TRƯỜNG ĐẠI HỌC ĐẠI NAM

KHOA CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN

HỌC PHẦN: DỮ LIỆU LỚN

ĐỀ TÀI: PHÂN TÍCH PHÂN KHÚC KHÁCH HÀNG VÀ ĐIỂM CHI TIÊU

Giảng viên: TS. TRẦN QUÝ NAM

ThS. LÊ THỊ THÙY TRANG

TT	Họ và Tên	Mã sv	Ngày Sinh	Lớp
1	NGUYỄN ĐỨC DUY	1671020060	26/12/2004	CNTT 16-01
2	NGUYỄN MINH ĐỨC	1671020089	16/08/2004	CNTT 16-01
3	NGUYỄN TẤT TOÀN	1671020330	18/01/2004	CNTT 16-01
4	NGUYỄN TIẾN DŨNG	1671020068	19/02/2004	CNTT 16-01

Hà Nội, năm 2025

TRƯỜNG ĐẠI HỌC ĐẠI NAM

KHOA CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN

HỌC PHẦN: DỮ LIỆU LỚN

ĐỀ TÀI: PHÂN TÍCH PHÂN KHÚC KHÁCH HÀNG VÀ DỰ ĐOÁN ĐIỂM CHI TIÊU

TT	Họ và Tên	Mã sv	Ngày Sinh	Điểm	
				Bằng Số	Bằng Chữ
1	**NGUYỄN ĐỨC DUY**	**1671020060**	**26/12/2004**		
2	**NGUYỄN MINH ĐỨC**	**1671020089**	**16/08/2004**		
3	**NGUYỄN TẤT TOÀN**	**1671020330**	**18/01/2004**		
4	**NGUYỄN TIẾN DŨNG**	**1671020068**	**19/02/2004**		

CÁN BỘ CHẤM THI 1 **CÁN BỘ CHẤM THI 2**

Trần Quý Nam **Lê Thị Thùy Trang**

Hà Nội, năm 2025

LỜI NÓI ĐẦU

Trong bối cảnh công nghệ thông tin ngày càng phát triển, khái niệm Dữ liệu lớn (Big Data) đã trở thành một trong những yếu tố quan trọng góp phần thúc đẩy sự đổi mới và tối ưu hóa quy trình ra quyết định trong nhiều lĩnh vực khác nhau. Dữ liệu lớn không chỉ đề cập đến khối lượng dữ liệu khổng lồ mà còn bao gồm các đặc tính quan trọng như tốc độ xử lý, tính đa dạng và giá trị tiềm năng. Việc khai thác và phân tích dữ liệu lớn giúp doanh nghiệp có thể hiểu rõ hơn về khách hàng, xu hướng thị trường cũng như tối ưu hóa hoạt động kinh doanh. Môn học Dữ liệu lớn cung cấp cho chúng em nền tảng lý thuyết vững chắc cùng với kỹ năng thực hành nhằm ứng dụng các phương pháp phân tích dữ liệu hiện đại trong giải quyết các bài toán thực tiễn.

Trong khuôn khổ học phần này, chúng em thực hiện nghiên cứu đề tài "Phân tích phân khúc khách hàng và điểm chi tiêu" dựa trên tập dữ liệu từ Kaggle. Tập dữ liệu này bao gồm thông tin về khách hàng như tuổi, giới tính, thu nhập hàng năm và điểm chi tiêu, giúp chúng em có cái nhìn sâu sắc về hành vi tiêu dùng của khách hàng. Mục tiêu chính của đề tài là áp dụng các phương pháp phân tích dữ liệu và kỹ thuật học máy để nhóm khách hàng thành các phân khúc có đặc điểm tương đồng, từ đó dự đoán mức độ chi tiêu của từng nhóm. Thông qua quá trình này, chúng em sẽ sử dụng các phương pháp như phân cụm, hồi quy và mô hình học máy khác nhằm tìm ra những đặc điểm quan trọng ảnh hưởng đến hành vi chi tiêu của khách hàng.

Ý nghĩa của môn học Dữ liệu lớn không chỉ dừng lại ở khía cạnh lý thuyết mà còn thể hiện rõ qua những ứng dụng thực tiễn. Trong lĩnh vực kinh doanh, việc phân tích phân khúc khách hàng và dự đoán hành vi tiêu dùng đóng vai trò quan trọng trong việc xây dựng chiến lược tiếp thị, tối ưu hóa chính sách giá cả và nâng cao trải nghiệm khách hàng. Ngoài ra, đề tài này cũng giúp chúng em tiếp cận với quy trình thực tế của một dự án khoa học dữ liệu, từ khâu thu thập, tiền xử lý dữ liệu, lựa chọn mô hình phù hợp cho đến đánh giá kết quả. Qua đó, chúng em không chỉ củng cố kiến thức chuyên môn mà còn phát triển kỹ năng tư duy phản biện, giải quyết vấn đề và làm việc nhóm – những kỹ năng cần thiết trong môi trường làm việc hiện đại.

Với tầm quan trọng của dữ liệu lớn trong thời đại số, việc nghiên cứu và ứng dụng các kỹ thuật phân tích dữ liệu vào thực tiễn không chỉ giúp tối ưu hóa hoạt động kinh doanh mà còn góp phần nâng cao năng lực cạnh tranh của doanh nghiệp. Đề tài "Phân tích phân khúc khách hàng và điểm chi tiêu" không chỉ là một bài tập ứng dụng kiến thức môn học mà còn mang ý nghĩa thực tiễn sâu sắc, giúp chúng em hiểu rõ hơn về cách dữ liệu có thể được khai thác để mang lại giá trị cho doanh nghiệp và xã hội.

MỤC LỤC

CHƯƠNG 1. TỔNG QUAN CƠ SỞ LÝ THUYẾT

1.1. Tổng quan về dữ liệu lớn

1.1.1. Khái niệm về dữ liệu truyền thống có cấu trúc (Structured data)

Dữ liệu truyền thống có cấu trúc là dạng dữ liệu được tổ chức theo một mô hình cố định, thường lưu trữ trong các hệ quản trị cơ sở dữ liệu quan hệ (RDBMS – Relational Database Management System) như MySQL, PostgreSQL, SQL Server và Oracle. Dữ liệu này được tổ chức theo bảng với các hàng và cột, mỗi cột đại diện cho một thuộc tính cụ thể của thực thể được quản lý.

Dữ liệu có cấu trúc cho phép dễ dàng thực hiện các thao tác truy vấn, cập nhật và phân tích thông qua các câu lệnh SQL. Đây là dạng dữ liệu phổ biến trong các hệ thống quản lý khách hàng (CRM), hệ thống kế toán, tài chính, và nhiều ứng dụng doanh nghiệp khác. Tuy nhiên, do tính cứng nhắc trong mô hình lưu trữ, dữ liệu có cấu trúc thường gặp khó khăn trong việc mở rộng khi có nhu cầu xử lý dữ liệu phi cấu trúc hoặc dữ liệu lớn.

1.1.2. Khái niệm dữ liệu lớn (Big Data)

Dữ liệu lớn (Big Data) là tập hợp dữ liệu có kích thước khổng lồ, tốc độ sinh ra nhanh và có sự đa dạng về cấu trúc. Dữ liệu lớn thường được mô tả thông qua mô hình 5V:

- Volume (Khối lượng): Dữ liệu lớn thường có dung lượng tính bằng terabyte (TB) hoặc petabyte (PB).
- Velocity (Tốc độ): Dữ liệu được tạo ra với tốc độ cao từ nhiều nguồn như mạng xã hội, cảm biến IoT, giao dịch tài chính.
- Variety (Đa dạng): Bao gồm dữ liệu có cấu trúc, bán cấu trúc và phi cấu trúc như văn bản, hình ảnh, video.
- Veracity (Độ tin cậy): Dữ liệu có thể bị nhiễu, sai lệch, đòi hỏi các phương pháp làm sạch dữ liệu.
- Value (Giá trị): Dữ liệu phải mang lại giá trị cho doanh nghiệp hoặc xã hội.

Công nghệ xử lý dữ liệu lớn bao gồm Hadoop, Apache Spark, cơ sở dữ liệu NoSQL như MongoDB và các công cụ phân tích dữ liệu. Việc khai thác dữ liệu lớn giúp các doanh

nghiệp có thể tối ưu hóa hoạt động, dự đoán xu hướng thị trường và nâng cao hiệu suất kinh doanh.

1.1.3. Phân tích dữ liệu tĩnh

Phân tích dữ liệu tĩnh là quá trình xử lý các tập dữ liệu đã được thu thập và lưu trữ theo một khoảng thời gian nhất định. Đây là loại dữ liệu không thay đổi theo thời gian thực, cho phép các nhà phân tích thực hiện thống kê mô tả, khai phá dữ liệu (Data Mining) và áp dụng các mô hình học máy giám sát (Supervised Learning).

Các phương pháp phân tích dữ liệu tĩnh bao gồm:

- Phân tích mô tả: Xác định các đặc trưng chung của dữ liệu như trung bình, phương sai, độ lệch chuẩn.
- Phân tích dự đoán: Sử dụng các mô hình hồi quy (Regression), cây quyết định (Decision Tree) để dự đoán giá trị tương lai.
- Phân tích phân cụm: Nhóm các đối tượng có đặc điểm tương tự nhau bằng các thuật toán như K-Means, Hierarchical Clustering.

Phân tích dữ liệu tĩnh được ứng dụng trong nhiều lĩnh vực như tài chính, tiếp thị, quản lý chuỗi cung ứng để đưa ra quyết định chiến lược dựa trên dữ liệu lịch sử.

1.1.4. Dữ liệu khi phân tích động, thời gian thực

Dữ liệu động là dạng dữ liệu thay đổi liên tục theo thời gian thực, yêu cầu các hệ thống phân tích phải xử lý ngay lập tức để cung cấp thông tin kịp thời. Điều này đặc biệt quan trọng trong các lĩnh vực như giao dịch chứng khoán, giám sát an ninh, quản lý hệ thống IoT và mạng xã hội.

Các công nghệ hỗ trợ phân tích dữ liệu thời gian thực bao gồm:

- Apache Kafka: Một hệ thống xử lý dữ liệu luồng mạnh mẽ, cho phép thu thập và phân phối dữ liệu theo thời gian thực.
- Apache Flink và Apache Storm: Các công cụ xử lý dữ liệu luồng mạnh mẽ hỗ trợ các thuật toán phức tạp.

- Cơ sở dữ liệu thời gian thực: Redis, Google BigQuery, Amazon Kinesis giúp xử lý và phân tích dữ liệu ngay lập tức.

Ứng dụng của phân tích dữ liệu thời gian thực có thể thấy rõ trong các hệ thống giao thông thông minh, giám sát y tế từ xa, hệ thống phát hiện gian lận tài chính và các nền tảng quảng cáo trực tuyến.

1.1.5. Bài toán phân tích dữ liệu lớn về text, hình ảnh, âm thanh, video và Web

Dữ liệu lớn không chỉ bao gồm dữ liệu có cấu trúc mà còn bao gồm dữ liệu phi cấu trúc như văn bản, hình ảnh, âm thanh, video và dữ liệu trên Web. Việc phân tích các dạng dữ liệu này đòi hỏi sự kết hợp của nhiều kỹ thuật và công nghệ tiên tiến.

