|  |
| --- |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO**  **TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH**  **KHOA CÔNG NGHỆ THÔNG TIN**  Description: C:\Documents and Settings\Administrator\Desktop\logo dai hoc_khong nen.png  **TIỂU LUẬN MÔN HỌC**  **DỮ LIỆU LỚN**  **SỬ DỤNG SPARK ĐÁNH GIÁ NHÀ HÀNG**  **Giảng viên hướng dẫn: ThS. PHẠM ĐÌNH TÀI**  **Sinh viên thực hiện: TRẦN THANH TÙNG**  **MSSV: 2100009418**  **Khoá: 2021**  **Ngành/ chuyên ngành: KHOA HỌC DỮ LIỆU**  **Tp.HCM, ngày 04 tháng 09 Năm 2024** |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO**  **TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH**  **KHOA CÔNG NGHỆ THÔNG TIN**  Description: C:\Documents and Settings\Administrator\Desktop\logo dai hoc_khong nen.png  **TIỂU LUẬN MÔN HỌC**  **DỮ LIỆU LỚN**  **SỬ DỤNG SPARK ĐÁNH GIÁ NHÀ HÀNG**  **Giảng viên hướng dẫn: ThS. PHẠM ĐÌNH TÀI**  **Sinh viên thực hiện: TRẦN THANH TÙNG**  **MSSV: 2100009418**  **Khoá: 2021**  **Ngành/ chuyên ngành: KHOA HỌC DỮ LIỆU**  **Tp.HCM, ngày 04 tháng 09 Năm 2024** |

|  |  |
| --- | --- |
| Trường Đại học Nguyễn Tất Thành  **Khoa Công Nghệ Thông Tin**  🙜 🙜 🙝 🙝 | CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM  **Độc lập – Tự do – Hạnh phúc**  🙜 🙜 🙝 🙝 |

**NHIỆM VỤ ĐỒ ÁN**

*(Sinh viên phải đóng tờ này vào báo cáo)*

Họ và tên: **Trần Thanh Tùng** MSSV: **2100009418**

Chuyên ngành: **Khoa học dữ liệu** Lớp: **21DTH2C** Email: **2100004924@nttu.edu.vn** SĐT: **0902813467** Tên đề tài: Sử dụng Spark đánh già nhà hàng

Giảng viên hướng dẫn: **ThS .Phạm Đình Tài**

Thời gian thực hiện:  **/ /2024 đến / /2024**

**MÔ TẢ ĐỀ TÀI:**

Spark là một công cụ xử lý dữ liệu lớn, nó được dùng để phân tích và đánh giá các nhà hàng dựa trên dữ liệu thu thập được từ các nguồn khác nhau như đánh giá của khách hàng, thông tin từ mạng xã hội, và các dữ liệu khác liên quan đến hoạt động kinh doanh nhà hàng.

**NỘI DUNG VÀ PHƯƠNG PHÁP:**

* Giới thiệu về Spark
* Cách hoạt động của Spark
* Các thành phần của Spark
* Xây dựng ứng dụng Spark
* Kết luận

**YÊU CẦU:**

* Có kiến thức, đam mê, hiểu biết về công nghệ mới như IoT, Mạng neural, Machine Learning, Deep Learning. …Đọc hiểu tài liệu tiếng Anh, kỹ năng trình bày văn bản trên máy tính tốt.

**Nội dung và yêu cầu đã được thông qua Bộ môn.**

*TP.HCM, ngày 04 tháng 09 năm 2024*

|  |  |
| --- | --- |
| **Q. TRƯỞNG BỘ MÔN**  *(Ký và ghi rõ họ tên)* | **GIÁO VIÊN HƯỚNG DẪN**  *(Ký và ghi rõ họ tên)* |

**ThS. Vương Xuân Chí ThS. Phạm Đình Tài**

**LỜI MỞ ĐẦU**

Trong bối cảnh hiện nay, khi mà dữ liệu ngày càng đóng vai trò quan trọng trong việc ra quyết định, việc tận dụng các công cụ và công nghệ xử lý dữ liệu lớn trở thành yếu tố then chốt cho sự thành công của nhiều lĩnh vực, bao gồm cả ngành ẩm thực. Đặc biệt, với sự bùng nổ của các nền tảng đánh giá và chia sẻ trải nghiệm ăn uống, việc xử lý và phân tích lượng lớn dữ liệu người dùng là cần thiết để cung cấp những thông tin chính xác, kịp thời cho cả nhà hàng lẫn khách hàng.

Apache Spark, một nền tảng xử lý dữ liệu lớn mạnh mẽ, đã trở thành lựa chọn hàng đầu cho các nhiệm vụ phân tích dữ liệu quy mô lớn nhờ vào khả năng xử lý dữ liệu nhanh chóng và hiệu quả. Việc ứng dụng Apache Spark vào đánh giá nhà hàng không chỉ giúp tăng cường độ chính xác của các hệ thống đề xuất mà còn mở ra những góc nhìn mới về sở thích và xu hướng ẩm thực của khách hàng.

Đề tài này sẽ đi sâu vào việc sử dụng Apache Spark để thu thập, xử lý và phân tích dữ liệu đánh giá nhà hàng, từ đó đưa ra các đánh giá chi tiết và khách quan hơn. Qua đó, chúng ta sẽ thấy được tiềm năng to lớn của công nghệ dữ liệu lớn trong việc nâng cao chất lượng dịch vụ và trải nghiệm của khách hàng.

**LỜI CẢM ƠN**

Với lòng biết ơn sâu sắc nhất, em xin gửi đến quý Thầy Cô ở Khoa Công Nghệ Thông Tin Trường Đại Học Nguyễn Tất Thành đã truyền đạt vốn kiến thức quý báu của quý Thầy Cô cho chúng em trong suốt thời gian học tập tại trường. Nhờ có những lời hướng dẫn, dạy bảo của Thầy Phạm Đình Tài nên đề tài sử dụng Sprak đánh giá nhà hàng thành công tốt đẹp.

Một lần nữa, em xin chân thành cảm ơn Thầy Phạm Đình Tài, người đã dẫn lối và hướng em vào lĩnh vực dữ liệu lớn, mảng mà em chưa từng tiếp xúc trước đây. Cảm ơn thầy đã vô cùng nhiệt huyết chỉ bảo em từng bước một.

Bài báo cáo của em còn hạn chế và còn nhiều bỡ ngỡ nên không tránh khỏi những thiếu sót, em rất mong nhận được những ý kiến đóng góp quý báu của quý Thầy Cô để kiến thức của em trong lĩnh vực này được hoàn thiện hơn đồng thời có điều kiện bổ sung, nâng cao ý thức và trình độ của mình.

Em xin chân thành cảm ơn các quý Thầy Cô rất nhiều!

|  |  |
| --- | --- |
| TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH  **TRUNG TÂM KHẢO THÍ** | **KỲ THI KẾT THÚC HỌC PHẦN**  **HỌC KỲ III NĂM HỌC 2023 - 2024** |

**PHIẾU CHẤM THI TIỂU LUẬN/ĐỒ ÁN**

BM-ChT-11

Môn thi: Dữ liệu lớn Lớp học phần:21DTH2C

Nhóm sinh viên thực hiện :

1. Trần Thanh Tùng Tham gia đóng góp: 100%

2. Trần Lê Nhựt Trường Tham gia đóng góp: 100%

3. Phạm An Ninh Tham gia đóng góp: 100%

Ngày thi: 04/09/2024 Phòng thi:

Đề tài tiểu luận/báo cáo của sinh viên: Sử dụng Spark đánh giá nhà hàng

Phần đánh giá của giảng viên (căn cứ trên thang rubrics của môn học):

|  |  |  |  |
| --- | --- | --- | --- |
| **Tiêu chí (theo CĐR HP)** | **Đánh giá của GV** | **Điểm tối đa** | **Điểm đạt được** |
| Cấu trúc của báo cáo |  | 1 |  |
| Nội dung |  |  |  |
| Các nội dung thành phần |  | 5 |  |
| Lập luận |  | 2 |  |
| Kết luận |  | 1 |  |
| Trình bày |  | 1 |  |
| **TỔNG ĐIỂM** |  | **10** |  |

|  |  |
| --- | --- |
|  | **Giảng viên chấm thi**  *(ký, ghi rõ họ tên)* |

**NHẬN XÉT CỦA GIẢNG VIÊN GIẢNG DẠY**

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………...

……………………………………………………………………………………………………...

……………………………………………………………………………………………………...

……………………………………………………………………………………………………...

……………………………………………………………………………………………………...

