TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

TRƯƠNG BỈNH THUẬN - 52100322
NGUYỄN ĐÌNH DANH - 52100878
NGUYỄN THANH TÚ - 52100349
ĐẶNG VIẾT TRUNG - 52100342
TRẦN THỊ VẸN - 52100674

BÁO CÁO CUỐI KỲ
XỬ LÝ DỮ LIỆU LỚN

Người hướng dẫn
ThS. Nguyễn Thành An

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

TRƯƠNG BỈNH THUẬN - 52100322
NGUYỄN ĐÌNH DANH - 52100878
NGUYỄN THANH TÚ - 52100349
ĐẶNG VIẾT TRUNG - 52100342
TRẦN THỊ VẸN - 52100674

BÁO CÁO CUỐI KỲ
XỬ LÝ DỮ LIỆU LỚN

Người hướng dẫn
ThS. Nguyễn Thành An

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024

LỜI CẢM ƠN

Trong suốt quá trình học tập và rèn luyện, chúng em đã nhận được rất nhiều sự giúp đỡ tận tình, sự quan tâm, chăm sóc của thầy Nguyễn Thành An. Ngoài ra, chúng em còn được thầy truyền đạt những kiến thức thú vị, thầy còn giúp sinh viên có được nhiều niềm vui trong việc học và cảm thấy thoải mái, … Chúng em xin chân thành cảm ơn thầy rất nhiều trong suốt quá trình học tập này!

Bởi lượng kiến thức của chúng em còn hạn hẹp và gặp nhiều vấn đề trong quá trình học nên báo cáo này sẽ còn nhiều thiếu sót và cần được học hỏi thêm. Chúng em rất mong em sẽ nhận được sự góp ý của thầy về bài báo cáo này để chúng em rút kinh nghiệm trong những môn học sắp tới. Cuối cùng, chúng em xin chân thành cảm ơn thầy.

TP. Hồ Chí Minh, ngày 26 tháng 05 năm 2024

Tác giả

(Ký tên và ghi rõ họ tên)

Đặng Viết Trung Nguyễn Thanh Tú

Nguyễn Đình Danh Trần Thị Vẹn

Trương Bỉnh Thuận

CÔNG TRÌNH ĐƯỢC HOÀN THÀNH
TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Tôi xin cam đoan đây là sản phẩm của riêng tôi/chúng tôi và được sự hướng dẫn của thầy Nguyễn Thành An. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây.

Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình. Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 26 tháng 05 năm 2024

Tác giả

(Ký tên và ghi rõ họ tên)

Đặng Viết Trung Nguyễn Thanh Tú

Nguyễn Đình Danh Trần Thị Vẹn

Trương Bỉnh Thuận

TÓM TẮT

Câu 1: Phân cụm dữ liệu

Thực hiện:

- Sử dụng PySpark DataFrame để khai thác dữ liệu.
- Sử dụng thư viện pyspark.ml.clustering.KMeans để cài đặt thuật toán k-Means.
- Sử dụng Matplotlib để vẽ biểu đồ trực quan.
- Tính toán trung bình khoảng cách từ các điểm dữ liệu tới centroid cho từng cụm và vẽ biểu đồ cột thể hiện kết quả.

Kết quả:

- Biểu đồ cột cho thấy trung bình khoảng cách từ các điểm dữ liệu tới centroid của từng cụm, giúp hiểu rõ hơn về độ chặt chẽ của các cụm.

Câu 2: Giảm số chiều với SVD

Thực hiện:

- Sử dụng PySpark và thuật toán SVD để giảm số chiều.
- Chọn ngẫu nhiên 100 điểm dữ liệu sau khi giảm chiều.
- Sử dụng kết quả phân cụm từ câu 1 để vẽ biểu đồ 3D phân bố của các điểm này với Matplotlib.

Kết quả:

- Biểu đồ 3D trực quan thể hiện sự phân bố của 100 điểm dữ liệu trong không gian 3 chiều, giúp dễ dàng quan sát và hiểu rõ hơn về cấu trúc dữ liệu sau khi giảm chiều.

Câu 3: Khuyến nghị sản phẩm với Collaborative Filtering

Thực hiện:

- Chia tập dữ liệu thành tập train (70%) và test (30%).
- Sử dụng PySpark và thuật toán ALS để xây dựng mô hình.

- Đánh giá hiệu suất của mô hình theo độ đo MSE cho các giá trị số lượng người dùng "tương đồng".
- Vẽ biểu đồ cột thể hiện giá trị MSE.

Kết quả:

- Biểu đồ cột minh họa giá trị MSE cho từng giá trị số lượng người dùng "tương đồng", giúp đánh giá hiệu suất của mô hình và lựa chọn tham số phù hợp.

Câu 4: Dự đoán giá chứng khoán

Thực hiện:

- Tạo cột "fluctuation" chứa biên độ dao động của giá cổ phiếu.
- Tạo DataFrame với 2 cột: Biên độ 5 ngày trước và Biên độ ngày tiếp theo.
- Xây dựng mô hình Linear Regression với PySpark.
- Tính MSE trên tập train và test.
- Vẽ biểu đồ cột thể hiện giá trị MSE.

Kết quả:

- Biểu đồ cột thể hiện giá trị MSE trên tập training và test, giúp đánh giá độ chính xác của mô hình dự đoán giá chứng khoán.

Câu 5: Phân loại đa lớp với PySpark

Thực hiện:

- Sử dụng các thuật toán phân loại: Multi-layer Perceptron, Random Forest, và Linear SVM.
- Đánh giá độ chính xác (Accuracy) của các mô hình trên tập train và test.
- Vẽ biểu đồ cột đôi thể hiện độ chính xác của ba mô hình.

Kết quả:

- Biểu đồ cột đôi so sánh độ chính xác của ba mô hình trên tập training và test, giúp lựa chọn mô hình phân loại tốt nhất.

Tổng Kết

Báo cáo đã thực hiện phân tích và xử lý dữ liệu lớn thông qua các bài toán thực tế, từ phân cụm dữ liệu, giảm chiều, khuyến nghị sản phẩm, dự đoán giá chứng khoán đến phân loại đa lớp. Sử dụng PySpark và Matplotlib để khai thác, trực quan hóa và đánh giá các mô hình, báo cáo cung cấp cái nhìn tổng quan về hiệu suất và khả năng ứng dụng của từng phương pháp trong xử lý dữ liệu lớn.

MỤC LỤC

DANH MỤC HÌNH VẼ

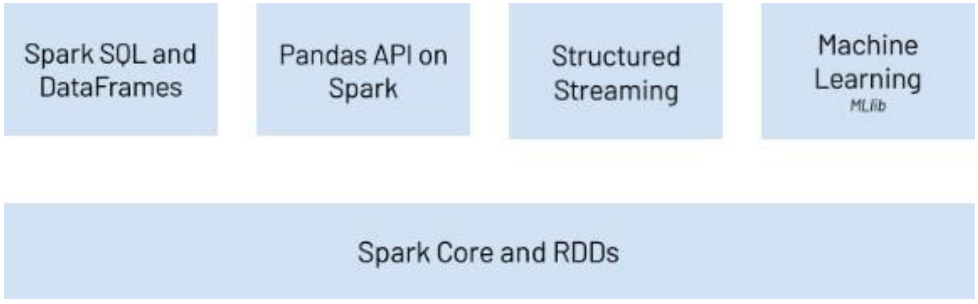
CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

1.1 PySpark

PySpark là API Python cho Apache Spark. Nó cho phép bạn thực hiện thời gian thực, xử lý dữ liệu quy mô lớn trong môi trường phân tán bằng Python. Nó cũng cung cấp một PySpark Shell để phân tích tương tác dữ liệu của bạn.

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

PySpark hỗ trợ tất cả các tính năng của Spark như Spark SQL, DataFrames, Structured Streaming, Machine Learning (MLlib) và Spark Core.