- Phân tích văn bản (Text Mining & NLP): Sử dụng Xử lý ngôn ngữ tự nhiên (NLP) để trích xuất thông tin từ văn bản, hỗ trợ trong các ứng dụng chatbot, tìm kiếm thông tin và phân tích cảm xúc khách hàng.
- Phân tích hình ảnh (Computer Vision): Sử dụng các mô hình học sâu (Deep Learning) để nhận diện khuôn mặt, phát hiện vật thể, và chẩn đoán hình ảnh y tế.
- Nhận dạng âm thanh (Speech Recognition): Công nghệ nhận dạng giọng nói như Google Speech-to-Text, IBM Watson Speech giúp chuyển đổi giọng nói thành văn bản.
- Phân tích video: Kết hợp kỹ thuật thị giác máy tính và học máy để nhận diện chuyển động, giám sát giao thông, phân tích nội dung truyền thông.
- Trích xuất dữ liệu từ Web (Web Scraping): Sử dụng các công cụ như BeautifulSoup, Scrapy để thu thập dữ liệu từ các trang web, hỗ trợ trong phân tích hành vi người dùng và nghiên cứu thị trường.

1.2. Hadoop cơ bản

1.2.1. Giới thiệu chung về Hadoop

Hadoop là một nền tảng mã nguồn mở hỗ trợ lưu trữ và xử lý dữ liệu lớn phân tán trên nhiều máy tính bằng cách sử dụng mô hình lập trình đơn giản. Được phát triển bởi Apache Software Foundation, Hadoop giúp giải quyết bài toán xử lý dữ liệu lớn bằng cách chia nhỏ

dữ liệu và phân phối chúng trên nhiều nút trong cụm máy tính. Nhờ khả năng mở rộng linh hoạt và chi phí thấp, Hadoop đã trở thành nền tảng quan trọng trong lĩnh vực phân tích dữ liệu lớn.

Hadoop được thiết kế để chạy trên phần cứng phổ thông, có khả năng chịu lỗi cao và hỗ trợ xử lý dữ liệu hàng loạt cũng như thời gian thực. Nó bao gồm một hệ thống tệp phân tán mạnh mẽ (HDFS) và một hệ thống xử lý dữ liệu phân tán (MapReduce), cùng với nhiều công cụ bổ trợ khác để tối ưu hóa hiệu suất.

1.2.2. Chức năng nhiệm vụ của Hadoop

Hadoop có các chức năng và nhiệm vụ chính như sau:

- Lưu trữ dữ liệu lớn: Hệ thống tệp phân tán HDFS cho phép lưu trữ dữ liệu có dung lượng lớn trên nhiều máy tính.
- Xử lý dữ liệu song song: Sử dụng mô hình MapReduce để xử lý dữ liệu song song trên nhiều nút trong cụm máy tính.
- Chịu lỗi và phục hồi dữ liệu: Khi một nút gặp sự cố, Hadoop tự động sao chép dữ liệu sang các nút khác để đảm bảo tính sẵn sàng.
- Khả năng mở rộng: Hadoop có thể mở rộng dễ dàng bằng cách thêm các nút mới vào hệ thống mà không làm gián đoạn hoạt động.
- Hỗ trợ nhiều kiểu dữ liệu: Không chỉ xử lý dữ liệu có cấu trúc, Hadoop còn có thể làm việc với dữ liệu phi cấu trúc như văn bản, hình ảnh, video.

1.2.3. Kiến trúc Hadoop

Hadoop có kiến trúc phân tán bao gồm các thành phần chính:

- HDFS (Hadoop Distributed File System): Là hệ thống tệp phân tán chịu lỗi, cho phép lưu trữ dữ liệu lớn bằng cách chia nhỏ và phân phối trên nhiều nút.
- MapReduce: Là mô hình lập trình giúp xử lý dữ liệu song song bằng cách chia thành hai giai đoạn chính: Map (ánh xạ) và Reduce (tổng hợp).
- YARN (Yet Another Resource Negotiator): Quản lý tài nguyên trong cụm Hadoop và điều phối các ứng dụng xử lý dữ liệu.

- Common Utilities: Các thư viện hỗ trợ cho các mô-đun khác trong Hadoop hoạt động hiệu quả.

Kiến trúc Hadoop giúp tối ưu hóa hiệu suất xử lý dữ liệu lớn bằng cách phân chia và xử lý dữ liệu trên nhiều máy tính, giúp cải thiện tốc độ và độ tin cậy của hệ thống.

1.2.4. Quản trị Hadoop

Quản trị Hadoop bao gồm các công việc quan trọng để đảm bảo hệ thống hoạt động ổn định và hiệu quả:

- Cấu hình và triển khai cụm Hadoop: Cài đặt Hadoop trên nhiều nút và thiết lập các thông số phù hợp với nhu cầu sử dụng.
- Quản lý tài nguyên: Theo dõi và tối ưu hóa tài nguyên hệ thống để đảm bảo hiệu suất cao nhất.
- Bảo trì và giám sát: Kiểm tra nhật ký hệ thống, phát hiện lỗi và thực hiện các biện pháp khắc phục.
- Bảo mật dữ liệu: Thiết lập quyền truy cập, mã hóa dữ liệu và bảo vệ hệ thống khỏi các mối đe dọa an ninh.
- Sao lưu và phục hồi: Xây dựng chiến lược sao lưu dữ liệu để tránh mất mát thông tin quan trọng.

1.2.5. Các thành phần của Hadoop

Hadoop có nhiều thành phần mở rộng để hỗ trợ các nhu cầu xử lý dữ liệu khác nhau:

- HBase: Một cơ sở dữ liệu NoSQL chạy trên HDFS, hỗ trợ lưu trữ dữ liệu lớn theo mô hình key-value.
- Hive: Hệ thống kho dữ liệu giúp thực hiện truy vấn dữ liệu bằng ngôn ngữ SQL.
- Pig: Một nền tảng xử lý dữ liệu lớn có cú pháp dễ học, giúp người dùng phân tích dữ liệu mà không cần viết MapReduce.
- Sqoop: Công cụ hỗ trợ nhập xuất dữ liệu giữa Hadoop và các cơ sở dữ liệu quan hệ.
- Flume: Công cụ thu thập và xử lý dữ liệu log theo thời gian thực.

- Oozie: Hệ thống lập lịch và quản lý luồng công việc trong Hadoop.

Với hệ sinh thái phong phú, Hadoop không chỉ hỗ trợ xử lý dữ liệu lớn mà còn mở rộng khả năng phân tích chuyên sâu trong nhiều lĩnh vực như thương mại điện tử, tài chính, y tế, và truyền thông.

1.3. Giới thiệu ngôn ngữ lập trình R

1.3.1. Giới thiệu ngôn ngữ R

R là một ngôn ngữ lập trình và môi trường tính toán thống kê mạnh mẽ, được phát triển bởi Ross Ihaka và Robert Gentleman vào những năm 1990. R được sử dụng rộng rãi trong phân tích dữ liệu, thống kê và học máy nhờ vào thư viện phong phú và tính năng hỗ trợ trực quan hóa dữ liệu mạnh mẽ. Đây là một công cụ quan trọng trong lĩnh vực khoa học dữ liệu, giúp các nhà phân tích khai thác, xử lý và trực quan hóa dữ liệu một cách hiệu quả.

Ngôn ngữ R là một phần mềm mã nguồn mở, có thể chạy trên nhiều hệ điều hành như Windows, macOS và Linux. Với sự phát triển mạnh mẽ của cộng đồng người dùng, R liên tục được cập nhật với các gói thư viện hỗ trợ đa dạng lĩnh vực, từ tài chính, y tế đến trí tuệ nhân tạo.

Ngôn ngữ R được thiết kế chủ yếu để thực hiện các phép tính thống kê và phân tích dữ liệu. Một số đặc điểm nổi bật của R bao gồm:

- Hỗ trợ các phép tính thống kê phong phú: R cung cấp nhiều phương pháp phân tích dữ liệu như hồi quy, kiểm định giả thuyết, phân cụm và phân tích chuỗi thời gian.
- Thư viện mở rộng: Hàng ngàn gói thư viện có sẵn trên CRAN (Comprehensive R Archive Network) hỗ trợ xử lý dữ liệu, học máy, trực quan hóa và tối ưu hóa.
- Tích hợp dễ dàng với các ngôn ngữ khác: R có thể kết hợp với Python, C++, SQL để mở rộng khả năng xử lý dữ liệu.
- Khả năng trực quan hóa mạnh mẽ: R hỗ trợ nhiều thư viện như ggplot2, plotly để tạo biểu đồ chuyên nghiệp và trực quan hóa dữ liệu một cách hiệu quả.

- Môi trường tương tác: R có giao diện dễ sử dụng thông qua RStudio, giúp người dùng thực hiện các thao tác lập trình và phân tích dữ liệu một cách thuận tiện.

1.3.2. Giới thiệu cú pháp R

1.3.2.1. Biến và kiểu dữ liệu

R hỗ trợ nhiều kiểu dữ liệu như số, chuỗi, logic, vector, danh sách và ma trận. Ví dụ:

```
x <- 10  # Biến số nguyên
y <- "Hello"  # Chuỗi ký tự
z <- c(1, 2, 3, 4, 5)  # Vector số
```

1.3.2.2. Toán tử

R cung cấp nhiều loại toán tử:

Toán tử số học: +, -, *, /, %%, ^

Toán tử so sánh: ==, !=, >, <, >=, <=

Toán tử logic: &, |, !

1.3.2.3. Cấu trúc điều kiện và vòng lặp

R hỗ trợ các câu lệnh điều kiện và vòng lặp như:

```
if (x > 5) {
  print("x lớn hơn 5")
} else {
  print("x nhỏ hơn hoặc bằng 5")
}
for (i in 1:5) {
  print(i)
}
```

1.3.2.4. Hàm trong R

R cho phép người dùng tạo hàm để sử dụng lại trong nhiều tình huống:

```
my_function <- function(a, b) {
  return(a + b)
}
result <- my_function(3, 4)
print(result)
```

1.3.3. Sử dụng R trong phân tích dữ liệu lớn

1.3.3.1. Xử lý dữ liệu lớn bằng R

- Dplyr: Hỗ trợ xử lý dữ liệu lớn bằng các thao tác như lọc, sắp xếp, nhóm dữ liệu.
- Data.table: Cung cấp khả năng xử lý dữ liệu nhanh hơn so với data.frame truyền thống.
- SparkR và sparklyr: Tích hợp R với Apache Spark để phân tích dữ liệu lớn phân tán.

1.3.3.2. Trực quan hóa dữ liệu lớn

R cung cấp nhiều công cụ giúp trực quan hóa dữ liệu lớn một cách hiệu quả:

- ggplot2: Thư viện tạo biểu đồ với cú pháp đơn giản, mạnh mẽ.
- Plotly: Hỗ trợ trực quan hóa dữ liệu tương tác.
- Shiny: Xây dựng ứng dụng web tương tác để trình bày dữ liệu.

1.3.3.3. Phân tích thống kê và học máy

R có nhiều gói hỗ trợ các thuật toán học máy và phân tích thống kê:

- Caret: Cung cấp công cụ để huấn luyện và đánh giá mô hình học máy.
- RandomForest: Hỗ trợ thuật toán rừng ngẫu nhiên trong phân loại và hồi quy.
- XGBoost: Một thuật toán học máy hiệu suất cao cho các bài toán dự báo.

1.3.3.4. Tích hợp với hệ thống dữ liệu lớn

R có thể kết nối với các hệ thống dữ liệu lớn như:

- Cơ sở dữ liệu SQL (MySQL, PostgreSQL, Oracle): Thông qua gói RMySQL, RPostgreSQL.
- Hadoop và Spark: Sử dụng SparkR hoặc RHadoop để xử lý dữ liệu lớn trên nền tảng phân tán.
- API và Web Scraping: Kết nối với API, thu thập dữ liệu từ Web bằng rvest, httr.

