TRƯỜNG ĐẠI HỌC SÀI GÒN

KHOA CÔNG NGHỆ THÔNG TIN

Tiểu luận học phần Khai phá dữ liệu

TÌM HIỂU THUẬT TOÁN K-MEANS

Sinh viên thực hiện:

Vũ Thị Hồng Xương 3118410492

GVHD: ThS. TRỊNH TẤN ĐẠT

Thành phố Hồ Chí Minh, tháng 5 năm 2022

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Thành phố Hồ Chí Minh, ngày tháng 5 năm 2022

Giảng viên hướng dẫn

MỤC LỤC

DANH MỤC HÌNH ẢNH

Chương 1: Giới thiệu

Với sự phát triển của công nghệ thông tin, lượng thông tin lưu trữ trên các thiết bị điện tử không ngừng tăng lên. Sự tích lũy dữ liệu này xảy ra với một tốc độ bùng nổ. Từ đó rất nhiều vấn đề đặt ra như giải quyết lưu trữ, khai thác dữ liệu. Câu hỏi đặt ra là liệu chúng ta có thể khai thác được gì từ những dữ liệu khổng lồ đó. Từ đó, khai phá dữ liệu ra đời như một hướng giải quyết hữu hiệu cho câu hỏi vừa đặt ra ở trên. Khai phá dữ liệu (datamining) được định nghĩa như là một quá trình chắt lọc hay khai phá tri thức từ một lượng lớn dữ liệu. Trong đó, Phương pháp phân cụm là kỹ thuật rất quan trọng trong khai phá dữ liệu.

K-mean là phương pháp phân cụm phổ biến nhất trong các phương pháp dựa trên phân hoạch (partition-based clustering) của phương pháp phân cụm. Trong bài tiểu luận này tôi sẽ tập trung vào tìm hiểu thuật toán K-means và ứng dụng thuật toán K-means để xây dựng một chương trình phân loại khách hàng đơn giản.

Bài tiểu luận gồm 4 chương và có bố cục như sau:

Chương 1: Giới thiệu

Chương 2: Cơ sở lý thuyết

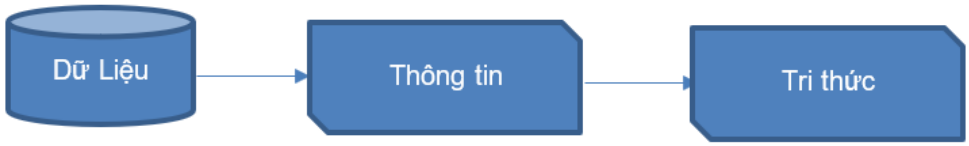
Chương 3: Xây dựng thuật giải chương trình phân loại khách hàng

Chương 4: Tổng kết

Chương 2: cơ sở lý thuyết

2.1. Khai phá dữ liệu (Data mining)

Khai phá dữ liệu (Data mining) là quá trình trích xuất tri thức từ một lượng lớn dữ liệu. Các dữ liệu được trích xuất từ những thông tin ẩn, hữu ích chưa được khám phá, chưa được biết trước từ dữ liệu. Với sự phát triển của công nghệ lưu trữ dữ liệu và sự phát triển của dữ liệu lớn, việc áp dụng các kỹ thuật khai thác dữ liệu đã nhanh chóng tăng tốc trong vài thập kỷ qua, hỗ trợ các công ty bằng cách chuyển đổi dữ liệu thô của họ thành kiến thức hữu ích, cho phép các doanh nghiệp có thể dự đoán được xu hướng tương lai.



Hình 2.1. Khai phá dữ liệu

Các phương pháp trong khai phá dữ liệu: phương pháp luật kết hợp, phương pháp cây quyết định, phương pháp phân cụm, các phương pháp dựa trên mẫu.

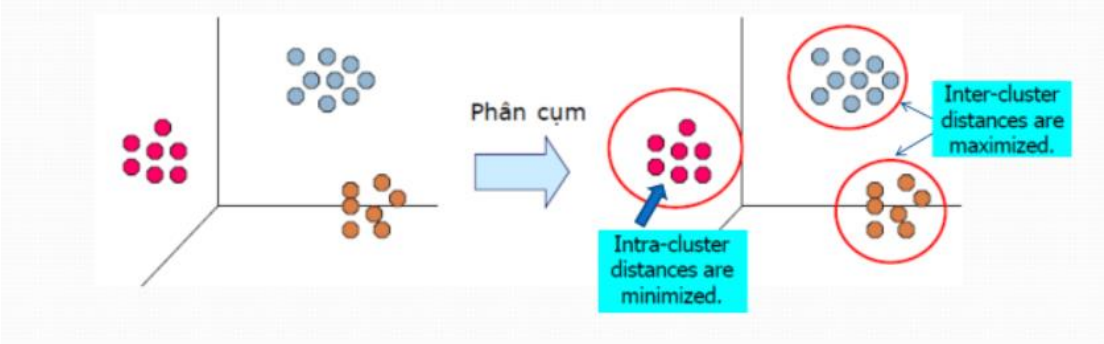
2.2. Tổng quan phương pháp phân cụm (Clustering)

- Phương pháp phân cụm

Phương pháp phân cụm là kỹ thuật rất quan trọng trong khai phá dữ liệu, nó thuộc lớp các phương pháp học không giám sát (Unsupervised Learning) trong máy học (Machine Learning).

Phương pháp phân cụm là quá trình gom nhóm/cụm các đối tượng dữ liệu tương tự nhau vào trong một nhóm/cụm. Mục đích phân cụm là để gom tập các đối tượng

thành các nhóm/cụm có ý nghĩa, những đối tượng bên trong một nhóm/cụm thì tương tự (có liên quan) với nhau và chúng khác (không liên quan) với những đối tượng trong các nhóm khác. Các đối tượng dữ liệu được gom nhóm chỉ dựa trên thông tin được tìm thấy trong dữ liệu mô tả những đối tượng đó hay những mối quan hệ của chúng. Nếu sự tương tự trong một nhóm càng lớn và sự khác nhau giữa các nhóm càng nhiều thì phép gom nhóm càng tốt hơn hay dễ phân biệt hơn.



Hình 2.2. Mô tả quá trình phân cụm

- Các ứng dụng của phương pháp phân cụm

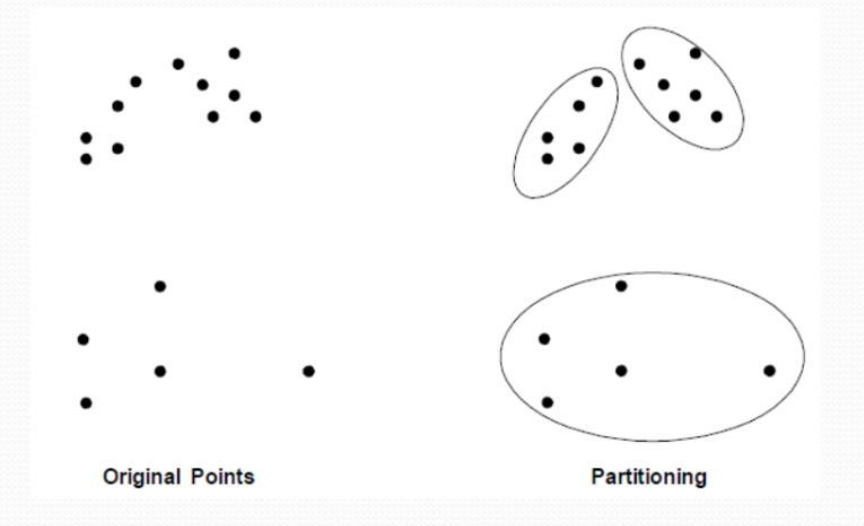
Phương pháp phân cụm có rất nhiều ứng dụng đối với những vấn đề thực tiễn. Một số ứng dụng phổ biến của phân cụm:

- Phân loại khách hàng dựa vào CSDL mua hàng.
- Phân nhóm tài liệu dựa trên các chủ đề.
- Phân đoạn hoặc nén hình ảnh.
- …

- Các phương pháp phân cụm chủ yếu

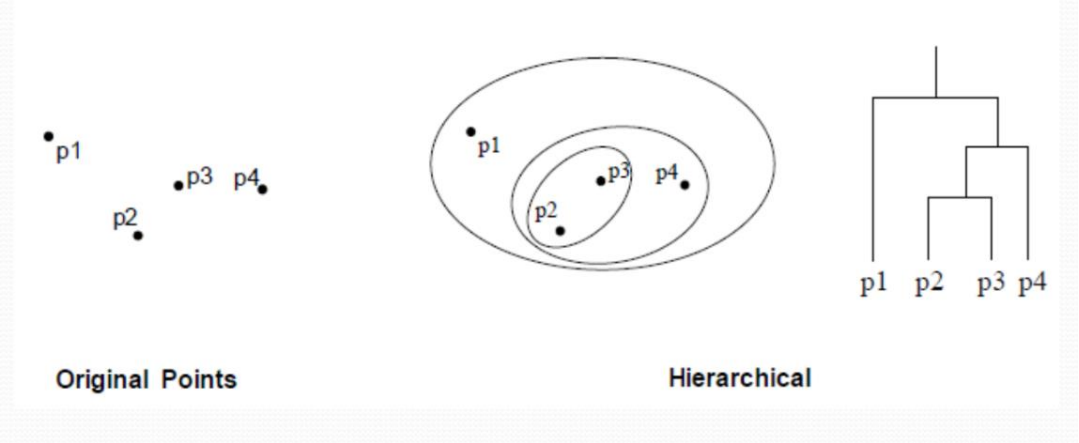
- Phân hoạch (partitioning): phân hoạch tập dữ liệu n phần tử thành k cụm sao cho: mỗi nhóm chứa ít nhất là một đối tượng và mỗi đối tượng thuộc về

đúng một nhóm. Các thuật toán phổ biến trong phương pháp này là: Kmeans, Fuzzy C-mean,...



Hình 2.3. Mô tả phương pháp phân cụm phân hoạch

▫ Phân cấp (hierarchical):xây dựng các nhóm và tổ chức như cây phân cấp. Phân cấp nhóm thường tạo cây các nhóm hay còn được gọi là dendrogram để lưu lại quá trình phân chia nhóm. Các lá của cây biểu diễn các đối tượng riêng lẻ. Các nút trong của cây biểu diễn các nhóm. Phương pháp này ta không cần biết trước số nhóm k và xác định số nhóm cần thiết bằng việc cắt ngang sơ đồ hình cây tại mức thích hợp. Các thuật toán phổ biến trong phương pháp này là: AGNES (Agglomerative NESting), DIANA (Divisive ANAlysis) ,...



Hình 2.4. Mô tả phương pháp phân cụm phân cấp

▫ Dựa trên mật độ (density-based): dựa trên hàm mật độ, số đối tượng lân cận của đối tượng dữ liệu. Các thuật toán phổ biến trong phương pháp này là: DBSCAN, OPTICS, MeanShift ,...

▫ Dựa trên mô hình (model-based): một mô hình giả thuyết được đưa ra cho mỗi cụm; sau đó hiệu chỉnh các thông số để mô hình phù hợp với cụm dữ liệu/đối tượng nhất. Các thuật toán phổ biến trong phương pháp này là: EM, SOMs ,...

▫ Spectral clustering : phân cụm dựa trên đồ thị

▫ ...

2.3.Tổng quan thuật toán K-mean

2.3.1. Giới thiệu

K-mean là phương pháp phân cụm phổ biến nhất trong các phương pháp dựa trên phân hoạch (partition-based clustering). Thuật toán K-means tiêu chuẩn được giới thiệu lần đầu tiên bởi Lloyd vào năm 1957.

Thuật toán K-means sử dụng phương pháp tạo và cập nhật trọng tâm (centroid) để phân nhóm các điểm dữ liệu cho trước vào các nhóm khác nhau và được ứng dụng cho những đối tượng trong một không gian n chiều liên tục.

Đầu tiên chúng sẽ tạo ra các điểm trọng tâm (centroid) ngẫu nhiên. Sau đó gán mỗi điểm trong tập dữ liệu vào điểm trọng tâm gần nó nhất. Sau đó chúng sẽ cập nhật lại trọng tâm và tiếp tục lặp lại các bước đã kể trên. Điều kiện dừng của thuật toán: Khi các trọng tâm không thay đổi trong 2 vòng lặp kế tiếp nhau. Tuy nhiên, việc đạt được 1 kết quả hoàn hảo là rất khó và rất tốn thời gian, vậy nên thường người ta sẽ cho dừng thuật toán khi đạt được 1 kết quả gần đúng và chấp nhận được.

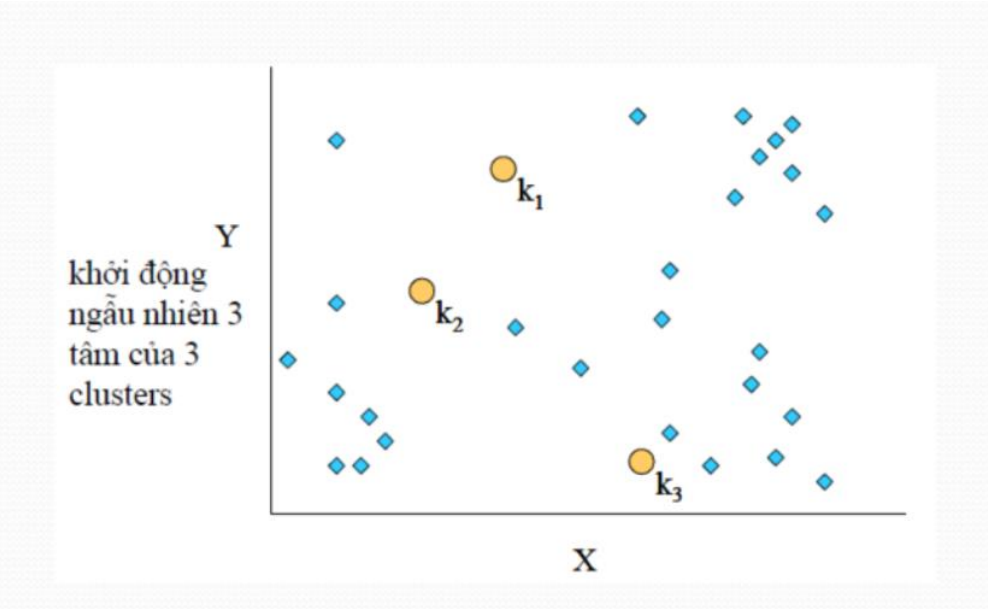
2.3.2. Thuật toán K-means

Giải thuật K-means phân chia tập dữ liệu thành k cụm, mỗi cụm (cluster) có một điểm trung tâm/ trọng tâm, được gọi là centroid. Mỗi đối tượng được gán vào một cụm nếu khoảng cách từ đối tượng đó đến trọng tâm của cụm đang xét là nhỏ nhất. Sau đó cập nhật centroid, ta tính toán lại vị trí của mỗi trọng tâm của mỗi cụm. Ta lặp lại việc gán đối tượng và cập nhật trọng tâm cho đến khi không còn đối tượng nào thay đổi nhóm nữa.

