ĐẠI HỌC BÁCH KHOA HÀ NỘI

KHOA TOÁN - TIN

----- ☙ 🕮 ❧ -----

ĐỒ ÁN I

BIG DATA FOR QUANTITATIVE TRADING

Giảng viên hướng dẫn: TS. Đào Thành Chung

Chữ ký của GVHD

Sinh viên thực hiện: Vũ Danh Trung Hiếu

MSSV: 20216925

Lớp: Hệ thống thông tin quản lý 01 – K66

HÀ NỘI, 06/2024

ĐẠI HỌC BÁCH KHOA HÀ NỘI

KHOA TOÁN - TIN

----- ☙ 🕮 ❧ -----

ĐỒ ÁN I

Big Data for Quantitative Trading

Vũ Danh Trung Hiếu

hieu.vdt216925@sis.hust.edu.vn

Chuyên ngành: Hệ thống thông tin quản lý

Giảng viên hướng dẫn: TS. Đào Thành Chung

Khoa: Toán - Tin

HÀ NỘI, 06/2024

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

1. Mục tiêu và nội dung đồ án
   a) Mục tiêu: . . . . . .
   . . . . . .
   . . . . . .
   . . . . . .
   b) Nội dung: . . . . . .
   . . . . . .
   . . . . . .
   . . . . . .
2. Kết quả đạt được: . . . . . .
   . . . . . .
   . . . . . .
   . . . . . .
   . . . . . .
3. Ý thức làm việc của sinh viên: . . . . . .
   . . . . . .
   . . . . . .
   . . . . . .
   . . . . . .

Hà Nội, ngày.....tháng.....năm 2024

Giảng viên hướng dẫn

TS. Đào Thành Chung

LỜI CẢM ƠN

Lời đầu tiên, em xin gửi lời cảm ơn chân thành nhất đến thầy Đào Thành Chung, người đã tận tình hướng dẫn, chỉ bảo và giúp đỡ em trong suốt quá trình thực hiện đồ án này. Sự nhiệt tình, kiến thức sâu rộng và những góp ý quý báu của thầy đã giúp em hoàn thiện đề tài này một cách tốt nhất.

Em cũng xin gửi lời cảm ơn chân thành đến các thầy, cô trong khoa Toán - Tin, Đại học Bách Khoa Hà Nội đã truyền đạt kiến thức và hỗ trợ em trong suốt thời gian học tập và thực hiện đồ án. Những kiến thức và kỹ năng mà em đã được học từ thầy cô là nền tảng vững chắc để em hoàn thành đồ án này và chúng sẽ là hành trang trên con đường sự nghiệp tương lai của em.

Bên cạnh đó em xin gửi lời cảm ơn đến những người bạn đã luôn bên cạnh, động viên và chia sẻ những khó khăn. Sự giúp đỡ và ủng hộ của các bạn là nguồn động lực to lớn để em vượt qua những khó khăn thử thách và hoàn thành đồ án này.

Cuối cùng em xin gửi lời cảm ơn sâu sắc đến gia đình, những người thân yêu đã luôn ở bên cạnh tạo điều kiện tốt nhất cho em về mọi mặt để em có thể yên tâm học tập và nghiên cứu trong suốt quãng thời gian qua.

Với điều kiện thời gian cũng như kiến thức của bản thân còn nhiều hạn chế dẫn đến đồ án này không tránh khỏi những thiếu sót. Vì vậy, em mong sẽ nhận được sự góp ý từ các thầy cô để đồ án được hoàn thiện hơn.

Em xin chân thành cảm ơn!

Hà Nội, tháng 06 năm 2024

Tác giả Đồ Án

Vũ Danh Trung Hiếu

TÓM TẮT NỘI DUNG ĐỒ ÁN

Trong bối cảnh nền kinh tế toàn cầu ngày càng phát triển, thị trường tài chính trở nên phức tạp và biến động liên tục, đòi hỏi các nhà đầu tư phải không ngừng tìm kiếm và áp dụng các phương pháp để nâng cao hiệu quả giao dịch. Trong số các phương pháp hiện đại, giao dịch định lượng đã và đang chứng minh được tiềm năng và hiệu quả của mình trong lĩnh vực này.

Giao dịch định lượng dựa trên việc sử dụng các mô hình và thuật toán toán học để đưa ra quyết định giao dịch, từ đó tối ưu hóa lợi nhuận và giảm thiểu rủi ro. Một trong những yếu tố then chốt góp phần tạo nên sự thành công của chiến lược này là khả năng phân tích và xử lý khối lượng dữ liệu khổng lồ (Big Data) để dự đoán biến động giá và xác định các cơ hội giao dịch tiềm năng một cách chính xác và nhanh chóng.

Đồ án này tập trung nghiên cứu và ứng dụng Big Data trong giao dịch định lượng. Cụ thể, đồ án sẽ gồm các phần chính:

- Tổng quan về Big Data và giao dịch định lượng: Giới thiệu về Big Data, các công nghệ và phương pháp xử lý dữ liệu lớn, cùng với các mô hình và thuật toán trong giao dịch định lượng.
- Xây dựng chiến lược tối ưu hóa danh mục đầu tư: Áp dụng các mô hình toán học và thuật toán để tối ưu hóa danh mục đầu tư, đảm bảo lợi nhuận tối đa và rủi ro tối thiểu.
- Phân tích dữ liệu lịch sử giao dịch: Sử dụng các thuật toán và kỹ thuật phân tích dữ liệu lớn để xử lý và khai thác dữ liệu lịch sử giao dịch, từ đó tìm ra các tín hiệu giao dịch tiềm năng.
- Thử nghiệm và đánh giá: Thực hiện các thử nghiệm trên dữ liệu thực tế để đánh giá hiệu quả của các chiến lược đã xây dựng, từ đó đưa ra các đề xuất cải thiện và hướng phát triển trong tương lai.

Em hy vọng rằng đồ án này sẽ cung cấp cái nhìn sâu sắc về vai trò của Big Data trong giao dịch định lượng, từ đó mở ra những cơ hội nghiên cứu và phát triển mới trong lĩnh vực này.

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

1.1 Đặt vấn đề

Trong thời đại cách mạng công nghệ 4.0, dữ liệu lớn (Big Data) đã trở thành một yếu tố quan trọng và phổ biến trong hầu hết các lĩnh vực, trong đó có lĩnh vực đầu tư tài chính. Sự phát triển mạnh mẽ của công nghệ thông tin và truyền thông đã tạo ra một lượng dữ liệu khổng lồ từ nhiều nguồn khác nhau trên mạng. Điều này đặt ra một thách thức lớn trong việc thu thập, lưu trữ, xử lý và phân tích dữ liệu một cách hiệu quả để mang lại giá trị.

Trong bối cảnh thị trường tài chính ngày càng phức tạp và biến động, việc tìm kiếm các phương pháp giao dịch hiệu quả và đáng tin cậy đã trở thành một thách thức lớn đối với các nhà đầu tư. Trước đây, các quyết định đầu tư thường dựa trên kinh nghiệm và cảm nhận của các nhà đầu tư. Tuy nhiên, với sự phát triển mạnh mẽ của công nghệ thông tin và khoa học dữ liệu, một cách tiếp cận mới dựa trên các phương pháp định lượng đã xuất hiện và ngày càng được ưa chuộng. Giao dịch định lượng, hay còn gọi là giao dịch thuật toán, sử dụng các mô hình toán học và phân tích dữ liệu lớn để đưa ra các quyết định đầu tư một cách khách quan và chính xác.

Với sự bùng nổ của dữ liệu và lượng thông tin khổng lồ được tạo ra mỗi giây mỗi phút từ giao dịch, tin tức, mạng xã hội và dữ liệu kinh tế khiến thị trường tài chính ngày càng trở nên phức tạp và nhiều biến động. Trong môi trường cạnh tranh gay gắt này, khả năng khai thác và ứng dụng Big Data hiệu quả đã trở thành yếu tố then chốt dẫn đến thành công cho các nhà giao dịch định lượng để đưa ra quyết định đầu tư. Điều này đặt ra một vấn đề cấp bách: làm thế nào để hiệu quả hóa việc sử dụng dữ liệu khổng lồ này để đưa ra quyết định giao dịch đúng đắn trong thời gian ngắn nhất ?

Việc hiểu dữ liệu giúp các nhà đầu có thể hiểu rõ hơn về thị trường,xu hướng tài chính để đưa ra quyết định chính xác. Phân tích dữ liệu cũng giúp cải thiện trải nghiệm người dùng, tối ưu hoạt động và nâng cao hiệu quả chiến dịch đầu tư tài chính, quản trị rủi ro. Do đó, việc xây dựng một hệ thống để thu thập, xử lý và phân tích dữ liệu từ thị trường tài chính để phục vụ các nhu cầu liên quan đến phân tích thị trường, dự báo xu hướng cổ phiếu, tối ưu hóa chiến lược đầu tư và quản trị rủi ro một cách hiệu quả.

Với tầm quan trọng của việc tối ưu hóa sức mạnh của Big Data trong giao dịch định lượng, em quyết định chọn đề tài "Big Data for Quantitative Trading".

1.2 Mục tiêu và phạm vi đề tài

1.2.1 Mục tiêu

Mục tiêu của đề tài này là xây dựng một hệ thống giao dịch định lượng hiệu quả thông qua việc ứng dụng công nghệ dữ liệu lớn (Big Data) và các thuật toán giao dịch vào quá trình phân tích và đầu tư. Đầu tiên, đề tài phát triển quy trình thu thập và làm sạch các dữ liệu đã được thu thập. Tiếp theo, đề tài sẽ áp dụng lý thuyết danh mục đầu tư hiện đại để tối ưu hóa phân bổ tài sản để giảm thiểu rủi ro và tối đa hóa lợi nhuận. Ngoài ra, các chỉ số kỹ thuật cũng sẽ được áp dụng để phát triển các chiến lược giao dịch, xác định tín hiệu mua và bán cổ phiếu một cách hiệu quả. Để đảm bảo tính khả thi và hiệu quả của các chiến lược này, một hệ thống backtest sẽ được thiết lập để kiểm tra và đánh giá hiệu suất trên dữ liệu lịch sử.

1.2.2 Phạm vi đề tài

Phạm vi của đề tài bao gồm các nội dung chính như phân tích giá của các mã cổ phiếu để tối ưu danh mục đầu tư, tìm tín hiệu giao dịch của các mã cổ phiếu, và thực hiện backtest để kiểm tra hiệu suất của các chiến lược giao dịch được đặt ra. Cụ thể, hệ thống sẽ thu thập và xử lý dữ liệu về giá của các mã cổ phiếu đang được niêm yết trên sàn chứng khoán, sử dụng phương pháp phân tích kỹ thuật để đánh giá hiệu suất và xây dựng mô hình tối ưu hóa danh mục đầu tư, và áp dụng các thuật toán phân tích các chỉ số kỹ thuật để nhận diện các tín hiệu mua và bán. Cuối cùng, hệ thống sẽ thực hiện backtest để kiểm tra và đánh giá hiệu quả của các chiến lược đầu tư để đảm bảo tính khả thi và hiệu quả. Bằng cách đạt được các mục tiêu và giới hạn trong phạm vi đề tài này, nghiên cứu sẽ cung cấp một cái nhìn tổng quan và các giải pháp thực tiễn cho việc ứng dụng dữ liệu lớn trong giao dịch định lượng, đồng thời đưa ra các khuyến nghị và hướng phát triển cho các nghiên cứu và ứng dụng sau này.

1.3 Định hướng giải pháp

1.3.1 Phương pháp nghiên cứu

Các vấn đề cần nghiên cứu trong đề tài này bao gồm việc thu thập và tiền xử lý dữ liệu lớn, phân tích và mô hình hóa dữ liệu để tạo ra các tín hiệu giao dịch, tối ưu hóa danh mục đầu tư để nâng cao hiệu suất đầu tư, cũng như thiết lập hạ tầng công nghệ để triển khai và thử nghiệm các mô hình giao dịch trên dữ liệu thực tế.

Để thành công trong lĩnh vực này, cần phải có sự kết hợp của kiến thức về tài chính, khoa học dữ liệu và công nghệ thông tin, cũng như khả năng áp dụng các phương pháp và công cụ vào thực tế. Các nghiên cứu không chỉ tập trung vào việc phát triển các mô hình và công cụ, mà còn đề xuất các giải pháp ứng dụng thực tế để cải thiện hiệu suất giao dịch và quản lý danh mục trong môi trường thị trường

tài chính đầy biến động và phức tạp.

1.3.2 Phương án thực hành

Đầu tiên, quá trình bắt đầu với việc thu thập dữ liệu lịch sử giao dịch cổ phiếu từ các sàn chứng khoán. Dữ liệu thu thập được sau đó sẽ được tiền xử lý để loại bỏ nhiễu, điền giá trị thiếu để tạo ra một bộ dữ liệu có chất lượng tốt nhất để sử dụng cho việc phân tích và mô hình hóa.

Tiếp theo, sẽ sử dụng các phương pháp phân tích để tìm ra các tín hiệu giao dịch. Các mô hình giao dịch dựa trên các tín hiệu này sẽ được xây dựng và thử nghiệm trên dữ liệu lịch sử để đánh giá hiệu suất của chúng.