1.4. Mô hình lập trình MapReduce

1.4.1. Giới thiệu MapReduce

MapReduce là một mô hình lập trình được phát triển bởi Google để xử lý dữ liệu lớn trên các hệ thống phân tán. Nó cung cấp một cách tiếp cận đơn giản nhưng mạnh mẽ để xử lý dữ liệu song song trên nhiều máy tính. Mô hình MapReduce bao gồm hai bước chính:

- Map: Chia nhỏ dữ liệu đầu vào thành các phần nhỏ hơn và xử lý chúng độc lập.
- Reduce: Tổng hợp kết quả từ bước Map và tạo ra đầu ra cuối cùng.

MapReduce đặc biệt phù hợp với các bài toán xử lý dữ liệu lớn như phân tích log, tìm kiếm văn bản, phân loại dữ liệu và trích xuất thông tin từ tập dữ liệu lớn.

1.4.2. Lập trình MapReduce

Lập trình MapReduce bao gồm việc viết hai hàm chính:

- Hàm Map: Nhận vào một tập dữ liệu, xử lý từng phần và tạo ra các cặp khóa-giá trị.
- Hàm Reduce: Nhận các cặp khóa-giá trị từ bước Map, tổng hợp và xử lý dữ liệu để tạo kết quả cuối cùng.

Ví dụ về lập trình MapReduce trong R:

```
library(dplyr)

# Dữ liệu đầu vào
```

```
data <- data.frame(text = c("apple banana apple", "banana orange apple", "orange banana orange"))
 # Hàm Map: Tách từ và đếm số lần xuất hiện
map_function <- function(text) {
  words <- unlist(strsplit(text, " "))
  data.frame(word = words, count = 1)
}
 # Áp dụng Map
mapped_data <- bind_rows(lapply(data$text, map_function))
 # Hàm Reduce: Tính tổng số lần xuất hiện của từng từ
reduce_function <- function(mapped_data) {
  mapped_data %>% group_by(word) %>% summarise(total_count = sum(count))
}
 # Áp dụng Reduce
reduced_data <- reduce_function(mapped_data)
print(reduced_data)
```

Hàm map_function tách các từ từ văn bản và tạo ra cặp khóa-giá trị, còn hàm reduce_function tổng hợp số lần xuất hiện của mỗi từ bằng cách nhóm theo từ và tính tổng.

1.4.3. Lập trình MapReduce dùng Hadoop MapReduce

Hadoop cung cấp một nền tảng mạnh mẽ để thực thi MapReduce trên các cụm máy tính lớn. Quy trình lập trình MapReduce trên Hadoop với R sử dụng thư viện rmr2 như sau:

```
 library(rmr2)
 # Hàm Map
map_function <- function(k, lines) {
```

```
  words <- unlist(strsplit(lines, " "))
  keyval(words, rep(1, length(words)))
}
 # Hàm Reduce
reduce_function <- function(word, counts) {
  keyval(word, sum(counts))
}
 # Tạo dữ liệu đầu vào trên HDFS
hdfs.data <- to.dfs(c("apple banana apple", "banana orange apple", "orange banana orange"))
 # Chạy MapReduce trên Hadoop
wordcount <- mapreduce(input = hdfs.data,
                       map = map_function,
                       reduce = reduce_function)
 # Lấy kết quả từ HDFS
from.dfs(wordcount)
```

Ví dụ này sử dụng rmr2 để thực thi MapReduce trên Hadoop, trong đó hàm map_function tách các từ và gán giá trị 1, còn reduce_function cộng tổng số lần xuất hiện của mỗi từ.

1.5. Mô hình lập trình Hadoop Spark

1.5.1. Các khái niệm xử lý dữ liệu lớn thời gian thực

Xử lý dữ liệu lớn thời gian thực là quá trình xử lý dữ liệu ngay khi dữ liệu được tạo ra, giúp đưa ra quyết định nhanh chóng mà không cần chờ đợi toàn bộ dữ liệu được thu thập. Các hệ thống xử lý thời gian thực thường yêu cầu độ trễ thấp, khả năng mở rộng cao và xử lý dòng dữ liệu liên tục. Một số ứng dụng phổ biến của xử lý dữ liệu lớn thời gian thực bao gồm:

- Phân tích dữ liệu giao dịch tài chính
- Giám sát và cảnh báo hệ thống mạng
- Dự đoán xu hướng thị trường
- Xử lý log hệ thống

Hadoop Spark là một trong những công nghệ hỗ trợ xử lý dữ liệu lớn thời gian thực hiệu quả nhờ khả năng xử lý dữ liệu nhanh hơn so với MapReduce truyền thống.

1.5.2. Khác biệt giữa Spark và Map Reduce

Apache Spark và Hadoop MapReduce đều là các công nghệ xử lý dữ liệu lớn nhưng có sự khác biệt quan trọng:

Đặc điểm	Hadoop MapReduce	Apache Spark
Kiểu xử lý	Batch Processing	Real-time & Batch Processing
Tốc độ	Chậm do lưu trữ trung gian trên HDFS	Nhanh hơn do lưu trữ dữ liệu trên RAM
Dễ sử dụng	Phải viết nhiều dòng code	Cung cấp API dễ sử dụng hơn
Hỗ trợ Streaming	Không hỗ trợ xử lý thời gian thực	Hỗ trợ Spark Streaming để xử lý dữ liệu real-time
Hỗ trợ Machine Learning	Không tích hợp sẵn	Có thư viện MLlib hỗ trợ Machine Learning

Spark tận dụng bộ nhớ RAM để tăng tốc xử lý, giúp giảm thời gian đọc/ghi dữ liệu từ đĩa, trong khi MapReduce phụ thuộc vào HDFS để lưu trữ dữ liệu trung gian.

1.5.3. Phân tích dữ liệu với mô hình Hadoop Spark

Hadoop Spark hỗ trợ nhiều thư viện mạnh mẽ cho phân tích dữ liệu lớn, bao gồm:

- Spark SQL: Hỗ trợ truy vấn dữ liệu bằng ngôn ngữ SQL.

- Spark Streaming: Xử lý dữ liệu thời gian thực từ Kafka, Flume, và các nguồn khác.
- MLlib: Thư viện học máy để xây dựng mô hình dự đoán.
- GraphX: Hỗ trợ xử lý dữ liệu đồ thị.

Ví dụ phân tích dữ liệu với Spark trong R sử dụng sparklyr:

```
library(sparklyr)
library(dplyr)
# Kết nối với Spark
sc <- spark_connect(master = "local")
# Đọc dữ liệu vào Spark
data <- copy_to(sc, mtcars, "mtcars")
# Phân tích dữ liệu
data %>% group_by(cyl) %>% summarise(avg_mpg = mean(mpg)) %>% collect()
# Ngắt kết nối Spark
spark_disconnect(sc)
```

Sparklyr để kết nối với Spark, tải dữ liệu vào Spark và thực hiện phân tích dữ liệu bằng cú pháp dplyr. Điều này giúp tận dụng khả năng xử lý phân tán của Spark mà vẫn giữ được sự tiện lợi của R.

1.6. Một số công cụ phát triển ứng dụng dữ liệu lớn

1.6.1. Apache Pig

Apache Pig là một nền tảng xử lý dữ liệu lớn trên Hadoop, cung cấp một ngôn ngữ lập trình cấp cao có tên Pig Latin để xử lý và phân tích dữ liệu một cách đơn giản hơn so với việc viết trực tiếp bằng MapReduce. Pig được thiết kế để xử lý các tập dữ liệu lớn một cách hiệu quả, cho phép người dùng thao tác trên dữ liệu mà không cần quan tâm đến các chi tiết kỹ thuật của MapReduce.

Đặc điểm chính của Apache Pig:

- Dễ sử dụng: Cung cấp Pig Latin, một ngôn ngữ lập trình đơn giản và dễ đọc.
- Hiệu suất cao: Có thể tối ưu hóa việc thực thi để giảm thời gian xử lý.

Linh hoạt: Hỗ trợ nhiều định dạng dữ liệu và có thể mở rộng với UDF (User Defined Functions).

1.6.2. Ngôn ngữ JAQL

JAQL (JSON Query Language) là một ngôn ngữ truy vấn chuyên biệt được thiết kế để xử lý dữ liệu dưới định dạng JSON trên Hadoop. Nó hỗ trợ khả năng mở rộng và có thể tích hợp với Hadoop để thực hiện các phép toán phân tán trên dữ liệu lớn.

Đặc điểm chính của JAQL:

- Hỗ trợ dữ liệu JSON một cách tự nhiên.
- Tích hợp với Hadoop để thực hiện các truy vấn dữ liệu lớn.
- Hỗ trợ nhiều thao tác xử lý dữ liệu như lọc, nhóm, và tổng hợp.

1.6.3. Chuyển dữ liệu vào Hadoop với Flume và Sqoop

- Apache Flume: Flume là một công cụ thu thập, tổng hợp và di chuyển dữ liệu log vào Hadoop một cách hiệu quả. Nó chủ yếu được sử dụng để thu thập dữ liệu từ các hệ thống ghi log như máy chủ web, máy chủ ứng dụng và gửi chúng vào HDFS.
- Apache Sqoop: Sqoop là một công cụ hỗ trợ chuyển đổi dữ liệu giữa hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) và Hadoop. Nó giúp nhập dữ liệu từ MySQL, PostgreSQL vào HDFS và ngược lại.

1.6.4. Sử dụng Hbase để truy xuất dữ liệu lớn

HBase là một hệ quản trị cơ sở dữ liệu phi quan hệ (NoSQL) được xây dựng trên Hadoop, hỗ trợ lưu trữ và truy xuất dữ liệu lớn theo mô hình bảng phân tán.

Đặc điểm chính của HBase:

- Hỗ trợ lưu trữ dữ liệu dạng bảng với kích thước lớn.
- Có khả năng truy xuất dữ liệu nhanh hơn so với HDFS.

- Hỗ trợ cập nhật dữ liệu theo thời gian thực.

1.7. Phân tích dữ liệu lớn BigSheets

1.7.1. BigSheets là gì?

BigSheets là một công cụ phân tích dữ liệu lớn dựa trên nền tảng Hadoop, được phát triển bởi IBM như một phần của IBM BigInsights. Nó cung cấp một giao diện bảng tính giúp người dùng có thể trích xuất, tổng hợp và trực quan hóa dữ liệu lớn mà không cần lập trình phức tạp. BigSheets giúp xử lý dữ liệu từ nhiều nguồn khác nhau như web, cơ sở dữ liệu và hệ thống lưu trữ.

1.7.2. Chức năng của BigSheets

BigSheets mang lại nhiều chức năng quan trọng trong phân tích dữ liệu lớn, bao gồm:

- Thu thập dữ liệu: Cho phép nhập dữ liệu từ nhiều nguồn khác nhau như tệp văn bản, cơ sở dữ liệu quan hệ, dữ liệu web và dữ liệu phi cấu trúc.
- Xử lý và làm sạch dữ liệu: Hỗ trợ các thao tác lọc, biến đổi, chuẩn hóa dữ liệu trước khi phân tích.
- Trực quan hóa dữ liệu: Cung cấp các biểu đồ, đồ thị giúp phân tích và hiển thị dữ liệu một cách trực quan.
- Tích hợp với Hadoop: Hoạt động trên Hadoop, giúp tận dụng khả năng mở rộng và xử lý dữ liệu phân tán.
- Phân tích nâng cao: Hỗ trợ các chức năng phân tích dữ liệu lớn như tìm xu hướng, phân cụm và dự báo.

