Lab 04: MapReduce hóa với các thuật toán học máy

Mục tiêu:

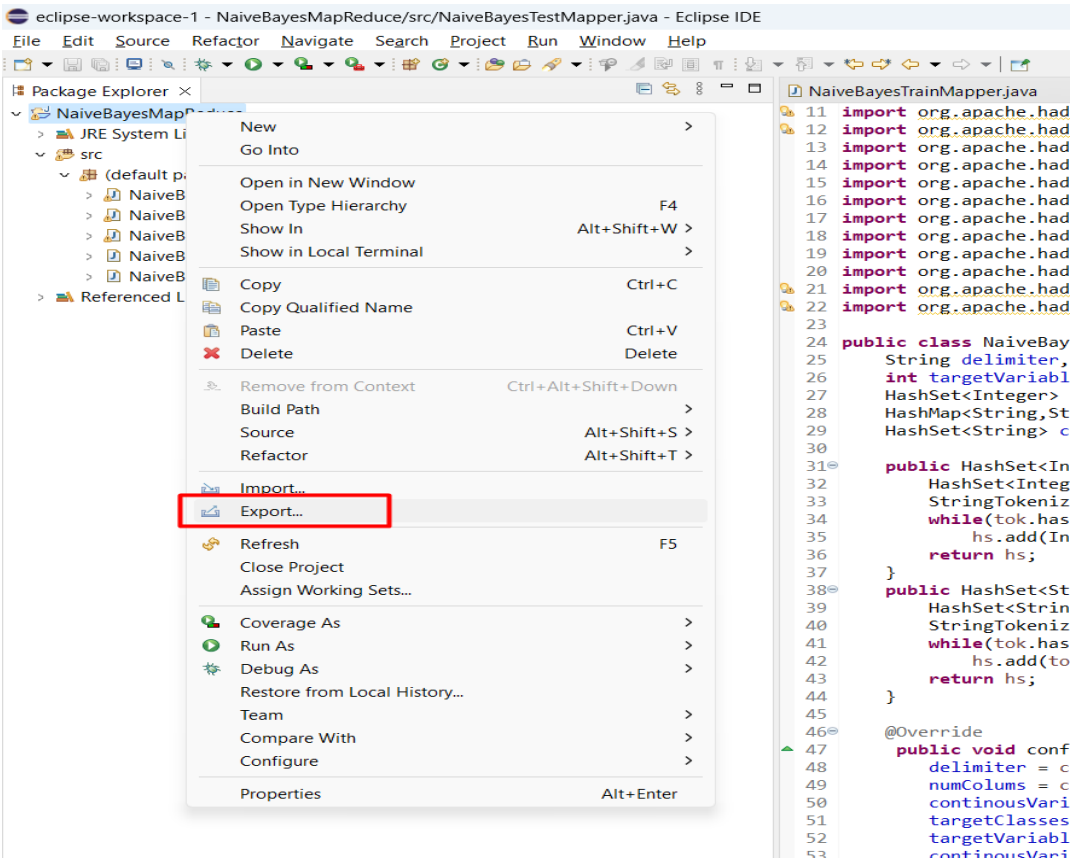
- Sử dụng MapReduce hóa với các thuật toán xử lý dữ liệu Naïve Bayes, K-Means
- Triển khai xây dựng cài đặt Hadoop MapReduce hóa với các thuật toán xử lý dữ liệu.
- Kiểm tra kết quả thực hiện trên màn hình giao diện ứng dụng, xem tiến trình của tác vụ trong bảng điều khiển và một URL để xem thông tin chi tiết hơn về tác vụ.
- Submit kết quả thực hiện Project lên hệ thống (LMS Canvas, Github, jupyter notebook…)

Nội dung:

Phần I: Kiểm tra đánh giá nội dung Lab03

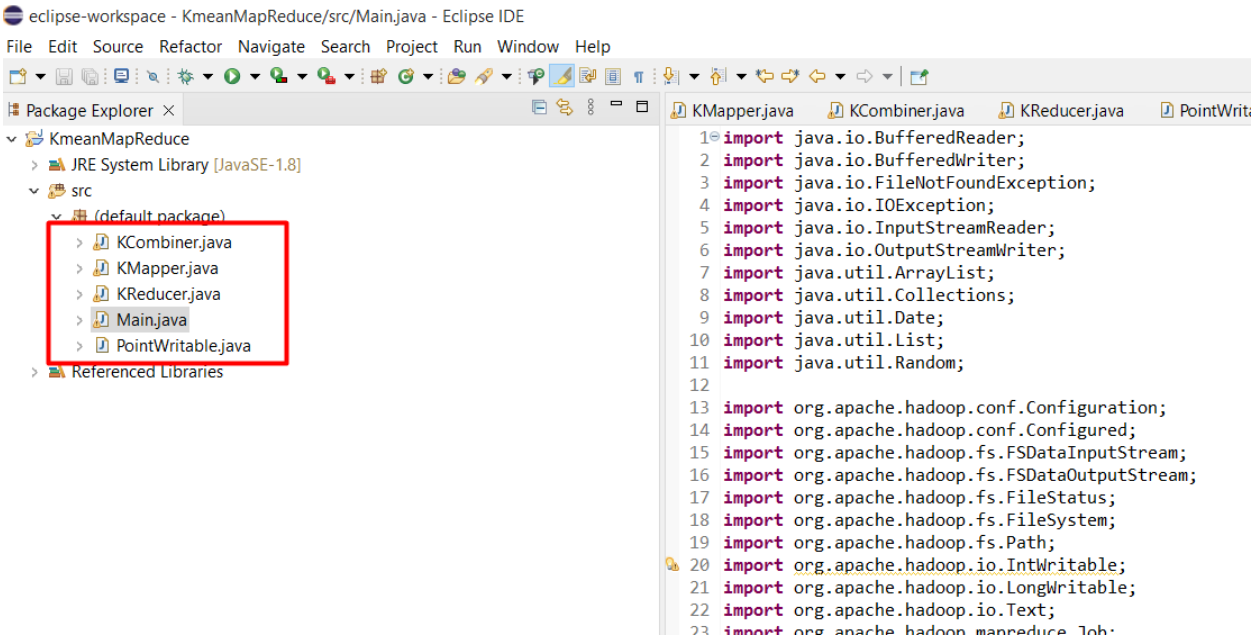
1. MapRedure hóa với thuật toán Naïve Bayes

- MapReduce - Phân loại nhị phân Naive Bayes
- Lập trình MapReduce cho bài toán phân loại hoa Iris sử dụng thuật toán Naive Bayes



Dữ liệu: https://www.kaggle.com/datasets/uciml/iris

2. MapRedure hóa với thuật toán K-Means



3. Bài tập về MapReduce hóa trên Hadoop

Classifier sử dụng MapReduce trên Hadoop Triển khai Naive Bayes được thực hiện bằng MapReduce và triển khai Local Machine, tập dữ liệu là DBPedia. Các bước thực hiện:

1. Naive_Bayes_classifier.ipynb Tập lệnh python trên Jupyter Notebook triển khai bộ phân loại Naive Bayes và tính toán thời gian tính toán và độ chính xác trên tập dữ liệu đào tạo, phát triển và thử nghiệm trên Local Machine.
2. Naive_Bayes_classifier.py Tập lệnh python triển khai bộ phân loại Naive Bayes và tính toán thời gian tính toán và độ chính xác trên tập dữ liệu đào tạo, phát triển và thử nghiệm trên Local Machine.
3. Python_mapreduce.py Tập lệnh python triển khai thuật toán Naive Bayes trên tập dữ liệu đào tạo, thử nghiệm và phát triển và ghi lại độ chính xác tương ứng cùng với thời gian đào tạo thuật toán. Từ điển được chuẩn bị trên nền tảng hadoop mapreduce.
4. mapper.py Tập lệnh python mapper được sử dụng để lập bản đồ bằng cách sử dụng hadoop stream để tạo đầu ra luồng (nhãn, từ, 1).
5. reducer.py Tập lệnh python reducer được sử dụng để lập bản đồ bằng hadoop stream để tạo luồng đầu ra (nhãn, từ, số lượng).
6. dictionary.pickle Tệp Pickle để lưu trữ từ điển để mô hình không phải được đào tạo mỗi lần.
7. log.txt Tệp nhật ký chứa các bản ghi nhật ký hadoop.

