

TRƯỜNG ĐẠI HỌC SÀI GÒN

KHOA CÔNG NGHỆ THÔNG TIN



KHAI PHÁ DỮ LIỆU

ĐỀ TÀI : THUẬT TOÁN PHÂN CỤM DỮ LIỆU K-MEANS ÁP DỤNG TRONG VIỆC PHÂN LOẠI KHÁCH HÀNG

- | | |
|----------------------|------------|
| 1) Nguyễn Hữu Thắng | 3118410402 |
| 2) Nguyễn Minh Thông | 3118410416 |
| 3) Trần Huy Khánh | 3118410191 |

GIẢNG VIÊN: TRỊNH TÂN ĐẠT

TP. HCM tháng 5 / 2022

MỤC LỤC

DANH SÁCH CÁC HÌNH	3
DANH SÁCH CÁC BẢNG	6
LỜI NÓI ĐẦU	7
CHƯƠNG 1 : TỔNG QUAN	9
1.1 Giới thiệu về học máy	9
1.1.1 Khái niệm.....	9
1.1.2 Phân nhóm các thuật toán học máy	10
1.1.3 Nhu cầu thực tế	12
1.1.4 Bài toán phân loại khách hàng.....	13
1.1.5 Những khó khăn của bài toán phân loại khách hàng	13
1.1.6 Các ứng dụng của học máy hiện nay	14
1.2 Các hướng tiếp cận của bài toán phân loại khách hàng	14
1.2.1 Các hướng tiếp cận	14
1.2.2 Một số công trình nghiên cứu hiện nay	16
CHƯƠNG 2 : CƠ SỞ LÝ THUYẾT	18
2.1 Tìm hiểu về ngôn ngữ lập trình Python	18
2.1.1 Python là gì ?	18
2.1.2 Lịch sử hình thành (Các phiên bản).....	18
2.1.3 Các ứng dụng phổ biến	19
2.1.4 Các thư viện thông dụng được sử dụng trong học máy	20
2.2 Tổng quan về thuật toán phân cụm dữ liệu.....	21
2.2.1 Giới thiệu	21
2.2.2 Ưu điểm và khuyết điểm so với các thuật toán phân lớp	22
2.2.3 Các thuật toán phân cụm dữ liệu phổ biến hiện nay.....	23

2.2.4 Thuật toán K-Means	24
2.2.5 Thuật toán Mini Batch K-Means	30
CHƯƠNG 3 : XÂY DỰNG THUẬT GIẢI	33
3.1 Tối ưu hóa việc lựa chọn số cụm dữ liệu	33
3.1.1 Lý do phải tối ưu hóa việc lựa chọn số cụm	33
3.1.2 Thuật toán Elbow	33
3.1.3 Thuật toán Silhouette	35
3.2 Chuẩn hóa dữ liệu	40
3.2.1 Định nghĩa dữ liệu nhiễu	40
3.2.2 Phương pháp phân tích RFM (Mô hình RFM)	40
3.2.3 Loại bỏ các dữ liệu ngoại lai (Outliers)	42
3.3 Gom nhóm cơ sở dữ liệu theo các đặc trưng chung.....	42
3.4 Chọn lựa đặc trưng và phân cụm dữ liệu	43
CHƯƠNG 4 : CÀI ĐẶT CHƯƠNG TRÌNH THỰC NGHIỆM, KẾT QUẢ VÀ ĐÁNH GIÁ	45
4.1 Thu thập và chuẩn hóa tập dữ liệu	45
4.2 Cài đặt chương trình thực nghiệm.....	52
4.3 Kết quả	63
4.4 Kết luận	79
4.4.1 Đánh giá	79
4.4.2 Ưu điểm	80
4.4.3 Khuyết điểm.....	80
4.4.4 Hướng phát triển	81
TÀI LIỆU THAM KHẢO	83

DANH SÁCH CÁC HÌNH

Hình 1.1: Mô hình máy học có giám sát	10
Hình 1.2: Mô hình máy học không giám sát	11
Hình 1.3: Sự khác nhau giữa 2 mô hình máy học	12
Hình 2.1: Minh họa thuật toán K-Means	25
Hình 2.2: Minh họa thuật toán K-Means [2]	29
Hình 2.3: So sánh thuật toán Mini Batch K-Means và K-Means	31
Hình 3.1: Phương pháp Elbow	34
Hình 3.2: Biểu đồ Elbow sử dụng tham số 'Calinski Harabasz'	34
Hình 3.3: Khoảng cách tính theo phương pháp Silhouette	36
Hình 3.4: Điểm Silhouette trung bình	37
Hình 3.5: Điểm Silhouette tương ứng với cụm 3	37
Hình 3.6: Điểm Silhouette tương ứng với cụm 5	38
Hình 3.7: Điểm Silhouette tương ứng với cụm 6	38
Hình 3.8: Điểm Silhouette tương ứng với cụm 2	39
Hình 3.9: Điểm Silhouette tương ứng với cụm 4	39
Hình 3.10: Mô hình RFM	41
Hình 4.1: Kiểm tra giá trị	45
Hình 4.2: Lọc dữ liệu giá trị null	45
Hình 4.3: Kiểm tra giá trị trùng lặp	45
Hình 4.4: Lọc dữ liệu trùng lặp	46
Hình 4.5: Lọc các đơn hàng bị huỷ	46
Hình 4.6: Lọc các sản phẩm có giá trị “Price” bằng 0	46
Hình 4.7: Code làm sạch đầu vào	47
Hình 4.8: Biểu đồ RFM trước khi dữ liệu nhiễu được làm sạch	48
Hình 4.9: Biểu đồ RFM sau khi được làm sạch	49
Hình 4.10 : Gom nhóm theo Tổng chi tiêu	50
Hình 4.11: Kiểm tra dữ liệu để gom nhóm	50
Hình 4.12: Gom nhóm dữ liệu dựa trên mô hình RFM	50
Hình 4.13: Kết quả sau khi thực nghiệm gom nhóm	50

Hình 4.14: Tiến hành gom nhóm theo mô hình RFM nâng cao	51
Hình 4.15: Kết quả thực nghiệm mô hình RFM	51
Hình 4.16: Thông tin về dữ liệu đầu vào	52
Hình 4.17: Import các thư viện cần thiết	53
Hình 4.18: Đọc file dữ liệu dạng xlsx	54
Hình 4.19: Hàm kiểm tra các thông tin cơ bản	54
Hình 4.20: Gom nhóm dữ liệu theo mô hình RFM	55
Hình 4.21: Phân bố của các khách hàng ở các quốc gia khác nhau	55
Hình 4.22: Phần trăm khách hàng đến từ Anh và ngoài nước Anh	55
Hình 4.23: Dữ liệu đánh giá tỉ lệ quay lại của khách hàng	57
Hình 4.24: Phân loại khách hàng thành các bậc dựa trên RFM_Score	57
Hình 4.25: Chuẩn hoá dữ liệu theo Logarit	57
Hình 4.26: Thuật toán Elbow để tìm số cụm phù hợp	58
Hình 4.27: Kiểm tra lại số cụm bằng thuật toán Silhouette	59
Hình 4.28: Phân cụm khách hàng theo 2 thuật toán	59
Hình 4.29: Dữ liệu khách hàng sau khi được phân cụm được xử lý tiếp	59
Hình 4.30: Dữ liệu cho biểu đồ so sánh kết quả giữa các loại phân cụm	60
Hình 4.31: Kết quả phân cụm của K-Means theo mô hình 3D	60
Hình 4.32: Kết quả phân cụm của Mini Batch K-Means theo mô hình 3D	61
Hình 4.33: So sánh chi tiết giữa 2 thuật toán phân cụm khách hàng	61
Hình 4.34: Dữ liệu kết quả được hiển thị trực quan hơn thông qua biểu đồ	62
Hình 4.35: Learning Curves cho 2 thuật toán trên	63
Hình 4.36: Biểu đồ thể hiện sự phân bố khách hàng	64
Hình 4.37: Tỉ lệ phần trăm trong tập dữ liệu đầu vào	64
Hình 4.38: Số lượng khách hàng quay lại	65
Hình 4.39: Tỉ lệ khách hàng quay lại	65
Hình 4.40: Biểu đồ nhiệt thể hiện phần trăm khách hàng quay lại	66
Hình 4.41: Biểu đồ nhiệt thể hiện số lượng sản phẩm đã bán mỗi tháng	66
Hình 4.42: Kết quả phân bậc khách hàng	67
Hình 4.43: Dữ liệu sau khi đã được lọc và chuẩn hoá	68
Hình 4.44: Kết quả thuật toán Elbow tìm số cụm	69

Hình 4.45: Kết quả tìm số cụm của thuật toán Silhouette	70
Hình 4.46: Thời gian thực thi của K-Means	71
Hình 4.47: Thời gian thực thi của Mini Batch K-Means	71
Hình 4.48: Kích thước của mỗi cụm sau khi đã được phân cụm	72
Hình 4.49: So sánh giữa các cách phân cụm khách hàng	72
Hình 4.50: Mô hình 3D phân cụm khách hàng theo thuật toán K-Means	73
Hình 4.51: Mô hình 3D phân cụm theo thuật toán Mini Batch K-Means	74
Hình 4.52: Phân cụm khách hàng theo thuật toán K-Means	75
Hình 4.53: Phân cụm khách hàng theo thuật toán Mini Batch K-Means	76
Hình 4.54: Số lượng khách hàng của mỗi cụm	77
Hình 4.55: So sánh các chỉ số RFM sau khi đã phân cụm	78
Hình 4.56: Learning Curves của K-Means	78
Hình 4.57: Learning Curves của Mini Batch K-Means	79

DANH SÁCH CÁC BẢNG

Bảng 3.1: Ký hiệu Euclidean

43

LỜI NÓI ĐẦU

Xã hội ngày nay càng phát triển thì chất lượng cuộc sống của mỗi người cũng từ đó được tăng theo, kéo theo đó là sự đòi hỏi nhu cầu cơ bản và thiết yếu về tinh thần lẫn vật chất cũng cao hơn. Chẳng hạn như khi xưa chỉ cần nhập từ cần tìm kiếm và hiển thị ra kết quả là đã thỏa mãn nhu cầu về tìm kiếm thông tin, nhưng nay mọi người lại yêu cầu cao hơn về “chất lượng” của kết quả tìm kiếm cũng như thời gian tìm kiếm. Bởi vì xã hội ngày càng bận rộn hơn, năng suất lao động cũng như chất lượng công việc được đặt lên hàng đầu trong khi thời gian thì có giới hạn.

Năm bắt được xu thế đó thì vài năm trở lại đây công nghệ đã có những bước phát triển vượt bậc hơn đáng kể cho với thế kỷ trước. Công nghệ ngày càng phổ biến, dễ tiếp cận đến đại đa số người dùng hơn và không ai có thể phủ nhận được tầm quan trọng của nó trong cuộc sống hằng ngày. Đặc biệt là sự hiện diện của Trí tuệ nhân tạo (AI) sẽ giúp nâng cao năng suất lao động, hoàn thành công việc nhanh, gọn gàng hơn và khám phá ra những giới hạn mới nơi mà nằm ngoài khả năng và sự hiểu biết của con người. Và gần đây thì cụm từ “Machine Learning” đang được rất nhiều người quan tâm và theo dõi.

Machine Learning (ML) là một nhánh của Trí tuệ nhân tạo (AI), thay vì lập trình viên phải viết phần mềm theo các yêu cầu của bên khách hàng hay các hướng dẫn cụ thể để hoàn thành nhiệm vụ đặt ra thì ML lại cho phép máy tính có khả năng cải thiện chính bản thân chúng dựa trên dữ liệu mẫu (training data) hoặc dựa vào kinh nghiệm (những gì đã được học). Từ đó có thể tự đưa ra dự đoán hoặc quyết định để giải quyết vấn đề đặt ra mà không cần được lập trình cụ thể góp phần chuyển đổi lượng dữ liệu khổng lồ thành các tri thức có ích. Con người hiện nay chỉ đang ở những bước đầu tiên của AI hay ML trong khi tiềm năng phát triển của lĩnh vực này là rất lớn trong tương lai và con người còn có thể tiến xa hơn những gì đang có ở hiện tại. Bởi vì nó có thể thực hiện một cách tự động và nhanh chóng để tạo ra các mô hình mẫu cho phép phân tích những dữ liệu có dung lượng lớn, phức tạp và đưa ra những kết quả nhanh chóng và chính xác nơi mà con người không thể thực hiện thủ công.

Một trong những ứng dụng thông dụng và phổ biến của Machine Learning nói chung hay Data Mining nói riêng trong thực tế là phân loại khách hàng (Customer Segmentation). Đặc biệt là trong khoảng thời gian dịch bệnh Covid-19 đang diễn ra phức tạp, mọi người hạn

chế ra ngoài nên việc mua sắm trực tuyến cũng trở thành nhu cầu thiết yếu và cấp bách hơn bao giờ hết. Chính vì lẽ đó mà dữ liệu về người dùng cũng tăng lên chóng mặt từ vài Gigabyte đến vài Terabyte buộc các nhà cung cấp dịch vụ cũng phải tăng khả năng chịu tải cũng như lưu trữ. Việc phân loại khách hàng sẽ giúp các doanh nghiệp tiết kiệm được các khoản chi phí như chi phí marketing nhưng vẫn giữ được phần trăm lợi nhuận của công ty. Phân loại khách hàng sẽ phân tách từ một cụm dữ liệu khách hàng lớn thành nhiều cụm nhỏ khác nhau có cùng chung một vài đặc điểm, việc này sẽ giúp doanh nghiệp đưa ra các chương trình khuyến mãi, ưu đãi đến đúng đối tượng khách hàng nhằm giữ chân các khách hàng cũ cũng như tìm kiếm các khách hàng tiềm năng mới, từ đó nâng cao năng lực cạnh tranh của doanh nghiệp.

Các thuật toán cho việc phân loại khách hàng cũng rất đa dạng và phong phú như thuật toán K-Means, DBSCAN, GMM Algorithm, MeanShift,... Và hiện nay hỗ trợ tốt nhất cho Machine Learning cũng như Data Mining bao gồm hai ngôn ngữ lập trình chính là R và Python. Mỗi thuật toán cũng như ngôn ngữ lập trình đều có ưu và nhược điểm riêng như thời gian tính toán, khả năng tiếp cận, cộng đồng hỗ trợ,... Bài báo cáo này của chúng em sẽ sử dụng ngôn ngữ lập trình Python và thuật toán K-Means để ứng dụng trong việc phân loại khách hàng.

Nhóm chúng em xin gửi lời cảm ơn chân thành đến các thầy cô giáo đang học tập và giảng dạy tại Trường Đại học Sài Gòn nói chung và các thầy cô giáo đang hoạt động tại Khoa Công nghệ thông tin nói riêng đã tận tình giúp đỡ cũng như giảng dạy và truyền đạt những kiến thức hay, bổ ích và quý báu để làm hành trang cho chặng đường đời của chúng em sau này. Đặc biệt là em xin gửi lời cảm ơn đến Thầy Trịnh Tấn Đạt đã tận tình giám sát theo dõi, trực tiếp chỉ bảo, hướng dẫn cũng như truyền đạt những kinh nghiệm thực tế trong công việc trong suốt quá trình nghiên cứu và học tập tại trường của chúng em.

CHƯƠNG 1 : TỔNG QUAN

1.1 Giới thiệu về học máy

1.1.1 Khái niệm

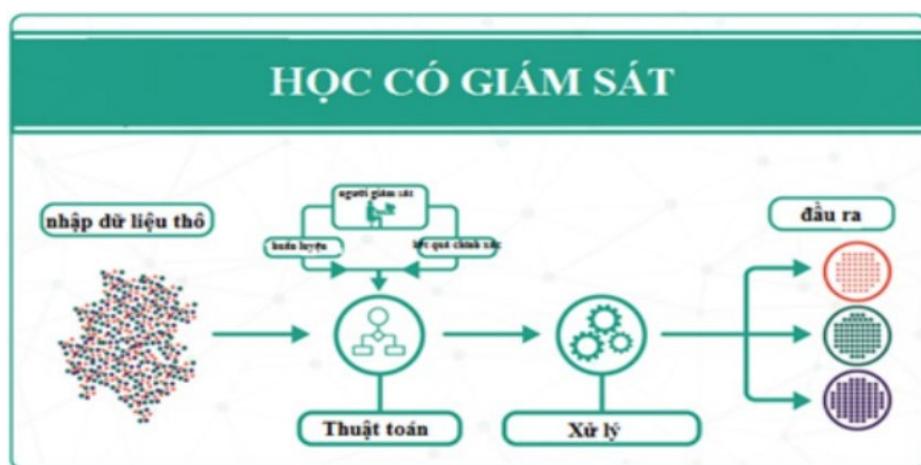
- Máy học là một lĩnh vực nghiên cứu cần rất nhiều thời gian nhưng tiềm năng đem lại là rất lớn cho con người. Nó là một phần nhỏ của AI (Artificial Intelligence) mà hiện đang phổ biến hiện nay. Tập trung vào việc tạo ra các cỗ máy có trí thông minh, suy nghĩ tương tự con người để có thể đưa ra những kết quả, quyết định đúng đắn nhất nhưng không cần phải lập trình một cách cụ thể (ví dụ lập trình các thuật toán, dự báo thời tiết,...). Dữ liệu được máy học tiếp thu và đưa ra chuẩn đoán, từ đó có thể dựa vào các chuẩn đoán mà máy đưa ra để đưa ra quyết định.
- Công nghệ Machine Learning (Máy học) được chia thành 2 loại học máy chính bao gồm “Supervised Learning” và “Unsupervised Learning” (hay với 1 cái tên khác là “Thuật toán máy học có giám sát” và “không giám sát”).
- Trong đó “Supervised Learning” là mỗi data (dữ liệu) được gán một nhãn (label) và thuật toán tự học dựa trên các nhãn này để training đầu vào. (ví dụ của phương thức này là “Phân Loại” và “Hồi Quy”).
- Còn đối với “Unsupervised Learning” là cho máy tính học trên dữ liệu không được gán nhãn và cũng không cần phải giám sát mô hình. Các thuật toán này giúp ích cho việc tìm các dữ liệu được ẩn giấu hay các dữ liệu “unknown patterns”. Từ đó phát hiện ra các nhóm dữ liệu, nhóm tính chất có một vài đặc trưng chung.
- Các mô hình học máy hiện nay yêu cầu lượng dữ liệu đủ lớn để “huấn luyện” và đánh giá mô hình. Điều này được xem như rào cản ở thế kỷ trước khi internet chưa phát triển, việc tiếp cận được với dữ liệu người dùng gặp nhiều khó khăn do các chính sách pháp lý và bảo mật khắt khe cũng như công nghệ không đáp ứng đủ nhu cầu của học máy. Nhưng sự tăng trưởng và phát triển nhanh chóng của các “Data Center” kéo theo “Big Data” đã cung cấp cho học máy một nguồn dữ liệu dồi dào để có thể cải thiện mô hình cũng như kết quả cho ra.
- Học máy vẫn đang còn ở những giai đoạn đầu nên vẫn cần sự can thiệp của con người trong việc tìm hiểu cơ sở dữ liệu và lựa chọn các kỹ thuật phù hợp để phân tích dữ liệu nhằm đem

lại kết quả tối ưu nhất. Đồng thời, trước khi bắt đầu quá trình học thì dữ liệu phải được làm sạch, không có sai lệch và không có dữ liệu giả (hay còn gọi là loại bỏ các outliers).

1.1.2 Phân nhóm các thuật toán học máy

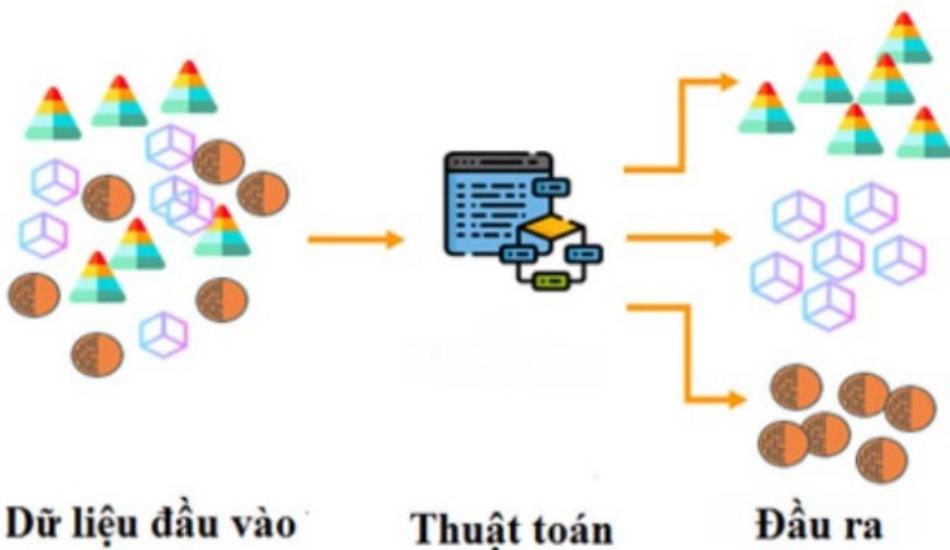
- Có 2 cách thức phân nhóm phổ biến dành cho các thuật toán học máy : “Phân nhóm dựa trên phương thức học” và “Phân nhóm dựa trên chức năng”.
- Trong đó “phân nhóm dựa trên phương thức học” bao gồm :

- Học có giám sát (Supervised Learning) : đây là loại hình máy học có sự giám sát. Dự đoán kết quả đầu ra của một dữ liệu mới được nhập vào khi đã có được lượng thông tin/dữ liệu để học từ trước (quan hệ giữa đầu vào và đầu ra). Là loại hình phổ biến nhất của Machine Learning được ứng dụng rộng rãi trong dự đoán giá chứng khoán, phân loại email,... Các mô hình phổ biến của hình thức học này như SVM, CNN,...



Hình 1.1. Mô hình máy học có giám sát

- Học không giám sát (Unsupervised Learning): đây là loại hình máy học không có sự giám sát, nghĩa là không thể biết được kết quả đầu ra,... mà chỉ biết được kết quả đầu vào (dữ liệu học máy). Mô hình sẽ được huấn luyện cách để tìm ra cấu trúc hoặc mối quan hệ giữa các đầu vào. Từ đó giúp máy học học dựa vào cấu trúc dữ liệu (ví dụ phân nhóm, giảm số chiều dữ liệu, lọc dữ liệu,...). Ứng dụng phổ biến nhất của học không giám sát là gom cụm (cluster) bao gồm các phương pháp như phát hiện điểm bất thường (Anomaly detection), Singular Value Decomposition,...



Hình 1.2. Mô hình máy học không giám sát

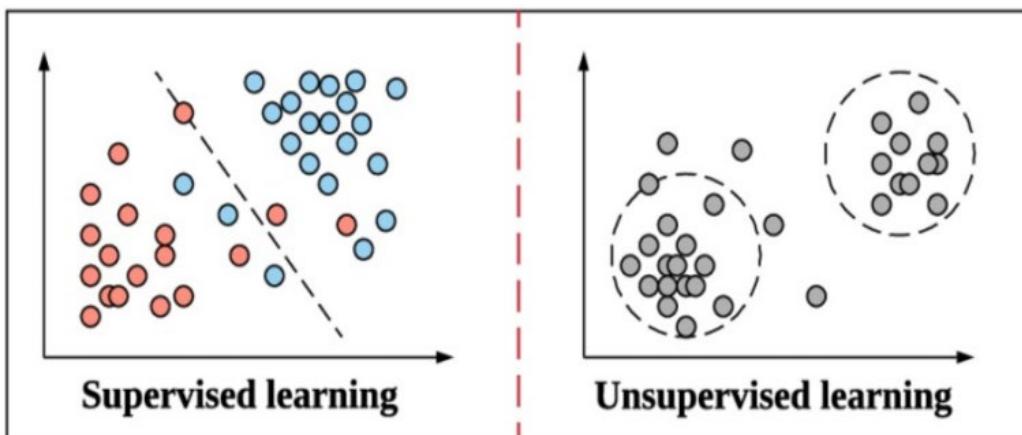
- Học bán giám sát (Semi Supervised Learning) : đây là loại hình học máy khi có một lượng dữ liệu lớn và các dữ liệu này được chia thành 2 phần (1 phần chưa được dán nhãn và 1 phần đã dán nhãn). Ví dụ như trong bài toán chỉ có 1 phần ảnh, văn bản được dán nhãn(Bức ảnh về con người, cây cối,...) và phần lớn phần còn lại được thu thập từ internet(chưa được dán nhãn). Sau đó sử dụng loại hình bán giám sát để tiến hành học máy. Loại hình học này thường khá phổ biến do dữ liệu dán nhãn tồn rất nhiều thời gian và đắt đỏ trong khi dữ liệu chưa dán nhãn có thể được thu thập dễ dàng từ internet với chi phí thấp hơn đáng kể.
- Học tăng cường (Reinforcement Learning): đây là loại hình sẽ giúp học máy xác định tự động các hành vi dựa trên hoàn cảnh để đạt được kết quả tối ưu nhất có thể (Áp dụng cho các bài toán về lý thuyết trò chơi). Khác với học có giám sát thì nó không có cặp dữ liệu gán nhãn trước làm đầu vào và cũng không có cách nào để đánh giá được việc học là đúng hay sai.

- Trong đó “phân nhóm dựa trên chức năng” bao gồm:

- Regression Algorithms
- Classification Algorithms
- Instance-based Algorithms
- Regularization Algorithms
- Bayesian Algorithms

- Clustering Algorithms
- Artificial Neural Network Algorithms
- Dimensionality Reduction Algorithms
- Ensemble Algorithms

- Tuy nhiên trên thực tế, đa số các học máy sẽ được phân chia theo 2 loại chính đó là máy học giám sát và không giám sát. Sự khác nhau giữa 2 loại máy học nè sẽ được biểu diễn thông qua hình sau :



Hình 1.3. Sự khác nhau giữa học giám sát và không giám sát

- Học có giám sát là cách huấn luyện 1 máy học sao cho dữ liệu học có đầu vào và đầu ra là tương ứng với nó. Được lặp đi lặp lại (thử sai) để tìm ra được một cách học cho ra kết quả, quyết định chính xác nhất có thể. Ngược lại, học không giám sát chính là cách huấn luyện 1 máy học sao cho dữ liệu học chỉ có đầu vào, phải tự tìm cách liên kết, kết nối, phân tích cấu trúc dữ liệu để tìm kiếm thông tin cho việc phân loại hoặc phân cụm dữ liệu đó 1 cách chính xác nhất.

1.1.3 Nhu cầu thực tế

- Ngày nay việc áp dụng các công nghệ tiên tiến như Máy học (Machine Learning) hay một số loại khác như DeepLearning , AI,... vào các lĩnh vực khác nhau là điều cần thiết và dần trở nên phổ biến rộng rãi. Chính vì việc công nghệ có những bước phát triển vượt bậc trong những năm trở lại đây (như máy tính lượng tử) nên việc thu thập các thông tin liên quan và cần thiết cho nhu cầu của các cá nhân, doanh nghiệp, tổ chức,... trở nên cần thiết hơn bao giờ hết và đã giúp ích rất nhiều cho nhân loại nói chung và mỗi cá nhân nói riêng. Các lĩnh vực đã áp dụng thành công học máy trong thực tế không thể không kể đến như công nghệ phát

triển trợ lý ảo (Siri của Apple) hay là Cortana (của Microsoft), ngoài ra trong lĩnh vực y học, kinh doanh, sản xuất,... cũng được áp dụng rất nhiều (ví dụ máy học có thể sử dụng để bảo trì, dự đoán và kiểm soát chất lượng sản xuất). Chính vì nhu cầu sử dụng nhiều nên ngày nay công nghệ AI có thể được coi là một chuyên ngành đang rất phổ biến vì có thể áp dụng cho rất nhiều thứ trong đời sống con người giúp cho cuộc sống của mỗi người ngày càng trở nên dễ dàng và tiện lợi hơn.

1.1.4 Bài toán phân loại khách hàng

- Không nằm ngoài những mục đích trên thì dữ liệu khách hàng của các trung tâm mua sắm, các sàn giao dịch thương mại điện tử,... thường có dung lượng rất lớn vượt ngoài khả năng tính toán của con người. Và đây chính là lúc mà Machine Learning thực hiện công việc của mình. Nó sẽ giúp chia khách hàng thành nhiều nhóm khác nhau từ tập dữ liệu ban đầu. Việc phân loại và xác định được các tệp khách hàng là yếu tố hàng đầu ảnh hưởng đến việc hoạt động của doanh nghiệp. Ngoài ra, việc phân tích dựa trên máy học có thể giúp ích được cho nhiều trường hợp khác như đánh giá độ phổ biến,... Chính vì thế, việc áp dụng máy học vào bài toán phân loại khách hàng cũng là một đề tài được quan tâm và bàn luận rất nhiều gần đây. Trong bài báo cáo này chúng em sẽ thực hiện bài toán phân loại khách hàng dựa trên thuật toán khá phổ biến là thuật toán K-Means với sự hỗ trợ của ngôn ngữ Python.