Hình 1.1: Tính năng của Spark

1.1.1 PySpark – RDD

RDD (Resilient Distributed Dataset) là một cấu trúc dữ liệu cơ bản trong thư viện PySpark, được thiết kế để xử lý dữ liệu trên các cụm dữ liệu phân tán (distributed data clusters). Dưới đây là một số thông tin chi tiết về RDD trong thư viện PySpark:

Khái niệm cơ bản: RDD là một tập hợp các phần tử không thay đổi, được phân tán trên nhiều nút trong cụm dữ liệu. RDD có thể được tạo ra từ các tập dữ liệu có sẵn trong bộ nhớ hoặc từ các phần tử trong hệ thống tệp.

Bền vững (Resilient): Tính bền vững của RDD đề cập đến khả năng tái tạo dữ liệu trong trường hợp một phần của cụm dữ liệu bị mất do lỗi. RDD có thể tái tạo dữ liệu mất bằng cách sử dụng thông tin gốc và các phép biến đổi.

Phân tán (Distributed): RDD được phân tán trên nhiều nút trong cụm dữ liệu, cho phép xử lý song song trên dữ liệu lớn.

Khả năng biến đổi (Transformations): RDD hỗ trợ các phép biến đổi, như map(), filter(), flatMap(), reduceByKey(),... để thực hiện các thao tác xử lý dữ liệu trên toàn bộ RDD hoặc một phần của RDD.

Khả năng hành động (Actions): RDD hỗ trợ các phép hành động như collect(), count(), reduce(), saveAsTextFile(),... để tính toán và trả về kết quả từ dữ liệu.

Immutability: RDD là không thay đổi (immutable), có nghĩa là một khi đã tạo, bạn không thể thay đổi nó. Thay vào đó, bạn có thể tạo một RDD mới từ RDD hiện có thông qua các phép biến đổi.

Lười biến đổi (Lazy evaluation): PySpark sử dụng lười biến đổi, nghĩa là các phép biến đổi trên RDD không được thực hiện ngay lập tức khi gọi, mà chỉ khi một phép hành động được gọi. Điều này giúp tối ưu hóa hiệu suất bằng cách chỉ tính toán khi cần thiết.

Caching: RDD có khả năng lưu trữ cache, giúp tăng tốc độ tính toán bằng cách lưu trữ dữ liệu trung gian trong bộ nhớ hoặc đĩa.

1.1.2 Dataframe

Trong Apache Spark, DataFrame là một cấu trúc dữ liệu phân tán giống như một bảng trong SQL hoặc một bảng Excel, có thể được sử dụng để xử lý và phân tích dữ liệu phân tán. DataFrame cung cấp một giao diện lập trình cao cấp hơn so với RDD, làm cho việc xử lý và truy vấn dữ liệu dễ dàng hơn. Dưới đây là một số khái niệm và tính năng của DataFrame trong PySpark:

- Cấu trúc dữ liệu phân tán: DataFrame là một tập hợp các dữ liệu phân tán, được tổ chức thành các cột có tên.

- Khả năng xử lý dữ liệu có cấu trúc: DataFrame hỗ trợ xử lý dữ liệu có cấu trúc, giúp cho việc làm việc với dữ liệu dạng bảng trở nên dễ dàng.
- Tích hợp với các ngôn ngữ khác nhau: DataFrame có thể được sử dụng trong các ngôn ngữ lập trình khác nhau như Python, Scala hoặc Java
- Tính tương tự với SQL: DataFrame cung cấp các phương thức cho phép thực hiện các thao tác dữ liệu tương tự như trong SQL như lọc (filter), nhóm (group by), kết hợp (join),...

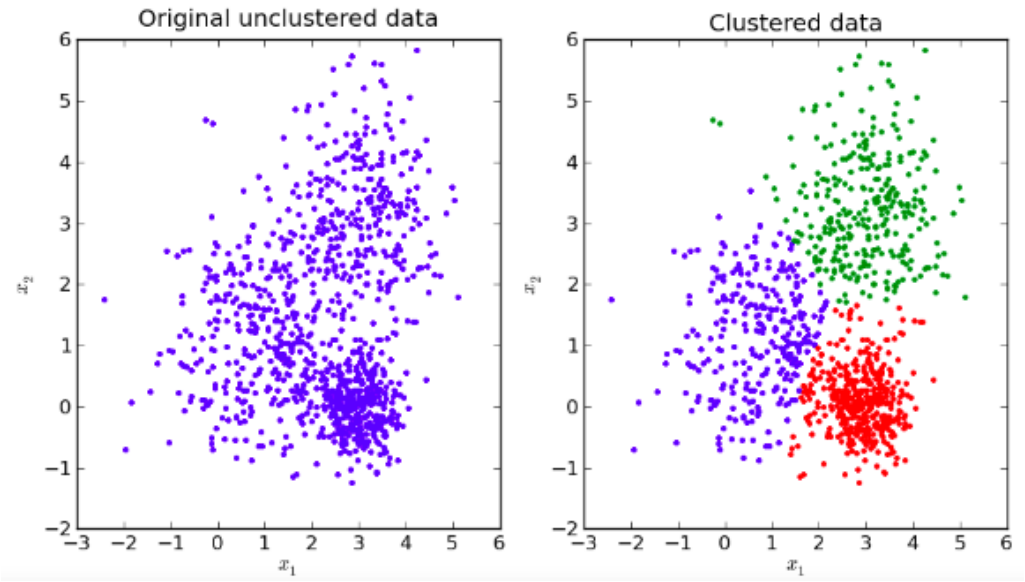
Hỗ trợ tối ưu hóa hiệu suất: DataFrame có khả năng tối ưu hóa hiệu suất trong các tác vụ xử lý dữ liệu phức tạp thông qua Catalyst Optimizer.

Hỗ trợ đa nguồn dữ liệu: DataFrame có thể đọc dữ liệu từ nhiều nguồn khác nhau như HDFS, S3, Hive, JSON, CSV, Parquet, JDBC, và nhiều nguồn dữ liệu khác.

1.2 Phân Cụm Dữ Liệu (Clustering)

1.2.1 Khái niệm

Phân cụm dữ liệu là bài toán gom nhóm các đối tượng dữ liệu vào thánh từng cụm (cluster) sao cho các đối tượng trong cùng một cụm có sự tương đồng theo một tiêu chí nào đó.



Hình 1.2: Minh họa phân cụm

1.2.2 Đặc điểm

Số cụm dữ liệu không được biết trước

Có nhiều các tiếp cận, mối cách lại có vài kỹ thuật

Các kỹ thuật khác nhau thường mang lại kết quả khác nhau.

1.2.3 Các độ đo khoảng cách

Tính chất của độ đo khoảng cách:

- Tính không âm (non-negative): d(x,y) >=0 và d(x, y) = 0 khai và chỉ khi x trùng y.
- Tính đối xứng (symmetic): d(x, y) = d(y, x)
- Tính tam giác (traingle inequality): d(x, y) + d(y, z) >= d(x, z)

1.2.3.1 Độ do Euclid chuẩn và độ đo Manhattan

- Cho hai điểm x = (x1, x2, ..., xm) và y = (y1, y2, ..., ym)
- Độ đo Euclid được xác định theo công thức

$$d(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{m} |x_i - y_i|^r \right)^{\frac{1}{r}}$$

- Độ đo Euclid chuẩn (r = 2)

