**ĐẠI HỌC UEH**

**TRƯỜNG CÔNG NGHỆ VÀ THIẾT KẾ**

**KHOA CÔNG NGHỆ THÔNG TIN KINH DOANH**

Logo

Description automatically generated

**ĐỒ ÁN MÔN HỌC**

**MÁY HỌC**

**ĐỀ TÀI:**

|  |  |
| --- | --- |
| **Giảng viên:** | TS. Nguyễn An Tế |
| **Mã lớp học phần:** | 23C1INF50904402 |
| **Sinh viên - MSSV:** | Lê Thị Cẩm Giang - 31211027269 |
|  | Bùi Tấn Hải - 31201024395 |
|  | Trương Gia Hào - 31211027638 |
|  | Lưu Quang Huy - 31211027642 |
|  | Bùi Lê Khang - 31211027644 |

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

LỜI CẢM ƠN

Mục lục:

[Phần 1: TỔNG QUAN 5](#_Toc153092257)

[1.1. Giới thiệu 5](#_Toc153092258)

[1.1.1. Hạn chế của thuật toán K-Means 5](#_Toc153092259)

[1.1.2. Hạn chế của thuật toán K-Mode 5](#_Toc153092260)

[1.1.3. Giới thiệu về K-Prototypes 5](#_Toc153092261)

[1.2. Cơ sở toán học 5](#_Toc153092262)

[1.2.1. Đo lường độ tương tự của K-Prototypes (Measure of Similarity) 5](#_Toc153092263)

[1.2.2. Hàm chi phí (Cost function) 7](#_Toc153092264)

[1.2.3. Cập nhật centroid 7](#_Toc153092265)

[Phần 2: Thực hiện các hàm bằng Python dựa trên cơ sở toán học 8](#_Toc153092266)

[2.1. Các hàm measure similarity 8](#_Toc153092267)

[2.1.1. Đo lường độ tương tự trên dữ liệu định lượng 8](#_Toc153092268)

[2.1.2. Đo lường độ tương tự trên dữ liệu định tính 8](#_Toc153092269)

[2.1.3. Hàm chi phí 9](#_Toc153092270)

[2.2. Các thuộc tính quan trọng trong class Kprototypes 9](#_Toc153092271)

[Phần 3: Phân cụm trên tập dữ liệu bằng thuật toán K-Prototypes 10](#_Toc153092272)

[3.1. Tổng quan về tập dữ liệu 10](#_Toc153092273)

[3.2. Áp dụng thuật toán K-Prototypes lên tập dữ liệu 11](#_Toc153092274)

[3.2.1. Silhouette Score 11](#_Toc153092275)

[3.2.2. Cost 12](#_Toc153092276)

[3.2.3. Thực hiện thuật toán K-Prototypes 13](#_Toc153092277)

[Phần 4: Đánh giá thuật toán K-Prototypes và so sánh với K-Means 14](#_Toc153092278)

[4.1. Scatter plot 14](#_Toc153092279)

[4.2. Silhouette score 15](#_Toc153092280)

[4.3. Đánh giá ưu nhược điểm so với K-Means 16](#_Toc153092281)

Mục lục hình ảnh:

[Hình 1 Thuật toán tính khoảng cách Euclide 9](#_Toc153276984)

[Hình 2 Matching Similarity 9](#_Toc153276985)

[Hình 3 Jaccard Similarity 9](#_Toc153276986)

[Hình 4 Hàm tính toán Cost 10](#_Toc153276987)

[Hình 5 Biểu đồ Silhouette Score 13](#_Toc153276988)

[Hình 6 Biểu đồ kết quả hàm Cost 13](#_Toc153276989)

[Hình 7 Áp dung thuật toán K-Prototypes 14](#_Toc153276990)

[Hình 8 Biểu đồ Scatter cho các cụm 14](#_Toc153276991)

[Hình 9 Đánh giá thuật toán khi k=3 15](#_Toc153276992)

[Hình 10 Đánh giá thuật toán khi k=7 15](#_Toc153276993)