1.7.3. BigSheets chuyên sâu

BigSheets cung cấp các tính năng nâng cao giúp người dùng khai thác tối đa sức mạnh của Hadoop:

- Truy vấn dữ liệu dễ dàng: BigSheets cung cấp giao diện bảng tính trực quan giúp người dùng dễ dàng truy vấn và phân tích dữ liệu mà không cần kiến thức sâu về Hadoop.

- Tích hợp với các nguồn dữ liệu lớn: Có khả năng xử lý dữ liệu từ HDFS, cơ sở dữ liệu quan hệ, API web và nhiều nguồn khác.
- Tùy chỉnh phân tích: Hỗ trợ người dùng tạo các công thức, bộ lọc tùy chỉnh để phân tích dữ liệu theo nhu cầu cụ thể.
- Tích hợp với các công cụ phân tích khác: BigSheets có thể tích hợp với các công cụ như IBM Watson, Spark và các hệ thống phân tích dữ liệu khác để mở rộng khả năng xử lý và khai thác dữ liệu.

1.7.4. IBM BigInsight

IBM BigInsights là một nền tảng phân tích dữ liệu lớn dựa trên Hadoop, trong đó BigSheets là một thành phần quan trọng. BigInsights mở rộng khả năng của Hadoop bằng cách cung cấp các công cụ phân tích mạnh mẽ, quản lý dữ liệu hiệu quả hơn và hỗ trợ nhiều tính năng nâng cao như:

- Công cụ quản lý Hadoop: Giúp triển khai và tối ưu hóa các cụm Hadoop một cách hiệu quả.
- Tích hợp với AI và Machine Learning: Cho phép phân tích dữ liệu nâng cao bằng các thuật toán học máy.
- Hỗ trợ xử lý dữ liệu phi cấu trúc: Có khả năng phân tích dữ liệu văn bản, hình ảnh và dữ liệu phi cấu trúc khác.
- Bảo mật và quản trị: Cung cấp các công cụ bảo mật và quản lý dữ liệu lớn theo tiêu chuẩn doanh nghiệp.

CHƯƠNG 2. MÔ TẢ TẬP DỮ LIỆU VÀ CÁC CÔNG NGHỆ SỬ DỤNG

2. 1. Mô tả tập dữ liệu

2.1.1. Giới thiệu về tập dữ liệu

Tập dữ liệu Customer Segmentation được lấy từ Kaggle, cung cấp thông tin về hồ sơ khách hàng trong một hệ thống bán lẻ. Mục tiêu chính của tập dữ liệu là hỗ trợ phân tích hành vi mua sắm và phân nhóm khách hàng để có thể xây dựng các chiến lược tiếp thị phù hợp.

Phân khúc khách hàng là một bài toán quan trọng trong khoa học dữ liệu, giúp doanh nghiệp hiểu rõ hơn về nhóm khách hàng tiềm năng và tối ưu hóa dịch vụ.

- Nguồn gốc và đặc điểm của tập dữ liệu
- Nguồn dữ liệu: Kaggle
- Số lượng mẫu (quan sát):có 143,505 mẫu quan sát (dòng dữ liệu), tương ứng với số lượng giao dịch mua bán được ghi nhận.
- Số lượng biến (đặc trưng): 9
  - InvoiceNo – Mã hóa đơn
  - StockCode – Mã sản phẩm
  - Description – Mô tả sản phẩm
  - Quantity – Số lượng mua
  - InvoiceDate – Ngày giao dịch
  - UnitPrice – Giá sản phẩm
  - CustomerID – Mã khách hàng
  - Country – Quốc gia của khách hàng
  - TotalPrice – Tổng giá trị đơn hàng (Quantity * UnitPrice)

Loại dữ liệu: Dữ liệu có cấu trúc, dạng bảng, gồm cả biến định lượng và biến định tính

Định dạng tập dữ liệu: CSV

2.1.2. Cấu trúc và mô tả các biến trong tập dữ liệu

Tập dữ liệu Online Retail gồm 143,505 quan sát và 9 biến, các biến bao gồm:

- InvoiceNo (Mã hóa đơn)
  - Loại dữ liệu: Số nguyên.
  - Mô tả: Mã số duy nhất xác định từng đơn hàng.
- StockCode (Mã sản phẩm)
  - Loại dữ liệu: Chuỗi ký tự.
  - Mô tả: Mã nhận diện duy nhất của sản phẩm trong danh mục hàng hóa.
- Description (Mô tả sản phẩm)
  - Loại dữ liệu: Chuỗi ký tự.
  - Mô tả: Tên và mô tả ngắn gọn về sản phẩm.
- Quantity (Số lượng)
  - Loại dữ liệu: Số nguyên.
  - Mô tả: Số lượng sản phẩm được mua trong mỗi đơn hàng.
- InvoiceDate (Ngày giao dịch)
  - Loại dữ liệu: Chuỗi ký tự (ISO datetime format).
  - Mô tả: Thời gian giao dịch, giúp theo dõi xu hướng mua hàng theo thời gian.
- UnitPrice (Giá sản phẩm)
  - Loại dữ liệu: Số thực.
  - Mô tả: Giá bán lẻ của từng sản phẩm trong đơn vị tiền tệ nhất định.
- CustomerID (Mã khách hàng)
  - Loại dữ liệu: Số nguyên.
  - Mô tả: Định danh duy nhất của khách hàng trong hệ thống.

- Country (Quốc gia)
  - Loại dữ liệu: Chuỗi ký tự.
  - Mô tả: Quốc gia của khách hàng thực hiện giao dịch.
- TotalPrice (Tổng giá trị đơn hàng)
  - Loại dữ liệu: Số thực.
  - Mô tả: Giá trị đơn hàng được tính bằng Quantity * UnitPrice.

2.1.3. Thống kê chung

- Số lượng quan sát: 143,505 giao dịch.
- Số lượng khách hàng: Nhiều khách hàng khác nhau được xác định bằng CustomerID.
- Trung bình số sản phẩm mỗi đơn hàng: 7.47 sản phẩm.
- Khoảng giá sản phẩm: Từ 0.01 đến 100 USD.
- Giá trị trung bình mỗi giao dịch: 12.77 USD.

2.1.4. Ứng dụng thực tiễn của tập dữ liệu

Tập dữ liệu này có thể áp dụng vào nhiều lĩnh vực khác nhau, bao gồm:

- Marketing và chiến lược bán hàng: Giúp doanh nghiệp tối ưu hóa chiến lược tiếp thị cho từng nhóm khách hàng.
- Dịch vụ khách hàng: Xây dựng chương trình ưu đãi phù hợp với từng phân khúc khách hàng.
- Dự đoán xu hướng tiêu dùng: Xác định mô hình chi tiêu của khách hàng để dự báo nhu cầu sản phẩm.

Dữ liệu cung cấp thông tin về hành vi mua sắm, hỗ trợ áp dụng các thuật toán phân cụm (K-Means) để nhóm khách hàng theo thói quen tiêu dùng. Đồng thời, dữ liệu này cũng giúp xây dựng mô hình dự đoán chi tiêu bằng các phương pháp hồi quy, cây quyết định hoặc mạng nơ-ron. Ứng dụng thực tiễn bao gồm tối ưu hóa chiến lược tiếp thị, quản lý khách hàng, và dự báo nhu cầu sản phẩm nhằm nâng cao hiệu quả kinh doanh.

2.2. Công nghệ sử dụng

2.2.1. Phân tích phân khúc khách hàng

2.2.1.1. Các thư viện sử dụng

Thư viện xử lý và thao tác dữ liệu:

- tidyverse: Tập hợp nhiều thư viện con giúp xử lý dữ liệu một cách hiệu quả, bao gồm:
  - dplyr: Hỗ trợ lọc (filter()), nhóm (group_by()), tổng hợp (summarise()), sắp xếp tidyverse: Tập hợp nhiều thư viện con giúp xử lý dữ liệu một cách hiệu quả, bao gồm:
  - dplyr: Hỗ trợ lọc (filter()), nhóm (group_by()), tổng hợp (summarise()), sắp xếp (arrange()) dữ liệu theo cách trực quan, dễ hiểu.
  - ggplot2: Hỗ trợ vẽ biểu đồ chuyên nghiệp với cú pháp linh hoạt.
  - readr: Hỗ trợ đọc dữ liệu từ các tệp CSV nhanh chóng.
  - tibble: Giúp hiển thị dữ liệu theo dạng bảng trực quan hơn so với data.frame.
  - purrr: Cung cấp các hàm lập trình hàm (functional programming), giúp tối ưu hóa việc xử lý dữ liệu.
- lubridate: Hỗ trợ xử lý ngày tháng, giúp dễ dàng chuyển đổi định dạng và trích xuất thông tin như năm, tháng, ngày từ dữ liệu thời gian.
- scales: Hỗ trợ định dạng dữ liệu số, chẳng hạn như hiển thị số có dấu phân cách (comma()), biểu diễn dữ liệu tỷ lệ (percent()), hoặc định dạng giá trị tiền tệ (dollar()).(arrange()) dữ liệu theo cách trực quan, dễ hiểu.

Thư viện làm việc với Apache Spark:

- sparklyr: Kết nối R với Apache Spark để xử lý dữ liệu lớn bằng Spark DataFrame thay vì sử dụng data.frame thông thường. Điều này giúp tối ưu hiệu suất khi làm việc với tập dữ liệu có dung lượng lớn.

- dbplyr: Hỗ trợ viết các truy vấn SQL trên Spark theo cú pháp dplyr, giúp làm việc với dữ liệu Spark thuận tiện hơn mà không cần viết SQL thuần.

Thư viện trực quan hóa dữ liệu:

- ggplot2: Cung cấp khả năng vẽ biểu đồ với giao diện đẹp mắt, hỗ trợ nhiều loại biểu đồ như biểu đồ cột, biểu đồ tròn, biểu đồ đường, và biểu đồ phân tán.
- ggthemes: Cung cấp nhiều chủ đề giao diện khác nhau cho ggplot2, giúp cải thiện tính thẩm mỹ của biểu đồ.
- maps: Chứa dữ liệu bản đồ thế giới, hỗ trợ vẽ bản đồ địa lý để hiển thị dữ liệu doanh thu theo từng quốc gia.