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{m} (x_i - y_i)^2}$$

- Độ đo Manhattan

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{m} |x_i - y_i|$$

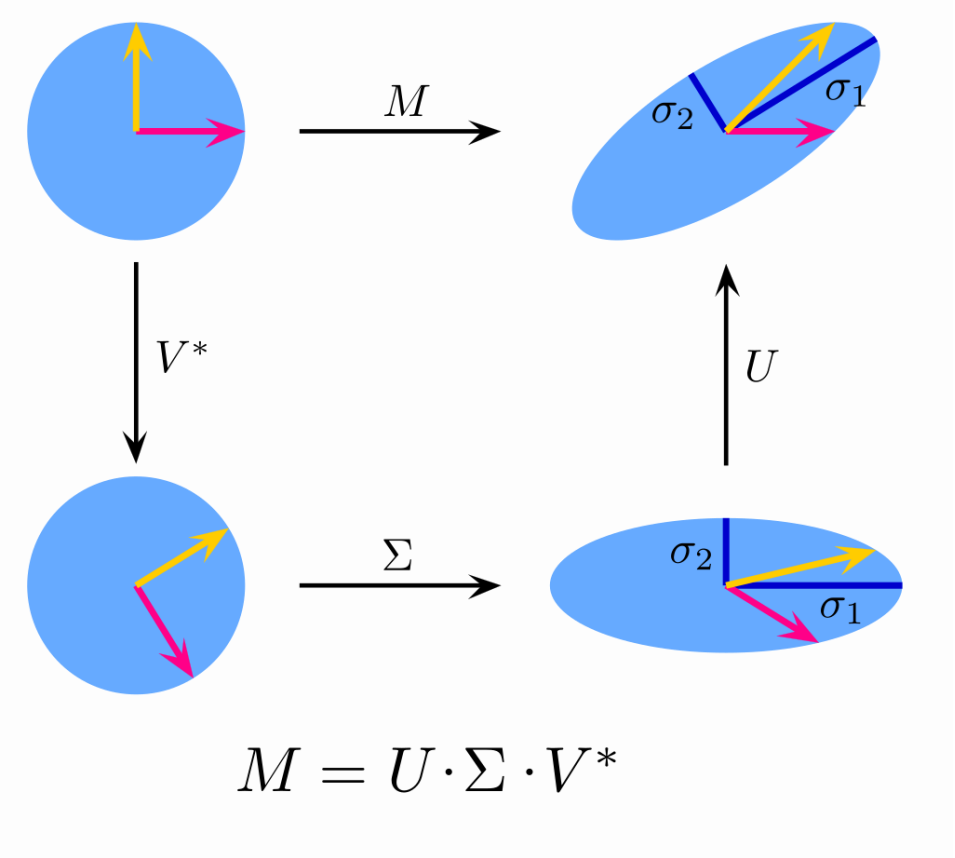
1.3 Singular Value Decomposition

Singular Value Decomposition (SVD) là một kỹ thuật phân rã ma trận rất quan trọng và phổ biến trong nhiều lĩnh vực, bao gồm xử lý dữ liệu, học máy, và nén dữ liệu. SVD phân rã một ma trận thành ba ma trận khác, giúp giảm số chiều và làm nổi bật các đặc điểm quan trọng trong dữ liệu.

1.3.1 Khái Niệm Cơ Bản

SVD là một phương pháp phân rã một ma trận A có kích thước m×n thành ba ma trận: A=UΣVT Trong đó:

- U là ma trận trực giao kích thước m×m, chứa các vector riêng (eigenvectors) của AA^T
- Σ là ma trận đường chéo kích thước m×n, chứa các giá trị riêng (singular values) của A. Các giá trị này sắp xếp theo thứ tự giảm dần.
- V là ma trận trực giao kích thước n×n, chứa các vector riêng của A^TA.



1.3.2 Ứng Dụng của SVD

Giảm Số Chiều (Dimensionality Reduction): SVD giúp giảm số chiều của dữ liệu bằng cách chọn k giá trị riêng lớn nhất và tương ứng với các vector riêng, giảm nhiễu và tăng hiệu quả tính toán.

Nén Dữ Liệu (Data Compression): Giảm kích thước lưu trữ và tăng tốc độ xử lý.

Hệ Thống Khuyến Nghị (Recommendation Systems): SVD được sử dụng trong việc giảm số chiều của ma trận người dùng-sản phẩm, giúp hệ thống khuyến nghị hoạt động hiệu quả hơn.

Xử Lý Hình Ảnh (Image Processing): SVD giúp giảm nhiễu và nén hình ảnh.

1.4 Collaborative Filtering

Collaborative Filtering (lọc cộng tác) là một kỹ thuật được sử dụng rộng rãi trong hệ thống khuyến nghị, nhằm đưa ra các gợi ý về sản phẩm hoặc dịch vụ dựa trên sở thích và hành vi của người dùng. Kỹ thuật này hoạt động bằng cách tìm ra các mẫu tương tự trong dữ liệu người dùng hoặc sản phẩm và sử dụng các mẫu này để dự đoán sở thích của người dùng.

1.4.1 Nguyên Lý Cơ Bản

Có hai phương pháp chính trong Collaborative Filtering:

User-Based Collaborative Filtering: Đề xuất các mục cho một người dùng dựa trên sự tương đồng giữa người dùng đó và những người dùng khác.

Item-Based Collaborative Filtering: Đề xuất các mục cho một người dùng dựa trên sự tương đồng giữa các mục mà người dùng đó đã thích hoặc sử dụng.

1.4.2 Thuật Toán ALS (Alternating Least Squares)

Alternating Least Squares (ALS) là một trong những thuật toán phổ biến nhất để thực hiện Collaborative Filtering. ALS hoạt động bằng cách phân rã ma trận người dùng-mục thành hai ma trận nhỏ hơn: ma trận người dùng và ma trận mục. Quá trình này lặp đi lặp lại để tối ưu hóa cả hai ma trận.

Các bước thực hiện ALS:

- Khởi tạo ma trận người dùng và ma trận mục với các giá trị ngẫu nhiên.
- Cập nhật ma trận người dùng bằng cách cố định ma trận mục và giải bài toán tối ưu hóa least squares.
- Cập nhật ma trận mục bằng cách cố định ma trận người dùng và giải bài toán tối ưu hóa least squares.
- Lặp lại bước 2 và 3 cho đến khi hội tụ hoặc đạt số lần lặp tối đa.

1.4.3 Độ Đo Mean Squared Error (MSE)

Mean Squared Error (MSE) là một độ đo thường được sử dụng để đánh giá hiệu suất của mô hình lọc cộng tác. MSE tính trung bình bình phương sai số giữa giá trị dự đoán và giá trị thực tế.

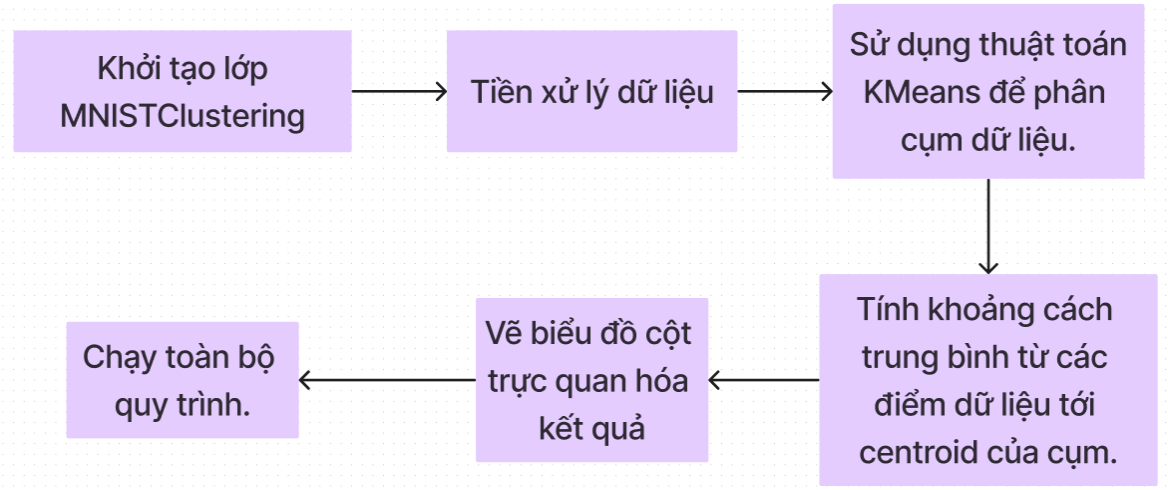
$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Trong đó:

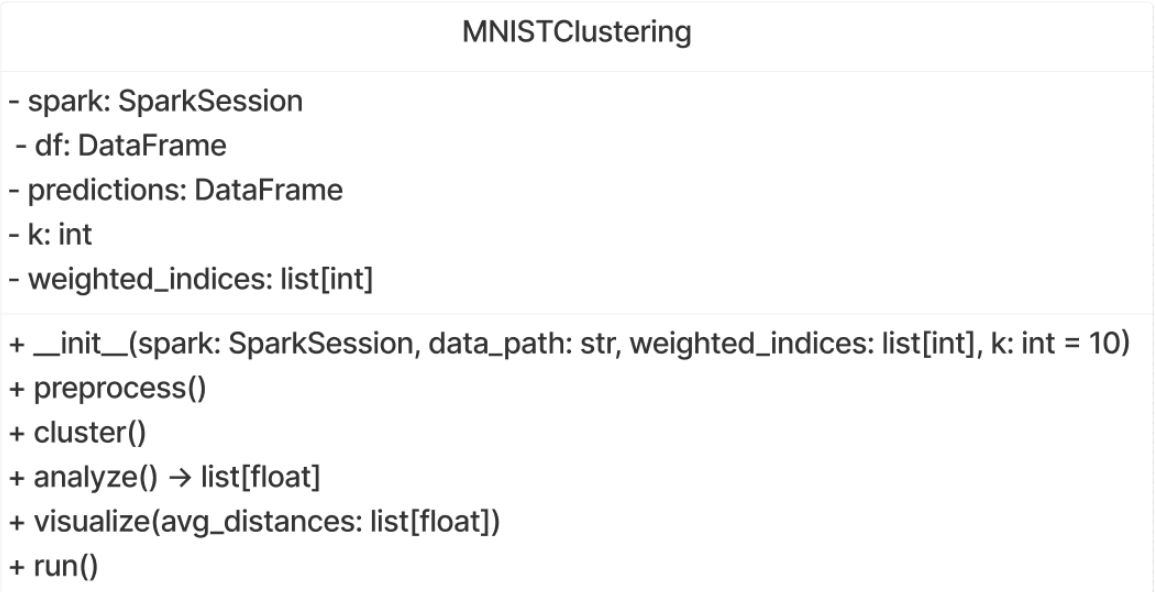
- n là số lượng dự đoán.
- y_i là giá trị thực tế.
- y^i là giá trị dự đoán.

