**TRƯỜNG ĐẠI HỌC THỦY LỢI**

**KHOA CÔNG NGHỆ THÔNG TIN**

**----------------------------------------**  
  


**BÁO CÁO BÀI TẬP LỚN**

**HỌC PHẦN PHÂN TÍCH DỮ LIỆU LỚN**

**ỨNG DỤNG HADOOP MAPREDUCE VÀ THUẬT TOÁN K-MEANS TRONG BÀI TOÁN PHÂN CỤM DỮ LIỆU BỆNH NHÂN**

Nhóm sinh viên thực hiện: **Nhóm 15**

*Vũ Văn Thủy - 62TH4*

*Trần Trọng Tấn - 62TH4*

*Nguyễn Phú Trường - 62TH4*

**Giảng viên hướng dẫn: TS. Tạ Quang Chiểu**

**Hà Nội, Tháng 8 Năm 2023.**

**MỤC LỤC**

[***CHƯƠNG 1: TỔNG QUAN, GIỚI THIỆU BÀI TOÁN 2***](#_Toc143508884)

[*1.1. Giới thiệu bài toán 2*](#_Toc143508885)

[*1.2. Tổng quan các kỹ thuật, phương pháp liên quan đến bài toán 2*](#_Toc143508886)

[***CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 2***](#_Toc143508887)

[*2.1. Tổng quan về dữ liệu lớn 2*](#_Toc143508888)

[*2.2. Tổng quan về Apache Hadoop 3*](#_Toc143508889)

[*2.3. Tổng quan về Mapreduce 4*](#_Toc143508890)

[*2.4. Tổng quan về thuật toán K-means 5*](#_Toc143508891)

[*2.2.1. Thuật toán K-means 5*](#_Toc143508892)

[*2.4.2. MapReduce hóa thuật toán K-means: 6*](#_Toc143508893)

[***CHƯƠNG 3: KẾT QUẢ VÀ ĐÁNH GIÁ MÔ HÌNH 8***](#_Toc143508894)

[*3.1. Chi tiết bài toán 8*](#_Toc143508895)

[*3.1.1. Phân tích dữ liệu thô 8*](#_Toc143508896)

[*3.1.2. Tiền xử lý dữ liệu 8*](#_Toc143508897)

[*3.1.3. Xác định số lượng cụm K 10*](#_Toc143508898)

[*3.1.4. Sử dụng MongoDB để lưu trữ dữ liệu 12*](#_Toc143508899)

[*3.1.5. Xây dựng các API bằng Node.js để thao tác với MongoDB 13*](#_Toc143508900)

[*3.1.6. Sử dụng Hadoop MapReduce để phân cụm dữ liệu 14*](#_Toc143508901)

[*3.2. Kết quả bài toán 16*](#_Toc143508902)

[*3.2.1. Demo cài đặt chương trình 16*](#_Toc143508903)

[*3.2.2. Kết quả 17*](#_Toc143508904)

[*3.2.3. Demo ứng dụng dự đoán nhóm sức khỏe của bệnh nhân 19*](#_Toc143508905)

[***CHƯƠNG 4: KẾT LUẬN 20***](#_Toc143508906)

**DANH MỤC HÌNH ẢNH**

[*Hình 2.1 Các đặc điểm của dữ liệu lớn 4*](#_Toc143522604)

[*Hình 2.2 Biểu tượng của Apache Hadoop 4*](#_Toc143522605)

[*Hình 2.3 Các thành phần chính của Apache Hadoop 5*](#_Toc143522606)

[*Hình 2.4 Mô hình MapReduce 5*](#_Toc143522607)

[*Hình 2.5 Lưu đồ thuật toán K-means 7*](#_Toc143522608)

[*Hình 2.6 Lưu đồ MapReduce hóa thuật toán K-means 7*](#_Toc143522609)

[*Hình 3.1 Dữ liệu ban đầu 9*](https://d.docs.live.net/93b2a7f392a6d5cd/Tài%20liệu/Nhóm%2015%20-%20Báo%20cáo%20bài%20tập%20lớn%20-%20Phân%20tích%20dữ%20liệu%20lớn.docx#_Toc143522631)

[*Hình 3.2 Dữ liệu sau khi đã được chuẩn hóa 11*](#_Toc143522632)

[*Hình 3.3 Biểu đồ Elbow 12*](#_Toc143522633)

[*Hình 3.4 Import file CSV vào MongoDB (1) 13*](#_Toc143522634)

[*Hình 3.5 Import file CSV vào MongoDB (2) 14*](#_Toc143522635)

[*Hình 3.6 Dữ liệu sau khi đã được import vào MongoDB 14*](#_Toc143522636)

[*Hình 3.7 Kết nối tới MongoDB 17*](#_Toc143522637)

[*Hình 3.8 Khởi chạy các API 17*](#_Toc143522638)

[*Hình 3.9 Cấu trúc thư mục của project 18*](#_Toc143522639)

[*Hình 3.10 Câu lệnh chạy file jar 18*](#_Toc143522640)

[*Hình 3.11 Dữ liệu sau khi đã được đọc từ MongoDB vào file input 18*](#_Toc143522641)

[*Hình 3.12 Kết quả in ra file 3 tâm cụm 19*](#_Toc143522642)

[*Hình 3.13 Dữ liệu 3 tâm cụm được lưu vào MongoDB 19*](#_Toc143522643)

[*Hình 3.14 Dữ liệu các điểm thuộc mỗi tâm cụm 20*](#_Toc143522644)

[*Hình 3.15 Ứng dụng dự đoán nhóm sức khỏe của bệnh nhân 21*](#_Toc143522645)

# CHƯƠNG 1: TỔNG QUAN, GIỚI THIỆU BÀI TOÁN

## Giới thiệu bài toán

Bài toán tập trung vào việc ứng dụng kỹ thuật phân tích dữ liệu sử dụng Hadoop MapReduce kết hợp với thuật toán phân cụm K-means trong bài toán phân cụm dữ liệu bệnh nhân.

Bộ dữ liệu được sử dụng cho bài toán chứa các thông tin đa dạng về chỉ số sức khỏe của bệnh nhân. Mục tiêu chính của bài toán là thực hiện phân cụm dữ liệu để xác định nhóm tình trạng sức khỏe của bệnh nhân dựa trên các đặc trưng liên quan đến sức khỏe của họ.

Kết quả của bài toán sẽ là việc phân loại các bệnh nhân vào các nhóm tương ứng với tình trạng sức khỏe của họ dựa trên sự tương tự về đặc trưng sức khỏe. Điều này có thể giúp cung cấp thông tin quan trọng về xu hướng và mối liên quan giữa các chỉ số sức khỏe khác nhau và tình trạng bệnh. Từ đó có thể đưa ra các đề xuất về biện pháp chăm sóc sức khỏe và phòng ngừa bệnh cho bệnh nhân.

## Tổng quan các kỹ thuật, phương pháp liên quan đến bài toán

* **K-means Clustering**: là một thuật toán phân cụm không giám sát, sử dụng để phân chia các điểm dữ liệu của các bệnh nhân vào các cụm dựa trên sự tương tự về đặc điểm giữa các bệnh nhân.
* **MapReduce**: là một mô hình lập trình và hệ thống xử lý dữ liệu song song phục vụ cho việc xử lý các tập dữ liệu lớn. Bằng cách chia nhỏ nhiệm vụ tính toán thành các phần nhỏ và thực hiện trên nhiều node khác nhau.
* **HDFS (Hadoop Distributed File System)**: là hệ thống lưu trữ dữ liệu phân tán của Hadoop**.**
* **MongoDB**: là một hệ quản trị cơ sở dữ liệu không quan hệ, sử dụng để lưu trữ dữ liệu của bệnh nhân, và lưu trữ thông tin của các cụm, phục vụ cho ứng dụng dự đoán bệnh nhân thuộc nhóm đặc điểm về sức khỏe nào.
* **Node.js**: là môi trường thực thi Javascript phía máy chủ, sử dụng để xây dựng ứng dụng và các API để thao tác với cơ sở dữ liệu MongoDB.
* **Python**: sử dụng cho việc tiền xử lý dữ liệu và xác định số lượng cụm.

# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

## 2.1. Tổng quan về dữ liệu lớn

Theo Wikipedia: Dữ liệu lớn là một thuật ngữ cho việc xử lý một tập hợp dữ liệu rất lớn và phức tạp mà các ứng dụng xử lý dữ liệu truyền thống không xử lý được. [1]



Hình 2.1 Các đặc điểm của dữ liệu lớn

Dữ liệu lớn thường có bốn đặc điểm chính:

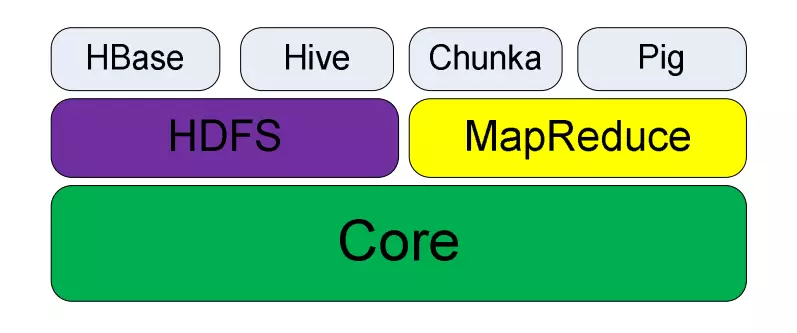
* **Khối lượng (Volume):** Dữ liệu lớn đề cập đến khối lượng thông tin khổng lồ, thường là hàng tỷ hoặc triệu tỷ bản ghi dữ liệu.
* **Tốc độ (Velocity):** Dữ liệu lớn được tạo ra và cập nhật với tốc độ nhanh chóng từ nhiều nguồn khác nhau, đặc biệt trong thời gian thực.
* **Đa dạng (Variety):** Dữ liệu lớn có đa dạng về nguồn gốc và định dạng, bao gồm văn bản, hình ảnh, video, dữ liệu cảm biến, và nhiều loại dữ liệu khác.
* **Chính xác (Veracity):** Chính xác của dữ liệu là yếu tố quan trọng, đảm bảo rằng dữ liệu không bị nhiễu, thiếu hoặc chứa thông tin sai lệch.

## 2.2. Tổng quan về Apache Hadoop

Apache Hadoop là một framework dùng để chạy những ứng dụng trên 1 cluster lớn được xây dựng trên những phần cứng thông thường. Hadoop hiện thực mô hình Map/Reduce, đây là mô hình mà ứng dụng sẽ được chia nhỏ ra thành nhiều phân đoạn khác nhau, và các phần này sẽ được chạy song song trên nhiều node khác nhau. [2]



Hình 2.2 Biểu tượng của Apache Hadoop



Hình 2.3 Các thành phần chính của Apache Hadoop

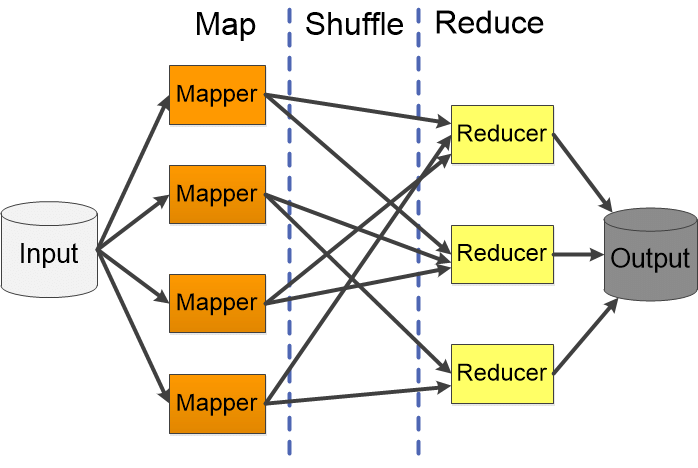
Các thành phần chính của Apache Hadoop bao gồm: HDFS, MapReduce, Hbase, Hive, Pig, Chunka,…Tuy nhiên hai thành phần chính và quan trọng nhất là HDFS và MapReduce

* Hadoop Distributed File System (HDFS): HDFS là hệ thống lưu trữ dữ liệu phân tán của Hadoop, chia nhỏ dữ liệu thành các khối và lưu trữ trên nhiều node khác nhau. Điều này đảm bảo tính sẵn sàng, độ tin cậy và khả năng mở rộng của dữ liệu lớn.
* MapReduce: MapReduce là mô hình lập trình và hệ thống xử lý dữ liệu song song trong Hadoop. Qua hai bước "map" và "reduce", nó phân tách và xử lý dữ liệu lớn trên nhiều máy tính, giúp tối ưu hóa quá trình xử lý và phân tích dữ liệu.

## 2.3. Tổng quan về Mapreduce

MapReduce là mô hình được thiết kế độc quyền bởi Google, nó có khả năng lập trình xử lý các tập dữ liệu lớn song song và phân tán thuật toán trên 1 cụm máy tính.

Mô hình MapReduce chia quá trình xử lý dữ liệu thành hai phần chính: pha ‘Map’ và pha ‘Reduce’



Hình 2.4 Mô hình MapReduce

* **Map Phase**: Trong phần này, dữ liệu được chia nhỏ thành các cặp khóa-giá trị. Một hàm map (map function) được áp dụng cho mỗi cặp khóa-giá trị, tạo ra các cặp khóa-giá trị trung gian. Quá trình ánh xạ này giúp tiến hành xử lý song song trên nhiều máy tính.
* **Shuffle và Sort Phase**: Sau khi giai đoạn "Map" hoàn thành, các cặp khóa-giá trị trung gian được tổng hợp lại dựa trên khóa, để chuẩn bị cho giai đoạn "Reduce". Quá trình này được gọi là "shuffle" và "sort".
* **Reduce Phase**: Trong giai đoạn này, các cặp khóa-giá trị có cùng khóa được gom nhóm lại và áp dụng hàm reduce (reduce function) để tính toán kết quả cuối cùng. Quá trình này giúp tổng hợp dữ liệu từ các máy tính khác nhau thành kết quả cuối cùng.

## 2.4. Tổng quan về thuật toán K-means

### 2.2.1. Thuật toán K-means

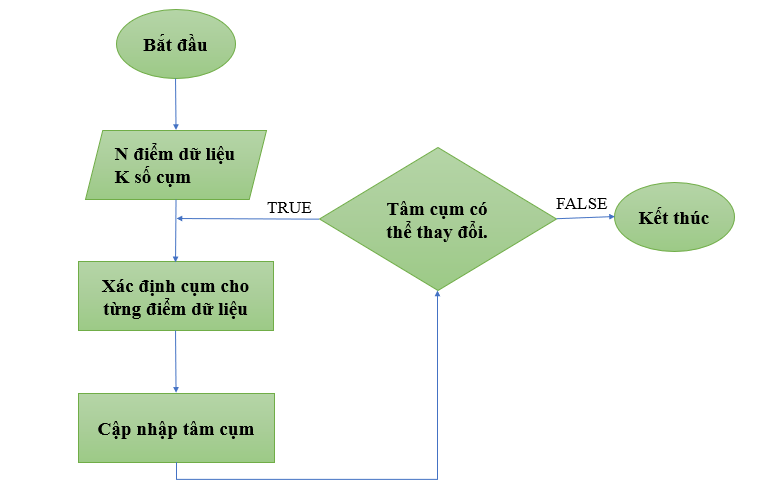
Thuật toán K-means là một trong các thuật toán phân cụm đơn giản và điển hình nhất, do MacQueen đề xuất trong lĩnh vực thống kê năm 1967. Mục đích của thuật toán là sinh ra *k* cụm dữ liệu từ một tập dữ liệu ban đầu gồm *n* đối tượng trong không gian *p* chiều.

