Dbscan

Quản trị điều hành (Trường Đại học Kinh tế Thành phố Hồ Chí Minh)

BỘ GIÁO DỤC VÀ ĐÀO TẠO

ĐẠI HỌC KINH TẾ HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN KINH DOANH

ĐỒ ÁN CUỐI KỲ

MÔN: MÁY HỌC

Đề tài:

THUẬT TOÁN DBSCAN

Nhóm 9

Thành viên: Nguyễn Thị Thu Trang

Lê Minh Triều

Nguyễn Ngọc Phương Trinh

Đỗ Nguyễn Thiên Trúc

Trần Duy Tuấn

Giảng viên: TS. Nguyễn An Tế

TP. Hồ Chí Minh , ngày 12 tháng 12 năm 2023

MỤC LỤC

DANH MỤC HÌNH ẢNH

LỜI NÓI ĐẦU

Lời đầu tiên, tác giả xin gửi lời cảm ơn đến trường Đại Học UEH đã đưa bộ môn Máy học vào trong chương trình giảng dạy. Đặc biệt, tác giả xin trình bày tỏ lòng biết ơn sâu sắc đến thầy Nguyễn An Tế – Giảng viên trường Đại Học UEH, người đã giảng dạy môn Máy học cho lớp DS001 một cách tận tình, nhiệt huyết và truyền đạt cho lớp những kiến thức quý báu trong suốt thời gian vừa qua.

Thời gian tham dự lớp học của thầy đã giúp tác giả bổ sung nhiều kiến thức bổ ích, điều đó đã góp phần không nhỏ vào sự thành công của bài tiểu luận cuối kỳ này.

Bộ môn Máy học là một môn học thú vị, vô cùng bổ ích đối với mỗi sinh viên ngành Khoa học dữ liệu. Tuy nhiên lượng kiến thức và thời gian còn nhiều hạn chế nên trong quá trình làm bài khó tránh được mắc phải nhiều sai sót, kính mong thầy xem xét và góp ý để giúp bài tiểu luận của tác giả được hoàn thiện hơn.

Xin chân thành cảm ơn!

Nhóm 9

BẢNG PHÂN CÔNG

STT	Thành viên	Phân công	Đánh giá
1	Nguyễn Thị Thu Trang	Chương 1, Chương 2 (2.2, 2.3)	100%
2	Lê Minh Triều	Chương 3, Chương 4	100%
3	Nguyễn Ngọc Phương Trinh	Chương 2 (2.1, 2.2, 2.4), Chương 3 (nhận xét)	100%
4	Đỗ Nguyễn Thiên Trúc	Chương 4	100%
5	Trần Duy Tuấn	Chương 2 (2.1, 2.2, 2.3)	100%

CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI

1.1 Giới thiệu đề tài

Clustering (phân cụm) là một kỹ thuật quan trọng trong lĩnh vực học máy và khám phá dữ liệu, giúp đem lại nhiều lợi ích khi các nhà chuyên gia cần tìm kiếm tri thức từ dữ liệu. Đặc biệt, trong lĩnh vực khoa học dữ liệu, các thuật toán phân cụm là một công cụ quan trọng giúp tách biệt và hiểu rõ các nhóm dữ liệu khác nhau. Trong đồ án này, nhóm tập trung nghiên cứu thuật toán DBSCAN (Density-Based Spatial Clustering of Applications with Noise) - một phương pháp phân cụm dựa trên mật độ.

1.2 Mục tiêu nghiên cứu

Nhóm sẽ so sánh sự khác biệt giữa thuật toán này với các thuật toán không giám sát khác như k-Means và Hierarchical Clustering. Sau đó, nhóm sẽ áp dụng thuật toán DBSCAN vào một bộ dữ liệu cụ thể để phân cụm dữ liệu.

1.3 Phương pháp nghiên cứu

Sử dụng thuật toán DBSCAN để phân cụm các khối nhà trong bộ dữ liệu California housing.

1.4 Tài nguyên sử dụng

Ngôn ngữ lập trình phân tích dữ liệu: Python

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 Tổng quan

2.1.1 Thuật toán phân cụm

Thuật toán phân cụm (clustering) là một kỹ thuật học máy không giám sát, được sử dụng để phân chia một tập dữ liệu thành các nhóm dựa trên sự tương đồng giữa các đối tượng trong cùng một nhóm. Trong thuật toán phân cụm, người dùng không cần phải cung cấp nhãn lớp cho các đối tượng trong tập dữ liệu, mà chỉ cần cung cấp một số thông tin về các đặc tính của các đối tượng.

Các phương pháp gom cụm phổ biến là:

Cách tiếp cận	**Mô tả**	**Phương pháp điển hình**
Dựa trên phân hoạch (Partitioning approach)	Xây dựng nhiều phân hoạch và chọn cách tốt nhất (sai số tối thiểu)	k-Means, k-Medoids, Fuzzy C-Means
Dựa trên phân cấp (Hierarchical approach)	Xây dựng cây phân rã tập dữ liệu theo một số tiêu chí	Diana, Agnes, BIRCH, CAMELEON
Dựa trên mật độ (Density-based approach)	Dựa trên mật độ kết nối	DBSCAN, OPTICS, DenClue
Dựa trên lưới (Grid-based approach)	Dựa trên cấu trúc của lưới	STING, WaveCluster, CLIQUE
Dựa trên mô hình (Model-based approach)	Xác định mô hình cho mỗi cluster	EM, SOM, COBWEB

2.1.2 Thuật toán DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) là một thuật toán phân cụm phi tham số dựa trên mật độ. Thuật toán này được đề xuất bởi Martin Ester, Hans-Peter Kriegel, Jörg Sander và Xiaowei Xu vào năm 1996. Thuật toán hình thành các cụm bằng cách xác định các điểm trung tâm (có đủ số lượng điểm lân cận) và mở rộng chúng để tiếp cận các điểm láng giềng. Các điểm không thuộc về bất kỳ cụm nào được phân loại là nhiễu.

2.1.3 Tại sao cần sử dụng thuật toán DBSCAN?

Phân cụm là kỹ thuật học không giám sát dùng để nhóm các điểm dữ liệu dựa trên các đặc điểm cụ thể. Có nhiều thuật toán phân cụm khác nhau trong đó K - means và HAC là các thuật toán được sử dụng phổ biến nhất. Tuy nhiên, khi áp dụng các thuật toán để tạo ra các cụm có hình dạng tùy ý, hoặc cụm trong cụm thì 2 thuật không mang lại kết quả tốt. Điều này có nghĩa là các phần tử trong cụm không thật sự tương đồng hoặc hiệu suất phân cụm không cao.

Phân cụm Dựa trên Mật độ, như DBSCAN, lại tiếp cận vấn đề theo cách khác. Thuật toán này xác định các khu vực có mật độ điểm dữ liệu cao, cách biệt với các khu vực mật độ thấp xung quanh. Mật độ ở đây được hiểu là số lượng điểm dữ liệu nằm trong một bán kính nhất định. Nhờ vậy, DBSCAN có thể phát hiện các cụm có hình dạng đặc biệt và cụm con bên trong một cụm lớn, điều mà các kỹ thuật truyền thống không làm được. Đây là lý do tại sao DBSCAN trở nên hữu ích trong việc phân cụm dữ liệu phức tạp.

2.1.4 So sánh DBSCAN và K-means

DBSCAN	K-means
Không cần xác định trước số cụm.	Cần thiết phải xác định cụ thể số cụm.
Có thể tách biệt được các dữ liệu nhiễu mà không cần xử lý trước đó.	Không hoạt động tốt với các dữ liệu có outlier. Do đó, cần xử lý outliers trước phân cụm.

Yêu cầu 2 tham số (eps, minPts)	Chỉ yêu cầu 1 tham số (k)

Trong đó:

- **Eps (Epsilon)**: Là khoảng cách tối đa cho phép giữa hai điểm để chúng được coi là lân cận.
- **MinPts (Minimum Points)**: Là số lượng điểm tối thiểu cần thiết trong một khu vực lân cận để có một điểm được xem là điểm trung tâm.

2.2. Các định nghĩa trong thuật toán DBSCAN

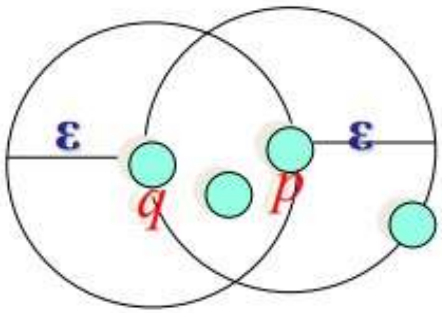
2.2.1 Eps-neighborhood

Vùng lân cận epsilon (*Eps-neighborhood*) của một điểm dữ liệu P được định nghĩa là tập hợp tất cả các điểm dữ liệu nằm trong phạm vi bán kính *epsilon* (kí hiệu €) xung quanh điểm P. Kí hiệu tập hợp những điểm này là:

$$N_{eps}(P) = \{Q \in D : d(P,Q) \leq €\}$$

Trong đó:

- D là tập hợp tất cả các điểm dữ liệu của tập huấn luyện



Hình 1. Ảnh minh họa Esp – neigborhood

Nguồn: Jing Gao - SUNY Buffalo

1.2.2 Điểm trung tâm (core point)

Một điểm được xem là điểm trung tâm nếu nó có ít nhất MinPts điểm trong vùng lân cận Epsilon của chính nó.

2.2.3 Điểm biên (border point)

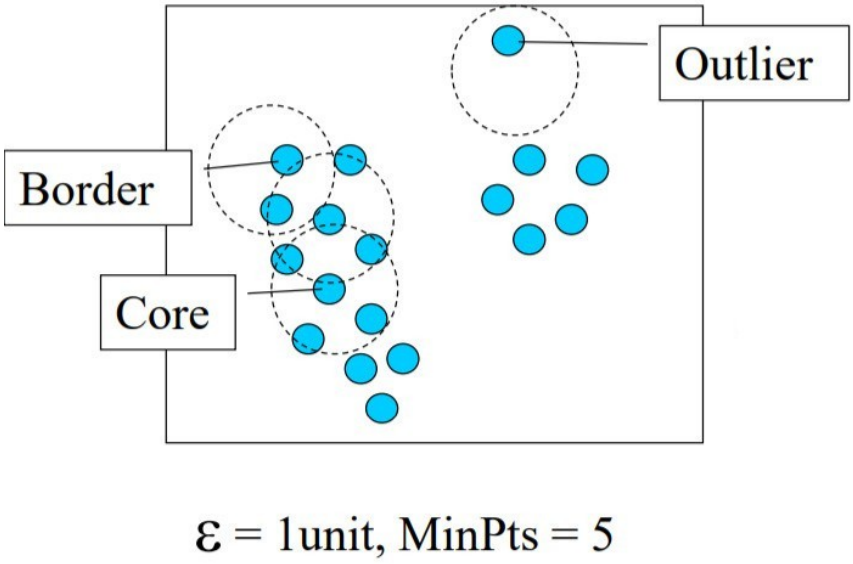
Một điểm được xem là điểm biên nếu nó có ít nhất một điểm trung tâm nằm ở vùng lân cận Epsilon nhưng mật độ không đủ MinPts điểm.

2.2.4 Điểm nhiễu (noise)

Điểm nhiễu là các điểm không thuộc bất kỳ cụm nào và không có khả năng tiếp cận mật độ từ bất kỳ điểm trung tâm nào.

Cho D là tập hợp các điểm. C_1, …, C_k là các cụm của D tương ứng với các tham số Eps_i và $MinPts_i$ ($i = 1,..., k$).

Kí hiệu: $\{p \in D \mid \forall i: p \notin Ci\}$.