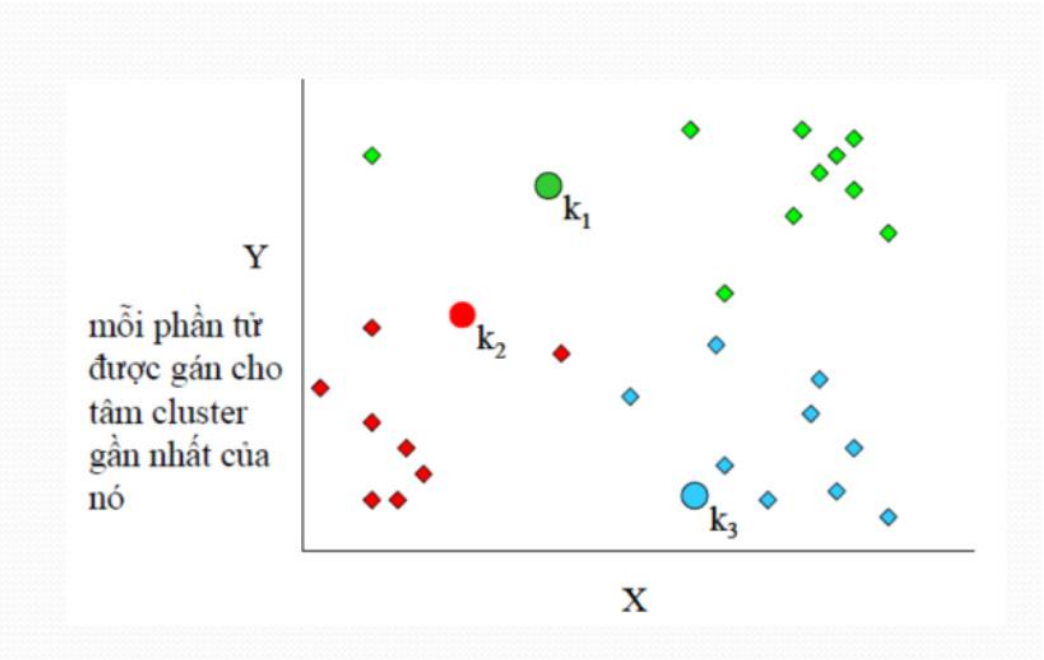
Thuật toán K-Means thực hiện qua các bước chính sau:

1. Chọn ngẫu nhiên K tâm (centroid) cho K cụm (cluster). Mỗi cụm được đại diện bằng các tâm của cụm.
2. Tính khoảng cách giữa các đối tượng (objects) đến K tâm (thường dùng khoảng cách Euclidean)
3. Nhóm các đối tượng vào nhóm gần nhất
4. Xác định lại tâm mới cho các nhóm
5. Thực hiện lại bước 2 cho đến khi không có sự thay đổi nhóm nào của các đối tượng

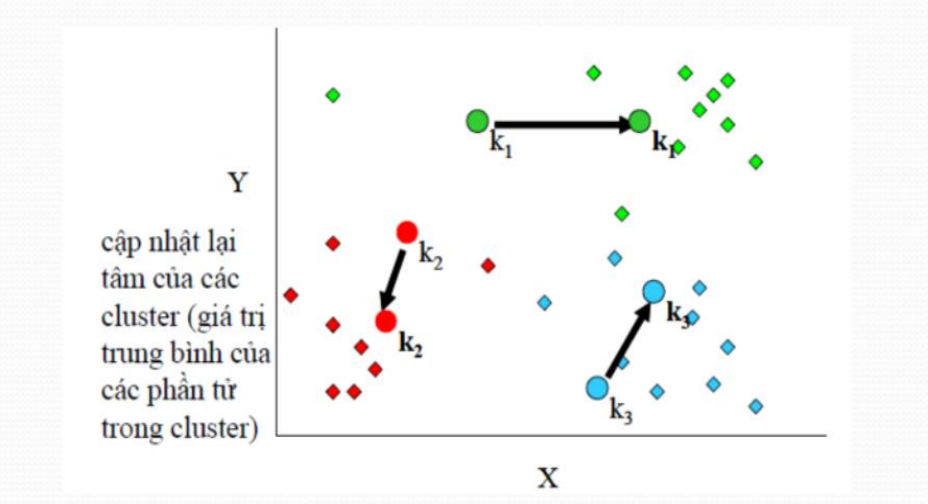
Ta xét mỗi bước trong thuật toán k-means một cách chi tiết hơn như sau



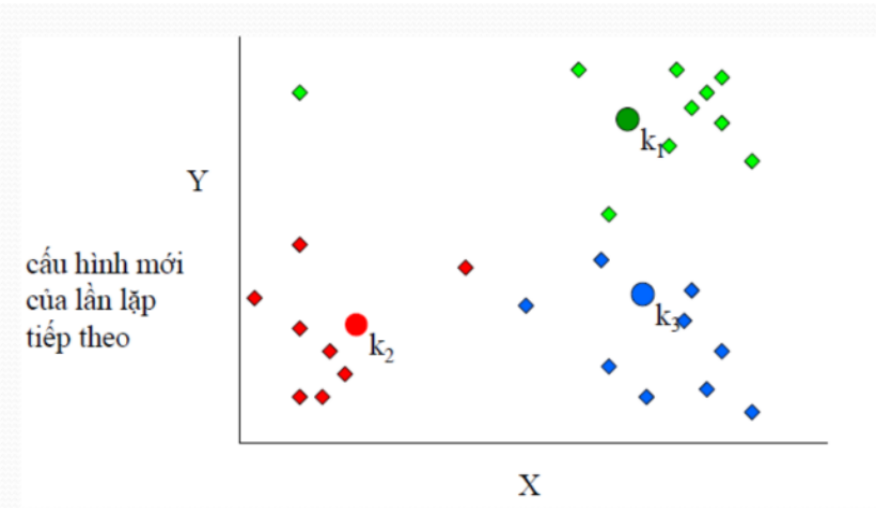
Hình 2.5.a



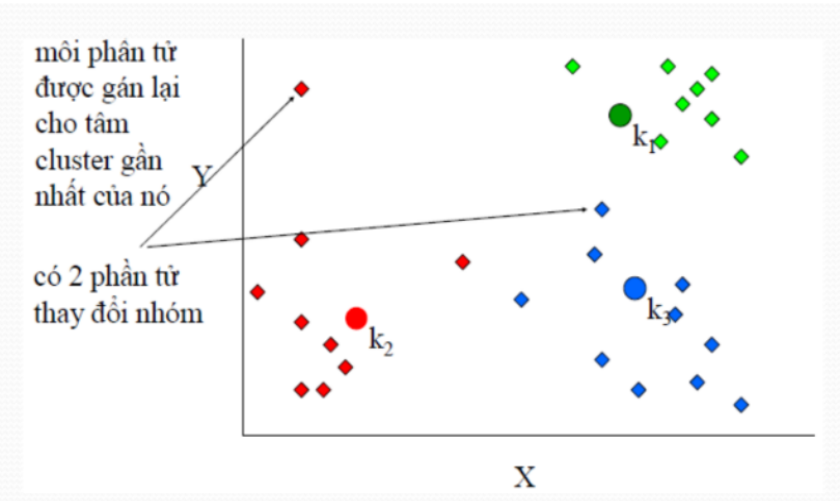
Hình 2.5.b



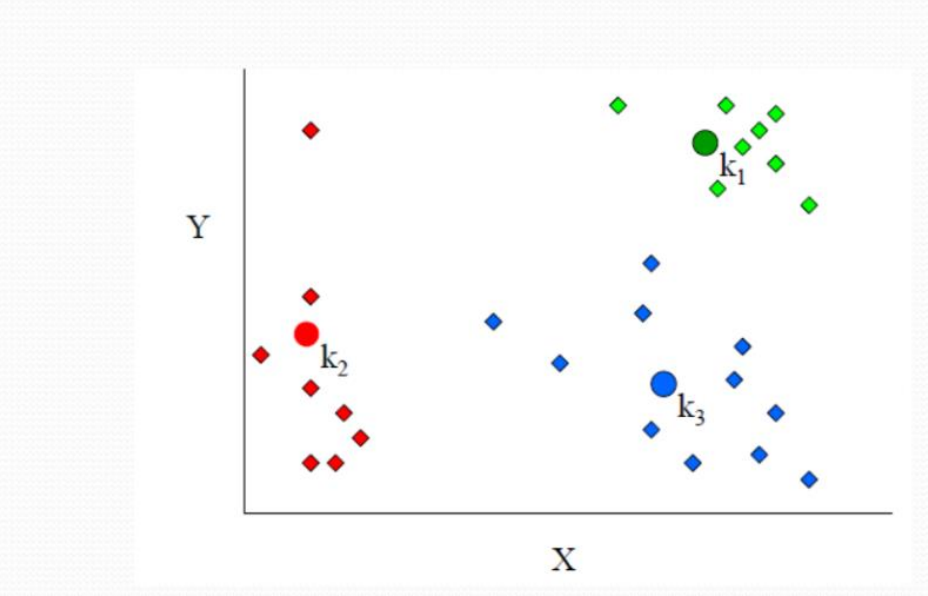
Hình 2.5.c



Hình 2.5.d



Hình 2.5.e



Hình 2.5. Gom tập dữ liệu thành 3 nhóm dùng thuật toán k-means.

Quá trình phâ cụm kết thúc nếu không có (hoặc có không đáng kể) đối tượng nào dịch chuyển từ cụm này sang cụm khác hoặc không có (hoặc có không đáng kể) sự thay đổi các trọng tâm (centroids) của các cụm, hoặc giảm không đáng kể về tổng lỗi phân cụm. Tổng lỗi phân cụm được tính theo công thức:

$$Error = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} d(\mathbf{x}, \mathbf{m_i})^2$$

- C_i: Cụm thứ i
- $\mathbf{m_i}$: Điểm trung tâm (centroid) của cụm C_i
- $d(\mathbf{x}, \mathbf{m_i})$: Khoảng cách (khác biệt) giữa quan sát $\mathbf{x}$ và điểm trung tâm $\mathbf{m_i}$

Hình 2.6. Công thức tính tổng lỗi phân cụm

Ví dụ: **:** Có 1 tập hợp các điểm trên không gian tọa độ Oxy. Mỗi điểm sẽ có tọa độ (x, y) xác định. Bài toán cần giải quyết là chia các điểm này thành K cụm khác nhau phân biệt.

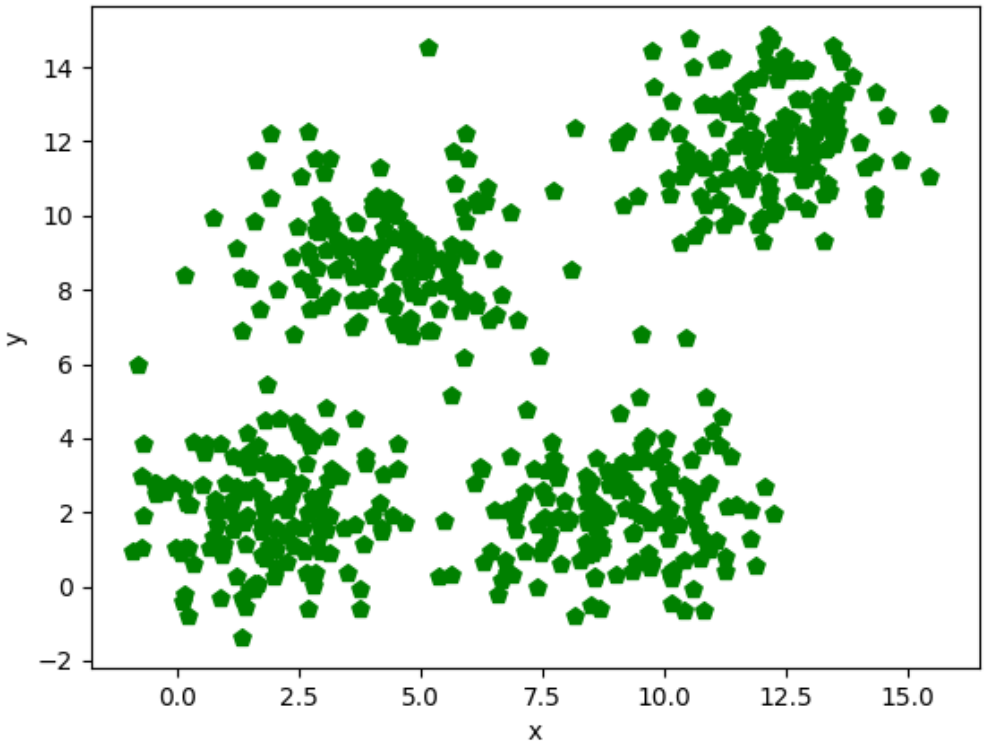
Viết code

Tôi sẽ khởi tạo ngẫu nhiên các điểm dữ liệu. Tuy nhiên, để thể hiện tính chất cụm của dữ liệu, ta sẽ khởi tạo các mẫu dữ liệu này xung quanh 3 tâm cụm (2, 2), (9, 2) và (4,9)