**Phát biểu bài toán phân cụm:**

* Input: *n* đối tượng và số các cụm *k*
* Output: các cụm *Ci (i = 1…k)* sao cho hàm tiêu chuẩn E đạt giá trị tối thiểu

**Các bước của thuật toán K-mean: [3]**

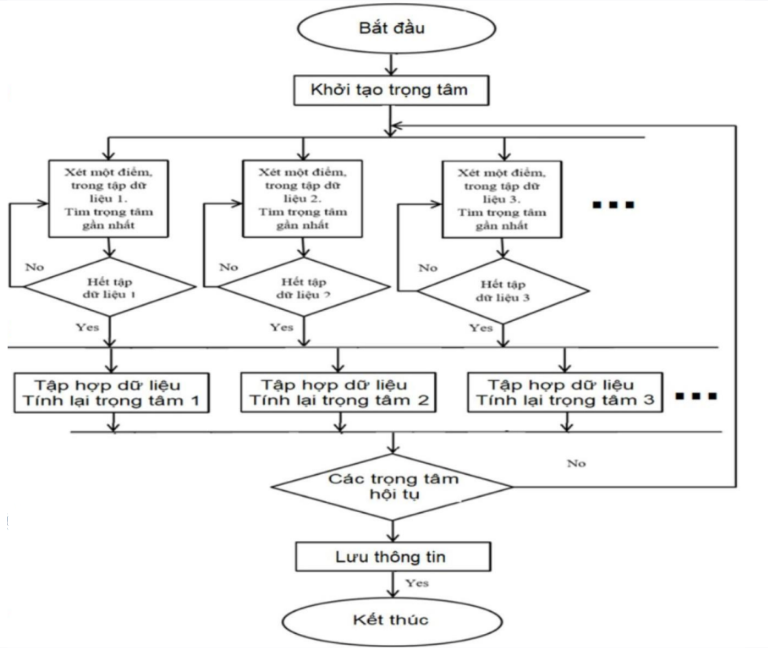
* Bước 1: Khởi tạo tâm cụm
  + Chọn *k* đối tượng *mj (j=1…k)* là trọng tâm ban đầu của *k* cụm từ tập dữ liệu
* Bước 2: Tính toán khoảng cách và gán cụm
  + Với mỗi điểm dữ liệu, tính toán khoảng cách từ nó tới mỗi tâm cụm đã được khởi tạo, sau đó tìm tâm cụm gần nhất đối với mỗi đối tượng
  + Công thức tính khoảng cách Euclidean giữa hai điểm A (a1, a2, ..., an) và B (b1, b2, ..., bn) trong không gian n chiều là:
* 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 lại các bước 2 và 3 cho đế khi các tâm cụm không thay đổi



Hình 2.5 Lưu đồ thuật toán K-means

### 2.4.2. MapReduce hóa thuật toán K-means:

MapReduce hóa thuật toán K-means giúp tận dụng khả năng xử lý song song của MapReduce để thực hiện thuật toán K-means trên dữ liệu lớn.



Hình 2.6 Lưu đồ MapReduce hóa thuật toán K-means

**Các bước thực hiện:**

* 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
* 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 có thể xử lý song song và tăng tốc tính toán
* Trong mỗi vòng lặp
  + 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: gom các điểm thuộc cùng một cụm 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 tọa độ của một điểm
  + keyIn là giá trị byte offset của dòng
  + valIn là vector biểu diễn tọa 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 tâm gần nhất (trọng tâm hoặc chỉ số tâm)
  + valInt là tọa độ điểm thuộc cụm có trọng tâm là keyInt

**Xây dựng hàm Reduce\_K-means**

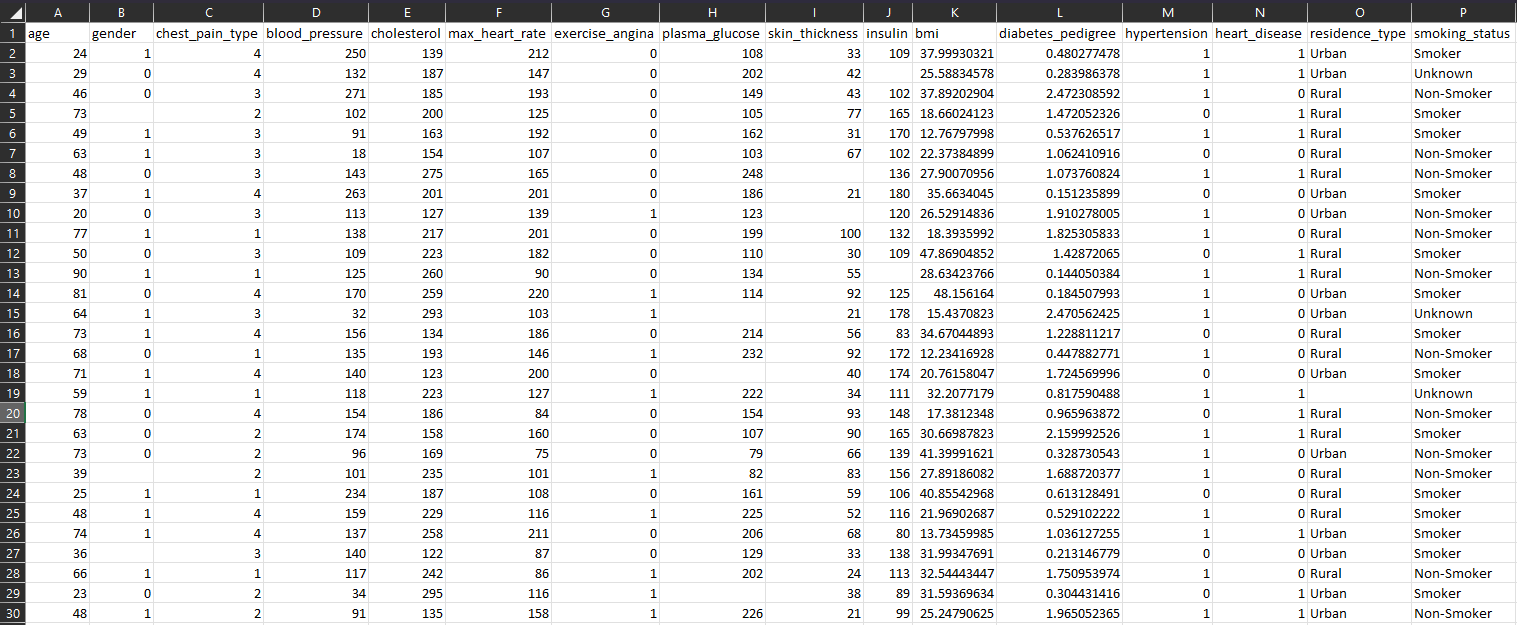
* Trước khi hàm reduce thực hiện
  + Kết quả của hàm map được trộn lại
  + Các cặp cùng keyInt được gom thành một nhóm
* Đầu vào:
  + keyInt được chuyển từ hàm map
  + list(valInt) là list các điểm vallInt thuộc về cụm thứ keyInt
* Xử lý:
  + Tính trung bình cộng từng thành phần của các điểm cùng cụm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

# CHƯƠNG 3: KẾT QUẢ VÀ ĐÁNH GIÁ MÔ HÌNH

## 3.1. Chi tiết bài toán

### 3.1.1. Phân tích dữ liệu thô

Nguồn dữ liệu thô: [Patient Dataset for Clustering (Raw Data)](https://www.kaggle.com/datasets/arjunnsharma/patient-dataset-for-clustering-raw-data), bộ dữ liệu thô về bệnh nhân dùng cho việc phân cụm. [4]

* Số lượng thuộc tính: 16
* Số lượng bản ghi: 2000

Hình 3.1 Dữ liệu ban đầu

### 3.1.2. Tiền xử lý dữ liệu

#### 3.1.2.1 . Làm sạch dữ liệu

Làm sạch dữ liệu là quá trình nhận dạng dữ liệu đã có để tiến hành xử lý các dữ liệu bị thiết (missing data), xử lý dữ liệu bị nhiễu (noisy data) và dữ liệu không nhất quán (inconsistent data).

