

Bộ giáo dục và đào tạo  
ĐẠI HỌC QUỐC GIA TP HCM  
TRƯỜNG ĐẠI HỌC BÁCH KHOA TP HCM



## Báo cáo bài tập lớn

Môn: Máy học cơ bản và ứng dụng cơ bản

**Đề tài: Ứng dụng của Kmeans phân loại tập khách hàng**

**LỚP:L01**

**NHÓM:4**

**GVHD: Nguyễn Khánh Lợi**

STT	Họ và tên	MSSV	Đóng góp
1	Phan Nguyễn Xuân Hoàng	2111255	100%
2	<b>Đỗ Thanh Hà</b>	2111117	100%
3	Ngô Thời Lân	2113891	100%
4	Phạm Tú Tân	2114731	100%
5	Quách Tấn Kiệt	2111609	100%

## *Lời mở đầu*

*Nhóm xin gửi lời cảm ơn chân thành và sự tri ân sâu sắc đối với thầy Nguyễn Khánh Lợi, Giảng viên trường Đại học Bách Khoa – Đại học Quốc gia Thành phố Hồ Chí Minh, đã tạo điều kiện cho chúng em có nhiều thời gian cho môn học Máy học và ứng dụng cơ bản. Và đồng thời chúng em cũng xin chân thành cảm ơn thầy đã nhiệt tình hướng dẫn hướng dẫn giúp nhóm em hoàn thành tốt Bài tập lớn này.*

*Trong quá trình học tập, cũng như là trong quá trình làm bài báo cáo Bài tập lớn, do điều kiện khó khăn và thời gian gấp rút, khó tránh khỏi sai sót, rất mong Thầy có thể thông cảm. Đồng thời do trình độ lý luận cũng như kinh nghiệm thực tiễn còn hạn chế nên bài báo cáo không thể tránh khỏi những thiếu sót, em rất mong nhận được ý kiến đóng góp từ Thầy để em học thêm được nhiều kinh nghiệm và sẽ hoàn thành tốt những Đồ án, Luận văn tốt nghiệp trong tương lai.*

*Nhóm xin chân thành cảm ơn Thầy!*

*Chúc Thầy sức khỏe và thành đạt.*

# MỤC LỤC

<b>1.GIỚI THIỆU VỀ PHÂN KHÚC KHÁCH HÀNG.....</b>	<b>3</b>
<b>2.CƠ SỞ LÝ THUYẾT VỀ THUẬT TOÁN K-MEANS .....</b>	<b>4</b>
2.1 Tổng quan về K-Means .....	4
2.2 Nguyên lý hoạt động của thuật toán.....	5
2.2.1 Phân cụm K-means .....	5
2.3 Phương pháp xác định số lượng cụm tối ưu .....	6
2.3.1 Phương pháp Elbow: .....	6
2.3.2 Silhouette Score: .....	6
2.3.3 Gap Statistic: .....	7
<b>3. PYSPARK VÀ ỨNG DỤNG TRONG PHÂN TÍCH DỮ LIỆU LỚN .....</b>	<b>8</b>
3.1 Giới thiệu về PySpark.....	8
3.2 Kiến trúc và nguyên lý hoạt động của PySpark.....	9
<b>4.KẾT HỢP K-MEANS VỚI PYSPARK CHO PHÂN KHÚC KHÁCH HÀNG.....</b>	<b>12</b>
<b>5.BIG DATA ONLINE RETAIL WITH PYSPARK .....</b>	<b>14</b>
5.1 Recency.....	20
5.2 Frequency.....	21
5.3 Monetary .....	23
5.4 Kết quả :.....	28
<b>6.ỨNG DỤNG THỰC TIỄN .....</b>	<b>32</b>
<b>7.KẾT LUẬN .....</b>	<b>34</b>
<b>8. Tài liệu tham khảo .....</b>	<b>35</b>

# 1. GIỚI THIỆU VỀ PHÂN KHÚC KHÁCH HÀNG

Trong môi trường kinh doanh hiện đại, việc hiểu rõ nhu cầu và hành vi của khách hàng đóng vai trò then chốt trong chiến lược marketing của doanh nghiệp. Tuy nhiên, mỗi khách hàng có những đặc điểm và hành vi tiêu dùng khác nhau, do đó không thể áp dụng một chiến lược tiếp cận chung cho tất cả. Đây là lý do vì sao phân khúc khách hàng (Customer Segmentation) trở thành một công cụ quan trọng.

Phân khúc khách hàng là quá trình chia tập khách hàng thành các nhóm nhỏ dựa trên các đặc điểm tương đồng như độ tuổi, thu nhập, thói quen mua sắm hay mức độ tương tác với thương hiệu. Nhờ đó, doanh nghiệp có thể:

- Cá nhân hóa chiến lược tiếp thị và chăm sóc khách hàng.
- Dự đoán hành vi tiêu dùng trong tương lai.
- Phân bổ ngân sách marketing hiệu quả hơn.

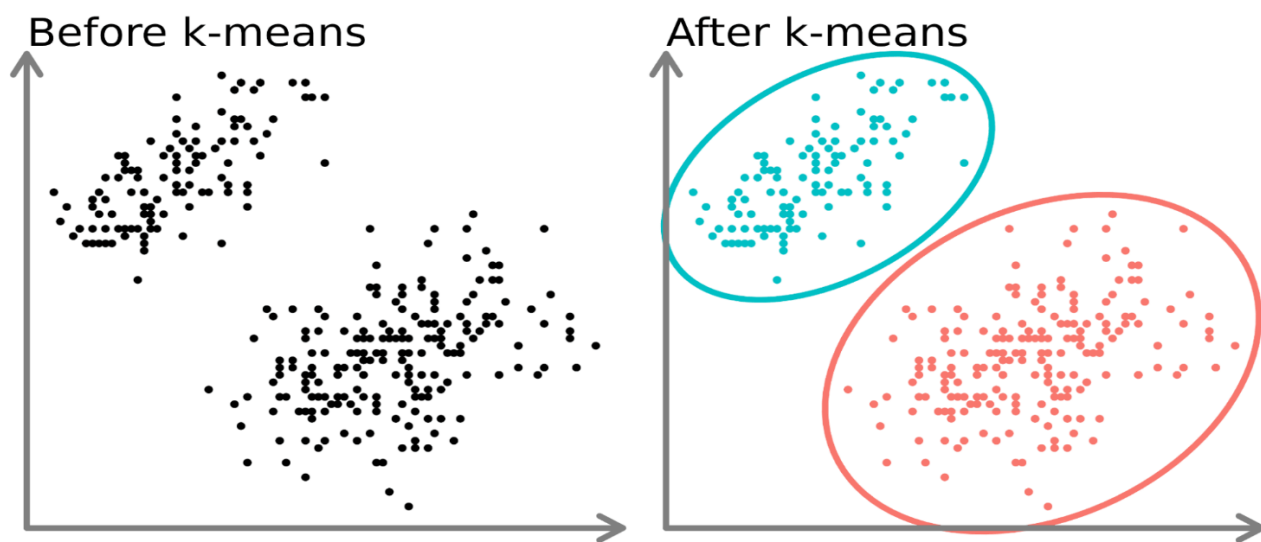
Trong nghiên cứu này, chúng tôi tập trung vào dữ liệu khách hàng của một trung tâm thương mại bao gồm các đặc trưng như tuổi, thu nhập hàng năm, và điểm chi tiêu. Tuy nhiên, dữ liệu này không có nhãn phân loại khách hàng cụ thể (ví dụ: khách hàng tiềm năng, trung thành, hay ít tương tác), vì vậy không thể áp dụng các thuật toán học có giám sát.

Để giải quyết bài toán này, chúng tôi sử dụng thuật toán K-Means – một kỹ thuật học không giám sát phổ biến trong phân cụm dữ liệu. K-Means cho phép tự động chia khách hàng thành các nhóm (cụm) dựa trên sự tương đồng về đặc trưng, từ đó giúp doanh nghiệp xây dựng chiến lược tiếp cận phù hợp cho từng nhóm.

## 2. CƠ SỞ LÝ THUYẾT VỀ THUẬT TOÁN K-MEANS

### 2.1 Tổng quan về K-Means

Phân cụm K-means là một kỹ thuật được sử dụng để sắp xếp dữ liệu chưa có nhóm hoặc chưa có nhãn thành các nhóm khác nhau dựa trên sự tương đồng của chúng. Ví dụ, một cửa hàng trực tuyến sử dụng K-means để nhóm khách hàng dựa trên tần suất mua hàng và mức chi tiêu, tạo ra các phân đoạn như "Khách hàng có chi tiêu tiết kiệm", "Khách hàng thường xuyên" và "Khách hàng có nhu cầu chi tiêu lớn" để tiến hành các phương thức tiếp thị nhằm vào các tệp khách hàng cụ thể.



Thuật ngữ "K-means" được James MacQueen sử dụng lần đầu tiên vào năm 1967, mặc dù ý tưởng này có từ năm 1956 bởi Hugo Steinhaus. Thuật toán chuẩn được Stuart Lloyd tại Bell Labs đề xuất lần đầu vào năm 1957 như một kỹ thuật cho điều chế mã xung, nhưng mãi đến năm 1982 mới được công bố dưới dạng bài báo tạp chí. Năm 1965, Edward W. Forgy công bố một phương pháp gần giống, vì vậy thuật toán này đôi khi được gọi là thuật toán Lloyd–Forgy.

Do tính chất đơn giản và dễ hiểu, dễ triển khai nên thuật toán K-means được sử dụng khá rộng rãi trong lĩnh vực máy học, đặc biệt với các vấn đề về dữ liệu có dạng số. Sự hiệu quả trong tính toán và khả năng mở rộng với nhiều biến thể khác nhau cũng là một trong số ít ưu điểm của phương pháp giải thuật này.

## 2.2 Nguyên lý hoạt động của thuật toán

Chúng ta được cung cấp một tập dữ liệu gồm các mục, mỗi mục có các đặc trưng và giá trị cụ thể cho những đặc trưng này (giống như một vector). Điều này đặt ra nhiệm vụ là phân loại các mục này thành các nhóm. Để thực hiện điều này, chúng ta sẽ sử dụng thuật toán K-means. Chữ ‘K’ trong tên thuật toán đại diện cho số lượng nhóm hoặc cụm mà chúng ta muốn phân loại các mục vào.

