BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT HƯNG YÊN

BÀI TẬP LỚN

ÁP DỤNG CÔNG NGHỆ BIGDATA ĐỂ DỰ ĐOÁN MỨC TIÊU THỤ NHIÊN LIỆU CỦA XE HƠI

NGÀNH: KHOA HỌC MÁY TÍNH

SINH VIÊN: **HOÀNG QUANG HUY**

MÃ LỚP: **12421TN**

HƯỚNG DẪN: **TS. NGUYỄN VĂN QUYẾT**

HƯNG YÊN – 2024

NHẬN XÉT

Nhận xét của giáo viên hướng dẫn

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

GIÁO VIÊN HƯỚNG DẪN

NGUYỄN VĂN QUYẾT

LỜI CAM ĐOAN

Em xin cam đoan bài tập lớn "Áp dụng công nghệ BigData để dự đoán mức tiêu thụ nhiên liệu của xe hơi" là sản phẩm của bản thân. Những phần sử dụng tài liệu tham khảo trong bài tập lớn đã được nêu rõ trong phần tài liệu tham khảo. Các số liệu, kết quả trình bày trong bài tập lớn là hoàn toàn trung thực, nếu sai em xin chịu hoàn toàn trách nhiệm và chịu mọi kỷ luật của bộ môn và nhà trường đề ra.

Hưng yên, ngày ... tháng ... năm 2024

Sinh viên

LỜI CẢM ƠN

Em xin bày tỏ lòng biết ơn chân thành tới Thầy Nguyễn Văn Quyết, người đã tận tình hướng dẫn và hỗ trợ em trong suốt quá trình thực hiện bài tập lớn này. Sự nhiệt huyết và tận tâm của Thầy đã mang lại cho em những kiến thức quý báu và giúp em vượt qua những khó khăn, thách thức trong quá trình nghiên cứu và hoàn thiện đồ án.

Những lời khuyên, sự chỉ dẫn cụ thể và chi tiết của Thầy đã giúp em mở rộng tầm nhìn, hiểu sâu hơn về lĩnh vực mà em đang nghiên cứu. Sự kiên nhẫn và sẵn sàng dành thời gian để giải đáp mọi thắc mắc của Thầy đã tiếp thêm động lực cho em, giúp em không ngừng cố gắng và hoàn thiện bản thân.

Mặc dù em đã cố gắng hết sức để hoàn thành bài tập lớn này, nhưng với kinh nghiệm và trình độ còn hạn chế, chắc chắn sẽ không tránh khỏi những thiếu sót. Em rất mong nhận được những ý kiến đóng góp, nhận xét từ Thầy để em có thể học hỏi và cải thiện trong tương lai. Em sẵn sàng tiếp thu mọi ý kiến phê bình với tinh thần cầu tiến, vì em hiểu rằng chỉ qua đó, em mới có thể tiếp tục phát triển và hoàn thiện bản thân hơn.

Một lần nữa, em xin chân thành cảm ơn Thầy Nguyễn Văn Quyết vì sự giúp đỡ quý báu và những đóng góp quan trọng mà Thầy đã dành cho em trong suốt quá trình thực hiện bài tập lớn. Em kính chúc Thầy luôn mạnh khỏe, hạnh phúc và tiếp tục gặt hái nhiều thành công trong sự nghiệp giảng dạy và nghiên cứu.

Em xin trân thành cảm ơn!

Mục Lục

DANH MỤC TỪ VIẾT TẮT

Từ viết tắt	Viết đầy đủ
RF	Random Forest
GBT	Gradient Boosting
LR	Linear Regresstion
MSE	Mean Squared Error
MAE	Mean Absolute Error
R^2	R-squared

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG BIỂU

CHƯƠNG 1: TÌM HIỂU VỀ ĐỀ TÀI

1.1 Lý do chọn đề tài

Việc chọn đề tài "Dự đoán mức tiêu thụ nhiên liệu của xe hơi" xuất phát từ sự cần thiết phải cải thiện hiệu quả sử dụng năng lượng trong ngành ô tô, đặc biệt khi mức tiêu thụ nhiên liệu đang là một yếu tố quan trọng đối với cả người tiêu dùng và các nhà sản xuất xe hơi. Trong bối cảnh kinh tế hiện nay, chi phí nhiên liệu ngày càng trở thành một gánh nặng lớn đối với người sử dụng phương tiện giao thông, đặc biệt là đối với những xe có mức tiêu thụ cao. Vì vậy, việc phát triển một hệ thống dự đoán mức tiêu thụ nhiên liệu có thể giúp người tiêu dùng đưa ra quyết định hợp lý khi lựa chọn xe, đồng thời giúp các nhà sản xuất xe hơi tối ưu hóa thiết kế sản phẩm của mình, giảm thiểu lãng phí năng lượng và cải thiện hiệu quả vận hành.

Ngoài ra, việc dự đoán mức tiêu thụ nhiên liệu còn có ý nghĩa quan trọng trong việc thúc đẩy sự phát triển của các phương tiện tiết kiệm năng lượng và tối ưu hóa công nghệ động cơ. Các yếu tố như khối lượng xe, công suất động cơ, loại nhiên liệu sử dụng và các đặc tính kỹ thuật khác đều ảnh hưởng trực tiếp đến mức tiêu thụ nhiên liệu. Việc áp dụng các phương pháp phân tích dữ liệu và học máy sẽ giúp mô hình hóa mối quan hệ giữa các yếu tố này, từ đó cung cấp những dự đoán chính xác về mức tiêu thụ nhiên liệu của các loại xe khác nhau trong các điều kiện vận hành cụ thể. Điều này không chỉ giúp các nhà sản xuất cải tiến công nghệ động cơ mà còn giúp họ đáp ứng nhu cầu ngày càng cao về các phương tiện tiết kiệm nhiên liệu.

Bên cạnh đó, trong bối cảnh ngành công nghiệp ô tô đang đối mặt với sự cạnh tranh khốc liệt, các công ty cần có những công cụ chính xác và hiệu quả để phân tích và tối ưu hóa sản phẩm của mình. Việc áp dụng các kỹ thuật học máy như hồi quy, cây quyết định hay mạng nơ-ron trong việc dự đoán mức tiêu thụ nhiên liệu không chỉ góp phần nâng cao hiệu quả sản xuất mà còn tạo ra các sản phẩm đáp ứng nhu cầu thực tế của người tiêu dùng, từ đó gia tăng tính cạnh tranh trên thị trường.

Ngoài yếu tố tiết kiệm chi phí, một trong những lý do quan trọng khác khi chọn đề tài này là sự thay đổi trong hành vi tiêu dùng và yêu cầu ngày càng cao đối với các phương tiện giao thông. Người tiêu dùng hiện nay không chỉ quan tâm đến khả năng vận hành mà còn đánh giá các yếu tố liên quan đến mức độ tiết kiệm nhiên liệu của phương tiện. Vì vậy, việc dự đoán chính xác mức tiêu thụ nhiên liệu sẽ giúp các nhà sản xuất hiểu rõ hơn về nhu cầu thị trường, từ đó đưa ra các sản phẩm phù hợp với xu hướng tiêu dùng và cải thiện chất lượng dịch vụ khách hàng.

Tóm lại, việc chọn đề tài này không chỉ mang lại lợi ích thiết thực trong việc tối ưu hóa mức tiêu thụ nhiên liệu mà còn giúp thúc đẩy sự phát triển của ngành công nghiệp ô tô, đặc biệt trong bối cảnh người tiêu dùng ngày càng quan tâm đến hiệu quả năng lượng và chi phí vận hành. Dự đoán chính xác mức tiêu thụ nhiên liệu sẽ mở ra nhiều cơ hội cải tiến công nghệ, tăng trưởng thị trường và phát triển bền vững cho ngành công nghiệp ô tô trong tương lai.

CHƯƠNG 2: CHƯƠNG2: CƠ SỞ LÝ THUYẾT

2.1 Big Data là gì?

Trên thế giới ngày nay, chúng ta đã bước vào một kỷ nguyên số hóa mạnh mẽ. Cùng với sự phát triển của công nghệ thông tin và truyền thông, lượng thông tin mà chúng ta tạo ra và thu thập hàng ngày đã trở nên khổng lồ và phức tạp hơn bao giờ hết. Điều này đặt ra một thách thức lớn cho việc xử lý, phân tích và tận dụng thông tin một cách hiệu quả. Đó chính là lý do Big Data (dữ liệu lớn) đã trở thành một thuật ngữ quan trọng và được quan tâm rộng rãi.

Big Data ám chỉ đến khối lượng dữ liệu rất lớn, phức tạp và đa dạng, vượt quá khả năng của công nghệ thông tin truyền thống để xử lý và phân tích bằng phương pháp truyền thống. Đây không chỉ là dữ liệu từ các nguồn truyền thống như cơ sở dữ liệu và hệ thống thông tin, mà còn bao gồm dữ liệu từ các nguồn mới như mạng xã hội, thiết bị di động, cảm biến và internet vạn vật (Internet of Things).

Big Data mang lại nhiều lợi ích đáng kể trong nhiều lĩnh vực khác nhau. Trong lĩnh vực kinh doanh, Big Data cho phép doanh nghiệp phân tích hành vi của khách hàng, dự đoán xu hướng và thị trường, từ đó đưa ra quyết định kinh doanh thông minh. Các ngành công nghiệp khác như y tế, giáo dục, nông nghiệp và giao thông cũng đang tận dụng Big Data để nâng cao chất lượng dịch vụ, tối ưu hóa quy trình và tăng cường hiệu suất.

Tuy nhiên, để tận dụng tối đa tiềm năng của Big Data, chúng ta cần đối mặt với những thách thức. Một trong những thách thức đó là khả năng lưu trữ và xử lý dữ liệu lớn. Dữ liệu lớn yêu cầu hệ thống lưu trữ mạnh mẽ và sự phân tán thông tin để đảm bảo tính khả dụng và độ tin cậy. Hơn nữa, để phân tích dữ liệu lớn, chúng ta cần sự kết hợp của khoa học dữ liệu, trí tuệ nhân tạo và các công cụ phân tích mạnh mẽ.

Cùng với những tiềm năng và thách thức, Big Data cũng đặt ra các vấn đề liên quan đến quyền riêng tư và an ninh thông tin. Với lượng thông tin lớn và phức tạp, việc bảo vệ thông tin cá nhân và đảm bảo an toàn cho dữ liệu trở thành một vấn đề cấp bách. Việc áp dụng các biện pháp bảo mật hiệu quả là cần thiết để đảm bảo sự tin tưởng và sự phát triển bền vững của Big Data.