1.1.5 Những khó khăn của bài toán phân loại khách hàng

- Đây là một bài toán khá phổ biến kể cả trên mạng hay ngoài đời thực. Có rất nhiều tài liệu liên quan đã nói về vấn đề này. Tuy nhiên nó đòi hỏi phải có một lượng dữ liệu tương đối lớn để có thể cho máy học một cách chính xác nhất. Ngoài ra, nếu lượng dữ liệu có sự chênh lệch quá lớn do các thành phần ngoại lai gây ra cũng sẽ dẫn đến việc tạo ra một mô hình máy học không chính xác và hiệu quả (sai số quá nhiều). Vì thế cần áp dụng nhiều kỹ thuật như “làm sạch dữ liệu”, “lọc dữ liệu nhiễu”,... Tạo ra một dataset hoàn chỉnh nhất có thể thì mới cho ra được kết quả như kỳ vọng.

- Ngoài ra, việc thu thập dữ liệu cũng có thể gọi là một vấn đề khá khó khăn. Vì đây là bài toán cần dữ liệu thật để khởi tạo cũng như tiến hành phân tích, chạy chương trình một cách chính xác nhất. Có thể tiếp cận với dữ liệu khách hàng thông qua các trang mạng xã hội, tạo form xin ý kiến hoặc có thể đi xin ý kiến một cách trực tiếp tuy nhiên vẫn khó khả thi. Không

những thế, việc thu thập dữ liệu người dùng cũng là một điều hết sức tê nhị, có thể vi phạm các chính sách bảo mật thông tin hiện nay.

1.1.6 Các ứng dụng của học máy hiện nay

- Học máy hiện nay đã và đang được áp dụng vào nhiều lĩnh vực trong cuộc sống nhằm mục đích giúp cho con người cảm thấy tiện lợi và thoải mái hơn. Nhờ vậy mà nhiều vấn đề, bài toán gian nan đã được giải quyết 1 cách nhanh chóng mà không cần mất quá nhiều công sức hay thời gian, bao gồm các ứng dụng phổ biến như :

- Y tế : Xác định loại thuốc hay vaccine phù hợp cho từng người riêng biệt
- Thương mại : Xe tự hành của Google và Tesla rất có ích trong việc di chuyển những quãng đường dài hay các hệ thống gợi ý kết bạn của Facebook, gợi ý phim Netflix,...
- Ngân hàng : Xác định rủi ro khi cho vay, phân tích điểm tín dụng của mỗi người để đưa ra hạn mức cho vay hợp lý
- Giáo dục : Phân loại học sinh, sinh viên từ đó đưa ra những gói hỗ trợ hợp lý
- Kinh tế : Dự báo giá vàng hay giá dầu thế giới, xác định xu hướng thị trường chứng khoán, tiền ảo,...

- Hay hệ thống đánh cờ vây nổi tiếng AlphaGo do Google phát triển đã đánh bại cả người chơi cờ vây hàng đầu thế giới,... Các ứng dụng trên cho thấy rằng lĩnh vực AI hay ML đang là một đề tài nghiên cứu rất phổ biến và được ưa chuộng hiện nay, một chuyên đề nghiên cứu tiềm năng và có thể gặt hái được rất nhiều thành tựu trong tương lai gần.

1.2 Các hướng tiếp cận của bài toán phân loại khách hàng

1.2.1 Các hướng tiếp cận

- Bài toán phân loại khách hàng cần xác định ba vấn đề chính bao gồm :

- Dùng thông tin nào để phân loại ví dụ như ngày mua hàng, hoá đơn, tổng số tiền đã chi tiêu của khách hàng,...
- Sử dụng mô hình nào để phù hợp với dữ liệu đầu vào tùy vào mục đích khác nhau (chẳng hạn như phân khúc theo vị trí địa lý, nhân khẩu học,...)
- Phương pháp hay giải thuật nào dùng để huấn luyện cho máy có thể nhận biết nguồn thông tin/dữ liệu đó (như phân theo cụm, theo lớp,...)

- Nhiều phương pháp phân loại khách hàng đã được đề xuất trong khoảng vài năm trở lại đây, và có rất nhiều công trình nghiên cứu khoa học và bài báo đã được công bố rộng rãi trên toàn thế giới. Phân loại khách hàng là một vấn đề rất được quan tâm nhất là các doanh nghiệp do hiệu quả mà nó mang lại là rất lớn như giúp doanh nghiệp hiểu rõ khách hàng của mình hơn, đưa ra những chương trình khuyến mãi hợp lý,... Từ đó vừa giữ chân được khách hàng cũ đồng thời tìm kiếm được những khách hàng tiềm năng trong tương lai do chi phí để tìm khách hàng mới cao hơn nhiều so với việc duy trì khách hàng cũ.

- Cụ thể có 4 hướng tiếp cận chính cho bài toán phân loại khách hàng trong thực tế :

- Phương pháp phân loại theo nhân khẩu học : Đây có lẽ là cách đơn giản nhất để xác định các nhóm khách hàng nhưng nó vẫn rất hiệu quả ở thời điểm hiện tại. Nó sẽ xem xét các đặc điểm phi tính cách có thể nhận dạng được chẳng hạn như tuổi, giới tính, tôn giáo, dân tộc,... Chẳng hạn như phương pháp này sẽ nhắm đến các khách hàng tiềm năng dựa trên thu nhập của họ, do đó chi phí tiếp thị, quảng cáo sẽ không bị lãng phí khi hướng thông điệp đến những người không có khả năng mua sản phẩm.
- Phương pháp phân loại theo tâm lý : Phương pháp này sẽ tập trung vào tính cách và sở thích của khách hàng để phân loại. Nó cho phép việc tiếp thị cực kỳ dễ dàng và hiệu quả nhưng khách hàng vẫn cảm thấy thoải mái do nói chuyện với họ về cuộc sống xung quanh hằng ngày là chủ yếu.
- Phương pháp phân loại theo vị trí địa lý : Đây thường là một trong những cách dễ dàng nhất để xác định cũng như phân nhóm khách hàng liên quan đến vị trí thực tế của họ như khu vực, quốc gia,... Lựa chọn tuyệt vời cho các doanh nghiệp nếu muốn tiếp cận trực tiếp người dân địa phương để hiểu rõ họ hơn. Nhận thức được vị trí của khách hàng cho phép cân nhắc các loại hình quảng cáo để phù hợp với người dân ở đó tránh các hiểu lầm có thể xảy ra.
- Phương pháp phân loại theo hành vi : Hữu ích nhất đối với các doanh nghiệp thương mại điện tử. Thu thập dữ liệu mua sắm của khách hàng từ đó biết được các thói quen chi tiêu, mua sắm, tổng thời gian mua sắm, ... của họ. Từ đó có thể phân biệt được khách hàng truy cập lần đầu hay khách hàng đã truy cập nhiều lần nhưng chưa mua hàng. Giúp doanh nghiệp có thể điều chỉnh các ưu đãi để phù hợp với từng khách hàng.

- Hướng tiếp cận được xem là đơn giản và dễ dàng nhất là thông qua các công trình đã nghiên cứu của các cá nhân, tổ chức khác. Ngoài ra nhóm cũng sẽ thực hiện tìm hiểu các thuật toán khác tương tự để đánh giá độ tối ưu của thuật toán đã sử dụng. Nhóm sẽ thực hiện tìm hiểu dựa trên thuật toán K-Means thông qua ngôn ngữ lập trình Python.

1.2.2 Một số công trình nghiên cứu hiện nay

- Các công trình nghiên cứu bao gồm :

- Bài toán “Mall Customer Segmentation With K-Means Clustering” của tác giả *Aishwarya Wuntkal*. Tại bài viết, tác giả đã mô tả được cách thức sử dụng, áp dụng của lĩnh vực trí tuệ nhân tạo vào cuộc sống (phân loại Khách Hàng tại Mall - một trung tâm thương mại). Ngoài ra, với lượng dữ liệu to lớn được cung cấp chính xác (bởi nguồn thông tin đáng tin cậy) đã giúp cho bài toán của tác giả mô tả một cách trực quan nhất về cách khai thác dữ liệu khách hàng thông qua thuật toán K-Means. Tuy nhiên vẫn có mặt hạn chế, do bài viết thực hiện việc khai phá một cách đơn sơ, đơn giản nhất nên chưa thể mô tả đúng hết toàn bộ các mặt khác của dữ liệu.
- Bài toán “Customer Segmentation with RFM & K-Means” của tác giả *Data Queen* (là một tác giả trên nền tảng chuyên về lĩnh vực AI - Kaggle). Bài viết này đã được tác giả mô tả chính xác lượng dữ liệu cần khai phá, sử dụng, ngoài ra áp dụng được chiến lược RFM (Recency - Frequency - Monetary). RFM là một chiến lược khai phá dữ liệu khách hàng hay còn gọi là mô hình phân loại khách hàng. Mô hình này dựa trên 3 yếu tố để phân nhóm khách hàng, số lượng sẽ tùy thuộc vào bài toán mình tự định nghĩa.
- Bài toán “CVM Model of Customer Purchasing Behavior Based on Clustering Analysis” của tác giả *Rui Zhao* nói về thước đo giá trị của khách hàng. Trong bài viết mô tả qua các thuộc tính của cơ sở dữ liệu mà tác giả sử dụng đồng thời lựa chọn số cụm trong thuật toán K-Means thông qua phương pháp Silhouette.
- Bài toán “Data mining for the online retail industry: A case study of RFM model-based customer segmentation using data mining” của nhóm tác giả *Daqing Chen, Sai Laing Sain, Kun Guo*. Bài viết này đã mô tả cách mà tác giả biến đổi dữ liệu đầu vào theo mô hình RFM để có thể quan sát được những đặc trưng của dữ liệu. Qua đó tác

giả trình bày kết quả thông qua các biểu đồ cây hay biểu đồ 3D để người đọc có thể dễ dàng quan sát.

- Bài toán “Mall Customer Segmentation Using Clustering Algorithm” của tác giả *Asith Ishantha* đi sâu vào việc phân tích cơ sở dữ liệu đầu vào cũng như các thuật toán liên quan đến K-Means như Mini Batch K-Means. Đồng thời tác giả so sánh cũng như đánh giá hiệu quả của các thuật toán trên. Sau đó thống kê các kết quả dựa trên những kết quả đó.
- Bài toán “Customer Segmentation using K-means Clustering” của nhóm tác giả *Tushar Kansal, Suraj Bahuguna, Vishal Singh, Tanupriya Choudhury* tập trung chủ yếu vào việc phân loại khách hàng cũng như so sánh hiệu năng của 3 thuật toán chính là K-Means, Mean Shift và Agglomerative. Từ đó giúp người đọc có thể lựa chọn thuật toán phù hợp và tối ưu nhất với tập dữ liệu của mình.

CHƯƠNG 2 : CƠ SỞ LÝ THUYẾT

2.1 Tìm hiểu về ngôn ngữ lập trình Python

2.1.1 Python là gì ?

- Python là ngôn ngữ lập trình hướng đối tượng, cấp cao và mạnh mẽ, được tạo ra bởi *Guido van Rossum*, được xem là một trong các ngôn ngữ lập trình phổ biến hiện nay với mục đích lập trình đa năng.
- Nó hoàn toàn tạo kiểu động và sử dụng cơ chế cấp phát bộ nhớ tự động, do đó lập trình viên không cần khai báo kiểu dữ liệu một cách tường minh như ngôn ngữ Java hay C++.
- Đồng thời Python rất là dễ đọc, học và nhớ, cú pháp lệnh là điểm cộng vô cùng lớn vì sự rõ ràng, dễ hiểu và cách gõ linh động. Rất phù hợp cho những người mới học lập trình lần đầu còn nhiều khó khăn bỡ ngỡ.
- Python được cho là ngôn ngữ lập trình dễ học, được sử dụng rộng rãi trong trí tuệ nhân tạo bên cạnh ngôn ngữ R. Cấu trúc của Python còn cho phép người sử dụng viết mã lệnh với số lần gõ phím tối thiểu do bản thân Python đã cung cấp hầu như đầy đủ các thư viện cũng như các hàm dựng sẵn, người dùng chỉ cần gọi hàm và truyền tham số.

2.1.2 Lịch sử hình thành (Các phiên bản)

- Python được tạo ra bởi Guido van Rossum vào năm 1980 và lần đầu ra mắt công chúng vào tháng 2/1991. Cái tên “Python” cũng được tác giả đặt theo một chương trình hài vào cuối những năm 1970. Python bao gồm các phiên bản chính như :

- Phiên bản 1.0 được phát hành vào tháng 01/1994, đây được xem là bản phát hành chuẩn đầu tiên
- Phiên bản 1.6 là phiên bản thế hệ 1.x cuối cùng ra mắt vào ngày 05/09/2000
- Phiên bản 2.0 ra mắt vào ngày 16/10/2000 giới thiệu nhiều tính năng mới mẻ, bao gồm một bộ dọn rác theo chu kỳ, khả năng hỗ trợ Unicode cũng như List Comprehension.
- Phiên bản 2.7 phát hành vào 03/07/2010 là phiên bản 2.x cuối cùng được ra mắt
- Phiên bản 3.0 được ra mắt vào năm 03/12/2008. Đây là một phiên bản cập nhật lớn của Python không tương thích ngược hoàn toàn như người tiền nhiệm. Nhiều tính

năng lón của nó đã được chuyển mã ngược (backport) về loạt các phiên bản Python 2.6.x và 2.7.x. Các bản phát hành của Python 3 còn đi kèm với công cụ 2to3, có tác dụng tự động hóa việc dịch mã từ Python 2 sang Python 3.

- Phiên bản 3.8.8 và 3.9.2 được phát hành lần lượt vào năm 2019 và 2020. Cả 2 phiên bản này được xúc tiến phát triển vì tất cả các phiên bản trước của Python (bao gồm cả 2.7) gặp một số vấn đề bảo mật, có thể dẫn đến thực thi mã từ xa và “đầu độc” bộ nhớ đệm máy tính của người dùng.
- Phiên bản 3.10.4 được xem là phiên bản mới nhất hiện tại và được phát hành vào năm 2021. Đây là bản phát hành đặc biệt giúp sửa lỗi hồi quy được phát hiện bởi BPO 46968 khiến Python không còn được xây dựng trên Red Hat Enterprise Linux 6 nữa.

2.1.3 Các ứng dụng phổ biến

- Một số ứng dụng phổ biến của Python như :

- Ngôn ngữ phù hợp để dạy lập trình trong giáo dục cho những người mới bắt đầu
- Hỗ trợ cho lập trình ứng dụng web thông qua các framework như Django, Flask, Pyramid, Plone,...
- Còn hỗ trợ cho người dùng các framework như PyGame hay Panda3D để tạo ra các trò chơi 2D, 3D
- Trong lĩnh vực khoa học và tính toán, thường dùng để phân tích dữ liệu cũng như làm về khoa học và số liệu ứng dụng. Đặc biệt là các thư viện như SciPy và NumPy giúp ích rất nhiều trong việc phân tích các dữ liệu có độ phức tạp cao. Cũng như trong AI & ML, Python là một công cụ hoàn hảo để phát triển vì tính đơn giản, nhất quán, độc lập của nền tảng, bộ sưu tập thư viện tài nguyên và cộng đồng hỗ trợ mạnh mẽ và rộng lớn được xem là một ưu thế của Python so với các ngôn ngữ lập trình khác.
- Tuy không nhanh bằng các ngôn ngữ biên dịch như C++ hay Java do Python là ngôn ngữ thuộc nhóm thông dịch nhưng Python là ngôn ngữ tuyệt vời để tạo những bản mẫu mà không cần tốn quá nhiều công sức và thời gian. Từ đó có thể tiết kiệm được các khoản chi phí cần thiết.
- Python còn được sử dụng rộng rãi trong kỹ thuật Robot kết hợp với các nền tảng phần cứng như Raspberry Pi. Như các cánh tay Robot trong công nghiệp được lập trình bằng Python để có thể thực hiện các công việc phức tạp và tốn sức người.

- Cuối cùng nhưng không kém phần quan trọng là tự động hóa, nó giúp tự động hóa các hệ thống và quy trình lặp đi lặp lại trong các doanh nghiệp, công ty như việc chấm công, trả lời email của khách hàng,...

2.1.4 Các thư viện thông dụng được sử dụng trong học máy

- Trong các dự án AI / ML thì Python có các thư viện hoàn hảo để hỗ trợ cho lập trình viên giúp việc phát triển trở nên dễ dàng và thuận tiện hơn :

- Numpy : Thư viện này cung cấp khả năng vector hóa cách vận hành về toán trong mảng giúp cải thiện hiệu suất và tốc độ thực thi cũng như cung cấp các tính năng hữu ích về tính toán đại số trên mảng đa chiều. Đây là một trong những thư viện lõi của Machine Learning. Một số ưu điểm của Numpy có thể kể đến như : tốc độ truy xuất rất nhanh do dữ liệu được lưu trữ trên những vùng ô nhớ liền kề, các hàm đại số được tối ưu để có thể tính toán với tốc độ cao,...
- SciPy : Bao gồm các modules cho đại số tuyến tính, tối ưu, tích hợp và thống kê. Thư viện này mang đến rất nhiều hoạt động hữu ích liên quan đến số học như tích hợp số thông qua các submodules chuyên biệt.
- Pandas : Thư viện này được thiết kế để làm việc với dữ liệu đơn giản, trực quan như dữ liệu dạng bảng. Là công cụ hoàn hảo để tinh chỉnh và làm sạch dữ liệu trước khi được đưa vào các mô hình để xử lý. Nó cho phép thực hiện các phép biến đổi và thống kê trên các dữ liệu cơ bản với tốc độ rất nhanh. Pandas còn hỗ trợ đọc dữ liệu từ đa số các định dạng từ phổ biến đến hiếm gặp hiện nay như: csv, txt, xlsx, json, hdf5, dat,...
- Matplotlib : Là một trong những thư viện được phát triển đầu tiên trên Python về dựng đồ thị. Matplotlib hỗ trợ dựng đồ thị cho hầu hết các biểu đồ 2D, 3D phổ biến hiện nay. Các biểu đồ được vẽ trên matplotlib có thể được tùy biến và can thiệp sâu để điều chỉnh style, định dạng cũng như các thông số cần hiển thị trên biểu đồ. Do đã được phát triển lâu nên nó cũng có một số hạn chế nhất định như không có khả năng chia sẻ đồ thị thông qua API.
- Seaborn : Là phiên bản kế nhiệm của Matplotlib, thừa hưởng những ưu điểm của người đàn anh đồng thời có thêm những cải tiến khác như bổ sung thêm heat maps dùng để tổng hợp dữ liệu nhưng vẫn mô tả được toàn bộ mức độ phân tán của dữ liệu. Từ đó trực quan hóa dữ liệu giúp người dùng dễ theo dõi và quan sát hơn.

- Plotly : Thư viện đồ họa tương tác với giấy phép mã nguồn mở và dựa trên nền tảng trình duyệt. Để sử dụng Plotly sẽ cần phải đăng ký Key API riêng. Plotly có lợi thế lớn hơn so với các thư viện khác khi tạo ra các biểu đồ tương tác.
- SciKit Learn : Là thư viện mạnh mẽ nhất dành cho các thuật toán học máy được viết trên ngôn ngữ lập trình Python. Thư viện cung cấp một tập các công cụ xử lý các bài toán trong Machine Learning như Classification, Regression, Clustering,... Hỗ trợ mạnh mẽ trong việc xây dựng các sản phẩm giúp cho việc lập trình trở nên nhanh chóng và đơn giản hơn.
- TensorFlow : Thư viện mã nguồn mở phục vụ chủ yếu cho Machine Learning và được phát triển bởi Google. Hỗ trợ cho việc tính toán bằng cách tiếp cận mạnh mẽ các phép tính và bài toán phức tạp. Nó cho phép lập trình viên tính toán song song trên nhiều máy tính khác nhau, thậm chí trên nhiều CPU, GPU trong cùng 1 máy tính.
- Keras : Là thư viện mạng neural được viết bằng Python với các chức năng tương tự TensorFlow nhưng dễ sử dụng và nắm bắt hơn, phù hợp với những người mới bắt đầu. Nó có thể sử dụng đồng thời với các thư viện khác như CNTK, Theano,...

2.2 Tổng quan về thuật toán phân cụm dữ liệu

2.2.1 Giới thiệu

- Phân cụm dữ liệu là tách một tập dữ liệu đầu vào thành các cụm dữ liệu mà những đối tượng trong đó sẽ giống nhau ở một số tiêu chí nào đó so với những đối tượng nằm ở cụm khác. Đây là kỹ thuật trong data mining, các cụm thu được sẽ cung cấp các thông tin, tri thức để từ đó đưa ra quyết định chính xác. Phân cụm dữ liệu thuộc nhóm học không giám sát (Unsupervised Learning).
- Các đối tượng trong các cụm là thuần nhất và phân biệt với nhau đồng thời thỏa mãn các tiêu chí cơ bản như :
 - Tính liên kết chặt : Các đối tượng trong cùng một cụm có độ tương tự là cao nhất có thể
 - Tính tách rời : Các đối tượng trong các cụm khác nhau phải phân biệt nhau nhất có thể

- Phân cụm hỗ trợ nhiều kiểu dữ liệu đầu vào như dùng để xử lý hình ảnh, phân nhóm tài liệu, nhận dạng mẫu, hỗ trợ phân tích dữ liệu, giảm kích thước dữ liệu giúp tiết kiệm tài

nguyên lưu trữ,... Đặc điểm của phương pháp này là sẽ không biết trước số cụm mà phải dựa vào các phương pháp/thuật toán hỗ trợ khác để có thể xác định số cụm. Vì thế nên mỗi cách tiếp cận khác nhau sẽ cho ra kết quả khác nhau.

- Các bước phân cụm dữ liệu bao gồm :

- Thu thập dữ liệu
- Tính độ tương đồng của các đối tượng (Phổ biến nhất là độ đo Euclid chuẩn và độ đo Manhattan)
- Lựa chọn một thuật toán phân cụm phù hợp và thực hiện dựa trên dữ liệu đầu vào
- Đưa ra kết quả, đánh giá và nhận xét

- Một ví dụ của phân cụm trong thực tế là tiếp thị bán hàng trong siêu thị : siêu thị sẽ lấy các thông tin cơ bản của khách hàng như tổng số tiền đã chi tiêu, các mặt hàng đã mua,... Các khách hàng từ sẽ được nhóm thành 3 cụm cơ bản : cụm 1 là nhóm khách hàng có chi tiêu cao, cụm 2 có chi tiêu trung bình, cụm 3 là chi tiêu thấp. Sau đó, siêu thị sẽ gửi quảng cáo các sản phẩm cũng như đưa ra các chương trình khuyến mãi phù hợp với từng cụm khách hàng này.

- Một số thuật toán phân cụm thường thấy như :

- Phân hoạch (Partitioning) : K-Means, K-Modes, Fanny, CLARA, FCM,...
- Phân cấp (Hierarchical) : DIANA (Divisive Analysis), AGNES (Agglomerative Nesting), BIRCH, ROCK,...
- Dựa trên mật độ (Density Based) : DBSCAN, MeanShift , OPTICS, Spectral Method, Subtractive Method,...
- Dựa trên mô hình (Model Based) : EM, SOMs, COBWEB, CLASSIT,...
- Dựa trên lưới (Grid Based) : WaveCluster, OptiGrid, STING, CLIQUE,...

2.2.2 Ưu điểm và khuyết điểm so với các thuật toán phân lớp

- Ưu điểm :

- Đơn giản, ít tính toán, khá nhanh và tốn ít chi phí do dữ liệu đầu vào không cần gán nhãn

- Dữ liệu không cần gán nhãn trước cũng như không cần có sẵn các mô hình phân lớp (Model)
- Không nhất thiết phải xác định trước số cụm do mục đích chính của phân cụm là nhóm các đối tượng nhờ cấu trúc ẩn của dữ liệu trong khi phân lớp sẽ phân loại các phần tử mới vào các lớp đã xác định trước
- Có thể tiến hành triển khai nhanh chóng không tốn nhiều thời gian

- Khuyết điểm :

- Có thể xác định số lượng cụm bằng các phương pháp như Elbow hay Silhouette
- Số lượng cụm cũng ảnh hưởng đến kết quả đầu ra
- Giá trị tâm cụm ban đầu được khởi tạo ngẫu nhiên nên kết quả của mỗi lần chạy là khác nhau nên thiếu nhất quán
- Hiệu năng thường không ổn định tùy thuộc vào lượng dữ liệu đầu vào và tâm cụm
- Chỉ áp dụng với dữ liệu có thuộc tính số
- Dễ bị nhạy cảm với dữ liệu nhiễu và các phần tử ngoại lai nên dữ liệu cần được tiền xử lý

2.2.3 Các thuật toán phân cụm dữ liệu phổ biến hiện nay

- Mean Shift : Là thuật toán tìm kiếm theo chế độ, chuyển các điểm dữ liệu lặp đi lặp lại theo chế độ đã chọn. Giúp xác định vị trí cực đại của một hàm mật độ. thường được ứng dụng trong xử lý hình ảnh và thị giác máy tính.

• Ưu điểm :

- ❖ Không cần xác định số lượng cụm
- ❖ Tâm cụm hội tụ tới cụm có mật độ thành viên tối đa

• Khuyết điểm :

- ❖ Kích thước cụm có thể hơi lớn

- Density-Based Spatial Clustering of Applications with Noise (DBSCAN) : Là một thuật toán cơ sở để phân nhóm dựa trên mật độ. Nó có thể phát hiện ra các cụm có hình dạng và kích thước khác nhau từ một lượng lớn dữ liệu đầu vào chứa nhiễu. Bởi vì có thể xử lý dữ liệu chứa nhiễu nên thuật toán này thường được sử dụng trong thực tế,

• Ưu điểm :

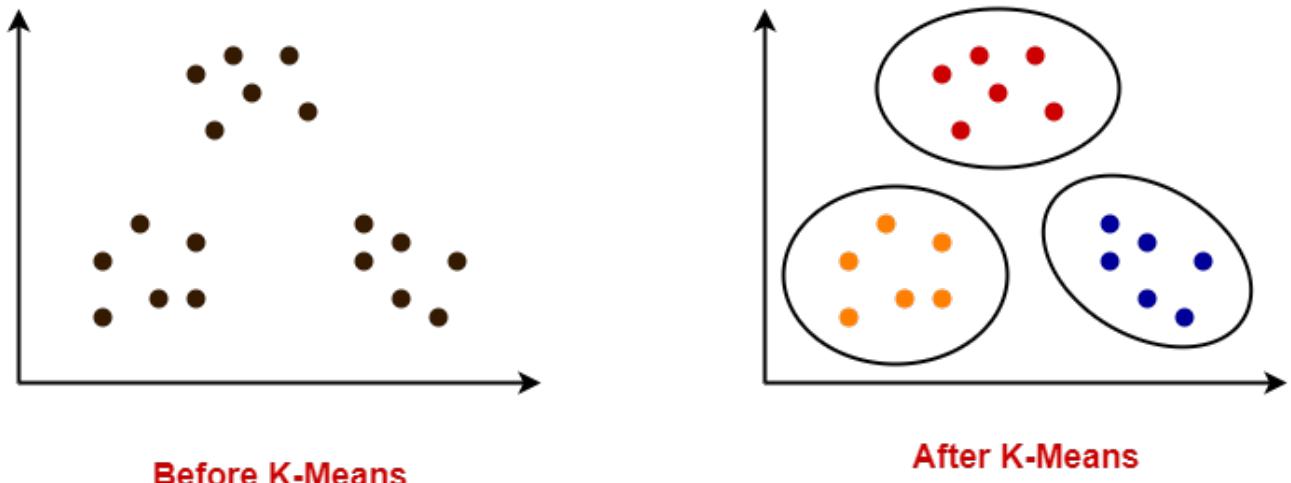
- ❖ Không cần xác định số lượng cụm

- ❖ Xác định các điểm nhiễu để không đưa vào cụm
- ❖ Kích thước cụm khá tốt
- Khuyết điểm :
 - ❖ Các tập dữ liệu có sự khác biệt lớn về mật độ thì DBSCAN sẽ không hoạt động tốt
 - ❖ DBSCAN cũng không hoạt động tốt với dữ liệu đa chiều
- Spectral Clustering : Hay còn gọi với cái tên khác là phân cụm dựa trên đồ thị, có chức năng là phân tích cụm nhằm tìm kiếm các nhóm đỉnh có liên quan trong một đồ thị. Nó cho ra kết quả là trong mỗi cụm các đỉnh có nhiều cạnh kết nối gần với nhau nhất trong khi giữa các cụm thì chỉ có vài cạnh nối. Thường được ứng dụng trong nhiều lĩnh vực như phân đoạn hình ảnh, phân cụm quang phổ của chuỗi protein, tách giọng nói,...
- Ưu điểm :
 - ❖ Các cụm không cần tuân theo một hình dạng hay khuôn mẫu nhất định
 - ❖ Hiệu quả đối với các dữ liệu có hình dạng và kích thước khác nhau
- Khuyết điểm :
 - ❖ Thời gian tính toán tùy thuộc vào tập dữ liệu
 - ❖ Các giá trị riêng và vector cần được tính toán trước khi phân cụm

2.2.4 Thuật toán K-Means

- Là một thuật toán do Stuart Lloyd giới thiệu lần đầu tiên vào năm 1957 ở Bell Labs như một kỹ thuật điều chỉnh, thuộc lớp giải thuật học không giám sát (Unsupervised learning). Là phương pháp phân cụm phổ biến nhất trong các phương pháp thuộc nhóm phân hoạch (Partitioning). Được sử dụng để phân loại hoặc phân nhóm một nhóm đối tượng/ dữ liệu nào đó dựa trên đặc tính/thuộc tính chung giữa các đối tượng với nhau. Việc quyết định phân một đối tượng vào một cụm phụ thuộc vào độ tương đồng của đối tượng đó với trọng tâm của các cụm.
- Về lý thuyết sẽ có n đối tượng và mỗi đối tượng có m thuộc tính thì ta sẽ chia các đối tượng đó thành k nhóm (dựa trên các thuộc tính chung) bằng chính thuật toán K-Means này.
- Ý tưởng chung của K-means là xác định vị trí tâm cụm và xếp đối tượng vào cụm mà có tâm cụm gần nó nhất dựa trên khoảng cách Euclidean hay khoảng cách Manhattan. Khoảng

cách giữa các cụm khác nhau cần đạt giá trị lớn nhất và khoảng cách bên trong một cụm phải đạt giá trị nhỏ nhất.