2.2.1.2. Xử lý dữ liệu

- Kết nối với Apache Spark: Dữ liệu lớn sẽ được xử lý bằng Apache Spark thông qua sparklyr, giúp tăng tốc độ xử lý thay vì dùng các phương pháp thông thường trong R.

```
sc <- spark_connect(master = "local")
```

- Đọc dữ liệu từ tập dữ liệu: Dữ liệu được nhập vào từ file CSV và lưu vào Spark DataFrame để tối ưu hiệu suất xử lý

```
df_spark <- spark_read_csv(sc, name = "retail_data", path = "D:/BTLBigDaTa/Online Retail.csv",

infer_schema = TRUE, delimiter = ",", header = TRUE)
```

- Làm sạch, tiền xử lý dữ liệu :
  - Loại bỏ các giá trị bị thiếu: Những dòng có giá trị NA trong cột CustomerID hoặc Description bị loại bỏ để tránh ảnh hưởng đến phân tích

```
df_spark <- df_spark %>%

 filter(!is.na(CustomerID) & !is.na(Description))
```

  - Loại bỏ những giao dịch không hợp lệ: Những giao dịch có số lượng sản phẩm (Quantity) hoặc giá sản phẩm (UnitPrice) nhỏ hơn hoặc bằng 0 bị loại bỏ.

```
df_spark <- df_spark %>%
```

```
  filter(Quantity > 0, UnitPrice > 0)
```

- Chuyển đổi kiểu dữ liệu: InvoiceDate được chuyển về kiểu ngày tháng (Date); CustomerID được chuyển về kiểu ký tự (character); Tạo thêm cột TotalPrice = Quantity * UnitPrice để tính tổng tiền của mỗi giao dịch; Trích xuất Year từ InvoiceDate để hỗ trợ phân tích theo năm.

```
df_spark <- df_spark %>%
 mutate(
  InvoiceDate = to_date(InvoiceDate),
  CustomerID = as.character(CustomerID),
  TotalPrice = Quantity * UnitPrice,
  Year = year(InvoiceDate)
 )
```

2.2.1.3. Kỹ thuật phân tích dữ liệu

- Tính doanh thu theo quốc gia: Nhóm dữ liệu theo Country, tính tổng doanh thu (TotalPrice), sau đó sắp xếp theo thứ tự giảm dần.

```
revenue_by_country <- df_spark %>%
 group_by(Country) %>%
 summarise(TotalRevenue = sum(TotalPrice, na.rm = TRUE)) %>%
 arrange(desc(TotalRevenue)) %>%
 collect()
```

- Xác định sản phẩm bán chạy nhất: Nhóm theo Description, tính tổng số lượng sản phẩm bán ra (Quantity) và sắp xếp giảm dần.

```
top_products <- df_spark %>%
 group_by(Description) %>%
```

```
summarise(TotalQuantity = sum(Quantity)) %>%
arrange(desc(TotalQuantity)) %>%
head(10) %>%
collect()
```

- Phân tích khách hàng theo RFM:Recency (R): Số ngày kể từ lần mua gần nhất.; Frequency (F): Số lần mua hàng duy nhất; Monetary (M): Tổng tiền chi tiêu.

```
rfm_df <- df_spark %>%
 group_by(CustomerID) %>%
 summarise(
  Recency = as.numeric(datediff(max(InvoiceDate), Sys.Date())),
  Frequency = n_distinct(InvoiceNo),
  Monetary = sum(TotalPrice, na.rm = TRUE)
 ) %>%
 collect()
```

2.2.1.4. Trực quan hóa dữ liệu

- Biểu đồ cột (Bar Chart):
  - Hiển thị Top 10 quốc gia có doanh thu cao nhất.
  - Hiển thị Top 10 sản phẩm bán chạy nhất.
- Bản đồ doanh thu theo quốc gia:
  - Sử dụng geom_map() để hiển thị doanh thu theo từng quốc gia trên bản đồ thế giới.
- Biểu đồ phân khúc khách hàng:
  - Biểu đồ cột để hiển thị số lượng khách hàng theo từng phân khúc.
  - Biểu đồ tròn (Pie Chart) để thể hiện tỷ lệ từng phân khúc khách hàng.

2.2.1.5. Lưu trữ và xuất dữ liệu

- Xuất dữ liệu phân khúc khách hàng RFM ra file CSV để sử dụng cho các phân tích tiếp theo.

  write.csv(rfm_df, "D:/BTLBigData/RFM_Segmentation.csv", row.names = FALSE)

- Ngắt kết nối với Spark sau khi hoàn thành xử lý để giải phóng tài nguyên.

  spark_disconnect(sc)

2.2.2. Dự đoán điểm chi tiêu

2.2.2.1. Các thư viện sử dụng

- tidyverse: Gói tổng hợp bao gồm nhiều thư viện con như dplyr, ggplot2, tidyr, giúp xử lý và trực quan hóa dữ liệu.
- ggplot2: Hỗ trợ vẽ biểu đồ trực quan để phân tích dữ liệu.
- caret: Hỗ trợ chia dữ liệu, xây dựng mô hình học máy và đánh giá hiệu suất.
- Metrics: Cung cấp các chỉ số đánh giá mô hình như RMSE, MAE.
- reshape2: Chuyển đổi dữ liệu sang định dạng phù hợp để trực quan hóa.
- dplyr: Hỗ trợ thao tác dữ liệu như lọc, nhóm và biến đổi dữ liệu.

2.2.2.2. Xử lý dữ liệu

- Đọc dữ liệu từ file CSV bằng read.csv().
- Chuyển đổi kiểu dữ liệu: CustomerID được chuyển thành kiểu ký tự để phù hợp với phân tích.
- Chia tập dữ liệu thành tập huấn luyện (80%) và tập kiểm tra (20%) bằng cách sử dụng sample().

2.2.2.3. Kỹ thuật phân tích

- Hồi quy tuyến tính (lm()): Dự đoán Điểm Chi Tiêu (Monetary Score) dựa trên các biến đầu vào như Recency, Frequency và Monetary.
- Dự đoán trên tập kiểm tra bằng predict().

- Đánh giá mô hình bằng các chỉ số:
  - RMSE (Root Mean Squared Error): Sai số bình phương trung bình.
  - MAE (Mean Absolute Error): Sai số tuyệt đối trung bình.
  - R^2 (R-squared): Đo lường độ phù hợp của mô hình với dữ liệu.

2.2.2.4. Trực quan hóa dữ liệu

- Phân phối điểm chi tiêu dự đoán bằng biểu đồ cột (geom_bar()).
- So sánh điểm chi tiêu thực tế và dự đoán:
  - Chọn ngẫu nhiên 20 khách hàng.
  - Chuyển đổi dữ liệu bằng melt() để đưa về định dạng phù hợp cho trực quan hóa.
  - Vẽ biểu đồ cột nhóm (geom_bar()) để so sánh điểm chi tiêu thực tế và dự đoán.

CHƯƠNG 3. KẾT QUẢ XỬ LÝ, PHÂN TÍCH DỮ LIỆU

3.1. Kết quả phân tích phân khúc khách hàng

3.1.1. Kiểm tra tổng quan dữ liệu

```
> # Kiểm tra tổng quan dữ liệu
> df_spark %>% glimpse()
Rows: ??
Columns: 8
Database: spark_connection
$ InvoiceNo   <chr> "536365", "536365", "536365", "536365", "536365", "536365", "536365", "53…
$ StockCode   <chr> "85123A", "71053", "84406B", "84029G", "84029E", "22752", "21730", "22633…
$ Description <chr> "WHITE HANGING HEART T-LIGHT HOLDER", "WHITE METAL LANTERN", "CREAM CUPID…
$ Quantity    <int> 6, 6, 8, 6, 6, 2, 6, 6, 6, 32, 6, 6, 8, 6, 6, 3, 2, 3, 3, 4, 4, 6, 3, 3, …
$ InvoiceDate <chr> "12/1/2010 8:26", "12/1/2010 8:26", "12/1/2010 8:26", "12/1/2010 8:26", "12/1/2010 8:26", "12/1/2010 8:26", "12/1/2010 8:26", "…
$ UnitPrice   <dbl> 2.55, 3.39, 2.75, 3.39, 3.39, 7.65, 4.25, 1.85, 1.85, 1.69, 2.10, 2.10, 3…
$ CustomerID  <int> 17850, 17850, 17850, 17850, 17850, 17850, 17850, 17850, 17850, 13047, 130…
$ Country     <chr> "United Kingdom", "United Kingdom", "United Kingdom", "United Kingdom", "…
```

3.1.2. Phân tích doanh thu theo quốc gia

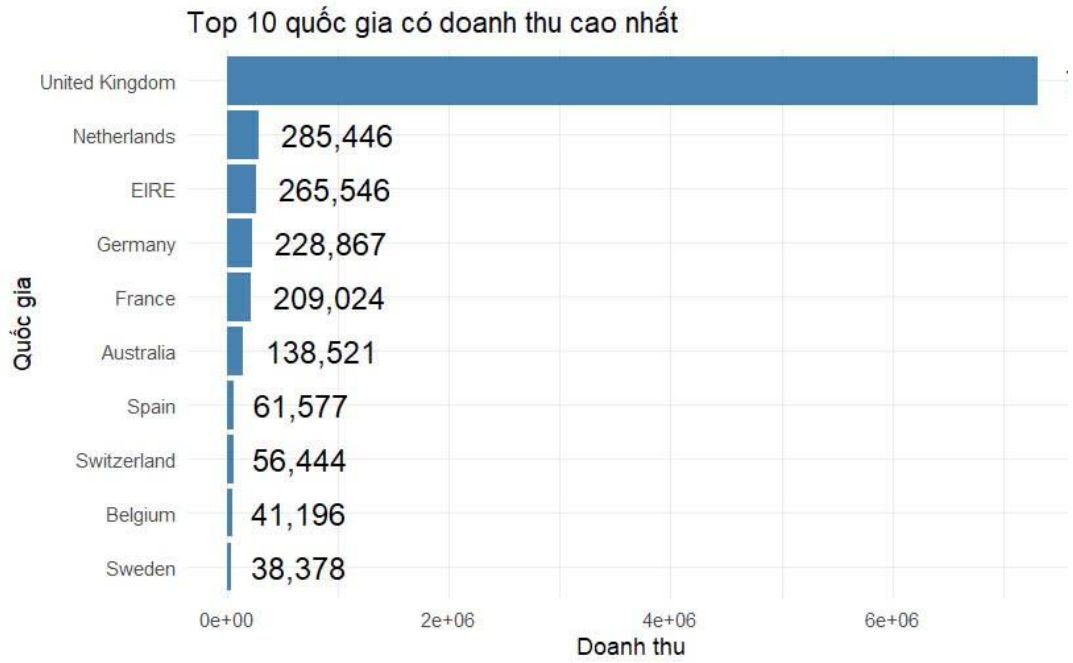
Hiển thị 10 quốc gia có doanh thu cao nhất:

```
> # Hiển thị top 10 quốc gia có doanh thu cao nhất
> head(revenue_by_country, 10)
# A tibble: 10 × 2
   Country        TotalRevenue
   <chr>                 <dbl>
 1 United Kingdom     7308392.
 2 Netherlands         285446.
 3 EIRE                265546.
 4 Germany             228867.
 5 France              209024.
 6 Australia           138521.
 7 Spain                61577.
 8 Switzerland          56444.
 9 Belgium              41196.
10 Sweden               38378.
```

- Biểu đồ top 10 quốc gia có doanh thu cao nhất:
  - United Kingdom có doanh thu cao hơn đáng kể so với các quốc gia còn lại, chiếm phần lớn tổng doanh thu.

Điều này cho thấy thị trường chính của doanh nghiệp tập trung vào Anh, có thể do số lượng khách hàng lớn hoặc doanh nghiệp có trụ sở tại đây.

- Nhóm quốc gia có doanh thu trung bình: Bao gồm Netherlands (Hà Lan), EIRE (Ireland), Germany (Đức), France (Pháp) với doanh thu dao động từ 209,024 đến 285,546.