Đọc file dữ liệu và thực hiện làm sạch dữ liệu bằng Python:

* Xóa những thuộc tính không cần thiết: những thuộc tính không cần thiết như ‘gender’, ‘skin\_thickness’, ‘residence\_type’ sẽ được xóa bỏ
* Xử lý dữ liệu bị thiếu: các dữ liệu bị thiếu thay thế bằng giá trị trung bình của thuộc tính đó sử dụng **mean()** của thư viện pandas
* Xử lý thuộc tính ‘smoking\_status’ bị thiếu hoặc có giá trị là ‘unknown’: thay thế bằng giá trị xuất hiện thường xuyên nhất của thuộc tính đó, sử dụng **mode()** của thư viện pandas
* Xử lý dữ liệu bị nhiễu: thuộc tính ‘blood\_pressure’ – huyết áp có các giá trị bất thường, không hợp lệ

#### 3.1.2.2. Biến đổi dữ liệu

Biến đổi dữ liệu là quá trình thay đổi cấu trúc hoặc nội dung của dữ liệu để tạo ra các biến mới hoặc thay đổi dạng của dữ liệu hiện có. Mục tiêu của bước này là cải thiện tính phù hợp của dữ liệu với mục tiêu phân tích hoặc mô hình hóa.

Đọc file dữ liệu và thực hiện biến đổi dữ liệu bằng Python:

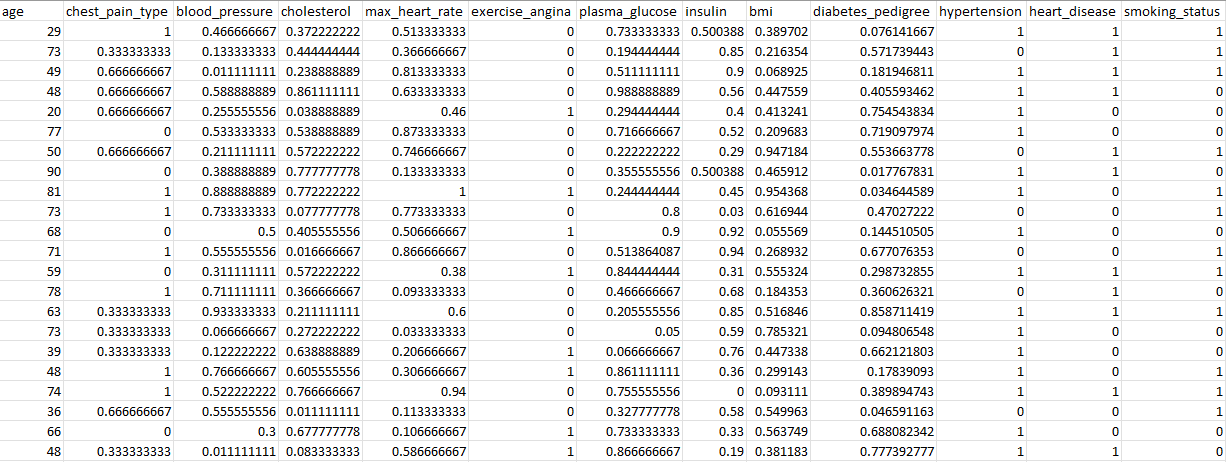
* Biến đổi dữ liệu bằng phương pháp Min Max Scaling: đưa các giá trị dữ liệu về khoảng giá trị cụ thể, thường là trong khoảng từ 0 đến 1.
* Công thức:

Trong đó:

* v = [minA, maxA] là giá trị cũ
* v’ = [0,1] là giá trị mới

Ví dụ:

* Thuộc tính ‘blood\_pressure’: có v = [90, 180]
* Chuẩn hóa về giá trị: v’ = [0, 1]
* Với v = 105
  + - v’ = (105 - 90) / (180 - 90) = 0.1667
* Biến đổi dữ liệu bằng phương pháp One-hot encoding: biến các giá trị cố định ở dạng ký tự về dạng giá trị số, sử dụng **get\_dummies()** trong thư viện Pandas



Hình 3.2 Dữ liệu sau khi đã được chuẩn hóa

|  |  |  |
| --- | --- | --- |
| **STT** | **Thuộc tính** | **Ý nghĩ thuộc tính** |
| 1 | age | Tuổi |
| 2 | chest\_pain\_type | Loại đau ngực |
| 3 | blood\_pressure | Huyết áp |
| 4 | cholesterol | Cholesterol |
| 5 | max\_heart\_rate | Nhịp tim tối đa |
| 6 | exercise\_angina | Đau ngực do tập thể dục |
| 7 | plasma\_glucose | Glucose huyết tương |
| 8 | insulin | Insulin |
| 9 | bmi | Chỉ số BMI |
| 10 | diabetes\_pedigree | Khả năng mắc bệnh tiểu đường tùy thuộc vào độ tuổi hoặc tiền sử gia đình mắc bệnh tiểu đường |
| 11 | hypertension | Tăng huyết áp |
| 12 | heart\_disease | Bệnh tim |
| 13 | smoking\_status | Tình trạng hút thuốc |

Bảng 3.1 Liệt kê các thuộc tính của bộ dữ liệu sau khi đã chuẩn hóa

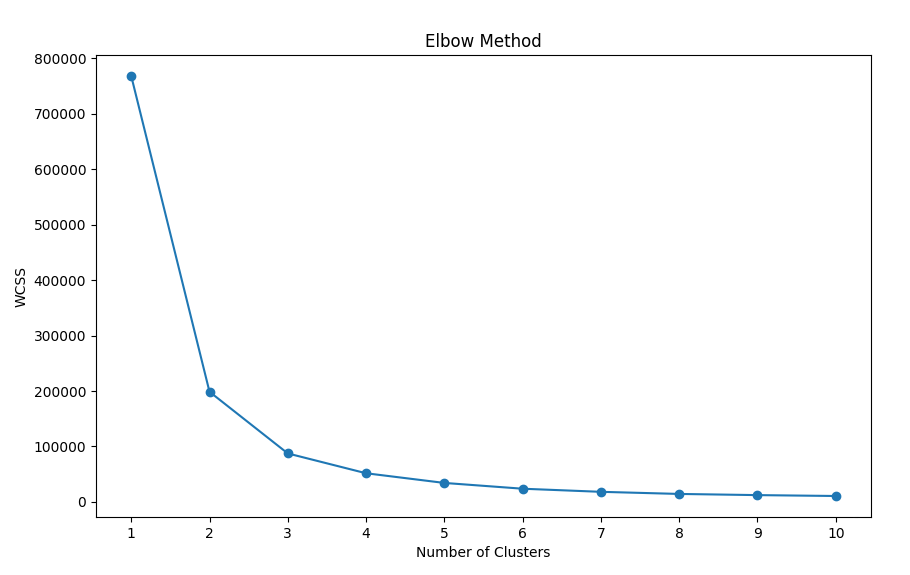
### 3.1.3. Xác định số lượng cụm K

Để xác định số lượng cụm K, cần sử dụng một biểu đồ được tạo bằng Elbow Method (phương pháp khuỷu tay) – là một phương pháp thường được sử dụng để chọn số lượng tốt nhất của các cụm trong thuật toán phân cụm. Phương pháp này giúp xác định số lượng cụm tối ưu dựa trên sự thay đổi của tổng bình phương khoảng cách (Within-Cluster Sum of Squares - WCSS) giữa các điểm dữ liệu và trung tâm của cụm.

