dàng phân loại và phân tích hành vi khách hàng một cách chi tiết hơn, từ đó hỗ trợ thiết kế các chiến lược marketing phù hợp cho từng nhóm.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 19. Định nghĩa hàm tính điểm RFM cho từng khách hàng.

Áp dụng hàm này lên toàn bộ tập dữ liệu RFM, đồng thời tạo thêm cột Customer\_RFM\_Score để lưu trữ kết quả tính toán nhằm phục vụ cho bước phân loại và phân nhóm khách hàng ở các giai đoạn tiếp theo.

Sau khi tạo mã điểm RFM thành công, nhóm tiến hành vẽ biểu đồ thể hiện sự phân bố các nhóm khách hàng theo RFM Score, cho phép quan sát rõ số lượng khách hàng tương ứng với từng tổ hợp điểm.

A graph of a graph showing a number of bars

AI-generated content may be incorrect.

Hình 20. Biểu đồ phân phối các nhóm khách hàng theo RFM Score.

Biểu đồ phân phối điểm RFM cho thấy sự phân cực rõ rệt trong tập khách hàng. Cụ thể, số lượng khách hàng tập trung rất lớn ở hai nhóm điểm 111 và 444. Nhóm 111 – đại diện cho những khách hàng có thời gian mua hàng xa, tần suất thấp và giá trị giao dịch thấp – chiếm tỷ trọng rất cao, phản ánh rằng phần lớn khách hàng hiện tại có mức độ tương tác và giá trị mang lại cho doanh nghiệp còn thấp. Ngược lại, nhóm 444 – những khách hàng mua hàng gần đây, thường xuyên và chi tiêu cao – cũng có số lượng đáng kể, cho thấy sự tồn tại của một nhóm khách hàng trung thành và rất tiềm năng. Ngoài hai nhóm cực trị này, các điểm RFM khác phân bố thưa thớt và không đồng đều, cho thấy tệp khách hàng chưa đa dạng ở các mức độ giá trị trung gian. Thực trạng này đặt ra yêu cầu cho doanh nghiệp cần có chiến lược chăm sóc khách hàng phân hóa rõ ràng: ưu tiên duy trì và gia tăng giá trị nhóm khách hàng 444 bằng các chương trình tri ân đặc biệt, đồng thời cần có các chiến dịch tái kích hoạt nhóm khách hàng 111, như gửi ưu đãi hấp dẫn hoặc khảo sát tìm hiểu nguyên nhân rời bỏ, nhằm gia tăng tần suất mua và giá trị giao dịch. Việc nắm bắt sự phân cực trong tập khách hàng sẽ giúp doanh nghiệp tối ưu nguồn lực và nâng cao hiệu quả chăm sóc khách hàng.

Dựa vào các điểm R, F, M đã được tính ở trên cho từng khách hàng, ta tiến hành phân nhóm khách hàng theo bảng quy tắc phân nhóm khách hàng dựa trên Customer\_RFM\_Score như sau:



Hình 21. Tạo DataFrame với các phân khúc khách hàng và các biểu thức RFM.

Bảng quy tắc này xác định từng phân khúc khách hàng dựa trên tổ hợp các điểm Recency, F và M, cụ thể:

* Champion: Đây là những khách hàng xuất sắc nhất, có giá trị giao dịch cao và mua hàng rất thường xuyên. Họ có Recency ở mức 2, 3 hoặc 4, cùng với Frequency và Monetary đều đạt điểm cao là 4. Đây là nhóm khách hàng cần được duy trì và chăm sóc đặc biệt.
* Top Loyal Customer: Là những khách hàng có giá trị giao dịch lớn và mua hàng đều đặn. Recency đạt điểm 3, Frequency dao động từ 1 đến 4, và Monetary đạt điểm 3 hoặc 4. Nhóm này có tiềm năng trở thành những khách hàng trung thành lâu dài nếu được đầu tư chăm sóc đúng cách.
* Loyal Customer: Bao gồm những khách hàng duy trì mua hàng tương đối ổn định, nhưng giá trị giao dịch hoặc tần suất mua chưa thật sự cao. Recency đạt điểm 3, trong khi Frequency và Monetary chỉ ở mức điểm 1 hoặc 2. Nhóm này cần có các chương trình khuyến khích để tăng giá trị giao dịch.
* Top Recent Customer: Là nhóm khách hàng mới mua hàng rất gần đây với giá trị đơn hàng cao, Frequency và Monetary đều đạt mức 3 hoặc 4. Đây là nhóm khách hàng tiềm năng cần được thúc đẩy để trở thành khách hàng trung thành.
* Recent Customer: Là những khách mới mua hàng gần đây đạt mức 4, nhưng tần suất và giá trị giao dịch còn thấp tức Frequency và Monetary chỉ ở mức 1 hoặc 2. Nhóm này cần theo dõi và có chiến dịch thúc đẩy mua hàng thêm.
* Top Customer Needed Attention: Nhóm khách hàng từng có giá trị cao và tần suất giao dịch tốt Frequency và Monetary ở mức 3 hoặc 4, nhưng hiện tại đã lâu không mua hàng Recency chỉ ở mức 2 hoặc 3. Đây là đối tượng cần có chương trình tái kích hoạt khẩn cấp.
* Customer Needed Attention: Là những khách hàng đang có dấu hiệu rời xa, với Recency ở mức 2 hoặc 3, và cả Frequency lẫn Monetary chỉ đạt 1 hoặc 2. Cần triển khai các biện pháp quan tâm, nhắc nhớ để níu giữ nhóm này.
* Top Lost Customer: Nhóm khách hàng từng rất có giá trị với Frequency và Monetary cao từ 3 hoặc 4, nhưng đã rất lâu không giao dịch Recency ở mức thấp nhất là 1. Đây là nhóm khó hồi phục nhưng nếu thành công sẽ rất giá trị.
* Lost Customer: Là những khách hàng gần như đã mất hoàn toàn, với Recency thấp ở mức 1, và Frequency cùng Monetary cũng thấp đạt mức 1 hoặc 2. Xác suất quay lại thấp, nhưng vẫn có thể cân nhắc chiến dịch kích hoạt lại quy mô nhỏ.

Bảng quy tắc này giúp phân loại khách hàng chính xác hơn, dựa trên hành vi thực tế về tần suất mua hàng, giá trị giao dịch và thời gian mua gần nhất. Qua đó, việc phân nhóm khách hàng chính xác theo RFM giúp nhóm thực hiện các bước tiếp theo một cách chính xác và hiệu quả hơn.

Từ bảng quy tắc trên, nhóm tiến hành phân loại khách hàng dựa trên Customer\_RFM\_Score. Nhóm sẽ duyệt qua từng phần tử trong Customer\_RFM\_Score và kiểm tra từng điều kiện trong bảng quy tắc. Đối với mỗi khách hàng, nhóm sẽ kiểm tra điểm số RFM của họ. Nếu điểm số này thỏa mãn một trong các điều kiện trong bảng quy tắc, khách hàng sẽ được phân vào nhóm tương ứng.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 22. Định nghĩa hàm phân loại khách hàng trên điểm số RFM.

Sau khi định nghĩa xong, nhóm áp dụng hàm categorizer() cho toàn bộ dữ liệu khách hàng để hoàn tất bước phân loại. Kết quả trả về là tên phân khúc khách hàng tương ứng với các điều kiện điểm số RFM đã cho. Sau khi đã biết được phân khúc khách hàng của từng khách hàng, nhóm sẽ tiến hành gán tên phân khúc vào cột Customer\_Category của mỗi khách hàng để giúp trực quan và dễ phân tích hơn sau này.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 23. Áp dụng hàm Categorizer vào cột “Customer\_RFM\_Score” để phân loại khách hàng.

Sau khi phân loại khách hàng theo các nhóm RFM thành công, nhóm đã tiến hành trực quan hóa kết quả phân nhóm bằng hai biểu đồ: biểu đồ cột thể hiện số lượng khách hàng theo từng nhóm và biểu đồ tròn thể hiện tỷ lệ phần trăm khách hàng của mỗi nhóm so với tổng số. Các biểu đồ này giúp dễ dàng nhận diện quy mô và tỷ trọng của từng nhóm khách hàng, từ đó hỗ trợ xây dựng các chiến lược chăm sóc phù hợp.

A close-up of a pie chart

AI-generated content may be incorrect.

Hình 24. Biểu đồ thể hiện tỷ lệ, số lượng của các nhóm khách hàng theo RFM.

* Quan sát biểu đồ, nhóm Champion chiếm tỷ trọng lớn nhất, với khoảng 19,9% tổng số khách hàng. Đây là nhóm cực kỳ quan trọng, cần được duy trì và chăm sóc đặc biệt để gia tăng sự trung thành. Tiếp theo là nhóm Lost Customer với 18,2%, cho thấy có một lượng lớn khách hàng đã ngưng tương tác với doanh nghiệp, đặt ra thách thức lớn cho việc tái kích hoạt. Các nhóm khác như Customer Needed Attention chiếm 14,7% và Loyal Customer chiếm khoảng 9,7% cũng chiếm tỷ trọng đáng kể, thể hiện sự tồn tại của tệp khách hàng tiềm năng có thể phát triển thêm nếu được chăm sóc đúng cách. Các nhóm còn lại như Top Loyal Customer, Top Recent Customer, Recent Customer, Top Customer Needed Attention và Top Lost Customer có tỷ trọng dao động từ khoảng 6,7% đến 8,5%.
* Biểu đồ cột bổ sung thêm thông tin về số lượng khách hàng cụ thể trong từng nhóm, với nhóm Champion có số lượng khách hàng lớn nhất khoảng 1950 khách hàng, tiếp theo là Lost Customer với khoảng 1750 khách hàng, cho thấy sự phân hóa rõ rệt giữa các nhóm khách hàng về quy mô.
* Từ hai biểu đồ, có thể rút ra nhận xét rằng tập khách hàng hiện tại vừa có nhóm trung thành cao cần được duy trì, vừa có nhóm khách hàng đã rời bỏ cần có chiến lược tái kích hoạt. Do đó, doanh nghiệp cần ưu tiên hai hướng hành động: (1) Xây dựng các chương trình tri ân, cá nhân hóa ưu đãi dành cho nhóm Champion và Loyal Customer để gia tăng giá trị lâu dài; (2) Triển khai các chiến dịch tiếp thị lại (re-engagement) đối với nhóm Lost Customer và Customer Needed Attention nhằm phục hồi và khai thác thêm doanh thu từ nhóm này.

### Biến đổi dữ liệu bằng Log Transformation

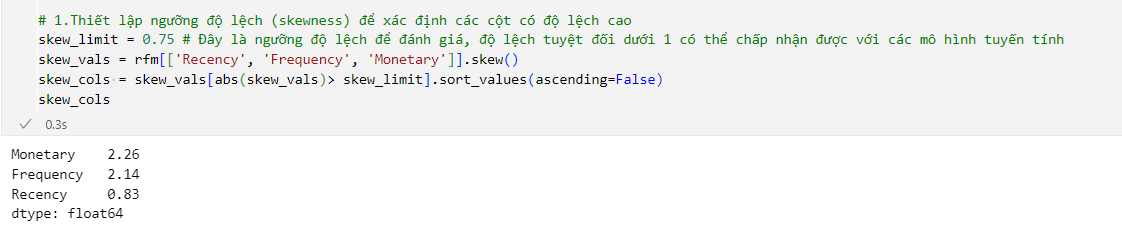
Trong quá trình phân tích dữ liệu khách hàng bằng mô hình RFM, nhóm đã trực quan hóa phân phối của các biến Recency, Frequency và Monetary thông qua biểu đồ histogram. Kết quả cho thấy cả ba biến đều có xu hướng lệch phải rõ rệt. Cụ thể, ở biến Recency phần lớn khách hàng thực hiện giao dịch gần đây, trong khi một nhóm nhỏ khách hàng có thời gian mua hàng rất lâu, tạo ra đuôi dài về phía phải. Biến Frequency cho thấy đa số khách hàng chỉ thực hiện 1 đến 2 giao dịch, số lượng khách hàng mua nhiều lần rất ít, gây mất cân đối dữ liệu. Với biến Monetary, giá trị chi tiêu chủ yếu tập trung ở mức thấp, chỉ một số ít khách hàng chi tiêu rất cao, hình thành các giá trị ngoại lệ.