2.2 Apache Hadoop

2.2.1 Hadoop là gì ?

Hadoop là một dạng framework, cụ thể là Apache. Apache Hadoop là một mã nguồn mở cho phép sử dụng các distributed processing (ứng dụng phân tán) để quản lý và lưu trữ những tệp dữ liệu lớn. Hadoop áp dụng mô hình MapReduce trong hoạt động xử lý Big Data.

Vậy MapReduce là gì? MapReduce vốn là một nền tảng được Google tạo ra để quản lý dữ liệu của họ. Nhiệm vụ của MapReduce là tiếp nhận một khối lượng dữ liệu lớn. Sau đó sẽ tiến hành tách các dữ liệu này ra thành những phần nhỏ theo một tiêu chuẩn nào

đó. Từ đó sẽ sắp xếp, trích xuất các tệp dữ liệu con mới phù hợp với yêu cầu của người dùng. Đây cũng là cách mà thanh tìm kiếm của Google hoạt động troCòn bản thân Hadoop cũng là một dạng công cụ mẫu giúp phân tán dữ liệu theo mô hình như vậy. Cho nên MapReduce được sử dụng như một nền tảng lý tưởng của Hadoop. Về cơ bản, Hadoop sẽ giúp người dùng tổng hợp và xử lý một lượng thông tin lớn trong thời gian ngắn bằng MapReduce.

Còn với chức năng lưu trữ, Hadoop sẽ dùng HDFS. HDFS là gì? Nó được biết đến như một kho thông tin có độ truy cập nhạy và chi phí thấp.

Hadoop được phát triển nên từ ngôn ngữ Java. Tuy nhiên nó vẫn hỗ trợ một số ngôn ngữ lập trình khác như C++, Python hay Pearl nhờ cơ chế streaming.ng khi chúng ta sử dụng hằng ngày.

2.1.2 Kiến trúc của Hadoop

Một cụm Hadoop sẽ bao gồm 1 master node (node chủ) và rất nhiều worker/slave node (node nhân viên). Một cụm cũng bao gồm 2 phần là MapReduce layer và HDFS layer. Master node bao gồm JobTracker, TaskTracker, NameNode, và DataNode. Còn Worker/Slave node bao gồm DataNode và TaskTracker. Trong một số trường hợp, Worker/Slave node được dùng để làm dữ liệu hoặc tính toán.

Hadoop Apache bao gồm 4 module khác nhau. Cụ thể:

- Hadoop Common: Hadoop Common được dùng như một thư viện lưu trữ các tiện ích của Java. Tại đây có những tính năng cần thiết để các modules khác sử dụng. Những thư viện này mang đến hệ thống file và lớp OS trừu tượng
- Hadoop Yarn: Phần này được dùng như một framework. Nó hỗ trợ hoạt động quản lý thư viện tài nguyên của các cluster và thực hiện chạy phân tích tiến trình.
- Hadoop Distributed File System (HDFS): phân tán cung cấp truy cập thông lượng cao giúp cho ứng dụng chủ. Cụ thể, khi HDFS nhận được một tệp tin, nó sẽ tự động chia file đó ra thành nhiều phần nhỏ. Các mảnh nhỏ này được nhân lên nhiều lần và chia ra lưu trữ tại các máy chủ khác nhau để phân tán sức nặng mà dữ liệu tạo nên.
- Hadoop MapReduce: cho phép phân tán dữ liệu từ một máy chủ sang nhiều máy con. Mỗi máy con này sẽ nhận một phần dữ liệu khác nhau và tiến hành xử lý cùng lúc. Sau đó chúng sẽ báo lại kết quả lên máy chủ. Máy chủ tổng hợp thông tin lại rồi trích xuất theo như yêu cầu của người dùng.

2.1.3 Cách hoạt động của Hadoop

Giai đoạn 1: người dùng hoặc ứng dụng sẽ gửi một job lên Hadoop để yêu cầu xử lý và thao tác. Job này sẽ đi kèm các thông tin cơ bản như: nơi lưu trữ dữ liệu input và output, các java class chứa các dòng lệnh thực thi, các thông số thiết lập cụ thể.

Giai đoạn 2: Sau khi nhận được các thông tin cần thiết, máy chủ sẽ chia khối lượng công việc đến cho các máy trạm. Máy chủ sẽ tiến hành theo dõi quá trình hoạt động của các máy trạm và đưa ra các lệnh cần thiết khi có lỗi xảy ra.

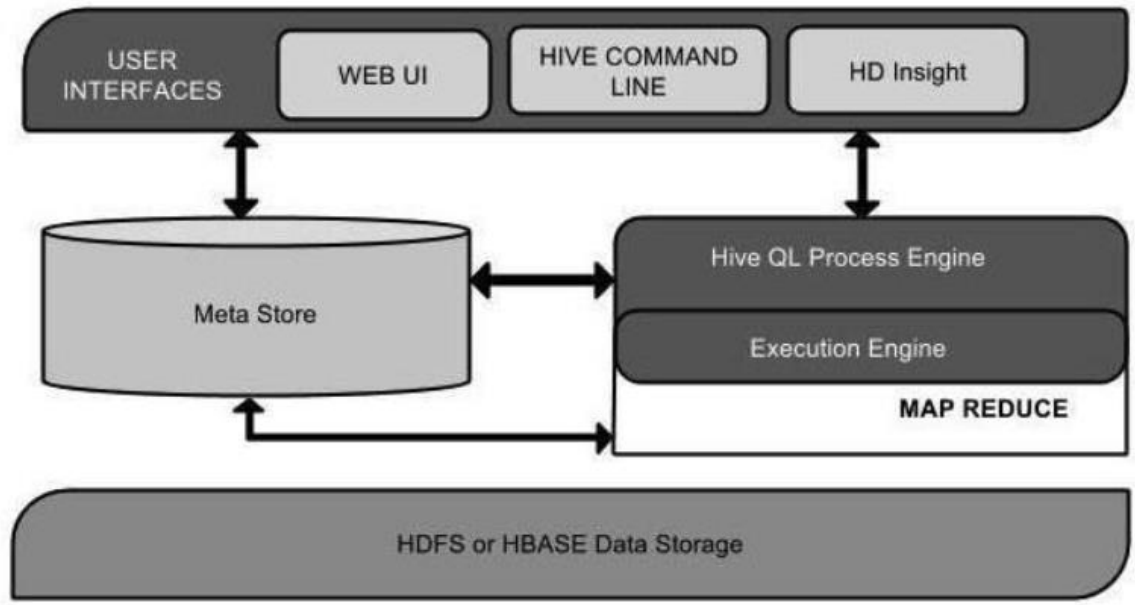
Giai đoạn 3: các nodes khác nhau sẽ tiến hành chạy tác vụ MapReduce. Nó chia nhỏ các khối và thay phiên nhau xử lý dữ liệu. Khi Hadoop hoạt động, nó sử dụng một tệp tin nền làm địa chỉ thường trú. Tệp tin này có thể tồn tại trên 1 hoặc nhiều máy chủ khác nhau.

2.3 Apache Hive

2.3.1 Hive là gì ?

Hive là một công cụ lưu trữ dữ liệu được triển khai dựa trên Hadoop Distributed File System. Hệ thống này hỗ trợ Hive thực hiện nhiều công việc khác nhau như đóng gói dữ liệu, truy vấn đặc biệt và phân tích khối dữ liệu lớn.

2.3.2 Cấu trúc của Hive



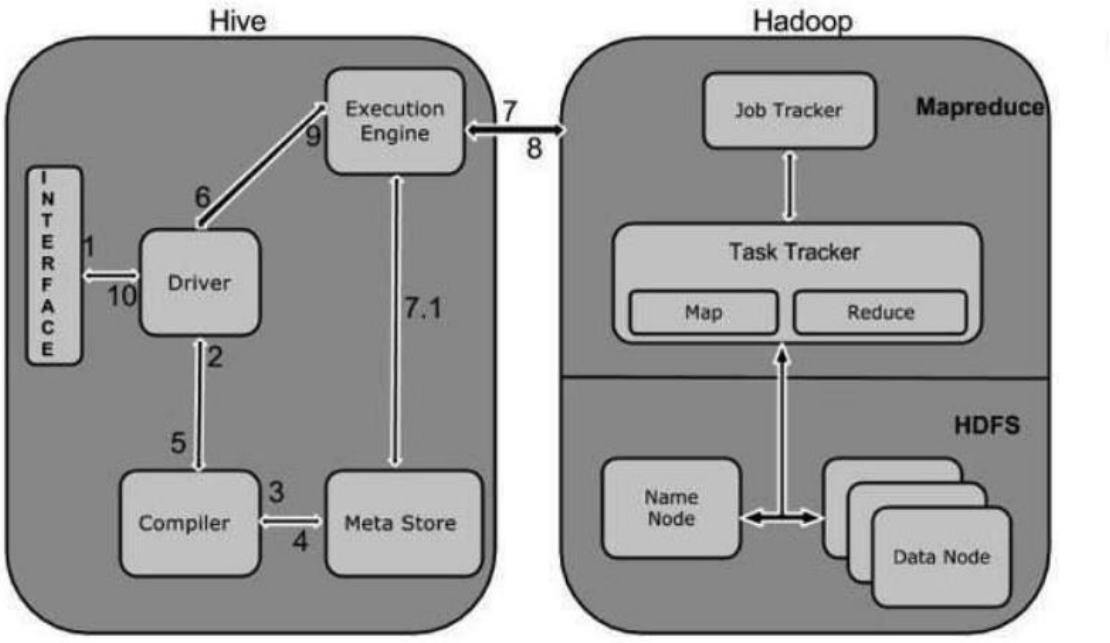
Hình 2. 1: Kiến trúc của Apache Hive

- User Interface: Hive là một phần mềm cơ sở hạ tầng kho dữ liệu có thể tạo ra sự tương tác giữa người dùng và HDFS. Các giao diện người dùng mà Hive hỗ trợ

là Hive Web UI, Hive command line và Hive HD Insight (Trong máy chủ Windows).

- Meta Store: Hive chọn các máy chủ cơ sở dữ liệu tương ứng để lưu trữ lược đồ hoặc metadata của các bảng, cơ sở dữ liệu, các cột trong một bảng, các loại dữ liệu của chúng và ánh xạ HDFS.
- HiveQL Process Engine: HiveQL tương tự như SQL để truy vấn thông tin lược đồ trên Metastore. Đây là một trong những thay thế của phương pháp truyền thống cho chương trình MapReduce. Thay vì viết chương trình MapReduce bằng Java, chúng ta có thể viết một truy vấn cho công việc MapReduce và xử lý nó.
- Execution Engine: Phần kết hợp của công cụ xử lý HiveQL và MapReduce là Công cụ thực thi Hive (Hive Execution Engine). Công cụ thực thi xử lý truy vấn và tạo kết quả giống như kết quả MapReduce.
- HDFS hoặc HBASE: Hệ thống tệp phân tán Hadoop hoặc HBASE là các kỹ thuật lưu trữ dữ liệu để lưu trữ dữ liệu vào hệ thống tệp.