Cuối cùng, các kết quả từ việc triển khai sẽ được sử dụng để tối ưu hóa và cải tiến hệ thống giao dịch. Quá trình này bao gồm việc điều chỉnh các tham số của mô hình và hệ thống dựa trên phản hồi từ thị trường và hiệu suất giao dịch.

Tóm lại, phương án thực hành này cung cấp một quy trình cụ thể và có cấu trúc để nghiên cứu và phát triển các mô hình giao dịch dựa trên dữ liệu lớn. Bằng cách kết hợp các phương pháp phân tích dữ liệu lớn với quy trình thực hành để mọi người có thể tiếp cận và giải quyết các vấn đề quan trọng trong lĩnh vực này một cách có hệ thống và hiệu quả.

1.4 Bố cục đồ án

Các phần còn lại của bài báo cáo này được tổ chức như sau:

Chương 2: Dữ liệu lớn và giao dịch định lượng Chương này sẽ cung cấp một cái nhìn tổng quan về dữ liệu lớn và cách nó ảnh hưởng đến lĩnh vực giao dịch định lượng. Đầu tiên, chúng ta sẽ tìm hiểu khái niệm về dữ liệu lớn và những đặc điểm của nó. Sau đó, sẽ xem xét cách mà dữ liệu lớn đã thay đổi cách mà các nhà đầu tư tiếp cận thị trường tài chính, đặc biệt là trong việc ra quyết định giao dịch dựa trên dữ liệu và mô hình hóa thị trường.

Chương 3: Giới thiệu về các công cụ xử lý dữ liệu lớn Trong chương này, em sẽ tập trung vào việc giới thiệu các công cụ và kỹ thuật phổ biến được sử dụng để xử lý dữ liệu lớn. Điều này sẽ cung cấp một cái nhìn tổng quan về cách hoạt động của các công cụ này từ đó giúp chúng ta hiểu rõ hơn về cách các công cụ này được áp dụng trong các ứng dụng xử lý dữ liệu lớn.

Chương 4: Xây dựng chương trình Trong chương này, chúng ta sẽ xác định cấu trúc chính của chương trình giao dịch dựa trên dữ liệu lớn. Đầu tiên, chúng ta sẽ mô tả mục đích sử dụng của chương trình và giải thích quy trình hoạt động của từng phần của chương trình, từ việc thu thập dữ liệu đến việc xử lý và phân tích dữ liệu, tối ưu danh mục đầu tư, tạo ra các tín hiệu giao dịch và cuối cùng là xây dựng

chương trình backtest để kiểm tra và đánh giá thuật toán.

Chương 5: Kết luận và hướng phát triển Cuối cùng, chương này sẽ tổng kết các kết quả đạt được từ đề tài và đề xuất hướng phát triển và cải tiến cho hệ thống trong tương lai.

CHƯƠNG 2. DỮ LIỆU LỚN VÀ GIAO DỊCH ĐỊNH LƯỢNG

2.1 Giới thiệu về dữ liệu lớn

Dữ liệu lớn là thuật ngữ để mô tả các tập dữ liệu có kích thước lớn và phức tạp, đặc biệt là khi kích thước của chúng vượt xa khả năng xử lý của các công cụ và phương pháp truyền thống [1]. Đây thường là các tập dữ liệu mà không thể được xử lý hoặc phân tích bằng cách sử dụng phần cứng và phần mềm thông thường mà yêu cầu phải sử dụng các công nghệ và công cụ đặc biệt được thiết kế để xử lý dữ liệu lớn.

Dữ liệu lớn không chỉ đến từ các nguồn truyền thống như cơ sở dữ liệu doanh nghiệp, mà còn bắt nguồn từ một loạt các nguồn mới, bao gồm trang web, mạng xã hội, thiết bị di động, cảm biến, máy móc, và hệ thống giao dịch. Sự phát triển nhanh chóng của Internet và Công nghệ thông tin đã tạo ra một luồng dữ liệu không ngừng, mang theo những cơ hội và thách thức mới cho xã hội và kinh tế.

Dữ liệu lớn có 5 đặc trưng cơ bản như sau (mô hình "5V"):

- Khối lượng dữ liệu (Volume): Đây là đặc điểm tiêu biểu nhất của dữ liệu lớn, khối lượng dữ liệu rất lớn. Dữ liệu truyền thống có thể lưu trữ trên các thiết bị đĩa mềm, đĩa cứng. Nhưng với dữ liệu lớn chúng ta sẽ sử dụng công nghệ “đám mây” mới đáp ứng khả năng lưu trữ được dữ liệu lớn.
- Tốc độ (Velocity): Dữ liệu lớn được tạo ra, thu thập và truyền đi với tốc độ nhanh chóng. Dữ liệu có thể được tạo ra liên tục từ nhiều nguồn khác nhau trong thời gian thực hoặc gần thời gian thực.
- Dữ liệu lớn có tính đa dạng và độ phức tạp cao. Nó bao gồm không chỉ dữ liệu cấu trúc như trong cơ sở dữ liệu truyền thống, mà còn dữ liệu phi cấu trúc như văn bản, hình ảnh, video, âm thanh, dữ liệu xã hội vànhiều nguồn dữ liệu khác.
- Chính xác (Veracity): Đề cập đến tính tin cậy và độ chính xác của dữ liệu, đảmbảo sự đáng tin cậy khi sử dụng dữ liệu trong quá trình phân tích và ra quyết định.
- Giá trị (Value): Giá trị là đặc điểm quan trọng nhất của dữ liệu lớn, vì khi bắt đầu triển khai xây dựng dữ liệu lớn thì việc đầu tiên chúng ta cần phải làm đó là xác định được giá trị của thông tin mang lại như thế nào, khi đó chúng ta mới có quyết định có nên triển khai dữ liệu lớn hay không.

Dữ liệu lớn mang đến cơ hội vô tận cho các tổ chức và doanh nghiệp. Khả năng

phân tích và khai thác dữ liệu lớn giúp tối ưu hoá hoạt động kinh doanh, đưa ra quyết định thông minh và tạo ra lợi thế cạnh tranh. Ngoài ra, dữ liệu lớn còn là nguồn tài nguyên quý giá cho nghiên cứu khoa học, phát triển sản phẩm và dịch vụ mới. Bằng cách hiểu và khai thác tốt dữ liệu lớn, các tổ chức có thể tiếp cận thông tin chính xác và chi tiết, khám phá những tiềm năng, và tận dụng cơ hội để đạt được sự phát triển và thành công bền vững.

Tuy nhiên, dữ liệu lớn mang đến không chỉ cơ hội mà còn nhiều thách thức đáng kể. Xử lý và quản lý khối lượng dữ liệu lớn, đa dạng và không chắc chắn yêu cầu sự phát triển của công nghệ và phương pháp mới. Ngoài ra, tốc độ sinh ra dữ liệu ngày càng nhanh đòi hỏi khả năng xử lý kịp thời để đáp ứng hiệu quả và đáng tin cậy.

Tóm lại, dữ liệu lớn đem lại cả cơ hội và thách thức. Việc hiểu và vượt qua những thách thức này sẽ mở ra tiềm năng to lớn để tận dụng dữ liệu và đạt được lợi ích sâu sắc từ dữ liệu lớn.

2.2 Giới thiệu về giao dịch định lượng

Giao dịch định lượng là việc sử dụng các mô hình và tính toán toán học, thống kê phức tạp để xác định các cơ hội sinh lời trên thị trường tài chính [2]. Giao dịch định lượng được biết là sử dụng các công nghệ hiện đại tiên tiến trên cơ sở dữ liệu khổng lồ để cung cấp các phân tích toàn diện về các cơ hội hiện có trên thị trường.

Ý tưởng của giao dịch định lượng là tạo ra các ý tưởng giao dịch vững chắc hoàn toàn bằng cách sử dụng các mô hình toán học. Một nhà giao dịch định lượng sẽ nghiên cứu và phân tích dữ liệu lịch sử, sau đó tiến hành áp dụng các mô hình toán học và thống kê tiên tiến để chọn ra các cơ hội giao dịch trên thị trường. Các ý tưởng giao dịch sau đó có thể được thực hiện thủ công hoặc tự động trên thị trường.

Giao dịch định lượng hoạt động bằng cách phát triển các mô hình dựa trên dữ liệu lịch sử và phân tích thống kê để dự đoán diễn biến thị trường. Nhà giao dịch sử dụng các mô hình này để xác định các cơ hội giao dịch có lợi nhuận. Quá trình này bao gồm:

- Thu thập dữ liệu: Thu thập dữ liệu lịch sử thị trường, báo cáo tài chính và mọi thông tin liên quan.
- Phát triển mô hình: Sử dụng phương pháp thống kê để phát triển mô hình dự đoán.
- Backtesting: Kiểm tra mô hình dựa trên dữ liệu lịch sử để đánh giá tính hiệu quả của nó.
- Thực thi: Triển khai mô hình trong giao dịch thời gian thực, thường sử dụng

hệ thống giao dịch tự động.

Giao dịch định lượng mang lại nhiều lợi ích, bao gồm khả năng xử lý và phân tích lượng lớn dữ liệu một cách nhanh chóng, tính khách quan trong quyết định đầu tư, và khả năng tối ưu hóa danh mục đầu tư dựa trên các tiêu chí rủi ro và lợi nhuận.

Tuy nhiên, nó cũng đối mặt với các thách thức như yêu cầu về chất lượng dữ liệu cao, rủi ro từ mô hình không chính xác và sự cạnh tranh gay gắt trong ngành. Mặc dù vậy, giao dịch định lượng đã trở thành một phần quan trọng của thị trường tài chính hiện đại, đóng góp vào việc cải thiện hiệu quả và tính minh bạch của hoạt động giao dịch. Do vậy, phương pháp này thường phù hợp với những nhà đầu tư có kinh nghiệm và am hiểu về thị trường tài chính.

2.2.1 Dữ liệu lớn trong giao dịch định lượng

Dữ liệu lớn đã cách mạng hóa nhiều lĩnh vực, bao gồm cả giao dịch định lượng. Dữ liệu lớn trong giao dịch định lượng đề cập đến việc sử dụng khối lượng dữ liệu khổng lồ, đa dạng và được tạo ra nhanh chóng để phân tích và ra quyết định đầu tư. Sự kết hợp giữa dữ liệu lớn và các mô hình, thuật toán toán học phức tạp đã mở ra những cơ hội mới và nâng cao hiệu quả của các chiến lược đầu tư.

Giao dịch định lượng sử dụng lượng lớn dữ liệu từ nhiều nguồn khác nhau, bao gồm dữ liệu thị trường, báo cáo tài chính, tin tức kinh tế, mạng xã hội. Dữ liệu tài chính được tạo ra liên tục với tốc độ chóng mặt, chẳng hạn như dữ liệu giao dịch mỗi giây hoặc thậm chí mili giây, đòi hỏi hệ thống giao dịch phải có khả năng xử lý dữ liệu theo thời gian thực. Hơn nữa, dữ liệu lớn trong giao dịch định lượng không chỉ bao gồm dữ liệu số truyền thống mà còn bao gồm các dạng dữ liệu phi cấu trúc như văn bản, hình ảnh và video.

Ứng dụng của dữ liệu lớn trong giao dịch định lượng rất đa dạng. Sử dụng mô hình học máy và trí tuệ nhân tạo, nhà đầu tư có thể phân tích dữ liệu lịch sử và dự đoán xu hướng trong tương lai, từ đó cải thiện các quyết định đầu tư. Khai thác các mô hình và mối quan hệ ẩn trong dữ liệu lớn giúp khám phá các cơ hội đầu tư tiềm năng. Ngoài ra, dữ liệu lớn còn được sử dụng để đo lường và quản lý rủi ro trong danh mục đầu tư, tối ưu hóa việc phân bổ tài sản để đạt được sự cân bằng tốt nhất giữa rủi ro và lợi nhuận. Áp dụng thuật toán giao dịch tự động dựa trên dữ liệu lớn cũng giúp thực hiện giao dịch nhanh chóng và chính xác, tối ưu hóa lợi nhuận.

Dữ liệu lớn mang lại nhiều lợi ích cho giao dịch định lượng. Nó làm tăng khả năng phân tích và đưa ra quyết định nhanh hơn và chính xác hơn các phương pháp truyền thống. Dựa hoàn toàn vào dữ liệu và mô hình, giao dịch định lượng giảm thiểu ảnh hưởng của tâm lý và cảm xúc trong quyết định đầu tư.

Tuy nhiên, nó cũng gặp nhiều thách thức. Chất lượng dữ liệu phải chính xác và đáng tin cậy, vì dữ liệu không chính xác có thể dẫn đến những quyết định sai lầm. Các mô hình phức tạp dễ mắc lỗi nếu không được kiểm tra và xác nhận kỹ lưỡng. Ngoài ra, khai thác dữ liệu lớn đòi hỏi đầu tư lớn vào công nghệ, phần mềm và phần cứng để xử lý và phân tích dữ liệu.