**A graph of a graph of a graph

AI-generated content may be incorrect.**

Hình 25. Biểu đồ Histogram của 3 biến RFM trước khi biến đổi dữ liệu.

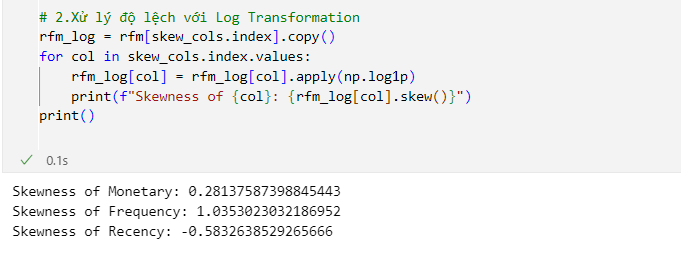
Để củng cố nhận định, nhóm cũng tính toán chỉ số độ lệch (Skewness) cho từng biến. Với ngưỡng lệch phổ biến là 0.75, kết quả cho thấy cả 3 biến đều vượt qua ngưỡng này. Cụ thể, biến Monetary có độ lệch phải là 2.26, Frequency có độ lệch phải 2.14, và Recency có độ lệch phải 0.83.



Hình 26. Thiết lập độ lệch để xác định các cột có độ lệch cao trong 3 biến RFM.

Sự lệch mạnh của các biến này có thể tác động tiêu cực đến thuật toán phân cụm K-Means, vì thuật toán này dựa trên khoảng cách Euclidean và giả định rằng dữ liệu có phân phối tương đối đối xứng. Khi dữ liệu bị lệch, các cụm phân loại có thể bị méo mó và không phản ánh chính xác các nhóm khách hàng tiềm năng.

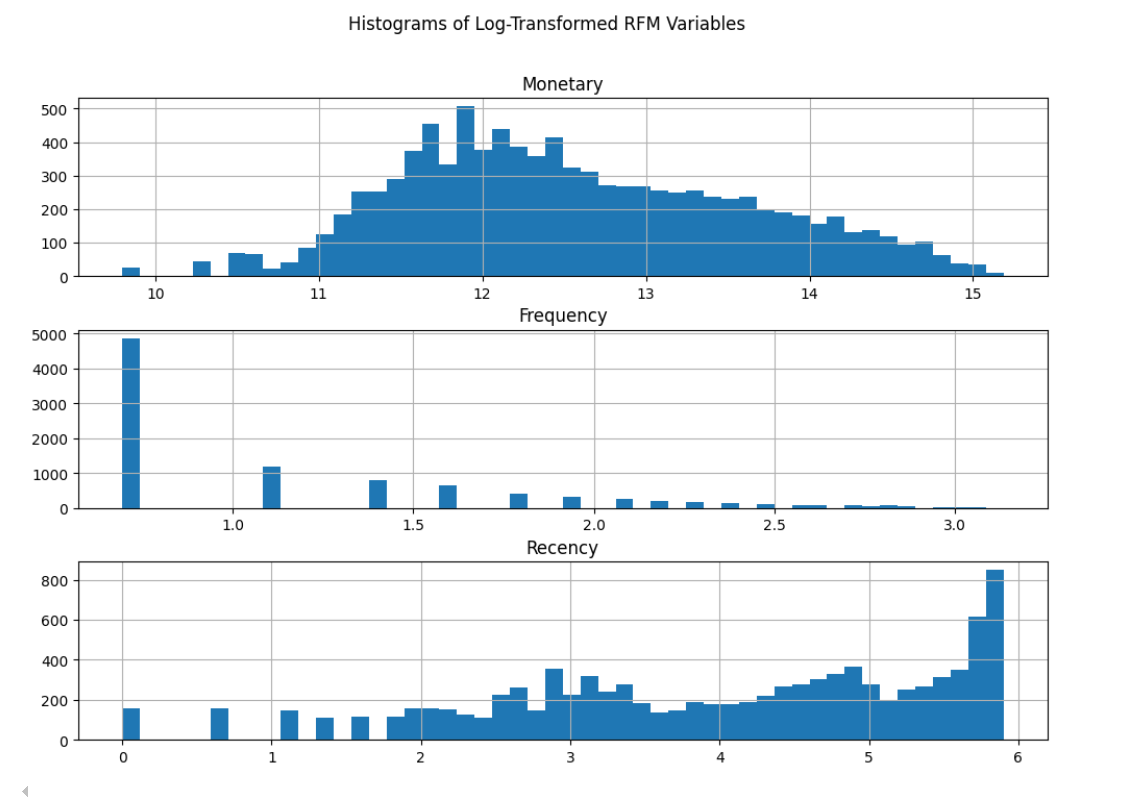
Vì vậy, nhóm đã quyết định thực hiện phép biến đổi dữ liệu bằng cách sử dụng Log Transformation để giảm độ lệch, giúp các biến có phân phối gần đối xứng hơn. Điều này sẽ giúp đảm bảo quá trình phân cụm diễn ra chính xác, tạo ra các nhóm khách hàng rõ ràng và hợp lý hơn trong mô hình phân tích. Cụ thể, nhóm áp dụng hàm np.log1p cho từng biến, sau đó tiến hành kiểm tra lại giá trị skewness của các biến đã biến đổi.



Hình 27. Xử lý độ lệch của dữ liệu bằng Log transformation.

Kết quả cho thấy: Monetary giảm độ lệch từ 2.26 xuống 0.28 và Recency từ 0.83 xuống -0.58, đều nằm trong ngưỡng [-0.75, 0.75], cho thấy phân phối của hai biến này đã gần đối xứng. Trong khi đó, Frequency tuy vẫn còn độ lệch 1.04 (vượt nhẹ ngưỡng), nhưng so với mức 2.14 ban đầu thì đã có sự cải thiện đáng kể.

Dựa trên các ngưỡng đánh giá, nhóm nhận định rằng Monetary và Recency sau biến đổi đã đạt được phân phối gần đối xứng, trong khi Frequency tuy vẫn còn đôi chút lệch nhưng ở mức độ không đáng kể. Điều này cho thấy quá trình Log Transformation đã phát huy hiệu quả trong việc điều chỉnh sự bất cân đối của dữ liệu.



Hình 28. Biểu đồ Histograms của 3 biến RFM sau khi biến đổi log.

Cuối cùng, nhóm tiến hành trực quan hóa lại phân phối của các biến RFM sau khi thực hiện Log Transformation nhằm đánh giá trực quan mức độ cải thiện. Quan sát biểu đồ histogram, có thể thấy biến Monetary sau biến đổi đã đạt được phân phối gần như đối xứng, với phần lớn dữ liệu tập trung quanh giá trị trung tâm, cho thấy sự cân đối và ổn định cao hơn so với trước. Đối với biến Frequency, mặc dù vẫn còn xuất hiện một đỉnh lớn ở giá trị thấp – phản ánh thực tế rằng phần lớn khách hàng có tần suất giao dịch không cao – nhưng phạm vi trải dài của dữ liệu đã thu hẹp đáng kể, giúp giảm ảnh hưởng của các giá trị ngoại lệ. Biến Recency cũng cho thấy sự cải thiện rõ rệt khi phân phối trở nên trải rộng và đồng đều hơn, không còn hiện tượng lệch phải mạnh như trước.

### Loại bỏ Outliers

Sau khi biến đổi log các biến trong phân tích RFM (Recency, Frequency, Monetary), việc sử dụng biểu đồ Boxplot để mô tả các biến trở nên cần thiết. Biến đổi log giúp làm giảm sự chênh lệch lớn giữa các giá trị và tạo ra phân bố dữ liệu đều hơn, thuận lợi cho việc phân tích. Biểu đồ Boxplot sẽ giúp chúng ta quan sát sự phân bố của từng biến sau khi biến đổi log, từ đó dễ dàng nhận diện các giá trị ngoại lai.

**A graph of a diagram

AI-generated content may be incorrect.**

Hình 29. Biểu đồ Boxplot của 3 biến RFM trước khi xử lý outlier.

Kết quả biểu diễn bằng biểu đồ boxplot cho thấy sự hiện diện rõ ràng của các giá trị ngoại lệ trong các đặc trưng Frequency. Trong khi biến Recency, Monetary phân bố khá đồng đều và không xuất hiện giá trị ngoại lệ rõ rệt, thì biến Frequency lại tồn tại nhiều giá trị vượt xa khỏi khoảng tứ phân vị, được thể hiện bằng các điểm tròn rời rạc phía trên hộp. Những ngoại lệ này, nếu không được xử lý, có thể làm sai lệch trung tâm dữ liệu, làm méo phân cụm, và gây ảnh hưởng tiêu cực đến hiệu quả của thuật toán K-Means – vốn nhạy cảm với khoảng cách Euclidean.

Do đó, để đảm bảo tính khách quan và độ chính xác trong quá trình phân tích phân cụm, bước xử lý ngoại lệ sẽ được thực hiện bằng phương pháp cắt ngưỡng theo khoảng IQR.

**A screenshot of a graph

AI-generated content may be incorrect.**

Hình 30. Biểu đồ Boxplot của 3 biến RFM sau khi xử lý Outlier bằng IQR.

Hình ảnh biểu đồ boxplot sau khi loại bỏ outlier cho thấy không còn hiển thị điểm nào nằm ngoài rìa (whiskers), chứng tỏ các giá trị ngoại lai đã được loại bỏ triệt để. Việc loại bỏ outlier đã cải thiện chất lượng dữ liệu, giảm ảnh hưởng của các giá trị cực đoan đến việc phân cụm K-Means và kiểm định sau này.

### Power Transformer

Sau khi xử lý outlier trong dữ liệu, bước tiếp theo là sử dụng Power Transformer với phương pháp Yeo-Johnson để chuẩn hóa các biến và giảm độ lệch (skewness).

A screenshot of a computer

AI-generated content may be incorrect.

Hình 31. Kết quả sử dụng Power Transformer.

Kết quả thu được cho thấy các biến Monetary, Frequency, và Recency đều có độ lệch gần 0, cụ thể là Monetary có giá trị 0.01, Frequency có giá trị 0.34, và Recency có giá trị -0.24. Những giá trị này đều nằm trong khoảng từ -0.75 đến 0.75, cho thấy phân phối của các biến này gần như đối xứng, với độ lệch nhẹ nhưng không đáng kể.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 32. Diễn giải độ lệch sau khi sử dụng Power Transformer.

Dựa trên các ngưỡng đánh giá độ lệch (skewness), nhóm nhận thấy rằng cả ba biến Recency, Frequency và Monetary đều có độ lệch nằm trong khoảng từ -0.75 đến 0.75, cho thấy phân phối của chúng gần như đối xứng. Đặc biệt, Monetary và Recency có độ lệch rất thấp, gần sát với 0, phản ánh phân phối gần chuẩn. Mặc dù Frequency có độ lệch nhẹ, nhưng vẫn nằm trong ngưỡng chấp nhận được. Điều này cho thấy phương pháp Power Transformer đã phát huy hiệu quả trong việc điều chỉnh phân phối, giúp các biến tiếp cận gần hơn với phân phối chuẩn.

A graph of a graph showing a number of blue and white bars

AI-generated content may be incorrect.

Hình 33. Biểu đồ Histogram của 3 biến RFM sau khi sử dụng Power Transformer.

Cuối cùng, nhóm đã tiến hành trực quan hóa phân phối của các biến RFM sau khi áp dụng Log Transformation để đánh giá mức độ cải thiện thông qua biểu đồ histogram. Quan sát cho thấy biến Monetary đã đạt được phân phối gần đối xứng, với phần lớn dữ liệu tập trung quanh giá trị trung tâm từ -1 đến 1, thể hiện sự cân đối và ổn định hơn đáng kể so với trước. Biến Frequency, dù vẫn có một đỉnh lớn ở giá trị thấp khoảng -1 – phản ánh thực tế rằng nhiều khách hàng có tần suất giao dịch thấp – nhưng phạm vi phân bố đã thu hẹp rõ rệt, giúp giảm tác động của các giá trị ngoại lệ. Biến Recency cũng cho thấy sự cải thiện đáng kể, với phân phối trở nên trải rộng và đồng đều hơn, không còn lệch phải mạnh, khi dữ liệu phân bố đều từ -2 đến 1.5.

