BÀI TẬP 3: K-MEANS TRÊN SPARK

LƯU Ý: BÀI TOÁN CẦN ĐƯỢC XÂY DỰNG CỤ THỂ TRÊN SPARK, KHÔNG ĐƯỢC DÙNG THƯ VIỆN SPARK MLlib CÓ SẴN CỦA SPARK.

Bài tập này giúp các bạn hiểu rõ chi tiết cài đặt của thuật toán phân cụm (clustering algorithms) trên Spark, đồng thời cũng giúp bạn hiểu rõ tầm quan trọng của các độ đo khoảng cách và chiến lược khởi tạo ban đầu.

Cho tập *X* có *n* điểm dữ liệu trong không gian $\mathbb{R}^d$, số cụm (cluster) cho trước k và k điểm trung tam C. Bây giờ ta định nghĩa một vài độ đo khoảng cách và hàm lỗi tương ứng cần cực tiểu hoá.

Euclidean distance: Cho hai điểm A và B trong không gian d chiều, có thể viết A=[a_1,a_2,…,a_d] và B=[b_1,b_2,…,b_d] khoảng cách Euclid giữa A và B được định nghĩa:

$$\|A - B\| = \sqrt{\sum_{i=1}^{d} (a_i - b_i)^2} \quad (1)$$

Hàm lỗi (hay còn gọi là hàm chi phí hay hàm mất mát) Ø tương ứng cần cực tiểu hoá khi gián điểm vào cluster sử dụng khoảng cách Euclid là:

$$\emptyset = \sum_{x \in X} \min_{c \in C} \|x - c\|^2 \quad (2)$$

Lưu ý khoảng cách trong hàm lỗi được bình phương để đảm bảo có thể tối thiểu hoá.

Manhattan distance: Cho hai điểm A và B trong không gian d chiều, có thể viết A=[a_1,a_2,…,a_d] và B=[b_1,b_2,…,b_d] khoảng cách Manhattan giữa A và B được định nghĩa:

$$|A - B| = \sum_{i=1}^{d} |a_i - b_i| \quad (3)$$

Hàm lỗi ψ tương ứng cần cực tiểu hoá khi gián điểm vào cluster sử dụng khoảng cách Manhattan là:

$$\psi = \sum_{x \in X} \min_{c \in C} |x - c| \quad (4)$$

Lặp lại trong thuật toán k-means: Chúng ta đã học thuật toán k-means với các bước sau: Khởi tạo k điểm trung tâm của k cluster; với mỗi điểm trong X ta gán vào cluster có điểm trung tâm gần nó nhất; tính lại điểm trung tâm sau khi đã gán hết các điểm vào cluster. Trong thực tế các bước trên được chạy trong một vài lần lặp thể hiện trong thuật toán bên dưới.

Lặp lại của thuật toán k-mean trên Spark: Cài đặt thuật toán k-mean với số lần lặp tối đa cụ thể trên Spark sử dụng dữ liệu trong thư mục **q2/data**, thư mục gồm 3 file:

1. File data.txt chứa tập dữ liệu với 4601 dòng và 58 cột. Mỗi dòng là một biểu diễn của một tài liệu dùng vector 58 chiều. Mỗi chiều trong vector thể hiện độ quan trong của một từ trong một tài liệu. Bạn có thể tải file này trực tiếp trên Google Colab bằng cách dùng ID 1E-voIV2ctU4Brw022Na8RHVVRGOoNkO1 (Xem code tải file trong file Colab 0 đã cung cấp ở bài tập 1)
2. File c1.txt chứa k điểm khởi tạo trung tâm của k cluster. Các điểm khởi tạo này được chọn ngẫu nhiên với k=10. Bạn có thể tải file này trực tiếp trên Google Colab bằng cách dùng ID 1yXNlZWMqUcAwDScBrkFChOHJwR1FZXmI (Xem code tải file trong file Colab 0 đã cung cấp ở bài tập 1)
3. File c2.txt chứa các điểm được chọn xa nhau nhất có thể dùng khoảng cách Euclid dùng để khởi tạo trung tâm của các cluster. (Bạn có thể tự chọn các điểm này bằng cách chọn điểm c1 đầu tiên ngẫu nhiên sau đó chọn điểm c2 xa với c1, sau đó chọn điểm c3 xa với c1 và c2, …). Bạn có thể tải file này trực tiếp trên Google Colab bằng cách dùng ID 1vfovle9DgaeK0LnbQTH0j7kRaJjsvLtb (Xem code tải file trong file Colab 0 đã cung cấp ở bài tập 1)