### 2.2.1 Phân cụm K-means

Thuật toán sẽ phân loại các mục thành **k** nhóm hoặc cụm dựa trên sự tương đồng. Để tính toán sự tương đồng này, chúng ta sẽ sử dụng **khoảng cách Euclidean** làm thước đo. Thuật toán hoạt động như sau:

1. Đầu tiên, chúng ta khởi tạo ngẫu nhiên **k** điểm, được gọi là **trung bình** hoặc **centroid cụm**.

$$c_i = \arg \min_j \|x_j - \mu_j\|_2^2$$

2. Chúng ta phân loại mỗi mục vào **trung bình gần nhất**, sau đó cập nhật tọa độ của trung bình, là giá trị trung bình của các mục đã được phân loại vào cụm đó cho đến thời điểm hiện tại.

$$\mu_j = \frac{\sum_{i=1}^N 1(c_i = j)x_i}{\sum_{i=1}^N 1(c_i = j)}$$

3. Chúng ta lặp lại quá trình cho đến khi toàn bộ các điểm dữ liệu được phân về đúng cụm hoặc số lượt cập nhật tâm chạm ngưỡng.

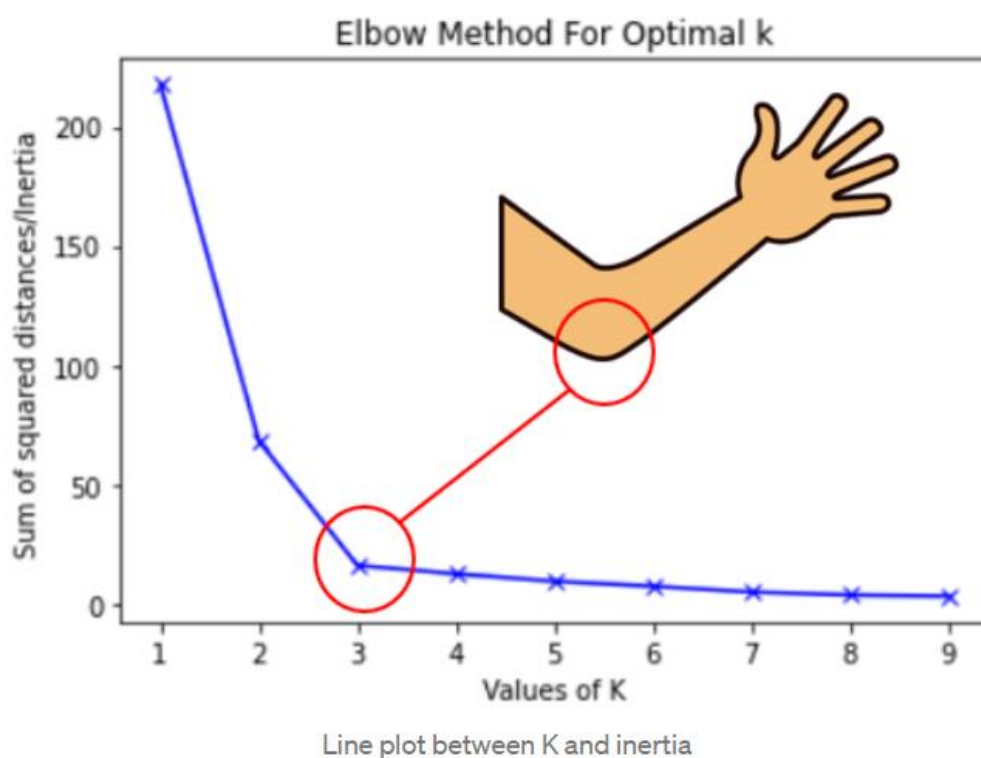
Các “điểm” được đề cập ở trên được gọi là **trung bình** vì chúng là giá trị trung bình của các mục được phân loại trong cụm đó. Để khởi tạo các trung bình này, chúng ta có nhiều lựa chọn. Một trong những phương pháp thường được sử dụng là khởi tạo trung bình tại các mục ngẫu nhiên trong tập dữ liệu. Một phương pháp khác là khởi tạo trung bình tại các giá trị ngẫu nhiên trong khoảng giới hạn của tập dữ liệu. Ví dụ, đối với một đặc trưng **x** có giá trị trong khoảng [0,3], chúng ta sẽ khởi tạo trung bình với các giá trị cho **x** nằm trong [0,3].

## 2.3 Phương pháp xác định số lượng cụm tối ưu

Trong quá trình sử dụng thuật toán để giải quyết các vấn đề về số liệu, việc xác định số lượng cụm/ nhóm tối ưu được xem là thách thức lớn nhất. Việc xác định số lượng K chính xác sẽ giúp tiết kiệm quá trình tính toán, tối ưu thời gian thực hiện các phép tính và tiết kiệm năng lượng. Một số phương pháp phổ biến thường được sử dụng bao gồm:

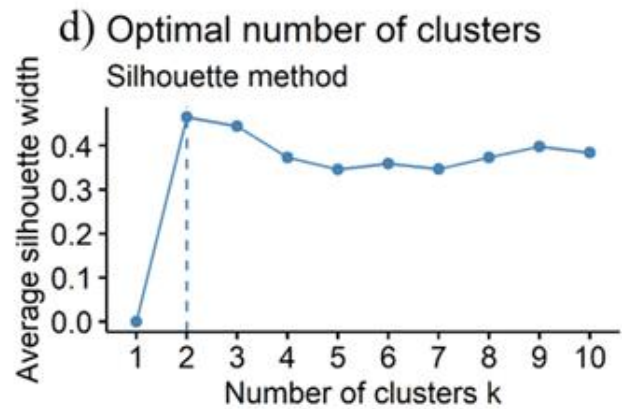
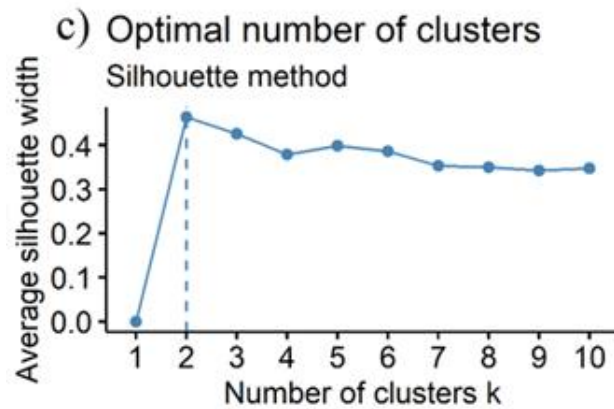
### 2.3.1 Phương pháp Elbow:

Vẽ đồ thị giữa số lượng cụm và tổng bình phương khoảng cách trong cụm (WCSS - Within-Cluster Sum of Squares). Thực hiện phân cụm với số lượng điểm từ 1 đến 10. Tính toán WCSS cho mỗi giá trị và từ đó xác định điểm “Khủy tay”. Điểm này trên đồ thị được xem là số lượng cụm tối ưu vì việc tăng thêm cụm sau điểm này không ảnh hưởng đáng kể đến kết quả phân cụm.



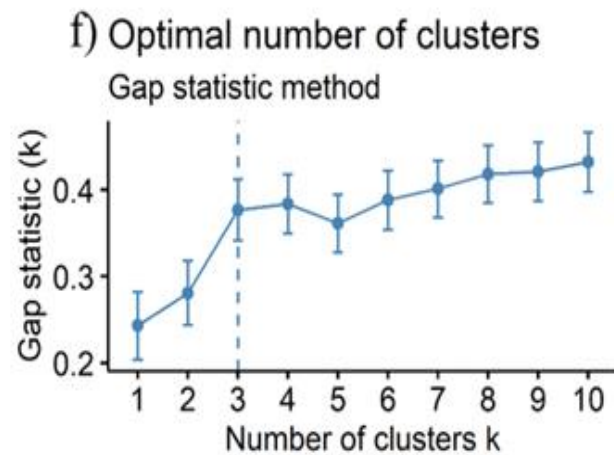
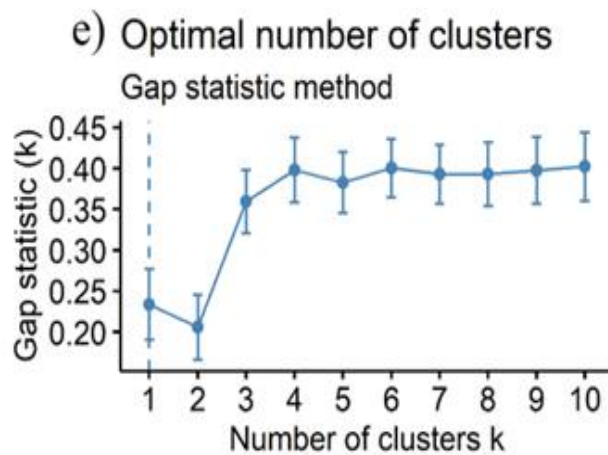
### 2.3.2 Silhouette Score:

Đo lường mức độ tương đồng của một đối tượng với cụm của nó so với các cụm khác. Giá trị Silhouette cao hơn cho biết phân cụm tốt hơn. Số lượng cụm tối ưu là giá trị K mà trung bình của điểm Silhouette trên toàn bộ dữ liệu đạt giá trị cao nhất



### 2.3.3 Gap Statistic:

Phương pháp này so sánh hiệu suất của thuật toán phân cụm trên dữ liệu thực tế với hiệu suất trên một tập dữ liệu ngẫu nhiên (null reference distribution). Thống kê khoảng cách (gap statistic) là sự khác biệt giữa WCSS quan sát được và WCSS mong đợi dưới phân phối null. Số lượng cụm tối ưu là giá trị K mà gap statistic đạt giá trị lớn nhất





### 3. PYSPARK VÀ ỨNG DỤNG TRONG PHÂN TÍCH DỮ LIỆU LỚN

#### 3.1 Giới thiệu về PySpark

PySpark là một giao diện lập trình ứng dụng (API) kết hợp khả năng ứng dụng dễ dàng của Python với sức mạnh của Apache Spark, đây được xem là một công cụ mạnh mẽ giúp thực hiện các tác vụ xử lý big data và thời gian thực trong một môi trường phân tán đối với những người quen thuộc với Python.

Trong đó Apache Spark là một hệ thống xử lý phân tán nguồn mở thường được sử dụng rộng rãi trong lĩnh vực BigData. Bằng cách sử dụng khả năng ghi vào bộ nhớ đệm nằm trong bộ nhớ và thực thi truy vấn tối ưu hóa nhằm phân tích nhanh dữ liệu có kích thước bất kỳ.



Ngoài ra tương tự như Apache Spark, PySpark cũng đồng thời cung cấp đầy đủ hỗ trợ cho tất cả các tính năng chính của Spark, bao gồm Spark SQL cho truy vấn dữ liệu dạng SQL, DataFrames cho xử lý dữ liệu cấu trúc, Structured Streaming cho xử lý dữ liệu luồng cấu trúc, Machine Learning cho các ứng dụng, và Spark Core cho các tác vụ xử lý dữ liệu cơ bản.