2.3.3 Cách hoạt động của Hive



Hình 2. 2: Sơ đồ hoạt động của Hive

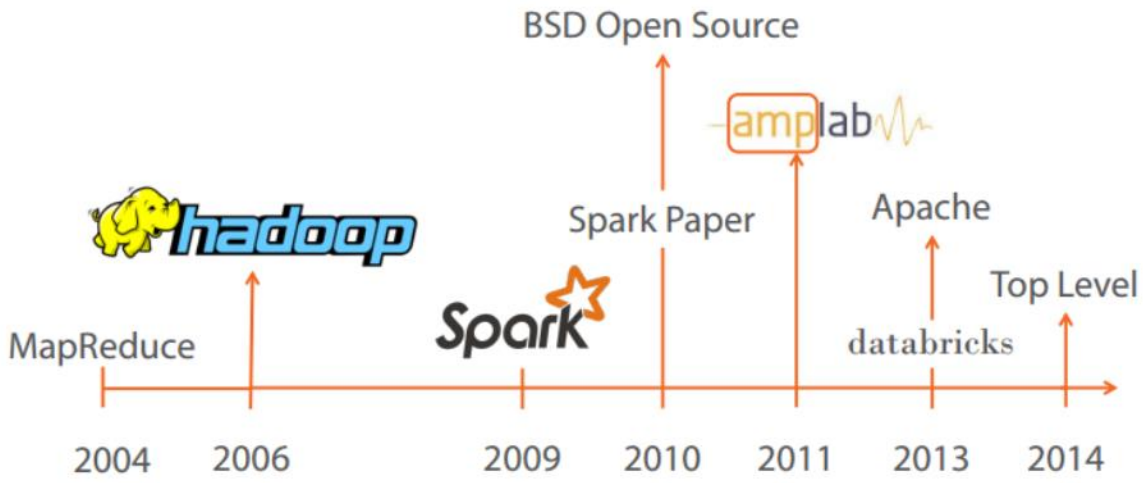
Hive tương tác với framework Hadoop qua các bước sau:

- Thực thi query: Giao diện Hive như Command line hoặc Giao diện người dùng web gửi truy vấn đến Trình điều khiển (bất kỳ trình điều khiển cơ sở dữ liệu nào như JDBC, ODBC, v.v.) để thực thi.

- Nhận kế hoạch: Trình điều khiển có sự trợ giúp của trình biên dịch truy vấn để phân tích cú pháp truy vấn để kiểm tra cú pháp và kế hoạch truy vấn hoặc yêu cầu của truy vấn.
- Nhận metadata: Trình biên dịch gửi yêu cầu metadata đến Metastore (bất kỳ cơ sở dữ liệu nào).
- Gửi metadata: Metastore gửi metadata như một phản hồi cho trình biên dịch.
- Gửi kế hoạch: Trình biên dịch kiểm tra yêu cầu và gửi lại kế hoạch cho trình điều khiển. Đến đây, việc phân tích cú pháp và biên dịch một truy vấn đã hoàn tất.
- Kế hoạch thực hiện: Trình điều khiển gửi kế hoạch thực hiện đến công cụ thực thi.
- Thực thi công việc: Trong nội bộ, quá trình thực thi công việc là một công việc MapReduce. Công cụ thực thi gửi công việc đến JobTracker, trong node Name và nó gán công việc này cho TaskTracker, trong node Data. Ở đây, truy vấn thực thi công việc MapReduce. Hoạt động metadata: Trong khi thực hiện, công cụ thực thi có thể thực thi các hoạt động metadata với Metastore.
- Lấy kết quả: Công cụ thực thi nhận kết quả từ các node Data.
- Gửi kết quả: Công cụ thực thi gửi các giá trị kết quả đó đến trình điều khiển.
- Gửi kết quả: Trình điều khiển gửi kết quả đến Giao diện Hive.

2.4 Apache Pyspark

2.4.1 Spark là gì

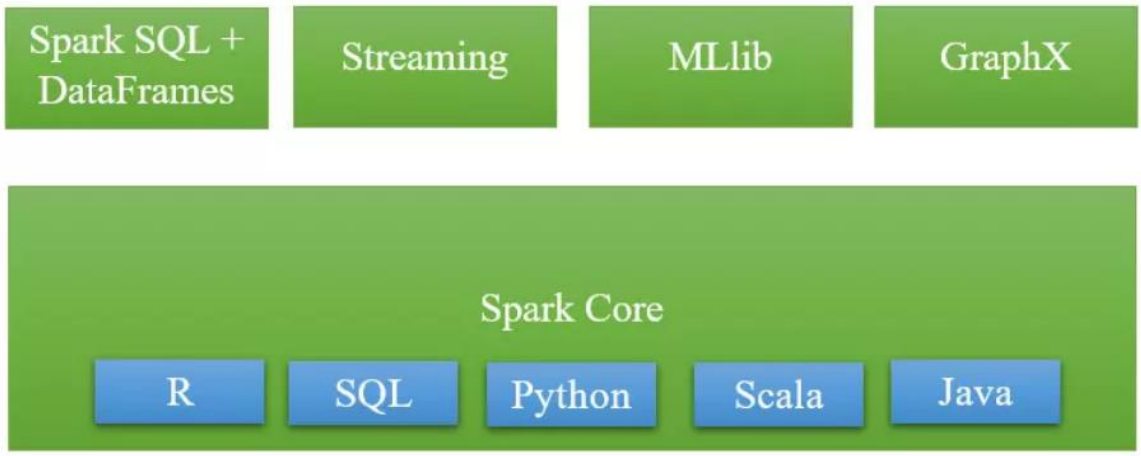


Hình 2. 3: Các mã nguồn của Apache

Apache Spark là một framework mã nguồn mở tính toán cụm, được phát triển sơ khởi vào năm 2009 bởi AMPLab.

Spark cho phép xử lý dữ liệu theo thời gian thực, vừa nhận dữ liệu từ các nguồn khác nhau đồng thời thực hiện ngay việc xử lý trên dữ liệu vừa nhận được (Spark Streaming). Spark không có hệ thống file của riêng mình, nó sử dụng hệ thống file khác như: HDFS, Cassandra, S3,… Spark hỗ trợ nhiều kiểu định dạng file khác nhau (text, csv, json…) đồng thời nó hoàn toàn không phụ thuộc vào bất cứ một hệ thống file nào.

b) Các thành phần của Spark



Hình 2. 4: Các thành phần của Apache Spark

- **Spark Core:** là nền tảng cho các thành phần còn lại và các thành phần này muốn khởi chạy được thì đều phải thông qua Spark Core do Spark Core đảm nhận vai trò thực hiện công việc tính toán và xử lý trong bộ nhớ (In-memory computing) đồng thời nó cũng tham chiếu các dữ liệu được lưu trữ tại các hệ thống lưu trữ bên ngoài.
- **Spark SQL:** cung cấp một kiểu data abstraction mới (SchemaRDD) nhằm hỗ trợ cho cả kiểu dữ liệu có cấu trúc (structured data) và dữ liệu nửa cấu trúc (semi-structured data – thường là dữ liệu dữ liệu có cấu trúc nhưng không đồng nhất và cấu trúc của dữ liệu phụ thuộc vào chính nội dung của dữ liệu ấy). Spark SQL hỗ trợ DSL (Domain-specific language) để thực hiện các thao tác trên DataFrames bằng ngôn ngữ Scala, Java hoặc Python và nó cũng hỗ trợ cả ngôn ngữ SQL với giao diện command-line và ODBC/JDBC server.
- **Spark Streaming**: được sử dụng để thực hiện việc phân tích stream bằng việc coi stream là các mini-batches và thực hiệc kỹ thuật RDD transformation đối với các dữ liệu mini-batches này. Qua đó cho phép các đoạn code được viết cho xử lý batch có thể được tận dụng lại vào trong việc xử lý stream, làm cho việc phát

triển lambda architecture được dễ dàng hơn. Tuy nhiên điều này lại tạo ra độ trễ trong xử lý dữ liệu (độ trễ chính bằng mini-batch duration) và do đó nhiều chuyên gia cho rằng Spark Streaming không thực sự là công cụ xử lý streaming giống như Storm hoặc Flink.

- **MLlib (Machine Learning Library):** MLlib là một nền tảng học máy phân tán bên trên Spark do kiến trúc phân tán dựa trên bộ nhớ. Theo các so sánh benchmark Spark MLlib nhanh hơn 9 lần so với phiên bản chạy trên Hadoop (Apache Mahout).
- **GrapX:** Grapx là nền tảng xử lý đồ thị dựa trên Spark. Nó cung cấp các Api để diễn tảcác tính toán trong đồ thị bằng cách sử dụng Pregel Api.

2.4.2 Cách hoạt động của Spark

Cách hoạt động của Spark gồm các bước sau:

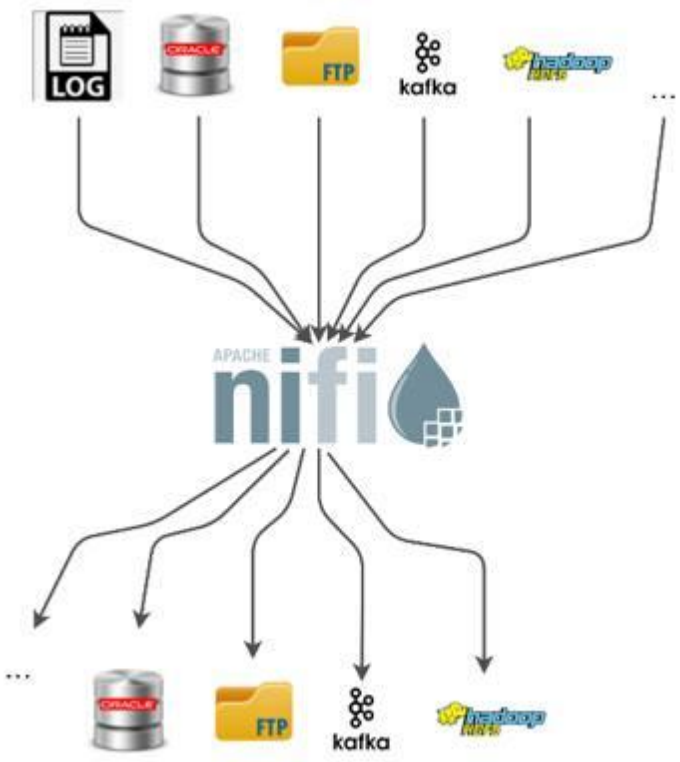
- Load data: Dữ liệu được load vào Spark từ nhiều nguồn khác nhau, chẳng hạn như Hadoop Distributed File System (HDFS), Apache Cassandra, Amazon S3, hay bất kỳ nguồn dữ liệu nào khác.
- Transform data: Apache Spark cung cấp nhiều API cho việc biến đổi dữ liệu, bao gồm các thư viện mã nguồn mở phổ biến như Spark SQL, Spark Streaming, Spark MLlib và Spark GraphX. Dữ liệu có thể được lọc, sắp xếp, thống kê, xử lý chuỗi, xử lý đồ thị, v.v.
- Persist data: Khi dữ liệu được biến đổi, Spark cung cấp nhiều lựa chọn để lưu trữ dữ liệu, bao gồm bộ nhớ, ổ đĩa, và HDFS.
- Execute actions: Spark cho phép thực thi các hành động (actions) trên dữ liệu, chẳng hạn như tính toán tổng, tính trung bình, thống kê, hay xuất dữ liệu ra các hệ thống lưu trữ khác.
- Parallel processing: Apache Spark sử dụng cơ chế xử lý phân tán để thực hiện các tác vụ trên cụm máy tính, tận dụng tối đa khả năng tính toán của cụm máy tính để tăng hiệu suất xử lý.