### Chuẩn hóa dữ liệu bằng StandardScaler

Sau khi thực hiện biến đổi dữ liệu bằng Power Transformer nhằm điều chỉnh phân phối các biến về gần chuẩn, nhóm tiếp tục tiến hành bước chuẩn hóa dữ liệu bằng StandardScaler. Mục tiêu của bước này là đưa tất cả các biến về cùng một thang đo với trung bình bằng 0 và độ lệch chuẩn bằng 1, giúp đảm bảo tính công bằng giữa các biến trong các phân tích tiếp theo như phân cụm hoặc giảm chiều dữ liệu.

A graph of a graph of a graph

AI-generated content may be incorrect.

Hình 34. Biểu đồ sau khi Chuẩn hóa dữ liệu bằng StandardScaler.

Quan sát cho thấy biến Monetary đạt được phân phối gần đối xứng hơn, với dữ liệu tập trung chủ yếu quanh giá trị trung tâm từ -1 đến 1, thể hiện sự cân bằng và ổn định tốt hơn so với trước. Biến Frequency vẫn có một đỉnh cao ở giá trị thấp khoảng -1, phản ánh rằng nhiều khách hàng có tần suất giao dịch thấp, nhưng phạm vi phân bố đã được thu hẹp đáng kể, giảm ảnh hưởng của các giá trị ngoại lệ. Biến Recency cũng cho thấy sự cải thiện rõ rệt, với phân phối trở nên trải rộng và đồng đều hơn, không còn lệch phải mạnh, khi dữ liệu phân bố đều từ -2 đến 1.5.

### Phân cụm dữ liệu

#### *Phương pháp Elbow*

Sau khi tiến hành loại bỏ các giá trị outliers và chuẩn hóa các biến số RFM nhằm đảm bảo dữ liệu đồng nhất về mặt thang đo, ta tiến hành sử dụng phương pháp Elbow để chọn được k - số cụm tối ưu.

A graph with a red line

AI-generated content may be incorrect.

Hình 35. Biểu đồ Elbow.

Nhìn vào biểu đồ, có thể thấy rằng từ k = 1 đến k = 2, Inertia giảm rất mạnh, chứng tỏ phân cụm ban đầu mang lại lợi ích lớn. Từ k = 2 đến k = 3 và k = 3 đến k = 4, Inertia tiếp tục giảm nhưng tốc độ giảm đã chậm lại. Sau k = 4, đường cong bắt đầu dẹt dần, tức là việc tăng thêm số lượng cụm chỉ mang lại sự cải thiện rất nhỏ. Dựa trên quy tắc “Elbow” thì số lượng cụm tối ưu nên chọn là k = 3 hoặc k = 4.

A graph with a red line and blue line

AI-generated content may be incorrect.

Hình 36. Biểu đồ Elbow xác định số lượng cụm tối ưu.

Mặc dù ban đầu, khi chỉ dựa vào sự giảm dần của Inertia, ta có thể cân nhắc lựa chọn k = 3 hoặc k = 4, vì cả hai giá trị này đều giúp giảm distortion đáng kể. Tuy nhiên, sau khi phân tích kỹ hơn và dựa vào phương pháp “Elbow”, ta nhận thấy rằng k = 4 là lựa chọn tối ưu hơn, vì tại điểm này, đường cong distortion bắt đầu dẹt lại, cho thấy lợi ích giảm distortion đã đạt đến mức tối đa. Đồng thời, chi phí tính toán (thời gian huấn luyện) sau k = 4 không tăng quá nhiều nhưng vẫn đảm bảo chất lượng phân cụm cao.

#### *Phân cụm K-Means*

Sau khi chọn được số cụm tối ưu bằng phương pháp Elbow ta tiến hành áp dụng thuật toán K-Means Clustering để phân chia khách hàng thành 4 nhóm dựa trên các đặc trưng RFM đã được chuẩn hóa.

Biểu đồ 3D dưới đây thể hiện kết quả phân cụm khách hàng theo mô hình K-Means dựa trên các chỉ số RFM sau khi đã chuẩn hóa. Trục X thể hiện Recency, trục Y thể hiện Frequency và trục Z thể hiện Monetary.

A graph showing different colors of different sizes

AI-generated content may be incorrect.

Hình 37. Biểu đồ phân tán kết quả phân cụm K-Means (k=4) theo Recency, Frequency và Monetary.

Mỗi điểm trên biểu đồ tương ứng với một khách hàng, được gán màu sắc khác nhau tương ứng với các cụm mà thuật toán K-Means đã xác định. Cụ thể:

* Cụm màu xanh lá (Cluster 0) – Cụm khách hàng tiềm năng ổn định: Đặc trưng bởi Recency cao, Frequency cao và Monetary cao, cụm này gồm những khách hàng mua sắm thường xuyên với giá trị giao dịch lớn. Họ không chỉ quay lại liên tục mà còn chi tiêu đáng kể, là nhóm lõi mang lại doanh thu ổn định. Cần duy trì và gia tăng sự trung thành cho nhóm này qua các chương trình VIP, ưu đãi độc quyền.
* Cụm màu vàng (Cluster 1 ) – Cụm khách hàng rủi ro cao: Nhóm này có Recency thấp, Frequency thấp và Monetary thấp, phản ánh việc ít quan tâm hoặc đã rời bỏ sản phẩm/dịch vụ. Khả năng chi tiêu và tần suất mua hàng của họ rất hạn chế. Nếu muốn tái thu hút, doanh nghiệp cần xây dựng những chiến lược marketing sáng tạo, thử nghiệm các ưu đãi đặc biệt hoặc chương trình chăm sóc cá nhân để kích thích nhu cầu.
* Cụm màu xanh dương (Cluster 2) – Cụm khách hàng giá trị cao: Nhóm này bao gồm những khách hàng có điểm Recency và Frequency cao, tức là họ đã mua hàng gần đây và có tần suất mua hàng khá tốt. Tuy nhiên, Monetary vẫn còn thấp cho thấy mặc dù họ mua hàng thường xuyên nhưng chi tiêu của họ chưa cao. Đây là nhóm khách hàng tiềm năng, có thể được chăm sóc và khuyến khích chi tiêu nhiều hơn qua các chương trình ưu đãi, sản phẩm mới.
* Cụm màu tím (Cluster 3) – Cụm khách hàng suy giảm tiềm năng: Khách hàng trong cụm này có Monetary cao nhưng Recency thấp và Frequency thấp. Họ từng thực hiện những giao dịch giá trị lớn, nhưng đã lâu không quay lại và mua hàng không thường xuyên. Đây là nhóm khách hàng quan trọng cần được tái kích hoạt thông qua các chương trình khuyến mãi hoặc ưu đãi cá nhân hóa để thu hút họ quay lại.

Biểu đồ cho thấy sự phân chia rõ ràng giữa các cụm, phản ánh rằng kỹ thuật RFM kết hợp với K-Means có hiệu quả trong việc phân khúc khách hàng theo hành vi mua hàng.

### Kiểm đinh giả thuyết

Để đảm bảo tính chính xác và độ tin cậy trong phân tích dữ liệu, việc kiểm tra các giả định thống kê là bước quan trọng không thể thiếu. Trong nghiên cứu này, kiểm định Levene được sử dụng để đánh giá xem phương sai giữa các nhóm có đồng nhất hay không. Nếu kết quả kiểm định Levene chỉ ra rằng phương sai không đồng nhất, phương pháp Welch’s ANOVA sẽ được áp dụng như một sự thay thế hợp lý cho ANOVA truyền thống. Welch’s ANOVA không yêu cầu giả định phương sai đồng nhất giữa các nhóm, từ đó giúp xác định sự khác biệt về giá trị trung bình giữa các nhóm một cách chính xác hơn.

Việc kiểm tra các giả định thống kê trước khi lựa chọn phương pháp kiểm định không chỉ giúp đảm bảo sự phù hợp của phương pháp mà còn nâng cao độ tin cậy và tính chính xác của kết quả phân tích.

Trong phân tích thống kê, mức ý nghĩa (significance level) thường được xác định ở mức 0.05, có nghĩa là ta chấp nhận sai số tối đa là 5% trong quá trình kiểm định giả thuyết:

* Nếu p-value < 0.05, chúng ta bác bỏ giả thuyết H₀ và chấp nhận giả thuyết H₁.
* Nếu p-value 0.05, không đủ cơ sở để bác bỏ giả thuyết H₀.

#### *Kiểm định Levene’s Test*

Ta tiến hành kiểm định kết quả phân cụm bằng phương pháp kiểm định Levene. Kiểm định Levene được xây dựng dựa trên các giả thuyết:

* H₀: Phương sai của các biến Recency, Frequency và Monetary là bằng nhau giữa các cụm.
* H₁: Có ít nhất một trong ba biến Recency, Frequency hoặc Monetary có phương sai khác nhau giữa các cụm.

A screenshot of a computer program

AI-generated content may be incorrect.

Hình 38. Kết quả kiểm định Levene trên 3 biến RFM.

Kết quả kiểm định Levene chỉ ra sự không đồng nhất phương sai giữa các nhóm khách hàng đối với ba biến Recency, Frequency và Monetary như sau:

* Biến Recency: Giá trị thống kê kiểm định Levene là 20.92 với p-value là 1.665e-13, nhỏ hơn mức ý nghĩa 0.05. Điều này cho thấy chúng ta bác bỏ giả thuyết H₀ (phương sai đồng nhất), kết luận phương sai giữa các nhóm khách hàng đối với biến Recency là không đồng nhất.
* Biến Frequency: Kiểm định Levene có giá trị thống kê là 836.14 và p-value là 0.0, cũng nhỏ hơn mức ý nghĩa 0.05. Vì vậy, giả thuyết H₀ bị bác bỏ, chứng tỏ phương sai giữa các nhóm khách hàng đối với biến Frequency là không đồng nhất.
* Biến Monetary: Với giá trị thống kê kiểm định Levene là 11.84 và p-value là 9.774e-08, nhỏ hơn mức ý nghĩa 0.05, ta bác bỏ giả thuyết H₀ và kết luận phương sai giữa các nhóm khách hàng đối với biến Monetary là không đồng nhất.

Như vậy, tất cả ba biến đều cho thấy phương sai giữa các nhóm khách hàng không đồng nhất.

#### *Kiểm định Welch’s ANOVA*

Kết quả kiểm định Levene đã chỉ ra sự không đồng nhất phương sai giữa các nhóm khách hàng đối với ba biến Recency, Frequency và Monetary. Vì vậy, để kiểm tra sự khác biệt về trung bình giữa các nhóm trong trường hợp không đồng nhất phương sai, kiểm định Welch’s ANOVA là lựa chọn phù hợp. Kiểm định này sẽ kiểm tra cặp giả thuyết sau:

* H₀: Các nhóm có trung bình giống nhau.
* H₁: Các nhóm có trung bình khác nhau.

**A close-up of numbers

AI-generated content may be incorrect.**

Hình 39. Kết quả kiểm định Welch’s ANOVA đối với 3 biến RFM.

Dựa trên kết quả kiểm định Welch’s ANOVA đối với ba đặc trưng Recency, Frequency và Monetary, ta có thể nhận xét như sau:

Tất cả các giá trị p-value đều bằng 0.0000, nhỏ hơn mức ý nghĩa 0.05, điều này cho phép bác bỏ giả thuyết H₀ và chấp nhận H₁. Kết luận này cho thấy có sự khác biệt có ý nghĩa thống kê về giá trị trung bình giữa các nhóm khách hàng đối với từng đặc trưng.

* Biến Recency: Giá trị F = 10562.4307 cho thấy sự khác biệt rõ rệt về thời gian quay lại mua sắm giữa các nhóm khách hàng. Mức độ khác biệt này phản ánh hành vi mua sắm gần đây của khách hàng, cho thấy phân cụm đã phân loại chính xác các nhóm khách hàng dựa trên khả năng quay lại mua sắm trong tương lai.
* Biến Frequency: Giá trị F = 31214.4059 khẳng định sự khác biệt đáng kể giữa các nhóm về tần suất mua hàng. Điều này chứng tỏ mô hình phân cụm đã phân tách các nhóm khách hàng theo mức độ tương tác và tần suất mua sắm, từ nhóm khách hàng thường xuyên đến nhóm ít tương tác hơn.
* Biến Monetary: Giá trị F = 7516.8964 chỉ ra sự khác biệt rõ rệt về mức chi tiêu giữa các nhóm khách hàng. Phân cụm đã phân loại thành công các nhóm có giá trị chi tiêu khác nhau, giúp doanh nghiệp nhận diện được nhóm khách hàng có giá trị kinh tế cao, từ đó có thể áp dụng các chiến lược chăm sóc và tiếp thị phù hợp.

