



TIỂU LUẬN

MÔN PHÂN TÍCH DỮ LIỆU LỚN

**Đề tài: “Xây dựng ứng dụng học máy phân cụm K-Means dùng thư viện
MLlib trong Spark để xử lý dữ liệu trên hệ thống dữ liệu phân tán
HDFS”**

Giảng viên hướng dẫn : T.S Trần Thị Xuân

Thành viên nhóm : Nguyễn Thái Học

Lý Quốc Huy

Hoàng Văn Thảo

Trịnh Việt Hoàng

Nhóm thực hiện : Nhóm 4

Lớp : CNTT K19 CLC

Thái Nguyên, năm 2024

MỤC LỤC

PHÂN CÔNG NHIỆM VỤ	1
DANH SÁCH HÌNH VẼ	3
LỜI NÓI ĐẦU	4
1 TỔNG QUAN VỀ DỮ LIỆU LỚN VÀ APACHE SPARK	5
1.1 Tổng quan về dữ liệu lớn	5
1.1.1 Khái niệm về dữ liệu lớn	5
1.1.2 Đặc điểm của dữ liệu lớn	6
1.1.3 Phân loại dữ liệu lớn	7
1.1.4 Ứng dụng của dữ liệu lớn	8
1.1.5 Các công cụ của dữ liệu lớn	9
1.1.6 Ưu điểm và nhược điểm của dữ liệu lớn	10
1.2 Apache Spark	11
1.2.1 Khái niệm	11
1.2.2 Ứng dụng của Apache Spark	12
1.2.3 Tính năng của Apache Spark	13
1.2.4 Kiến trúc của Apache Spark	15
1.2.5 Thành phần của Apache Spark	18
2 PYSPARK VÀ THUẬT TOÁN PHÂN CỤM K-MEANS	20
2.1 Pyspark	20
2.1.1 Khái niệm	20
2.1.2 Thành phần của Pyspark	20
2.1.3 Các tính năng chính của Pyspark	21

2.2	Kỹ thuật phân cụm và thuật toán K-Means	22
2.2.1	Kỹ thuật phân cụm	22
2.2.2	Thuật toán K-Means Clustering	26
2.2.3	Phương pháp Elbow	29
2.2.4	Độ đo bóng (Silhouette)	30
3	TRIỂN KHAI CÀI ĐẶT THUẬT TOÁN K-MEANS TRÊN ỨNG DỤNG APACHE SPARK	32
3.1	Ngôn ngữ và công cụ	32
3.1.1	Ngôn ngữ	32
3.1.2	Công cụ	32
3.2	Cài đặt và triển khai ứng dụng trên Apache Spark	33
3.2.1	Tập dữ liệu	33
3.2.2	Tiền xử lý dữ liệu	34
3.2.3	Huấn luyện mô hình	35
3.3	Triển khai ứng dụng lên Apache Spark	36
3.3.1	Cài đặt Spark	36
3.3.2	Triển khai ứng dụng	37
	KẾT LUẬN	39
	TÀI LIỆU THAM KHẢO	41
	PHỤ LỤC CODE	42
	NHẬN XÉT CỦA GIẢNG VIÊN	45

PHÂN CÔNG NHIỆM VỤ

STT	Tên	Nhiệm vụ
1	Lý Quốc Huy	Nghiên cứu và thuyết trình tổng quan về ứng dụng Spark. Nghiên cứu và cài đặt Spark trên Ubuntu. Làm slides thuyết trình.
2	Hoàng Văn Thảo	Nghiên cứu và thuyết trình về tổng quan dữ liệu lớn. Làm slides thuyết trình.
3	Trịnh Việt Hoàng	Nghiên cứu và thuyết trình về Pyspark. Thực hiện tiền xử lý cho tập dữ liệu. Làm slides báo cáo.
4	Nguyễn Thái Học	Nghiên cứu và thuyết trình về kỹ thuật phân cụm, thuật toán K-Means và một số kỹ thuật chọn cụm tối ưu. Tổng hợp nội dung viết báo cáo và làm slides hoàn chỉnh. Xây dựng thuật toán K-Means và triển khai thành công trên Apache Spark.

Bảng 1: Bảng phân công nhiệm vụ

DANH SÁCH HÌNH VẼ

1.1	Dữ liệu lớn trong đời sống hiện nay	5
1.2	Đặc trưng của dữ liệu lớn	6
1.3	Apache Spark tích hợp trong Apache Hadoop	12
1.4	Ứng dụng của Apache Spark	13
1.5	Tính năng của Spark	14
1.6	Kiến trúc căn bản của Spark	15
1.7	Cụm độc lập	16
1.8	Hadoop YARN	17
1.9	Apache Mesos	17
1.10	Thành phần của Spark	19
2.1	Thành phần của Pyspark	20
2.2	Mình họa về kỹ thuật phân cụm	22
2.3	Phân cụm dựa trên mật độ	24
2.4	Phân cụm dựa trên mô hình phân phối	24
2.5	Phân cụm theo cấp bậc	25
2.6	Thuật toán K-means Clustering	26
2.7	Phương pháp Elbow	29
3.1	Giao diện Neovim	33
3.2	Tập dữ liệu Customer	33
3.3	Xử lý giá trị bị thiếu	34
3.4	Dữ liệu sau khi được tính toán chỉ số RFM	34
3.5	Kỹ thuật VectorAssembler	34
3.6	Chuẩn hóa dữ liệu	35

3.7	Tính điểm silhouette cho các cụm ngẫu nhiên	35
3.8	Mô hình dự đoán cụm cho điểm dữ liệu	35
3.9	Cài đặt các gói phụ thuộc cho Spark	36
3.10	Cài đặt Spark	36
3.11	Cấu hình môi trường cho Pyspark	36
3.12	Khởi tạo master và worker	37
3.13	Khởi tạo phiên ứng dụng Spark	37
3.14	Khởi chạy ứng dụng	38
3.15	Kết thúc ứng dụng	38

LỜI NÓI ĐẦU

Hiện nay AI là một ngành khoa học đang rất phát triển trên thế giới. Chúng ta không thể phủ nhận vai trò của những mô hình và thuật toán đang ngày càng quan trọng đối với con người và đang dần len lỏi vào cuộc sống hàng ngày của mọi người, thay đổi thói quen của bạn mà bạn có thể không nhận ra. Khi chúng ta lướt một trang web, facebook ta nhận được những banner quảng cáo đúng với những sản phẩm mà chúng ta đang cần. Bạn đến bệnh viện và kết quả của bạn nhận được có thể một phần được đóng góp từ dự đoán của thuật toán. Tất cả những điều này giúp cuộc sống của chúng ta trở nên tiện ích hơn rất nhiều. Ngày nay AI đang thu hút được sự đầu tư mạnh mẽ từ các công ty công nghệ trên toàn cầu và thậm chí nó nằm trong chiến lược cạnh tranh giữa các quốc gia. Việc công nghệ ngày càng phát triển kéo theo dữ liệu ngày càng gia tăng một cách nhanh chóng. Trong thế giới hiện nay dữ liệu là chìa khóa, tổ chức nào công ty nào có được nhiều dữ liệu hơn thì càng nắm nhiều lợi thế. Việc sở hữu những khối dữ liệu lớn cũng có rất nhiều điều đáng lo ngại vì các công cụ quản lý dữ liệu truyền thống không thể phân tích xử lý và đáp ứng kịp thời so với tốc độ dữ liệu được tạo ra mỗi ngày.

Nhận thấy tầm quan trọng của AI cũng như các ứng dụng kỹ thuật quản lý, xử lý dữ liệu lớn đối với sự phát triển của thời đại, nên nhóm chúng em đã chọn và nghiên cứu đề tài liên quan đến lĩnh vực dữ liệu lớn (Big Data) cụ thể là “Xây dựng ứng dụng học máy phân cụm K-Means dùng thư viện MLlib trong Spark để xử lý dữ liệu trên hệ thống dữ liệu phân tán HDFS” làm đề tài kết thúc học phần nhằm mục đích khai thác sâu hơn về Big Data cũng như công cụ Apache Spark để phân tích và xử lý dữ liệu.

CHƯƠNG 1: TỔNG QUAN VỀ DỮ LIỆU LỚN VÀ APACHE SPARK

1.1 Tổng quan về dữ liệu lớn

1.1.1 Khái niệm về dữ liệu lớn

Dữ liệu lớn hay còn gọi là Big Data đề cập đến các tập dữ liệu cực lớn và phức tạp không thể được xử lý hoặc phân tích hiệu quả bằng các phương pháp xử lý dữ liệu truyền thống. Chúng có các đặc trưng bởi khối lượng, tốc độ và sự đa dạng của dữ liệu và bao gồm cả dữ liệu có cấu trúc và bán cấu trúc, không cấu trúc.



Hình 1.1: Dữ liệu lớn trong đời sống hiện nay

Thuật ngữ "Big Data" thường được sử dụng để chỉ dữ liệu quá lớn hoặc phức tạp mà cơ sở dữ liệu, công cụ và ứng dụng truyền thống không thể xử lý được. Với sự ra đời của các công nghệ như điện toán đám mây, học máy và trí tuệ nhân tạo. Dữ liệu lớn đã trở thành một lĩnh vực nghiên cứu và ứng dụng ngày càng quan trọng.

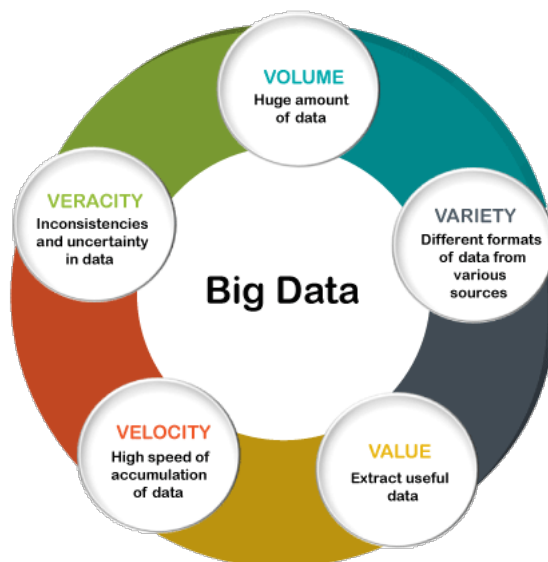
Lịch sử của dữ liệu lớn bắt đầu từ những năm 1960 và 1970, khi máy tính lần đầu tiên được giới thiệu để xử lý dữ liệu. Tuy nhiên phải đến năm 1990, thuật ngữ "Big Data" mới được đặt ra để mô tả khối lượng, sự đa dạng và tốc độ ngày càng tăng của dữ liệu được tạo ra từ nhiều nguồn khác nhau. Đầu những năm 2000, sự xuất hiện của internet và sự phổ biến của các thiết bị kỹ thuật số đã dẫn đến lượng dữ liệu được tạo và thu thập tăng mạnh.

Ngược lại điều này đã tạo ra nhu cầu về các công cụ và công nghệ mới để lưu trữ, xử lý và phân tích dữ liệu lớn. Năm 2004, Google giới thiệu một công nghệ mới có tên là Mapreduce, cho phép xử lý dữ liệu quy mô lớn trên các hệ thống phân tán bằng phần cứng phổ thông. Công nghệ này đã trở thành nền tảng của Hadoop, một nền tảng mã nguồn mở để lưu trữ và xử lý dữ liệu phân tán được phát hành vào năm 2006.

1.1.2 Đặc điểm của dữ liệu lớn

Có năm đặc điểm chính của dữ liệu lớn, thường được gọi là 5V của Big Data đó là:

- Volume (Khối lượng): Đề cập đến quy mô của dữ liệu được tạo và thu thập. Dữ liệu lớn thường liên quan đến lượng dữ liệu khổng lồ không thể xử lý dễ dàng bằng các công cụ quản lý dữ liệu truyền thống. Khối lượng dữ liệu lớn thường được đo bằng terabyte, petabyte hoặc thậm chí là exabyte.



Hình 1.2: Đặc trưng của dữ liệu lớn

- **Velocity (Vận tốc):** Đề cập đến tốc độ dữ liệu được tạo và thu thập. Dữ liệu lớn thường được tạo theo thời gian thực hoặc gần thời gian thực và đòi hỏi phải xử lý và phân tích nhanh chóng. Vận tốc đặc biệt quan trọng đối với các ứng dụng yêu cầu ra quyết định nhanh chóng, chẳng hạn như giao dịch tài chính hoặc phát hiện gian lận.
- **Variety (Đa dạng):** Đề cập đến các loại và nguồn dữ liệu khác nhau tạo nên Big Data. Dữ liệu lớn có thể bao gồm dữ liệu có cấu trúc, bán cấu trúc và phi cấu trúc cũng như các dữ liệu từ các nguồn khác nhau như phương tiện truyền thông xã hội, cảm biến và thiết bị di động. Sự đa dạng cũng đề cập đến các định dạng dữ liệu, bao gồm văn bản, âm thanh, hình ảnh và video.
- **Veracity (Xác thực):** Đề cập đến tính chính xác và độ tin cậy của dữ liệu. Dữ liệu lớn có thể có sai sót, sai lệch và không nhất quán, điều này có thể ảnh hưởng đến tính chính xác của thông tin chi tiết và việc ra quyết định. Độ chính xác đặc biệt quan trọng đối với các ứng dụng đòi hỏi độ chính xác và độ tin cậy cao, chẳng hạn như nghiên cứu khoa học và chẩn đoán y tế.
- **Values (Giá trị):** Đề cập đến việc khi chúng ta bắt đầu triển khai ứng dụng dữ liệu lớn thì chúng ta cần phải xác định được giá trị của thông tin mang lại là như thế nào.