Dù đối mặt với nhiều thách thức, sự phát triển của dữ liệu lớn và giao dịch định lượng là một xu hướng không thể đảo ngược trong ngành tài chính. Các công nghệ mới như trí tuệ nhân tạo và học máy tiếp tục phát triển, mang lại những cơ hội mới và cải tiến hơn nữa cho giao dịch định lượng. Trong tương lai, chúng ta có thể mong đợi sự gia tăng hơn nữa về khả năng và hiệu quả của các mô hình giao dịch, đồng thời mở ra những hướng đi mới trong việc phân tích và quản lý tài sản tài chính.

2.2.2 Kết luận

Dữ liệu lớn đã và đang thay đổi cách thức giao dịch và đầu tư trên thị trường tài chính. Bằng cách tận dụng sức mạnh của dữ liệu và công nghệ tiên tiến, các nhà đầu tư có thể đưa ra các quyết định thông minh hơn, tối ưu hóa danh mục đầu tư và quản lý rủi ro một cách hiệu quả. Tuy nhiên, để thành công trong việc áp dụng dữ liệu lớn vào giao dịch định lượng, các nhà đầu tư cần phải quản lý tốt các thách thức về chất lượng dữ liệu, rủi ro mô hình và bảo mật thông tin. Sự phát triển liên tục của công nghệ và việc áp dụng các phương pháp tiếp cận mới sẽ tiếp tục thúc đẩy lĩnh vực này tiến lên phía trước, mang lại những giá trị to lớn cho thị trường tài chính toàn cầu.

CHƯƠNG 3. CÔNG NGHỆ XỬ LÝ DỮ LIỆU LỚN

3.1 Công nghệ Apache Spark

Apache Spark là một hệ thống phân tích dữ liệu đa năng và nhanh chóng, được sử dụng cho xử lý dữ liệu quy mô lớn trên các cụm máy tính phân tán. Apache Spark cung cấp các API phát triển bằng ngôn ngữ Java, Scala, Python và R và hỗ trợ tái sử dụng mã trên nhiều khối lượng công việc, chẳng hạn như xử lý lô dữ liệu, truy vấn tương tác, phân tích theo thời gian thực, máy học và xử lý đồ thị [3].

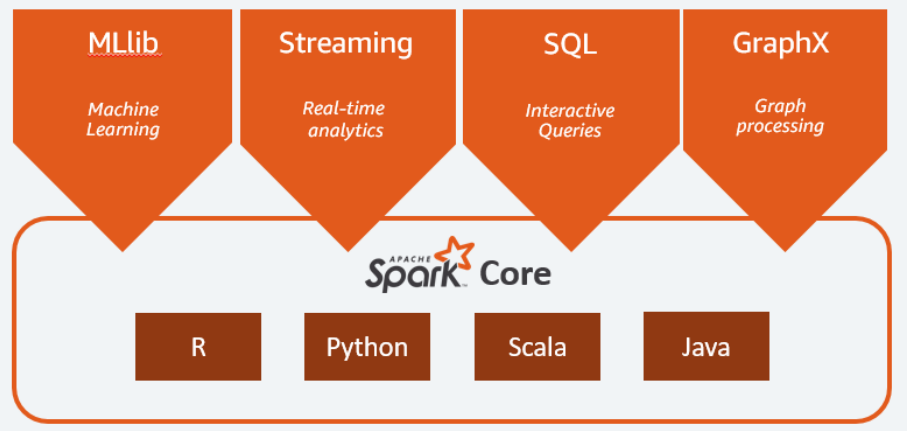
Hình 3.1: Apache Spark

Apache Spark có nhiều đặc điểm quan trọng làm nổi bật framework này trong lĩnh vực xử lý dữ liệu lớn. Đầu tiên, Spark cho phép xử lý dữ liệu phân tán, tức là nó có khả năng chia nhỏ và phân phối dữ liệu trên nhiều máy tính để tăng tốc độ xử lý. Thứ hai, Spark tích hợp nhiều công cụ và ngôn ngữ lập trình, giúp cho người dùng có thể sử dụng ngôn ngữ ưa thích của họ để phát triển ứng dụng.

Apache Spark bao gồm các thành phần chính sau:

- Spark Core: Spark Core là thành phần cốt lõi của Apache Spark, cung cấp các tính năng và khả năng cơ bản cho việc xử lý dữ liệu lớn. Nó cung cấp API cho việc phân chia dữ liệu, quản lý tác vụ và tương tác với dữ liệu trên các cụm máy tính phân tán. Spark Core là nền tảng cho tất cả các thư viện và thành phần khác của Spark.
- Spark SQL: Spark SQL là một thành phần của Spark được sử dụng để xử lý dữ liệu cấu trúc thông qua các truy vấn SQL hoặc các biểu thức dựa trên DataFrame. Điều này cho phép người dùng truy vấn và xử lý dữ liệu dưới dạng bảng quan hệ, giống như cách xử lý trong cơ sở dữ liệu quan hệ thông thường.
- Spark Streaming: Spark Streaming là một thư viện cho việc xử lý dữ liệu trực tuyến (real-time) trong Spark. Nó cho phép xử lý dữ liệu đến từ các nguồn như Apache Kafka, Flume hoặc Twitter trong các batch nhỏ, giúp xử lý dữ liệu theo thời gian thực.

- MLlib (Machine Learning Library): Thư viện hỗ trợ xây dựng và triển khai các mô hình học máy trên dữ liệu lớn. Spark MLlib cung cấp nhiều thuậttoán máy học phổ biến như phân loại, gom cụm, hồi quy và phân tích lân cận gần
- GraphX: Spark GraphX là một khung xử lý đồ thị phân tán được xây dựng trên Spark. GraphX cung cấp ETL, phân tích thăm dò và điện toán đồ thị lặp lại để giúp người dùng xây dựng một cách tương tác và chuyển đổi cấu trúc dữ liệu đồ thị ở quy mô lớn. Công cụ này đi kèm với một API rất linh hoạt và một bộ các thuật toán Đồ thị phân tán được tuyển chọn.



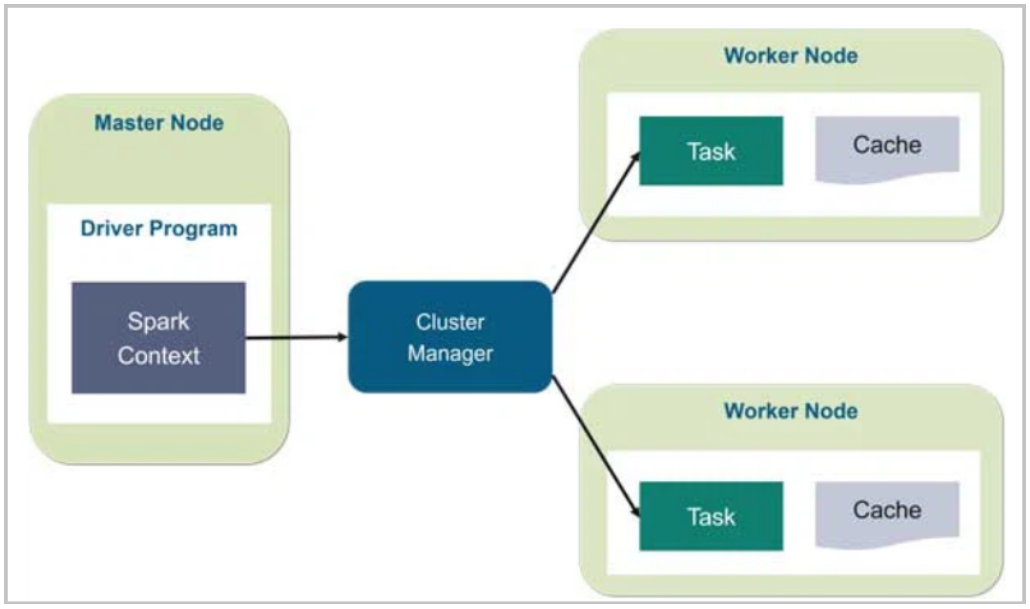
Hình 3.2: Apache Spark Core

Kiến trúc của Apache spark gồm 2 trình cơ bản, đó là: trình điều khiển và trình thực thi. Trong đó

- Trình điều khiển thực hiện chức năng chuyển đổi mã từ người dùng thành các tác vụ khác nhau. Loại này sẽ phân phối trên các nút xử lý.
- Trình thực thi hoạt động trên các nút xử lý. Nó có chức năng thực hiện các nhiệm vụ được giao trên các nút xử lý đó.

Trong kiến trúc này, Apache spark tạo ra các lệnh xử lý dữ liệu từ người dùng. Lệnh này ở dạng đồ thị vòng có hướng hoặc tạo ra lớp lập lịch DAG. Chúng có thể xác định các tác vụ được thực thi trên từng nút tương ứng.

Apache Spark là một công cụ mạnh mẽ cho việc xử lý dữ liệu lớn, cung cấp tính linh hoạt và hiệu suất cao cho các ứng dụng xử lý dữ liệu phân tán và phân tích dữ liệu lớn. Với sự tích hợp đa ngôn ngữ, hiệu suất cao và nhiều tính năng mạnh mẽ, Spark đóng vai trò quan trọng trong việc phát triển các ứng dụng dữ liệu lớn và phân tích dữ liệu.



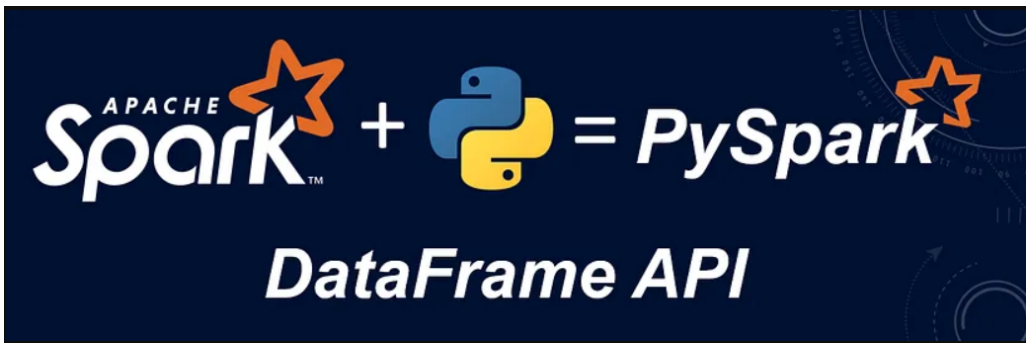
Hình 3.3: Mô hình kiến trúc Apache Spark

3.2 Công cụ xử lý dữ liệu lớn trong Python PySpark

PySpark là một thư viện mã nguồn mở của Apache Spark được viết bằng ngôn ngữ lập trình Python. Nó cung cấp một giao diện dễ sử dụng để lập trình ứng dụng xử lý dữ liệu lớn trên cụm máy tính phân tán. PySpark cho phép người dùng tận dụng các tính năng và khả năng mạnh mẽ của Apache Spark mà không cần phải sử dụng ngôn ngữ Scala, ngôn ngữ chính thức của Spark [4].

Điều đặc biệt của PySpark không chỉ là khả năng xử lý dữ liệu mạnh mẽ, mà còn là sự cung cấp của shell PySpark, giúp người dùng có thể phân tích dữ liệu một cách tương tác, linh hoạt và trực quan.

PySpark kết hợp khả năng sử dụng dễ dàng của Python với sức mạnh của Apache Spark, nhằm mục đích xử lý và phân tích dữ liệu ở mọi quy mô cho mọi người, đặc biệt là những người quen thuộc với Python.



Hình 3.4: PySpark

Ngoài ra, PySpark cung cấp đầy đủ hỗ trợ cho tất cả các tính năng chính của

Spark, bao gồm Spark SQL cho truy vấn dữ liệu dạng SQL, DataFrames cho xử lý dữ liệu cấu trúc, Structured Streaming cho xử lý dữ liệu luồng cấu trúc, Machine Learning cho các ứng dụng, và Spark Core cho các tác vụ xử lý dữ liệu cơ bản.

PySpark mang lại nhiều tính năng và lợi ích, trong đó bao gồm:

- Cung cấp một API Python quen thuộc, giúp người dùng làm việc với Spark một cách dễ dàng.
- Hỗ trợ nhiều nguồn và định dạng dữ liệu khác nhau, bao gồm dữ liệu có cấu trúc, bán cấu trúc và không có cấu trúc.
- Tích hợp tốt với các thư viện và công cụ Python phổ biến khác, như NumPy, pandas và scikit-learn.
- Khả năng mở rộng các công việc Spark trên một cụm máy tính, giúp tăng hiệu suất và xử lý được lượng dữ liệu lớn hơn.

Trong hệ sinh thái Python, chúng ta có nhiều lựa chọn khác nhau để xử lý dữ liệu lớn một cách hiệu quả, nhưng PySpark có những ưu điểm nổi bật:

Ưu điểm	PySpark	Pandas	Dask
Phù hợp với Big Data	Có	Không	Không
Mở rộng	Có	Không	Có
Độ phức tạp	Cao	Thấp	Thấp
Đặc điểm	Xử lý dữ liệu lớn	Xử lý dữ liệu nhỏ	Linh hoạt
Đỗ dễ sử dụng	Trung bình	Cao	Cao
Yêu cầu tài nguyên	Cao	Thấp	Thấp

Bảng 3.1: So sánh các công cụ xử lý dữ liệu

Như vậy ta có thể thấy rằng:

- PySpark: Là công cụ xử lý dữ liệu quy mô lớn và có khả năng mở rộng cao, nhưng yêu cầu người dùng có nền tảng về big data và chấp nhận độ phức tạp cao hơn so với các công cụ khác.
- Pandas: Là công cụ làm việc với dữ liệu nhỏ và vừa, có giao diện thân thiện và tiêu thụ ít tài nguyên hơn.
- Dask: Là công cụ linh hoạt và nhẹ nhàng hơn PySpark, nhưng không đủ hiệu năng hoặc khả năng mở rộng cho các nhiệm vụ xử lý dữ liệu quy mô rất lớn.