*Tp.HCM, Ngày tháng năm 2024*

**Giảng viên giảng dạy**

(Ký tên, ghi rõ họ tên)

**MỤC LỤC**

[CHƯƠNG 1: GIỚI THIỆU 1](#_Toc176051629)

[1.Giới thiệu đề tài. 1](#_Toc176051630)

[1.2. Lý do chọn đề tài 1](#_Toc176051631)

[1.3. Mục tiêu đề tài. 1](#_Toc176051632)

[1.4. Phương pháp đề tài. 2](#_Toc176051633)

[1.5. Đối tượng và phạm vi nghiên cứu. 2](#_Toc176051634)

[CHƯƠNG 2 ỨNG DỤNG THUẬT TOÁN 3](#_Toc176051635)

[2.1 Mô tả bài toán. 3](#_Toc176051636)

[2.2. Bộ dữ liệu. 3](#_Toc176051637)

[2.3 Lý thuyết trong bài toán 4](#_Toc176051638)

[2.3.1 Spark là gì? 4](#_Toc176051639)

[2.3.2 Tại sao phải dùng Spark 4](#_Toc176051640)

[2.3.3 Spark Concepts 6](#_Toc176051641)

[2.3.4 Thành phần của Spark 7](#_Toc176051642)

[2.3.5 Spark Hoạt Động Như Thế Nào? 9](#_Toc176051643)

[2.3.6 Lập trình với RDDs 10](#_Toc176051644)

[2.3.7 Khi nào không cần dùng Spark? 12](#_Toc176051645)

[2.4 Thực nghiệm với thư viện Python và PySpark 12](#_Toc176051646)

[CHƯƠNG 3 XÂY DỰNG ỨNG DỤNG NGÔN NGỮ PYTHON 14](#_Toc176051647)

[3.1 Xây dựng ứng dụng và giải thích 14](#_Toc176051648)

[3.1.1 Các đoạn code và giải thích 14](#_Toc176051649)

[3.1.2 Cầu hình môi trường PySpark 14](#_Toc176051650)

[3.1.3 Đọc và hiểu dữ liệu đánh giá 15](#_Toc176051651)

[3.1.4 Chuẩn bị dữ liệu 18](#_Toc176051652)

[3.1.4 Phân tích dữ liệu 20](#_Toc176051653)

[3.1.6 Modeling dữ liệu 25](#_Toc176051654)

[KẾT LUẬN 30](#_Toc176051655)

[Kết quả đạt được. 30](#_Toc176051656)

[Hạn chế và hướng phát triển. 31](#_Toc176051657)

[TÀI LIỆU THAM KHẢO 32](#_Toc176051658)

**DANH MỤC HÌNH**

[Hình 1. Cấu trúc 1 đoạn file json 3](#_Toc176079963)

[Hình 2. So sánh tốc độ Hadoop và Spark 5](#_Toc176079964)

[Hình 3. Các tầng của Spark 6](#_Toc176079965)

[Hình 4. Các nền tảng dùng được trong Spark 6](#_Toc176079966)

[Hình 5. Quy trình của Spark 7](#_Toc176079967)

[Hình 6. Hoạt động của một ứng dụng Spark 9](#_Toc176079968)

[Hình 7. Kiến trúc Spark 10](#_Toc176079969)

[Hình 8. Những đặc điểm chính của RDDs 11](#_Toc176079970)

[Hình 9. Cấu hình 1 Session 15](#_Toc176079971)

[Hình 10. Truy cập vào drive và giải nén lên thư mục content 16](#_Toc176079972)

[Hình 11. Đọc dữ liệu từ tệp JSON vào Spark DataFrame và hiển thị 10 dòng dữ liệu đầu 16](#_Toc176079973)

[Hình 12. 10 dòng đầu của dữ liệu 17](#_Toc176079974)

[Hình 13. Sơ đồ thuộc tính, tổng bình luận 18](#_Toc176079975)

[Hình 14. Làm sạch dữ liệu 19](#_Toc176079976)

[Hình 15. Lọc các bản ghi 19](#_Toc176079977)

[Hình 16. Kiểu dữ liệu ngày 19](#_Toc176079978)

[Hình 17. Thêm 1 cột mới vào DataFrame 20](#_Toc176079979)

[Hình 18. Kiểm tra dữ liệu đã được làm sạch 21](#_Toc176079980)

[Hình 19. Tính toán các thống kê 21](#_Toc176079981)

[Hình 20. Phân tích dữ liệu đánh giá 21](#_Toc176079982)

[Hình 21. Dùng textblob để tính toán mức độ phổ biến 22](#_Toc176079983)

[Hình 22. Điểm Rating trung bình theo xếp hạng sao 23](#_Toc176079984)

[Hình 23. Phân phối xếp hạng Rating trong data 24](#_Toc176079985)

[Hình 24. Số lượng đánh giá theo thời gian 25](#_Toc176079986)

[Hình 25. Xếp hạng sao trung bình theo thời gian 26](#_Toc176079987)

[Hình 26. Loại các cột không cần thiết 26](#_Toc176079988)

[Hình 27. Dữ liệu cho mô hình 27](#_Toc176079989)

[Hình 28. Loại bỏ các Stopword và xây dựng Pipeline 27](#_Toc176079990)

[Hình 29. Train dữ liệu 28](#_Toc176079991)

[Hình 30. Độ chính xác của mô hình 29](#_Toc176079992)

[Hình 31. Trực quan hóa đường cong ROC 30](#_Toc176079993)

[Hình 32. Kết quả đường ROC 31](#_Toc176079994)

**KÍ HIỆU CÁC CỤM TỪ VIẾT TẮT**

|  |  |
| --- | --- |
| Chữ viết tắt | Ý nghĩa |
| RDDs | Resilient Distributed Datasets |
| HDFS | Hadoop Distributed File System |
| SQL | Structured Query Language |
| DAG | Directed Acyclic Graph |
| RAM | Random-access memory |
| YARN | Yet Another Resource Negotiator |

# CHƯƠNG 1: GIỚI THIỆU

## 1.Giới thiệu đề tài.

Trong thời đại công nghệ số hiện nay, lượng dữ liệu được tạo ra và lưu trữ ngày càng lớn, đặc biệt là trong lĩnh vực đánh giá và phản hồi của người tiêu dùng. Với sự phát triển của các nền tảng đánh giá trực tuyến như Yelp, TripAdvisor, và Google Reviews, các nhà hàng có thể thu thập hàng ngàn ý kiến từ khách hàng mỗi ngày. Tuy nhiên, việc phân tích khối lượng dữ liệu lớn này để rút ra những thông tin hữu ích cũng như xử lý với tốc độ cao cũng không phải là điều đơn giản. Từ đó, hình hanh Spark và các công cụ dữ liệu lớn.

## 1.2. Lý do chọn đề tài

Dữ liệu đánh giá nhà hàng là một chủ đề khá quan trọng trong xã hội hiện nay vì nó ảnh hưởng sâu sắc đến mức độ yêu thích của thực khách đối với nhà hàng họ ăn. Việc có thể phân tích và dự đoán được đánh giá của thực khách sẽ giúp tăng doanh số của nhà hàng cũng như tăng chất lượng phục vụ của nhà hàng.

## 1.3. Mục tiêu đề tài.

Ứng dụng dữ liệu lớn vào bài toán đánh giá nhà hàng từ việc thực nghiệm bằng công cụ Apache Spark.

Trong nghiên cứu, sẽ xây dựng mô hình để dự đoán đánh giá trong tương lai, mô hình sẽ nâng cao khả năng ra quyết định của khách hàng và cải thiện chất lượng dịch vụ trong ngành nhà hàng.

## 1.4. Phương pháp đề tài.

* Data Processing
* Data isualization

Trong bài báo cáo này. chúng em sử dụng mô hình LogisticRegression. Để thực hiện trainning data và lấy kết quả trainning từ mô hình và đánh giá hiệu suất.

## 1.5. Đối tượng và phạm vi nghiên cứu.

* Đối tượng: đánh giá nhà hàng từ nguồn dữ liệu nhà hàng từ trang web đánh giá YELP.
* Phạm vi nghiên cứu: Bình luận và đánh giá của thực khách đối với nhà hàng.