[Hình 11 Silhouette Score cho mỗi giá trị k khác nhau 16](#_Toc153276994)

# TỔNG QUAN

## Giới thiệu

### Giới thiệu

* Học máy không giám sát là một kiểu vấn đề ở trong các bài toán học máy khi mà input của các dữ liệu huấn huyện là các vector nhưng không có target. Ý tưởng của các bài toán học máy này là sẽ gom các cụm có cùng độ tương tự (similarity), quy luật (pattern) và sự khác nhau (diferences).
* Một trong những thuật toán thường được sử dụng trong bài toán học máy không giám sát là K-Means clustering. Tuy nhiên thuật toán không hoạt động được trên các dữ liệu có chứa dạng định tính. Điều này xảy ra bởi vì hàm chi phí của K-Means được tính bởi Euclide Distance chỉ sử dụng được trên dữ liệu định lượng.
* Một thuật toán khác tương tự với K-Mean là K-Mode hoạt động được trên dữ liệu định tính nhờ vào việc sử dụng Matching Similarity để làm hàm chi phí, và update các centroid bằng cách lấy mode thay vì lấy mean như K-Mean. Vì thế K-Mode chỉ sử dụng được trên dữ liệu định tính.
* Đối mặt với vấn đề đó, Huang đã đưa ra một thuật toán là K-Prototype, được sử dụng được cả trên dữ liệu định lượng và định tính. K-Prototype kết hợp giữa K-Mean và K-Mode ở cách tính hàm chi phí và update các centroid để có thể cải thiện được điểm yếu của hai thuật toán trên.

### Hạn chế của thuật toán K-Means

* Trong thuật toán K-Means, để đo lường sự tương đồng (similarity) giữa các điểm dữ liệu và trung tâm của các cụm, thường sử dụng khoảng cách Euclidean. Khoảng cách Euclidean là một phép đo khoảng cách giữa hai điểm trong không gian nhiều chiều.
* Tuy nhiên khoảng Euclidean có khuyết điểm trong ngữ cảnh đặc trưng, để hiểu rõ hơn ta cùng điểm qua một số tính chất quan trọng của nó:
* Hiệu quả trong việc xử lý các tập dữ liệu lớn.
* Nó thường kết thúc ở mức tối ưu cục bộ (MacQueen, 1967; Selim và Ismail, 1984).
* Nó chỉ hoạt động trên các giá trị số.
* Các cụm có hình dạng lồi (Anderberg, 1973).
* Dựa vào đặc tính K-Means chỉ hoạt động dựa trên các giá trị số và sự cố này xảy ra khi hàm chi phí trong K-Means được tính bằng khoảng cách Euclidean. Do vậy nếu trong trường hợp dữ liệu là phân loại thì không thể phân cụm được, chúng ta có thể sử dụng phương pháp encoding để chuyển dữ liệu phân loại thành số, tuy nhiên việc áp dụng phương pháp đó có một số hạn chế như độ phức tạp sẽ tăng lên, số chiều tăng lên.

### Hạn chế của thuật toán K-Mode

* Khi làm việc với dữ liệu phân loại (categorical data), phương pháp K-Modes được phát triển nhằm mục đích phân nhóm các quan sát dựa trên sự tương đồng giữa các giá trị categorical. Để đo lường sự tương đồng trong K-Modes, cần sử dụng các phép đo tương đồng phù hợp với tính chất của dữ liệu categorical.
* Ngược lại với K-Means chỉ thực hiện để với kiểu dữ định tính (numerical data) thì K-Modes chỉ phù hợp với kiểu dữ liệu định tính (categorical data). Tuy vậy nhưng vẫn có thể sử dụng phương pháp này trong trường hợp kiểu dữ là số bằng cách chia các biến số thành các phạm vi (bins) và sau đó xem xét chúng như là các biến phân loại. Điều này có thể giúp bạn áp dụng K-Modes trên dữ liệu số, mặc dù giữa các giá trị cụ thể có thể không có sự giống nhau như giữa các giá trị phân loại. Tóm lại chúng ta có thể sử dụng các cách để biến đổi chúng về lại kiểu dữ liệu phù hợp để tính toán nhưng việc làm ấy là không phổ biến mà chỉ mang tính chất chữa cháy khi không có giải pháp thay thế.