Hình 2. Ảnh minh họa core point, border point và outlier

Nguồn: Jing Gao - SUNY Buffalo

2.2.5 Khả năng tiếp cận trực tiếp mật độ (directly density-reachable)

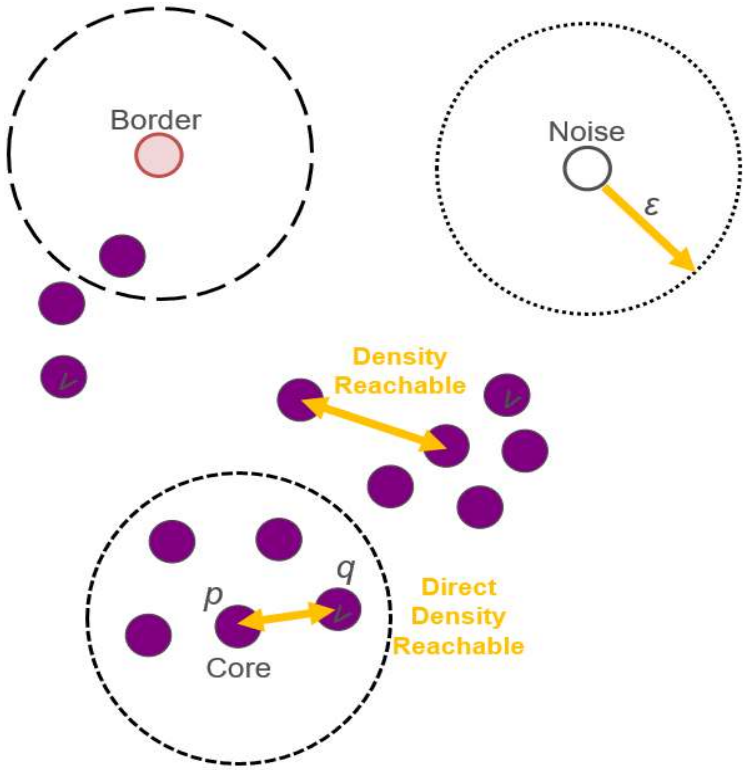
Khái niệm này được sử dụng để xác định xem một điểm dữ liệu có thể được kết nối với các điểm dữ liệu nằm trong *vùng lân cận epsilon* hay không. Điểm P được coi là có thể tiếp cận trực tiếp tới điểm Q (tương ứng với tham số epsilon và minPts) nếu thỏa mãn hai điều kiện:

- Q nằm trong *vùng lân cận epsilon* của P:

  $Q \in N_{eps}(P)$

- Số lượng các điểm dữ liệu nằm trong *vùng lân cận epsilon* tối thiểu là minPts:

$|N_{eps}(Q)| \geq minPts$



Hình 3. Ảnh minh họa directly density – reachable

Nguồn: Jing Gao - SUNY Buffalo

Như vậy, một điểm dữ liệu có thể tiếp cận được trực tiếp tới một điểm khác phải thỏa mãn điều kiện về khoảng cách giữa chúng và mật độ các điểm dữ liệu trong vùng lân cận epsilon phải tối thiểu bằng minPts. Khi đó, *vùng lân cận* được coi là có mật độ cao và sẽ được phân vào các cụm. Trái lại thì *vùng lân cận* sẽ có mật độ thấp. Trong trường hợp mật độ thấp thì điểm dữ liệu ở trung tâm được coi là không kết nối trực tiếp tới những điểm khác trong *vùng lân cận* và những điểm này có thể rơi vào biên của cụm hoặc là một điểm dữ liệu *nhiễu* không thuộc về cụm nào.

2.2.6 Khả năng tiếp cận mật độ (density-reachable)

Khả năng tiếp cận mật độ (*density-reachable*) liên quan đến cách hình thành một chuỗi liên kết điểm trong cụm.

Điểm p có khả năng tiếp cận mật độ bởi điểm q nếu dựa trên tham số Eps và MinPts, có một chuỗi các điểm $p_1, \ldots, p_n$ với $p_1 = q$, $p_n = p$ sao cho mỗi điểm p_{i+1} đều có khả năng tiếp cận mật độ trực tiếp (*directly density-reachable*) bởi p_i.

Khả năng tiếp cận mật độ (*density-reachable*) là khái niệm mở rộng từ khả năng tiếp cận trực tiếp mật độ (*directly density-reachable*). Mối quan hệ trong khái niệm này có tính bắc cầu nhưng không đối xứng (not symmetric).

Hai điểm biên của cùng một cụm không có khả năng tiếp cận mật độ lẫn nhau vì hai điểm này có thể không đáp ứng đúng các điều kiện về xác định điểm lõi (the core point condition). Tuy nhiên, vẫn có điểm lõi trong cụm này mà hai điểm biên có thể tiếp cận mật độ (*density-reachable*) được.

Ví dụ:

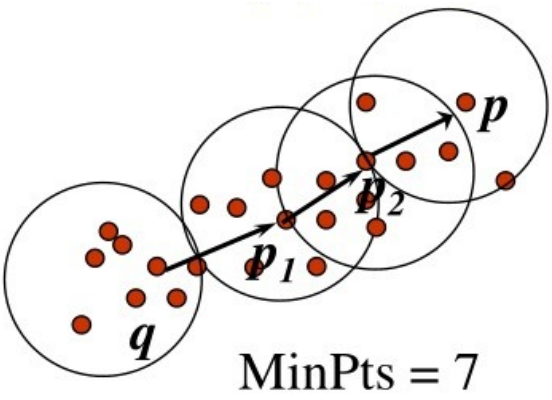
Một điểm p có thể tiếp cận mật độ trực tiếp (*directly density-reachable)* bởi p_2.

p_2 có thể tiếp cận mật độ trực tiếp bởi p_1.

p_1 có thể tiếp cận mật độ trực tiếp bởi q.

Từ đó, ta có chuỗi: $q \Rightarrow p_1 \Rightarrow p_2 \Rightarrow p$.

Vậy, p có khả năng tiếp cận mật độ (*density-reachable)* bởi q.

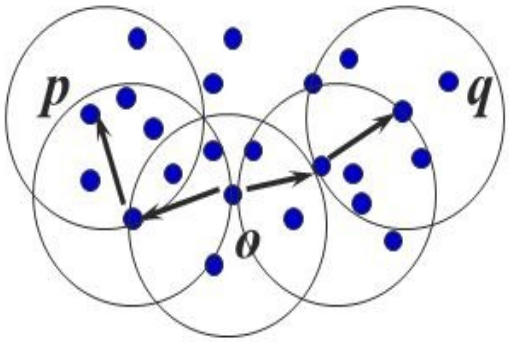


Hình 4. Ảnh minh họa density-reachable

Nguồn: Jing Gao - SUNY Buffalo

2.2.7 Kết nối mật độ (density-connected)

Điểm p kết nối mật độ được với điểm q nếu dựa trên tham số Eps và MinPts, có điểm o mà cả p và q đều có khả năng tiếp cận mật độ (*density-reachable*) bởi o.



Hình 5. Ảnh minh họa density-connected

Nguồn: Jing Gao - SUNY Buffalo

- Xuất phát từ một điểm dữ liệu ta có thể tìm được các điểm có khả năng *kết nối mật độ* tới nó theo lan truyền chuỗi để xác định cụm.
- Kết nối mật độ (density-connected) có tính đối xứng.

2.2.8 Cụm (cluster)

Cụm là tập hợp các điểm mà mỗi điểm trong đó có khả năng tiếp cận mật độ từ những điểm khác trong cùng một cụm.

Cho D là một tập hợp các điểm. Cụm C tương ứng với các tham số Eps và MinPts là tập con không rỗng của D. Cụm C thỏa mãn những điều kiện sau:

- **Tính tối đa (Maximality):**
  Với mọi điểm p, q: nếu điểm p thuộc cụm C và điểm q có khả năng tiếp cận mật độ từ p tương ứng với Eps và MinPts thì điểm q cũng thuộc cụm C.
- **Tính kết nối mật độ (Connectivity):**
  Với mọi điểm p, q thuộc cụm C: điểm p có khả năng kết nối mật độ với điểm q tương ứng với Eps và MinPts.

2.3 Giải thuật DBSCAN

2.3.1 Giải thuật

Giải thuật DBSCAN giúp phát hiện các cụm và nhiễu dựa trên định nghĩa 1.2.4 (Nhiễu) và 1.2.8 (Cụm). Giải thuật cần phải biết trước hai tham số Eps và MinPts. Để

tìm một nhóm, DBSCAN bắt đầu với một điểm p và tìm tất cả các điểm density-reachable từ p. Nếu p là một điểm lõi thì thuật toán hình thành một cụm. Nếu p là một điểm biên thì không có điểm nào density-reachable từ p và DBSCAN đi đến điểm kế tiếp của tập dữ liệu.

Input:

Tập dữ liệu gồm m phần tử: X =$\{x_i\}_{i=1,m}$ với X, ∈ R

Bán kính vùng lân cận: eps

Số lượng điểm tối thiểu: minPts

Giải thuật:

```
//Khởi tạo
ClusterId = 1
 for i = 1 to m do
        if (x_i chưa được duyệt qua)
                N = regionQuery(x_i, X, eps)
                if (|N| <minPts) //x_i không là điểm lõi
                        changeCluster(x_i, 0) //Gán x_i vào nhiễu
                else
                        changeCluster(N, Clusterld)
                        N = N \{x_i}
                        while (N != rỗng)
                                N' = regionQuery(p, X, eps) // p∈ N
                                if (|N'| >= minPts)
                                        for j=1 to sizeof(N') do
                                                q = N'.get(j)
                                                if (q chưa được duyệt hoặc là noise)
                                                        if (q chưa được duyệt)
                                                                N = N ∪ {q}
                                                        changeCluster(q, ClusterId)
                                N =N \{p}
```

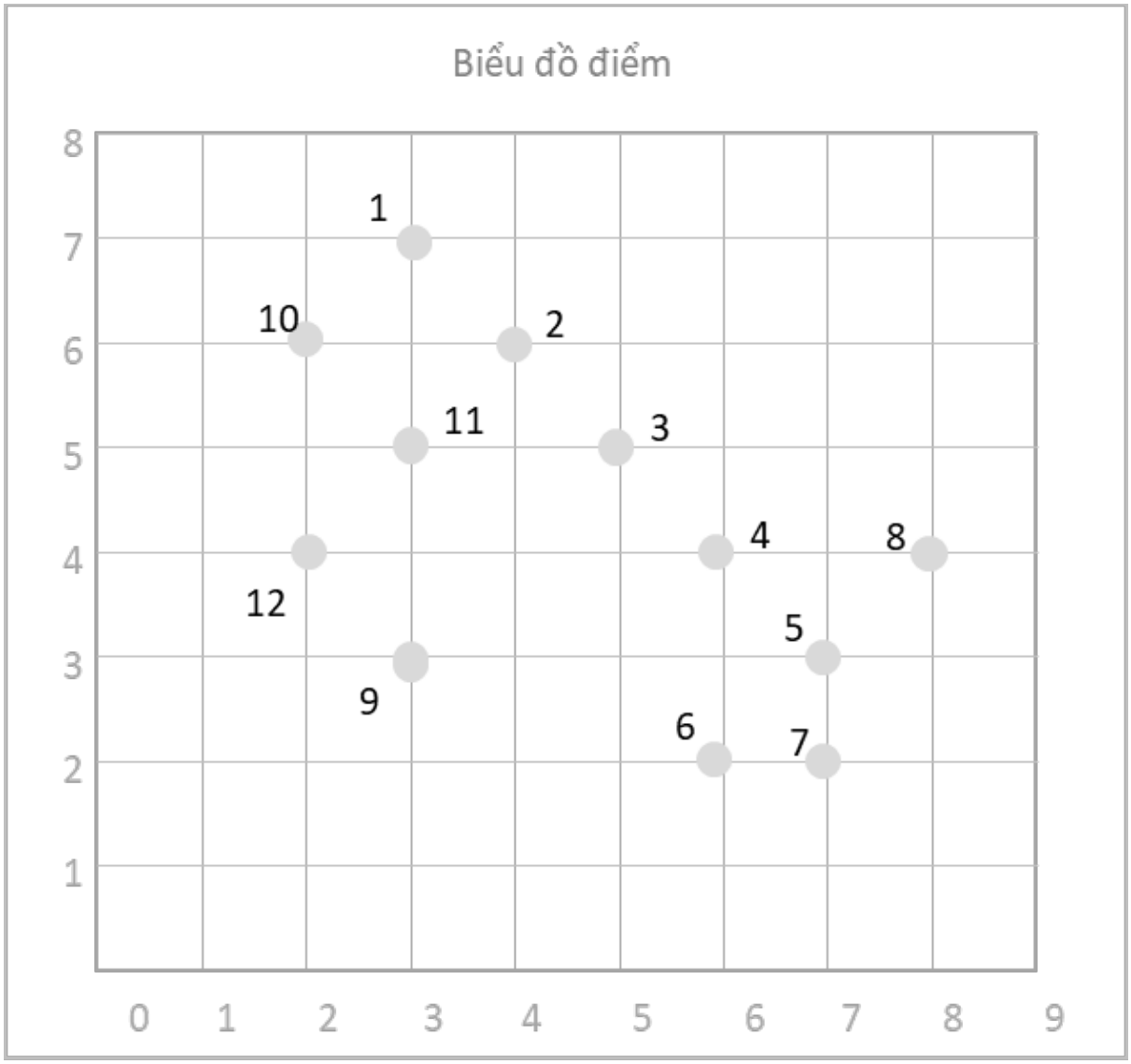
ClusterId++

Hàm regionQuery(x_i, X, eps) được gọi trong giải thuật DBSCAN trả về các điểm láng giềng trong bán kính eps của điểm x_i. Hàm này sẽ cần duyệt qua toàn tập dữ liệu để tìm k láng giềng.