Hình 2.1. Dữ liệu sau khi đã áp dụng thuật toán K-Means

- Các bước của thuật toán K-means :

Bước 1 : Xác định K – số cụm (ngẫu nhiên hoặc nhờ sự giúp đỡ của các thuật toán khác)

Bước 2 : Chọn K điểm ngẫu nhiên làm tâm cụm

Bước 3 : Gán từng điểm vào cụm có tâm gần nó nhất dựa theo khoảng cách hình học Euclidean, Manhattan hay Cosine

Bước 4 : Tính phương sai và đặt lại tâm cụm mới (tâm cụm được tính bằng trung bình cộng toạ độ các thành phần trong cụm)

Bước 5 : Nếu tâm cụm không đổi hoặc không còn việc gán lại điểm nào vào các cụm khác (điều kiện hội tụ được thỏa mãn) thì kết thúc thuật toán. Nếu không thì lặp lại bước 3

- Có trường hợp thuật toán không dừng được thì điều kiện dừng có thể thay từ tâm cụm không đổi sang thành tâm cụm chênh lệch 1 khoảng dưới Delta nào đó với tâm cụm cũ.

- Xác định K : Việc xác định K có thể tách ra thành 1 bài toán riêng biệt. Trong trường hợp hiểu rõ tập dữ liệu, việc xác định K sẽ rất dễ dàng. Nhưng trong trường hợp khác thì không, khi đó ta có thể sử dụng phương pháp thử và sai (try and error) hoặc dùng một số phương pháp hỗ trợ việc này như Elbow hay Silhouette.

- Thuật toán K-Means được chứng minh là hội tụ sau một khoảng thời gian nhất định và có độ phức tạp tính toán là :

$$O((3nkd)7T^{\text{flop}}))$$

- Trong đó :

n là số đối tượng dữ liệu

k là số cụm dữ liệu

d là số chiều của bài toán

7 là số vòng lặp

T^{flop} là thời gian thực hiện 1 phép tính cơ sở (nhân, chia, cộng,...)

A. Phân tích toán học :

- Giả sử ta có N điểm dữ liệu đầu vào và được biểu diễn theo từng đối tượng x như sau:

$$X = [x_1, x_2, x_3, x_4, \dots, x_N] \in R^{d \times N} \text{ và } K < N$$

- Trong đó :

d là số chiều của ma trận

N là tổng điểm dữ liệu đầu vào

K là số lượng phần tử trung tâm

R là ma trận theo d x N

- Khi đó ta cần tìm các phần tử trung tâm $m_1, m_2, m_3, \dots, m_K \in R^{d \times 1}$.

- Với mỗi điểm dữ liệu x_i thì ta đặt $y_i = [y_{i1}, y_{i2}, y_{i3}, \dots, y_{iK}]$ là các label vector của nó. Khi x_i được phân vào cluster k thì khi đó $y_{ik} = 1$ và $y_{ij} = 0 \forall j \neq k$. Nghĩa là chỉ duy nhất một phần tử của vector y_i là bằng 1 (chính là vị trí cluster x_i) và các phần tử còn lại bằng 0. Việc biểu diễn mã hóa label dữ liệu như thế này được gọi là one-hot và rất phổ biến trong Machine Learning.

- Ví dụ $y_i = [1, 0, 0, 0]$ nghĩa là y_i thuộc vào cluster 1; $y_i = [0, 0, 1, 0]$ là y_i thuộc vào cluster

➤ Có thể biểu diễn ràng buộc giữa x và y như sau :

$$y_{ik} \in \{0,1\}, \sum_{k=1}^K y_{ik} = 1 \quad (1)$$

- Giả sử m_k chính là phần tử trung tâm của mỗi cluster và ước lượng, dự đoán số lượng điểm sẽ được phân vào cluster này (có m_k là phần tử trung tâm). Một điểm x_i khi được phân vào cluster sẽ có sai số là $(x_i - m_k)$. Vì vậy ta cần phải tìm cách để đại lượng sai số sau đây đạt được giá trị nhỏ nhất có thể.

$$\|x_i - m_k\|_2^2 \quad (2)$$

- Nói dễ hiểu hơn đại lượng (2) chính là tổng bình phương của các phần tử bên trong vector $(x_i - m_k)$.
- Vì y_i phụ thuộc vào x_i nên ta sẽ có biểu thức như sau với sự ràng buộc tương tự (1)

$$y_{ik} \|x_i - m_k\|_2^2 = \sum_{j=1}^K y_{ik} \|x_i - m_k\|_2^2$$

- Khi đó sai số cho toàn bộ dữ liệu là :

$$\tau(Y, M) = \sum_{i=1}^N \sum_{j=1}^K y_{ij} \|x_i - m_k\|_2^2$$

Trong đó $Y = [y_1, y_2, \dots, y_N]$, $M = [m_1, m_2, \dots, m_K]$ lần lượt là các ma trận được tạo bởi label vector của mỗi điểm dữ liệu đang được xét và điểm trung tâm của mỗi cluster đó. Thì khi đó hàm số mất mát chính là $\tau(Y, M)$ với ràng buộc dữ liệu được nêu ở (1). Có nghĩa là ta phải tối ưu làm sao cho hàm số mất mát đạt giá trị nhỏ nhất có thể. Khi đó bài toán của mình sẽ đạt được tốc độ cũng như thời gian xử lý,... tốt nhất.

- Khoảng cách Euclidean : là khoảng cách giữa 2 điểm trong hệ trực tọa độ và được tính theo công thức sau (công thức tính theo hệ trực tọa độ Oxy)

$$\partial_{ji} = \sqrt{\sum_{s=1}^m (x_{is} - x_{js})^2}$$

- Với ∂_{ji} là khoảng cách từ ai tới c_j

x_{is} là thuộc tính thứ s của đối tượng ai

x_{js} là thuộc tính thứ s của phần tử trung tâm c_j

- Phần tử trung tâm: k phần tử trung tâm (k nhóm) ban đầu được chọn ngẫu nhiên và kết quả sẽ thay đổi xấp xỉ dựa vào việc chọn phần tử trung tâm và có công thức như sau :

$$c_{ji} = \frac{\sum_{s=1}^t x_{sj}}{t}$$

- Trong đó :

c_{ji} là tọa độ thứ j của phần tử trung tâm nhóm i

x_{sj} là thuộc tính thứ j của phần tử s ($s=1,\dots,t$)

t là số phần tử hiện có của nhóm thứ i

$i = 1,\dots,k$ là k số cluster

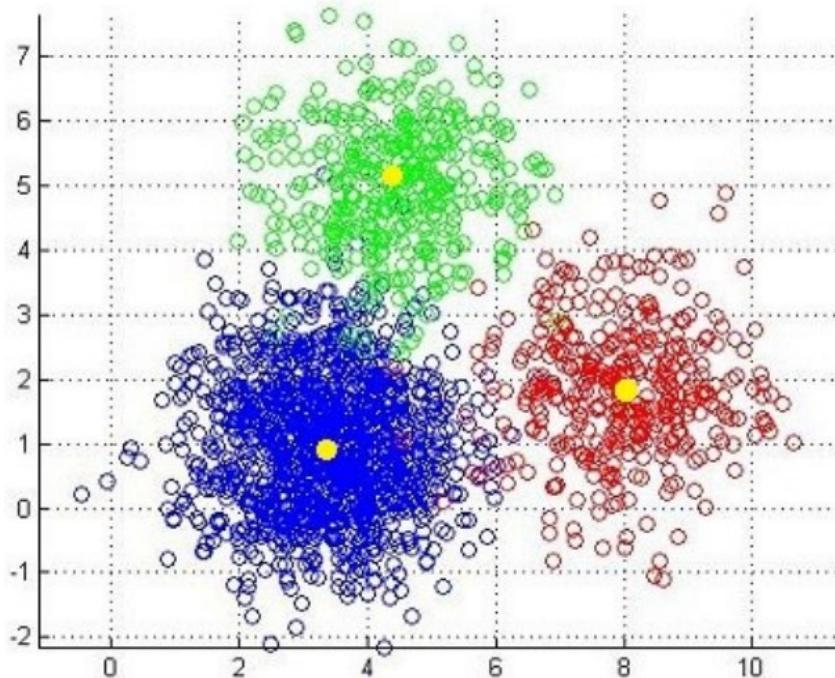
$j = 1,\dots,m$ là m số thuộc tính

- Độ chính xác của thuật toán : Hàm mất mát của thuật toán K-Means biểu hiện cho việc độ chính xác của nó càng lớn khi khoảng cách mỗi điểm so với trung tâm cụm càng lớn.

$$J = \frac{1}{m} \sum_{i=1}^m \partial(c_i, x_i)$$

B. Ví dụ cụ thể về việc phân cụm theo K-Means :

- Gọi điểm trung bình chính là điểm/phần tử trung tâm và từ phần tử đó tiến hành tính toán khoảng cách đến mỗi phần tử khác bên trong tập dữ liệu ban đầu. Như vậy nếu chọn $K=3$ (nghĩa là có 3 phần tử trung tâm) thì toàn bộ dữ liệu sẽ được phân thành từng cụm như sau và trong đó các điểm màu vàng chính là các phần tử trung tâm



Hình 2.2. Mô hình về K-Means và chứa 3 phần tử trung tâm

- Đầu tiên để thực hiện phân cụm thì ta phải chọn K điểm trung tâm (để phân chia thành K cụm) và được chọn ngẫu nhiên từ m dữ liệu đầu vào. Sau đó với điểm dữ liệu x_i ta sẽ gán nó cho cụm c_i (chính là cụm có trung tâm gần nó nhất) - tính theo công thức Euclidean. Gọi C_i chính là các điểm được chọn làm trung tâm và x_i là các điểm dữ liệu ta có công thức sau:

$$\partial(c_1, x) = \sqrt{\sum_{i=1}^m (x_i - c_1)^2}$$

là khoảng cách từ trung tâm C_1 tới từng điểm x

$$\partial(c_2, x) = \sqrt{\sum_{i=1}^m (x_i - c_2)^2}$$

là khoảng cách từ trung tâm C_2 tới từng điểm x

$$\partial(c_3, x) = \sqrt{\sum_{i=1}^m (x_i - c_3)^2}$$

là khoảng cách từ trung tâm C_3 tới từng điểm x

- Sau đó xét từng ∂ và chọn ra một khoảng cách gần nhất \rightarrow gán điểm x_i được xét vào cụm c_j .
- Sau khi đã xét được hết toàn bộ các điểm rồi thì tiến hành bước tiếp theo chính là xác định lại vị trí phần tử trung tâm bằng cách tính trung bình tọa độ các điểm thuộc chính cụm đó và được lặp đi lặp lại liên tục đến khi vị trí phần tử trung tâm cụm không thay đổi sau một số lần lặp nào đó (sự hội tụ).

$$\phi_k = \frac{1}{n} (x_1 + x_2 + x_3 + \dots + x_n).$$

- Trong đó :

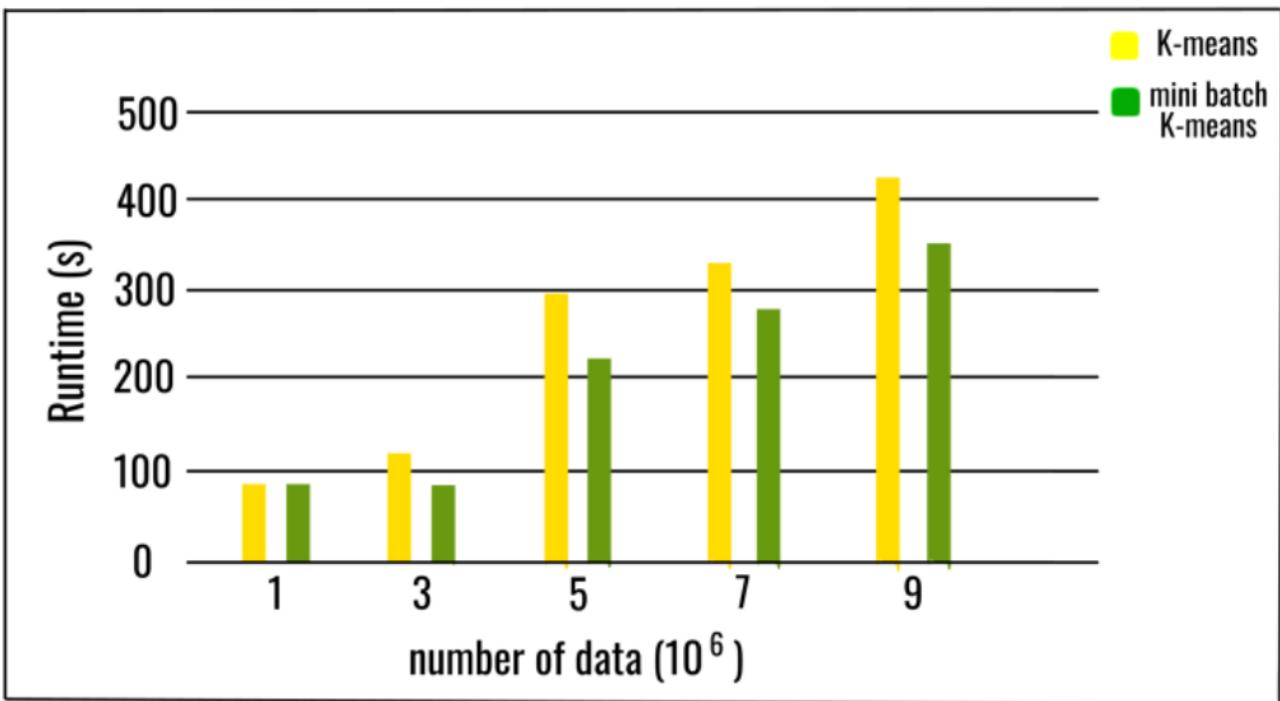
$x_{1, 2, \dots, n}$ chính là chỉ số dữ liệu thuộc cụm thứ k

ϕ_k chính là tọa độ của phần tử trung tâm k sau khi được cập nhật

2.2.5 Thuật toán Mini Batch K-Means

- Thuật toán K-Means là một thuật toán phân cụm phổ biến nhất và được biết tới do hiệu suất tốt và thời gian phân cụm nhanh của nó. Tuy nhiên khi làm việc với lượng dữ liệu có kích thước lớn và mật độ dày đặc thì việc này trở nên bất khả thi, có thể tốn rất nhiều thời gian để chạy. Việc huấn luyện một lần trên toàn bộ dữ liệu là điều không thể vì làm như vậy sẽ gây lãng phí về chi phí lưu trữ, chi phí tính toán. Vì thế nên đã xuất hiện nhiều phương pháp, thuật toán khác được đề xuất sử dụng trước khi tiến hành thực nghiệm trên K-Means để đảm bảo được chất lượng dữ liệu đầu ra cũng như thời gian chạy tối ưu hết mức có thể. Chính vì thế thuật toán Mini Batch K-Means được sinh ra để hỗ trợ, giảm chi phí thời gian và không gian của thuật toán K-Means bằng cách tiếp cận bằng các luồng nhỏ hơn (lượng dữ liệu ít hơn).

- Ý tưởng chính của thuật toán Mini Batch K-Means chính là sử dụng các luồng dữ liệu nhỏ ngẫu nhiên và có kích thước cố định. Khi chạy thuật toán này, sau mỗi lần lặp thì một mẫu ngẫu nhiên mới từ tập dữ liệu ban đầu (dữ liệu thu thập) được sử dụng để cập nhật lại các cụm và chạy liên tục cho đến khi hội tụ.



Hình 2.3. So sánh thuật toán K-Means và Mini Batch K-Means

- Thuật toán này lấy các luồng dữ liệu nhỏ hơn và được chọn ngẫu nhiên từ tập dữ liệu ban đầu để đưa vào thuật toán sau mỗi lần lặp. Mỗi dữ liệu sẽ được gán cho các cụm (cụm gần nhất) và phụ thuộc vào vị trí trước đó của phần tử trung tâm cụm đang xét đến. Tương tự K-Means thì sau mỗi lần lặp thì các phần tử trung tâm cũng sẽ được cập nhật lại cho đến khi hội tụ (không có sự thay đổi vị trí của phần tử trung tâm).
- Thuật toán Mini Batch K-Means có thể tiết kiệm đáng kể thời gian tính toán, giảm được các chi phí đáng kể để tối ưu hóa thuật toán K-Means ban đầu. Chính vì thế nó có thể khắc phục được nhược điểm của K-Means khi chạy trên một tập dữ liệu đầu vào lớn.
- Phiên bản được xem như là cải tiến mới nhất của Mini Batch K-Means là Online Learning K-Means (sẽ được nhóm thực nghiệm bên dưới). Trong đó sẽ cập nhật K tâm cụm đối với mỗi đối tượng dữ liệu đầu vào. Ta có công thức sau :

$$Q(w) = \sum_{i=1}^M ||x_i - w(x_i)||_2^2$$

- Trong đó : $w(x_i)$ là tâm gần nhất với x_i

- Tại bước t lấy một điểm x_t , sau đó tìm tâm w_t gần nhất với x_t và cập nhật lại tâm w_t theo công thức sau :

$$w_{t+1} = w_t + \gamma_t (x_t - w_t)$$

CHƯƠNG 3 : XÂY DỰNG THUẬT GIẢI

3.1 Tối ưu hóa việc lựa chọn số cụm dữ liệu

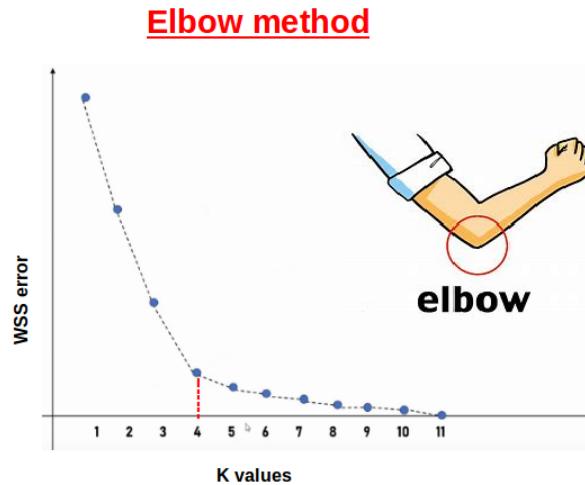
3.1.1 Lý do phải tối ưu hóa việc lựa chọn số cụm

- Trong bài toán phân loại khách hàng, do nhóm sử dụng thuật toán K-means để phân cụm khách hàng thành nhiều cụm khác nhau nên đòi hỏi phải có một sự lựa chọn các điểm trung tâm (phần tử trung tâm) một cách tốt nhất và để tối ưu hóa việc lựa chọn số cụm dữ liệu thì cách tốt nhất đó chính là xác định số k cụm trước khi xây dựng thuật toán. Lý do là vì phân cụm K-Means không chỉ cần k ban đầu để thực hiện thuật toán, mà còn số k để kiểm soát quá trình phân cụm, phân cụm sao cho tối ưu bởi vì số cụm ảnh hưởng rất lớn kết quả của bài toán.
- Chính vì mỗi khách hàng đại diện cho một lượng dữ liệu lớn (các dữ liệu bên trong đối tượng khách hàng). Gọi N chính là số lượng khách hàng trong tập dữ liệu nhóm thu thập được, ứng với mỗi khách hàng sẽ được đại diện bởi một vector đặc trưng và có số chiều là k. Như vậy sẽ có được N vector đặc trưng sử dụng để làm tập dữ liệu mẫu huấn luyện. Tuy nhiên phải thực hiện các bước chọn điểm trung tâm để tối ưu hóa thuật toán và một số các giải pháp mà nhóm tìm được để có thể khắc phục được khuyết điểm của thuật toán K-Means như thuật toán Elbow hay thuật toán Silhouette.

3.1.2 Thuật toán Elbow

- Thuật toán Elbow là một phương pháp heuristic dùng để xác định số k cụm cho thuật toán phân cụm K-Means phổ biến nhất hiện nay. Phương pháp này giúp chọn được số lượng các cụm phù hợp dựa vào đồ thị trực quan hóa bằng cách nhìn vào sự suy giảm của hàm biến dạng và lựa chọn ra điểm khuỷu tay (elbow point).
- Cách hoạt động :
 1. Giả sử ban đầu , chúng ta áp dụng phân cụm K-Means trên dataset cho số k cụm (k chạy từ 2 đến 15) để xác định các nhóm không có nhãn trong dataset.
 2. Sau đó, điểm trung bình đã được tính theo tham số 'metric'. Giá trị mặc định cho tham số số liệu là 'biến dạng', tính tổng các khoảng cách bình phương từ mỗi điểm đến các trọng tâm của nó.

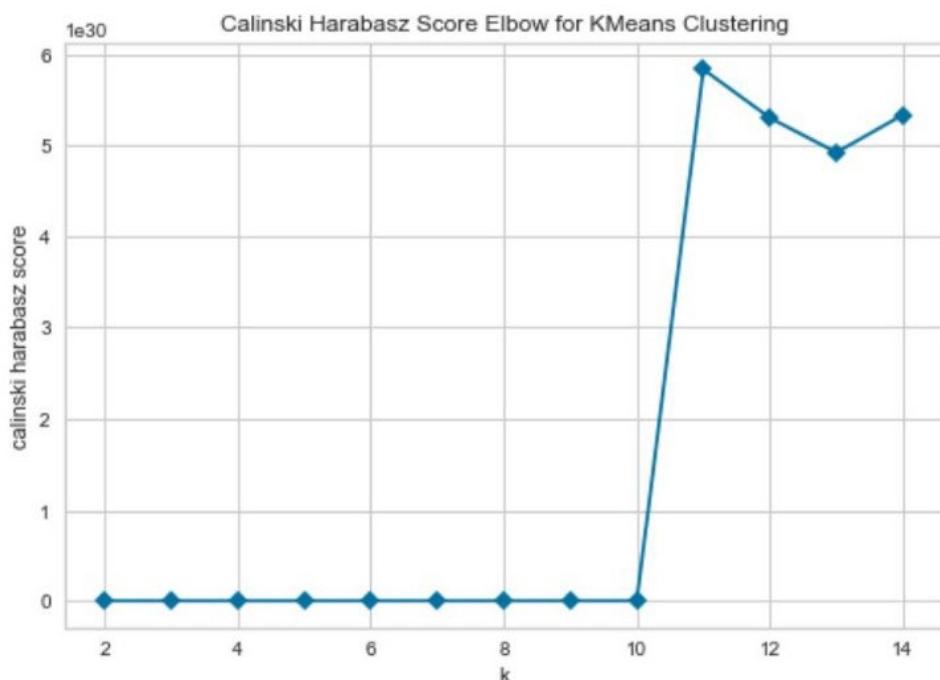
3. Khi đồ thị được biểu thị cho giá trị 'biến dạng', ta sẽ có cấu trúc dạng hình cánh tay, phần khuỷu tay và số lượng cụm như thể hiện trong Hình 3.1 bên dưới.



Hình 3.1. Phương pháp khuỷu tay với tham số số liệu 'biến dạng'

4. Sau đó sẽ dùng một giá trị khác của tham số số liệu gọi là 'calinski harabasz'. Tham số này tính toán tỷ lệ tán xạ giữa và trong các cụm để đưa ra các giả thuyết tốt hơn. Ở đây cấu trúc khuỷu tay sẽ có hình dạng đảo ngược.

5. Cuối cùng, điểm số trung bình sẽ được hiển thị trên biểu đồ (k so với điểm calinski harabasz trung bình) được hiển thị trong Hình 3.2.



Hình 3.2. Biểu đồ khuỷu tay sử dụng tham số chỉ số 'Calinski Harabasz'

3.1.3 Thuật toán Silhouette

- Thuật toán Silhouette đo lường mức độ gần của một điểm với các điểm lân cận gần nhất của nó, trên tất cả các cụm. Nó cung cấp thông tin về chất lượng phân cụm và có thể được sử dụng để xác định xem liệu có nên thực hiện tinh chỉnh thêm bằng cách phân nhóm dựa trên phân nhóm hiện tại hay không. Mục đích của nó là để đo lường mức độ tối ưu khi một quan sát, một điểm dữ liệu được phân vào một cụm bất kỳ.
- Giá trị Silhouette là thước đo mức độ tương tự của một điểm dữ liệu đối với cụm của chính nó (gắn kết) so với các cụm khác (tách biệt). Giá trị của Silhouette nằm trong khoảng [1, -1], trong đó giá trị cao chỉ ra rằng điểm dữ liệu được đối sánh tốt với cụm của chính nó và đối sánh kém với các cụm lân cận. Nếu hầu hết các điểm dữ liệu có giá trị cao, thì việc phân cụm là hợp lý. Nếu nhiều điểm dữ liệu có giá trị thấp hoặc âm, thì việc phân cụm có thể đang có quá nhiều hoặc quá ít cụm. Từ đó cho biết được các điểm dữ liệu nằm gọn bên trong cụm (tốt) hay nằm gần ngoài rìa cụm (không tốt) để đánh giá hiệu quả của việc phân cụm.

Tính toán hệ số Silhouette:

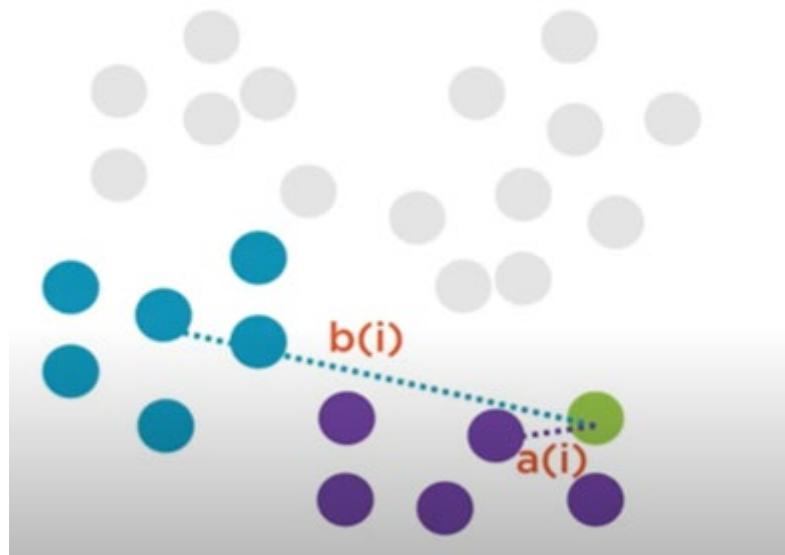
a (i) : Khoảng cách trung bình của điểm đó với tất cả các điểm khác trong cùng một cụm.

b (i) : Khoảng cách trung bình của điểm đó với tất cả các điểm trong cụm gần nhất với cụm của nó.

s (i) : Hệ số silhouette hoặc điểm thứ i sử dụng công thức được đề cập bên dưới đây.

$$s(i) = \frac{b(i) - a(i)}{\max(b(i), a(i))}$$

Sau khi tính toán hệ số Silhouette cho mỗi điểm thì sẽ lấy trung bình để tính điểm Silhouette.