# CHƯƠNG 2: ỨNG DỤNG THUẬT TOÁN

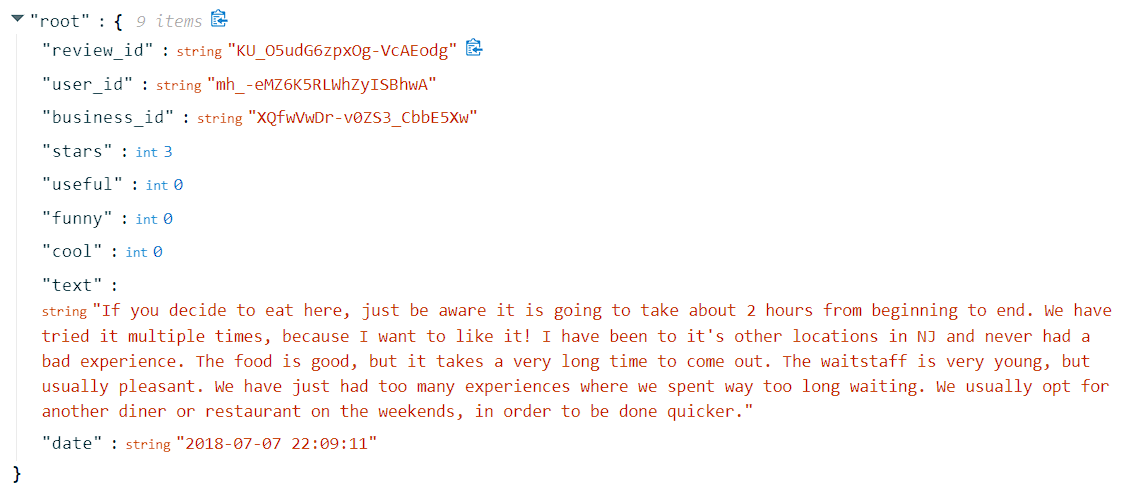
## 2.1 Mô tả bài toán.

Trong bối cảnh ngành dịch vụ nhà hàng ngày càng phát triển, việc cung cấp thông tin chính xác và đáng tin cậy về chất lượng dịch vụ của các nhà hàng là rất quan trọng. Tuy nhiên, việc phân tích và đánh giá một cách toàn diện từ các nguồn dữ liệu không đồng nhất và khộng đủ nhanh là một thách thức lớn. Bài toán của dự án này là xây dựng một hệ thống sử dụng Apache Spark để thu thập, xử lý, phân tích và đánh giá dữ liệu liên quan đến các nhà hàng nhằm cung cấp thông tin chính xác và hữu ích cho người tiêu dùng.

## 2.2. Bộ dữ liệu.

Đầu tiên, lấy số liệu từ file json trên Kaggle

Ta có được bảng dữ liệu như sau:



Hình . Cấu trúc 1 đoạn file json

Trong đó:

* **review\_id:** Mã định danh duy nhất của đánh giá này.
* **user\_id:** Mã định danh duy nhất của người viết đánh giá.
* **business\_id:** Mã định danh duy nhất của doanh nghiệp được đánh giá.
* **stars:** Số sao đánh giá, thường nằm trong khoảng từ 1 đến 5.
* **date:** Ngày và giờ đánh giá được tạo ra.
* **useful:** Số người đánh giá đánh giá này là hữu ích.
* **funny:** Số người đánh giá đánh giá này là hài hước.
* **cool:** Số người đánh giá đánh giá này là tuyệt vời.
* **text:** Nội dung văn bản của đánh giá, bao gồm ý kiến, cảm xúc và trải nghiệm của người đánh giá.

## 2.3 Lý thuyết trong bài toán

### **2.3.1 Spark là gì?**

Apache Spark là một framework mã nguồn mở tính toán cụm, được phát triển sơ khởi vào năm 2009 bởi AMPLab. Sau này, Spark đã được trao cho Apache Software Foundation vào năm 2013 và được phát triển cho đến nay.

Tốc độ xử lý của Spark có được do việc tính toán được thực hiện cùng lúc trên nhiều máy khác nhau. Đồng thời việc tính toán được thực hiện ở bộ nhớ trong (in-memories) hay thực hiện hoàn toàn trên RAM.

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.

### **2.3.2 Tại sao phải dùng Spark**

1. Tốc độ

Spark chạy các chương trình nhanh hơn tới 100 lần so với Hadoop MapReduce trong bộ nhớ hoặc nhanh hơn 10 lần trên đĩa.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

Hình . So sánh tốc độ Hadoop và Spark

1. Dễ sử dụng

Viết ứng dụng nhanh chóng bằng Java, Scala, Python, R.

Spark cung cấp hơn 80 nhà khai thác cấp cao giúp dễ dàng xây dựng các ứng dụng song song.

1. Tổng quát

Kết hợp SQL, phát trực tuyến và phân tích phức tạp.

Spark cung cấp năng lượng cho một chồng thư viện bao gồm SQL và DataFrames, MLlib cho học máy.

GraphX và Spark Streaming. Kết hợp các thư viện này một cách liền mạch trong cùng một ứng dụng.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, Hình chữ nhật

Mô tả được tạo tự động

Hình . Các tầng của Spark

1. Chạy trên đa nền tảng

Spark chạy trên Hadoop, Mesos, độc lập hoặc trên đám mây. Nó có thể truy cập các nguồn dữ liệu đa dạng bao gồm: HDFS, Cassandra, HBase và S3.

Ảnh có chứa văn bản, Phông chữ, biểu tượng, Đồ họa

Mô tả được tạo tự động

Hình . Các nền tảng dùng được trong Spark

### **2.3.3 Spark Concepts**

Một số thuật ngữ của Apache Spark:

* **Job** : Một đoạn mã thực thi đọc dữ liệu đầu vào từ HDFS hoặc máy cục bộ, thực hiện một số tính toán trên dữ liệu và ghi kết quả đầu ra.
* **Stages** : Các công việc được chia thành các giai đoạn. Các giai đoạn được phân loại thành giai đoạn Map hoặc Reduce. Các giai đoạn được chia dựa trên ranh giới tính toán, tất cả các phép tính (toán tử) không thể được thực hiện trong một giai đoạn duy nhất. Chúng xảy ra qua nhiều giai đoạn.
* **Tasks** : Mỗi giai đoạn có một số nhiệm vụ, một nhiệm vụ cho mỗi phân vùng. Một nhiệm vụ được thực thi trên một phân vùng dữ liệu trên một executor (máy).
* **DAG (****Directed Acyclic Graph - Đồ thị có hướng không chu trình)**: Trong ngữ cảnh hiện tại, DAG là đồ thị có hướng không chu trình của các toán tử.
* **Executor** : Quá trình chịu trách nhiệm thực thi một nhiệm vụ.
* **Master** : Máy mà chương trình Driver chạy.
* **Slave** : Máy mà chương trình Executor chạy.

### **2.3.4 Thành phần của Spark**

**Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, thiết kế

Mô tả được tạo tự động**

Hình . Quy trình của Spark

**Spark Driver (Trình điều khiển Spark)**

* **Quy trình riêng biệt để thực thi các ứng dụng người dùng**: Spark Driver là một quá trình riêng biệt chịu trách nhiệm thực thi các ứng dụng được gửi bởi người dùng. Nó quản lý và kiểm soát toàn bộ quá trình thực thi của ứng dụng Spark.
* **Tạo SparkContext để lập lịch thực thi công việc và tương tác với trình quản lý cụm**: Spark Driver tạo ra một đối tượng gọi là SparkContext, đối tượng này chịu trách nhiệm lên lịch thực thi các công việc (jobs) và đàm phán với trình quản lý cụm (Cluster Manager) để có được tài nguyên cần thiết cho việc chạy các công việc đó.

**Executors (Các bộ thực thi)**

* **Chạy các nhiệm vụ do driver lập lịch**: Executors là các tiến trình được tạo ra trên các nút làm việc trong cụm (cluster), và chịu trách nhiệm thực thi các nhiệm vụ (tasks) được lên lịch bởi Spark Driver.
* **Lưu trữ kết quả tính toán trong bộ nhớ, trên đĩa, hoặc ngoài heap**: Executors có khả năng lưu trữ kết quả của các tính toán trong bộ nhớ (RAM), trên đĩa cứng, hoặc ngoài heap (off-heap), tùy thuộc vào cấu hình và yêu cầu của ứng dụng.
* **Tương tác với các hệ thống lưu trữ**: Executors cũng tương tác với các hệ thống lưu trữ dữ liệu như HDFS, S3, Cassandra, HBase, và các hệ thống khác để đọc và ghi dữ liệu trong quá trình thực thi nhiệm vụ.

**Cluster Manager (Trình quản lý cụm)**

* **Mesos**: Một hệ thống quản lý tài nguyên cụm, cho phép chạy và quản lý các ứng dụng phân tán trên các tài nguyên của cụm.
* **YARN**: một hệ thống quản lý tài nguyên cụm được sử dụng trong Hadoop để quản lý và điều phối các ứng dụng phân tán.
* **Spark Standalone**: Một trình quản lý cụm đơn giản và nhẹ tích hợp sẵn trong Apache Spark, không phụ thuộc vào bất kỳ hệ thống quản lý cụm bên ngoài nào và có thể được sử dụng khi cần thiết.