2.3.2 Ví dụ

Cho các điểm dữ liệu sau với minPts = 4 và eps = 1.5:

P1: (3, 7)	P2: (4, 6)	P3: (5, 5)
P4: (6, 4)	P5: (7, 3)	P6: (6, 2)
P7: (7, 2)	P8: (8, 4)	P9: (3, 3)
P10: (2, 6)	P11: (3, 5)	P12: (2, 4)



Hình 6. Ảnh minh họa các điểm

Bước 1: Áp dụng công thức khoảng cách Euclide để tính khoảng cách giữa các điểm:

$d(A(x_1,y_1), B(x_2,y_2)) = \sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$

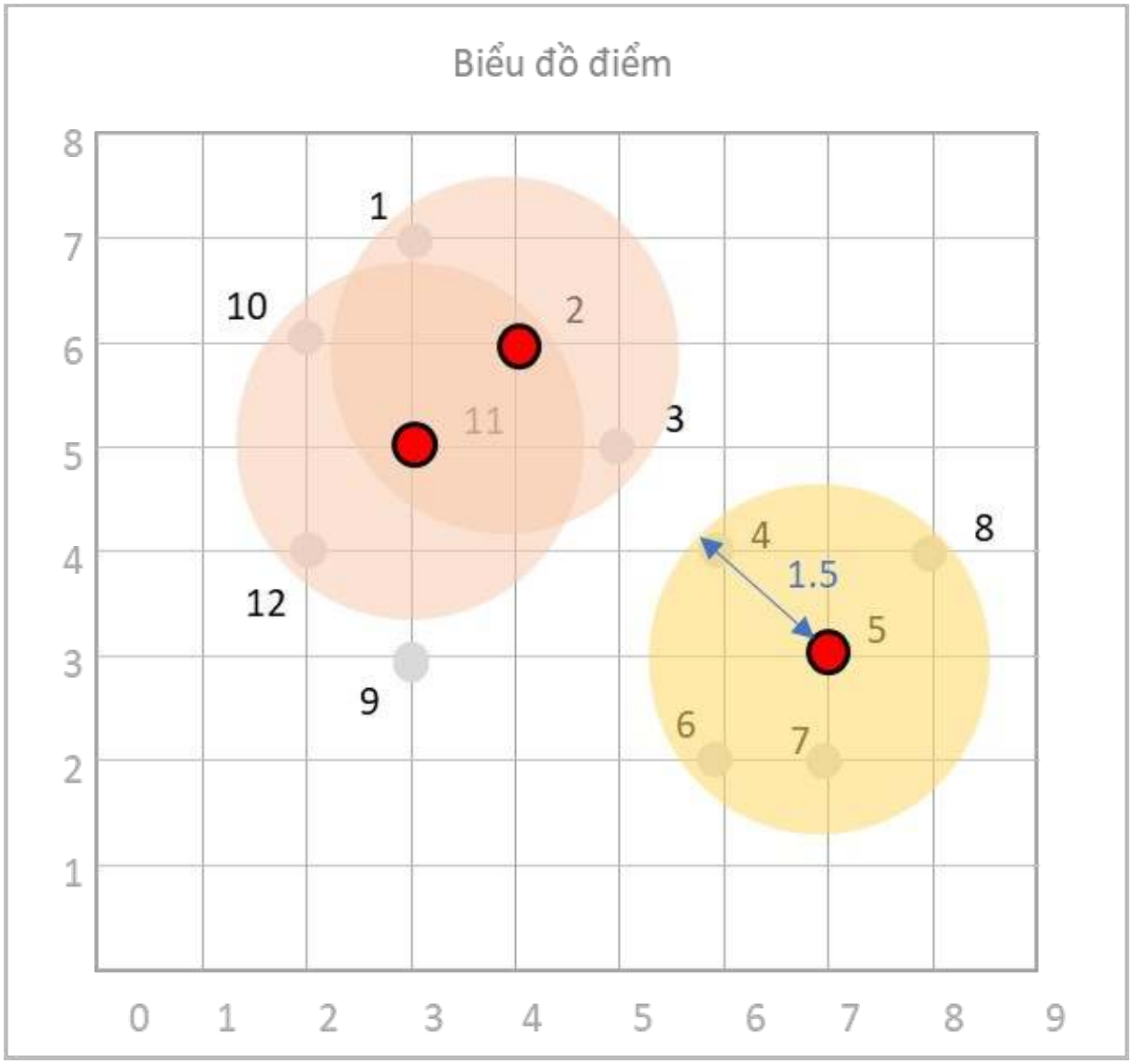
Bảng kết quả khoảng cách giữa các điểm:

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
P1	0											
P2	1.41	0										
P3	2.83	1.41	0									

P4	4.24	2.83	1.41	0								
P5	5.66	4.24	2.83	1.41	0							
P6	5.83	4.47	3.16	2.00	1.41	0						
P7	6.40	5.00	3.61	2.24	1.00	1.00	0					
P8	5.83	4.47	3.16	2.00	1.41	2.83	2.24	0				
P9	4.00	3.16	2.83	3.16	4.00	3.16	4.12	5.10	0			
P10	1.41	2.00	3.16	4.47	5.83	5.66	6.40	6.32	3.16	0		
P11	2.00	1.41	2.00	3.16	4.47	4.24	5.00	5.10	2.00	1.41	0	
P12	3.16	2.83	3.16	4.00	5.10	4.47	5.39	6.00	1.41	2.00	1.41	0

Bước 2: Với eps = 1.5, chọn các khoảng cách có giá trị nhỏ hơn hoặc bằng eps này (duyệt từng điểm theo cả hàng ngang và dọc). Sau đó, với minPts = 4, xét từng điểm nếu điểm đó có tối thiểu 3 điểm khoảng cách gần nhất từ kết quả trên (hoặc tối thiểu 4 điểm tính cả điểm đang xét) thì đó là điểm lõi.Từ đó, được kết quả:

P1: P2, P10.	P7: P5, P6.
P2: P1, P3, P11.	P8: P5.
P3: P2, P4.	P9: P12.
P4: P3, P5.	P10: P1, P11.
P5: P4, P6, P7, P8.	**P11: P2, P10, P12.**
P6: P5, P7.	P12: P9, P11.



Hình 7. Ảnh minh họa các điểm dựa trên vùng lân cận

Bước 3: Xác định điểm biên (border) dựa trên điểm lõi.

Điểm	Loại điểm	
P1	Điểm nhiễu (noise)	Điểm biên (border)
P2	**Điểm lõi (core)**	
P3	Điểm nhiễu (noise)	Điểm biên (border)
P4	Điểm nhiễu (noise)	Điểm biên (border)
P5	**Điểm lõi (core)**	

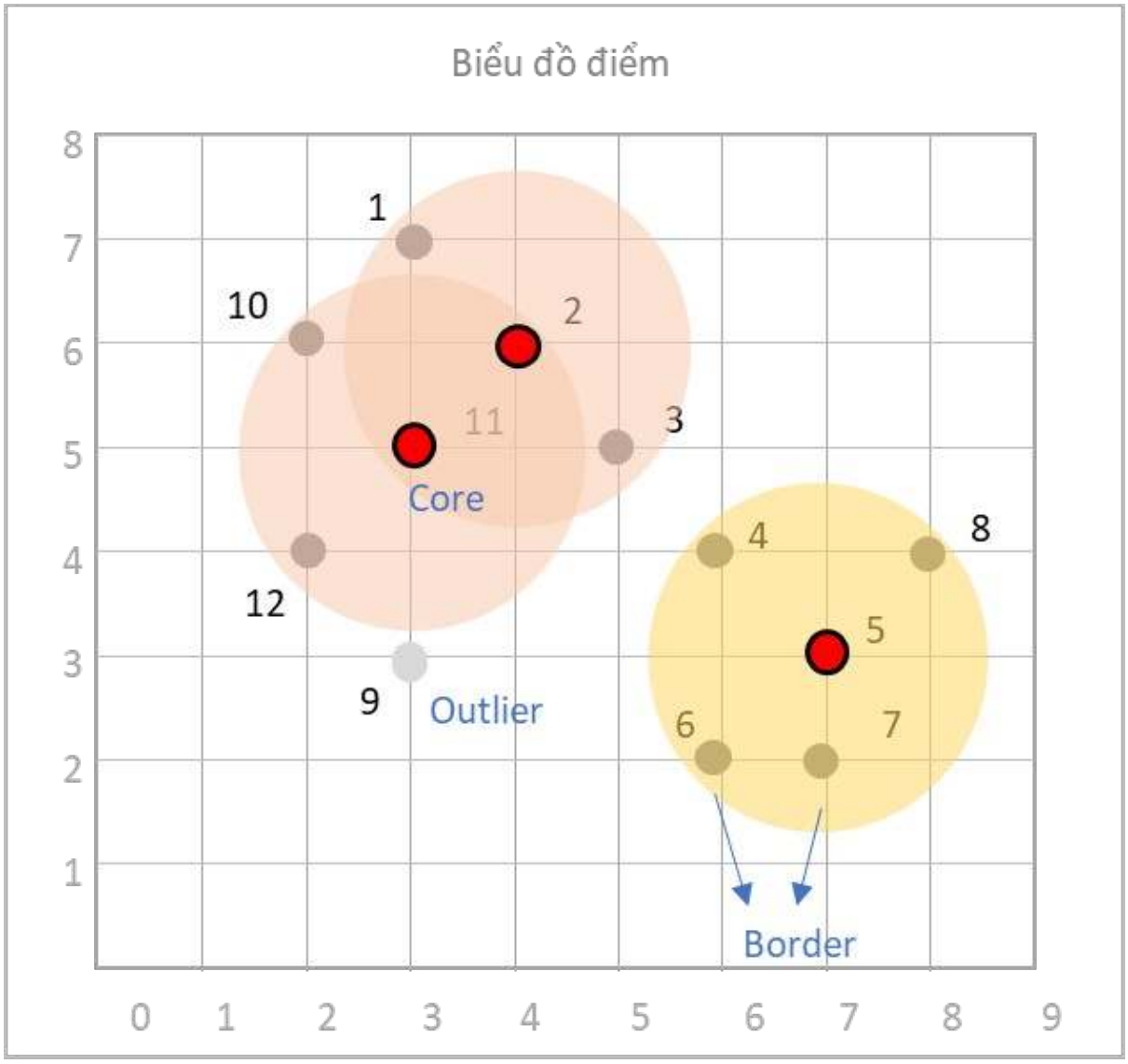
P6	Điểm nhiễu (noise)	Điểm biên (border)
P7	Điểm nhiễu (noise)	Điểm biên (border)
P8	Điểm nhiễu (noise)	Điểm biên (border)
P9	Điểm nhiễu (noise)	
P10	Điểm nhiễu (noise)	Điểm biên (border)
P11	**Điểm lõi (core)**	
P12	Điểm nhiễu (noise)	Điểm biên (border)

Với ba điểm lõi là:

P2: P1, P3, P11. (Vùng lân cận P1, P2, P3, P11 với P2 là điểm lõi)

P5: P4, P6, P7, P8. (Vùng lân cận P4, P5, P6, P7, P8 với P5 là điểm lõi)

P11: P2, P10, P12. (Vùng lân cận P2, P10, P11, P12 với P11 là điểm lõi)



Hình 8. Ảnh minh họa các điểm (core, border, outlier)

Nhận thấy, P9 không có khoảng cách gần 3 điểm này (không nằm trong cả 3 vùng lân cận). Do đó, P9 sẽ là điểm nhiễu.