### Giới thiệu về K-Prototypes

* K-Prototypes là một biến thể của thuật toán K-Means và k-Mode được thiết kế để xử lý dữ liệu kết hợp giữa các biến số và các biến phân loại. Trong khi K-Means và K-Modes tập trung lần lượt vào dữ liệu số hoặc dữ liệu phân loại, K-Prototypes kết hợp cả hai loại biến để phân cụm dữ liệu.
* Thuật toán này sử dụng khoảng cách Euclidean để đo lường sự tương đồng giữa các biến định lượng và sử dụng độ đo Jaccard, Matching để đo lường sự tương đồng giữa các biến định tính. Bằng cách này, K-Prototypes giúp giải quyết vấn đề phân cụm trong các tập dữ liệu đa dạng về loại dữ liệu

## Cơ sở toán học

### Đo lường độ tương tự của K-Prototypes (Measure of Similarity)

* Công thức toán học

Giả sử là một tập hợp gồm n phần tử và trong đó m biểu thị vị trí các phần tử và i biểu thị cụm thứ i.

* + - 1. Công thức chung đo lường độ tương tự
* Công thức chung để đo độ tương tự được ký hiệu như sau:

Trong đó là centroid cho cụm L.

* + - 1. Công thức đo lường mức độ tương tự của biến định lượng:
* Euclidean Distance:
* Thước đo độ tương tự của các biến định lượng được sử dụng là khoảng cách Euclidean được ký hiệu như sau:

Với là giá trị của biến định lượng thứ j, là giá trị trung bình của centroid cho biến định lượng thứ j cụm m.

* + - 1. Công thức đo lường mức độ tương tự của biến định tính
* Mức độ tương tự cho các biến định tính được biểu thị như sau:

Ở trên là công thức tổng quát đo lường mức độ tương tự của biến định tính. Ta sẽ giới thiệu một vài hàm đo độ tương tự ở phần tiếp theo

1. Matching Similarity Function

* Độ tương tự bằng Matching Similarity Function cho các biến định tính được biểu thị như sau:

Trong đó biểu thị trọng số của các biến phân loại cho cụm L là độ lệch chuẩn của các biến số trong mỗi cụm. biểu thị các biến phân loại, là chế độ cho các biến j cụm L, và biểu thị số lượng biến phân loại.

1. Jaccard Similarity Function

* Ta có thể dùng Jaccard Similarity Function để đo độ tương tự cho biến định tính như sau:
* Công thức trên làm tăng độ tương tự đối tượng trong cụm với các biến phân loại sao cho kết quả sẽ tốt hơn khi biểu thị trọng số cho trong đó

(6)

Trong đó là tần số của trong cụm và [] là số lượng đối tượng trong cụm , và là tần số của trong toàn bộ dữ liệu.

* + - 1. Công thức tổng quát đo lường độ tương tự
* Từ (1) và (5), ta có công thức tổng quát đo lường độ tương tự của cả biến định tính và biến định lượng:

### Hàm chi phí (Cost function)

* Cost Function cho kiểu dữ liệu hỗn hợp (định lượng và định tính) như sau
* là chi phí đại diện cho tổng chi phí của tất cả các biến định lượng và định tính cho toàn bộ đối tượng trong cụm L . Chi phí đạt cực tiểu khi được tính theo phương trình sau.

Trong đó, là số lượng các đối tượng trong cụm L.

