TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

Trần Minh Trí - Nguyễn Nhật Trường

Nghiên cứu cải tiến dự đoán giá chứng khoán dựa vào độ tương đồng trên chuỗi thời gian

KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN
CHƯƠNG TRÌNH CHÍNH QUY

Tp. Hồ Chí Minh, tháng 06/2021

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

Trần Minh Trí - 1712834
Nguyễn Nhật Trường - 1712852

Nghiên cứu cải tiến dự đoán giá chứng khoán dựa vào độ tương đồng trên chuỗi thời gian

KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN
CHƯƠNG TRÌNH CHÍNH QUY

GIÁO VIÊN HƯỚNG DẪN
Ths. Trần Văn Quý

Tp. Hồ Chí Minh, tháng 06/2021

Lời cảm ơn

Chúng em xin chân thành cảm ơn quý Thầy, Cô đã nhiệt tình truyền tải kiến thức và chia sẻ các kinh nghiệm quý báu cho chúng em trong suốt 4 năm qua, để hiện tại chúng em có được nền tảng kiến thức vững chắc để có thể thực hiện hoàn thành bài khóa luận tốt nghiệp này.

Chúng em xin gửi lời cảm ơn chân thành đến thầy Trần Văn Quý, người đã tận tình hướng dẫn và hỗ trợ nhóm trong suốt quá trình thực hiện khóa luận này. Nhờ thầy tạo điều kiện mà tụi em mới có thể tiếp xúc và tiếp thu được một phương pháp, kiến thức mới. Mặc dù công việc giảng dạy bận rộn và nhiều khó khăn mùa dịch trong suốt thời gian làm khóa luận, thầy vẫn không ngần ngại chỉ dẫn, chỉnh sửa, uốn nắn con đường thực hiện khóa luận của chúng em cho đạt được kết quả tốt nhất. Khóa luận này chắc chắn không thể hoàn thành nếu không có sự hướng dẫn tận tâm của thầy.

Cuối cùng, chúng em xin bày tỏ lòng biết ơn đến gia đình, người thân và bạn bè đã luôn hỗ trợ, giúp đỡ, động viên và tạo điều kiện cho chúng em thực hiện khóa luận tốt nghiệp này.

Tuy đã cố gắng thực hiện hết sức mình trong phạm vi, kiến thức và khả năng cho phép nhưng các thiếu sót, sai lầm là điều không thể tránh khỏi. Chúng em mong nhận được các góp ý cho bài khóa luận này.

Chúng em xin chân thành cảm ơn,

TP Hồ Chí Minh, ngày 11 tháng 06 năm 2021

Trần Minh Trí, Nguyễn Nhật Trường

Mục lục

Danh sách hình

Danh sách bảng

Tóm tắt

Dự đoán xu hướng giá chứng khoán sử dụng máy học là một chủ đề nóng hiện nay, với các thuật toán dự đoán dựa trên lịch sử của một loại cổ phiếu nào đó. Một nghiên cứu gần đây [1] chỉ ra được sử dụng dữ liệu nhiều loại cổ phiếu có tương đồng cao so với cổ phiếu cần dự đoán để tăng cường tập huấn luyện có thể cho ra kết quả dự đoán tốt hơn so với chỉ áp dụng dữ liệu của duy nhất một cổ phiếu. Thấy được đây là một nghiên cứu mới có thể bổ sung, cải thiện, nên trong bài khóa luận này, lấy nghiên cứu trên làm cơ sở, nhóm em đã xây dựng một quy trình xử lí và áp dụng các loại cổ phiếu tương đồng trong huấn luyện đã được cải tiến và sửa nhiều lỗi mắc phải từ nghiên cứu cơ sở. Quy trình này được áp dụng lên một tập dữ liệu (các cổ phiếu) cơ sở lấy từ nghiên cứu trước đó để có thể so sánh hiệu suất giữa quy trình này và kết quả của nghiên cứu cơ sở, và một tập dữ liệu khác do nhóm tự thu thập để tiếp tục chứng minh rằng phương pháp này mang lại kết quả cao hơn so với cách làm truyền thống. Kết quả thu được ở cả hai tập dữ liệu, việc áp dụng tương đồng vào huấn luyện dự đoán máy học mang lại lợi nhuận cao hơn ở mọi mô hình sử dụng, cụ thể lợi nhuận tốt nhất giữa không và áp dụng phương pháp này qua từng tập dữ liệu là 67.45 - 85.9 và 130 - 211.9. Và kết quả so sánh trên tập dữ liệu cơ sở, tất cả mô hình qua quy trình của nhóm đều đạt độ chính xác cao hơn 0.55 của kết quả tốt nhất cơ sở, với độ chính xác cao nhất đạt được 0.5979; điểm F1 của 5/7 mô hình qua quy trình của nhóm cao hơn 0.495 tốt nhất của kết quả cơ sở, với điểm F1 cao nhất đạt 0.5593.

Chương 1

Giới thiệu tổng quan về đề tài

1.1 Lý do nghiên cứu

Thị trường chứng khoán hiện càng ngày càng phát triển mạnh mẽ và thu hút nhiều người đầu tư giao dịch. Tuy nhiên, nó luôn có những yếu tố bất ngờ ngoài ý muốn, có lúc tăng nhanh sau đó giảm một cách đột ngột, có lúc thì liên tục tăng mà không có dấu hiệu giảm xuống. Từ đó, nhu cầu phân tích và dự đoán giá chứng khoán được hình thành. Dự đoán về giá cổ phiếu hoặc bất kỳ vốn chủ sở hữu tài chính nào cũng đã và đang được tiếp cận bởi nhiều nhà nghiên cứu khoa học, cũng như các quỹ đầu tư lớn trong giới, với nhiều công trình phân tích trải dài qua nhiều năm.

Tuy nhiên, hầu hết các mô hình dự đoán hiện nay chỉ đào tạo trên dữ liệu thu thập được của một cổ phiếu cụ thể. Thị trường chứng khoán thì thường có một xu hướng chung. Người giao dịch không đưa ra quyết định mua/bán chỉ dựa vào một loại cổ phiếu bất kỳ mà dựa vào biến động của nhiều loại cổ phiếu, chỉ số index sàn giao dịch. Một nghiên cứu mới gần đây thực hiện bởi Lior Sidi cho thấy có thể cải tiến mô hình dự đoán giá chứng khoán khi áp dụng những cổ phiếu tương đồng vào dữ liệu [1] so với các mô hình hiện tại. Cho thấy ta có thể áp dụng nhiều hơn một loại cổ

phiếu vào dự đoán, đào tạo được mô hình máy học mô phỏng quyết định của một người giao dịch ngoài đời thực.

Trong bài khóa luận này, nhóm lựa chọn đề tài "Nghiên cứu cải tiến dự đoán giá chứng khoán dựa vào độ tương đồng trên chuỗi thời gian", áp dụng các kỹ thuật tính độ tương đồng giữa các chuỗi thời gian để kết hợp dữ liệu nhiều cổ phiếu khác có tính tương đồng với cổ phiếu đang dự đoán trong huấn luyện, mong muốn mô phỏng được quá trình đưa ra quyết định dựa vào xu hướng của sàn giao dịch với nhiều loại cổ phiếu, và từ đó đưa ra được dự đoán mang về lợi nhuận cao hơn.

1.2 Mục tiêu nghiên cứu

Mục tiêu nghiên cứu chính mà nhóm nhắm đến là dựa vào nghiên cứu hiện có [1], áp dụng được các kỹ thuật tương đồng vào huấn luyện kèm theo các chỉnh sửa, khắc phục các hạn chế của nghiên cứu trước này để đưa ra kết quả cải tiến tốt hơn.

1.3 Cách tiếp cận

Nhóm xây dựng một quy trình thử nghiệm các thuật toán tương đồng và các phép cân bằng độ dài chuỗi thời gian để áp dụng số lượng k cổ phiếu có độ tương đồng cao nhất với cổ phiếu đang xét. Cùng với đó sử dụng các nhóm thuộc tính, phương pháp giảm chiều, độ dài cửa sổ trượt và mô hình khác nhau. Các kết quả được chuyển về dạng phân lớp và áp dụng hai chiến lược giao dịch đơn giản để có được lợi nhuận ước tính. Quy trình này áp dụng từng tổ hợp các cấu hình trên 5 đoạn (folds) và các số đo được tính trung bình để xác nhận chéo (cross validation) giữa các thí nghiệm.

Đề tài sử dụng hai tập dữ liệu, một tập dữ liệu S&P 500, bao gồm dữ liệu qua từng ngày trong 5 năm của 500 cổ phiếu trong chỉ số này, được lấy từ nguồn của nghiên cứu [1] sẵn có thể so sánh kết quả của việc cải

tiến với các kết quả báo cáo trong nghiên cứu này. Tập dữ liệu còn lại là dữ liệu S&P 500 qua từng giờ, nhóm thu thập sử dụng yahoo finance api, trong khoản thời gian từ ngày 06/04/2020 đến 07/04/2021. Tập dữ liệu thứ hai này dùng để chứng minh phương pháp áp dụng tương đồng đã qua cải tiến của nhóm có thể áp dụng với các tập dữ liệu có khung thời gian khác nhau.

1.4 Đóng góp

Nhóm hi vọng có thể cải tiến, tinh chỉnh hợp lý việc áp dụng nhiều dữ liệu của các loại cổ phiếu liên quan trong huấn luyện dự đoán giá một loại cổ phiếu và thực nghiệm phương pháp của nhóm trên tập dữ liệu với khung thời gian khác. Từ đó cho thấy được sự đáng tin cậy của phương pháp này, mở ra một cách xử lí dữ liệu mới trong dự đoán giá chứng khoán bằng máy học.

Chương 2

Tổng quan

2.1 Tổng quan các nội dung về lĩnh vực chứng khoán

2.1.1 Khái niệm chứng khoán

Theo Wikipedia [2], chứng khoán (securities) là một tài sản tài chính có thể giao dịch. Thuật ngữ "chứng khoán" thường được sử dụng theo cách nói hàng ngày để chỉ bất kỳ hình thức công cụ tài chính nào, mặc dù chế độ pháp lý và quy định cơ bản có thể không có định nghĩa rộng như vậy.

Định nghĩa pháp lý của chứng khoán khác nhau tùy theo thẩm quyền của từng quốc gia. Tại Việt Nam, Điều 4 Luật Chứng khoán 2019 quy định:

Chứng khoán là tài sản, bao gồm các loại sau đây:
a) Cổ phiếu, trái phiếu, chứng chỉ quỹ;
b) Chứng quyền, chứng quyền có bảo đảm, quyền mua cổ phần, chứng chỉ lưu ký;
c) Chứng khoán phái sinh;
d) Các loại chứng khoán khác do Chính phủ quy định.

Trong đó, cũng theo Điều 4, các loại tài sản là chứng khoán được quy

định như sau:

- Cổ phiếu (stock) là loại chứng khoán xác nhận quyền và lợi ích hợp pháp của người sở hữu đối với một phần **vốn** *cổ phần* (share) của tổ chức phát hành.

- Trái phiếu (bond) là loại chứng khoán xác nhận quyền và lợi ích hợp pháp của người sở hữu đối với một phần **nợ** của tổ chức phát hành.

- Chứng chỉ quỹ (fund certificate) là loại chứng khoán xác nhận quyền sở hữu của nhà đầu tư đối với một phần vốn góp của quỹ đầu tư chứng khoán.

- Chứng quyền (warrant) là loại chứng khoán được phát hành cùng với việc phát hành trái phiếu hoặc cổ phiếu ưu đãi, cho phép người sở hữu chứng quyền được quyền mua một số cổ phiếu phổ thông nhất định theo mức giá đã được xác định trước trong khoảng thời gian xác định.

- Chứng quyền có bảo đảm (covered warrant, viết tắt: CW) là loại chứng khoán có tài sản bảo đảm do công ty chứng khoán phát hành, cho phép người sở hữu được quyền mua (chứng quyền mua) hoặc được quyền bán (chứng quyền bán) chứng khoán cơ sở với tổ chức phát hành chứng quyền có bảo đảm đó theo mức giá đã được xác định trước, tại một thời điểm hoặc trước một thời điểm đã được ấn định hoặc nhận khoản tiền chênh lệch giữa giá thực hiện và giá chứng khoán cơ sở tại thời điểm thực hiện.

- Quyền mua cổ phần (share purchase right) là loại chứng khoán do công ty cổ phần phát hành nhằm mang lại cho *cổ đông* (shareholder) hiện hữu quyền được mua cổ phần mới theo điều kiện đã được xác định.

- Chứng chỉ lưu ký (depositary receipt; viết tắt: DR) là loại chứng

khoán được phát hành trên cơ sở chứng khoán của tổ chức được thành lập và hoạt động hợp pháp tại Việt Nam.

- Chứng khoán phái sinh (derivative) là công cụ tài chính dưới dạng hợp đồng, bao gồm hợp đồng quyền chọn, hợp đồng tương lai, hợp đồng kỳ hạn, trong đó xác nhận quyền, nghĩa vụ của các bên đối với việc thanh toán tiền, chuyển giao số lượng tài sản cơ sở nhất định theo mức giá đã được xác định trong khoảng thời gian hoặc vào ngày đã xác định trong tương lai.

2.1.2 Trị trường chứng khoán

Khái niệm

Thị trường chứng khoán (securities market) là nơi diễn ra các hoạt động phát hành, trao đổi các loại chứng khoán, tập hợp gồm những người mua và người bán. Hầu hết các giao dịch chứng khoán đều thực hiện thông qua các *nhà môi giới chứng khoán* (stock broker) và nền tảng giao dịch điện tử.

Thị trường chứng khoán, cơ bản, trao đổi hai loại chứng khoán chính là cổ phiếu và trái phiếu, và có thể lần lượt được gọi là *thị trường cổ phiếu* (stock market) và *thị trường trái phiếu* (bond market). Tuy nhiên, trong thực tế không tồn tại sự phân biệt rõ rệt giữa hai thị trường này, do cả hai loại chứng khoán này, cũng như các loại khác, đều được các *sở giao dịch chứng khoán* (stock exchange) và các công ty chứng khoán tổ chức trao đổi.

Phân loại thị trường chứng khoán

Thị trường chứng khoán có thể được chia thành hai loại chính, dựa vào tính chất của chứng khoán, là thị trường chứng khoán sơ cấp và thị trường chứng khoán thứ cấp:

- Thị trường chứng khoán sơ cấp (primary market) là thị trường mua bán các chứng khoán mới phát hành và là nền tảng ra đời và phát

triển của thị trường chứng khoán. Do bản chất của nó, thị trường chứng khoán sơ cấp không hoạt động nếu không có chứng khoán mới nào được phát hành. Vì thế, nhịp độ và lượng giao dịch tại đây thấp hơn nhiều so với thị trường thứ cấp.

- Thị trường chứng khoán thứ cấp (secondary market) là thị trường giao dịch các chứng khoán đã được phát hành trên thị trường sơ cấp. Chủ yếu, hoạt động chuyển giao quyền sở hữu chứng khoán và tiền sẽ diễn ra tại đây giữa các nhà đầu tư. Ngoài ra, lượng cổ phiếu được phát hành bởi các tổ chức sẽ được mua rất nhanh, vì những ai đến sau sẽ không kịp giao dịch và sẽ phải tìm đến thị trường thứ cấp. Do đó, thị trường thứ cấp mang tính cạnh tranh cao với khối lượng và nhịp độ giao dịch lớn và liên tục.

Đặc điểm và chức năng của thị trường chứng khoán

Đặc điểm:

- Tính cạnh tranh hoàn hảo: mọi người đều có thể tự do tham gia giao dịch, và hơn nữa, giá cả trên thị trường chứng khoán không được quy định mà hoàn toàn được hình thành dựa trên quan hệ cung - cầu.

- Tính liên tục: các chứng khoán sau khi được phát hành trên thị trường sơ cấp sẽ có thể được mua - bán liên tục qua lại nhiều lần hoặc các chứng khoán đã được sở hữu có thể được đổi thành tiền bởi các nhà đầu tư.

Chức năng:

- Hỗ trợ huy động vốn đầu tư cho nền kinh tế: thị trường chứng khoán chuyển đổi các khoản tiền cá nhân của nhà đầu tư thành chứng khoán và đưa khoản tiền này vào hoạt động sản xuất.

- Tạo ra một môi trường đầu tư đa dạng: với nhiều các loại chứng khoán trên thị trường, các nhà đầu tư sẽ có nhiều sự lựa chọn với nhiều mức độ rủi ro hay khả năng sinh lời khác nhau.

- Cung cấp khả năng thanh toán cho chứng khoán: chứng khoán có thể được quy đổi thành tiền hoặc các chứng khoán khác.

- Đánh giá tình hình kinh tế cúa một doanh nghiệp: trạng thái của doanh nghiệp có thể được nhận xét thông qua giá chứng khoán của họ.

- Đánh giá tình hình kinh tế cúa một quốc gia: tương tự như trên, giá cả chứng khoán của các doanh nghiệp trong một nước liên hệ đến nền kinh tế của nước đó.

Nguyên tắc hoạt động của thị trường chứng khoán

Thị trường chứng khoán hoạt động theo 3 nguyên tắc cơ bản: nguyên tắc trung gian, nguyên tắc đấu giá và nguyên tắc công khai. Các nguyên tắc này nhằm đảm bảo giá chứng khoán được hình thành một cách thống nhất, công bằng cho tất cả các bên giao dịch.

- Nguyên tắc trung gian: Mọi hoạt động giao dịch, mua bán chứng khoán trên thị trường chứng khoán đều được thực hiện thông qua các trung gian, hay còn gọi là các nhà môi giới. Họ sẽ hưởng hoa hồng khi thực hiện giao dịch cho khách hàng thông qua các lệnh. Ngoài ra, các nhà môi giới còn có thể cung cấp thông tin hay tư vấn đầu tư cho khách hàng.
  Nguyên tắc trung gian hạn chế hành vi trực tiếp thoả thuận giữa các nhà đầu tư với nhau, từ đó, bảo toàn tính cạnh tranh và ổn định của thị trường chứng khoán.

