MỘT SỐ THUẬT TOÁN XỬ LÝ DỮ LIỆU LỚN

Giảng viên: Nguyễn Tu Trung, Trần Mạnh Tuấn
BM HTTT, Khoa CNTT, Trường ĐH Thủy Lợi

Hà Nội, 2019

Nội dung

- Thuật toán K-Means
- Thuật toán MapReduce_K-Means
- Thuật toán Naïve Bayes
- Thuật toán MapReduce_Bayes

Thuật toán K-Means

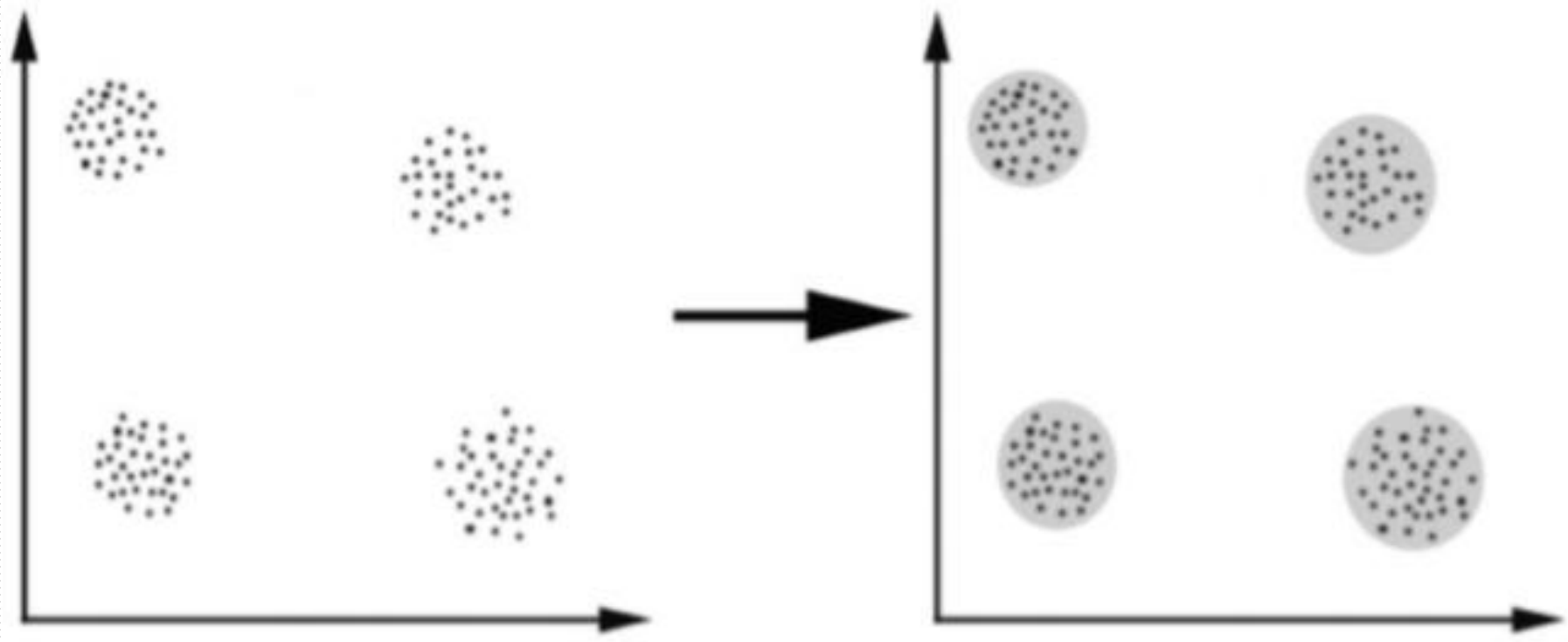
- Giới thiệu thuật toán K-Means
- Phát biểu bài toán phân cụm
- Các bước thuật toán K-Means
- Điều kiện dừng và chất lượng phân cụm
- Nhận xét thuật toán K-Means

Giới thiệu thuật toán K-Means

- Là một trong các thuật toán phân cụm đơn giản và điển hình nhất
- Do MacQueen đề xuất trong lĩnh vực thống kê năm 1967
- Mục đích:
  - Sinh ra k cụm dữ liệu từ một tập dữ liệu ban đầu gồm n đối tượng trong không gian p chiều $X_i = \{x_{i1}, x_{i2}, \ldots, x_{ip}\}$, i = 1..n, sao cho hàm tiêu chuẩn E đạt giá trị tối thiểu
    - $E = \sum_{i=1}^{k} \sum_{x \in C_i} d(x, m_i)^2$
  - m_i là vector trọng tâm của cụm C_i, giá trị của mỗi phần tử là trung bình cộng các thành phần tương ứng của các đối tượng vector dữ liệu trong cụm đang xét
  - d là khoảng cách Euclide giữa hai đối tượng