Ứng với mỗi tâm cụm, ta sẽ khởi tạo 150 điểm dữ liệu xung quanh nó. Ở đây lần lượt là X_0, X_1, X_2,X_3

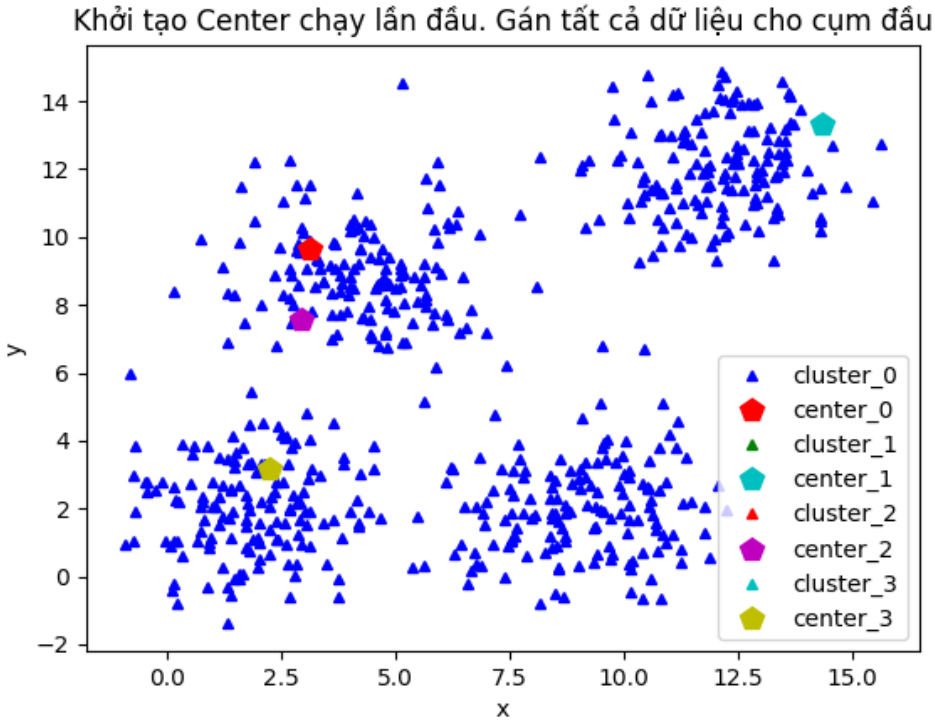
Tôi sẽ chỉ định số tâm cụm cho bài toán này là n_cluster = 4

Với mỗi điểm dữ liệu trong tập dữ liệu, tâm cụm của nó sẽ là 1 trong số n_cluster tâm cụm gần với nó nhất. Việc tính toán khoảng cách giữa 2 điểm trong bài này sử dụng Euclidean distance. Viết hàm khởi tạo n_cluster tâm cụm

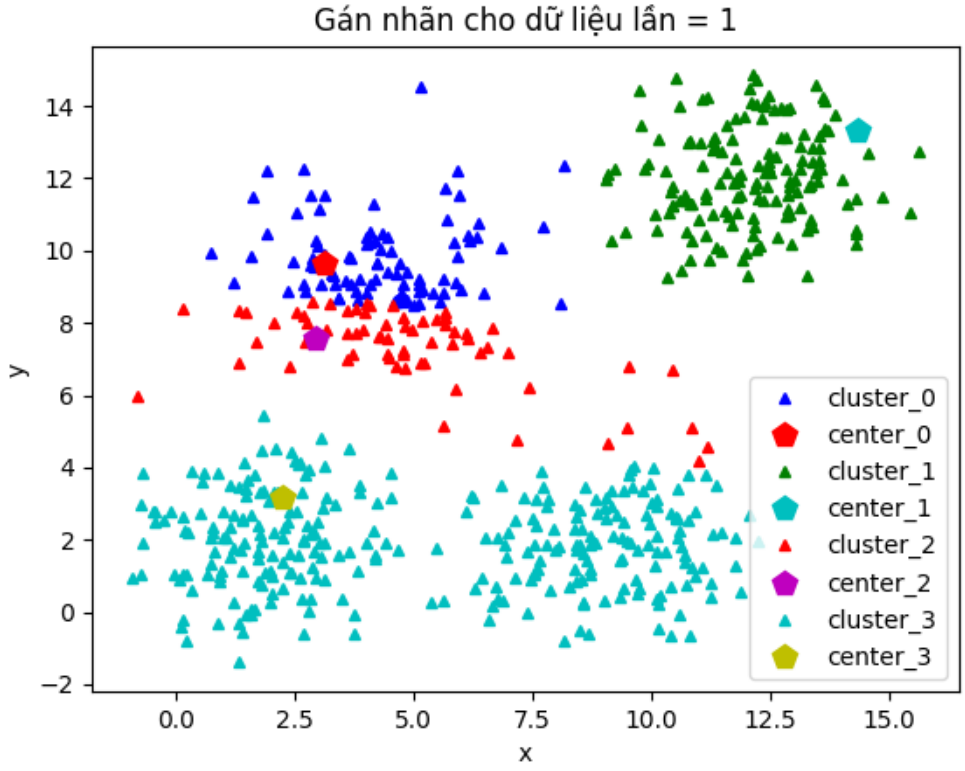


Hình 2.7.1. Dữ liệu mẫu 4 cụm cấp ngẫu nhiên

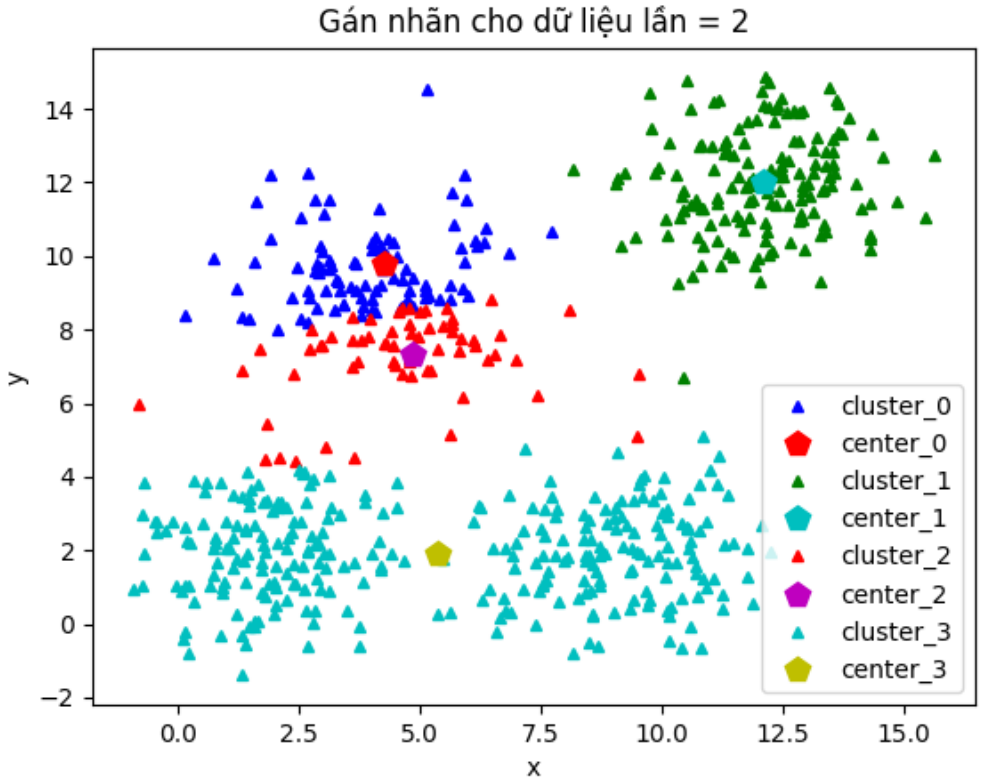
Cập nhật lại vị trí của các tâm cụm cho đến khi mà tâm cụm mới bằng tâm cụm trước đó thì dừng lại. Trong bài ví dụ này bài toán dừng lại sau 9 lần thay đổi tâm cụm.



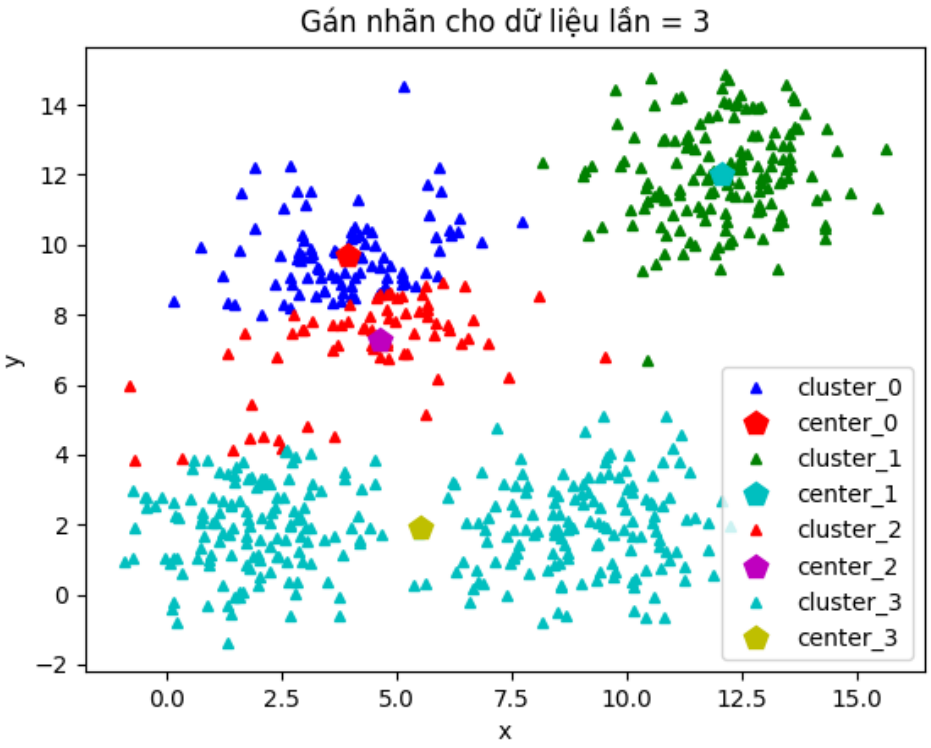
Hình 2.7.2. Khởi tạo center, gán dữ liệu cho cụm



