**TRƯỜNG ĐẠI HỌC THỦY LỢI**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO BÀI TẬP LỚN MÔN HỌC**

**PHÂN TÍCH DỮ LIỆU LỚN**

**Đề tài:**

**ỨNG DỤNG MÔ HÌNH MAPREDUCE CHO THUẬT TOÁN NAIVE BAYES**

***Nhóm 6:***

**Các sinh viên thực hiện : ĐỖ THU THẢO - 1851161594**

**TRƯƠNG DANH TÙNG - 1851160008**

**NGUYỄN ĐỨC LONG - 1851161775**

**ĐOÀN HẢI LONG - 1851161680**

**Giảng viên phụ trách môn học : NGUYỄN TU TRUNG**

**Ngành : CÔNG NGHỆ THÔNG TIN**

**Lớp : 60HT**

**LỜI MỞ ĐẦU**

Hiện nay, với sự phát triển của công nghệ, lượng dữ liệu lưu trữ càng ngày càng lớn. Điều này đặt ra nhiều thách thức. Để xử lý lượng dữ liệu khổng lồ đó, rất nhiều công nghệ đã ra đời. Trong đó phải kể đến công nghệ tính toán phân tán. Ý tưởng chính của việc tính toán phân tán là chia bài toán thành những bài toán con và giải quyết trên các máy riêng biệt nhau được kết nối trong một cluster. Chúng ta có thể thấy sự thành công của các công ty như Google, Facebook trong thời đại bùng nổ về công nghệ hiện nay. Đằng sau sự thành công đó có sự đóng góp không nhỏ của một mô hình lập trình được đưa ra bởi Google – đó là mô hình lập trình phân tán MapReduce.

# **CHƯƠNG I. MÔ HÌNH MAPREDUCE**

## **1.1. Mô hình MapReduce.**

## **1.1.1. Tổng quan về mô hình toán MapReduce**.

MapReduce là một “mô hình lập trình” (programming model), lần đầu được giới thiệu trong bài báo của Jeffrey Dean và Sanjay Ghemawat ở hội nghị OSDI 2004 [thêm trích dẫn vào đây]. MapReduce chỉ là một mô hình hay là một khung (framework) để người lập trình xây dựng chương trình. Để thực hiện chƣơng trình theo mô hình MapReduce thì cần có một nền tảng hay một hiện thực (implementation) cụ thể. Google có một nền tảng của MapReduce bằng C++. Apache có Hadoop, một implementation mã nguồn mở khác trên Java (người dùng dùng Hadoop qua một Java interface).

MapReduce là quy trình giúp xử lý tập hợp dữ liệu lớn đặt tại các máy tính phân tán, có thể xử lý được dữ liệu không cấu trúc (dữ liệu lưu trữ dạng tệp tin hệ thống) và dữ liệu cấu trúc (dữ liệu quan hệ 2 chiều). Trong MapReduce, các máy tính chứa dữ liệu đơn lẻ được gọi là các nút (node).

Đơn giản thì MapReduce là một cách tiếp cận chia để trị, tức là chia vấn đề lớn thành các vấn đề nhỏ, xử lý song song từng vấn đề nhỏ và tổng hợp kết quả lại. Để thực hiện được công việc, mô hình MapReduce chia công việc thành hai phần chính là map và reduce. Trong đó, hàm map thƣờng rút trích thông tin cần thiết các từng phần tử, qua một bước trung gian để trộn và sắp xếp lại kết quả, sau đó tại reduce thì tổng hợp kết quả trung gian và tính ra kết quả cuối cùng.

Ví dụ, trong bài đếm từ, công việc chủ yếu là đếm số lần xuất hiện của các từ trong văn bản. Công việc nhƣ vậy thì có thể phân chia thành hai phần như sau:

\*Mapper :

• Đầu vào : Một dòng của văn bản

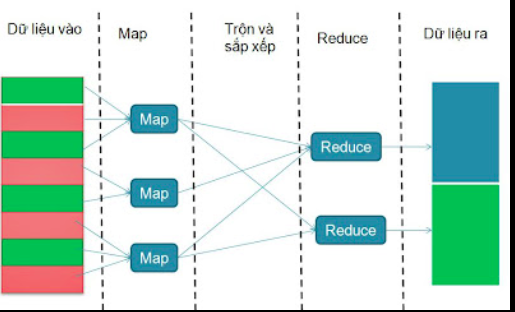
• Đầu ra : key : từ, value : 1

\*Reduce :

• Đầu vào : key : từ, values : tập hợp các giá trị đếm được của mỗi từ

• Đầu ra : key : từ, value : tổng

Dữ liệu là văn bản, được chia thành nhiều dòng khác nhau, lưu trữ ở nhiều nơi khác nhau. Hàm map chạy song song tạo ra các giá trị trung gian khác nhau từ tập dữ liệu khác nhau. Hàm reduce cũng chạy song song, mỗi reducer xử lý một tập khóa khác nhau. Tất cả các giá trị được xử lý một cách độc lập. Và giai đoạn reduce chỉ bắt đầu khi giai đoạn map kết thúc.