**Spark Driver** bao gồm các thành phần quan trọng để chuyển đổi mã người dùng thành các công việc thực tế được thực thi trên cụm:

Ảnh có chứa văn bản, Phông chữ, biểu đồ, hàng

Mô tả được tạo tự động

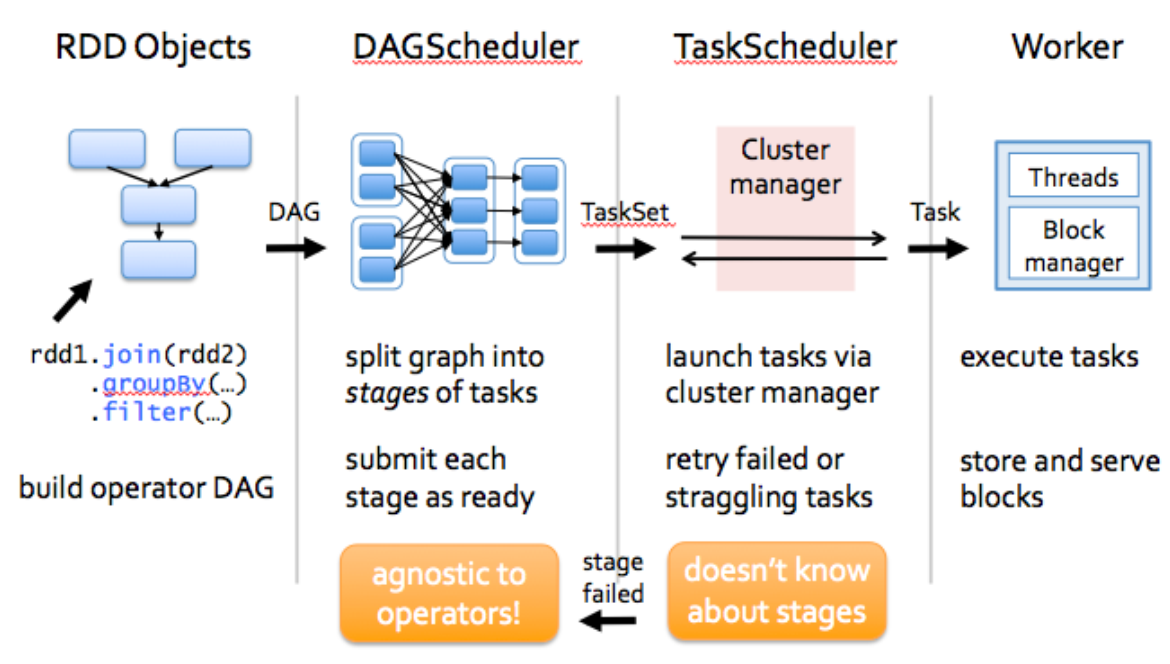
Hình 6. Hoạt động của một ứng dụng Spark

* **SparkContext**: Đại diện cho kết nối đến một cụm Spark, được sử dụng để tạo RDDs, accumulators, và biến broadcast trên cụm đó.
* **DAGScheduler**: Tính toán DAG của các giai đoạn cho mỗi công việc và gửi chúng đến TaskScheduler; xác định vị trí ưu tiên cho các nhiệm vụ dựa trên trạng thái bộ nhớ đệm hoặc vị trí của các tệp shuffle, và tìm lịch trình tối ưu để chạy các công việc.
* **TaskScheduler**: Chịu trách nhiệm gửi nhiệm vụ đến cụm, chạy chúng, thử lại khi có lỗi, và giảm thiểu các nhiệm vụ chậm.
* **SchedulerBackend**: Giao diện nền tảng cho các hệ thống lập lịch, cho phép tích hợp các triển khai khác nhau (Mesos, YARN, Standalone, local).
* **BlockManager**: Cung cấp giao diện để đặt và lấy các khối dữ liệu cả cục bộ và từ xa vào các kho lưu trữ khác nhau (bộ nhớ, đĩa, và ngoài heap).

### **2.3.5 Spark Hoạt Động Như Thế Nào?**

Spark có mã nguồn nhỏ và hệ thống được chia thành nhiều lớp khác nhau. Mỗi lớp có một số trách nhiệm riêng. Các lớp này độc lập với nhau.

Lớp đầu tiên là bộ thông dịch (interpreter), Spark sử dụng một bộ thông dịch Scala với một số chỉnh sửa. Khi bạn nhập mã lệnh của mình vào bảng điều khiển Spark (tạo các RDD và áp dụng các toán tử), Spark sẽ tạo ra một đồ thị toán tử (operator graph). Khi người dùng thực hiện một hành động (như collect), đồ thị này sẽ được gửi đến DAG Scheduler. DAG Scheduler chia đồ thị toán tử thành các giai đoạn (stage) (map và reduce). Một giai đoạn bao gồm các tác vụ (task) dựa trên các phân vùng của dữ liệu đầu vào. DAG Scheduler kết hợp các toán tử lại với nhau để tối ưu hóa đồ thị. Ví dụ, nhiều toán tử map có thể được lập lịch trong cùng một giai đoạn. Việc tối ưu hóa này là chìa khóa cho hiệu suất của Spark. Kết quả cuối cùng của DAG Scheduler là một tập hợp các giai đoạn. Các giai đoạn này được chuyển đến Task Scheduler. Task Scheduler khởi chạy các tác vụ thông qua trình quản lý cụm (Spark Standalone/Yarn/Mesos). Task Scheduler không biết về sự phụ thuộc giữa các giai đoạn.

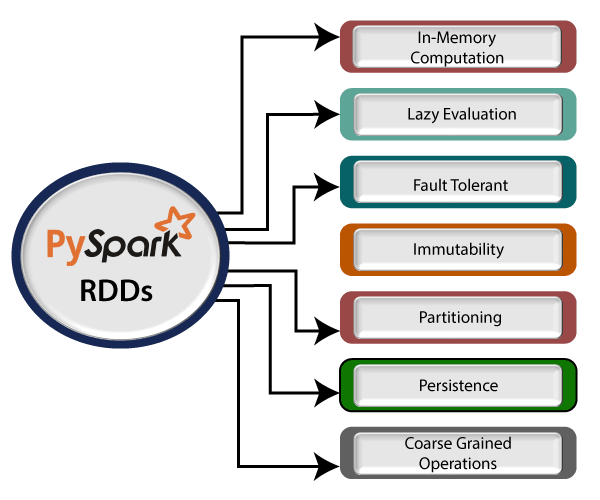


Hình . Kiến trúc Spark

### **2.3.6 Lập trình với RDDs**

Lập trình với RDDs (Resilient Distributed Datasets) là một khái niệm cốt lõi trong Apache Spark. RDDs là một tập hợp các đối tượng được phân tán và bất biến, có thể được xử lý song song trên nhiều nút trong một cụm máy tính.

* 1. **Các đặc điểm chính của RDDs**
* Bất biến: Một khi RDDs được tạo ra, nó không thể thay đổi. Điều này giúp đảm bảo tính nhất quán và dễ dàng quản lý.
* Phân tán: Dữ liệu trong RDDs được chia thành các phân vùng logic, có thể được tính toán trên các nút khác nhau trong cụm.
* Chịu lỗi: RDDs có khả năng tự động phục hồi từ các lỗi nút, giúp đảm bảo tính tin cậy của hệ thống.



Hình . Những đặc điểm chính của RDDs

* 1. **Các cách tạo RDDs**
* **Parallelizing**: Tạo RDDs từ một bộ sưu tập hiện có trong chương trình driver.
* **External Datasets**: Tham chiếu đến một tập dữ liệu trong hệ thống lưu trữ bên ngoài như HDFS, HBase, hoặc bất kỳ nguồn dữ liệu nào hỗ trợ Hadoop InputFormat.
  1. **Các thao tác trên** **RDDs**
* **Transformations**: Các phép biến đổi như map, filter, reduceByKey giúp tạo ra RDD mới từ RDDs hiện có.
* **Actions**: Các hành động như collect, count, saveAsTextFile giúp tính toán và trả về kết quả từ RDDs.

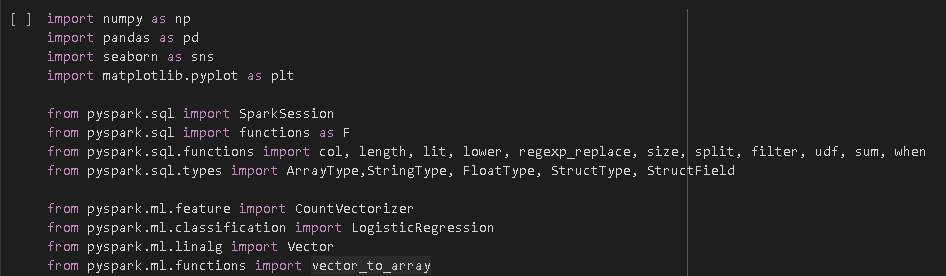
### **2.3.7 Khi nào không cần dùng Spark?**

* **Dữ liệu nhỏ**: Nếu chỉ làm việc với một lượng dữ liệu nhỏ, việc sử dụng Spark có thể là quá mức cần thiết. Các công cụ đơn giản hơn như Pandas trong Python có thể đủ để xử lý.
* **Yêu cầu thời gian thực**: Mặc dù Spark có khả năng xử lý thời gian thực với Spark Streaming, nhưng nếu cần độ trễ cực thấp, các công cụ khác như Apache Flink có thể phù hợp hơn.
* **Chi phí**: Spark yêu cầu tài nguyên phần cứng mạnh mẽ và có thể tốn kém khi triển khai trên đám mây hoặc duy trì cơ sở hạ tầng cần thiết.
* **Độ phức tạp**: Nếu không có kinh nghiệm với Spark, việc triển khai và duy trì có thể phức tạp và tốn thời gian. Trong trường hợp này, các giải pháp đơn giản hơn có thể là lựa chọn tốt hơn.
* **Tích hợp hệ thống**: Nếu hệ thống không tương thích tốt với Spark hoặc đang sử dụng các công nghệ khác mà Spark không hỗ trợ tốt, có thể cần xem xét các giải pháp thay thế.