- Nguyên tắc đấu giá: Giá chứng khoán được xác định thông qua việc đấu giá giữa các lệnh mua và các lệnh bán. Việc xác định giá là hoàn

toàn không thể bị can thiệp bởi bất cứ ai trong thị trường. Có hai hình thức đấu giá là đấu giá trực tiếp và đấu giá tự động:

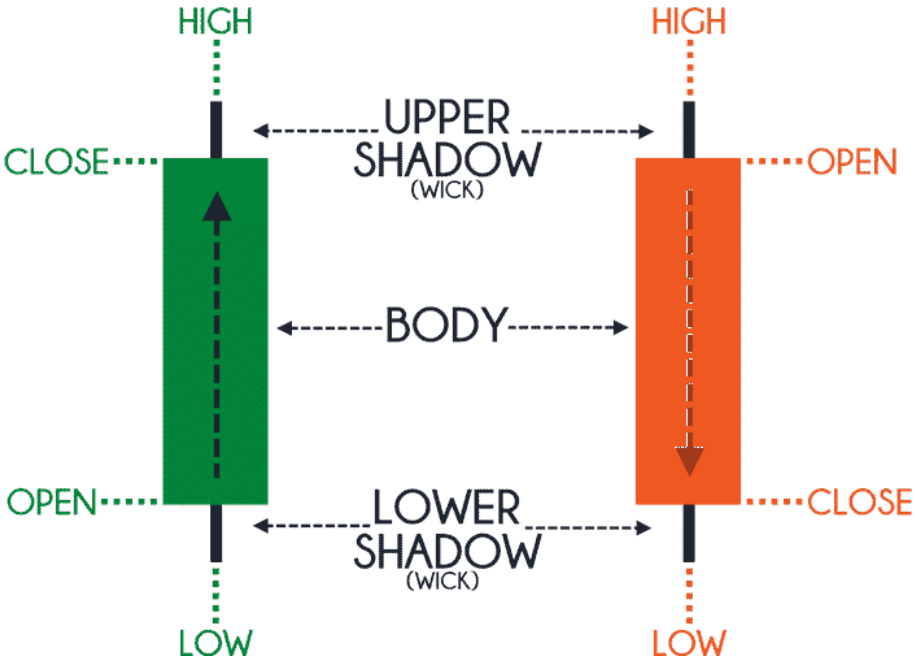
  - Đấu giá trực tiếp là việc các nhà môi giới gặp nhau trên sàn giao dịch và trực tiếp đấu giá.
  - Đấu giá tự động là việc các lệnh giao dịch từ các nhà môi giới được nhập vào hệ thống máy chủ của Sở giao dịch chứng khoán. Hệ thống máy chủ này sẽ xác định mức giá sao cho tại mức giá này, chứng khoán giao dịch với khối lượng cao nhất.

- Nguyên tắc công khai: Tất cả các hoạt động trên thị trường chứng khoán đều phải đảm bảo tính công khai. Các thông tin về giao dịch chứng khoán trên thị trường phải được Sở giao dịch công bố cho công chúng. Một công ty hay doanh nghiệp càng minh bạch về tình hình tài chính thì chứng khoán của họ càng thu hút nhiều nhà đầu tư.

2.1.3 Dữ liệu chứng khoán

Trong trao đổi cổ phiếu, tồn tại 4 thành phần giá mà nhà đầu tư cần lưu ý tới, chính là: giá mở cửa ("Open" prices), giá cao nhất ("High" prices), giá thấp nhất ("Low" prices) và giá đóng cửa ("Close" prices). Chúng thường được gọi chung là "Open High Low Close" (OHLC). Ngoài ra, còn có một đại lượng nữa mang tên là khối lượng giao dịch (Volume).

Trước khi định nghĩa các thành phần này, ta cần phải biết giá trị của chúng sẽ phụ thuộc vào một yếu tố vô cùng quan trọng, đó là thời gian, cụ thể là *khoảng thời gian* (period) mà chứng khoán đó được quan sát và ghi nhận diễn biến. Thông thường, period này sẽ là hằng ngày, nhưng một số nhà đầu tư sẽ kết hợp nhiều period khác nhau để nhận xét hành vi của giá chứng khoán.



Hình 2.1: Minh họa 4 thành phần giá OHLC trên một period. [3]

Ta có các định nghĩa sau:

- Open: giá của cổ phiếu khi thị trường bắt đầu ở period đang xét.
- High và Low: giá cao nhất và thấp nhất trong period đó.
- Close: giá của cổ phiếu khi thị trường kết thúc ở period đang xét.
- Volume: tổng số cổ phần được trao đổi (cả mua và bán) trong period đó.

Biểu đồ OHLC là công cụ được sử dụng để biểu diễn giá theo thời gian, với trục thời gian được chia theo period và mỗi period sẽ biểu thị giá bằng biểu đồ hình nến (như hình 2.1), và với màu xanh hay đỏ biểu thị giá tăng hay giảm tương ứng. Biểu đồ này thường sẽ được kết hợp với các loại biểu đồ khác như biểu đồ đường (biểu diễn trung bình động) hay biểu đồ cột (biểu diễn khối lượng giao dịch. Ứng dụng này được sử dụng trong quá trình phân tích kỹ thuật sẽ được nói đến ở phần dưới đây.

2.1.4 Dự đoán chứng khoán

Những nhà đầu tư khi tham gia vào thị trường chứng khoán sẽ luôn tìm cách dự đoán chứng khoán để thu được lợi nhuận nhiều nhất có thể với mức độ hiệu quả tùy vào kinh nghiệm và kiến thức cá nhân. Hiện nay, có ba nhóm phương pháp dự đoán giá cổ phiếu chính đang được sử dụng kết hợp lẫn nhau, bao gồm:

- Phân tích cơ bản (fundamental analysis)
- Phân tích kỹ thuật (technical analysis)
- Phương pháp công nghệ (technological methods)

Phân tích cơ bản (fundamental analysis)

Phân tích cơ bản là việc phân tích cổ phiếu dựa vào các nhân tố có tính chất nền tảng có tác động hoặc dẫn tới sự thay đổi giá cả của cổ phiếu nhằm **xác định giá trị nội tại của cổ phiếu** trên thị trường. Giá trị nội tại nói trên được tạo ra bởi hoạt động của công ty và chính là cơ sở quyết định giá cổ phiếu của công ty đó. Ngoài ra, giá trị nội tại còn xác định chiều hướng thay đổi giá cổ phiếu khi được quan sát dài hạn. Do đó, các nhà đầu tư theo trường phái phân tích cơ bản sẽ tập trung theo dõi giá trị thực của một công ty với các chỉ tiêu như: lợi nhuận và tăng trưởng doanh thu; những rủi ro có thể gặp; dòng tiền... để tìm cơ hội kiếm lời từ sự chênh lệch của giá trị thị trường so với giá trị thực của công ty.

Nói tóm lại, phân tích cơ bản đi sâu vào đánh giá triển vọng tăng trưởng và lợi nhuận của công ty trên cơ sở xem xét triển vọng của nền kinh tế thế giới, kinh tế trong nước, của các ngành kinh tế và của chính bản thân công ty. Như vậy, về mặt logic, phân tích cơ bản là một quá trình phân tích các vấn đề chủ yếu, bao gồm:

- **Phân tích kinh tế vĩ mô** nhận xét các chính sách của chính phủ và GDP; tình hình lạm phát; tỷ giá hối đoái; xu hướng lãi suất;... Ngoài ra, các nhà đầu tư có thể cố gắng xác định chu kỳ kinh tế ở thời điểm hiện tại và các lĩnh vực nổi trội cần chú ý để thu lời.

- **Phân tích kinh tế ngành**, hay nói cách khác là phân tích ngành kinh tế mà công ty đang hoạt động quan sát tình hình cung - cầu của sản phẩm thuộc ngành, sức mua của nó và sức mạnh nhà cung cấp, hơn nữa, sản phẩm thay thế và các đối thủ cạnh tranh cũng cần được quan tâm đến. Ngoài ra, các nhà đầu tư cũng phải nhìn nhận về chuỗi cung ứng của ngành kinh tế trong tương lai để có thể xác định độ hấp dẫn, tiềm năng phát triển và tuổi thọ của ngành về sau.

- **Phân tích công ty** xác định các tiêu chí gồm: chuỗi cung ứng, nguồn cung ứng nguyên vật liệu, giá trị gia tăng trong chuỗi cung

ứng, công suất hoạt độn, tỷ lệ sản xuất/công suất và thời điểm hòa vốn. Các yếu tố này sẽ hỗ trợ việc nhận định tiềm năng phát triển cũng như rủi ro của công ty. Ngoài ra, các nhà đầu tư cũng cần theo dõi báo cáo tài chính của công ty và ghi nhận các chỉ số như EPS (Earning per share - thu nhập trên mỗi cổ phiếu của các cổ đông), ROA (Return on asset - lợi nhuận trên tổng tài sản xác định mức độ hiệu quả của việc sử dụng tài sản), leverage(chỉ số đòn bẩy),... thêm vào đó, là mức độ sử dụng nợ vay, các khoản nợ hiện tại và tỷ lệ thanh toán chúng.

Như vậy, phân tích kinh tế vĩ mô và phân tích ngành kinh tế có mục đích đánh giá môi trường kinh doanh và tác động của môi trường kinh doanh đến hoạt động của công ty, từ đó tác động đến giá cổ phiếu của công ty. Còn phân tích công ty nhằm đánh giá bản chất của công ty đó để rút ra các nhận xét về khả năng tăng trưởng trong hiện tại lẫn tương lai.

Ta có thể nói, thực chất phân tích cơ bản bao gồm hai nội dung chủ yếu sau: một, tìm các nguyên nhân gây ra sự biến động giá cổ phiếu dựa trên cơ sở phân tích 3 yếu tố trên; và hai, xác địch giá trị nội tại của của cổ phiếu và so sánh với giá thị trường hiện hành để đưa ra các quyết định đầu tư hợp lý.

Ưu - nhược của phân tích cơ bản: Ưu điểm:

- Phương pháp này phù hợp hơn cho việc dự đoán giá cổ phiếu và cho quyết định đầu trong dài hạn.
- Giúp cho nhà đầu tư có thể lựa chọn công ty tốt để đầu tư và nhận biết được các yếu tố chủ yếu tác động đến giá trị của công ty.

Nhược điểm:

- Sử dụng phương pháp này tốn nhiều thời gian và công sức do phải tiếp cận và xử lí một khối lượng lớn các thông tin kinh tế và tài chính.

- Mức độ chính xác của kết quả phân tích bị hạn chế, bởi lẽ nó phụ thuộc vào tính chính xác của thông tin, đặc biệt là báo cáo tài chính.

- Mặt khác, trong phân tích cơ bản có nhiều biến số phải tính đến và giá trị của các biến số này một phần mang tính chủ quan của người phân tích.

- Một hạn chế không nhỏ của phân tích cơ bản là bỏ qua yếu tố tâm lí của nhà đầu tư trên thị trường.

Phân tích kỹ thuật (technical analysis)

Phân tích kỹ thuật là dự đoán vận động giá cả tương lai dựa trên khảo sát giá trong quá khứ. Khác với phân tích cơ bản xem xét nhiều yếu tố xung quanh giá của một tài sản, phân tích kỹ thuật chỉ tập trung xem xét diễn biến giá cả trong lịch sử. Hai tham số quan trọng được khảo sát trong phân tích kỹ thuật là giá cả và khối lượng giao dịch, nghiên cứu những diễn biến trong lịch sử để đưa ra những dự báo cho tương lai.

Phân tích kỹ thuật hoạt động dựa trên các nguyên tắc sau:

- Hành động thị trường phản ánh tất cả diễn biến: do toàn bộ thông tin liên quan đã được phản ánh bởi giá cả, các nhà phân tích kỹ thuật tin rằng việc phải hiểu suy nghĩ và cảm nhận của các nhà đầu tư về thông tin đó là vô cùng quan trọng.

- Giá cả di chuyển theo xu hướng: các nhà phân tích kỹ thuật tin rằng xu hướng của giá cả mang tính trực tiếp. Tức là, giá cả chỉ có thể thay đổi theo các chiều lên, xuống, hoặc ngang (không đổi) hay kết hợp.

- Lịch sử sẽ tự lặp lại: các nhà phân tích kỹ thuật tin rằng các nhà đầu tư mới sẽ tuần tự theo lại đường đi cũ của các nhà đầu tư trước. Hành vi này được cấu thành do cảm xúc, dù các cảm xúc này không hợp lý trong quá trình đầu tư, nhưng chúng hoàn toàn tồn tại. Do

đó, dựa trên tính lặp này, các nhà phân tích kỹ thuật tin rằng các *pattern* giá có thể được nhận dạng (và có thể được dự đoán) và áp dụng chúng để đưa ra các quyết định giao dịch mang xác suất thành công cao hơn.

Thông thường, các nhà phân tích thường sử dụng các biểu đồ gồm nhiều loại *chỉ bảo kỹ thuật* (technical indicator) khác nhau để phục vụ việc xác định các xu hướng của thị trường. Dưới đây là một số chỉ báo kỹ thuật phổ biến:

- **Simple Moving Average (SMA)**, với Moving Average (MA) được dịch là trung bình động, là một trong những chỉ báo được sử dụng và biết đến nhiều nhất, một phần do sự đơn giản của nó. SMA sẽ thực hiện tính trung bình của giá (thường là giá đóng cửa) trong một khoảng thời gian nào đó. Ví dụ, nếu ta có dữ liệu giá đóng cửa của từng ngày trong một tháng thì SMA_{15} sẽ là trung bình giá của 15 ngày cuối, SMA_{10} sẽ là trung bình giá của 10 ngày cuối và tương tự.

- **Exponential Moving Average (EMA)** là một phiên bản sửa đổi từ SMA, với hướng coi trọng giá gần đây hơn bằng cách gán cho chúng một "cân nặng" (weight) cao hơn so với các giá cũ.

- **Moving Average Convergence Divergence (MACD)**, với tên gọi mang hai từ "hội tụ" và "phân kỳ" ("convergence" và "divergence"), là một chỉ số dùng để nhận xét mối quan hệ giữa hai MA của một giá chứng khoán. Công thức của MACD là hiệu của EMA_{12} trừ cho EMA_{26}.

- **Relative Strength Index (RSI)** thuộc loại *chỉ báo dao động* (momentum indicator) với mục đích đo mức độ thay đổi gần đây của giá. RSI sẽ có giá trị từ 0 đến 100, và trong thực tế sử dụng, RSI có giá trị trên 70 có nghĩa là một chứng khoán đang bị mua vượt mức hay định giá vượt mức và có thể sẽ tiếp diễn theo xu hướng ngược lại với

hiện tại, còn RSI với giá trị dưới 30 có nghĩa là chứng khoán đó đang bị mua dưới mức hay định giá dưới mức.

- **Bollinger Bands (BB)** là một chỉ báo dao động khác cũng khá phổ biến, gồm hai *dải* (band) nằm ở hai phía (trên và dưới) của MA. Chỉ báo BB chủ yếu được sử dụng để phát hiện tình trạng mua và bán quá mức của thị trường.

- **Rate of Change (ROC)** là một chỉ báo dao động được dùng để tính phần trăm thay đổi của giá hiện tại với giá trước cách một khoảng thời gian nào đó. Chỉ báo ROC được trực quan hóa so với trục hoành, với giá có xu hướng tăng nếu ROC nằm ở phần dương (trên trục hoành), và ngược lại.

Ưu - nhược của phân tích kỹ thuật: Ưu điểm:

- Phân tích kỹ thuật có thể xác định thời điểm thích hợp để mua giữ hoặc bán cổ phiếu. Trong thực tế, một số nhà phân tích sử dụng kết hợp cả hai phương thức phân tích cơ bản và phân tích kỹ thuật để quyết định mua cái gì và thời điểm mua nó, cũng như bán.

- Giá tương lai có thể được dự đoán dựa trên dòng diễn biến giá cả.

- Phân tích kỹ thuật áp dụng việc sử dụng các biểu đồ biểu diễn giá chứng khoán và các chỉ báo khiến quá trình phân tích tường minh và chi tiết hơn.

Nhược điểm:

- Các chỉ báo hay mô hình kỹ thuật không bảo đảm độ chính xác tuyệt đối.

- Phân tích kỹ thuật mang tính chủ quan cao, tức là, dù đã có các tiêu chuẩn quy định sẵn, với cùng một biểu đồ, nhưng hai nhà phân tích có thể sẽ đưa ra được hai nhận xét hoàn toàn khác nhau, với lập luận và minh chứng riêng của họ.

Tóm lại, cả hai phương pháp truyền thống phân tích cơ bản và phân tích kỹ thuật đều có những ưu và nhược điểm riêng nên ta hoàn toàn không thể nói phương pháp nào là tối ưu hơn. Mức độ hiệu quả của từng phương pháp sẽ phụ thuộc vào khả năng, trình độ và quan trọng nhất, là mục tiêu của mỗi nhà đầu tư. Hơn nữa, như đã nói ở trên, chiến lược kết hợp cả phân tích cơ bản và phân tích kỹ thuật sẽ hỗ trợ đưa ra các quyết định đầu tư mang góc nhìn đa chiều hơn, dựa theo ứng dụng riêng của mỗi phương pháp: phân tích cơ bản là nền tảng nhà đầu tư phải có khi mua hoặc bán bất kỳ loại cổ phiếu nào và phân tích kỹ thuật giúp nhà đầu tư xác định thời điểm mua vào hay bán ra tốt nhất để đem lại lợi nhuận tối đa.

Phương pháp công nghệ (technological methods)

Ngày nay, khoa học - công nghệ hiện đại đã trải qua rất nhiều bước tiến lớn để phục vụ cuộc sống con người tốt hơn. Chính vì thế, công nghệ càng ngày càng được áp dụng vào nhiều lĩnh vực khác nhau và một trong số đó là dự đoán thị trường chứng khoán.