2.5 Apache NiFi

2.5.1 Nifi là gì?

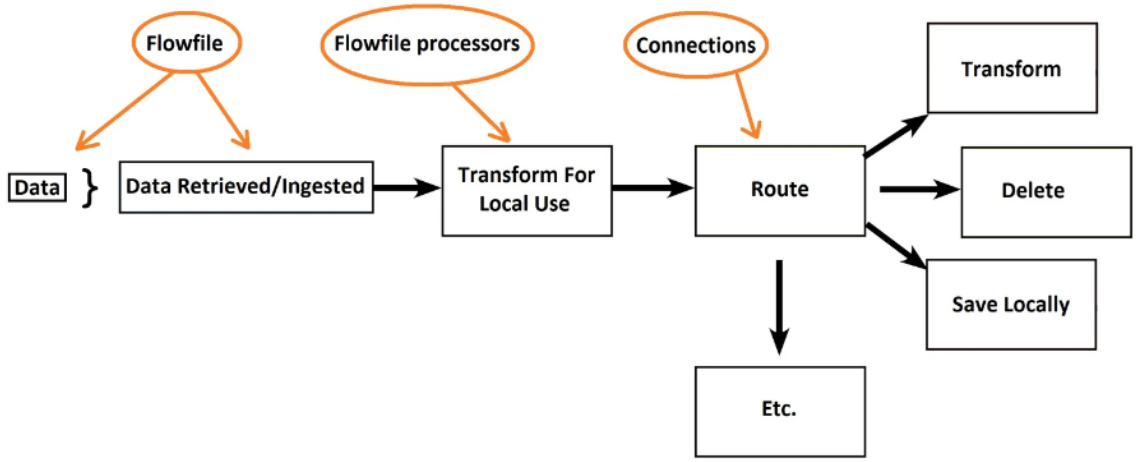
Apache NiFi là một phần mềm open-source viết bằng Java, được tạo ra để tự động hóa luồng dữ liệu giữa các hệ thống phần mềm với nhau. Nó được xây dựng từ năm 2006

dựa trên phần mềm "NiagaraFiles" phát triển bởi anh NSA, sau đó được chuyển sang open-source vào năm 2014



Hình 2. 5: Cách Nifi thu thập và truyền dữ liệu

2.5.2 Các thành phần chính trong Nifi



Hình 2. 6: Các thành phần trong Nifi

Flowfile: đại diện cho đơn vị dữ liệu được thực hiện trong luồng ví dụ như 1 bản ghi text, 1 file ảnh… gồm 2 phần:

- Content: chính là dữ liệu nó đại diện

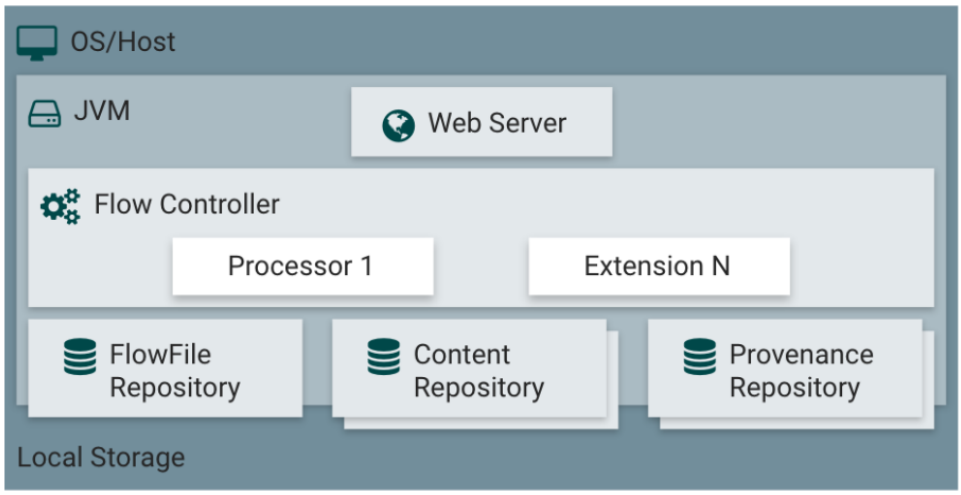
- Attribute: thuộc tính của flow file (key-value)

Flowfile Processor: đây là những thứ thực hiện công việc trong nifi, bên trong nó đã có chứa sẵn code thực thi các tác vụ trong các trường hợp với input và output. Khối xử lý sinh ra các flowfile. Các processor hoạt động song song với nhau

Connection: đóng vai trò kết nối giữa các processors. Ngoài ra nó còn là một hàng đợi chứa các flowfile chưa được xử lý:

- Xác định thời gian flowfile tồn tại trong queue
- Phân chia flowfile đến các node trong cụm (load balancing)
- Xác định tần suất flowfile nhả ra cho hệ thống

2.5.3 Kiến trúc hệ thống trong Nifi



Hình 2. 7: Kiến trúc hệ thống của Nifi

- Web server: cung cấp giao diện cho người dùng sử dụng các thao tác
- Flow Controller: cung cấp tài nguyên cho quá trình hoạt động của hệ thống
- Extensions: Bao gồm các thành phần xây dựng nên luồng dữ liệu trong nifi: các processors có nhiệm vụ xử lý, điều hướng; Các log, Controller service chứa các chức năng dùng cho các extensions khác
- Flowfile repository: Chỉ lưu lại các metadata của flowfile vì flowfile lưu dữ liệu rồi.
- Content repository: Lưu trữ dữ liệu thực đang được xử lý trong luồng. Nifi lưu lại tất cả các phiên bản dữ liệu trước và sau khi được xử lý

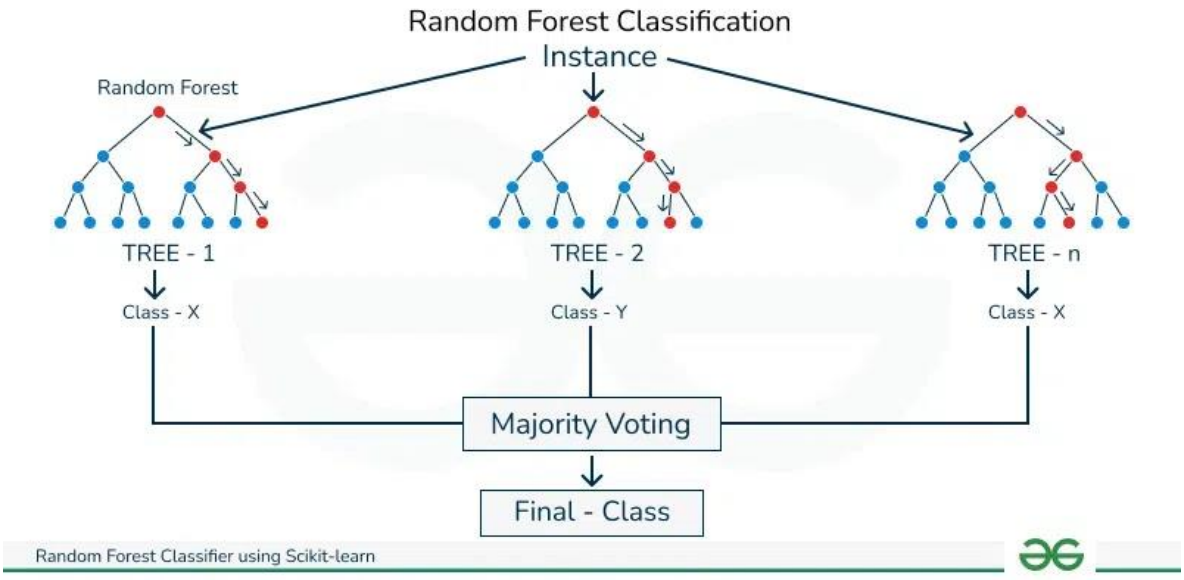
Provenance repository: Lưu lại toàn bộ lịch sử của flowfile.

2.6 Nghiên cứu một số kỹ thuật học máy được áp dụng trong xử lý dữ liệu lớn

2.6.1 Mô hình Random Forest

Rừng ngẫu nhiên hoặc Rừng quyết định ngẫu nhiên là một thuật toán Machine learning được giám sát được sử dụng để phân loại, hồi quy và các tác vụ khác sử dụng cây quyết định. Rừng ngẫu nhiên đặc biệt phù hợp để xử lý các tập dữ liệu lớn và phức tạp, xử lý các không gian đối tượng nhiều chiều và cung cấp thông tin chi tiết về tầm quan trọng của đối tượng. Khả năng duy trì độ chính xác dự đoán cao của thuật toán này trong khi giảm thiểu việc trang bị quá mức khiến nó trở thành lựa chọn phổ biến trên nhiều lĩnh vực khác nhau, bao gồm tài chính, chăm sóc sức khỏe và phân tích hình ảnh, cùng nhiều lĩnh vực khác.

Trình phân loại rừng ngẫu nhiên tạo một tập hợp cây quyết định từ tập hợp con được chọn ngẫu nhiên của tập huấn luyện. Đó là một tập hợp các cây quyết định (DT) từ một tập hợp con được chọn ngẫu nhiên của tập huấn luyện và sau đó Nó thu thập phiếu bầu từ các cây quyết định khác nhau để quyết định dự đoán cuối cùng.