Link: https://github.com/dolaram/Naive-Bayes-using-MapReduce

Phần II: Thực hiện project tại lớp

1. Xử lý dữ liệu MapRedure hóa với Naïve Bayes

Bài toán: Dự đoán có đi chơi Tennis hay không

• Có 2 lớp dự báo:

– C1 = "yes" => Có đi chơi Tennis

– C2 = "no" => Không đi chơi Tennis

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

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
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

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

• Thuật toán MapReduce trong Naive Bayes cũng được chia thành 2 phần:

– Phần 1 - MapReduce hoá để huấn luyện Naive Bayes:

• Nhiệm vụ đếm số lần xuất hiện của các C_i và $x_k|C_i$

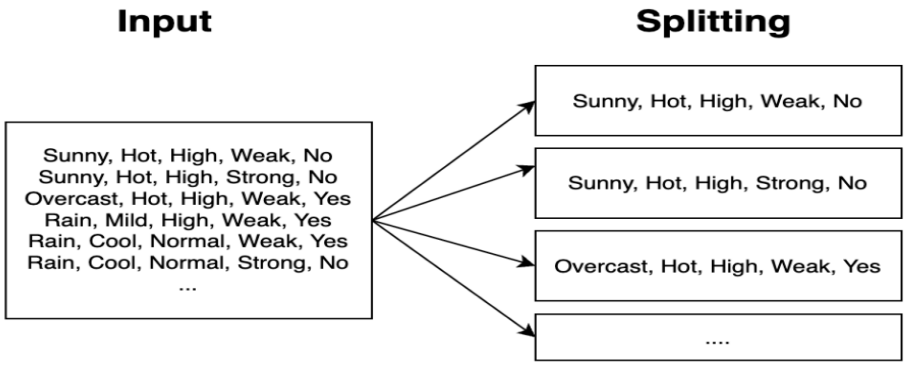
• Tổng hợp kết quả để phục vụ tính $P(C_i)$ và $P(x_k|C_i)$

– Phần 2 - MapReduce hoá để phân loại:

• Sử dụng các xác suất $P(C_i)$ và $P(x_k|C_i)$ để phân lớp tập dữ liệu X^{new}

MapRedure hóa để huấn luyện:

- Bước 1: Input - Chuẩn bị dữ liệu
  - Dữ liệu đầu vào là danh sách các bản ghi đã phân loại ở trong ví dụ dự đoán có đi chơi tennis hay không.
- Bước 2: Splitting
  - Chia dữ liệu thành từng dòng để xử lý

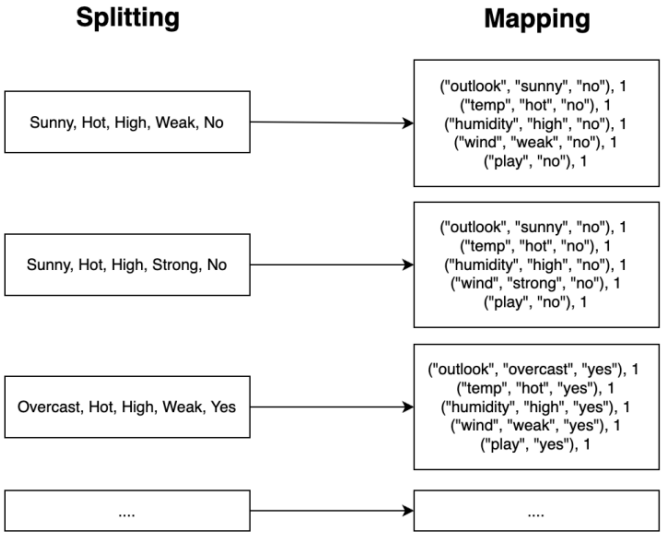


- Bước 3: Mapping
  - Tạo thành các cặp key/value, với key là C_i và $x_k|C_i$, value là 1
  - Ví dụ ở dòng Sunny, Hot, High, Weak, No