Hình 1. Mô hình làm việc của một công việc MapReduce

MapReduce định nghĩa dữ liệu (cấu trúc và không cấu trúc) dưới dạng cặp khóa/giá trị ***(key/value).*** Ví dụ, key có thể là tên của tập tin (file) và ***value*** nội dung của tập tin, hoặc key là địa chỉ URL và ***value*** là nội dung của URL,… Việc định nghĩa dữ liệu thành cặp ***key/value*** này linh hoạt hơn các bảng dữ liệu quan hệ 2 chiều truyền thống (quan hệ cha – con hay còn gọi là khóa chính – khóa phụ).

Để xử lý khối dữ liệu này, lập trình viên viết hai hàm ***map và reduce***. Hàm ***ma***p có input là một cặp (k1, v1) và output là một danh sách các cặp (k2, v2). Chú ý rằng các input và output keys và values có thể thuộc về các kiểu dữ liệu khác nhau. Như vập hàm ***map*** có thể được viết một cách hình thức như sau:

***map (k1, v1) -> list (k2, v2)***

MapReduce sẽ áp dụng hàm ***map*** (mà ngƣời dùng viết) vào từng cặp (***key, value***) trong khối dữ liệu vào, chạy rất nhiều phiên bản của map song song với nhau trên các máy tính của cluster. Sau giai đoạn này thì chúng ta có một tập hợp rất nhiều cặp (***key, value***) thuộc kiểu (k2, v2) gọi là các cặp (***key, value***) trung gian. MapReduce cũng sẽ nhóm các cặp này theo từng key, như vậy các cặp (***key, value***) trung gian có cùng k2 sẽ nằm cùng một nhóm trung gian.

Giai đoạn hai MapReduce sẽ áp dụng hàm reduce (mà người dùng viết) vào từng nhóm trung gian. Một cách hình thức, hàm này có thể mô tả như sau:

***reduce (k2, list (v2)) -> list (v3)***

Trong đó k2 là key chung của nhóm trung gian, list(v2) là tập các values trong nhóm, và list(v3) là một danh sách các giá trị trả về của reduce thuộc kiểu dữ liệu v3. Do reduce được áp dụng vào nhiều nhóm trung gian độc lập nhau, chúng lại một lần nữa có thể đƣợc chạy song song với nhau.

## 

## **1.1.2. Ưu điểm của mô hình MapReduce**

MapReduce được xây dựng từ mô hình lập trình hàm và lập trình song song. Nó giúp cải thiện tốc độ tính toán trên tập dữ liệu lớn bằng cách tăng tốc độ đọc ghi và xử lý dữ liệu.

Mô hình MapReduce có thể áp dụng hiệu quả có nhiều bán toán. Mô hình này làm ẩn đi các chi tiết cài đặt và quản lý như:

* Quản lý tiến trình song song và phân tán
* Quản lý, sắp xếp lịch trình truy xuất I/O
* Theo dõi trạng thái dữ liệu
* Quản lý số lượng lớn dữ liệu có quan hệ phụ thuộc nhau
* Xử lý lỗi
* Cung cấp mô hình lập trình đơn giản

Các ứng dụng viết bằng MapReduce có tính linh hoạt, khả năng mở rộng tốt.

# **CHƯƠNG II. THUẬT TOÁN NAIVE BAYES**

## **2.1. Giới thiệu thuật toán Naive Bayes cơ bản.**

Naïve Bayes là phương pháp phân loại dựa vào xác suất được sử dụng rộng rãi trong lĩnh vực học máy, đƣợc sử dụng lần đầu tiên trong lĩnh vực phân loại bởi Maron vào năm 1960, sau đó trở nên phổ biến trong nhiều lĩnh vực như các công cụ tìm kiếm, các bộ lọc email 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 là ở chỗ giả định rằng sự xuất hiện của tất cả các từ trong văn bản đều độc lập với nhau. Như thế Naïve bayes không khai thác sự phụ thuộc của nhiều từ vào trong một chủ đề cụ thể.

