00TRƯỜNG ĐẠI HỌC PHENIKAA
KHOA CÔNG NGHỆ THÔNG TIN

PHENIKAA
UNIVERSITY

Báo cáo bài tập lớn Tích hợp và phân tích dữ liệu lớn

Đề tài : So sánh việc chạy Spark và không Spark

GVHD: TS. Trịnh Thành
SVTH: Phạm Bá Khương
Trịnh Gia Khiêm
Lê Hoàng Ngọc Tú

MỤC LỤC

Chương I : Giới thiệu tổng quan

1. Toàn cảnh về Dữ liệu lớn

Trong thời đại ngày nay, mọi kinh nghiệm, hành động và cảm xúc của con người đều được coi là dữ liệu. Chúng ta đang sống trong môi trường Dữ liệu lớn(Big Data/Large Scale Data), tất cả những hoạt động, tư duy và sự kiện trong đời sống đều trở thành dữ liệu. Dữ liệu trở thành mạch máu, duy trì hoạt động của toàn bộ các lĩnh vực và ngày càng phát triển theo cấp số nhân. So với trước kia, lượng dữ liệu ngày nay mà chúng ta sở hữu là vô cùng lớn, phong phú và vấn đề lưu trữ, khai thác lượng dữ liệu này đang rất được quan tâm trong những năm gần đây. Nhất là trong lĩnh vực khoa học máy tính.

Ta có thể thấy rằng, Dữ liệu lớn đang được áp dụng ngày càng rộng rãi trong hầu hết mọi lĩnh vực Khoa học-Kinh tế hiện nay. Như hệ thống Ngân hàng sử dụng các kỹ thuật phân cụm giúp phân tích các địa điểm chi nhánh tập trung nhiều nhu cầu của khách hàng tiềm năng, dự đoán lượng tiền mặt cần thiết sẵn sàng cung ứng ở một chi nhánh tại thời điểm cụ thể hàng năm. Hay trong thương mại điện tử, Thông qua dữ liệu lớn và Machine Learning, ta có thể tạo ra dữ liệu và phán đoán yêu cầu của khách hàng ngay cả trước khi bắt đầu giao dịch. Nhà quản lý trang thương mại điện tử có thể xác định các sản phẩm được xem nhiều nhất và tối ưu thời gian hiển thị của các trang sản phẩm này. Hoặc đánh giá hành vi của khách hàng và đề xuất các sản phẩm tương tự. Điều này làm tăng khả năng bán hàng, từ đó tạo ra doanh thu cao hơn.

Một thách thức đặt ra, đó là làm sao có thể khai thác triệt để nguồn thông tin khổng lồ này, nói đúng hơn là: “Làm thế nào để thu thập được nội dung của dữ liệu một cách nhanh chóng và hiệu quả nhất?”. Việc nghiên cứu toàn bộ dữ liệu tốn rất nhiều thời gian, chi phí và công sức, do đó, cần có những phương pháp có thể giải quyết vấn đề thông qua việc khai thác các tính chất đặc trưng của dữ liệu.

Vấn đề khi dữ liệu là quán lớn, việc làm thế nào để phân tích và thực

thi nó trên máy tính là các thách thức hiện nay

2. Big Data/ Large Scale Data.

Mặc dù định nghĩa về Dữ liệu lớn rất phổ biến, nhưng nó cũng tương đối mơ hồ. Nhìn chung, ta có thể định nghĩa khái niệm về Dữ liệu lớn như sau: “Big Data/ Dữ liệu lớn là một thuật ngữ cho việc xử lý một tập hợp dữ liệu có kích thước rất lớn và phức tạp mà các ứng dụng xử lý dữ liệu truyền thống không xử lý được”.

Hiện nay, chúng ta luôn có thể tìm được những nguồn dữ liệu vô cùng lớn và phong phú, tuy nhiên không phải bất kỳ loại dữ liệu nào cũng có thể được coi là Big Data. Về cơ bản, Big Data có những đặc trưng nhất định, được gọi là “Đặc trưng 4V”, bao gồm:

- **(1)Volume (Dung lượng):** Số lượng dữ liệu được tạo ra và lưu trữ. Kích thước của dữ liệu là một trong những yếu tố chính xác định xem chúng có thể được coi là Dữ liệu lớn hay không.
- **(2)Variety (Đa dạng):** Dữ liệu được thu từ nhiều nguồn khác nhau và có nhiều kiểu cấu trúc khác nhau như: Dữ liệu có cấu trúc, dữ liệu bán cấu trúc và dữ liệu phi cấu trúc.
- **(3)Velocity (Vận tốc):** Dữ liệu được tạo ra với tốc độ nhanh cỡ nào để có thể đáp ứng được các nhu cầu của người sử dụng chúng. Dữ liệu tốc độ truyền đến từ các nguồn như máy móc, mạng, mạng xã hội, điện thoại di động,…Điều này xác định tiềm năng của dữ liệu thông qua tốc độ dữ liệu được tạo ra và xử lý để đáp ứng nhu cầu.
- **(4)Veracity (Tính xác thực):** Không phải thông tin nào cũng đúng hoàn toàn. Chất lượng của dữ liệu thu được có thể khác nhau rất nhiều, ảnh hưởng đến tính chính xác của chúng

Chương II: Tìm hiểu về spark

1.Khái niệm

Apache Spark là một Open Source Cluster Computing Framework (công cụ phân tích hợp nhất mã nguồn mở) được sử dụng rộng rãi trong thế giới Big Data. Được phát triển sơ khởi vào năm 2009 bởi AMPLab tại đại học California. Sau này, Spark đã được trao cho Apache Software Foundation vào năm 2013 và được phát triển cho đến nay.