Cách thức thực hoạt động của Elbow Method như sau:

* Thực hiện thuật toán KMeans với một loạt số lượng cụm khác nhau (thường từ 1 đến một giới hạn tối đa đã xác định từ trước).
* Đối với mỗi số lượng cụm, tính giá trị WCSS (tổng bình phương khoảng cách giữa các điểm dữ liệu và trung tâm cụm tương ứng).
* Vẽ biểu đồ Elbow Method với trục x là số lượng cụm và trục y là giá trị WCSS. Đường cong trên biểu đồ thường có dạng "cổ điển" như cùi chỏ, giống như đoạn của một cánh tay và một bàn tay đang giơ lên.
* Xác định điểm "bẻ" của đường cong (nơi mà đường cong thay đổi hướng từ việc giảm dần sang giảm chậm hơn). Điểm này thường gọi là "đỉnh của cùi chỏ" (elbow point).
* Số lượng cụm tốt nhất sẽ là số lượng cụm tương ứng với điểm "bẻ" trên biểu đồ Elbow Method.

Ý nghĩa của Elbow Method là khi số lượng cụm tăng, WCSS thường giảm vì các điểm dữ liệu được gom vào cụm một cách gần hơn với trung tâm cụm. Tuy nhiên, tại một điểm nào đó, việc tăng số lượng cụm không còn gây ra sự giảm đáng kể trong WCSS, và đây chính là điểm mà đường cong bẻ. Điều này xảy ra vì thêm một cụm mới không còn đáng giá trong việc giảm tổng bình phương khoảng cách nữa, và sự thay đổi giữa số lượng cụm đã đạt một mức ổn định.



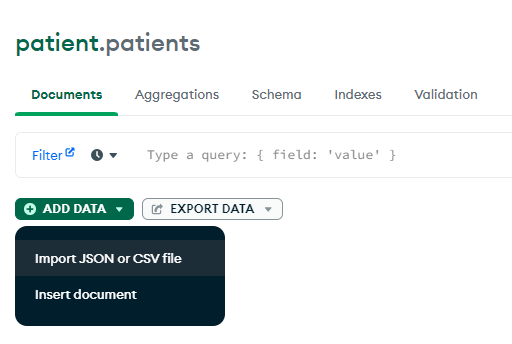
Hình 3.3 Biểu đồ Elbow

Từ biểu đồ trên, nhận thấy rằng tại số tượng cụm bằng 3 thì biểu đồ thay đổi hướng từ việc giảm dần sang giảm chậm hơn. Kết luận rằng số lượng cụm tối ưu cho dữ liệu là 3 cụm

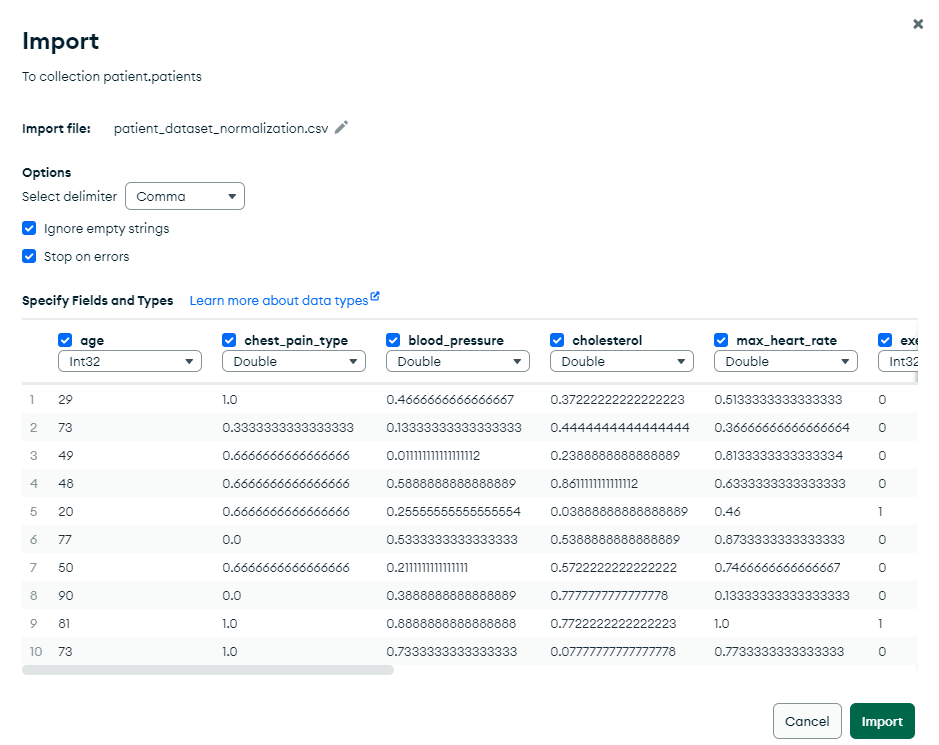
### 3.1.4. Sử dụng MongoDB để lưu trữ dữ liệu

Dữ liệu sau khi đã được chuẩn hóa, thực hiện import file csv vào trong MongoDB:

