Tìm hiểu về thuật toán K-Means

Nguyễn Văn Huy & Lê Duy An

Ngày 28 tháng 7 năm 2020

Mục lục

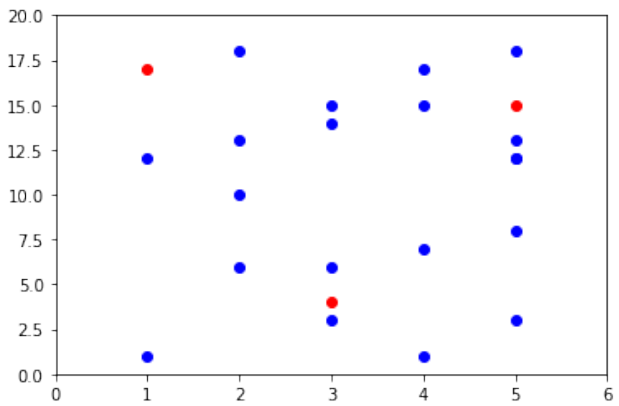
1 Giới thiệu

2 Phân tích toán học

Với dữ liệu đầu vào của thuật toán là tập hợp các điểm dữ liệu $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_N] \in \mathbb{R}^{d \times N}$ với $\mathbf{x}_i$ (có d phần tử) là một vector mang giá trị của mỗi điểm, N là số lượng các vector và số lượng K các nhóm cần phân loại từ các điểm dữ liệu đó với $K < N$ (vì số lượng nhóm cần phân loại không được lớn hơn số lượng các phần tử). Điều mà chúng ta cần phải làm là làm thế nào để xác định các điểm thuộc về nhóm nào một cách gắn kết nhất, ở đây để cho dễ gọi và tính toán thì chúng ta cho rằng K nhóm cần phần loại được gọi là nhóm $1, 2, 3, ..K$. Trong phần này chúng ta chỉ đề cập đến bài toán chỉ có một điểm dữ liệu thuộc vào một nhóm duy nhất.
Ban đầu chúng ta phải có được các điểm gốc ban đầu của các nhóm có thể chọn k điểm bất kì hoặc có thể lấy các điểm dữ liệu có sẵn trong tập dữ liệu ban đầu. Gọi các điểm gốc ban đầu là $\mathbf{m} = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3, ..., \mathbf{m}_K]$ với mỗi điểm $\mathbf{m}_k$ cũng có có d các giá trị tương tự như các điểm dữ liệu $\mathbf{x}_i$. Dựa vào tập các điểm gốc $\mathbf{m}_k$ chúng ta phải xác định xem điểm $\mathbf{x}_i$ thuộc vào nhóm nào và gán nhãn cho các điểm đó bằng vector $\mathbf{y}$ có dạng:

$$\mathbf{y}_{ij} \in \{0, 1\}, \ \forall i, j; \ \sum_{j=1}^{K} = 1, \ \forall i$$

Trong đó $\mathbf{y}_{ij} = 0$ và $\mathbf{y}_{ik} = 1$, nghĩa là vector $\mathbf{y}$ có K giá trị và vị trí ở vị trí k có giá trị bằng 1 thì đồng nghĩa là vector $\mathbf{x}_i$ được gán vào nhóm k. Tập hợp các nhãn là $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, ..., \mathbf{y}_N] \in \mathbb{R}^{K \times N}$.
Một ví dụ để hình dung rõ hơn về các tập dữ liệu. Ví dụ đề cập đến việc phân nhóm các điểm của tập $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_N]$ được biểu diễn trong Bảng 2 nằm trong không gian 2𝔻 gồm có số lượng các điểm là $N = 20$ được biểu diễn trong Hình 2.1 bằng các điểm màu xanh, vì trong không gian 2𝔻 nên $d = 2$ tương ứng là tọa độ x và y. Điều kiện của bài toán là các điểm đó thành ba cụm với ba điểm ban đầu là $\mathbf{m} = [(1; 17), (3; 4), (5; 15)]$ có $K = 3$ được biểu thị bằng các điểm màu đỏ trên Hình 2.1.



Hình 2.1: Hình các các dữ liệu đầu vào.