Ứng dụng đa dạng của Apache Spark

Spark cung cấp một giao diện để lập trình toàn bộ các cụm với Tính song song dữ liệu ngầm và Khả năng chịu lỗi. Spark cho phép xây dựng các mô hình dự đoán nhanh chóng với việc tính toán được thực hiện trên một nhóm các máy tính, có có thể tính toán cùng lúc trên toàn bộ tập dữ liệu mà không cần phải trích xuất mẫu tính toán thử nghiệm. Tốc độ xử lý của Spark có được do việc tính toán được thực hiện cùng lúc trên nhiều máy khác nhau. Đồng thời việc tính toán được thực hiện ở bộ nhớ trong (in-memories) hay thực hiện hoàn toàn trên RAM

2.Ưu điểm

Dễ sử dụng: Spark cung cấp một mô hình lập trình đơn giản cho người dùng do Map Reduce cung cấp. Spark cung cấp các giao diện lập trình ứng dụng (API) để phát triển các ứng dụng dữ liệu lớn, đi kèm với hơn 80 toán tử xử lý dữ liệu. Các dòng mã của Spark cũng ngắn hơn các nền tảng tương tự nó, như Hadoop

Xử lý nhanh gọn: Apache spark có thể xử lý các dữ liệu có dung lượng vừa nhanh hơn hàng chục lần so với Hadoop, bản thân Spark có khả năng điều chỉnh độ trễ giúp công việc nhanh chóng hơn.

Khả năng tương thích: Apache spark có thể tích hợp với tất cả các định dạng tệp và các nguồn dữ liệu do cụm Hadoop hỗ trợ. Spark có thể xử lý hàng loạt, phân tích tương tác, xử lý luồng, học máy và tính toán đồ thị.

Khả năng chịu lỗi: Spark tự động sửa lỗi của các nút trong cụm.

Hỗ trợ ngôn ngữ: Các ngôn ngữ được hỗ trợ bao gồm Java, Python, Scala và R.

Dễ quản lý: Bạn có thể dễ dàng quản lý (submit, bắt đầu, dừng lại, xem trạng thái, sparkcontext, spark job) giao diện REST.

Apache spark tuy là một khái niệm mới nhưng tiềm năng phát triển trong tương lai là rất lớn, nhất là trong lĩnh vực IT với các công nghệ cốt lõi

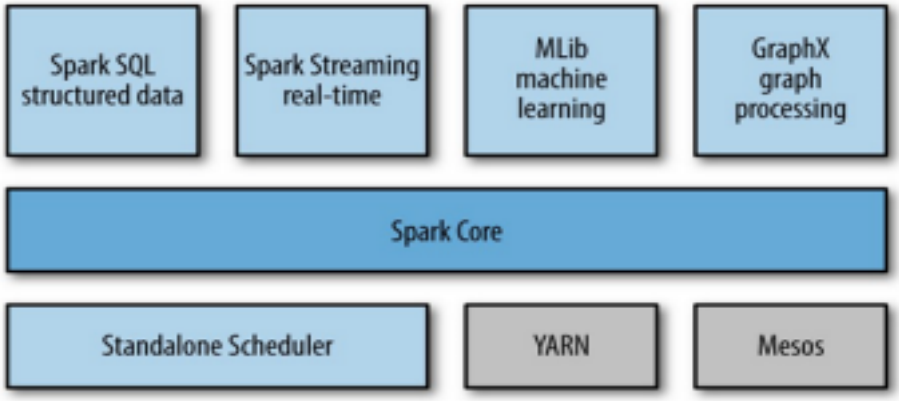
3.Nhược điểm

Spark không có hệ thống Filesytem riêng, do đó nó phụ thuộc vào một số nền tảng khác như Hadoop hoặc một hệ thống nền tảng dựa trên đám mây

Spark đòi hỏi rất nhiều RAM để chạy trong bộ nhớ, do đó chi phí của Spark khá cao

4. Thành phần của Spark

Về cơ bản, Spark được chia làm 5 thành phần: Spark Core, Spark SQL, Spark Streaming, MLlib và GraphX.



4.1: Spark Core

Có thể coi Spark Core là cốt lõi của Apache Spark Các thành phần khác của Spark khi hoạt động cần phải thông qua sự trợ giúp từ Spark Core. Công việc của Spark Core giúp thực hiện các công việc tính toán và xử lý dữ liệu bên trong bộ nhớ (In Memory Computing). Đồng thời thiết bị này có thể tham chiếu các dữ liệu được lưu trữ bên ngoài.

4.2: Spark SQL

Là một thư viện chạy trên Spark, thành phần có nhiệm vụ cung cấp giao diện SQL và thao tác với cơ sở dữ liệu có cấu trúc bằng cách cung cấp các hàm xử lý dữ liệu. Dữ liệu có cấu trúc bao gồm dữ liệu được lưu trữ trong cơ sở dữ liệu, kho dữ liệu NoSQL, Parquet, ORC, Avro, JSON, CSV hoặc bất kỳ định dạng có cấu trúc nào khác.