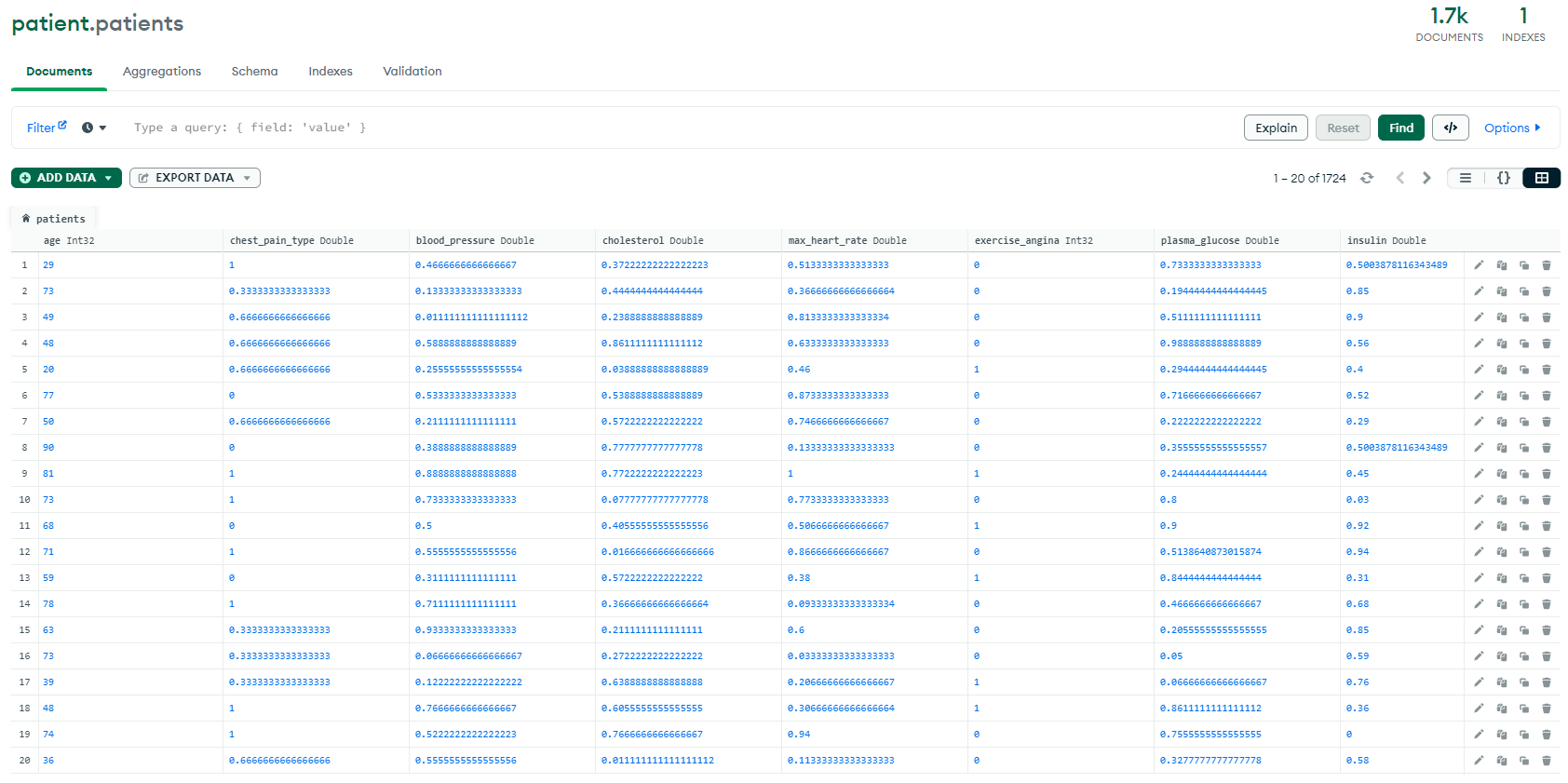
* Chọn ‘Import JSON or CSV file’ trong MongoDB Compass
* Chọn file CSV cần import
* Kiểm tra lại kiểu dữ liệu của các trường dữ liệu
* Ấn ‘Import’ và kiểm tra dữ liệu đã được import đầy đủ hay chưa



Hình 3.4 Import file CSV vào MongoDB (1)



Hình 3.5 Import file CSV vào MongoDB (2)



Hình 3.6 Dữ liệu sau khi đã được import vào MongoDB

### 3.1.5. Xây dựng các API bằng Node.js để thao tác với MongoDB

* Lấy dữ liệu của tất cả bệnh nhân:
  + URL: <http://127.0.0.1:3000/api/patients>
  + Phương thức: GET
  + Mô tả: API cho phép truy cập dữ liệu của tất cả các bệnh nhân trong MongoDB để phục vụ cho việc phân cụm dữ liệu bệnh nhân bằng Hadoop MapReduce
  + Phản hồi: trả về tất cả dữ liệu của bệnh nhân ở dạng chuỗi
* Lưu tâm cụm vào MongoDB
  + URL: <http://127.0.0.1:3000/api/centroids>
  + Phương thức: POST
  + Body: dữ liệu của các tâm cụm dạng chuỗi
  + Mô tả: API cho phép lưu trữ các tâm cụm sau khi đã thực hiện phân cụm bằng thuật toán K-means vào MongoDB
  + Phản hồi: trả về dữ liệu của các tâm cụm nếu lưu thành công
* Dự đoán nhóm sức khỏe của bệnh nhân:
  + URL: <http://127.0.0.1:3000/api/centroid-predict>
  + Phương thức: POST
  + Body: dữ liệu về các thuộc tính sức khỏe của một bệnh nhân dạng JSON
  + Mô tả: API thực hiện chuẩn hóa dữ liệu đầu vào, sau đo tính toán khoảng cách từ điểm dữ liệu đầu tới các tâm cụm xem tâm cụm nào gần nhất, sau đó thực hiện lưu dữ liệu vào MongoDB
  + Phản hồi: trả về dữ liệu của bệnh nhân đã được chuẩn hóa, dữ liệu của tâm cụm gần nhất và những đặc trưng về sức khỏe của tâm cụm đó

### 3.1.6. Sử dụng Hadoop MapReduce để phân cụm dữ liệu

#### 3.1.6.1. Xây dựng lớp MongoDBProcess

* Hàm *fetchAndWriteDataToHDFS(conf, inputFilePath):* thực hiện gọi đến API lấy dữ liệu của bệnh nhân, lưu kết quả phản hồi của API là dữ liệu của các bệnh nhân vào file input trên HDFS
* Hàm *saveCentroidsToMongoDB(centroids):* thực hiện gửi dữ liệu của các tâm cụm đến API lưu tâm cụm vào MongoDB và trả về thông báo.

#### 3.1.6.2. Xây dựng lớp PointWritable

* Lớp PointWritable để khởi tạo các đối tượng điểm dữ liệu bao gồm các thuộc tính
  + attibutes: mảng chứa các giá trị thuộc tính của mỗi điểm dữ liệu
  + dim: số chiều của mỗi điểm dữ liệu, tức là số lượng thuộc tính trong mỗi điểm
  + nPoints: đếm số lượng điểm dữ liệu được gộp lại với nhau, để tính tổng của tất cả các cụm để tính ra tâm cụm mới
* Cung cấp các phương thức để đọc và ghi dữ liệu của các điểm dữ liệu, tính tổng các giá trị thuộc tính, tính khoảng cách dữ điểm hiện tại và điểm p dựa vào các thuộc tính của chúng, tính trung bình của các thuộc tính trong dữ liệu dựa trên số lượng điểm đã gộp

#### 3.1.6.3. Xây dựng lớp Kmapper

* Thực hiện việc tính toán khoảng cách từ điểm dữ liệu đến tất cả các tâm cụm, sau đó xác định cụm gần nhất.
* Đầu vào: key là vị trí dòng của điểm trong tập tin đầu vào. value là dòng dữ liệu chứa thông tin của một điểm, được biểu diễn dưới dạng chuỗi văn bản.
* Đầu ra: centroidId là id của tâm cụm gần nhất cho điểm. pointInput là điểm dữ liệu với thông tin của các thuộc tính và số lượng điểm trong cụm.

#### 3.1.6.4. Xây dựng lớp KCombiner

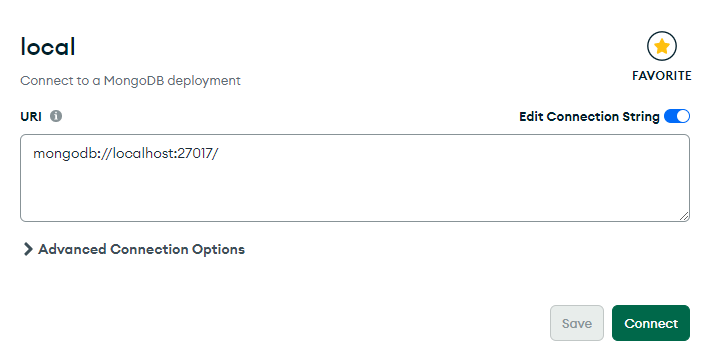
* Thực hiện tính toán tổng các điểm dữ liệu thuộc cùng một cụm và ghi thông tin điểm vào tập tin đầu ra, để lưu trữ các điểm thuộc cùng một cụm vào các tập tin riêng biệt.
* Đầu vào : centroidId là khóa đại diện cho id của tâm cụm, và points là danh sách các điểm thuộc tâm cụm.
* Đầu ra : centroidId vẫn là khóa đại diện cho id của tâm cụm, và ptSum là tổng của các điểm trong cùng một cụm sau khi đã kết hợp.