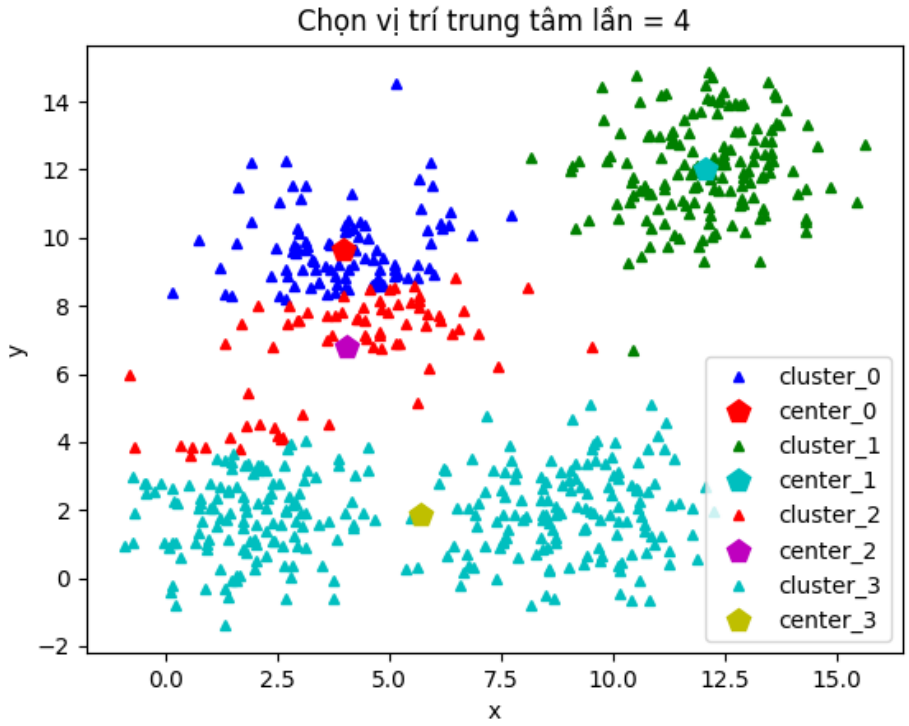
Hình 2.7.3. Gán nhãn cho dữ liệu lần 1



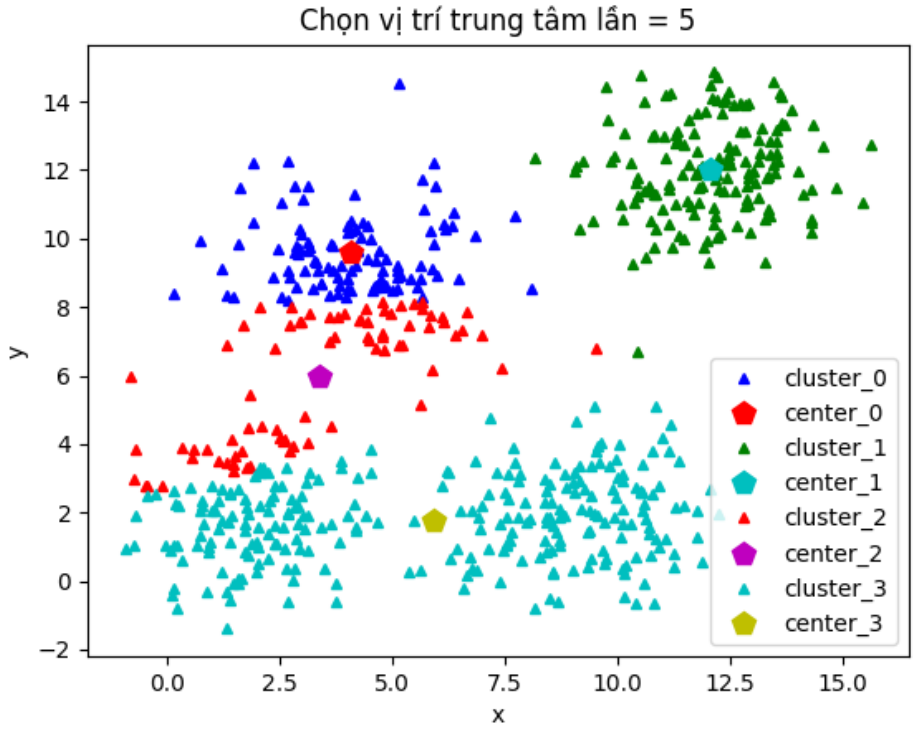
Hình 2.7.4. Gán nhãn cho dữ liệu lần 2



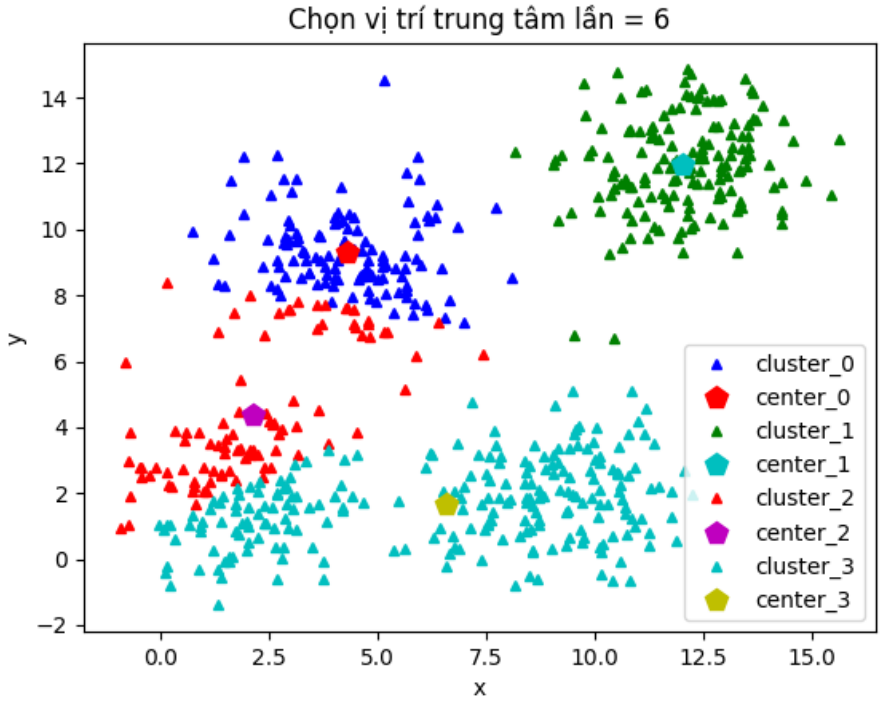
Hình 2.7.5. Gán nhãn cho dữ liệu lần 3



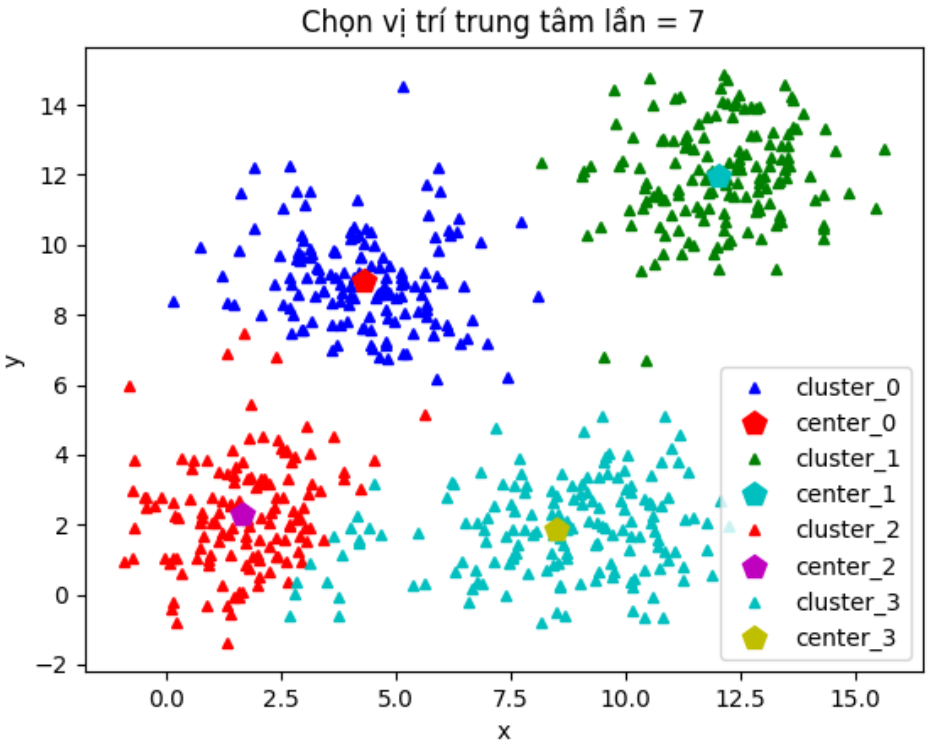
Hình 2.7.6. Gán nhãn cho dữ liệu lần 4



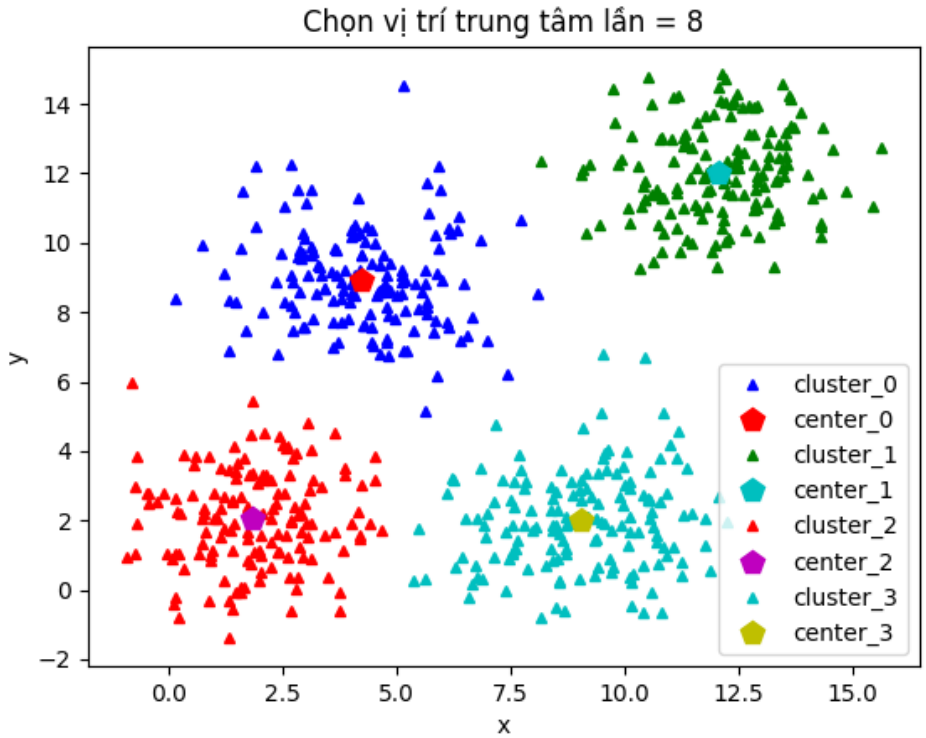
Hình 2.7.7. Gán nhãn cho dữ liệu lần 5



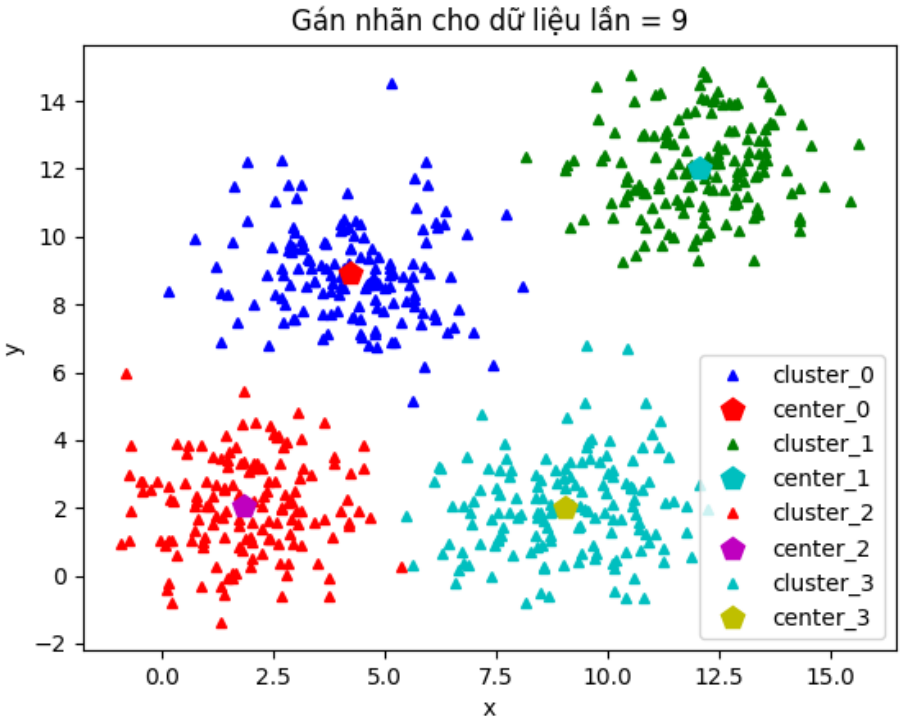
Hình 2.7.8. Gán nhãn cho dữ liệu lần 6



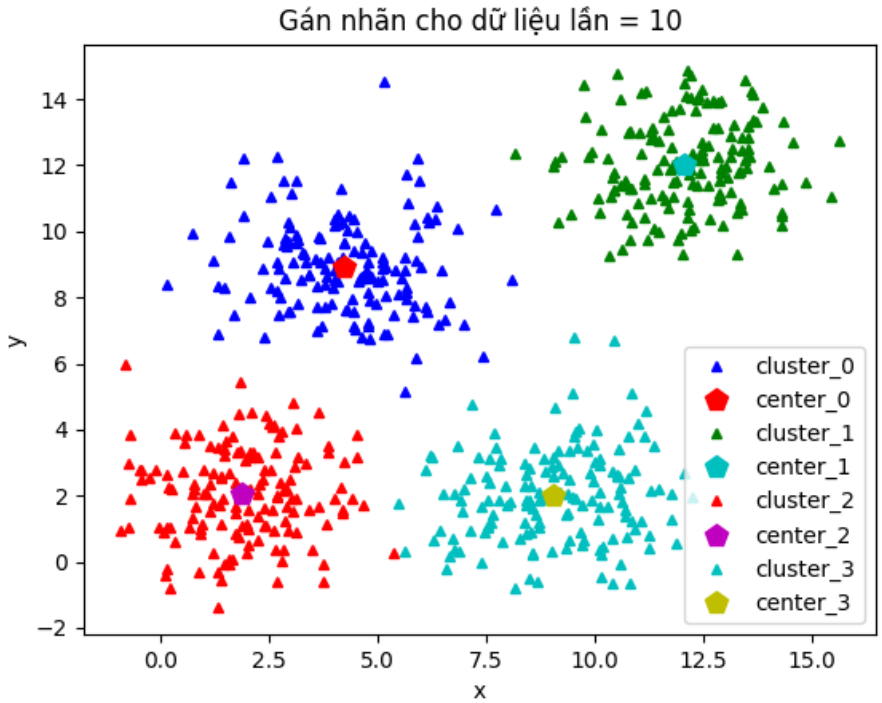
Hình 2.7.9. Gán nhãn cho dữ liệu lần 7



Hình 2.7.10. Gán nhãn cho dữ liệu lần 8



Hình 2.7.11. Gán nhãn cho dữ liệu lần 9



Hình 2.7. Ví dụ minh họa thuật toán K-means

- Gán các đối tượng đến trọng tâm gần nhất

Để gán một đối tượng đến trọng tầm gần nhất, ta cần một phép đo độ lân cận để xác định khái niệm "gần nhất". Một nhóm các phép đo lân cận phổ biến như độ đo khoảng cách cho biến tỷ lệ theo khoảng dùng để xác định sự tương đồng (giống nhau hay khác nhau) giữa hai đối tượng là khoảng cách Minkowski:

$$d(i,j) = \sqrt[q]{(|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + ... + |x_{ip} - x_{jp}|^q)}$$

với $i = (x_{i1}, x_{i2}, ..., x_{ip})$ và $j = (x_{j1}, x_{j2}, ..., x_{jp})$ là các đối tượng dữ liệu p-chiều và q là số nguyên dương.

Nếu q = 1, độ đo khoảng cách là Manhattan:

$$d(i,j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \ldots + |x_{ip} - x_{jp}|$$

Nếu q = 2, độ đo khoảng cách là khoảng cách Euclide:

$$d(i,j) = \sqrt{(|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + ... + |x_{ip} - x_{jp}|^2)}$$

Khoảng cách Euclide (L2) thường được dùng cho những đối tượng trong không gian Euclide. Tuy nhiên, có thể có nhiều phép đo độ lân cận thích hợp cho một nhóm dữ liệu được cho. Chẳng hạn, khoảng cách Manhattan (L1) có thể dùng cho dữ liệu Euclide, trong khi phép đo Jaccard thường dùng cho những tài liệu hay các biến nhị phân. Các kiểu dữ liệu khác nhau, yêu cầu độ đo sự khác nhau cũng khác nhau, các biến tỷ lệ theo khoảng thì dùng khoảng cách Euclide, các biến nhị phân thì dùng hệ số so khớp hay hệ số Jaccard, các biến tên, thứ tự, tỷ lệ thì dùng khoảng cách Minkowski, các dạng hỗn hợp thì dùng công thức trọng lượng. Thông thường, những phép đo lân cận dùng cho k-means thường đơn giản vì thuật toán tính toán lặp đi lặp lại sự lân cận của mỗi điểm với trọng tâm. Tuy nhiên, trong một số trường hợp, khi dữ liệu nằm trong không gian Euclide có số chiều nhỏ, có thể tránh được việc tính toán nhiều sự lân cận, do đó tăng đáng kể tốc độ của thuật toán k-means.