Đây là thuật toán đƣợc xem là đơn giản nhất trong các phƣơng pháp. Bộ phân lớp Bayes có thể dự báo các xác suất là thành viên của lớp, chẳng hạn xác suất mẫu cho trước thuộc về một lớp xác định. Chúng giả định các thuộc tính là độc lập nhau (độc lập điều kiện lớp).

Thuật toán Naïve Bayes dựa trên định lý Bayes đƣợc phát biểu như sau:



Trong đó:

* Y đại diện một giả thuyết, giả thuyết này được suy luận khi có được chứng cứ mới X.
* P(X): xác suất X xảy ra (Xác suất biên duyên của X).
* P(Y): xác suất Y xảy ra (Điều kiện tiên nghiệm của Y).
* P(X|Y): xác suất X xảy ra khi Y xảy ra (xác suất có điều kiện, khả năng của X khi Y đúng).
* P(Y|X) : xác suất hậu nghiệm của Y nếu biết X

Áp dụng trong bài toán phân loại, các dữ kiện cần có :

* D: tập dữ liệu huấn luyện đã được vector hoá dƣới dạng ⃗ (x1,x2,…xn)
* Ci : tập các tài liệu của D thuộc lớp Ci với i={1,2,3,…}
* Các thuộc tính x1,x2,…xn độc lập xác suất đôi một với nhau.

## **2.2. Phân loại thuật toán Naive Bayes**

Naive Bayes là một thuật toán phân loại cho các vấn đề phân loại nhị phân (hai lớp) và đa lớp. Kỹ thuật này dễ hiểu nhất khi được mô tả bằng các giá trị đầu vào nhị phân hoặc phân loại.

Thuật toán Naive Bayes tính xác suất cho các yếu tố, sau đó chọn kết quả với xác suất cao nhất.

Tuy nhiên, ta cần lưu ý giả định của thuật toán Naive Bayes là các yếu tố đầu vào được cho là độc lập với nhau.

Thuật toán này là một thuật toán mạnh mẽ trong các bài toán:

* **Dự đoán với thời gian thực**
* **Phân loại Text/ Lọc thư rác**
* **Hệ thống Recommendation**

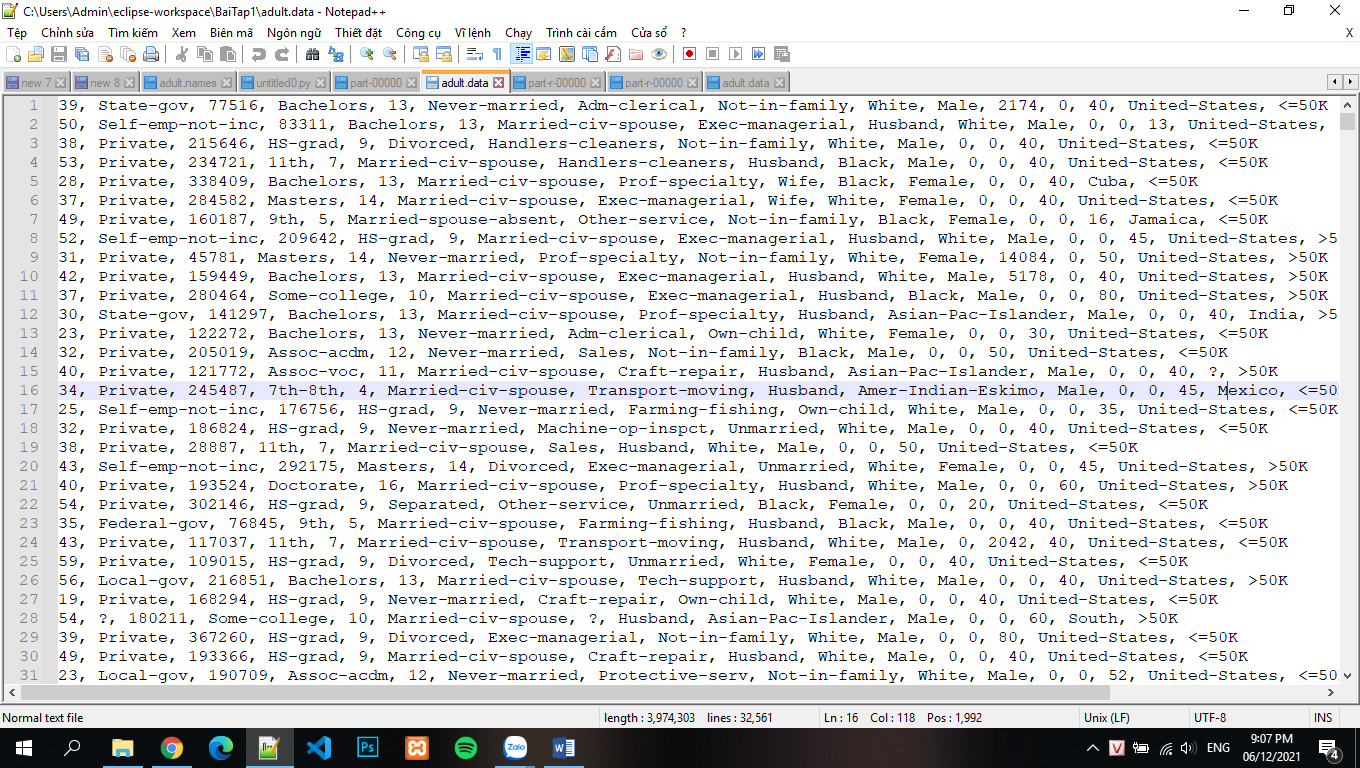
Về mặt toán học, ta có thể viết như sau:

Nếu ta có một Class E và các điểm dữ liệu x1,x2,x3, etc.

Đầu tiên ta sẽ phải tính xác suất P(x1| E) , P(x2 | E) … (xác suất của x1 thuộc class E xảy ra) và sau đó ta sẽ chọn class có xác suất xảy ra x1 cao nhất.

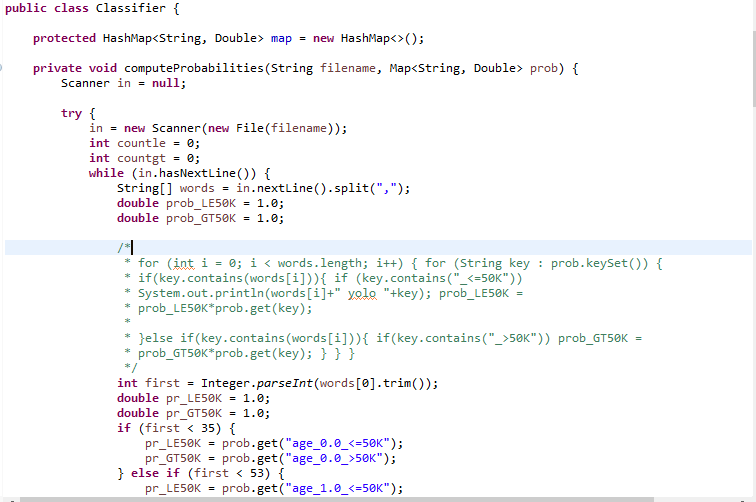
# **Chương III.Thử nghiệm thuật toán**

## **3.1.Đầu vào của thuật toán**

****

## **3.2.Source code**

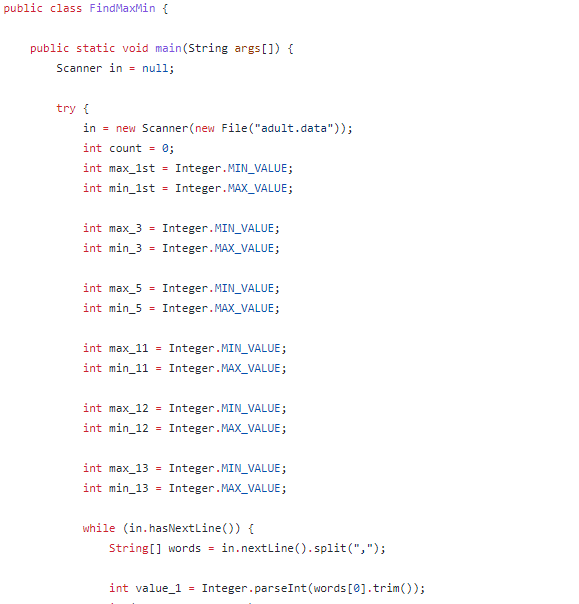