PySpark là một trong những công cụ phổ biến nhất trong lĩnh vực Data Science và Machine Learning, vì nó sử dụng Python – ngôn ngữ lập trình có nhiều thư viện hỗ trợ cho khoa học dữ liệu như NumPy và TensorFlow.

PySpark giúp xây dựng các ứng dụng học máy hiệu quả và mạnh mẽ trên dữ liệu khổng lồ, có thể lên đến hàng tỷ hay cả nghìn tỷ bản ghi, với tốc độ cao gấp 100 lần so với các ứng dụng Python bình thường.

Nhiều công ty lớn như Amazon, Walmart, Trivago, Sanofi, và Runtastic đã áp dụng PySpark vào hoạt động của mình. PySpark còn được sử dụng trong đa dạng các ngành nghề khác nhau:

- Y tế
- Tài chính
- Giáo dục
- Giải trí
- Dịch vụ tiện ích
- Thương mại điện tử
- Và nhiều lĩnh vực khác nữa

Tóm lại, PySpark là một công cụ mạnh mẽ và linh hoạt cho việc phát triển các ứng dụng xử lý dữ liệu lớn bằng ngôn ngữ Python, giúp người dùng tận dụng tối đa sức mạnh của Apache Spark mà không cần phải học một ngôn ngữ lập trình mới.

CHƯƠNG 4. XÂY DỰNG CHƯƠNG TRÌNH

4.1 Tổng quan chương trình

Sau khi đã tìm hiểu về các công nghệ xử lý dữ liệu lớn, em sẽ tiến hành xây dựng chương trình. Chương trình sẽ được thiết kế và triển khai gồm các bước:

- Thu thập và xử lý dữ liệu: Để bắt đầu xây dựng các thuật toán, chương trình sẽ tiến hành thu thập dữ liệu lịch sử giao dịch của các mã cổ phiếu. Dữ liệu được thu thập sẽ bao gồm mã cổ phiếu, giá mở cửa, giá đóng cửa, giá thấp nhất trong phiên, giá cao nhất trong phiên và khối lượng giao dịch trong từng phiên. Sau đó ghi dữ liệu đã được thu thập vào dataframe, tiến hành các bước tiền xử lý dữ liệu để thực hiện phân tích.
- Tối ưu danh mục đầu tư: Từ dữ liệu đã thu thập được, lấy các chỉ tiêu cần thiết để xây dựng chương trình. Áp dụng thuật toán tối ưu hóa Sharpe Ratio để tối ưu danh mục chọn ra các mã cổ phiếu đáng đầu tư và trọng số đầu tư của từng mã.
- Xác định tín hiệu giao dịch: Từ các mã cổ phiếu đã đưa ra ở trên, sử dụng đường trung bình động đơn giản (SMA) và chỉ số sức mạnh tương đối (RSI) để xác định các tín hiệu mua bán của từng mã cổ phiếu.
- Sau khi xây dựng các hệ thống trên, tiến hành xây dựng hệ thống backtest áp dụng các chiến thuật đầu tư đã xác định để tiến hành mô phỏng giao dịch dựa trên lịch sử giao dịch của các mã cổ phiếu đã chọn từ đó đưa ra đánh giá và nhận định về các thuật toán đầu tư trên.

Quá trình này không chỉ giúp chúng ta hiểu rõ hơn về các công nghệ xử lý dữ liệu lớn và các thuật toán tối ưu hóa đầu tư, mà còn mang lại những công cụ hữu ích để hỗ trợ quyết định đầu tư một cách thông minh và hiệu quả.

4.2 Thu thập và xử lý dữ liệu

Trong quá trình xây dựng chương trình, việc thu thập dữ liệu đóng vai trò vô cùng quan trọng trong việc cung cấp dữ liệu phục vụ cho việc phân tích và nghiên cứu. Em sẽ tiến hành thu thập dữ liệu từ chuyên trang thông tin tài chính CafeF. Dữ liệu sẽ được thu thập mới nhất đến phiên giao dịch 31/05/2024. Dữ liệu được thu thập về bao gồm tên các mã cổ phiếu, giá mở cửa của từng phiên giao dịch, giá đóng cửa của từng phiên giao dịch, giá thấp nhất và giá cao nhất trong phiên giao dịch.

```
from pyspark.sql import SparkSession, Row
from pyspark.sql.functions import col, when, to_date, count,first,lit,
monotonically_increasing_id, mean, lag, lead, round
from pyspark.sql import functions as F
from pyspark.sql.window import Window
import requests
import pandas as pd
from datetime import datetime, timedelta
spark = SparkSession.builder.appName("Trading")\
        .config("spark.driver.memory", "10g")\
        .getOrCreate()
```

Hình 4.1: Nhập các thư viện và khởi tạo SparkSession

```
# Thu thập các mã cổ phiếu
trade_centers = ['HOSE', 'UPCOM', 'VN30', 'HATSC'] # Các sàn chứng khoán
tickers = []
def crawl_ticker(trade_center):
  response = requests.get("""https://s.cafef.vn/Ajax/PageNew/DataGDNN/
  GDNuocNgoai.ashx?TradeCenter={}&Date=""".format(trade_center))
  if response.status_code == 200:
    records = response.json().get('Data', {}).get('ListDataNN', [])
    for record in records:
      tickers.append({'ticker': record.get('Symbol')})

for trade_center in trade_centers:
  crawl_ticker(trade_center)
tickers = spark.createDataFrame(tickers)
tickers.show()
```

```
+--------+
|  ticker|
+--------+
|     POW|
|     HAH|
|     DCM|
|     YEG|
|     HVN|
|FUEVFVND|
|     GEX|
|     CTG|
|     VNM|
|     DPM|
|     KBC|
|     CCL|
|     AGG|
|     SAB|
|     PLX|
|     VTO|
|     BAF|
```

Hình 4.2: Thu thập tên các mã cổ phiếu

```
schema = StructType([
    StructField("ticker", StringType(), True),
    StructField("day", StringType(), True),
    StructField("open", FloatType(), True),
    StructField("close", FloatType(), True),
    StructField("volume", IntegerType(), True),
    StructField("high", FloatType(), True),
    StructField("low", FloatType(), True)
])
@pandas_udf(schema, PandasUDFType.GROUPED_MAP)
def crawl_data_udf(data):
    stock_data = []
    for _, row in data.iterrows():
        symbol = row['ticker']
        response = requests.get("""https://s.cafef.vn/Ajax/PageNew/DataHistory/
        PriceHistory.ashx?Symbol={}&StartDate=&EndDate=&PageIndex=1&PageSize=10000""".format(symbol))
        if response.status_code == 200:
            records = response.json().get('Data', {}).get('Data', [])
            for record in records:
                stock_data.append({
                    'ticker': symbol,
                    'day': record.get('Ngay'),
                    'open': record.get('GiaMoCua'),
                    'close': record.get('GiaDongCua'),
                    'volume': record.get('KhoiLuongKhopLenh'),
                    'high': record.get('GiaCaoNhat'),
                    'low': record.get('GiaThapNhat')
                })
    return pd.DataFrame(stock_data)
data = tickers.groupby('ticker').apply(crawl_data_udf)
data = data.coalesce(1)  # Gộp tất cả partitions thành 1
data.show()
```

```
/usr/local/lib/python3.10/dist-packages/pyspark/sql/pandas/group_ops.py:104: UserWarning: It is prefer
  warnings.warn(
+------+----------+----+-----+------+----+----+
|ticker|       day|open|close|volume|high| low|
+------+----------+----+-----+------+----+----+
|   A32|07/06/2024|36.0| 36.0|     0| 0.0| 0.0|
|   A32|06/06/2024|36.0| 36.0|     7|36.0|36.0|
|   A32|05/06/2024|36.0| 36.0|    17|36.0|36.0|
|   A32|04/06/2024|36.0| 36.0|     0|36.0|36.0|
|   A32|03/06/2024|36.0| 36.0|   106|36.0|36.0|
|   A32|31/05/2024|37.3| 37.3|     0|37.3|37.3|
|   A32|30/05/2024|37.3| 37.5|    10|37.3|37.3|
|   A32|29/05/2024|37.3| 37.5|    17|37.3|37.3|
```

Hình 4.3: Thu thập thông tin giao dịch

Sau khi thu thập, dữ liệu sẽ được lưu vào dataframe và tiến hành xử lý dữ liệu. Vì các thuật toán trong đề tài này chỉ sử dụng giá đóng cửa của các cổ phiếu nên em sẽ tiến hành loại bỏ các cột không cần thiết và xoay dataframe để tên các mã cổ phiếu và ngày giao dịch thành tên các cột, còn giá đóng cửa của của các phiên sẽ là giá trị tương ứng.

```
data = data.withColumn('day', to_date(data['day'], 'dd/MM/yyyy')).sort('day')
# Chuyển dữ liệu bảng với các cột là tên các ticker, giá trị là giá đóng cửa
table = data.groupBy('day').pivot('ticker').agg(first("close"))
table.orderBy(table["day"].desc()).show()
```

```
+----------+----+-----+---+----+---+----+---+----+----+-----+----+-----+----+-----+-----+----+
|       day| A32|  AAA|AAH| AAM|AAS| AAT|ABB| ABC| ABI|  ABR| ABS|  ABT| ABW|  ACB|  ACC| ACE|
+----------+----+-----+---+----+---+----+---+----+----+-----+----+-----+----+-----+-----+----+
|2024-05-31|37.3|11.85|6.4|8.35|8.4|4.54|8.6|12.2|26.0|13.35|5.32|35.15|10.3|24.65| 14.5|35.4|
|2024-05-30|37.5|12.15|6.2|8.75|8.4|4.51|8.7|12.3|25.8| 13.1|5.37|35.15|10.4| 29.4| 14.5|35.4|
|2024-05-29|37.5|12.25|6.2|8.75|8.5|4.45|8.8|12.4|26.4|13.35|5.42|35.15|10.5|29.25| 14.5|35.5|
|2024-05-28|37.5|11.65|6.4|8.75|8.6|4.38|8.5|12.1|26.3| 13.3|5.28| 34.8|10.0|29.65| 14.8|35.5|
|2024-05-27|37.5| 11.2|6.5|8.81|8.4|4.45|8.6|11.9|26.1| 13.1|5.26| 35.3| 9.9|29.45| 14.8|35.7|
|2024-05-24|35.0| 11.0|5.8|8.87|8.5|4.65|8.6|12.0|25.4| 13.6|5.26| 34.8| 9.9|29.25| 14.9|35.5|
|2024-05-23|33.5|11.45|6.1| 8.8|8.7| 4.7|8.5|12.0|25.4|12.85|5.39| 35.4|10.3|28.45| 14.9|35.7|
|2024-05-22|33.5| 11.3|6.6| 8.8|8.7|4.74|8.5|12.1|24.5| 12.9|5.39| 35.4|10.3|27.95| 14.9|35.6|
|2024-05-21|32.2|11.55|6.7|8.68|8.7|4.77|9.1|12.5|24.6| 12.9|5.38| 35.0|10.4| 28.2|14.85|34.2|
|2024-05-20|32.2|11.45|6.3|8.82|8.6|4.82|8.2|12.1|24.2|13.05|5.37| 33.8| 9.8|28.25| 14.9|35.9|
|2024-05-17|32.2| 11.5|6.0| 8.9|8.6|4.82|7.8|12.0|23.6|13.05|5.29| 35.0| 9.8|28.25|14.15|35.6|
|2024-05-16|32.2| 11.1|5.9| 8.6|8.5| 5.0|7.7|12.0|23.4| 13.9|5.33| 35.0| 9.8|28.25|14.15|34.5|
|2024-05-15|32.2| 11.1|7.0| 8.6|8.5|5.02|7.8|12.0|23.4| 16.9| 5.3| 35.0| 9.8| 27.7|14.15|35.1|
|2024-05-14|32.2| 11.1|6.2| 9.0|8.5| 4.7|7.8|12.0|23.6|17.05|5.35| 33.3| 9.7|27.75| 14.1|35.9|
|2024-05-13|32.2|11.15|5.5| 9.0|8.5| 4.7|7.7|11.8|23.9|16.75|5.39| 33.2| 9.7| 27.7| 14.0|35.9|
|2024-05-10|32.2| 10.9|5.0| 8.9|8.4|4.77|7.9|12.8|23.8| 15.7|5.31| 34.6| 9.7| 27.7| 13.9|35.9|
|2024-05-09|32.2|10.65|4.4| 8.9|8.3|4.75|7.9|12.2|24.0| 14.8|5.16| 34.5| 9.7| 27.6| 13.9|35.9|
|2024-05-08|32.2| 10.7|4.0| 8.9|8.4|4.75|7.8|11.8|24.0|15.25|5.18|34.35| 9.7|27.55|13.85|35.9|
|2024-05-07|32.2|10.65|3.6| 9.0|8.3|4.81|7.7|12.1|24.0|14.45|5.18|34.35| 9.5|27.65|13.95|35.9|
|2024-05-06|32.3| 10.5|3.5| 8.9|8.4|4.71|7.8|12.1|24.0| 14.2|5.17|34.35| 9.7| 27.6| 14.0|36.3|
+----------+----+-----+---+----+---+----+---+----+----+-----+----+-----+----+-----+-----+----+
only showing top 20 rows
```