Hình 2. 8: Sơ đồ tổng quan về RandomForest

Ngoài ra, trình phân loại rừng ngẫu nhiên có thể xử lý cả nhiệm vụ phân loại và hồi quy, đồng thời khả năng cung cấp điểm quan trọng của tính năng khiến nó trở thành một công cụ có giá trị để hiểu tầm quan trọng của các biến khác nhau trong tập dữ liệu.

- **Tầm quan trọng của tính năng rừng ngẫu nhiên**

Nếu bạn không biết cây quyết định hoạt động như thế nào hoặc lá hoặc nút là gì, thì đây là một mô tả hay từ Wikipedia: "Trong cây quyết định, mỗi nút bên trong đại diện cho một 'kiểm tra' trên một thuộc tính (ví dụ: liệu một đồng xu có lật xuất hiện mặt ngửa hoặc mặt ngửa), mỗi nhánh biểu thị kết quả của bài kiểm tra và mỗi nút lá biểu thị một nhãn lớp (quyết định được đưa ra sau khi tính toán tất cả các thuộc tính). Một nút không có nút con là một chiếc lá."

Bằng cách xem xét tầm quan trọng của tính năng, bạn có thể quyết định những tính năng nào có thể bị loại bỏ vì chúng không đóng góp đủ (hoặc đôi khi không đóng góp gì cả) cho quá trình dự đoán. Điều này rất quan trọng vì nguyên tắc chung trong học máy là bạn càng có nhiều tính năng thì mô hình của bạn càng có nhiều khả năng bị trang bị quá mức và ngược lại.

- **Cách rừng ngẫu nhiên hoạt động**

Tập hợp các Cây quyết định: Rừng ngẫu nhiên tận dụng sức mạnh của việc học tập tổng hợp bằng cách xây dựng một đội quân Cây quyết định . Những cây này giống như các chuyên gia riêng lẻ, mỗi chuyên gia chuyên về một khía cạnh cụ thể của dữ liệu. Điều quan trọng là chúng hoạt động độc lập, giảm thiểu rủi ro mô hình bị ảnh hưởng quá mức bởi các sắc thái của một cây duy nhất.

Lựa chọn tính năng ngẫu nhiên: Để đảm bảo rằng mỗi cây quyết định trong tập hợp mang lại một góc nhìn độc đáo, Rừng ngẫu nhiên sử dụng lựa chọn tính năng ngẫu nhiên. Trong quá trình huấn luyện mỗi cây, một tập hợp con các đặc tính ngẫu nhiên sẽ được chọn. Tính ngẫu nhiên này đảm bảo rằng mỗi cây tập trung vào các khía cạnh khác nhau của dữ liệu, thúc đẩy một tập hợp các yếu tố dự đoán đa dạng trong quần thể.

Tổng hợp hoặc đóng bao Bootstrap: Kỹ thuật đóng bao là nền tảng trong chiến lược đào tạo của Random Forest, bao gồm việc tạo nhiều mẫu bootstrap từ tập dữ liệu gốc, cho phép lấy mẫu các phiên bản thay thế. Điều này dẫn đến các tập hợp dữ liệu khác nhau cho mỗi cây quyết định, tạo ra tính biến đổi trong quá trình đào tạo và làm cho mô hình trở nên mạnh mẽ hơn.

Ra quyết định và biểu quyết: Khi đưa ra dự đoán, mỗi cây quyết định trong Rừng ngẫu nhiên sẽ bỏ phiếu. Đối với các nhiệm vụ phân loại, dự đoán cuối cùng được xác định theo chế độ (dự đoán thường xuyên nhất) trên tất cả các cây. Trong các tác vụ hồi

quy, giá trị trung bình của các dự đoán cây riêng lẻ được lấy. Cơ chế bỏ phiếu nội bộ này đảm bảo quá trình ra quyết định tập thể và cân bằng.

2.6.2 Mô hình Gradient-Boosting

Gradient Boosting là một dạng tổng quát hóa của AdaBoost. Cụ thể như sau, vẫn vấn đề tối ưu ban đầu

$$\min_{c_n, w_n} L(y, W_{n-1} + c_n w_n))$$

Hình 2. 9: Công thức tính Confidence score

Phía trên là công thức cập nhật tham số mô hình theo hướng giảm của đạo hàm (Gradient Descent). Công thức này được sử dụng không gian tham số, tuy nhiên, để liên hệ với bài toán chúng ta đang xét, mình chuyển công thức sang góc nhìn của không gian hàm số.

Nếu chúng ta coi chuỗi các model boosting là một hàm số W, thì mỗi hàm learner có thể coi là một tham số w. Đến đây, để cực tiểu hóa hàm loss L(y,W) chúng ta áp dụng Gradient Descent.

$$W_n = W_{n-1} - \eta \frac{\partial}{\partial w} L(W_{n-1})$$

Hình 2. 10: Boosting Gradient Descent

Ta có thể thấy mối quan hệ liên quan sau:

$$c_n w_n \approx -\eta \frac{\partial}{\partial w} L(W_{n-1})$$

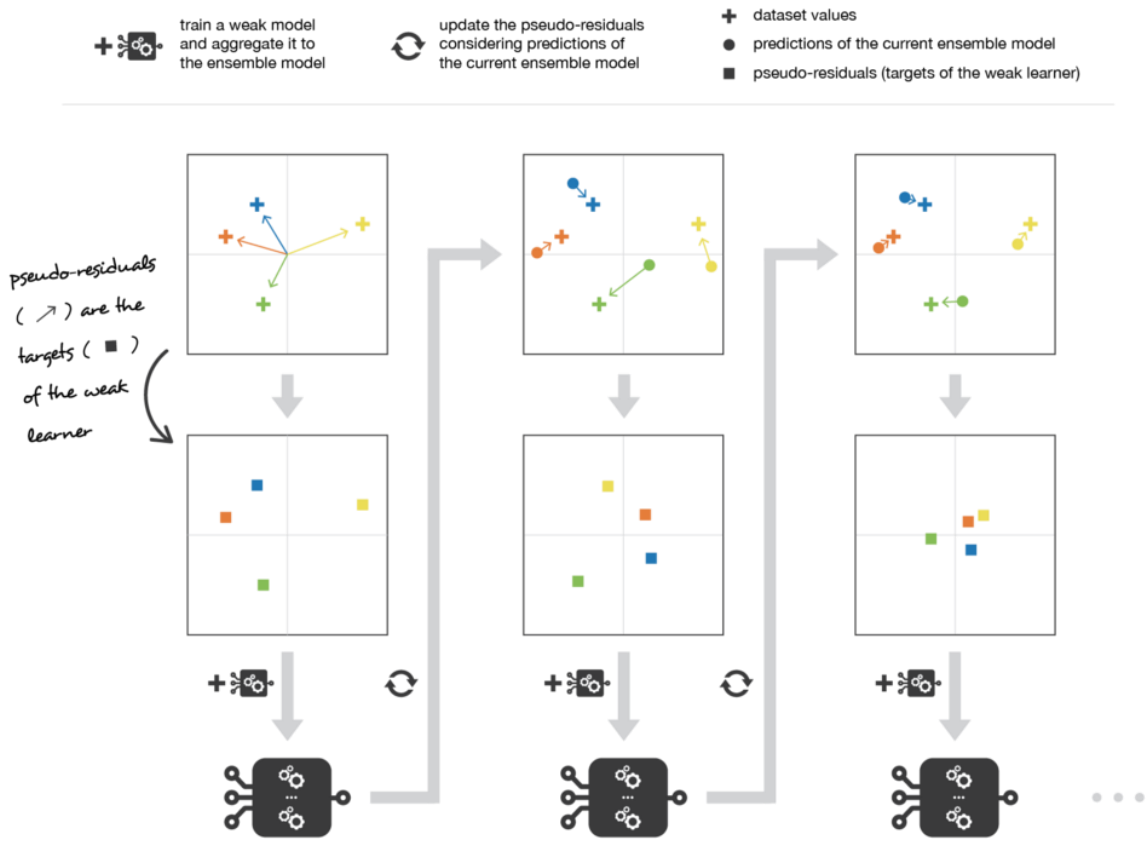
Hình 2. 11: Pseudo-Residuals Gradient Descent

Chúng ta có thể tóm tắt quá trình triển khai thuật toán như sau:

- Khởi tạo giá trị pseudo-residuals là bằng nhau cho từng điểm dữ liệu
- Tại vòng lặp thứ i
    - Train model mới được thêm vào để fit vào giá trị của pseudo-residuals đã có
    - Tính toán giá trị confidence score c_i của model vừa train

- Cập nhật model chính $W=W+ci*wiW=W+c_i*w_i$
- Cuối cùng, tính toán giá trị pseudo-residuals $-\eta\partial\partial wL(Wn-1)-\eta_{\partial w\partial}L(W_{n-1})$ để làm label cho model tiếp theo

- Sau đó lặp lại với vòng lặp i + 1.



Hình 2. 12: Gradient Boosting

2.6.3 Mô hình Linear Regresstion

"Hồi quy tuyến tính" là một phương pháp thống kê để hồi quy dữ liệu với biến phụ thuộc có giá trị liên tục trong khi các biến độc lập có thể có một trong hai giá trị liên tục hoặc là giá trị phân loại. Nói cách khác "Hồi quy tuyến tính" là một phương pháp để dự đoán biến phụ thuộc (Y) dựa trên giá trị của biến độc lập (X). Nó có thể được sử dụng cho các trường hợp chúng ta muốn dự đoán một số lượng liên tục. Ví dụ, dự đoán giao thông ở một cửa hàng bán lẻ, dự đoán thời gian người dùng dừng lại một trang nào đó hoặc số trang đã truy cập vào một website nào đó v.v...

Để bắt đầu với Hồi quy tuyến tính, chúng ta hãy đi lướt qua một số khái niệm toán học về thống kê.