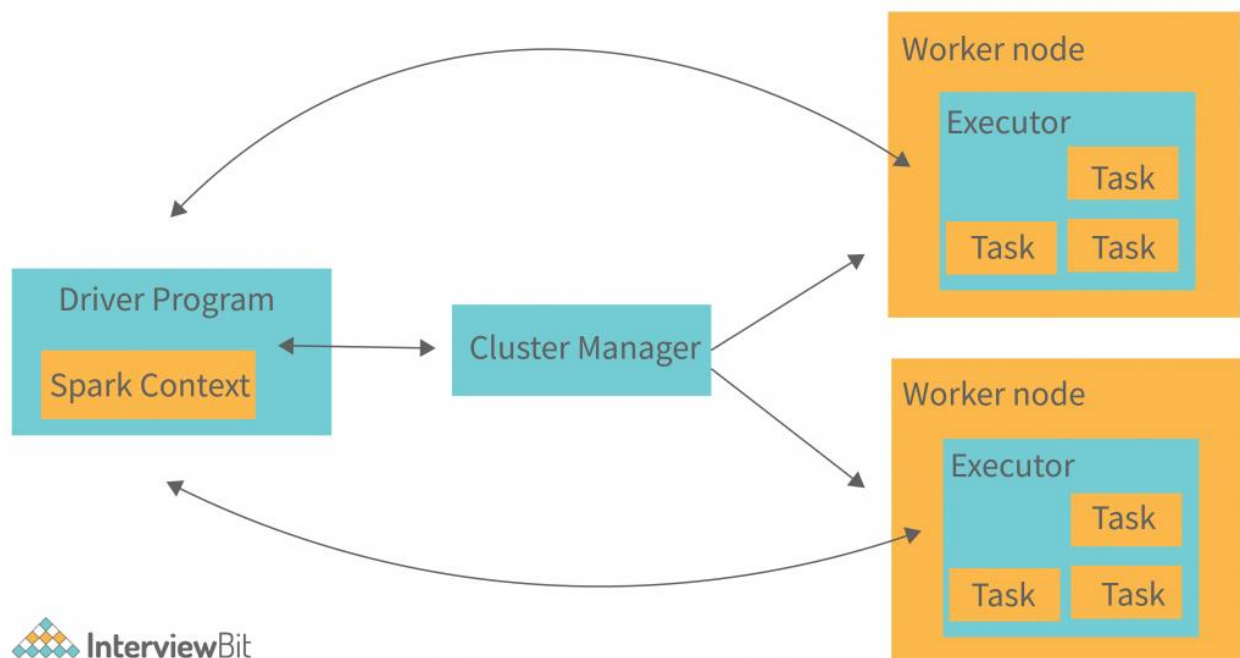
PySpark mang lại nhiều tính năng và lợi ích khác nhau, bao gồm:

- Cung cấp một giao diện lập trình sử dụng Python quen thuộc để làm việc hiệu quả với Spark

- Hỗ trợ nhiều nguồn và định dạng dữ liệu khác nhau, bao gồm dữ liệu có cấu trúc, bán cấu trúc và không có cấu trúc.
- Tích hợp tốt với các thư viện và công cụ Python phổ biến khác, như NumPy, pandas và scikit-learn.
- Khả năng mở rộng các công việc Spark trên một cụm máy tính, giúp tăng hiệu suất và xử lý được lượng dữ liệu lớn hơn.

### 3.2 Kiến trúc và nguyên lý hoạt động của PySpark

PySpark kế thừa kiến trúc của Apache Spark, được biểu thị như biểu đồ sau:



**Trong đó:**

1. **Driver Program:** Đây là chương trình chạy hàm main() của ứng dụng và tạo SparkContext, là điểm vào cho mọi chức năng Spark. Điều phối việc thực thi tác vụ trên toàn bộ cụm, bao gồm các thành phần như DAG Scheduler, Task Scheduler, Backend Scheduler, và Block Manager.
2. **Cluster Manager:** Quản lý việc phân bổ tài nguyên và thực thi công việc.

3. **Worker Nodes:** Các nút thực hiện tính toán và lưu trữ dữ liệu sau đó trả kết quả về Spark Context. Mỗi phân vùng dữ liệu được xem như một đơn vị xử lý công việc, giúp thực hiện các tác vụ song song và đồng thời.
4. **Executor:** Nơi thực hiện các tác vụ được giao bởi driver. Chạy các tác vụ song song, lưu trữ dữ liệu trong cache. Hoạt động trong một tiến trình Java và có thể phân bố một cách chủ động dựa trên khối lượng công việc.
5. **SparkContext:** Đại diện cho kết nối đến cụm Spark và điều phối tài nguyên.
6. **SparkSession:** API chính để làm việc với Spark trong ứng dụng Python, kết hợp SparkContext, SQLContext và HiveContext.

Nguyên lý hoạt động của Pyspark dựa trên các nguyên lý cốt lõi sau:

1. **RDD (Resilient Distributed Dataset):** Cấu trúc dữ liệu cơ bản, phân tán và bất biến, cho phép xử lý song song và chịu lỗi, hỗ trợ các phép biến đổi và hành động khác nhau.
2. **Directed Acyclic Graph (DAG):** Khi chạy ứng dụng Spark, driver chuyển đổi mã người dùng thành DAG, đại diện kế hoạch thực thi. DAG được chia thành nhiều giai đoạn khác nhau, mỗi giai đoạn bao gồm các tác vụ có thể thực thi song song, tối ưu hóa hiệu suất.
3. **Tính toán trong bộ nhớ:** PySpark thực hiện tính toán trong bộ nhớ, làm cho nó nhanh hơn nhiều so với các hệ thống dựa trên đĩa như Hadoop MapReduce điều này giúp tăng tốc độ xử lý BigData.
4. **DataFrame:** Tập dữ liệu phân tán có cấu trúc tương tự bảng trong cơ sở dữ liệu quan hệ.
5. **Hỗ trợ dữ liệu có cấu trúc:** Qua Spark SQL và DataFrame API, PySpark cho phép truy vấn SQL và thao tác dữ liệu có cấu trúc, tương

tự như pandas DataFrame trong Python cung cấp tối ưu hóa sâu hơn, phù hợp cho phân tích BigData.

6. **MLlib**: Thư viện machine learning của Spark, cung cấp các thuật toán và công cụ để xây dựng các mô hình machine learning, bao gồm cả K-means.

## 4.KẾT HỢP K-MEANS VỚI PYSPARK CHO PHÂN KHÚC KHÁCH HÀNG

PySpark MLlib cung cấp một triển khai hiệu quả của thuật toán K-means, được thiết kế để xử lý dữ liệu phân tán trên các cụm tính toán. Lớp `pyspark.ml.clustering.KMeans` hỗ trợ cả thuật toán K-means truyền thống và biến thể `K-means||`, một phương pháp khởi tạo centroid được tối ưu hóa để cải thiện tốc độ và chất lượng phân cụm.

Để áp dụng K-means phân khúc khách hàng với Pyspark, quy trình thực hiện nhìn chung như sau

### **Thu thập và chuẩn bị dữ liệu:**

- Thu thập dữ liệu từ nhiều nguồn (cơ sở dữ liệu, CRM, logs, ...)
- Tích hợp dữ liệu vào Spark DataFrame
- Làm sạch dữ liệu (xử lý giá trị thiếu, loại bỏ nhiễu)

### **Tiền xử lý dữ liệu:**

- Chuyển đổi dữ liệu (encoding cho biến phân loại)
- Chuẩn hóa dữ liệu (scaling)
- Giảm chiều dữ liệu nếu cần (PCA)

### **Xác định số lượng cụm tối ưu:**

- Sử dụng phương pháp Elbow hoặc Silhouette Score
- Đánh giá trực quan kết quả phân cụm

### **Huấn luyện mô hình K-means:**

- Khởi tạo mô hình với số cụm K đã xác định
- Huấn luyện mô hình trên dữ liệu đã chuẩn bị

### **Đánh giá và diễn giải kết quả:**

- Phân tích đặc điểm của từng cụm
- Đặt tên và mô tả các phân khúc khách hàng
- Trực quan hóa các phân khúc

### **Ứng dụng kết quả vào chiến lược kinh doanh:**

- Phát triển chiến lược marketing cho từng phân khúc
- Cá nhân hóa trải nghiệm khách hàng
- Tối ưu hóa sản phẩm và dịch vụ

Thông qua các bước thực hiện hóa thuật toán K-means với Pyspark trong ứng dụng phân khúc khách hàng, nhóm cũng đã rút ra được những phương hướng nhằm tối ưu hiệu suất, tiết kiệm thời gian trong quá trình thực hiện đề tài. Các phương thức đó như sau :

**Cấu hình tài nguyên phù hợp:**

- Điều chỉnh số lượng executors, cores và memory
- Sử dụng persist() hoặc cache() cho các DataFrame sử dụng nhiều lần

**Tiền xử lý hiệu quả:**

- Loại bỏ các tính năng không liên quan
- Giảm chiều dữ liệu nếu có quá nhiều biến

**Sampling dữ liệu:**

- Với dữ liệu rất lớn, có thể lấy mẫu để tìm số lượng cụm tối ưu trước khi huấn luyện trên toàn bộ dữ liệu

**Chọn chiến lược khởi tạo phù hợp:**

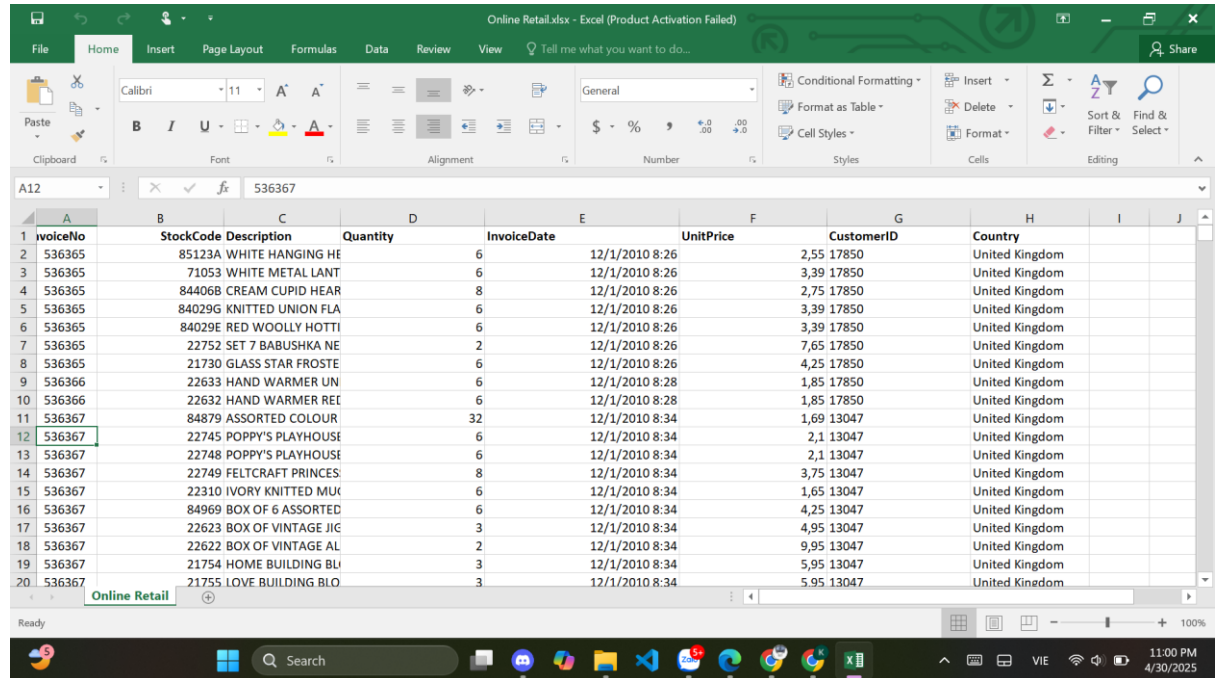
- Sử dụng K-means|| thay vì khởi tạo ngẫu nhiên