$\mathbf{X}$	x	y
$\mathbf{x}_1$	1	4
$\mathbf{x}_2$	5	12
$\mathbf{x}_3$	2	18
...	...	...
$\mathbf{x}_N$	5	13

Bảng 1: Bảng biểu diễn một vài giá trị của các điểm trong tập $\mathbf{X}$

$\mathbf{Y}$	1	2	3
$\mathbf{y}_1$	0	0	1
$\mathbf{y}_2$	1	0	0
$\mathbf{y}_3$	0	1	0
...	...	...	...
$\mathbf{y}_N$	0	0	1

Bảng 2: Bảng biểu diễn mội vài nhãn $\mathbf{y}_i$ trong tập $\mathbf{Y}$

2.1 Hàm mất mát và bài toán tối ưu

2.2 Thuật toán tối ưu hàm mất mát

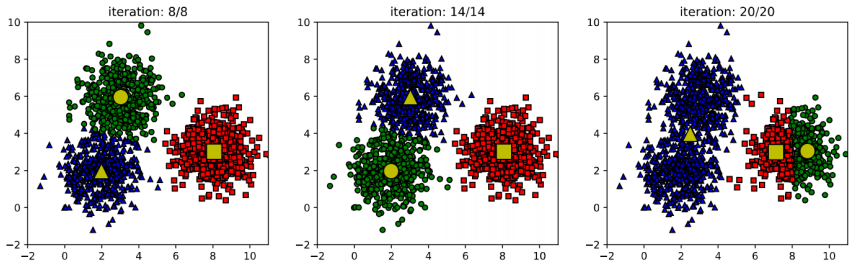
3 Ưu và nhược điểm.

3.1 Ưu điểm

- Thuật toán đơn giản, hiệu quả với độ phức tạo là $O(tKn)$, t, k « n, với:
  - t: số lần lặp
  - K: số cluster
  - n: số mẫu
- Sử dụng được với bộ số liệu lớn

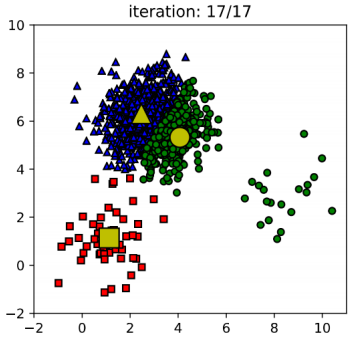
3.2 Nhược điểm

- Cần phải xác định trước số lượng cluster. Trong thực tế, cần phải sử dụng thêm một số biện pháp giúp xác định giá trị K, chẳng hạn như phương pháp Elbow.
- Thuật toán KMeans không đảm bảo tìm được nghiệm tối ưu toàn cục nên nghiệm cuối cùng phụ thuộc rất nhiều vào các centroid ban đầu. Hình 3.1 thể hiện các kết quả khác nhau khi các centroid khởi tạo khác nhau. Ngoài ra, việc chọn các centroid cũng ảnh hưởng đến hiệu suất làm việc của thuật toán. Với cùng một kết quả tối ưu như nhau, số lần chạy ở hình 2 lớn gần gấp đôi so với hình 1.



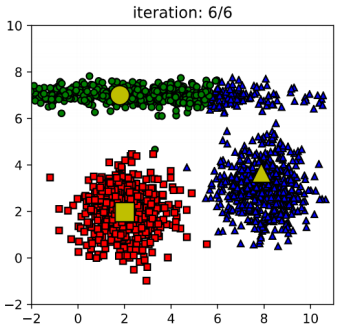
Hình 3.1: Các nghiệm khác nhau do khởi tạo ban đầu khác nhau

- Các cluster cần phải có số lượng điểm gần bằng nhau. Ở hình 3.2 minh hoạ kết quả thuật toán KMeans với bộ dữ liệu có các cluster có số điểm chênh lệch. Trong trường hợp này, nhiều điểm đáng lẽ thuộc vào cụm xanh lam đã bị nhầm vào cụm xanh lục.