Hình 3.3. Cách tính khoảng cách trong Silhouette

Silhouette Analysis:

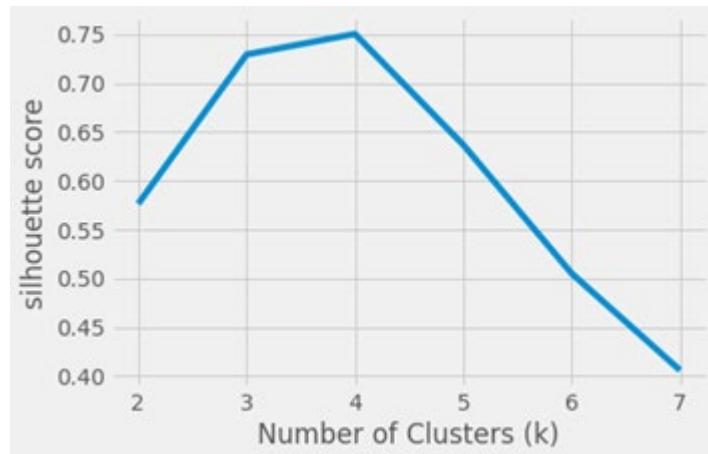
- Silhouette là một thước đo về cách một thuật toán phân cụm đang hoạt động. Sau khi tính toán hệ số Silhouette của mỗi điểm trong tập dữ liệu đầu vào, hãy vẽ biểu đồ của thuật toán đó để có được một cái nhìn trực quan về mức độ hiệu quả của tập dữ liệu được nhóm thành k cụm. Biểu đồ Silhouette hiển thị thước đo mức độ gần cho mỗi điểm trong một cụm với các điểm trong các cụm lân cận và do đó cung cấp một cách để đánh giá các thông số như số lượng cụm một cách trực quan. Số đo này có miền giá trị trong khoảng từ [-1, 1].

Điểm quan trọng:

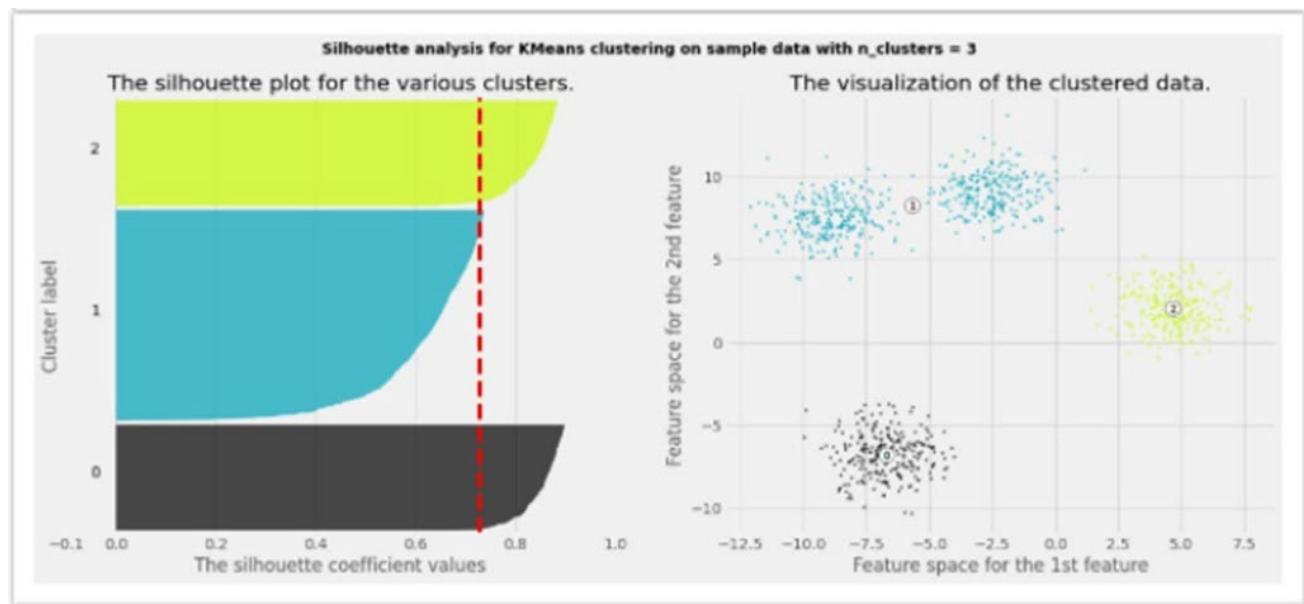
- Hệ số Silhouette + 1 chỉ ra rằng mẫu ở xa các cụm lân cận.
- Hệ số Silhouette bằng 0 chỉ ra rằng mẫu nằm trên hoặc rất gần ranh giới quyết định giữa hai cụm lân cận.
- Hệ số Silhouette < 0 cho thấy rằng những mẫu đó có thể đã được chỉ định vào nhóm sai hoặc là các mẫu ngoại lai.

Tìm giá trị tối ưu của ‘k’ bằng Silhouette Analysis :

- Tương tự như phương pháp Elbow trước đó, chọn một phạm vi giá trị của k (số cụm), sau đó huấn luyện phân cụm K-Means cho mỗi giá trị của k. Đối với mỗi mô hình phân cụm, K-Means được đại diện bằng các hệ số Silhouette trên một biểu đồ. Từ đó quan sát được các biến động và ngoại lệ của mỗi cụm.

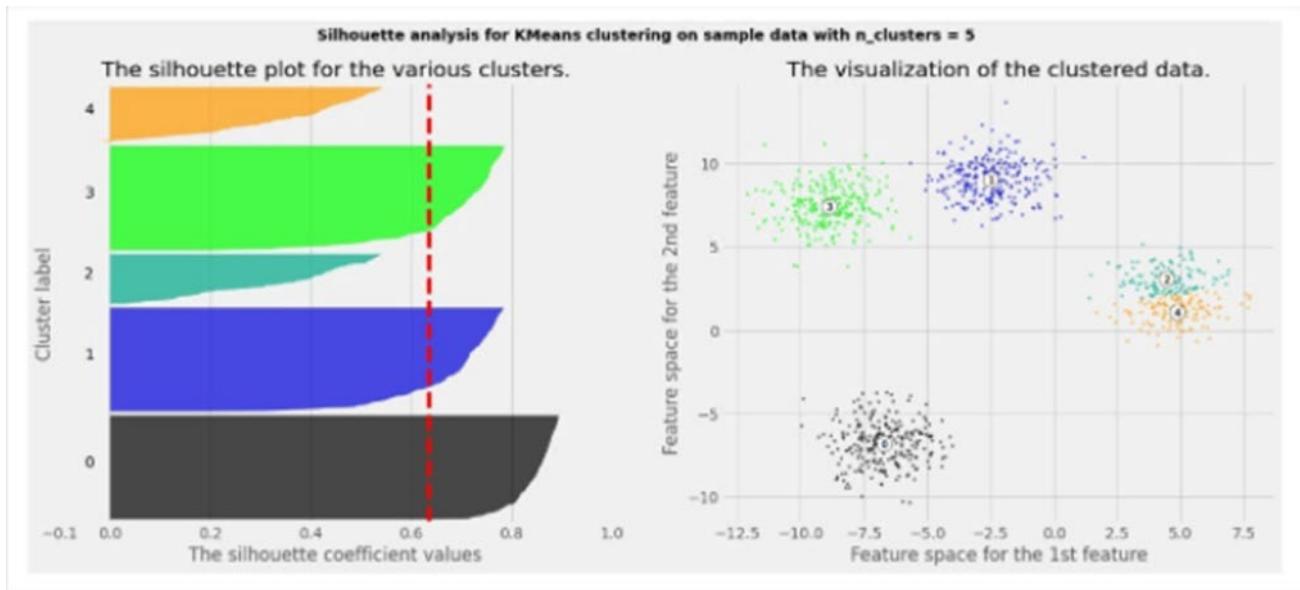


Hình 3.4. Điểm Silhouette trung bình tương ứng với số cụm



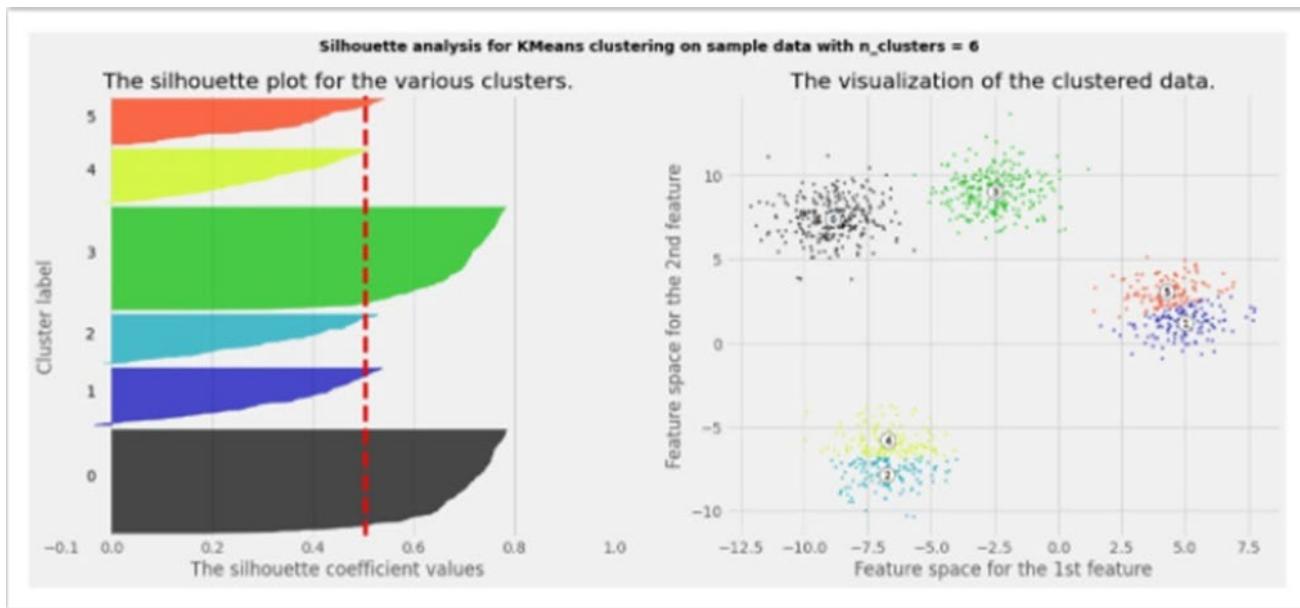
Hình 3.5. Điểm Silhouette tương ứng với số cụm là 3

- Biểu đồ Silhouette cho thấy rằng giá trị n_cluster bằng 3 là một lựa chọn không tốt, vì tất cả các điểm trong cụm với cluster_label = 0 đều có điểm Silhouette dưới mức trung bình.



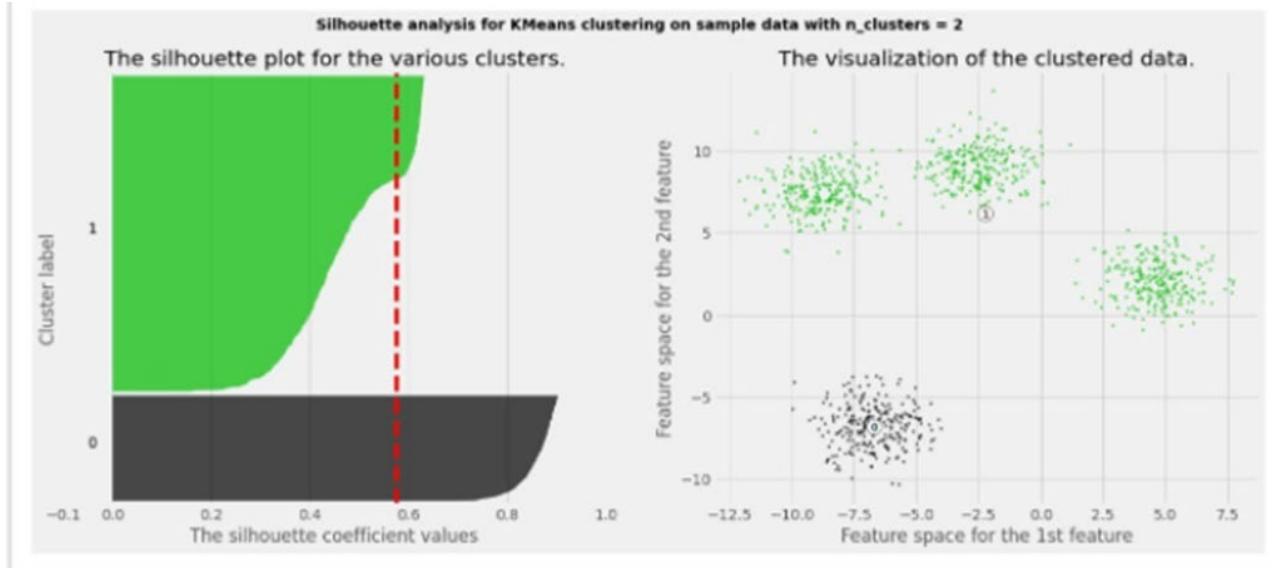
Hình 3.6. Điểm Silhouette tương ứng với số cụm là 5

- Biểu đồ Silhouette cho thấy rằng giá trị n_{cluster} bằng 5 là một lựa chọn không tốt, vì tất cả các điểm trong cụm có $\text{cluster_label} = 2$ và 4 đều có điểm Silhouette dưới mức trung bình.



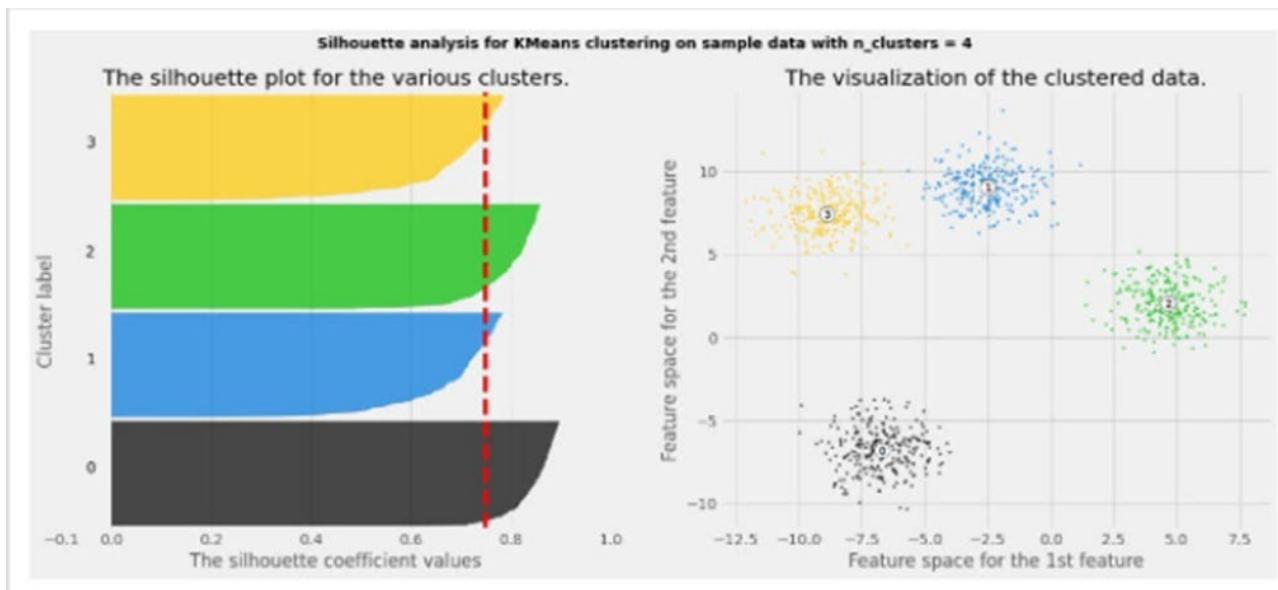
Hình 3.7. Điểm Silhouette tương ứng với số cụm là 6

- Biểu đồ Silhouette cho thấy rằng giá trị n_cluster bằng 6 lại là một lựa chọn không tốt, vì tất cả các điểm trong cụm với cluster_label = 1, 2, 4 và 5 đều có điểm Silhouette dưới mức trung bình mặc dù có sự hiện diện của một vài điểm ngoại lệ.



Hình 3.8. Điểm Silhouette tương ứng với số cụm là 2

- Độ dày của biểu đồ Silhouette cho cụm với cluster_label = 1 khi n_clusters = 2, có kích thước lớn hơn do nhóm 3 cụm con thành một cụm lớn hơn.



Hình 3.9. Điểm Silhouette tương ứng với số cụm là 4

- Đối với $n_clusters = 4$ thì tất cả các cụm có độ dày tương tự hoặc ít hơn không đáng kể, do đó nên có kích thước tương tự, có thể được coi là ‘k’ cụm tốt nhất.

3.2 Chuẩn hóa dữ liệu

3.2.1 Định nghĩa dữ liệu nhiễu

- Dữ liệu nhiễu có thể được xem là một dạng dữ liệu ngoại lai, có giá trị khác biệt so với những giá trị khác có trong tập dữ liệu. Sự khác biệt này có thể dựa trên nhiều tiêu chí khác nhau như giá trị hay thuộc tính. Chính vì những giá trị nhiễu đó có thể làm cho các thuật toán tính toán có thể xảy ra sai sót (các thuật toán học máy sẽ cho ra kết quả sai, không dự đoán được chính xác). Vì thế nên cần phải lọc các dữ liệu nhiễu, làm mịn dữ liệu để các giá trị có thể chuyển về dạng gần tương tự nhau, giúp các thuật toán nhạy cảm với dữ liệu nhiễu không xảy ra sai sót. Từ đó đưa ra các quyết định chính xác, tối ưu hơn. Không những thế, việc làm sạch giá trị nhiễu đó còn có thể giúp cho thuật toán có thời gian xử lý nhanh hơn, thời gian chạy ngắn hơn nhưng vẫn đạt được kết quả như mong muốn.

- Dữ liệu nhiễu được chia thành 2 loại cơ bản bao gồm :

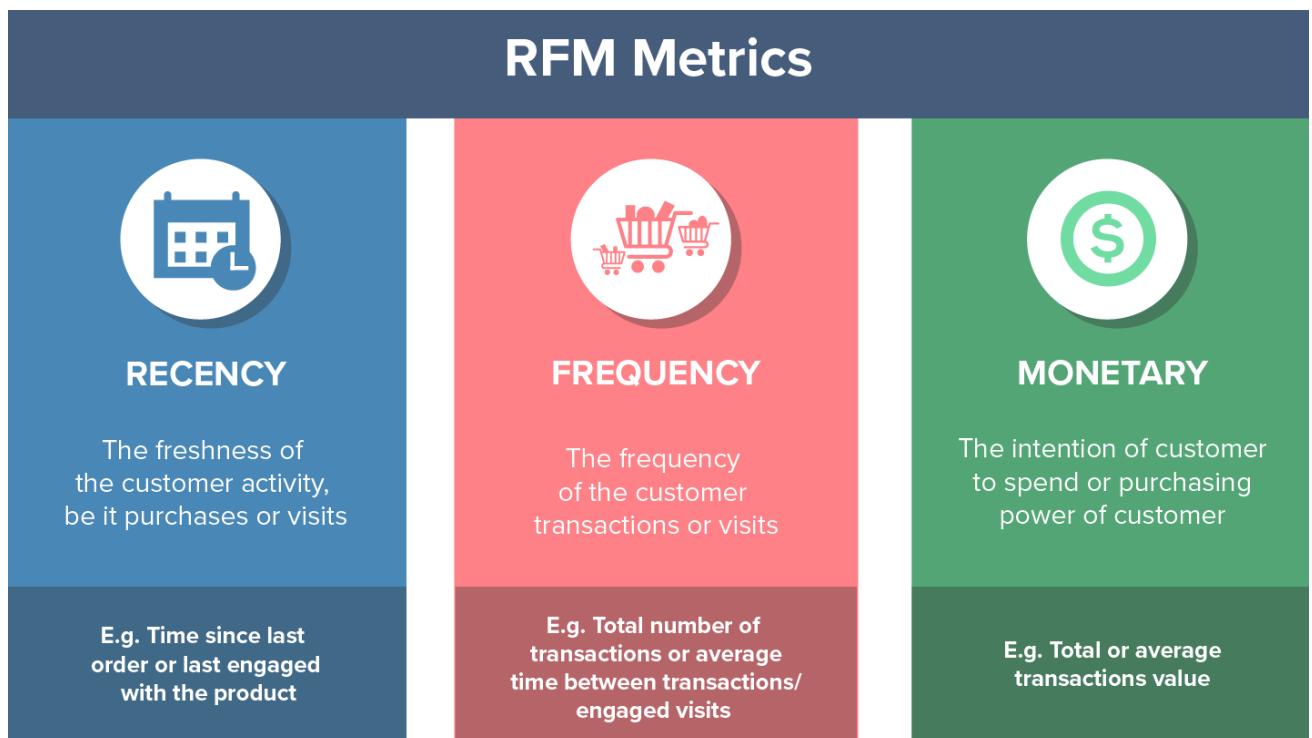
- Global Outlier : Là kiểu dữ liệu khác hoàn toàn so với dữ liệu bên trong dataset (Ví dụ việc dữ liệu tuổi của người lớn hơn 200).
- Contextual Outlier : Là kiểu dữ liệu khác so với 1 phần dataset nhưng vẫn trong khoảng bình thường so với 1 phần khác (Ví dụ như việc chênh lệch số tiền chi tiêu của khách hàng khi vào đầu tháng và cuối tháng).

3.2.2 Phương pháp phân tích RFM (Mô hình RFM)

- RFM là viết tắt của cụm từ :

- Recency : Thời gian chênh lệch kể từ lần mua hàng cuối cùng so với hiện tại
 $R(\text{days}) = \text{now} - \text{last_purchase_date}$
- Frequency : Tần suất mua hàng của khách hàng
 $F = \text{total order}$
- Monetary : Tổng số tiền đã chi tiêu của khách hàng. Đây là yếu tố quan trọng nhất ảnh hưởng trực tiếp tới doanh số của công ty, doanh nghiệp
 $M = \text{total money}$

- Đây là một mô hình phổ biến được sử dụng ở nhiều lĩnh vực (như Marketing, kinh doanh,...) chủ yếu sử dụng 3 yếu tố kể trên để phân loại được giá trị của khách hàng cũng như tìm kiếm các khách hàng tiềm năng dành cho cửa hàng, doanh nghiệp hoặc tổ chức. Nó còn giúp doanh nghiệp biết được tình trạng kinh doanh hiện tại, như chỉ số Recency càng lớn cho thấy xu hướng rời bỏ của khách hàng càng cao. Ngoài ra việc phân tích như vậy cũng sẽ giúp ích rất nhiều cho việc phân cụm các khách hàng trong bài toán của nhóm vì dựa vào mô hình này sẽ giúp giảm số chiều của dữ liệu từ đó giảm thời gian xử lý, tăng hiệu năng mà vẫn giữ được kết quả tốt nhất có thể.



Hình 3.10. Hình ảnh mô hình RFM

- Mô hình sẽ được áp dụng tùy thuộc vào mục đích của công ty, tổ chức mong muốn như thế nào (ví dụ có thể chia khách hàng thành 4 nhóm : tiềm năng, ít lui đến, thường xuyên, VIP). Nó còn được sử dụng để tạo ra hay cải thiện mối quan hệ thân thiết giữa doanh nghiệp với khách hàng, ngoài ra còn có thể sử dụng để gửi các ưu đãi, chương trình khuyến mãi (bằng email hoặc tin nhắn) đến các khách hàng nhằm “lôi kéo” đến với trung tâm hay cửa hàng của doanh nghiệp. Từ đó giúp gia tăng doanh thu, lợi nhuận của công ty lên một cách đáng kể.
- Tại đây, nhóm sẽ sử dụng mô hình RFM kết hợp với thuật toán K-Means để có thể áp dụng, đưa ra quyết định và kết quả chính xác nhất có thể. Bằng cách sử dụng mô hình để tính toán các giá trị như R-F-M.

3.2.3 Loại bỏ các dữ liệu ngoại lai (Outliers)

- Việc làm sạch dữ liệu là một việc vô cùng khó khăn bởi dữ liệu nhiễu hay ngoại lai còn tùy thuộc vào định nghĩa của mỗi người. Nhóm sẽ sử dụng các thư viện đã có sẵn trong Python để thực hiện việc loại bỏ các dữ liệu có thể làm ảnh hưởng đến việc phân cụm khi đưa vào thuật toán K-Means như : loại bỏ các giá trị null, số âm, các giá trị không thể tính toán,...
- Ngoài ra còn kết hợp với việc kiểm tra sau mỗi bước hoặc sau khi lọc nhằm kiểm định lại độ chính xác cũng như không làm cho dữ liệu ban đầu trở nên khó kiểm soát hơn.
- Bởi vì thuật toán K-Means rất nhạy cảm với các dữ liệu nhiễu nên việc loại bỏ dữ liệu nhiễu là việc làm cần thiết trước khi dữ liệu được đưa vào xử lý các bước tiếp theo.

3.3 Gom nhóm cơ sở dữ liệu theo các đặc trưng chung

- Với bài toán phân loại khách hàng, nhóm sẽ tập trung vào việc lựa chọn các đặc trưng để phân tích, dễ nhận biết nhất đối với nguồn dữ liệu то lớn đó là Tổng số tiền chi tiêu của khách hàng. Ngoài ra dựa trên mô hình RFM mà nhóm tiến hành gom nhóm dựa trên các yếu tố được nói tại mô hình kết hợp để gom một số cột, giá trị lại từ đó có thể triển khai ra các luồng phân tích sâu hơn vào bài toán (ví dụ phân tích tỷ lệ giữ chân của khách hàng, ...).
- Dựa theo RFM, nhóm sẽ tiến hành gom nhóm theo 3 yếu tố chính và biểu thị như sau:
 - Recency: ‘InvoiceData’ = (today_date - date.max()).days
 - Frequency: ‘Invoice’ = inv.nunique()
 - Monetary: ‘TotalPrice’ = price.sum()
- Ngoài ra nhóm sẽ áp dụng thuật toán K-Means như đã trình bày trong chương 2 để tiến hành gom nhóm các khách hàng. Mỗi khách hàng như vậy được xem như một đối tượng mà thuật toán K-Means sẽ phân loại. Để sử dụng được K-Means thì cần phải chỉ rõ số cụm k là bao nhiêu, trọng tâm xác định thế nào, sử dụng hàm độ đo lân cận thế nào để gán mỗi khách hàng với trọng tâm của từng cụm tương ứng.
- Hàm độ đo lân cận mà nhóm sử dụng chính là khoảng cách Euclidean bình phương trong đó trọng tâm của mỗi nhóm là trung bình cộng của tất cả các khách hàng trong nhóm đó.

Kí hiệu	Mô tả
t	Là một điểm

C_i	Nhóm thứ i
c_i	Trọng tâm của nhóm i
c	Trọng tâm của tất cả các điểm
m_i	Số điểm trong nhóm thứ i
M	Số đối tượng trong tập dữ liệu
k	Số nhóm

Bảng 3.1. Bảng ký hiệu Euclide

- Khoảng cách Euclide bình phương giữa hai điểm được xác định như sau :

$$d(t_i, t_j)^2 = (t_{i1} - t_{j1})^2 + (t_{i2} - t_{j2})^2 + \dots + (t_{iT} - t_{jT})^2$$

- Trọng tâm của mỗi nhóm được xác định như sau :

$$c_i = \frac{1}{m_i} \sum_{t \in C_i} t, i = 1, \dots, k$$

- Khoảng cách Euclide bình phương từ một điểm đến trọng tâm được xác định như sau:

$$d(t, c_i)^2 = (t_1 - c_{i1})^2 + (t_2 - c_{i2})^2 + \dots + (t_T - c_{iT})^2$$

- Trong tập dữ liệu mà nhóm thu thập và tìm kiếm được, nó bao gồm rất nhiều dạng dữ liệu như :"Invoice", "Quantity", "Price", "InvoiceData", Và có đầy đủ loại dữ liệu từ số, chữ đến kiểu date. Tuy nhiên do sử dụng gom nhóm kết hợp với mô hình RFM và chia nhỏ lượng dữ liệu để phân tích ra, nhóm có thể giảm số lượng chiều của đồ thị một cách đáng kể. Ví dụ việc chọn "Recency" yêu cầu tối thiểu 1 vector "InvoiceDate" để thực hiện việc lọc dữ liệu → giảm lượng chiều của vector mà không cần sử dụng đến một cách đáng kể như "Price", "CustomerID",...