2.4. Xác định các tham số

Xác định tham số là một bước quan trọng khi áp dụng thuật toán bởi một thay đổi nhỏ trong tham số có thể ảnh hưởng đáng kể đến kết quả của thuật toán. Đối với thuật toán DBSCAN, hai tham số quan trọng cần xác định ở đây là **MinPts** và **Eps**.

Theo quy tắc chung, MinPts tối thiểu có thể được tính theo số chiều D trong tập dữ liệu đó là minPts ≥ D+1. Giá trị MinPts = 1 không có ý nghĩa, vì khi đó mọi điểm bản thân nó đều là một cụm. Do đó, minPts phải được chọn ít nhất là 3. Tuy nhiên, các

giá trị lớn hơn thường tốt hơn cho các tập dữ liệu có nhiễu và kết quả phân cụm thường hợp lý hơn. Theo quy tắc chung thì thường chọn minPts = 2×dim. Trong trường hợp dữ liệu có nhiễu hoặc có nhiều quan sát lặp lại thì cần lựa chọn giá trị minPts lớn hơn nữa tương ứng với những bộ dữ liệu lớn.

Lý do dùng k - distance để tìm eps:

Trong biểu đồ k-distance, trục x thể hiện số điểm láng giềng gần nhất, và trục y thể hiện giá trị khoảng cách. Khi giá trị của Eps tăng lên, biểu đồ k-distance sẽ có xu hướng đi lên. Một điểm khuỷu (elbow point) trong biểu đồ k-distance cho thấy giá trị tốt của Eps. Giá trị Eps ở điểm khuỷu thường là giá trị mà tại đó số lượng cụm trong dữ liệu thay đổi đáng kể.

Đối với Esp, ý tưởng là với các điểm trong một cụm, điểm lân cận thứ k của chúng có khoảng cách xấp xỉ nhau. Điểm nhiễu sẽ có điểm lân cận thứ k ở khoảng cách xa hơn. Dựa trên ý tưởng đó, giá trị của Eps có thể được xác định từ đồ thị K - distance. Đây là biểu đồ thể hiện giá trị khoảng cách trong thuật toán K - Means đến k = MinPts - 1 điểm láng giềng gần nhất. Ứng với mỗi điểm ta lựa chọn khoảng cách lớn nhất trong k khoảng cách. Những khoảng cách này trên đồ thị được sắp xếp theo thứ tự giảm dần. Giá trị Eps nên chọn là điểm có độ cong lớn nhất (elbow) - điểm mà từ đó khoảng cách bắt đầu tăng đột biến, cho thấy sự chuyển tiếp từ các điểm thuộc cụm đến các điểm nhiễu. Nếu giá trị của Eps được chọn quá nhỏ thì một phần lớn dữ liệu sẽ không được phân cụm và được xem là nhiễu. Ngược lại, nếu Eps được chọn quá lớn, các cụm nhỏ sẽ được hợp nhất và phần lớn các điểm sẽ nằm trong cùng một cụm.

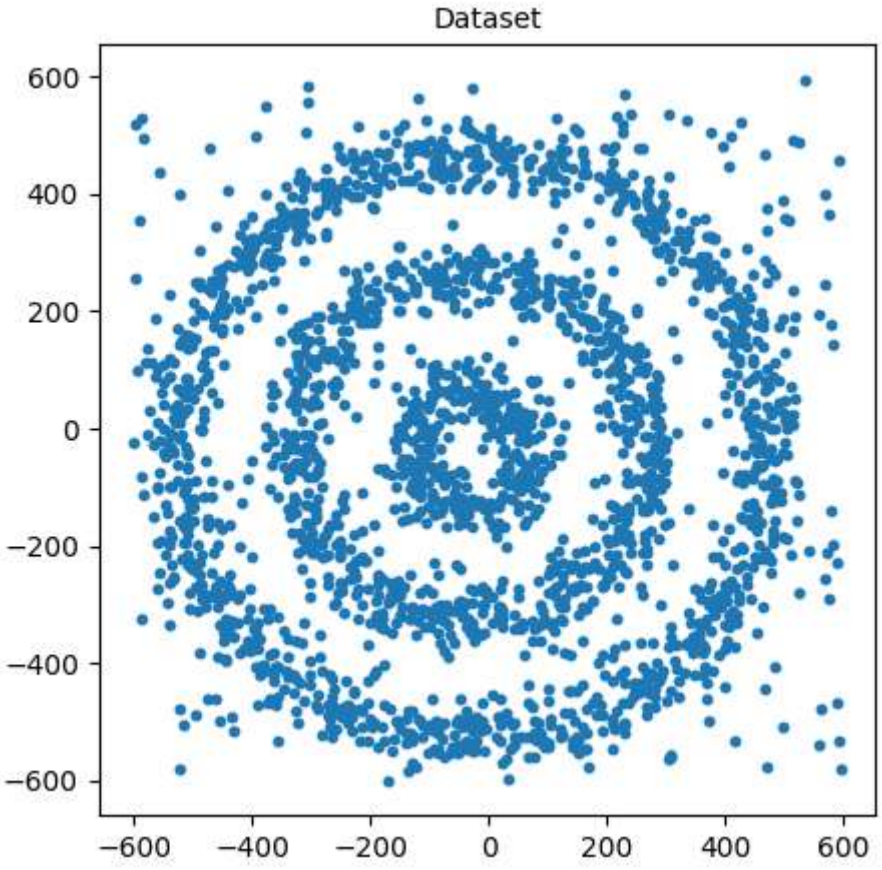
Ngoài ra, phương pháp thử và sai cũng có thể được áp dụng trong việc tìm các tham số Eps và MinPts. Phương pháp bao gồm việc thực hiện các lần lặp với các giá trị khác nhau cho Eps và MinPts, sau đó đánh giá kết quả phân cụm để tìm ra bộ tham số tối ưu. Ở mỗi lần thử, nếu kết quả không tốt thì cần điều chỉnh lại tham số và thử lại. Bắt đầu với một giá trị Eps và MinPts ước lượng và điều chỉnh dựa trên kết quả phân cụm. Nếu các cụm quá nhỏ hoặc có quá nhiều điểm nhiễu thì cần tăng giá trị Eps và MinPts. Nếu các cụm quá lớn và có xu hướng hợp nhất thì nên giảm giá trị Eps và MinPts. Mục tiêu là tìm ra bộ tham số mà với đó, thuật toán có thể tạo ra các cụm có ý nghĩa và phân biệt rõ ràng giữa các cụm và điểm nhiễu.

CHƯƠNG 3: DBSCAN VÀ CÁC MÔ HÌNH PHÂN CỤM KHÁC

Nhóm tiến hành đánh giá hiệu quả của thuật toán DBSCAN dựa trên ưu điểm của nó so với những thuật toán phân cụm thông thường như K- Means và HAC.

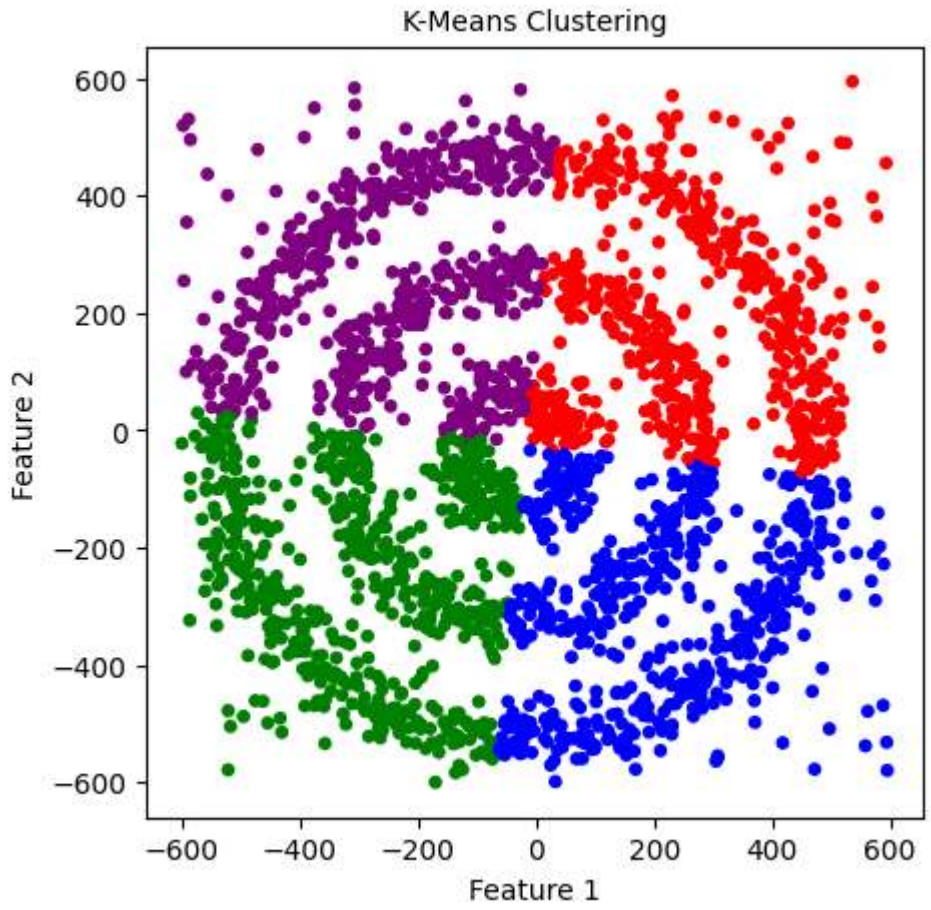
Để thể hiện điều đó, nhóm tạo ra một tập dữ liệu gồm các điểm dữ liệu ngẫu nhiên được sắp xếp có chủ đích thành một hình dạng đặc biệt và thêm vào đó một số điểm nhiễu với mô tả như bảng dưới.