Năm đặc điểm này của Big Data tương tác với nhau và đặt ra những thách thức đáng kể cho các tổ chức mong muốn làm việc với dữ liệu lớn. Để quản lý và phân tích dữ liệu lớn một cách hiệu quả, các tổ chức phải phát triển các chiến lược và công cụ có thể xử lý khối lượng, tốc độ, sự đa dạng và tính xác thực trong dữ liệu của họ. Điều này thường yêu cầu sử dụng các công nghệ chuyên dụng như điện toán phân tán, khai thác dữ liệu và học máy.

1.1.3 Phân loại dữ liệu lớn

Có ba loại dữ liệu lớn chính, được đặc trưng bởi loại dữ liệu và nguồn dữ liệu đó được tạo ra:

- **Dữ liệu có cấu trúc:** Là dữ liệu có tính tổ chức cao và có thể dễ dàng lưu trữ và phân tích trong cơ sở dữ liệu. Dữ liệu có cấu trúc thường bao gồm các thông tin như ngày, số và danh mục. Ví dụ về dữ liệu có cấu trúc như: dữ liệu tài chính, dữ liệu khách hàng.

- Dữ liệu phi cấu trúc: Là dữ liệu không có cấu trúc hoặc định dạng được xác định trước. Loại dữ liệu này thường do con người tạo ra và bao gồm các tệp văn bản, hình ảnh, âm thanh và video. Ví dụ như các bài đăng trên mạng xã hội, email và đánh giá của khách hàng.
- Dữ liệu bán cấu trúc: Là sự kết hợp giữa dữ liệu có cấu trúc và phi cấu trúc. Chúng có cấu trúc xác định nhưng không vừa khít với cơ sở dữ liệu truyền thống. Dữ liệu bán cấu trúc thường bao gồm siêu dữ liệu. Ví dụ như tệp XML, JSON và nhật ký web.

Ngoài các dữ liệu này, thì dữ liệu lớn cũng có thể được phân loại theo nguồn mà chúng được tạo ra như:

- Dữ liệu do máy tạo ra: Dữ liệu do máy tạo được tạo bởi cảm biến, máy móc và các hệ thống tự động khác. Ví dụ như dữ liệu từ thiết bị IoT, hệ thống GPS và thiết bị sản xuất.
- Dữ liệu do con người tạo ra: Dữ liệu này được tạo ra bởi các cá nhân thông qua tương tác của họ với các hệ thống kỹ thuật số. Ví dụ như các bài đăng trên mạng xã hội, truy vấn tìm kiếm và giao dịch trực tuyến.

Những loại dữ liệu này là điều quan trọng đối với các tổ chức muốn quản lý và phân tích hiệu quả tài sản dữ liệu của họ. Bằng cách phân loại dữ liệu theo các đặc điểm này, các tổ chức có thể phát triển các phương pháp tiếp cận có mục tiêu hơn để quản lý và phân tích dữ liệu.

1.1.4 Ứng dụng của dữ liệu lớn

Ngày nay dữ liệu lớn là thành phần quan trọng của nhiều ngành, bao gồm chăm sóc sức khỏe, tài chính, sản xuất. Sự phát triển của trí tuệ nhân tạo và học máy đã đẩy nhanh hơn nữa sự phát triển của dữ liệu lớn, vì những công nghệ này đòi hỏi khối lượng lớn dữ liệu chất lượng cao để đào tạo và cải thiện mô hình của chúng. Dữ liệu lớn có nhiều ứng dụng trong nhiều lĩnh vực khác nhau, bao gồm chăm sóc sức khỏe, tài chính, tiếp thị và khoa học:

- Sử dụng để phân tích dữ liệu bệnh nhân nhằm cải thiện kết quả chăm sóc sức khỏe

- Phát hiện gian lận trong giao dịch tài chính
- Phân tích dữ liệu khoa học để thực hiện những khám phá mới

Ngoài ra còn nhiều ứng dụng nữa đã được áp dụng dữ liệu lớn và đều cho những kết quả đáng kinh ngạc và khả quan.

1.1.5 Các công cụ của dữ liệu lớn

Có nhiều công cụ có sẵn để quản lý và phân tích dữ liệu lớn, mỗi công cụ để có điểm mạnh và điểm yếu riêng. Một số công cụ dữ liệu lớn phổ biến bao gồm:

- Apache Hadoop: Apache Hadoop là một khung phần mềm nguồn mở được sử dụng rộng rãi để lưu trữ và xử lý phân tán các bộ dữ liệu lớn. Chúng cung cấp một hệ thống có khả năng mở rộng và có khả năng chịu lỗi để lưu trữ và xử lý dữ liệu, đồng thời bao gồm một số công cụ để xử lý và phân tích dữ liệu, chẳng hạn như hệ thống tệp phân tán HDFS và Mapreduce.
- Apache Spark: Apache Spark là một công cụ xử lý dữ liệu nguồn mở được thiết kế để xử lý và phân tích dữ liệu tốc độ cao. Chúng cung cấp một công cụ phân tích thống nhất để xử lý dữ liệu, học máy và xử lý đồ thị, đồng thời hỗ trợ nhiều ngôn ngữ lập trình, bao gồm Java, Python và Scala.
- Apache Cassandra: Là một hệ thống quản lý cơ sở dữ liệu phân tán nguồn mở được thiết kế để xử lý khối lượng lớn dữ liệu lớn trên nhiều máy chủ. Cung cấp một hệ thống có khả năng mở rộng cao và có khả năng chịu lỗi để lưu trữ và truy xuất dữ liệu, đồng thời đặc biệt phù hợp với các trường hợp sử dụng yêu cầu tính sẵn sàng cao và thông lượng ghi cao.
- Cơ sở dữ liệu NoSQL: Là một loại cơ sở dữ liệu được thiết kế để xử lý dữ liệu phi cấu trúc và bán cấu trúc. Chúng cung cấp một hệ thống linh hoạt và có thể mở rộng để lưu trữ và truy xuất dữ liệu, đồng thời chúng bao gồm một số cơ sở dữ liệu phổ biến như MongoDB, Couchbase và Apache CouchDB.
- Công cụ trực quan hóa dữ liệu: Các công cụ trực quan hóa dữ liệu được sử dụng để tạo các biểu diễn trực quan của dữ liệu, chẳng hạn như biểu đồ, đồ thị và bản đồ.

Chúng cung cấp một cách hiệu quả để truyền đạt thông tin chi tiết và xu hướng cho các bên liên quan và người ra quyết định, đồng thời bao gồm các công cụ phổ biến như Tableau, D3.js và QlikView.

- Thư viện máy học: Thư viện máy học được sử dụng để phát triển và triển khai các mô hình máy học có thể dùng cho nhiều ứng dụng khác nhau, chẳng hạn như phân tích dự đoán, xử lý ngôn ngữ tự nhiên và thị giác máy tính. Các thư viện máy học phổ biến bao gồm TensorFlow, Scikit-learn và Keras.

Đây chỉ là một vài công cụ trong số rất nhiều công cụ dữ liệu lớn hiện có. Việc chọn công cụ phù hợp cho trường hợp sử dụng nhất định phụ thuộc vào một số yếu tố, chẳng hạn như kích thước và độ phức tạp của dữ liệu, khả năng phân tích hoặc xử lý mong muốn cũng như các nguồn lực và kiến thức chuyên môn sẵn có.

1.1.6 Ưu điểm và nhược điểm của dữ liệu lớn

a. Ưu điểm

Dữ liệu lớn có một số lợi thế khiến chúng trở thành tài sản quý giá cho các tổ chức trong ngành khác nhau. Một số ưu điểm của dữ liệu lớn bao gồm:

- Cải thiện việc ra quyết định: Cung cấp cho các tổ chức quyền truy cập vào lượng dữ liệu khổng lồ, cho phép họ đưa ra quyết định sáng suốt hơn và dựa trên dữ liệu. Bằng cách phân tích dữ liệu lớn, các tổ chức có thể xác định xu hướng, mô hình và thông tin chi tiết khó hoặc không thể phân biệt được từ các tập dữ liệu nhỏ hơn.
- Tăng hiệu quả và năng suất: Công nghệ dữ liệu lớn cho phép các tổ chức xử lý và phân tích dữ liệu nhanh chóng và chính xác hơn. Điều này có thể giúp các tổ chức tối ưu hóa hoạt động, giảm lãng phí và kém hiệu quả, đồng thời tăng năng suất.
- Hiểu biết sâu sắc hơn về khách hàng: Cung cấp cho các tổ chức sự hiểu biết đầy đủ và chi tiết hơn về hành vi, sở thích và nhu cầu của khách hàng. Điều này có thể giúp các tổ chức cải thiện chiến lược tiếp thị và thu hút khách hàng, dẫn đến sự hài lòng và trung thành của khách hàng cao hơn.

- Tăng cường đổi mới sản phẩm và dịch vụ: Cung cấp cho các tổ chức những hiểu biết sâu sắc về các xu hướng mới nổi, sở thích của người tiêu dùng và cơ hội thị trường, cơ thể giúp thúc đẩy và đổi mới sản phẩm và dịch vụ. Bằng cách tận dụng dữ liệu lớn, các tổ chức có thể phát triển các sản phẩm dịch vụ đáp ứng tốt hơn nhu cầu và sở thích của khách hàng.
- Tiết kiệm chi phí: Bằng cách cải thiện hiệu quả và năng suất. Dữ liệu lớn có thể giúp các tổ chức giảm chi phí và tăng lợi nhuận. Ví dụ như sử dụng để tối ưu hóa hoạt động của chuỗi cung ứng, giảm chi phí tồn kho và cải thiện việc phân bổ nguồn lực.

b. Nhược điểm

Nhìn chung lợi thế của dữ liệu lớn có thể rất đáng kể và các tổ chức quản lý và phân tích hiệu quả tài sản dữ liệu của họ có thể đạt được lợi thế cạnh tranh trong các ngành tương ứng. Tuy nhiên dữ liệu lớn cũng có nhiều nhược điểm và thách thức đáng kể bao gồm nhu cầu về chuyên môn, công cụ và cơ sở hạ tầng chuyên dụng để quản lý và phân tích các bộ dữ liệu lớn.

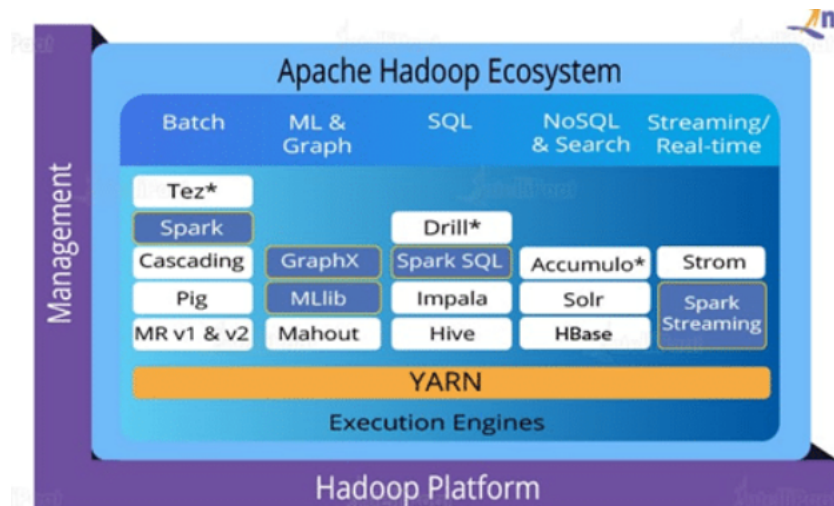
Do đó, các tổ chức mong muốn làm việc với dữ liệu lớn phải đầu tư vào cơ sở hạ tầng chuyên môn cần thiết để phân tích và rút ra những hiểu biết sâu sắc từ chúng một cách hiệu quả.

1.2 Apache Spark

1.2.1 Khái niệm

Apache Spark là một khung tính toán cụm có tốc độ cao được thiết kế để xử lý thời gian thực. Spark là một dự án nguồn mở của quỹ phần mềm Apache. Spark khắc phục những hạn chế của Hadoop Mapreduce và mở rộng mô hình Mapreduce để sử dụng hiệu quả cho việc xử lý dữ liệu.