Ứng dụng *Artificial Neural Networks* (ANNs, tạm dịch là mạng thần kinh nhân tạo) là một trong những hướng nghiên cứu có sức ảnh hưởng lớn nhất đến phương pháp này. Cơ chế chính của ANNs là một xấp xỉ hàm tùy ý được xây dựng, hay chính xác hơn là "tự học" được từ dữ liệu quan sát. Thường có hai cách tiếp cận khi sử dụng ANNs là mạng độc lập và mạng chung. Cách tiếp cận độc lập sử dụng một ANN duy nhất cho mỗi mốc thời gian (ví dụ: 1, 2 hôặc 5 ngày), còn cách tiếp cận chung sẽ kết hợp nhiều mốc thời gian với nhau để chúng được xác định đồng thời. Tuy nhiên, ở cách tiếp cận chung, lỗi dự báo mạng cho một mốc có thể làm ảnh hưởng đến các mốc khác, chứ không độc lập như cách kia, dẫn đến làm giảm hiệu suất. Trong thực tế, phần lớn các nghiên cứu học thuật sẽ chuộng ANN độc lập hơn để dự báo cổ phiếu.

Ngoài ra, một phương pháp phổ biến khác là phương pháp chuỗi thời gian (time-series). Trong toán học, chuỗi thời gian là một "chuỗi" các điểm dữ liệu đã có *chỉ số* (indexes), hoặc đã thuộc một danh sách hay đồ thị,

theo thứ tự thời gian. Tuy nhiên, thông thường, chuỗi thời gia có thể được định nghĩa là một dãy giá trị trong các thời điểm liên tiếp và cách đều nhau. Ta có thể nói, chuỗi thời gian là một dãy gồm các dữ liệu với thời gian *rời rạc* (discrete).

Phương pháp chuỗi thời gian có hai dạng chính:

- Phân tích chuỗi thời gian (Time series analysis) là các phương pháp khai thác các thông tin ý nghĩa của dữ liệu trong lĩnh vực học thống kê.

- Dự báo chuỗi thời gian (Time series forecasting) là việc sử dụng các mô hình máy học để dự đoán các giá trị trong tương lai dựa trên dữ liệu đã có.

2.2 Tổng quan về bài toán dự đoán trên chuỗi thời gian

2.2.1 Dữ liệu cổ phiếu dạng chuỗi thời gian

Dữ liệu tài chính tiêu chuẩn thường bao gồm một tập hợp các dữ liệu giá cổ phiếu trong một khoảng thời gian nhất định. Tập hợp này thường là bộ giá Open High Low Close (OHLC) và Volume, như đã đề cập ở trên. Ngoài ra, một số bài nghiên cứu cũng áp dụng và tính toán một sô các chỉ bảo kỹ thuật quan trọng như RSI, MACD và ROC, ngoài ra, còn có một độ đo khác là Sharpe Ratio. Sharpe Ratio được dùng để tính mức rủi ro trong một khoảng thời gian nhất định bằng cách chia kỳ vọng cho độ lệch chuẩn của lợi nhuận.

Trong quá trình chuẩn bị dữ liệu, một khía cạnh cần quan tâm đến là giai đoạn cần được dự đoán. Các khoảng ngắn như một ngày, một tháng hay một năm sẽ phù hợp hơn khi quá trình dự đoán áp dụng chỉ báo kỹ thuật.

2.2.2 Giảm chiều dữ liệu

Giảm chiều dữ liệu chuỗi thời gian là một bước vô cùng quan trọng khi làm việc với dữ liệu cổ phiếu dạng chuỗi thời gian. Mục đích là để giảm độ phức tạp của dữ liệu và hỗ trợ việc xác định *"hình mẫu"* (pattern), *phân cụm* (clustering) và dự đoán.

2.2.3 Dự đoán giá cổ phiếu

Việc dự đoán cho dữ liệu tài chính sử dụng các mô hình máy học như Artificial Neural Networks (ANNs), Support Vector Machine (SVM),... đã được nghiên cứu thử nghiệm nhiều với kết quả mang lại rất tích cực.

Khi khảo sát các yếu tố trong quá trình dự đoán, việc sử dụng các kỹ thuật máy học, giai đoạn dự đoán hay biến đầu vào khác nhau sẽ ảnh hưởng lớn đến kết quả. Cụ thể, theo thống kê, ANNs hiện đang là kỹ thuật máy học tốt nhất hiện tại, tuy nhiên, các mô hình đơn giản hơn như cây quyết định, hồi quy logistic, k-nearest neighbors hay Naive Bayes cũng mang lại kết quả dự đoán không kém. Do đó, sử dụng các thuật toán đơn giản là một cách phù hợp để *đối chuẩn* (benchmark) các phương pháp chuẩn bị hay làm giàu dữ liệu.

2.2.4 Đánh giá kết quả dự đoán

Việc đánh giá một mô hình dự đoán cổ phiếu trên chuỗi thời gian sẽ sử dụng hai loại số liệu: một là các độ đo lỗi hay độ chính xác thông thường của mô hình dự đoán như mean absolute error (MAE), mean absolute percentage error (MAPE), và root mean square error (RMSE); hai là lợi nhuận, cụ thể sẽ được tính khi áp dụng một chiến lược trao đổi nào đó.

Một trong những chiến lược phổ biến nhất là *chiến lược mua và giữ* (Buy & Hold strategy). Chiến lược này rất đơn giản, với nguyên tắc là mua khi giá cổ phiếu đó được đoán là sẽ tăng và bán trong trường hợp giảm. Tuy nhiên, bất kỳ chiến lược nào cũng áp dụng một cơ chế kiểm

soát rủi ro, chẳng hạn như *cắt lỗ* (stop loss).

2.3 Tổng quan về bài toán tương đồng trên chuỗi thời gian

Việc phân cụm cho dữ liệu cổ phiếu dạng chuỗi thời gian sẽ hỗ trợ quá trình khám phá hình mẫu, xác định các công ty tương tự nhau, thực hiện dự đoán và cũng như là đề xuất. Một mô hình phân cụm cần ba yếu tố chính: thuật toán phân cụm, hàm tính *độ tương đồng* (similarity) và phương pháp đánh giá kết quả. Cần lưu ý, bước phân cụm và tính độ tương đồng bị ảnh hưởng rất lớn bởi bước xử lý và giảm chiều dữ liệu.

Ở phần dưới đây, ta sẽ đi tìm hiểu hai loại độ tương đồng khác nhau.

2.3.1 Khoảng cách bằng số

Một trong những cách tính độ tương đồng cơ bản để phục vụ cho việc phân cụm chuỗi thời gian và phân loại các công ty tương tự nhau là áp dụng khoảng cách Euclid lên dữ liệu cổ phiếu. Tuy nhiên, một điểm yếu của khoảng cách Euclid là nó không thể nhận dạng được xu hướng của dữ liệu trên dòng thời gian. Thuật toán Dynamic Time Warping (DTW) khắc phục được vấn đề này với khả năng xử lý các chuỗi không đồng đều về độ dài và giải quyết các chênh lệch trong chuỗi thời gian để thực hiện tìm các "shape" tương tự giữa hai chuỗi. Ngoài ra, còn có một cách tính khác là sử dụng hệ số tương quan Pearson (Pearson correlation coefficient hay PCC). Một hạn chế của PCC là nó không phù hợp cho dữ liệu chuỗi thời gian không hài hòa, đều và khác độ dài.

2.3.2 Khoảng cách tượng trưng

Độ tương đồng thay vì được xem là một con số hay khoảng cách bằng số như trên, mà ở đây, nó sẽ được nhận xét qua các biểu tượng, ví dụ: UP

(tăng), DOWN (giảm) và SAME (không thay đổi) với "khoảng cách tượng trưng" tương ứng sẽ là tổng số biểu tượng trùng khớp.

Symbolic ApproXimation Aggregation (SAX) là một phương pháp giảm chiều bằng cách chuyển đổi dữ liệu chuỗi thời gian bằng một bộ biểu tượng (hay ký tự) định sẵn. Các chuỗi thời gian sau khi đã được áp dụng SAX sẽ được tính độ tương đồng hay trong trường hợp này, được gọi là khoảng cách SAX, bằng hai thuật toán chính là SAX MINDIST và SAX APXDIST. Trong đó, **SAX MINDIST** sẽ tính khoảng cách SAX dựa vào khoảng cách giữa các biểu tượng. Một khía cạnh nhỏ của MINDIST là nó sẽ xem các biểu tượng liền kề nhau có khoảng cách là 0. **SAX APXDIST** khắc phục điều này và sẽ cân nhắc gía trị khoảng cách lớn nhất và nhỏ nhất của toàn bộ các biểu tượng để tính khoảng cách SAX.

2.4 Công trình liên quan

Đề tài khóa luận này lấy cảm hứng từ bài báo khoa học 'Improving S&P stock prediction with time series stock similarity' bởi Lior Sidi [1]. Trong nghiên cứu này, tác giả chứng minh được nếu áp dụng thêm dữ liệu các cổ phiếu có độ tương đồng cao có thể cải tiến kết quả của mô hình dự đoán máy học và đưa ra được một vài cấu hình tương đồng tốt nhất. Để có được kết quả đó tác giả thực hiện hai bước sau:

- Thí nghiệm 1: Tác giả thực hiện mô hình hóa không áp dụng tương đồng để thiết lập một cấu hình cơ sở (baseline)
- Thí nghiệm 2: Từ cấu hình cơ sở trên, tác giả áp dụng tương đồng và đưa ra cấu hình tương đồng có kết quả dự đoán cho lợi nhuận cao nhất.

Tuy từ những thực nghiệm trên tác giả đưa ra được mô hình cho kết quả tốt hơn so với mô hình không áp dụng tương đồng, nhóm tìm hiểu và phát hiện nhiều hạn chế có thể cải tiến như sau:

- Tuy nhiên, ở thí nghiệm 1, tác giả lựa chọn từng tham số cho cấu hình bằng hiệu suất tổng thể. Dựa vào kết quả hình 2.2 và 2.3, tác giả lựa chọn phương pháp SAX cho việc biến đổi dữ liệu cùng giá trị dự đoán là tỉ lệ thay đổi giá (price rate of change) để làm cơ sở cho thí nghiệm 2 do chúng cho kết quả tốt trong **toàn bộ** các cấu hình. Việc dựa vào kết quả tổng thể này không hề tính đến 2 tham số riêng này tốt do các cấu hình nào và liệu 2 tham số này có ảnh hưởng xấu đến nhau hay không.

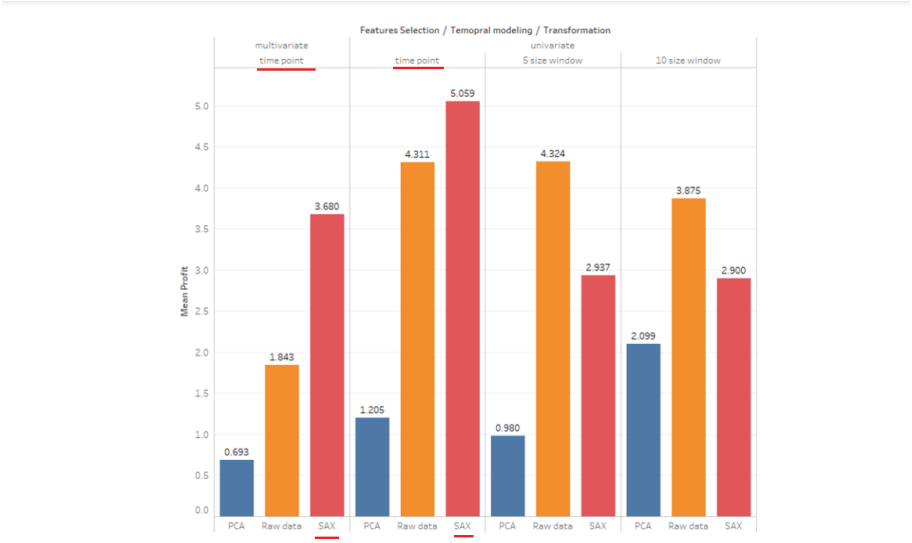
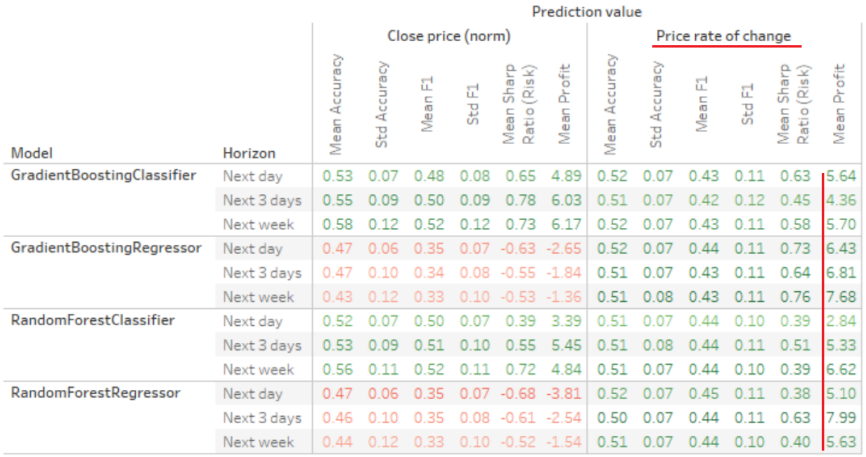


Figure 7: Experiment 1 processing parameters results - mean profit values per transformation configuration

Hình 2.2: KQ thí nghiệm 1 [1] - Mục 5.2.1 - Hình 7

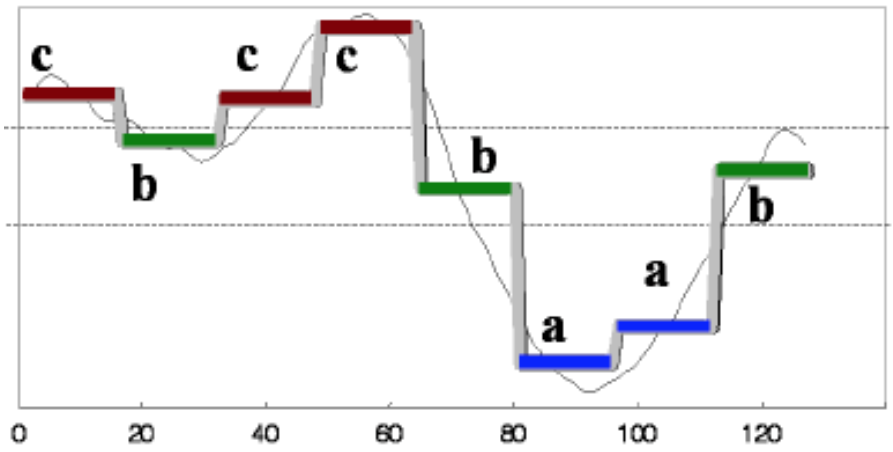


		Prediction value											
		Close price (norm)						Price rate of change					
Model	Horizon	Mean Accuracy	Std Accuracy	Mean F1	Std F1	Mean Sharp Ratio (Risk)	Mean Profit	Mean Accuracy	Std Accuracy	Mean F1	Std F1	Mean Sharp Ratio (Risk)	Mean Profit
GradientBoostingClassifier	Next day	0.53	0.07	0.48	0.08	0.65	4.89	0.52	0.07	0.43	0.11	0.63	5.64
	Next 3 days	0.55	0.09	0.50	0.09	0.78	6.03	0.51	0.07	0.42	0.12	0.45	4.36
	Next week	0.58	0.12	0.52	0.12	0.73	6.17	0.52	0.07	0.43	0.11	0.58	5.70
GradientBoostingRegressor	Next day	0.47	0.06	0.35	0.07	-0.63	-2.65	0.52	0.07	0.44	0.11	0.73	6.43
	Next 3 days	0.47	0.10	0.34	0.08	-0.55	-1.84	0.51	0.07	0.43	0.11	0.64	6.81
	Next week	0.43	0.12	0.33	0.10	-0.53	-1.36	0.51	0.08	0.43	0.11	0.76	7.68
RandomForestClassifier	Next day	0.52	0.07	0.50	0.07	0.39	3.39	0.51	0.07	0.44	0.10	0.39	2.84
	Next 3 days	0.53	0.09	0.51	0.10	0.55	5.45	0.51	0.08	0.44	0.11	0.51	5.33
	Next week	0.56	0.11	0.52	0.11	0.72	4.84	0.51	0.07	0.44	0.10	0.39	6.62
RandomForestRegressor	Next day	0.47	0.06	0.35	0.07	-0.68	-3.81	0.52	0.07	0.45	0.11	0.38	5.10
	Next 3 days	0.46	0.10	0.35	0.08	-0.61	-2.54	0.50	0.07	0.44	0.11	0.63	7.99
	Next week	0.44	0.12	0.33	0.10	-0.52	-1.54	0.51	0.07	0.44	0.10	0.40	5.63

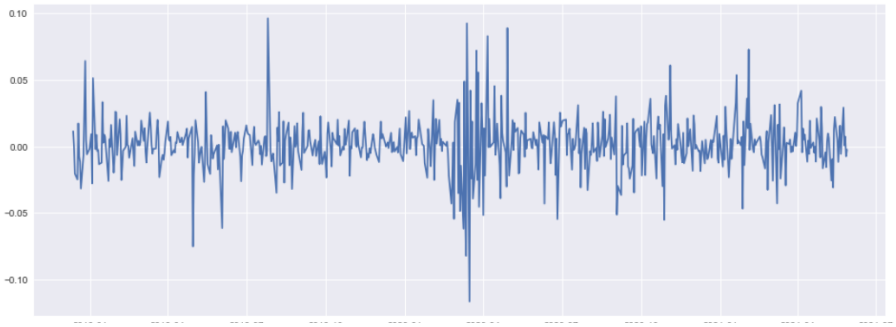
Figure 8: Experiment 1 prediction parameters results - prediction model, Horizon and Value (rows - configuration, columns - prediction value with metrics and color - profit scale)

Hình 2.3: KQ thí nghiệm 1 [1] - Mục 5.2.1 - Hình 8

Qua tìm hiểu thêm, nhóm thấy được áp dụng SAX vào chuỗi PROC gây ảnh hưởng xấu đến dữ liệu. Hình 2.4 minh họa cách SAX chuyển chuỗi thời gian về chuỗi ký tự. Hình 2.5 là minh họa biểu đồ PROC. Ta dễ thấy được do các giá trị của PROC rất nhỏ và đa số chênh lệch ít (giá thay đổi ít qua các điểm thời gian kề nhau), do đó áp dụng SAX vào chuỗi PROC sẽ cho một chuỗi ký tự có ít số ký tự, gây mất dữ liệu.