3.4 Chọn lựa đặc trưng và phân cụm dữ liệu

- Việc chọn lựa các đặc trưng cũng như phân cụm dữ liệu mà nhóm thực hiện tương tự như việc gom nhóm. Trong đó nhóm sẽ dùng mô hình RFM như đã nêu ở phía trên để có thể chọn

ra những điểm đặc trưng nhất của tập dữ liệu đầu vào để có thể đạt được kết quả tốt nhất có thể.

- Ngoài ra nhóm dựa vào lượng dữ liệu đã có sẵn và thực hiện phân tích sâu hơn thông qua các giải pháp như Mini Batch K-Means hay Online Learning K-Means đã được nêu ở trên.
- Bằng cách áp dụng Mini batch K-Means, nhóm có thể so sánh độ tối ưu giữa K-Means thuận và chính thuật toán này. Không những thế, nó có thể giúp cho nhóm có thể phân tích được nguồn dữ liệu lớn mà không bị sai sót hay đi sai hướng phân tích (phân tích theo hướng K-Means).
- Nhóm cũng sẽ thực hiện tìm hiểu các giải pháp khác để có thể biểu thị dữ liệu ban đầu thành những biểu đồ, hình ảnh trực quan nhất có thể và tìm tòi các phương pháp có thể có nhằm tối ưu hóa thuật toán.

CHƯƠNG 4 : CÀI ĐẶT CHƯƠNG TRÌNH THỰC NGHIỆM, KẾT QUẢ VÀ ĐÁNH GIÁ

4.1 Thu thập và chuẩn hóa tập dữ liệu

- Nhận xét về tập dữ liệu: Tập dữ liệu khách hàng mà nhóm sử dụng một phần được lấy từ nguồn UCI Machine Learning Repository với hơn 500K dòng dữ liệu, bao gồm các khách hàng đến từ khắp nơi trên thế giới nhưng chủ yếu là ở Anh trong những năm 2009-2010. Tập dữ liệu bao gồm các trường như Invoice, Stock Code, Description, Quantity, Invoice Date, Price, Customer ID và Country.
- Để phân loại khách hàng nhóm em sẽ tiến hành chuẩn hóa dữ liệu trong tập dữ liệu thu thập ở trên từ đó xây dựng nên tập dữ liệu được sử dụng để tiến hành phân loại trong bài báo cáo này. Quá trình chuẩn hóa dữ liệu bao gồm các thao tác sau :
- Tiến hành lọc dữ liệu theo các bước sau :

```
# begin cleaning data
# check null or empty data
df.isnull().sum()
```

Hình 4.1. Kiểm tra các giá trị “null” hay rỗng

```
# drop all null values
df.dropna(inplace=True)
df = df.dropna(subset=['Customer ID'])
```

Hình 4.2. Lọc dữ liệu giá trị “null”

```
# check duplicate data
df.duplicated().sum()
```

Hình 4.3. Kiểm tra các giá trị trùng lặp

```
# drop all duplicate data  
df = df.drop_duplicates()
```

```
# check again duplicate data for sure it is removed  
df.duplicated().sum()
```

Hình 4.4. Lọc các dữ liệu trùng lặp và kiểm tra lại

```
# remove refund invoices  
df = df[~df['Invoice'].str.contains('C', na=False)]  
df.describe().T
```

Hình 4.5. Lọc các đơn hàng bắt đầu bằng “C” do đây là các đơn hàng bị huỷ

```
indx = df.loc[df['Price'] == 0].index  
df.drop(index= indx, inplace= True)
```

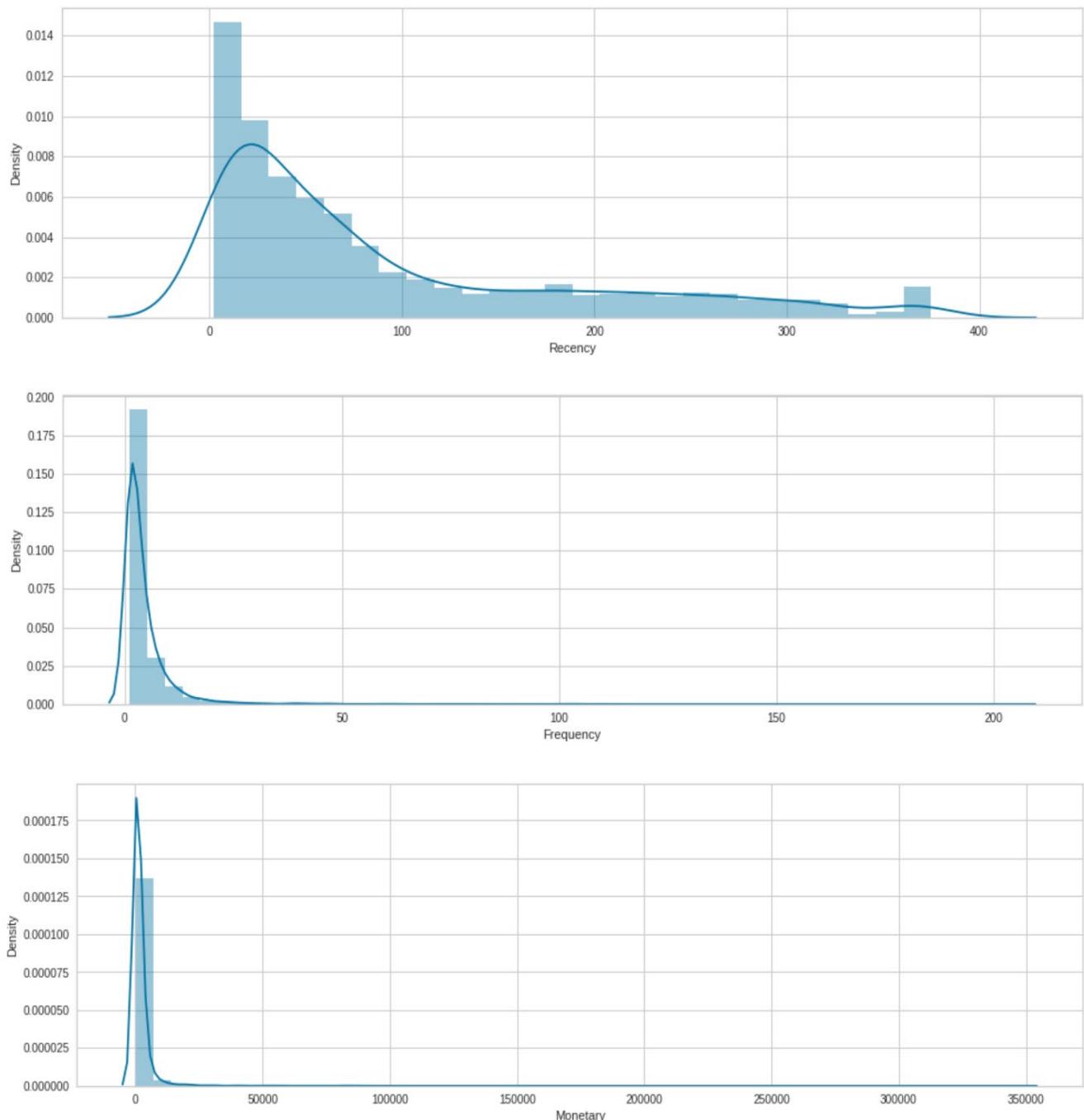
Hình 4.6. Lọc các giá trị Price nếu bằng 0 do đây là hàng tặng kèm

```
# customers having more than Mean + 3 Std (Z-score > 3). will be dropped
# define frequency threshold value and drop customers who exceed the threshold
freq_stats = rfm_seg['Frequency'].describe()
freq_threshold = freq_stats['mean'] + 3 * freq_stats['std']
indx = rfm_seg.loc[rfm_seg['Frequency'] > freq_threshold].index
rfm_seg.drop(index = indx, inplace= True)
```

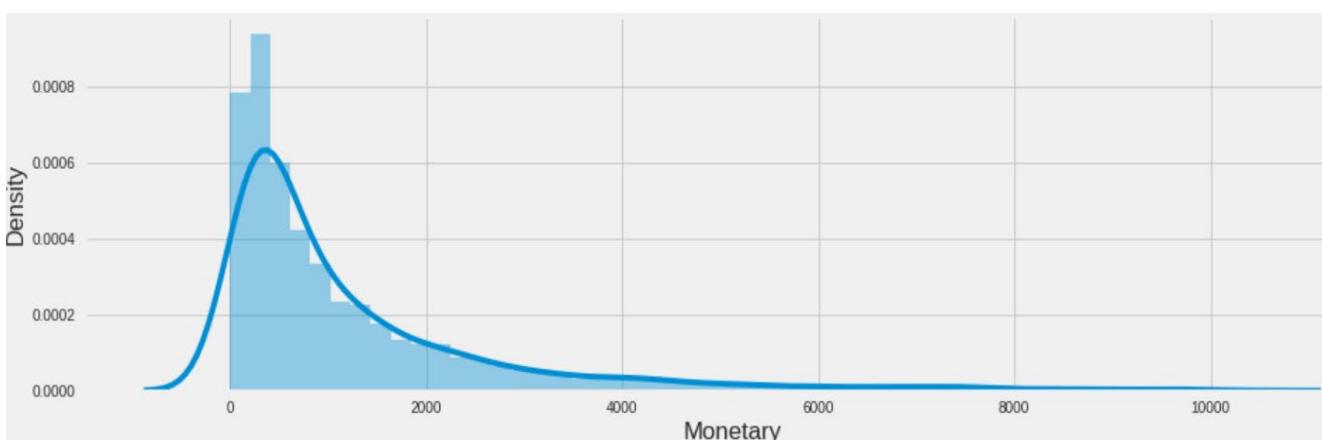
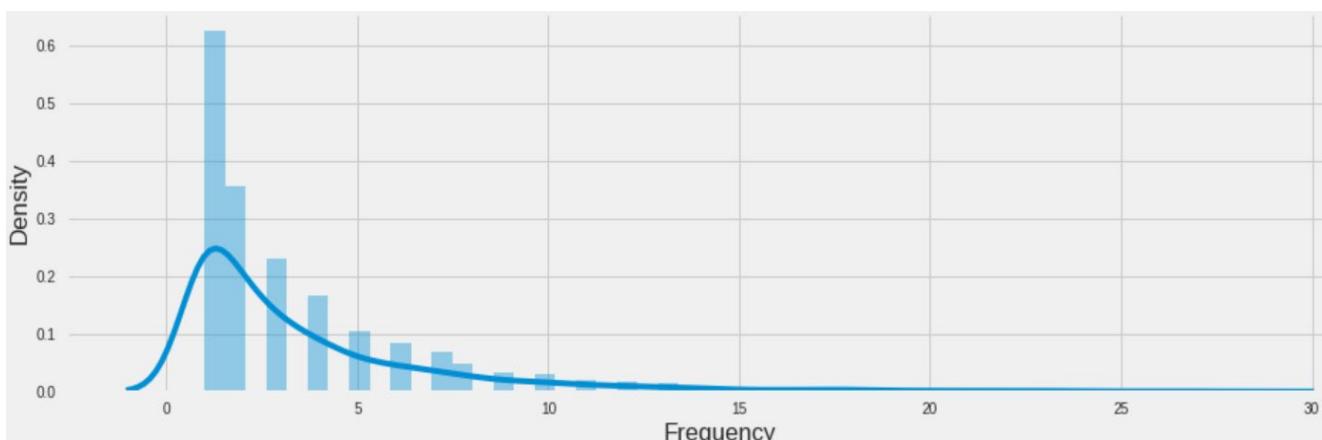
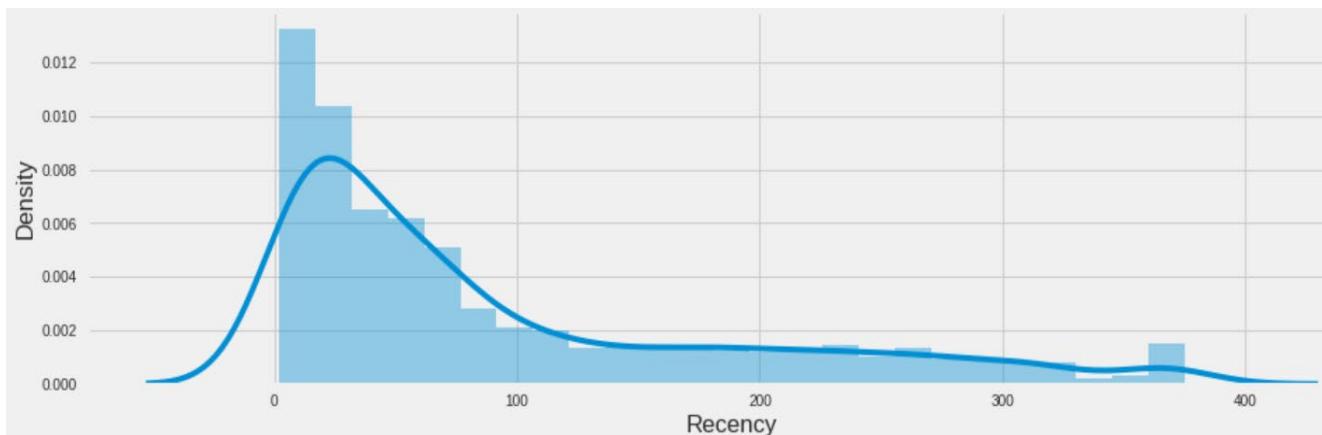
```
# define Monetary value threshold value and drop customers who exceed the threshold
m_stats = rfm_seg['Monetary'].describe()
m_threshold = m_stats['mean'] + 3 * m_stats['std']
indx = rfm_seg.loc[rfm_seg['Monetary'] > m_threshold].index
rfm_seg.drop(index = indx, inplace= True)
```

```
f, ax = plt.subplots(figsize=(14, 13))
plt.subplot(3, 1, 1)
sns.distplot(rfm_seg['Recency'], label= 'Recency')
plt.subplot(3, 1, 2)
sns.distplot(rfm_seg['Frequency'], label= 'Frequency')
plt.subplot(3, 1, 3)
sns.distplot(rfm_seg['Monetary'], label= 'Monetary')
plt.style.use('fivethirtyeight')
plt.tight_layout()
plt.show()
```

Hình 4.7. Code cho việc làm sạch dữ liệu đầu vào



Hình 4.8. Các biểu đồ RFM trước khi được làm sạch



Hình 4.9. Các biểu đồ RFM sau khi được làm sạch các giá trị nhiễu

- Các bước gom nhóm, lấy đặc trưng của dữ liệu

```
# create a new variable by multiplying the price per unit with Quantity
df['TotalPrice'] = df['Quantity'] * df['Price']
df.head()
```

Hình 4.10. Gom nhóm theo Tổng chi tiêu “Total Price”

```
# check min and max invoice date
print('Min Invoice Date:', df.InvoiceDate.dt.date.min(), '\nMax Invoice Date:', df.InvoiceDate.dt.date.max())
```

```
today_date = df['InvoiceDate'].max() + dt.timedelta(days=2)
today_date
```

Hình 4.11. Kiểm tra dữ liệu để có thể gom nhóm, phân tích chính xác hơn

```
rfm = df.groupby('Customer ID').agg({'InvoiceDate': lambda date: (today_date - date.max()).days, 'Invoice': lambda inv: inv.nunique(), 'TotalPrice': lambda total: total.sum()})
rfm.columns = ['Recency', 'Frequency', 'Monetary']
rfm.head()
```

Hình 4.12. Gom nhóm dữ liệu dựa trên mô hình RFM

Customer ID	Recency	Frequency	Monetary
12346.00000	166	11	372.86000
12347.00000	4	2	1323.32000
12348.00000	75	1	222.16000
12349.00000	44	3	2671.14000
12351.00000	12	1	300.93000

Hình 4.13. Kết quả thực nghiệm gom nhóm

```

rfm_seg = rfm.copy()

r_labels = range(4, 0, -1)
f_labels = range(1, 5)
m_labels = range(1, 5)
r_quartiles = pd.qcut(rfm_seg['Recency'], q=4, labels= r_labels)
# f_quartiles = pd.qcut(rfm_seg['Frequency'], q=4, labels= f_labels)
m_quartiles = pd.qcut(rfm_seg['Monetary'], q=4, labels= m_labels)
# rfm_seg = rfm_seg.assign(R= r_quartiles, F= f_quartiles, M= m_quartiles)

# frequency is duplicated, solution : https://stackoverflow.com/questions/
def pct_rank_qcut(series, n):
    edges = pd.Series([float(i) / n for i in range(n + 1)])
    f = lambda x: (edges >= x).values.argmax()
    return series.rank(pct=1).apply(f)
f_quartiles = rfm_seg['Frequency'].astype(float)
f_quartiles = pct_rank_qcut(rfm_seg['Frequency'], 4)
rfm_seg = rfm_seg.assign(R= r_quartiles, F= f_quartiles, M= m_quartiles)

# build RFM segments and RFM score
def add_rfm_seg(x):
    return str(int(x['R'])) + str(int(x['F'])) + str(int(x['M']))
rfm_seg['RFM_Segment'] = rfm_seg.apply(add_rfm_seg, axis=1)
rfm_seg['RFM_Score'] = rfm_seg[['R', 'F', 'M']].sum(axis=1)
rfm_seg.head()

```

Hình 4.14. Tiến hành thử nghiệm tính gom nhóm theo mô hình RFM nâng cao

	Recency	Frequency	Monetary	R	F	M	RFM_Segment	RFM_Score
Customer ID								
12346.00000	166	11	372.86000	1	4	2	142	7
12347.00000	4	2	1323.32000	4	2	3	423	9
12348.00000	75	1	222.16000	2	1	1	211	4
12349.00000	44	3	2671.14000	3	3	4	334	10
12351.00000	12	1	300.93000	4	1	1	411	6

Hình 4.15. Kết quả sau khi thực nghiệm

4.2 Cài đặt chương trình thực nghiệm

- Mô tả sơ lược về bài toán như sau :

- Đầu vào : Tập dữ liệu không được dán nhãn đã được chuẩn hóa theo mô hình RFM
- Đầu ra : Các khách hàng được phân vào 6 cụm riêng biệt theo thuật toán K-Means và Mini Batch K-Means

- Nhóm chúng em cài đặt chương trình để cho “máy học” có thể học được thông qua Colab (hay còn gọi là Colaboratory) do Google phát triển. Thích hợp để thực nghiệm trên môi trường này do đã được cấu hình sẵn môi trường dành cho ngôn ngữ Python với sự hỗ trợ của ứng dụng chạy trên nền web nổi tiếng là Jupyter Notebook – nơi cho phép đưa cả code Python, công thức, hình ảnh, video, biểu thức,... vào trong cùng 1 file duy nhất. Ngoài ra còn có thể truy cập ngay tại trình duyệt mà không yêu cầu cấu hình, sử dụng miễn phí CPU, GPU (ở mức độ nhất định) và việc chia sẻ code giữa các thành viên trong nhóm cũng dễ dàng hơn.

- Về dữ liệu đầu vào bao gồm các đặc trưng là các thông tin của khách hàng khi đến mua hàng và được miêu tả như sau như hình sau :

	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
0	489434	85048	15CM CHRISTMAS GLASS BALL 20 LIGHTS	12	2009-12-01 07:45:00	6.95000	13085.00000	United Kingdom
1	489434	79323P	PINK CHERRY LIGHTS	12	2009-12-01 07:45:00	6.75000	13085.00000	United Kingdom
2	489434	79323W	WHITE CHERRY LIGHTS	12	2009-12-01 07:45:00	6.75000	13085.00000	United Kingdom
3	489434	22041	RECORD FRAME 7" SINGLE SIZE	48	2009-12-01 07:45:00	2.10000	13085.00000	United Kingdom
4	489434	21232	STRAWBERRY CERAMIC TRINKET BOX	24	2009-12-01 07:45:00	1.25000	13085.00000	United Kingdom

Hình 4.16. Thông tin về dữ liệu đầu vào

1. Invoice : mã đơn hàng khách hàng đã mua
2. StockCode : mã sản phẩm trong đơn hàng
3. Description : thông tin về sản phẩm trong đơn hàng
4. Quantity : số lượng sản phẩm đã được mua
5. InvoiceDate : ngày mà khách hàng đến mua hàng (tính theo dạng yyyy-mm-dd hh:mm:ss)
6. Price : giá tiền của sản phẩm
7. CustomerID : ID của khách hàng, sử dụng để phân biệt giữa các khách hàng với nhau
8. Country : thông tin quốc gia

- Tiến hành import các thư viện cần thiết cần thiết

```
# install library
!pip install openpyxl
# !pip install pqkmeans

import time
import numpy as np
import pandas as pd
import datetime as dt
import warnings
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import plotly.express as px
import seaborn as sns
import plotly.graph_objs as go
import sklearn.preprocessing as pp

from plotly.offline import init_notebook_mode, iplot
from sklearn.metrics.pairwise import pairwise_distances_argmin
from sklearn.datasets import make_blobs
from sklearn.preprocessing import MinMaxScaler, StandardScaler, OneHotEncoder
from sklearn.cluster import KMeans, MiniBatchKMeans
from yellowbrick.cluster import KElbowVisualizer, SilhouetteVisualizer
from IPython.core.pylabtools import figsize
from mpl_toolkits.mplot3d import Axes3D
from IPython.display import display
from ipywidgets import widgets, HBox
from sklearn.metrics import silhouette_score
from sklearn.manifold import MDS
from plotly.subplots import make_subplots
from mlxtend.plotting import plot_learning_curves
```

Hình 4.17. Import các thư viện cần thiết

- Câu hình và đọc file dữ liệu theo định dạng xlsx lên trên Colab

```
for dirname, _, filenames in os.walk('/content/drive/MyDrive'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
# result : /content/drive/MyDrive/online_retail_II.xlsx
```

```

# set options
pd.set_option('display.max_columns', None)
pd.set_option('display.float_format', lambda x: '%.5f' % x)

# set warnings
warnings.filterwarnings('ignore')
warnings.simplefilter(action='ignore', category=FutureWarning)

# load data from xlsx files
df_ = pd.read_excel('/content/drive/MyDrive/online_retail_II.xlsx', sheet_name='Year 2009-2010', engine='openpyxl')
df = df_.copy()
df.head()

```

Hình 4.18. Đọc file dữ liệu định dạng xlsx

- Tiến hành viết hàm kiểm tra các thông tin cơ bản của dữ liệu đầu vào

```

# exploratory data
def check_df(dataframe):
    print('===== Shape =====')
    print(dataframe.shape)
    print('===== Types =====')
    print(dataframe.dtypes)
    print('===== Head =====')
    print(dataframe.head(5))
    print('===== Tail =====')
    print(dataframe.tail(5))
    print('===== NA =====')
    print(dataframe.isnull().sum())
    print('===== Quantiles =====')
    print(dataframe.quantile([0, 0.05, 0.50, 0.95, 0.99, 1]).T)

```

Hình 4.19. Hàm kiểm tra các thông tin cơ bản

- Sau đó sẽ loại bỏ, lọc các dữ liệu nhiễu, ngoại lai để giúp dữ liệu đầu vào trở nên tốt hơn. Đã được nêu ra ở phần 4.1
- Gom nhóm dữ liệu lại theo các đặc trưng

```

# check number of unique customers
df['Customer ID'].nunique()

```

```

# create a new variable by multiplying the price per unit with Quantity
df['TotalPrice'] = df['Quantity'] * df['Price']
df.head()

```

```
today_date = df['InvoiceDate'].max() + dt.timedelta(days=2)
today_date
```

```
rfm = df.groupby('Customer ID').agg({'InvoiceDate': lambda date: (today_date - date.max()).days, 'Invoice': lambda inv: inv.nunique()})
rfm.columns = ['Recency', 'Frequency', 'Monetary']
rfm.head()
```

```
# summary metrics per RFM score
# 'count' column is number of customers
rfm_seg.groupby('RFM_Score').agg({'Recency': 'mean', 'Frequency': 'mean', 'Monetary': ['mean', 'count']})).round(1)
```

Hình 4.20. Gom nhóm dữ liệu theo mô hình RFM

- Thực nghiệm việc kiểm tra nơi ở của khách hàng và hiện thực hóa bằng thư viện. Kết quả [1].

```
# simple map of countries
data_map = dict(type='choropleth', locations = countries.index, locationmode = 'country names', z = countries, title='Number of rows per country', geo=dict(showframe=True,
layout = dict(autosize=False, width=1000, height=600, title='Number of rows per country', geo=dict(showframe=True,
map_of_countries = go.Figure(data=[data_map], layout=layout)
iplot(map_of_countries, validate=False)
```

Hình 4.21. Phân bố của các khách hàng ở các quốc gia khác nhau

- Tiến hành các bước phân tích. Kết quả [2].

```
# percentage customers from UK and Outside UK
x = df['Country'].apply(lambda x: x if x == 'United Kingdom' else 'Not United Kingdom').value_counts().rename('Total Rows')
y = (x/df.shape[0]).rename('% Rows')
pd.concat([x, y], axis=1)
```

Hình 4.22. Phần trăm khách hàng đến từ Anh và ngoài nước Anh

- Viết hàm chuẩn bị dữ liệu và tiến hành thực nghiệm tỉ lệ quay lại của khách hàng. Kết quả [3], [4], [5].

```

# data for heatmap
# data cointains from 2009/12/01 to 2010/12/09
def get_month(x):
    return dt.datetime(x.year, x.month, 1)

df_heat['InvoiceMonth'] = df_heat['InvoiceDate'].apply(get_month)
grouping = df_heat.groupby('Customer ID')['InvoiceMonth']
df_heat['CohortMonth'] = grouping.transform('min')
df_heat.tail()

```

```

# get year, month, day and calculate cohort index
def get_month_int(dframe, column):
    years = dframe[column].dt.year
    months = dframe[column].dt.month
    days = dframe[column].dt.day
    return years, months, days

invoice_year, invoice_month,_ = get_month_int(df_heat, 'InvoiceMonth')
cohort_year, cohort_month,_ = get_month_int(df_heat, 'CohortMonth')

year_diff = invoice_year - cohort_year
month_diff = invoice_month - cohort_month

df_heat['CohortIndex'] = year_diff * 12 + month_diff + 1

```

```

# count monthly active customers from each cohort
grouping = df_heat.groupby(['CohortMonth', 'CohortIndex'])
cohort_data = grouping['Customer ID'].apply(pd.Series.nunique)

# return number of unique elements in the object
cohort_data = cohort_data.reset_index()
cohort_counts = cohort_data.pivot(index= 'CohortMonth', columns= 'CohortIndex', values= 'Customer ID')
print(cohort_counts)

```

```

# create retention rate table
# get column 0 as a SERIES of shape (n,) (cohort sizes)
cohort_size = cohort_counts.iloc[:,0]
# axis=0 to ensure the divide along the row axis
retention = cohort_counts.divide(cohort_size, axis= 0)
# to show the number as percentage
retention.round(3) * 100

```

```
# build a heatmap for retention rates of customers
plt.figure(figsize=(15, 7))
plt.title('Retention rates of customers')
sns.heatmap(data=retention, annot=True, fmt= '.0%', vmin= 0.0, vmax= 0.5, cmap='PuBu_r')
plt.show()
```

Hình 4.23. Dữ liệu đánh giá tỉ lệ quay lại của khách hàng

- Viết hàm phân loại theo RFM_Score. Kết quả [6].

```
# use RFM score to group customers into Diamond, Platinum, Gold, Silver, Bronze and Iron segments
def segments(df):
    if (df['RFM_Score'] > 11):
        return 'Diamond'
    elif (df['RFM_Score'] > 10) and (df['RFM_Score'] <= 11):
        return 'Platinum'
    elif (df['RFM_Score'] > 8) and (df['RFM_Score'] <= 10):
        return 'Gold'
    elif (df['RFM_Score'] > 6) and (df['RFM_Score'] <= 8):
        return 'Silver'
    elif (df['RFM_Score'] > 4) and (df['RFM_Score'] <= 6):
        return 'Bronze'
    else:
        return 'Iron'

# 'count' column is number of customers
rfm_seg['General_Segment'] = rfm_seg.apply(segments, axis=1)
rfm_seg.groupby('General_Segment').agg({'Recency': 'mean', 'Frequency': 'mean', 'Monetary': ['mean', 'count']}).round(1)
```