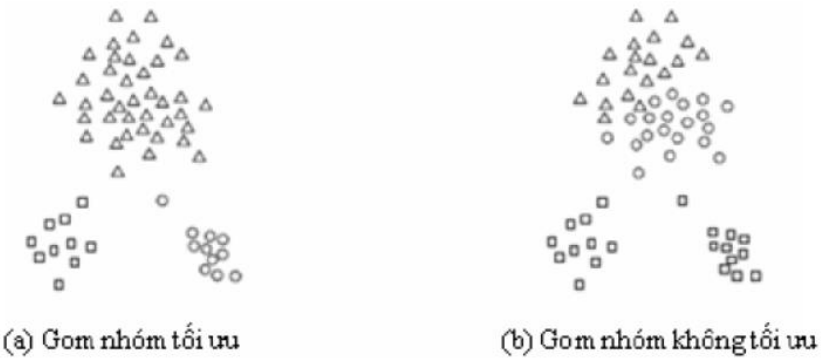
Bisecting k means [15, 22] là một cách tiếp cận khác làm tăng tốc độ của k-means bằng cách giảm số lượng tính toán lân cận.

- Những trọng tâm và hàm mục tiêu

Trong bước 3 của thuật toán k-means, vì trọng tâm có thể thay đổi, tùy thuộc trên phép đo lân cận cho dữ liệu và mục đích của việc gom nhóm. Mục đích của việc gom nhóm được biểu diễn bởi một hàm mục tiêu - hàm dùng để đo chất lượng của phép gom nhóm - tùy thuộc trên độ lân cận của các đối tượng với nhau hay với trọng tâm của nhóm. Chẳng hạn, cực tiểu hóa khoảng cách bình phương của mỗi với trọng tâm gần nhất của nó. Tuy nhiên, điểm quan trọng là một khi ta đã chỉ rõ phép đo lân cận và một hàm mục tiêu, trọng tâm có thể được tính toán.

- Chọn các trọng tâm ban đầu

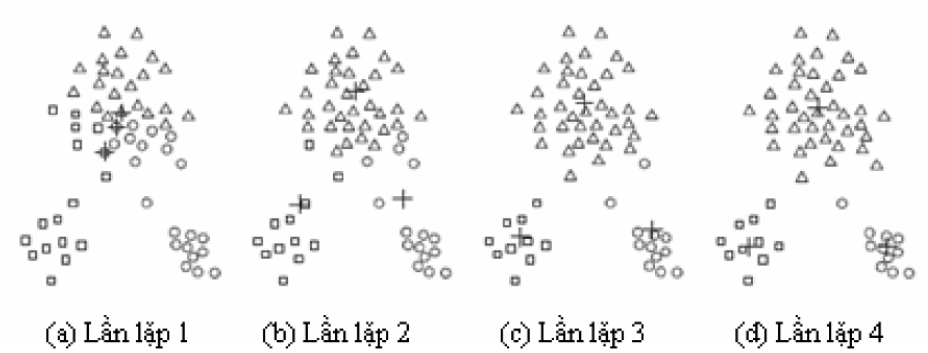
Khi khởi tạo các trọng tâm ngẫu nhiên, các lần chạy khác nhau của k-means sinh ra các tổng SSE khác nhau. Ta minh họa điều này với tập hợp các điểm hai chiều trong hình 2.8 với ba nhóm điểm. Hình 2(a) biểu diễn một cách gom nhóm với cực tiểu toàn cục của SSE cho ba nhóm, trong khi hình 2.8(b) biểu diễn một cách gom nhóm không tối ưu với chỉ một cực tiểu địa phương.



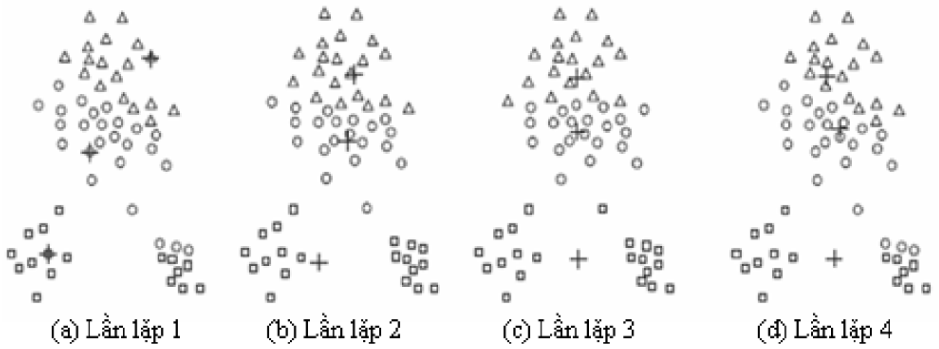
Hình 2.8 – Gom nhóm tối ưu và không tối ưu.

Chọn những trọng tâm ban đầu chính xác là bước quan trọng của thuật toán k-means. Một cách tiếp cận thông thường là chọn những trọng tâm ban đầu một cách ngẫu nhiên, nhưng những kết quả thường xấu. Ta có một ví dụ minh họa điều này

dùng cùng bộ dữ liệu như trong hình 2.9 và hình 2.10. Hình 2.9 và hình 2.10 biểu diễn các nhóm dẫn đến hai lựa chọn đặc biệt những trọng tâm ban đầu. (Ở cả hai hình, vị trí của những trọng tâm nhóm trong những vòng lặp khác nhau được biểu diễn bởi dấu gạch chéo).



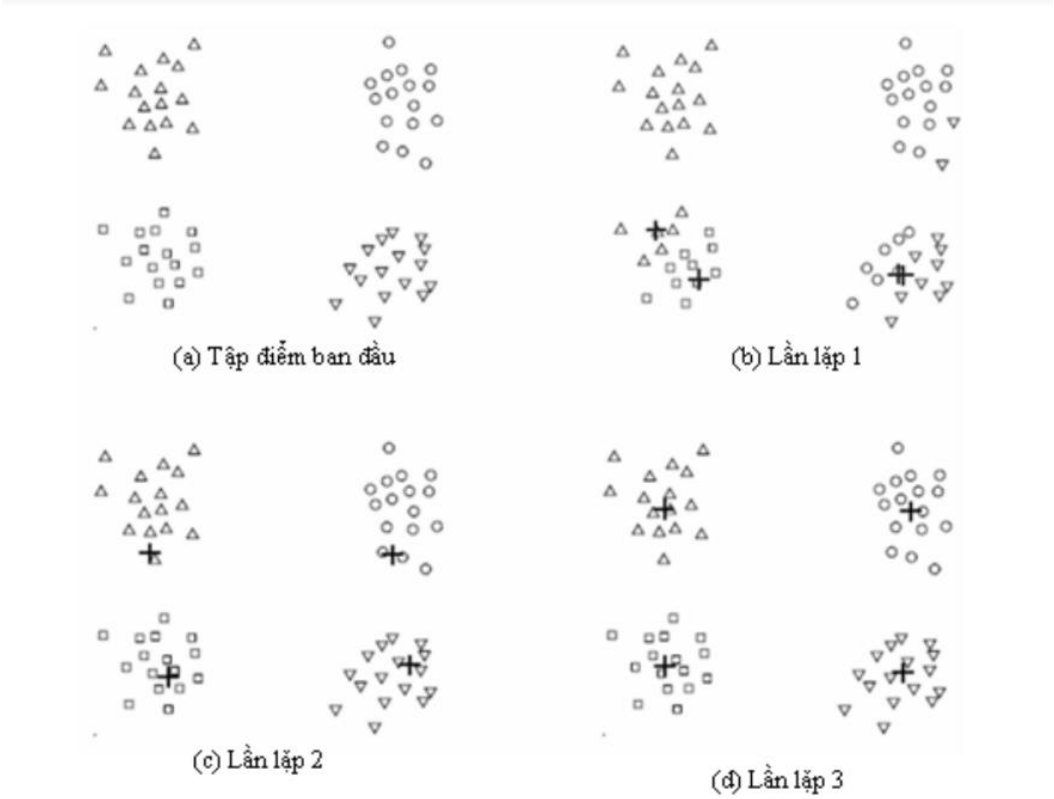
Hình 2.9. Chọn trọng tâm ban đầu tốt.



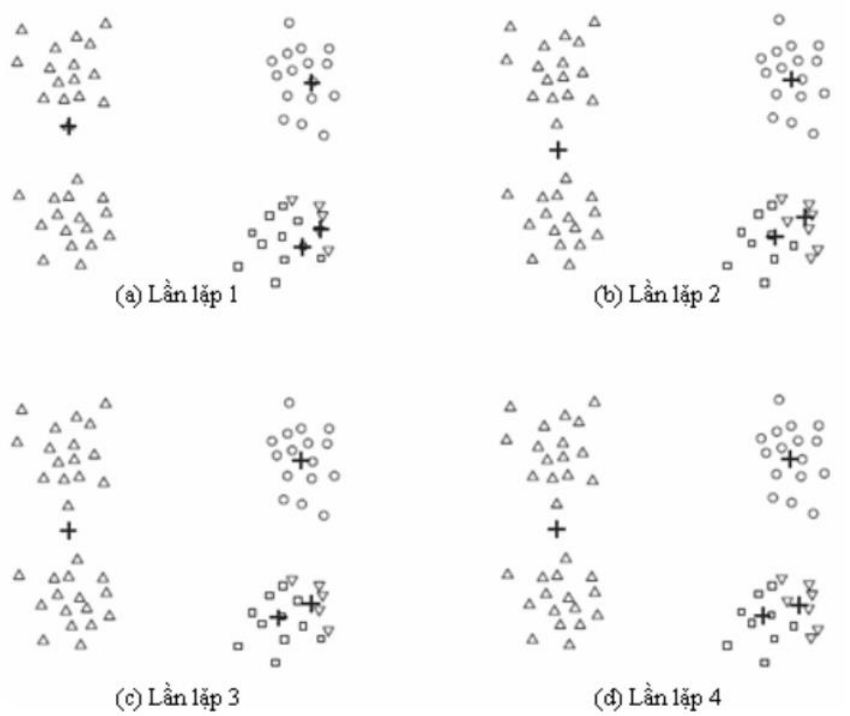
Hình 2.10. Chọn trọng tâm ban đầu không tốt.

Một cách khác để chọn những trọng tâm ban đầu là chọn ngẫu nhiên đầu tiên hay lấy trọng tâm của tất cả các điểm. Sau đó, chọn điểm xa nhất đối với các trọng tâm ban đầu đã chọn. Theo cách này, ta có được một tập các trọng tâm ban đầu được đảm bảo không chỉ được chọn ngẫu nhiên mà còn tách rời nhau. Không may là cách như vậy có thể có chọn những giá trị ngoại lệ hơn là các điểm trong nhóm. Nó cũng

tốn kém để tính toán điểm xa nhất từ tập các trọng tâm ban đầu hiện tại. Để khắc phục vấn đề này, cách này thường dùng cho một bộ mẫu các điểm. Vì những giá trị ngoại lệ hiếm khi xảy ra, chúng có khuynh hướng không xuất hiện trong một mẫu ngẫu nhiên. Ngược lai, các điểm từ mỗi nhóm có thể được bao gồm trừ phi kích thước mẫu rất nhỏ. Việc tính toán liên quan đến việc tìm những trọng tâm ban đầu cũng được giảm đáng kể vì kích thước mẫu nhỏ hơn nhiều so với số điểm.



Hình 2.11. Hai cặp nhóm vói mỗi cặp trọng tâm ban đầu thuộc một nhóm.



Hình 2.12. Hai cặp nhóm, mỗi cặp có ít hơn hoặc nhiều hơn hai trọng tâm ban đầu.

2.3.3. Ưu và khuyết điểm của thuật toán k-means

Thuật toán k-means đơn giản, dễ hiểu, tương đối hiệu quả và có thể dùng cho rất nhiều loại dữ liệu. Nó thật sự hiệu quả thậm chí thực hiện nhiều lần chạy. Một số biến thể bao gồm bisecting k-means thậm chí hiệu quả hơn và ít bị ảnh hưởng hơn trong vấn đề khởi tạo trọng tâm ban đầu. Các đối tượng tự động gán vào các nhóm. Yêu cầu về thời gian thực hiện k-means tương đối - tuyến tính với số đối tượng dữ liệu. Cụ thể thời gian đòi hỏi là O(tkn), với n là số đối tượng, k là số cụm, và t là số lần lặp. Thông thường k, t << n. Thường kết thúc ở điểm tối ưu cục bộ; có thể tìm được tối ưu toàn cục dùng các kỹ thuật như thuật toán di truyền.

Bên cạnh đó, k-means cũng có một số khuyết điểm sau: k-means có thể áp dụng chỉ khi xác định được trị trung bình của các đối tượng, cần chỉ định trước số k (số các nhóm), tất cả các đối tượng phải gán vào các nhóm, phụ thuộc vào việc chọn các nhóm đầu tiên, không phù hợp với tất cả các loại dữ liệu, không thể xử lý dữ liệu chuỗi và outliers - các đối tương bất tương tự với phần dữ liệu còn lại, không phù

hợp để khám phá các nhóm với dạng không lồi, nhóm có kích thước và mật độ khác nhau.

- **Khắc phục hạn chế**: để khắc phục yếu tố chọn K cho thuật toán K-means ta dùng Elbow hoặc Silhouette để chọn K

  Vậy thuật toán của bài toán áp dụng trở thành:

1. Chọn K tâm (centroid) cho K cụm (cluster) bằng phương thức Elbow hoặc Silhouette
2. Tính khoảng cách giữa các đối tượng (objects) đến K tâm (thường dùng khoảng cách Euclidean)
3. Nhóm các đối tượng vào nhóm gần nhất
4. Xác định lại tâm mới cho các nhóm
5. Thực hiện lại bước 2 cho đến khi không có sự thay đổi nhóm nào của các đối tượng

Chương 3: Xây dựng thuật giải chương trình phân loại khách hàng

3.1. Xây dựng thuật giải

Trong thực tế, các khách hàng của một cửa hàng thường cho thấy xu hướng phân thành các nhóm khác nhau, với mỗi nhóm có một xu hướng mua sắm, thói quen, thị yếu khác nhau. Việc phát hiện ra các nhóm phân khúc khách hàng đóng một vai trò vô cùng quan trọng đối với nhà bán lẻ vì nó giúp họ xây dựng các chiến lược kinh doanh, tiếp thị đặc thù cho từng nhóm, từ đó nâng cao chất lượng phục vụ khách hàng, qua đó làm tăng lợi nhuận bán lẻ.