**Tối ưu hóa số lần lặp và ngưỡng hội tụ:**

- Điều chỉnh maxIter và tol tùy theo bài toán cụ thể

## 5.BIG DATA ONLINE RETAIL WITH PYSPARK

Tập dữ liệu bao gồm các đơn đặt hàng được thực hiện ở các quốc gia khác nhau từ tháng 12 năm 2010 đến tháng 12 năm 2011.



InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
536365	85123A	WHITE HANGING HE	6	12/1/2010 8:26	2,55	17850	United Kingdom
536365	71053	WHITE METAL LANT	6	12/1/2010 8:26	3,39	17850	United Kingdom
536365	84406B	CREAM CUPID HEAR	8	12/1/2010 8:26	2,75	17850	United Kingdom
536365	84029G	KNITTED UNION FLA	6	12/1/2010 8:26	3,39	17850	United Kingdom
536365	84029E	RED WOOLLY HOTTI	6	12/1/2010 8:26	3,39	17850	United Kingdom
536365	22752	SET 7 BABUSHKA NE	2	12/1/2010 8:26	7,65	17850	United Kingdom
536365	21730	GLASS STAR FROSTE	6	12/1/2010 8:26	4,25	17850	United Kingdom
536366	22633	HAND WARMER UN	6	12/1/2010 8:28	1,85	17850	United Kingdom
536366	22632	HAND WARMER REC	6	12/1/2010 8:28	1,85	17850	United Kingdom
536367	84879	ASSORTED COLOUR	32	12/1/2010 8:34	1,69	13047	United Kingdom
536367	22745	POPPY'S PLAYHOUSE	6	12/1/2010 8:34	2,1	13047	United Kingdom
536367	22748	POPPY'S PLAYHOUSE	6	12/1/2010 8:34	2,1	13047	United Kingdom
536367	22749	FELTCRAFT PRINCES	8	12/1/2010 8:34	3,75	13047	United Kingdom
536367	22310	IVORY KNITTED MUK	6	12/1/2010 8:34	1,65	13047	United Kingdom
536367	84969	BOX OF 6 ASSORTED	6	12/1/2010 8:34	4,25	13047	United Kingdom
536367	22623	BOX OF VINTAGE JIG	3	12/1/2010 8:34	4,95	13047	United Kingdom
536367	22622	BOX OF VINTAGE AL	2	12/1/2010 8:34	9,95	13047	United Kingdom
536367	21754	HOME BUILDING BLU	3	12/1/2010 8:34	5,95	13047	United Kingdom
536367	21755	LOVE BUILDING BLO	3	12/1/2010 8:34	5,95	13047	United Kingdom

Các ý nghĩa các trường dữ liệu trong file:

- Invoice NO: ID của đơn hàng, nếu ID bắt đầu bằng chữ "c" thể hiện đơn hàng đó bị hủy (Cancel).
- Stock Code: Mã sản phẩm.
- Description: Tên sản phẩm.
- Quantity: Số lượng sản phẩm trên đơn đặt hàng.
- Invoice Date: Ngày và giờ khi đơn hàng được tạo.
- UnitPrice: Giá sản phẩm trên mỗi đơn vị, tính bằng pound.
- CustomerID: ID của khách hàng..
- Country: Quốc gia nơi khách hàng cư trú.

### ***Bài toán đặt ra:***

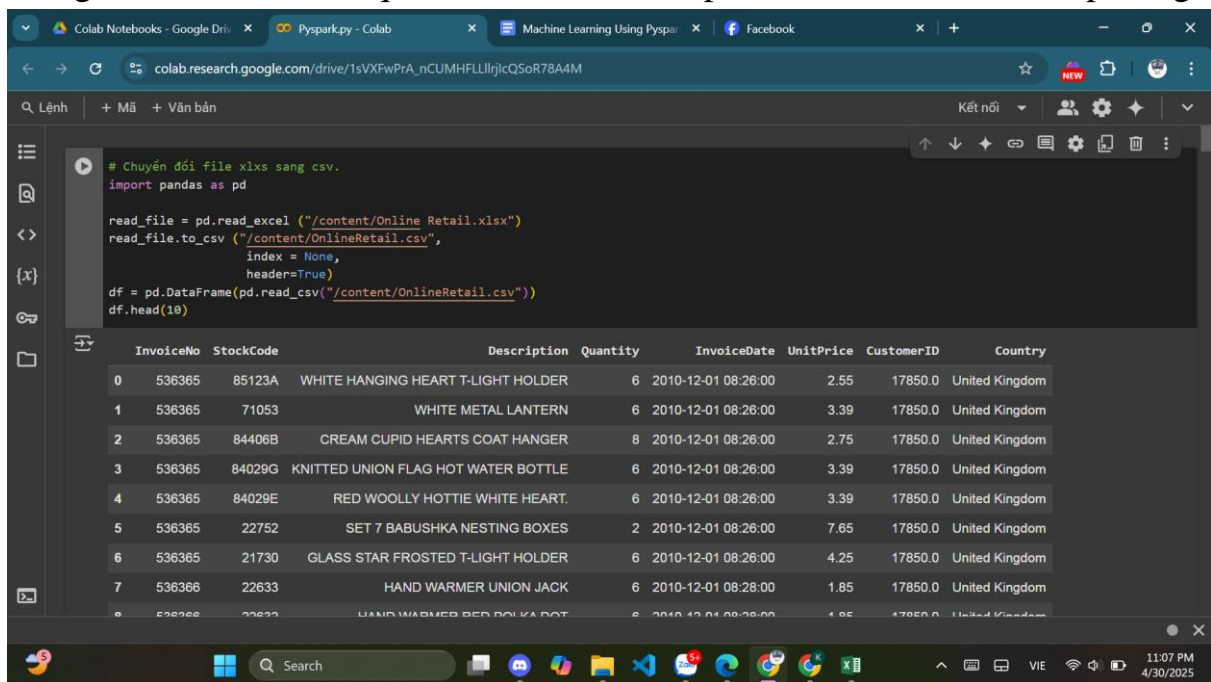
Phân khúc khách hàng là một cụm từ chắc hẳn bạn đã từng nghe khi ai đó nói về chủ đề kinh doanh hay marketing. Đây là một trong những chiến lược quan

trọng mà các công ty sử dụng để phân nhóm khách hàng của mình, từ đó có những chiến lược quảng cáo, chăm sóc khách hàng phù hợp. Nếu bạn ghé vào cửa hàng quần áo X vào mỗi dịp đầu tháng, có thể bạn được xếp vào nhóm những người tiêu tiền đầu tháng, nghèo cuối tháng, hay lương được trả vào cuối tháng, thích mua sắm hàng tháng, thích mua quần áo,... Và khi đó, công ty X sẽ có những quảng cáo với mào "Chỉ dành riêng cho bạn" hay "Duy nhất trong ngày hôm nay" và những quảng cáo này sẽ xuất hiện cho những người cùng nhóm với bạn, xuất hiện vào đầu tháng, các mặt hàng sale chủ yếu là quần áo, ...Nhóm sẽ thực hiện phân khúc khách hàng dựa trên K-means với tập dữ liệu trên.

## *Phân tích dữ liệu tổng quan*

### *Convert dữ liệu sang csv*

Bởi vì file dataset tải xuống dưới dạng excel, mà việc phân tích dữ liệu sẽ thuận tiện hơn dưới dạng .csv, nên nhóm sẽ convert nó sang csv bằng python thông qua pandas package.



```
# Chuyển đổi file xlsx sang csv.
import pandas as pd

read_file = pd.read_excel("/content/Online Retail.xlsx")
read_file.to_csv("/content/OnlineRetail.csv",
                 index=None,
                 header=True)
df = pd.DataFrame(pd.read_csv("/content/OnlineRetail.csv"))
df.head(10)
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
5	536365	22752	SET 7 BABUSHKA NESTING BOXES	2	2010-12-01 08:26:00	7.65	17850.0	United Kingdom
6	536365	21730	GLASS STAR FROSTED T-LIGHT HOLDER	6	2010-12-01 08:26:00	4.25	17850.0	United Kingdom
7	536366	22633	HAND WARMER UNION JACK	6	2010-12-01 08:28:00	1.85	17850.0	United Kingdom
8	536366	22633	HAND WARMER RED POLKA DOT	6	2010-12-01 08:28:00	1.85	17850.0	United Kingdom



## Cài đặt thư viện Pyspark

Ta sử dụng câu lệnh **!pip install pyspark**.

```
[ ] # Cài đặt thư viện Pyspark
!pip install pyspark

Requirement already satisfied: pyspark in /usr/local/lib/python3.11/dist-packages (3.5.5)
Requirement already satisfied: py4j==0.10.9.7 in /usr/local/lib/python3.11/dist-packages (from pyspark) (0.10.9.7)
```

## Phân tích dữ liệu cơ bản

Để bắt đầu các thao tác dữ liệu với PySpark, chúng ta cần khởi tạo session với SparkSession. Nó xây dựng một khung dữ liệu trong PysPark để chúng ta có thể sử dụng các chức năng của PysPark lên dữ liệu của mình.