Hình 2.4: Ví dụ: SAX



Hình 2.5: Ví dụ: Biểu đồ PROC

- Tác giả không hiệu chỉnh các mô hình máy học, sử dụng giá trị mặc định (Random Forest 100 cây, Gradient Boost learning rate 0.02 [1]) nhưng cũng không hề có bất kì phương pháp xác định mô hình quá khớp/chưa khớp (over/underfit) với các cấu hình. Mô hình của tác giả có thể dự đoán xu hướng toàn tăng/toàn giảm mà không bị phát hiện.

- Tác giả hạn chế việc sử dụng nhiều thuộc tính khi áp dụng khung thời gian cho cửa sổ trượt [1]. Nhóm thấy việc này là không cần thiết và có thể ảnh hưởng đến kết quả dự đoán. Trong mô hình cửa sổ trượt nhiều ngày nếu sử dụng hợp lý nhiều hơn 1 thuộc tính cũng có thể nâng cao kết quả dự đoán.

Trong bài khóa luận này, giải quyết các hạn chế này và áp dụng các cải tiến khác để đưa ra được quy trình áp dụng độ tương đồng giữa các chuỗi thời gian hoàn thiện, tốt hơn là mục tiêu chính của nhóm.

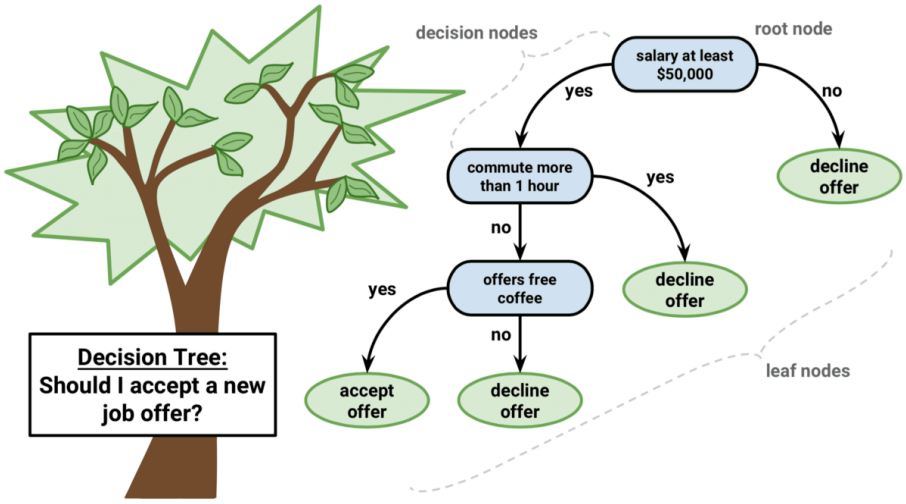
Chương 3

Cơ sở lý thuyết

3.1 Giới thiệu các mô hình máy học

3.1.1 Khái niệm: Cây quyết định

Cây quyết định (Decision Tree), là một kiểu mô hình dự báo, có thể được áp dụng vào bài toán phân lớp, hồi quy. Cây quyết định có cấu trúc hình cây, tượng trưng cho một phương pháp quyết định dựa trên những thuộc tính đã biết.



Hình 3.1: Minh họa: cây quyết định

Ở cây quyết định:

- Mỗi nút trong (internal node) là một phép thử thuộc tính.
- Mỗi nhánh (branch) tương ứng là một kết quả phép thử của nút nối đến cây con bên dưới. Mỗi nút trong sẽ có số nhánh tương ứng với số kết quả có thể có của phép thử.
- Mỗi lá (leaf node) là kết quả phân lớp, hồi quy.

Cây quyết định hoạt động bằng cách bắt đầu đi từ nút gốc của cây và đi xuyên qua cây theo các nhánh cho tới khi gặp nút lá, khi đó ta có kết quả của bài toán.

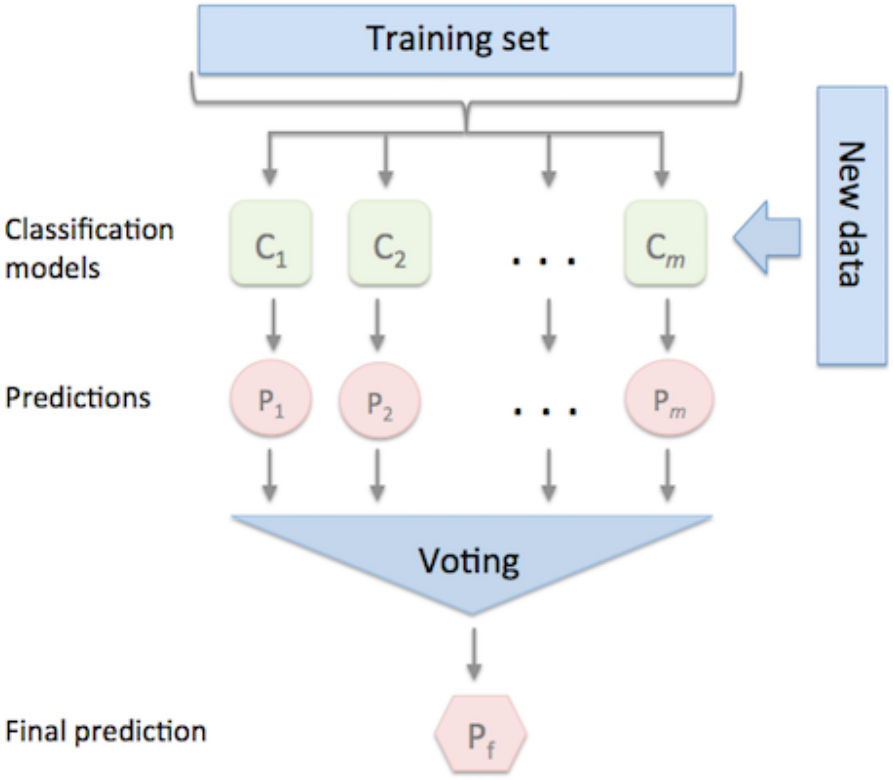
3.1.2 Khái niệm: Ensemble learning

Với mỗi bài toán cụ thể, ta thường có thể áp dụng nhiều thuật toán học khác nhau để xây dựng mô hình học. Tuy nhiên, các thuật toán này thường tuân thủ định lý "Không có bữa trưa miễn phí" (No free lunch theorem) [4], tức là không tồn tại một thuật toán mà luôn tốt cho mọi ứng dụng và mọi tập dữ liệu, không có thuật toán tốt hơn hẳn các thuật toán khác mà mỗi thuật toán có ưu/nhược điểm riêng.

Các thuật toán máy học thường dựa trên một tập các tham số (hyper-parameters) hoặc một giả thiết nhất định nào đó về phân bố dữ liệu. Do đó với mỗi bài toán, ta sẽ cần nhiều thời gian để thử nghiệm, hiệu chỉnh các tham số (fine tunning hyperparameters) của thuật toán để thu được độ chính xác cao.

Một cách khác để tăng độ chính xác cho bài toán cụ thể là kết hợp nhiều mô hình với nhau. Việc kết hợp hợp lý các mô hình có thể cho ta một mô hình mới có nhiều ưu điểm hơn. Phương pháp kết hợp này gọi là **Ensemble learning** (học máy tập hợp)

Một số cách kết hợp các mô hình phổ biến như xếp chồng (stacking), tăng cường (boosting), đóng bao (bagging)... đi kèm với phương pháp bỏ phiếu (voting) để cho ra kết quả cuối cùng.



Hình 3.2: Minh họa: Ensemble learning [5]

3.1.3 Khái niệm: Phương pháp bỏ phiếu (voting)

Bỏ phiếu là một cách đơn giản để kết hợp các mô hình học máy với nhau. Kết quả của phương pháp bỏ phiếu là kết quả tổng hợp có trọng số của các mô hình thành phần. Đối với dữ liệu mới x, mỗi mô hình thành phần C_i cho dự đoán P_i với trọng số ý kiến w_i tương ứng thì kết quả dự đoán của mô hình học tập hợp là:

$$P(x) = \sum_{i=1}^{N} w_i P_i \tag{3.1}$$

đối với bài toán hồi quy

$$P(x) = \underset{i}{\text{argmax}} \sum_{j=1}^{N} w_i (P_j = i) \tag{3.2}$$

đối với bài toán phân lớp [5]

3.1.4 Mô hình Random Forest

Khái niệm: Bootstrapping và Bagging

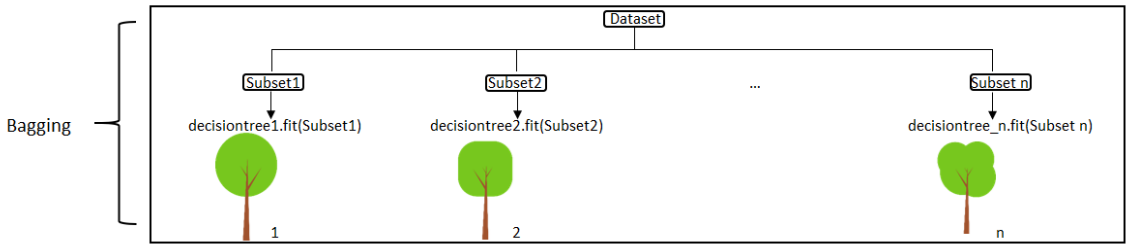
Bootstrapping là một phương pháp nổi tiếng trong thống kê được giới thiệu bởi Bradley Efron vào năm 1979 [6]. Phương pháp này thực hiện lặp lại nhiều lần:

- Lấy ra một mẫu có hoàn lại từ mẫu ban đầu
- Ước tính các tham số mong muốn từ mẫu lấy được

Từ đó, ước lượng được các tham số mong muốn từ mẫu ban đầu. Phương pháp này chủ yếu dùng để ước lượng lỗi chuẩn (standard errors), độ lệch (bias) và tính toán khoảng tin cậy (confidence interval) cho các tham số.

Bootstrap Aggregating (Bagging) là thuật toán kết hợp các mô hình máy học. Thuật toán sử dụng phương pháp Bootstrap để tạo nhiều bộ dữ liệu đầu vào từ một bộ dữ liệu, sau đó xây dựng các độc lập mô hình có

cùng thuật toán nhưng với từng bộ dữ liệu đầu vào khác nhau. Các mô hình này sẽ được kết hợp bằng phương pháp biểu quyết đa số (majority voting).



Hình 3.3: Minh họa: Bagging

Khái niệm: Random Forest

Random Forest [7], hay còn gọi là Random Decision Forest, là một phương pháp ensemble, kết hợp nhiều mô hình cây quyết định phân lớp/hồi quy (Classification and regression trees - CART), cải tiến của phương pháp Bagging. RF được phát triển bởi Leo Breiman. Ông đồng thời cũng là tác giả của CART [8]

Mô tả thuật toán xây dựng Random Forest

- Bước 1: Chọn ngẫu nhiên k từ n thuộc tính (k<n) có trong bộ dữ liệu huấn luyện.
- Bước 2: Sử dụng phương pháp boostrapping, chọn có hoàn lại từ bộ dữ liệu huấn luyện để tạo ra một bộ dữ liệu mới.
- Bước 3: Xây dựng cây quyết định với bộ dữ liệu ở bước 2, chỉ sử dụng k thuộc tính đã chọn ở bước 1
- Bước 4: Lặp lại Bước 2-3 để tạo nhiều cây quyết định khác.
- Bước 5: Kết hợp các cây quyết định thành phần bằng phương pháp bỏ phiếu.

Hình 3.4: Minh họa: Random Forest [9]

Dự đoán bằng Random Forest

Với x là một điểm dữ liệu mới, mỗi cây thành phần dự đoán hồi quy $T_i(x)$, dự đoán hồi quy của mô hình là:

$$\hat{f}_{rf}^{N}(x) = \frac{1}{N}\sum_{i=1}^{N} T_i(x) \tag{3.3}$$

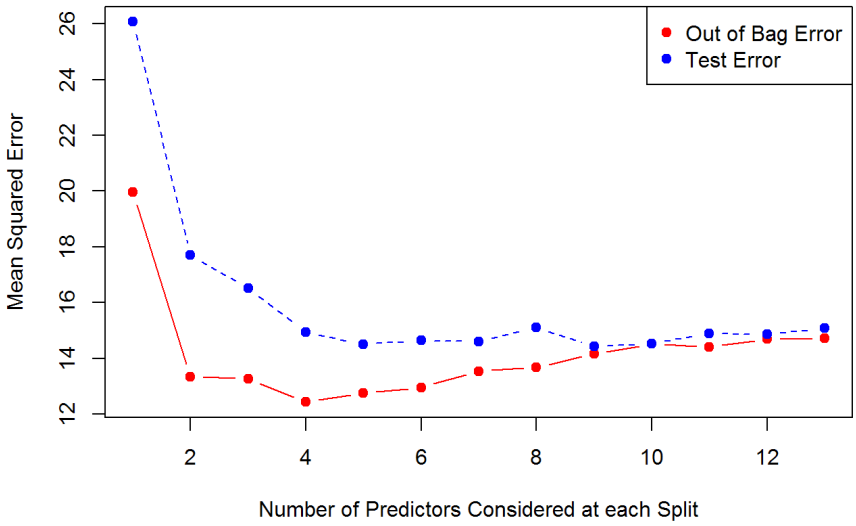
Với $\hat{T}_i(x)$ là kết quả phân lớp của cây thành phần, dự đoán phân lớp của mô hình là:

$$\hat{C}_{rf}^{N}(x) = majority\ vote\ \{\hat{T}_i(x)\}_1^N \tag{3.4}$$

Đánh giá Random Forest

Do sử dụng phương pháp boostrapping để tạo các mẫu huấn luyện cho các CART thành phần, chỉ có khoảng $\frac{2}{3}$ dữ liệu không trùng lặp trong bộ huấn luyện ban đầu được sử dụng cho việc huấn luyện. $\frac{1}{3}$ dữ liệu còn lại

không tham gia huấn luyện được gọi là 'Out-of-bag dataset' (OOB). Phần dữ liệu này có thể xem như là một tập kiểm thử (validation dataset) dùng để đánh giá và tính toán độ quan trọng các thuộc tính của các CART trong rừng. Độ lỗi của RF trên tập dữ liệu OOB được gọi là 'Out-of-bag Error' (E_{OOB}).



Hình 3.5: Tương quan Out-of-bag error và test error

Có thể so sánh E_{OOB} giữa các RF có số thuộc tính được chọn k (ở bước 1) khác nhau và số lượng cây trong rừng khác nhau để chọn được mô hình RF có độ lỗi thấp nhất.

Điểm mạnh của Random Forest

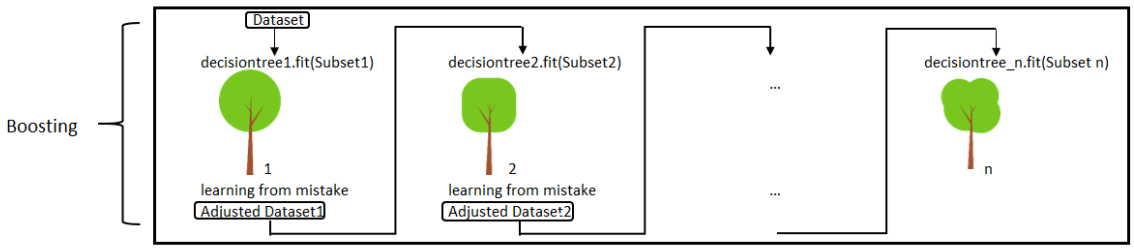
Thuật toán xây dựng có sự ngẫu nhiên trong mẫu dữ liệu (dùng phương pháp bootstrapping) và ngẫu nhiên trong số thuộc tính của mẫu so với tập thuộc tính ban đầu. Do đó, các tập huấn luyện con được tạo ra có tính đa dạng, ít liên quan. Mỗi CART xây dựng từ những tập dữ liệu con này không dùng tất cả dữ liệu training, cũng như không dùng tất cả các thuộc tính của dữ liệu để xây dựng nên mỗi cây có bias cao. Tuy nhiên, kết quả cuối của RF là kết hợp của nhiều cây thành phần, thông tin từ các cây sẽ bổ sung thông tin cho nhau, dẫn đến mô hình có low bias và low variance, tức là mô hình có kết quả dự đoán tốt.

Trong rừng, mỗi cây thành phần chỉ được huấn luyện trên một tập nhỏ các thuộc tính thay vì toàn bộ (bước 1), cơ chế này giúp RF thực thi nhanh khi áp dụng trên tập dữ liệu có số lượng lớn thuộc tính. Hơn nữa, việc xây dựng từng cây thành phần là độc lập nên có thể dễ dàng thực hiện song song.

3.1.5 Mô hình Gradient Boosting

Khái niệm: Boosting

Boosting là một kỹ thuật ensemble với mục đích từ một số mô hình học yếu (weak learner) tạo ra mô hình học mạnh hơn (strong learner) bằng cách cho những mô hình sau sửa lỗi (học từ lỗi) của các mô hình trước. Các mô hình được thêm vào theo cách này cho đến khi đạt số lượng mô hình tối đa cho phép hoặc dự đoán hoàn toàn trùng khớp với tập huấn luyện.



Hình 3.6: Minh họa: Boosting

Thuật toán boosting đơn giản nhất là Ada Boost, được giới thiệu vào năm 1995 bởi Freund và Schapire [10]. Thuật toán này sử dụng các CART có độ sâu nhỏ, hay còn gọi là một gốc (stump), làm các weak learner thành phần.