#### 3.1.6.5. Xây dựng lớp KReducer

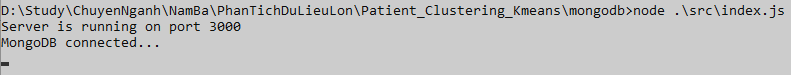
* Tính toán trung bình của các điểm trong cùng một cụm và tìm ra các điểm tâm cụm mới.
* Đầu vào: centroidId là khóa đại diện id của tâm cụm. partialSums là danh sách các điểm thuộc cụm tương ứng đã được tổng hợp từ bước KCombiner.
* Đầu ra: newCentroidId là khóa đại diện id của tâm cụm mới. newCentroidValue là giá trị của tâm cụm mới.

## 3.2. Kết quả bài toán

### 3.2.1. Demo cài đặt chương trình

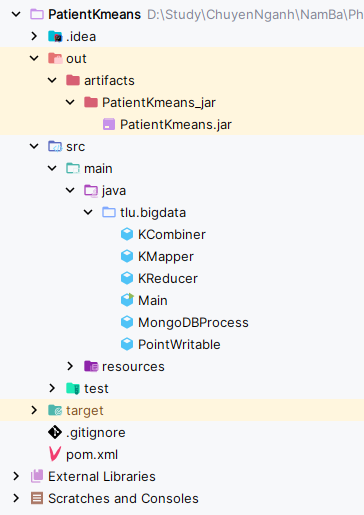
* Kết nối tới MongoDB:

Hình 3.7 Kết nối tới MongoDB

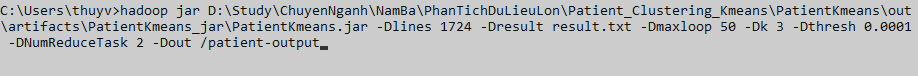
* Di chuyển vào thư mục chứa các API thao tác với MongoDB và giao diện của ứng dụng dự đoán nhóm sức khỏe của bệnh nhân, thực hiện câu lệnh ‘*node .\src\index.js’*  để khởi chạy các API và kết nối đến MongoDB: 

Hình 3.8 Khởi chạy các API

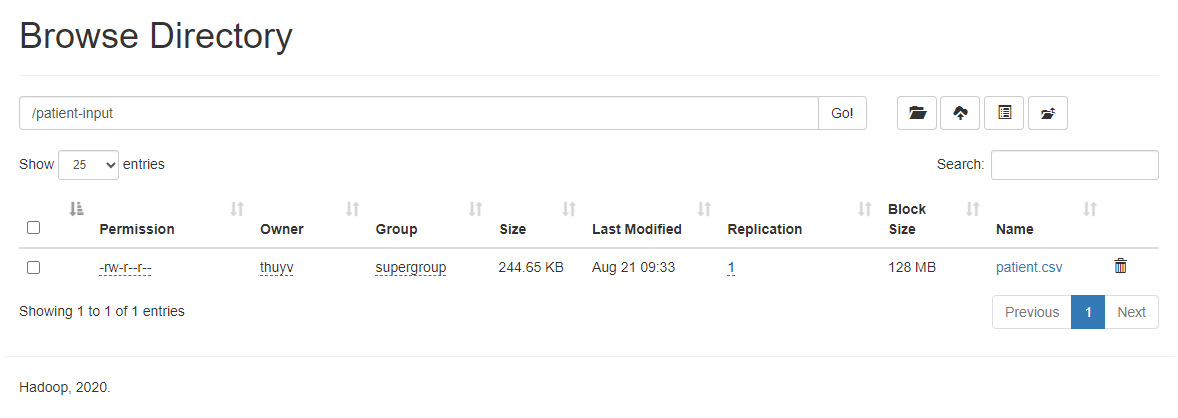
* Thực hiện build project thành file jar:



Hình 3.9 Cấu trúc thư mục của project

* Chạy file jar bằng câu lệnh

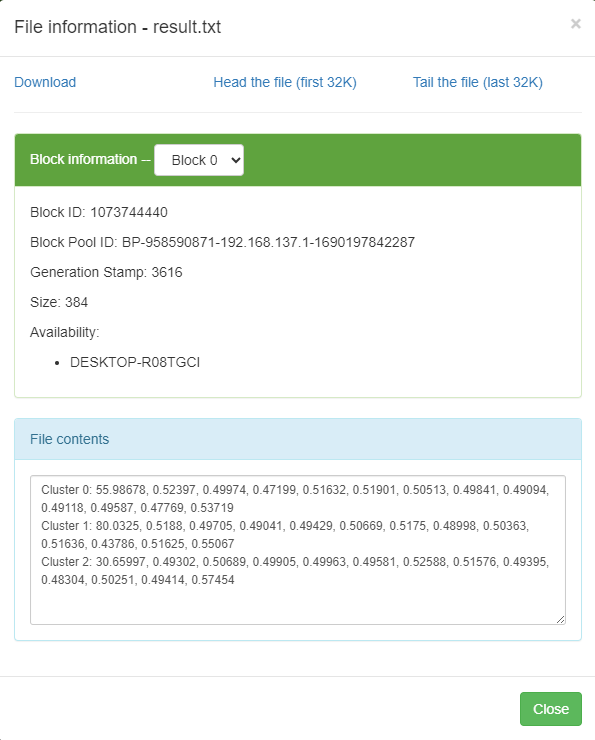
Hình 3.10 Câu lệnh chạy file jar

* Chương trình sẽ gọi đến API lấy dữ liệu bệnh nhân và đọc kết quả trả về của API đó để lưu file input vào HDFS

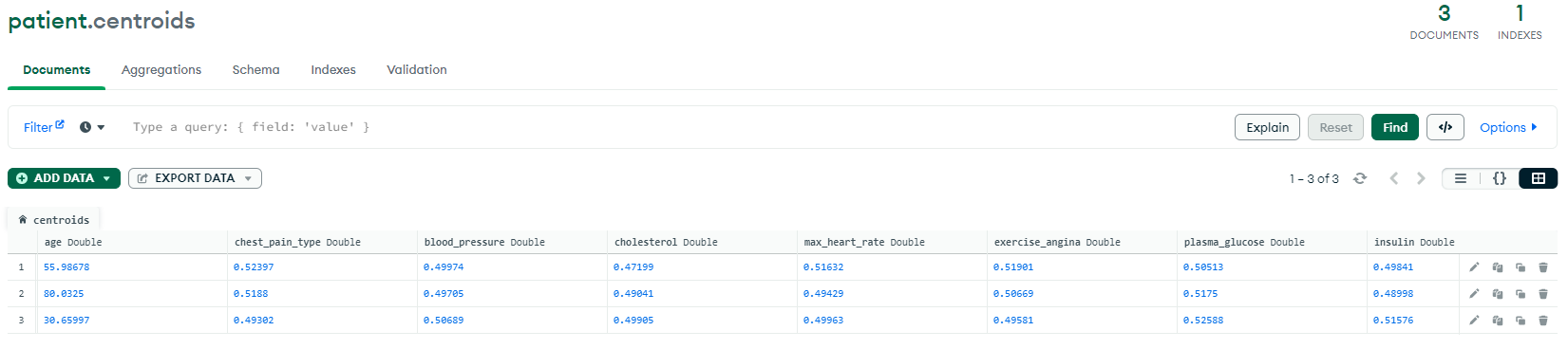
Hình 3.11 Dữ liệu sau khi đã được đọc từ MongoDB vào file input