Phát biểu bài toán phân cụm

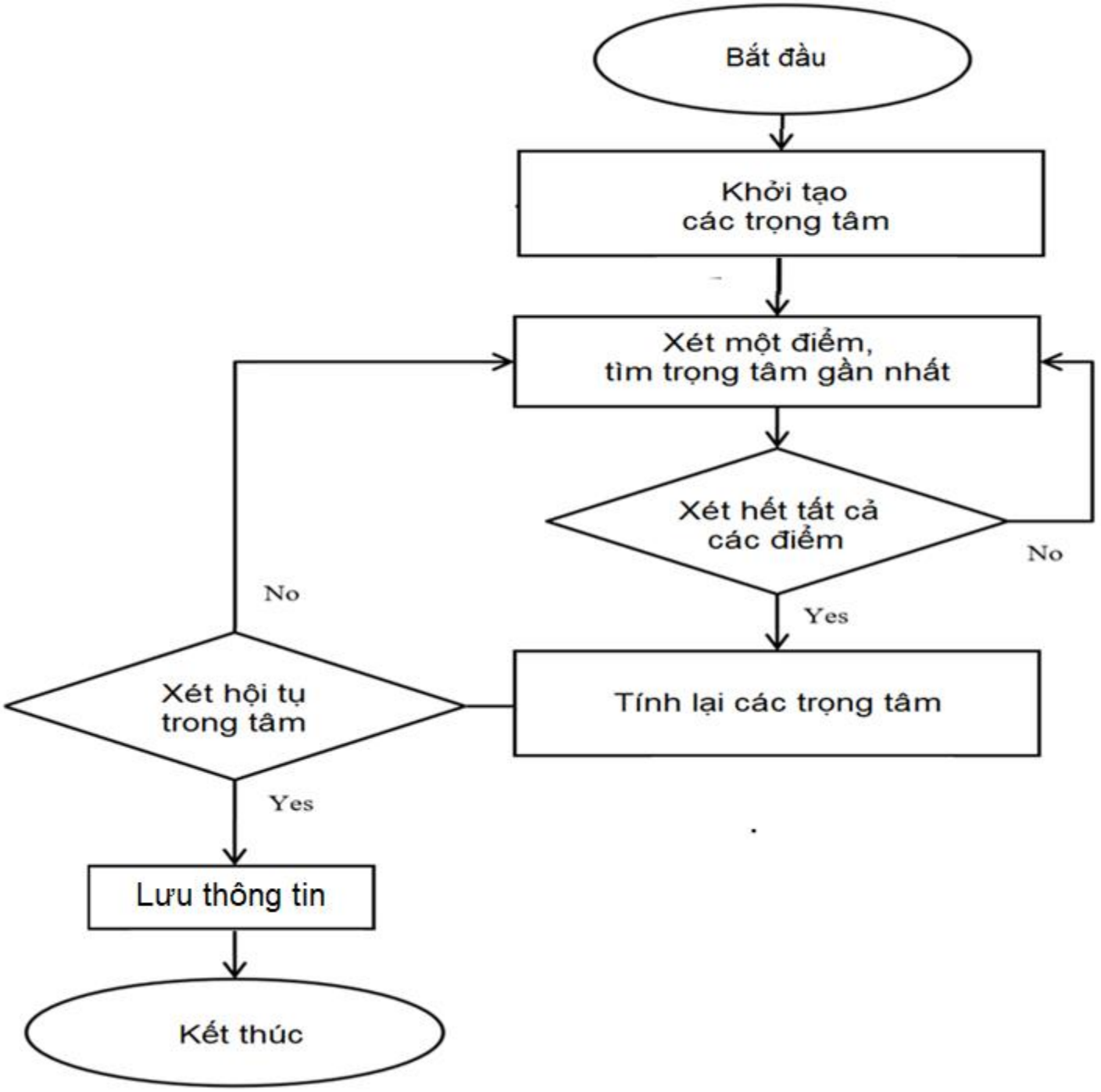
- Input: n đối tượng và số các cụm k
- Output: Các cụm C_i (i=1... k) sao cho hàm tiêu chuẩn E đạt giá trị tối thiểu