Trong chương này chúng tôi sẽ trình bày phương pháp cài đặt thực nghiệm, cách chương trình ứng dụng và các kết quả đạt được của bài toán phân loại khách hàng áp dụng thuật toán K-means.

Dựa trên tập dữ liệu đầu vào có chứa thông tin chi tiết về mua sắm của khách hàng từ các thành phố khác nhau để phân cụm phân tách các dữ liệu tương tự với nhau. Bộ dữ liệu dựa trên chi tiết mua sắm của khách hàng ở một số thành phố. Dữ liệu đã được điền một cách ngẫu nhiên và không có mối liên hệ nào với bất kỳ dữ liệu hệ thống thực hoặc dữ liệu bí mật nào. Nó có 8 thuộc tính với 200 dữ liệu khách hàng. Các thuộc tính là:

- CustomerID (mã khách hàng)
- CustomerGender (giới tính)
- CustomerAge (tuổi)
- CustomerCity (thành phố)
- AnnualIncome (thu nhập)
- CreditScore (điểm tín dụng)
- SpendingScore (điểm chi tiêu)
- CustomerCityID (mã thành phố)

Đầu tiên ta cần nhập các thư viện bắt buộc:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.cluster import KMeans
from sklearn.metrics import accuracy_score,confusion_matrix
import warnings
warnings.filterwarnings("ignore")
df1 = pd.read_csv('./Shopping_CustomerData.csv')
df1.head()
```

	Customer ID	Customer Gender	Customer Age	Customer City	AnnualIncome	Credit Score	Spending Score	Customer CityID
0	1001	Male	49	Bengaluru	527547.58850	653	78	1
1	1002	Male	59	Bengaluru	207143.19760	630	63	1
2	1003	Female	54	Delhi	164423.84570	555	69	4
3	1004	Female	42	Bengaluru	56220.36443	699	30	1
4	1005	Female	30	Bengaluru	256194.36190	793	6	1

Tôi sẽ lấy hai thuộc tính và 400 dữ liệu hàng đầu của mỗi thuộc tính để dễ hình dung các bước.

df_new = df1[["CustomerAge","SpendingScore"]]

df_new.head()

	CustomerAge	SpendScore
0	49	78
1	59	63
2	54	69
3	42	30
4	30	6

df_new.describe()

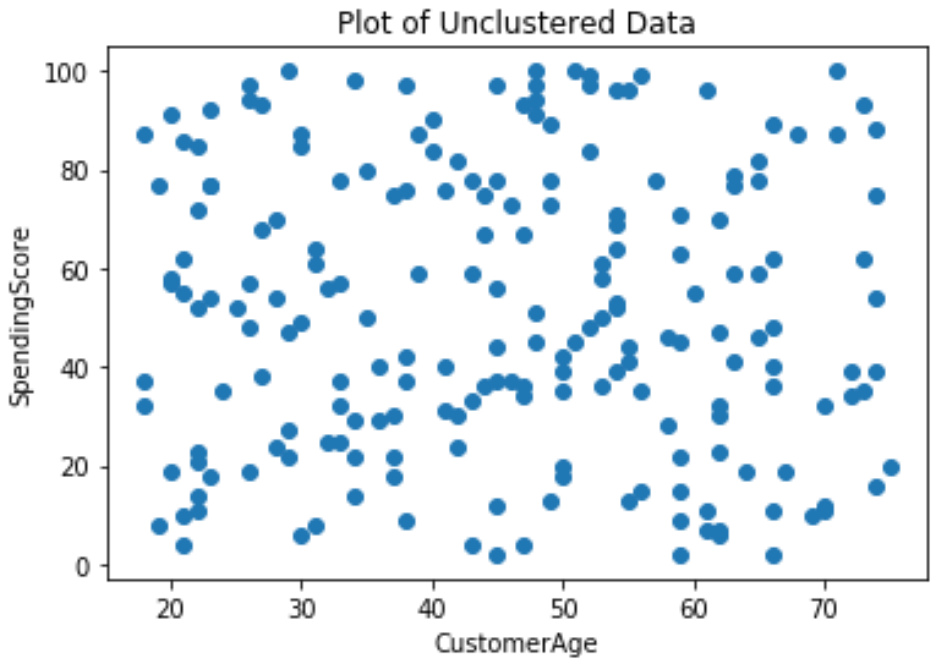
	CustomerAge	SpendingScore
count	200.000000	200.00000
mean	45.520000	50.70500
std	16.113592	28.72269
min	18.000000	2.00000
25%	31.750000	27.75000
50%	46.500000	48.00000
75%	59.000000	77.00000
max	75.000000	100.00000

Kiểm tra giá trị null

```
df_new.isnull().sum()
```

Vẽ biểu đồ 2 biến CustomerAge và SpendingScore

```
plt.scatter(df_new.iloc[:,0],df_new.iloc[:,1])
plt.xlabel('CustomerAge')
plt.ylabel('SpendingScore')
plt.title('Plot of Unclustered Data')
plt.show()
```



Biểu đồ trên biểu diễn dữ liệu khách hàng dựa trên 2 thuộc tính tuổi và điểm tiêu dùng và dữ liệu trên chưa đc phân nhóm.

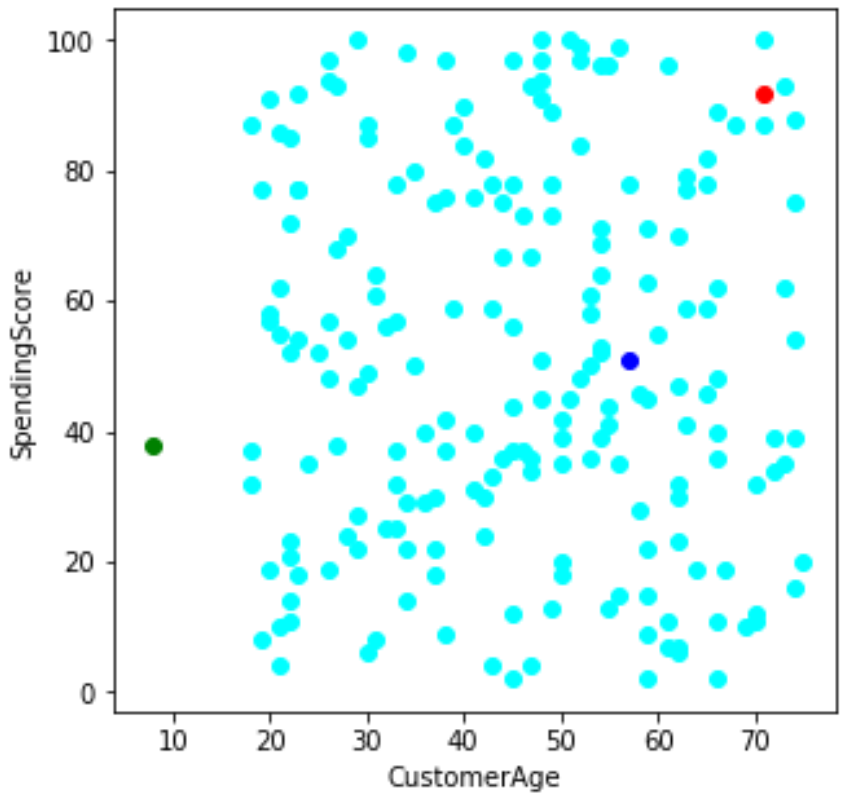
Triển khai thuật toán K-means từ Scratch.

Thuật toán sẽ dừng lại khi trọng tâm của các cụm không thay đổi sau nhiều lần lặp.

```
#Step-1
k = 3#gia su gia tri cua k la 3
np.random.seed(40)

#chon ngau nhien 2 trong tam
random_centroids=[]
for i in range(1,k+1):
  random_centroids.append([np.random.randint(1,100),
np.random.randint(1,100)])#chon cac gia tri ngau nhien tu 1-100
print('Randomly selected points as random_centroids:',random_centroids)
```

Các điểm trọng tâm được chọn ngẫu nhiên là : [[71, 92], [8, 38], [57, 51]]

```
#ve bieu do trong tam da chon
plt.figure(figsize=(5, 5))
plt.scatter(df_new.iloc[:,0], df_new.iloc[:,1], color='cyan')
length = len(random_centroids)
colors1=['r','g','b','cyan','yellow','black']
for i in range(length):
  plt.scatter(*random_centroids[i],color=colors1[i])
plt.xlabel('CustomerAge')
plt.ylabel('SpendingScore')
plt.show()
```



#Step-2

#tinh toan cac diem du lieu gan trong tam

```
 def assignment(df_new,random_centroids):

  for i in range(length) :
    df_new['Distance from Centroid {}'.format(i)]=(np.sqrt((df_new.iloc[:,0] -
random_centroids[i][0]) ** 2 + (df_new.iloc[:,1] - random_centroids[i][1]) ** 2))

  list1=[]
  list2=[]
  for a,b,c in zip(df_new['Distance from Centroid 0'],df_new['Distance from
Centroid 1'],df_new['Distance from Centroid 2']):
    d = min(a,b,c)

    if d == a:
      list1.append(0)
```

```
            list2.append('r')
        elif d == b:
            list1.append(1)
            list2.append('g')
        else:
            list1.append(2)
            list2.append('b')
    df_new['Closest_Centroid'] = list1
    df_new['Color']=list2
    return df_new
```

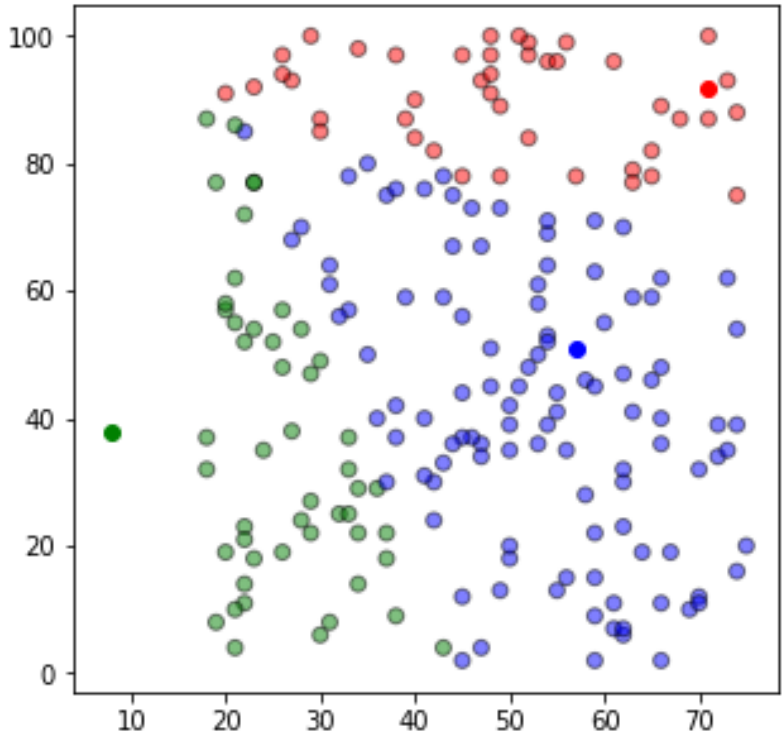
Bảng một số số liệu khoảng cách của các điểm dữ liệu tới trọng tâm

```
df_new = assignment(df_new, random_centroids)
df_new.head()
```

	Customer Age	**Spending Score**	**Distance from Centroid 0**	**Distance from Centroid 1**	**Distance from Centroid 2**	**Closest Centroid**	**Color**
0	49	78	26.076810	57.280014	28.160256	0	r
1	59	63	31.384710	56,797887	12.165525	2	b
2	54	69	28.600699	55.470713	18.248288	2	b
3	42	30	68.447060	34,928498	25,806976	2	b
4	30	6	95,273291	38.832976	52.478567	1	g

```
#ve cac cum
plt.figure(figsize=(5,5))
```

```
plt.scatter(df_new.iloc[:,0],df_new.iloc[:,1],color =df_new['Color'],alpha =
0.5,edgecolor = 'k')
for i in range(length):
  plt.scatter(*random_centroids[i],color=colors1[i])
```



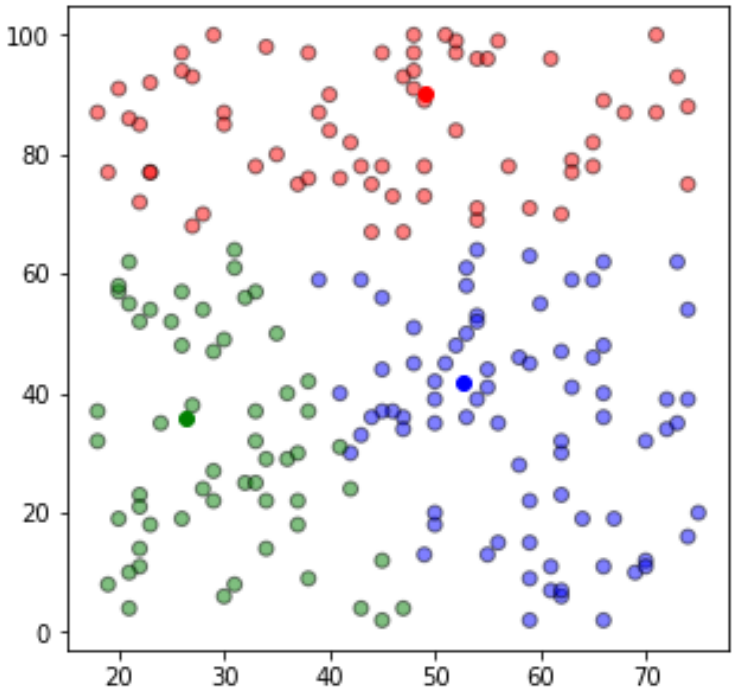
Cập nhật các trọng tâm

```
#Step-3

def update(parameter):
    for i in range(length):
        random_centroids[i][0] = np.mean(df_new[df_new['Closest_Centroid'] ==
i]['CustomerAge'])
        random_centroids[i][1] = np.mean(df_new[df_new['Closest_Centroid'] ==
i]['SpendingScore'])
    return parameter
random_centroids = update(random_centroids)
random_centroids
```

Các trọng tâm mới cập nhật có tọa độ là : [[49.16279069767442,
90.09302325581395],
[26.387755102040817, 35.97959183673469],
[52.75, 41.7037037037037]]

Cập nhật lại trọng tâm

```
df_new = assignment(df_new, random_centroids)
df_new.head()
plt.figure(figsize=(5, 5))
plt.scatter(df_new.iloc[:,0], df_new.iloc[:,1], color=df_new['Color'], alpha=0.5,
edgecolor='k')
for i in range(length):
  plt.scatter(*random_centroids[i], color=colors1[i])
plt.show()
```



Ta lặp lại việc gán đối tượng và cập nhật trọng tâm cho đến khi không còn đối tượng nào thay đổi nhóm nữa.