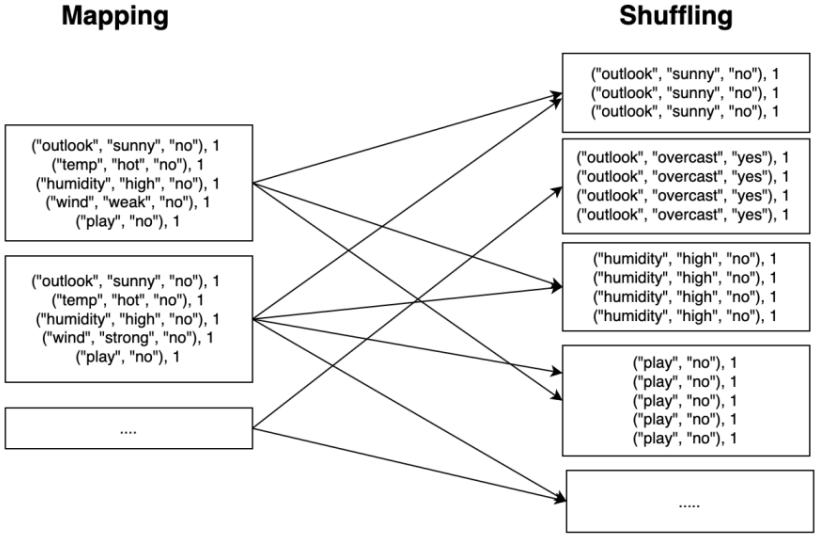
  Ta có:
  - Với $x_k|C_i$:

    ("outlook", "sunny", "no"), 1
    ("temp", "hot", "no"), 1
    ("humidity", "high", "no"), 1
    ("wind", "weak", "no"), 1
  - Với C_i:

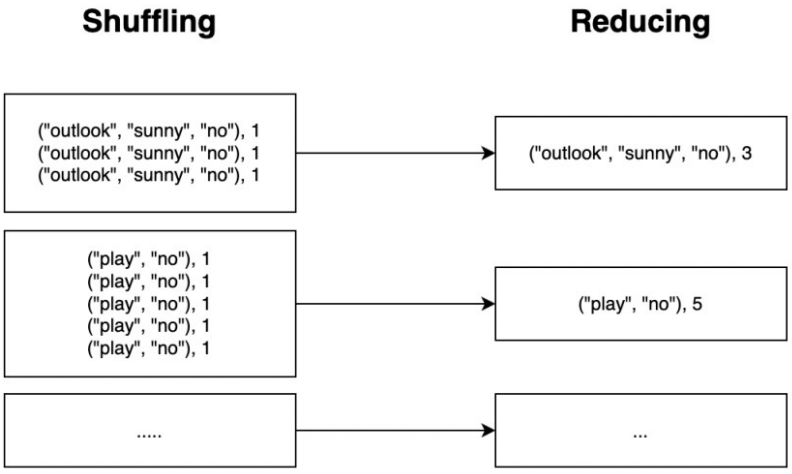
    ("play", "no"), 1



- Bước 4: Shuffling and Sort



- Bước 5: Reducing
  - Tính tổng các giá trị giống như bài toán WordCount



Mapredure hóa để phân loại:

- Bước 1: Input - Chuẩn bị dữ liệu
  - Tập dữ liệu chưa được gán nhãn, ví dụ:

Sunny, Cool, High, Strong

Rain, Cold, High, Weak

Overcast, Cold, High, Strong

- Bước 2: Splitting
  - Chia dữ liệu thành từng dòng để xử lý

- Bước 3: Mapping
  - Từ các kết quả tổng hợp đếm của các biến C_i và $x_k|C_i$ ở phần 1, ta đã có đủ dữ liệu để tính xác xuất $P(C_i|x_k)$. Ví dụ: với dữ liệu Sunny, Cool, High, Strong
    - $P(No) = \frac{val(\text{“play”},\text{“no”})}{val(\text{“play”},\text{“no”}) + val(\text{“play”},\text{“yes”})} = \frac{5}{14}$
    - $P(Sunny|No) = \frac{val(\text{“outlook”},\text{“sunny”},\text{“no”})}{val(\text{“no”})} = \frac{3}{5}$
    - ...
  - Sau khi tính được xác xuất thành phần ta có thể tính được $P(C_i|x_k)$ theo định lý Naive Bayes và so sánh các kết quả