```
] from pyspark.sql import SparkSession
from pyspark.sql.functions import *
from pyspark.sql.types import *
# Init SparkSession
spark = SparkSession.builder.appName("Pyspark Tutorial").config("spark.memory.offHeap.enabled", "true").config("spark.memory.offHeap.size", "10g").getOrCreate()
```

Đọc dữ liệu trong file CSV và convert bằng PySpark

```
# Read data frame
df = spark.read.csv('/content/OnlineRetail.csv', header=True, escape="\"")
df.show(5)
```

InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
536365	85123A	WHITE HANGING HEA...	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
536365	84406B	CREAM CUPID HEART...	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom
536365	84029G	KNITTED UNION FLA...	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
536365	84029E	RED WOOLLY HOTTIE...	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom

only showing top 5 rows

Một số truy vấn cơ bản sử dụng Pyspark

Đếm số dòng dữ liệu

```
[ ] df.count()

541909
```

Đếm xem có bao nhiêu khách hàng

```
[ ] df.select('CustomerID').distinct().count()
```

⇒ 4373

Các quốc gia có số lượng khách hàng thẻ nào

```
[ ] df = df_filter
# Các quốc gia có số lượng khách hàng thẻ nào
df.groupBy('Country').agg(countDistinct('CustomerID').alias('country_count')).orderBy(desc('country_count')).show()
```

```
+-----+-----+
|      Country|country_count|
+-----+-----+
| United Kingdom|      3921|
|      Germany|       94|
|      France|       87|
|      Spain|       30|
|      Belgium|       25|
| Switzerland|       21|
|      Portugal|       19|
|      Italy|       14|
|      Finland|       12|
|      Austria|       11|
|      Norway|       10|
|      Denmark|        9|
| Channel Islands|        9|
|      Australia|        9|
|      Netherlands|        9|
|      Sweden|        8|
|      Cyprus|        8|
|      Japan|        8|
|      Poland|        6|
|      Greece|        4|
+-----+-----+
only showing top 20 rows
```

Định dạng lại trường dữ liệu ngày tháng năm thành timestamp có thể sort, tìm min, max.

```
spark.sql("set spark.sql.legacy.timeParserPolicy=LEGACY")
df = df.withColumn('date',to_timestamp("InvoiceDate", 'yy-MM-dd HH:mm:ss'))
```

Ngày có đơn hàng gần đây nhất

```
# Ngày có đơn hàng gần đây nhất
df.select(max("date")).show()
```

```

+-----+
|      max(date) |
+-----+
|2011-12-09 12:50:00|
+-----+
```

Ngày có đơn hàng đầu tiên

```
#Ngày có đơn hàng đầu tiên
df.select(min("date")).show()
```

```

+-----+
|      min(date) |
+-----+
|2010-12-01 08:26:00|
+-----+
```

## Tiền xử lý dữ liệu

Có thể thấy trong file dữ liệu có tới 7 trường dữ liệu, không phải trường dữ liệu nào cũng có ích cho phân cụm, và cũng không phải sẽ dùng được dữ liệu luôn.

RFM (bao gồm Recency (Tần suất mua hàng gần đây), Frequency (Tần suất mua hàng), và Monetary (Giá trị đặt hàng)) là một phương pháp phân tích khách hàng được sử dụng rộng rãi trong lĩnh vực tiếp thị để đánh giá giá trị của khách hàng dựa trên hành vi mua hàng của họ.

- **Recency:** đo thời điểm mà khách hàng đã mua hàng lần cuối. Khách hàng mới mua hàng gần đây được xem là có giá trị cao hơn so với khách hàng mua hàng lâu đến không mua hàng nữa.
- **Frequency:** đo tần suất mà khách hàng mua hàng trong một khoảng thời gian nhất định. Khách hàng mua hàng thường xuyên được xem là có giá trị hơn so với những khách hàng mua hàng ít lần.
- **Monetary:** đo giá trị đặt hàng của khách hàng. Khách hàng đặt hàng có giá trị cao hơn được xem là có giá trị cao hơn so với những khách hàng đặt hàng có giá trị thấp.

Trước tiên, khi kiểm tra dữ liệu, chúng tôi nhận thấy rằng các hóa đơn có “Invoice No” chứa ký tự “C” (tức là các đơn hàng bị hủy) sẽ dẫn đến giá trị “quantity” âm, kéo theo “Monetary value” cũng âm tương ứng. Điều này có thể gây sai lệch trong việc đánh giá hành vi khách hàng, đặc biệt với những khách hàng có tần suất mua hàng thấp nhưng tỷ lệ hủy đơn cao. Để đảm bảo tính chính xác và phù hợp của dữ liệu, nhóm đề xuất thực hiện bước lọc dữ liệu ngay từ đầu, loại bỏ hoàn toàn các đơn hàng bị hủy trước khi tiến hành phân tích. Phương pháp này sẽ giúp giảm thiểu rủi ro sai lệch và mang lại kết quả đáng tin cậy hơn cho các bước phân tích tiếp theo.

```
+ Mã + Văn bản
[ ] df_filter = df.select("*").filter(~col("InvoiceNo").startswith("C"))
df_filter.count()

⇒ 532621

[ ] df_filter.select('CustomerID').distinct().count()

⇒ 4340
```

Như vậy sau khi lọc thì dữ liệu giảm đi đáng kể, số khách hàng giảm đi 33 người.

## 5.1 Recency

Trong phần này, nhóm sẽ có mục tiêu tính toán ra một giá trị đại diện cho việc thời điểm khách hàng mua lần cuối so với 1 mốc 0 nhất định (ở đây mình chọn mốc 0 là thời gian đầu tiên có đơn hàng đã tính ra bên trên). Sau đó chỉ cần lấy thời gian gần nhất khách hàng đặt đơn trừ đi mốc thời gian đó, ta sẽ có 1 giá trị đại diện cho Recency. Rõ ràng giá trị này càng lớn chứng tỏ khách hàng càng mua gần đây. Việc tính toán này sẽ thông qua 1 số bước như sau

- Tạo 1 cột mới, đặt giá trị của tất cả cột đó là ngày đầu tiên có đơn hàng. Cột này có tên "from\_date"

```
] df = df.withColumn("from_date", lit("2010-12-01 08:26:00"))
```

Lấy giá trị thời gian mua của từng đơn hàng trừ đi from\_date, ta sẽ biết đơn hàng đo được đặt cách mốc thời gian 0 là bao nhiêu (theo đơn vị timestamp), giá trị sẽ được lưu tại cột 'recency'.

```
df = df.withColumn('from_date', to_timestamp("from_date", 'yy-MM-dd HH:mm'))
df2 = df.withColumn('from_date', to_timestamp(col('from_date'))).withColumn('recency', col("date").cast("long") - col('from_date').cast("long"))
df2 = df2.alias('df2').groupBy('CustomerID').agg(max('recency').alias('recency')).withColumn('recency', col('recency').cast('timestamp'))
```

Mỗi khách hàng có thể mua nhiều lần vào nhiều mốc thời gian khác nhau, nên ta chỉ quan tâm lần cuối cùng họ mua, vì vậy cần xử lý lại cột 'recency'.

```
df2 = df2.join(df2.groupBy('CustomerID').agg(max('recency').alias('recency')), on='recency', how='leftsemi')
df2.show()
```

Agency	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	date	from_date
5220	536384	82484	WOOD BLACK BOARD ...	3	2010-12-01 09:53:00	6.45	18074.0	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00
5220	536384	84755	COLOUR GLASS T-LI...	48	2010-12-01 09:53:00	0.65	18074.0	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00
5220	536384	22464	HANGING METAL HEA...	12	2010-12-01 09:53:00	1.65	18074.0	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00
5220	536384	21324	HANGING MEDINA LA...	6	2010-12-01 09:53:00	2.95	18074.0	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00
5220	536384	22457	NATURAL SLATE HEA...	12	2010-12-01 09:53:00	2.95	18074.0	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00
5220	536384	22469	HEART OF WICKER S...	40	2010-12-01 09:53:00	1.45	18074.0	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00
5220	536384	22470	HEART OF WICKER L...	40	2010-12-01 09:53:00	2.55	18074.0	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00
5220	536384	22224	WHITE LOVEBIRD LA...	6	2010-12-01 09:53:00	2.95	18074.0	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00
5220	536384	21340	CLASSIC METAL BIR...	2	2010-12-01 09:53:00	12.75	18074.0	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00
5220	536384	22189	CREAM HEART CARD ...	4	2010-12-01 09:53:00	3.95	18074.0	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00
5220	536384	22427	ENAMEL FLOWER JUG...	3	2010-12-01 09:53:00	5.95	18074.0	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00
5220	536384	22428	ENAMEL FIRE BUCKE...	6	2010-12-01 09:53:00	6.95	18074.0	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00
5220	536384	22424	ENAMEL BREAD BIN ...	8	2010-12-01 09:53:00	10.95	18074.0	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00
7860	536393	22180	RETROSPOT LAMP	8	2010-12-01 10:37:00	9.95	13747.0	United Kingdom	2010-12-01 10:37:00	2010-12-01 08:26:00
10860	536403	22867	HAND WARMER BIRD ...	96	2010-12-01 11:27:00	1.85	12791.0	Netherlands	2010-12-01 11:27:00	2010-12-01 08:26:00
10860	536403	POST	POSTAGE	1	2010-12-01 11:27:00	15.0	12791.0	Netherlands	2010-12-01 11:27:00	2010-12-01 08:26:00
11940	536409	90199C	5 STRAND GLASS NE...	3	2010-12-01 11:45:00	6.35	17908.0	United Kingdom	2010-12-01 11:45:00	2010-12-01 08:26:00
11940	536409	21479	WHITE SKULL HOT W...	1	2010-12-01 11:45:00	3.75	17908.0	United Kingdom	2010-12-01 11:45:00	2010-12-01 08:26:00
11940	536409	22111	SCOTTIE DOG HOT W...	1	2010-12-01 11:45:00	4.95	17908.0	United Kingdom	2010-12-01 11:45:00	2010-12-01 08:26:00
11940	536409	22785	SQUARECUSHION COV...	1	2010-12-01 11:45:00	6.75	17908.0	United Kingdom	2010-12-01 11:45:00	2010-12-01 08:26:00

ly showing top 20 rows

## 5.2 Frequency

Phần này thì chúng ta sẽ tính tần suất một khách hàng mua một đồ gì đó. Chúng ta chỉ cần nhóm theo từng ID khách hàng và đếm số mặt hàng họ đã mua.

```
[ ] df_freq = df2.groupby('CustomerID').agg(count('InvoiceNo').alias('frequency'))
df_freq.show()
```

CustomerID	frequency
17786.0	72
16917.0	9
15891.0	62
17955.0	1
14542.0	5
12891.0	1
16553.0	1
14532.0	20
17536.0	3
14722.0	29
13827.0	65
17353.0	4
15070.0	1
16351.0	8
18085.0	20
12350.0	17
14727.0	10
16838.0	11
13533.0	33
15016.0	20

only showing top 20 rows

Lúc này frequency dataframe của chúng ta chỉ có 2 cột đứng riêng lẻ, chúng ta sẽ nối nó vào dataframe chúng ta đang làm việc để thống nhất, cũng như check xem mình truy xuất có đúng không.