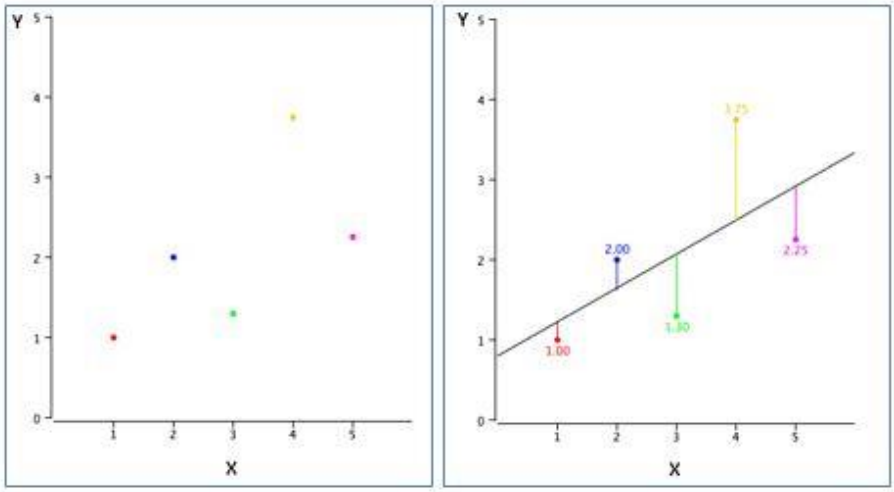
- Tương quan (r) - Giải thích mối quan hệ giữa hai biến, giá trị có thể chạy từ -1 đến +1

- Phương sai (σ2) - Đánh giá độ phân tán trong dữ liệu của bạn
- Độ lệch chuẩn (σ) - Đánh giá độ phân tán trong dữ liệu của bạn (căn bậc hai của phương sai)
- Phân phối chuẩn
- Sai số (lỗi) - {giá trị thực tế - giá trị dự đoán}

Không một kích thước nào phù hợp cho tất cả, điều này cũng đúng đối với Hồi quy tuyến tính. Để thoả mãn hồi quy tuyến tính, dữ liệu nên thoả mãn một vài giả định quan trọng. Nếu dữ liệu của bạn không làm theo các giả định, kết quả của bạn có thể sai cũng như gây hiểu nhầm.

1. Tuyến tính & Thêm vào : Nên có một mối quan hệ tuyến tính giữa biến độc lập và biến không độc lập và ảnh hưởng của sự thay đổi trong giá trị của các biến độc lập nên ảnh hưởng thêm vào tới các biến phụ thuộc.
2. Tính bình thường của phân bổ các lỗi : Sự phân bổ sai khác giữa các giá trị thực và giá trị dự đoán (sai số) nên được phân bổ một cách bình thường.
3. Sự tương đồng: Phương sai của các lỗi nên là một giá trị không đổi so với ,
    - Thời gian
    - Dự đoán
    - Giá trị của các biến độc lập
4. Sự độc lập về thống kê của các lỗi: Các sai số (dư) không nên có bất kỳ mối tương quan nào giữa chúng. Ví dụ: Trong trường hợp dữ liệu theo chuỗi thời gian, không nên có sự tương quan giữa các sai số liên tiếp nhau.

Trong khi sử dụng hồi quy tuyến tính, mục tiêu của chúng ta là để làm sao một đường thẳng có thể tạo được sự phân bố gần nhất với hầu hết các điểm. Do đó làm giảm khoảng cách (sai số) của các điểm dữ liệu cho đến đường đó.



Hình 2. 13:Đường hồi quy tuyến tính

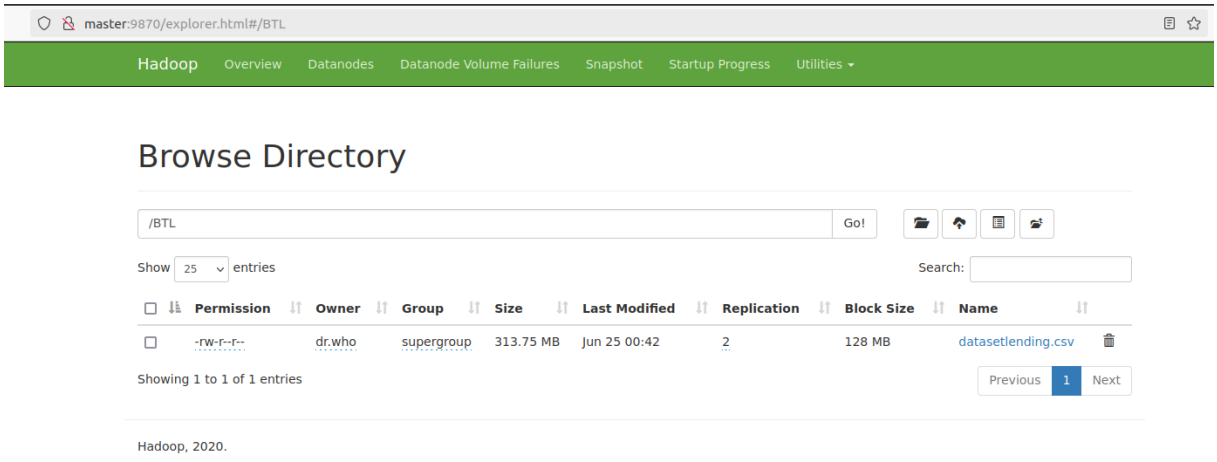
CHƯƠNG 3: CHI TIẾT THỰC HIỆN

3.1 Thu thập dữ liệu

3.1.1 Thu thập dữ liệu

Dữ liệu được thu thập trên trang Kaggle: Cerebral Stroke Prediction-Imbalanced Dataset (kaggle.com)

Thông tin về dữ liệu cần để phục vụ cho việc hỗ dự đoán mức tiêu thụ nhiên liệu ở ô tô: Dữ liệu ban đầu của 10734656 samples và 11 colums.



Hình 3. 1: Dữ liệu ban đầu được đưa lên hadoop

3.2 Phân tích dữ liệu thu thập

3.2.1 Mô tả dữ liệu

- Mô tả các trường dữ liệu ban đầu

Bảng 3. 1: Mô tả các trường dữ liệu

STT	Thông tin	Tên cột	Mô tả thông tin
1.	r	1.0	Phạm vi, có thể chỉ ra quãng đường lái xe tổng thể của phương tiện.
2.	m (kg)	2.0	Khối lượng của phương tiện tính bằng kilogram.
3.	Mt	3.0	Lượng CO2 phát thải tính bằng tấn.
4.	Ewltp (g/km)	4.0	Lượng CO2 phát thải đo được theo Quy trình Kiểm tra Xe Cơ giới Nhẹ Toàn cầu

STT	Thông tin	Tên cột	Mô tả thông tin
			(WLTP) tính bằng gram trên mỗi kilomet.
5.	Ft	5.0	Loại nhiên liệu sử dụng bởi phương tiện (ví dụ: xăng, diesel, điện)..
6.	Fm	6.0	Thành phần của hỗn hợp nhiên liệu sử dụng bởi phương tiện.
7.	ec (cm^3)	7.0	Dung tích động cơ tính bằng centimet khối.
8.	ep (KW)	8.0	Công suất động cơ tính bằng kilowatt..
9.	z (Wh/km)	9.0	Lượng năng lượng tiêu thụ tính bằng watt-giờ trên mỗi kilomet.
10.	Erwltp (g/km)	10.0	Giảm lượng CO2 phát thải tính bằng gram trên mỗi kilomet đo được theo WLTP.
11.	Fuel consumption	11.0	Mức tiêu thụ nhiên liệu của phương tiện.
12.	Electric range (km)	12.0	Quãng đường tối đa mà phương tiện có thể di chuyển chỉ bằng điện.

- Xem cấu trúc Dữ liệu: Sử dụng hàm ***data.show(), data.prinschema()*** và ***data.describe()***

r	m (kg)	Mt	Ewltp (g/km)	Ft	Fm	ec (cm3)	ep (KW)	z (Wh/km)	Erwltp (g/km)	Fuel consumption	Electric range (km)
1	1337.0	1446.0	126.0	lpg	B	999.0	74.0	NULL	1.7	7.8	NULL
1	1670.0	1782.0	125.0	petrol	H	2487.0	131.0	NULL	0.8	5.5	NULL
1	2044.0	2187.0	0.0	electric	E	NULL	221.0	172.0	NULL	NULL	440.0
1	1493.0	1576.0	135.0	petrol	M	1199.0	96.0	NULL	2.0	6.0	NULL
1	1649.0	1814.0	131.0	petrol	H	1598.0	132.0	NULL	0.59	5.8	NULL

Hình 3. 2: Tổng quan về dữ liệu

```
root
 |-- r: integer (nullable = true)
 |-- m (kg): double (nullable = true)
 |-- Mt: double (nullable = true)
 |-- Ewltp (g/km): double (nullable = true)
 |-- Ft: string (nullable = true)
 |-- Fm: string (nullable = true)
 |-- ec (cm3): double (nullable = true)
 |-- ep (KW): double (nullable = true)
 |-- z (Wh/km): double (nullable = true)
 |-- Erwltp (g/km): double (nullable = true)
 |-- Fuel consumption : double (nullable = true)
 |-- Electric range (km): double (nullable = true)
```

Hình 3. 3:Thông tin tập dữ liệu

3.2.2 Tiền xử lý dữ liệu và trực quan hóa

Như trình bày ở trên dữ liệu đầu vào của bài toán mục tiêu của chúng ta sẽ là dự đoán mức tiêu thụ nhiên liệu của xe hơi. Các thông tin này sẽ giúp chúng ta xây dựng một mô hình dự đoán có thể dự đoán mức tiêu thụ nhiên liệu của xe hơiQuá trình này không chỉ giúp tăng khả năng dự đoán mức tiêu thụ nhiên liệu cho xe mà còn tối ưu hóa việc sử dụng nhiên liệu và giảm thiểu tác động môi trường. Để có thể cài đặt được thuật toán, cài đặt được chương trình chúng ta cần xử lí dữ liệu đầu vào

- Lấy ra cột "fuel consumption" làm mục tiêu dự đoán, đây là cột đại diện cho mức tiêu thụ nhiên liệu của xe.
- Làm sạch dữ liệu dưới hình thức tìm và xử lý các giá trị null trong các hàng.
- Xem vào phân phối của các thuộc tính liên tục và nếu chúng không phân phối bình thường thì xử lý nó.
- Thực hiện một số phân tích thống kê trên dữ liệu để có được một số suy luận
- Chuyển đổi biến phân loại của kiểu dữ liệu đối tượng thành kiểu float.
- Chuẩn hóa biến liên tục bằng cách sử dụng StandardScaler.
-Tạo các pipeline

```
[Stage 3:=============================================================>  (48 + 2) / 50]
+---+------+------+------------+---+---+--------+-------+---------+-------------+-----------------+-------------------+
|  r|m (kg)|    Mt|Ewltp (g/km)| Ft| Fm|ec (cm3)|ep (KW)|z (Wh/km)|Erwltp (g/km)|Fuel consumption |Electric range (km)|
+---+------+------+------------+---+---+--------+-------+---------+-------------+-----------------+-------------------+
|  0|   425|523557|       20891|  0|  1| 1670374|  35372|  8374652|      4966748|          3183164|            8398594|
+---+------+------+------------+---+---+--------+-------+---------+-------------+-----------------+-------------------+
```

Hình 3. 4: Kiểm tra các dữ liệu null trong từng cột

- Ta thấy trong các cột có dữ liệu null, vì vậy ta xử lý các giá trị null bằng cách điền giá trị 0 vào.