## 2.4 Thực nghiệm với thư viện Python và PySpark

Chúng em sử dụng các thư viện mà gần gũi với người dùng chẳng hạn như thư viện numpy, thư viện pandas, thư viện matplotlib,… Ở đây chúng em sẽ trình bày cụ thể mục đích sử dụng các thư viện đó.

Đầu tiên việc khai báo thư viện cần:



* Numpy, Pandas, Matplotlib: Các thư viện cho phân tích và trực qua hóa dữ liệu
* from pyspark.sql import SparkSession: Khởi tạo một phiên làm việc với Spark, là điểm vào chính để sử dụng API Spark SQL.
* from pyspark.sql import functions as F: Import module functions từ PySpark SQL với bí danh F, chứa nhiều hàm hỗ trợ xử lý dữ liệu trong DataFrame.
* from pyspark.sql.functions import col, length, lit, lower, regexp\_replace, size, split, filter, udf, sum, when: Import các hàm cụ thể để thao tác và xử lý dữ liệu trong DataFrame như truy cập cột, xử lý chuỗi, tạo điều kiện logic, v.v.
* from pyspark.sql.types import ArrayType, StringType, FloatType, StructType, StructField: Import các kiểu dữ liệu và cấu trúc dữ liệu để định nghĩa schema cho DataFrame, ví dụ: mảng, chuỗi, số thực, và cấu trúc phức tạp.
* from pyspark.ml.feature import CountVectorizer: Công cụ để chuyển đổi văn bản thành vector số lượng từ (mô hình túi từ).
* from pyspark.ml.classification import LogisticRegression: Thuật toán phân loại tuyến tính trong Spark MLlib.
* from pyspark.ml.linalg import Vector: Kiểu dữ liệu biểu diễn vector cho các phép toán trong học máy.
* from pyspark.ml.functions import vector\_to\_array: Hàm chuyển đổi một Vector thành mảng trong Spark MLlib.

# CHƯƠNG 3 XÂY DỰNG ỨNG DỤNG NGÔN NGỮ PYTHON

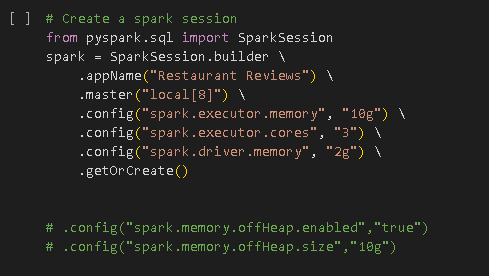
## 3.1 Xây dựng ứng dụng và giải thích

### **3.1.1 Các đoạn code và giải thích**

Công cụ mà chúng em sử dụng để viết trong lần này là Google Colab vì nó khá phổ biến và dễ dàng sử dụng đối với hầu hết người lập trình.

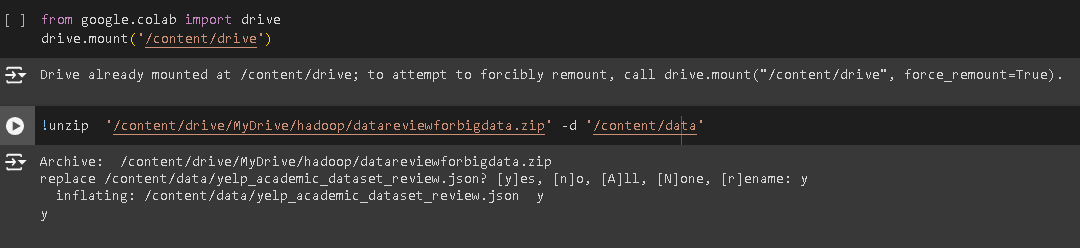
Dùng TPU v2 trong Colab để tăng tốc độ xử lý học sâu của bài toán đánh giá nhà hàng của thực khách.

### **3.1.2 Cấu hình môi trường PySpark**



Hình . Cấu hình 1 Session

* from pyspark.sql import SparkSession: Import SparkSession, đối tượng trung tâm để làm việc với Spark SQL.
* spark = SparkSession.builder ...getOrCreate(): Tạo hoặc lấy một phiên SparkSession với các cấu hình:
* .appName("Restaurant Reviews"): Đặt tên cho ứng dụng.
* .master("local[8]"): Chạy trên máy cục bộ sử dụng 8 lõi CPU.
* .config("spark.executor.memory", "10g"): Cấp 10 GB RAM cho mỗi executor.
* .config("spark.executor.cores", "3"): Mỗi executor dùng 3 lõi CPU.
* .config("spark.driver.memory", "2g"): Driver có 2 GB RAM.
* .getOrCreate(): Tạo phiên mới hoặc dùng phiên đã tồn tại với cấu hình trên.



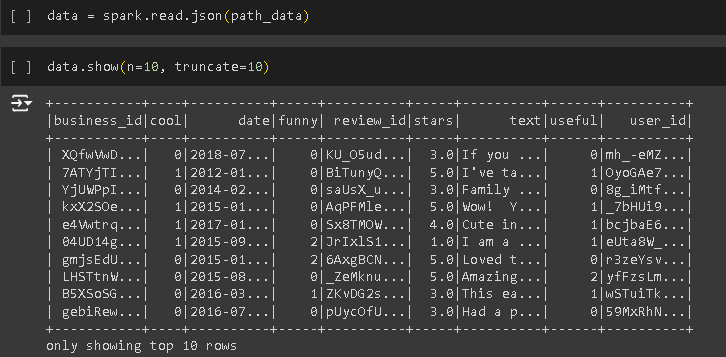
Hình . Truy cập vào drive và giải nén lên thư mục content

Sau đó chúng ta đọc dữ liệu từ drive lên colab, gắn đường dẫn file thành url và gán tên thành df để đọc url.

!unzip '/content/drive/MyDrive/hadoop/datareviewforbigdata.zip' -d '/content/data':

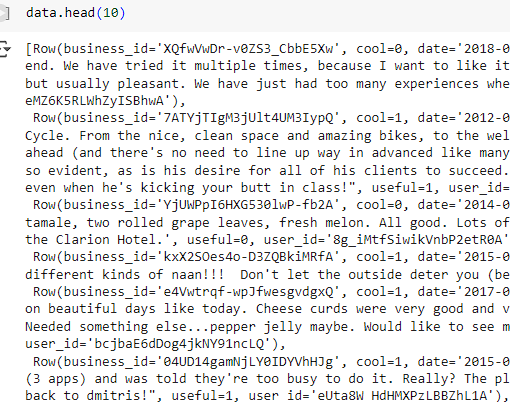
* Sử dụng lệnh unzip trong hệ điều hành Linux để giải nén tệp ZIP có tên datareviewforbigdata.zip, nằm trong Google Drive tại đường dẫn /content/drive/MyDrive/hadoop/.
* Tệp này được giải nén vào thư mục /content/data trong hệ thống tệp của Google Colab.