Hình 4.24. Phân loại khách hàng thành các bậc dựa trên RFM_Score

- Tiến hành viết hàm và thực nghiệm việc chuẩn hoá dữ liệu đối với 3 loại dữ liệu “Recency”, “Frequency”, “Monetary” theo hàm Logarit. Kết quả [7].

```
# unskew the data - log transformation
rfm_log = rfm[['Recency', 'Frequency', 'Monetary']].apply(np.log, axis=1).round(3)
# rfm_log = np.log(rfm_rfm)
# plot the distribution of RFM values
f, ax = plt.subplots(figsize=(14, 13))
plt.subplot(3, 1, 1)
sns.distplot(rfm_log['Recency'], label= 'Recency')
plt.subplot(3, 1, 2)
sns.distplot(rfm_log['Frequency'], label= 'Frequency')
# type of some values monetary is category
plt.subplot(3, 1, 3)
# print(rfm_log['Monetary'].astype('category'))
# rfm_log[['Monetary']] = rfm_log[['Monetary']].apply(lambda col:pd.Categorical(col).codes)
# print(rfm_log['Monetary'] / 100)
sns.distplot(rfm_log['Monetary'], label= 'Monetary')
plt.style.use('fivethirtyeight')
plt.tight_layout()
plt.show()
```

Hình 4.25. Chuẩn hoá dữ liệu theo Logarit

- Bắt đầu thực nghiệm thuật toán Elbow để tìm số cụm phù hợp. Kết quả [8].

```
# elbow method will be used to determine the optimum number of clusters for rfm dataframe
kmeans = KMeans()
# kmeans = MiniBatchKMeans()
ssd = []
K = range(1, 30)

for k in K:
    kmeans = KMeans(n_clusters=k).fit(df)
    # kmeans = MiniBatchKMeans(n_clusters=k).fit(df)
    ssd.append(kmeans.inertia_)

plt.plot(K, ssd, 'bx-')
plt.xlabel('Distance Residual Sums Per Different k Values')
plt.title('Elbow Method for Optimum Number of Clusters')
plt.show()

# make clear the number of clusters
kmeans = KMeans()
# kmeans = MiniBatchKMeans()
visu = KEElbowVisualizer(kmeans, k=(2, 20))
visu.fit(df)
visu.show()
```

Hình 4.26. Thuật toán Elbow để tìm số cụm phù hợp

- Bắt đầu kiểm tra lại bằng thuật toán Silhouette. Kết quả [9].

```
# make sure the number of clusters by checking with Silhouette Method
sil_avg=[]
for s in range(2, 20):
    cluster_sil = KMeans(n_clusters = s).fit_predict(df)
    silhouette_avg = silhouette_score(df, cluster_sil)
    sil_avg.append([s, silhouette_avg])

# create new array
sil_avg = np.array(sil_avg)
plt.plot(sil_avg[:, 0], sil_avg[:, 1], linestyle='dashed')
plt.xlabel('Number of Clusters')
plt.ylabel('Average Silhouette Score')
plt.legend(['avg_sil'])
plt.show()
```

```

# average Silhouette Score from 4 to 7 clusters
fig, ax = plt.subplots(2, 2, figsize=(20,17))
axli = ax.flatten()
j = 0
for s in range(4, 8):
    cluster_silhouette = KMeans(n_clusters = s)
    visualizer = SilhouetteVisualizer(cluster_silhouette, colors='yellowbrick', ax= axli[j])
    visualizer.fit(df)
    visualizer.finalize()
    j+=1

```

Hình 4.27. Kiểm tra lại số cụm bằng thuật toán Silhouette

- Tiến hành việc phân cụm theo 2 thuật toán K-Means và Mini Batch K-Means cũng như so sánh tốc độ thực thi giữa 2 thuật toán này. Kết quả [10].
- Ngoài ra ở phương pháp Mini Batch K-Means nhóm đã sử dụng hàm “partial_fit” thay vì là “fit” như bình thường (được hỗ trợ sẵn trong thư viện Scikit Learn). Đây là phương pháp Online Learning K-Means giúp cập nhật K tâm cụm với mỗi đối tượng dữ liệu đầu vào.

```

%time kmeans = KMeans(n_clusters=6, init= 'k-means++', n_init=10, random_state=0, max_iter=1000, verbose=0)
# calculate running time of kmeans
start_time = time.time()
kmeans.fit(df)
end_time = time.time() - start_time
# print(end_time)
# kmeans.fit_predict(df)
clusters = kmeans.labels_
print('\nTotal cell runtime: ', )

%time mbk = MiniBatchKMeans(n_clusters=6, init= 'k-means++', n_init=10, random_state=0, max_no_improvement=10, batch_size=100, verbose=0)
# calculate running time of MiniBatchKMeans
start_time_mbk = time.time()
# NOTE : reduce memory usage for big dataset, let you update the model with incremental data when the whole dataset cannot be loaded into
# partial_fit is for online clustering were fit is for offline
# partial_fit update k means estimate on a single mini-batch
mbk.partial_fit(df_mbk)
# mbk.fit(df_mbk)
end_time_mbk = time.time() - start_time_mbk
# print(end_time_mbk)
clusters_mbk = mbk.labels_
print('\nTotal cell runtime: ', )

```

Hình 4.28. Phân cụm khách hàng theo 2 thuật toán

- Tính toán số lượng khách hàng theo RFM giữa các cluster. Kết quả [11].

```

# calculate average RFM values and size for each cluster
rfm_rfms = rfm_rfm.copy()
rfm_rfm_kmeans = rfm_rfm.assign(Clusters= clusters + 1)
rfm_rfm_mbk = rfm_rfms.assign(Clusters= clusters_mbk + 1)

print(rfm_rfm_kmeans.groupby('Clusters').agg({'Recency': 'mean', 'Frequency': 'mean', 'Monetary': ['mean', 'count']})).round(0))
print(rfm_rfm_mbk.groupby('Clusters').agg({'Recency': 'mean', 'Frequency': 'mean', 'Monetary': ['mean', 'count']})).round(0))

```

Hình 4.29. Dữ liệu khách hàng sau khi được phân cụm được xử lý tiếp

- Thực nghiệm việc phân nhóm theo “RFM_Score” và so sánh các kết quả với nhau. Kết quả [12].

```
# snake plots to understand and compare segments
rfm_normalized = pd.DataFrame(rfm_normalized, index=rfm_rfms.index, columns=rfm_rfms.columns)
rfm_normalized['Clusters'] = clusters + 1
rfm_normalized['General_Segment'] = rfm_seg['General_Segment']
rfm_normalized.reset_index(inplace=True)

rfm_melt = pd.melt(rfm_normalized, id_vars=['Customer ID','General_Segment','Clusters'],value_vars=['Recency', 'Frequency', 'Monetary'])
rfm_melt.head()

# snake plots for mini batch kmeans
rfm_normalized_mbk = pd.DataFrame(rfm_normalized_mbk, index=rfm_rfms.index, columns=rfm_rfms.columns)
rfm_normalized_mbk['Clusters'] = clusters_mbk + 1
rfm_normalized_mbk['General_Segment'] = rfm_seg['General_Segment']
rfm_normalized_mbk.reset_index(inplace=True)

rfm_melt_mbk = pd.melt(rfm_normalized_mbk, id_vars=['Customer ID', 'General_Segment', 'Clusters'], value_vars=['Recency', 'Frequency', 'Monetary'])
rfm_melt_mbk.head()

# snake plot
f, (ax1, ax2, ax3) = plt.subplots(1, 3, figsize=(20, 8))
sns.lineplot(x= 'Metric', y= 'Value', hue= 'General_Segment', data= rfm_melt, ax=ax1)
sns.lineplot(x = 'Metric', y = 'Value', hue = 'Clusters', data = rfm_melt, ax=ax2)
sns.lineplot(x = 'Metric', y = 'Value', hue = 'Clusters', data = rfm_melt_mbk, ax=ax3)
plt.suptitle('Snake Plot of RFM & KMeans & MiniBatchKMeans', fontsize=20)
plt.show()
```

Hình 4.30. Dữ liệu cho biểu đồ so sánh kết quả giữa các loại phân cụm

- Viết hàm hiển thị kết quả K-Means dưới dạng mô hình 3D. Kết quả [13].

```
# 3d version of kmeans
fig_3d_kmeans = px.scatter_3d(
    kmeans_3,
    x= 'Recency',
    y= 'Frequency',
    z= 'Monetary',
    color= (clusters + 1).astype(str),
    opacity= 0.7,
    height= 650,
    width= 700,
    title= 'Clusters Obtained using KMeans<br><sup>Training time: %.3fs & Inertia: %f</sup>' % (end_time, kmeans.inertia_),
    color_discrete_sequence = px.colors.qualitative.Set2
)
fig_3d_kmeans.show()
```

Hình 4.31. Kết quả phân cụm của K-Means theo mô hình 3D

- Tương tự với Mini Batch K-Means. Kết quả [14].

```

# 3d version of Mini Batch Kmeans
fig_3d_mbk = px.scatter_3d(
    mbk_3,
    x= 'Recency',
    y= 'Frequency',
    z= 'Monetary',
    color= (clusters_mbk + 1).astype(str),
    opacity= 0.7,
    height= 650,
    width= 700,
    title= 'Clusters Obtained using Mini Batch Kmeans<br><sup>Training time: %.3fs & Inertia: %f</sup>' % (end_time_mbk, mbk.inertia_),
    color_discrete_sequence = px.colors.qualitative.Set2
)
fig_3d_mbk.show()

```

Hình 4.32. Kết quả phân cụm của Mini Batch K-Means theo mô hình 3D

- Sau đó thực nghiệm, viết các hàm để so sánh giữa 2 loại thuật toán (nâng cao). Kết quả [15].

```

# create 3D chart compare 2 options
# first for kmeans algorithm
fig = plt.figure(figsize=(10, 9))
ax = fig.add_subplot(projection='3d')

ax.scatter(df[:,0], df[:,1], df[:,2], c= clusters, cmap= 'viridis', s= 70, alpha= 0.5)
ax.scatter(kmeans.cluster_centers_[:,0], kmeans.cluster_centers_[:,1], kmeans.cluster_centers_[:,2], s = 350, c = 'r', marker='x', edgecolor='k',
plt.title('Clusters Obtained using KMeans\nTraining time: %.3fs & Inertia: %f' % (end_time, kmeans.inertia_), fontsize = 20)
ax.set_xlabel("Recency")
ax.set_ylabel("Frequency")
ax.set_zlabel("Monetary")
plt.autoscale(enable=True, axis='x', tight=True)
kmeansImage = plt.savefig('Kmeans.png', bbox_inches='tight')
plt.show()
plt.close(fig)

```

```

# create 3D chart compare 2 options
# second for Mini Batch Kmeans
fig_mbk = plt.figure(figsize=(10, 9))
ax_mbk = fig_mbk.add_subplot(projection='3d')

ax_mbk.scatter(df[:,0], df[:,1], df[:,2], c= clusters_mbk, cmap= 'viridis', s= 70, alpha= 0.5)
ax_mbk.scatter(mbk.cluster_centers_[:,0], mbk.cluster_centers_[:,1], mbk.cluster_centers_[:,2], s = 350, c = 'r', marker='x', edgecolor='k',
plt.title('Clusters Obtained using Mini Batch KMeans\nTraining time: %.3fs & Inertia: %f' % (end_time_mbk, mbk.inertia_), fontsize = 20)
ax_mbk.set_xlabel("Recency")
ax_mbk.set_ylabel("Frequency")
ax_mbk.set_zlabel("Monetary")
plt.autoscale(enable=True, axis='x', tight=True)
mbkImage = plt.savefig('MiniBatchKmeans.png', bbox_inches='tight')
plt.show()
plt.close(fig_mbk)

```

```

# show and compare final result image
# inertia is the sum of squared distances of samples to their closest cluster center
imageKmeans = widgets.Image(value=open('Kmeans.png', 'rb').read())
imageMiniBatchKmeans = widgets.Image(value=open('MiniBatchKmeans.png', 'rb').read())
hbox = HBox([imageKmeans, imageMiniBatchKmeans])
display(hbox)

```

Hình 4.33. So sánh chi tiết giữa 2 thuật toán phân cụm khách hàng

- Tiến hành viết code để hiển thị các dạng dữ liệu theo mô hình RFM dưới dạng đồ thị để hiển thị kết quả một cách trực quan hơn. Kết quả [16], [17].

```

# distribution of customers
df_dis = pd.DataFrame()
df_dis['Distribution Of The KMeans Clusters']= rfm["Cluster"]
df_dis['Distribution Of The Mini Batch KMeans Clusters']= rfm_mbk["Cluster"]

fig, ax=plt.subplots(1,2, figsize=(20,7))
sns.countplot(df_dis['Distribution Of The KMeans Clusters'], ax=ax[0])
sns.countplot(df_dis['Distribution Of The Mini Batch KMeans Clusters'], ax=ax[1])
fig.show()

```

```

# recency each clusters
df_re = pd.DataFrame()
df_re['KMeans Clusters']= rfm["Cluster"]
df_re['Mini Batch KMeans Clusters']= rfm_mbk["Cluster"]

fig, (ax1, ax2)=plt.subplots(1,2, figsize=(20,7))
sns.swarmplot(x=df_re['KMeans Clusters'], y=rfm['Recency'], color= "#CBEDDD", alpha=0.5, ax=ax1)
sns.boxenplot(x=df_re['KMeans Clusters'], y=rfm['Recency'], ax=ax1)

sns.swarmplot(x=df_re['Mini Batch KMeans Clusters'], y=rfm_mbk['Recency'], color= "#CBEDDD", alpha=0.5, ax=ax2)
sns.boxenplot(x=df_re['Mini Batch KMeans Clusters'], y=rfm_mbk['Recency'], ax=ax2)
fig.show()

```

```

# frequency each clusters
df_fre = pd.DataFrame()
df_fre['KMeans Clusters']= rfm["Cluster"]
df_fre['Mini Batch KMeans Clusters']= rfm_mbk["Cluster"]

fig, (ax1, ax2)=plt.subplots(1,2, figsize=(20,7))
sns.swarmplot(x=df_fre['KMeans Clusters'], y=rfm['Frequency'], color= "#CBEDDD", alpha=0.5, ax=ax1)
sns.boxenplot(x=df_fre['KMeans Clusters'], y=rfm['Frequency'], ax=ax1)

sns.swarmplot(x=df_fre['Mini Batch KMeans Clusters'], y=rfm_mbk['Frequency'], color= "#CBEDDD", alpha=0.5, ax=ax2)
sns.boxenplot(x=df_fre['Mini Batch KMeans Clusters'], y=rfm_mbk['Frequency'], ax=ax2)
fig.show()

```

```

# monetary each clusters
df_mon = pd.DataFrame()
df_mon['KMeans Clusters']= rfm["Cluster"]
df_mon['Mini Batch KMeans Clusters']= rfm_mbk["Cluster"]

fig, (ax1, ax2)=plt.subplots(1,2, figsize=(20,7))
sns.swarmplot(x=df_mon['KMeans Clusters'], y=rfm['Monetary'], color= "#CBEDDD", alpha=0.5, ax=ax1)
sns.boxenplot(x=df_mon['KMeans Clusters'], y=rfm['Monetary'], ax=ax1)

sns.swarmplot(x=df_mon['Mini Batch KMeans Clusters'], y=rfm_mbk['Monetary'], color= "#CBEDDD", alpha=0.5, ax=ax2)
sns.boxenplot(x=df_mon['Mini Batch KMeans Clusters'], y=rfm_mbk['Monetary'], ax=ax2)
fig.show()

```

Hình 4.34. Dữ liệu kết quả được hiển thị trực quan hơn thông qua biểu đồ

- Ngoài ra, nhóm còn nghiên cứu thêm Learning Curves (Đường cong học tập) nhằm so sánh mức độ chính xác giữa 2 thuật toán cho các bộ dữ liệu đầu vào khác nhau cũng như hiệu suất học tập của các thuật toán trên. Tiến hành thực thi và cho ra kết quả như sau [18].

```
# sample data for training learning curve kmeans
X, y = mnist_data()
X, y = shuffle_arrays_unison(arrays=[X, y], random_seed=123)
X_train, X_test = X[:4000], X[4000:]
y_train, y_test = y[:4000], y[4000:]
plot_learning_curves(X_train, y_train, X_test, y_test, kmeans, scoring='accuracy', style='fast')
kmeansImage = plt.savefig('KmeansLearningCurve.png', bbox_inches='tight')
plt.show()
plt.close()
```

```
# sample data for training learning curve mini batch kmeans
X, y = mnist_data()
X, y = shuffle_arrays_unison(arrays=[X, y], random_seed=123)
X_train, X_test = X[:4000], X[4000:]
y_train, y_test = y[:4000], y[4000:]
plot_learning_curves(X_train, y_train, X_test, y_test, mbk, scoring='accuracy', style='fast')
mbkImage = plt.savefig('MiniBatchKmeansLearningCurve.png', bbox_inches='tight')
plt.show()
plt.close()
```

```
# show and compare final result image of learning curve
imageKmeansLearningCurve = widgets.Image(value=open('KmeansLearningCurve.png', 'rb').read())
imageMiniBatchKmeansLearningCurve = widgets.Image(value=open('MiniBatchKmeansLearningCurve.png', 'rb').read())
hbox_curve = HBox([imageKmeansLearningCurve, imageMiniBatchKmeansLearningCurve])
display(hbox_curve)
```

Hình 4.35. Learning Curves cho 2 thuật toán trên

4.3 Kết quả

- Các kết quả thu được sau khi thực nghiệm chương trình do nhóm viết ra.
- Kết quả dựa theo thuộc tính country và được đưa lên biểu đồ [1].

Number of rows per country



Hình 4.36. Biểu đồ thể hiện sự phân bố khách hàng

- Kết quả tỉ lệ phần trăm trong dữ liệu [2].

	Total Rows	% Rows
United Kingdom	372705	0.90735
Not United Kingdom	38058	0.09265

Hình 4.37. Tỉ lệ phần trăm trong tập dữ liệu đầu vào

- Kết quả phân tích số lượng khách hàng quay lại [3].

CohortIndex	1	2	3	4	5	6	\
CohortMonth							
2009-12-01	955.00000	337.00000	319.00000	406.00000	363.00000	343.00000	
2010-01-01	383.00000	79.00000	119.00000	117.00000	101.00000	115.00000	
2010-02-01	374.00000	89.00000	84.00000	109.00000	92.00000	75.00000	
2010-03-01	443.00000	84.00000	102.00000	107.00000	103.00000	90.00000	
2010-04-01	294.00000	57.00000	57.00000	48.00000	54.00000	66.00000	
2010-05-01	254.00000	40.00000	43.00000	44.00000	45.00000	65.00000	
2010-06-01	270.00000	47.00000	51.00000	55.00000	62.00000	77.00000	
2010-07-01	186.00000	29.00000	34.00000	55.00000	54.00000	19.00000	
2010-08-01	162.00000	33.00000	48.00000	52.00000	19.00000	NaN	
2010-09-01	243.00000	55.00000	57.00000	24.00000	NaN	NaN	
2010-10-01	377.00000	97.00000	35.00000	NaN	NaN	NaN	
2010-11-01	325.00000	35.00000	NaN	NaN	NaN	NaN	
2010-12-01	46.00000	NaN	NaN	NaN	NaN	NaN	

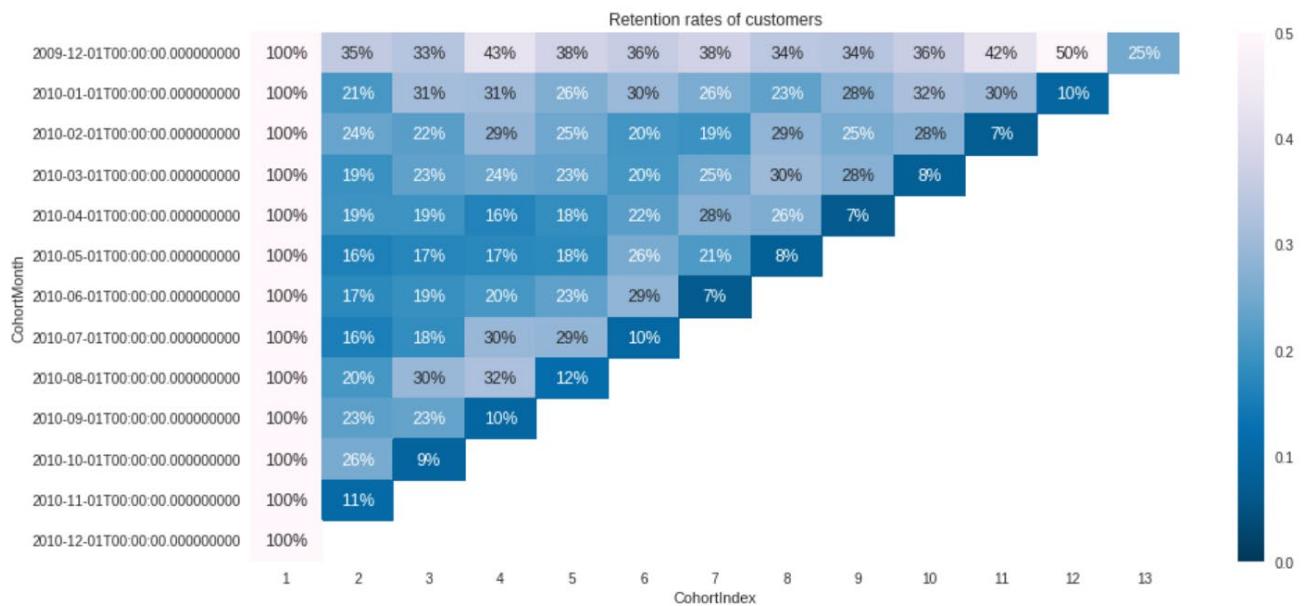
Hình 4.38. Số lượng khách hàng quay lại

- Kết quả phân tích tỉ lệ quay lại của khách hàng nâng cao [4].

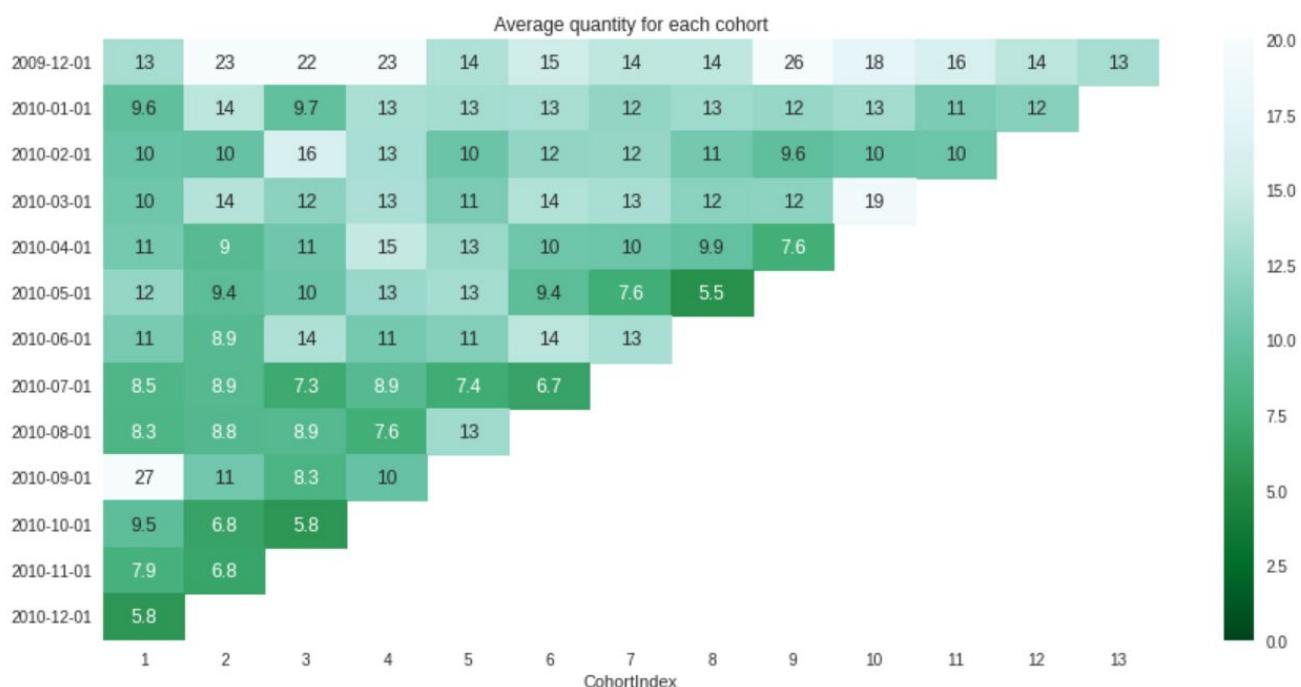
CohortIndex	1	2	3	4	5	6	7	8	9	10	11	12	13
CohortMonth													
2009-12-01	100.00000	35.30000	33.40000	42.50000	38.00000	35.90000	37.70000	34.20000	33.60000	36.20000	42.20000	49.50000	24.80000
2010-01-01	100.00000	20.60000	31.10000	30.50000	26.40000	30.00000	25.80000	23.00000	27.90000	31.90000	30.30000	9.90000	NaN
2010-02-01	100.00000	23.80000	22.50000	29.10000	24.60000	20.10000	19.30000	28.60000	25.40000	27.50000	7.20000	NaN	NaN
2010-03-01	100.00000	19.00000	23.00000	24.20000	23.30000	20.30000	24.60000	30.20000	27.50000	7.90000	NaN	NaN	NaN
2010-04-01	100.00000	19.40000	19.40000	16.30000	18.40000	22.40000	27.60000	26.20000	6.80000	NaN	NaN	NaN	NaN
2010-05-01	100.00000	15.70000	16.90000	17.30000	17.70000	25.60000	21.30000	7.90000	NaN	NaN	NaN	NaN	NaN
2010-06-01	100.00000	17.40000	18.90000	20.40000	23.00000	28.50000	6.70000	NaN	NaN	NaN	NaN	NaN	NaN
2010-07-01	100.00000	15.60000	18.30000	29.60000	29.00000	10.20000	NaN						
2010-08-01	100.00000	20.40000	29.60000	32.10000	11.70000	NaN							
2010-09-01	100.00000	22.60000	23.50000	9.90000	NaN								
2010-10-01	100.00000	25.70000	9.30000	NaN									
2010-11-01	100.00000	10.80000	NaN										
2010-12-01	100.00000	NaN											

Hình 4.39. Tỉ lệ khách hàng quay lại

- Kết quả tỉ lệ phần trăm quay lại của khách hàng được biểu diễn thông qua biểu đồ nhiệt [5].



Hình 4.40. Biểu đồ nhiệt thể hiện phần trăm khách hàng quay lại



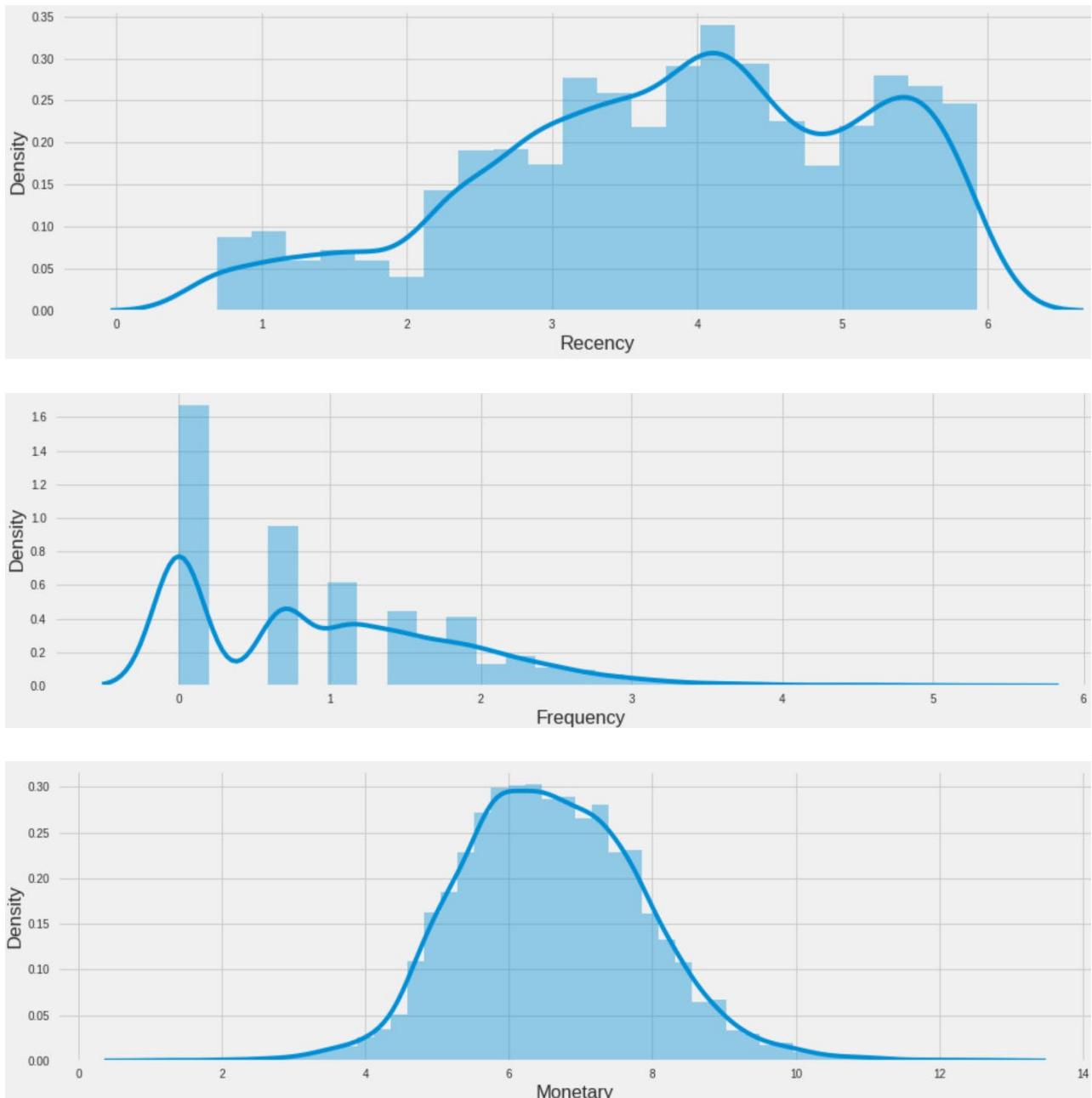
Hình 4.41. Biểu đồ nhiệt thể hiện số lượng sản phẩm đã bán tương ứng với mỗi tháng

- Kết quả phân bậc khách hàng theo hệ số RFM_Score [6].

