K-mean Neighbor

Time Series Team

Ngày 12 tháng 8 năm 2025

Mục lục

Phần I: Lý Thuyết

1. Giới thiệu

1 Bối cảnh và Động lực

Trong thời đại bùng nổ dữ liệu hiện nay, các tổ chức và doanh nghiệp đều đối mặt với khối lượng dữ liệu ngày càng tăng lên một cách nhanh chóng. Việc khai thác, phân tích và tổ chức dữ liệu trở thành yếu tố then chốt giúp đưa ra các quyết định chiến lược, tối ưu hóa hoạt động và nâng cao trải nghiệm khách hàng. Tuy nhiên, dữ liệu thô thường không có cấu trúc rõ ràng hoặc không có nhãn phân loại, khiến việc phân nhóm hoặc nhận diện các mẫu dữ liệu trở nên phức tạp.

Ví dụ, giả sử bạn có một cơ sở dữ liệu lớn chứa thông tin chi tiết về hàng ngàn khách hàng, nhưng không có thông tin phân loại cụ thể về nhóm khách hàng. Làm thế nào để bạn có thể:

- Phân chia khách hàng thành các nhóm có hành vi tiêu dùng tương đồng như nhóm VIP và nhóm khách hàng thông thường?
- Phân loại các sản phẩm dựa trên đặc điểm chung nhằm phục vụ tốt hơn cho việc tiếp thị và phát triển sản phẩm?
- Phát hiện ra các mẫu hoặc xu hướng ẩn trong dữ liệu mà không cần dựa vào dữ liệu đã được gán nhãn trước?

Đây chính là lúc các thuật toán học máy không giám sát (unsupervised learning) như **K-means clustering** trở nên rất hữu ích. Thuật toán này cho phép tự động phân nhóm dữ liệu dựa trên sự tương đồng giữa các điểm dữ liệu mà không cần thông tin nhãn từ trước. Nhờ vậy, K-means giúp phát hiện cấu trúc tiềm ẩn trong dữ liệu và hỗ trợ các bước phân tích tiếp theo.

2 Vấn đề cốt lõi

Để minh họa cho bài toán phân nhóm không giám sát, hãy xem xét một ví dụ đơn giản với dữ liệu về tuổi và chi tiêu hàng tháng của 9 khách hàng như sau:

Index	**Tuổi**	**Chi tiêu (USD)**
1	18	80
2	20	90
3	22	85
4	30	50
5	34	64
6	40	60
7	60	30
8	66	40
9	70	25

Bảng 1: Dữ liệu mẫu về khách hàng

Mục tiêu của bài toán là phân chia tự động 9 khách hàng này thành 3 nhóm khác nhau, ví dụ như nhóm *Trẻ*, *Trung niên* và *Cao tuổi*, dựa trên đặc điểm tuổi tác và chi tiêu, mà không cần biết trước nhóm nhãn phân loại. Việc này sẽ giúp doanh nghiệp hiểu được cấu trúc phân khúc khách hàng một cách khách quan và hiệu quả hơn, phục vụ cho các chiến lược marketing và chăm sóc khách hàng cá nhân hóa.

Bài toán trên là ví dụ điển hình cho việc ứng dụng kỹ thuật *clustering* trong khai phá dữ liệu, đặc biệt là thuật toán **K-means clustering**, một trong những phương pháp phân nhóm phổ biến và đơn giản nhất hiện nay.

2. Lý thuyết K-Means

1 Định nghĩa và Khái niệm cơ bản

K-Means là thuật toán học không giám sát (unsupervised learning) thuộc nhóm phân cụm (clustering), nhằm chia tập dữ liệu gồm n điểm dữ liệu $\{x_1, x_2, \ldots, x_n\}$ với $x_i \in \mathbb{R}^d$ thành k cụm $\{C_1, C_2, \ldots, C_k\}$ sao cho:

- **Cohesion (Tính gắn kết)**: Các điểm trong cùng một cụm có độ tương tự cao nhất có thể
- **Separation (Tính phân tách)**: Các điểm thuộc cụm khác nhau có độ tương tự thấp nhất có thể
- **Completeness (Tính đầy đủ)**: Mọi điểm dữ liệu đều được gán vào đúng một cụm

2 Khái niệm Centroid và Cluster

Centroid μ_i của cụm C_i là trung tâm hình học (geometric center) của tất cả các điểm trong cụm đó:

$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x \tag{1}$$

Trong đó $|C_i|$ là số lượng điểm trong cụm C_i.

Cluster Assignment là việc gán điểm x_j vào cụm C_i được thực hiện dựa trên nguyên tắc khoảng cách gần nhất:

$$\text{label}(x_j) = \arg \min_{i \in \{1,2,\ldots,k\}} d(x_j, \mu_i) \tag{2}$$

Trong đó $d(x_j, \mu_i)$ là hàm khoảng cách giữa điểm x_j và centroid μ_i.

3 Công thức Toán học

Cho tập dữ liệu $\{x_1, x_2, \ldots, x_n\}$ với $x_i \in \mathbb{R}^d$.