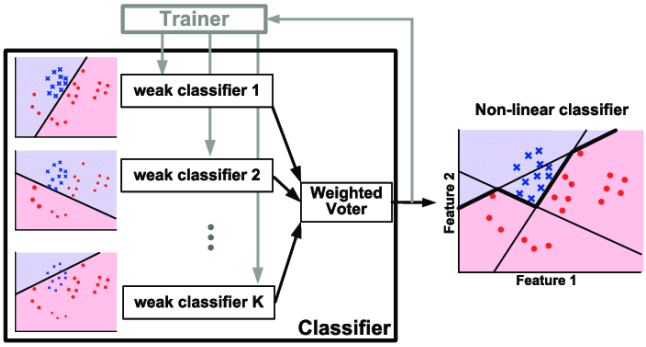
Ada Boost thực hiện việc học tăng cường bằng cách gán cho mỗi điểm dữ liệu trong bộ huấn luyện bộ huấn luyện một trọng số mẫu (sample wieght). Ban đầu, tất cả điểm dữ liệu đều được khởi tạo trọng số này bằng $\frac{1}{n}$, với n là tổng số điểm dữ liệu. Sau đó, với mỗi gốc được tạo, tổng lỗi của gốc đó sẽ quyết định trọng số của gốc trong việc bỏ phiếu (wieghted

voting) cho kết quả dự đoán cuối. Những điểm dữ liệu cũng được cập nhật lại sample wieght dựa vào kết quả gốc hiện tại dự đoán đúng (sample wieght giảm) hay dự đoán sai (sample wieght tăng) thể hiện gốc sau đó cần dự thiết dự đoán đúng các điểm dữ liệu đang được dự đoán sai (gốc tiếp theo cố gắng sửa lỗi của gốc trước đó)

Cụ thể thuật toán Ada Boost như sau [11]:

- Khởi tạo $w_1 = \{\frac{1}{N}\}_1^N$
- Với M là số gốc tối đa, lặp lại m=1,...M:
  - Huấn luyện gốc sử dụng sample weight w_m
  - Tính độ lỗi:
$$\epsilon_t = Pr_{i\ D_t}[h_t \neq y_i] \tag{3.5}$$
  - Thiết lập trọng số cho gốc này:
$$\alpha_t = \frac{1}{2} \ln \frac{1 - \epsilon_t}{\epsilon_t} \tag{3.6}$$
  - Cập nhật sample wieght cho việc huấn luyện gốc tiếp theo:
$$D_{t+1} = \frac{D_t(i) e^{-\alpha_t}}{Z_t} \tag{3.7}$$

    (với Z_t là hệ số chuẩn hóa để tổng sample wieght bằng 1)
- Cuối cùng, kết hợp các gốc bằng bỏ phiếu có trọng số:
$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x)) \tag{3.8}$$



Hình 3.7: Minh họa: Ada boost [12]

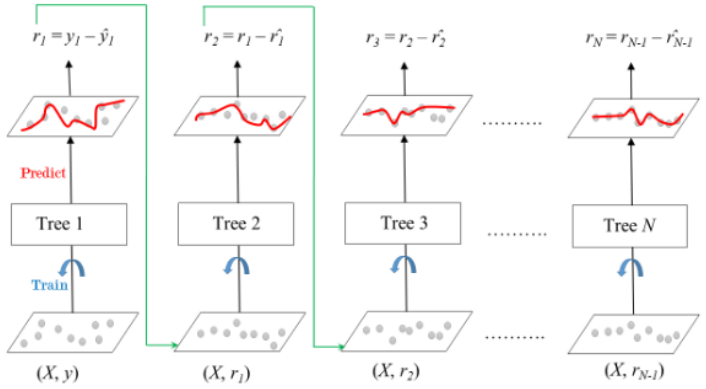
Gradient Boosting

Ý tưởng của thuật toán Gradient Boosting (GB) bắt nguồn từ Leo Breiman. Ông cho rằng phương pháp boosting có thể xem như là một thuật toán tối ưu hóa trên một hàm mất mát (Loss function) phù hợp [13]. Thuật toán Gradient boosting sau đó được phát triển bởi Jerome H. Friedman [14] [15].

Khác với Ada Boost, ở Gradient Boosting:

- Sử dụng cây phân lớp, hồi quy làm các weaker learner thành phần thay vì gốc
- Những CART thành phần không huấn luyện dựa trên trọng số mẫu (mô hình tiếp theo cố gắng dự đoán đúng những điểm lỗi của mô hình trước), mà dựa trên 'độ lệch (lỗi) giả'(pseudo-residual) của CART tiền nhiệm trước đó (mô hình tiếp theo cố gắng giảm độ lỗi có được từ mô hình trước).
- Những mô hình thành phần không có trọng số riêng, thay vào đó chúng có chung một tỉ lệ tốc độ học (learning rate) trong khoảng 0-1 để điều chỉnh tốc độ học của từng mô hình mới.

Thuật toán kết thúc khi đạt số lượng cây tối đa hoặc việc thêm những cây mới không giảm pseudo-residual.



Hình 3.8: Minh họa: Gradient boosting [16]

Thuật toán Gradient Boosting [17] [18]:

Input: Dữ liệu huấn luyện $\{(x_i, y_i)\}_{i=1}^n$, một hàm Loss $L(y, F(x))$ và số lượng cây tối đa M

- Bước 1: Khởi tạo dự đoán ban đầu với hằng số

$$F_0(x) = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, \gamma) \tag{3.9}$$

- Bước 2: Lặp m=1 đến M để tạo M cây thành phần:

  (A) Tính 'độ lệch giả' (pseudo-residual) của các điểm dữ liệu trong tập huấn luyện: Với mỗi i = 1, ..., n

$$r_{im} = -[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}]_{F(x)=F_{m-1}(x)} \tag{3.10}$$

  (B) Xây dựng cây dự đoán các giá trị r_{im}, gọi R_{jm} là các lá của cây, vớij=1, ...,J_m (cây có J_m lá)

  (C) Tính giá trị output ở từng lá của cây vừa tạo :

$$\gamma_{jm} = \underset{\gamma}{\operatorname{argmin}} \sum_{x_i \in R_{ij}} L(y_i, F_{m-1}(x_i) + \gamma) \tag{3.11}$$

(D) Cập nhật dự đoán mới:

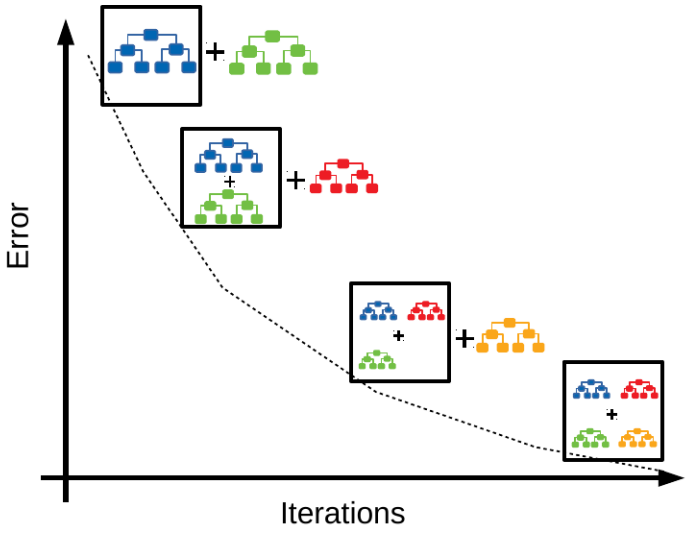
$$F_m(x) = F_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_m I(x \in Rjm) \tag{3.12}$$

với ν là tỷ lệ tốc độ học (learning rate)

Output: Kết quả dự đoán là kết quả của cây cuối cùng: $F_M(x)$

Điểm mạnh của Gradient Boosting

Qua mỗi cây mới được thêm vào, độ lỗi của mô hình sẽ ngày càng được giảm. Với các siêu tham số (hyperparameter) hợp lý, mô hình có thể cho được độ chính xác cao.



Hình 3.9: Độ lỗi qua các iteration - GB [19]

Gradient Boosting có thể được tối ưu với nhiều hàm Loss khác nhau và cho phép hiệu chỉnh (fine tunning) nhiều siêu tham số (hyperparameter) giúp mô hình có độ linh hoạt cao giữa nhiều bộ dữ liệu khác nhau.

3.1.6 Mô hình XGBoost

XGBoost, viết tắt cho eXtreme Gradient Boosting, là một phiên bản Gradient Boosting được tối ưu hóa bởi Tianqi Chen [20].

Để cải tiến GB, mục tiêu của XGBoost không chỉ cố gắng tối thiểu hóa hàm Loss mà còn kèm theo một hàm chính quy hóa (regularization). Ta có thể gọi hàm mục tiêu (objective function) của XGBoost là: [21]

$$obj(\theta) = \sum_{i=1}^{n} L(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{3.13}$$

với L là hàm Loss, Ω là hàm regularization, f_k là mô hình thành phần thứ k. Việc thêm hàm regularization giúp cải tiến vấn đề overfit của mô hình Gradient Boosting, do sau mỗi lần thêm cây thành phần, độ lỗi của mô hình GB luôn được giảm cho đến tối thiểu hoặc mô hình đạt tối đa số cây thành phần.

Xây dựng CART trong mô hình XGBoost

Một trong những cải tiến của XGBoost so với Gradient Boost là ở việc tạo các cây hồi quy/phân lớp thành phần.

Ở iteration thứ t, cây được tạo cho dự đoán

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i) = \sum_{k=1}^{T} \Omega(f_k(x_i)) \tag{3.14}$$

Độ phức tạp của cây được định nghĩa là [21]

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{3.15}$$

với γ và λ là các hệ số regularization, T là số lá của cây. Ta thấy $\sum_{j=1}^{T} w_j^2$ là tổng bình phương các lá, hay chính là tổng bình phương output của cây. Ta có thể đặt:

$$O_{value}^2 = f_t(x_i)^2 = \sum_{j=1}^{T} w_j^2 \tag{3.16}$$

khi đó độ phức tạp của cây là:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda O_{value}^2 \tag{3.17}$$

Lúc này, hàm mục tiêu của XGBoost là [21] [22]

$$\begin{aligned} obj^{(t)} &= \sum_{i=1}^{n} L(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) + constant \\ &= \sum_{i=1}^{n} L(y_i, \hat{y}_i^{(t-1)} + O_{value}) + \gamma T + \frac{1}{2}\lambda O_{value}^2 + constant \end{aligned} \tag{3.18}$$

Vậy, XGBoost chỉ thêm cây có output làm tối thiểu hóa được hàm mục tiêu này. Các hằng số sẽ không ảnh hưởng đến bài toán tối thiểu hóa nên ta có thể bỏ chúng đi

$$obj^{(t)} = \sum_{i=1}^{n} L(y_i, \hat{y}_i^{(t-1)} + O_{value}) + \frac{1}{2}\lambda O_{value}^2 \tag{3.19}$$

Áp dụng xấp xỉ Taylor bậc 2 (second order Taylor Approximation) [23]:

$$\begin{aligned} L(y_i, \hat{y}_i^{(t-1)} + O_{value}) \approx L(y_i, \hat{y}_i^{(t-1)}) &+ \left[\frac{d}{d_{\hat{y}_i^{(t-1)}}} L(y_i, \hat{y}_i^{(t-1)})\right] O_{value} \\ &+ \frac{1}{2}\left[\frac{d^2}{d^2_{\hat{y}_i^{(t-1)}}} L(y_i, \hat{y}_i^{(t-1)})\right] O_{value}^2 \end{aligned} \tag{3.20}$$

Đặt

$$\begin{aligned} g &= \left[\frac{d}{d_{\hat{y}_i^{(t-1)}}} L(y_i, \hat{y}_i^{(t-1)}) \right] \\ h &= \left[\frac{d^2}{d^2_{\hat{y}_i^{(t-1)}}} L(y_i, \hat{y}_i^{(t-1)}) \right] \end{aligned} \tag{3.21}$$

Khi đó, hàm mục tiêu có thể được viết thành

$$obj^{(t)} = \sum_{i=1}^{n} L(y_i, \hat{y}_i^{(t-1)}) + \sum_{i=1}^{n} g_i O_{value} + \frac{1}{2} \sum_{i=1}^{n} h_i O_{value}^2 + \frac{1}{2} \lambda O_{value}^2 \tag{3.22}$$

Ta thấy được $\sum_{i=1}^{n} L(y_i, \hat{y}_i^{(t-1)})$ không chứa O_{value}, nên nó sẽ không ảnh hưởng đến quá trình tối thiểu hóa và có thể được bỏ đi. Hàm mục tiêu bây giờ là

$$obj^{(t)} = O_{value} \sum_{i=1}^{n} g_i + \frac{1}{2} O_{value}^2 (\sum_{i=1}^{n} h_i + \lambda) \tag{3.23}$$

Để tìm O_{value} làm tối thiểu hàm mục tiêu, ta lấy đạo hàm hàm mục tiêu theo O_{value} và tìm nghiệm bằng 0:

$$\begin{gathered} \sum_{i=1}^{n} g_i + O_{value} (\sum_{i=1}^{n} h_i + \lambda) = 0 \\ \Leftrightarrow O_{value} = - \frac{\sum_{i=1}^{n} g_i}{\sum_{i=1}^{n} h_i + \lambda} \end{gathered} \tag{3.24}$$

Ở bài toán hồi quy, giả sử ta sử dụng hàm Loss MSE $L(y_i, \hat{y}_i^{(t-1)}) =$

$\frac{1}{2}(y_i - \hat{y}_i^{(t-1)})^2$, ta có thể tính g_i và h_i

$$\begin{aligned} g &= -(y_i - \hat{y}_i^{(t-1)}) \\ h &= 1 \end{aligned} \tag{3.25}$$

thì hàm mục tiêu là:

$$O_{value} = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i^{(t-1)})}{\sum_{i=1}^{n} 1 + \lambda} \tag{3.26}$$

Ta thấy được ở phân số trên, tử số chính là tổng độ lỗi (sum of residuals) và mẫu số chính là tổng số phần tử trong kết quả (number of residuals) trong output + λ

Vậy hàm mục tiêu của mô hình hồi quy này có thể hiểu là [22]

$$O_{value} = \frac{\text{Sum of Residuals}}{\text{Number of Residuals} + \lambda} \tag{3.27}$$

Đây chính là công thức kết quả của một lá trên cây hồi quy.

Ở bài toán phân lớp, giả sử ta sử dụng hàm Loss

$$L(y_i, \hat{y}_i^{(t-1)}) = -[y_i \log(\hat{y}_i^{(t-1)}) + (1 - y_i) \log(1 - \hat{y}_i^{(t-1)})] \tag{3.28}$$

ta có thể tính g_i và h_i [22]

$$\begin{aligned} g &= -(y_i - \hat{y}_i^{(t-1)}) \\ h &= \hat{y}_i^{(t-1)}(1 - \hat{y}_i^{(t-1)}) \end{aligned} \tag{3.29}$$

thì hàm mục tiêu là:

$$O_{value} = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i^{(t-1)})}{\sum_{i=1}^{n}(\hat{y}_i^{(t-1)}(1 - \hat{y}_i^{(t-1)})) + \lambda} \tag{3.30}$$

hay dễ hiểu chính là: [22]

$$O_{value} = \frac{\text{Sum of Residuals}}{\Sigma[\text{Previous Probability x (1 - Previous Probability)}] + \lambda} \quad (3.31)$$

Đây chính là công thức kết quả của một lá trên cây phân lớp.

Tách lá ở cây

Ở XGBoost, khi tách một lá, ta dùng hàm Gain để xác định xem các cây con mới này phân cụm tốt hơn lá ban đầu hay không.

$$\begin{aligned}\text{Gain} = &\text{ Similarity Score(Lá trái)} + \text{Similarity Score(Lá phải)} \\ &- \text{Similarity Score(Gốc)} - \gamma\end{aligned} \quad (3.32)$$

và tất nhiên cây con có Gain lớn nhất sẽ được chọn để tách lá.

Công thức của Similarity Score (được miêu tả trong bản thảo XGBoost ban đầu [20]) ở hàm Gain trong XGBoost chính là âm của hàm mục tiêu đã được tối thiểu hóa.Tức là, để tính công thức Similarity Score, ta thế đáp án O_{value} ta tính được ở (3.24) vào (3.23)

$$\begin{aligned}\text{Similarity Score} &= -\sum_{i=1}^{n} g_i\left(-\frac{\sum_{i=1}^{n} g_i}{\sum_{i=1}^{n} h_i + \lambda}\right) - \frac{1}{2}\left(\sum_{i=1}^{n} h_i + \lambda\right)\left(-\frac{\sum_{i=1}^{n} g_i}{\sum_{i=1}^{n} h_i + \lambda}\right)^2 \\ &= \frac{1}{2}\frac{\left(\sum_{i=1}^{n} g_i\right)^2}{\sum_{i=1}^{n} h_i + \lambda}\end{aligned} \quad (3.33)$$

Tuy nhiên, do Similarity Score là tương đối giữa các cây con và để giảm số tính toán cần thiết (tối ưu hóa), Similarity Score được áp dụng thực tế

được bỏ đi phần $\frac{1}{2}$

$$\text{Similarity Score} = \frac{(\sum_{i=1}^{n} g_i)^2}{\sum_{i=1}^{n} h_i + \lambda} \tag{3.34}$$

Tương tự như cách tính O_{value} cho bài toán hồi quy và phân lớp đã nêu ở trên, ta có thể tính được hàm Similarity Score với các hàm Loss khác nhau. Giả sử bài toán hồi quy và phần lớp sử dụng hàm Loss nêu trên, ta có thể tính hàm Similarity Score:

$$\text{Similarity Score} = \frac{(\sum_{i=1}^{n} (y_i - \hat{y}_i^{(t-1)}))^2}{\sum_{i=1}^{n} 1 + \lambda} \tag{3.35}$$

hay dễ hiểu chính là: [22]

$$\text{Similarity Score} = \frac{(\text{Sum of Residuals})^2}{\text{Number of Residuals} + \lambda} \tag{3.36}$$

cho hồi quy, và

$$\text{Similarity Score} = \frac{(\sum_{i=1}^{n} (y_i - \hat{y}_i^{(t-1)}))^2}{\sum_{i=1}^{n} (\hat{y}_i^{(t-1)}(1 - \hat{y}_i^{(t-1)})) + \lambda} \tag{3.37}$$

hay dễ hiểu chính là: [22]

$$\text{Similarity Score} = \frac{(\text{Sum of Residuals})^2}{\Sigma[\text{Previous Probability x (1 - Previous Probability)}] + \lambda} \tag{3.38}$$

cho phân lớp.