	Recency	Frequency	Monetary	
	mean	mean	mean	count
General_Segment				
Bronze	99.30000	1.50000	473.00000	918
Diamond	8.50000	15.90000	9507.30000	474
Gold	46.50000	4.80000	2006.40000	790
Iron	220.10000	1.10000	217.10000	885
Platinum	23.70000	8.20000	3204.20000	406
Silver	72.80000	2.70000	929.00000	839

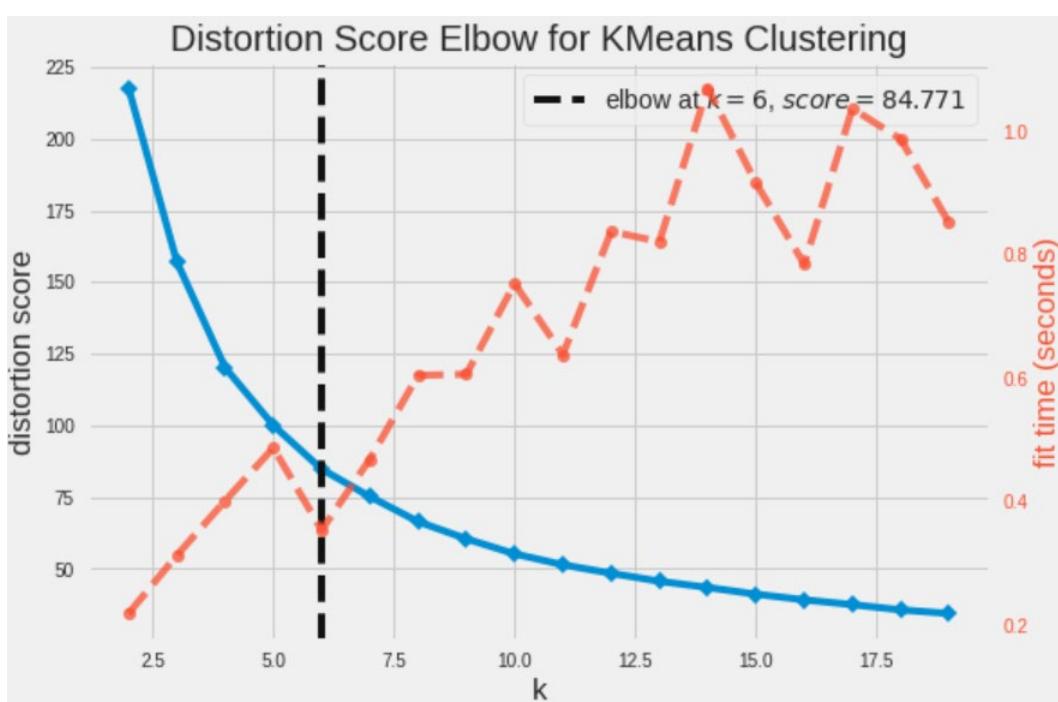
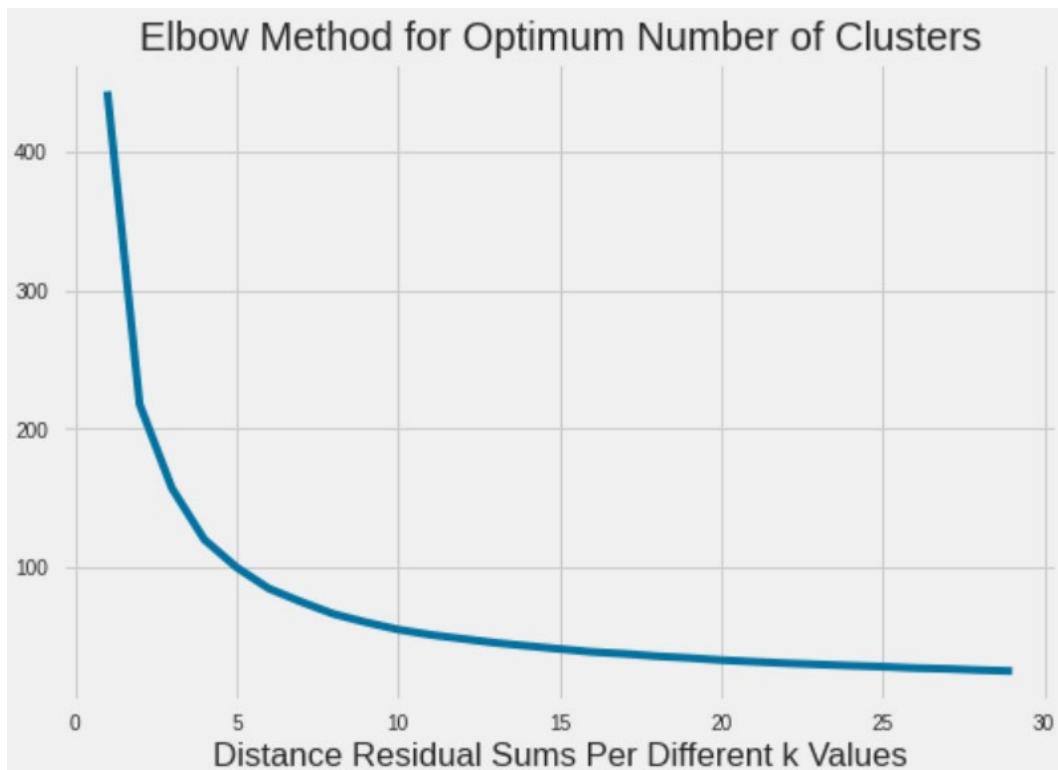
Hình 4.42. Kết quả phân bậc khách hàng

- Kết quả sau khi đã lọc và chuẩn hóa dữ liệu theo cơ số Logarit [7].



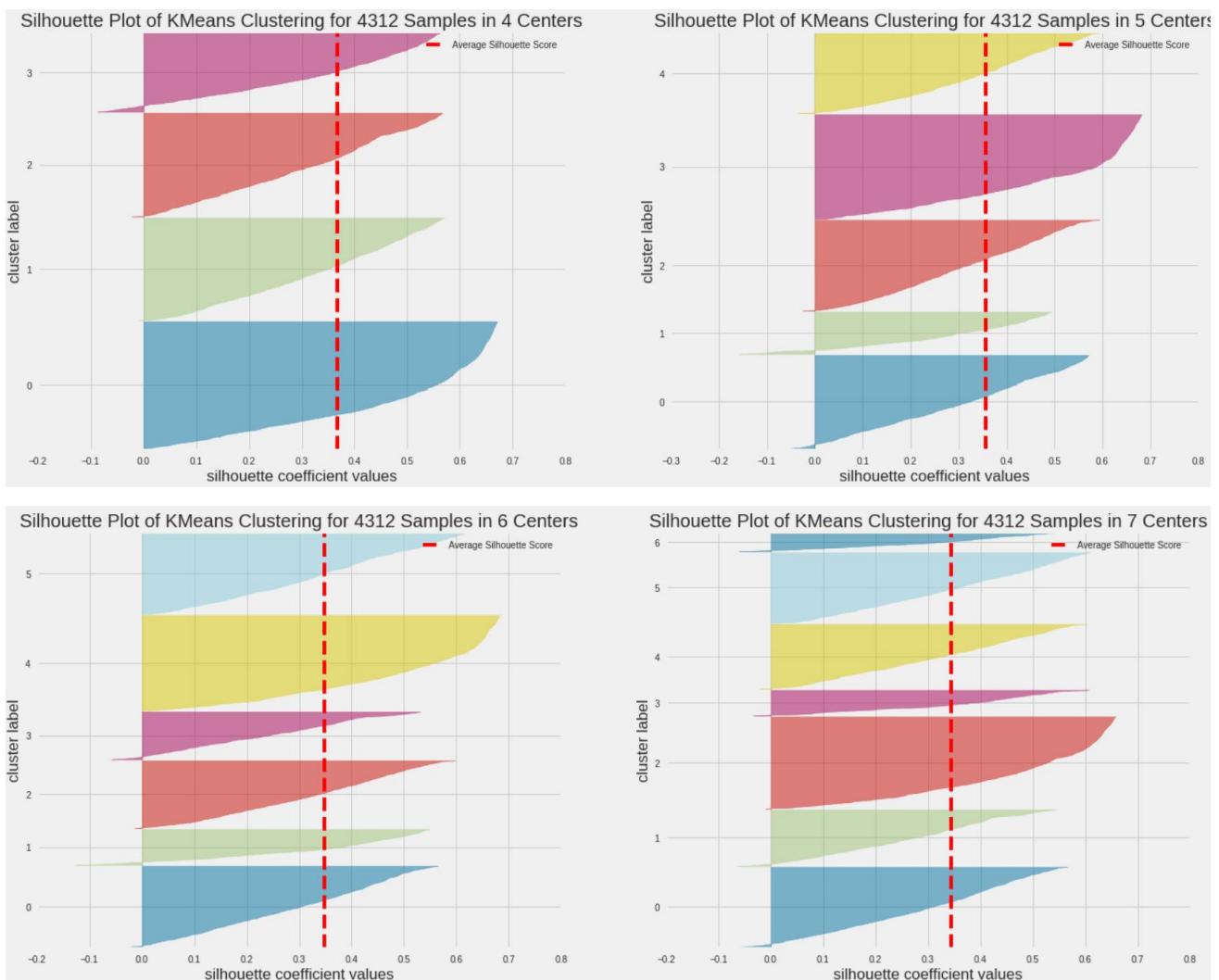
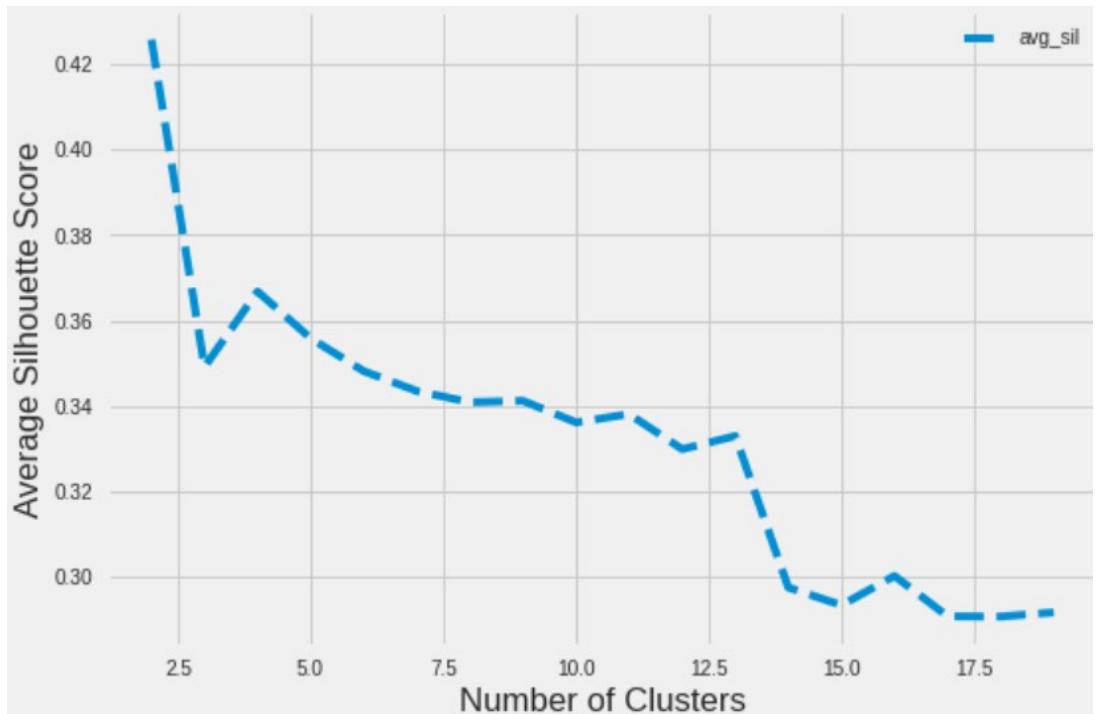
Hình 4.43. Dữ liệu sau khi đã được lọc và chuẩn hóa

- Kết quả của thuật toán Elbow tìm số cụm phù hợp [8].



Hình 4.44. Kết quả thuật toán Elbow tìm số cụm

- Kết quả thuật toán Silhouette để kiểm tra lại việc tìm số cụm [9].



Hình 4.45. Kết quả tìm số cụm của thuật toán Silhouette

- Kết quả so sánh thời gian thực thi giữa MiniBatchKmeans và K-Means [10].

```
CPU times: user 14 µs, sys: 2 µs, total: 16 µs
Wall time: 21.2 µs
```

Total cell runtime:

```
CPU times: user 547 ms, sys: 195 ms, total: 742 ms
Wall time: 397 ms
```

Hình 4.46. Thời gian thực thi của K-Means

```
CPU times: user 72 µs, sys: 0 ns, total: 72 µs
Wall time: 40.3 µs
```

Total cell runtime:

```
CPU times: user 75.1 ms, sys: 42.9 ms, total: 118 ms
Wall time: 64 ms
```

Hình 4.47. Thời gian thực thi của Mini Batch K-Means

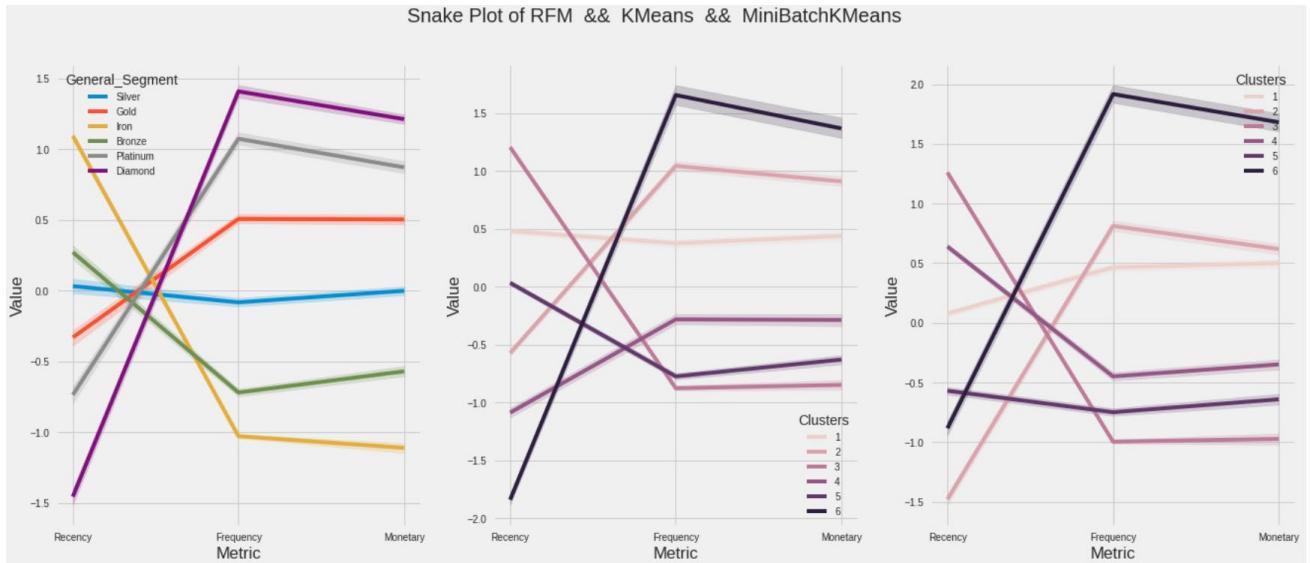
- Kết quả giá trị và kích thước của mỗi cluster được phân cụm dựa trên mô hình RFM (Phía trên là K-Means và phía dưới là Mini Batch K-Means) [11].

Clusters	Recency	Frequency	Monetary	count
	mean	mean	mean	
1	99.00000	4.00000	1813.00000	854
2	25.00000	8.00000	3182.00000	715
3	238.00000	1.00000	370.00000	1008
4	13.00000	2.00000	664.00000	504
5	53.00000	1.00000	430.00000	854
6	5.00000	17.00000	10345.00000	377

Clusters	Recency	Frequency	Monetary	count
	mean	mean	mean	
1	59.00000	4.00000	1909.00000	982
2	9.00000	6.00000	2218.00000	651
3	254.00000	1.00000	311.00000	803
4	121.00000	2.00000	674.00000	872
5	26.00000	1.00000	428.00000	645
6	20.00000	20.00000	12163.00000	359

Hình 4.48. Kích thước của mỗi cụm sau khi đã được phân cụm (K-Means và Mini Batch K-Means)

- Kết quả so sánh giữa các loại thuật toán thông qua biểu đồ SnakePlot [12].

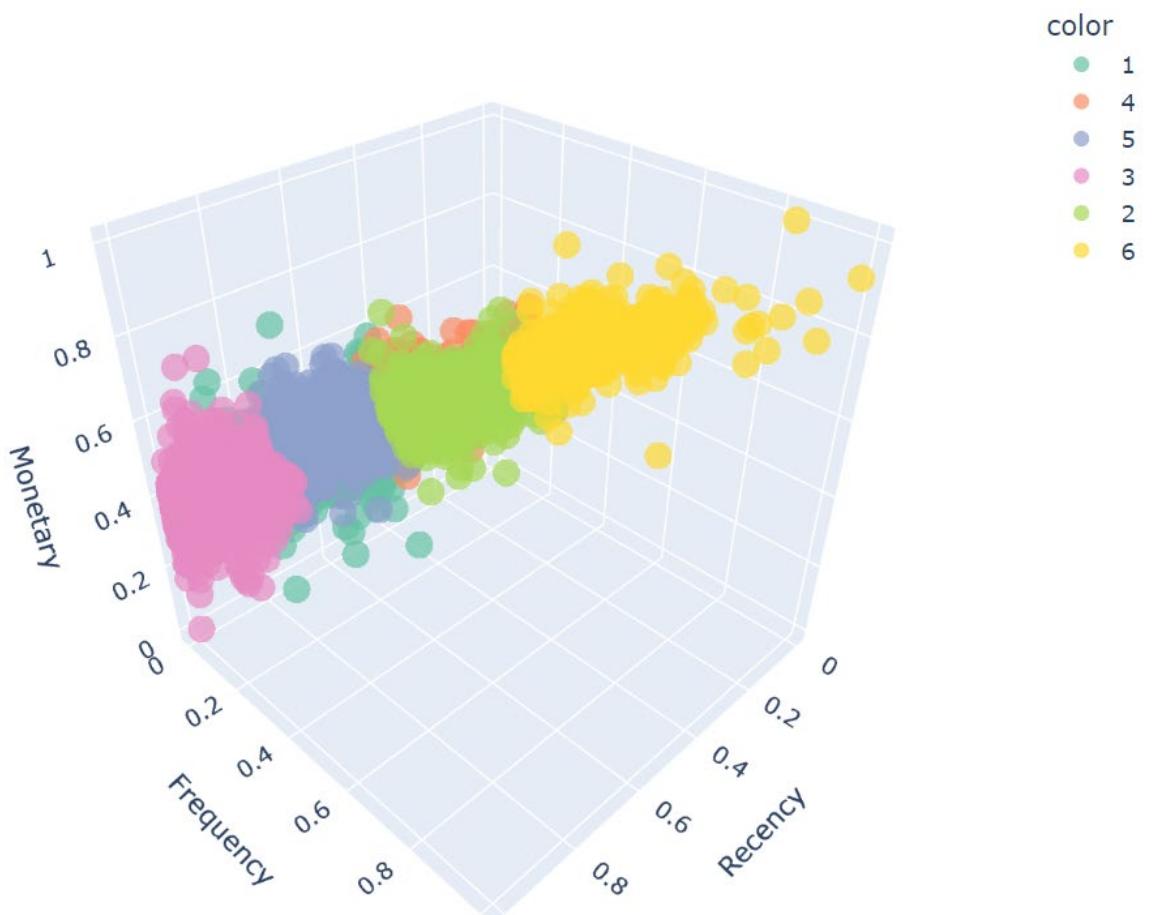


Hình 4.49. So sánh giữa các cách phân cụm khách hàng

- Kết quả thuật toán K-Means theo mô hình 3D [13].

Clusters Obtained using KMeans

Training time: 0.394s & Inertia: 84.771653

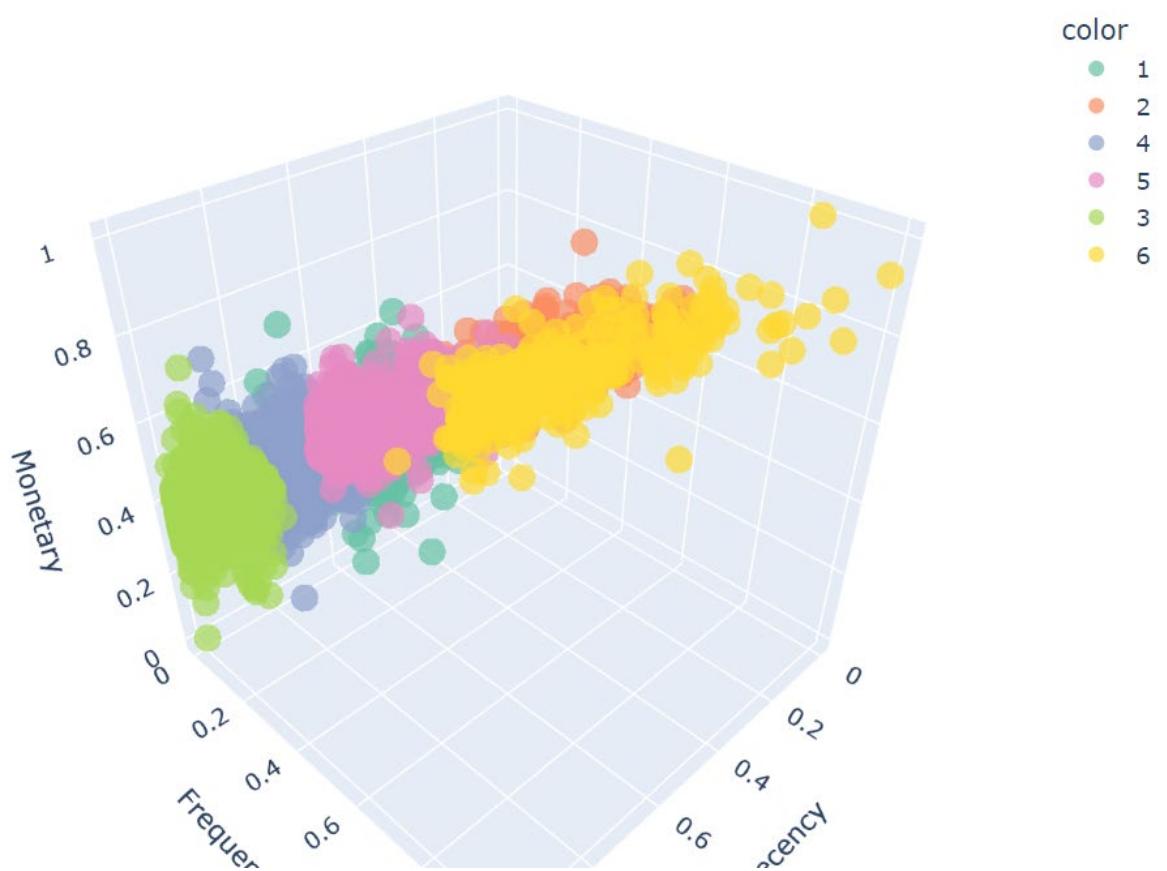


Hình 4.50. Mô hình 3D phân cụm khách hàng theo thuật toán K-Means

- Kết quả thuật toán Mini Batch Kmeans theo mô hình 3D [14].