```
#join data
df3=df2.join(df_freq,on='CustomerID',how='inner')
df3.show()
```

agency	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	Country	date	from_date	frequency
5220	536384	82484	WOOD BLACK BOARD ...	3	2010-12-01 09:53:00	6.45	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00	13
5220	536384	84755	COLOUR GLASS T-LI...	48	2010-12-01 09:53:00	0.65	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00	13
5220	536384	22464	HANGING METAL HEA...	12	2010-12-01 09:53:00	1.65	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00	13
5220	536384	21324	HANGING MEDINA LA...	6	2010-12-01 09:53:00	2.95	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00	13
5220	536384	22457	NATURAL SLATE HEA...	12	2010-12-01 09:53:00	2.95	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00	13
5220	536384	22469	HEART OF WICKER S...	40	2010-12-01 09:53:00	1.45	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00	13
5220	536384	22470	HEART OF WICKER L...	40	2010-12-01 09:53:00	2.55	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00	13
5220	536384	22224	WHITE LOVEBIRD LA...	6	2010-12-01 09:53:00	2.95	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00	13
5220	536384	21340	CLASSIC METAL BIR...	2	2010-12-01 09:53:00	12.75	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00	13
5220	536384	22189	CREAM HEART CARD ...	4	2010-12-01 09:53:00	3.95	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00	13
5220	536384	22427	ENAMEL FLOWER JUG...	3	2010-12-01 09:53:00	5.95	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00	13
5220	536384	22428	ENAMEL FIRE BUCKE...	6	2010-12-01 09:53:00	6.95	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00	13
5220	536384	22424	ENAMEL BREAD BIN ...	8	2010-12-01 09:53:00	10.95	United Kingdom	2010-12-01 09:53:00	2010-12-01 08:26:00	13
7860	536393	22180	RETROSPOT LAMP	8	2010-12-01 10:37:00	9.95	United Kingdom	2010-12-01 10:37:00	2010-12-01 08:26:00	1
10860	536403	22867	HAND WARMER BIRD ...	96	2010-12-01 11:27:00	1.85	Netherlands	2010-12-01 11:27:00	2010-12-01 08:26:00	2
10860	536403	POST	POSTAGE	1	2010-12-01 11:27:00	15.0	Netherlands	2010-12-01 11:27:00	2010-12-01 08:26:00	2
11940	536409	90199C	5 STRAND GLASS NE...	3	2010-12-01 11:45:00	6.35	United Kingdom	2010-12-01 11:45:00	2010-12-01 08:26:00	58
11940	536409	21479	WHITE SKULL HOT W...	1	2010-12-01 11:45:00	3.75	United Kingdom	2010-12-01 11:45:00	2010-12-01 08:26:00	58
11940	536409	22111	SCOTTIE DOG HOT W...	1	2010-12-01 11:45:00	4.95	United Kingdom	2010-12-01 11:45:00	2010-12-01 08:26:00	58
11940	536409	22785	SQUARECUSHION COV...	1	2010-12-01 11:45:00	6.75	United Kingdom	2010-12-01 11:45:00	2010-12-01 08:26:00	58

op 20 rows

## 5.2 Monetary

Phần này tính xem mỗi khách hàng đã chi bao nhiêu tiền để mua sắm, phần này sẽ chia làm 2 bước


- Tính số lượng và đơn giá của một lần mua hàng

```
[ ] m_val = df3.withColumn('TotalAmount',col("Quantity") * col("UnitPrice"))
```

- Tính tổng số tiền mà khách hàng đã chi

```
m_val = m_val.groupBy('CustomerID').agg(sum('TotalAmount').alias('monetary_value'))
m_val.show()
```





CustomerID	monetary_value
17786.0	278.74
16917.0	391.52000000000001
15891.0	524.52
17955.0	163.2
14542.0	103.25000000000001
12891.0	85.0
16553.0	204.0
14532.0	140.55
17536.0	76.5
14722.0	187.91999999999996
13827.0	299.31
17353.0	870.0
15070.0	106.2
16351.0	153.9
18085.0	386.04999999999995
12350.0	334.40000000000003
14727.0	268.58
16838.0	196.73000000000002
13533.0	124.14999999999999
15016.0	170.04

only showing top 20 rows

- Join dữ liệu vào dataframe đang tổng hợp

```
[ ] # join
    final_df = m_val.join(df3,on='CustomerID',how='inner')
    final_df.show()
```

monetary_value	recency	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	Country	date	fr
278.74	4860540	542239	21166	COOK WITH WINE ME...	3	2011-01-26 14:35:00	1.95	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	21181	PLEASE ONE PERSON...	2	2011-01-26 14:35:00	2.1	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	85150	LADIES & GENTLEME...	1	2011-01-26 14:35:00	2.55	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	85152	HAND OVER THE CHO...	3	2011-01-26 14:35:00	2.1	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	21988	CHOCOLATE THIS WA...	1	2011-01-26 14:35:00	2.1	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	22413	METAL SIGN TAKE I...	1	2011-01-26 14:35:00	2.95	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	85123A	WHITE HANGING HEA...	2	2011-01-26 14:35:00	2.95	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	22189	CREAM HEART CARD ...	1	2011-01-26 14:35:00	3.95	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	22349	DOG BOWL CHASING ...	1	2011-01-26 14:35:00	3.75	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	22627	MINT KITCHEN SCALES	1	2011-01-26 14:35:00	8.5	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	22464	HANGING METAL HEA...	1	2011-01-26 14:35:00	1.65	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	22625	RED KITCHEN SCALES	1	2011-01-26 14:35:00	8.5	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	171648	ASS COL SMALL SAN...	1	2011-01-26 14:35:00	0.42	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	47578A	ENGLISH ROSE SMAL...	1	2011-01-26 14:35:00	0.85	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	21487	BROWN CHECK CAT D...	1	2011-01-26 14:35:00	4.25	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	850408	SET/4 BLUE FLOWER...	1	2011-01-26 14:35:00	1.65	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	21114	LAVENDER SCENTED ...	10	2011-01-26 14:35:00	1.25	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	22189	CREAM HEART CARD ...	1	2011-01-26 14:35:00	3.95	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	21166	COOK WITH WINE ME...	1	2011-01-26 14:35:00	1.95	United Kingdom	2011-01-26 14:35:00	2010-12-01
278.74	4860540	542239	21174	POTTERING IN THE ...	1	2011-01-26 14:35:00	1.95	United Kingdom	2011-01-26 14:35:00	2010-12-01

Từ đây chúng ta cũng chỉ cần quan tâm 4 trường dữ liệu này

```
[ ] final_df = final_df.select(['recency','frequency','monetary_value','CustomerID']).distinct()
final_df.show()
```

recency	frequency	monetary_value	CustomerID
4860540	72	278.74	17786.0
8393340	9	391.5200000000001	16917.0
12715740	62	524.52	15891.0
15132840	1	163.2	17955.0
16242780	5	103.2500000000001	14542.0
16257900	1	85.0	12891.0
18170160	1	204.0	16553.0
11853540	20	140.55	14532.0
16952220	3	76.5	17536.0
19549920	29	187.91999999999996	14722.0
20747940	65	299.31	13827.0
21113580	4	870.0	17353.0
97020	1	106.2	15070.0
3549600	8	153.9	16351.0
3739860	20	386.04999999999995	18085.0
5470500	17	334.40000000000003	12350.0
8576340	10	268.58	14727.0
15223620	11	196.73000000000002	16838.0
16514880	33	124.14999999999999	13533.0
17379600	20	170.04	15016.0

## Chuẩn hóa dữ liệu

Đây là một phần vô cùng quan trọng và không thể thiếu trong các bài toán về dữ liệu. Mỗi trường dữ liệu có một đơn vị khác nhau, nếu không chuẩn hóa thì chắc chắn sẽ nảy sinh nhiều vấn đề về sau. PySpark hỗ trợ việc chuẩn hóa này với package pyspark.ml

```
[ ] from pyspark.ml.feature import VectorAssembler
    from pyspark.ml.feature import StandardScaler

    assemble=VectorAssembler(inputCols=[
        'recency', 'frequency', 'monetary_value'
    ], outputCol='features')

    assembled_data=assemble.transform(final_df)

    scale=StandardScaler(inputCol='features',outputCol='standardized')
    data_scale=scale.fit(assembled_data)
    data_scale_output=data_scale.transform(assembled_data)
    data_scale_output.select('standardized').show(2,truncate=False)
```

Dữ liệu sau khi chuẩn hóa

```
+-----+
|standardized|
+-----+
|[0.5600258964872293,1.8173891485441467,0.09947228534979743]|
|[0.9670710986890596,0.22717364356801834,0.13971941293015963]|
+-----+
only showing top 2 rows
```

Triển khai học máy

```
from pyspark.ml.clustering import KMeans
from pyspark.ml.evaluation import ClusteringEvaluator
import numpy as np
from tqdm import tqdm

cost = np.zeros(10)