Biểu đồ thể hiện Top 10 quốc gia có doanh thu cao nhất

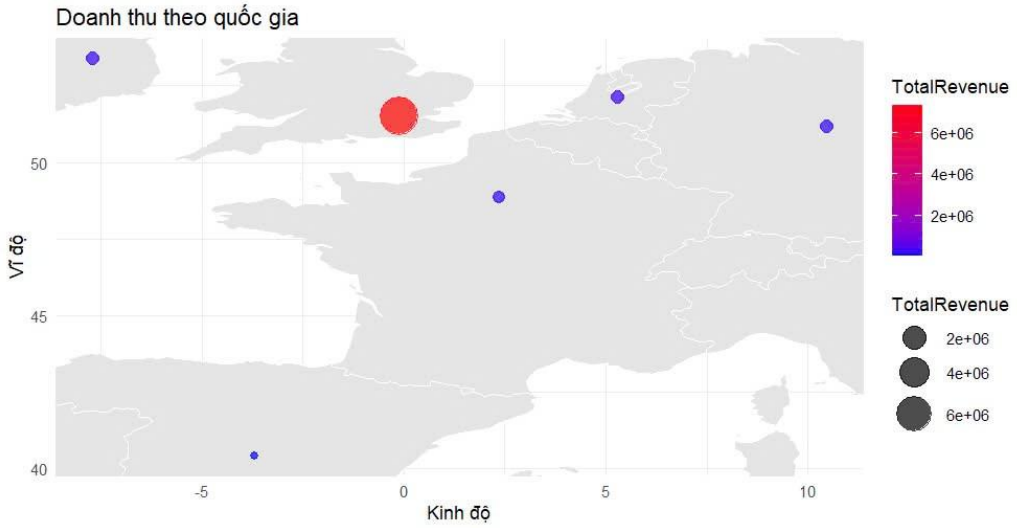
Đây có thể là các thị trường tiềm năng nhưng chưa phát triển mạnh bằng Anh.

- Nhóm quốc gia có doanh thu thấp hơn: Australia, Spain, Switzerland, Belgium, Sweden có doanh thu từ 38,378 đến 138,521.

Những nước này có thể là thị trường phụ hoặc có ít khách hàng hơn so với các nước trên.

Doanh thu của United Kingdom áp đảo hoàn toàn, có thể gấp nhiều lần so với quốc gia đứng thứ hai. Sự chênh lệch này có thể đến từ số lượng khách hàng lớn hơn, nhu cầu cao hơn hoặc chiến lược kinh doanh tập trung vào Anh.

Vẽ bản đồ doanh thu theo quốc gia:

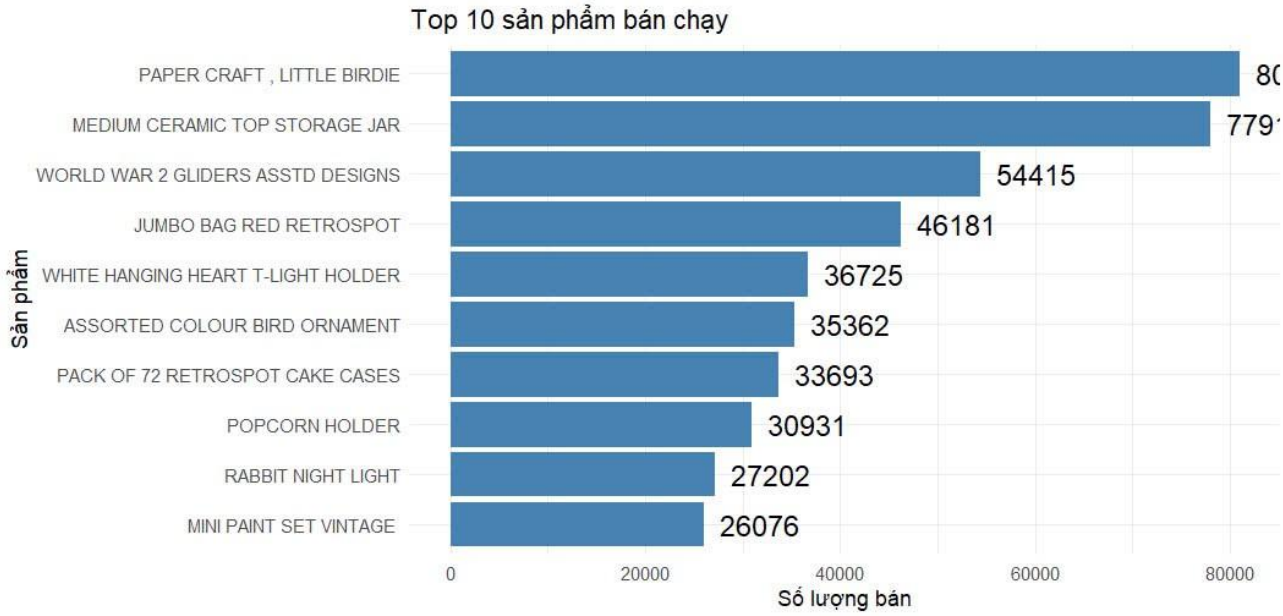


Bản đồ thể hiện doanh thu các quốc gia

- Bản đồ thể hiện doanh thu của các quốc gia trên bản đồ châu Âu dựa trên dữ liệu bán hàng.
- Mỗi chấm trên bản đồ tương ứng với một quốc gia có doanh thu trong dữ liệu.
- Kích thước của chấm biểu thị mức doanh thu: chấm càng lớn thì doanh thu càng cao.
- Màu sắc của chấm cũng thể hiện mức doanh thu:
  - Màu đỏ đậm → Doanh thu cao nhất.
  - Màu xanh dương → Doanh thu thấp hơn.
- Vương quốc Anh (United Kingdom) có doanh thu cao nhất, thể hiện qua chấm lớn màu đỏ nổi bật.
- Các quốc gia khác như Pháp, Đức, Hà Lan, Ireland, Tây Ban Nha có doanh thu thấp hơn, thể hiện qua chấm nhỏ hơn và có màu xanh.
- Bản đồ giúp xác định thị trường quan trọng, hỗ trợ chiến lược kinh doanh tập trung vào các quốc gia có doanh thu cao.

3.1.3. Phân tích sản phẩm bán chạy

Biểu đồ top 10 sản phẩm bán chạy:



Biểu đồ thể hiện Top 10 sản phẩm bán chạy

Biểu đồ thể hiện top 10 sản phẩm có số lượng bán cao nhất trong dữ liệu.

- Trục ngang (Số lượng bán) thể hiện số lượng đơn vị sản phẩm đã được bán ra.
- Trục dọc (Sản phẩm) liệt kê tên các sản phẩm bán chạy nhất.
- Chiều dài của thanh biểu diễn số lượng bán, thanh dài hơn tương ứng với số lượng bán cao hơn.
    - Sản phẩm bán chạy nhất là PAPER CRAFT, LITTLE BIRDIE với khoảng 80,000 đơn vị.
    - MEDIUM CERAMIC TOP STORAGE JAR đứng thứ hai với 77,900 đơn vị, chỉ thấp hơn một chút so với sản phẩm đứng đầu.
    - Các sản phẩm tiếp theo như WORLD WAR 2 GLIDERS ASSTD DESIGNS, JUMBO BAG RED RETROSPOT, và WHITE HANGING HEART T-LIGHT HOLDER có số lượng bán giảm dần.
    - Sản phẩm xếp thứ 10 là MINI PAINT SET VINTAGE với hơn 26,000 đơn vị bán ra.

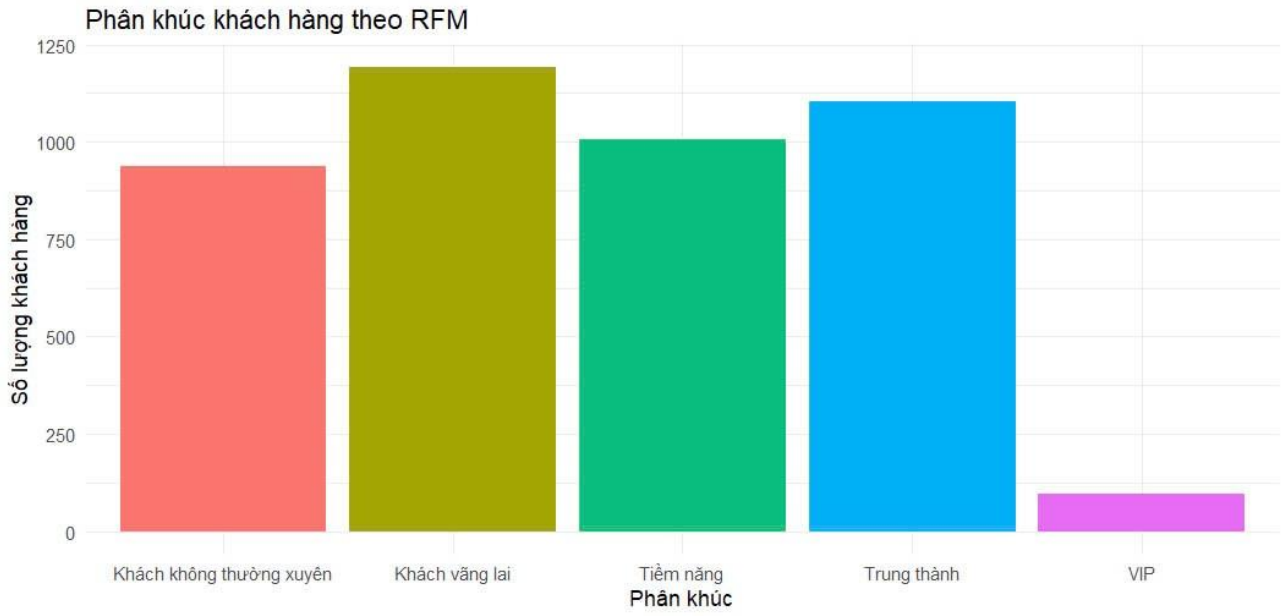
PAPER CRAFT, LITTLE BIRDIE và MEDIUM CERAMIC TOP STORAGE JAR là những sản phẩm được ưa chuộng nhất, doanh nghiệp có thể cân nhắc nhập kho nhiều hơn.

Dữ liệu này hữu ích cho việc tối ưu hóa danh mục sản phẩm, tập trung vào những mặt hàng có nhu cầu cao để tăng doanh thu.

3.1.4. Phân khúc khách hàng bằng RFM

Biểu đồ thể hiện phân khúc khách hàng theo mô hình RFM (Recency, Frequency, Monetary). Mô hình này phân loại khách hàng dựa trên:

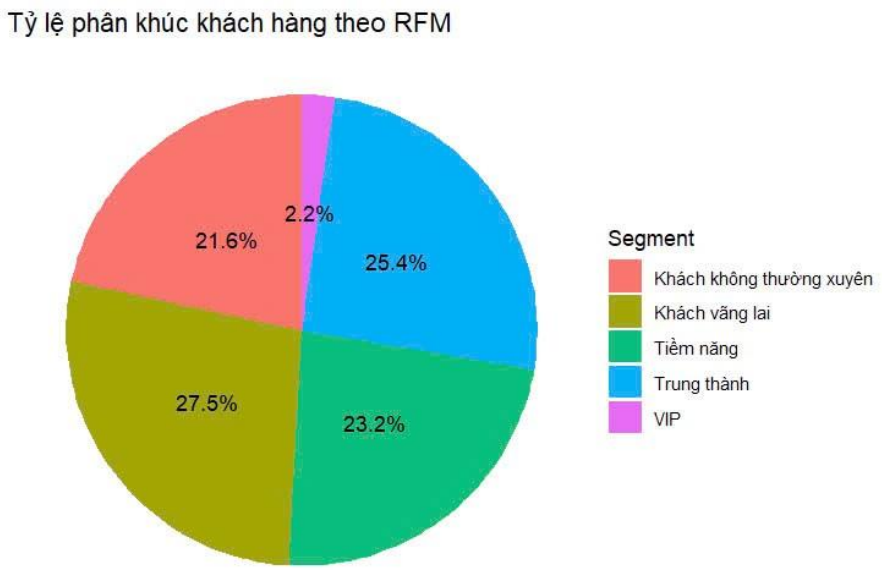
- Recency (R): Lần mua hàng gần nhất.
- Frequency (F): Tần suất mua hàng.
- Monetary (M): Tổng giá trị mua hàng.