CHƯƠNG 2. ÁP DỤNG

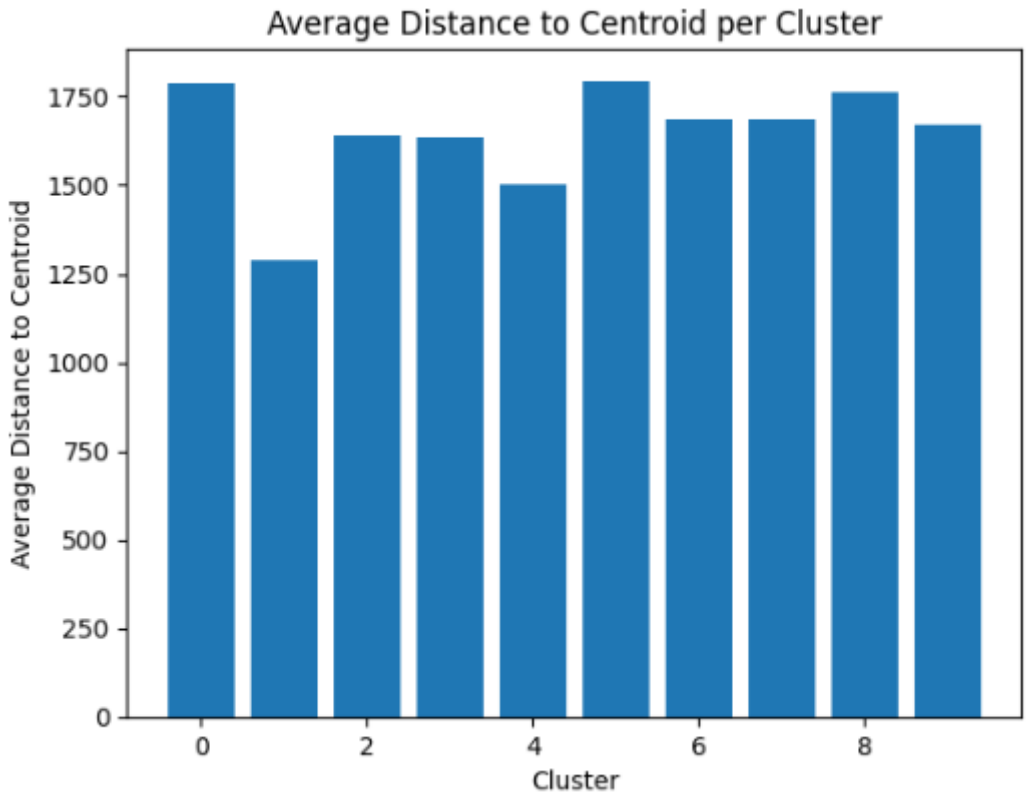
2.1 Câu 1



Hình 2.1: Sơ đồ thực hiện phân cụm dữ liệu

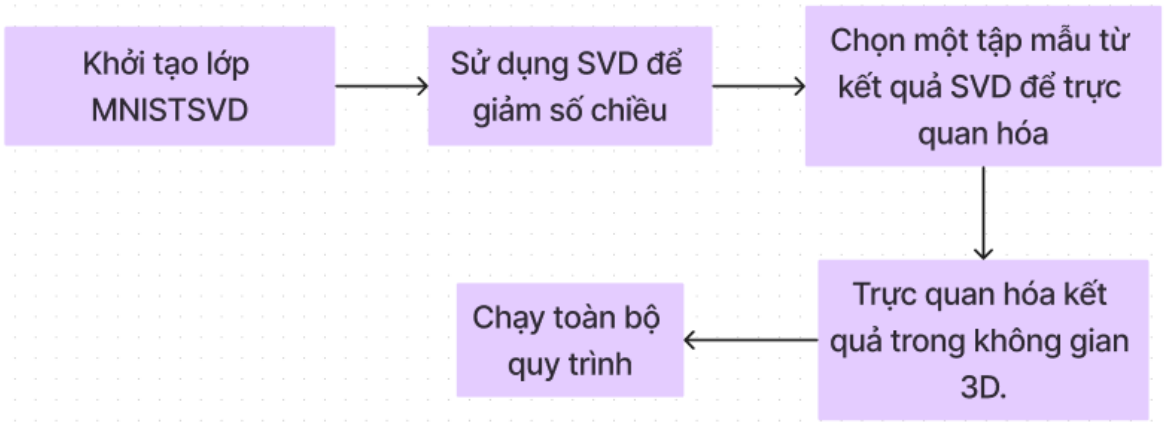


Hình 2.2: Sơ đồ lớp phân cụm



Hình 2.3: Kết quả phân cụm dữ liệu

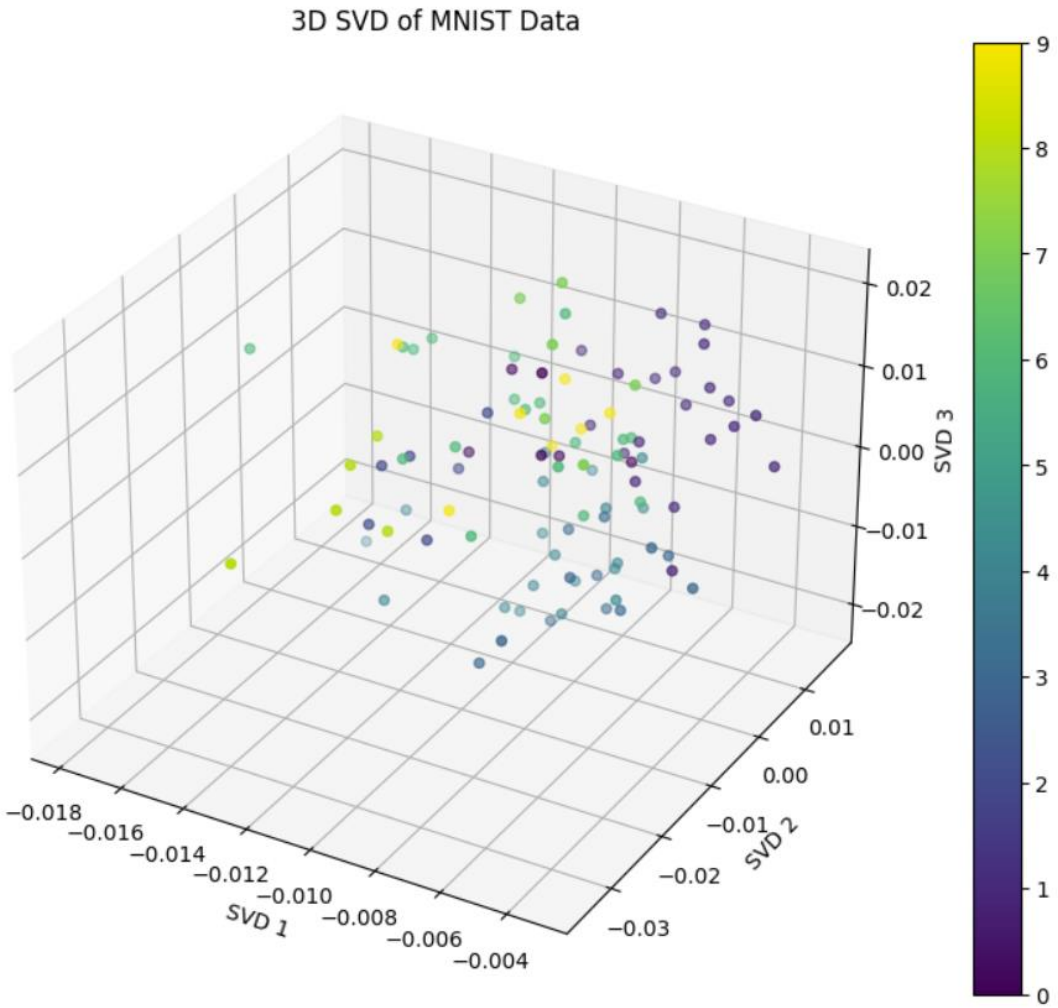
2.2 Câu 2



Hình 2.4: Sơ đồ thực hiện giảm số chiều với SVD

MNISTSVD
- spark: SparkSession - predictions: DataFrame - k: int - n: int
+ __init__(spark: SparkSession, predictions: DataFrame, k: int = 3, n: int = 100) + reduce_dimensions() → DataFrame + sample_data(svd_result: DataFrame) → Tuple + plot_3d(svd_features_np: np.array, sampled_predictions: np.array) + run()

Hình 2.5: Sơ đồ lớp giảm chiều dữ liệu

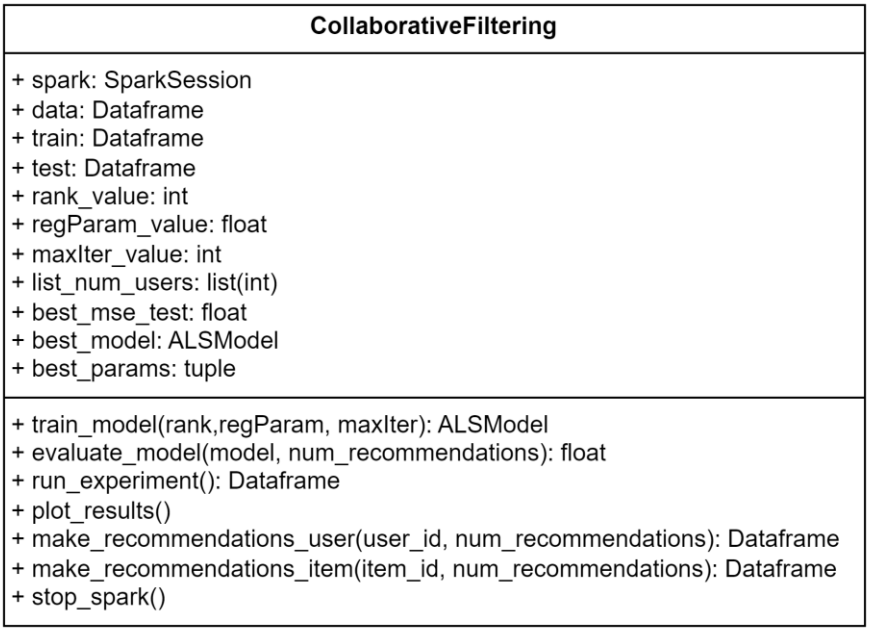


Hình 2.6: Kết quả giảm số chiều

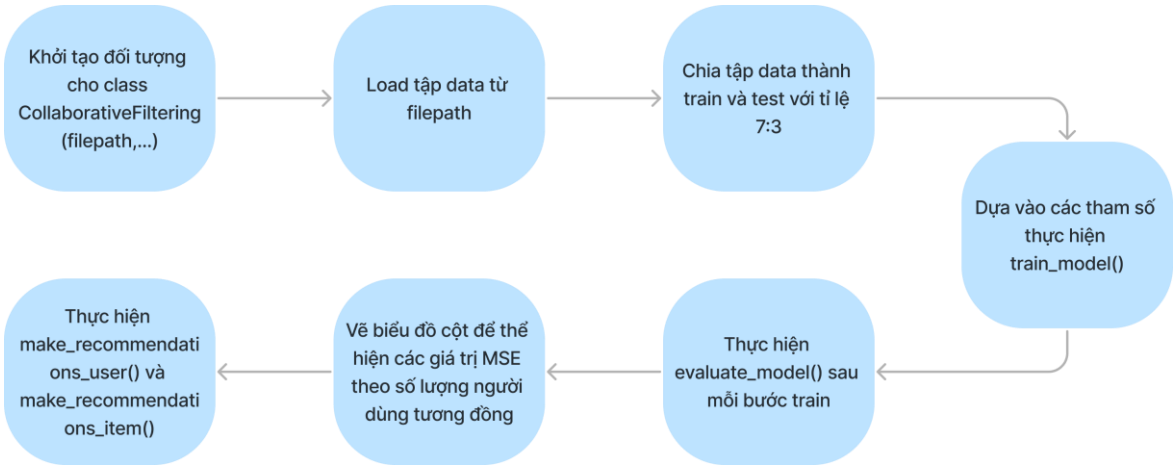
2.3 Câu 3

Bài toán khuyến nghị sản phẩm với Collaborative Filtering cho tập ratings2k.csv dựa trên thuật toán ALS để khảo sát hiệu suất của mô hình theo độ đo Mean Squared Error (MSE) với các giá trị số lượng người dùng "tương đồng" trong đoạn [10; 20].

Sơ đồ lớp:



Hình 2.7: Sơ đồ lớp CollaborativeFiltering.