-File Classifer.java

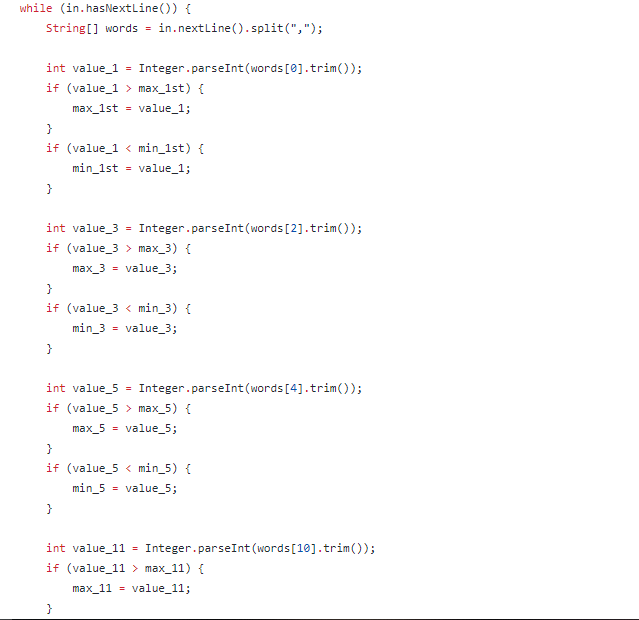


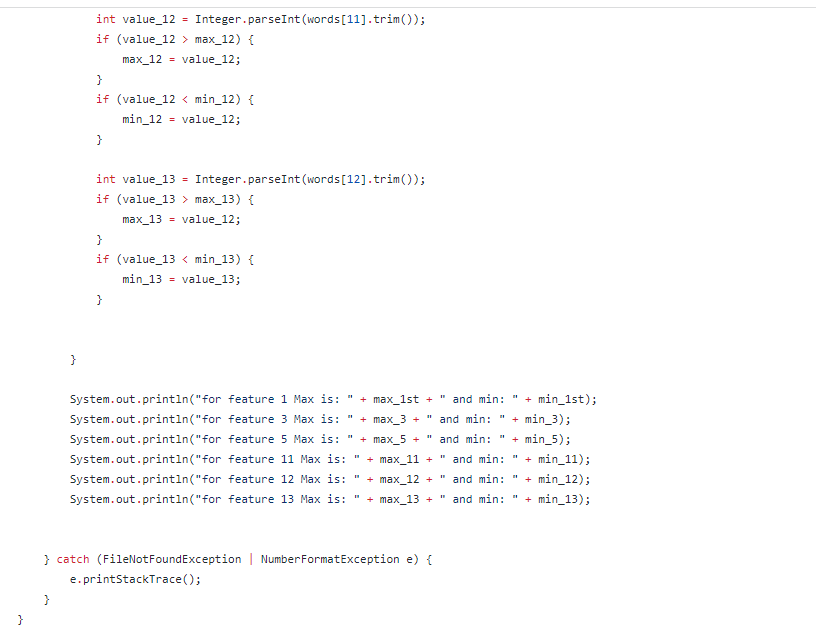
-File Evaluate.java



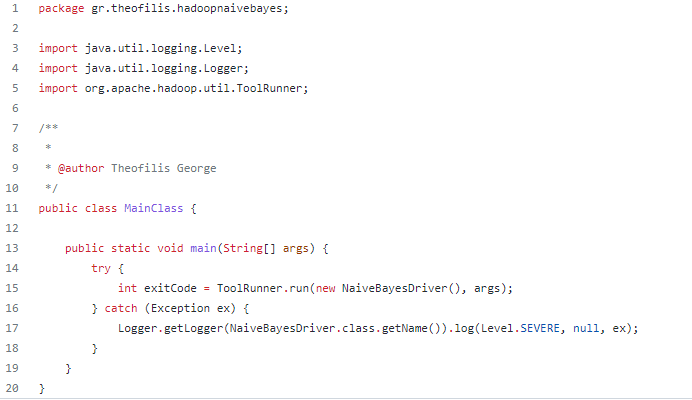
-File FindMaxMin.java



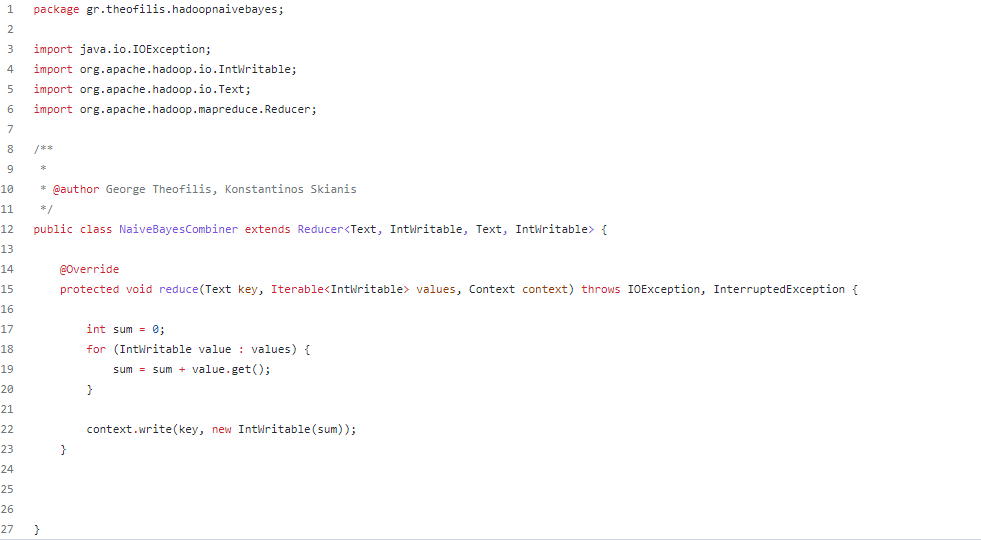




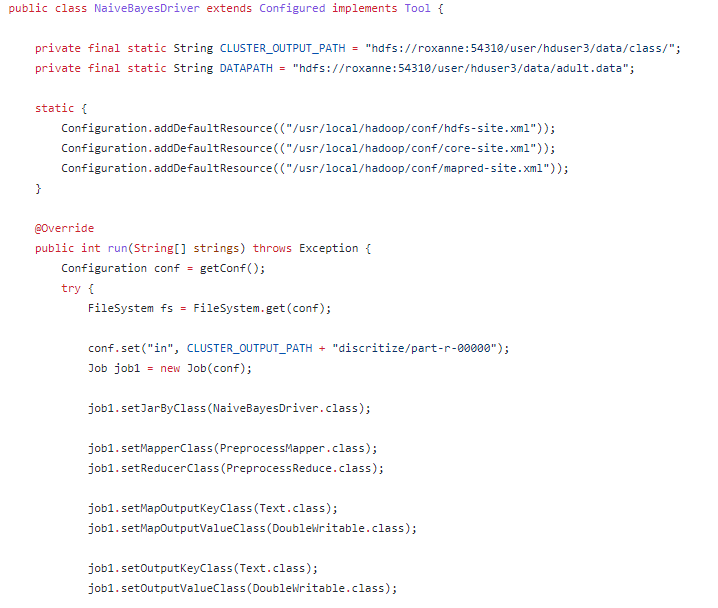
-File MainClass.java



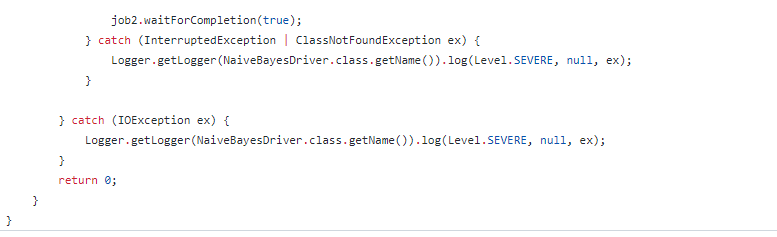
-File NaiveBayesCombiner.java