Một trong những lợi ích của việc sử dụng Spark SQL là nâng cao năng suất. Nó cho phép ta xử lý dữ liệu có cấu trúc với ít mã hơn so với sử dụng API lõi Spark

Lập trình viên có thể viết các ứng dụng Spark bằng nhiều ngôn ngữ khác nhau như Scala, Java, Python, R..

4.3. Spark Streaming

Là một khung xử lý luồng dữ liệu phân tán Spark Streaming phát triển các ứng dụng phân tán để xử lý các luồng dữ liệu trong thời gian gần thực. Nó không chỉ cung cấp một mô hình lập trình đơn giản (API) mà còn cho phép ứng dụng xử lý dữ liệu dòng (Stream data) tốc độ cao. Nó cũng cho phép kết hợp các luồng dữ liệu và dữ liệu lịch sử để xử lý. Spark Streaming là một thư viện chạy trên Spark. Nó mở rộng Spark để xử lý luồng dữ liệu, chịu được lỗi và thông lượng cao. Khả năng xử lý ứng dụng của Spark Streaming có thể được nâng cao bằng cách thêm nhiều nút hơn vào cụm Spark.

Ngoài ra, Spark Streaming kế thừa những tính năng và lợi ích của Spark Core, nó có thể được sử dụng cùng với các thư viện Spark khác, chẳng hạn như Spark SQL, MLlib, Spark ML và Graphx.

Spark Streaming chạy trên lõi của Spark Core.

4.4. MLlib.

Là một nền tảng học máy, Spark MLlib nhanh hơn gấp 9 lần so với phiên bản chạy trên Hadoop (theo so sánh của benchmark) nhờ kiến trúc phân tán dựa trên bộ nhớ. MLlib Cung cấp rất nhiều thuật toán của học máy như Classification (phân lớp), Regression (hồi quy), Clustering phân cụm), Neighborhood-Based Collaborative Filtering (Lọc cộng tác),...

4.5. GraphX

Là thư viện để xử lý đồ thị. Nó cung cấp các API và được sử dụng để diễn tả tất cả các tính toán có trong đô thị thông qua Pregel Api. Theo thống

kê, tại các thư viện mà Spark cung cấp thì người dùng Spark SQL là 69%, người sử dụng Dataframes là khoảng 62% và 58% người sử dụng Spark Streaming và MLlib + GraphX.

Chương III: Cài đặt môi trường và kết quả thực nghiệm

1.Chuẩn bị

- Đề xuất cài đặt môi trường trên 2 máy ảo chạy hệ điều hành ubuntu 18.04 với mỗi máy có cấu hình 2 core CPU và 4GB ram. Cả hai máy đều được sử dụng làm máy worker trong đó sẽ có một máy vừa làm máy worker vừa làm máy master (cả hai máy phải được kết nối mạng LAN).

- Địa chỉ IP của 2 máy:
  + Máy 1 vừa làm máy master vừa làm máy worker: 192.168.3.118
  + Máy 2 làm máy worker: 192.168.3.119

- Các file cần chuẩn bị trên cả 2 máy:
  + spark-3.3.2-bin-hadoop3.tgz
  + Anaconda3-2021.05-Linux-x86_64.sh
  + data2K.csv (3.5 MB)
  + data500K_10c.csv (887 MB)
  + data1M_2.csv (1.7 GB)
  + data1M_10c.csv (1.7 GB)
  + data2M_10c.csv (3.5 GB)
  + test.py

```
def run_with_spark():



    conf = SparkConf()
    conf.setMaster('spark://192.168.3.118:7077')
#    conf.setMaster('local')
    conf.setAppName('spark-basic')
    conf.set('spark.executor.memory','2g')
    conf.set("spark.driver.maxResultSize", "2g")
    sc = SparkContext(conf=conf)



    df= sc.textFile("/home/khiem/Desktop/testcode/data2K.csv").map(lambda line:
line.split(","))
    dataset  = df.map(lambda x: LabeledPoint(x[0], x[1:]))
    (trainingData, testData) = dataset.randomSplit([0.7, 0.3])

    # decison treee
    start = datetime.now()
    model = DecisionTree.trainClassifier(trainingData,  numClasses=20,
categoricalFeaturesInfo={}, impurity='gini', maxDepth=8, maxBins=32)
    end = datetime.now() -start

    # Evaluate model on test instances and compute test error
    predictions = model.predict(testData.map(lambda x: x.features))
    labelsAndPredictions = testData.map(lambda lp: lp.label).zip(predictions)

    acc =0
    acc = (labelsAndPredictions.filter( lambda lp: lp[0] == lp[1]).count()) /
float(testData.count())
    print("\n")
    print("Decison Tree")
    print ('time : ', end)
    print('Accuracy= ' + str(acc))
```

2.Các bước thực hiện

- Bước 1: Cài đặt spark cho cả 2 máy
  - + Cài đặt java : sudo apt-get install default-jdk
  - + Giải nén file spark : tar -xvf spark-3.3.2-bin-hadoop3.tgz
  - + Thêm biến môi trường vào file bashrc: nano ~/.bashrc

```
SPARK_HOME=/opt/spark/spark-3.3.2-bin-hadoop3
export PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin
```