Tỉa cành cây

Quay lại với hàm Gain (3.32) sau khi ta đã tính được công thức của Similariry Score (3.34)

$$\text{Gain} = \frac{(\sum_{i=1}^{n} g_i^{\text{Lá trái}})^2}{\sum_{i=1}^{n} h_i^{\text{Lá trái}} + \lambda} + \frac{(\sum_{i=1}^{n} g_i^{\text{Lá phải}})^2}{\sum_{i=1}^{n} h_i^{\text{Lá phải}} + \lambda} - \frac{(\sum_{i=1}^{n} g_i^{\text{Ngọn}})^2}{\sum_{i=1}^{n} h_i^{\text{Ngọn}} + \lambda} - \gamma \tag{3.39}$$

Khi đã xây dựng xong cây, từ dưới lên trên, những nút đã tách lá nhưng cho Gain nhỏ hơn 0 sẽ bị 'tỉa cảnh' (tree pruning), gộp lại hai lá mà nút đó đã tách thành trở về một nút ban đầu. Tỉa cành ở đây giúp giảm độ phức tạp của các cây thành phần (regularization), giúp XGBoost giảm overfit

Dựa vào công thức hàm Gain (3.39), ta thấy được hệ số regularization λ và α càng cao thì việc tỉa cành sẽ càng mạnh.

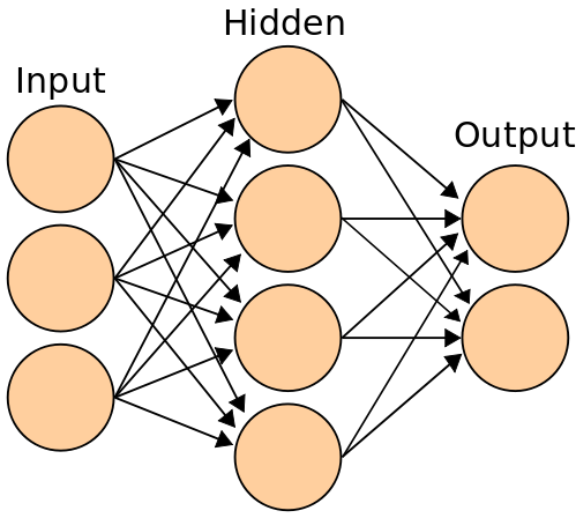
3.1.7 Mô hình LSTM

Artificial Neural Network (ANN)

Artificial Neural Network hay mạng thần kinh nhân tạo, gọi tắt là mạng thần kinh hoặc mạng neural, là một mô hình xử lý thông tin lấy cảm hứng từ mạng neural sinh học. Trong thực tế sử dụng, mạng neural được ứng dụng cho việc mô hình hóa dữ liệu *thống kê phi tuyến* (nonlinear stochastic), ngoài ra, mạng neural còn được sử dụng để mô hình hóa các mối quan hệ phức tạp giữa dữ liệu vào và kết quả hoặc để tìm *hình mẫu* (pattern) trong dữ liệu.

Một mạng neural bao gồm một số lượng lớn các *nút* (hay node) kết nối với nhau, mô phỏng các neural thần kinh của não người, thông qua các liên kết để thống nhất giải quyết một vấn đề cụ thể nào đó. Trong nhiều trường hợp, mạng nơ-ron là một hệ thống thích ứng với khả năng tự thay đổi cấu trúc của mình dựa trên các thông tin mới thu thập trong quá trình học.

Kiến trúc của một mạng neural gồm 3 thành phần đó là lớp vào - input layer, lớp ẩn - hidden layer và lớp ra - output layer.



Hình 3.10: Minh họa cấu trúc của một mạng neural. [24]

Trong đó, lớp vào thể hiện cho các đầu vào của mạng; lớp ẩn chứa các neural có nhiệm vụ tiếp nhận dữ liệu đầu vào từ các neural ở lớp trước, sau đó chuyển đổi lượng thông tin này để truyền cho các lớp tiếp theo; lớp ra thể hiện cho các đầu ra của mạng. Lưu ý, một mạng neural có thể chứa nhiều lớp ẩn ở giữa lớp vào và lớp ra.

Hoạt động của một mạng nơ-ron nhân tạo được cấu thành từ các "building block" (tạm dịch là đơn thể) gồm các bước:

- Đầu vào: mỗi đầu vào tương ứng với 1 đặc trưng của dữ liệu. Trong đó, mỗi đầu vào sẽ nhận một giá trị trọng số liên kết thể hiện độ mạnh của từng đầu vào đối với quá trình xử lý thông tin để chuyển đổi dữ liệu từ lớp này sang lớp khác. Ta có thể nói, quá trình học của ANN chỉ là quá trình điều chỉnh các trọng số này để có được kết quả như ý:

- Hàm tổng: tính tổng trọng số của tất cả các đầu vào được đưa vào mỗi neural bằng công thức: $\sum_{i=1}^{n} x_i w_i$ với w_i là trọng số của các đầu vào $x_i \in \mathbf{x} \in \mathbb{R}^n$.

- Hàm truyền: thể hiện mối quan giữa hàm tổng và kết quả đầu ra. Hàm truyền nhận đầu vào là **kết quả của hàm tổng** và **"bias"** để

đưa ra đầu ra **y**. Chú ý, hàm truyền tồn tại nhiều lựa chọn và mỗi lựa chọn có ảnh hưởng lớn đến kết quả đầu ra của mạng.

- Đầu ra: kết quả của mạng neural hay chính là một giải pháp cho một vấn đề,

Hàm truyền (hay còn có tên là hàm kích hoạt - activation fuction) là một hàm vô hướng có kết quả là một giá trị, gọi là mức độ kích hoạt của đơn vị. Các đặc điểm của tín hiệu đầu ra sẽ quyết định hàm truyền của mạng ở lớp ra. Một số hàm truyền phổ biến là:

- **Hàm đồng nhất** (Indentity): $f(x) = x \in (-\infty, \infty)$.
- **Hàm sigmoid** (có ký hiện là σ): $\sigma(x) = \dfrac{1}{1+e^{-x}} \in (0,1)$.
- **Hàm tanh** (Hyperbolic tangent, còn được viết là tanH): $\tanh(x) = \dfrac{e^z - e^{-z}}{e^x + e^{-z}} \in (-1,1)$.
- **Hàm bước nhị phân** (Binary step): $f(x) = \begin{cases} 0 \text{ if } x < 0 \\ 1 \text{ if } x \geq 0 \end{cases} \in \{0,1\}$.

Ngoài ra, còn có các hàm thuộc nhóm đơn vị tuyến tính::

- **ReLU** (Rectified Linear Unit): $f(x) = \max(0,x) \in [0,\infty)$.
- **GELU** (Gaussian Error Linear Unit): $f(x) = \dfrac{1}{2}x\left(1 + \operatorname{erf}\left(\dfrac{x}{\sqrt{2}}\right)\right) \in (-0.17\ldots,\infty)$, với $\operatorname{erf}(t) = \dfrac{2}{\sqrt{\pi}}\displaystyle\int_0^t e^{-s^2}ds$ là hàm lỗi Gauss.
- **ELU** (Exponential Linear Unit): $f(x) = \begin{cases} \alpha(e^x - 1) \text{ if } x \leq 0 \\ x \text{ if } x > 0 \end{cases} \in \{-\alpha, \infty\}$ với tham số $\alpha > 0$.

- **SELU** (Scaled Exponential Linear Unit): $f(x) = \lambda \begin{cases} \alpha(e^x - 1) \text{ if } x < 0 \\ x \text{ if } x \geq 0 \end{cases} \in \{-\lambda\alpha, \infty\}$ với tham số $\lambda = 1.0507$ và $\alpha = 1.67326$.
- **Leaky ReLU**: $f(x) = \begin{cases} 0.01x \text{ if } x < 0 \\ x \text{ if } x \geq 0 \end{cases} \in \{-\infty, \infty\}$.
- **Parameteric Rectified Linear Unit** (PReLU): $f(x) = \begin{cases} \alpha x \text{ if } x < 0 \\ x \text{ if } x \geq 0 \end{cases} \in \{-\infty, \infty\}$ với tham số $\alpha > 0$.
- **Sigmoid Linear Unit** (SiLU): $f(x) = \dfrac{x}{1 + e^{-x}} \in [-0.278\ldots, \infty)$.

Còn lại là các hàm:

- **Softplus**: $f(x) = \ln(1 + e^x) \in (0, \infty)$.
- **Mish**: $f(x) = x\tanh(\ln(1 + e^x)) \in [-0.308\ldots, \infty)$.
- **Gaussian**: $f(x) = e^{-x^2} \in (0, 1]$.

Dưới đây là các hàm truyền đặc biệt mà không nhận một x duy nhất từ một hoặc nhiều lớp trước:

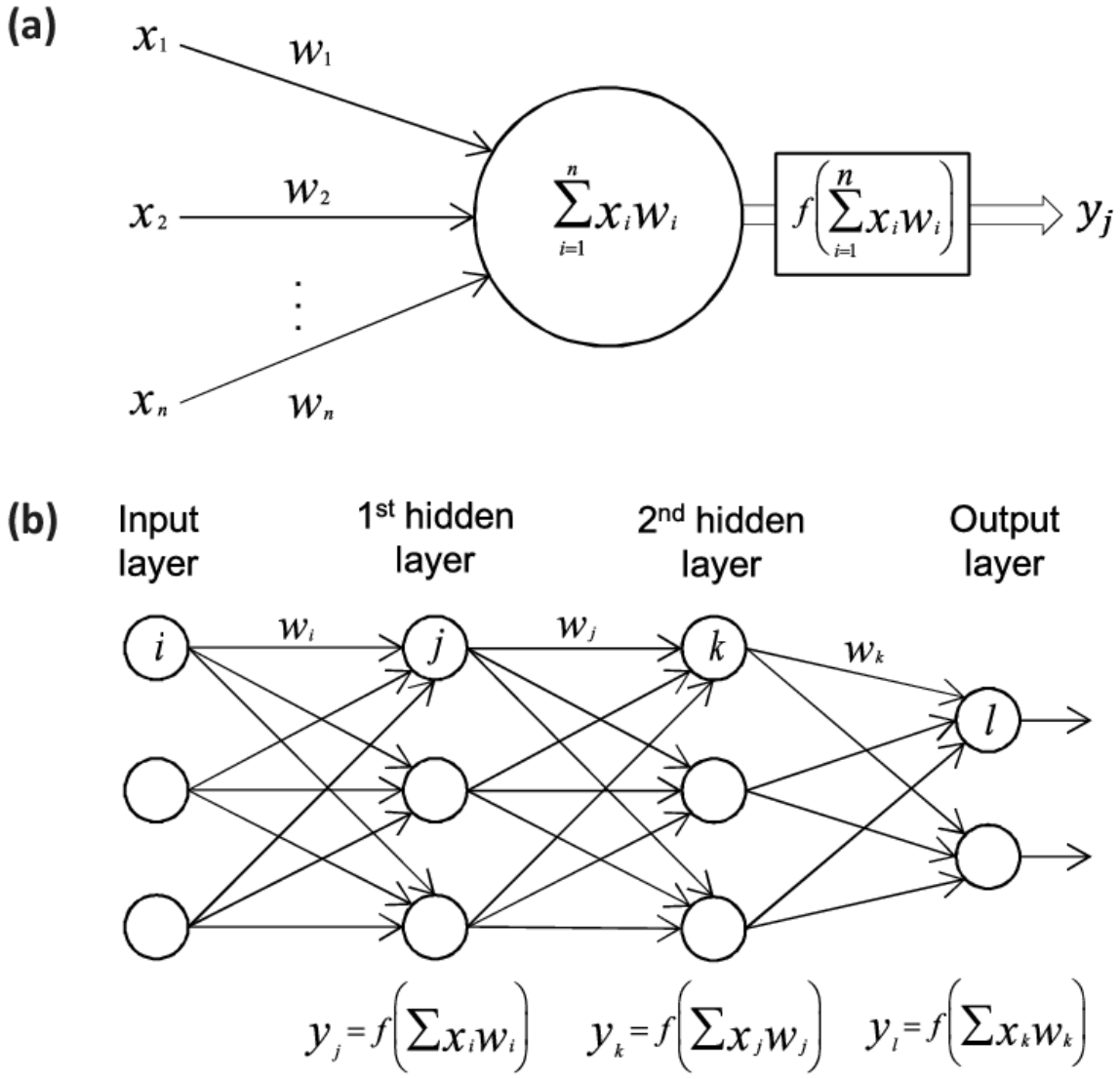
- **Softmax**: $f_i(\vec{x}) = \dfrac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}} \in (0, 1)$.
- **Maxout**: $f_i(\vec{x}) = \max_i x_i \in \{-\infty, \infty\}$.

Tóm lại, ta có thể mô tả mạng neural bằng phương trình:

$$y_i^{(l)} = f_i^{(l)} \left(\sum_{j=1}^{n^{(l-1)}} w_{ij}^{(l)} y_j^{(l-1)} + b_i^{(l)} \right). \tag{3.40}$$

Trong đó:

- $l = 1 \dots L$, với L là số lớp của mạng.
- $y_i^{(l)}$ là đầu ra của neural thứ i ở lớp thứ l, với $i = 1 \dots n^l$.
- n^l là số neural ở lớp thứ l.
- $b_i^{(l)}$ là *ngưỡng* (bias) của neural thứ i ở lớp thứ l.
- $f_i^{(l)}$ là hàm truyền của neural thứ i ở lớp thứ l.
- Đầu vào của mạng là $y^{(0)} = \mathbf{x}$ và đầu ra cuối là $\mathbf{y} = y^{(L)}$.



Hình 3.11: Mô tả hoạt động của mạng neural (lược bỏ ngưỡng). [25]

Hình 3.11 (a) minh họa các đơn thể của một mạng neural hoàn chỉnh. Còn ở (b), đây là một ví dụ cho *mạng neural nhân tạo truyền thẳng* (Feed-forward Neural Network - FNN), hay là một *mạng perceptron nhiều lớp* (Multilayer Perceptron - MLP) gồm hai lớp ẩn kết nối hoàn toàn với nhau ở mỗi nút (fully connected network). Trong đó, mỗi neural thứ j ở lớp ẩn thứ nhất sẽ được áp dụng hàm tổng và sau đó, kết quả sau khi truyền là y_j tại đây sẽ đóng vai trò là đầu vào cho lớp ẩn thứ hai.

Như vậy, quá trình học của mạng neural, hay chính là quá trình tìm bộ trọng số $\mathbf{w}$, phù hợp sẽ lặp liên tục và có thể không dừng đến khi tìm ra kết quả như ý. Vì vậy, trong thực tế, khi huấn luyện một mô hình mạng neural, một tiêu chuẩn dựa trên một giá trị sai số nào giữa đầu ra của mạng và đầu ra mong muốn cần được thiết lập, hoặc một số lần lặp tối đa xác định nào đó. Từ đây, ta tiếp cận thuật toán lan truyền ngược (Backpropagation), được sử dụng để điều chỉnh các trọng số liên kết sao cho tổng sai số nhỏ nhất. Với các mạng neural hiện đại, giải thuật được sử dụng kết hợp với một phương pháp tối ưu hóa như gradient descent để rút ngắn thời gian chạy của mạng.

Backpropagation

Trước hết, Gradient descent là một thuật toán tìm giá trị nhỏ nhất của hàm số f(x) dựa trên đạo hàm của nó. Thuật toán gồm 3 bước chính: (1) Khởi tạo giá trị $x = x_0$ tùy ý; (2) Tính $x_1 = x_0 - \eta f'(x_0)$, với η là *learning rate* (tạm dịch là tốc độ học); (3) Tính lại $f(x)$ với các giá trị x mới, sao cho nếu $f(x)$ đủ nhỏ thì dừng lại, còn không, lặp lại bước (2) với $x_2, x_3, \ldots$

Từ đó, giả sử, với bộ trọng số $\mathbf{w} = \{w_{ij}^{(l)}\}$, ta sẽ thu được kết quả của hàm từ đầu vào $\mathbf{x}$. Ta đặt liên hệ này là $h(\mathbf{x})$, vậy, hàm tính sai số hay lỗi cho một cặp $(\mathbf{x}_n, y_n)$ nào đó là $e(h(\mathbf{x}_n), y_n)$. Tuy nhiên, do y_n là hằng số, và $h(\mathbf{x}_n)$ chỉ phụ thuộc vào $\mathbf{w}$, nên ta có thể viết lại biểu diễn hàm lỗi trên thành $e(\mathbf{w})$. Ta có thể nói, thuật toán lan truyền ngược áp dụng Stochastic Gradient Descent (gradient descent ngẫu nhiên - SGD) có mục

tiêu là tìm giá trị nhỏ nhất cho hàm $e(\mathbf{w})$ qua đạo hàm $\nabla e(\mathbf{w})$: $\dfrac{\partial e(\mathbf{w})}{\partial w_{ij}^{(l)}}$ của nó với toàn bộ i, j, l.

Áp dụng quy tắc chuỗi đạo hàm (chain rule), ta có:

$$\begin{aligned}\frac{\partial e(\mathbf{w})}{\partial w_{ij}^{(l)}} &= \frac{\partial e(\mathbf{w})}{\partial s_j^{(l)}} \times \frac{\partial s_j^{(l)}}{\partial w_{ij}^{(l)}} \\ &= \delta_j^{(l)} \times x_i^{(l-1)}\end{aligned}$$

với s là ký hiệu hàm tổng. Như vậy, ta chỉ cần tìm $\dfrac{\partial e(\mathbf{w})}{\partial s_j^{(l)}} = \delta_j^{(l)}$.