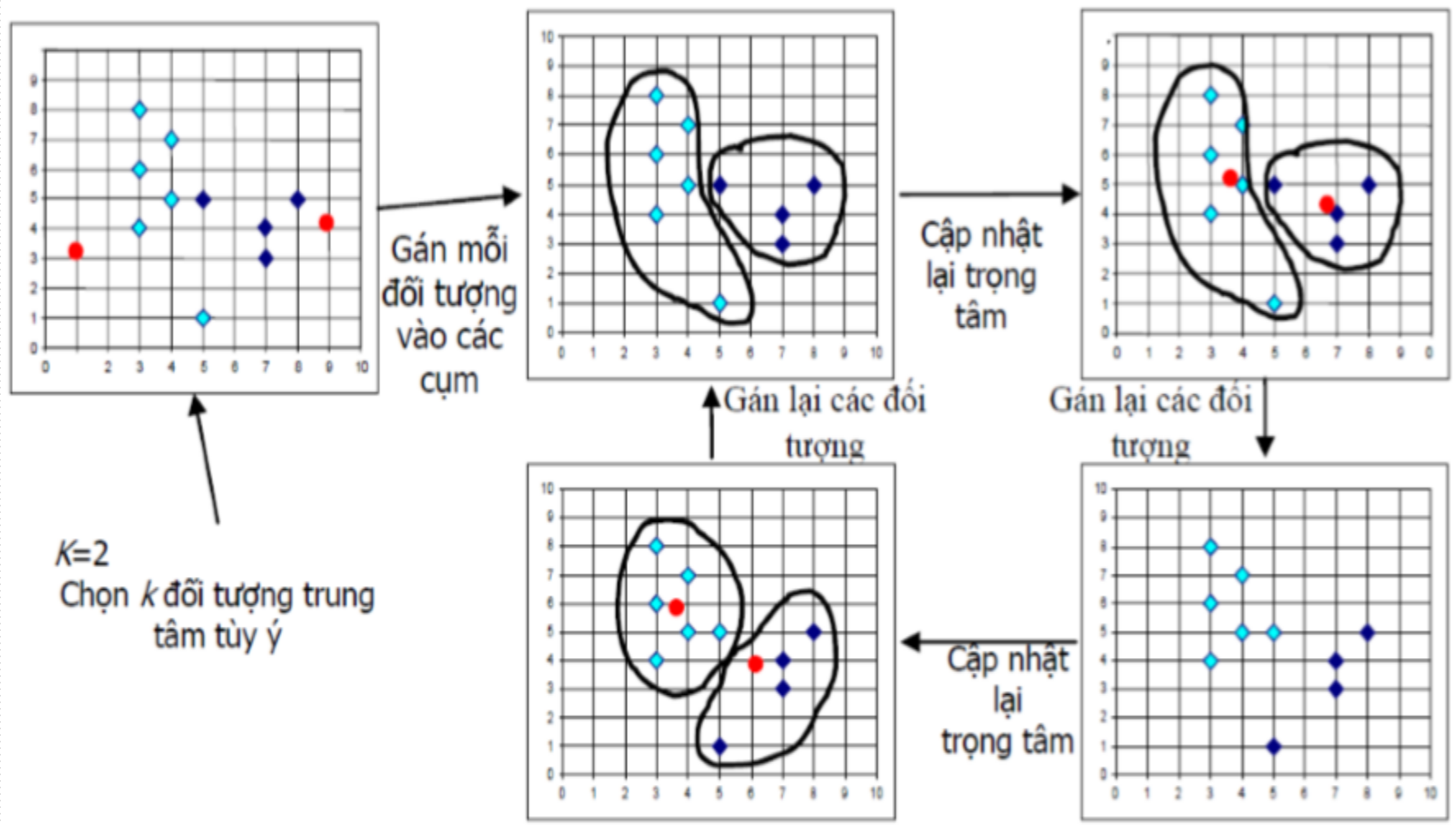
Các bước thuật toán K-Means

- Bước 1: Khởi tạo tâm cụm
  - Chọn k đối tượng m_j (j=1...k) là trọng tâm ban đầu của k cụm từ tập dữ liệu (Việc lựa chọn này có thể là ngẫu nhiên hoặc theo kinh nghiệm)
- Bước 2: Tính toán khoảng cách và gán cụm
  - Với mỗi đối tượng Xi (1 i n), tính toán khoảng cách từ nó tới mỗi trọng tâm mj với j=1,..,k, sau đó tìm trọng tâm gần nhất đối với mỗi đối tượng
- Bước 3: Cập nhật lại trọng tâm
  - Với mỗi j=1,..,k, cập nhật trọng tâm cụm mj bằng cách xác định trung bình cộng của các vector đối tượng dữ liệu
- Bước 4: Kiểm tra điều kiện dừng
  - Lặp các bước 2 và 3 cho đến khi các trọng tâm của cụm không thay đổi

Lưu đồ thuật toán K-Means



Minh họa quá trình phân cụm



Điều kiện dừng và chất lượng phân cụm

- Điều kiện dừng
  - Không có (hoặc có không đáng kể) việc gán lại các ví dụ vào các cụm khác
  - Không có (hoặc có không đáng kể) thay đổi về các điểm trung tâm (centroids) của các cụm
  - Giảm không đáng kể về tổng lỗi phân cụm E
- Chất lượng phân cụm
  - Phụ thuộc nhiều vào các tham số đầu vào như: số cụm k và k trọng tâm khởi tạo ban đầu
  - Nếu các trọng tâm khởi tạo ban đầu quá lệch so với các trọng tâm cụm tự nhiên thì kết quả phân cụm của k-means là rất thấp => các cụm dữ liệu được khám phá rất lệch so với các cụm trong thực tế

Nhận xét thuật toán K-Means

- Phần lớn khối lượng tính toán tập trung ở bước tính khoảng cách từ mỗi điểm (đối tượng) tới các tâm cụm
- Số lượng đối tượng trong tập dữ liệu càng lớn, thời gian cần cho bước này càng nhiều
- Việc tính toán khoảng cách từ một điểm tới tâm cụm là độc lập, không phụ thuộc vào điểm khác
- => Việc tính khoảng cách từ các điểm có thể thực hiện song song, đồng thời với nhau