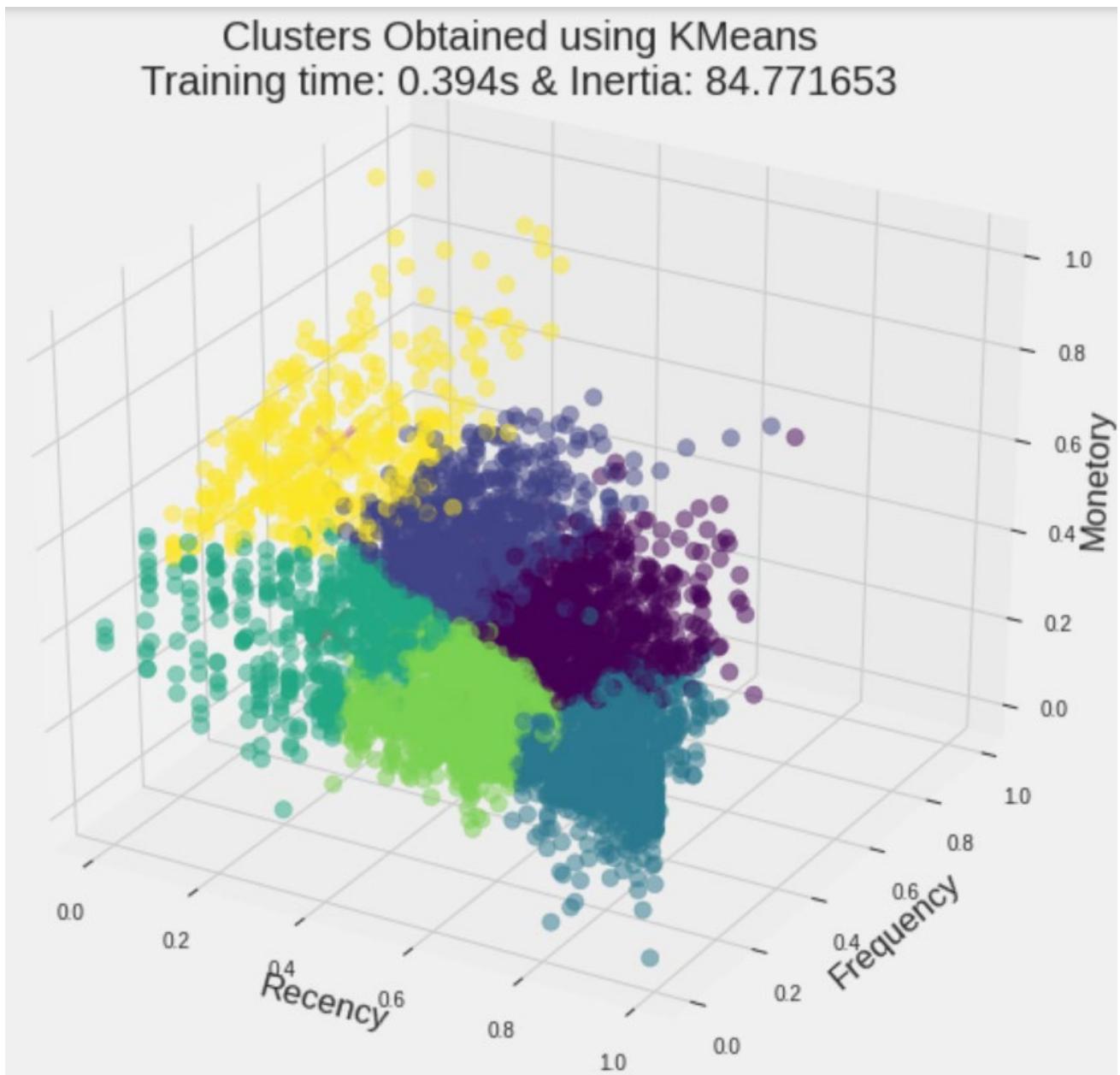
Clusters Obtained using Mini Batch Kmeans

Training time: 0.062s & Inertia: 98.335240



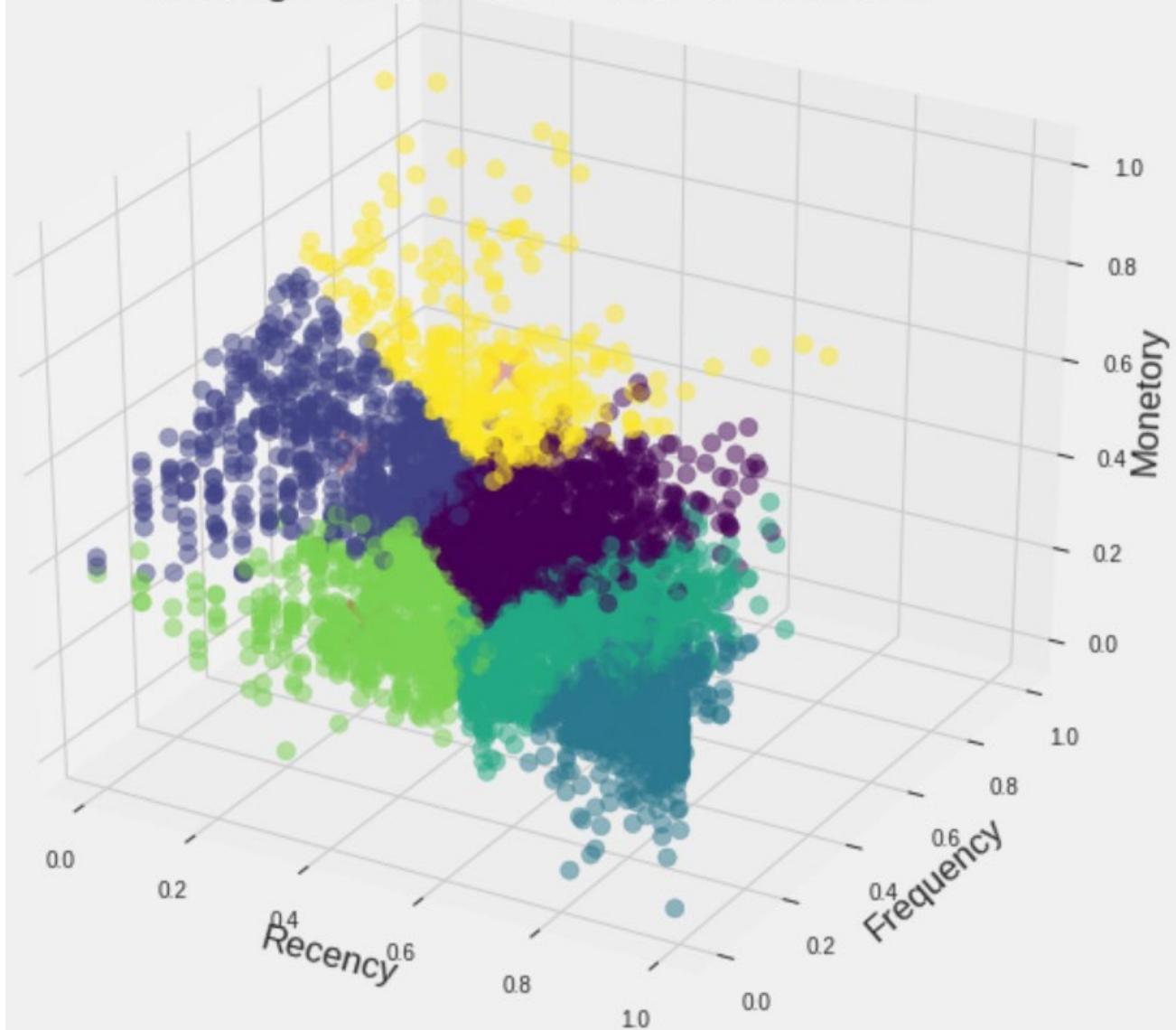
Hình 4.51. Mô hình 3D phân cụm khách hàng theo thuật toán Mini Batch K-Means

- So sánh kết quả của thuật toán K-Means và Mini Batch K-Means nâng cao [15].



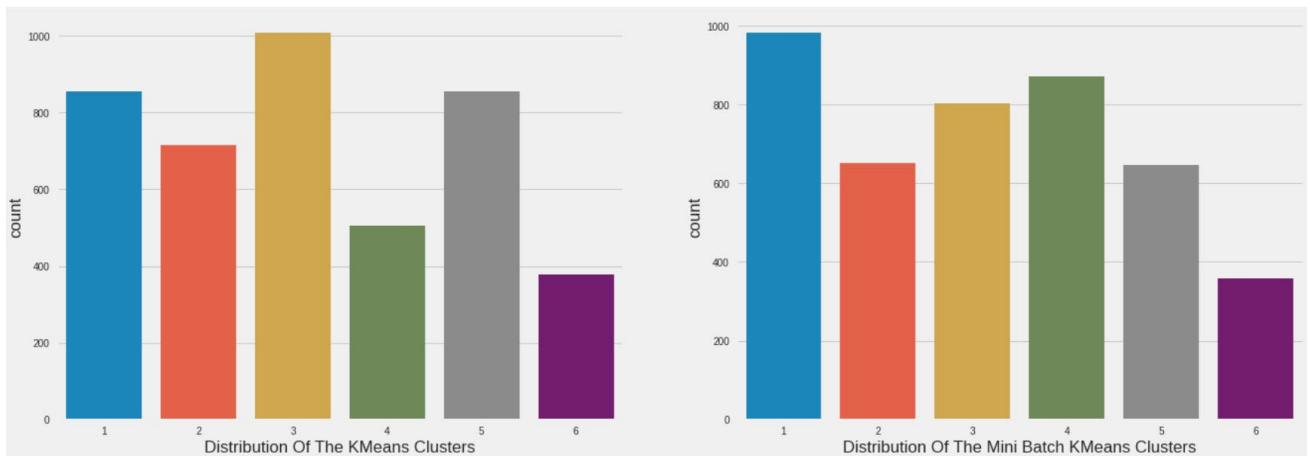
Hình 4.52. Phân cụm khách hàng theo thuật toán K-Means (Đáu X màu đỏ thể hiện tâm cụm)

Clusters Obtained using Mini Batch KMeans
Training time: 0.062s & Inertia: 98.335240



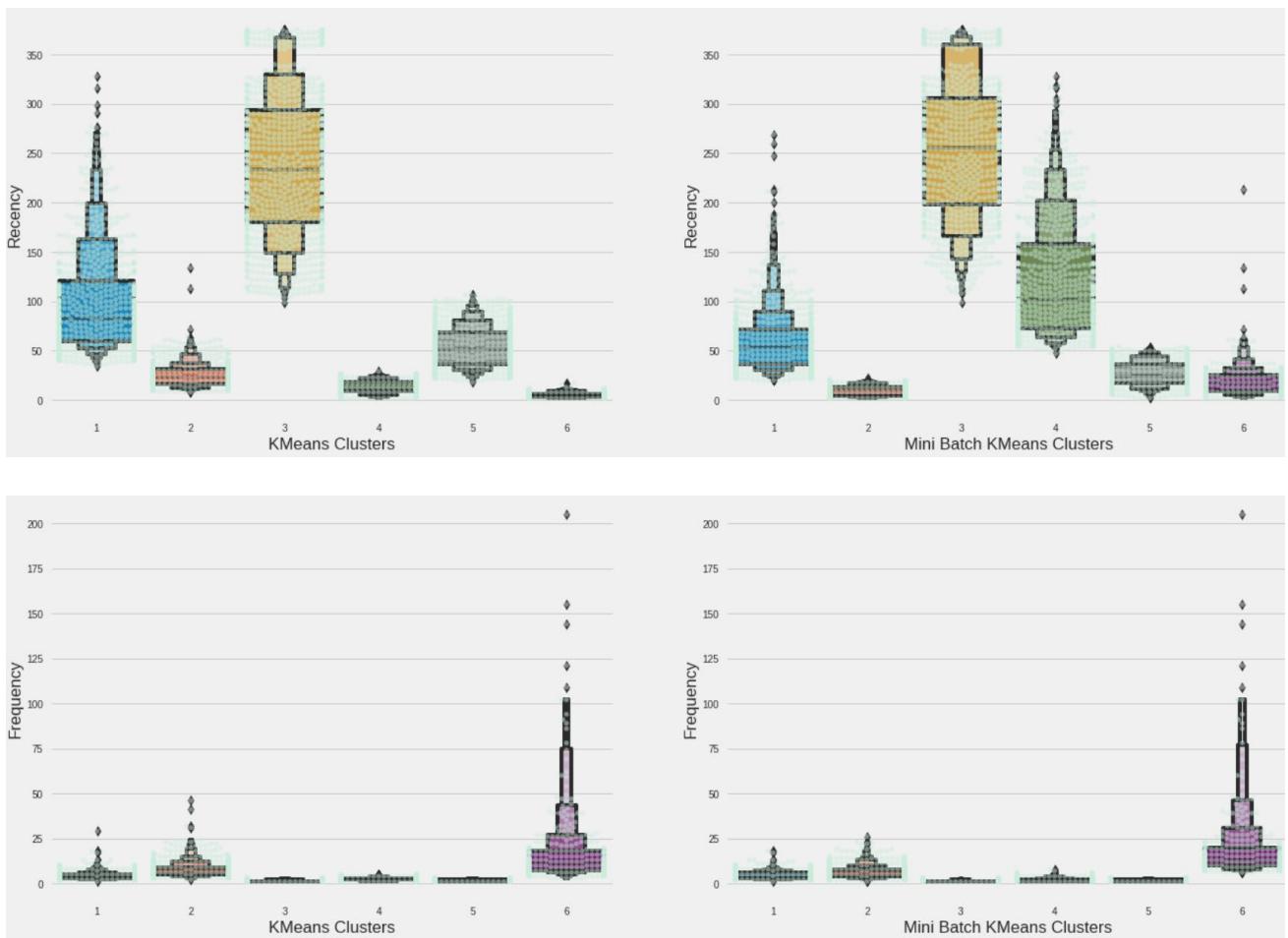
Hình 4.53. Phân cụm khách hàng theo thuật toán Mini Batch K-Means (Đầu X màu đỏ
thể hiện tâm cụm)

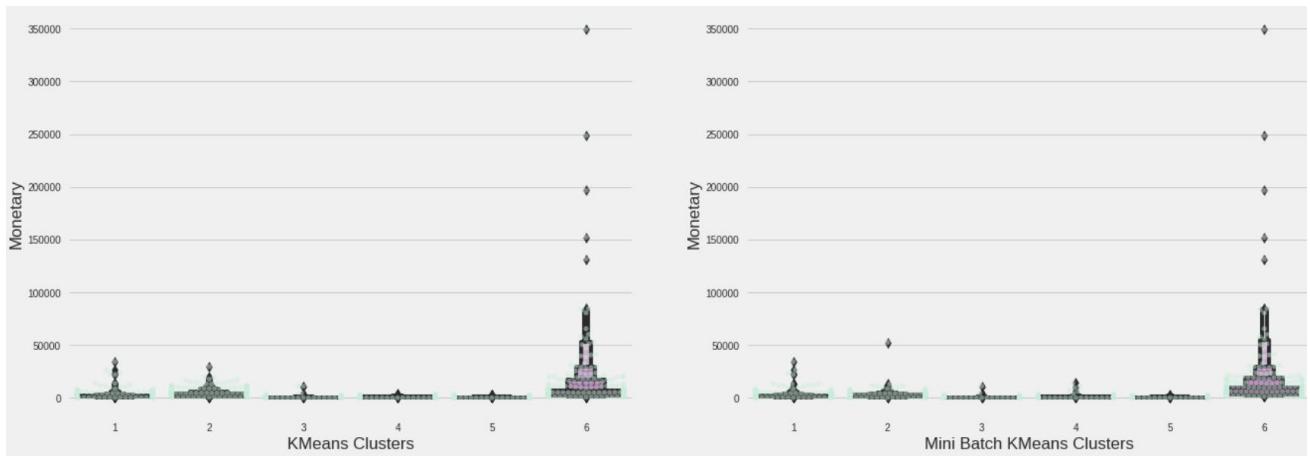
- Kết quả phân cụm khách hàng giữa các cluster của 2 thuật toán là KMeans và Mini Batch Kmeans [16].



Hình 4.54. Số lượng khách hàng của mỗi cụm

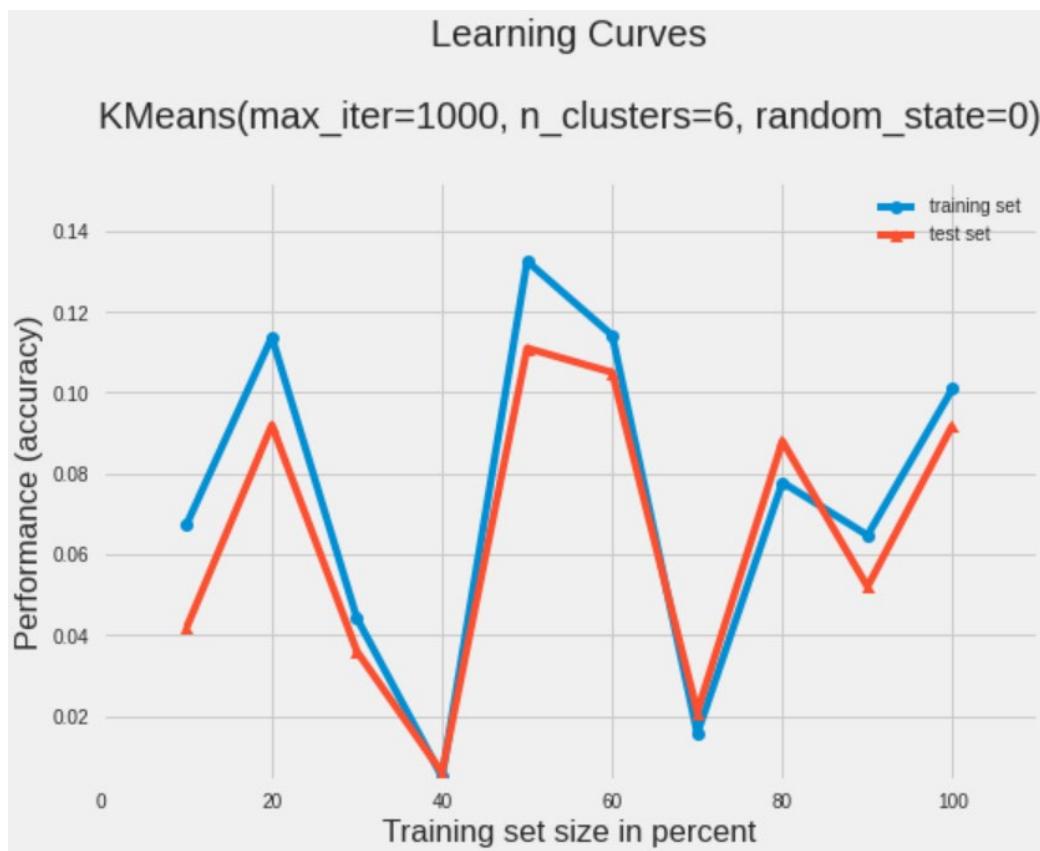
- So sánh kết quả theo mô hình RFM giữa K-Means và Mini Batch K-Means [17].



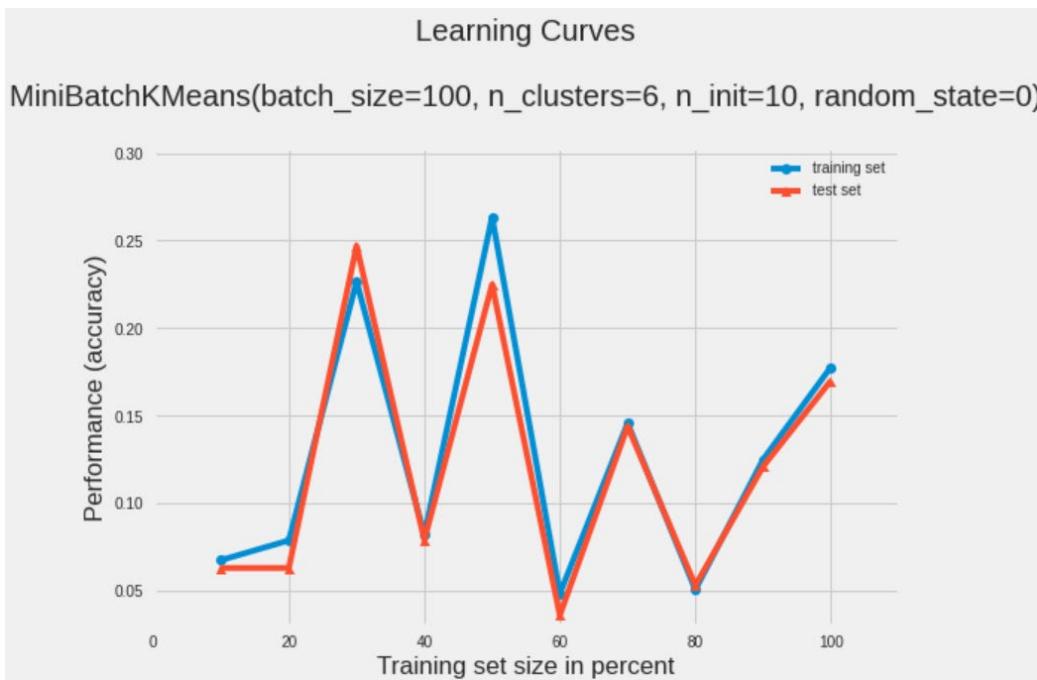


Hình 4.55. So sánh các chỉ số RFM sau khi đã phân cụm của 2 thuật toán

- Kết quả của Learning Curves (Đường cong học tập) thể hiện hiệu suất học tập cũng như mức độ chính xác của 2 thuật toán với tập dữ liệu đầu vào và tập kiểm thử ngẫu nhiên (4000 điểm dữ liệu) [18].



Hình 4.56. Learning Curves của K-Means



Hình 4.57. Learning Curves của Mini Batch K-Means

4.4 Kết luận

4.4.1 Đánh giá

- Là một thuật toán yêu cầu việc thử và sai (try and error) nhiều lần để đánh giá một cách chính xác và khách quan nhất. Ngoài ra trong các bước tính toán như trên, vị trí mà phần tử trung tâm không thay đổi sau một bước lặp nào đó thì tại đó hàm mất mát sẽ đạt giá trị nhỏ nhất (min). Nghĩa là tại vị trí đó sẽ là tối ưu nhất có thể.
- Tuy nhiên khi k (số lượng cụm) càng nhỏ so với m (số lượng dữ liệu) thì thuật toán sẽ dễ dàng cho ra các kết quả không tối ưu vì thuật toán phụ thuộc rất lớn vào việc chọn phần tử trung tâm ban đầu. Để khắc phục nhược điểm này thì nhóm đã khởi tạo k cụm bằng phương pháp “kmean++” được hỗ trợ trong Python, trong đó thay vì khởi tạo 1 lần duy nhất k cụm thì “kmean++” chỉ khởi tạo 1 tâm cụm đầu tiên, sau đó từ tâm cụm đầu tiên này tính toán khoảng cách nào xa nhất tới điểm khác trong tập dữ liệu và lựa chọn điểm đó là tâm cụm thứ 2 và cứ thế lặp lại cho đến khi đạt đủ tâm cụm đề ra. Việc này sẽ giúp ích rất nhiều cho việc phân loại khách hàng bởi nó giúp giảm số lần lặp để điều chỉnh tâm cụm cũng như thời gian chạy. Từ đó nâng cao hiệu suất của thuật toán.

- Dựa vào các kết quả trên, cho thấy rằng việc khởi tạo thuật toán, thực nghiệm cho ra một kết quả khá tốt và khi so sánh với các bài nghiên cứu khác thì thấy rằng kết quả của nhóm khá tương đồng. Nhóm lựa chọn thuật toán Mini Batch K-Means để so sánh với K-Means, thông qua quá trình thực nghiệm cho thấy kết quả của phương pháp Mini Batch K-Means chỉ đạt khoảng 80% so với phương pháp gốc là K-Means nhưng thời gian chạy lại nhanh hơn rất nhiều. Việc này rất có ý nghĩa trong thực tiễn bởi lẽ khi tập dữ liệu lớn đến mức mà K-Means phải mất hàng năm để tìm ra lời giải thì Mini Batch K-Means lại lựa chọn tối ưu hơn cả.

4.4.2 Ưu điểm

- Phân loại khách hàng là một bài toán không quá mới nhưng nó vẫn còn đang mang các giá trị thực tiễn trong cuộc sống hàng ngày. Trong đề tài này nhóm đã tìm hiểu về một cải tiến của K-Means là Mini Batch K-Means, tuy kết quả không được tốt và chính xác như K-Means nhưng Mini Batch K-Means lại mang giá trị thực tiễn hơn trong thực tế bởi dữ liệu khách hàng ngày càng gia tăng chứ không giảm nên việc dùng K-Means cho tập dữ liệu lớn sẽ tiêu tốn rất nhiều chi phí và thời gian. Thay vào đó có thể sử dụng Mini Batch K-Means với kết quả chấp nhận được do thời gian chạy là nhanh hơn rất nhiều so với K-Means.

- Trong thuật toán Mini Batch K-Means nhóm đã sử dụng phương pháp có tên là “Online K-Means Clustering” thông qua hàm “partial_fit” đã được sử dụng trong quá trình thực nghiệm ở trên để tăng thêm hiệu năng cho thuật toán này. Bởi mặc dù đã chia nhỏ dữ liệu ra cho nhiều lần chạy nhưng trong thực tế lượng dữ liệu đã được chia nhỏ này vẫn còn quá “lớn” nên khi huấn luyện thì toàn bộ lượng dữ liệu này sẽ được nạp lên RAM gây nghẽn RAM cũng như ảnh hưởng đến cái tài nguyên khác của máy tính, do đó hàm “partial_fit” sẽ cập nhật lại k cụm cho mỗi luồng dữ liệu vào. Từ đó giảm áp lực lên tài nguyên máy tính, thứ vốn dĩ có hạn.

4.4.3 Khuyết điểm

- Tuy tốc độ nhanh hơn nhiều so với K-Means nhưng Mini Batch K-Means lại cho thấy sự mất ổn định hơn so với K-Means bởi phương pháp này không có cái nhìn toàn cục về tập dữ liệu cũng như cấu trúc của dữ liệu đầu vào. Mini Batch K-Means sẽ phụ thuộc rất nhiều vào mỗi tập dữ liệu “nhỏ” đầu vào, sau mỗi lần chạy khi tập dữ liệu này thay đổi thì kết quả cũng sẽ bị thay đổi ít nhiều.

- Cả 2 thuật toán trên đều cho ra các kết quả khác nhau sau mỗi lần chạy, điều này là do việc khởi tạo các trung tâm cụm ban đầu là ngẫu nhiên. Từ đó dẫn đến việc phân loại khách hàng cũng chỉ mang tính chất tương đối. Do đó để có thể đạt được kết quả tối ưu nhất thì phải trải qua thực nghiệm với nhiều bộ dữ liệu khách hàng khác nhau mới có thể tìm ra được một phương pháp hợp lý và chính xác nhất.
- Độ lớn của tập cơ sở dữ liệu được nhóm sử dụng còn khá hạn chế, chỉ hơn 500.000 dòng với hơn 4000 khách hàng và dữ liệu chỉ ở dạng số. Điều này có thể khiến kết quả không đúng trong thực tế khi dữ liệu hỗn hợp, thay đổi liên tục và tăng lên gấp nhiều lần.
- Ngoài ra dữ liệu trước khi xử lý cần nạp lên bộ nhớ tạm là RAM nên thuật toán K-Means sẽ không chạy được đối với các tập dữ liệu rất lớn hoặc luồng dữ liệu (stream data), có thể thay thế bằng các phương pháp như Mini Batch K-Means giúp chia nhỏ tập dữ liệu đầu vào hay Online Learning K-Means sẽ cập nhật K tâm cụm đối với mỗi đối tượng dữ liệu đầu vào như nhóm đã nêu ở trên.

4.4.4 Hướng phát triển

- Việc lựa chọn thuật toán nào còn tùy thuộc vào mục đích cuối cùng là kết quả hay thời gian chạy của từng người. Tuy nhiên có thể cải thiện độ ổn định của 2 thuật toán trên bằng cách tìm ra cách tối ưu để khởi tạo tâm cụm cũng như giảm thiểu độ “nhạy cảm” của thuật toán với tập có chứa dữ liệu nhiễu, dữ liệu hỗn hợp từ đó nâng cao hiệu năng cũng như thời gian chạy.
- Ngoài ra còn có thể sử dụng kết hợp với các phương pháp như RFM như nhóm đã đề cập ở trên hay PCA (Principal Component Analysis) để giảm số chiều dữ liệu đầu vào nhưng vẫn giữ được tối đa lượng thông tin quan trọng cần thiết từ đó thu được kết quả tốt nhất có thể. Hay sử dụng các phương pháp phân cụm mới hơn như phân cụm nửa giám sát trong đó sử dụng các thông tin bổ trợ như tập các cặp ràng buộc, một lượng nhỏ dữ liệu được dán nhãn,... để hướng dẫn cho quá trình phân cụm của thuật toán. Các thuật toán phân cụm nửa giám sát phổ biến và thường gặp như Seeded K-Means, Constrained K-Means,...
- Với những hạn chế về mặt kiến thức, trình độ cũng như thời gian thực hiện, nhóm chúng em xin kết thúc đề tài nghiên cứu này ở đây và hy vọng đóng góp được một phần nhỏ về bài toán phân loại khách hàng nói riêng cũng như khoa học máy tính nói chung. Với những định hướng đưa ra như đã nêu ở trên, nhóm chúng em hy vọng sẽ có thể thực hiện được để giúp

cho việc phân loại khách hàng ở các công ty, doanh nghiệp ngày càng dễ dàng và thuận lợi hơn cũng như thuật toán K-Means có thể loại bỏ các điểm yếu để trở nên hiệu quả hơn trong các bài toán. Từ đó có thể tiết kiệm được các nguồn tài nguyên máy tính có hạn.

TÀI LIỆU THAM KHẢO

- [1] S.J.Russell and P.Norvig, “Artificial Intelligence : A Modern Approach”, Prentice Hall Seriesin Artificial Intelligence, Pearson Education, 1995.
- [2] A.Hareendran, V.Chandra, “Artifical Intelligence and Machine Learning”, PHI Learning, 1 January 2014.
- [3] P.N.Tan, M.Steinbach and V.Kumar, “Introduction to Data Mining”, Pearson Addison-Wesley, 2006.
- [4] Vũ Minh Đông, “Một số phương pháp phân cụm dữ liệu”, Đại học Dân Lập Hải Phòng, 2010.
- [5] Charu C.Aggarwal, “Data Mining : The Textbook”, Springer, 13 April 2015.
- [6] A.K.Jain and R.C.Dubes, “Algorithms for Clustering Data”, Prentice Hall Advanced Reference Series, Prentice Hall, March 1988.
- [7] Rui Zhao, “CVM Model of Customer Purchasing Behavior Based on Clustering Analysis”, Advances in Economics, Business and Management Research, Atlantis Press International B.V, 15 December 2021.
- [8] D.Chen, S.L.Sain and K.Guo, “Data mining for the online retail industry: A case study of RFM model-based customer segmentation using data mining”, Database Marketing & Customer Strategy Management, Macmillan Publishers Ltd, 27 August 2012.
- [9] T.Kansal, S.Bahuguna, V.Singh, T.Choudhury, “Customer Segmentation using K-means Clustering”, 2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS), Belgaum, India, 21-22 December 2018.
- [10] Junjie Wu, “Advances in K-means Clustering: A Data Mining Thinking”, Springer Theses, Springer Science & Business Media, 2012.
- [11] K.Peng, V.C.M.Leung, Q. Huang, “Clustering Approach Based on Mini Batch Kmeans for Intrusion Detection System Over Big Data”, IEEE Access (Volume 6), IEEE, 28 February 2018.
- [12] S.R.Fitriyani, H.Murfi, “The K-Means with Mini Batch Algorithm for Topics Detection on Online News”, 2016 Fourth International Conference on Information and Communication Technologies (ICoICT), Bandung, Indonesia, 2016.