Hoàn thành và submit project trên LMS

2. Xử lý dữ liệu MapRedure hóa với K-means

Bài toán: cho tập dữ liệu là các điểm, yêu cầu chia tập dữ liệu thành các cụm

Dữ liệu đầu vào:

- Là danh sách các điểm và tọa độ (có thể lưu trên file txt)
- Mỗi hàng là dữ liệu về điểm, tọa độ các điểm: A,3,4
- Được chuyển sang kiểu key/value làm đầu vào cho thuật toán

Mô hình cơ bản của MapRedure hóa với K-means:

Ý tưởng: tách dữ liệu thành các nhóm nhỏ. Với mỗi nhóm ta tính:

– **Map:**

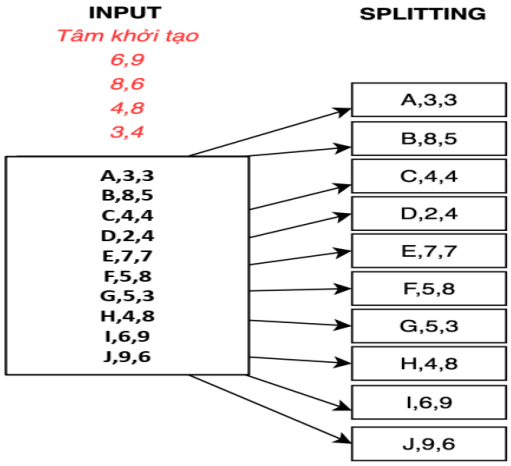
- Phân cụm trên từng nhóm nhỏ dữ liệu
- Với mỗi điểm dữ liệu, tìm trọng tâm gần nhất
- Tạo cặp key/value, với key là trọng tâm gần nhất, value là toạ độ của điểm đang xét

– **Shuffle and Sort**: Tất cả dữ liệu được gom theo từng tâm

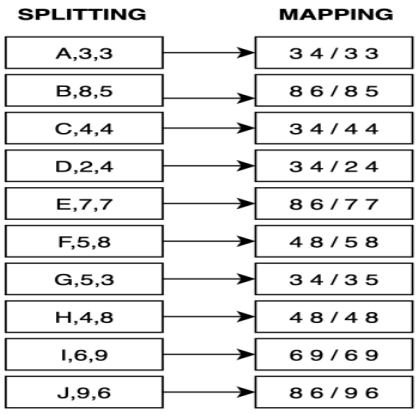
– **Reduce**:

- Tính tâm mới của các dữ liệu được gom (theo từng tâm)

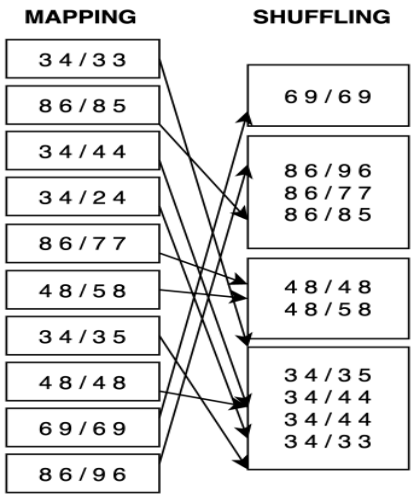
- Bước 1: Input
  - Thu thập dữ liệu cần phân cụm
  - Khởi tạo K tâm cụm
- Bước 2: Splitting
  - Chia tập dữ liệu thành 1 hoặc một nhóm các điểm



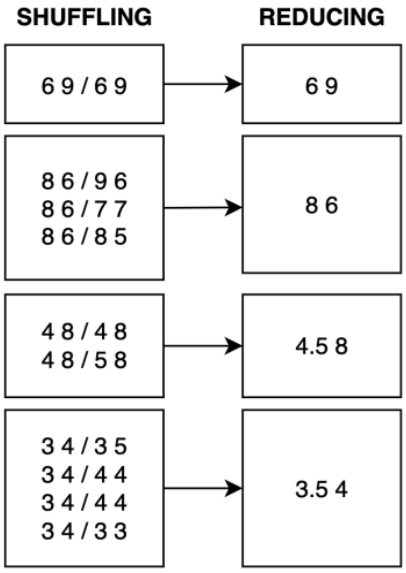
- Bước 3: Mapping
  - Tính khoảng cách các điểm đến các tâm cụm
  - Tạo cặp key/value, với key là trọng tâm gần nhất, value là toạ độ của điểm đang xét