Spark là công ty dẫn đầu thị trường về xử lý dữ liệu lớn. Chúng được sử dụng rộng rãi trong các tổ chức theo nhiều cách và vượt qua Hadoop khi chạy nhanh hơn gấp 100 lần trong bộ nhớ và 10 lần trên đĩa.



Hình 1.3: Apache Spark tích hợp trong Apache Hadoop

1.2.2 Ứng dụng của Apache Spark

Ngày nay việc triển khai rộng rãi các công cụ dữ liệu lớn. Mỗi ngày trôi qua, yêu cầu của doanh nghiệp ngày càng tăng và do đó, cần có một hình thức xử lý dữ liệu nhanh hơn và hiệu quả hơn. Hầu hết các dữ liệu trực tuyến thường là các dữ liệu phi cấu trúc, dày và nhanh liên tục. Spark hầu như đã đáp ứng được những vấn đề đó và được sử dụng thành công trong nhiều lĩnh vực, ngành nghề khác nhau:

- Ngân hàng: Spark đang càng ngày được ngành ngân hàng áp dụng. Chủ yếu được sử dụng ở đây để phát hiện gian lận tài chính với sự trợ giúp của SparkML. Các ngân hàng sử dụng Spark để xử lý việc đánh giá rủi ro tín dụng, phân khúc khách hàng và quảng cáo. Apache Spark cũng được sử dụng để phân tích hồ sơ truyền thông và xã hội, thảo luận trên diễn đàn, trò chuyện hỗ trợ khách hàng. Các phân tích dữ liệu này giúp các tổ chức đưa ra quyết định kinh doanh tốt hơn.
- Thương mại điện tử: Spark được sử dụng rộng rãi trong ngành thương mại điện tử. Được sử dụng để phân cụm dữ liệu theo thời gian thực, các doanh nghiệp có thể chia sẻ những phát hiện của mình với các nguồn dữ liệu khác để đưa ra khuyến nghị tốt hơn cho khách hàng. Hệ thống khuyến nghị chủ yếu được sử dụng trong ngành thương mại điện tử để hiển thị các xu hướng mới.
- Chăm sóc sức khỏe: Spark là một công cụ tính toán mạnh mẽ để thực hiện phân tích

nâng cao trên hồ sơ bệnh nhân. Giúp theo dõi hồ sơ sức khỏe của bệnh nhân một cách dễ dàng. Ngành chăm sóc sức khỏe sử dụng Spark để triển khai các dịch vụ nhằm thu thập thông tin chi tiết như phản hồi của bệnh nhân và dịch vụ của bệnh viện cũng như theo dõi dữ liệu y tế.



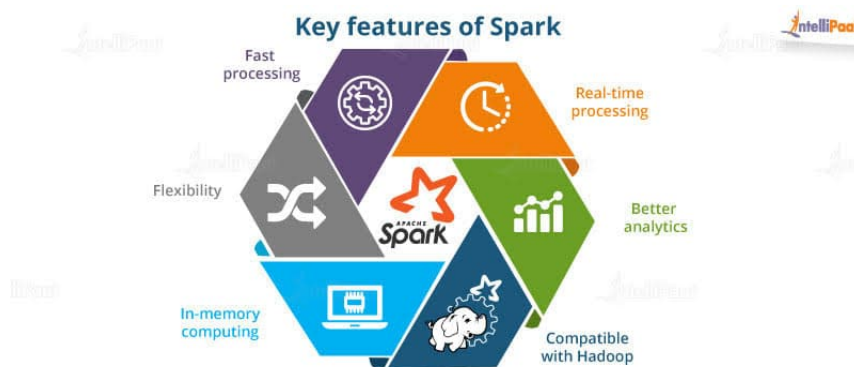
Hình 1.4: Ứng dụng của Apache Spark

- Phương tiện truyền thông: Nhiều công ty trò chơi sử dụng Spark để tìm kiếm các mẫu từ sự kiện trong trò chơi theo thời gian thực của họ. Với điều này họ có thể có thêm cơ hội kinh doanh bằng cách tùy chỉnh những thứ như tự động điều chỉnh mức độ phức tạp của trò chơi theo hiệu suất của người chơi, v.v. Một số công ty truyền thông như Yahoo, sử dụng Spark để tiếp thị có mục tiêu tùy chỉnh các trang tin tức dựa trên độc giả, sở thích, v.v. Họ sử dụng các công cụ như thuật toán Machine Learning để xác định danh mục sở thích của độc giả. Cuối cùng, họ phân loại những tin tức như vậy thành nhiều phần khác nhau và cập nhật kịp thời cho người đọc.
- Du lịch: Nhiều người hợp tác với các nhà lập kế hoạch du lịch để biến kỳ nghỉ của họ trở nên hoàn hảo và các công ty du lịch này phụ thuộc vào Spark để cung cấp các gói du lịch khác nhau. TripAdvisor là một trong những công ty sử dụng Spark để so sánh các gói du lịch khác nhau từ các nhà cung cấp khác nhau. Chúng quét hàng trăm web để tìm giá khách sạn, gói chuyển đi tốt nhất và hợp lý nhất, v.v.

1.2.3 Tính năng của Apache Spark

Được phát triển để phân tích tốc độ cao, dễ sử dụng và chuyên sâu. Mặc dù được cài đặt trên cụm Hadoop, nhưng khả năng xử lý song song của chúng cũng cho phép Spark chạy độc lập. Một số tính năng của Apache Spark nổi bật có thể kể đến như:

- **Xử lý nhanh:** Tính năng quan trọng nhất của Apache Spark đã khiến thế giới dữ liệu lớn lựa chọn công nghệ này hơn các công nghệ khác là tốc độ của chúng. Dữ liệu lớn được đặc trưng bởi khối lượng, sự đa dạng, tốc độ, giá trị và tính xác thực do đó nó cần được xử lý ở tốc độ cao hơn. Spark chứa bộ dữ liệu phân tán linh hoạt (RDD) giúp tiết kiệm thời gian thực hiện các thao tác đọc ghi do đó chạy nhanh hơn gần 10 đến 100 lần so với Hadoop.
- **Tính linh hoạt:** Hỗ trợ nhiều ngôn ngữ và cho phép các nhà phát triển viết ứng dụng bằng Java, Scala, Python, R. Được trang bị hơn 80 toán tử cao cấp, công cụ này khá phong phú về mặt toán học.



Hình 1.5: Tính năng của Spark

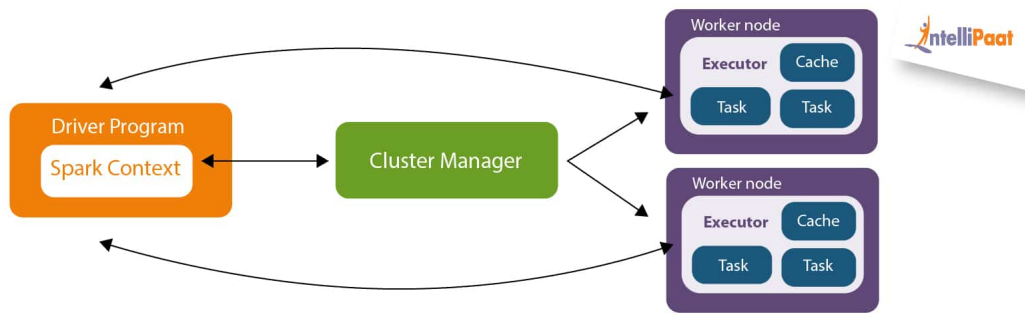
- **Điện toán trong bộ nhớ:** Spark lưu trữ dữ liệu trong RAM của máy chủ, cho phép nó truy cập dữ liệu nhanh chóng và từ đó tăng tốc độ phân tích.
- **Xử lý thời gian thực:** Có thể xử lý dữ liệu theo thời gian thực, không giống như Mapreduce xử lý dữ liệu được lưu trữ, Spark có thể xử lý dữ liệu theo thời gian thực và do đó có thể tạo ra kết quả tức thì.
- **Phân tích tốt hơn:** Ngược lại với Mapreduce bao gồm các chức năng Map và Reduce, Spark có nhiều tính năng hơn. Apache Spark bao gồm một tập hợp phong phú các truy vấn SQL, thuật toán Machine Learning, phân tích phức tạp, v.v. Với tất cả các chức năng này, phân tích dữ liệu lớn có thể được thực hiện theo cách tốt hơn.
- **Khả năng tương thích với Hadoop:** Spark không chỉ có khả năng hoạt động độc lập mà còn có thể hoạt động trên Hadoop. Không chỉ vậy, Spark còn chắc chắn tương

thích với cả hai phiên bản của hệ sinh thái Hadoop.

1.2.4 Kiến trúc của Apache Spark

a. Kiến trúc căn bản của Spark

Driver Program trong kiến trúc Apache Spark gọi chương trình chính của ứng dụng và tạo SparkContext. SparkContext bao gồm tất cả các chức năng cơ bản. SparkDriver chứa nhiều thành phần khác như DAG Scheduler, Task Scheduler, Backend Scheduler, và Block Manager, chịu trách nhiệm dịch mã do người dùng viết thành các công việc thực sự được thực thi trên cụm.



Hình 1.6: Kiến trúc căn bản của Spark

SparkDriver và SparkContext cùng nhau giám sát việc thực hiện công việc trong cụm. SparkDriver hoạt động với Cluster Manager để quản lý nhiều công việc khác. Cluster Manager thực hiện công việc phân bổ tài nguyên, sau đó công việc được chia thành nhiều nhiệm vụ nhỏ hơn và phân bổ tiếp cho các worker node.

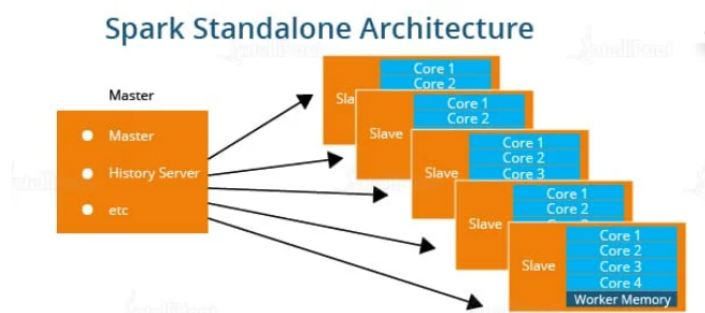
Bất cứ khi nào RDD được tạo trong SparkContext, chúng có thể được phân phối trên nhiều worker node và cũng có thể được lưu trữ ở đó. Worker node thực thi các nhiệm vụ do Cluster Manager giao và trả lại SparkContext. Executor có trách nhiệm thực hiện các nhiệm vụ này. Nếu muốn tăng hiệu suất của hệ thống, chúng ta có thể tăng số lượng công nhân để có thể chia công việc thành nhiều phần hợp lý hơn.

Spark Driver giống như buồng lái của ứng dụng Spark, thực hiện vai trò của bộ điều khiển thực thi của ứng dụng Spark. Spark Driver theo dõi tất cả các trạng thái ứng dụng cho cụm Spark. Cluster Manager phải được giao tiếp với SparkDriver để lấy tài nguyên vật lý và khởi động trình thực thi. Các tác vụ do SparkDriver giao được thực hiện bởi Executor. Trách

nhiệm cốt lõi của Executor là thực hiện các nhiệm vụ được giao, chạy chúng và báo cáo lại trạng thái cũng như kết quả thành công hay thất bại của chúng. Mỗi ứng dụng Spark có các quy trình thực thi riêng.

SparkContext có thể hoạt động với nhiều Cluster Manager khác nhau, như Standardlone Manager, YARN hoặc Mesos, phân bổ tài nguyên cho các vùng chứa trong worker node. Công việc được thực hiện trong các thùng chứa này:

- Standardlone: Cụm độc lập bao gồm một master độc lập có chức năng như trình quản lý tài nguyên, cùng với worker độc lập đóng vai trò là worker node. Trong chế độ cụm này, mỗi worker node lưu trữ một Executor duy nhất chịu trách nhiệm thực thi các tác vụ. Để bắt đầu quá trình thực thi, khách hàng sẽ kết nối với master độc lập, yêu cầu các tài nguyên cần thiết. Đóng vai trò là chủ ứng dụng, khách hàng cộng tác với trình quản lý tài nguyên để mua các tài nguyên cần thiết. Giao diện người dùng dựa trên web có thể truy cập được trong Cluster Manager, cho phép người dùng trực quan hóa chi tiết toàn diện về tất cả các cụm và thống kê công việc.