Tên tập dữ liệu	**Số thuộc tính**	**Số mẫu**
Dataset	2	2300



Hình 9. Ảnh minh họa bộ dữ liệu ban đầu

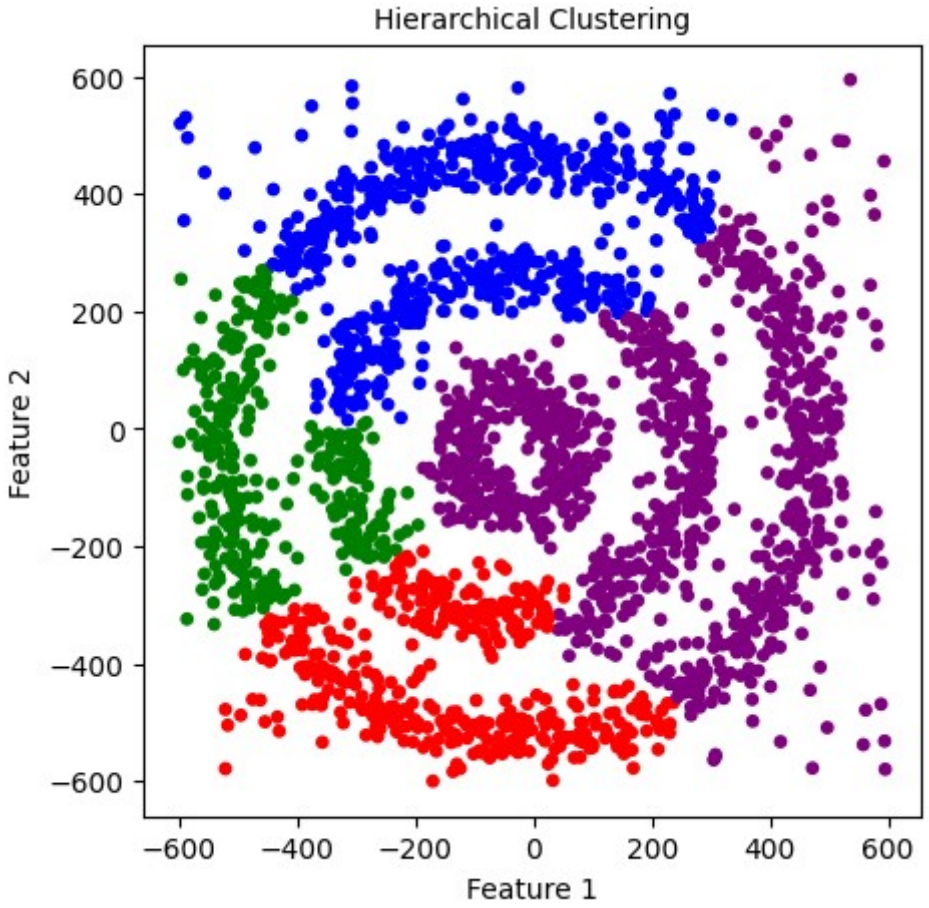
3.1 Kết quả phân cụm với K-means



Hình 10. Ảnh minh họa phân cụm với K - means

Nhận xét: K - means có xu hướng gom nhiễu thành các cụm vì thuật toán cố gắng tối thiểu hóa tổng bình phương khoảng cách giữa các điểm dữ liệu và trung tâm cụm. Điều này có thể dẫn đến việc các điểm nhiễu được coi là một phần của cụm thay vì được xác định là nhiễu và loại bỏ.

3.2 Kết quả phân cụm với HAC

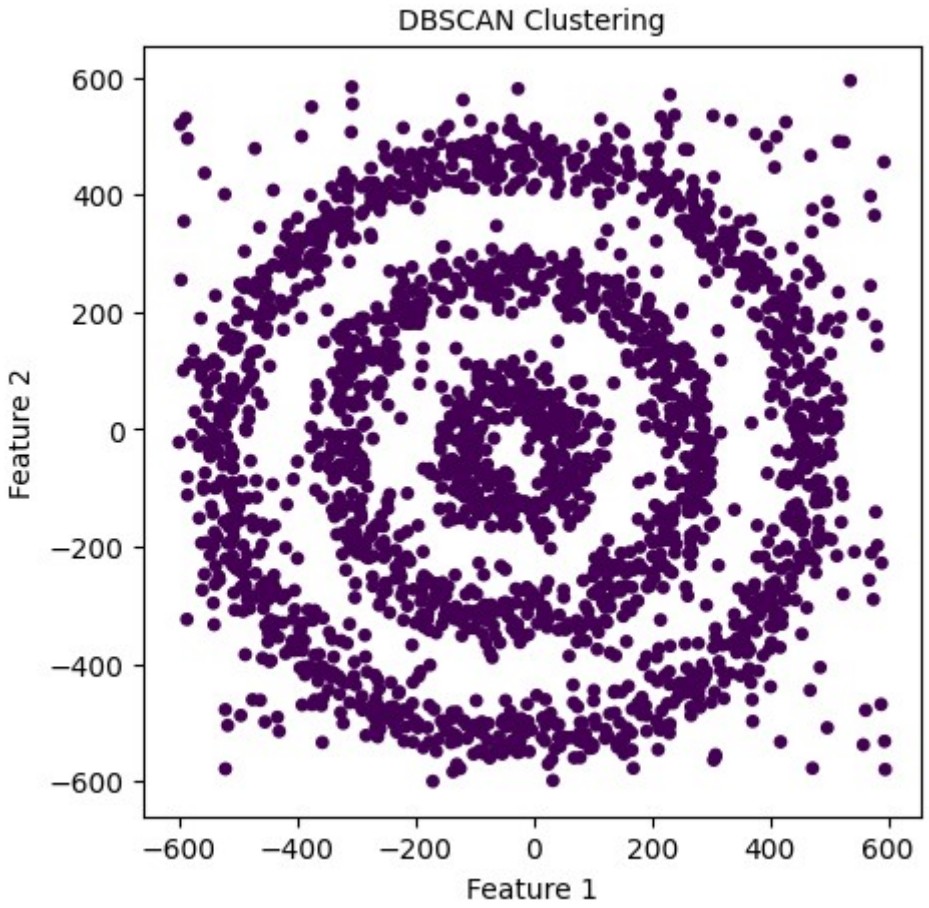


Hình 11. Ảnh minh họa phân cụm với HAC

Nhận xét: Tương tự K - means, HAC cũng có xu hướng gom các dữ liệu nhiễu vào trong cụm. Khi phân cụm, HAC coi mỗi điểm dữ liệu là một cụm riêng lẻ. Sau đó, thuật toán sẽ từng bước kết hợp các cụm này dựa trên một tiêu chí khoảng cách đã chọn (ở đây là khoảng cách Euclidean). Điều này có thể dẫn đến việc các điểm nhiễu được gom vào cùng một cụm với các điểm dữ liệu khác nếu nhiễu nằm gần các điểm dữ liệu đó.

3.3. Kết quả phân cụm bằng DBSCAN

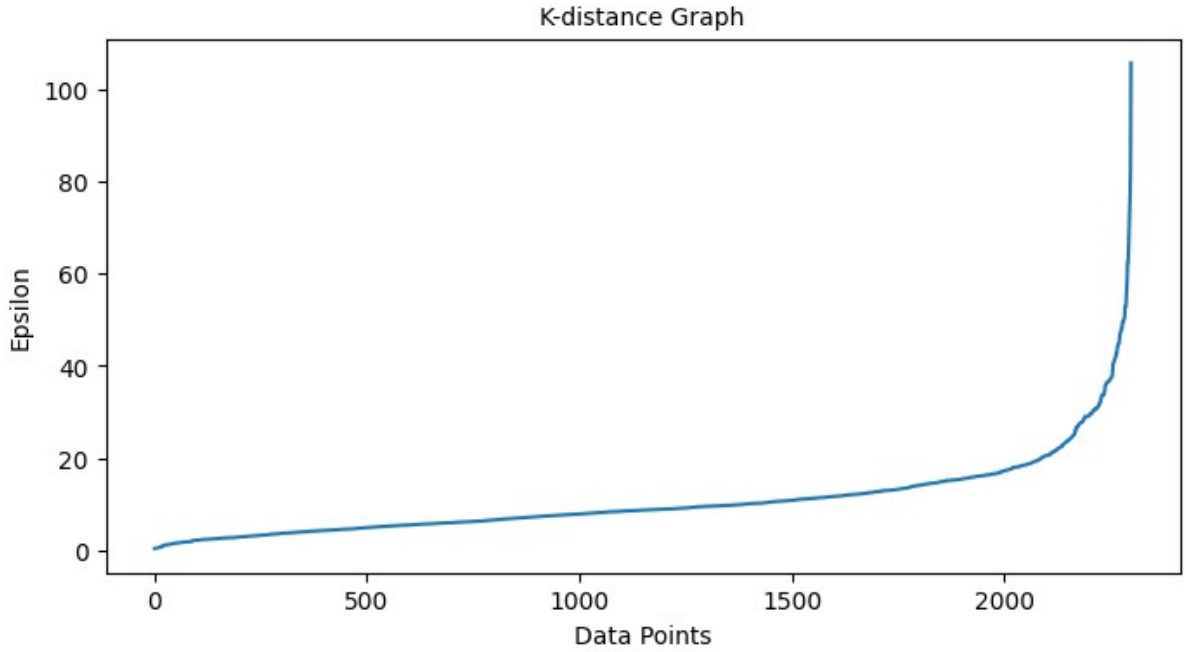
Với tham số mặc định (epsilon =0.5, minPoints = 5)



Hình 12. Kết quả phân cụm bằng DBSCAN (Esp = 0.5, MinPts = 5)

Nhận xét: Tất cả các giá trị đều được gán labels -1, tức là outlier. Có thể thấy kết quả phân cụm rất tệ do sử dụng siêu tham số chưa hợp lý. Do đó cần tìm giá trị của epsilon và minPoints, sau đó huấn luyện lại mô hình.

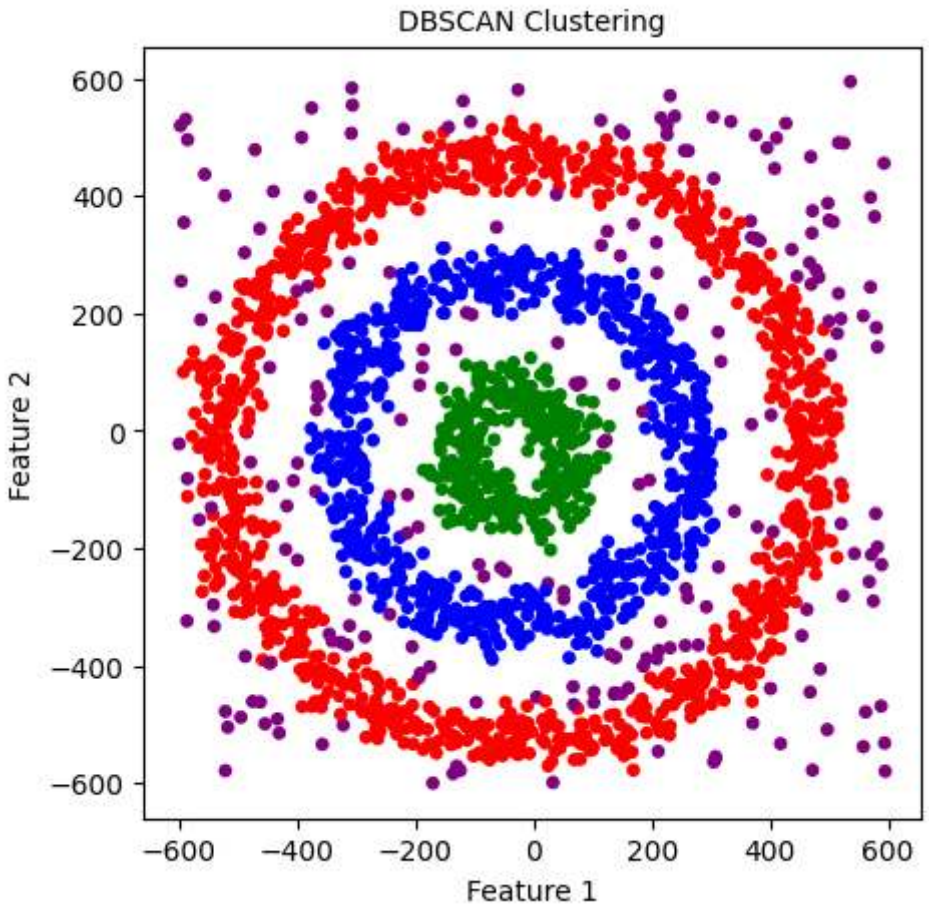
Sử dụng K- distance để tìm Eps phù hợp



Hình 13. Biểu đồ K - distance

Từ biểu đồ K - distance và qua nhiều lần thử, nhóm tìm ra được 2 tham số tối ưu epsilon =30, minPoints = 6

Kết quả phân cụm bằng DBSCAN sau khi điều chỉnh tham số



Hình 14. Kết quả phân cụm bằng DBSCAN (Esp = 30, MinPts = 6)

Nhận xét: DBSCAN phân dữ liệu thành 3 cụm rõ ràng, đặc biệt là thuật toán có thể phát hiện các outliers (các điểm dữ liệu màu tím) và không gom cụm chúng.