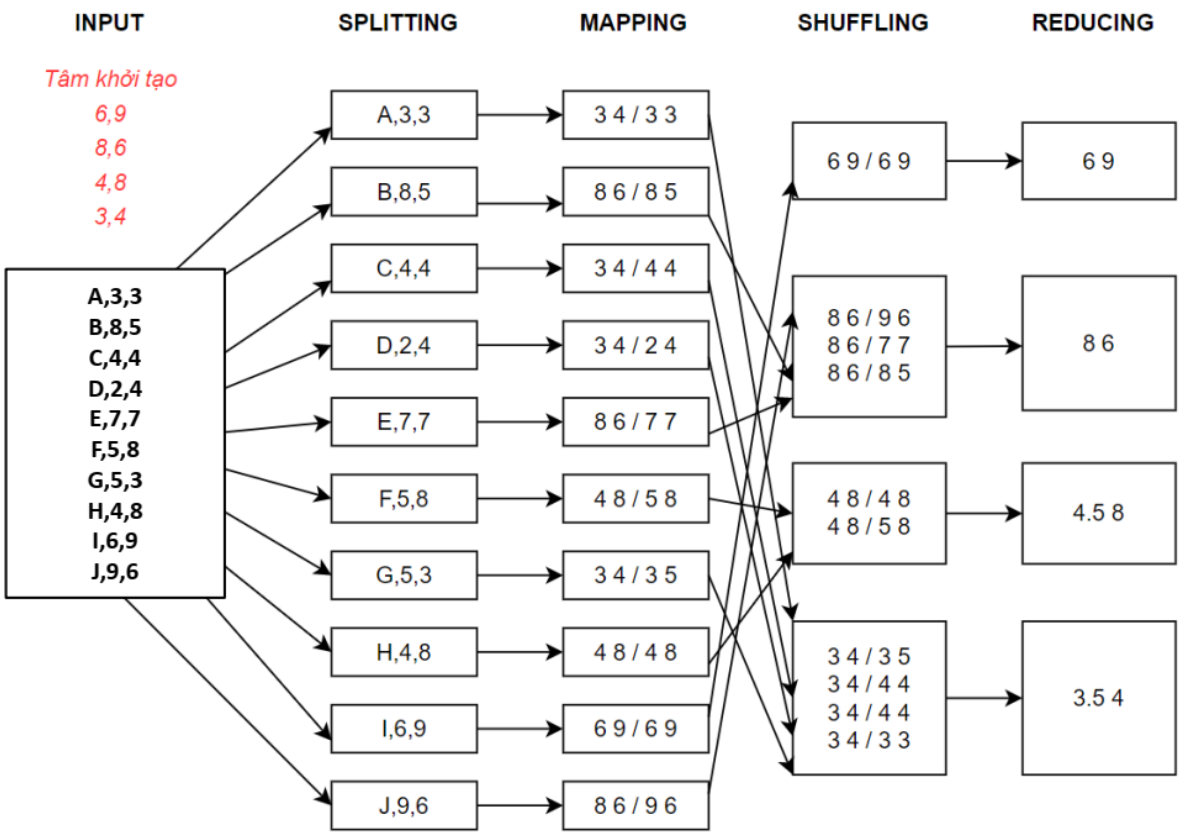
- Bước 4: Shuffling
  - Gom các dữ liệu có cùng key thành 1 nhóm



- Bước 5: Reducing
  - Tính toạ độ tâm mới bằng cách tính trung bình cộng các điểm đã được gán trong tâm



- Bước 6: Lặp lại và kiểm tra điểm dừng
  - Gán tâm mới vào tâm khởi tạo ban đầu và thực hiện một vòng lặp mới
  - Điều kiện dừng khi các tâm cũ không đổi so với tâm mới



Hoàn thành và submit project trên LMS