Biểu đồ thể hiện phân khúc khách hàng theo điểm RFM

- Khách vãng lai (cao nhất, ~1.200 khách): Đây là nhóm khách hàng đã từng mua hàng nhưng không thường xuyên, có thể bị mất nếu không có chiến lược giữ chân.
- Trung thành (~1.100 khách): Nhóm này mua hàng thường xuyên, có giá trị cao, cần được duy trì bằng ưu đãi hoặc chương trình chăm sóc khách hàng.

- Khách không thường xuyên (~1.000 khách): Nhóm này có tần suất mua thấp, có thể là khách mới hoặc không gắn bó.
- Tiềm năng (~950 khách): Những khách hàng có tiềm năng trở thành trung thành nếu có chiến lược tiếp thị phù hợp.
- VIP (thấp nhất, ~100 khách): Đây là nhóm khách hàng có giá trị cao nhất, có thể chi tiêu lớn nhưng số lượng ít. Cần ưu đãi đặc biệt để duy trì.
- Nhận xét và ứng dụng thực tế:
  - Nhóm khách vãng lai và khách không thường xuyên khá lớn, cần chiến lược tiếp cận để biến họ thành khách hàng trung thành.
  - Nhóm khách trung thành chiếm tỷ lệ cao, cần duy trì bằng chương trình ưu đãi hoặc chăm sóc đặc biệt.
  - Nhóm VIP ít nhưng quan trọng, nên có các chương trình đặc quyền như ưu đãi cá nhân hóa, chăm sóc đặc biệt.
  - Nhóm tiềm năng có thể chuyển thành trung thành, nên đầu tư vào quảng cáo và khuyến mãi để tăng mức độ gắn kết.
  - Doanh nghiệp có thể tối ưu hóa chiến lược tiếp thị dựa trên dữ liệu này để cải thiện doanh thu và tỷ lệ giữ chân khách hàng.
- Biểu đồ tỷ lệ phân khúc khách hàng theo RFM:



Biểu đồ thể hiện tỷ lệ phân khúc khách hàng

- Khách vãng lai (27.5%): Đây là nhóm khách hàng lớn nhất, từng mua hàng nhưng không thường xuyên. Doanh nghiệp cần có chiến lược giữ chân để tăng tần suất mua.
- Trung thành (25.4%): Nhóm khách hàng ổn định, thường xuyên mua hàng và có giá trị cao.
- Tiềm năng (23.2%): Đây là những khách hàng có thể chuyển thành trung thành nếu có chiến lược marketing phù hợp.
- Khách không thường xuyên (21.6%): Nhóm này có tần suất mua thấp, có thể bao gồm khách hàng mới hoặc không gắn bó lâu dài.
- VIP (2.2%): Đây là nhóm khách hàng có giá trị mua cao nhất nhưng chiếm tỷ lệ nhỏ.

3.2. Kết quả dự đoán điểm chi tiêu

3.2.1. Các chỉ số đánh giá mô hình

Chương trình thực hiện dự đoán Monetary Score (M_Score) dựa trên ba yếu tố chính:

- Recency (R): Khoảng thời gian kể từ lần mua hàng gần nhất.
- Frequency (F): Số lần mua hàng của khách hàng.
- Monetary (M): Tổng số tiền khách hàng đã chi tiêu.

Mô hình này sử dụng hồi quy tuyến tính (Linear Regression) để dự đoán giá trị M_Score dựa trên dữ liệu huấn luyện.Sau khi huấn luyện mô hình, ta đánh giá chất lượng dự đoán bằng 3 chỉ số chính:

a) RMSE (Root Mean Squared Error - Sai số bình phương trung bình căn bậc hai)

Công thức:

$$RMSE = \sqrt{\frac{1}{n}\sum(y_{\text{thực tế}} - y_{\text{dự đoán}})^2}$$

RMSE phản ánh độ lệch trung bình của giá trị dự đoán so với thực tế. Giá trị càng nhỏ thì mô hình càng chính xác.

b) MAE (Mean Absolute Error - Sai số tuyệt đối trung bình)

Công thức:

$$MAE = \frac{1}{n}\sum|y_{\text{thực tế}} - y_{\text{dự đoán}}|$$

MAE đo lường sai số trung bình theo giá trị tuyệt đối, không bị ảnh hưởng nhiều bởi các giá trị ngoại lệ như RMSE.

c) R^2 (Hệ số xác định - R-squared score)

Công thức:

$$R^2 = 1 - \frac{\sum(y_{\text{thực tế}} - y_{\text{dự đoán}})^2}{\sum(y_{\text{thực tế}} - \bar{y})^2}$$

R^2 thể hiện tỷ lệ phương sai của dữ liệu thực tế được mô hình giải thích.

- $R^2 \approx 1$: Mô hình dự đoán tốt.
- $R^2 < 0.5$: Mô hình chưa hiệu quả, cần cải thiện.

3.2.2. Kết quả tính toán

```
> # Hiển thị kết quả
> cat("RMSE:", rmse_value, "\n")
RMSE: 1.094496
> cat("MAE:", mae_value, "\n")
MAE: 0.9343732
> cat("R²:", r2_value, "\n")
R²: 0.4144396
```

Sau khi huấn luyện mô hình hồi quy tuyến tính để dự đoán Monetary Score, ta thu được các chỉ số đánh giá sau:

- RMSE (Root Mean Square Error) = 1.094 → Sai số trung bình khoảng 1.094 đơn vị, cho thấy mô hình có mức độ sai số tương đối nhỏ nhưng vẫn có thể cải thiện.
- MAE (Mean Absolute Error) = 0.934 → Sai số tuyệt đối trung bình khoảng 0.934, thấp hơn RMSE, chứng tỏ sai số trung bình giữa dự đoán và thực tế là dưới 1 đơn vị.
- R^2 (R-squared) = 0.414 → Mô hình giải thích được 41.4% biến động của M_Score, cho thấy mô hình chỉ có độ chính xác trung bình và chưa thực sự mạnh.

Kết luận: Sai số RMSE và MAE tương đối nhỏ, nhưng vẫn có thể giảm xuống bằng cách cải thiện mô hình. R^2 = 0.414 thấp, mô hình chưa mô tả được phần lớn sự biến động của M_Score, cần bổ sung các yếu tố quan trọng khác để nâng cao độ chính xác.

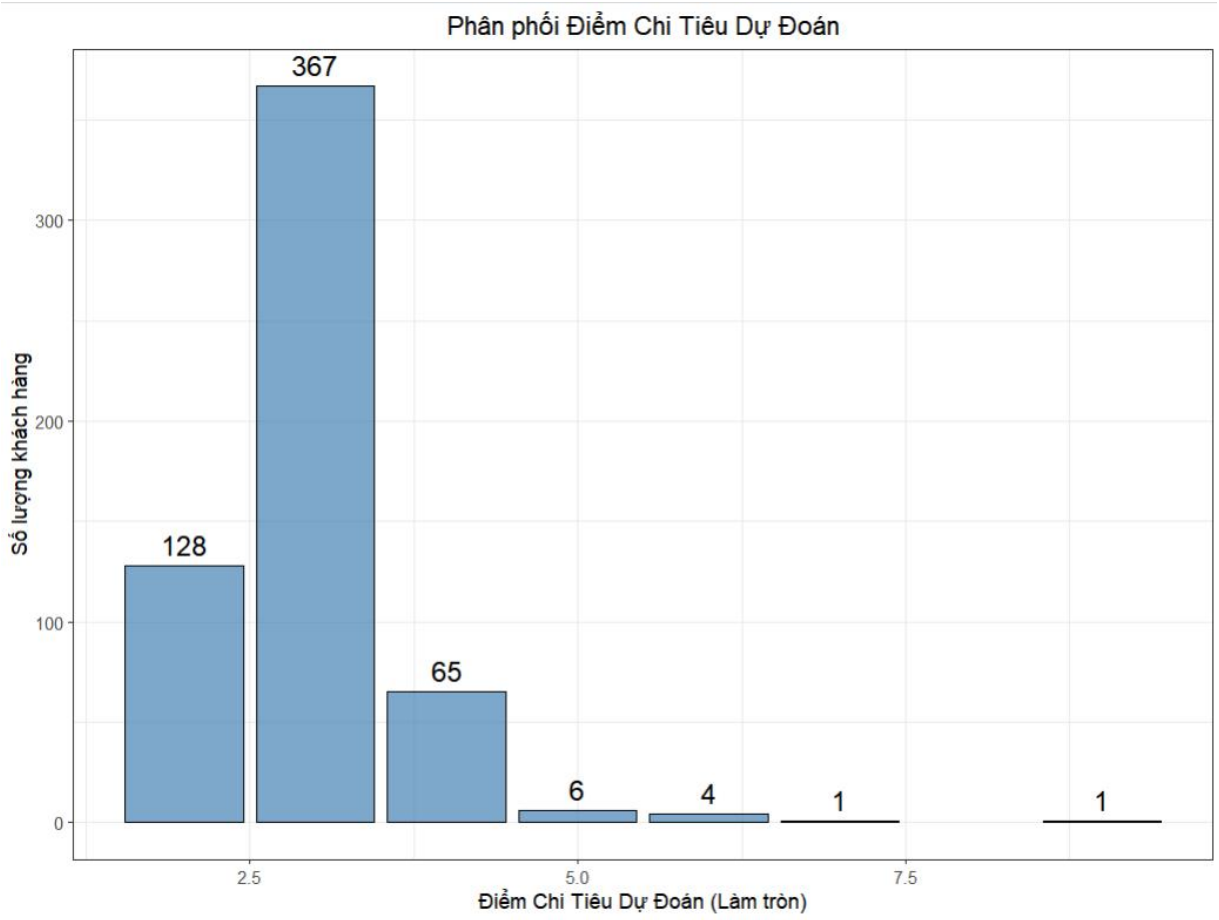
3.2.3. Trực quan hóa kết quả dự đoán điểm chi tiêu

Biểu đồ hiển thị sự phân phối của điểm chi tiêu dự đoán (M_Score) của khách hàng.

Trục X: Điểm chi tiêu dự đoán đã làm tròn.

Trục Y: Số lượng khách hàng có điểm chi tiêu dự đoán tương ứng.

Các thanh màu xanh dương đại diện cho số lượng khách hàng thuộc từng mức điểm chi tiêu.