```
#Step-4

count=1
while True:
  old_random_centroids = np.round(random_centroids)
  print('Old Centroid',old_random_centroids)
  count+=1
  random_centroids = update(random_centroids)

  new_random_centroids = np.round(random_centroids)
  print('New Centroid',new_random_centroids)

  df_new = assignment(df_new, random_centroids)
  result=np.allclose(old_random_centroids,new_random_centroids)#np.allclose()
function checks if two arrays are equal element-wise
  print(result)
  if result == True:
    break

print(count)
```

Những lần cập nhật trọng tâm

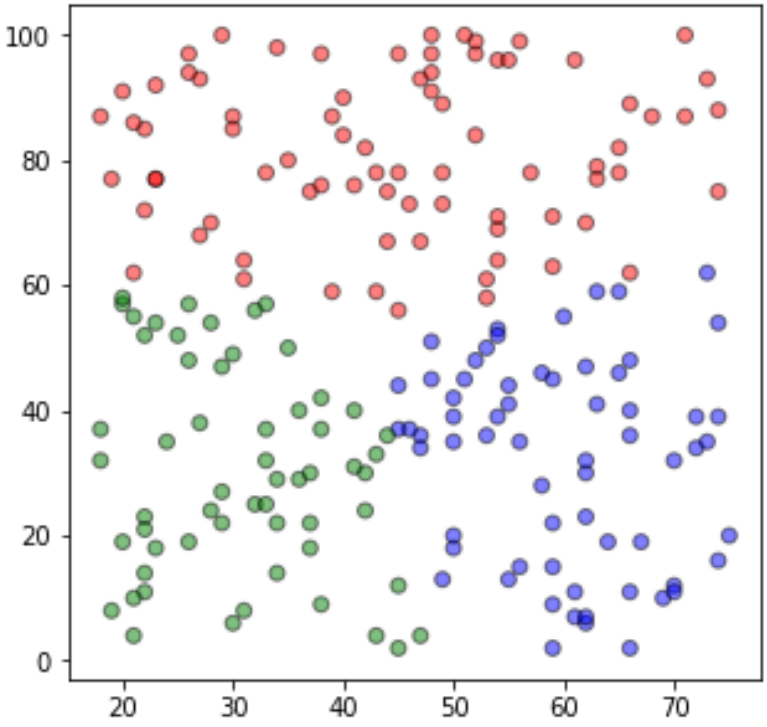
```
Old Centroid [[49. 90.]
 [26. 36.]
 [53. 42.]]
New Centroid [[45. 85.]
 [30. 32.]
 [58. 35.]]
False
Old Centroid [[45. 85.]
 [30. 32.]
 [58. 35.]]
```

New Centroid [[45. 82.]
 [30. 31.]
 [59. 33.]]
False
Old Centroid [[45. 82.]
 [30. 31.]
 [59. 33.]]
New Centroid [[45. 82.]
 [31. 30.]
 [60. 32.]]
False
Old Centroid [[45. 82.]
 [31. 30.]
 [60. 32.]]
New Centroid [[45. 81.]
 [31. 30.]
 [60. 32.]]
False
Old Centroid [[45. 81.]
 [31. 30.]
 [60. 32.]]
New Centroid [[45. 81.]
 [31. 30.]
 [60. 32.]]
True
6

Vẽ các cụm

```
plt.figure(figsize=(5, 5))
plt.scatter(df_new.iloc[:,0], df_new.iloc[:,1], color=df_new['Color'], alpha=0.5,
edgecolor='k')
```

plt.show()



Triển khai K-Means bằng cách sử dụng scikit learning

df = df1[["CustomerAge","SpendingScore"]]

df.head()

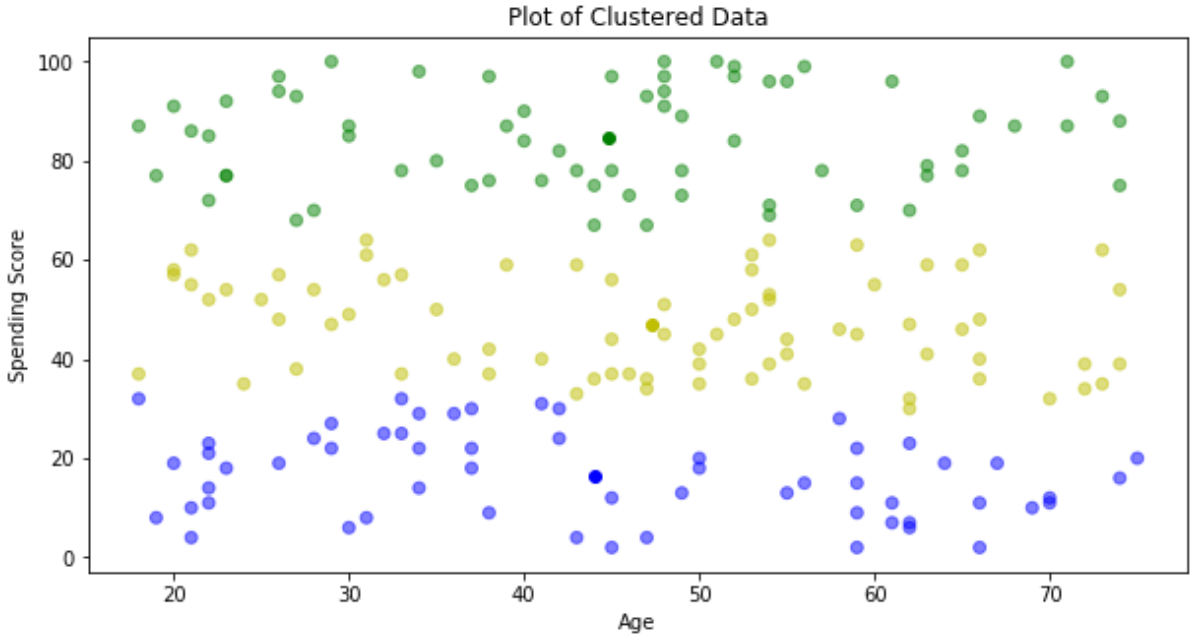
	CustomerAge	SpendScore
0	49	78
1	59	63
2	54	69
3	42	30
4	30	6

```
#gia su gia tri k la 3
kmeans = KMeans(n_clusters=3)#Creating a K-Means Object
kmeans.fit(df)#Fitting the Model
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
#tao label cho tung diem
labels = kmeans.predict(df)
labels
array([1, 0, 1, 2, 2, 1, 2, 1, 2, 1, 2, 0, 0, 2, 0, 0, 2, 0, 1, 1, 1, 0,
       1, 0, 1, 1, 0, 0, 2, 0, 2, 1, 1, 2, 0, 0, 1, 1, 1, 1, 2, 1, 0, 2,
       1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 2, 2, 0, 2, 0, 0, 0, 2,
       1, 0, 0, 1, 0, 2, 1, 2, 0, 1, 1, 2, 2, 0, 0, 0, 1, 0, 2, 0, 1, 0,
       2, 0, 0, 0, 2, 0, 0, 0, 2, 1, 1, 2, 2, 2, 2, 0, 0, 0, 0, 1, 0, 1,
       2, 1, 0, 2, 1, 2, 2, 2, 1, 2, 1, 1, 2, 1, 2, 0, 0, 1, 1, 1, 0, 2,
       1, 1, 0, 1, 1, 0, 2, 2, 2, 2, 2, 2, 1, 2, 0, 2, 2, 1, 1, 0, 1, 0,
       0, 0, 2, 1, 0, 1, 0, 1, 2, 1, 2, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 2,
       1, 2, 2, 0, 0, 1, 1, 0, 2, 0, 1, 2, 0, 2, 1, 0, 2, 0, 1, 0, 0, 0,
       2, 2], dtype=int32)
#in cac trong tam cua moi cum
centroids = kmeans.cluster_centers_
centroids
array([[47.26666667, 46.82666667],
       [44.82089552, 84.65671642],
       [44.06896552, 16.5       ]])
```

Phân nhóm các dữ liệu

```
plt.figure(figsize=(10, 5))
colmap = {1:'y',2:'g',3:'b',4:'r',5:'c'}
colors = map(lambda x: colmap[x+1], labels)
print(colors)
colors1=list(colors)


plt.scatter(df['CustomerAge'], df['SpendingScore'], color=colors1, alpha=0.5)
for idx, centroid in enumerate(centroids):
```

```
    plt.scatter(*centroid, color=colmap[idx+1])
plt.xlabel('Age')
plt.ylabel('Spending Score')
plt.title('Plot of Clustered Data')
plt.show()
```



3.2. Chương trình chạy thực nghiệm trên Python:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.cluster import KMeans
from sklearn.metrics import accuracy_score, confusion_matrix
import warnings

warnings.filterwarnings("ignore")

df1 = pd.read_csv('./Shopping_CustomerData.csv')
df1.head()
```

```
df_new = df1[["CustomerAge", "SpendingScore"]]
df_new.head()

df_new.describe()

# Checking for Null Values
df_new.isnull().sum()

plt.scatter(df_new.iloc[:, 0], df_new.iloc[:, 1])
plt.xlabel('CustomerAge')
plt.ylabel('SpendingScore')
plt.title('Plot of Unclustered Data')
plt.show()

# Step-1
k = 3
np.random.seed(40)

# Step-2
random_centroids = []
for i in range(1, k + 1):
    random_centroids.append([np.random.randint(1, 100), np.random.randint(1,
                                                        100)])
print('Randomly selected points as random_centroids:', random_centroids)

plt.figure(figsize=(5, 5))
plt.scatter(df_new.iloc[:, 0], df_new.iloc[:, 1], color='cyan')
length = len(random_centroids)
colors1 = ['r', 'g', 'b', 'cyan', 'yellow', 'black']
for i in range(length):
    plt.scatter(*random_centroids[i], color=colors1[i])
plt.xlabel('CustomerAge')
plt.ylabel('SpendingScore')
plt.show()
```

```
# Step-3

def assignment(df_new, random_centroids):
    for i in range(length):
        df_new['Distance from Centroid {}'.format(i)] = (np.sqrt(
            (df_new.iloc[:, 0] - random_centroids[i][0]) ** 2 + (df_new.iloc[:, 1] -
random_centroids[i][1]) ** 2))

    list1 = []
    list2 = []
    for a, b, c in zip(df_new['Distance from Centroid 0'], df_new['Distance from Centroid 1'],
                       df_new['Distance from Centroid 2']):
        d = min(a, b, c)

        if d == a:
            list1.append(0)
            list2.append('r')
        elif d == b:
            list1.append(1)
            list2.append('g')
        else:
            list1.append(2)
            list2.append('b')
    df_new['Closest_Centroid'] = list1
    df_new['Color'] = list2
    return df_new
    df_new = assignment(df_new, random_centroids)
    df_new.head()
    # Plotting the clusters


plt.figure(figsize=(5, 5))
plt.scatter(df_new.iloc[:, 0], df_new.iloc[:, 1], color=df_new['Color'], alpha=0.5, edgecolor='k')
for i in range(length):
    plt.scatter(*random_centroids[i], color=colors1[i])
```

```
# Step-4

def update(parameter):
    for i in range(length):
        random_centroids[i][0] = np.mean(df_new[df_new['Closest_Centroid'] == i]['CustomerAge'])
        random_centroids[i][1] = np.mean(df_new[df_new['Closest_Centroid'] ==
i]['SpendingScore'])
    return parameter


random_centroids = update(random_centroids)
random_centroids

df_new = assignment(df_new, random_centroids)
df_new.head()
plt.figure(figsize=(5, 5))
plt.scatter(df_new.iloc[:, 0], df_new.iloc[:, 1], color=df_new['Color'], alpha=0.5, edgecolor='k')
for i in range(length):
    plt.scatter(*random_centroids[i], color=colors1[i])
plt.show()

# Step-5

count = 1
while True:
    old_random_centroids = np.round(random_centroids)
    print('Old Centroid', old_random_centroids)
    count += 1
    random_centroids = update(random_centroids)

    new_random_centroids = np.round(random_centroids)
    print('New Centroid', new_random_centroids)

    df_new = assignment(df_new, random_centroids)
    result = np.allclose(old_random_centroids,
```