* Tiếp theo, các biến phân loại, ví dụ , là tập hợp các giá trị duy nhất trong mỗi biến phân loại j và là xác suất cho c trong cụm L. Vì vậy, chi phí có thể được viết lại như sau:

Chi phí đạt cực tiểu khi và chỉ khi cho với mọi biến định tính

### Cập nhật centroid

Các bước update centroid:

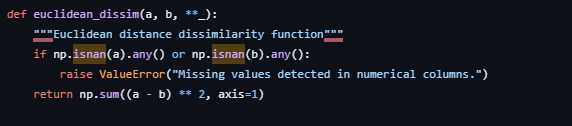
* Bước 1: Chọn số lượng phân cụm và khởi tạo các centroid
* Bước 2: Tính khoảng cách từ từng điểm dữ liệu đến các centroid bằng num\_dissim(data, centroid) + cat\_dissim(data, centroid)
* Bước 3: Gắn phân cụm cho các điểm dữ liệu cho các centroid có khoảng cách thấp nhất.
* Bước 4: Reset lại các centroid dựa vào các điểm dữ liệu đã được phân cụm.   
  Ở thuộc tính dạng định lượng thì sẽ lấy trung bình mỗi thuộc tính của tất cả các điểm dữ liệu trong mỗi cụm.   
  Ở thuộc tính dạng định tính thì sẽ lấy mode của các thuộc tính của điểm dữ liệu đó trong mỗi cụm.
* Bước 5: Lặp lại quá trình cho đến khi gặp điều kiện dừng hoặc thuật toán hội tụ
* Thuật toán ở trong thư viện mà nhóm sử dụng, tác giả có tính Cost để chạy thuật toán nhiều lần với khởi tạo centroid khác nhau, rồi chọn thuật toán có tổng Cost thấp nhất.
* Ở bước 2 ta có thể thêm các hệ số alpha\*num\_dissim(data, centroid) + beta\*cat\_dissim(data, centroid) để gán trọng số cho chi phí của kiểu dữ liệu định lượng và định tính như công thức ở phía trên

# Thực hiện các hàm bằng Python dựa trên cơ sở toán học

## Các hàm measure similarity

### Đo lường độ tương tự trên dữ liệu định lượng

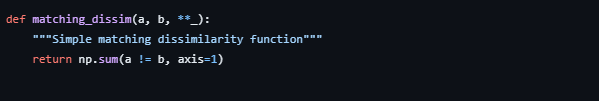
Euclide Distance



Hình 1 Thuật toán tính khoảng cách Euclide

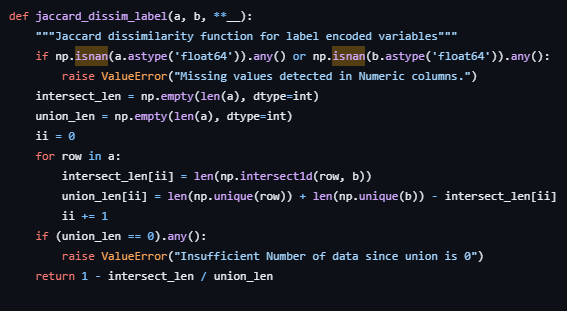
### Đo lường độ tương tự trên dữ liệu định tính

* + - 1. Matching Similarity



Hình 2 Matching Similarity

* + - 1. Jaccard Similarity



Hình 3 Jaccard Similarity

### Hàm chi phí



Hình 4 Hàm tính toán Cost

## Các thuộc tính quan trọng trong class Kprototypes

* Hàm \_\_init\_\_(args):

|  |  |  |
| --- | --- | --- |
| Parameters | Kiểu dữ liệu | Ý nghĩa |
| **n\_clusters** | int | Số cluster muốn phân cụm |
| **max\_iter** | int | Điều kiện dừng cho mỗi lần chạy |
| **num\_dissim** | function | Hàm tính similarity cho dữ liệu định lượng (numerical variable) |
| **cat\_dissim** | function | Hàm tính similarity cho dữ liệu định tính (categorical variable) |
| **gamma** | float | Tham số để cho rằng biến định lượng quan trọng hơn hay biến định tính quan trọng hơn |
| **n\_init** | int | Chạy n lần với khác nhau, thuật toán sẽ chọn lần chạy có tổng cost thấp nhất |
| **init** | str | Ở đây có hai phương pháp init là “Huang” và “Cao”. Default là “Cao” |