Mục tiêu của K-Means là tìm k cụm $S = \{S_0, S_1, \ldots, S_{k-1}\}$ để minimize:

$$\text{WCSS} = \sum_{i=0}^{k-1} \sum_{x \in S_i} \|x - c_i\|^2 \tag{3}$$

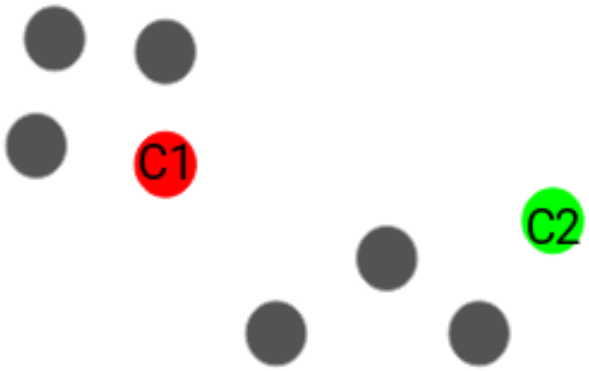
Trong đó:

- c_i là centroid của cụm S_i
- $\|x - c_i\|^2$ là khoảng cách Euclidean bình phương
- WCSS (Within-Cluster Sum of Squares) là tổng bình phương khoảng cách trong cụm

4 Thuật toán K-Means

Thuật toán K-Means hoạt động như sau:

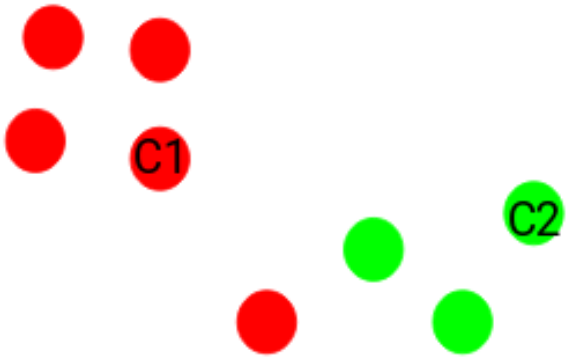
1. **Đầu vào:** Tập dữ liệu $X = \{x_1, x_2, \ldots, x_n\}$ và số lượng cụm k.
2. **Đầu ra:** Tập các centroid $\{c_1, c_2, \ldots, c_k\}$ và nhãn cụm tương ứng cho mỗi điểm dữ liệu $\{l_1, l_2, \ldots, l_n\}$.
3. **Khởi tạo:** Chọn ngẫu nhiên k centroid ban đầu.



Hình 1: Chọn k centroid ban đầu

4. **Khởi tạo biến** `converged` bằng `false` để kiểm soát vòng lặp.
5. **Lặp cho đến khi hội tụ hoặc đạt số lần lặp tối đa:**
   - **Bước gán nhãn (Assignment Step):** Mỗi điểm dữ liệu x_i được gán nhãn cụm l_i bằng cách chọn centroid gần nhất:

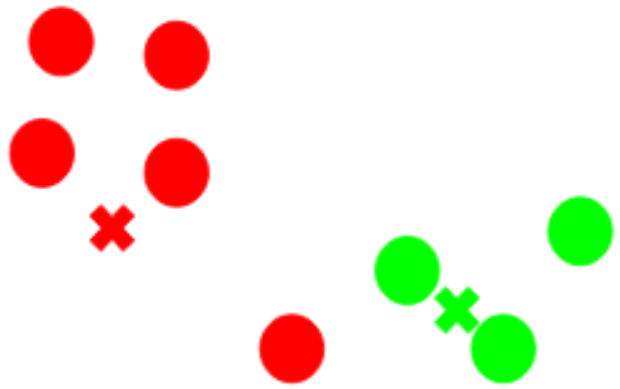
$$l_i = \arg\min_j \|x_i - c_j\|^2$$



Hình 2: Bước gán nhãn

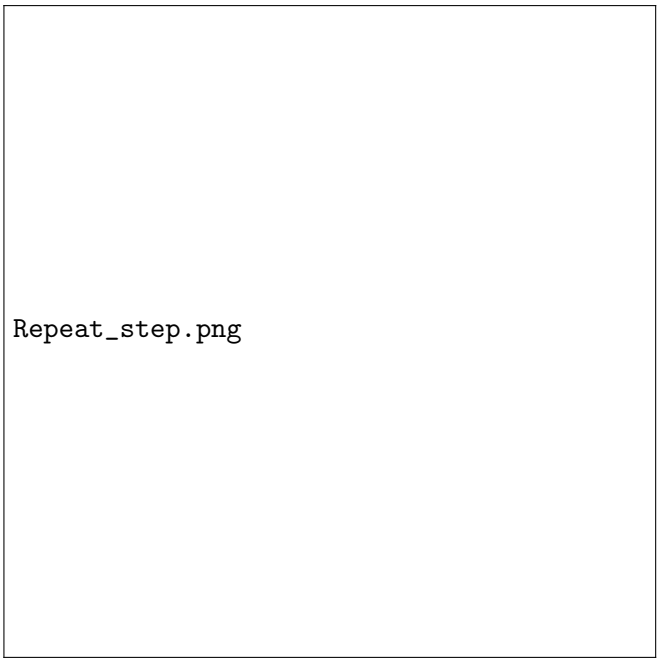
- **Bước cập nhật centroid (Update Step):** Tính lại vị trí centroid cho mỗi cụm j dựa trên trung bình các điểm thuộc cụm:

$$c_j^{(t+1)} = \frac{1}{|S_j^{(t)}|} \sum_{x \in S_j^{(t)}} x$$



Hình 3: Cập nhật centroids

- **Cập nhật biến** `converged` dựa trên sự thay đổi của centroid. Nếu các centroid không thay đổi hoặc thay đổi rất nhỏ, đặt `converged` = `true`.