-Loại bỏ các cột không phục vụ cho bào toán dự đoán mức tiêu thụ nhiên liệu.

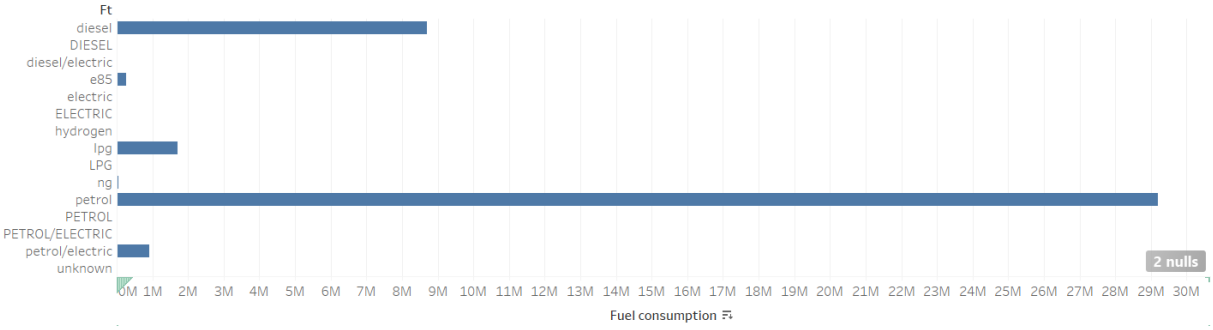
Bảng 3. 2:Các cột loại bỏ

STT	Thông tin
1.	r
2.	Mt
3.	Ewltp (g/km)
4.	z (Wh/km)
5.	Erwltp (g/km)

Bảng 3. 3: Mô tả các dữ liệu trong cột Ft

STT	Thông tin	Tên cột	Mô tả thông tin
1.	petrol	1.0	Xe sử dụng **xăng** (petrol) làm nhiên liệu chính.
2.	ng	2.0	Xe sử dụng **khí tự nhiên** (Natural Gas). Đây là loại nhiên liệu thay thế, ít phát thải hơn so với xăng và diesel.
3.	diesel/electric	3.0	Xe sử dụng **diesel và điện** (hybrid). Là xe hybrid kết hợp động cơ diesel với động cơ điện để tối ưu hóa hiệu quả nhiên liệu.
4.	e85	4.0	Xe sử dụng **E85**, một loại nhiên liệu sinh học chứa 85% ethanol và 15% xăng. Đây là một dạng nhiên liệu thay thế cho xăng.
5.	lpg	5.0	Xe sử dụng **LPG** (Liquefied Petroleum

STT	Thông tin	Tên cột	Mô tả thông tin
			Gas – khí hóa lỏng dầu mỏ), là một loại nhiên liệu thay thế, phổ biến trong xe ô tô.
6.	diesel	7.0	Loại nhiên liệu **diesel**.
7.	LPG	9.0	Xe sử dụng **LPG** (Liquefied Petroleum Gas – khí hóa lỏng dầu mỏ)

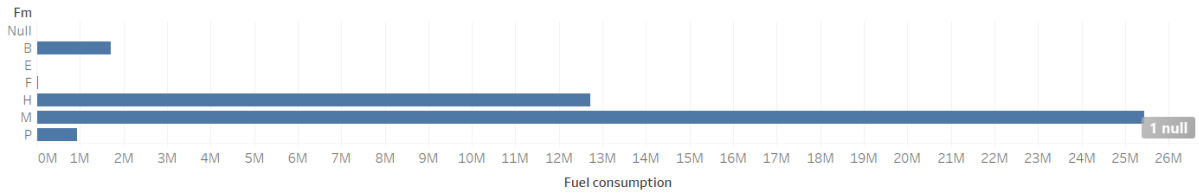


Hình 3. 5: Phân bố dữ liệu trong cột Ft

Bảng 3. 4: Mô tả thông tin các giá trị trong cột Fm

STT	Thông tin	Tên cột	Mô tả thông tin
1.	F	1.0	**Fuel** – Nhiên liệu chính được sử dụng bởi phương tiện (có thể là xăng, diesel, hoặc nhiên liệu thay thế khác).
2.	B	2.0	**Biofuel** – Phương tiện sử dụng **nhiên liệu sinh học** (biofuel), như ethanol hoặc biodiesel, làm nguồn nhiên liệu.
3.	P	3.0	**Plug-in** – Phương tiện có khả năng **sạc điện** từ nguồn ngoài (điện), thường gặp ở xe **hybrid** hoặc **xe điện thuần túy** (plug-in hybrid or electric vehicles).
4.	M	4.0	**Mixed** – Phương tiện sử dụng **nhiều loại nhiên liệu** (mixed fuels), ví dụ như kết hợp xăng và điện hoặc diesel và điện.

STT	Thông tin	Tên cột	Mô tả thông tin
5.	H	5.0	**Hybrid** – Phương tiện là **hybrid**, sử dụng **kết hợp giữa động cơ xăng/diesel và động cơ điện** để giảm mức tiêu thụ nhiên liệu và khí thải.



Hình 3. 6:Phân bố dữ liệu trong cột Fm

Chuyển đổi các giá trị phân loại trong cột Ft và Fm thành chỉ số số nguyên qua StringIndexer, sau đó mã hóa chúng thành dạng nhị phân bằng OneHotEncoder. Cuối cùng, các đặc trưng số và đã mã hóa được kết hợp thành một vector duy nhất bằng VectorAssembler, sẵn sàng cho việc huấn luyện mô hình.

```
# Mã hóa cột phân loại
indexer_Ft = StringIndexer(inputCol="Ft", outputCol="Ft_index")
indexer_Fm = StringIndexer(inputCol="Fm", outputCol="Fm_index")
encoder_Ft = OneHotEncoder(inputCol="Ft_index", outputCol="Ft_encoded")
encoder_Fm = OneHotEncoder(inputCol="Fm_index", outputCol="Fm_encoded")

# Tạo VectorAssembler cho các đặc trưng trước khi scale
assembler = VectorAssembler(
    inputCols=["m (kg)", "Ft_encoded", "Fm_encoded", "ec (cm3)", "ep (KW)", "Electric range (km)"],
    outputCol="assembled_features"
)
```

Hình 3. 7:OneHot 2 cột Ft và Fm

3.3 Xây dựng mô hình học máy

3.3.1 Xây dựng mô hình

Để thực hiện thực nghiệm đánh giá các mô hình, em sử dụng ngôn ngữ Python chạy trên môi trường Ubuntu để cài đặt chương trình. Đối với bài toán dự đoán mức tiêu thụ nhiên liệu, mô hình dự đoán được xây dựng dựa trên ba thuật toán phân lớp đó là:

RF, LR, GBT. Tất cả các thuật toán này được sử dụng trong thư viện PySpark của Apache Spark, một công cụ mạnh mẽ và phổ biến trong lĩnh vực Big Data.

3.4 Phân tích kết quả thực nghiệm

3.4.1 Chuẩn bị thực nghiệm

Bảng 3. 5: Mô tả các trường dữ liệu

Train	Test
80%	20%

-Tạo 3 pipeline cho 3 mô hình:

```
# 1. Mô hình Linear Regression
lr = LinearRegression(featuresCol="features", labelCol="Fuel consumption", regParam=0.1)
pipeline_lr = Pipeline(stages=[indexer_Ft, indexer_Fm, encoder_Ft, encoder_Fm, assembler, scaler, lr])
```

Hình 3. 8:PipeLine mô hình Linear Regression

```
# 2. Mô hình Random Forest Regressor
rf = RandomForestRegressor(featuresCol="features", labelCol="Fuel consumption", numTrees=50, maxDepth=10)

pipeline_rf = Pipeline(stages=[indexer_Ft, indexer_Fm, encoder_Ft, encoder_Fm, assembler, scaler, rf])
```

Hình 3. 9:PipeLine mô hình Random Forest

```
# 3. Mô hình Gradient Boosting Regressor
gbt = GBTRegressor(featuresCol="features", labelCol="Fuel consumption")
pipeline_gbt = Pipeline(stages=[indexer_Ft, indexer_Fm, encoder_Ft, encoder_Fm, assembler, scaler, gbt])
```

Hình 3. 10:PipeLine mô hình Gradient Boosting

3.4.2 Tiến hành thực nghiệm

Dựa vào ta tiến hành thực nghiệm với 3 mô hình huấn luyện khác nhau, cụ thể:

- Mô hình 1: Hồi quy tuyến tính (Linear Regression)
- Mô hình 2: Rừng ngẫu nhiên (Random Forest)
- Mô hình 3: Gradient Boosting

Sau đó, thực hiện tinh chỉnh siêu tham số (hyperparameter tuning) để tối ưu hóa hiệu suất của các mô hình, nhằm đạt được kết quả dự đoán chính xác hơn.

3.4.3 Kết quả thực nghiệm

Bảng 3. 6: Kết quả thực nghiệm của 3 mô hình

Thuật toán	RMSE	MAE	R^2
LR	0.64	0.45	0.87
RF	0.45	0.29	0.93
GBT	0.44	0.28	0.94

-Sau đấy ta thực hiện tinh chỉnh 3 mô hình:

Bảng 3. 7: Mô tả các tham số của LR

Parameter	Values
regParam	[0.1, 0.01, 0.001, 0.0001]
elasticNetParam	[0.0, 0.5, 1.0]
maxIter	[10, 20]
tol	[1e-6, 1e-4]

Bảng 3. 8: Kết quả của LR sau tinh chỉnh

LR						
RMSE	MAE	R^2	Best regParam	Best elasticNetParam	Best maxIter	Best tol
0.63	0.45	0.87	0.0001	0.0	10	0.0001

Bảng 3. 9: Mô tả các tham số của RF

Parameter	Values
numTrees	[50]
maxDepth	[10]

Bảng 3. 10: Kết quả của RF sau tinh chỉnh

RF				
RMSE	MAE	R^2	Best numTrees	Best maxDepth
0.44	0.29	0.94	50	10

Bảng 3. 11: Mô tả các tham số của GBT

Parameter	Values
maxIter	[10, 20]
maxDepth	[5]
stepSize	[0.1, 0.2]

Bảng 3. 12: Kết quả GBT sau tinh chỉnh

GBT					
RMSE	MAE	R^2	Best numTrees	Best maxDepth	Best
0.40	0.25	0.94	20	5	0.2

3.5 Kết chương

Trong chương này, em đã tiến hành những công việc cụ thể để áp dụng công nghệ Big Data để dự đoán mức tiêu thụ nhiên liệu xe:

Thứ nhất, em đã thu thập dữ liệu, phân tích và xử lý để chuẩn bị cho quá trình huấn luyện mô hình.