Phân bố bị lệch về phía điểm chi tiêu thấp:

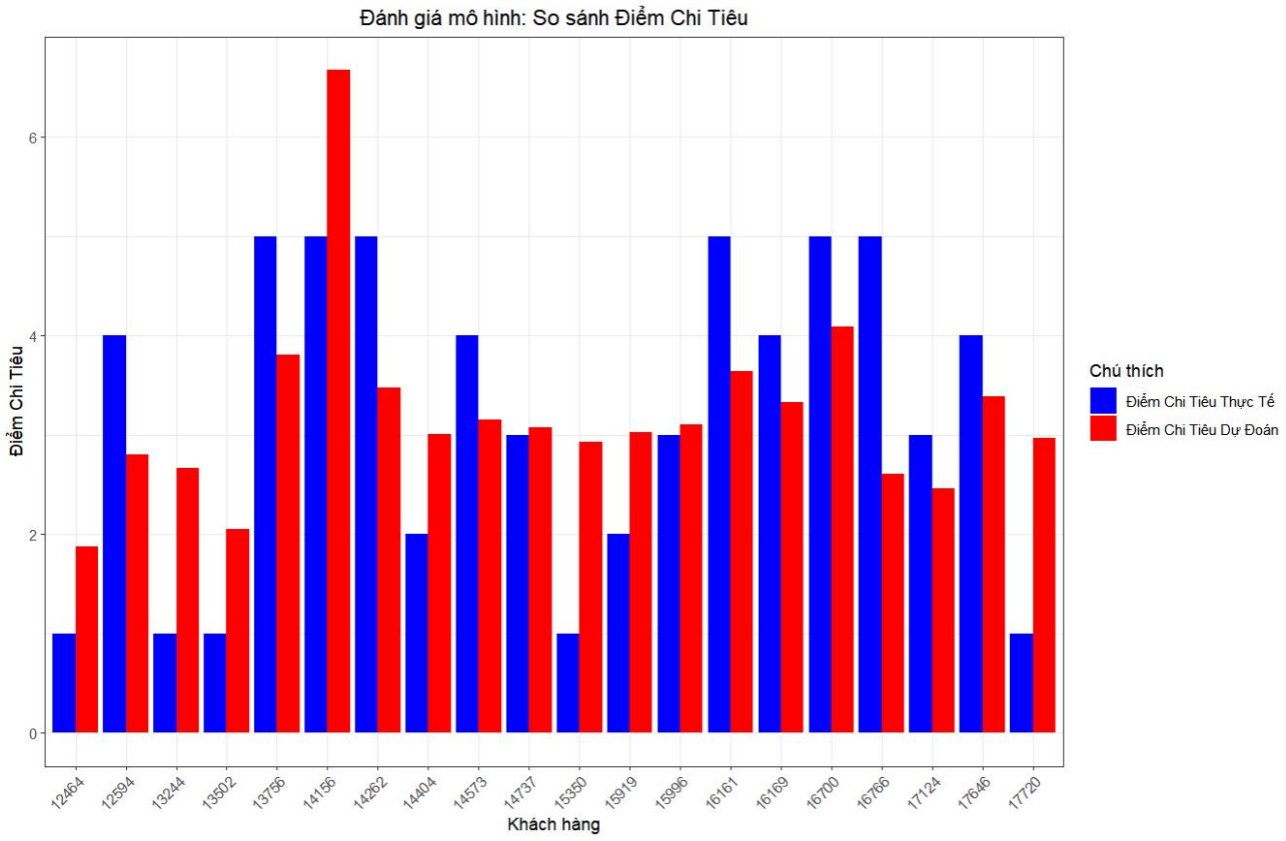
- Phần lớn khách hàng có M_Score dự đoán từ 2 đến 3, với nhóm M_Score = 3 chiếm nhiều nhất (367 khách hàng).
- Rất ít khách hàng có điểm dự đoán cao (chỉ 1 khách hàng có điểm chi tiêu khoảng 8).

Mô hình có xu hướng dự đoán điểm chi tiêu thấp:

- Vì phần lớn dữ liệu rơi vào khoảng 2 - 3, mô hình có thể chưa phản ánh tốt các khách hàng có mức chi tiêu cao hơn.

3.2.4. So sánh điểm chi tiêu

Biểu đồ trên thể hiện sự so sánh giữa Điểm Chi Tiêu Thực Tế (màu xanh) và Điểm Chi Tiêu Dự Đoán (màu đỏ) của một số khách hàng ngẫu nhiên. Điều này giúp đánh giá độ chính xác của mô hình dự đoán. (Chọn ngẫu nhiên 20 khách hàng)



Xu hướng chung giữa thực tế và dự đoán khá tương đối

- Ở nhiều khách hàng, chiều cao của cột màu đỏ (dự đoán) gần bằng với cột màu xanh (thực tế), cho thấy mô hình có khả năng dự đoán khá tốt.

Sai lệch ở một số khách hàng

- Có những trường hợp mô hình dự đoán thấp hơn thực tế (cột đỏ ngắn hơn cột xanh), mô hình đã đánh giá thấp mức chi tiêu của một số khách hàng.
- Một số khách hàng có mô hình dự đoán cao hơn thực tế (cột đỏ cao hơn cột xanh), mô hình đã phóng đại chi tiêu của họ.

Ví dụ cụ thể:

- Khách hàng 14156 có điểm chi tiêu thực tế thấp nhưng mô hình dự đoán cao hơn hẳn.
- Khách hàng 16706 có điểm chi tiêu thực tế cao nhưng dự đoán lại thấp hơn.
- Những khách hàng khác có sai lệch nhỏ hơn, mô hình dự đoán khá ổn với sai số không quá lớn.

Kết luận: Mô hình dự đoán khá ổn với sai số nhỏ và cần cải thiện độ chính xác cho các khách hàng có điểm chi tiêu cao hoặc thấp.

3.3. Ứng dụng thực tiễn

Hai chương trình phân tích và dự đoán này có nhiều ứng dụng thực tiễn trong thương mại điện tử, ngân hàng, tiếp thị, bán lẻ, bảo hiểm và quản lý khách hàng. Với chương trình phân tích phân khúc khách hàng bằng RFM, doanh nghiệp có thể xác định nhóm khách hàng tiềm năng, tối ưu chiến lược quảng cáo, cá nhân hóa chương trình khuyến mãi và chăm sóc khách hàng để tăng mức độ trung thành. Bên cạnh đó, việc dự đoán khách hàng có nguy cơ rời bỏ giúp doanh nghiệp triển khai các biện pháp giữ chân phù hợp. Trong lĩnh vực ngân hàng và bảo hiểm, phân khúc khách hàng hỗ trợ đánh giá rủi ro tín dụng, thiết kế các gói sản phẩm phù hợp với từng nhóm khách hàng dựa trên hành vi chi tiêu. Đối với chương trình dự đoán điểm chi tiêu bằng mô hình hồi quy, doanh nghiệp có thể dự báo xu hướng mua sắm, tối ưu chiến lược giá, dự đoán doanh thu và điều chỉnh chính sách marketing. Các công ty tài chính và bảo hiểm có thể sử dụng mô hình này để đánh giá khả năng chi tiêu và rủi ro của khách hàng, từ đó tối ưu hóa chiến lược cấp tín dụng. Trong thương mại điện tử và bán lẻ, dự đoán nhu cầu tiêu dùng giúp doanh nghiệp quản lý hàng tồn kho hiệu quả và cá nhân hóa gợi ý sản phẩm. Khi triển khai hệ thống trên Spark để xử lý dữ liệu lớn, các ngân hàng, chuỗi siêu thị và nền tảng thương mại điện tử như Amazon, Shopee có thể áp dụng mô hình trên quy mô hàng triệu khách hàng, giúp tối ưu hóa trải nghiệm người dùng theo thời gian thực. Nhìn chung, hai chương trình này đóng vai trò quan trọng trong việc giúp doanh nghiệp hiểu rõ hành vi khách hàng, tối ưu chiến lược kinh doanh, giảm thiểu rủi ro tài chính và nâng cao hiệu quả tiếp thị.

KẾT LUẬN

Trong bối cảnh bùng nổ dữ liệu hiện nay, việc khai thác và phân tích dữ liệu lớn đóng vai trò quan trọng trong nhiều lĩnh vực, đặc biệt là kinh doanh và tiếp thị. Đề tài "Phân tích phân khúc khách hàng và điểm chi tiêu" đã giúp chúng em áp dụng các phương pháp khoa học dữ liệu để khám phá hành vi tiêu dùng của khách hàng thông qua tập dữ liệu thực tế từ Kaggle. Quá trình thực hiện nghiên cứu đã mang lại nhiều kết quả đáng chú ý, từ việc phân khúc khách hàng dựa trên mô hình RFM đến việc xây dựng mô hình hồi quy dự đoán điểm chi tiêu, qua đó cung cấp những thông tin hữu ích giúp tối ưu hóa chiến lược kinh doanh.

Kết quả phân tích phân khúc khách hàng cho thấy sự khác biệt rõ rệt giữa các nhóm khách hàng, giúp doanh nghiệp xác định được nhóm khách hàng trung thành, khách hàng tiềm năng cũng như những đối tượng có nguy cơ rời bỏ. Việc ứng dụng mô hình RFM không chỉ giúp nhận diện hành vi mua sắm mà còn hỗ trợ doanh nghiệp trong việc xây dựng các chiến lược tiếp cận phù hợp, tối ưu hóa chương trình chăm sóc khách hàng và nâng cao hiệu quả tiếp thị. Đặc biệt, nhóm khách hàng trung thành và khách hàng VIP có giá trị mua sắm cao cần được ưu tiên duy trì, trong khi nhóm khách hàng vãng lai và khách không thường xuyên có thể trở thành mục tiêu của các chiến dịch tiếp thị cá nhân hóa nhằm tăng cường mức độ gắn kết.

Bên cạnh đó, mô hình dự đoán điểm chi tiêu dựa trên hồi quy tuyến tính đã cho thấy khả năng ước lượng tương đối chính xác xu hướng chi tiêu của khách hàng. Mặc dù hệ số R^2 còn hạn chế, nhưng kết quả thu được đã phản ánh phần nào mối quan hệ giữa tần suất mua hàng, lần mua gần nhất và tổng giá trị giao dịch với điểm chi tiêu. Mô hình có thể được cải thiện hơn nữa bằng cách bổ sung các đặc trưng quan trọng khác hoặc thử nghiệm các thuật toán học máy nâng cao hơn như Random Forest hay XGBoost để nâng cao độ chính xác của dự đoán. Những ứng dụng thực tiễn từ nghiên cứu này rất rộng rãi, không chỉ dừng lại ở việc phân tích khách hàng mà còn giúp doanh nghiệp dự đoán nhu cầu, tối ưu hóa hàng tồn kho và cải thiện chiến lược giá. Việc triển khai mô hình trên các nền tảng dữ liệu lớn như Apache Spark cũng mở ra tiềm năng áp dụng trên quy mô lớn hơn, giúp các công ty thương mại điện tử, ngân hàng và bảo hiểm tận dụng dữ liệu để nâng cao năng lực cạnh tranh.

Nhìn chung, thông qua đề tài này, chúng em đã có cơ hội tiếp cận và thực hành các kỹ thuật phân tích dữ liệu lớn, từ tiền xử lý, khai thác thông tin, xây dựng mô hình đến đánh giá kết quả. Quá trình thực hiện không chỉ giúp chúng em củng cố kiến thức về dữ liệu lớn và học máy mà còn phát triển các kỹ năng quan trọng như tư duy phân tích, giải quyết vấn đề và làm việc nhóm. Những bài học rút ra từ nghiên cứu này sẽ là nền tảng quan trọng để chúng em tiếp tục phát triển trong lĩnh vực khoa học dữ liệu và ứng dụng công nghệ vào thực tiễn.

DANH MỤC TÀI LIỆU THAM KHẢO

[1]. Kaggle.(n.d.).CustomerSegmentationDataset.Truycậptừ: https://www.kaggle.com/datasets

[2]. Apache Spark. (n.d.). Apache Spark Documentation. Truy cập từ: https://spark.apache.org/docs/latest/

[3]. Viện Công nghệ Thông tin. (2021). Phân tích và xử lý dữ liệu lớn với Apache Spark. Nhà xuất bản Thông tin và Truyền thông.

[4]. Lê Hoàng Nam. (2021). Phân tích dữ liệu và dự đoán hành vi khách hàng. Nhà xuất bản Thông tin và Truyền thông.

[5]. Trịnh Minh Hoàng. (2020). Big Data và ứng dụng trong kinh doanh. Nhà xuất bản Thống kê.