* Hàm fit(args)

|  |  |  |
| --- | --- | --- |
| **Parameters** | **Kiểu dữ liệu** | **Ý nghĩa** |
| **X** | array, shape=[n\_samples, n\_features] | Dữ liệu huấn luyện |
| **categorical** | List (e.x. [1,2]) | List chứa indice của các cột categorical |

* Hàm predict(args)

|  |  |  |
| --- | --- | --- |
| **Parameters** | **Kiểu dữ liệu** | **Ý nghĩa** |
| **X** | array, shape=[n\_samples, n\_features] | Dữ liệu dự đoán |
| **categorical** | List | List chứa indice của các cột categorical |

# Phân cụm trên tập dữ liệu bằng thuật toán K-Prototypes

## Tổng quan về tập dữ liệu

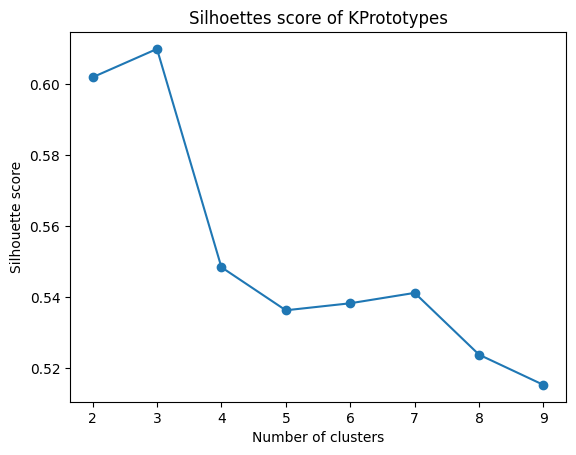
* Bộ dữ liệu đưa ra các thông tin về khách hàng của một công ty. Giúp doanh nghiệp hiểu rõ về khách hàng và làm cho việc thay đổi các sản phẩm dễ dàng hơn dựa trên các nhu cầu cụ thể, thói quen mua sắm cũng như các quan tâm về nhiều mặt hàng khác nhau của khách hàng
* Bộ dữ liệu có thông tin về khách hàng, sản phẩm mà khách hàng mua, các mã giảm giá khách hàng đã dùng và thông tin về nơi mua của khách hàng.
* Trong phạm vi thực hiện K-Prototypes, ta sẽ chỉ dùng thông tin của khách hàng để ứng dụng thuật toán.
* Các biến không hữu ích như ID, Birthday đã được xử lý bằng cách loại bỏ, cũng như ta sẽ chỉ trích xuất năm sinh của khách hàng từ Birthday. Các missing value, duplicate data cũng như blank rows đã được loại bỏ.

|  |  |  |
| --- | --- | --- |
| **Tên biến** | **Kiểu** | **Ý nghĩa** |
| **Year\_Birth** | Numeric | Năm sinh của khách hàng |
| **Education** | Categorical | Trình độ học vấn của khách hàng |
| **Marital\_Status** | Categorical | Tình trạng hôn nhân của khách hàng |
| **Income** | Numeric | Thu nhập của khách hàng hàng năm |
| **Recency** | Numeric | Số ngày kể từ lần cuối khách hàng mua |
| **Kidhome** | Numeric | Số lượng trẻ em trong nhà |
| **Teenhome** | Numeric | Số lượng trẻ vị thành niên trong nhà |
| **Complain** | Numeric | 1 nếu khách hàng có phàn nàn trong vòng 2 năm, 0 nếu không có |

## Áp dụng thuật toán K-Prototypes lên tập dữ liệu

### Silhouette Score

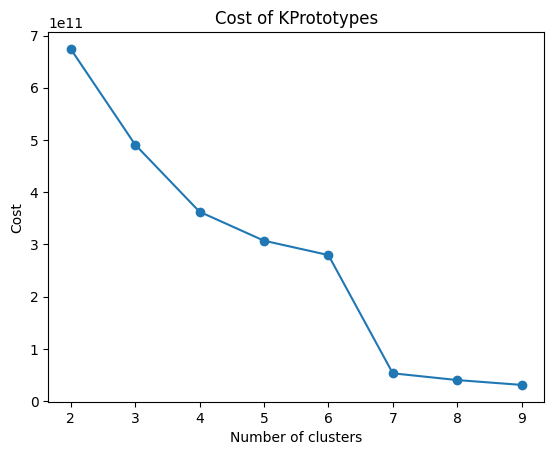
* Silhouette Score ở số cluster bằng 3 là cao nhất, số cluster bằng 4 thì Silhouette Score bị giảm mạnh.