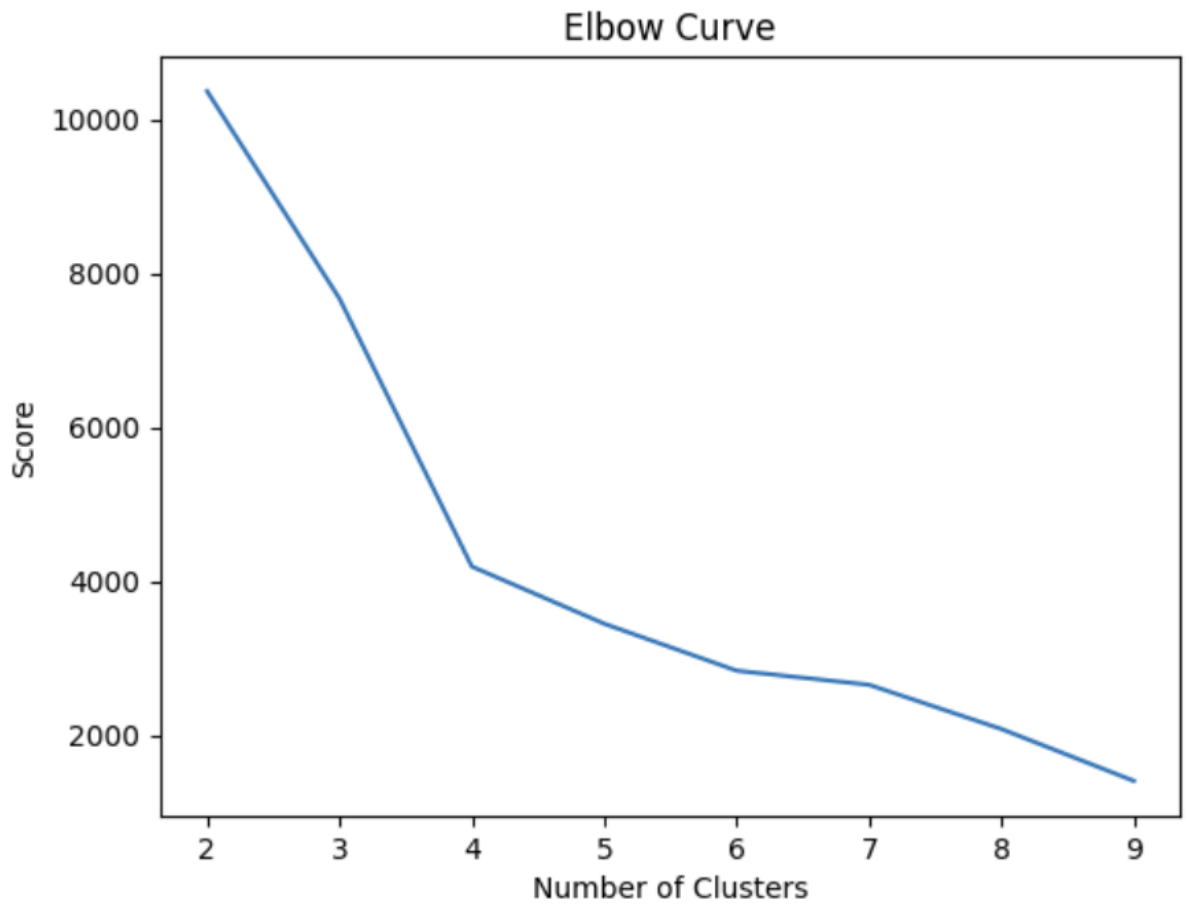
evaluator = ClusteringEvaluator(predictionCol='prediction', featuresCol='standardized',metricName='silhouette', distanceMeasure='squaredEuclidean')

for i in tqdm(range(2,10)):
    KMeans_algo=KMeans(featuresCol='standardized', k=i)
    KMeans_fit=KMeans_algo.fit(data_scale_output)
    output=KMeans_fit.transform(data_scale_output)
    cost[i] = KMeans_fit.summary.trainingCost
```

100% | 8/8 [06:43<00:00, 50.38s/it]

Đoạn code trên thực hiện mô phỏng quá trình phân cụm dữ liệu bằng thuật toán K-Means và đánh giá hiệu suất của mô hình. Cụ thể, đoạn code sử dụng các thư viện PySpark (pyspark.ml.clustering, pyspark.ml.evaluation), NumPy và TQDM để triển khai quy trình. Đầu tiên, một mảng cost được khởi tạo với giá trị 0 để lưu kết quả chi phí huấn luyện. Tiếp theo, một vòng lặp chạy từ 2 đến 19 được thiết lập để thử nghiệm K-Means với số lượng cụm (k) khác nhau. Trong mỗi lần lặp, dữ liệu được chuẩn hóa thông qua featuresCol='standardized', sau đó mô hình K-Means được huấn luyện và áp dụng lên tập dữ liệu đã chuẩn hóa. Kết quả phân cụm được đánh giá bằng chỉ số Silhouette với khoảng cách Euclid bình phương (metricName='silhouette', distanceMeasure='squaredEuclidean'). Cuối cùng, chi phí huấn luyện của mô hình tại mỗi giá trị k được lưu vào mảng cost để phân tích và so sánh, nhằm tìm ra số lượng cụm tối ưu cho bài toán phân cụm.

```
[ ] import pandas as pd
import pylab as pl
df_cost = pd.DataFrame(cost[2:])
df_cost.columns = ["cost"]
new_col = range(2,10)
df_cost.insert(0, 'cluster', new_col)
pl.plot(df_cost.cluster, df_cost.cost)
pl.xlabel('Number of Clusters')
pl.ylabel('Score')
pl.title('Elbow Curve')
pl.show()
```



## Chọn K=4

```
[ ] kmeans_algo=KMeans(featuresCol='standardized', k=4)
    kmeans_fit=kmeans_algo.fit(data_scale_output)
```

## Tiến hành dự đoán

```
preds=kmeans_fit.transform(data_scale_output)
preds.show(5)
```

recency	frequency	monetary_value	CustomerID	features	standardized	prediction
4860540	72	278.74	17786.0	[4860540.0, 72.0, 2...	[0.56002589648722...	1
8393340	9	391.5200000000001	16917.0	[8393340.0, 9.0, 39...	[0.96707109868905...	1
12715740	62	524.52	15891.0	[1.271574E7, 62.0, ...]	[1.46509311578517...	1
15132840	1	163.2	17955.0	[1.513284E7, 1.0, 1...	[1.74358863159191...	1
16242780	5	103.2500000000001	14542.0	[1.624278E7, 5.0, 1...	[1.87147465733124...	1

only showing top 5 rows

## 5.4 Kết quả :

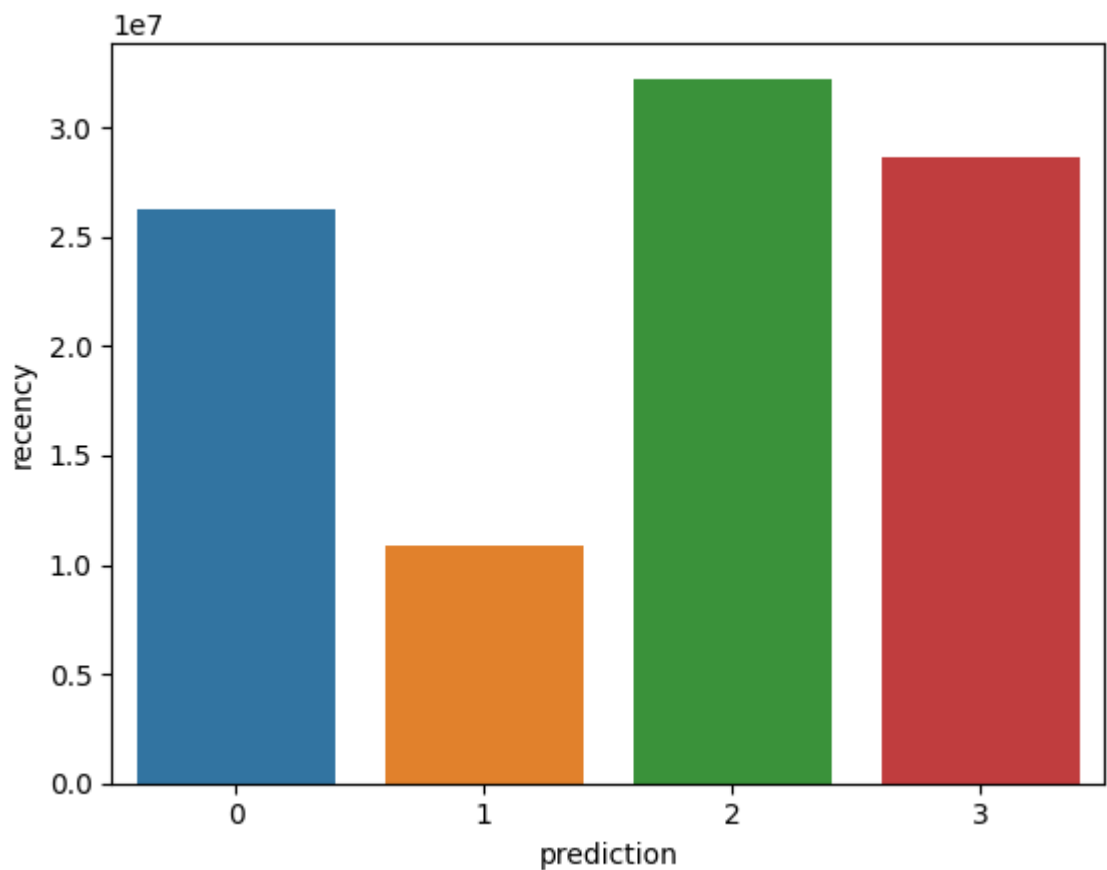
Chúng ta xem kết quả trực quan hóa phân khúc khách hàng dựa vào matplotlib

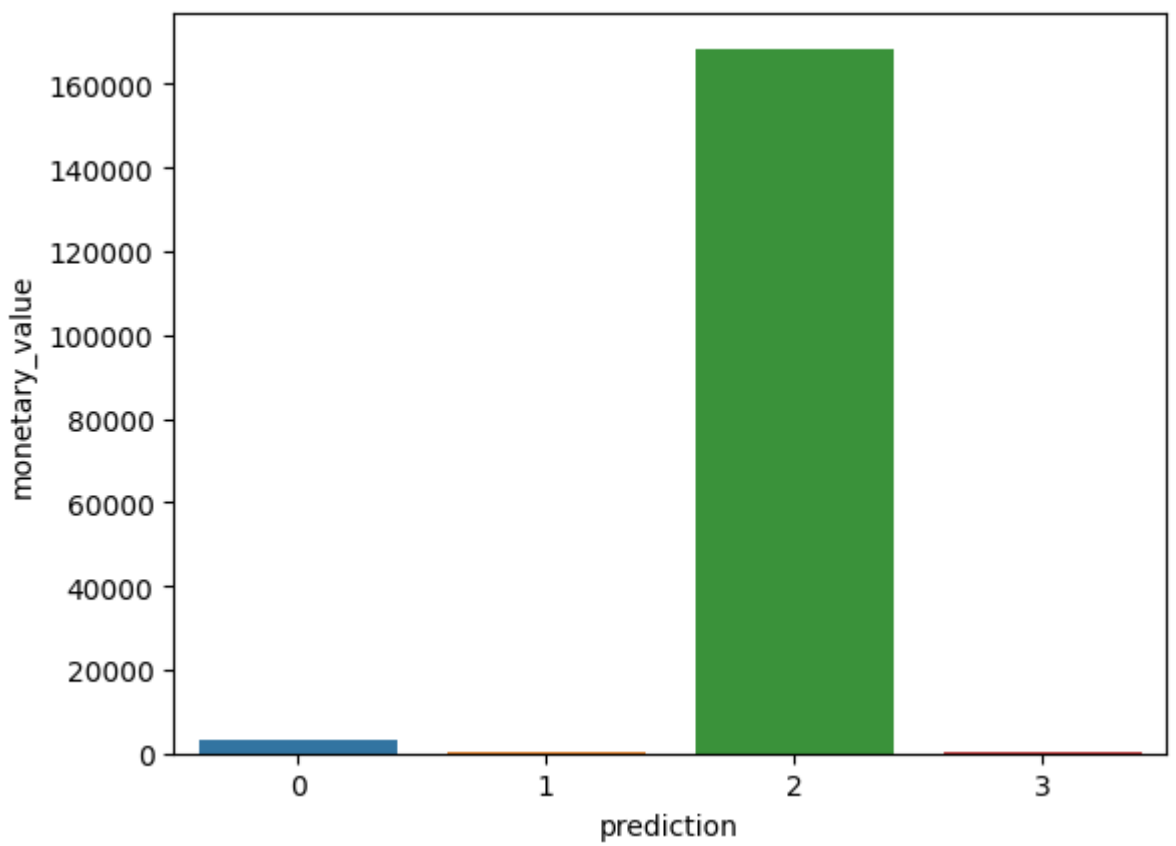
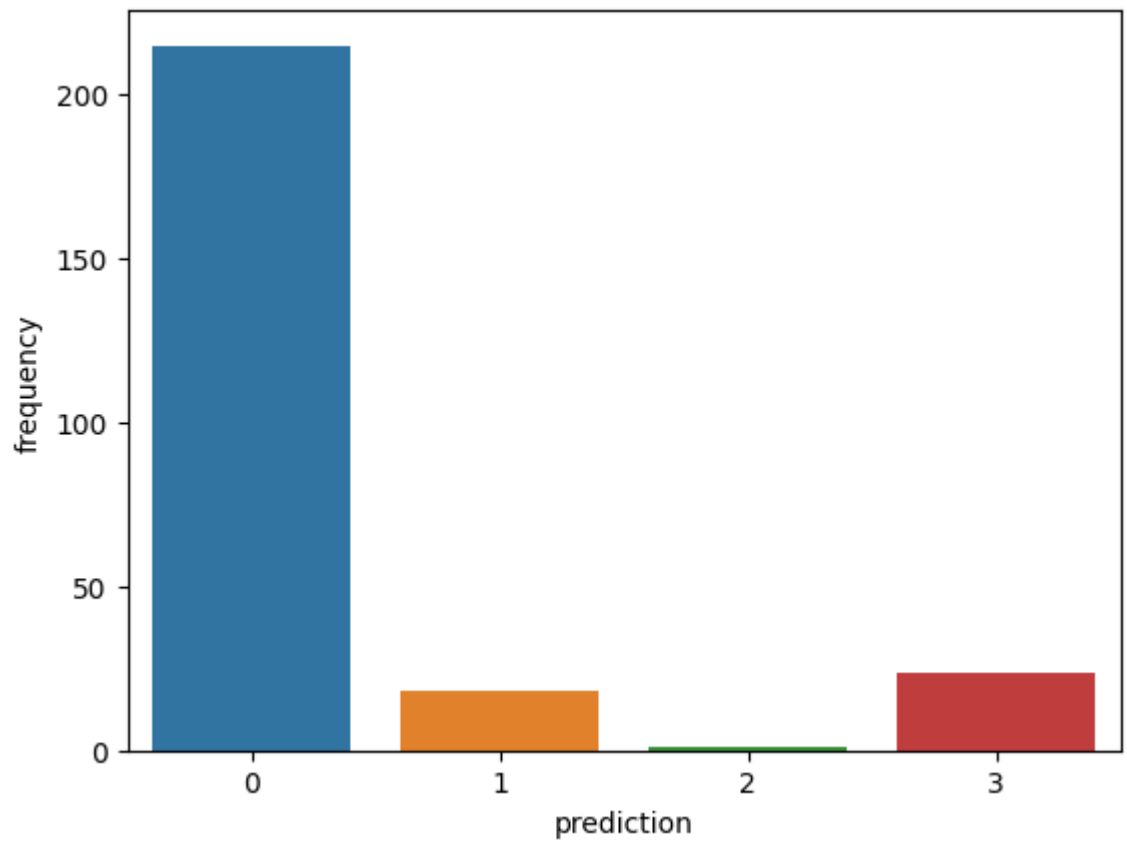
```
import matplotlib.pyplot as plt
import seaborn as sns