  - + Áp dụng biến môi trường vừa thêm: source ~/.bashrc
  - + Kiểm tra: spark-shell

- Bước 2: Cài đặt anaconda, python trên cả 2 máy:
  + Cài python: sudo apt-get install python3
  + Cài đặt pip: sudo apt-get install python3-pip
  + Cài đặt file anaconda: bash spark-3.3.2-bin-hadoop3.tgz

- Bước 3: Cài đặt ssh trên cả 2 máy và tạo key trên master
  + Cài đặt ssh: sudo apt-get install openssh-server
  + Tạo key trên máy master: ssh-keygen
  + Di chuyển vào thư mục ssh để copy file id_rsa.pub sang máy worker: ssh-copy-id -i id_rsa.pub khiem@192.168.3.119
  + Vì máy 1 vừa làm máy master vừa làm máy worker nên tạo luôn một file authorized_keys tại thư mục ssh của máy 1: cat id_rsa.pub >> authorized_keys

- Bước 4: Cấu hình cluster ở cả 2 máy:
  + Di chuyển vào thư mục /spark/conf tạo một file tên là spark-env.sh sau đấy mở file đó lên rôi thêm địa chỉ IP của máy master:

```
export SPARK_MASTER_HOST=192.168.3.118
export SPARK_MASTER_PORT=7077
```

  + Tạo file worker rồi thêm địa chỉ IP của 2 máy worker:

```
192.168.3.118
192.168.3.119
```

- Bước 5 Khởi động cluster:
  + Trên máy master khởi động máy master bằng câu lệnh start-master.sh
  + Khởi động worker trên cả 2 máy :start-slave.sh spark://192.168.3.118:7077
  + Submit file code lên cluster: /opt/spark/spark-3.3.2-bin-hadoop3/bin/spark-submit test.py --master spark://192.186.3.118:7077

3.Kết quả chạy code

- Để ổn định và trực quan nhất thì em sẽ chỉ so sánh accuracy và thời gian chạy ở các lần chạy code bằng thuật toán decision tree.

- Chạy thường trên máy tính hệ điều hành windows có cấu hình 4 core cpu và 16gb ram:
  + data2K.csv: Accuracy: 0.84
    Thời gian chạy: 0.01s

  + data500K_10c.csv: Accuracy: 0.92
    Thời gian chạy: 4p 27s

  + data1M_2.csv: Accuracy: 0.97
    Thời gian chạy: 11p 21s

  + data1M_10c.csv: Accuracy: 0.93
    Thời gian chạy: 16p 1s

  + data2M_10c.csv: Accuracy: 0.94
    Thời gian chạy:41p 8s

- Chạy bằng spark trên máy tính hệ điều hành windows có cấu hình 4 core cpu và 16gb ram:
  + data2K.csv: Accuracy: 0.84
    Thời gian chạy: 24s

  + data500K_10c.csv: Accuracy: 0.83
    Thời gian chạy: 1p 34s

  + data1M_2.csv: Accuracy: 0.96
    Thời gian chạy: 2p 19s

  + data1M_10c.csv: Accuracy: 0.83
    Thời gian chạy: 3p 3s

  + data2M_10c.csv: Accuracy: 0.83

Thời gian chạy:5p 0s

- Chạy spark đơn bằng máy ảo hệ điều hành ubuntu 18.04 với cấu hình 2 core cpu và 4gb ram
  + data2K.csv: Accuracy: 0.82
    Thời gian chạy: 8.07s

  + data500K_10c.csv: Accuracy: 0.82
    Thời gian chạy: 1p 12s

  + data1M_2.csv(bị tràn ram): Accuracy: 0.96
    Thời gian chạy: 2p 56s

  + data1M_10c.csv(bị tràn ram): Accuracy: 0.84
    Thời gian chạy: 3p 2s

  + data2M_10c.csv: Không chạy nổi

- Chạy spark cluster bằng 2 máy ảo hệ điều hành ubuntu 18.04 với cấu hình mỗi máy là 2 core cpu và 4gb ram
  + data2K.csv: Accuracy: 0.84
    Thời gian chạy: 16s