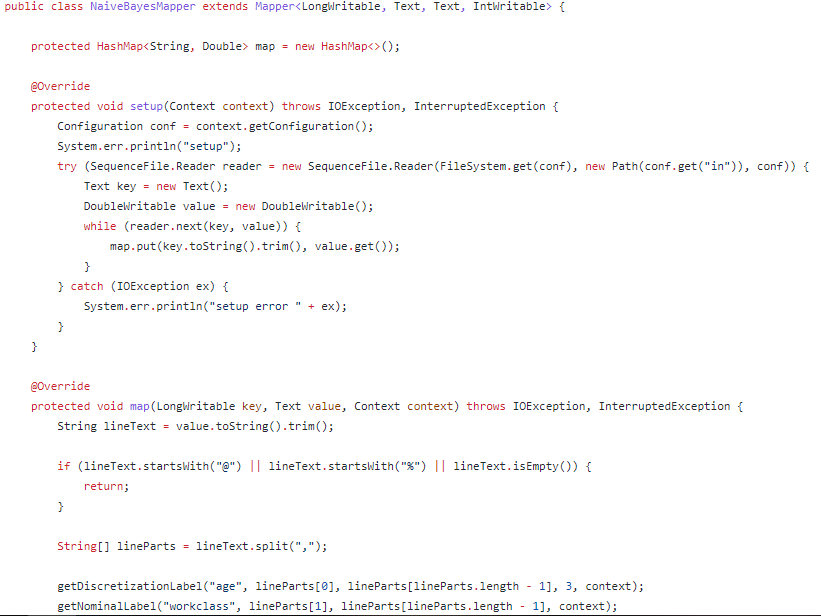
-File NaiveBayesDriver.java



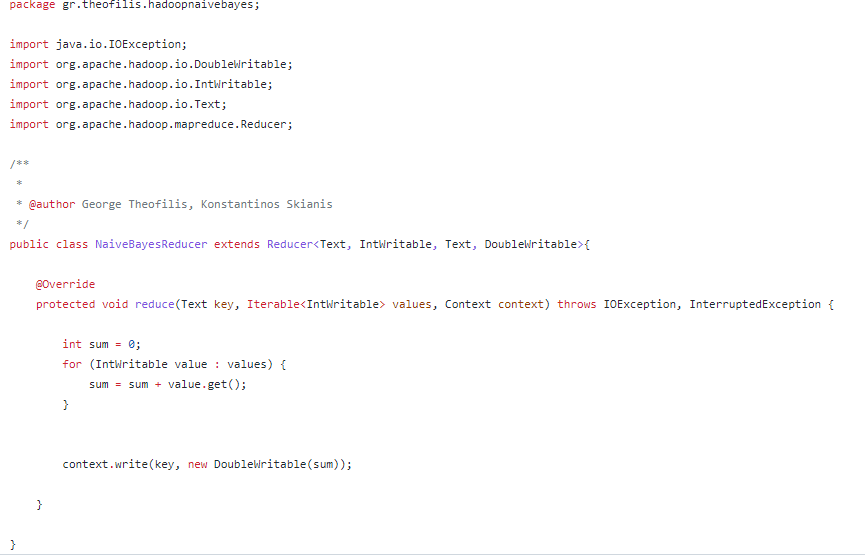




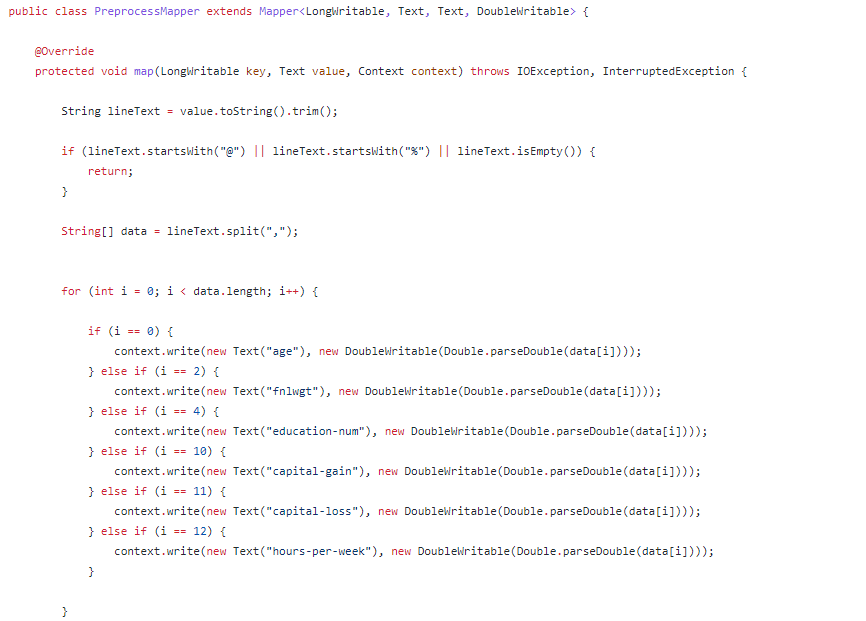
-File NaiveBayesMapper.java



-File NaiveBayesReducer.java



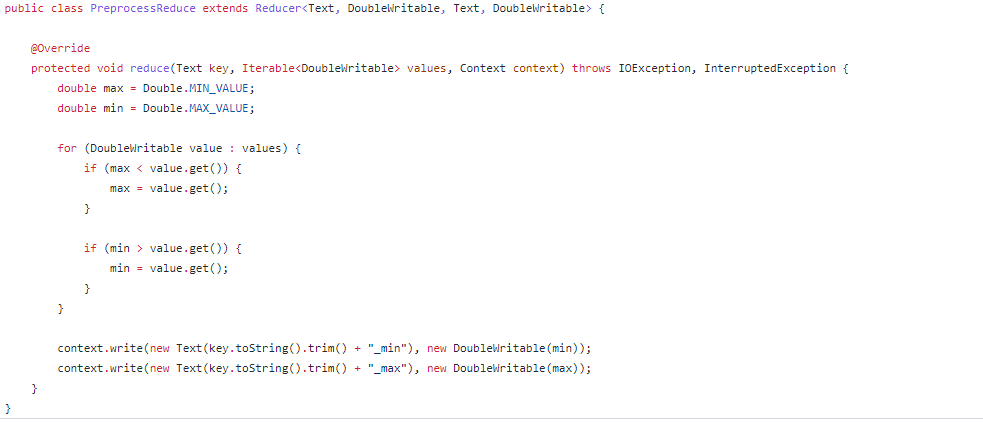
-File PreprocessMapper.java



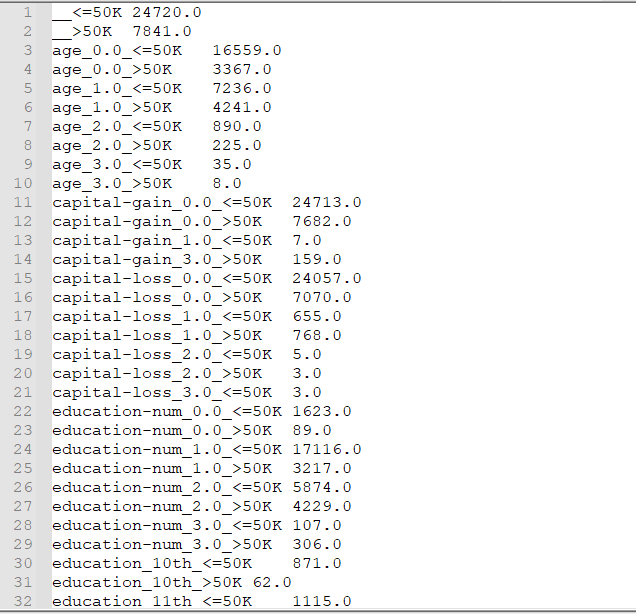
# 

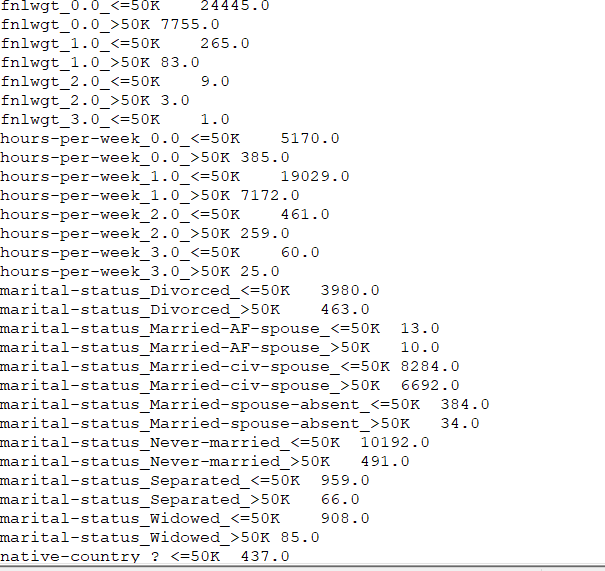
-File PreprocessReduce.java

# 



## **3.3 Kết quả thu được**





# **IV. Kết luận.**

Qua đề tài **ỨNG DỤNG MÔ HÌNH MAPREDUCE CHO THUẬT TOÁN NAIVE BAYES** nhóm chúng em đã nỗ lực để thực hiện và tối ưu hóa mô hình mapreduce.Vì bài toán vẫn còn nhiều hạn chế nên vẫn mong có sự góp ý của thầy giáo và các bạn. Qua đó nhóm cũng xin cảm ơn sự chỉ dẫn nhiệt tình của thầy Nguyễn Tu Trung trong quá trình học tập và thực hiện đề tài, sự góp ý của thầy sẽ góp phần làm đề tài nhóm chúng em trở nên hoàn thiện hơn. Nhóm xin chân thành cảm ơn!

Tài liệu tham khảo:

-File dữ liệu : <https://archive-beta.ics.uci.edu/ml/datasets/adult>

-Source code tham khảo : https://github.com/theofilis/hadoop-naive-bayes/tree/master/src/main/java/gr/theofilis/hadoopnaivebayes