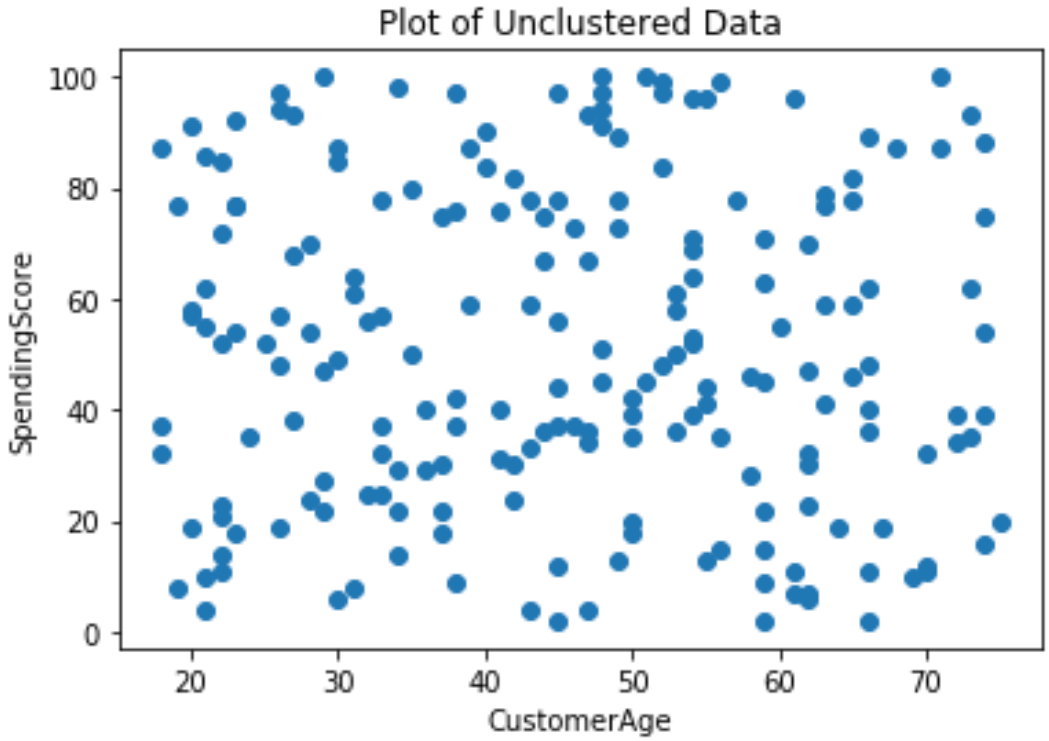
```
                    new_random_centroids)
    print(result)
    if result == True:
        break

print(count)

plt.figure(figsize=(5, 5))
plt.scatter(df_new.iloc[:, 0], df_new.iloc[:, 1], color=df_new['Color'], alpha=0.5, edgecolor='k')
plt.show()

df = df1[["CustomerAge", "SpendingScore"]]
df.head()

kmeans = KMeans(n_clusters=3)  # Creating a K-Means Object
kmeans.fit(df)  # Fitting the Model

labels = kmeans.predict(df)
labels

centroids = kmeans.cluster_centers_
centroids

kmeans.inertia_

plt.figure(figsize=(10, 5))
colmap = {1: 'y', 2: 'g', 3: 'b', 4: 'r', 5: 'c'}
colors = map(lambda x: colmap[x + 1], labels)
print(colors)
colors1 = list(colors)

plt.scatter(df['CustomerAge'], df['SpendingScore'], color=colors1, alpha=0.5)
for idx, centroid in enumerate(centroids):
    plt.scatter(*centroid, color=colmap[idx + 1])
plt.xlabel('Age')
plt.ylabel('Spending Score')
```

```
plt.title('Plot of Clustered Data')
plt.show()

inertia_list = []
for num_clusters in np.arange(1, 21):
    kmeans = KMeans(n_clusters=num_clusters)
    kmeans.fit(df)
    inertia_list.append(kmeans.inertia_)

inertia_list

plt.figure(figsize=(10, 5))
plt.plot(np.arange(1, 21), inertia_list)
plt.grid(True)
plt.xlabel('Values of K')
plt.ylabel('Inertia')
plt.title('Elbow Curve')
plt.show()
```

Chương 4: Tổng kết

4.1. Kết quả

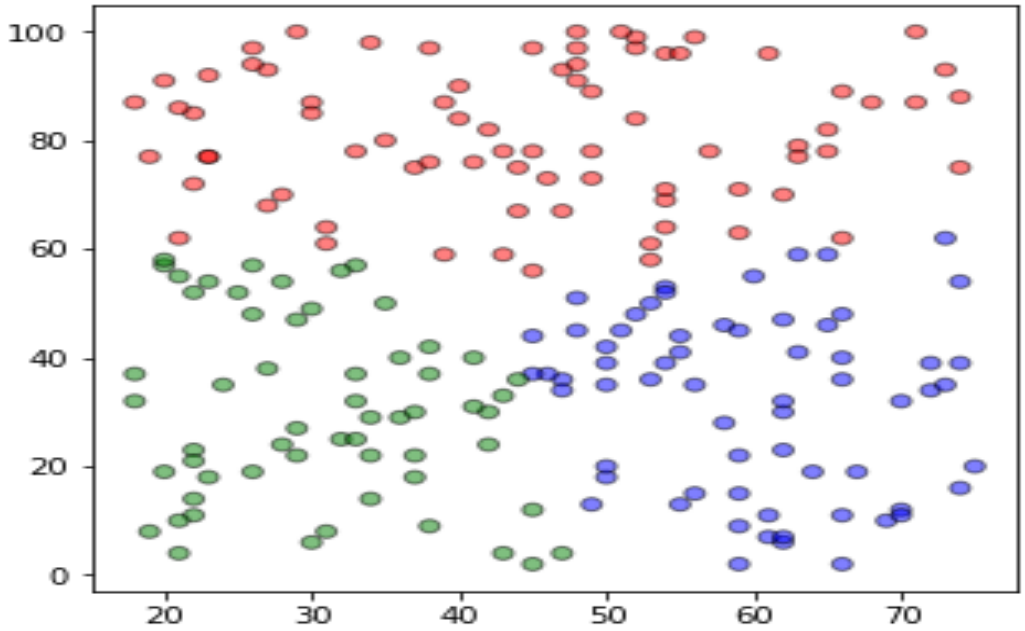
Ứng dụng phân loại khách hàng dựa trên tập dữ liệu đầu vào có chứa thông tin chi tiết về mua sắm của khách hàng từ các thành phố khác nhau có giá trị đầu vào được hình dung theo bảng sau, dữ liệu khách hàng dựa trên 2 thuộc tính tuổi và điểm tiêu dùng:



Kết quả khi chạy chương trình phân loại khách hàng trên python:

- Khi triển khai thuật toán K-means từ Scratch.

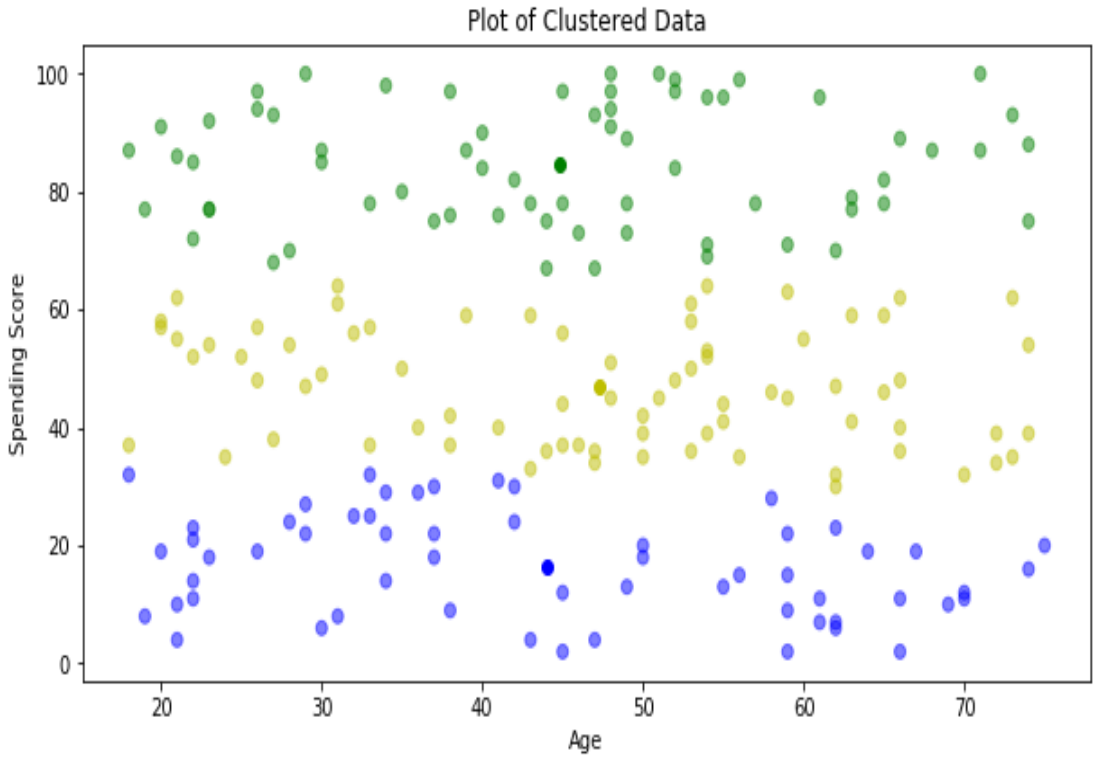
Với giá trị k là 3 thuật toán K-means từ thư viện Scratch phân loại khách hàng thành 3 nhóm như sau:



Hình 4.1. Kết quả phân nhóm K-means từ Scratch.

- Khi triển khai K-Means bằng cách sử dụng scikit learning:

Với giá trị k là 3 và dựa vào tổng bình phương của các điểm dữ liệu đến trọng tâm, bảng dữ liệu khách hàng được chia thành 3 nhóm khách hàng như sau:



Hình 4.2. Kết quả phân nhóm K-Means bằng cách sử dụng scikit learning.

4.2. Đánh giá

Trong đề tài này, trên cơ sở tìm hiểu thuật thoán K-means tôi đã áp dụng thuật toán để xây dựng một ứng dụng phân loại khách hàng đơn giản.

Ứng dụng phân cụm khách hàng theo hai tiêu trên tập dữ liệu đầu vào có chứa thông tin chi tiết về mua sắm của khách hàng, tùy vào mục đích và các chiến lược bán hàng, giúp doanh nghiệp cung cấp sản phẩm/dịch vụ đến đúng đối tượng khách hàng.

Thuật toán K-means đơn giản, dễ hiểu, tương đối hiệu quả và có thể dùng cho rất nhiều loại dữ liệu, phù hợp để giải quyết vấn đề liên quan trong bài toán phân nhóm khách hàng.

Ngoài ra việc áp dụng thuật toán K-means để xây dựng bài toán phân loại khách hàng có những khuyết điểm sau: cần phải chỉ định k nhóm, phụ thuộc vào các điểm trọng tâm được khởi tạo ngẫu nhiên, nên các nhóm phân cụm có thể sẽ không tối ưu và không phù hợp với mục đích phân loại.

4.3. Kết luận

Trong bài tiểu luận tôi đã trình bày các khái niệm và các bước cơ bản để xây dựng mô hình K-means để phân nhóm khách hàng. Tiểu luận tập trung tìm hiểu thuật toán K-means. Tôi đã tìm hiểu về cơ sở lý thuyết, nội dung cách tiếp cận, các vấn đề có liên quan. Thuật toán K-means đơn giản, dễ cài đặt và ứng dụng rộng rãi trong thực tiễn cuộc sống, góp phần quan trọng trong ứng dụng công nghệ thông tin vào đời sống.

Các kết quả đã đạt được trong luận văn:

- Trình bày tổng quan về khai phá văn bản
- Tìm hiểu cơ sở lý thuyết của thuật toán K-means.
- Trình bày các bước thực hiện thuật toán K-means.

Định hướng phát triển tiếp theo của tiểu luận: tìm hiểu, nghiên cứu khai thác rộng và sâu hơn các tri thức về lý thuyết khai phá văn bản đặc biệt là lĩnh vực khai phá dữ liệu để có thể vận dụng vào thực tiễn chính xác hơn.

Thử nghiệm và đánh giá kỹ hơn với các dữ liệu thuộc các lĩnh vực khác nhau.

Sau khi tìm hiểu và giải bài toán bằng thuật toán K-means bản thân tôi đã có thêm kiến thức về cách giải bài toán với thuật toán K-means, cách nghiên cứu, tìm tài liệu, được chuyên gia góp ý xây dựng để hoàn thành đề tài tìm hiểu. Qua đó tôi nhận thấy vấn đề khai thác dữ liệu với K-means ứng dụng rất nhiều trong thực tiễn, thúc đẩy tôi tích cực học tập và nghiên cứu nhiều hơn. Mở rộng kiến thức cho tôi trong quá trình học tôi chân thành cảm ơn Thầy **Trịnh Tấn Đạt** đã nhiệt tình hướng dẫn và giúp đỡ tôi trong suốt thời gian qua.

Tài liệu tham khảo

[1] Phân cụm K-Means (K-Means clustering) (https://viblo.asia/p/phan-cum-k-means-k-means-clustering-1VgZvX325Aw) truy cập lúc 8h ngày 23/04/2022

[2] Pang-Ning Tan, Michael Steinbach and Vipin Kumar, "*Introduction to Data Mining*", Chapter 8 - Cluster Analysis: Basic Concepts and Algorithms, Pearson Addison-Wesley, pp. 488-568, 2006.

[3] Phân cụm k-means (https://vi.wikipedia.org/wiki/Ph%C3%A2n_c%E1%BB%A5m_k-means) truy cập lúc 8h ngày 20/4/2022.

[4] Jain AK, Dubes RC (1988) Algorithms for clustering data. Prentice-Hall, Englewood Cliffs.

[5] Nguyễn Văn Chức - Thuật toán K-Means với bài toán phân cụm dữ liệu, BIS 2010 (http://bis.net.vn/forums/t/374.aspx). Truy cập lúc 8h Ngày 07/07/2021

[6] J.Han, M. Kamber and A.K.H. Tung, "*Spatial Clustering Methods in Data Mining"*, Sciences and Engineering Research Council of Canadia.

[7] IEEE, "*IEEE Transactions on Pattern Analysis and Machine Intelligence", chapter 3* - An efficient k-means clustering algorithm: analysis and implementation, pages 881-892, 2007.

[8] James Theiler and Galen Gisler, "*A contiguity-enhanced k-means clustering algorithm for unsupervised multispectral image segmentation*", pages 108--118, 1997.