Thứ hai, em đã triển khai các mô hình Random Forest (RF), Linear Regression (LR) và Gradient-Boosting (GBT) trong môi trường Big Data sử dụng công nghệ PySpark.

CHƯƠNG 4: XÂY DỰNG ỨNG DỤNG

4.1 Triển khai các chức năng nghiệp vụ

Trang dự đoán mức tiêu thụ nhiên liệu ở ô tô (Fuel Consumption Prediction) bao gồm:

Bảng 4. 1: Mô tả dữ liệu đầu vào của trang web

m (kg)	1.0	Khối lượng của phương tiện tính bằng kilogram.
Ft	2.0	Loại nhiên liệu sử dụng bởi phương tiện (ví dụ: xăng, diesel, điện)..
Fm	3.0	Thành phần của hỗn hợp nhiên liệu sử dụng bởi phương tiện.
ec (cm^3)	4.0	Dung tích động cơ tính bằng centimet khối.
ep (KW)	5.0	Công suất động cơ tính bằng kilowatt..
Electric range (km)	6.0	Quãng đường tối đa mà phương tiện có thể di chuyển chỉ bằng điện.

-Trong đó các giá trị của Ft bao gồm:

Bảng 4. 2: Mô tả các giá trị nhập vào của Ft

STT	Thông tin	Tên cột	Mô tả thông tin
1.	petrol	1.0	Xe sử dụng **xăng** (petrol) làm nhiên liệu chính.
2.	ng	2.0	Xe sử dụng **khí tự nhiên** (Natural Gas). Đây là loại nhiên liệu thay thế, ít phát thải hơn so với xăng và diesel.
3.	diesel-electric	3.0	Xe sử dụng **diesel và điện** (hybrid). Là xe hybrid kết hợp động cơ diesel với động cơ điện để tối ưu hóa hiệu quả nhiên liệu.
4.	e85	4.0	Xe sử dụng **E85**, một loại nhiên liệu sinh học chứa 85% ethanol và 15%

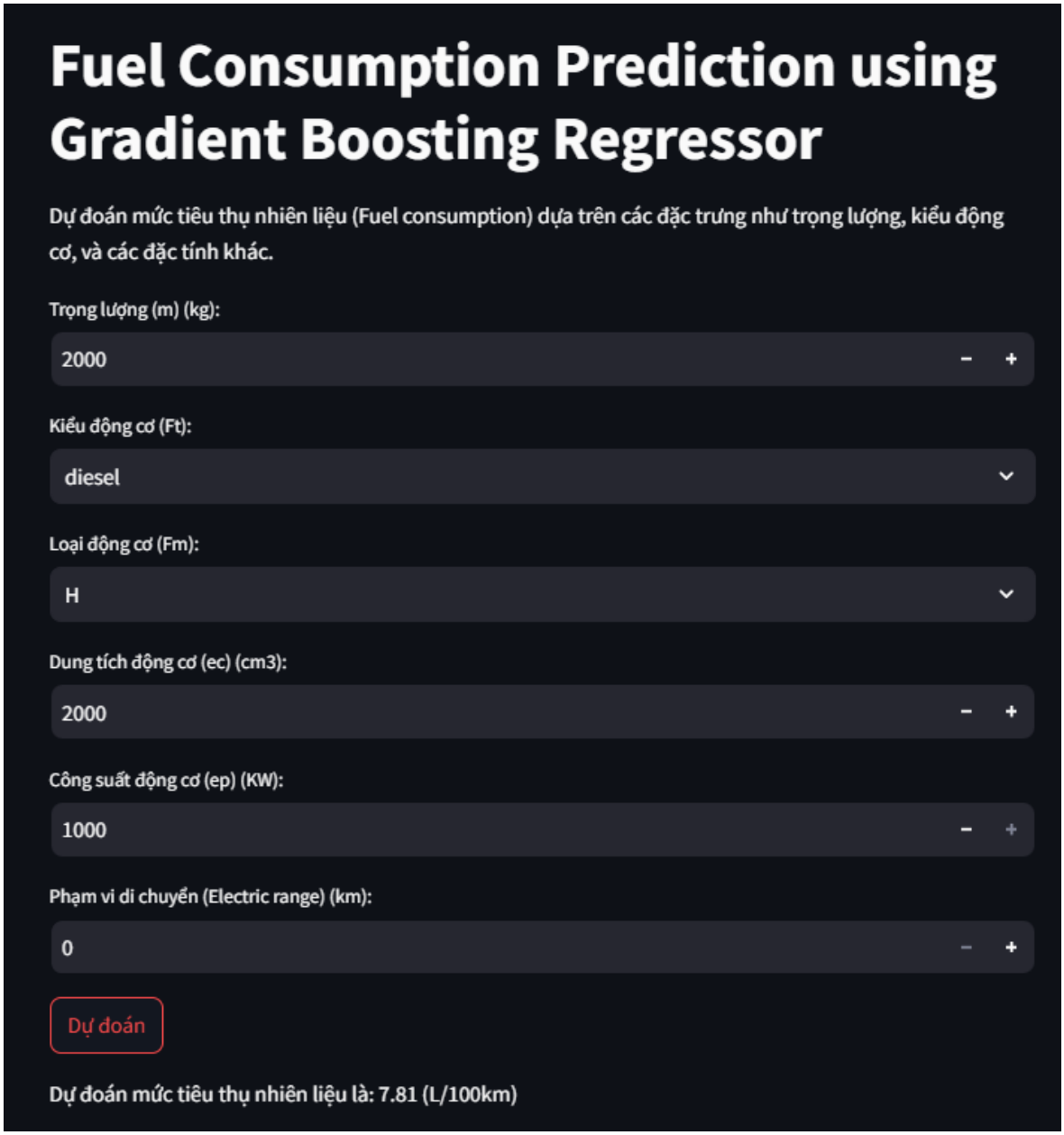
STT	Thông tin	Tên cột	Mô tả thông tin
			xăng. Đây là một dạng nhiên liệu thay thế cho xăng.
5.	lpg	5.0	Xe sử dụng **LPG** (Liquefied Petroleum Gas – khí hóa lỏng dầu mỏ), là một loại nhiên liệu thay thế, phổ biến trong xe ô tô.
6.	diesel	6.0	Loại nhiên liệu **diesel**.
7.	lgp	7.0	Xe sử dụng **LPG** (Liquefied Petroleum Gas – khí hóa lỏng dầu mỏ)

-Các giá trị của cột Fm bao gồm:

Bảng 4. 3: Mô tả các giá trị đầu vào của Fm

STT	Thông tin	Tên cột	Mô tả thông tin
1.	F	1.0	**Fuel** – Nhiên liệu chính được sử dụng bởi phương tiện (có thể là xăng, diesel, hoặc nhiên liệu thay thế khác).
2.	B	2.0	**Biofuel** – Phương tiện sử dụng **nhiên liệu sinh học** (biofuel), như ethanol hoặc biodiesel, làm nguồn nhiên liệu.
3.	P	3.0	**Plug-in** – Phương tiện có khả năng **sạc điện** từ nguồn ngoài (điện), thường gặp ở xe **hybrid** hoặc **xe điện thuần túy** (plug-in hybrid or electric vehicles).
4.	M	4.0	**Mixed** – Phương tiện sử dụng **nhiều loại nhiên liệu** (mixed fuels), ví dụ như kết hợp xăng và điện hoặc diesel và điện.
5.	H	5.0	**Hybrid** – Phương tiện là **hybrid**, sử dụng **kết hợp giữa động cơ xăng/diesel và động cơ điện** để giảm

STT	Thông tin	Tên cột	Mô tả thông tin
			mức tiêu thụ nhiên liệu và khí thải.



Hình 4. 1: Trang Demo

KẾT LUẬN

Kết quả đạt được

Nắm bắt được các kiến thức trong học phần "Khai phá dữ liệu lớn".

Biết sử dụng các framework để giải quyết cho các bài toán cơ bản.

Xử lý dữ liệu: Thành thạo các kỹ thuật làm sạch dữ liệu, xử lý giá trị thiếu, và chuẩn bị dữ liệu cho mô hình học máy.

Tiền xử lý dữ liệu:Áp dụng các phương pháp như mã hóa nhãn (onehot encoding), Biết cách viết các Pipeline.

Hiểu biết về huấn luyện mô hình và thực nghiệm: Biết cách chia dữ liệu thành tập huấn luyện và tập kiểm tra, huấn luyện mô hình học máy và đánh giá hiệu suất của mô hình.

Chuẩn hóa dữ liệu: Sử dụng các kỹ thuật như StandarScaler để chuẩn hóa dữ liệu, đảm bảo mô hình hoạt động hiệu quả hơn.

Công việc đã hoàn thành: Xây dựng được demo bằng streamlit để dự đoán về mức tiêu thụ nhiên liệu ở ô tô.

Hạn chế của đề tài

Hạn chế về kiến thức:

- Do hạn chế về kiến thức, mô hình chưa được tối ưu.

Triển khai mô hình:

- Mới chỉ triển khai được mô hình lên Streamlit, chưa có giải pháp triển khai toàn diện.

Hướng phát triển:

- **Tối ưu hóa tham số (Hyperparameter Tuning)**: Sử dụng các kỹ thuật như Grid Search, Random Search, hoặc Bayesian Optimization để tìm ra bộ tham số tối ưu cho mô hình.
- **Kỹ thuật tiền xử lý nâng cao**: Tìm hiểu và áp dụng các kỹ thuật tiền xử lý như Feature Engineering, xử lý outliers và khắc phục vấn đề imbalance trong dữ liệu (nếu cần).

Triển khai mô hình chuyên nghiệp:

- Triển khai trên các nền tảng cloud: Sử dụng các dịch vụ như AWS, GCP hoặc Azure để triển khai mô hình, đảm bảo khả năng mở rộng và độ tin cậy.
- Tích hợp CI/CD: Thiết lập các pipeline CI/CD để tự động hóa quá trình triển khai và cập nhật mô hình.

Phân tích và giám sát mô hình:

- Theo dõi hiệu suất mô hình: Sử dụng các công cụ như TensorBoard, MLflow hoặc các dịch vụ giám sát khác để theo dõi hiệu suất của mô hình theo thời gian.
- Phân tích lỗi (Error Analysis): Thực hiện phân tích lỗi để hiểu rõ nguyên nhân các dự đoán sai và cải thiện mô hình.

TÀI LIỆU THAM KHẢO

[1] https://viblo.asia/p/linear-regression-hoi-quy-tuyen-tinh-trong-machine-learning-4P856akRlY3

[2] https://viblo.asia/p/gradient-boosting-tat-tan-tat-ve-thuat-toan-manh-me-nhat-trong-machine-learning-YWOZrN7vZQ0

[3] https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/