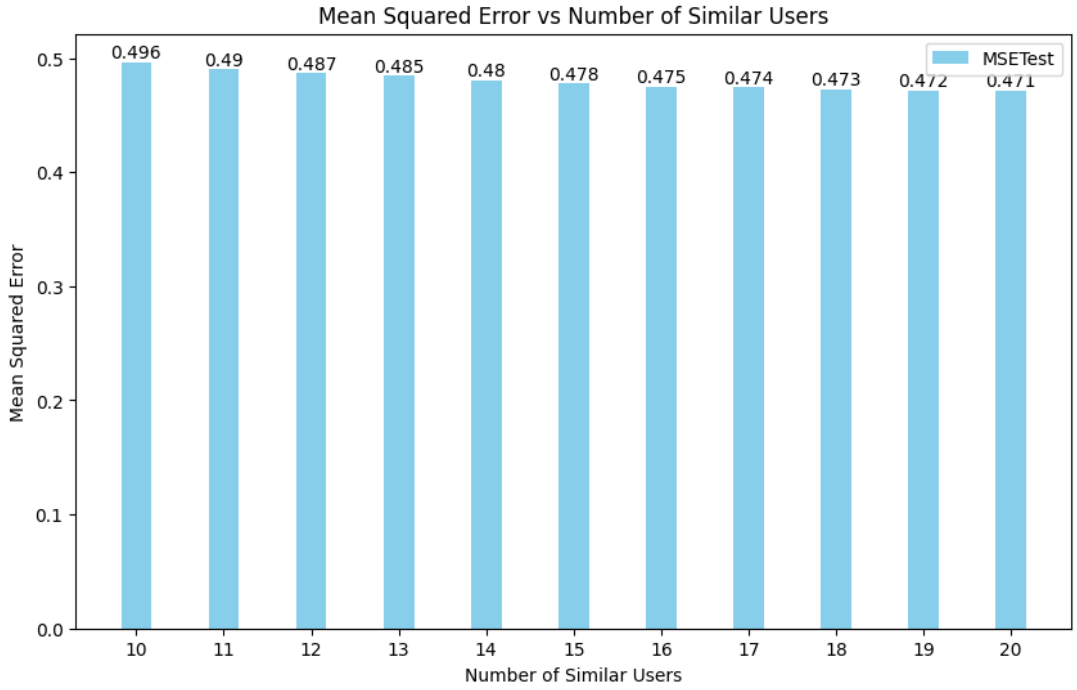
Hình 2.8: Sơ đồ hoạt động tổng quát.

Output của hàm run_experiment() là giá trị của các tham số và MSE tương ứng:

```
Rank: 10, RegParam: 0.1, MaxIter: 10, NumUsers: 10, MSETest: 0.49636994522557415
Rank: 10, RegParam: 0.1, MaxIter: 10, NumUsers: 11, MSETest: 0.49033658573306454
Rank: 10, RegParam: 0.1, MaxIter: 10, NumUsers: 12, MSETest: 0.48653170522134875
Rank: 10, RegParam: 0.1, MaxIter: 10, NumUsers: 13, MSETest: 0.4846540727902628
Rank: 10, RegParam: 0.1, MaxIter: 10, NumUsers: 14, MSETest: 0.4804252765011511
Rank: 10, RegParam: 0.1, MaxIter: 10, NumUsers: 15, MSETest: 0.477521864309743
Rank: 10, RegParam: 0.1, MaxIter: 10, NumUsers: 16, MSETest: 0.4751585465192135
Rank: 10, RegParam: 0.1, MaxIter: 10, NumUsers: 17, MSETest: 0.47418431349033796
Rank: 10, RegParam: 0.1, MaxIter: 10, NumUsers: 18, MSETest: 0.47282209427268984
Rank: 10, RegParam: 0.1, MaxIter: 10, NumUsers: 19, MSETest: 0.4716137384081107
Rank: 10, RegParam: 0.1, MaxIter: 10, NumUsers: 20, MSETest: 0.4714880484016054
```

Hình 2.9: Kết quả của run_experiment() với NumUser=[10,20].

Output của hàm plot_results() là biểu đồ thể hiện các MSE vừa tính được:



Hình 2.10: Kết quả của hàm plot_results().

Output của hàm make_recommendations_user() là Dataframe của 1 UserID thể hiện các dự đoán rating của user đó với các item với số item dự đoán bằng với N số lượng người dùng "tương tự":

user	item	rating
55	352	4.43102
55	335	3.9418976
55	199	3.8839781
55	413	3.7169306
55	422	3.65177
55	196	3.6078062
55	123	3.6049364
55	36	3.5598989
55	8	3.5472276
55	200	3.509641
55	438	3.480188
55	95	3.4742172
55	251	3.4534976
55	257	3.4314704
55	163	3.4163554
55	144	3.402484
55	433	3.3593829
55	176	3.3400552
55	80	3.3173904
55	437	3.3109953

Hình 2.11: Kết quả của hàm make_recommendations_user().

Output của hàm make_recommendations_item() là tương tự với make_recommendations_user() nhưng khác biệt ở điểm là dựa theo item và dự đoán user:

item	user	rating
352	35	5.3315716
352	13	5.135572
352	41	4.994187
352	69	4.9313583
352	1	4.8797145
352	71	4.8002105
352	28	4.6533933
352	14	4.582856
352	5	4.534957
352	20	4.4505987
352	23	4.4416738
352	73	4.434429
352	55	4.43102
352	16	4.4310017
352	10	4.418131
352	40	4.40754
352	42	4.367618
352	37	4.3671885
352	50	4.337681
352	57	4.3349557

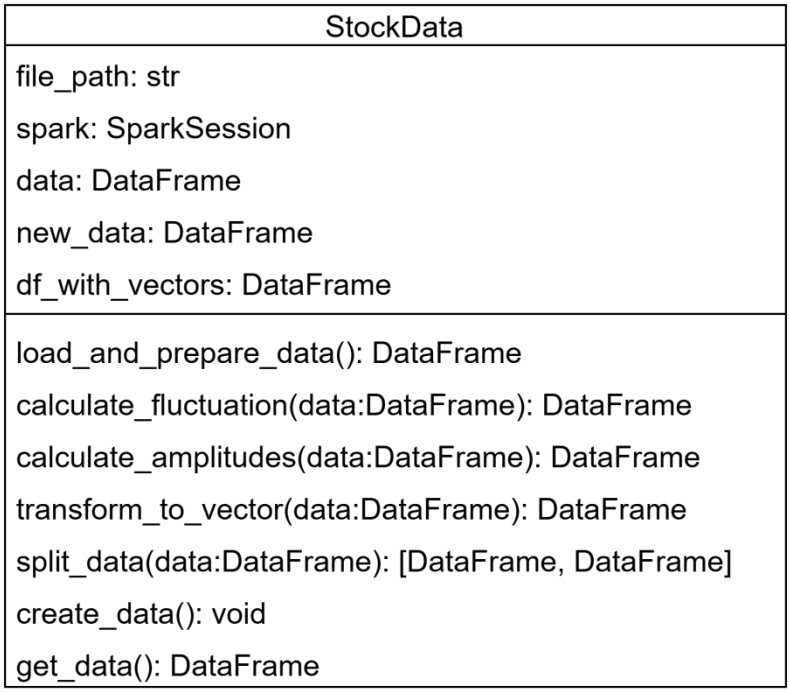
Hình 2.12: Kết quả của hàm make_recommendations_item().