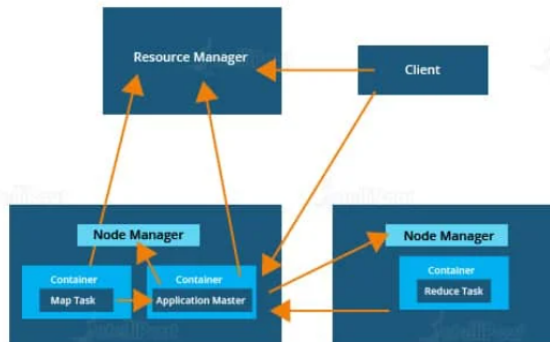


Hình 1.7: Cụm độc lập

- Hadoop YARN: YARN đảm nhiệm việc quản lý tài nguyên cho Hadoop chúng có hai thành phần là Resource Maneger (trình quản lý tài nguyên) và Node Maneger (trình quản lý nút). Trình quản lý tài nguyên quản lý tài nguyên trên tất cả các ứng dụng trong hệ thống. Bao gồm trình lập lịch biểu và trình quản lý ứng dụng. Bộ lập lịch phân bổ tài nguyên cho các ứng dụng khác nhau. Trình quản lý nút bao gồm trình quản lý ứng dụng và vùng chứa. Mỗi tác vụ của Mapreduce chạy trong một vùng chứa. Do đó, một ứng dụng hoặc công việc yêu cầu một hoặc nhiều vùng chứa và trình quản lý nút sẽ giám sát các vùng chứa này và việc sử dụng tài nguyên. Tất

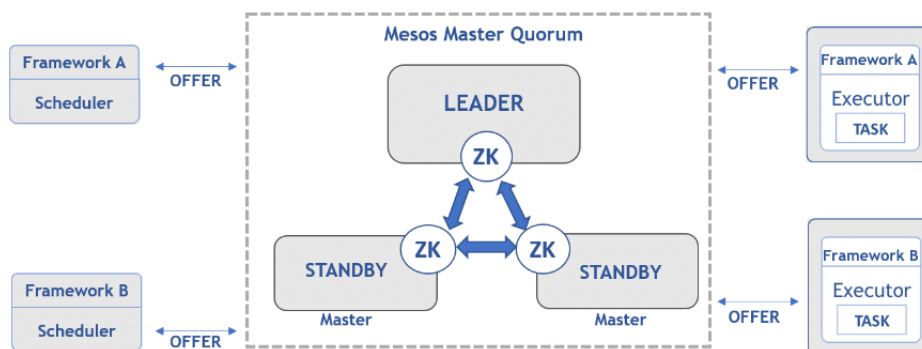
cả các điều này được ghi lại và báo cáo cho Resource Manager

YARN Architecture



Hình 1.8: Hadoop YARN

- Apache Mesos: xử lý khối công việc từ nhiều nguồn bằng cách sử dụng tính năng chia sẻ và cách ly tài nguyên động. Giúp triển khai và quản lý các ứng dụng trong môi trường cụm quy mô lớn. Apache Mesos bao gồm 3 thành phần thứ nhất là Mesos master cung cấp khả năng chịu lỗi (khả năng vận hành và phục hồi tổn thất khi xảy ra lỗi). Một cụm chứa nhiều Mesos master. Thứ hai là Mesos slave là một phiên bản cung cấp tài nguyên cho cụm. Mesos slave chỉ phân công tài nguyên khi Mesos master giao nhiệm vụ. Cuối cùng là Mesos Frameworks cho phép các ứng dụng yêu cầu tài nguyên từ cụm để ứng dụng có thể thực hiện các tác vụ.



Hình 1.9: Apache Mesos

Cluster Manager duy trì một cụm máy sẽ chạy các ứng dụng Spark. Chúng có các trình điều khiển riêng được gọi là "master" và "worker". Chúng được gắn với các máy vật lý thay

vì các quy trình như trong Spark. Khi chạy một ứng dụng Spark, Cluster Manager sẽ yêu cầu tài nguyên để chạy ứng dụng đó. Tùy thuộc vào cấu hình của ứng dụng, chúng có thể là nơi chạy SparkDriver hoặc đơn giản là tài nguyên cho người thực thi ứng dụng Spark. Trong quá trình thực thi ứng dụng Spark, Cluster Manager sẽ quản lý các máy cơ bản mà ứng dụng đang chạy.

b. Chế độ thực thi ứng dụng

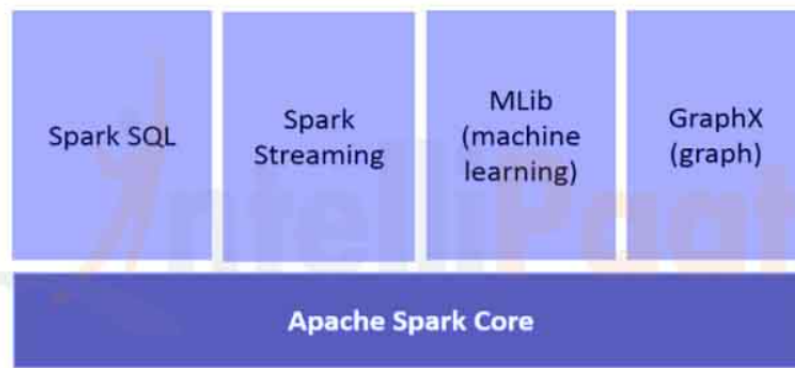
Mô hình thực thi giúp xác định vị trí thực tế của các tài nguyên được đề cập trước đó khi ứng dụng được chạy. Có ba chế độ thực hiện có thể được lựa chọn là:

- **Chế độ cụm:** Là chế độ phổ biến nhất để chạy các ứng dụng Spark trong đó tập lệnh Python, Java hoặc R được biên dịch trước và được người dùng gửi tới Cluster Manager. Sau đó, quy trình SparkDriver được Cluster Manager khởi chạy trên worker node bên trong cụm, bên cạnh các quy trình thực thi. Điều này ngụ ý rằng Cluster Manager chịu trách nhiệm duy trì tất cả các quy trình liên quan đến ứng dụng Spark.
- **Chế độ máy khách:** Gần giống như chế độ cụm, ngoại trừ SparkDriver vẫn còn trên máy khách đã được gửi ứng dụng. Điều này có nghĩa là máy khách duy trì quy trình SparkDriver và Cluster Manager duy trì quy trình thực thi. Những máy này thường được gọi là máy cổng hoặc nút biên.
- **Chế độ cục bộ:** Ở chế độ này, toàn bộ ứng dụng Spark được chạy trên một máy. Chúng quan sát sự song song thông qua các luồng trên máy đơn lẻ đó. Đây là một cách phổ biến để thử nghiệm các ứng dụng hoặc thử nghiệm sự phát triển của máy đơn. Tuy nhiên không được khuyến khích để chạy các ứng dụng trong sản xuất.

1.2.5 Thành phần của Apache Spark

Một số thành phần của Apache Spark và mỗi thành phần đều có chức năng nhiệm vụ riêng:

- **Spark Core:** Bao gồm một công cụ thực thi chung cho nền tảng Spark được xây dựng theo yêu cầu. Cung cấp các bộ dữ liệu tham chiếu và tính toán bộ nhớ tích hợp được lưu trữ trong các hệ thống lưu trữ bên ngoài.



Hình 1.10: Thành phần của Spark

- **Spark SQL:** Là một thành phần trên Spark Core giới thiệu một bộ trừu tượng hóa dữ liệu mới có tên là SchemaRDD. SchemaRDD cung cấp hỗ trợ cho cả dữ liệu có cấu trúc và bán cấu trúc.
- **Spark Streaming:** Là một trong những thành phần Apache Spark và cho phép Spark xử lý dữ liệu truyền phát theo thời gian thực. Cung cấp API để thao tác các luồng dữ liệu phù hợp với API RDD. Cho phép các lập trình viên hiểu dự án và chuyển qua các ứng dụng thao tác dữ liệu và đưa ra kết quả theo thời gian thực. Tương tự như Spark Core, Spark Streaming cố gắng làm cho hệ thống có khả năng chịu lỗi và có thể mở rộng.
- **MLlib:** Apache Spark được trang bị một thư viện phong phú có tên là MLlib. Thư viện này chứa một loạt các thuật toán học máy như: hồi quy, phân loại, bộ lọc cộng tác, v.v. Ngoài ra còn được trang bị các hàm toán học cao cấp. Tất cả các chức năng này giúp Spark mở rộng quy mô trên một cụm.
- **Graphx:** Spark cũng đi kèm với một thư viện để thao tác với biểu đồ và thực hiện các phép tính, được gọi là GraphX. Giống như Spark Streaming và Spark SQL, GraphX cũng mở rộng API Spark RDD, tạo ra biểu đồ có hướng. Chúng cũng chứa nhiều toán tử để thao tác trên đồ thị, cùng với các thuật toán đồ thị.

CHƯƠNG 2: PYSPARK VÀ THUẬT TOÁN PHÂN CỤM K-MEANS

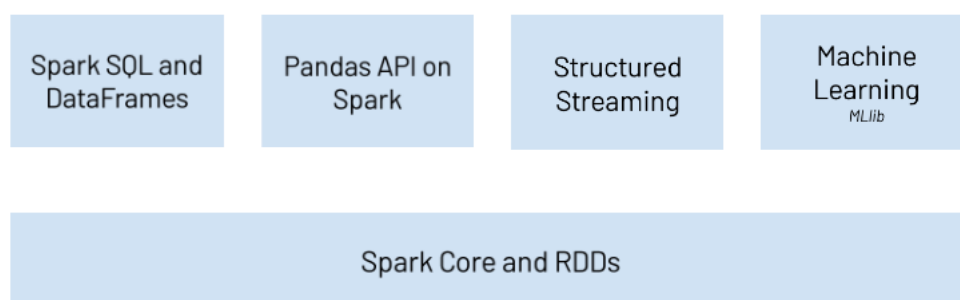
2.1 Pyspark

2.1.1 Khái niệm

Pyspark là API Python cho Apache Spark. Cho phép thực hiện xử lý dữ liệu quy mô lớn, thời gian thực trong môi trường phân tán bằng Python. Chúng cung cấp các công cụ để phân tích tương tác dữ liệu.

Pyspark kết hợp khả năng học hỏi và tính dễ sử dụng của Python với sức mạnh của Apache Spark để cho phép xử lý và phân tích dữ liệu ở mọi quy mô cho người quen thuộc với Python.

2.1.2 Thành phần của Pyspark



Hình 2.1: Thành phần của Pyspark

Pyspark hỗ trợ tất cả các tính năng của Spark như Spark SQL, DataFrames, structured Streaming, Machine Learning và Spark Core.

- **Spark SQL và DataFrames:** Spark SQL là mô-đun của Apache Spark để làm việc với dữ liệu có cấu trúc. Nó cho phép bạn kết hợp liền mạch các truy vấn SQL với các chương trình Spark. Với PySpark DataFrames, bạn có thể đọc, ghi, chuyển đổi và phân tích dữ liệu một cách hiệu quả bằng Python và SQL. Cho dù bạn sử dụng Python hay SQL, cùng một công cụ thực thi cơ bản sẽ được sử dụng để bạn luôn tận dụng toàn bộ sức mạnh của Spark.
- **Pandas API on Spark:** API Pandas trên Spark cho phép bạn mở rộng quy mô khối lượng công việc gấu trúc của mình theo bất kỳ kích thước nào bằng cách chạy nó được phân phối trên nhiều nút. Nếu bạn đã quen thuộc với gấu trúc và muốn tận dụng Spark cho dữ liệu lớn, API gấu trúc trên Spark giúp bạn làm việc hiệu quả ngay lập tức và cho phép bạn di chuyển ứng dụng của mình mà không cần sửa đổi mã. Bạn có thể có một cơ sở mã duy nhất hoạt động với cả gấu trúc (thử nghiệm, bộ dữ liệu nhỏ hơn) và với Spark (bộ dữ liệu sản xuất, phân tán), đồng thời bạn có thể chuyển đổi giữa API gấu trúc và API Pandas trên Spark một cách dễ dàng và không tốn chi phí.
- **Structured Streaming:** Là một công cụ xử lý luồng có khả năng mở rộng và có khả năng chịu lỗi được xây dựng trên công cụ Spark SQL. Bạn có thể biểu diễn phép tính phát trực tuyến giống như cách bạn biểu diễn phép tính hàng loạt trên dữ liệu tĩnh. Công cụ Spark SQL sẽ đảm nhiệm việc chạy nó tăng dần và liên tục cũng như cập nhật kết quả cuối cùng khi dữ liệu truyền phát tiếp tục đến.
- **MLlib:** Được xây dựng dựa trên Spark, MLlib là một thư viện máy học có thể mở rộng, cung cấp một bộ API cấp cao thống nhất giúp người dùng tạo và điều chỉnh các quy trình máy học thực tế.
- **Spark Core và RDD:** Spark Core là công cụ thực thi chung cơ bản cho nền tảng Spark mà tất cả các chức năng khác đều được xây dựng dựa trên đó. Nó cung cấp RDD (Bộ dữ liệu phân tán linh hoạt) và khả năng tính toán trong bộ nhớ.