Hình 5 Biểu đồ Silhouette Score

### Cost

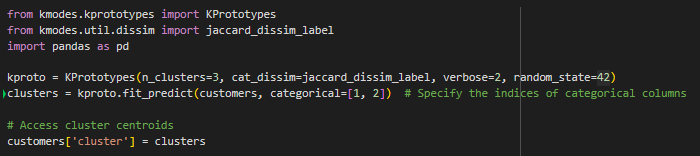
* Cost giảm nhiều ở số cluster bằng 7
* Vì ta muốn segment customer, nên để dễ nhìn vào các cụm để phân tích, ta sẽ chọn số cluster là 3



Hình 6 Biểu đồ kết quả hàm Cost

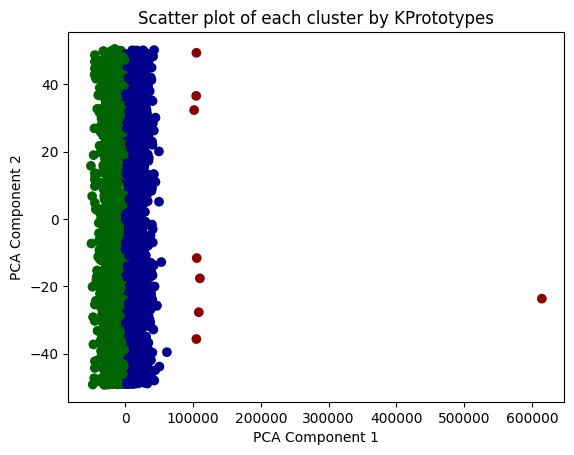
### Thực hiện thuật toán K-Prototypes

* Dựa vào đánh giá qua Silhoette Score và Cost, ta sẽ để số cluster bằng 3.
* Hàm đo lường độ tương tự cho biến định tính ta dùng ở đây là Jaccard Similarity



Hình 7 Áp dung thuật toán K-Prototypes

* Ta One Hot Encoding các dữ liệu định tính, và thực hiện giảm chiều dữ liệu bằng PCA. Bên dưới là kết quả phân cụm.