Kết luận: Như vậy, từ tập dữ liệu Dataset, nhóm có thể kết luận được rằng DBSCAN hoạt động cực kì tốt trên những bộ dữ liệu có hình dạng đặc biệt và có thể nhận biết được các điểm outlier

CHƯƠNG 4: ỨNG DỤNG DBSCAN VÀO BỘ DỮ LIỆU CỤ THỂ

4.1 Tổng quan bộ dữ liệu thu thập

4.1.1 Giới thiệu bộ dữ liệu

Bộ dữ liệu California housing được lấy trên Kaggle, chứa các thông tin liên quan tới các lô nhà ở California, được thu thập bởi Cục Thống kê Dân số Hoa Kỳ năm 1990. Bộ dữ liệu có tổng cộng 8 thuộc tính và 1700 bản ghi được ghi nhận.

4.1.2 Các thuộc tính của bộ dữ liệu

STT	Tên thuộc tính	Mô tả	Giá trị	Kiểu dữ liệu
1	longitude	Kinh độ (nếu giá trị âm càng xa giá trị 0 thì vị trí ngôi nhà càng gần phía Tây)	(-124.3) - (-114.3)	float64
2	latitude	Vĩ độ (nếu giá trị âm càng xa giá trị 0 thì vị trí ngôi nhà càng gần phía Bắc)	32.5 - 42.5	float64
3	housingMedianAge	Trung vị của tuổi các ngôi nhà trong một lô đất (giá trị thấp hơn nghĩa là tòa nhà mới hơn)	1.0 - 52.0	float64
4	totalRooms	Số phòng trong một lô đất	2.0 - 37937.0	float64
5	totalBedrooms	Số phòng ngủ trong một lô đất	1.0 - 6445.0	float64
6	population	Số người sống trong một lô đất	3.0 - 35682.0	float64
6	households	Số hộ gia đình sống trong một lô đất	1.0 - 6082.0	float64
7	medianIncome	Giá trị thu nhập trung bình của các	0.5 - 15.0	float64

		hộ gia đình trong một lô đất (tính bằng chục nghìn Đô la Mỹ)		
8	medianHouseValue	Giá trị nhà trung bình của các hộ gia đình trong một lô đất (tính bằng Đô la Mỹ)	14999.0 - 500001.0	float64
9	ID	Số thứ tự	1700 giá trị	

4.2 Tiền xử lý dữ liệu

Tiền xử lý dữ liệu là một bước quan trọng trong quá trình phân tích dữ liệu. Bước này nhằm đảm bảo dữ liệu đầu vào cho mô hình chính xác và phù hợp với mục đích phân tích.

Hai bước quan trọng trong tiền xử lý dữ liệu là xóa bỏ trùng lặp và tìm các điểm dữ liệu bị thiếu. Xóa bỏ trùng lặp giúp loại bỏ các quan sát trùng lặp trong dữ liệu, từ đó giúp phân tích dữ liệu chính xác hơn.

```
remove_duplicates(df)
```

```
'Không có dữ liệu trùng lặp'
```

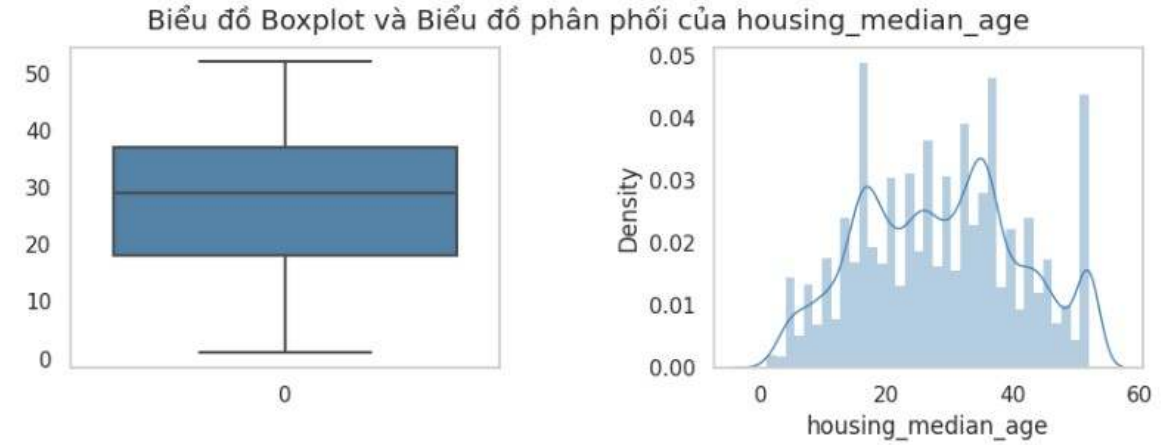
Tìm các điểm dữ liệu bị thiếu để tìm ra phương pháp khắc phục hiệu quả vì tất cả những mô hình học máy đều không chấp nhận giá trị null

```
remove_null_values(df)
```

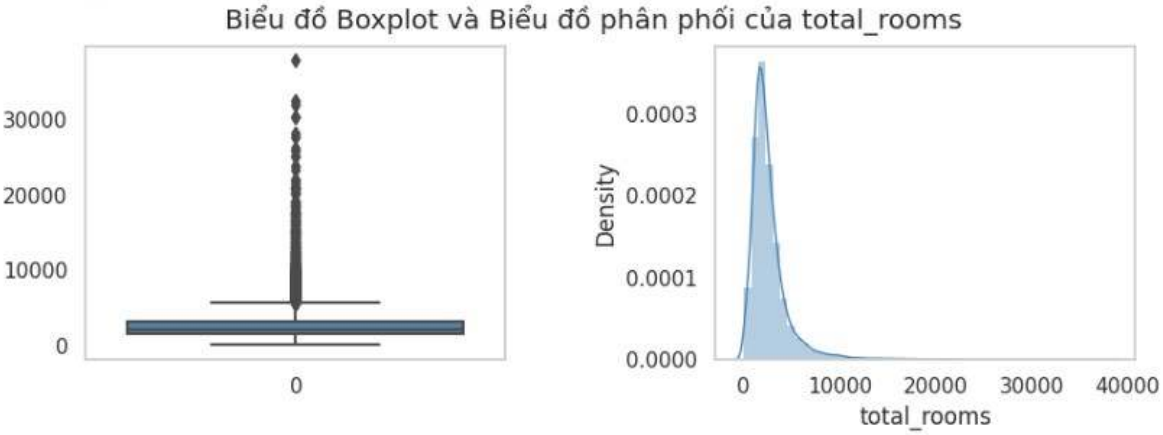
```
'Không có dữ liệu có giá trị null'
```

Như vậy, bộ dữ liệu đã hoàn chỉnh.

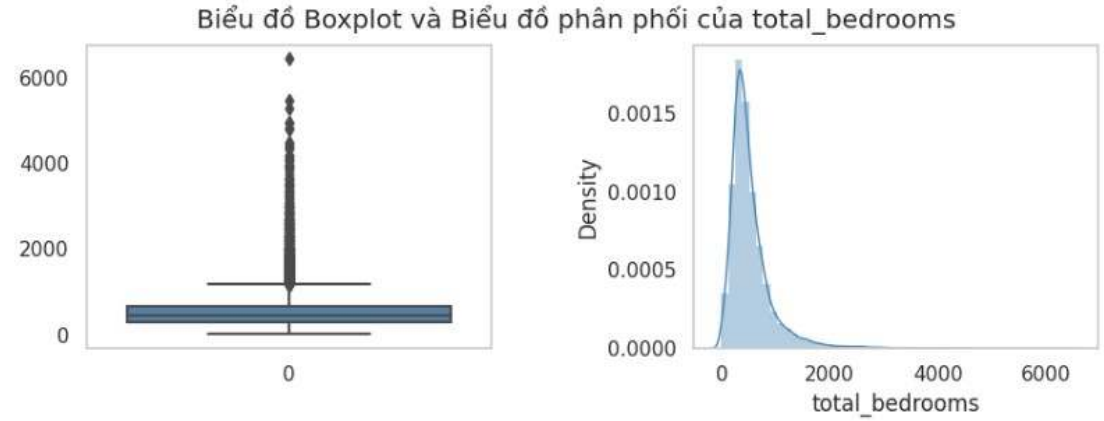
4.3 Trực quan & chuẩn hóa dữ liệu



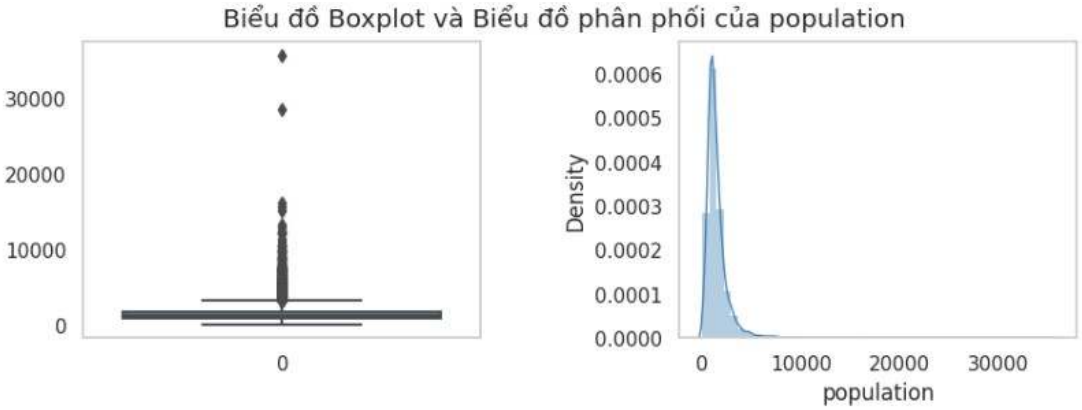
Hình 15. Biểu đồ boxplot và biểu đồ phân phối của housing_meadian_age



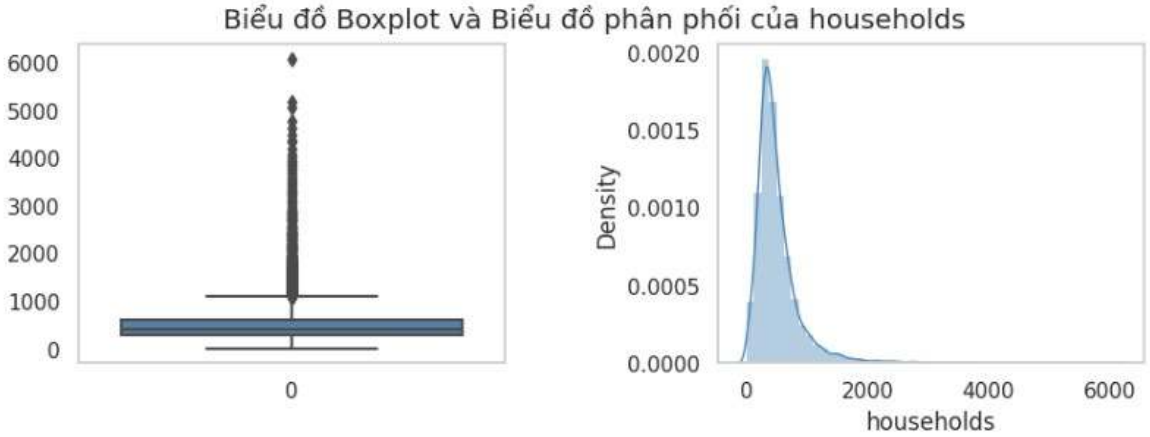
Hình 16. . Biểu đồ boxplot và biểu đồ phân phối của total_rooms