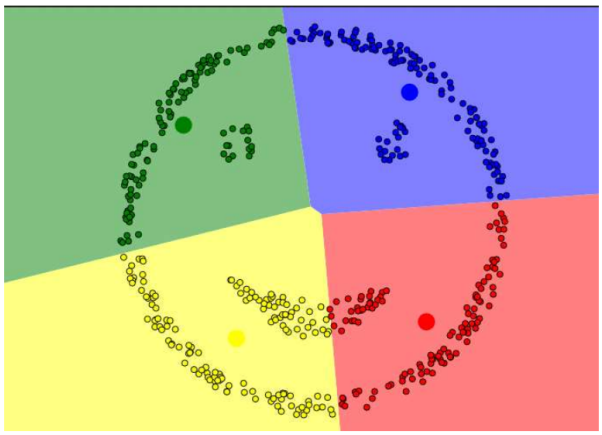
Hình 3.2: Các nghiệm trong cluster này bị nhầm vào cluster khác

- Các cluster cần có dạng hình tròn (cầu), nếu không thì KMeans sẽ hoạt động không hiệu quả. Lý do chính là vì Kmeans quyết định cluster của một điểm dữ liệu thông qua khoảng cách của nó đến centroid.



Hình 3.3: Các nghiệm trong cluster này bị nhầm vào cluster khác

- Centroid có thể bị xê dịch bởi các ngoại lệ, hoặc các ngoại lệ có thể có cụm riêng thay vì bị bỏ qua.
- Cho kết quả sai khi một cluster này bị bao bọc bằng một cluster khác. Hình 3.3 là một minh hoạ điển hình cho việc KMeans không thể phân cụm dữ liệu. Một cách tự nhiên, chúng ta chia hình mặt thành 4 cụm: mắt trái, mắt phải, miệng, hình bao quanh mặt. Nhưng vì các bộ phận mắt, miệng nằm bên trong mặt nên KMeans phân cụm không chính xác.



Hình 3.4: KMeans phân cụm hình mặt không chính xác

4 Cách tìm K cụm tối ưu nhất

Xác định số lượng cụm tối ưu trong một tập dữ liệu là vấn đề cơ bản trong phân cụm Kmeans, yêu cầu người dùng chỉ định số lượng cụm k được tạo. Ý tưởng đằng sau Kmeans bao gồm xác định các cụm k sao cho tổng biến thể trong cụm là tối thiểu. Đây được xem là một nhược điểm của thuật toán này. Phần dưới đây trình bày một vài phương pháp giúp xác định số cụm k hợp lý nhất.

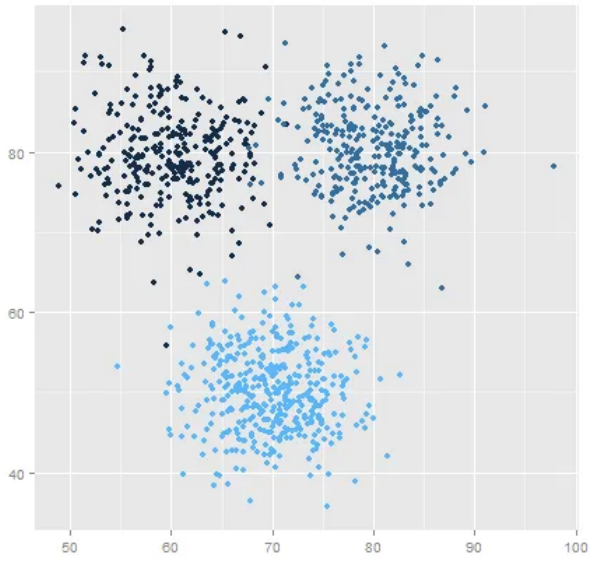
4.1 Thuật toán Elbow

Tư tưởng chính của phương pháp phân cụm phân hoạch (như KMeans) là định nghĩa 1 cụm sao cho tổng bình phương khoảng cách của tất cả các điểm đến đến trung tâm cụm là nhỏ nhất, tham số này là WSS (Within-cluster Sum of Square). Elbow method chọn số cụm k sao cho khi thêm vào một cụm khác thì không làm cho WSS thay đổi nhiều.