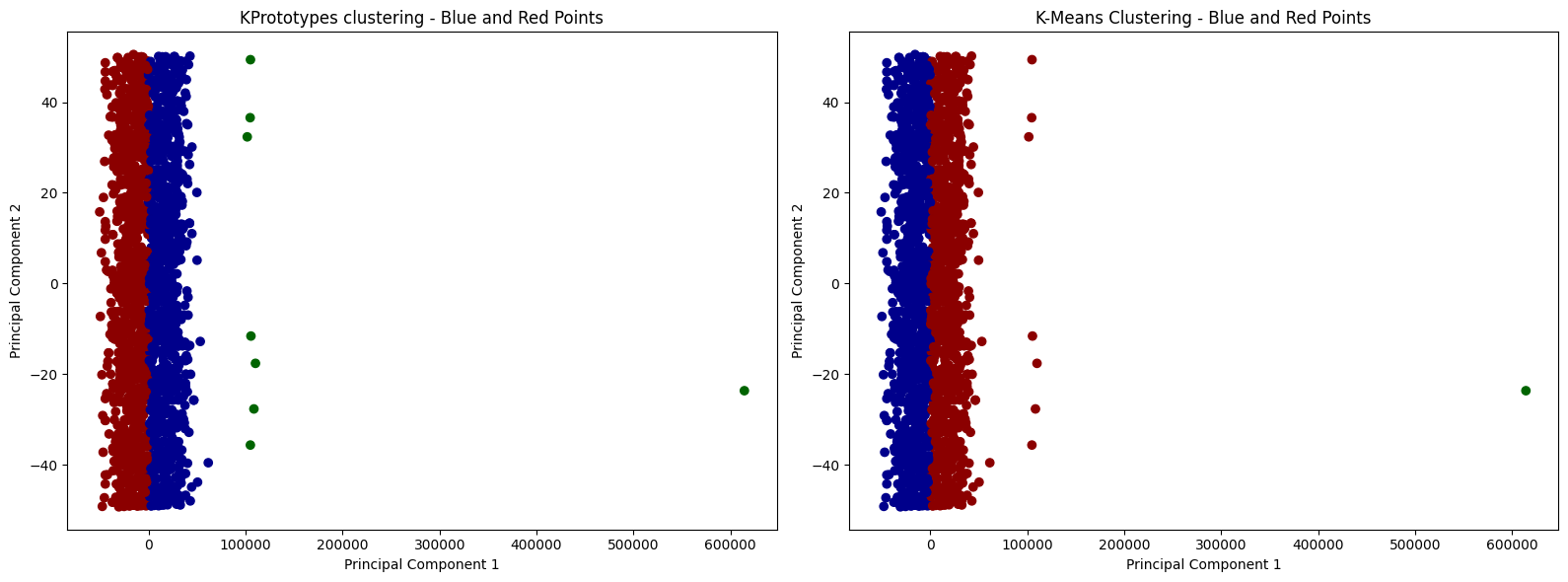


Hình 8 Biểu đồ Scatter cho các cụm

# Đánh giá thuật toán K-Prototypes và so sánh với K-Means

## Scatter plot

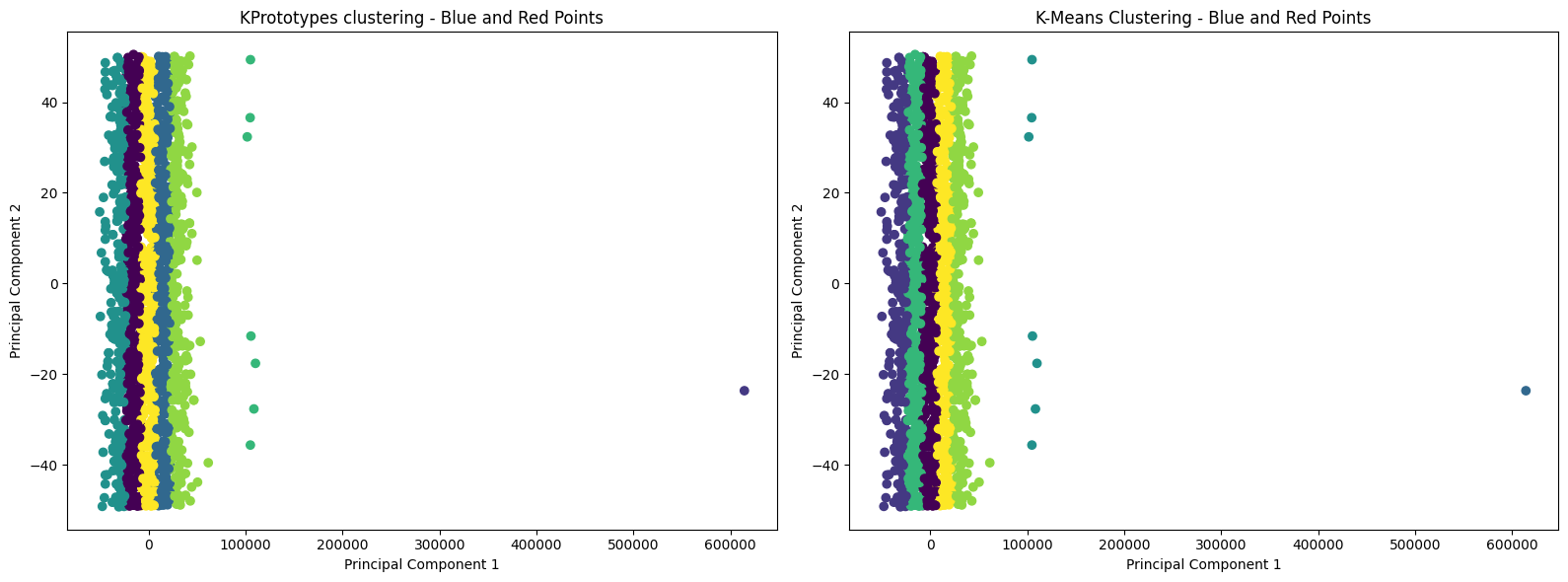
* Ở dưới là kết quả của thuật toán được trực quan bằng Scatter Plot so sánh với K-Means khi số cluster bằng 3