### 3.2.2. Kết quả

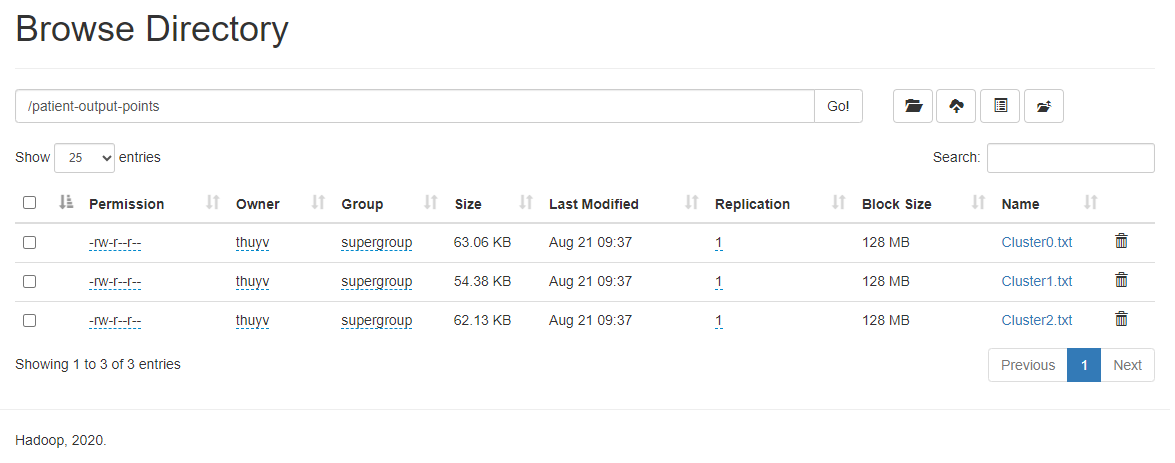
* Kết quả in ra file *result.txt* 3 tâm cụm cuối cùng và dữ liệu của 3 tâm cụm được lưu vào trong MongoDB



Hình 3.12 Kết quả in ra file 3 tâm cụm



Hình 3.13 Dữ liệu 3 tâm cụm được lưu vào MongoDB

* In ra 3 file kết quả, mỗi file chứa dữ liệu các điểm thuộc mỗi tâm cụm

Hình 3.14 Dữ liệu các điểm thuộc mỗi tâm cụm

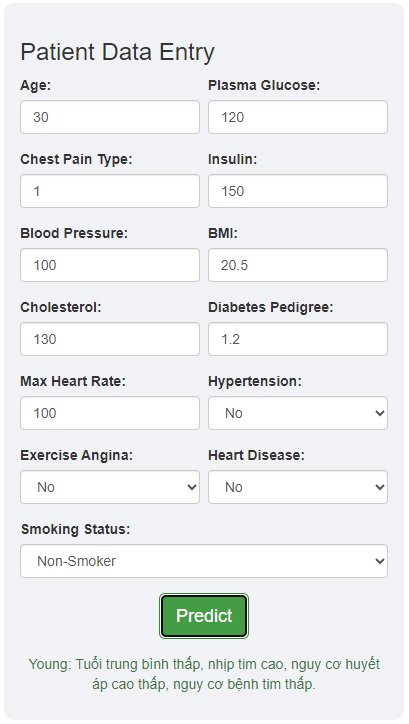
### 3.2.3. Demo ứng dụng dự đoán nhóm sức khỏe của bệnh nhân

Từ dữ liệu của 3 tâm cụm đã tìm thấy, có thể nhận xét những đặc điểm của mỗi cụm như sau:

* Cụm 0: nhóm ‘Old Person’, những bệnh nhân có đặc điểm như tuổi trung bình cao, huyết áp thấp, nhịp tim cao, nguy cơ tiểu đường thấp.
* Cụm 1: nhóm ‘Middle Age’, những bệnh nhân có đặc điểm như tuổi trung bình, nhịp tim trung bình, huyết áp tăng, nguy cơ tiểu đường cao.
* Cụm 2: nhóm ‘Young’, những bệnh nhân có đặc điểm là tuổi trung bình thấp, nhịp tim cao, nguy cơ tăng huyết áp thấp, nguy cơ bệnh tim thấp.

Từ những phân tích đặc điểm dữ liệu của các cụm trên, xây dựng ứng dụng dự đoán nhóm sức khỏe của bệnh nhân:

* Khởi chạy <localhost:3000>
* Nhập dữ liệu của một bệnh nhân và ấn nút ‘Predict’
* Dữ liệu sau đó sẽ được chuẩn hóa theo đúng dạng đã sử dụng để phân cụm
* Sau đó thực hiện truy vấn đến 3 tâm cụm trong MongoDB rồi thực hiện tính toán khoảng cách từ điểm dữ liệu nhập vào tới 3 tâm cụm
* Tìm ra tâm cụm gần nhất và hiển thị những đặc trưng của tâm cụm đó lên màn hình



Hình 3.15 Ứng dụng dự đoán nhóm sức khỏe của bệnh nhân

# CHƯƠNG 4: KẾT LUẬN

Ưu điểm của phân cụm K-means bằng Hadoop MapReduce:

* Xử lý dữ liệu lớn: MapReduce cho phép xử lý dữ liệu lớn một cách hiệu quả thông qua việc phân tách và phân phối công việc trên nhiều máy.
* Mở rộng dễ dàng: Khả năng mở rộng với việc thêm máy tính vào hệ thống, giúp tăng khả năng xử lý và giảm thời gian thực hiện.
* Tiết kiệm thời gian: MapReduce tự động quản lý các phần của quá trình tính toán và sắp xếp, giúp tiết kiệm thời gian và nguồn lực.

Nhược điểm của phân cụm K-means bằng MapReduce:

* Overhead giao tiếp: Việc trao đổi dữ liệu giữa các nút trong hệ thống MapReduce có thể tạo ra overhead lớn, đặc biệt đối với các tập dữ liệu nhỏ.
* Phụ thuộc vào số vòng lặp: Phương pháp K-Means yêu cầu nhiều vòng lặp để hội tụ. Các vòng lặp này có thể tạo ra nhiều thao tác MapReduce, dẫn đến thời gian xử lý dài.

Qua bài tập trên, nhóm em đã đạt được một số kết quả như sau:

* Hiểu tổng quan về Big Data, Apache Hadoop và mô hình MapReduce
* Hiểu chi tiết về thuật toán K-means
* Triển khai được ý tưởng và giải pháp MapReduce hóa cho thuật toán K-means
* Xây dựng chương trình demo thuật toán K-means thành công
* Xây dựng ứng dụng dự đoán nhóm sức khỏe của bệnh nhân
* Hiểu về cách thức các ứng dụng giao tiếp với nhau thông qua API
* Hiểu về cách lưu trữ dữ liệu trong MongoDB

TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | Big Data Wikipedia: [Dữ liệu lớn – Wikipedia tiếng Việt](https://vi.wikipedia.org/wiki/D%E1%BB%AF_li%E1%BB%87u_l%E1%BB%9Bn) |
| [2] | Apache Hadoop: [Giới thiệu về framework mã nguồn mở Apache Hadoop (viblo.asia)](https://viblo.asia/p/gioi-thieu-ve-framework-ma-nguon-mo-apache-hadoop-1ZnbRlNlG2Xo) |
| [3] | Một số thuật toán xử lý dữ liệu lớn - TS. Nguyễn Tu Trung, TS. Trần Mạnh Tuấn - Trường ĐH Thủy Lợi |
| [4] | Dữ liệu bệnh nhân: [Patient Dataset for Clustering (Raw Data) | Kaggle](https://www.kaggle.com/datasets/arjunnsharma/patient-dataset-for-clustering-raw-data) |