  + data500K_10c.csv: Accuracy: 0.83
    Thời gian chạy: 1p 27s

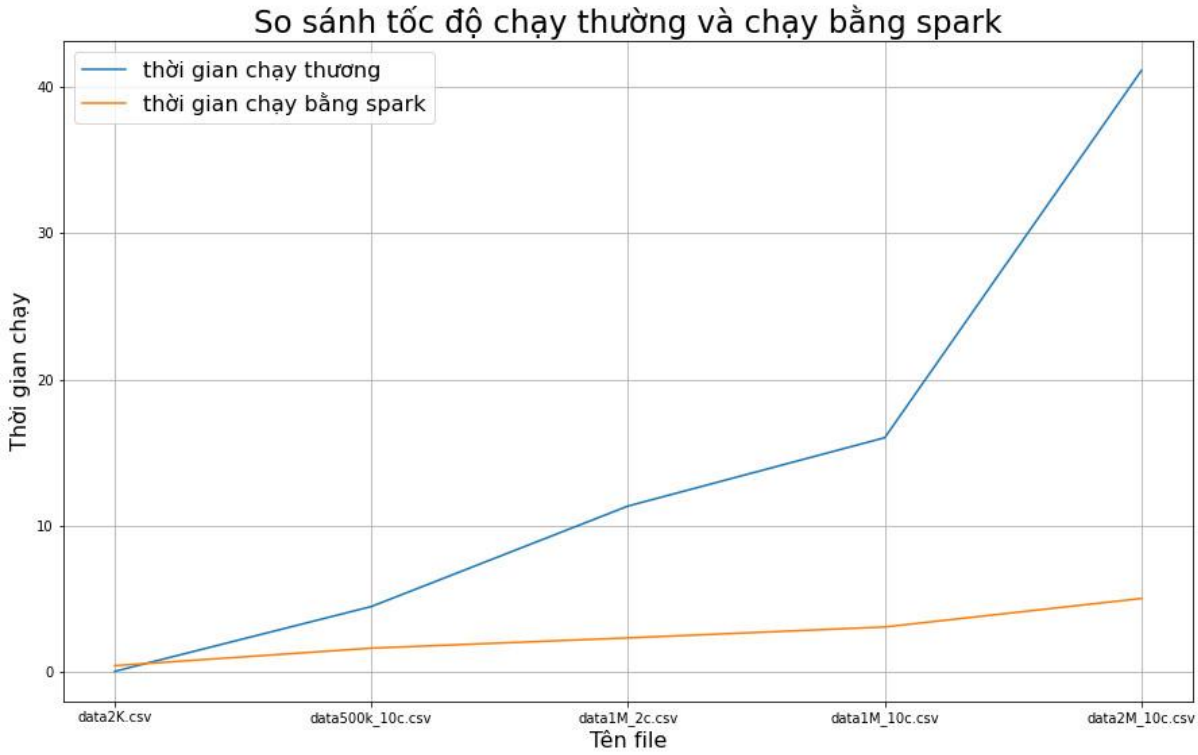
  + data1M_2.csv: Accuracy: 0.96
    Thời gian chạy: 1p 43s

  + data1M_10c.csv: Accuracy: 0.84
    Thời gian chạy: 1p 58s

  + data2M_10c.csv: Accuracy: 0.84
    Thời gian chạy:4p 38s

Chương IV: So sánh

1.So sánh giữa chạy có spark và không có spark



Hình 4.1 : So sánh tốc độ chạy thường và chạy bằng spark

1.1. Hiệu suất:

- **Chạy thường**: Việc sử dụng những thuật toán như Decision Tree , Random Forest.. cho việc phân tích và xử lí dữ liệu lớn có thể cho kết quả tốt tùy thuộc vào lượng dữ liệu hợp với những thuật toán đó. Nhưng rất mất nhiều thời gian cho xử lí dữ liệu
- **Spark:** Spark tận dụng tối đa khả năng tính toán phân tán. Với việc phân chia công việc và dữ liệu trên nhiều máy tính, Spark cung cấp hiệu suất cao hơn so với chạy thường, với tốc độ xử lí nhanh hơn nhưng bên cạnh đó đôi kết quả lại không như mong đợi thậm trí thấp hơn với việc dùng những thuật toán trên.
- Hình 4.1: Thời gian xử lí dữ liệu giữa Spark và những thuật toán khác cho thấy tốc độ xử lí rất nhanh chóng

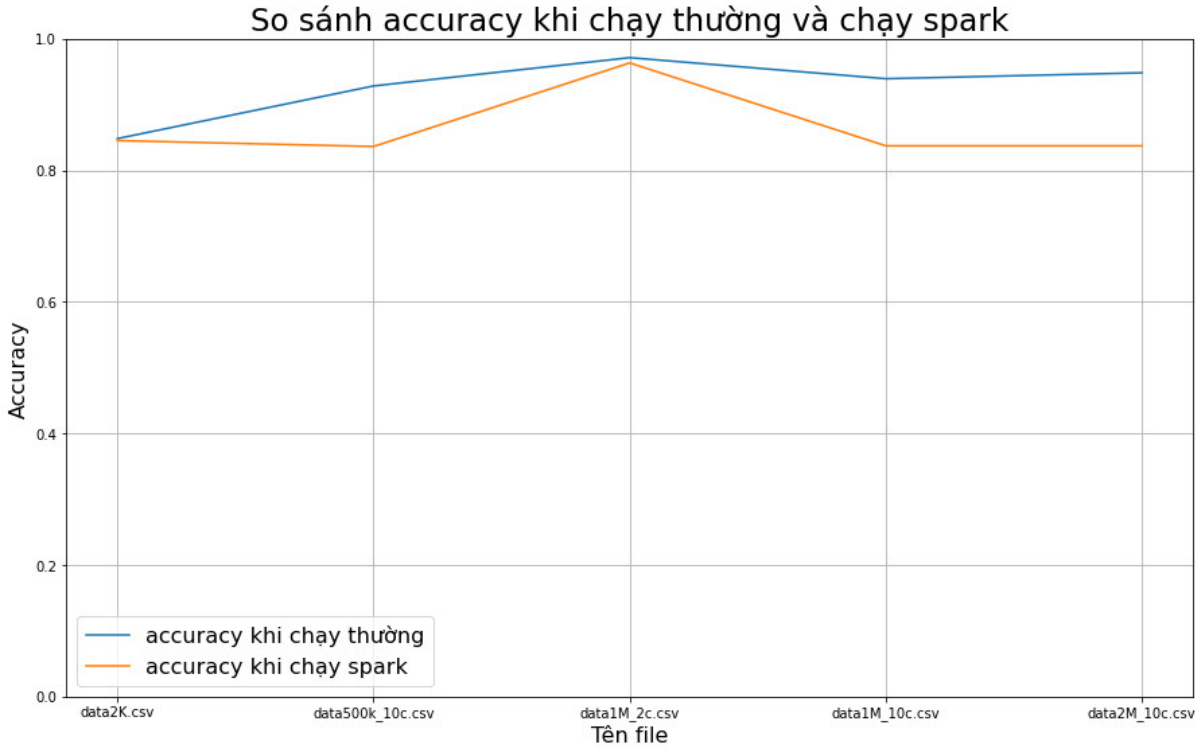
1.2. Khả năng mở rộng:

- **Chạy thường**: Với xử lý dữ liệu truyền thống thường sẽ phải thực hiện tuần tự trên một máy tính duy nhất nên bị giới hạn về khả năng xử lí dữ liệu lớn. Bị giới hạn tài nguyên. Khó trong việc mở rộng quy mô xử lý dẫn đến việc giảm hiệu suất cũng như tốc độ xử lý dữ liệu lớn
- **Spark:** Với Spark có thể xây dựng các cụm máy tính lớn để xử lý dữ liệu phân tán có nhiều node tận dụng được tài nguyên tính toán và lưu trữ của nhiều máy tính để tăng tốc độ xử lý. Tối ưu hóa bộ nhớ dữ liệu giảm thời và tăng tốc xử lý dữ liệu

1.3. Tính sẵn sàng sử dụng:

- **Chạy thường**: Chạy ứng dụng thông thường đòi hỏi việc cài đặt và cấu hình môi trường phát triển. Bạn cần cài đặt các thư viện, trình biên dịch và môi trường thực thi tương ứng.
- **Spark**: Spark có thể được cài đặt và sử dụng trên Windows, Linux, nhưng yêu cầu một số công việc cấu hình và cài đặt bổ sung. Điều này có thể đòi hỏi kiến thức kỹ thuật và thời gian để thiết lập môi trường Spark

1.4. So sánh về Accuracy



Hình 4.2: So sánh accuracy khi chạy thường và spark

Hình 4.2 cho thấy sự chính xác khi chạy thường và chạy Spark khi xử lí dữ liệu

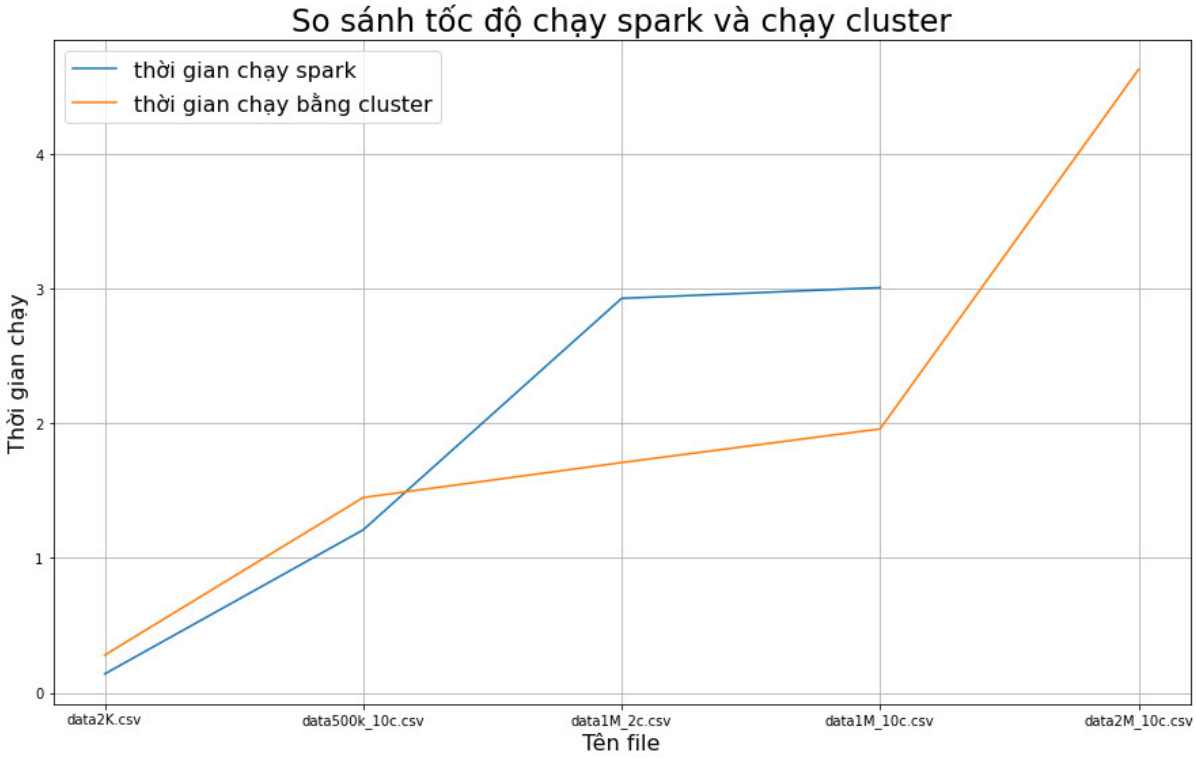
- **Chạy thường:** Xử lý dữ liệu truyền thống có thể đạt sự chính xác cao và con phải tùy thuộc vào thuật toán sử dụng với dữ liệu. Hình 4.2 cho thấy việc sử dụng thuật toán phù hợp với dữ liệu nên có sự chính xác cao hơn Spark
- **Spark:** Việc sử dụng Spark để xử lý dữ liệu do sử dụng sức mạnh tính toán và tài nguyên phân tán giúp gia tăng hiệu xuất nhưng đôi khi với những tập dữ liệu chưa đủ lớn sự chính xác có thể không cao. Hình 4.2 cho thấy việc xử lý dữ liệu với Spark có sự chính xác thấp hơn khi xử lý truyền thông.