Hình 4: Quay lại bước 5

Kết thúc khi thuật toán hội tụ hoặc đạt số lần lặp tối đa, trả về centroid và nhãn cụm của các điểm dữ liệu.

5 Các thuộc tính của thuật toán K-means Clustering

Hãy xem một ví dụ khác về thuật toán phân cụm K-means. Chúng ta sẽ dùng ví dụ về một ngân hàng muốn phân đoạn khách hàng của mình. Để đơn giản, giả sử ngân hàng chỉ dùng thu nhập và nợ để phân nhóm khách hàng. Họ thu thập dữ liệu khách hàng và dùng biểu đồ phân tán để trực quan hóa:

Trục X thể hiện thu nhập khách hàng, trục Y là số tiền nợ. Ở đây, chúng ta có thể dễ dàng nhận thấy khách hàng được phân thành 4 nhóm khác nhau, như hình dưới:

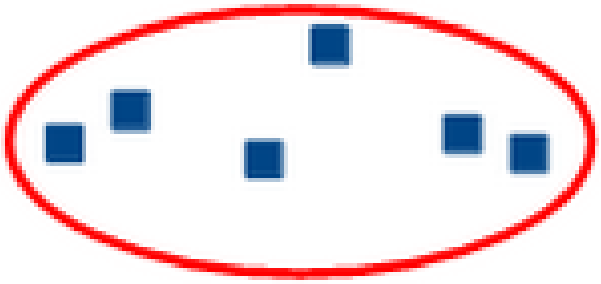
Đây chính là cách clustering (phân cụm) giúp tạo ra các nhóm (cụm) từ dữ liệu. Ngân hàng có thể dùng các cụm này để xây dựng chiến lược và đưa ra ưu đãi cho khách hàng. Bây giờ hãy cùng tìm hiểu các thuộc tính của các cụm này.

5.1 Thuộc tính thứ nhất của thuật toán K-means

Tất cả các điểm dữ liệu trong một cụm nên có tính tương đồng với nhau. Ví dụ với trường hợp trên:

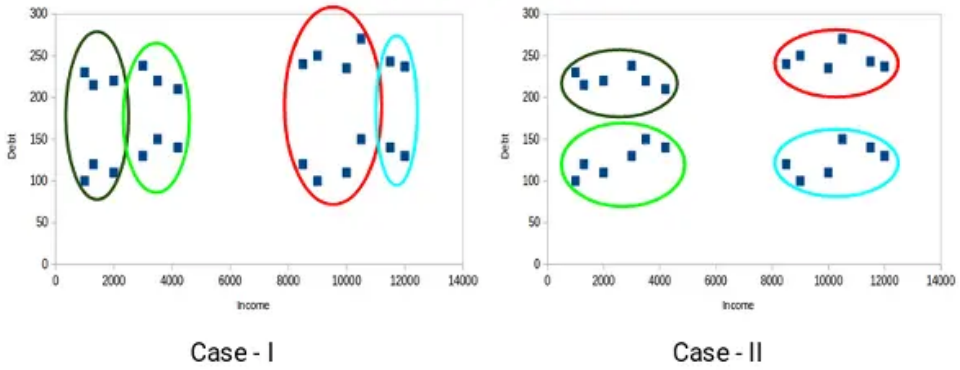
Nếu khách hàng trong một cụm không giống nhau, thì nhu cầu của họ cũng khác nhau, đúng không? Nếu ngân hàng áp dụng cùng một chương trình ưu đãi cho tất cả khách hàng đó, có thể họ sẽ không hài lòng, và sự quan tâm đến ngân hàng sẽ giảm đi. Điều này không lý tưởng.

Việc có các điểm dữ liệu tương đồng trong cùng một cụm giúp ngân hàng thực hiện các chiến dịch tiếp thị mục tiêu hiệu quả hơn. Bạn có thể liên tưởng các ví dụ tương tự trong cuộc sống và xem clustering ảnh hưởng như thế nào đến chiến lược kinh doanh.



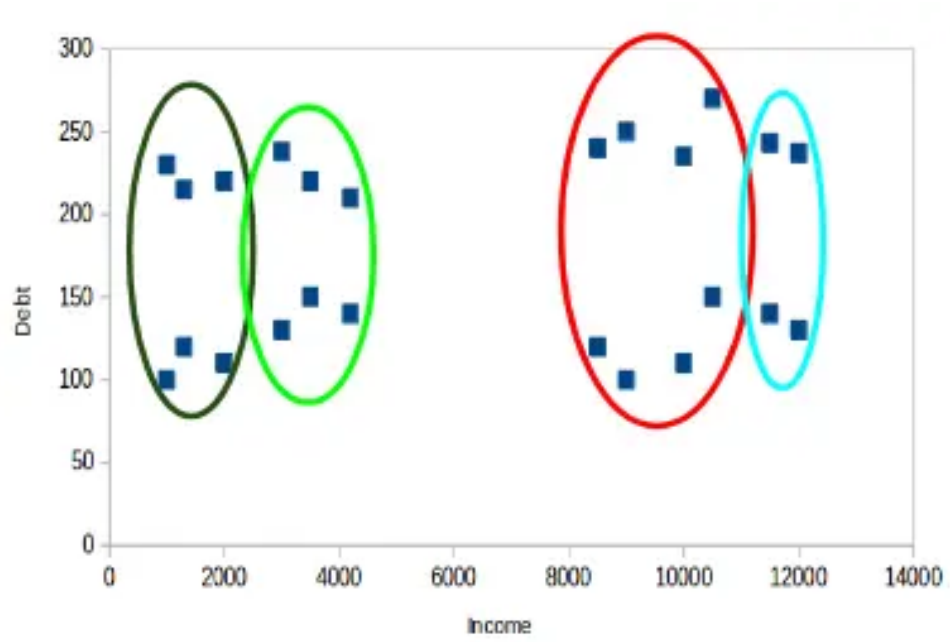
5.2 Thuộc tính thứ hai của thuật toán K-means