2.4 Câu 4

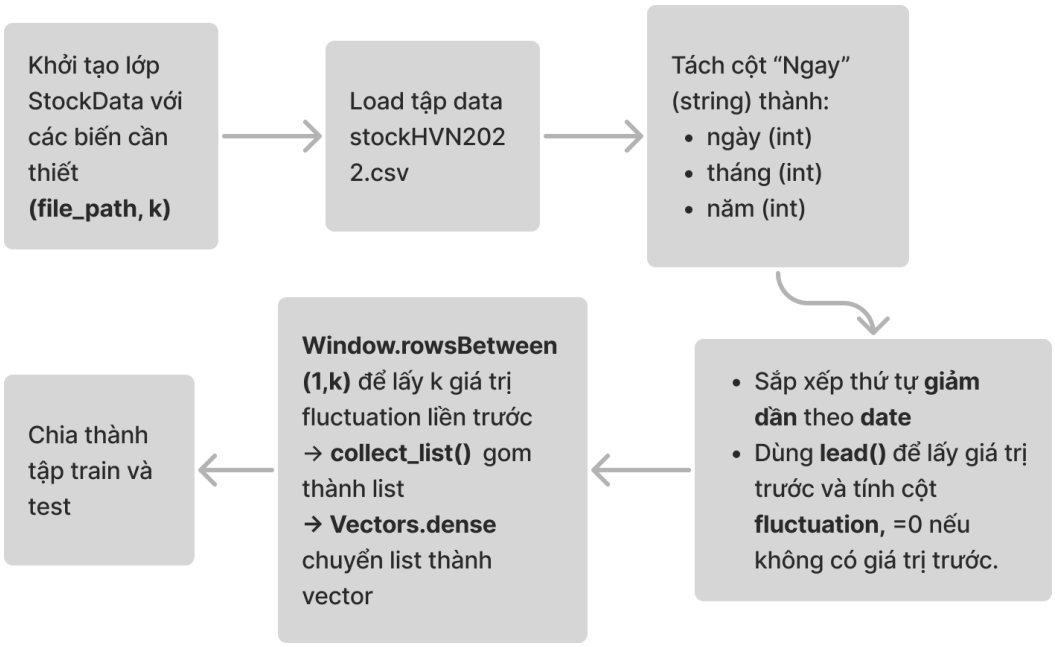
Bài toán xây dựng model Linear Regression để dự đoán giá chứng khoán theo mô hình hướng đối tượng bao gồm 2 class:

2.4.1 StockData: lớp xử lí dữ liệu.

Sơ đồ lớp:



Hình 2.13: Sơ đồ lớp StockData.



Hình 2.14: Quy trình xây dựng StockData class.

Output cuối từ StockData class là 2 tập train và test với 2 cột:

- last_k_fluctuation: Vector số dao động của k ngày liền trước.

- fluctuation: số dao động của ngày hiện tại.

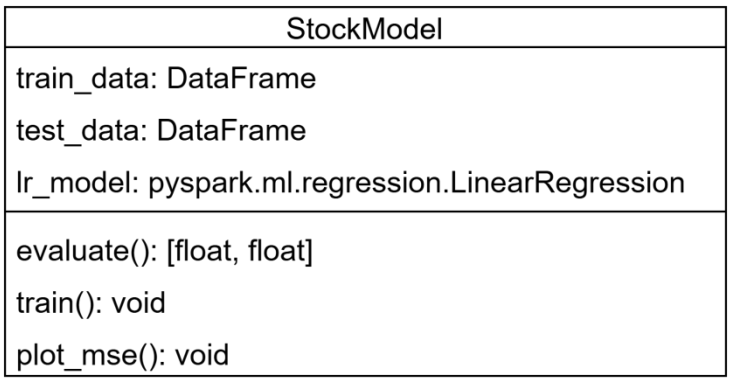
```
Train data:
+-------------------------------------------------------------+--------------------+
|last_k_fluctuation                                           |fluctuation         |
+-------------------------------------------------------------+--------------------+
|[0.06885245901639349,0.012269938650306704,-0.02727272727272723]  |-0.034267912772585715|
|[-0.003267973856209197,0.06885245901639349,0.012269938650306704] |-0.02727272727272723 |
|[-0.003257328990227944,-0.003267973856209197,0.06885245901639349]|0.012269938650306704 |
|[0.02675585284280939,-0.003257328990227944,-0.003267973856209197]|0.06885245901639349  |
|[0.023972602739726005,0.02675585284280939,-0.003257328990227944] |-0.003267973856209197|
+-------------------------------------------------------------+--------------------+
only showing top 5 rows

Test data:
+-------------------------------------------------------------+--------------------+
|last_k_fluctuation                                           |fluctuation         |
+-------------------------------------------------------------+--------------------+
|[-0.06918918918918925,0.06968641114982595,0.02280130293159599]  |-0.01273885350318463|
|[-0.06565656565656569,-0.06918918918918925,0.06968641114982595] |0.02280130293159599 |
|[0.014344262295082027,-0.06565656565656569,-0.06918918918918925]|0.06968641114982595 |
|[-0.06153846153846159,0.014344262295082027,-0.06565656565656569]|-0.06918918918918925|
|[0.014634146341463448,-0.06153846153846159,0.014344262295082027]|-0.06565656565656569|
+-------------------------------------------------------------+--------------------+
only showing top 5 rows
```

Hình 2.15: Train và test data với k=3

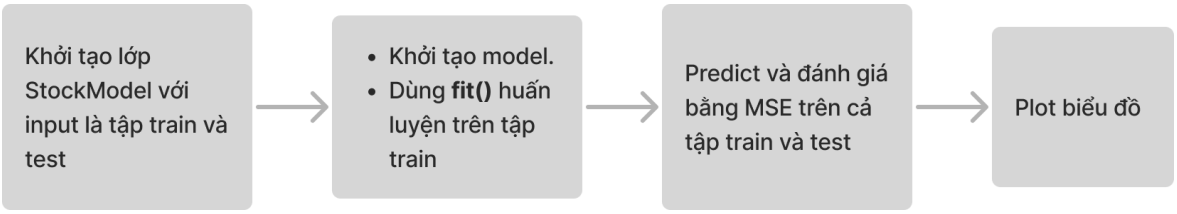
2.4.2 StockModel - lớp model Linear Regression cho StockData

Sơ đồ lớp:



Hình 2.16: Sơ đồ lớp StockModel

Quy trình xây dựng và đánh giá model:

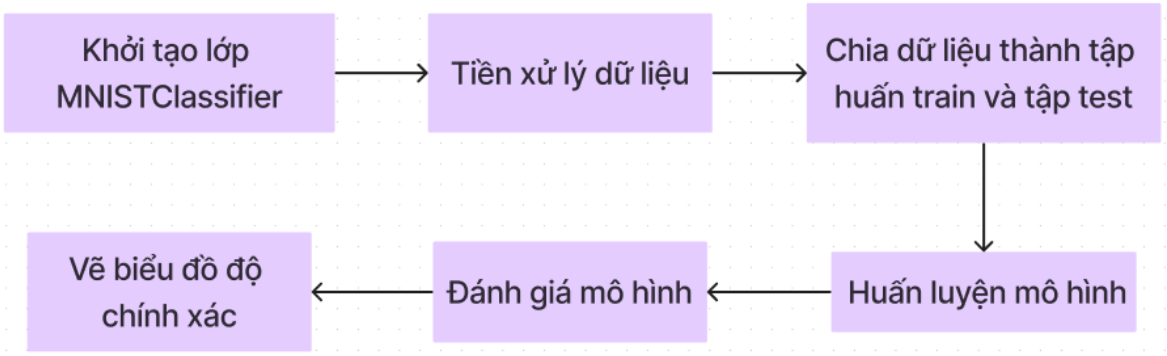


Hình 2.17: Quy trình xây dựng và đánh giá model.