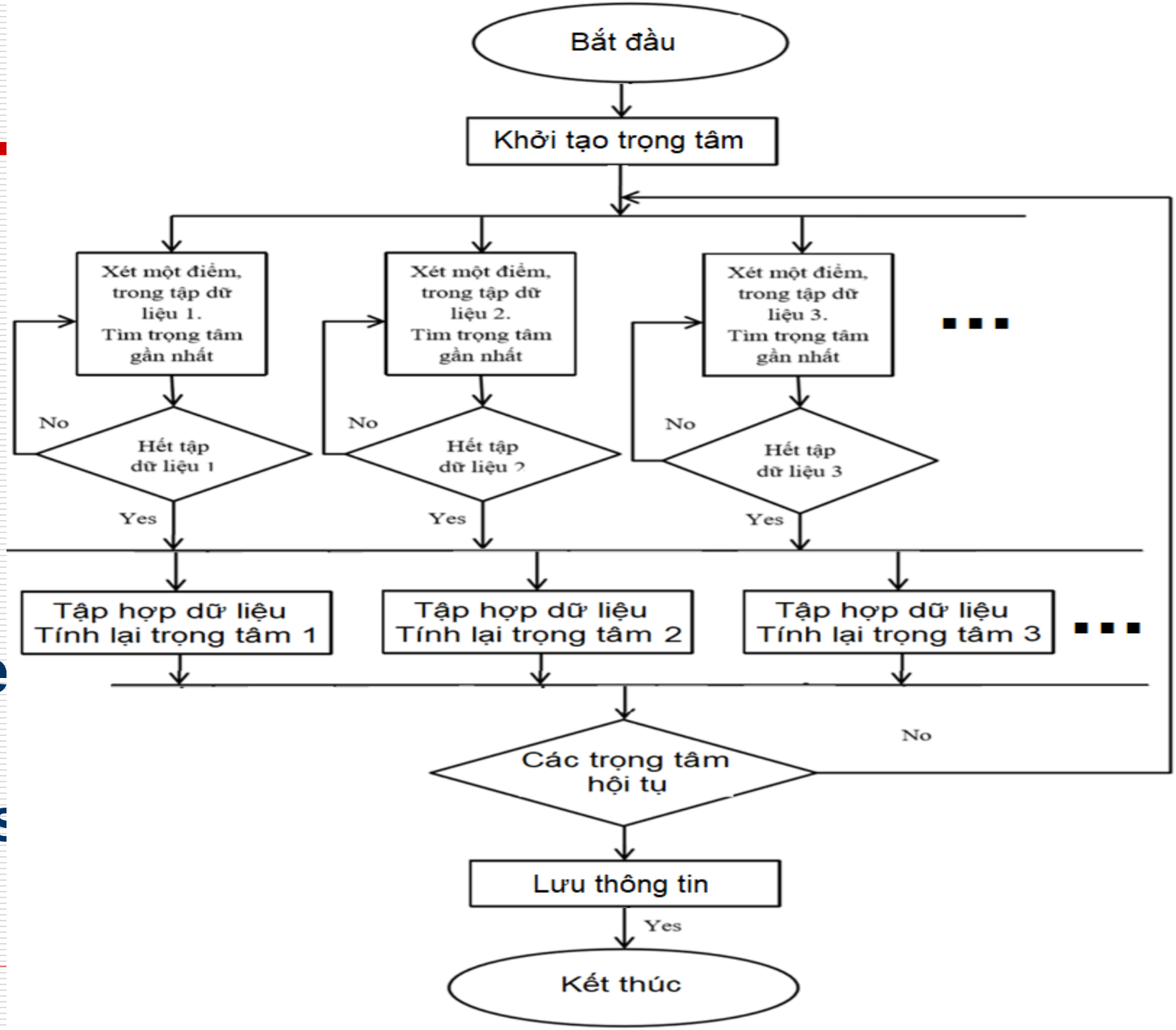
Thuật toán MapReduce_K-Means

- Ý tưởng MapReduce hóa K-Means
- Lưu đồ thuật toán MapReduce_K-Means
- Giải pháp MapReduce hóa K-Means
- Xây dựng hàm Map_K-Means
- Xây dựng hàm Reduce_K-Means
- Mã nguồn MapReduce_K-Means

Ý tưởng MapReduce hóa K-Means

- Với mỗi vòng lặp
  - Tách dữ liệu vào các nhóm nhỏ
  - Map:
    - Phân cụm trên từng nhóm nhỏ dữ liệu
    - Gom dữ liệu theo từng tâm
  - Tất cả dữ liệu được gom theo từng tâm
  - Reduce:
    - Tính tâm mới của các dữ liệu được gom (theo từng tâm)