Hình 4.4: Chuyển các mã cổ phiếu thành các cột

Để cho khách quan, chương trình sẽ chỉ lấy các các mã cổ phiếu hoạt động từ 01/01/2011 và có ít nhất 3250 phiên giao dịch (trung bình mỗi năm có 250 phiên giao dịch). Sau đó tiến hành xử lý các giá trị null do lỗi trong quá trình thu thập dữ liệu từ web (chỉ tính các giá trị null do mất mát dữ liệu chứ không xét những giá trị null do cổ phiếu không giao dịch) bằng cách thay các giá trị null bằng cách lấy trung bình 2 giá trị xung quanh (1 giá trị bên trái nếu giá trị đó không null và 1 giá trị bên phải nếu giá trị đó không null) rồi thay vào giá trị bị null.

```
# Lấy các cổ phiếu từ ngày 01/01/2011
table = table[table['day'] >= '2011-01-01']
columns = table.columns

# Tính số lượng giá trị không null cho mỗi cột
non_null_counts = table.select(
    [count(when(col(c).isNotNull(), 1)).alias(c) for c in columns]
).collect()[0]
non_null_counts_dict = {col: non_null_counts[col] for col in columns}

# Lọc ra các cột có số lượng giá trị không null lớn hơn 3250 phiên giao dịch
columns_to_keep=[col for col, count in non_null_counts_dict.items() if count>3250]

# Tạo DataFrame mới chỉ chứa các cột thỏa mãn điều kiện
table = table.select(columns_to_keep)
table.show()
```

```
+----------+----+----+----+----+----+----+----+----+----+----+----+----+----+----+---+---
|       day| AAA| AAM| ABI| ABT| ACB| ACC| ACL| AGF| AGR| ALV| ANV| APG| ASM|ASP| AT
+----------+----+----+----+----+----+----+----+----+----+----+----+----+----+----+---+---
|2011-01-04|33.5|23.6| 7.1|42.5|25.2|27.0|27.0|23.5|13.4|14.0|14.3|11.7|62.5|8.6|27.
|2011-01-05|32.9|23.4| 6.9|42.0|24.7|27.0|26.5|23.5|13.1|14.1|13.7|11.0|63.5|8.6|26.
|2011-01-06|33.0|23.4| 6.9|42.0|25.0|27.0|26.9|23.1|13.1|13.9|13.8|11.2|63.5|8.7|27.
|2011-01-07|33.3|23.3| 6.9|43.0|24.6|27.0|27.2|23.8|13.2|13.6|13.8|11.0|63.5|8.6|26.
|2011-01-10|33.9|23.2| 6.6|43.5|24.2|27.0|27.0|23.8|13.3|13.9|13.3|11.0|63.0|8.4|27.
|2011-01-11|33.8|23.0| 6.5|43.0|23.6|27.0|26.9|23.1|13.1|14.0|13.0|10.3|61.0|8.4|27.
|2011-01-12|35.0|23.5| 6.6|42.7|23.8|27.0|27.0|22.9|13.3|14.3|13.4|10.9|61.5|8.6|27
```

Hình 4.5: Lấy dữ liệu theo thời gian

```
# Thay các giá trị null bằng trung bình của giá trị xung quanh
def fill_nan_with_neighbors_avg(col_name):
    window_spec = Window.orderBy("id")
    lag_col = lag(col_name).over(window_spec)
    lead_col = lead(col_name).over(window_spec)
    avg_col = (lag_col + lead_col) / 2
    return when(col(col_name).isNull(), round(avg_col, 2)).otherwise(col(col_name))
# Thêm cột id để ghép dữ liệu
table = table.withColumn("id", monotonically_increasing_id()).cache()
# Thực hiện xử lý cho từng cột
for column in table.columns:
    if column != 'day' and column != 'id':
        table = table.withColumn(column, fill_nan_with_neighbors_avg(column))
# Loại bỏ cột "id"
table = table.drop("id")

table.show()
```

```
+----------+----+----+----+----+----+----+----+----+----+----+----+----+----+----+---+----+----
|       day| AAA| AAM| ABI| ABT| ACB| ACC| ACL| AGF| AGR| ALV| ANV| APG| ASM|ASP|ATA| BBC
+----------+----+----+----+----+----+----+----+----+----+----+----+----+----+----+---+----+----
|2011-01-04|33.5|23.6| 7.1|42.5|25.2|27.0|27.0|23.5|13.4|14.0|14.3|11.7|62.5|8.6|27.9|21.5
|2011-01-05|32.9|23.4| 6.9|42.0|24.7|27.0|26.5|23.5|13.1|14.1|13.7|11.0|63.5|8.6|26.6|20.9
|2011-01-06|33.0|23.4| 6.9|42.0|25.0|27.0|26.9|23.1|13.1|13.9|13.8|11.2|63.5|8.7|27.8|21.4
|2011-01-07|33.3|23.3| 6.9|43.0|24.6|27.0|27.2|23.8|13.2|13.6|13.8|11.0|63.5|8.6|26.5|21.4
|2011-01-10|33.9|23.2| 6.6|43.5|24.2|27.0|27.0|23.8|13.3|13.9|13.3|11.0|63.0|8.4|27.5|21.3
|2011-01-11|33.8|23.0| 6.5|43.0|23.6|27.0|26.9|23.1|13.1|14.0|13.0|10.3|61.0|8.4|27.4|20.5
|2011-01-12|35.0|23.5| 6.6|42.7|23.8|27.0|27.0|22.9|13.3|14.3|13.4|10.9|61.5|8.6|27.4|21.1
|2011-01-13|34.8|23.3| 6.8|42.8|24.0|27.0|26.8|23.7|13.4|14.9|13.5|11.0|63.0|8.7|27.3|21.1
|2011-01-14|34.6|23.3| 7.0|43.0|24.0|33.5|26.8|23.6|13.5|15.8|13.5|10.9|66.0|8.7|27.2|21.3
|2011-01-17|34.8|23.2| 7.0|43.5|24.1|33.5|26.8|23.8|13.5|16.9|13.6|11.0|68.5|8.7|27.3|21.0
```

Hình 4.6: Xử lý các giá trị thiếu

Sau khi thực hiện các bước xử lý dữ liệu trên, ta đã có bộ dữ liệu phù hợp cho việc phân tích và tính toán ở các bước sau. Tiếp theo em sẽ dùng bộ dữ liệu trên để tiến hành bước tiếp theo của bài toán là xây dựng chương trình tối ưu danh mục đầu tư dựa vào thuật toán tối ưu hóa Sharpe Ratio.

4.3 Tối ưu danh mục đầu tư

Trong đầu tư tài chính, việc xác định được các mã cổ phiếu nên đầu tư và tỷ lệ phân bổ nguồn vốn cho từng mã là việc rất quan trọng và ảnh hưởng lớn đến hiệu suất đầu tư của nhà đầu tư. Trong phần này em sẽ dựa vào thuật toán tối ưu hóa Sharpe Ratio để xây dựng mô hình tối ưu danh mục đầu tư nhằm rút ngắn danh mục và đưa ra tỷ trọng đầu tư để tối ưu lợi nhuận cho nhà đầu tư.

Thuật toán tối ưu hóa Sharpe Ratio là một phương pháp quan trọng trong việc xây dựng danh mục đầu tư hiệu quả. Sharpe Ratio là một chỉ số đo lường tỷ lệ giữa lợi nhuận của một khoản đầu tư và rủi ro mà nhà đầu tư phải chịu. Mục tiêu của thuật toán này là tìm ra một danh mục đầu tư tối ưu, tức là danh mục có tỷ lệ Sharpe Ratio cao nhất. Quy trình tối ưu hóa Sharpe Ratio bắt đầu bằng việc lựa chọn một tập hợp các tài sản đầu tư có sẵn để xây dựng danh mục. Sau đó, thuật toán sẽ thử nghiệm và tối ưu hóa tỷ lệ phân bổ của mỗi tài sản trong danh mục để tối đa hóa Sharpe Ratio [5]. Tỷ lệ Sharpe được tính bằng công thức:

$$SR = \frac{R_p}{\sigma_p}$$

$$R_p = \sum_{i=1}^{n} w_i r_i$$

$$\sigma_p = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j \sigma_{ij}}$$

Trong đó:

R_p : Lợi nhuận kỳ vọng của danh mục đầu tư.

σ_p : Độ biến động của danh mục đầu tư

w_i : Trọng số mã cổ phiếu thứ i ($0 \leq w_i \leq 1$)

r_i : Lợi nhuận kỳ vọng của cổ phiếu thứ i

σ_{ij} : Ma trận hiệp phương sai tại vị trí (i,j)

Ràng buộc được đặt ra là tổng các trọng số phải bằng 1: $\sum_{i=1}^{n} w_i = 1$

Với bộ dữ liệu bên trên, đầu tiên em sẽ tiến hành tính tỷ lệ thay đổi về giá đóng

cửa của từng mã qua các phiên giao dịch. Sau đó sẽ thực hiện tìm lợi suất kỳ vọng hàng năm bằng cách tính trung bình tỷ lệ thay đổi về giá rồi nhân với số ngày giao dịch trong năm, tiếp theo sẽ đi tính ma trận hiệp phương sai để phục vụ cho bước tối ưu hóa ở dưới.

```
# Tính % thay đổi giữa các ngày
window_spec = Window.orderBy("day")
columns = [col for col in table.columns if col != 'day']
returns_daily = table
column_trans = [((F.col(col) - F.lag(col).over(window_spec)) / F.lag(col).over(window_spec)).alias(col)
for col in returns_daily.columns if col != 'day']

# Sử dụng select() để áp dụng các biến đổi cho tất cả các cột cùng một lúc
returns_daily = returns_daily.select(column_trans)
returns_daily.show()
```

```
+--------------------+--------------------+--------------------+--------------------+--------------------
|                 AAA|                 AAM|                 ABI|                 ABT|                 AC
+--------------------+--------------------+--------------------+--------------------+--------------------
|                NULL|                NULL|                NULL|                NULL|                NUL
|-0.01791044776119...|-0.00847457627118656|-0.02816901408450...|-0.01176470588235...|-0.0198412698412698
|0.003039513677811...|                 0.0|                 0.0|                 0.0| 0.0121457489878542
|0.009090909090909005|-0.00427350427350...|                 0.0|0.023809523809523808|-0.01599999999999..
|0.018018018018018063|-0.00429184549356...|-0.04347826086956532|0.011627906976744186| -0.016260162601626
|-0.00294985250737...|-0.00862068965517...|-0.01515151515151...|-0.01149425287356...|-0.0247933884297519
| 0.03550295857988174|0.021739130434782608| 0.0153846153846153|-0.00697674418604...| 0.0084745762711864
|-0.00571428571428...|-0.00851063829787231| 0.0303030303030303|0.002341920374707127|0.00840336134453778
|-0.00574712643678...|                 0.0| 0.0294117647058823|0.004672897196261749|                 0.0
|0.005780346820809125|-0.00429184549356...|                 0.0|0.011627906976744186|0.00416666666666672
| 0.03448275862068974|-0.00431034482758...|-0.01428571428571...|-0.01609195402298857|-0.01244813278008..
|0.052777777777777726| 0.00865800865800...|0.007246376811594177|0.002336448598130...| 0.00420168067226
```

Hình 4.7: Tính tỷ lệ thay đổi giữa các ngày

```
# Tính trung bình và ma trận hiệp phương sai
returns_daily = returns_daily.toPandas()
expected_returns_annual = returns_daily.mean()
cov_matrix_annual = returns_daily.cov()
print("Loi suat ky vong: \n",expected_returns_annual)
print('==============================================================')
print("Ma tran hiep phuong sai: \n",cov_matrix_annual)
```

```
Loi suat ky vong:
 AAA    0.000042
AAM    0.000026
ABI    0.001170
ABT    0.000215
ACB    0.000182
         ...
VTL   -0.000259
VTO    0.000337
VTS    0.000646
WSB    0.000766
YBC    0.001314
Length: 330, dtype: float64
==============================================================
Ma tran hiep phuong sai:
          AAA       AAM       ABI           ABT       ACB       ACC       ACL
AAA  0.000689  0.000045  0.000110  6.028566e-05  0.000161  0.000072  0.000143
AAM  0.000045  0.000706  0.000033  2.594108e-05  0.000028  0.000005  0.000062
ABI  0.000110  0.000033  0.001609  1.718154e-05  0.000070  0.000088  0.000083
ABT  0.000060  0.000026  0.000017  5.510051e-04  0.000031  0.000019  0.000041
ACB  0.000161  0.000028  0.000070  3.057577e-05  0.000377  0.000044  0.000064
..        ...       ...       ...           ...       ...       ...       ...
```