Nhìn chung, kết quả kiểm định Welch’s ANOVA cho thấy sự khác biệt có ý nghĩa thống kê giữa các nhóm khách hàng đối với tất cả ba đặc trưng Recency, Frequency và Monetary. Các giá trị F lớn cho thấy phân cụm đã thành công trong việc chia nhóm khách hàng thành các nhóm có hành vi mua sắm và mức chi tiêu rất khác biệt

# Kết luận

## Kết quả nghiên cứu

Trong nghiên cứu này, nhóm đã triển khai thành công mô hình phân tích và phân khúc khách hàng dựa trên sự kết hợp giữa kỹ thuật phân tích RFM (Recency – Frequency – Monetary) và thuật toán phân cụm K-Means. Qua quá trình xử lý, chuẩn hóa và phân tích dữ liệu giao dịch thực tế, một số kết quả chính đạt được có thể được tổng kết như sau:

Thứ nhất, nhóm đã xây dựng được một quy trình phân tích dữ liệu khách hàng có hệ thống, bao gồm các bước: thu thập dữ liệu, xử lý dữ liệu thiếu và dữ liệu ngoại lai (sử dụng khoảng tứ phân vị – IQR), chuẩn hóa biến (StandardScaler), trích xuất đặc trưng RFM và cuối cùng là áp dụng thuật toán K-Means để thực hiện phân cụm.

Thứ hai, thông qua việc áp dụng mô hình RFM, nhóm đã phân loại khách hàng dựa trên hành vi mua sắm, từ đó chia thành các nhóm như: Champion, Top Loyal Customer, Loyal Customer, Top Recent Customer, Recent Customer, Top Customer Needed Attention, Customer Needed Attention, Top Lost Customer, Lost Customer. Việc này giúp nhận diện được nhóm khách hàng trung thành, nhóm khách hàng từng gắn bó cũng như những nhóm khách hàng tiềm năng. Từ đó hỗ trợ doanh nghiệp trong công tác chăm sóc và giữ chân khách hàng. Các đặc trưng RFM này sau đó được sử dụng làm nền tảng để phân cụm chi tiết hơn thông qua thuật toán K-Means, nhằm tối ưu hóa việc phân nhóm và đưa ra chiến lược cụ thể.

Thứ ba, dựa trên các đặc trưng RFM đã trích xuất, thuật toán K-Means đã được triển khai hiệu quả sau khi xác định được số lượng cụm tối ưu (k = 4) thông qua phương pháp Elbow. Kết quả phân cụm cho thấy bốn nhóm khách hàng với hành vi đặc trưng khác nhau. Cụm 0 là nhóm khách hàng trung thành, thường xuyên mua hàng với giá trị chi tiêu ổn định, cần tập trung vào duy trì và tri ân. Cụm 1 gồm những khách hàng rủi ro cao, ít mua sắm và chi tiêu thấp, cần chiến lược tái kích hoạt hiệu quả. Cụm 2 là nhóm khách hàng tiềm năng, mua hàng thường xuyên nhưng giá trị giao dịch chưa cao, phù hợp với các chương trình khuyến khích chi tiêu. Cụm 3 là nhóm khách hàng suy giảm tiềm năng, từng chi tiêu lớn nhưng hiện tại ít tương tác, cần được chăm sóc lại để tăng khả năng quay lại. Việc phân loại này giúp doanh nghiệp xây dựng chiến lược marketing cá nhân hóa và tối ưu hiệu quả giữ chân khách hàng.

Kết quả phân tích thống kê, bao gồm kiểm định Levene và Welch’s ANOVA, cho thấy sự khác biệt có ý nghĩa thống kê giữa các nhóm khách hàng đối với ba đặc trưng Recency, Frequency và Monetary. Cụ thể, kiểm định Levene bác bỏ giả thuyết đồng nhất phương sai, chỉ ra sự không đồng nhất giữa các nhóm, đồng thời kiểm định Welch’s ANOVA xác nhận sự khác biệt đáng kể về giá trị trung bình giữa các cụm. Các giá trị F lớn cùng p-value nhỏ hơn 0.05 đối với cả ba biến khẳng định rằng quá trình phân cụm RFM đã phân tách khách hàng thành những nhóm có hành vi mua sắm và mức độ chi tiêu khác biệt rõ rệt. Những kết quả này không chỉ chứng minh tính phù hợp và hiệu quả của phương pháp phân cụm RFM kết hợp K-Means, mà còn cung cấp cơ sở khoa học vững chắc cho việc phát triển các chiến lược tiếp thị và quản lý quan hệ khách hàng một cách tối ưu, góp phần nâng cao hiệu quả kinh doanh bền vững.

## Hàm ý quản trị

Dựa trên kết quả phân cụm khách hàng bằng thuật toán K-Means, tập dữ liệu được chia thành 4 cụm khách hàng có hành vi tiêu dùng khác biệt rõ rệt gồm: Cụm khách hàng tiềm năng ổn định, cụm khách hàng rủi ro cao, cụm khách hàng giá trị cao và cụm khách hàng suy giảm tiềm năng. Việc nhận diện đặc trưng của từng cụm khách hàng giúp doanh nghiệp có cơ sở để xây dựng các chiến lược marketing cá nhân hóa, phù hợp với hành vi và tiềm năng đóng góp doanh thu của từng cụm.

Đối với cụm khách hàng tiềm năng ổn định, mục tiêu chủ yếu là duy trì và gia tăng giá trị vòng đời khách hàng. Đây là những khách hàng có tần suất mua hàng cao và chi tiêu ổn định, là nguồn doanh thu vững chắc cho doanh nghiệp. Doanh nghiệp cần triển khai các chương trình khách hàng thân thiết, như hệ thống tích điểm và phân hạng thành viên, để tăng sự trung thành và khuyến khích họ tiếp tục gắn bó lâu dài. Ngoài ra, chiến lược up-selling và cross-selling cần được áp dụng hợp lý, dựa trên hành vi giao dịch của khách hàng, nhằm tăng giá trị đơn hàng và nâng cao doanh thu. Các ưu đãi cá nhân hóa vào những dịp đặc biệt như sinh nhật hoặc lễ Tết sẽ tạo ra cảm xúc tích cực và giúp khách hàng cảm nhận sự quan tâm từ thương hiệu. Để duy trì mối quan hệ lâu dài, doanh nghiệp cũng cần gửi các email cập nhật sản phẩm mới, các tin tức thương hiệu, hoặc nội dung hữu ích, giúp khách hàng luôn nhớ đến thương hiệu.

Đối với cụm khách hàng rủi ro cao, cần tái kích hoạt sự quan tâm và đưa họ trở lại với thương hiệu. Doanh nghiệp cần thực hiện các chiến dịch marketing tái kích hoạt, chẳng hạn như khuyến mãi đặc biệt (giảm giá, quà tặng miễn phí), hoặc gửi ưu đãi giới hạn để khôi phục sự chú ý của nhóm khách hàng này. Các chương trình chăm sóc cá nhân hóa (ví dụ như ưu đãi cho khách hàng lâu không mua) sẽ giúp tạo động lực cho họ quay lại. Bên cạnh đó, doanh nghiệp cần lắng nghe phản hồi từ nhóm khách hàng này để hiểu lý do họ không còn tương tác, từ đó có thể điều chỉnh sản phẩm/dịch vụ hoặc trải nghiệm khách hàng sao cho phù hợp.

Với cụm khách hàng giá trị cao, cần tăng cường giá trị đơn hàng và khuyến khích họ chi tiêu nhiều hơn. Doanh nghiệp cần áp dụng chiến lược up-selling và cross-selling để đề xuất sản phẩm bổ trợ hoặc gói combo phù hợp với thói quen mua hàng của khách. Các ưu đãi giới hạn thời gian (ví dụ: giảm giá cho đơn hàng tiếp theo) sẽ tạo ra động lực cho khách hàng chi tiêu nhiều hơn. Bên cạnh đó, việc đưa ra các chương trình khách hàng thân thiết (tích điểm, cấp hạng thành viên) sẽ giúp tăng tần suất mua hàng và duy trì sự gắn kết lâu dài. Doanh nghiệp cũng nên duy trì sự kết nối với nhóm này qua các email cập nhật sản phẩm mới hoặc chương trình khuyến mãi đặc biệt.

Đối với cụm khách hàng suy giảm tiềm năng, doanh nghiệp cần tái thu hút và khôi phục mối quan hệ với khách hàng đã từng có giá trị giao dịch lớn, nhưng hiện nay tần suất và chi tiêu giảm sút. Để làm được điều này, doanh nghiệp cần triển khai chiến dịch khuyến mãi cá nhân hóa, chẳng hạn như giảm giá cho những sản phẩm họ đã từng mua hoặc mời dùng thử sản phẩm mới. Việc cung cấp các ưu đãi đặc biệt cho nhóm khách hàng này sẽ là động lực để họ quay lại. Đồng thời, việc gửi email chăm sóc (thông tin về các sản phẩm liên quan, các sự kiện hoặc chương trình mới) cũng giúp duy trì sự kết nối. Doanh nghiệp cần theo dõi thường xuyên hành vi của nhóm khách hàng này và có chiến lược chăm sóc cá nhân để tái kích hoạt họ.

Việc áp dụng mô hình K-Means kết hợp với chỉ số RFM trong phân tích khách hàng không chỉ giúp doanh nghiệp nhận diện rõ hành vi tiêu dùng, mà còn mở ra khả năng xây dựng các chính sách marketing cá nhân hóa theo từng cụm khách hàng. Mỗi nhóm khách hàng có hành vi, giá trị và tiềm năng khác nhau đòi hỏi các chiến lược tiếp cận, giữ chân và phát triển khác biệt. Khi các chính sách này được triển khai đồng bộ và hợp lý, doanh nghiệp có thể tối ưu hóa nguồn lực tiếp thị, tăng tỷ lệ chuyển đổi, và nâng cao giá trị vòng đời của khách hàng một cách bền vững.

## Hạn chế và hướng phát triển trong tương lại

Mặc dù nghiên cứu đã đạt được những kết quả nhất định, nhưng vẫn còn tồn tại một số hạn chế cần được ghi nhận và có hướng giải quyết trong các nghiên cứu tương lai:

Đầu tiên, phạm vi nghiên cứu còn hạn chế chỉ sử dụng nguồn dữ liệu của một cửa hàng bán lẻ dẫn đến tính khái quát của mô hình còn hạn chế.

Thứ hai, bộ dữ liệu chỉ bao gồm các thông tin giao dịch trong năm 2022 nên còn hạn chế khả năng phân tích xu hướng dài hạn của khách hàng và hành vi mua sắm.

Cuối cùng, do hạn chế về thời gian và kiến thức lý thuyết cũng như kinh nghiệm thực tế của các nhà nghiên cứu, nên đóng góp của nghiên cứu này có thể bị hạn chế phần nào. Do đó, cần tiếp tục khám phá và sử dụng các thuật toán, kỹ thuật phân cụm khác như DBSCAN, Hierarchical Clustering,..

Từ đó, nhóm đề xuất một số định hướng phát triển trong tương lai như sau:

Thứ nhất, cần mở rộng phạm vi dữ liệu bằng cách áp dụng mô hình trên tập dữ liệu từ nhiều nguồn bán hàng khác nhau của ngành trà trên cả nước, giúp tăng độ tin cậy và khả năng tổng quát của mô hình.

Thứ hai, cần thu thập dữ liệu trong nhiều năm liên tiếp, nhằm tạo điều kiện cho việc phân tích xu hướng hành vi tiêu dùng dài hạn của khách hàng. Điều này không chỉ giúp phát hiện sự thay đổi trong hành vi mua sắm theo thời gian, mà còn hỗ trợ doanh nghiệp trong việc xây dựng các chiến lược chăm sóc và giữ chân khách hàng hiệu quả hơn.

Cuối cùng, thử nghiệm và so sánh nhiều thuật toán phân cụm khác nhau, bao gồm cả các phương pháp dựa trên mật độ (Density-Based) hoặc phân cụm phân cấp (Hierarchical Clustering), để đánh giá độ ổn định và độ chính xác của kết quả phân nhóm.