Thiết lập số lần lặc tối đa (MAX_ITER) là 20, số cluster là k=10 cho các thử nghiệm trong bài tập này. Bạn cần đảm bảo chương trình lặp đúng số lần thực tế chứ không phải lúc nào cũng 20 lần (tối đa là 20).

I. Thử nghiệm chiến lược khởi tạo dùng khoảng cách Euclid
   a. Sử dụng khoảng cách Euclid (Xem công thức 1) để làm độ đo khoảng cách, tính hàm lỗi Ø(i) (xem công thức 2) cho mỗi lần lặp i. Tức là trong lần lặp đầu tiên bạn sẽ tính hàm lỗi sử dụng điểm khởi trung tâm ở một trong hai file c1 hoặc c2. Chạy k-means trên data.txt sử dụng c1.txt và c2.txt. Vẽ đồ thị thể hiện hàm lỗi trong các lần lặp từ 1 đến MAX_ITER cho c1.txt và c2.txt. Bạn có thể vẽ trong cùng một đồ thị để dễ dàng trả lời câu hỏi theo yêu cầu.

      (*Gợi ý: Không cần viết Spark job độc lập để tính Ø(i), bạn có thể tính giá trị hàm lỗi khi phân bổ điểm vào cluster*)

   b. Hàm lỗi thay đổi bao nhiêu phần trăm sau 10 lần lặp của k-means khi khởi tạo điểm trung tâm sử dụng c1.txt so với c2.txt và sử dụng khoảng cách Euclid. Khởi tạo điểm trung tâm ngẫu nhiên dùng c1.txt có tốt hơn khởi tạo dùng c2.txt theo giá trị hàm lỗi Ø(i) không? Giải thích lý do.

      (*Gợi ý: để tính phần trăm thay đổi giữa lần lặp 1 và 10 ta dùng công thức (cost[0]-cost(10))/cost(0).*)

II. Thử nghiệm chiến lược khởi tạo dùng khoảng cách Manhattan
   a. Sử dụng khoảng cách Manhattan (xem công thức 3) làm độ đo khoảng cách, tính hàm mất lỗi $\psi(i)$ (xem công thức 4) cho mỗi lần lặp thứ i. Tức là ở lần lặp đầu tiên bạn cần tính hàm lỗi sử dụng điểm khởi tạo trung tâm ở một trong hai file trên. Chạy k-means trên data.txt sử dụng c1.txt và c2.txt. Vẽ đồ thị thể hiện hàm lỗi trong các lần lặp từ 1 đến MAX_ITER cho c1.txt và c2.txt. Bạn có thể vẽ trong cùng một đồ thị để dễ dàng trả lời câu hỏi theo yêu cầu.

      (*Gợi ý: Có thể giải quyết vấn đề tương tự như mục I. Giá trị hàm lỗi không phải luôn giảm, k-means chỉ đảm bảo đơn điệu giảm cho khoảng cách Euclid*)

b. Hàm lỗi thay đổi bao nhiêu phần trăm sau 10 lần lặp của k-means khi khởi tạo điểm trung tâm sử dụng c1.txt so với c2.txt và sử dụng khoảng cách Euclid. Khởi tạo điểm trung tâm ngẫu nhiên dùng c1.txt có tốt hơn khởi tạo dùng c2.txt theo giá trị hàm lỗi $\psi(i)$ không? Giải thích lý do.

Yêu cầu nộp:

1. Hai file code dạng colab cho I.a, II.a
2. File word thể hiện hình ảnh đồ thị cho I.a, II.a; kết quả tính phần trăm thay đổi của hàm lỗi và giải thích cho I.b, II.b