df_viz = preds.select('recency', 'frequency', 'monetary_value', 'prediction')
df_viz = df_viz.toPandas()
avg_df = df_viz.groupby(['prediction'], as_index=False).mean()

list_value = ['recency', 'frequency', 'monetary_value']

for i in list_value:
    sns.barplot(x='prediction', y=str(i), data=avg_df)
    plt.show()
```





**Nhận xét:**

- *Nhóm 0*: Nhóm này có tần suất mua hàng tương đối cao, cao trội hơn 3 nhóm còn lại, lần truy cập gần nhất cũng tương đối cao, giá trị tiền mua hàng tương đối nhỏ, cho thấy là một đối tượng đa số là cá nhân, hướng tới các sản phẩm giá rẻ
- *Nhóm 1*: Nhóm này có cả 3 chỉ số lần truy cập gần nhất, tần suất mua hàng và tổng tiền mua hàng rất thấp, không có quá nhiều hi vọng là khách hàng tiềm năng, khả năng cao sẽ ngừng mua hàng trong thời gian tới
- *Nhóm 2*: Tần suất đặt hàng rất ít, tuy nhiên gần đây lại đặt phổ biến, lượng tiền mua hàng cao vượt trội nhiều lần so với các nhóm khác. Nhóm này khả năng là các doanh nghiệp có xu hướng mua các loại hàng có giá trị cao hoặc mua có số lượng lớn
- *Nhóm 3*: Nhóm này có tổng tiền mua hàng rất thấp, tần suất mua hàng không quá nhiều, và truy cập gần đây tương đối cao. Nhóm này có khả năng mua theo đợt, hoặc là những người mới tham gia sàn thương mại.



## 6. ỨNG DỤNG THỰC TIỄN

Trong môi trường cạnh tranh ngày càng cao, việc hiểu rõ từng phân khúc khách hàng là yếu tố then chốt giúp doanh nghiệp đưa ra các chiến lược marketing hiệu quả, tối ưu hóa chi phí và nâng cao mức độ hài lòng của người tiêu dùng. Tuy nhiên, không phải lúc nào doanh nghiệp cũng có sẵn nhãn dữ liệu (labels) để nhận diện các nhóm khách hàng khác nhau. Do đó, các kỹ thuật học không giám sát như K-Means được ứng dụng rộng rãi để khám phá cấu trúc ẩn trong tập dữ liệu.

Trong đề tài này, dữ liệu khách hàng được trích từ tập “Mall Customer Segmentation Data” bao gồm các đặc trưng chính như:

- Age (Tuổi)
- Annual Income (Thu nhập hằng năm)
- Spending Score (Điểm chi tiêu, đại diện cho hành vi mua sắm)

Sau khi xử lý dữ liệu và chuẩn hóa, thuật toán K-Means được áp dụng để phân chia khách hàng thành các cụm riêng biệt. Kết quả phân cụm cho thấy có thể hình thành các nhóm khách hàng tiêu biểu như:

- Nhóm thu nhập cao – chi tiêu cao → Khách hàng VIP, cần ưu đãi đặc biệt.
- Nhóm thu nhập trung bình – chi tiêu thấp → Khách hàng tiềm năng, cần kích cầu.
- Nhóm thu nhập thấp – chi tiêu trung bình → Khách hàng giá nhạy, cần chiến lược giá hợp lý.

Việc phân cụm này mang lại nhiều lợi ích thực tiễn cho doanh nghiệp:

- Cá nhân hóa chiến dịch tiếp thị: gửi khuyến mãi hoặc nội dung phù hợp với từng nhóm.
- Tối ưu hóa ngân sách marketing: tập trung vào nhóm khách hàng sinh lời cao.
- Dự đoán hành vi tiêu dùng: từ dữ liệu phân cụm, doanh nghiệp có thể mô hình hóa hành vi tương lai.
- Tăng cường mức độ hài lòng và giữ chân khách hàng: nhờ vào việc hiểu rõ nhu cầu và khả năng chi tiêu.

Ngoài ra, thuật toán K-Means còn có thể mở rộng để ứng dụng trong nhiều ngành khác:

- Trong ngân hàng: phân loại khách hàng theo rủi ro tín dụng.

- Trong thương mại điện tử: gợi ý sản phẩm phù hợp theo nhóm người dùng.
- Trong chăm sóc sức khỏe: nhóm bệnh nhân theo đặc điểm sức khỏe để đưa ra phác đồ điều trị hợp lý.

Từ những ứng dụng thực tiễn trên, có thể khẳng định rằng K-Means là một công cụ mạnh mẽ, đơn giản nhưng hiệu quả, đặc biệt phù hợp với các bài toán phân tích khách hàng quy mô lớn.

## 7.KẾT LUẬN

Trong báo cáo này, chúng tôi đã trình bày quy trình ứng dụng thuật toán K-Means để phân cụm khách hàng dựa trên các đặc trưng như độ tuổi, thu nhập hàng năm và điểm chi tiêu. Kết quả cho thấy việc sử dụng K-Means không chỉ giúp nhận diện các nhóm khách hàng tiềm năng một cách tự động mà còn cung cấp nền tảng dữ liệu quan trọng để doanh nghiệp cá nhân hóa chiến lược marketing, tối ưu hóa trải nghiệm khách hàng và nâng cao hiệu quả kinh doanh.

Mặc dù K-Means là thuật toán đơn giản và dễ triển khai, nó vẫn tồn tại một số hạn chế như sự phụ thuộc vào số lượng cụm (K) được chọn trước và độ nhạy với các giá trị ngoại lai. Trong tương lai, có thể xem xét kết hợp thêm các thuật toán phân cụm khác như DBSCAN hoặc Gaussian Mixture Model để nâng cao độ chính xác và khả năng thích ứng với dữ liệu thực tế phức tạp hơn.

Tóm lại, phân cụm khách hàng bằng K-Means là một bước đi quan trọng trong việc chuyển đổi dữ liệu thô thành thông tin chiến lược, giúp doanh nghiệp hiểu sâu hơn về hành vi tiêu dùng và ra quyết định kinh doanh dựa trên dữ liệu.

## **8. Tài liệu tham khảo**

<https://www.geeksforgeeks.org/k-means-clustering-introduction/>

[https://en.wikipedia.org/wiki/K-means\\_clustering](https://en.wikipedia.org/wiki/K-means_clustering)

<https://medium.com/@cmukesh8688/silhouette-analysis-in-k-means-clustering-cefa9a7ad111>

<https://spark.apache.org/docs/latest/api/python/index.html>

<https://www.interviewbit.com/blog/apache-spark-architecture/>