# Tài liệu tham khảo

|  |  |
| --- | --- |
| [1] | UEL, "nganh phan tich du lieu la gi," UEL, 15 08 2024. [Online]. Available: https://tuyensinh.uel.edu.vn/nganh-phan-tich-du-lieu-la-gi/. [Accessed 30 4 2024]. |
| [2] | Beryls, "Beryls," Beryls, 15 7 2021. [Online]. Available: https://www.beryl8.com/vn/newsroom/insights/42. [Accessed 30 04 2025]. |
| [3] | phamdinhkhanh, "phamdinhkhanh," 08 11 2019. [Online]. Available: https://phamdinhkhanh.github.io/2019/11/08/RFMModel.html. [Accessed 30 04 2025]. |
| [4] | T. Vu, "Subiz," 13 11 2024. [Online]. Available: https://subiz.com.vn/blog/mo-hinh-rfm.html. [Accessed 30 04 2025]. |
| [5] | H. Noi, "200Lab," 10 07 2021. [Online]. Available: https://200lab.io/blog/phan-loai-khach-hang-rfm-segmentation. [Accessed 30 04 2025]. |
| [6] | E. Kavlakoglu and V. Winland, "IBM," IBM, 26 06 2024. [Online]. Available: https://www.ibm.com/think/topics/k-means-clustering. [Accessed 30 04 2025]. |
| [7] | J. Frost, "Statistics By Jim," 12 06 2024. [Online]. Available: https://statisticsbyjim.com/anova/welchs-anova-compared-to-classic-one-way-anova/. [Accessed 30 04 2025]. |
| [8] | CV, "Notebookvn," 23 03 2022. [Online]. Available: https://notebook.vn/kiem-dinh-levene/. [Accessed 30 04 2025]. |
| [9] | BigDataUni, "BigDataUni," 12 05 2024. [Online]. Available: https://bigdatauni.com/tin-tuc/cac-phuong-phap-danh-gia-trong-thuat-toan-clustering.html. [Accessed 30 04 2025]. |
| [10] | s. learn, "scikit-learn," [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html. [Accessed 30 04 2024]. |
| [11] | S. Weisberg, 26 10 2001. [Online]. Available: https://www.stat.umn.edu/arc/yjpower.pdf. [Accessed 30 04 2025]. |
| [12] | Thongke, "CESTI," 15 07 2021. [Online]. Available: https://thongke.cesti.gov.vn/dich-vu-thong-ke/tai-lieu-phan-tich-thong-ke/845-thong-ke-mo-ta-trong-nghien-cuu-dai-luong-do-phan-tan. [Accessed 30 04 2025]. |

PHỤ LỤC

Link video bài báo cáo: <https://www.youtube.com/watch?v=Dec-kaBHxDw>

1. Import

# Import thư viện cần dùng

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

import scipy.stats as stats

import datetime as dt

%matplotlib inline

import statsmodels.api as sm

import statsmodels.formula.api as smf

import missingno as msno

import plotly.express as px

import plotly.graph\_objects as go

import datetime

from sklearn.compose import make\_column\_transformer

# Scaling

from sklearn.preprocessing import scale

from sklearn.preprocessing import StandardScaler

from sklearn.preprocessing import PolynomialFeatures

from sklearn.preprocessing import OneHotEncoder

from sklearn.preprocessing import PowerTransformer

from sklearn.preprocessing import MinMaxScaler

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import RobustScaler

# Modelling

from sklearn.cluster import KMeans

from sklearn.decomposition import PCA

# Importing plotly and cufflinks in offline mode

import cufflinks as cf

import plotly.offline

cf.go\_offline()

cf.set\_config\_file(offline=False, world\_readable=True)

import plotly.graph\_objects as go

# Ignore Warnings

import warnings

warnings.filterwarnings("ignore")

warnings.warn("this will not show")

# Figure&Display options

plt.rcParams["figure.figsize"] = (16, 9)

pd.set\_option('max\_colwidth',200)

pd.set\_option('display.max\_rows', 1000)

pd.set\_option('display.max\_columns', 200)

pd.set\_option('display.float\_format', lambda x: '%.2f' % x)

!pip install termcolor

!pip install colorama

import colorama

from colorama import Fore, Style  # makes strings colored

from termcolor import colored

from termcolor import cprint

import ipywidgets

from ipywidgets import interact

!pip install wordcloud

from wordcloud import WordCloud

!pip install squarify

import squarify as sq

# 1. Import dữ liệu:

import pandas as pd

df = pd.read\_excel('Group-12-48K21.1.xlsx')

df

1. Tiền xử lý dữ liệu
   1. Thống kê mô tả

# 1. Một số hàm sử dụng trong tiền xử lý dữ liệu

# Hàm kiểm tra các giá trị thiếu trong DataFrame

def missing\_values(df):

    # Tính số lượng và tỷ lệ % các giá trị bị thiếu theo từng cột

    missing\_number = df.isnull().sum().sort\_values(ascending=False)

    missing\_percent = (df.isnull().sum() / len(df)\*100).sort\_values(ascending=False)

    # Gộp số lượng và phần trăm vào cùng một bảng

    missing\_values = pd.concat([missing\_number, missing\_percent], axis=1, keys=['Missing\_Number', 'Missing\_Percent'])

    return missing\_values[missing\_values['Missing\_Number']>0]

# Hàm in thông tin tổng quan về DataFrame

def first\_looking(df):

    # Đổi tên cột cho chuẩn trước

    df.columns = df.columns.str.lower().str.replace('&', '\_').str.replace(' ', '\_')

    # In ra các cột hiện tại

    print(colored("Current Columns:", attrs=['bold']), list(df.columns))

    print(colored('\*'\*100, 'red', attrs=['bold']))

    # Group by theo mã khách hàng

    grouped\_df = df.groupby('mã\_khách\_hàng').agg({

        'mã\_đơn\_hàng': 'nunique',               # Tổng số lượng mua

        'đơn\_giá': 'mean',          # Trung bình đơn giá

        'thành\_tiền': 'sum'         # Tổng thành tiền

    }).reset\_index()

    # In thông tin tổng quan

    print(colored("Shape:", attrs=['bold']), grouped\_df.shape, '\n',

          colored('\*'\*100, 'red', attrs=['bold']),

          colored("\nInfo:\n", attrs=['bold']), sep='')

    print(grouped\_df.info(), '\n',

          colored('\*'\*100, 'red', attrs=['bold']), sep='')

    # Số lượng giá trị duy nhất

    print(colored("Number of Uniques:\n", attrs=['bold']), grouped\_df.nunique(), '\n',

          colored('\*'\*100, 'red', attrs=['bold']), sep='')

    # Giá trị thiếu

    print(colored("Missing Values:\n", attrs=['bold']), missing\_values(grouped\_df), '\n',

          colored('\*'\*100, 'red', attrs=['bold']), sep='')

    # Tất cả cột sau group

    print(colored("All Columns:", attrs=['bold']), list(grouped\_df.columns), '\n',

          colored('\*'\*100, 'red', attrs=['bold']), sep='')

    # Thống kê mô tả cột số

    print(colored("Descriptive Statistics (Numeric)\n", attrs=['bold']), grouped\_df.describe().round(2), '\n',

          colored('\*'\*100, 'red', attrs=['bold']), sep='')

    # Thống kê mô tả cột chuỗi từ df ban đầu

    categorical\_columns = [

        'thời\_gian\_tạo\_đơn', 'mã\_đơn\_hàng', 'mã\_khách\_hàng', 'tên\_khách\_hàng',

        'mã\_pkkh', 'mô\_tả\_phân\_khúc\_khách\_hàng', 'mã\_nhóm\_hàng', 'tên\_nhóm\_hàng',

        'mã\_mặt\_hàng', 'tên\_mặt\_hàng'

    ]

    print(colored("Descriptive Statistics (Categorical Columns)\n", attrs=['bold']),

          df[categorical\_columns].describe(include=object).T, '\n',

          colored('\*'\*100, 'red', attrs=['bold']), sep='')

# Hàm in ra thông tin về các giá trị duy nhất cho tất cả các cột trong DataFrame

def unique\_values(df):

    for col in df.columns:

        print(f"\nColumn name    : {col}")

        print(f"Num of uniques : {df[col].nunique()}")

        print(f"Unique values  : {df[col].unique()}")

# Hàm kiểm tra và loại bỏ các giá trị trùng lặp trong DataFrame

def duplicate\_values(df):

    duplicate\_values = df.duplicated(subset=None, keep='first').sum()

    if duplicate\_values > 0:

        # Xóa các dòng trùng lặp (giữ dòng đầu tiên)

        df.drop\_duplicates(keep='first', inplace=True)

        print(duplicate\_values, colored(" Duplicates were dropped!"),'\n',

              colored('\*'\*100, 'red', attrs=['bold']), sep='')

    else:

        print(colored("There are no duplicates"),'\n',

              colored('\*'\*100, 'red', attrs=['bold']), sep='')

# Hàm loại bỏ các cột không cần thiết

def drop\_columns(df, drop\_columns):

    if drop\_columns !=[]:

        df.drop(drop\_columns, axis=1, inplace=True)

        print(drop\_columns, 'Dropped')

    else:

        print(colored('We will now check the missing values and if necessary, the realted columns will be dropped!', attrs=['bold']),'\n',

              colored('\*'\*100, 'red', attrs=['bold']), sep='')

# Hàm điền giá trị thiếu bằng giá trị xuất hiện nhiều nhất (mode) trong một nhóm theo cột nhóm

def fill\_most(df, group\_col, col\_name):

    for group in list(df[group\_col].unique()):

        cond = df[group\_col]==group

        mode = list(df[cond][col\_name].mode())

        if mode != []:

            df.loc[cond, col\_name] = df.loc[cond, col\_name].fillna(df[cond][col\_name].mode()[0])

        else:

            df.loc[cond, col\_name] = df.loc[cond, col\_name].fillna(df[col\_name].mode()[0])

    print(df[col\_name].value\_counts(dropna=False))

# Hàm kiểm tra các giá trị âm, giá trị bằng 0 và giá trị thiếu trong các cột số

def validate\_numeric\_data(df, cols):

    # Kiểm tra và in thông tin dữ liệu cho các cột

    for col in cols:

        print(f"{col}:")

        print(f" - Số giá trị âm: {(df[col] < 0).sum()}")

        print(f" - Số giá trị bằng 0: {(df[col] == 0).sum()}")

        print(f" - Số giá trị thiếu: {df[col].isna().sum()}")

    print("\nDữ liệu trước khi loại bỏ giá trị bất thường:", df.shape)

    # Loại bỏ các hàng có giá trị âm trong tất cả các cột được chỉ định

    for col in cols:

        df = df[df[col] >= 0]

    print("\nDữ liệu sau khi loại bỏ giá trị bất thường:", df.shape)

# 2. In ra thông tin tổng quan về dữ liệu

first\_looking(df)

# 3.1 Kiểm tra kiểu dữ liệu của các cột trong dữ liệu

df.dtypes

# 3.2 Tách cột thời gian và chuyển đổi kiểu dữ liệu thành date time

# Chuyển cột "Thời gian tạo đơn" thành kiểu datetime

df["thời\_gian\_tạo\_đơn"] = pd.to\_datetime(df["thời\_gian\_tạo\_đơn"])

# Tạo 2 cột mới từ cột "Thời gian tạo đơn"

df["ngày\_tạo\_đơn"] = df["thời\_gian\_tạo\_đơn"].dt.normalize()  # Chỉ lấy phần ngày và giữ kiểu datetime

df["giờ\_tạo\_đơn"] = df["thời\_gian\_tạo\_đơn"].dt.time  # Giữ giờ dưới dạng kiểu time (datetime.time)

# Xóa cột "Thời gian tạo đơn" sau khi đã tạo cột mới.

drop\_columns(df, "thời\_gian\_tạo\_đơn")

# Đưa cột "Ngày tạo đơn" và "Giờ tạo đơn" lên làm 2 cột đầu tiên trong bảng dữ liệu.

column\_order = ["ngày\_tạo\_đơn", "giờ\_tạo\_đơn"] + [col for col in df.columns if col not in ["ngày\_tạo\_đơn", "giờ\_tạo\_đơn"]]

# Sắp xếp lại DataFrame

df = df[column\_order]

df.dtypes

# 4. Kiểm tra giá trị trùng lặp

duplicate\_values(df)