Hình 4.8: Tính lợi suất kỳ vọng và ma trận hiệp phương sai

Sau khi có được dữ liệu đầu vào cần thiết, em sẽ tiến hành xây dựng hàm tối ưu hóa Sharpe Ratio dựa trên dữ liệu lợi nhuận và hiệp phương sai của các mã cổ phiếu.

```python
import numpy as np
from scipy.optimize import minimize
# Hàm tối ưu hóa Sharpe Ratio
def optimize_portfolio(returns, covariance):
    num_assets = len(returns)
    args = (returns, covariance)
    constraints = ({'type': 'eq', 'fun': lambda x: np.sum(x) - 1})
    bounds = tuple((0.0, 1.0) for asset in range(num_assets))

    def negative_sharpe_ratio(weights, returns, covariance):
        portfolio_return = np.dot(weights, returns)
        portfolio_volatility = np.sqrt(np.dot(weights.T, np.dot(covariance, weights)))
        sharpe_ratio = portfolio_return / portfolio_volatility
        return -sharpe_ratio

    initial_guess = num_assets * [1./num_assets,]
    result = minimize(negative_sharpe_ratio, initial_guess, args=args,
                      method='SLSQP', bounds=bounds, constraints=constraints)
    return result
# Tối ưu hóa danh mục đầu tư
optimal_result = optimize_portfolio(expected_returns_annual, cov_matrix_annual)

# Lấy 5 mã cổ phiếu có tỷ trọng cao nhất
top_indices = np.argsort(optimal_result.x)[-5:]
top_weights = optimal_result.x[top_indices]
top_tickers = [columns[i] for i in top_indices]
sum_weight = np.sum(top_weights)
for i in range(5):
  top_weights[i] = top_weights[i]/sum_weight
```

Hình 4.9: Hàm tối ưu danh mục

Hàm mục tiêu được xác định là `negative_sharpe_ratio`, với mục tiêu là tối đa hóa tỷ lệ Sharpe Ratio. Hàm này sẽ thực hiện tính toán tỷ lệ Sharpe Ratio của một danh mục đầu tư dựa trên các lợi nhuận kỳ vọng và ma trận hiệp phương sai của các tài sản, và trả về giá trị âm của các tỷ trọng (do hàm minimize thực hiện tìm giá trị nhỏ nhất). Cuối cùng, thuật toán tối ưu hóa sử dụng hàm minimize, trong đó thuật toán sử dụng phương pháp "Sequential Least SQuares Programming" (SLSQP) để tìm kiếm trọng số tối ưu của các ma cổ phiếu.

Kết quả cuối cùng được trả về là 1 mảng chứa giá trị trọng số của các mã cổ phiếu từ đó giúp nhà đầu tư hiểu và áp dụng phân bổ tài sản tối ưu cho các mục tiêu đầu tư của mình.

```python
for ticker, weight in zip(top_tickers, top_weights):
    print('Mã cổ phiếu: {}, Tỷ trọng: {}%'.format(ticker,100*weight))
```

```
Mã cổ phiếu: CHP, Tỷ trọng: 18.3406313829264%
Mã cổ phiếu: TDW, Tỷ trọng: 18.681834870965293%
Mã cổ phiếu: SHP, Tỷ trọng: 19.338460538578303%
Mã cổ phiếu: TMP, Tỷ trọng: 21.109226143375732%
Mã cổ phiếu: VCF, Tỷ trọng: 22.52984706415427%
```

Hình 4.10: Danh mục tối ưu

Vì danh sách cổ phiếu là rất lớn và không thể đầu tư hết vào các mã nên ở đây em sẽ thực hiện lấy ra 5 mã cổ phiếu có trọng số lớn nhất tức là 5 mã nên đầu tư nhất trong danh sách các cổ phiếu và lấy ra trọng số của chúng để phân bổ nguồn vốn một cách hợp lý. Các trọng số sẽ được chia lại dựa vào trọng số đã xác định ở thuật toán sao cho tổng trọng số của chúng là 1 tương ứng với tổng nguồn vốn.

4.4 Xác định tín hiệu giao dịch

Sau khi đã xác định được những mã cổ phiếu nên đầu tư nhất, em sẽ tiến hành phân tích để tìm ra các tín hiệu giao dịch mua bán tại từng thời điểm để nhà đầu tư có thể giao dịch hiệu quả từ đó giúp tối đa lợi nhuận nhận được.

4.4.1 Chỉ báo SMA

Ở phần này, em sẽ sử dụng chỉ báo SMA (đường trung bình động đơn giản) để xác định các tín hiệu mua và bán. Chỉ báo SMA được tính bằng công thức:

$$SMA_n = \frac{\sum_{i=1}^{n} P_i}{n}$$

Trong đó:

P : Giá đóng cửa từng phiên giao dịch

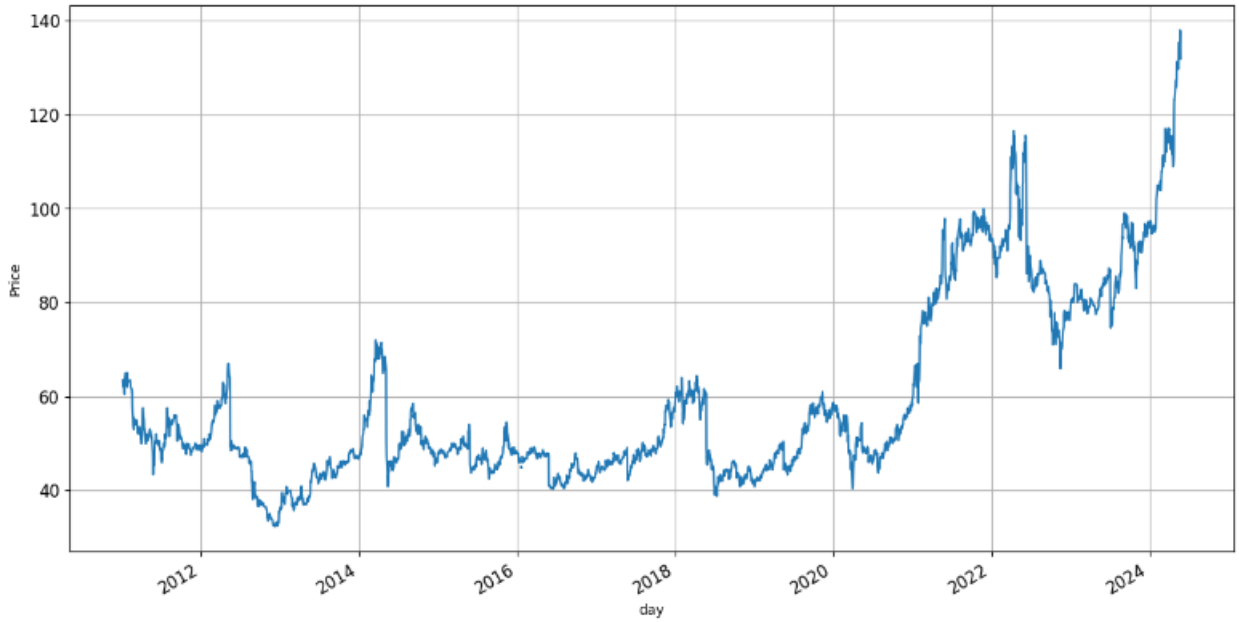
i : Số thứ tự của phiên gần nhất tính từ 1

n : Là số phiên được tính cho SMA

Trong thuật toán này em sẽ sử dụng đường chỉ báo SMA 50 và SMA 200. Để xác định tín hiệu mua và bán người ta sẽ quan sát các điểm cắt nhau của 2 đường này. Cụ thể:

- Tín hiệu mua (Golden Cross): Khi SMA 50 cắt lên trên SMA 200, điều này được coi là một tín hiệu mua. Nó cho thấy sự gia tăng trong đà tăng giá và có thể là dấu hiệu của một xu hướng tăng mới.
- Tín hiệu bán (Death Cross): Ngược lại, khi SMA 50 cắt xuống dưới SMA 200, điều này được coi là một tín hiệu bán. Nó chỉ ra sự suy giảm trong đà tăng giá và có thể là dấu hiệu của một xu hướng giảm mới.

Từ các lý thuyết trên, em sẽ tiến hành xây dựng thuật toán bao gồm việc tính toán các chỉ số SMA 50 và SMA 200 cho từng ngày giao dịch, sau đó dựa vào vị trí đường SMA 50 và SMA 200 để xác định tín hiệu giao dịch. Ở đây em sẽ thực hiện thuật toán cho 1 mã cổ phiếu rồi sau đó sẽ áp dụng để sử dụng cho các mã bất kỳ. Em sẽ chọn mã cổ phiếu của Công ty Cổ phần FPT (FPT) để xây dựng thuật toán.



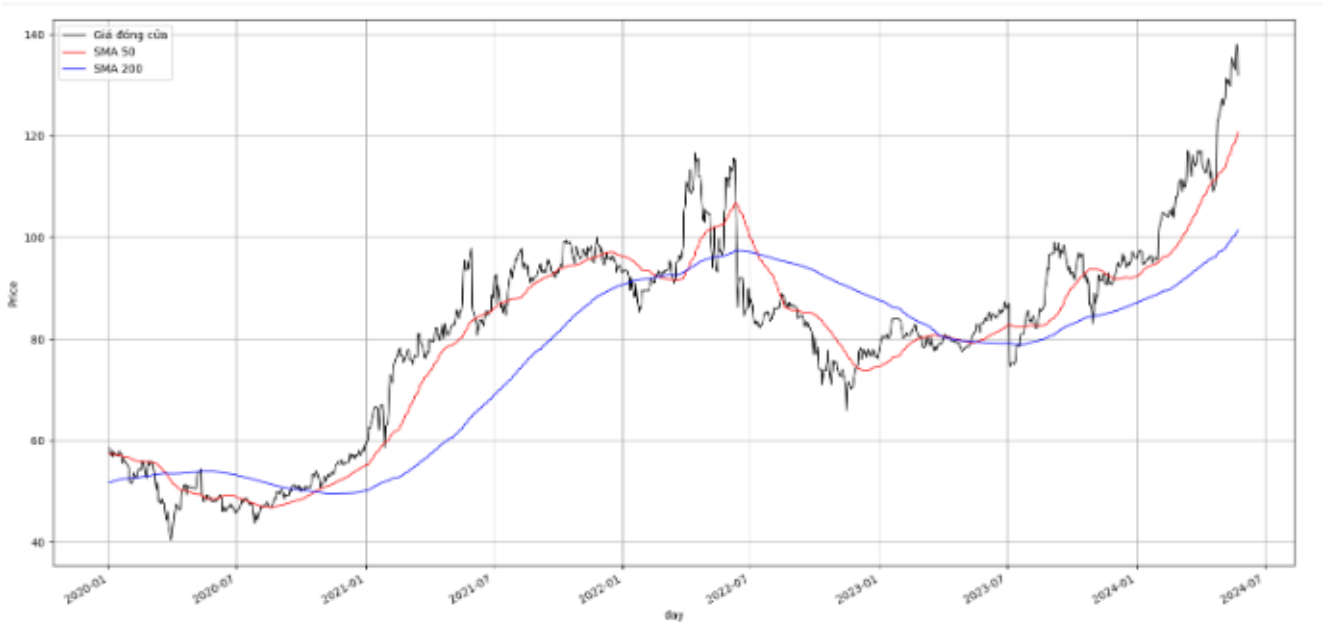
Hình 4.11: Biến động giá của FPT

Sau khi đã có được thông tin dữ liệu lịch sử giao dịch của FPT, em sẽ tiến hành đi tính các chỉ số SMA 50 và SMA 200 theo công thức bên trên.

```
# Tính chỉ số SMA 50 và SMA 200
FPT['50_SMA'] = round(FPT['price'].rolling(window = 50, min_periods = 1).mean(),2)
FPT['200_SMA'] = round(FPT['price'].rolling(window = 200, min_periods = 1).mean(), 2
FPT.head(10)
```

ticker	price	50_SMA	200_SMA
day			
2011-01-04	63.5	63.50	63.50
2011-01-05	62.5	63.00	63.00
2011-01-06	62.5	62.83	62.83
2011-01-07	62.0	62.62	62.62
2011-01-10	62.5	62.60	62.60
2011-01-11	62.5	62.58	62.58
2011-01-12	61.0	62.36	62.36
2011-01-13	60.5	62.12	62.12
2011-01-14	63.5	62.28	62.28
2011-01-17	65.0	62.55	62.55