Lưu đồ thuật toán MapReduce_K-Means



Giải pháp MapReduce hóa K-Means

- Đầu tiên: biểu diễn dữ liệu
  - Dữ liệu lưu trữ dưới dạng list các hàng
  - Mỗi hàng là list giá trị là các thành phần của vector biểu diễn cho một điểm
- Thứ hai: lưu trữ phân tán dữ liệu
  - Do các điểm được tính toán độc lập với nhau => có thể lưu trữ các phần của dữ liệu trên nhiều máy khác nhau để tăng tốc tính toán
- Thứ ba, trên mỗi máy tính, trong mỗi vòng lặp, mỗi máy
  - B1: Tính khoảng cách của mỗi điểm trong phần dữ liệu của nó với các trọng tâm
  - B2: Kiểm tra xem điểm đó gần trọng tâm nào nhất
  - B3: Gửi lại kết quả cho để gộp các điểm thuộc cùng một nhóm để tính lại trọng tâm sau mỗi vòng lặp

Giải pháp MapReduce hóa K-Means

- Dữ liệu cần phân cụm là danh sách các hàng (có thể lưu trên file txt) được chuyển sang kiểu key/value làm đầu vào cho thuật toán
- Mô hình cơ bản của MapReduce:
  - map (keyIn, valIn) -> list (keyInt, valInt)
  - reduce (keyInt, list (valInt)) -> list (keyOut, valOut)
- Áp dụng cho K-Means:
  - Xây dựng hàm Map_K-Means
  - Xây dựng hàm Reduce_K-Means

Xây dựng hàm Map_K-Means

- Đầu vào: cặp key/value biểu diễn toạ độ của một điểm
  - keyIn là giá trị bye offset của dòng
  - valIn là vector biểu diễn toạ độ của một điểm
- Xử lý:
  - Tính khoảng cách của điểm với các trọng tâm (chưa phải là trọng tâm cần tìm)
  - Chuyển về cụm có tâm gần nhất
- Đầu ra: cặp key/value trung gian
  - keyInt là trọng tâm
  - valInt là toạ độ điểm thuộc cụm có trọng tâm là keyInt

Xây dựng hàm Reduce_K-Means

- Trước khi hàm reduce thực hiện
  - Kết quả của hàm map được trộn lại
  - Các cặp cùng keyInt được gom thành một nhóm
- Đầu vào:
  - keyInt được chuyển từ hàm map
  - list(valInt) là list các điểm valInt thuộc về cụm thứ keyInt
- Xử lý:
  - Tính trung bình cộng từng thành phần của các điểm cùng cụm
  - Cập nhật lại trọng tâm của cụm đó
- Đầu ra:
  - keyOut là keyInt
  - valOut là giá trị trọng tâm mới

Mã nguồn MapReduce_K-Means

- Thảo luận mã nguồn chương trình…

Thuật toán Naïve Bayes

- Giới thiệu thuật toán Naïve Bayes
- Định lý Bayes
- Thuật toán phân lớp Bayes
- Lưu đồ thuật toán phân lớp Bayes
- Ví dụ minh họa
- Thực thi thuật toán với ví dụ
- Phân tích thuật toán Bayes

Giới thiệu thuật toán Naïve Bayes

- Là phương pháp phân loại dựa vào xác suất
- Sử dụng rộng rãi trong lĩnh vực học máy, phổ biến trong nhiều lĩnh vực như các công cụ tìm kiếm, các bộ lọc mail nói riêng và phân loại văn bản nói chung
- Ý tưởng cơ bản của cách tiếp cận Naïve Bayes là sử dụng xác suất có điều kiện giữa từ và chủ đề để dự đoán xác suất chủ đề của một văn bản cần phân loại
- Điểm quan trọng của phương pháp này chính:
  - Giả định rằng sự xuất hiện của tất cả các từ trong văn bản đều độc lập với nhau
  - Không khai thác sự phụ thuộc của nhiều từ vào trong một chủ đề cụ thể
- Được xem là thuật toán đơn giản nhất trong các phương pháp