# 5. Kiểm tra giá trị duy nhất ở các cột

unique\_values(df)

# 6.1 Kiểm tra giá trị bị thiếu trong tất cả các cột

missing\_values(df)

# 6.2 Điền giá trị bị thiếu

fill\_most(df, group\_col='mã\_khách\_hàng', col\_name='tên\_khách\_hàng')

# Kiểm tra lại giá trị bị thiếu trong tất cả các cột

missing\_values(df)

# 7. Kiểm tra các cột có kiểu là numeric

validate\_numeric\_data(df, ['sl', 'đơn\_giá', 'thành\_tiền'])

* 1. Tính điểm RFM

# 1. Tiến hành tính chỉ số RFM cho từng khách hàng

# Xác định ngày cuối cùng trong dữ liệu để tính Recency

max\_date = df["ngày\_tạo\_đơn"].max()

# Tính RFM

rfm = df.groupby('mã\_khách\_hàng').agg({

    'ngày\_tạo\_đơn': lambda x: (max\_date - x.max()).days,  # Recency

    'mã\_đơn\_hàng': 'nunique',  # Frequency

    'thành\_tiền': 'sum'  # Monetary

})

# Đổi tên các cột cho đúng với RFM

rfm.columns = ['Recency', 'Frequency', 'Monetary']

# Tính các giá trị phân vị (quantiles) để phân nhóm khách hàng

quantiles = rfm.quantile(q = [0.25, 0.50, 0.75])

# In ra các phân vị và bảng RFM

print(quantiles)

print(rfm)

# 2.1 Định nghĩa Hàm tính toán điểm Recency cho mỗi khách hàng

def recency\_scoring(rfm):

    # Xác định điểm Recency dựa trên giá trị 'Recency'

    if rfm.Recency <= 18.0:

        recency\_score = 4

    elif rfm.Recency <= 71.0:

        recency\_score = 3

    elif rfm.Recency <= 200.0:

        recency\_score = 2

    else:

        recency\_score = 1

    return recency\_score

# Áp dụng hàm recency\_scoring lên từng dòng của DataFrame rfm

rfm['Recency\_Score'] = rfm.apply(recency\_scoring, axis=1)

# Lấy ngẫu nhiên 10 dòng để kiểm tra

rfm.sample(10)

# 2.2 Đếm số lượng và vẽ biểu đồ Recency\_Score

# Đếm số lượng các giá trị khác nhau trong cột 'Recency\_Score' và sắp xếp theo chỉ số

recency\_counts = rfm['Recency\_Score'].value\_counts().sort\_index()

# Thiết lập kích thước của biểu đồ

plt.figure(figsize=(8, 8))

# Vẽ biểu đồ tròn

plt.pie(

    recency\_counts,                  # Giá trị cần vẽ

    labels=recency\_counts.index,      # Gán nhãn cho từng phần

    autopct='%1.1f%%',                # Hiển thị phần trăm với 1 số thập phân

    startangle=140,                   # Góc bắt đầu vẽ

    colors=plt.cm.Paired.colors       # Sử dụng bảng màu Paired

)

# Thêm tiêu đề cho biểu đồ

plt.title('Distribution of Recency Score \n')

# Đảm bảo biểu đồ tròn không bị méo

plt.axis('equal')

# Hiển thị biểu đồ

plt.show()

# In ra bảng đếm số lượng của từng điểm Recency

print(recency\_counts)

# 3.1 Định nghĩa hàm tính điểm Frequency cho mỗi khách hàng

def frequency\_scoring(rfm):

    # Xác định điểm Frequency dựa trên giá trị 'Frequency'

    if rfm.Frequency <= 1:

        frequency\_score = 1

    elif rfm.Frequency <= 2:

        frequency\_score = 2

    elif rfm.Frequency <= 4:

        frequency\_score = 3

    else:

        frequency\_score = 4

    return frequency\_score

# Áp dụng hàm 'frequency\_scoring' để tính điểm Frequency\_Score cho từng khách hàng

rfm['Frequency\_Score'] = rfm.apply(frequency\_scoring, axis=1)

# Lấy mẫu 10 dòng ngẫu nhiên để kiểm tra

rfm.sample(10)

# 3.2 Đếm số lượng và vẽ biểu đồ Recency\_Score

# Đếm số lượng các giá trị khác nhau trong cột 'Frequency\_Score' và sắp xếp theo chỉ số

frequency\_counts = rfm['Frequency\_Score'].value\_counts().sort\_index()

# Thiết lập kích thước của biểu đồ

plt.figure(figsize=(8, 8))

# Vẽ biểu đồ tròn (pie chart) cho phân phối Frequency Score

plt.pie(

    frequency\_counts,                  # Dữ liệu cần vẽ

    labels=frequency\_counts.index,      # Nhãn cho từng phần của biểu đồ

    autopct='%1.1f%%',                  # Hiển thị phần trăm với 1 chữ số thập phân

    startangle=140,                     # Góc bắt đầu vẽ biểu đồ

    colors=plt.cm.Paired.colors         # Sử dụng bảng màu Paired cho các phần

)

# Thêm tiêu đề cho biểu đồ

plt.title('Distribution of Frequency Score \n')

# Đảm bảo biểu đồ tròn không bị méo

plt.axis('equal')

# Hiển thị biểu đồ

plt.show()

# In ra bảng đếm số lượng của từng điểm Frequency

print(frequency\_counts)

# 4.1 Định nhĩa Hàm tính toán điểm Monetary cho mỗi khách hàng

def monetary\_scoring(rfm):

    # Xác định điểm Monetary dựa trên giá trị 'Monetary'

    if rfm.Monetary >= 614000:

        monetary\_score = 4

    elif rfm.Monetary >= 249000:

        monetary\_score = 3

    elif rfm.Monetary >= 131000:

        monetary\_score = 2

    else:

        monetary\_score = 1

    return monetary\_score

# Áp dụng hàm 'monetary\_scoring' lên toàn bộ DataFrame 'rfm' để tính điểm Monetary\_scoring

rfm['Monetary\_Score'] = rfm.apply(monetary\_scoring, axis=1)

rfm.sample(10)

# 4.2 Đếm số lượng và vẽ biểu đồ Monetary\_Score

# Đếm số lượng các giá trị khác nhau trong cột 'Monetary\_Score' và sắp xếp theo chỉ số

monetary\_counts = rfm['Monetary\_Score'].value\_counts().sort\_index()

# Thiết lập kích thước cho biểu đồ

plt.figure(figsize=(8, 8))

# Vẽ biểu đồ tròn (pie chart) cho phân phối Monetary Score

plt.pie(

    monetary\_counts,                  # Dữ liệu cần vẽ

    labels=monetary\_counts.index,      # Nhãn cho từng phần của biểu đồ (dùng monetary\_counts thay vì recency\_counts)

    autopct='%1.1f%%',                 # Hiển thị phần trăm với 1 chữ số thập phân

    startangle=140,                    # Góc bắt đầu vẽ biểu đồ

    colors=plt.cm.Paired.colors        # Sử dụng bảng màu Paired cho các phần

)

# Thêm tiêu đề cho biểu đồ

plt.title('Distribution of Monetary Score \n')

# Đảm bảo biểu đồ tròn không bị méo

plt.axis('equal')

# Hiển thị biểu đồ

plt.show()

# In ra bảng đếm số lượng của từng điểm Monetary

print(monetary\_counts)

# 5.1 Định nghĩa hàm tính điểm RFM (Recency, Frequency, Monetary) cho khách hàng

def rfm\_scoring(customer):

    # Tính toán điểm RFM bằng cách nối các giá trị của Recency\_Score, Frequency\_Score, và Monetary\_Score

    return str(int(customer['Recency\_Score'])) + str(int(customer['Frequency\_Score'])) + str(int(customer['Monetary\_Score']))

# Áp dụng hàm 'rfm\_scoring' vào toàn bộ DataFrame 'rfm' để tính điểm RFM cho mỗi khách hàng

rfm['Customer\_RFM\_Score'] = rfm.apply(rfm\_scoring, axis=1)

# In ra một số mẫu ngẫu nhiên trong DataFrame 'rfm' để kiểm tra kết quả

rfm.sample(8)

import plotly.io as pio

pio.renderers.default = 'colab'

# 5.2 Sử dụng plotly để tạo một biểu đồ cột hiển thị phân phối điểm RFM

# Đếm số lượng khách hàng cho mỗi điểm RFM trong cột 'Customer\_RFM\_Score' và sắp xếp theo chỉ số

score\_counts = rfm['Customer\_RFM\_Score'].value\_counts().sort\_index()

# Tạo biểu đồ cột (bar chart) với Plotly

fig = px.bar(

    x=score\_counts.index,                # Trục x: các giá trị điểm RFM (Customer\_RFM\_Score)

    y=score\_counts.values,               # Trục y: số lượng khách hàng cho mỗi điểm RFM

    title='Customer RFM Score Distribution',  # Tiêu đề cho biểu đồ

    labels={'x': 'Customer\_RFM\_Score', 'y': 'Counts'}  # Đặt nhãn cho các trục x và y

)

# Hiển thị biểu đồ

fig.show()

# 6.1 Tạo DataFrame với các phân khúc khách hàng và các biểu thức RFM

segments = {

    'Customer\_Segment': [

        'Champion',

        'Top Loyal Customer',

        'Loyal Customer',

        'Top Recent Customer',

        'Recent Customer',

        'Top Customer Needed Attention',

        'Customer Needed Attention',

        'Top Lost Customer',

        'Lost Customer'

    ],

    'RFM': [

        '(2|3|4)-(4)-(4)',        # Champion

        '(3)-(1|2|3|4)-(3|4)',    # Top Loyal Customer

        '(3)-(1|2|3|4)-(1|2)',    # Loyal Customer

        '(4)-(1|2|3|4)-(3|4)',    # Top Recent Customer

        '(4)-(1|2|3|4)-(1|2)',    # Recent Customer

        '(2|3)-(1|2|3|4)-(3|4)',  # Top Customer Needed Attention

        '(2|3)-(1|2|3|4)-(1|2)',  # Customer Needed Attention

        '(1)-(1|2|3|4)-(3|4)',    # Top Lost Customer

        '(1)-(1|2|3|4)-(1|2)'     # Lost Customer

    ]

}

segments

# 6.2 Định nghĩa hàm phân loại khách hàng dựa trên điểm số RFM

def categorizer(rfm):

# Duyệt qua các điều kiện phân loại

    if (rfm[0] in ['2', '3', '4']) & (rfm[1] in ['4']) & (rfm[2] in ['4']):

        rfm = 'Champion'

    elif (rfm[0] in ['3']) & (rfm[1] in ['1', '2', '3', '4']) & (rfm[2] in ['3', '4']):

        rfm = 'Top Loyal Customer'

    elif (rfm[0] in ['3']) & (rfm[1] in ['1', '2', '3', '4']) & (rfm[2] in ['1', '2']):

        rfm = 'Loyal Customer'

    elif (rfm[0] in ['4']) & (rfm[1] in ['1', '2', '3', '4']) & (rfm[2] in ['3', '4']):

        rfm = 'Top Recent Customer'

    elif (rfm[0] in ['4']) & (rfm[1] in ['1', '2', '3', '4']) & (rfm[2] in ['1', '2']):

        rfm = 'Recent Customer'

    elif (rfm[0] in ['2', '3']) & (rfm[1] in ['1', '2', '3', '4']) & (rfm[2] in ['3', '4']):

        rfm = 'Top Customer Needed Attention'

    elif (rfm[0] in ['2', '3']) & (rfm[1] in ['1', '2', '3', '4']) & (rfm[2] in ['1', '2']):

        rfm = 'Customer Needed Attention'

    elif (rfm[0] in ['1']) & (rfm[1] in ['1', '2', '3', '4']) & (rfm[2] in ['3', '4']):

        rfm = 'Top Lost Customer'

    elif (rfm[0] in ['1']) & (rfm[1] in ['1', '2', '3', '4']) & (rfm[2] in ['1', '2']):

        rfm = 'Lost Customer'

    return rfm

# 6.3 Áp dụng hàm categorizer vào cột 'Customer\_RFM\_Score' để phân loại khách hàng

rfm['Customer\_Category'] = rfm["Customer\_RFM\_Score"].apply(categorizer)

rfm

# 6.4 Vẽ biểu đồ từng nhóm phân loại

import matplotlib.pyplot as plt

# Đếm số lượng khách hàng theo từng nhóm phân loại

category\_counts = rfm['Customer\_Category'].value\_counts()