Các điểm dữ liệu thuộc các cụm khác nhau nên khác biệt càng nhiều càng tốt. Điều này có thể dễ hiểu nếu bạn đã hiểu thuộc tính đầu tiên. Cùng lấy lại ví dụ trên để minh họa:



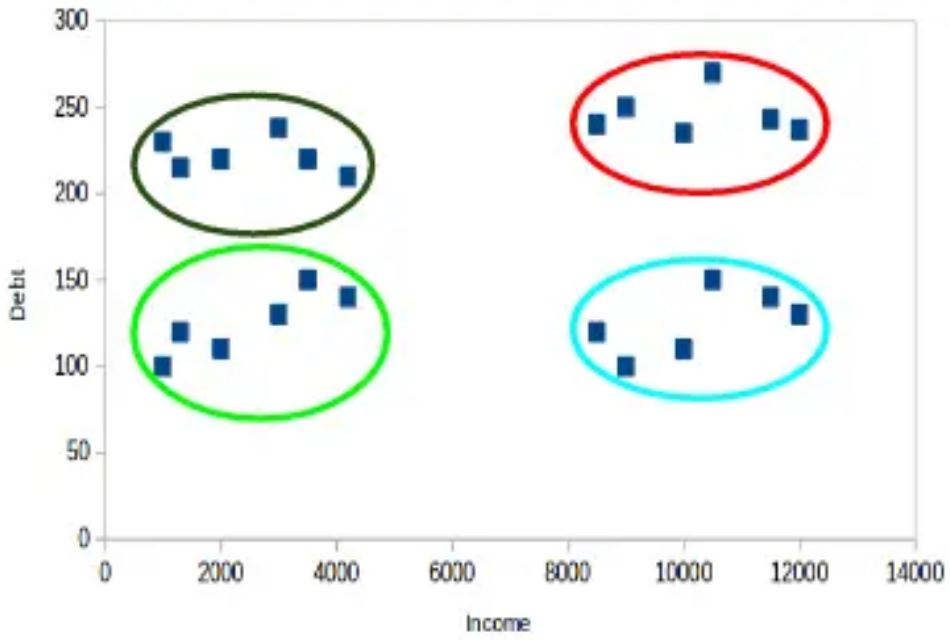
Case - I Case - II

Bạn nghĩ trường hợp nào sẽ cho kết quả phân cụm tốt hơn? Ở trường hợp I:



Case - I

Các khách hàng trong cụm đỏ và cụm xanh khá giống nhau. 4 điểm ở cụm đỏ có đặc tính tương tự với 2 điểm ở cụm xanh. Họ đều có thu nhập và nợ cao. Ở đây, chúng ta phân cụm sai lệch khi nhóm các khách hàng này riêng biệt. Còn trong trường hợp II:



Case - II

Các điểm ở cụm đỏ khác hoàn toàn so với cụm xanh. Tất cả khách hàng trong cụm đỏ có thu nhập

và nợ cao, trong khi cụm xanh có thu nhập cao nhưng nợ thấp. Rõ ràng cụm phân chia trong trường hợp II hợp lý hơn.

Như vậy, các điểm dữ liệu từ các cụm khác nhau nên khác biệt nhiều nhất có thể để tạo thành các cụm có ý nghĩa hơn. Thuật toán K-means dùng phương pháp lặp để tìm phân cụm tối ưu bằng cách giảm thiểu tổng bình phương khoảng cách giữa các điểm và centroid của cụm.

5.3 Tại sao chúng ta cần phân cụm?

Chúng ta đã hiểu phân cụm là gì và các thuộc tính khác nhau của cụm. Vậy tại sao phải dùng phân cụm? Phần tiếp theo sẽ giải đáp thắc mắc này và giới thiệu một số ứng dụng thực tế.