2.1.3 Các tính năng chính của Pyspark

Một số tính năng nổi bật của Pyspark là:

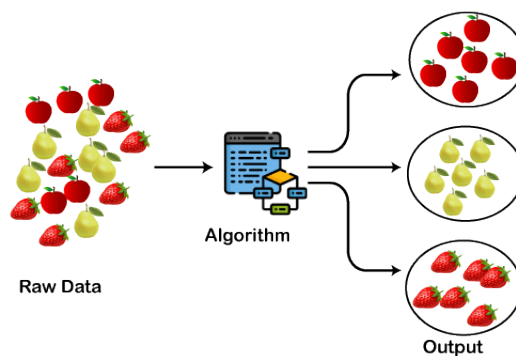
- Tính toán thời gian thực: Khung Pyspark có tính năng xử lý trong bộ nhớ giúp giảm độ trễ.
- Tính linh hoạt: Hỗ trợ nhiều ngôn ngữ khác nhau bao gồm Scala, Python, Java, Python và R, khiến chúng trở thành một trong những khung ưa thích để xử lý các tập dữ liệu khổng lồ.
- Bộ đệm và tính bền bỉ của ổ đĩa: Khung này cung cấp bộ nhớ đệm mạnh mẽ với khả năng duy trì ổ đĩa vượt trội.
- Tốc độ xử lý: Khung Pyspark cung cấp tốc độ xử lý dữ liệu lớn nhanh hơn nhiều so với các ứng dụng khác.
- Hoạt động hiệu quả với RDD: Làm việc hiệu quả và tuyệt vời với RDD.

2.2 Kỹ thuật phân cụm và thuật toán K-Means

2.2.1 Kỹ thuật phân cụm

a. Khái niệm

Phân cụm hay phân tích cụm là một kỹ thuật học máy, nhằm nhóm các tập dữ liệu khổng lồ không được gán nhãn. Một cách cụ thể hơn phân cụm là "Một cách nhóm các điểm dữ liệu thành các cụm khác nhau, bao gồm các điểm dữ liệu tương tự nhau. Các đối tượng có những điểm tương đồng có thể vẫn nằm trong một nhóm có ít hoặc không có điểm tương đồng với nhóm khác".



Hình 2.2: Minh họa về kỹ thuật phân cụm

Chúng thực hiện điều đó bằng cách tìm một số mẫu tương tự trong tập dữ liệu chưa được gán nhãn như hình dạng, kích thước, màu sắc, hành vi, v.v và phân chia chúng theo sự hiện diện và vắng mặt của các mẫu tương tự đó. Đây là một phương pháp học không giám sát, sau khi áp dụng kỹ thuật phân cụm này, mỗi cụm hoặc nhóm được cung cấp ID cụm. Hệ thống ML có thể sử dụng id này để đơn giản hóa việc xử lý các tập dữ liệu lớn và phức tạp.

b. Ứng dụng của kỹ thuật phân cụm

Kỹ thuật phân cụm có thể được sử dụng rộng rãi trong nhiều nhiệm vụ khác nhau. Một số lĩnh vực sử dụng phổ biến nhất kỹ thuật này là:

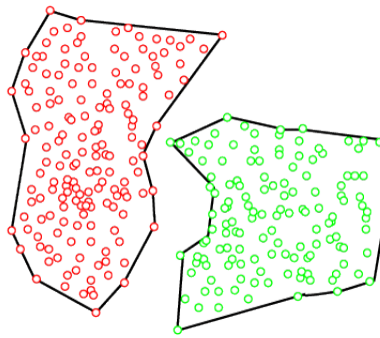
- Phân khúc thị trường
- Phân tích dữ liệu thống kê
- Phân tích mạng xã hội
- Phân đoạn hình ảnh
- Phát hiện bất thường

c. Các loại phương pháp phân cụm

Các phương pháp phân cụm được chia thành Hard Clustering (phân cụm cứng) điểm dữ liệu chỉ thuộc một nhóm và Soft Clustering (phân cụm mềm) điểm dữ liệu cũng có thể thuộc về một nhóm khác. Một số phương pháp phân cụm chính được sử dụng trong Machine Learning:

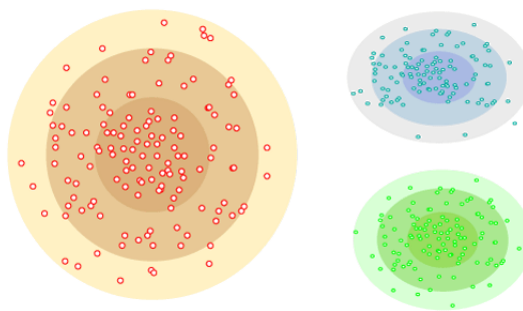
- Phân cụm phân vùng: Đây là một kiểu phân cụm chia dữ liệu thành các nhóm không phân cấp. Đây còn được gọi là phương pháp dựa trên centroid (tâm). Ở loại này, tập dữ liệu được chia thành một tập k nhóm, trong đó k sử dụng để xác định số lượng nhóm được xác định trước. Trung tâm cụm được tạo theo cách sao cho khoảng cách giữa các điểm dữ liệu của một cụm là tối thiểu so với tâm cụm khác.
- Phân cụm dựa trên mật độ: Phương pháp phân cụm dựa trên mật độ là kết nối các khu vực có mật độ cao thành các cụm và các phân bố có hình dạng tùy ý được hình thành miễn là khu vực dày đặc và có thể được kết nối. Thuật toán này thực hiện điều

đó bằng cách xác định các tâm cụm khác nhau trong tập dữ liệu và kết nối các khu vực có mật độ cao thành các cụm. Các vùng dày đặc trong không gian dữ liệu được phân chia với nhau bằng các vùng thưa thớt hơn. Các thuật toán này có thể gặp khó khăn trong việc phân cụm các điểm dữ liệu nếu tập dữ liệu có mật độ và kích thước khác nhau.



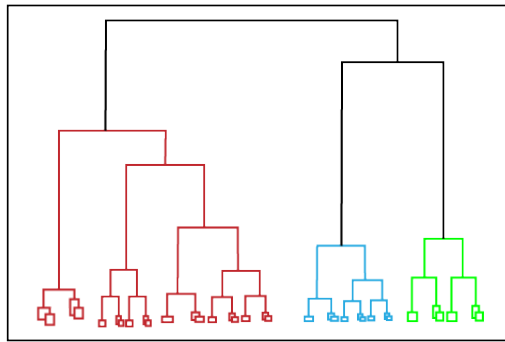
Hình 2.3: Phân cụm dựa trên mật độ

- Phân cụm dựa trên mô hình phân phối: Trong phương pháp phân cụm dựa trên mô hình phân phối, dữ liệu được phân chia dựa trên xác suất về cách tập dữ liệu thuộc về một phân phối cụ thể. Việc phân nhóm được thực hiện bằng cách giả sử một số phân phối phổ biến là phân phối Gaussian.



Hình 2.4: Phân cụm dựa trên mô hình phân phối

- Phân cụm theo cấp bậc: Phân cụm theo cấp bậc có thể được sử dụng thay thế cho phân cụm được phân vùng vì không có yêu cầu chỉ định trước số lượng của cụm sẽ được tạo. Trong kỹ thuật này, tập dữ liệu được chia thành các cụm để tạo ra cấu trúc dạng cây, còn được gọi là Dendrogram. Các quan sát hoặc số lượng cụm bất kỳ có thể được chọn bằng cách cắt cây ở mức chính xác.



Hình 2.5: Phân cụm theo cấp bậc

- Phân cụm mờ: Là một loại phương pháp mềm trong đó một đối tượng dữ liệu có thể thuộc về nhiều nhóm hoặc cụm. Mỗi tập dữ liệu có một tập hợp về các hệ số, phụ thuộc vào mức độ điểm dữ liệu trong một cụm.

d. Các thuật toán phân cụm

Các thuật toán phân cụm có thể được phân chia dựa trên mô hình của chúng đã được giải thích ở trên. Có nhiều loại thuật toán phân cụm khác nhau được công bố nhưng chỉ có một số loại được sử dụng phổ biến:

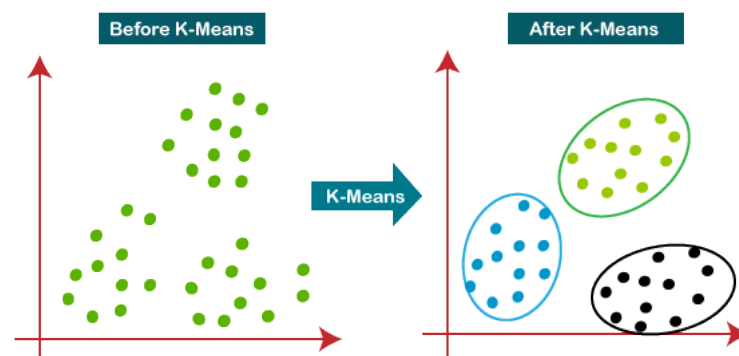
- Thuật toán K-Means: Thuật toán k-mean là một trong những thuật toán phân cụm phổ biến nhất. Nó phân loại tập dữ liệu bằng cách chia các mẫu thành các cụm khác nhau có phương sai bằng nhau. Số lượng cụm phải được chỉ định trong thuật toán này. Nó nhanh và cần ít tính toán hơn với độ phức tạp tuyến tính là $O(n)$.
- Mean-shift (thuật toán dịch chuyển trung bình): Thuật toán dịch chuyển trung bình cố gắng tìm các vùng dày đặc trong mật độ trơn tru của các điểm dữ liệu. Đây là một ví dụ về mô hình dựa trên centroid, hoạt động dựa trên việc cập nhật các ứng cử viên cho centroid trở thành trung tâm của các điểm trong một khu vực nhất định.
- DBSCAN (Density-Based Spatial Clustering of Applications with Noise): Đây là một ví dụ về mô hình dựa trên mật độ tương tự như độ dịch chuyển trung bình, nhưng có một số ưu điểm đáng chú ý. Trong thuật toán này, các vùng có mật độ cao được phân tách bằng các vùng có mật độ thấp. Bởi vì điều này, các cụm có thể được tìm thấy ở bất kỳ hình dạng tùy ý nào.

- GMM ((Gaussian Mixture Model): Thuật toán này có thể được sử dụng thay thế cho thuật toán K-Means hoặc cho những trường hợp K-Means có thể bị lỗi. Trong GMM, giả định rằng các điểm dữ liệu được phân phối Gaussian.
- Agglomerative Hierrarchical: Thuật toán phân cấp tổng hợp thực hiện phân cụm theo cấp bậc từ dưới lên. Trong đó, mỗi điểm dữ liệu được coi là một cụm duy nhất ngay từ đầu và sau đó được hợp nhất liên tục. Hệ thống phân cấp cụm có thể được biểu diễn dưới dạng cấu trúc cây.

2.2.2 Thuật toán K-Means Clustering

a. Khái niệm

Phân cụm K-means là một thuật toán học không giám sát, nhóm tập dữ liệu không được gán nhãn thành các cụm khác nhau. Ở đây k xác định số lượng cụm được xác định trước. Một cách trừu tượng hơn đây là một thuật toán lặp chia tập dữ liệu không được gán nhãn thành k cụm khác nhau sao cho mỗi tập dữ liệu chỉ thuộc một nhóm có các thuộc tính tương tự. Đây là một thuật toán dựa trên centroid (tâm), trong đó mỗi cụm được liên kết với một centroid.



Hình 2.6: Thuật toán K-means Clustering

Mục đích chính của thuật toán này là giảm thiểu tổng khoảng cách giữa điểm dữ liệu và cụm tương ứng của chúng. Thuật toán lấy tập dữ liệu không được gán nhãn làm đầu vào, chia tập dữ liệu thành số k cụm và lặp lại quy trình cho đến khi không tìm thấy cụm tốt nhất. Giá trị của k luôn luôn phải được xác định trước trong thuật toán này.

Thuật toán phân cụm K-means chủ yếu thực hiện hai nhiệm vụ:

- Xác định giá trị tốt nhất cho k điểm trung tâm hoặc trong tâm bằng một quá trình lặp đi lặp lại.
- Gán từng điểm dữ liệu cho tâm k gần nhất của chúng. Những điểm dữ liệu gần trung tâm k cụ thể sẽ tạo ra một cụm.

Do đó mỗi cụm có các điểm dữ liệu có một số điểm tương đồng và nằm xa các cụm khác.

b. Cách thức hoạt động của K-means

Hoạt động của thuật toán K-means có thể được giải thích theo các bước sau:

1. Chọn số k để quyết định số cụm
2. Chọn k điểm hoặc tâm ngẫu nhiên
3. Gán từng điểm dữ liệu cho trọng tâm gần nhất của chúng, tâm này sẽ tạo thành các cụm k được xác định trước.
4. Tính toán khoảng cách và đặt trong tâm mới của cụm.
5. Lặp lại bước thứ 3
6. Nếu số lượng điểm dữ liệu trong một cụm có sự thay đổi thì lặp lại bước 4, nếu không thuật toán sẽ dừng lại.