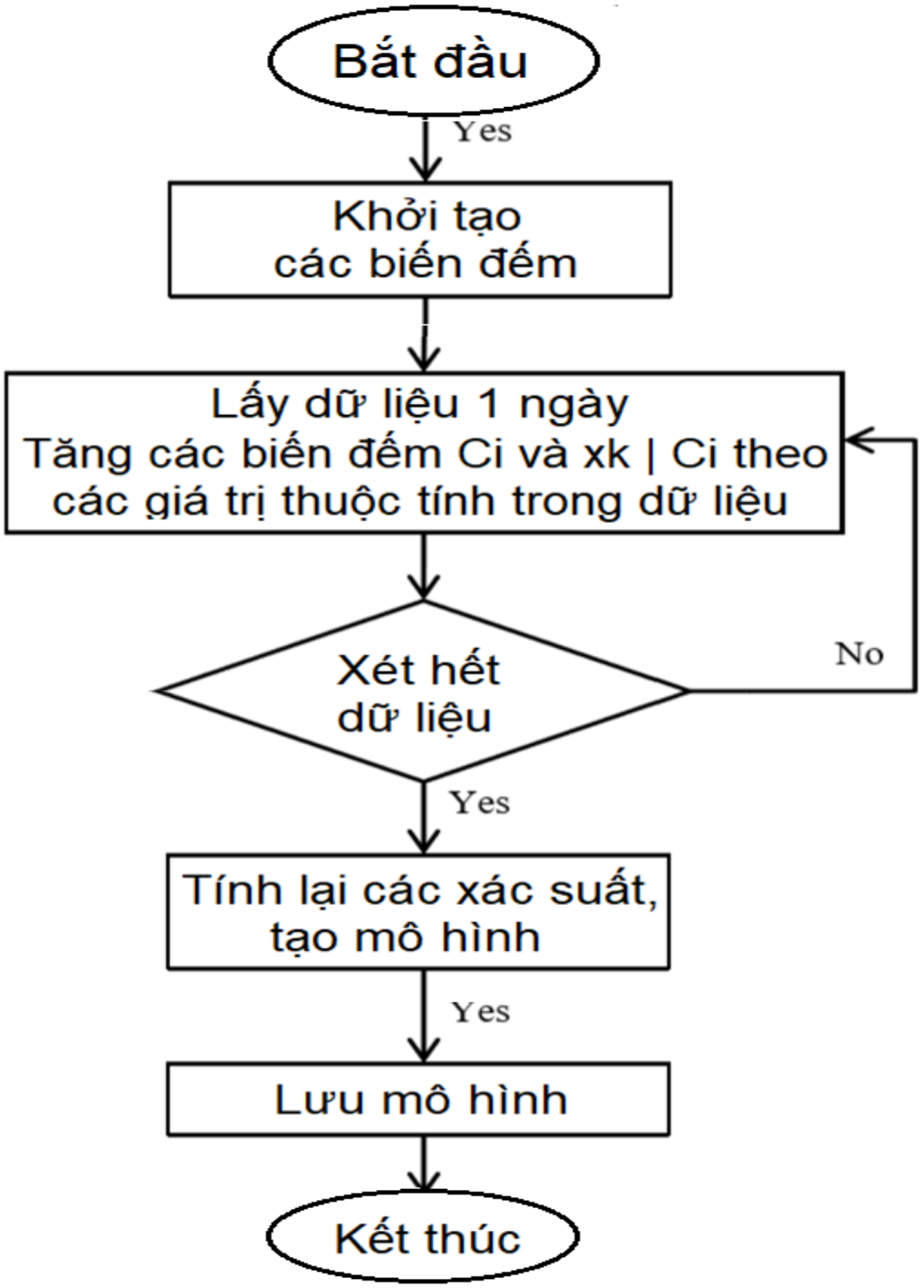
Định lý Bayes

- Định lý Bayes được phát biểu như sau:
  - $P(Y|X) = \frac{P(X|Y).P(Y)}{P(X)}$
- P(Y): Xác suất của sự kiện Y xảy ra
- P(X): Xác suất của sự kiện X xảy ra
- P(X|Y): Xác suất (có điều kiện) của sự kiện X xảy ra, nếu biết rằng sự kiện Y đã xảy ra
- P(Y|X): Xác suất (có điều kiện) của sự kiện Y xảy ra, nếu biết rằng sự kiện X đã xảy ra

Thuật toán phân lớp Bayes

- Dữ kiện cần có:
  - D: tập dữ liệu huấn luyện, được vector hoá dưới dạng $\vec{x} = (x_1, x_2, \dots, x_n)$
  - C_i : tập các tài liệu của D thuộc lớp C_i với i={1,2,3,…}
  - Các thuộc tính $x_1, x_2, \dots, x_n$ độc lập xác suất đôi một với nhau
- Thuật toán Naïve Bayes cơ bản:
  - Bước 1 : Huấn luyện Naïve Bayes (dựa vào tập dữ liệu)
    - Tính xác suất $P(C_i)$
    - Tính xác suất $P(x_k|C_i)$
  - Bước 2: Phân lớp X_{new}
    - Tính $F(X_{new}, C_i) = P(C_i) \prod_{k=1}^{n} P(x_k|C_i)$
    - X_{new} được gán vào lớp C_q sao cho
      - $F(X_{new}, C_q) = \max(F(X_{new}, C_i))$

Lưu đồ thuật toán phân lớp Bayes



Ví dụ minh họa thuật toán Bayes

- Yêu cầu: Dự đoán quyết định của người chơi có đi chơi Tennis hay không với các điều kiện về thời tiết đã được biết trước
- Bảng dữ liệu huấn luyện như sau:

Day	Outlook	Temp.	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No

Ví dụ minh họa

- Bảng dữ liệu huấn luyện như sau (tiếp):

Day	Outlook	Temp.	Humidity	Wind	Play Tennis
D7	Overcast	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Thực thi thuật toán Bayes với ví dụ

- Có 2 lớp dự báo:
  - C1 = “yes” => Có đi chơi Tennis
  - C2 = “no” => Không đi chơi Tennis
- B1: Huấn luyện Bayes
- B2: Phân lớp Bayes