Hình 9 Đánh giá thuật toán khi k=3

* Kết quả của thuật toán được trực quan bằng Scatter Plot so sánh với K-Means khi số cluster bằng 7

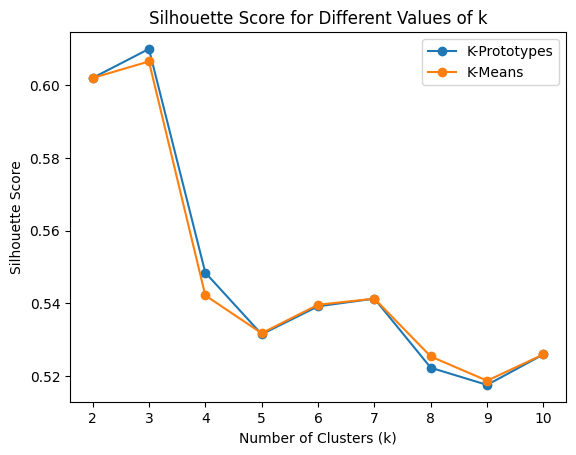
Kết quả phân cụm ở hai thuật toán có sự tương đồng.



Hình 10 Đánh giá thuật toán khi k=7

## Silhouette score

* Kết quả đánh giá qua Silhouete Score của K-Prototypes có phần tương đồng với K-Means. Ở trường hợp số lượng cluster bằng 3 thì K-Prototypes có Silhouette Score cao hơn nhưng không nhiều.
* Kết quả Silhouette Score của thuật toán K-Prototypes phụ thuộc vào cách khởi tạo centroid và n\_init. Nếu n\_init thấp thì kết quả phân cụm sẽ không được tối ưu



Hình 11 Silhouette Score cho mỗi giá trị k khác nhau

## Đánh giá ưu nhược điểm so với K-Means

* Ưu điểm
* Thuật toán có thể chạy được trên cả dữ liệu định lượng và định tính
* Giữ lại các thông tin về kiểu dữ liệu định tính, các centroid cũng có thông tin về dữ liệu định tính, điều này giúp mình có thể giúp ta diễn giải có ý nghĩa hơn về các biến định tính trong từng cluster
* Nhược điểm
* Nhạy cảm với quá trình init các centroid. Khởi tạo các centroid khác nhau có thể dẫn đến các kết quả khác nhau.
* Thuật toán hội tụ chậm hơn K-Means. Nếu chọn n\_init thấp thì thuật toán sẽ không chọn được kết quả tối ưu, tuy nhiên nếu chọn n\_init cao thì thuật toán sẽ chạy rất chậm. Ở trong trường hợp phía trên để có được Silhouette Scores cho K-Prototypes với các số lượng cluster khác nhau thì ta mất hơn 30 phút, trong khi đó K-Means chỉ mất chưa đến 2 phút (Phương pháp init của K-Mean ở trên là K-Mean++).

**Tài liệu tham khảo:**

1. [The k-prototype as Clustering Algorithm for Mixed Data Type (Categorical and Numerical)](https://freedium.cfd/https:/towardsdatascience.com/the-k-prototype-as-clustering-algorithm-for-mixed-data-type-categorical-and-numerical-fe7c50538ebb)
2. [Extensions to the k-Means Algorithm for Clustering](https://citeseerx.ist.psu.edu/pdf/3cb884ae32eb68b93ffd98f45404c4ecb208763b)
3. [Kmodes Github](https://github.com/nicodv/kmodes)