Hình 4.12: Chỉ số SMA



Hình 4.13: Biểu đồ SMA 50 và SMA 200

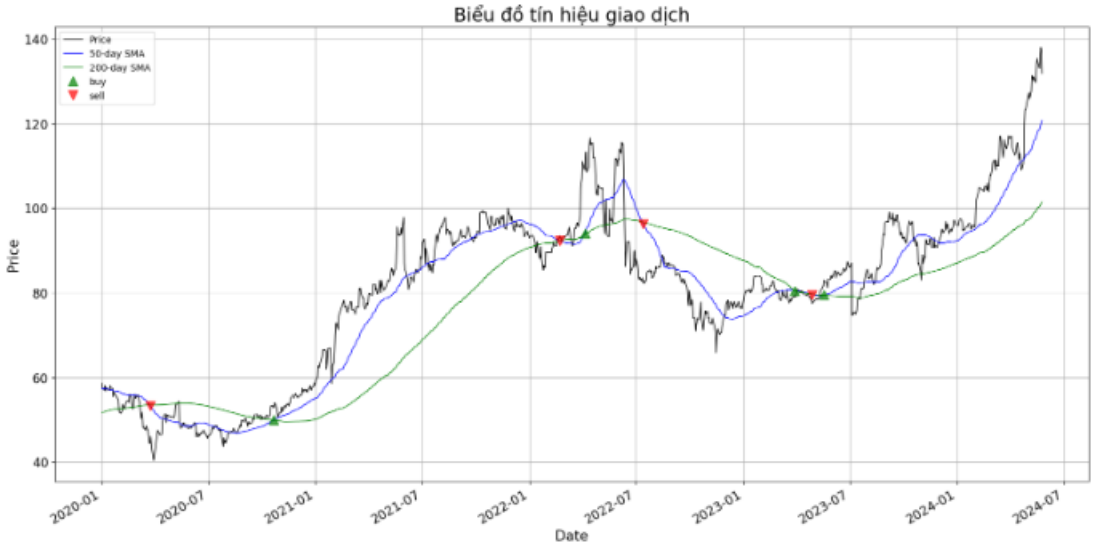
Để cho dễ nhìn em sẽ chỉ vẽ biểu đồ từ năm 2020. Dựa vào biểu đồ, ta có thể thấy tại các điểm đường SMA 50 (đường màu đỏ) cắt đường SMA 200 (đường màu xanh) và đi lên thì giá cổ phiếu có xu hướng tăng, ngược lại khi đường SMA 50 cắt SMA 200 và đi xuống thì giá cổ phiếu có xu hướng giảm. Vì vậy em sẽ tạo cột 'signal' để xác định xu hướng đi tăng giảm của giá cổ phiếu và cột 'position' để xác định thời điểm giao dịch dựa vào sự thay đổi cột 'signal'.

```
# Nếu SMA 50 > SMA 200 thì signal = 1, nếu ko thì signal = 0
FPT['signal'] = 0.0
FPT['signal'] = np.where(FPT['50_SMA'] > FPT['200_SMA'], 1.0, 0.0)
# nếu position = 1: Mua; position = -1: Bán
FPT['position'] = FPT['signal'].diff()
FPT.head(10)
```

ticker	price	50_SMA	200_SMA	signal	position
day					
2020-01-02	58.6	57.34	51.73	1.0	NaN
2020-01-03	57.6	57.32	51.79	1.0	0.0
2020-01-06	57.0	57.30	51.85	1.0	0.0
2020-01-07	58.1	57.30	51.92	1.0	0.0
2020-01-08	56.8	57.28	51.98	1.0	0.0
2020-01-09	57.6	57.28	52.04	1.0	0.0
2020-01-10	57.5	57.27	52.11	1.0	0.0
2020-01-13	57.1	57.24	52.16	1.0	0.0
2020-01-14	57.0	57.18	52.22	1.0	0.0
2020-01-15	57.0	57.13	52.28	1.0	0.0

Hình 4.14: Xác định tín hiệu giao dịch

Sau khi xác định được các chỉ số cần thiết, em sẽ tiến hành trực quan hóa dữ liệu để xác định tính đúng đắn của thuật toán.



Hình 4.15: Biểu đồ tín hiệu giao dịch

Từ biểu đồ có thể thấy thuật toán hoạt động chính xác khi ra tín hiệu mua tại các vị trí giá cổ phiếu đi lên và tín hiệu bán tại các vị trí giá cổ phiếu đi xuống. Nếu giả định rằng các tín hiệu mua và bán được thực hiện theo đúng thời điểm như biểu đồ chỉ ra, có thể thấy rằng chiến lược này có thể đã mang lại lợi nhuận, đặc biệt là khi giá tăng mạnh sau các tín hiệu mua.

4.4.2 Chỉ báo RSI

Chỉ báo RSI (Relative Strength Index) là chỉ báo động lượng dùng để xác định xu hướng tiếp diễn của cổ phiếu. Đây là một chỉ báo thông dụng được biểu hiện dưới dạng dao động (một đường dao động giữa 2 biên) từ 0 đến 100 mà ở đó dao động dưới 30 điểm được gọi là quá bán và dao động trên 70 điểm là quá mua. Ở vùng quá bán 30 điểm, cổ phiếu được cho là đã bị bán quá nhiều và sẽ có xu hướng đảo chiều tăng [6]. Chỉ số RSI được tính bằng công thức:

$$\text{RSI}_{14} = 100 - \frac{100}{1 + \frac{\text{Average Gain}}{\text{Average Loss}}}$$

Trong đó:

$$\text{Average Gain} = \text{Trung bình lượng tăng giá trong 14 phiên}$$

$$\text{Average Loss} = \text{Trung bình lượng giảm giá trong 14 phiên}$$

Tương tự như trên, em sẽ chọn mã cổ phiếu FPT để xây dựng thuật toán. Dựa

vào công thức RSI bên trên, em sẽ xác định chỉ số RSI cho mã cổ phiếu FPT

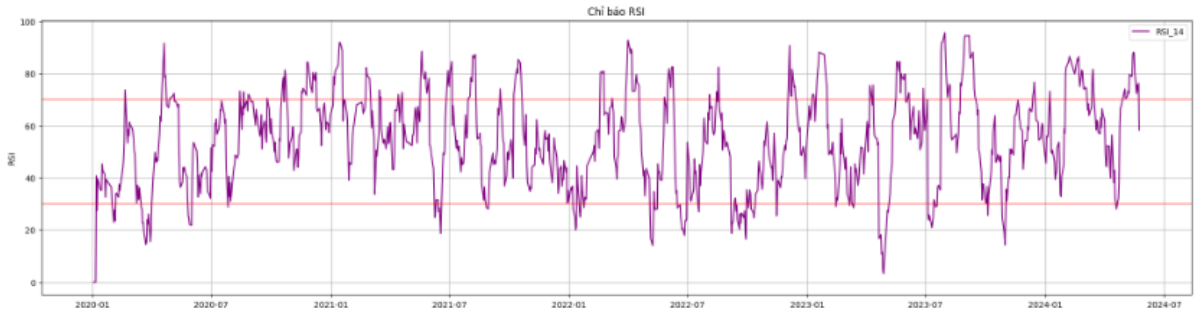
```
def RSI(data):
    delta = FPT['price'].diff()
    gain = (delta.where(delta > 0, 0)).fillna(0)
    loss = (-delta.where(delta < 0, 0)).fillna(0)

    avg_gain = gain.rolling(window=14, min_periods =1).mean()
    avg_loss = loss.rolling(window=14, min_periods =1).mean()

    rs = avg_gain / avg_loss
    rsi = 100 - (100 / (1 + rs))
    return rsi
FPT['RSI'] = RSI(RSI)
FPT.head(20)
```

ticker day	price	RSI
2011-01-04	63.5	NaN
2011-01-05	62.5	0.000000
2011-01-06	62.5	0.000000
2011-01-07	62.0	0.000000
2011-01-10	62.5	25.000000
2011-01-11	62.5	25.000000
2011-01-12	61.0	14.285714

Hình 4.16: Chỉ số RSI 14



Hình 4.17: Biểu đồ chỉ số RSI 14

4.4.3 Xác định tín hiệu mua và bán

Dựa vào 2 chỉ báo SMA và RSI, em sẽ xác định xu hướng tăng giảm của mã cổ phiếu rồi từ đó tìm điểm giao dịch. Xu hướng tăng giảm được xác định dựa trên:

- Xu hướng tăng: Nếu RSI vượt lên trên mức 50 và SMA 50 cắt trên SMA 200, đây là tín hiệu yếu đi của xu hướng giảm, thị trường đang dịch chuyển sang xu hướng tăng và là thời điểm thích hợp bắt đầu cơ hội mua lên.
- Xu hướng giảm: Nếu RSI vượt xuống mức 50 và SMA 50 cắt dưới SMA 200, đây là tín hiệu yếu đi của xu hướng tăng, thị trường đang dần dịch chuyển sang xu hướng giảm và là thời cơ phù hợp tìm cơ hội bán xuống.

```
FPT['signal'] = 0.0
FPT['signal'] = np.where((FPT['RSI'] <= 50) & (FPT['50_SMA'] > FPT['200_SMA']), 2.0,FPT['signal'])
FPT['signal'] = np.where((FPT['RSI'] >= 50) & (FPT['50_SMA'] < FPT['200_SMA']), -1.0,FPT['signal'])
# nếu position = 2: Mua; position = -1: Bán
FPT['position'] = FPT['signal'].diff()
FPT.head(10)
```

ticker day	price	50_SMA	200_SMA	RSI	signal	position
2020-01-02	58.6	57.34	51.73	62.121212	0.0	NaN
2020-01-03	57.6	57.32	51.79	53.333333	0.0	0.0
2020-01-06	57.0	57.30	51.85	53.333333	0.0	0.0
2020-01-07	58.1	57.30	51.92	68.000000	0.0	0.0
2020-01-08	56.8	57.28	51.98	57.471264	0.0	0.0
2020-01-09	57.6	57.28	52.04	59.782609	0.0	0.0
2020-01-10	57.5	57.27	52.11	55.813953	0.0	0.0
2020-01-13	57.1	57.24	52.16	55.813953	0.0	0.0
2020-01-14	57.0	57.18	52.22	53.571429	0.0	0.0
2020-01-15	57.0	57.13	52.28	46.575342	2.0	2.0

Hình 4.18: Xác định tín hiệu giao dịch

Hình 4.19: Biểu đồ tín hiệu giao dịch theo SMA và RSI

4.5 Thử nghiệm mô hình

Trong phần này em sẽ thực hiện 3 chương trình backtest để kiểm tra và đánh giá hiệu quả của thuật toán đầu tư bao gồm:

- Chương trình 1: Chọn ngẫu nhiên 5 mã cổ phiếu với trọng số vốn bất kỳ và thực hiện giao dịch theo tín hiệu giao dịch được đặt ra.
- Chương trình 2: Lấy 5 mã cổ phiếu và trọng số từ danh mục tối ưu và thực hiện giao dịch theo tín hiệu mua bán được đặt ra

Để thuận tiện cho việc thực hiện chương trình backtest em sẽ tạo các hàm `optimize_portfolio` chứa thuật toán tối ưu danh mục đầu tư và `signals` chứa thuật toán tìm tín hiệu giao dịch. Các dữ liệu đầu vào chung của cả 2 chương trình backtest gồm nguồn vốn đầu tư ban đầu là 1 tỷ (VND), ngày bắt đầu giao dịch 01/05/2023 và ngày kết thúc sẽ là ngày giao dịch cuối cùng trong bộ dữ liệu.

4.5.1 Chương trình 1

Chương trình này sẽ lấy ngẫu nhiên 5 mã cổ phiếu trong bộ dữ liệu và thực hiện giao dịch theo tín hiệu giao dịch được đặt ra. Các mã cổ phiếu được chọn và trọng số của các mã được chia như sau.

```
import random
ticker_algo_1 = random.sample(table.columns, 5)
weights_algo_1 = np.random.dirichlet(np.ones(5),size=1)
list_1 = signals(ticker_algo_1)
for i in range(5):
  print('Mã {} trọng số: {}%'.format(ticker_algo_1[i], 100*weights_algo_1[0][i]))
```

```
Mã DHA trọng số: 9.654954891411158%
Mã CLC trọng số: 25.147998833603953%
Mã HQC trọng số: 8.844334013357969%
Mã ABI trọng số: 2.3276410602965103%
Mã HPB trọng số: 54.02507120133041%
```

Hình 4.20: Danh mục giao dịch chương trình 1

Sau khi xác định được mã cổ phiếu và trọng số chia vốn, em sẽ thực hiện chương trình backtest và định lợi nhuận đưa ra.

```
test_day = start_day
# Chia số vốn theo tỷ trọng đã tính
capital = [0] * 5
volumn = [0] * 5
for i in range(5):
  capital[i] = weights_algo_1[0][i] * initial_capital
# Tạo dataframe để lưu tổng lợi nhuận và vốn
total_capital_1 = pd.DataFrame()
total_capital_1.index = list_1[ticker_algo_1[0]].index
total_capital_1['value'] = 0
# Giao dịch
while test_day <= end_day:
  test_day = check(test_day)
  a= 0
  for i, ticker in enumerate(ticker_algo_1):
    if test_day in list_1[ticker].index:
      price = list_1[ticker][ticker].loc[test_day]
      if volumn[i] == 0:     # Mua
          if list_1[ticker]['signal'].loc[test_day] == 2:
            volumn[i] += int((capital[i] * 0.1)/ price)
            capital[i] -= int((capital[i] * 0.1)/ price) * price
      if list_1[ticker]['position'].loc[test_day] == 2 and capital[i] >= price: #Mua
          volumn[i] += int((capital[i] * 0.9)/ price)
          capital[i] -= int((capital[i] * 0.9)/ price) * price
      elif list_1[ticker]['position'].loc[test_day] == -1:     # Bán
          capital[i] += int(volumn[i] * 0.9) * price
          volumn[i] -= int(volumn[i] * 0.9)
      a += (capital[i] + volumn[i] * price)
  total_capital_1['value'].loc[test_day] = a
  test_day += timedelta(days= 1)
  if test_day > end_day:
    break
```

Hình 4.21: Chương trình 1

```
# Tính lợi nhuận
print('Tổng tiền sau đầu tư: {} (tỷ VND)'.format(total_capital_1['value'].iloc[-1]/100000))
print('Lợi nhuận khi thực hiện theo thuật toán: {}%'.format(100*(total_capital_1['value'].iloc[-1] - initial_capital)/initial_capital))
```

```
Tổng tiền sau đầu tư: 1.0865741 (tỷ VND)
Lợi nhuận khi thực hiện theo thuật toán: 8.657409999999988%
```

Hình 4.22: Kết quả chương trình 1

Sau khi thực hiện có thể thấy lợi nhuận đạt được sau 1 khoảng thời gian đầu tư từ 01/05/2023 đến 31/05/2024 là 86.57 triệu VND tương ứng với 8.66% so với nguồn vốn ban đầu.