6 Ứng dụng của phân cụm trong thực tế

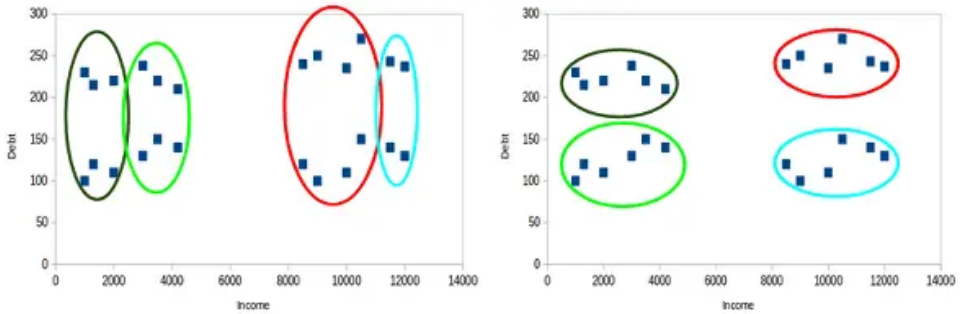
Phân cụm được sử dụng rộng rãi trong nhiều lĩnh vực, từ ngân hàng, hệ thống đề xuất, đến phân cụm văn bản và phân đoạn ảnh.

- **Phân đoạn khách hàng:** Đây là ứng dụng phổ biến nhất của phân cụm, không chỉ trong ngân hàng mà còn trong viễn thông, thương mại điện tử, thể thao, quảng cáo, bán hàng,...
- **Phân cụm văn bản:** Nếu bạn có nhiều tài liệu và muốn nhóm các tài liệu tương tự lại với nhau, phân cụm giúp gom các tài liệu giống nhau vào cùng một nhóm.
- **Phân đoạn ảnh:** Phân cụm cũng dùng để phân đoạn ảnh bằng cách nhóm các điểm ảnh giống nhau vào cùng một cụm.
- **Hệ thống đề xuất:** Ví dụ, muốn đề xuất bài hát cho bạn bè, bạn có thể xem những bài họ thích rồi dùng phân cụm để tìm những bài tương tự, từ đó đưa ra đề xuất phù hợp.

Ngoài ra còn rất nhiều ứng dụng khác mà bạn có thể chia sẻ thêm.

7 Đánh giá chất lượng cụm

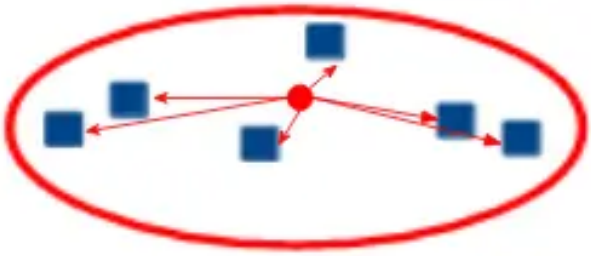
Mục tiêu của phân cụm không chỉ là tạo cụm mà còn phải tạo các cụm có ý nghĩa và chất lượng tốt. Ví dụ dưới đây minh họa:



Trong ví dụ này chỉ có 2 đặc trưng nên dễ dàng trực quan và đánh giá cụm.

Tuy nhiên, trong thực tế, ta có thể có rất nhiều đặc trưng (thu nhập, nghề nghiệp, tuổi, giới tính,...). Việc trực quan hóa và đánh giá cụm trở nên khó khăn. Lúc này, ta cần các chỉ số đánh giá chất lượng cụm.

- **Inertia (độ quán tính):** Đánh giá khoảng cách giữa các điểm trong cụm với centroid cụm. Thường dùng khoảng cách Euclid cho dữ liệu số. Inertia là tổng các khoảng cách này trong tất cả cụm. Giá trị inertia càng nhỏ thì cụm càng tốt (các điểm càng gần nhau).



Intra cluster distance

- **Dunn Index:** Ngoài khoảng cách nội cụm, Dunn index còn xem xét khoảng cách giữa các cụm (khoảng cách giữa các centroid). Đây là tỉ lệ giữa khoảng cách nhỏ nhất giữa hai cụm với khoảng cách lớn nhất trong một cụm. Giá trị Dunn index càng cao càng tốt, thể hiện các cụm vừa tách biệt rõ ràng vừa chặt chẽ.

$$\text{Dunn Index} = \frac{\min(\text{Inter cluster distance})}{\max(\text{Intra cluster distance})}$$

Intra cluster distance Inter cluster distance

- **Silhouette Score (điểm silhouette):** Đo mức độ tương đồng của một điểm với cụm của nó so với các cụm khác. Điểm silhouette cao thể hiện cụm rõ ràng, phân tách tốt. Điểm gần 0 hoặc âm cho thấy cụm có thể chồng lấn hoặc phân cụm kém.

8 Ví dụ Thực tế Chi tiết

8.1 Bước 1: Khởi tạo Centroids

Với dữ liệu khách hàng trên và $k = 3$, ta khởi tạo ngẫu nhiên 3 centroids:

$$C_0 = (22, 85) \tag{4}$$
$$C_1 = (30, 50) \tag{5}$$
$$C_2 = (34, 64) \tag{6}$$

8.2 Bước 2: Gán nhãn cho từng điểm

Tính khoảng cách từ điểm đầu tiên (18, 80) đến các centroids:

$$d(P_1, C_0) = \sqrt{(18-22)^2 + (80-85)^2} = \sqrt{16+25} = 6.4 \tag{7}$$

$$d(P_1, C_1) = \sqrt{(18-30)^2 + (80-50)^2} = \sqrt{144+900} = 32.31 \tag{8}$$

$$d(P_1, C_2) = \sqrt{(18-34)^2 + (80-64)^2} = \sqrt{256+256} = 22.62 \tag{9}$$

Vì $d(P_1, C_0)$ nhỏ nhất, điểm P_1 được gán cho cụm 0.
Tiếp tục tương tự cho tất cả điểm, ta được:

Index	Tuổi	Chi tiêu	D to C0	D to C1	D to C2
1	18	80	6.4	32.31	22.62
2	20	90	5.39	42.43	30.41
3	22	85	0.0	35.36	24.17
4	30	50	36.4	0.0	14.56
5	34	64	25.0	14.56	0.0
6	40	60	29.15	14.14	7.21

Bảng 2: Khoảng cách từ các điểm đến centroids ban đầu

8.3 Bước 3: Cập nhật Centroids

Sau khi gán nhãn, ta tính centroids mới:
Cụm 0: Chứa điểm (18,80), (20,90), (22,85)

$$C_0^{new} = \left(\frac{18+20+22}{3}, \frac{80+90+85}{3}\right) = (20, 85)$$

Cụm 1: Chứa điểm (60,30), (66,40), (70,25), (30,50)

$$C_1^{new} = \left(\frac{60+66+70+30}{4}, \frac{30+40+25+50}{4}\right) = (56.5, 36.25)$$

Cụm 2: Chứa điểm (34,64), (40,60)

$$C_2^{new} = \left(\frac{34+40}{2}, \frac{64+60}{2}\right) = (37, 62)$$

8.4 Iteration tiếp theo

Lặp lại quá trình assignment và update cho đến khi centroids không thay đổi hoặc thay đổi rất nhỏ.
Kết quả cuối cùng:

- **Nhóm Trẻ (Cụm 0):** (18,80), (20,90), (22,85)
- **Nhóm Trung niên (Cụm 2):** (30,50), (34,64), (40,60)
- **Nhóm Cao tuổi (Cụm 1):** (60,30), (66,40), (70,25)

9 Triển khai với NumPy

9.1 Implementation cơ bản

```python
import numpy as np
import matplotlib.pyplot as plt

class KMeans:
    def __init__(self, k=3, max_iters=100):
        self.k = k
        self.max_iters = max_iters

    def fit(self, X):
        # Initialize centroids randomly
        self.centroids = X[np.random.choice(X.shape[0], self.k, replace=False)]

        for i in range(self.max_iters):
            # Assignment step
            distances = np.linalg.norm(X[:, np.newaxis, :] - self.centroids, axis=2)
            labels = np.argmin(distances, axis=1)

            # Update step
            new_centroids = np.array([X[labels == j].mean(axis=0) for j in range(self.k)])

            # Check for convergence
            if np.all(self.centroids == new_centroids):
                break

            self.centroids = new_centroids

        return labels

    def predict(self, X):
        distances = np.linalg.norm(X[:, np.newaxis, :] - self.centroids, axis=2)
        return np.argmin(distances, axis=1)

# Usage example
X = np.array([[18, 80], [20, 90], [22, 85], [30, 50],
              [34, 64], [40, 60], [60, 30], [66, 40], [70, 25]])

kmeans = KMeans(k=3)
labels = kmeans.fit(X)
print("Final centroids:", kmeans.centroids)
print("Labels:", labels)
```

9.2 Tính toán WCSS (Within-Cluster Sum of Squares)

```python
def calculate_wcss(X, labels, centroids):
    wcss = 0
    for i in range(len(centroids)):
        cluster_points = X[labels == i]
        if len(cluster_points) > 0:
            wcss += np.sum(np.square(cluster_points - centroids[i]))
    return wcss

# Calculate WCSS for the results
wcss = calculate_wcss(X, labels, kmeans.centroids)
print(f"WCSS: {wcss}")
```

3. Ứng dụng nâng cao

1 Xử lý Dữ liệu Hình ảnh

K-Means có thể được áp dụng để phân cụm hình ảnh bằng cách:

1. **Phẳng hóa hình ảnh:** Chuyển từ (224, 224, 3) thành (224×224×3,)
2. **Áp dụng K-Means:** Phân cụm các pixel hoặc các đặc trưng
3. **Feature Extraction:** Sử dụng "Magic Function"để giảm chiều từ (224×224×3,) xuống (512,)

```python
def image_clustering(image_path, k=2):
    from PIL import Image

    # Read and preprocess the image
    img = Image.open(image_path)
    img_array = np.array(img)

    # Flatten into 2D array
    pixel_data = img_array.reshape(-1, 3)  # RGB channels

    # Apply K-Means
    kmeans = KMeans(k=k)
    labels = kmeans.fit(pixel_data)

    # Reconstruct image with centroids
    clustered_img = kmeans.centroids[labels]
    clustered_img = clustered_img.reshape(img_array.shape)

    return clustered_img.astype(np.uint8)
```

2 Xử lý Dữ liệu Văn bản

Đối với dữ liệu văn bản, ta sử dụng **Bag of Words (BoW)**:

```python
from sklearn.feature_extraction.text import CountVectorizer

# Sample documents
documents = [
    "gop gio gat bao",
    "co lam moi co an",
    "dat lanh chim dau",
    "an chao da bat",
    "gay ong dap lung ong",
    "qua cau rut van"
]

# Create Bag of Words vectors
vectorizer = CountVectorizer()
bow_matrix = vectorizer.fit_transform(documents)
vocabulary = vectorizer.get_feature_names_out()

print("Vocabulary:", vocabulary)
print("BoW matrix shape:", bow_matrix.shape)

# Apply K-Means clustering
```

```
kmeans_text = KMeans(k=2)
labels = kmeans_text.fit(bow_matrix.toarray())
print("Text clustering labels:", labels)
```