# Tính tỷ lệ phần trăm khách hàng theo từng nhóm phân loại

category\_percent = rfm['Customer\_Category'].value\_counts(normalize=True) \* 100

# Thiết lập kích thước của biểu đồ

plt.figure(figsize=(17, 9))

# 6.4.1 Biểu đồ cột

# Vẽ biểu đồ cột để hiển thị số lượng khách hàng trong mỗi nhóm

plt.subplot(1, 2, 1)

category\_counts.plot(kind='bar', color='#66c2a5', edgecolor='black')

plt.title('Số lượng khách hàng theo nhóm')

plt.xlabel('Nhóm khách hàng')

plt.ylabel('Số lượng')

plt.xticks(rotation=45)

# 6.4.2 Biểu đồ tròn

# Vẽ biểu đồ tròn để hiển thị tỷ lệ phần trăm khách hàng trong mỗi nhóm

plt.subplot(1, 2, 2)

plt.pie(category\_percent,

        labels=category\_percent.index,

        autopct='%1.1f%%',

        startangle=140,

        colors=plt.cm.Pastel1.colors)

plt.title('Tỷ lệ phần trăm khách hàng theo nhóm')

plt.axis('equal')

# Tự động điều chỉnh bố cục của các biểu đồ để tránh bị chồng lên nhau

plt.tight\_layout()

# Hiển thị các biểu đồ

plt.show()

* 1. Normalization và Outliers

# 1. Vẽ biểu đồ histogram biểu diễn biểu đồ trước khi chuẩn hóa và xử lý outliers

import matplotlib.pyplot as plt

# Chọn 3 cột RFM

rfm\_filtered = rfm[['Recency', 'Frequency', 'Monetary']]

# Vẽ histogram cho 3 cột RFM

ax = rfm\_filtered.hist(bins=50, figsize=(12, 8), layout=(3, 1))

# Thêm tiêu đề

plt.suptitle("Histograms of RFM Variables", fontsize=16)

# Điều chỉnh bố cục để tiêu đề không bị che khuất

plt.tight\_layout(rect=[0, 0.03, 1, 0.95])

# Hiển thị biểu đồ

plt.show()

# 2.1 Thiết lập ngưỡng độ lệch (skewness) để xác định các cột có độ lệch cao

skew\_limit = 0.75 # Đây là ngưỡng độ lệch để đánh giá, độ lệch tuyệt đối dưới 1 có thể chấp nhận được với các mô hình tuyến tính

# Tính độ lệch của các cột Recency, Frequency, Monetary

skew\_vals = rfm[['Recency', 'Frequency', 'Monetary']].skew()

# Lọc ra các cột có độ lệch tuyệt đối lớn hơn skew\_limit

skew\_cols = skew\_vals[abs(skew\_vals)> skew\_limit].sort\_values(ascending=False)

skew\_cols

# 2.2 Xử lý độ lệch với Log Transformation

# Tạo một bản sao của các cột có độ lệch cao (skewed columns) từ dataframe rfm

rfm\_log = rfm[skew\_cols.index].copy()

# Duyệt qua từng tên cột có độ lệch cao

for col in skew\_cols.index.values:

    # Áp dụng log transformation: log1p(x) = log(1 + x), giúp giảm độ lệch

    rfm\_log[col] = rfm\_log[col].apply(np.log1p)

    # In ra độ lệch mới (sau khi đã log transform) để kiểm tra sự cải thiện

    print(f"Skewness of {col}: {rfm\_log[col].skew()}")

# 2.3 Diễn giải độ lệch (skewness) sau khi log transform

for skew in rfm\_log.skew():  # Lặp qua độ lệch của từng cột đã log-transform

    # Nếu độ lệch nằm trong khoảng -0.75 đến 0.75: xem như xấp xỉ đối xứng (symmetric)

    if -0.75 < skew < 0.75:

        print ("A skewness value of", '\033[1m', Fore.GREEN, skew, '\033[0m',

               "means that the distribution is approx.", '\033[1m', Fore.GREEN, "symmetric", '\033[0m')

    # Nếu độ lệch chỉ hơi nghiêng (từ 0.75 đến 1 hoặc -0.75 đến -1): lệch trung bình

    elif  -0.75 < skew < -1.0 or 0.75 < skew < 1.0:

        print ("A skewness value of", '\033[1m', Fore.YELLOW, skew, '\033[0m',

               "means that the distribution is approx.", '\033[1m', Fore.YELLOW, "moderately skewed", '\033[0m')

    # Ngược lại: nếu độ lệch lớn hơn ±1 thì là lệch mạnh (highly skewed)

    else:

        print ("A skewness value of", '\033[1m', Fore.RED, skew, '\033[0m',

               "means that the distribution is approx.", '\033[1m', Fore.RED, "highly skewed", '\033[0m')

import matplotlib.pyplot as plt

# 2.4 Vẽ biểu đồ histogram cho các cột trong rfm\_log

rfm\_log.hist(bins=50, figsize=(12, 8), layout=(3, 1))

# Thêm tiêu đề cho biểu đồ

plt.suptitle("Histograms of Log-Transformed RFM Variables")

# Hiển thị biểu đồ

plt.show()

# 3.1 Vẽ các boxplot cho các biến trong rfm\_log trước khi xử lý outliers

# Tạo figure với kích thước 16x6 inches

plt.figure(figsize=(16, 6))

# Vòng lặp qua từng cột trong dataframe rfm\_log

for i, col in enumerate(rfm\_log.columns):

    # Tạo từng subplot cho mỗi biến, xếp theo 1 dòng, nhiều cột (tùy theo số biến)

    plt.subplot(1, len(rfm\_log.columns), i + 1)

    # Vẽ boxplot cho biến hiện tại

    sns.boxplot(y=rfm\_log[col])

    # Đặt tiêu đề cho từng biểu đồ con

    plt.title(f'Boxplot of {col}')

    # Điều chỉnh bố cục cho đẹp, tránh chồng lấn

    plt.tight\_layout()

# Đặt tiêu đề chung cho toàn bộ figure (lưu ý y > 1 để tiêu đề không bị đè lên các plot)

plt.suptitle('Boxplots of Log-Transformed RFM Variables', y=1.05, fontsize=16)

# Hiển thị biểu đồ

plt.show()

# 3.2 Vẽ biểu đồ Boxplot sau khi loại bỏ outlier bằng phương pháp IQR

# Tính Q1 (quartile 25%) cho từng biến trong rfm\_log

Q1 = rfm\_log.quantile(0.25)

# Tính Q3 (quartile 75%) cho từng biến trong rfm\_log

Q3 = rfm\_log.quantile(0.75)

# Tính khoảng IQR (Interquartile Range) = Q3 - Q1

IQR = Q3 - Q1

# Lọc ra các dòng KHÔNG phải là outlier:

# Điều kiện: giữ lại những dòng mà tất cả giá trị đều NẰM TRONG [Q1 - 1.5\*IQR, Q3 + 1.5\*IQR]

rfm\_log\_filtered = rfm\_log[~((rfm\_log < (Q1 - 1.5 \* IQR)) | (rfm\_log > (Q3 + 1.5 \* IQR))).any(axis=1)]

# Vẽ lại boxplot sau khi đã loại bỏ outliers

plt.figure(figsize=(16, 6))

# Vòng lặp qua từng biến để vẽ boxplot

for i, col in enumerate(rfm\_log\_filtered.columns):

    plt.subplot(1, len(rfm\_log\_filtered.columns), i + 1)

    # Vẽ boxplot cho từng biến với màu 'skyblue'

    sns.boxplot(y=rfm\_log\_filtered[col], color='skyblue')

    # Đặt tiêu đề cho từng biểu đồ

    plt.title(f'Boxplot of {col}')

    # Căn chỉnh layout cho đẹp

    plt.tight\_layout()

# Đặt tiêu đề tổng cho figure

plt.suptitle('Boxplots of Log-Transformed RFM Variables (Outliers Removed)', y=1.05, fontsize=16)

# Hiển thị tất cả biểu đồ

plt.show()

# 4.1 Sử dụng Power Transformer với phương pháp Yeo-Johnson để xử lý độ lệch (skewness) của các dữ liệu trong bộ dữ liệu

from sklearn.preprocessing import PowerTransformer

# Lấy lại cột 'mã\_khách\_hàng' từ chỉ mục (index)

customer\_ids = rfm\_log.index

# Lấy dữ liệu cần biến đổi (các cột có độ lệch)

rfm\_before\_trans = rfm\_log[skew\_cols.index].copy()

# Áp dụng PowerTransformer

pt = PowerTransformer(method='yeo-johnson')

trans = pt.fit\_transform(rfm\_before\_trans)

# Tạo DataFrame mới từ kết quả biến đổi và gán lại tên cột

rfm\_trans = pd.DataFrame(trans, columns=skew\_cols.index)

# Thêm cột 'mã\_khách\_hàng' vào DataFrame đã biến đổi

rfm\_trans['mã\_khách\_hàng'] = customer\_ids

# Hiển thị kết quả

print(rfm\_trans.head())

# Tính độ lệch (skewness) cho các cột số trong rfm\_trans (loại bỏ cột mã\_khách\_hàng)

print(rfm\_trans.drop(columns=['mã\_khách\_hàng']).skew())

# 4.2 Diễn giải độ lệch (Skewness)

# Vòng lặp qua từng giá trị skewness của các cột trong rfm\_trans (bỏ cột 'mã\_khách\_hàng' vì không phải dữ liệu số)

for skew in rfm\_trans.drop(columns=['mã\_khách\_hàng']).skew():

    # Nếu độ lệch nằm trong khoảng (-0.75, 0.75), tức là phân phối gần đối xứng

    if -0.75 < skew < 0.75:

        print ("A skewness value of", '\033[1m', Fore.GREEN, skew, '\033[0m',

               "means that the distribution is approx.", '\033[1m', Fore.GREEN, "symmetric", '\033[0m')

    # Nếu độ lệch nằm trong khoảng (-1.0, -0.75) hoặc (0.75, 1.0), tức là phân phối hơi lệch

    elif  -0.75 < skew < -1.0 or 0.75 < skew < 1.0:

        print ("A skewness value of", '\033[1m', Fore.YELLOW, skew, '\033[0m',

               "means that the distribution is approx.", '\033[1m', Fore.YELLOW, "moderately skewed", '\033[0m')

    # Các trường hợp còn lại: độ lệch lớn => phân phối bị lệch nhiều

    else:

        print ("A skewness value of", '\033[1m', Fore.RED, skew, '\033[0m',

               "means that the distribution is approx.", '\033[1m', Fore.RED, "highly skewed", '\033[0m')

import matplotlib.pyplot as plt

# 4.3 Vẽ biểu đồ histogram cho 3 cột RFM (đã log transform)

rfm\_trans.hist(bins=50, figsize=(12, 8), layout=(3, 1))

# Thêm tiêu đề chung cho toàn bộ biểu đồ

plt.suptitle("Histogram of RFM Transformed Features", fontsize=16)

# Hiển thị biểu đồ

plt.tight\_layout(rect=[0, 0.03, 1, 0.95])  # Để tránh tiêu đề bị đè

plt.show()

from sklearn.cluster import KMeans, AgglomerativeClustering

from pyclustertend import hopkins

from sklearn.preprocessing import scale

from sklearn.metrics.cluster import adjusted\_rand\_score

from sklearn.metrics import silhouette\_samples,silhouette\_score

from scipy.cluster.hierarchy import linkage, dendrogram

from yellowbrick.cluster import KElbowVisualizer

# 5.1 Chuẩn hóa dữ liệu bằng StandardScaler

from sklearn.preprocessing import StandardScaler

# Đảm bảo chỉ huấn luyện trên các cột số

scaler = StandardScaler()

scaler.fit(rfm\_trans.drop(columns=['mã\_khách\_hàng']))

# Áp dụng chuẩn hóa trên các cột số, giữ lại cột 'mã\_khách\_hàng'

rfm\_scaled = scaler.transform(rfm\_trans.drop(columns=['mã\_khách\_hàng']))