4.5.2 Chương trình 2

Trong chương trình này, các mã cổ phiếu sẽ được xác định từ danh mục tối ưu và áp dụng thuật toán xác định tín hiệu giao dịch để thực hiện giao dịch. Các mã cổ phiếu được xác định và trọng số chia vốn như sau.

```
top_tickers, top_weights = optimize_portfolio(start_day)
dfs = signals(top_tickers)
for ticker in top_tickers:
  dfs[ticker].set_index('day', inplace=True)
  dfs[ticker] = dfs[ticker].loc[start_day:end_day]
for i in range(5):
  print('Mã {} trọng số: {}%'.format(top_tickers[i], 100*top_weights[i]))
```

```
Mã ND2 trọng số: 18.38541406649654%
Mã TDW trọng số: 18.997795566898994%
Mã TMP trọng số: 19.086838426668713%
Mã PDN trọng số: 20.861391092324823%
Mã VCF trọng số: 22.66856084761092%
```

Hình 4.23: Danh mục giao dịch chương trình 2

Sau khi xác định được mã cổ phiếu và trọng số chia vốn, em sẽ thực hiện chương trình backtest và xác định lợi nhuận đưa ra.

```
test_day = start_day
# Chia số vốn theo tỷ trọng đã tính
capital = [0] * 5
volumn = [0] * 5
for i in range(5):
  capital[i] = top_weights[i] * initial_capital
# Tạo dataframe để lưu tổng lợi nhuận và vốn
total_capital = pd.DataFrame()
total_capital.index = dfs[top_tickers[0]].index
total_capital['value'] = 0
# Giao dịch
while test_day <= end_day:
  test_day = check(test_day)
  a= 0
  for i, ticker in enumerate(top_tickers):
    if test_day in dfs[ticker].index:
      price = dfs[ticker][ticker].loc[test_day]
      if volumn[i] == 0:     # Mua
          if dfs[ticker]['signal'].loc[test_day] == 2:
            volumn[i] += int((capital[i] * 0.1)/ price)
            capital[i] -= int((capital[i] * 0.1)/ price) * price
      if dfs[ticker]['position'].loc[test_day] == 2 and capital[i] >= price: #Mua
          volumn[i] += int((capital[i] * 0.9)/ price)
          capital[i] -= int((capital[i] * 0.9)/ price) * price
      elif dfs[ticker]['position'].loc[test_day] == -1:     # Bán
          capital[i] += int(volumn[i] * 0.9) * price
          volumn[i] -= int(volumn[i] * 0.9)
      a += (capital[i] + volumn[i] * price)
  total_capital['value'].loc[test_day] = a
  test_day += timedelta(days= 1)
  if test_day > end_day:
    break
```

Hình 4.24: Chương trình 2

```
# Tính lợi nhuận
print('Tổng tiền sau đầu tư: {} (tỷ VND)'.format(total_capital['value'].iloc[-1]/100000))
print('Lợi nhuận khi thực hiện theo thuật toán: {}%'.format(100*(total_capital['value'].iloc[-1] - initial_capital)/initial_capital))
```

```
Tổng tiền sau đầu tư: 1.1894665 (tỷ VND)
Lợi nhuận khi thực hiện theo thuật toán: 18.946649999999995%
```

Hình 4.25: Kết quả chương trình 2

Sau khi thực hiện có thể thấy lợi nhuận đạt được sau 1 khoảng thời gian đầu tư

từ 01/05/2023 đến 31/05/2024 là 189.47 triệu VND tương ứng với 18.95% so với nguồn vốn ban đầu.

4.6 Đánh giá kết quả thực nghiệm

Sau khi thực hiện chương trình backtest để kiểm tra hiệu quả hoạt động của thuật toán, em sẽ thực hiện trực quan hóa tổng số tiền thu được của cả 2 chương trình để đưa ra đánh giá hiệu quả hoạt động của các thuật toán.

Hình 4.26: Biểu đồ so sánh lợi nhuận qua từng thời kỳ

Qua biểu đồ biến động lợi nhuận của hai chương trình thử nghiệm, chúng ta có thể đưa ra một số đánh giá như sau:

Chương trình 1 (đường màu đỏ):

- Ban đầu, lợi nhuận của Chương trình 1 dao động quanh mức 1.0 tỷ VND.
- Vào khoảng tháng 7 năm 2023, lợi nhuận giảm mạnh xuống dưới 0.6 tỷ VND nhưng ngay lập tức phục hồi trở lại mức 1.0 tỷ VND.
- Sau đó, lợi nhuận ổn định và dao động nhẹ quanh mức 1.0 tỷ VND cho đến tháng 5 năm 2024.

Chương trình 2 (đường màu xanh lá cây):

- Biểu đồ cho thấy lợi nhuận của chương trình 2 có sự tăng trưởng ổn định và mạnh mẽ hơn so với chương trình 1.
- Lợi nhuận của Chương trình 2 bắt đầu tăng trưởng ngay từ đầu và tiếp tục tăng trong suốt giai đoạn triển khai chương trình.
- Mặc dù có một số dao động, nhưng xu hướng tổng thể là tăng, với mức lợi nhuận cao hơn đáng kể so với chương trình 1.

Dựa trên những đánh giá này, có thể khẳng định rằng chương trình 2 cho ra kết quả đầu tư tốt hơn chương trình 1, nhờ vào sự ổn định và mức lợi nhuận cao hơn so với chương trình 1. Qua đó có thể thấy các thuật toán được đề xuất là hợp lý và có thể áp dụng vào thực tế.

CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 Kết luận

Nhìn chung đề tài đã được xây dựng và đã hoàn thành các mục tiêu quan trọng bao gồm thu thập dữ liệu cổ phiếu từ web, xây dựng thuật toán tối ưu danh mục đầu tư, phát hiện tín hiệu giao dịch và tiến hành backtest. Tuy nhiên, vẫn còn một số hạn chế cần được khắc phục để nâng cao hiệu quả và tính ứng dụng thực tế của hệ thống. Đầu tiên, dữ liệu thu thập được chưa phong phú và đa dạng, điều này có thể ảnh hưởng đến khả năng phản ánh đầy đủ và chính xác của thị trường. Bên cạnh đó, dữ liệu thu thập hiện tại được sử dụng ngay mà không được lưu trữ lại, gây khó khăn cho việc phân tích và sử dụng lại trong tương lai. Hơn nữa, các thuật toán được áp dụng vẫn còn đơn giản và chưa đem lại hiệu quả tối ưu cho nhà đầu tư.

Mặc dù đã đạt được những kết quả bước đầu nhưng đề tài vẫn cần phải tiếp tục cải tiến và hoàn thiện. Việc mở rộng nguồn dữ liệu, lưu trữ và tái sử dụng dữ liệu, cũng như phát triển các thuật toán phức tạp hơn sẽ giúp hệ thống trở nên hiệu quả hơn và đáp ứng tốt hơn nhu cầu của nhà đầu tư. Những hướng phát triển này sẽ không chỉ nâng cao độ chính xác và khả năng dự báo mà còn tối ưu hóa lợi nhuận đầu tư, góp phần nâng cao khả năng cạnh tranh trên thị trường tài chính đầy biến động.

5.2 Hướng phát triển

Mặc dù đề tài đã đạt được thành công nhất định khi có thể đem lại lợi nhuận cho nhà đầu tư tuy nhiên hệ thống vẫn cần phải có những cải thiện đáng kể để đam lại hiệu quả tốt nhất cả về hệ thống lưu trữ xử lý dữ liệu lớn và cả về thuật toán để giúp tối ưu lợi nhuận đạt được cho nhà đầu tư.

Hướng phát triển đầu tiên sẽ tập trung vào việc mở rộng nguồn dữ liệu và phát triển các thuật toán qua đó giúp tối ưu lợi nhuận thu được. Nguồn dữ liệu sẽ không tập chỉ tập trung vào giá lịch sử của các mã cổ phiếu mà sẽ được mở rộng ra các dữ liệu thị trường như báo cáo tài chính, tin tức thị trường,... Thêm vào đó đề tài có thể áp dụng các mô hình học máy, học sâu kết hợp với thuật toán tối ưu để đưa ra danh mục tối ưu nhất có thể. Các mô hình Machine Learning, Deep Learning cũng có thể được áp dụng để dự báo giá bán tương lai của các mã cổ phiếu từ đó đưa ra các tín hiệu giao dịch hiệu quả hơn.

Việc mở rộng nguồn dữ liệu đầu vào kéo theo nhu cầu phải mở rộng và phát triển hệ thống lưu trữ và xử lý dữ liệu. Khi dữ liệu được thu thập từ nhiều nguồn khác

nhau, việc lưu trữ một cách hiệu quả và tái sử dụng dữ liệu này trở nên vô cùng quan trọng. Các hệ thống lưu trữ phân tán như HDFS, Apache HBase, Amazon DynamoDB,... có thể được sử dụng để đảm bảo rằng dữ liệu được lưu trữ an toàn, có thể mở rộng và dễ dàng truy cập cho các phân tích sau này. Điều này không chỉ giúp tối ưu hóa việc lưu trữ mà còn đảm bảo tính sẵn sàng và khả năng mở rộng của hệ thống dữ liệu khi quy mô dữ liệu ngày càng tăng.

Hơn nữa, việc tích hợp các công cụ và hệ thống xử lý dữ liệu lớn khác như Apache Kafka và Apache HBase sẽ giúp nâng cao khả năng xử lý và phân tích dữ liệu của hệ thống. Apache Kafka có thể xử lý các luồng dữ liệu lớn theo thời gian thực, trong khi Apache HBase cung cấp khả năng lưu trữ và truy xuất dữ liệu nhanh chóng. Việc kết hợp các công cụ này sẽ không chỉ giúp hệ thống xử lý và phân tích dữ liệu một cách hiệu quả hơn mà còn đảm bảo rằng dữ liệu luôn được cập nhật và sẵn sàng cho các quá trình ra quyết định quan trọng của nhà đầu tư.

Tổng kết lại, các hướng phát triển cụ thể của hệ thống sẽ tập trung vào việc mở rộng và cải thiện hệ thống lưu trữ, đồng thời nâng cao hiệu suất phân tích và xử lý dữ liệu lớn. Việc phát triển hệ thống sẽ tập trung vào việc sử dụng các hệ thống lưu trữ phân tán để đảm bảo dữ liệu được lưu trữ một cách hiệu quả và có thể dễ dàng mở rộng, đồng thời cải tiến các công cụ xử lý dữ liệu lớn để tăng cường khả năng xử lý dữ liệu theo thời gian thực. Ngoài ra, áp dụng các cải tiến về thuật toán và sử dụng các mô hình Machine Learning và Deep Learning sẽ giúp tối ưu hóa lợi nhuận đầu tư. Các mô hình học máy có khả năng học hỏi từ dữ liệu lịch sử và dự đoán các xu hướng trong tương lai, trong khi các mô hình học sâu có thể xử lý các tập dữ liệu phức tạp và phi cấu trúc. Việc áp dụng những công nghệ tiên tiến này sẽ giúp hệ thống không chỉ tối ưu hóa hiệu suất giao dịch mà còn nâng cao khả năng dự báo và ra quyết định, tạo lợi thế cạnh tranh trong thị trường tài chính đầy biến động.

TÀI LIỆU THAM KHẢO

[1] S. Tiao, *What is big data?* March 11, 2024. [Online]. Available: https://www.oracle.com/big-data/what-is-big-data/.

[2] R. SHARMA, *What is quantitative trading*, April 20, 2024. [Online]. Available: https://www.investopedia.com/terms/q/quantitative-trading.asp.

[3] *Apache spark*. [Online]. Available: https://spark.apache.org/docs/3.5.1/index.html.

[4] *Pyspark overview*. [Online]. Available: https://spark.apache.org/docs/latest/api/python/index.html.

[5] William F. Sharpe, Stanford University, *The sharpe ratio*. [Online]. Available: https://web.stanford.edu/%7Ewfsharpe/art/sr/SR.htm.

[6] J. FERNANDO, *Relative strength index (rsi) indicator explained with formula*, April 10, 2024. [Online]. Available: https://www.investopedia.com/terms/r/rsi.asp.