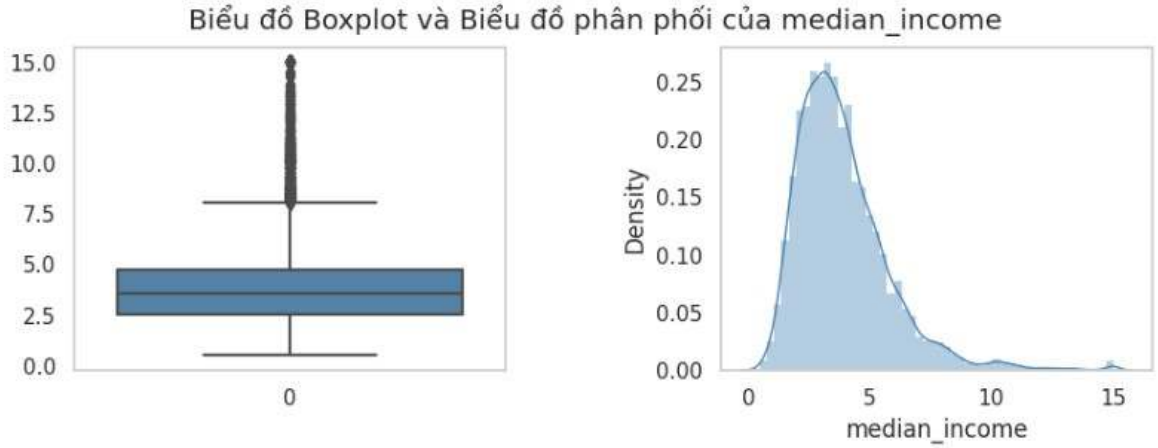
Hình 17. . Biểu đồ boxplot và biểu đồ phân phối của total_bedrooms



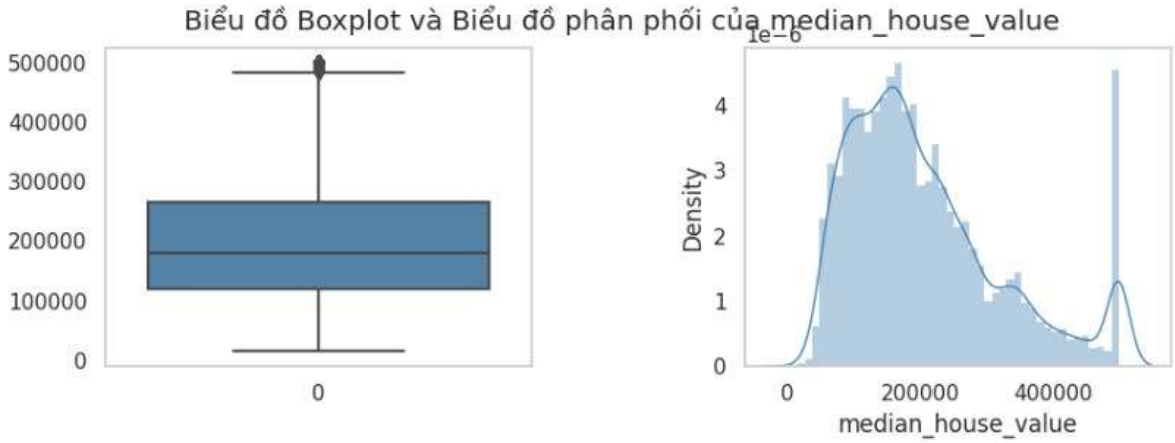
Hình 18. . Biểu đồ boxplot và biểu đồ phân phối của population



Hình 19. . Biểu đồ boxplot và biểu đồ phân phối của households

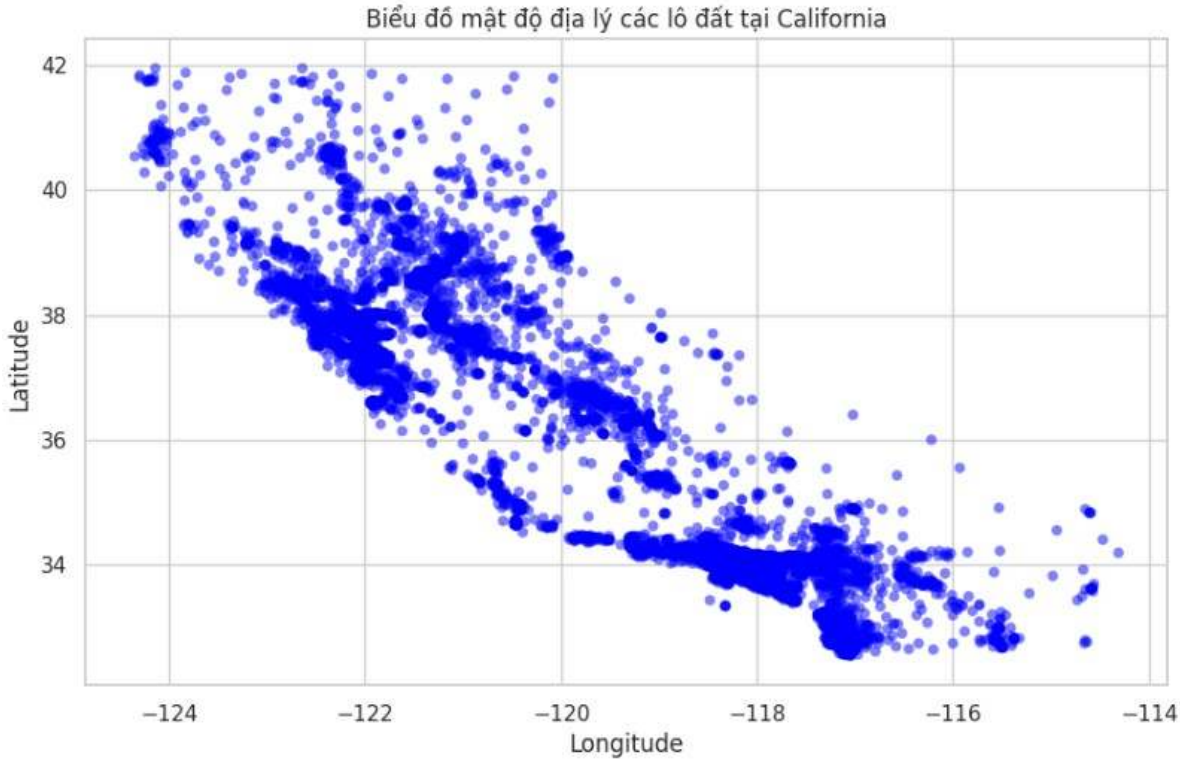


Hình 20. . Biểu đồ boxplot và biểu đồ phân phối của mean_income



Hình 21. . Biểu đồ boxplot và biểu đồ phân phối của median_house_income

Nhận xét: Sau khi trực quan hóa các biến trong dataset bằng box plot, nhóm nhận thấy có một vài giá trị ngoại lai xuất hiện ở 1 vài cột, tuy nhiên không nhiều. Chính vì DBSCAN có khả năng tự nhận diện outlier nên nhóm quyết định không loại bỏ những điểm này ra khỏi dataset.



Hình 22. Biểu đồ thể hiện mật độ địa lý các lô đất tại California

Nhận xét: Dựa trên các giá trị về kinh độ và vĩ độ, nhóm trực quan hóa được mật độ những lô đất trong bang California. Điều này khiến nhóm có thêm một ý tưởng, đó là phân cụm các lô đất tại California dựa trên vị trí địa lý.

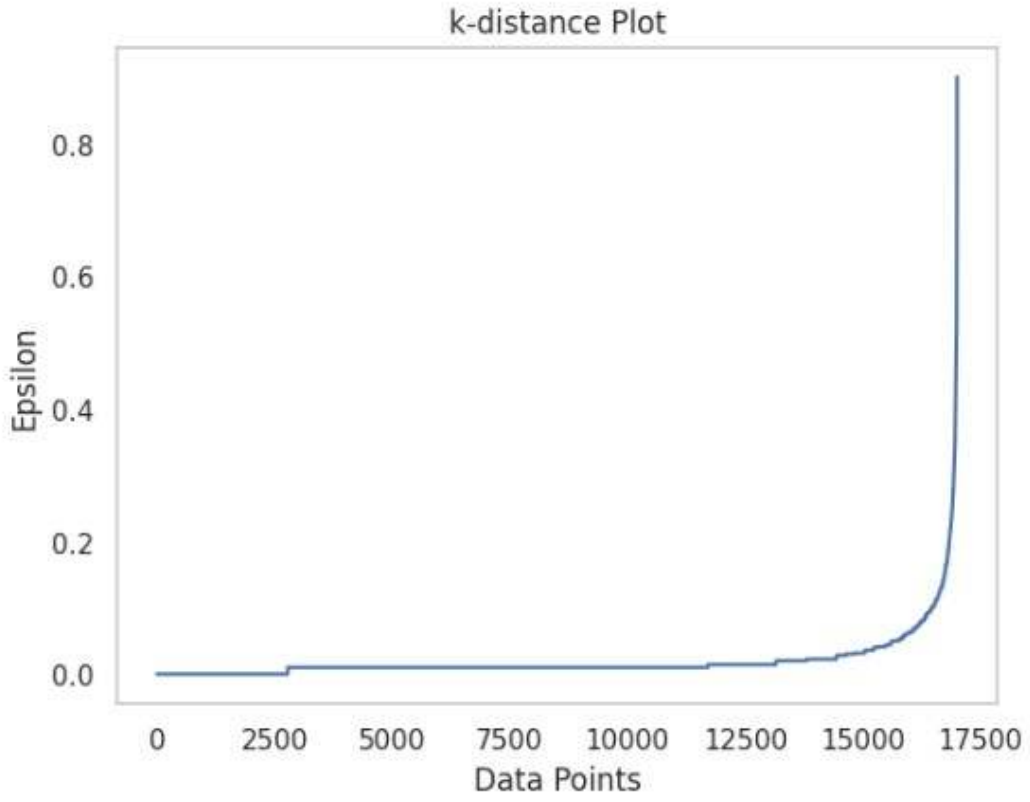
Sau đó nhóm tiến hành chuẩn hóa dữ liệu bằng StandardScaler

4.4 Xây dựng & đánh giá mô hình

4.4.1 Phân cụm các lô đất dựa trên tọa độ địa lý

Tìm kiếm giá trị Epsilon và MinPts tối ưu

Như đã trình bày ở 1.4 (Xác định các tham số), để tìm ra Epsilon tối ưu, nhóm tiền hành xác định từ đồ thị K - distance.



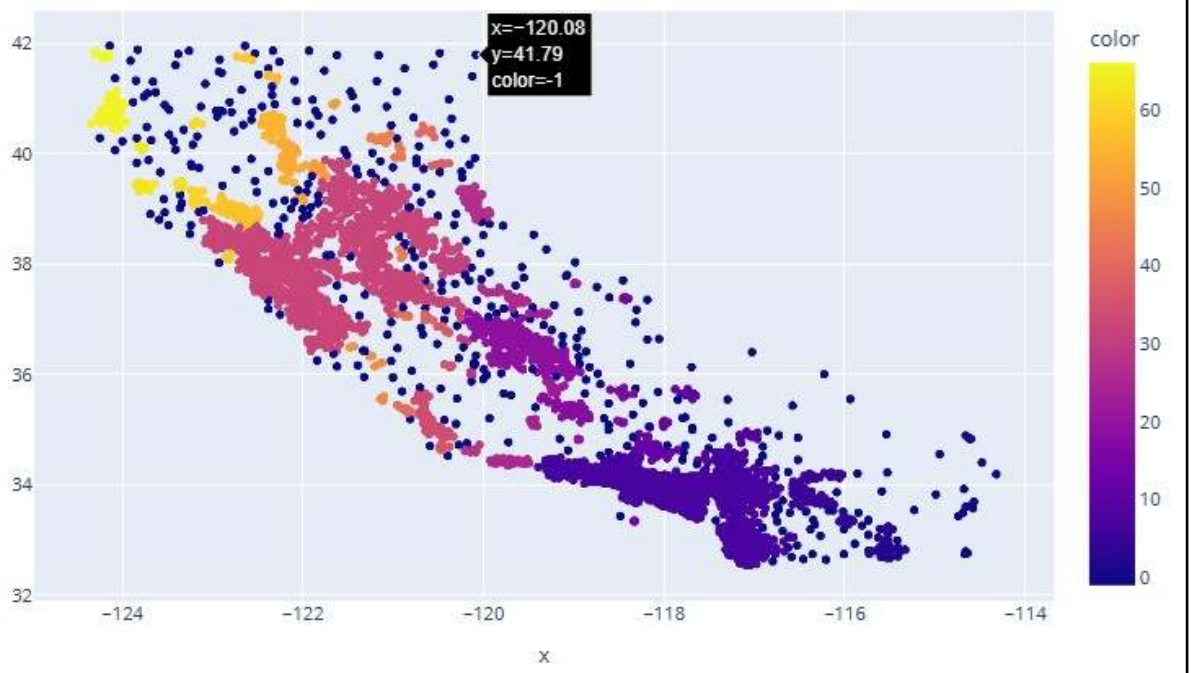
Hình 23. Biểu đồ K - distance

Nhận xét: Từ đồ thị, điểm cong cực đại tương ứng với Epsilon = 0.1. MinPts nhóm chọn bằng 2 lần số chiều trong dataset

Kết quả:

```
dbscan_cluster_model = DBSCAN(eps=0.1, min_samples=4).fit(X)
dbscan_cluster_model
```

```
DBSCAN(eps=0.1, min_samples=4)
```



Hình 24. Kết quả phân cụm dựa trên tọa độ địa lý (Esp = 0.1, MinPts = 4)

Kết quả phân cụm cho thấy Data được phân tới hơn 60 cụm. Rõ ràng chọn Epsilon dựa trên đồ thị K - Distance trong trường hợp này là không tốt. Số cụm bị chia thành nhiều như vậy nguyên nhân chính là do vùng lân cận Epsilon quá nhỏ.
Đánh giá kết quả phân cụm bằng điểm số Silhouette ta được:

```
Silhouette score: -0.12506709078950604
```

Điểm số Silhouette < 0 chứng tỏ các cụm trong mô hình được phân chưa tốt.