### **3.1.3 Đọc và hiểu dữ liệu đánh giá**



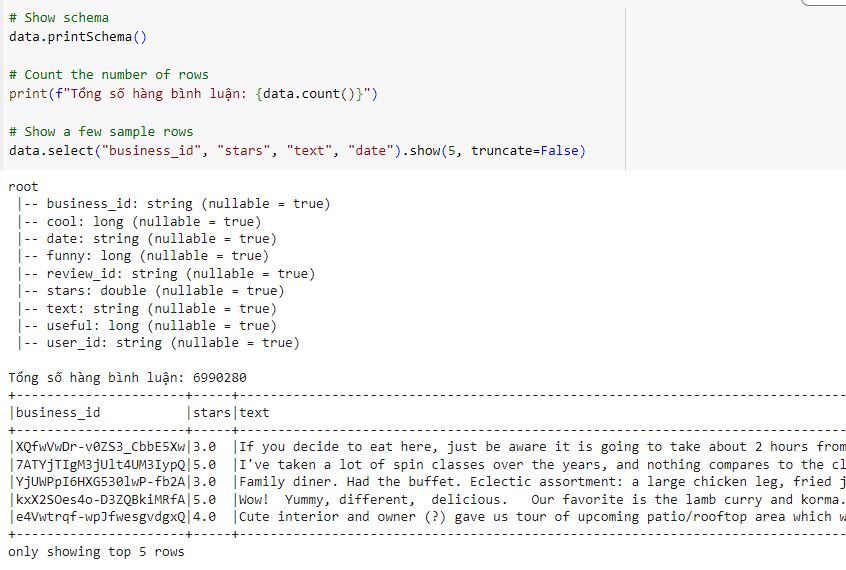
Hình . Đọc dữ liệu từ tệp JSON vào Spark DataFrame và hiển thị 10 dòng dữ liệu đầu

Dòng lệnh data = spark.read.json(path\_data) sử dụng Spark để đọc tệp JSON từ đường dẫn được lưu trong biến path\_data. Phương thức spark.read.json() của SparkSession cho phép đọc dữ liệu từ tệp JSON và trả về một DataFrame, một cấu trúc dữ liệu mạnh mẽ cho phép thực hiện các thao tác phân tích dữ liệu lớn. Trong trường hợp này, dữ liệu được nạp vào DataFrame với tên là data, giúp dễ dàng thao tác và phân tích các bản ghi từ tệp JSON bằng các công cụ và hàm của Spark.

****

Hình . 10 dòng đầu của dữ liệu

Dòng lệnh data.head(10) sử dụng phương thức head() của Spark DataFrame để trả về một danh sách chứa các dòng dữ liệu đầu tiên. Tham số 10 chỉ định rằng ta muốn lấy 10 dòng đầu tiên của DataFrame. Khác với phương thức show() (hiển thị dữ liệu dưới dạng bảng trong giao diện), head() trả về một danh sách các đối tượng Row. Mỗi đối tượng Row đại diện cho một dòng dữ liệu và cho phép thao tác với dữ liệu này như các đối tượng Python, giúp dễ dàng truy cập và xử lý các giá trị cụ thể trong các dòng.



Hình . Sơ đồ thuộc tính, tổng bình luận

* data.printSchema():

Phương thức này in ra schema (cấu trúc) của DataFrame data. Schema cung cấp thông tin về các cột trong DataFrame, bao gồm tên cột, kiểu dữ liệu (như StringType, IntegerType, v.v.), và thông tin liệu cột có thể chứa giá trị null hay không. Điều này giúp hiểu rõ cấu trúc và các loại dữ liệu mà DataFrame chứa đựng.

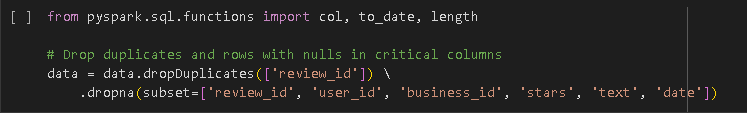
* print(f"Tổng số hàng bình luận: {data.count()}"):

Dòng này đếm số lượng hàng (bản ghi) trong DataFrame data bằng phương thức count() và in ra kết quả dưới dạng một câu thông báo: "Tổng số hàng bình luận: [số lượng hàng]". Đây là cách nhanh chóng để biết tổng số bản ghi hiện có trong DataFrame.

* data.select("business\_id", "stars", "text", "date").show(5, truncate=False):

Dòng này chọn các cột cụ thể từ DataFrame data bao gồm business\_id, stars, text, và date bằng phương thức select(). Sau đó, phương thức show(5, truncate=False) hiển thị 5 dòng đầu tiên của các cột đã chọn mà không cắt ngắn nội dung của các cột văn bản. Điều này giúp kiểm tra nhanh một số mẫu dữ liệu để hiểu rõ hơn về thông tin trong các cột được chọn.

### **3.1.4 Chuẩn bị dữ liệu**

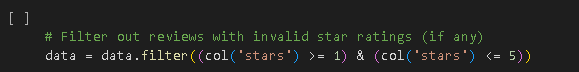


Hình . Làm sạch dữ liệu

**data = data.dropDuplicates(['review\_id'])**: Gọi phương thức này sẽ loại bỏ những bản ghi trùng lập

**data = data.dropna(subset=['review\_id', 'user\_id', 'business\_id', 'stars', 'text', 'date'])**: loại bỏ các giá trị của các cột quan trọng bao gồm reviewid, user\_id, business\_id,

stars, text, và date.



Hình . Lọc các bản ghi

**data.filter((col('stars') >= 1) & (col('stars') <= 5))**:

* Phương thức filter() của Spark DataFrame được sử dụng để lọc các hàng dữ liệu dựa trên điều kiện cho trước.
* Điều kiện trong trường hợp này là col('stars') >= 1 và col('stars') <= 5, nghĩa là chỉ những bản ghi mà giá trị trong cột stars nằm trong khoảng từ 1 đến 5 mới được giữ lại.
* col('stars') đại diện cho cột stars trong DataFrame.
* Toán tử & được sử dụng để kết hợp hai điều kiện thành một biểu thức logic duy nhất.
* Bất kỳ đánh giá nào có giá trị stars nhỏ hơn 1 hoặc lớn hơn 5 sẽ bị loại bỏ.



Hình . Kiểu dữ liệu ngày

**data.withColumn('date', ...)**:

* Phương thức withColumn() được sử dụng để tạo ra một cột mới hoặc thay thế một cột hiện có trong DataFrame data.
* Ở đây, cột 'date' sẽ được thay thế bằng phiên bản đã được chuyển đổi.

**to\_date(col('date'), 'yyyy-MM-dd HH:mm:ss')**:

* to\_date() là một hàm của Spark SQL dùng để chuyển đổi dữ liệu từ kiểu chuỗi thành kiểu ngày tháng (DateType).
* col('date') đại diện cho cột date trong DataFrame.
* 'yyyy-MM-dd HH:mm:ss' là định dạng của chuỗi ngày tháng trong cột date. Đây là mẫu mà Spark sẽ sử dụng để phân tích và chuyển đổi chuỗi thành kiểu DateType. Mẫu này cho biết rằng chuỗi ngày tháng có dạng năm-tháng-ngày giờ-phút-giây.



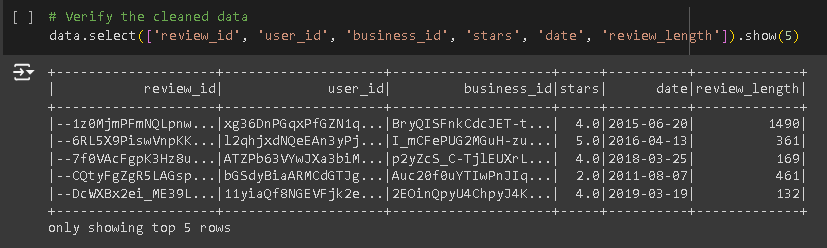
Hình . Thêm 1 cột mới vào DataFrame

**data.withColumn('review\_length', ...)**:

* Phương thức withColumn() được sử dụng để thêm một cột mới vào DataFrame hoặc để thay thế một cột hiện có. Ở đây, cột 'review\_length' sẽ được thêm vào DataFrame data.

**length(col('text'))**:

* length() là một hàm từ Spark SQL dùng để tính độ dài của một chuỗi văn bản.
* col('text') đại diện cho cột text trong DataFrame. Hàm length() sẽ tính số ký tự trong mỗi chuỗi văn bản ở cột text.



Hình . Kiểm tra dữ liệu đã được làm sạch

**data.select(['review\_id', 'user\_id', 'business\_id', 'stars', 'date', 'review\_length']). show(5)**:

Được sử dụng để chọn các cột cụ thể từ DataFrame data. Ở đây, các cột được chọn là review\_id, user\_id, business\_id, stars, date, và review\_length. Hiển thị 5 dòng đầu tiên từ DataFrame, chỉ bao gồm các cột đã chọn.

### **3.1.4 Phân tích dữ liệu**

Ảnh có chứa văn bản, Phông chữ, hàng, số

Mô tả được tạo tự động

Hình . Tính toán các thống kê

Cung cấp giá trị đếm, giá trị trung bình, độ lệch chuẩn, giá trị min và giá trị max cho các cột số đã chọn.

Ảnh có chứa văn bản, Phông chữ, hàng, ảnh chụp màn hình

Mô tả được tạo tự động

Hình . Phân tích dữ liệu đánh giá

Đoạn code này được sử dụng để phân tích dữ liệu đánh giá. Nó giúp ta trả lời các câu hỏi như:

* Mỗi mức sao có bao nhiêu đánh giá?
* Điểm đánh giá trung bình của mỗi doanh nghiệp là bao nhiêu?
* Điểm đánh giá trung bình thay đổi như thế nào theo từng năm?