B1: Huấn luyện Bayes

- Tính các xác suất $P(C_i)$
  - P(C1) = P("yes") = 9/14
  - P(C2) = P("no") = 5/14
- Tính các xác suất $P(x_k|C_i)$
  - Với thuộc tính Outlook
  - Với thuộc tính Temp
  - Với thuộc tính Humidity
  - Với thuộc tính Wind

Với thuộc tính Outlook

- Có các giá trị: sunny, overcast, rain
- P(sunny | yes) = 2/9
- P(sunny | no) = 3/5
- P(overcast | yes) = 4/9
- P(overcast | no) = 0/5
- P(rain | yes) = 3/9
- P(rain | no) = 2/5

Với thuộc tính Temp

- Có các giá trị: Hot, Cold, Mild
- P(hot | yes) = 2/9
- P(hot | no) = 2/5
- P(cold | yes) = 3/9
- P(cold | no) = 1/5
- P(mild | yes) = 4/9
- P(mild | no) = 2/5

Với thuộc tính Humidity

- Có các giá trị: Normal, High
- P(normal | yes) = 6/9
- P(normal | no) = 1/5
- P(high | yes) = 3/9
- P(high | no) = 4/5

Với thuộc tính Wind

- Có các giá trị: Weak, Strong
- P(weak | yes) = 6/9
- P(weak | no) = 2/5
- P(strong | yes) = 3/9
- P(strong | no) = 3/5

B2: Phân lớp Bayes

- X^{new} = {sunny, cool, high, strong}
- Tính các xác suất
  - $F(X^{new} | yes) = P(yes) * P(sunny | yes) * P(cool | yes) * P(high | yes) * P(strong | yes) = 9/14 * 2/9 * 3/9 * 3/9 * 3/9 = 0.0053$
  - $F(X^{new} | no) = P(no) * P(sunny | no) * P(cool | no) * P(high | no) * P(strong | no) = 5/14 * 3/5 * 1/5 * 4/5 * 3/5 = 0.0206$
- Kết luận: X^{new} thuộc vào lớp No

Phân tích thuật toán Bayes

- Tương tự như thuật toán k-means, Naïve bayes cũng có vấn đề khi xử lý lượng dữ liệu lớn
- Trong quá trình học (trainning), nếu số lượng dữ liệu quá lớn, dẫn đến các vấn đề về thiếu bộ nhớ, tốc độ xử lý
- Với lượng dữ liệu lớn (khoảng vài triệu bản ghi) thì hầu hết thời gian của Naïve bayes là đếm số lần xuất hiện của các biến => tính các xác suất cần thiết để xây dựng mô hình
- Tiêu thụ thời gian chủ yếu do tính: các $P(C_i)$ và $P(x_k|C_i)$
- Để tính các xác suất này, ta cần đếm số lần xuất hiện của các C_i và các $x_k|C_i$
- Việc tính các xác xuất, đếm các số lần xuất hiện của từng biến là độc lập => chia dữ liệu thành nhiều phần nhỏ và thực hiện song song