Quy trình triển khai thuật toán Elbow:

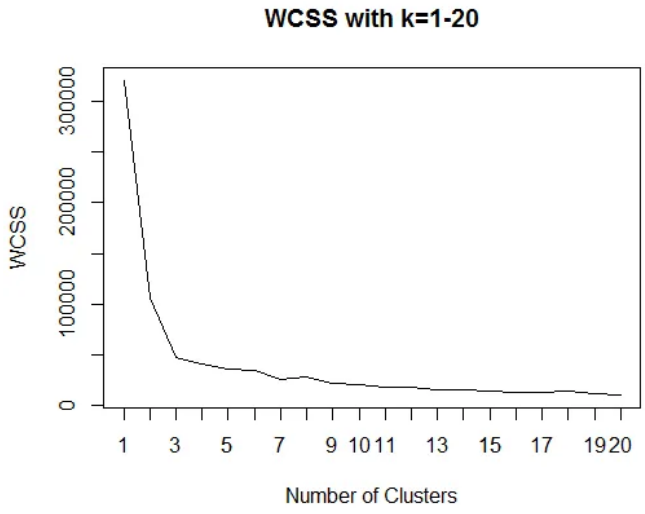
- Thực hiện phân cụm với số cụm thay đổi
- Với mỗi giá trị k, tính giá trị WSS
- Vẽ đường cong Elbow theo các giá trị k
- Dựa vào đường cong Elbow chọn số k thích hợp, là vị trí ở khúc cua

Xét ví dụ mẫu dữ liệu sau:



Hình 4.1: Mẫu dữ liệu

Tính toán giá trị WSS tương ứng với K từ 1 đến 20, ta thu được biểu đồ sau:



Hình 4.2: WSS tương ứng với k từ 1 đến 20

Ta chọn k = 3 làm số cụm cho mẫu dữ liệu trên.

4.2 Thuật toán Average Silhouette

Average silhouette dùng để đo chất lượng của một cụm. Giá trị Silhouette s(i) cho mỗi điểm dữ liệu i được xác định như sau:

$$s(i) = \frac{b(i)-a(i)}{max\{a(i),b(i)\}}(|C_i| > 1)$$

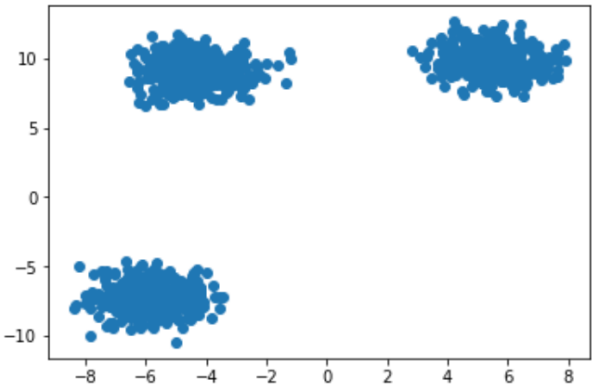
Trong đó:

- a(i) khoảng cách trung bình từ điểm i đến các điểm khác trong cụm. $a(i) = \frac{1}{|C_i|-1} \sum_{j \in C_i, i \neq j} d(i,j)$
- b(i) là khoảng cách trung bình giữa các cụm. $b(i) = \min_{i \neq j} \frac{1}{|C_j|} \sum_{j \in C_j} d(i,j)$

Với s(i) = 0 với những cụm có chỉ có 1 phần tử.
Quy trình triển khai thuật toán average silhouette:

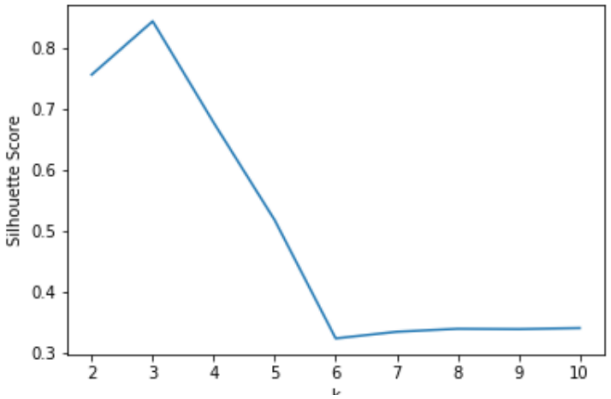
- Thực hiện phân cụm với số cụm thay đổi
- Với mỗi giá trị k, tính giá trị average silhouette
- Vẽ đường cong average silhouette theo các giá trị k
- Vị trí có average silhouette lớn nhất là số cụm cần tìm

Xét ví dụ về mẫu dữ liệu sau:



Hình 4.3: Mẫu dữ liệu

Vẽ đường cong average silhouette:



Ta chọn k = 3 làm số cụm cho mẫu dữ liệu trên.