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

Hình . Dùng textblob để tính toán mức độ phổ biến

Đoạn code này thực hiện các bước sau:

1. Cài đặt và nhập các thư viện cần thiết.
2. Định nghĩa một hàm để tính sentiment của một đoạn văn bản.
3. Đăng ký hàm này làm một UDF trong Spark SQL.
4. Áp dụng UDF cho từng đánh giá trong DataFrame để tính sentiment.
5. Hiển thị kết quả.

Có thêm một DataFrame mới với thêm một cột "sentiment" chứa giá trị sentiment của từng đánh giá. Sử dụng cột này để phân tích cảm xúc của khách hàng về sản phẩm hoặc dịch vụ của mình.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, số

Mô tả được tạo tự động

Hình . Điểm Rating trung bình theo xếp hạng sao

Một biểu đồ cột, trong đó trục hoành là các mức sao từ 1 đến 5, trục tung là điểm sentiment trung bình tương ứng. Chiều cao của mỗi cột đại diện cho điểm sentiment trung bình của các đánh giá có số sao tương ứng.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, vòng tròn

Mô tả được tạo tự động

Hình . Phân phối xếp hạng Rating trong data

Được một biểu đồ tròn, trong đó mỗi phần đại diện cho một mức sao, kích thước của phần tỷ lệ thuận với số lượng đánh giá ở mức sao đó.

Ảnh có chứa văn bản, hàng, Sơ đồ, biểu đồ

Mô tả được tạo tự động

Hình . Số lượng đánh giá theo thời gian

Một biểu đồ đường, trong đó trục hoành là các năm và trục tung là số lượng đánh giá. Đường biểu diễn sự thay đổi của số lượng đánh giá theo thời gian.

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình

Mô tả được tạo tự động

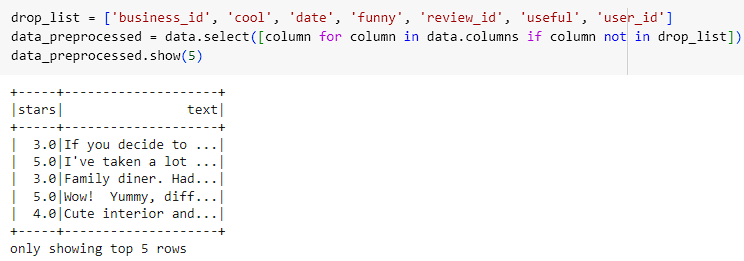
Ảnh có chứa văn bản, hàng, Sơ đồ, biểu đồ

Mô tả được tạo tự động

Hình . Xếp hạng sao trung bình theo thời gian

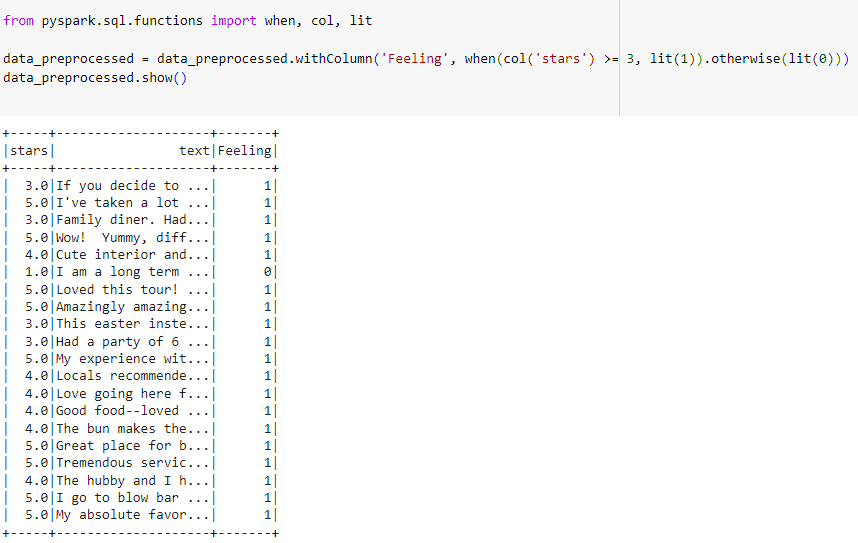
Một biểu đồ đường, trong đó trục hoành là các năm và trục tung là điểm đánh giá trung bình. Đường biểu diễn sự thay đổi của điểm đánh giá trung bình theo thời gian. Đường biểu diễn sẽ có màu san hô và các điểm dữ liệu được đánh dấu bằng các dấu tròn.

### **3.1.6 Modeling dữ liệu**



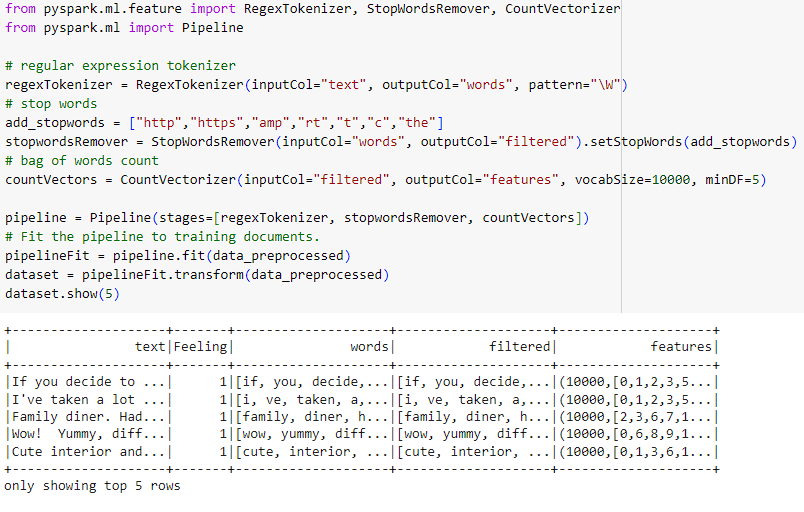
Hình . Loại các cột không cần thiết

Loại bỏ các cột thuộc tính chỉ giữ lại thuộc tinh “stars”, “text” để chuẩn bị dữ liệu cho mô hình phân loại



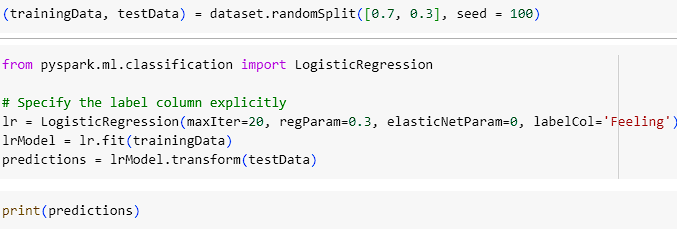
Hình . Dữ liệu cho mô hình

Thêm một cột mới có tên 'Feeling' vào DataFrame data\_preprocessed. Giá trị của cột 'Feeling' sẽ là 1 nếu giá trị của cột stars lớn hơn hoặc bằng 3, và 0 nếu nhỏ hơn 3. Nếu stars lớn hơn hoặc bằng 3, thì Feeling là 1 (có thể hiểu là cảm giác tích cực); ngược lại, nếu stars nhỏ hơn 3, thì Feeling là 0 (cảm giác tiêu cực).



Hình . Loại bỏ các Stopword và xây dựng Pipeline

Đoạn mã trên thiết lập một quy trình xử lý văn bản trong PySpark để chuẩn bị dữ liệu cho việc huấn luyện mô hình học máy. Đầu tiên, một đối tượng RegexTokenizer được sử dụng để phân tách các văn bản thành các từ riêng lẻ dựa trên biểu thức chính quy, giúp tách biệt các từ khỏi các ký tự không phải là chữ hoặc số. Tiếp theo, StopWordsRemover loại bỏ các từ dừng, như liên kết và các từ thông dụng, từ danh sách các từ đã phân tách để làm sạch dữ liệu và giảm thiểu nhiễu. Sau đó, CountVectorizer chuyển đổi danh sách các từ đã được lọc thành các vector số học, trong đó mỗi từ được đại diện bằng số lần xuất hiện của nó trong dữ liệu. Toàn bộ quy trình này được kết hợp trong một Pipeline, cho phép thực hiện liên tục các bước xử lý trên dữ liệu. Cuối cùng, pipeline được áp dụng lên dữ liệu đã tiền xử lý để chuyển đổi dữ liệu thành định dạng phù hợp cho việc huấn luyện mô hình học máy. Biểu đồ kết quả cho phép kiểm tra các đặc trưng đã được tạo ra và sẵn sàng cho các bước tiếp theo trong quy trình phân tích hoặc huấn luyện mô hình.