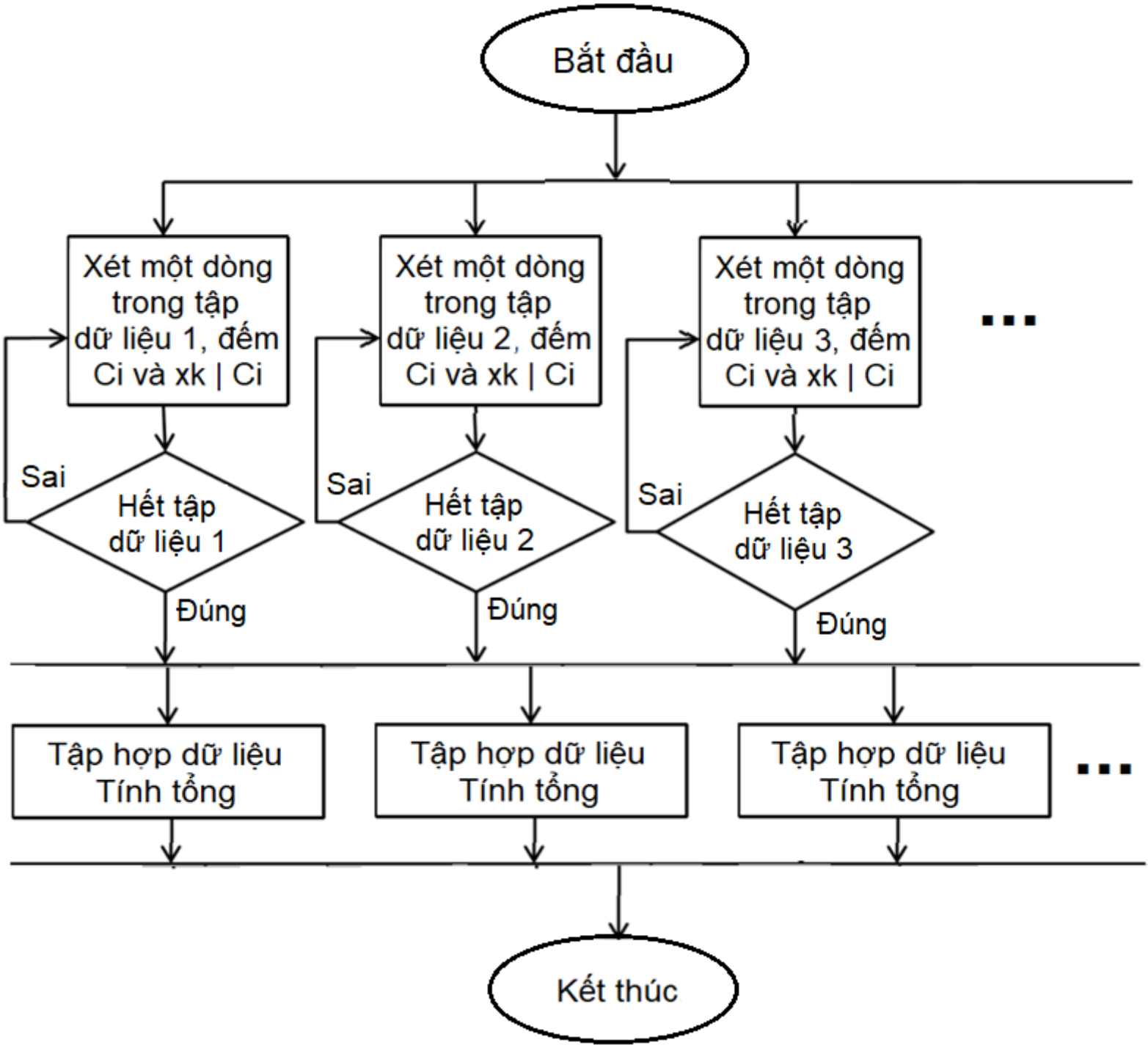
Thuật toán MapReduce_Bayes

- Ý tưởng MapReduce thuật toán Bayes
- Lưu đồ thuật toán MapReduce_Bayes
- Giải pháp MapReduce hoá thuật toán Bayes
- Xây dựng hàm Map_Bayes
- Xây dựng hàm Reduce_Bayes

Ý tưởng MapReduce hoá thuật toán Bayes

- Nhiệm vụ:
  - MapReduce hóa việc đếm số lần xuất hiện của các C_i và các $x_k|C_i$
- Ý tưởng:
  - Chia dữ liệu thành nhiều phần nhỏ
  - Đếm các số lần xuất hiện của từng biến C_i và $x_k|C_i$ trong hàm Map
  - Tập hợp kết quả và tính tổng theo từng biến trong hàm Reduce
  - Lưu thông tin số lần xuất hiện của từng biến C_i và $x_k|C_i$
  - Giai đoạn phân lớp: tính các xác suất $P(C_i)$ và $P(x_k|C_i)$ dựa trên dữ liệu về số lần xuất hiện của từng biến C_i và $x_k|C_i$ để tính các $F(X_{new}, C_i)$

Lưu đồ thuật toán MapReduce_Bayes



MapReduce hoá thuật toán Bayes

- Dữ liệu đầu vào:
  - Là danh sách các hàng (có thể lưu trên file txt)
  - Mỗi hàng là dữ liệu huấn luyện mô tả từng ngày: Sunny Hot High Weak No
  - Được chuyển sang kiểu key/value làm đầu vào cho thuật toán
- Mô hình cơ bản của MapReduce:
  - map (keyIn, valIn) -> list (keyInt, valInt)
  - reduce (keyInt, list (valInt)) -> list (keyOut, valOut)
- Áp dụng cho thuật toán Bayes:
  - Xây dựng hàm Map_Bayes
  - Xây dựng hàm Reduce_Bayes

Xây dựng hàm Map_Bayes

- Đầu vào:
  - cặp key/value biểu diễn dữ liệu một ngày
  - keyIn là giá trị bye offset của dòng
  - valIn là text biểu dữ liệu một ngày (Sunny Hot High Weak No)
- Xử lý:
  - Đếm 1 cho xuất hiện của C_i
  - Đếm 1 cho xuất hiện của $x_k|C_i$
- Đầu ra:
  - cặp key/value trung gian
  - keyInt là C_i hoặc $x_k|C_i$
  - valInt là giá trị 1

Xây dựng hàm Reduce_Bayes

- Trước khi hàm reduce thực hiện
  - Kết quả của hàm map được trộn lại
  - Các valInt cùng keyInt được gom thành một nhóm
- Đầu vào:
  - keyInt được chuyển từ hàm map (C_i hoặc $x_k|C_i$)
  - list(valInt) là list các giá trị 1
- Xử lý:
  - Tính tổng các giá trị 1 trong list(valInt)
- Đầu ra:
  - keyOut là keyInt (C_i hoặc $x_k|C_i$)
  - valOut là tổng các giá trị 1 trong list(valInt)