# Chuyển đổi lại thành DataFrame với cột 'mã\_khách\_hàng' được giữ lại

rfm\_scaled\_df = pd.DataFrame(rfm\_scaled, columns=rfm\_trans.drop(columns=['mã\_khách\_hàng']).columns)

rfm\_scaled\_df['mã\_khách\_hàng'] = rfm\_trans['mã\_khách\_hàng']

print(rfm\_scaled\_df)

import matplotlib.pyplot as plt

# 5.2 Vẽ biểu đồ histogram cho 3 cột RFM (đã StandardScaler)

# Loại bỏ cột 'mã\_khách\_hàng' để vẽ histogram

rfm\_scaled\_only = rfm\_scaled\_df.drop(columns=['mã\_khách\_hàng'])

# Vẽ histogram

rfm\_scaled\_only.hist(bins=50, figsize=(12,8), layout=(3, 1))

# Thêm tiêu đề chung

plt.suptitle("Histograms of Standardized RFM Variables")

# Hiển thị biểu đồ

plt.tight\_layout()

plt.show()

1. Phân cụm K-means

# 1.1 Vẽ đồ thị elbow để lựa chọn k tối ưu

# Tạo danh sách các giá trị k từ 1 đến 10

ks = range(1, 11)

# Danh sách để lưu giá trị inertia tương ứng với từng k

inertias=[]

# Vòng lặp qua từng giá trị k

for k in ks :

    kc = KMeans(n\_clusters=k, random\_state=42)

    kc.fit(rfm\_scaled)

    inertias.append(kc.inertia\_)

# Vẽ đồ thị Elbow

f, ax = plt.subplots(figsize=(15, 8))

plt.plot(ks, inertias, '-o')

plt.xlabel('Number of clusters, k')

plt.ylabel('Inertia')

plt.xticks(ks)

plt.style.use('ggplot')

plt.title('What is the Best Number for KMeans?')

plt.show()

# 1.2 Xác Định Số Cụm Tối Ưu (k) Qua Biểu Đồ Elbow Trong Phân Cụm KMeans

# Thiết lập màu nền cho figure là màu trắng để biểu đồ nhìn sáng và rõ hơn

plt.rcParams['figure.facecolor'] = 'white'

# Khởi tạo mô hình KMeans (chưa xác định số lượng cụm k)

model = KMeans()

# Khởi tạo KElbowVisualizer để tự động vẽ đường cong Elbow

# k=(1,12) nghĩa là kiểm tra số lượng cụm từ 1 đến 11

visualizer = KElbowVisualizer(model, k=(1, 12))

# Thiết lập kích thước của figure

plt.figure(figsize=(14, 8))

# Fit dữ liệu chuẩn hóa (rfm\_scaled) vào visualizer để vẽ biểu đồ inertia theo từng k

visualizer.fit(rfm\_scaled)

# Hiển thị biểu đồ Elbow

visualizer.show()

# 2. Chạy thuật toán K-Means để phân cụm khách hàng

# Khởi tạo mô hình KMeans với 4 cụm

kmeans = KMeans(n\_clusters=4).fit(rfm\_scaled)

# Huấn luyện mô hình và dự đoán nhãn phân cụm cho dữ liệu

kmeans.fit\_predict(rfm\_scaled)

# Lấy nhãn phân cụm từ mô hình

labels = kmeans.labels\_

# Thêm cột 'ClusterID' vào DataFrame rfm\_trans để lưu nhãn phân cụm

rfm\_trans['ClusterID'] = labels

# Hiển thị kết quả DataFrame sau khi thêm cột 'ClusterID'

rfm\_trans

# 2. Vẽ biểu đồ phân phối tỷ lệ phân cụm khách hàng

import plotly.express as px

# Tạo DataFrame từ value\_counts

cluster\_counts = rfm\_trans['ClusterID'].value\_counts().reset\_index()

cluster\_counts.columns = ['ClusterID', 'Count']

# Vẽ biểu đồ tròn

fig = px.pie(cluster\_counts,

             values='Count',

             names='ClusterID',

             title='Predicted Clusters Distribution')

# Hiển thị biểu đồ

fig.show()

# 3. Vẽ biểu đồ 3D trực quan 2 cụm khách hàng

import plotly.graph\_objects as go  # Import thư viện Plotly Graph Objects để tạo biểu đồ 3D

# Dữ liệu

x = rfm\_trans.iloc[:, 0]  # Chọn cột đầu tiên 'Recency' (Độ gần đây)

y = rfm\_trans.iloc[:, 1]  # Chọn cột thứ hai 'Frequency' (Tần suất, đã qua phép log-transformed)

z = rfm\_trans.iloc[:, 2]  # Chọn cột thứ ba 'Monetary' (Giá trị chi tiêu, đã qua phép log-transformed)

# Centroids là tâm của các cụm được KMeans tính toán

centroids = kmeans.cluster\_centers\_

# Khởi tạo một đối tượng figure

fig = go.Figure()

# Thêm trace (dữ liệu) cho biểu đồ 3D

fig.add\_trace(go.Scatter3d(

    x=x,  # Dữ liệu trục x (Recency)

    y=y,  # Dữ liệu trục y (Frequency)

    z=z,  # Dữ liệu trục z (Monetary)

    mode='markers',  # Dạng hiển thị là các điểm (markers)

    marker=dict(

        size=5,  # Kích thước của các điểm

        color=labels,  # Màu sắc của các điểm dựa trên cụm phân loại (labels)

        colorscale='Viridis',  # Sử dụng bảng màu Viridis

        opacity=0.8  # Độ trong suốt của các điểm

    ),

    name='Customers'  # Tên hiển thị cho dữ liệu này

))

# Cập nhật bố cục (layout) của biểu đồ

fig.update\_layout(

    scene=dict(

        xaxis\_title='Recency',  # Tiêu đề trục x

        yaxis\_title='Frequency',  # Tiêu đề trục y

        zaxis\_title='Monetary'  # Tiêu đề trục z

    ),

    title='3D Visualization of RFM Clusters with Centroids'  # Tiêu đề của biểu đồ

)

# Hiển thị biểu đồ

fig.show()

rfm['ClusterID'] = kmeans.labels\_

cluster\_category\_counts = rfm.groupby(['ClusterID', 'Customer\_Category']).size().unstack(fill\_value=0)

# Bảng số lượng khách hàng

styled\_counts = cluster\_category\_counts.style.set\_caption("SỐ LƯỢNG KHÁCH HÀNG THEO CUSTOMER\_CATEGORY TRONG TỪNG CỤM") \

                                             .background\_gradient(cmap='Blues') \

                                             .format("{:.0f}") \

                                             .set\_table\_styles([{"selector": "caption", "props": [("font-size", "16px"), ("font-weight", "bold")]}])

# Bảng tỷ lệ phần trăm (Cần tính tỷ lệ phần trăm)

# Tính tỷ lệ phần trăm cho mỗi cluster và customer category

cluster\_category\_percent = (cluster\_category\_counts / cluster\_category\_counts.sum(axis=1).values[:, None]) \* 100

styled\_percent = cluster\_category\_percent.round(2).style.set\_caption("TỶ LỆ % KHÁCH HÀNG THEO CUSTOMER\_CATEGORY TRONG TỪNG CỤM") \

                                                      .background\_gradient(cmap='Greens') \

                                                      .format("{:.2f}%") \

                                                      .set\_table\_styles([{"selector": "caption", "props": [("font-size", "16px"), ("font-weight", "bold")]}])

# Hiển thị trong notebook với một dòng trống giữa hai bảng

display(styled\_counts)

print("\n")  # In một dòng trống giữa các bảng

display(styled\_percent)

rfm\_trans['Labels'] = rfm\_trans['ClusterID'].map({

    0: 'Nhóm Khách Hàng Tiềm Năng Ổn Định',

    1: 'Nhóm Rủi Ro Cao',

    2: 'Nhóm Khách Hàng Giá Trị Cao',

    3: 'Nhóm Khách Hàng Suy Giảm Tiềm Năng'

})

rfm\_trans.sample(8)

1. Kiểm định

# 1. Kiểm định Levene

from scipy.stats import levene

from sklearn.cluster import KMeans

# Phân cụm với KMeans (4 cụm)

kmeans = KMeans(n\_clusters=4, random\_state=42)

labels = kmeans.fit\_predict(rfm\_scaled)

# Gán nhãn phân cụm vào DataFrame

rfm\_trans['Cluster'] = labels

# Chia dữ liệu theo nhóm

cluster0 = rfm\_trans[rfm\_trans['Cluster'] == 0]

cluster1 = rfm\_trans[rfm\_trans['Cluster'] == 1]

cluster2 = rfm\_trans[rfm\_trans['Cluster'] == 2]

cluster3 = rfm\_trans[rfm\_trans['Cluster'] == 3]

# Kiểm định Levene giữa các cụm cho từng đặc trưng

recency\_levene = levene(cluster0['Recency'], cluster1['Recency'], cluster2['Recency'], cluster3['Recency'])

frequency\_levene = levene(cluster0['Frequency'], cluster1['Frequency'], cluster2['Frequency'], cluster3['Frequency'])

monetary\_levene = levene(cluster0['Monetary'], cluster1['Monetary'], cluster2['Monetary'], cluster3['Monetary'])

# In kết quả

print(f"Recency Levene test - statistic: {recency\_levene.statistic}, p-value: {recency\_levene.pvalue}")

print(f"Frequency Levene test - statistic: {frequency\_levene.statistic}, p-value: {frequency\_levene.pvalue}")

print(f"Monetary Levene test - statistic: {monetary\_levene.statistic}, p-value: {monetary\_levene.pvalue}")

import numpy as np

import pandas as pd

from scipy import stats

from sklearn.cluster import KMeans

# Phân cụm với KMeans (4 cụm)

kmeans = KMeans(n\_clusters=4, random\_state=42)

labels = kmeans.fit\_predict(rfm\_scaled)

# Gán nhãn phân cụm vào DataFrame

rfm\_trans['Cluster'] = labels

# Chia dữ liệu theo nhóm

cluster0 = rfm\_trans[rfm\_trans['Cluster'] == 0]

cluster1 = rfm\_trans[rfm\_trans['Cluster'] == 1]

cluster2 = rfm\_trans[rfm\_trans['Cluster'] == 2]

cluster3 = rfm\_trans[rfm\_trans['Cluster'] == 3]

# Hàm tính Welch’s ANOVA

def welch\_anova(\*groups):

    k = len(groups)

    ni = np.array([len(g) for g in groups])

    mi = np.array([np.mean(g) for g in groups])

    si2 = np.array([np.var(g, ddof=1) for g in groups])  # Phương sai mẫu

    # Trọng số ngược phương sai

    wi = ni / si2

    # Trung bình có trọng số

    x\_bar = np.sum(wi \* mi) / np.sum(wi)

    # F statistic

    numerator = np.sum(wi \* (mi - x\_bar)\*\*2) / (k - 1)

    # Bậc tự do hiệu chỉnh

    denom = np.sum((1 - wi / np.sum(wi))\*\*2 / (ni - 1))

    df = (k - 1) / denom

    F\_stat = numerator

    p\_value = 1 - stats.f.cdf(F\_stat, dfn=k - 1, dfd=df)

    return F\_stat, p\_value

# Kiểm định Welch’s ANOVA cho từng đặc trưng

recency\_F, recency\_p = welch\_anova(cluster0['Recency'], cluster1['Recency'], cluster2['Recency'], cluster3['Recency'])

frequency\_F, frequency\_p = welch\_anova(cluster0['Frequency'], cluster1['Frequency'], cluster2['Frequency'], cluster3['Frequency'])

monetary\_F, monetary\_p = welch\_anova(cluster0['Monetary'], cluster1['Monetary'], cluster2['Monetary'], cluster3['Monetary'])

# In kết quả Welch’s ANOVA

print(f"Recency Welch’s ANOVA - F: {recency\_F:.4f}, p-value: {recency\_p:.4f}")

print(f"Frequency Welch’s ANOVA - F: {frequency\_F:.4f}, p-value: {frequency\_p:.4f}")

print(f"Monetary Welch’s ANOVA - F: {monetary\_F:.4f}, p-value: {monetary\_p:.4f}")

merged\_df = pd.merge(df, rfm\_trans, on='mã\_khách\_hàng', how='inner')

1. Xuất file

# 1. Xuất file CSV

merged\_df.to\_csv('Phan\_cum.csv', index=False)  # Xuất ra file CSV