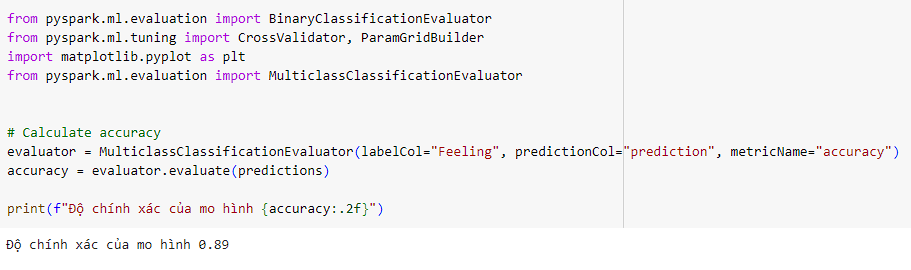


Hình . Train dữ liệu

Đoạn mã trên sử dụng PySpark để xây dựng một mô hình phân loại văn bản bằng thuật toán Logistic Regression. Đầu tiên, dữ liệu được chia thành hai tập con: tập huấn luyện và tập kiểm tra, với tỷ lệ 70% cho huấn luyện và 30% cho kiểm tra. Quá trình này được thực hiện bằng cách sử dụng phương thức randomSplit với một giá trị seed để đảm bảo tính ngẫu nhiên và khả năng tái tạo kết quả.

Tiếp theo, mô hình Logistic Regression được khởi tạo với các tham số cụ thể như số lần lặp tối đa là 20, tham số điều chuẩn (regularization) là 0.3, và sử dụng điều chuẩn L2 (Ridge Regression) bằng cách thiết lập tham số elasticNetParam bằng 0. Cột nhãn trong tập dữ liệu được chỉ định rõ ràng là 'Feeling', đây là cột mục tiêu mà mô hình cần dự đoán.

Sau khi thiết lập mô hình, quá trình huấn luyện được thực hiện trên tập huấn luyện (trainingData) để mô hình học các mối quan hệ giữa các đặc trưng của dữ liệu và nhãn mục tiêu. Cuối cùng, mô hình đã huấn luyện được áp dụng lên tập kiểm tra (testData) để dự đoán nhãn. Kết quả dự đoán bao gồm các nhãn dự đoán, xác suất dự đoán, và giá trị thô, tất cả đều được lưu trữ trong một DataFrame mới gọi là predictions.



Hình . Độ chính xác của mô hình

Đoạn mã này được sử dụng để đánh giá hiệu suất của mô hình phân loại đa lớp bằng cách tính toán độ chính xác (accuracy).

Đầu tiên, một đối tượng MulticlassClassificationEvaluator được tạo ra với các tham số cần thiết: cột nhãn thực tế (Feeling), cột nhãn dự đoán (prediction), và chỉ số hiệu suất cần tính toán là độ chính xác. Đối tượng này sẽ so sánh các nhãn dự đoán của mô hình với các nhãn thực tế trong tập dữ liệu dự đoán (predictions) để tính toán độ chính xác.

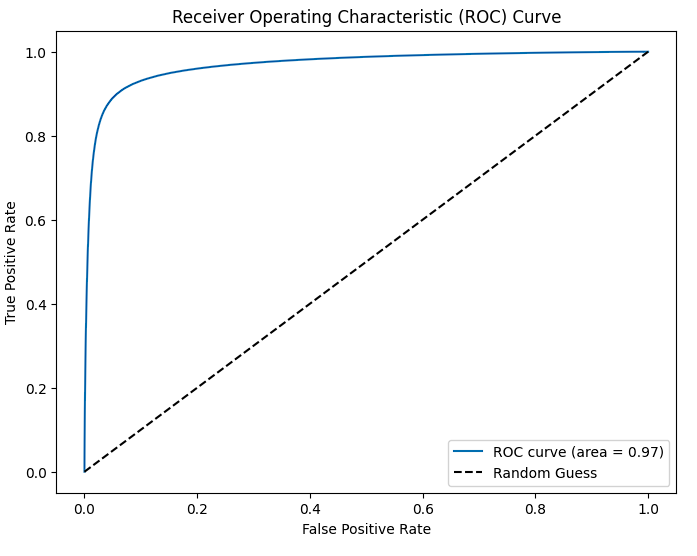
Sau khi thực hiện đánh giá, kết quả độ chính xác được lưu trữ trong biến accuracy. Cuối cùng, đoạn mã in ra độ chính xác của mô hình với định dạng hai chữ số thập phân, cho phép người dùng dễ dàng hiểu và đánh giá mức độ chính xác của mô hình phân loại đã được huấn luyện. Kết quả này cung cấp thông tin quan trọng về hiệu suất của mô hình trong việc phân loại đúng các ví dụ trong tập dữ liệu.



Hình . Trực quan hóa đường cong ROC

Đoạn mã này thực hiện việc đánh giá mô hình phân loại nhị phân bằng cách sử dụng chỉ số AUC-ROC (Area Under the Receiver Operating Characteristic Curve) và vẽ đường cong ROC để trực quan hóa hiệu suất của mô hình. Đầu tiên, một đối tượng BinaryClassificationEvaluator được tạo ra với chỉ số AUC-ROC để tính toán diện tích dưới đường cong ROC từ các dự đoán của mô hình. Kết quả AUC-ROC được in ra để đánh giá mức độ phân loại của mô hình, với giá trị càng cao cho thấy mô hình càng chính xác trong việc phân loại.

Sau đó, đoạn mã trích xuất dữ liệu đường cong ROC từ tóm tắt mô hình Logistic Regression và chuyển đổi dữ liệu này sang định dạng Pandas để dễ xử lý. Đường cong ROC được vẽ bằng cách sử dụng các tỷ lệ dương tính giả (FPR) và dương tính thật (TPR), và được so sánh với một đường chéo đại diện cho dự đoán ngẫu nhiên. Biểu đồ này cung cấp cái nhìn rõ ràng về hiệu suất của mô hình qua các ngưỡng phân loại khác nhau và cho phép người dùng đánh giá khả năng phân loại của mô hình một cách trực quan.



Hình . Kết quả đường ROC

# KẾT LUẬN

## Kết quả đạt được.

Sau quá trinh xử lý và huấn luyện dữ liệu và đưa ra được kết quả của mô hình có độ chính xác được 89%, hiệu suất phân loại rất tốt. Nó có khả năng phân biệt rất rõ ràng giữa các mẫu tích cực và tiêu cực của bài đánh giá thực khách.

Cụ thể:

* TPR cao: Mô hình có thể xác định được hầu hết các mẫu tích cực thực sự.
* FPR thấp: Mô hình ít khi đánh giá sai các mẫu tiêu cực thành tích cực.

## Hạn chế và hướng phát triển.

Đây là một đề tài tương đối hay và hấp dẫn, chúng em đã cố gắng đưa ứng dụng PySpark vào bài toán đánh giá nhà hàng Nhưng với những hạn chế về mặt trình độ cũng như thời gian thực hiện, kết thúc đề với nghiên cứu một mảng nhỏ trong lĩnh vực PySpark và DeepLearning, và kết quả dừng lại với việc dự đoán phân loại dữ liệu có sẳn. Sau này chúng em sẽ nghiên cứu thêm cách thu thập dữ liệu tự động để có thể xây dựng 1. Những định hướng dưới đây hi vọng sẽ cải thiện khả năng tương tác và tăng độ chính xác của dự đoán:

* Xây dựng một web app: để dự đoán bài đanh giá của khách hàng từ API của Google.
* Hệ thống huấn luyện dựa trên các thuật toán: sử dụng nhiều mô hình khác để tìm ra mô hình hiệu quả nhất.

# TÀI LIỆU THAM KHẢO

Phùng Thanh Bình (2011) dịch và diễn giải, mô hình hồi quy

Phùng Thanh Bình (2011) dịch và diễn giải, mô hình hồi quy

|  |  |
| --- | --- |
| [1] | "GPT-Neo: A Large Language Model for Natural Language Generation". |
| [2] | ThS. Hồ Khôi, "BÀI TẬP DEEPLAERNING\_THAMKHAO". |
| [3] | N. T. Tuấn, DEEP LEARNING CƠ BẢN. |
| [4] | Comunity, "Đường cong ROC," [Online]. Available: https://vi.wikipedia.org/wiki/%C4%90%C6%B0%E1%BB%9Dng\_cong\_ROC. [Accessed 2 12 2023]. |
| [5] | ThS. Phạm Đình Tài, "Tài liệu môn Dữ liệu lớn". |
| [6] | W. Feng, Learning Apache Spark with Python, December 05, 2021. |
| [7] | ThS. Bùi Tiến Đức, Tài liệu giảng môn Học máy và Ứng dụng. |
| [8] | TS. Võ Thị Hồng Thắm, Bài giảng môn Thu thập và tiền xử lý dữ liệu. |