Tiền hành phương pháp Grid Search để tìm ra tham số tốt nhất, tham số tốt nhất sẽ được chọn từ mô hình cho ra Silhouette Score tốt nhất. Như đã trình bày ở trên, DBSCAN có thể tự detect outlier nên khi phân cụm bằng DBSCAN, mô hình sẽ mặc định cụm mang giá trị -1 là những outlier, do đó mô hình chỉ có ý nghĩa nếu số cụm >2. Thêm nữa, để tránh trường hợp có quá nhiều cụm được phân ra dẫn tới việc các cụm có thể chồng lấn hoặc không rõ ràng, nhóm chỉ chọn ra tham số tốt nhất ở những mô hình cho ra số cụm < 50.

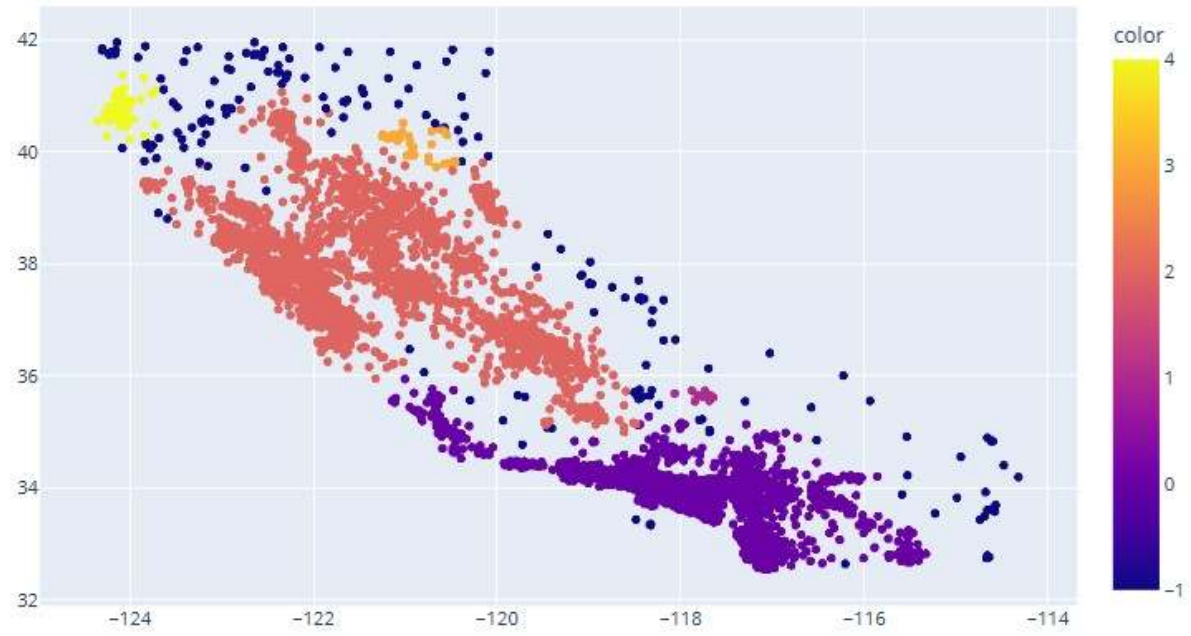
Từ đồ thị K - Distance, chia nhỏ các giá trị Epsilon trong khoảng từ 0.01 tới 1 và MinPts trong khoảng từ 2 tới 20.

Kết quả:

```
{'best_epsilon': 0.29285714285714287,
 'best_min_samples': 14,
 'best_labels': array([-1, -1, -1, ..., -1, -1,  4]),
 'best_score': 0.4066290757338104}
```

Kết quả sau khi sử dụng phương pháp grid search: Với Eps = 0.292, MinPts = 14 mô hình sẽ cho ra kết quả tốt nhất

Biểu diễn trực quan kết quả sau phân cụm:



Hình 25. Kết quả phân cụm dựa trên tọa độ địa lý (Esp = 0.29, MinPts = 14)

Nhận xét: Ở lần thử này kết quả cho ra tốt hơn rất nhiều so lần lần thử đầu tiên

4.4.2 Phân cụm lô đất dựa vào các thuộc tính còn lại

Tiếp tục sử dụng phương pháp grid search để tìm ra tham số tối ưu

Kết quả:

```
{'best_epsilon': 1.0,
 'best_min_samples': 5,
 'best_labels': array([-1, -1,  0, ...,  1,  1,  1]),
 'best_score': 0.6500872279887167}
```

Biểu diễn trực quan kết quả sau phân cụm cột **housing_median_age và total_rooms**

Hình 26. Kết quả phân cụm của housing_median_age và total_rooms

Biểu diễn trực quan kết quả sau phân cụm cột **housing_median_age** và **median_income**

Hình 27. Kết quả phân cụm cyả housing_median_age và median_income

4.4.3 Giảm chiều dữ liệu

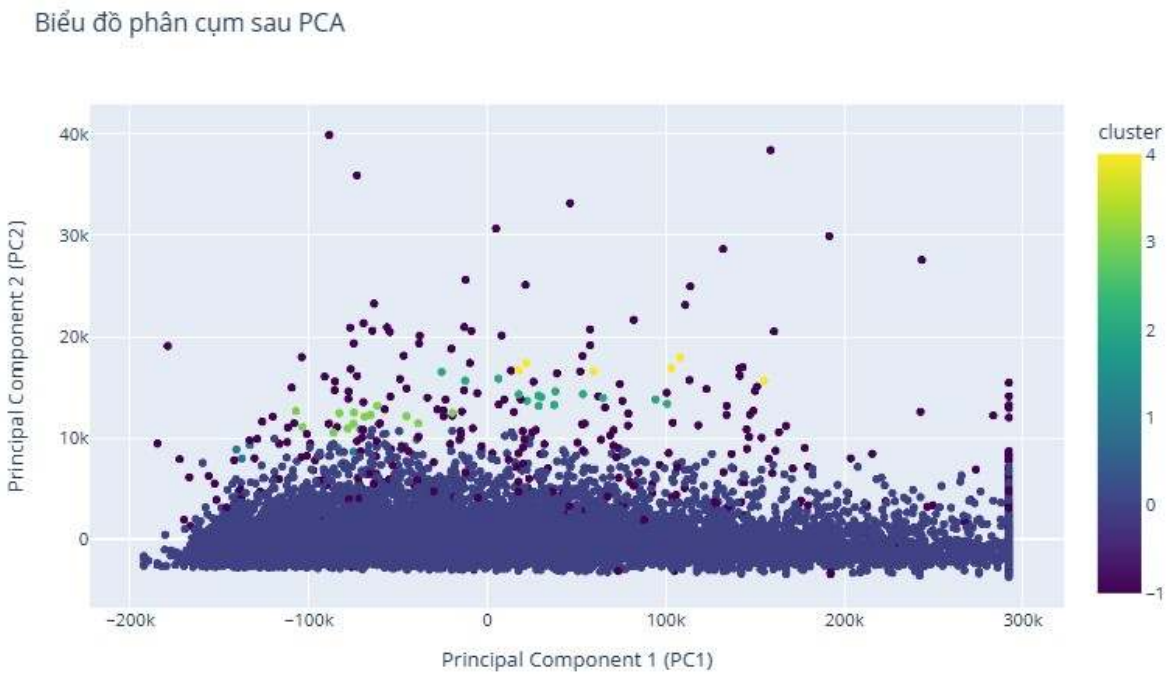
Ở phần này, nhóm dùng phương pháp PCA để giảm số chiều trong bộ dữ liệu về 2 chiều nhằm trực quan hóa tổng thể sau khi tiến hành phân cụm.

```
clusters = df['cluster']
df = df.drop(['cluster'], axis=1)

pca = PCA(n_components=2)
df_reduced = pd.DataFrame(pca.fit_transform(df), columns=['PC1', 'PC2'])
df_reduced['cluster'] = clusters
```

- ```
  fig = px.scatter(df_reduced, x='PC1', y='PC2', color='cluster', title='Biểu đồ PCA của df với cột cluster',
                   labels={'PC1': 'Principal Component 1 (PC1)', 'PC2': 'Principal Component 2 (PC2)'},
                   color_continuous_scale='viridis')
  ```
  ```

```
fig.show()
```



Hình 28. Biểu dồ phân cụm sau khi PCA

TÀI LIỆU THAM KHẢO

[1] Bài giảng học phần Khoa Học Dữ Liệu, “Orange Data Mining” khoa Công Nghệ Thông Tin Kinh Doanh, Đại học Kinh tế Tp.HCM, 2023.

[2] Bài giảng học phần Khoa Học Dữ Liệu, “Machine Learning” khoa Công Nghệ Thông Tin Kinh Doanh, Đại học Kinh tế Tp.HCM, 2023.

[3] Bài giảng học phần Biểu diễn trực quan dữ liệu, “Data Visualization” khoa Công Nghệ Thông Tin Kinh Doanh, Đại học Kinh tế Tp.HCM, 2023.

[4] Bộ dữ liệu "California housing" trên Kaggle.

[5] Jing, G. (no date) *CSE601 density-based clustering - university at Buffalo*, *cse.buffalo.edu*. Available at: https://cse.buffalo.edu/~jing/cse601/fa13/materials/clustering_density.pdf (Accessed: 11 December 2023).

[6] Ester, M. *et al.* (no date) *A density-based algorithm for discovering clusters in large spatial ...* Available at: https://cdn.aaai.org/KDD/1996/KDD96-037.pdf (Accessed: 11 December 2023).

[7] Deep *Ai Khanhblog* (no date a) *15.1. Phương pháp phân cụm dựa trên mật độ (Density-Based Clustering) - Deep AI KhanhBlog*. Available at: https://phamdinhkhanh.github.io/deepai-book/ch_ml/DBSCAN.html (Accessed: 12 December 2023).

[8] DBSCAN *clustering in ML: Density based clustering* (2023) *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/dbscan-clustering-in-ml-density-based-clustering/ (Accessed: 12 December 2023).

[9] Sklearn.*cluster.DBSCAN* (no date) *scikit*. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html (Accessed: 12 December 2023).

[10] Sharma, A. (2023) *How to master the popular DBSCAN Clustering Algorithm for machine learning*, *Analytics Vidhya*. Available at: https://www.analyticsvidhya.com/blog/2020/09/how-dbscan-clustering-works/ (Accessed: 12 December 2023).

[11] Sharma, A. (2023) *Clustering | Introduction, Different Methods, and Applications*, *Analytics Vidhya*. Available at: https://www.analyticsvidhya.com/blog/2016/11/an-introduction-to-clustering-and-different-methods-of-clustering/#Q1._What_is_agglomerative_clustering,_and_how_does_it_work (Accessed: 12 December 2023).