4. Các kỹ thuật nâng cao

1 Lựa chọn K tối ưu - Elbow Method

```
def elbow_method(X, max_k=10):
    wcss_values = []
    k_range = range(1, max_k + 1)

    for k in k_range:
        kmeans = KMeans(k=k)
        labels = kmeans.fit(X)
        wcss = calculate_wcss(X, labels, kmeans.centroids)
        wcss_values.append(wcss)

    # Plot the Elbow graph
    plt.figure(figsize=(10, 6))
    plt.plot(k_range, wcss_values, 'bo-')
    plt.xlabel('Number of clusters (k)')
    plt.ylabel('WCSS')
    plt.title('Elbow Method to select optimal k')
    plt.grid(True)
    plt.show()

    return wcss_values

# Use the Elbow Method
wcss_values = elbow_method(X, max_k=8)
```

2 K-Means++ Initialization

Cải thiện việc khởi tạo centroids để tránh local minima:

```
def kmeans_plus_plus_init(X, k):
    centroids = []

    # Randomly select the first centroid
    centroids.append(X[np.random.randint(X.shape[0])])

    for _ in range(1, k):
        # Calculate distance to the nearest centroid for each point
        distances = np.array([min([np.linalg.norm(x - c)**2 for c in centroids]) for x
    in X])

        # Select the next point with probability proportional to its distance
        probabilities = distances / distances.sum()
        cumulative_probabilities = probabilities.cumsum()
        r = np.random.rand()

        for j, p in enumerate(cumulative_probabilities):
            if r < p:
                centroids.append(X[j])
                break

```

```
    return np.array(centroids)

# Using K-Means++
class KMeansPlusPlus(KMeans):
    def fit(self, X):
        # Use K-Means++ initialization
        self.centroids = kmeans_plus_plus_init(X, self.k)

        # Continue as in regular K-Means
        for i in range(self.max_iters):
            distances = np.linalg.norm(X[:, np.newaxis, :] - self.centroids, axis=2)
            labels = np.argmin(distances, axis=1)

            new_centroids = np.array([X[labels == j].mean(axis=0) for j in range(self.k)])

            if np.all(np.abs(self.centroids - new_centroids) < 1e-4):
                break

            self.centroids = new_centroids

        return labels
```

Phần II: Triển khai Thuật toán K-Means từ đầu trong Python

Bây giờ là lúc khởi động Jupyter notebooks (hoặc IDE nào bạn sử dụng) và bắt tay vào làm việc với Python!

Chúng ta sẽ làm việc với bộ dữ liệu dự đoán vay vốn mà bạn có thể tải xuống. Tôi khuyến khích bạn đọc thêm về bộ dữ liệu và bài toán dự đoán vay vốn. Điều này sẽ giúp bạn hình dung những gì chúng ta đang làm (và tại sao chúng ta làm điều này). Đây là hai câu hỏi khá quan trọng trong bất kỳ dự án khoa học dữ liệu nào.

1 Bước 1: Import các thư viện cần thiết

Đầu tiên, import tất cả các thư viện cần thiết:

```
import pandas as pd
import numpy as np
import random as rd
import matplotlib.pyplot as plt
```

Listing 1: Import các thư viện

2 Bước 2: Đọc và khám phá dữ liệu

Bây giờ, chúng ta sẽ đọc file CSV và xem năm hàng đầu tiên của dữ liệu:

```
data = pd.read_csv('clustering.csv')
data.head()
```

Listing 2: Đọc dữ liệu

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term
0	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0
1	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0
2	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0
3	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0
4	LP001013	Male	Yes	0	Not Graduate	No	2333	1516.0	95.0	360.0

Hình 5: Đọc file

Đối với bài viết này, chúng ta sẽ chỉ lấy hai biến từ dữ liệu - "LoanAmount"và "ApplicantIncome". Điều này sẽ giúp dễ dàng hình dung các bước cũng như thực hiện. Hãy chọn hai biến này và hình dung các điểm dữ liệu:

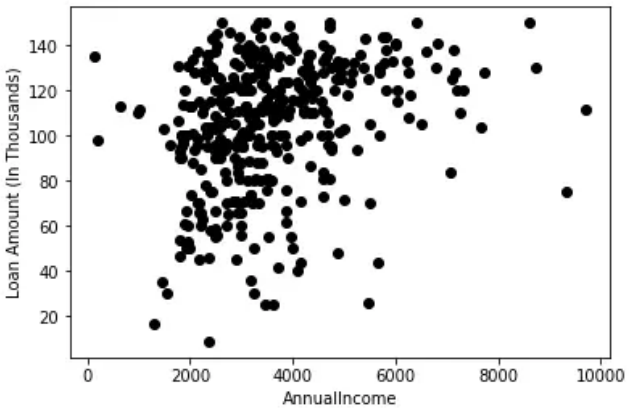
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

data = pd.read_csv('clustering.csv')
data.head()

X = data[["LoanAmount", "ApplicantIncome"]]
# Visualize the data points
plt.scatter(X["ApplicantIncome"], X["LoanAmount"], c='black')
plt.xlabel('Annual Income')
```