Giả sử có i điểm dữ liệu và có μ_j là tập các tâm cụm. Mô hình xác định nhãn cho từng điểm dữ liệu c_i bằng cách:

$$c_i = \arg \min_j \|\mathbf{x}_i - \mu_j\|_2^2 \quad (2.1)$$

Sau mỗi lần lặp thuật toán sẽ tính toán lại tâm cho từng cụm theo trung bình của toàn bộ các điểm đã được phân vào cùng một cụm:

$$\mu_j := \frac{\sum_{i=1}^n \mathbf{1}(c_i = j) \mathbf{x}_i}{\sum_{i=1}^n \mathbf{1}(c_i = j)} \quad (2.2)$$

Trong công thức 1 thì $\|\mathbf{x}\|_2^2$ là bình phương của norm chuẩn bậc 2, kí hiệu là L_2 . Trong công thức 2 hàm này trả về giá trị là 1 nếu nhãn của điểm dữ liệu c_i được dự báo thuộc về cụm j , trái lại thì trả về giá trị 0.

Nếu ta coi các μ_j là các tâm cụm của mỗi cụm và ước lượng tất cả các điểm được phân vào cụm này bởi μ_j thì một điểm dữ liệu i được phân vào cụm μ_j sẽ có sai số như công thức 1. Chúng ta luôn mong sai số này có giá trị tuyệt đối là nhỏ nhất. Vậy nên hàm mất mát của thuật toán này chính là tìm các giá trị j sao cho tổng bình phương khoảng cách giữa các điểm dữ liệu và cụm là nhỏ nhất.

c. Ưu điểm của thuật toán K-Means

Thuật toán K-Means có một số ưu điểm như:

- Đơn giản và dễ dàng cài đặt và sử dụng
- Độ phức tạp tính toán tương đối nhỏ, phù hợp cho tập dữ liệu nhỏ.

d. Hạn chế của K-Means

Bên cạnh những ưu điểm thì K-Means cũng có một số hạn chế:

- Cần phải xác định trước số cụm cho thuật toán: Vì bộ dữ liệu của chúng ta chưa được gán nhãn nên dường như chúng ta không có thông tin nào về số lượng cụm hợp lý. Chúng ta chỉ có thể thực hiện phương pháp thử và sai (try and error) và xác định số cụm thông qua một phương pháp chẳng hạn như Elbow.
- Vị trí tâm của cụm sẽ bị phụ thuộc vào điểm khởi tạo ban đầu của chúng: Những vị trí khởi tạo khác nhau có thể dẫn tới cách phân cụm khác nhau, mặc dù thuật toán có cùng thiết lập số cụm.
- Đối với những bộ dữ liệu có hình dạng phức tạp hoặc mất cân bằng thì thuật toán không hội tụ về qui luật phân chia tổng quát. Chẳng hạn như dữ liệu có dạng đường viền hình tròn bao ngoài một hình tròn ở bên trong nó; dữ liệu hình tròn ốc; dữ liệu có phân phối dẹt; dữ liệu bị mất cân bằng phân phối giữa các cụm.
- Thuật toán rất nhạy cảm với outliers: Khi xuất hiện outliers thì thường khiến cho tâm cụm bị chệch và do đó dự báo cụm không còn chuẩn xác. Chính vì thế chúng ta cần phải loại bỏ outliers trước khi huấn luyện thuật toán.

- Thuật toán k-Means yêu cầu phải tính khoảng cách từ một điểm tới toàn bộ các tâm cụm để tìm ra tâm cụm gần nhất. Như vậy chúng ta cần phải load toàn bộ dữ liệu lên RAM, đối với những bộ dữ liệu kích thước lớn thì sẽ vượt quá khả năng lưu trữ của RAM. Khi đó chúng ta cần phải huấn luyện thuật toán theo phương pháp online learning.

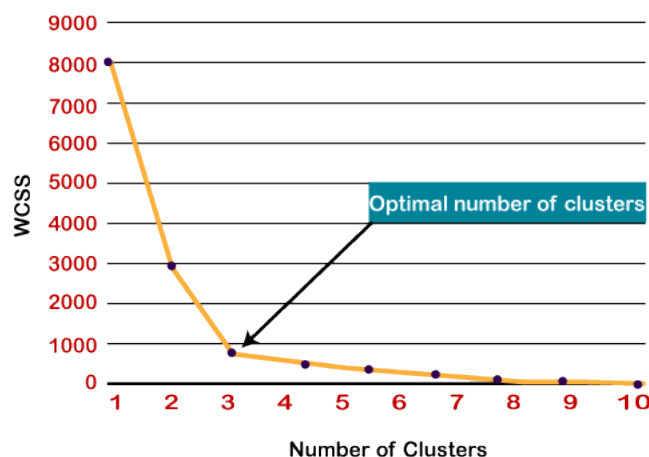
2.2.3 Phương pháp Elbow

Như đã đề cập ở phần hạn chế, trong thuật toán K-Means chúng ta cần phải xác định trước số cụm. Và phương pháp Elbow là một cách giúp chúng ta lựa chọn được số lượng các cụm phù hợp dựa vào đồ thị trực quan hóa bằng cách nhìn vào sự suy giảm của hàm biến dạng và lựa chọn ra điểm khuỷu tay (elbow point).

Phương pháp này sử dụng khái niệm giá trị WCSS (Within Cluster Sum of Squares) để xác định tổng số biến thể trong cụm. Công thức tính giá trị WCSS cho n cụm như sau:

$$WCSS = \sum_{p_i \in \mu_1} distance(p_i C_1)^2 + \sum_{p_i \in \mu_2} distance(p_i C_2)^2 + \sum_{p_i \in \mu_n} distance(p_i C_n)^2 \quad (2.3)$$

Trong đó $\sum_{p_i \in \mu_1} distance(p_i C_1)^2$ là tổng bình phương khoảng cách giữa mỗi điểm dữ liệu và trọng tâm của chúng trong cụm 1 và tương tự cho các số hạng còn lại. Để đo khoảng cách giữa các điểm dữ liệu và trọng tâm, chúng ta có thể sử dụng bất kỳ phương pháp nào như khoảng cách Euclidean hoặc khoảng cách Manhattan.



Hình 2.7: Phương pháp Elbow

Để tìm giá trị tối ưu của cụm, phương pháp khuỷu tay thực hiện theo các bước sau:

- Thực thi phân cụm K-Means trên một tập dữ liệu nhất định cho các giá trị k khác nhau ví dụ từ 1 đến 10.
- Với mỗi giá trị của k , tính WCSS.
- Vẽ đường cong giữa các giá trị WCSS được tính toán và số cụm k .
- Điểm uốn cong hoặc một điểm của đồ thị trông giống như một cánh tay thì điểm đó được coi là giá trị tốt nhất của k .

Điểm khuỷu tay là điểm mà ở đó tốc độ suy giảm của hàm biến dạng sẽ thay đổi nhiều nhất. Tức là sau vị trí này thì gia tăng thêm số lượng cụm cũng không giúp hàm biến dạng giảm đáng kể. Nếu thuật toán phân chia số lượng cụm tại vị trí này sẽ đạt được tính chất phân cụm một cách tổng quan nhất mà không gặp các hiện tượng overfitting.

Phương pháp Elbow là một phương pháp thường được sử dụng để lựa chọn số cụm phân chia hợp lý dựa trên biểu đồ, tuy nhiên có một số trường hợp chúng ta không thể dễ dàng phát hiện vị trí của Elbow, đặc biệt là đối với những bộ dữ liệu mà qui luật phân cụm không thực sự dễ dàng được phát hiện. Nhìn chung Elbow vẫn là một phương pháp tốt nhất được ứng dụng trong việc tìm kiếm số lượng cụm cần phân chia.

2.2.4 Độ đo bóng (Silhouette)

Đây cũng là một kĩ thuật giúp chọn được số lượng cụm tối ưu. Giả sử mạng lưới được chia thành k cụm.

Với mỗi node i đặt:

- a_i là khoảng cách trung bình từ i tới tất cả các node trong cùng cụm với i .
- $b(i)$ là khoảng cách trung bình ngắn nhất từ i tới bất kỳ cụm nào không chứa i . Cụm tương ứng với b_i này được gọi là cụm hàng xóm của i .

Khi đó:

$$s(i) = \frac{b_i - a_i}{\max[a_i, b_i]} \quad (2.4)$$

s_i nằm trong đoạn $[-1, 1]$, s_i càng gần 1 thì node i càng phù hợp với cụm mà nó được phân vào. $s_i = 0$ thì không thể xác định được i nên thuộc về cụm nào giữa cụm hiện tại và cụm

hàng xóm của chúng. s_i càng gần -1 thì chứng tỏ i bị phân sai cụm, chúng nên thuộc về cụm hàng xóm chứ không phải là cụm hiện tại.

CHƯƠNG 3: TRIỂN KHAI CÀI ĐẶT THUẬT TOÁN K-MEANS TRÊN ỨNG DỤNG APACHE SPARK

3.1 Ngôn ngữ và công cụ

3.1.1 Ngôn ngữ

Sử dụng ngôn ngữ Python làm ngôn ngữ chính để phát triển cài đặt và xây dựng chương trình trên Spark thông qua thư viện Pyspark và MLlib.

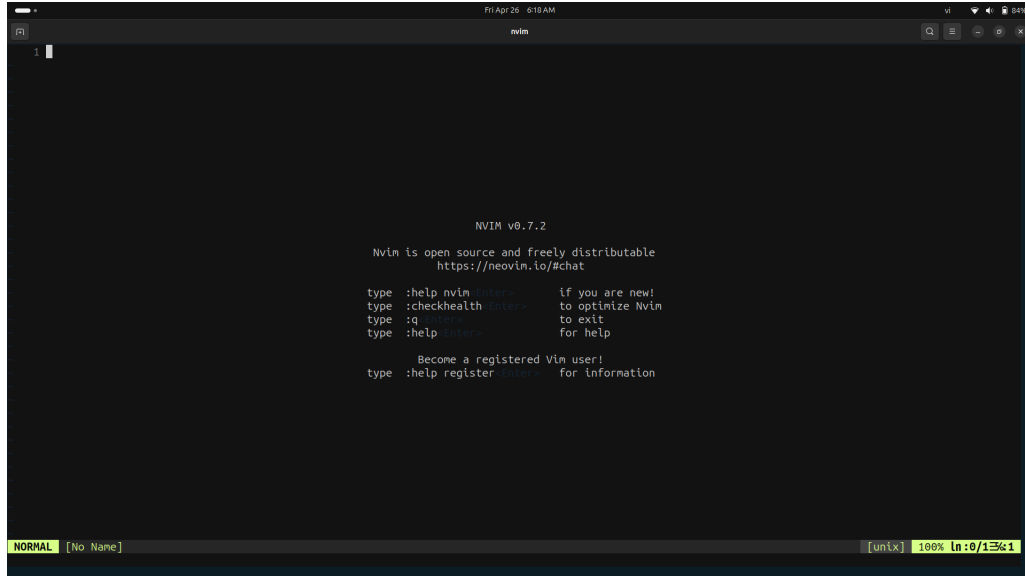
3.1.2 Công cụ

Sử dụng trình biên dịch Neovim để viết mã và xây dựng ứng dụng cho bài toán. Neovim là trình soạn thảo văn bản hay nói cách khác là một công cụ IDE có thể được sử dụng cho nhiều tác vụ.

Ưu điểm của Neovim:

- Giao diện tối giản, gần gũi, linh hoạt, nhanh nhạy.
- Cho phép người dùng trên cộng đồng có những đóng góp mới: như đóng góp về mặt chức năng cho Neovim như: themse (giao diện), syntax for language(cú pháp cho các ngôn ngữ),v.v.
- Tích hợp đa nền tảng một cách nhất quán.
- Tối ưu hóa cho người dùng.

Để cài đặt Neovim có hai cách để cài đặt. Đối với hệ điều hành Window truy cập vào: Neovim để tải và cài đặt. Đối với hệ điều hành Ubuntu tại terminal gõ "sudo apt update" sau đó gõ "sudo apt-get install neovim" để cài đặt.



Hình 3.1: Giao diện Neovim

3.2 Cài đặt và triển khai ứng dụng trên Apache Spark

3.2.1 Tập dữ liệu

Sử dụng tập dữ liệu có tên là "Customer". Đây là tập dữ liệu khách hàng như:

- InvoiceNo: Mã vùng.
- Quantity: Số lượng mua hàng.
- InvoiceDate: Ngày mua hàng.
- UnitPrice: Số tiền đã mua hàng.

InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
536365	85123A	WHITE HANGING HE	6	12/1/2010 8:26	2.55	17850	United Kingdom
536365	71053	WHITE METAL LANT	6	12/1/2010 8:26	3.39	17850	United Kingdom
536365	84406B	CREAM CUPID HEA	8	12/1/2010 8:26	2.75	17850	United Kingdom
536365	84029G	KNITTED UNION FL	6	12/1/2010 8:26	3.39	17850	United Kingdom
536365	84029E	RED WOOLLY HOTT	6	12/1/2010 8:26	3.39	17850	United Kingdom
536365	22752	SET 7 BABUSHKA N	2	12/1/2010 8:26	7.65	17850	United Kingdom
536365	21730	GLASS STAR FROS	6	12/1/2010 8:26	4.25	17850	United Kingdom
536366	22633	HAND WARMER UNI	6	12/1/2010 8:28	1.85	17850	United Kingdom
536366	22632	HAND WARMER REC	6	12/1/2010 8:28	1.85	17850	United Kingdom
536367	84879	ASSORTED COLOUR	32	12/1/2010 8:34	1.69	13047	United Kingdom

Hình 3.2: Tập dữ liệu Customer

3.2.2 Tiền xử lý dữ liệu

Sau khi kiểm tra tổng quan về tập dữ liệu, nhận thấy có các cột dữ liệu bị thiếu giá trị. Tiến hành xử lý dữ liệu này bằng cách xóa đi các hàng có dữ liệu bị thiếu những cột đó.

```
-----Kiểm tra các giá trị bị thiếu-----
+-----+-----+-----+-----+-----+-----+-----+-----+
|InvoiceNo|StockCode|Description|Quantity|InvoiceDate|UnitPrice|CustomerID|Country|
+-----+-----+-----+-----+-----+-----+-----+-----+
|      0|      0|      1454|      0|      0|      0|    135080|      0|
+-----+-----+-----+-----+-----+-----+-----+-----+
```

Hình 3.3: Xử lý giá trị bị thiếu

Chuẩn hóa dữ liệu bằng kĩ thuật RFM. Đây là kĩ thuật dùng để tính toán 3 chỉ số Recency (Lần gần nhất mua hàng), Frequency (Tần suất mua hàng) và Monetary (Tổng số tiền đã mua hàng).

```
-----Hiện thị tập dữ liệu sau khi tính toán chỉ số RFM-----
+-----+-----+-----+
|Recency|Frequency|Monetary|
+-----+-----+-----+
|    326|      10|     0.0|
|      2|     910|  21550.0|
|     75|     155|   8986.2|
|     19|     365|  8787.75|
|    310|      85|   1672.0|
+-----+-----+-----+
only showing top 5 rows
```

Hình 3.4: Dữ liệu sau khi được tính toán chỉ số RFM

VectorAssembler là một lớp trong thư viện PySpark, được sử dụng để tổng hợp các cột dữ liệu thành một cột vector duy nhất.

```
-----Dữ liệu sau khi được chuyển thành vector-----
+-----+-----+-----+-----+
|Recency|Frequency|Monetary|num_vector|
+-----+-----+-----+-----+
|    326|      10|     0.0|[326.0,10.0,0.0]|
|      2|     910|  21550.0|[2.0,910.0,21550.0]|
|     75|     155|   8986.2|[75.0,155.0,8986.2]|
|     19|     365|  8787.75|[19.0,365.0,8787.75]|
|    310|      85|   1672.0|[310.0,85.0,1672.0]|
+-----+-----+-----+-----+
```

Hình 3.5: Kỹ thuật VectorAssembler

Cho phép xây dựng các biểu đồ dữ liệu và mô hình học máy trong PySpark bằng cách kết hợp các đặc trưng vào một vector duy nhất, giúp thuận tiện cho việc xử lý dữ liệu và tạo

đầu vào cho các thuật toán học máy. Đầu tiên đưa 3 cột dữ liệu về dạng vector ở một cột duy nhất. Sau đó thực hiện chuẩn hóa dữ liệu bằng StandardScaler.

```
-----Dữ liệu sau khi được co giãn đặc trưng-----
+-----+-----+-----+-----+-----+
|Recency|Frequency|Monetary|num_vector|scaler_number|
+-----+-----+-----+-----+-----+
| 326| 10| 0.0| [326.0,10.0,0.0]| [2.32175727464055...|
| 2| 910| 21550.0| [2.0,910.0,21550.0]| [-0.8936310147844...|
| 75| 155| 8986.2| [75.0,155.0,8986.2]| [-0.1691762458707...|
| 19| 365| 8787.75| [19.0,365.0,8787.75]| [-0.7249223699689...|
| 310| 85| 1672.0| [310.0,85.0,1672.0]| [2.16297266775537...|
+-----+-----+-----+-----+-----+
```

Hình 3.6: Chuẩn hóa dữ liệu

3.2.3 Huấn luyện mô hình

Áp dụng kĩ thuật silhouette để tính điểm cho các cụm được khởi tạo ngẫu nhiên từ 2 đến 10. Thực hiện huấn luyện trên tập dữ liệu và thu được kết quả:

```
Sil score for k = 2 is 0.9915020384733066
Sil score for k = 3 is 0.7424681785900861
Sil score for k = 4 is 0.46959765706212575
Sil score for k = 5 is 0.7888786239464297
Sil score for k = 6 is 0.5702297934970758
Sil score for k = 7 is 0.5699367036943114
Sil score for k = 8 is 0.7587368655106982
Sil score for k = 9 is 0.5076012030012443
```

Hình 3.7: Tính điểm silhouette cho các cụm ngẫu nhiên

Quan sát vào hình 3.7 ta nhận thấy điểm silhouette với số k=3, k=5, k=7 cho các giá trị xấp xỉ, điều này cho thấy tập dữ liệu có phân bố và đặc điểm hình dạng không đều. Lựa chọn k=3 để tiến hành huấn luyện mô hình, sau đó sử dụng mô hình để dự đoán cụm cho từng điểm dữ liệu:

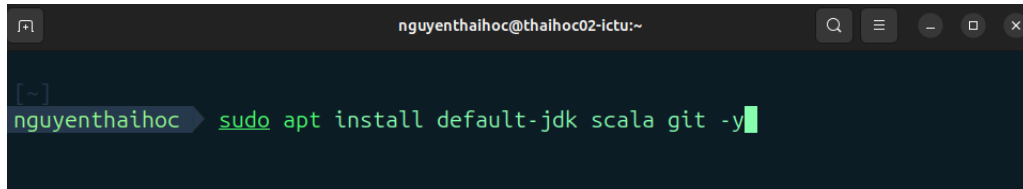
```
+-----+-----+-----+-----+
|Recency|Frequency|Monetary|Cluster|
+-----+-----+-----+-----+
| 326| 10| 0.0| 1|
| 2| 910| 21550.0| 0|
| 75| 155| 8986.2| 0|
| 19| 365| 8787.75| 0|
| 310| 85| 1672.0| 1|
+-----+-----+-----+-----+
```

Hình 3.8: Mô hình dự đoán cụm cho điểm dữ liệu

3.3 Triển khai ứng dụng lên Apache Spark

3.3.1 Cài đặt Spark

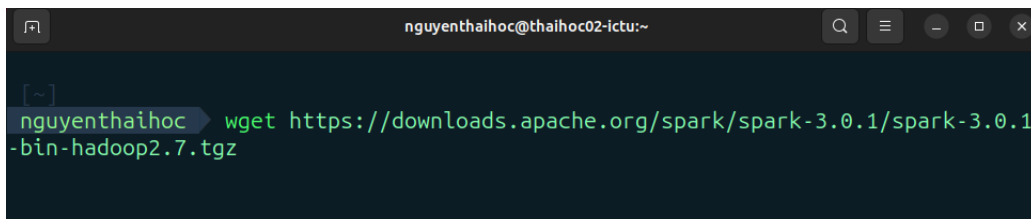
Trước khi tải xuống và thiết lập Spark, cần cài đặt một số phụ thuộc cần thiết. Đầu tiên cần cài đặt các gói JDK, Scala, Git.



```
nguyenthaihoc@thaihoc02-ictu:~$ sudo apt install default-jdk scala git -y
```

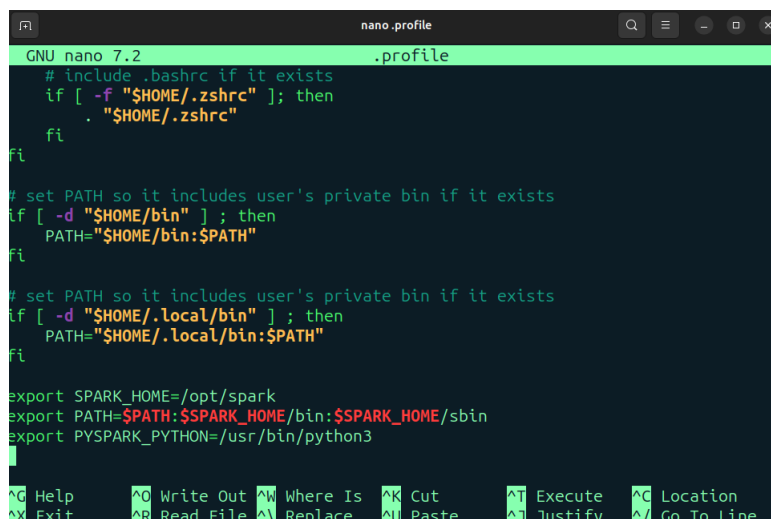
Hình 3.9: Cài đặt các gói phụ thuộc cho Spark

Tiếp theo lựa chọn phiên bản Spark muốn cài, ở đây tôi sử dụng phiên bản Spark 3.0.1.



```
nguyenthaihoc@thaihoc02-ictu:~$ wget https://downloads.apache.org/spark/spark-3.0.1/spark-3.0.1-bin-hadoop2.7.tgz
```

Hình 3.10: Cài đặt Spark



```
GNU nano 7.2 .profile
# include .bashrc if it exists
if [ -f "$HOME/.zshrc" ]; then
    . "$HOME/.zshrc"
fi

# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/bin" ] ; then
    PATH="$HOME/bin:$PATH"
fi

# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/.local/bin" ] ; then
    PATH="$HOME/.local/bin:$PATH"
fi

export SPARK_HOME=/opt/spark
export PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin
export PYSARK_PYTHON=/usr/bin/python3
```

Hình 3.11: Cấu hình môi trường cho Pyspark

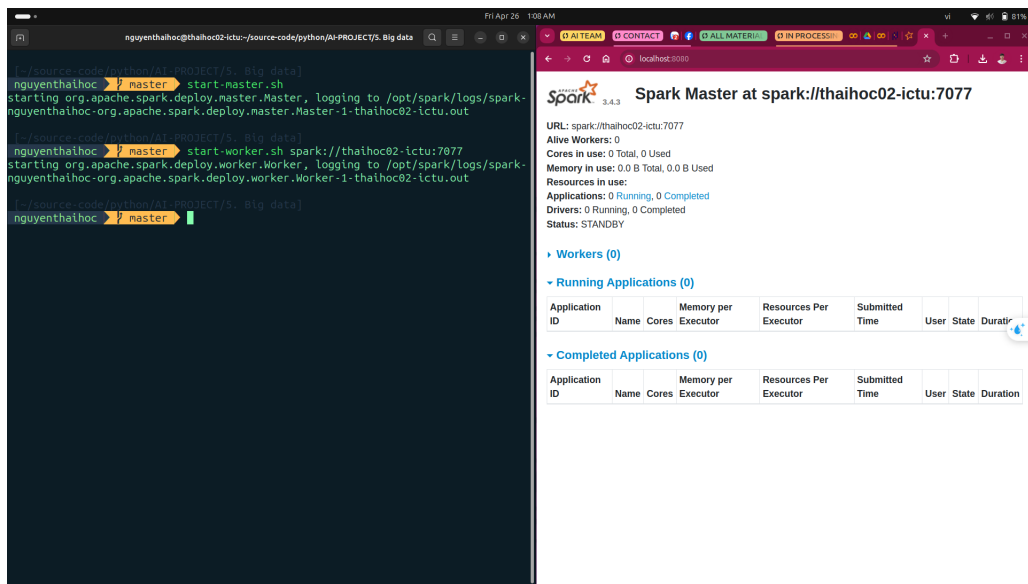
Tiến hành giải nén bằng cách gõ "tar xyz spark-*". Sau đó gõ "sudo mv spark-3.0.1-bin-hadoop2.7 /opt/spark" để di chuyển thư mục vừa được giải nén sang phân vùng mới.

CHƯƠNG 3. TRIỂN KHAI CÀI ĐẶT THUẬT TOÁN K-MEANS TRÊN ỨNG DỤNG APACHE SPARK

Mở file cấu hình môi trường ubuntu bằng cách gõ "sudo nano .profile". Khi màn hình hiện ra hãy thêm 3 dòng export như hình 3.11 ở phía cuối để cấu hình môi trường cho Pyspark sau đó nhấn "Ctrl + O" -> "Enter" -> "Ctrl + X" để lưu. Gõ "source .profile" để lưu và tải lại file cấu hình.