Tóm lại, backpropagation áp dụng SGD gồm các bước sau:

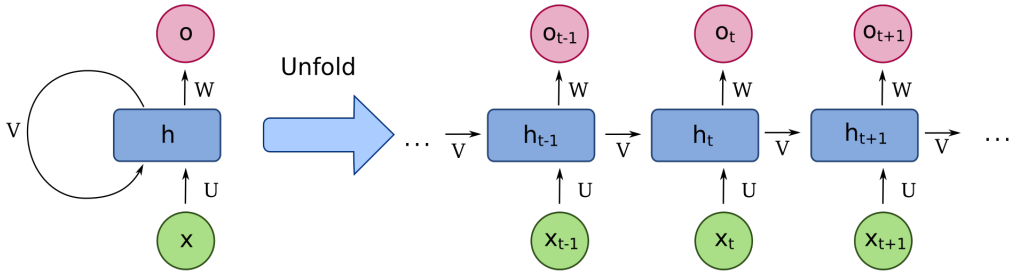
- Bước 1: Khởi tạo toàn bộ các trọng số $w_{ij}^{(l)}$ **ngẫu nhiên**.
- Bước 2: Chọn bất kỳ $j \in \{1, \ldots, N\}$
- Bước 3: Tính tất cả các $x_i^{(l)}$.
- Bước 4: Tính tất cả các $\delta_j^{(l)}$.
- Bước 5: Cập nhật $w_{ij}^{(l)} \leftarrow w_{ij}^{(l)} - \eta x_i^{(l-1)} \delta_j^{(l)}$
- Bước 6: Lặp lại bước 2, 3, 4, 5 cho đến khi cần dừng lại.

Recurrent Neural Network (RNN)

Ở hình 3.11 (b), như đã nói, kiến trúc mạng có hình dạng như vậy được gọi là kiến trúc truyền thẳng, tức là, không có các kết nối ngược trở lại từ các neural đầu ra về các neural đầu vào và mạng không lưu lại các giá trị output trước cũng như các trạng thái kích hoạt của neural. Các mạng neural truyền thẳng chỉ đưa tín hiệu di chuyển theo một hướng duy nhất: từ đầu vào tới đầu ra, và đầu ra của một tầng bất kì sẽ không ảnh hưởng tới tầng đó. Ngoài ra, mạng neural nhân tạo còn có kiểu kiến trúc phản hồi (feedback), hay còn có tên là mạng neural hồi quy (Recurrent Neural

Network). Ngược lại với FNN, RNN cho phép đưa tín hiệu theo cả hai hướng thẳng và ngược lại bằng các vòng lặp. Mạng lưu lại các trạng thái trước đó, và hơn nữa, các trạng thái tiếp theo không chỉ phụ thuộc vào các tín hiệu đầu vào, mà còn phụ thuộc vào các trạng thái trước đó của mạng.

Ý tưởng đằng sau RNN là RNN giống như trí nhớ ngắn hạn (Short Term Memory - STM) trong não người, tức là, RNN sẽ "học" và "ghi nhớ" lại thông tin ở chu kì trước và áp dụng ở bước "học" tiếp theo. Ngoài ra, ta có thể xem các trọng số có trí nhớ dài hạn (Long Term Memory - LTM). Ví dụ, nếu một mạng neural đã xác định được các trọng số liên kết, thì dù ta đưa đầu vào nào, cách xử lý cũng mạng vẫn sẽ giữ nguyên.



Hình 3.12: Mô hình mạng neural hồi quy. [26]

Mô hình của RNN được thể hiện ở hình 3.12 thực hiện lặp lại cùng một tác vụ, trong đó, kết quả đầu ra tại thời điểm hiện tại phụ thuộc vào kết quả tính toán của các thành phần ở thời điểm trước đó, và tương tự, kết quả này sẽ ảnh hưởng tới thành phần sau này. Nói cách khác, RNN là một mô hình có trí nhớ, có khả năng nhớ được thông tin đã tính toán trước đó, nhưng trí nhớ này ngắn hạn. Đặc biệt, khác với các mô hình mạng neural truyền thống, đó là thông tin đầu vào hoàn toàn độc lập với thông tin đầu ra.

Dựa theo hình 3.12, trong đó, x_t và o_t lần lượt là đầu vào và đầu ra tại thời điểm thứ t, h_t là trạng thái ẩn tại thời điểm thứ t và đóng vai trò là bộ nhớ của mạng. h_t sẽ được tính dựa trên sự kết hợp giữa các trạng thái

ẩn trước và đầu vào. Ta có thể viết:

$$h_t = f(Ux_t + VS_{t-1}).$$

Sau đó, o_t sẽ được tính dựa theo Wh_t, với (U, V, W) là ba tham số của mạng. Hai loại hàm truyền f phổ biến nhất là hàm tanh hoặc hàm ReLU.

Với khả năng "nhớ" được, mạng neural hồi quy được sử dụng để xử lý thông tin dạng chuỗi và các dữ liệu thời gian. Một số ứng dụng tiêu biểu là:

- Mô hình ngôn ngữ và phát sinh văn bản:
  Mô hình cho biết xác suất của một câu trong một ngôn ngữ là bao nhiêu, hay dự đoán xác suất từ tiếp theo của một câu cho trước là bao nhiêu. Từ bài toán này, chúng ta có thể mở rộng thành bài toán phát sinh văn bản. Mô hình này cho phép ta phát sinh ra văn bản mới dựa vào tập dữ liệu huấn luyện.

- Dịch máy (Machine Translation):
  Tương tự như mô hình ngôn ngữ, đầu vào là chuỗi các từ của ngôn ngữ cần dịch và đầu ra là chuỗi các từ của ngôn ngữ mong muốn. Điểm khác biệt giữa hai loại mô hình là đầu ra ở đây chỉ có thể dự đoán được khi đầu vào đã hoàn toàn được phân tích. Vì một từ chỉ được dịch chính xác khi thông tin của các từ trước đó đã có đầy đủ. Hay nói cách khác, mức độ hiệu quả khi dịch từ ngữ dựa vào ngữ cảnh đã được hiểu.

- Phát sinh mô tả cho ảnh (Generating Image Descriptions):
  Mô hình này, kết hợp với mạng neural tích chập (Convolutional Neural Network - CNN), hoạt động bằng cách phân tích các đặc trưng rút trích được trong bức ảnh để tạo ra những câu mô tả thích hợp.

Việc huấn luyện một RNN, cũng tương tự như mạng bình thường, sẽ sử dụng lan truyền ngược như trên. Tuy nhiên, do bản chất có bộ nhớ

của RNN, quá trình lan truyền ngược này sẽ được thực hiện qua từng thời điểm trong mạng. Quá trình này được gọi là *lan truyền ngược liên hồi* (backpropagation through time). Mục tiêu là tính đạo hàm của lỗi với tham số (U, V, W) tương ứng, sau đó, học các tham số này bằng cách sử dụng gradient descent. Tương tự như việc cộng tổng các lỗi, ta cũng sẽ cộng tổng các đạo hàm tại mỗi bước cho mỗi mẫu huấn luyện: $\nabla e(\mathbf{w}) : \sum_t \frac{\partial e_t(\mathbf{w})}{\partial \mathbf{w}}$.

Áp dụng quy tắc chuỗi đạo hàm với từng trạng thái h, ta có:

$$\sum_t \frac{\partial e_t(\mathbf{w})}{\partial \mathbf{w}} \propto \sum_t \left(\prod_{i=k+1} \frac{\partial h_i}{\partial h_{i-1}} \right) \frac{\partial h_k}{\partial \mathbf{w}}.$$

Công thức trên tính lượng cập nhật tham số bằng đạo hàm tại thời điểm k tới thời điểm t. Tuy nhiên, một vấn đề được sinh ra từ công thức trên, cụ thể là ở phần $\frac{\partial h_i}{\partial h_{i-1}}$. Giả sử, trong trường hợp, $\left\| \frac{\partial h_i}{\partial h_{i-1}} \right\|_2 < 1$, giá trị cập nhật sẽ bị giảm về gần 0 rất nhanh. Ví dụ, nếu lượng cập nhật chỉ là 0.01, nhưng qua 100 thời điểm, thì giá trị trên sẽ là $0.01^{100} \approx 0$. Khi đạo hàm bằng 0, có nghĩa là xảy ra hiện tượng bão hòa. Lúc đó các nút phía trước cũng sẽ bị bão hoà theo. Vấn đề này không chỉ xảy ra với mạng RNN mà ngay cả mạng neural thường khá sâu cũng có hiện tượng này. Người ta gọi đây là hiện tượng *mất mát đạo hàm* (vanishing gradient).

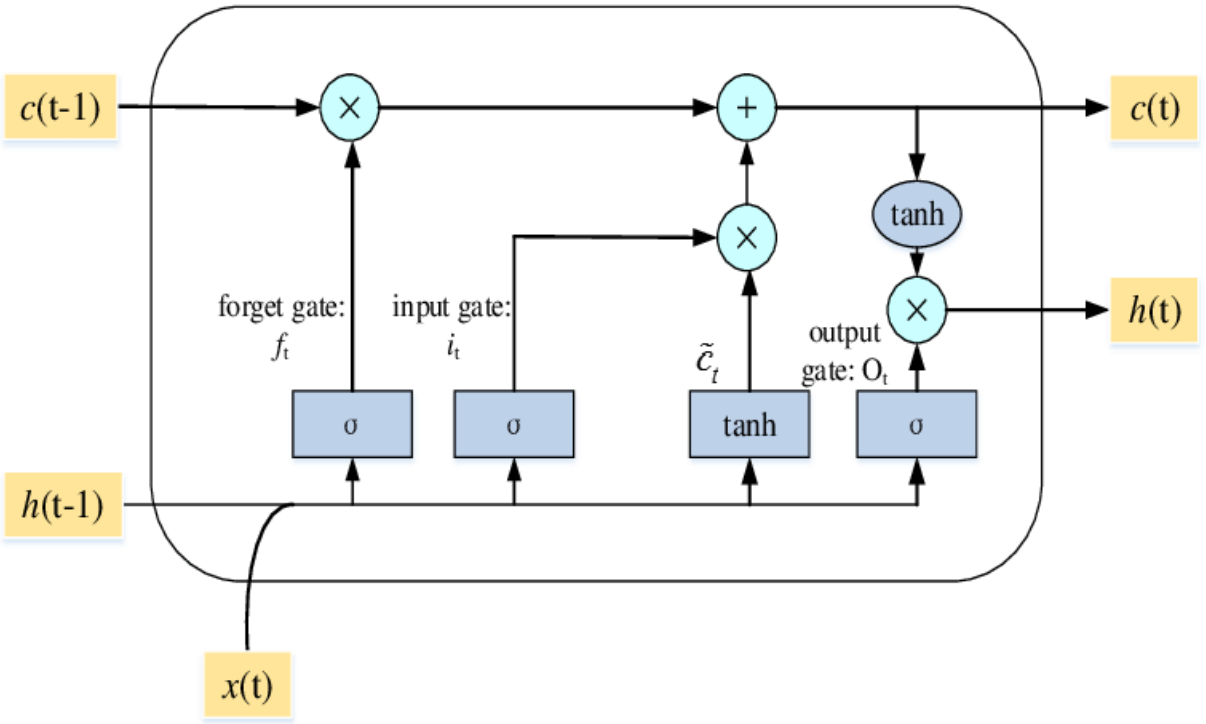
Một phương pháp phổ biến để giải quyết tình trạng mất mát đạo hàm là sử dụng kiến trúc mạng nhớ dài-ngắn hạn (Long Short-Term Memory - LSTM).

Long Short-Term Memory (LSTM)

LSTM là một mô hình cải tiến của RNN và có cấu trúc tương tự nhưng khác ở cách tính toán đối với các trạng thái ẩn. Ý tưởng của LSTM là kiểu bộ nhớ hạt nhân nhận thông tin đầu vào gồm trạng thái ẩn và giá trị. Bộ nhớ này quyết định thông tin nào cần lưu lại và thông tin nào cần xóa đi, nhờ đó, tất cả các bước của LSTM có thể truy vấn được thông tin từ

một tập thông tin lớn hơn. Do đó, LSTM có thể được ứng dụng cho vấn đề dự đoán chuỗi thời gian.

Mạng LSTM có cấu trúc mắt xích tương tự RNN, nhưng các thành phần đơn lặp có cấu trúc phức tạp hơn của RNN:



Hình 3.13: Cấu trúc của LSTM. [27]

Mấu chốt của khả năng "nhớ lâu" của LSTM là cấu trúc "trạng thái nhớ", là đường kẻ ngang phía trên trong thành phần đơn lặp ở hình 3.13. Ngoài ra, quá trình thêm hoặc loại bỏ thông tin sẽ dựa trên qui định của các cổng. Tóm lại, cốt lõi của mạng LSTM bao gồm trạng thái nhớ và các cổng.

Các cổng của LSTM là một phương pháp định nghĩa thông tin băng qua và chúng được tạo bằng hàm sigmoid σ. Cụ thể, hàm sigmoid có giá trị thuộc khoảng (0, 1) mang ý nghĩa độ lớn thông tin được phép truyền qua tại mỗi lớp mạng. Nếu kết quả là 0, điều này có nghĩa là không thông tin nào được phép đi qua, và ngược lại, 1 nghĩa là toàn bộ thông tin được đi qua.

Một mạng LSTM có 3 loại cổng, đều có đầu vào là trạng thái trước,

đặt tên là $S \in \{h, c\}$, và đầu vào, nhân với trọng số tương ứng. Lưu ý, các cổng sẽ nhận một loại trọng sô khác nhau:

- **Cổng quên (Forget gate)** $f_t = \sigma(W_f S_{t-1} + W_f X_t)$.
- **Cổng vào (Input gate)** $i_t = \sigma(W_i S_{t-1} + W_i X_t)$
- **Cổng ra (Output gat)** $o_t = \sigma(W_o S_{t-1} + W_o X_t)$

Sau đó, trạng thái nhớ trung gian có thể được tính bằng hàm tanh qua công thức $\tilde{C}_t = \tanh(W_c S_{t-1} + W_c X_t)$. Từ đây, **trạng thái nhớ** sẽ nhận trạng thái trung gian và trạng thái nhớ trước, kết hợp với cổng vào và cổng quên như sau:

$$c_t = (i_t \times \tilde{C}_t) + (f_t \times c_{t-1}). \tag{3.41}$$

Cuối cùng, trạng thái ẩn sẽ được tính bằng hàm tanh của trạng thái nhớ nhân với cổng ra:

$$h_t = o_t \times \tanh(c_t). \tag{3.42}$$

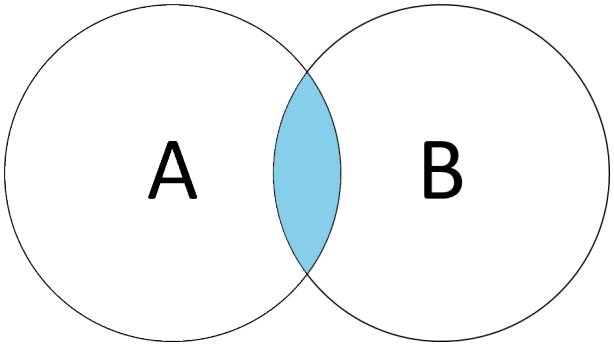
Lưu ý, phép nhân trong công thức 3.41 và 3.42 phụ thuộc vào S, tức là, các cổng sẽ thực hiện tính cho cả h và c, sau đó, thực hiện phép nhân tương ứng cho mỗi loại trạng thái.

3.2 Giới thiệu các thuật toán xử lí độ dài chuỗi thời gian

3.2.1 Time join

Time join (tạm dịch là nối thời gian) là một kỹ thuật cơ bản dùng để điều chỉnh độ dài chuỗi thời gian, với nguyên tắc loại bỏ các thời điểm không cùng thuộc cả hai chuỗi đang xét.

Nguyên tắc này hoàn toàn tương đương với inner join của SQL:



Hình 3.14: Minh họa inner join cho với hai bảng dữ liệu A, B. [28]

Time join là kỹ thuật thông dụng nhất, nhưng nó có thể làm suy giảm đáng kể dữ liệu gốc.

3.2.2 Delayed time join

Delayed time join (tạm dịch là nối trễ thời gian) có nguyên tắc tương tự như time join, với điểm khác biệt duy nhất là chuỗi thời gian được so sánh bị đẩy t "thời điểm" về trước (tức là, chuỗi bị làm trễ t giờ/ngày/... tùy theo chuỗi về quá khứ).

Với chuỗi thời gian là giá cổ phiểu, mục đích chính củ phương pháp này là để xác định cổ phiếu này có biểu diễn trạng thái của cố phiếu kia trong tương lai hay không.

3.2.3 Padding

Kỹ thuật đệm padding đơn giản chỉ kéo dài chuỗi thời gian ngắn hơn bằng cách lặp lại giá trị ban đầu của nó.

Ví dụ, kết quả sau khi thực hiện padding trên hai chuỗi $X_1 = (1, 6, 1, 2, 9, 9)$ và $X_2 = (9, 2, 4, 1)$ là: $\hat{X}_1 = (1, 6, 1, 2, 9, 9)$ và $\hat{X}_2 = (9, 9, 9, 2, 4, 1)$.

3.2.4 Perceptually important points (PIPs)

Theo bài nghiên cứu trong "Computational Finance and its Applications" [29], chuỗi thời gian được cấu tạo từ rất nhiều điểm dữ liệu, và một

tập hợp con của các điểm này sẽ hình thành nên xu hướng của chuỗi thời gian đó. Tuy nhiên, nếu ta tập trung quan sát vào một bộ phận điểm với số lượng thậm chí còn nhỏ hơn tập con trên, ta vẫn có thể xác định được xu hương hay chy kỳ của chuỗi thời gian ban đầu. Các điểm này được gọi là Perceptually Important Points (PIP(s), tạm dịch là các điểm có cảm giác quan trọng). PIP có thể được biểu diễn trong toán học là một tập $Z \subseteq Y \subseteq X$, trong đó, Y là tập hợp các điểm hình thành xu hướng của chuỗi thời gian ban đầu X.