```
plt.ylabel('Loan Amount (Thousands)')
plt.show()
```

Listing 3: Chọn biến và trực quan hóa dữ liệu



3 Bước 3: Chọn số cụm và khởi tạo centroids

Bước 1 và 2 của K-Means là về việc chọn số lượng cụm (k) và chọn các centroids ngẫu nhiên cho mỗi cụm. Chúng ta sẽ chọn 3 cụm và sau đó chọn các quan sát ngẫu nhiên từ dữ liệu làm centroids:

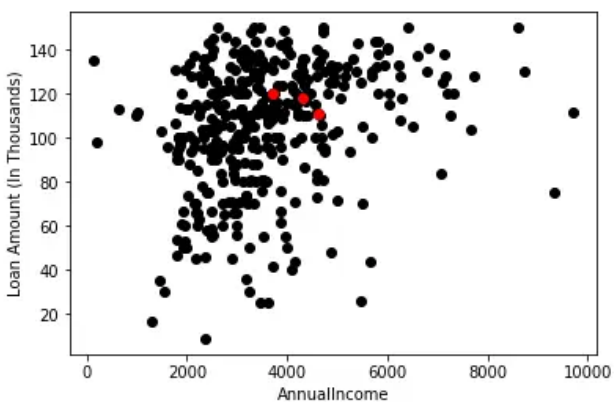
```
# Step 1 and 2 - Choose the number of clusters (k) and randomly select centroids for
    each cluster

# number of clusters
K = 3

# Randomly select observations as centroids
Centroids = X.sample(n=K)
plt.scatter(X["ApplicantIncome"], X["LoanAmount"], c='black')
plt.scatter(Centroids["ApplicantIncome"], Centroids["LoanAmount"], c='red')
plt.xlabel('Annual Income')
plt.ylabel('Loan Amount (Thousands)')
```

Listing 4: Chọn số cụm và khởi tạo centroids

Ở đây, các chấm đỏ đại diện cho 3 centroids của mỗi cụm. Lưu ý rằng chúng ta đã chọn những điểm này một cách ngẫu nhiên, và do đó mỗi lần bạn chạy mã này, bạn có thể nhận được các centroids khác nhau.



4 Bước 4: Triển khai thuật toán K-Means chính

Tiếp theo, chúng ta sẽ định nghĩa một số điều kiện để triển khai thuật toán K-Means Clustering. Hãy xem mã trước:

```
# Step 3 - Assign all points to the nearest cluster centroid
# Step 4 - Recalculate the centroids of the newly formed clusters
# Step 5 - Repeat steps 3 and 4

diff = 1
j = 0

while diff != 0:
    XD = X.copy()
    i = 1
    for index1, row_c in Centroids.iterrows():
        ED = []
        for index2, row_d in XD.iterrows():
            d1 = (row_c["ApplicantIncome"] - row_d["ApplicantIncome"])**2
            d2 = (row_c["LoanAmount"] - row_d["LoanAmount"])**2
            d = np.sqrt(d1 + d2)
            ED.append(d)
        X[i] = ED
        i += 1

    C = []
    for index, row in X.iterrows():
        min_dist = row[1]
        pos = 1
        for i in range(K):
            if row[i + 1] < min_dist:
                min_dist = row[i + 1]
                pos = i + 1
        C.append(pos)
    X["Cluster"] = C
    Centroids_new = X.groupby(["Cluster"]).mean()[["LoanAmount", "ApplicantIncome"]]

    if j == 0:
        diff = 1
        j += 1
    else:
        diff = (Centroids_new['LoanAmount'] - Centroids['LoanAmount']).sum() + (
    Centroids_new['ApplicantIncome'] - Centroids['ApplicantIncome']).sum()
        print(diff.sum())

    Centroids = Centroids_new
```

Listing 5: Triển khai thuật toán K-Means

Những giá trị này có thể thay đổi mỗi lần chúng ta chạy. Ở đây, chúng ta dừng quá trình huấn luyện khi các centroids không thay đổi sau hai lần lặp. Đây là tiêu chí hội tụ phổ biến nhất được sử dụng cho thuật toán K-Means clustering. Chúng ta đã định nghĩa diff ban đầu là 1, và bên trong toàn bộ vòng lặp, chúng ta tính toán diff này như là sự khác biệt giữa các centroids ở lần lặp trước và lần lặp hiện tại.

Khi sự khác biệt này bằng 0, chúng ta dừng quá trình huấn luyện.

5 Bước 5: Trực quan hóa kết quả

Bây giờ hãy trực quan hóa các cụm chúng ta đã có được:

```
color = ['blue', 'green', 'cyan']
for k in range(K):
    data = X[X["Cluster"] == k + 1]
    plt.scatter(data["ApplicantIncome"], data["LoanAmount"], c=color[k])
plt.scatter(Centroids["ApplicantIncome"], Centroids["LoanAmount"], c='red')
plt.xlabel('Income')
plt.ylabel('Loan Amount (Thousands)')
plt.show()
```

Listing 6: Trực quan hóa kết quả phân cụm

Cuối cùng thì chúng ta có thể thấy rõ ràng ba cụm. Các chấm đỏ đại diện cho centroid của mỗi cụm. Tôi hy vọng bây giờ bạn đã hiểu rõ cách thức hoạt động của K-Means.