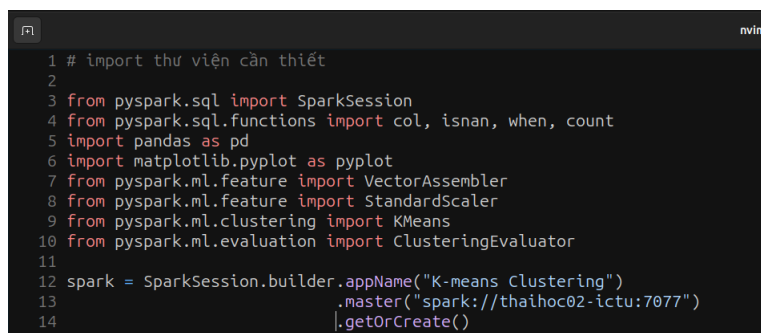
3.3.2 Triển khai ứng dụng

Khởi tạo một master và worker trên spark trong một máy cục bộ:



Hình 3.12: Khởi tạo master và worker

Khi muốn khởi tạo một worker trên một master mong muốn ta cần phải khởi chạy worker cùng với địa chỉ URL của master mong muốn như hình 3.12.

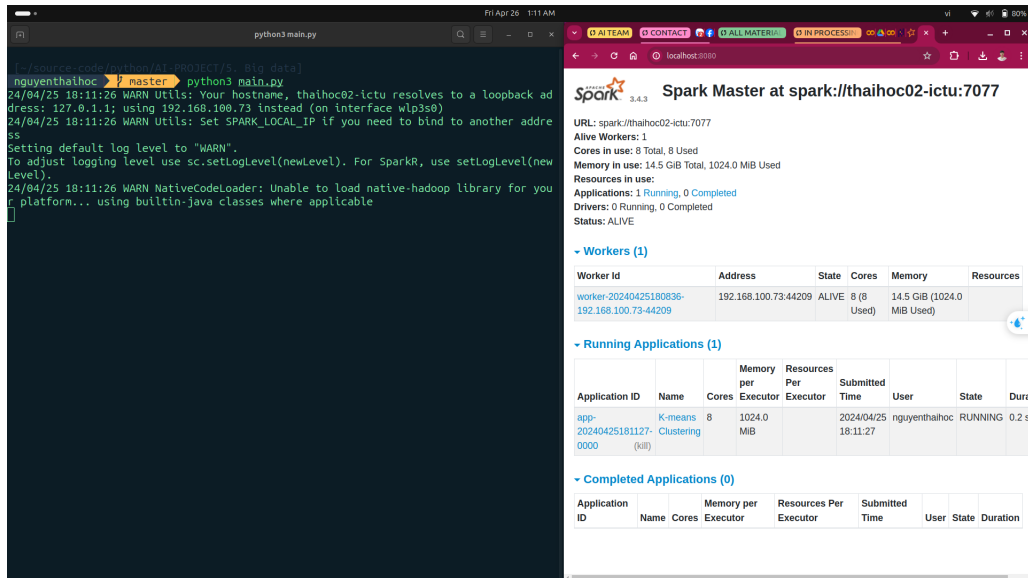


Hình 3.13: Khởi tạo phiên ứng dụng Spark

Truy cập vào địa chỉ localhost:8080 một giao diện của Spark hiện ra như hình 3.12 cùng với các thông tin về master và worker.

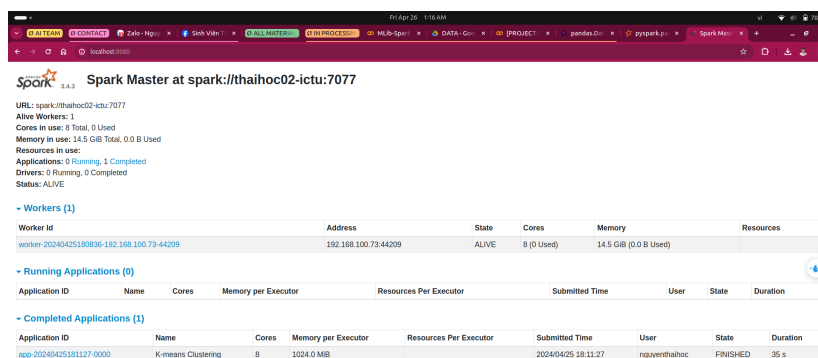
CHƯƠNG 3. TRIỂN KHAI CÀI ĐẶT THUẬT TOÁN K-MEANS TRÊN ỨNG DỤNG APACHE SPARK

Giả sử ta xây dựng mô hình và ứng dụng phía trên vào một file có tên là main.py. Để khởi chạy file này ta cần cấu hình một số thứ. Trong file main.py cần khởi tạo một phiên ứng dụng spark với appName là tên của ứng dụng và master chính là địa chỉ URL của master mà ta muốn chạy ứng dụng (hình 3.13). Để khởi chạy file ứng dụng lên spark chỉ cần gọi tới đúng đường dẫn của file là ứng dụng bắt đầu được khởi chạy.



Hình 3.14: Khởi chạy ứng dụng

Sau khi ứng dụng chạy xong tại giao diện spark phần Completed Application sẽ hiển thị ra các thông tin như thời gian khởi chạy, số thời gian chạy xong ứng dụng, tên người dùng và bộ nhớ.



Hình 3.15: Kết thúc ứng dụng

KẾT LUẬN

Bài báo cáo đã nghiên cứu về cơ sở lý thuyết của dữ liệu lớn và công cụ Apache Spark, bên cạnh đó nhóm đã ứng dụng thuật toán học máy cụ thể là K-Means để phân cụm khách hàng và kết hợp triển khai trên nền tảng xử lý dữ liệu lớn Apache Spark. Trong quá trình nghiên cứu nhóm đã:

- Hiểu được các định nghĩa về dữ liệu lớn, cách thức hoạt động và sử dụng công cụ Apache Spark.
- Hiểu và nắm bắt được các kỹ thuật để triển khai các ứng dụng học máy trên nền tảng Apache Spark.
- Triển khai thành công thuật toán K-Means phân cụm khách hàng trên công cụ Spark.

Mặc dù đã rất nỗ lực, song với kiến thức và thời gian còn hạn chế nên không thể tránh khỏi những thiếu sót. Nhóm rất mong nhận được sự cảm thông, nhận xét, đánh giá từ các thầy cô để bài của nhóm được hoàn thiện hơn nữa. Chúng em xin chân thành cảm ơn.

TÀI LIỆU THAM KHẢO

- [1] Adanma Cecilia Eberendu et al. Unstructured data: an overview of the data of big data. *International Journal of Computer Trends and Technology*, 38(1):46–50, 2016.
- [2] Muhammad Faaique. Overview of big data analytics in modern astronomy. *International Journal of Mathematics, Statistics, and Computer Science*, 2:96–113, 2024.
- [3] Zengyi Huang, Haotian Zheng, Chen Li, and Chang Che. Application of machine learning-based k-means clustering for financial fraud detection. *Academic Journal of Science and Technology*, 10(1):33–39, 2024.
- [4] Trupti M Kodinariya, Prashant R Makwana, et al. Review on determining number of cluster in k-means clustering. *International Journal*, 1(6):90–95, 2013.
- [5] Mohan Maheshwari, Kishan Pal Singh, Swati Agarwal, Rohit Kumar Singh Gautam, Avinash Ravi Raja, and Sudesh Singh. An overview of nano-particle reinforced copper metal matrix composites. In *AIP Conference Proceedings*, volume 3007. AIP Publishing, 2024.
- [6] Hans JP Marvin, Esmée M Janssen, Yamine Bouzembrak, Peter JM Hendriksen, and Martijn Staats. Big data in food safety: An overview. *Critical reviews in food science and nutrition*, 57(11):2286–2295, 2017.
- [7] Laurence Morissette and Sylvain Chartier. The k-means clustering technique: General considerations and implementation in mathematica. *Tutorials in Quantitative Methods for Psychology*, 9(1):15–24, 2013.
- [8] Shi Na, Liu Xumin, and Guan Yong. Research on k-means clustering algorithm: An improved k-means clustering algorithm. In *2010 Third International Symposium on intelligent information technology and security informatics*, pages 63–67. Ieee, 2010.

- [9] Phạm Đình Khánh. Thuật toán K-Means Clustering. https://phamdingkhanh.github.io/deepai-book/ch_ml/index_KMeans.html, May 2021. Accessed on May 31, 2020.
- [10] Richard EA Robertson, Jenni Barclay, EP Joseph, and RSJ Sparks. An overview of the eruption of la soufrière volcano, st vincent 2020–21. *Geological Society, London, Special Publications*, 539(1):1–24, 2024.
- [11] Glenn G Sparks. Media effects research: A basic overview. *Canadian Journal of Communication*, 40:2, 2006.
- [12] Douglas Steinley. K-means clustering: a half-century synthesis. *British Journal of Mathematical and Statistical Psychology*, 59(1):1–34, 2006.
- [13] Anusha Thakur. Market overview: Big data analytics and its impact on the healthcare industry.
- [14] Miklos A Vasarhelyi, Alexander Kogan, and Brad M Tuttle. Big data in accounting: An overview. *Accounting Horizons*, 29(2):381–396, 2015.
- [15] Hongyi Xiong, Jianhua Wang, Yiming Xiao, Fangjun Xiao, Renhuang Huang, Licong Hong, Bofei Wu, Jinfeng Zhou, Yongbin Long, and Yubin Lan. High-speed parallel segmentation algorithms of meanshift for litchi canopies based on spark and hadoop. *International Journal of Modeling, Simulation, and Scientific Computing*, 2024.

PHỤ LỤC CODE

```
1 from pyspark.sql import SparkSession
2 from pyspark.sql.functions import col, isnan, when, count
3 import pandas as pd
4 import matplotlib.pyplot as pyplot
5 from pyspark.ml.feature import VectorAssembler
6 from pyspark.ml.feature import StandardScaler
7 from pyspark.ml.clustering import KMeans
8 from pyspark.ml.evaluation import ClusteringEvaluator
9
10 spark = SparkSession.builder.appName("K-means Clustering")
11                                     .master("spark://thaihoc02-ictu:7077")
12                                     .getOrCreate()
13
14 customer_df = spark.read.options(header=True,
15 inferSchema=True).csv("customer-rfm-origin.csv")
16
17 customer_df.show(5)
18
19 customer_df.printSchema()
20
21 customer_df.describe().show()
22
23 customer_df.select([count(when(isnan(CustomerID) |
24 col(CustomerID).isNull(), CustomerID)).alias(CustomerID)
25 for CustomerID in customer_df.columns]).show()
26
27 a = customer_df.dropna(subset=["CustomerID"])
28
29 a = spark.read.options(header=True, inferSchema=True).csv("rfm.csv")
```

```
30
31 final_df = a.withColumnRenamed("InvoiceDate", "Recency").
    withColumnRenamed("InvoiceNo", "Frequency").withColumnRenamed("
    TotalPay", "Monetary")
32
33 final_df.show(5)
34
35 col_number = final_df.columns
36
37 num_vector = VectorAssembler(inputCols=col_number,
38                               outputCol="num_vector")
39
40 train = num_vector.transform(final_df)
41 train.show(5)
42
43 scaler = StandardScaler(inputCol="num_vector",
44                           outputCol="scaler_number",
45                           withMean=True, withStd=True)
46
47 scaler = scaler.fit(train)
48
49 train = scaler.transform(train)
50
51 train.show(5)
52
53 silhouette_score = []
54 evaluator = ClusteringEvaluator(predictionCol='prediction',
55                                  featuresCol="scaler_number",
56                                  metricName="silhouette",
57                                  distanceMeasure="squaredEuclidean")
58
59 for i in range(2, 10):
60     kmeans = KMeans(featuresCol="scaler_number", k=i, seed=42)
61     model = kmeans.fit(train)
62     predictions = model.transform(train)
63     score=evaluator.evaluate(predictions)
64     silhouette_score.append(score)
```

```
65     print("Sil score for k = ", i, "is", score)
66
67 kmeans = KMeans(featuresCol="scaler_number", k=3, seed=42)
68 model = kmeans.fit(train)
69 predictions = model.transform(train)
70
71 a = predictions.select("Recency").rdd.flatMap(lambda x: x).collect()
72 b = predictions.select("Frequency").rdd.flatMap(lambda x: x).collect()
73 c = predictions.select("Monetary").rdd.flatMap(lambda x: x).collect()
74 d = predictions.select("prediction").rdd.flatMap(lambda x: x).collect()
75
76 results_df = pd.DataFrame({
77     "Recency" : a,
78     "Frequency": b,
79     "Monetary" : c,
80     "Cluster" : d
81 })
82
83 results = spark.createDataFrame(results_df)
84 results.show(5)
```

NHẬN XÉT CỦA GIẢNG VIÊN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Thái nguyên, ngày ... tháng ... năm 2024

Giảng viên