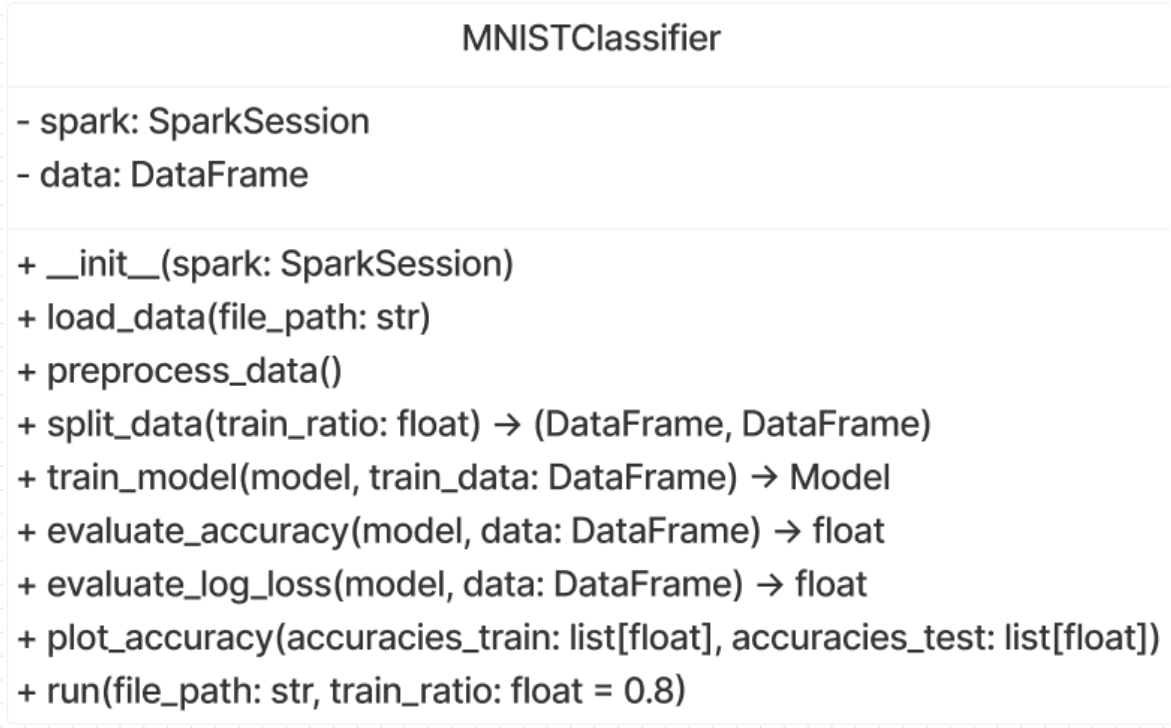
```
Train MSE: 0.0005788157807408016
Test MSE: 0.0006217807601481779
```

Hình 2.18: Điểm số MSE sau khi predict trên tập train và test

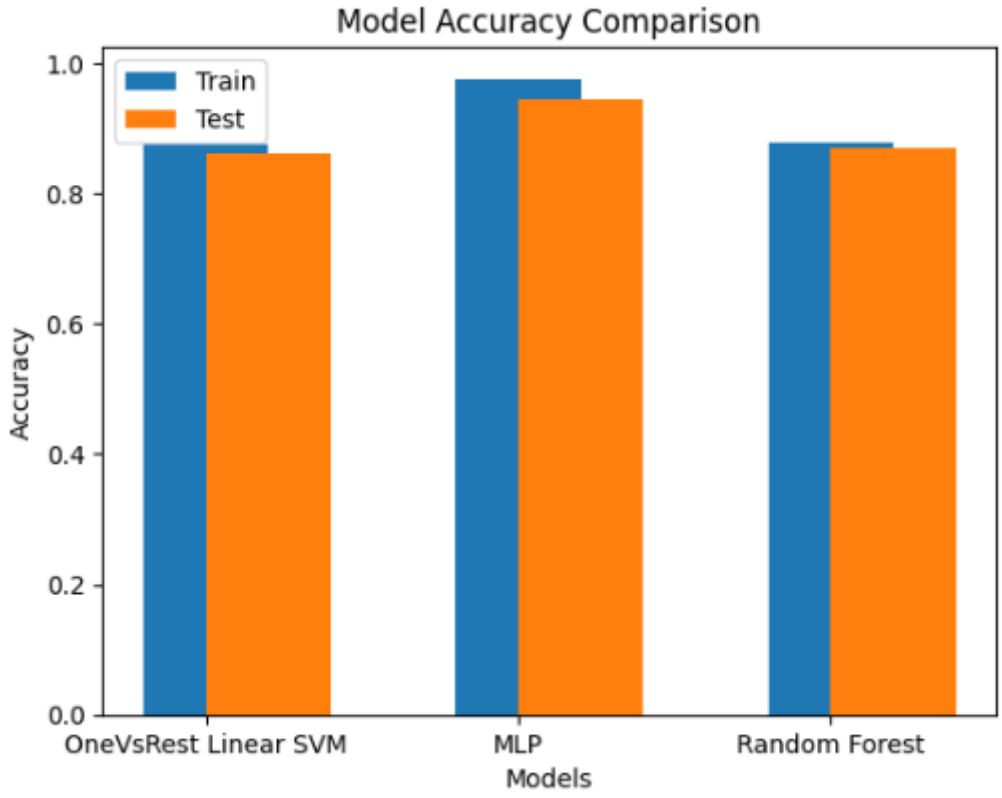
2.5 Câu 5



Hình 2.19: Sơ đồ thực hiện phân loại đa lớp



Hình 2.20: Sơ đồ lớp phan loại đa lớp



Hình 2.21: Kết quả độ chính xác của 3 mô hình

CHƯƠNG 3. TỔNG KẾT

3.1 Danh sách thành viên

HỌ VÀ TÊN	MSSV	EMAIL
Đặng Viết Trung	52100342	52100342@student.tdtu.edu.vn
Nguyễn Thanh Tú	52100349	52100349@student.tdtu.edu.vn
Trần Thị Vẹn	52100674	52100674@student.tdtu.edu.vn
Nguyễn Đình Danh	52100878	52100878@student.tdtu.edu.vn
Trương Bỉnh Thuận	52100322	52100322@student.tdtu.edu.vn

3.2 Bảng tự đánh giá

Thành viên	Yêu cầu	Mức độ hoàn thành
Nguyễn Thanh Tú	Câu 1	100%
Trần Thị Vẹn	Câu 2	100%
Trương Bỉnh Thuận	Câu 3	100%

Nguyễn Đình Danh	Câu 4	100%
Đặng Viết Trung	Câu 5	100%
Tất cả thành viên	Câu 6	100%

3.3 Thuận lợi và khó khăn

3.3.1 Thuận lợi

Khả năng xử lý dữ liệu lớn của PySpark: PySpark cung cấp khả năng xử lý dữ liệu lớn hiệu quả nhờ tính phân tán và khả năng làm việc với các tập dữ liệu lớn trên nhiều máy tính. Điều này giúp giảm thời gian xử lý và tăng khả năng mở rộng.

Thư viện phong phú và mạnh mẽ: Các thư viện như pyspark.ml, matplotlib cung cấp nhiều công cụ mạnh mẽ để phân tích và trực quan hóa dữ liệu, từ đó giúp sinh viên dễ dàng triển khai các thuật toán và biểu diễn kết quả một cách trực quan.

3.3.2 Khó khăn

Tính toán hiệu suất cao: Các bài toán như phân cụm dữ liệu với trọng số lớn hoặc giảm số chiều với SVD đòi hỏi khả năng tính toán cao. Nếu không có môi trường tính toán đủ mạnh, quá trình xử lý có thể rất chậm và không hiệu quả.

Khó khăn trong cài đặt thuật toán: Mặc dù PySpark cung cấp nhiều công cụ mạnh mẽ, việc cài đặt và tối ưu hóa các thuật toán như k-Means, ALS, hay Linear Regression vẫn đòi hỏi kiến thức sâu về cả lý thuyết và thực hành.

TÀI LIỆU THAM KHẢO

Tiếng Anh

[1] Viblo [Trực tuyến]. Địa chỉ: Hierarchical clustering - Phân cụm dữ liệu (viblo.asia). Ngày truy cập: [25/5/2024]

[2] Wikipedia [Trực tuyến]. Địa chỉ: Singular value decomposition - Wikipedia. Ngày truy cập: [25/5/2024]

[3] Wikipedia [Trực tuyến]. Địa chỉ: Collaborative filtering - Wikipedia. Ngày truy cập: [25/5/2024]