1.4. Kết luận

- Về độ chính xác xử lý truyền thống đang có sự chính xác cao hơn khi xử lý bằng Spark
- Xử lý dữ liệu truyền thống phù hợp với dữ liệu đơn giản không đòi hỏi về sự phân tán.
- Việc xử lí dữ liệu lớn bằng cách truyền thống là không khả thi. Còn đối với Spark cho thấy tốc độ xử lí vượt trội.
- Xử lý truyền thống bị hạn chế về khả năng mở rộng không , không tận dụng hết sức mạnh tính toán bằng Spark
- Spark đi kèm với nhiều công cụ cho phép việc xử lí dữ liệu phức tạp, đòi hỏi tính toán cao, …

2. So sánh giữa chạy Spark đơn (standalone model) và Spark cluster

***So sánh về tốc độ xử lí dữ liệu**



Hình 4.3: So sánh thời gian chạy Spark đơn và Spark cluster

2.1 Spark đơn

2.1.1 Ưu Điểm:

- Dễ cài đặt và cấu hình: Spark đơn không đòi hỏi việc cấu hình và triển khai cụm máy tính, giúp giảm thiểu công việc cài đặt và triển khai.
- Phù hợp cho việc phát triển và kiểm thử: Chế độ Spark đơn phù hợp cho việc phát triển và kiểm thử ứng dụng Spark mà không cần tài nguyên phân tán.

2.1.2 Nhược điểm:

- Giới hạn về tài nguyên: Spark đơn chỉ sử dụng tài nguyên của một máy tính duy nhất, giới hạn khả năng xử lý và bộ nhớ của ứng dụng.

- Không mở rộng được: Vì không sử dụng cụm máy tính phân tán, Spark đơn không thể mở rộng quá mức cho xử lý dữ liệu lớn hoặc tăng hiệu suất.
- Thời gian xử lý lâu hơn Spark cluster (Hình 4.2).

2.2 Spark cluster

2.2.1 Ưu Điểm:

- Hiệu suất cao: Spark cluster có thể mở rộng khả năng tính toán và xử lý dữ liệu lớn bằng cách phân chia công việc và dữ liệu trên nhiều node.
- Khả năng mở rộng: Với Spark cluster, bạn có thể thêm các node vào cụm để tăng khả năng xử lý và lưu trữ.
- Tính sẵn sàng sử dụng: Spark cluster được thiết kế để xử lý dữ liệu phân tán và có sẵn nhiều công cụ quản lý và giám sát, giúp dễ dàng quản lý và sử dụng.

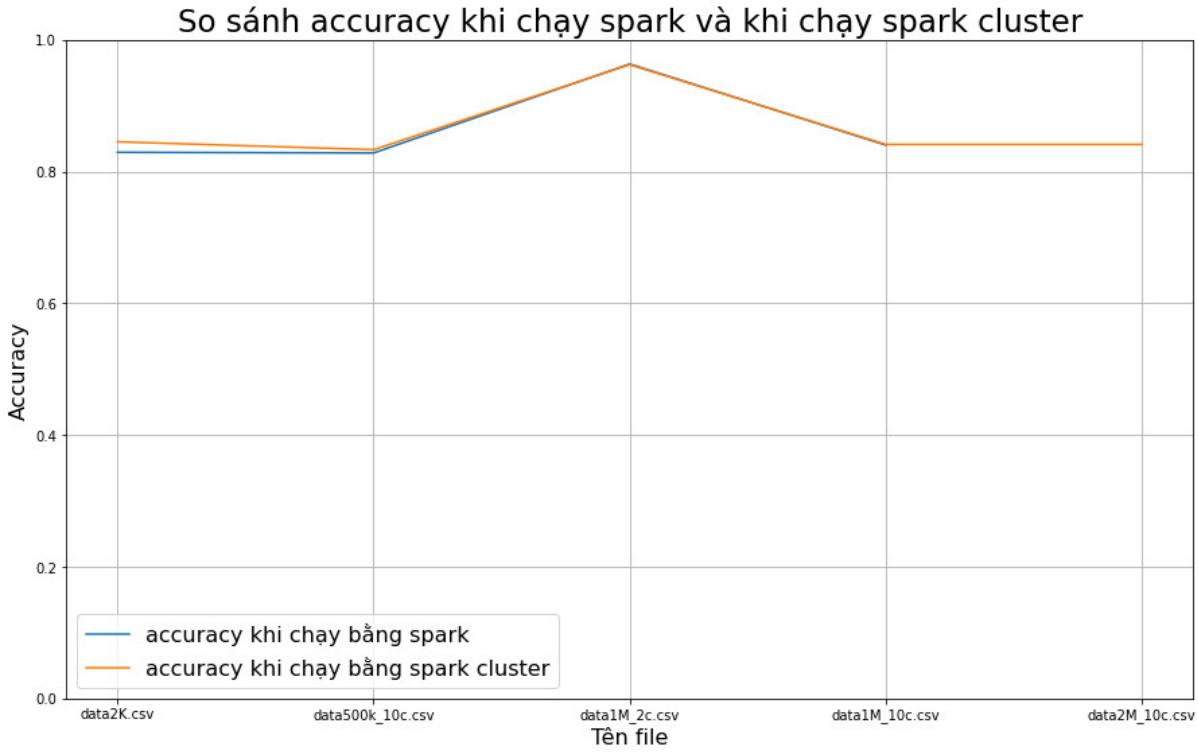
2.2.2 Nhược Điểm:

- Cần công việc cấu hình và triển khai: Spark cluster đòi hỏi công việc cấu hình và triển khai cụm máy tính phân tán, đòi hỏi kiến thức kỹ thuật và thời gian.
- Phức tạp hơn trong việc quản lý: Với Spark cluster, bạn cần quản lý và duy trì cụm máy tính phân tán, bao gồm việc giám sát, điều chỉnh và khắc phục sự cố.

Kết luận:

- Như hình 4.2 cho thấy thời gian xử lí dữ liệu lâu hơn khi dùng Spark đơn, với tập dữ liệu rất lớn Spark đơn không thể tránh việc bị Overfitting , khi sử dụng Spark cluster cho thấy hiệu suất cao hơn thời gian được rút gọn và không bị giới hạn tài nguyên của một máy
- Tùy thuộc vào một số hệ điều hành ví du như Windown thấy rằng việc xử lí dữ liệu lâu hơn so với hệ điều hành Linux. Linux tối ưu hơn để xử lý dữ liệu lớn nên việc Spark đơn thời gian xử lý nhanh hay chậm cũng phụ thuộc vào hệ điều hành.
- Việc dùng Spak cluster mất nhiều thời gian về vấn đề kĩ thuật hơn Spark đơn

***So sánh về Accuracy**



Hình 4.4: So sánh accuracy giữa chạy Spark đơn và Spark cluster

- Hình 4.4 thấy được sự chính xác giữa Spark đơn và Spark cluster là không đáng kể.

- Việc vận dụng hết sức mạnh tính toán để xử lí dữ liệu lớn giữa Spark đơn và Spark cluster thì sự chính xác gần như là tương đương nhau.
- Tuy nhiên, từ những tập dữ liệu thống kê trên hình 4.4 cho thấy hầu hết các trường hợp khi Spark cluster sẽ mang lại nhiều lợi ích nhất là về dữ liệu lớn để tăng hiệu suất tổng.

Chương V: Kết luận

Tổng kết lại, việc sử dụng Spark mang lại nhiều ưu điểm so với chạy ứng dụng thông thường. Spark cho phép xử lý dữ liệu phân tán, mở rộng khả năng tính toán và lưu trữ, cung cấp hiệu suất cao hơn và linh hoạt hơn trong việc xử lý dữ liệu lớn. Tuy nhiên, việc cài đặt và cấu hình Spark phải đòi hỏi thêm công việc và kiến thức kỹ thuật.

Việc lựa chọn giữa Spark đơn và Spark cluster phụ thuộc vào nhu cầu và yêu cầu cụ thể của dự án. Spark đơn đơn giản và phù hợp cho các dự án nhỏ với tài nguyên hạn chế, trong khi Spark cluster mang lại hiệu suất cao và khả năng mở rộng cho xử lý dữ liệu lớn. Tuy nhiên, Spark cluster đòi hỏi công việc cấu hình và triển khai phức tạp hơn, cùng với việc quản lý và duy trì cụm máy tính phân tán.

Tóm lại, Spark đơn và Spark cluster đều có ưu điểm và hạn chế riêng. Việc lựa chọn phụ thuộc vào yêu cầu và tài nguyên của dự án, đồng thời cần xem xét khả năng mở rộng và hiệu suất để đảm bảo lựa chọn phù hợp cho ứng dụng Spark trên hệ điều hành Windows.

Chương VI: Tài liệu tham khảo

- Chat GPT
- https://www.youtube.com/watch?v=jffQhcweGwY&ab_channel=edureka%21
- https://www.youtube.com/watch?v=YanzUI-30pI&ab_channel=BigTechTalk
- https://viblo.asia/p/cai-dat-apache-spark-cho-ubuntu-Az45bLJwZxY
- https://intellipaat.com/blog/tutorial/spark-tutorial/spark-architecture/?US

Chương VII: Phân công việc

Tên thành viên	Công việc
Trịnh Gia Khiêm	Cài đặt môi trường, chạy code, viết báo cáo phần III, làm slide, thuyết trình
Phạm Bá Khương	Tìm tài liệu , viết báo cáo , code
Lê Hoàng Ngọc Tú	Tìm tài liệu, viết báo cáo phần IV