Các PIPs được xác định, theo Hung, Ying Kit [30], bằng cách xác định mức ảnh hưởng của một điểm dữ liệu với hình dạnh của chuỗi thời gian. Tức là, một điểm có ảnh hưởng lớn hơn tới hình dạng của chuỗi thì sẽ được xem là quan trọng hơn.
Với chuỗi thời gian $P = (p_1, p_2 \ldots, p_m)$, ta có p_1 và p_m sẽ là hai PIPs đầu và cuối của P. PIP tiếp theo sẽ là một điểm p_i có khoảng cách cao nhất với hai PIP trên. PIP thứ tư sẽ là điểm p_j có khoảng cách cao nhất đến hai PIPs liền kề nó (giữa PIP đầu và PIP thứ 2, hoặc giữa PIP thứ 2 và PIP cuối). Quá trình này sẽ được lặp lại cho đến khi toàn bộ các điểm trong P được nhận hoặc đạt số lượng PIP mong muốn.

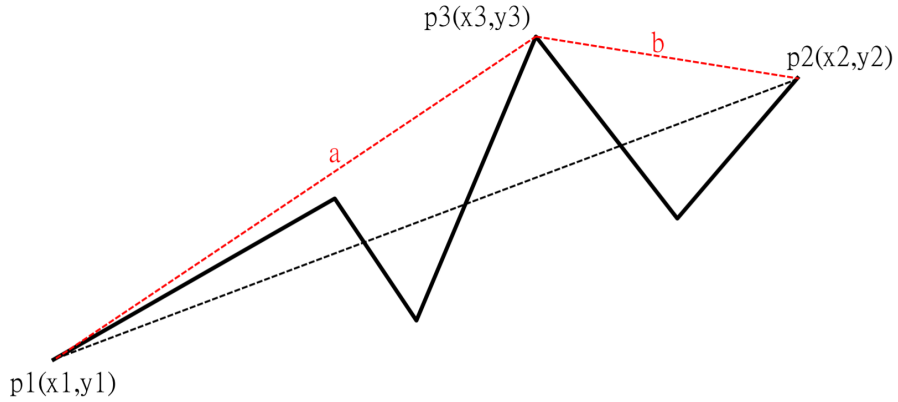
Ta nhận thấy, việc lựa chọn các PIPs như thế nào sẽ phụ thuộc vào hàm tính khoảng cách nào được sử dụng. Do đó, ta có ba loại PIP chính là:

- PIP-ED, với ED là khoảng cách Euclid (Euclidean Distance):
- PIP-PD, với PD là khoảng cách trực giao (Perpendicular Distance).
- PIP-VD, với VD là khoảng cách dọc (Verical Distance).

Sử dụng chuỗi thời gian P như trên, khoảng cách của điểm p_3 được xết đến hai PIPs kề p_1 và p_2 là:

- $ED(p_3, p_1, p_2)$ là tổng khoảng cách Euclid từ p_3 đến p_1 và p_2, tức là:

$$ED(p_3, p_1, p_2) = \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2} + \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2}.$$



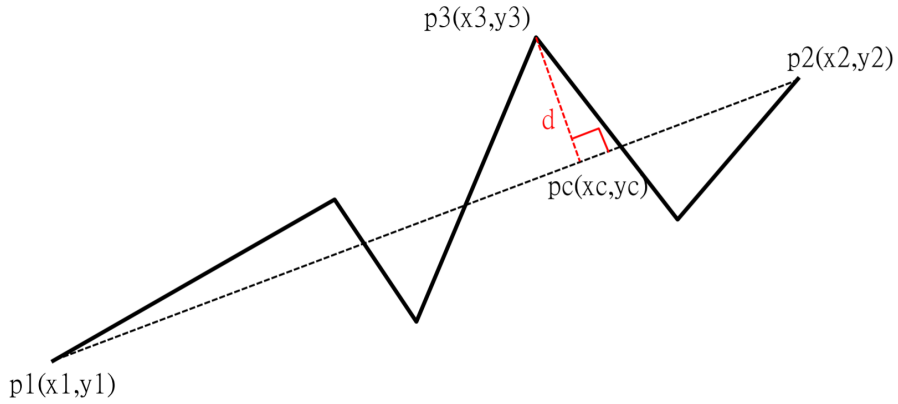
Hình 3.15: Khoảng cách Euclide của p_3 sẽ có giá trị là a + b. [30]

- Từ p_3, ta vẽ một đường vuông góc với đường thẳng đi qua hai điểm p_1 và p_2, cắt tại giao điểm là p_c, như vậy, $PD(p_3, p_1, p_2)$ sẽ là khoảng cách Euclid của hai điểm p_3 và p_c:

$$PD(p_3, p_1, p_2) = ED(p_3, p_c) = \sqrt{(x_3 - x_c)^2 + (y_3 - y_c)^2}.$$

  Trong đó, đặt $s = \dfrac{y_2 - y_1}{x_2 - x_1}$, ta có:

  - $x_c = \dfrac{x_3 + sy_3 - sy_2 + s^2 x_2}{1 + s^2}$.
  - $y_c = sx_c - sx_2 + y_2$.



Hình 3.16: Khoảng cách trực giao d của p_3 với p_1 và p_2. [30]

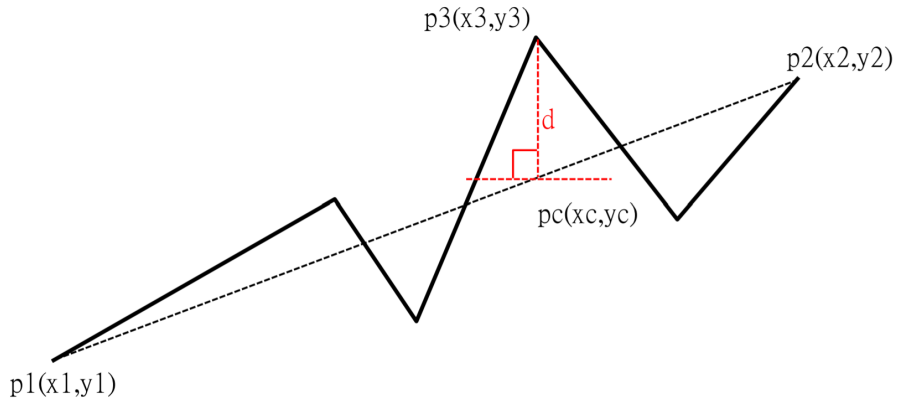
- Từ p_3, ta vẽ một đường thẳng đứng (song song với trục tung), cắt đường thẳng đi qua hai điểm p_1 và p_2 tại p_c, như vậy, $VD(p_3, p_1, p_2)$ là khoảng cách Euclid của hai điểm p_i và p_c:

$$VD(p_3, p_1, p_2) = ED(p_3, p_c) = \sqrt{(x_3 - x_c)^2 + (y_3 - y_c)^2}.$$

Do p_i và p_c cùng thuộc một đường thẳng song song với trục tung nên $x_3 = x_c$, suy ra:

$$VD(p_3, p_1, p_2) = ED(p_3, p_c) = |y_3 - y_c|.$$

Trong đó: $y_c = y_1 + (y_2 - y_1)\dfrac{x_3 - x_1}{x_2 - x_1}$.



Hình 3.17: Khoảng cách dọc d của p_3 với p_1 và p_2. [30]

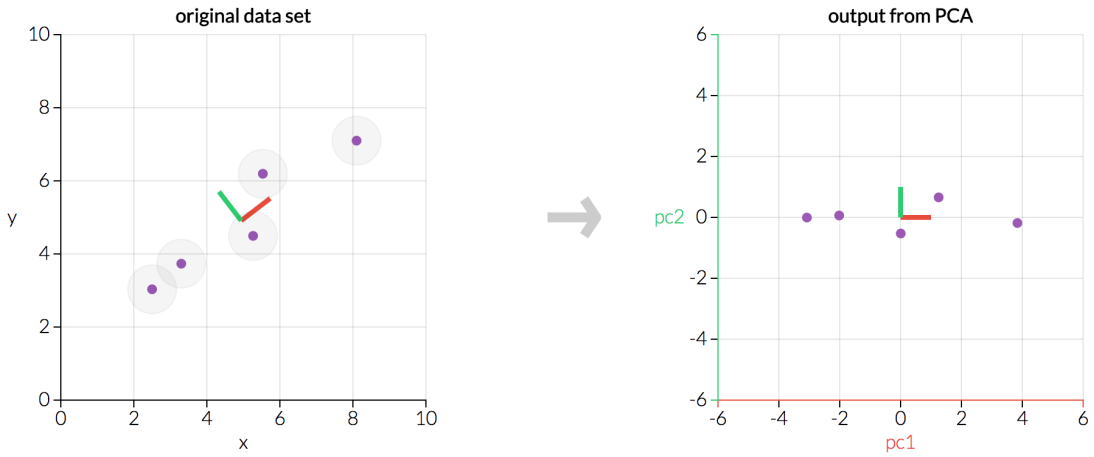
Trong vấn đề điều chỉnh độ dài chuỗi thời gian, phương pháp PIP sẽ được áp dụng để cùng chuyển đổi hai chuỗi ban đầu thành hai chuỗi có cùng độ dài là số lượng PIP mong muốn. Do bản chất của PIP, ta có thể an tâm rằng các đặc tính của hai chuỗi sẽ không bị mất đi hoàn toàn sau khi các điểm dữ liệu bị lược bỏ.

3.3 Giới thiệu các thuật toán phân khúc dữ liệu

3.3.1 Principals Component Analysis (PCA)

Các *thành phần chính* (principal component) của một tập hợp điểm trong không gian thực $\mathbb{R}^n$ là một dãy p vector đơn vị, sao cho vector thứ i chỉ hướng của đường "phù hợp nhất" (thường được biết đến là đường best fit). Vector này đồng thời cũng vuông góc với $i-1$ vector trước nó. Ở đây, đường best fit được hiểu là đường có trung bình khoảng cách từ các điểm đến nó nhỏ nhất [31].

Với định nghĩa trên, phép phân tích thành phần chính (Principals Component Analysis, thường được nhắc đến ở tên viết tắt là PCA), như tên, là một quá trình tính toán các thành phần chính và áp dụng chúng. PCA thường được sử dụng để khám phá và phân tích dữ liệu, nhưng phổ biến nhất là trong việc giảm chiều dữ liệu bằng cách chiếu mỗi điểm dữ liệu lên một vài thành phần chính đầu tiên. Quá trình này sinh ra một bộ dữ liệu có chiều thấp hơn ban đầu và bảo toàn độ biến thiên của dữ liệu nhiều nhất có thể. Thành phần chính thứ nhất có thể được định nghĩa là một chiều không gian mà của dữ liệu được chiếu có phương sai cao nhất. Và tuần tự, thành phần chính thứ i là một chiều không gian vuông góc với $i-1$ thành phần chính trước nó, nhưng vẫn ưu tiên tăng tối đa phương sai [31].



Hình 3.18: PCA thực hiện phép đổi cơ sở trên dữ liệu và chiếu dữ liệu lên cơ sở mới gồm hai chiều là hai thành phần chính đầu tiên. [32]

Nói cách khác, các thành phần chính có thể được hiểu là các *vector riêng* (eigenvector) của *ma trận hiệp phương sai* (covariance matrix) của dữ liệu (do bản chất tối đa phương sai của PCA). Do đó, tính toán thành phần chính thường sử dụng phương pháp phân rã giá trị riêng của ma trận hiệp phương sai của dữ liệu. Trước hết, ta cần giải thích một số định nghĩa mới:

- Vector riêng của 1 ma trận chỉ thay đổi độ dài mà không thay đổi hướng khi nhân với ma trận đó.
  Ví dụ, $\begin{pmatrix} 3 \\ 2 \end{pmatrix}$ là vector riêng của ma trận $\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix}$ do:

$$\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} 3 \\ 2 \end{pmatrix} = \begin{pmatrix} 12 \\ 8 \end{pmatrix} = 4 \begin{pmatrix} 3 \\ 2 \end{pmatrix}$$

  Trong ví dụ trên, hệ số vô hướng của vector riêng là 4. Ta gọi hệ số này *giá trị riêng* (eigenvalue), ký hiệu là λ.

- Ma trận hiệp phương sai của một tập hợp n biến ngẫu nhiên là một ma trận vuông $n \times n$, trong đó, các phần tử nằm trên đường chéo (từ trái sang phải, từ trên xuống dưới) lần lượt là phương sai tương

ứng của các biến, còn các phần tử còn lại là hiệp phương sai của đôi một từng cặp biến ngẫu nhiên khác nhau.
Ví dụ, với tập hợp $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$ có kỳ vọng của mỗi biến tương ứng là $\mu_1, \mu_2, \ldots, \mu_n$, ta có ma trận hiệp phương sai là:

$$\sum_{\mathbf{X}} = \begin{bmatrix} var(X_1) & cov(X_1, X_2) & \cdots & cov(X_1, X_n) \\ cov(X_2, X_1) & var(X_2) & \cdots & cov(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ cov(X_n, X_1) & cov(X_n, X_2) & \cdots & var(X_n) \end{bmatrix}$$

với $\sum_{\mathbf{X}ij} = cov(X_i, X_j) = \sum(X_i - \mu_i)(X_j - \mu_j)$ là hiệp phương sai của mỗi cặp biến. Tuy nhiên, do $var(X_i) = \sum(X_i - \mu_i)^2 = \sum(X_i - \mu_i)(X_i - \mu_i) = cov(X_i, X_i)$, ta có thể viết lại ma trận hiệp phương sai như sau:

$$\sum_{\mathbf{X}} = \begin{bmatrix} cov(X_1, X_1) & cov(X_1, X_2) & \cdots & cov(X_1, X_n) \\ cov(X_2, X_1) & cov(X_2, X_2) & \cdots & cov(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ cov(X_n, X_1) & cov(X_n, X_2) & \cdots & cov(X_n, X_n) \end{bmatrix}$$

PCA được tính bằng phương pháp phân rã giá trị riêng theo các bước sau [31]:

- Giả sử, ban đầu ta có bộ dữ liệu, đặt tên là $\mathbf{X}$, gồm n quan sát của d biến (n dòng, d cột).
  Ta có thể viết $\mathbf{X}$ dưới dạng một ma trận gồm n vector $x_1, x_2, \ldots, x_n$, mà mỗi x_i có vai trò là một nhóm quan sát của d cột. Tức là, dữ liệu ban đầu tương đương với:

$$\mathbf{X} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \in \mathbb{R}^{n \times d}.$$

- Ta tính kỳ vọng cho từng cột $j = 1, 2, \ldots, d$, và nhóm các kết quả thành một vector $\mu \in \mathbb{R}^{d\times 1} : \mu_j = \dfrac{1}{n}\sum_{i=1}^{n} x_{ij}$.

- **Chuẩn hóa dữ liệu** bằng cách trừ các giá trị ở mỗi cột cho kỳ vọng của cột tương ứng. Ta đặt tên ma trận mới này là $\mathbf{Z}$ và:

$$\mathbf{Z} = \mathbf{X} - h\mu^T \in \mathbb{R}^{n\times d}.$$

$$\text{Trong đó: } \begin{cases} \mathbf{X} \in \mathbb{R}^{n\times d} \\ h \in \mathbb{R}^{n\times 1} : h_j = 1 \\ \mu^T \in \mathbb{R}^{1\times d} \end{cases}$$

- **Tìm ma trận hiệp phương sai**, đặt lên là $\mathbf{C} \in \mathbb{R}^{d\times d}$ từ $\mathbf{Z}$.

- **Tính ma trận vector riêng và ma trận giá trị riêng**, đặt tên lần lượt là $\mathbf{V}$ và $\mathbf{D}$ thỏa: $\mathbf{V}^{-1}\mathbf{C}\mathbf{V} = \mathbf{D}$.
  Trong đó, ma trận $\mathbf{V} = (V_1, V_2, \ldots, V_d) \in \mathbb{R}^{d\times d}$ có d cột gồm các vector riêng V_j của $\mathbf{C}$. Còn ma trận $\mathbf{D} \in \mathbb{R}^{d\times d}$ là một ma trận chéo chứa các giá trị riêng của $\mathbf{C}$, được sắp xếp theo cặp với $\mathbf{V}$. Tức là, nếu gọi λ_j là giá trị riêng thứ j của $\mathbf{C}$, ta có:

$$\begin{cases} \mathbf{D}_{ij} = \lambda_j : j = i \\ \mathbf{D}_{ij} = 0 : j \neq i \\ \mathbf{C}\mathbf{V}_j = \lambda_j \mathbf{V}_j \end{cases} , j = 1, 2, \ldots, d.$$

- Ma trận vector riêng và ma trận giá trị riêng sau đó sẽ được sắp xếp lại theo chiều *giảm dần* của giá trị riêng.

- **Chọn tập con từ các vector riêng làm cơ sở mới**, giả sử, ta muốn giảm chiều dữ liệu lại thành K cột ($1 \leq K \leq d$). Ta có ma

trận $\mathbf{W} \in \mathbb{R}^{d \times K}$ gồm K cột là K vector riêng được chọn, sao cho:

$$\mathbf{W}_{lk} = \mathbf{V}_{kl}$$

với $l = 1, 2, \ldots, d$ và $k = 1, 2, \ldots, K$.

- **Kết quả** thu được là bộ dữ liệu mới được chuyển cơ sở:

$$\mathbf{T} = \mathbf{Z} \times \mathbf{W} \in \mathbb{R}^{n \times K}.$$

Trong đó, cột thứ nhất của $\mathbf{T}$ là kết quả chiếu lên thành phần chính thứ nhất, cột thứ hai là kết quả chiếu lên thành phần chính thứ hai và tương tự cho các cột sau (nếu có).

Để chọn được con số K thích hợp, ta cần phải xem xét đến mức độ bảo toàn dữ liệu khi thực hiện giảm chiều. Do giá trị riêng thể hiện độ phân chia "năng lượng" của dữ liệu gốc cho từng vector riêng, nên để xử lý vấn đề này, ta cần tính *năng lượng tích lũy* cho từng vector riêng, đặt tên là g. Năng lượng tích lũy của vector riêng thứ j là tổng các giá trị riêng từ 1 tới j:

$$g_j = \sum_{l=1}^{j} \mathbf{D}_{ll}, j = 1, 2, \ldots, d.$$

Sử dụng độ đo g để xác định K tốt nhất. Mục tiêu là chọn sao cho K bé nhất có thể nhưng phần trăm năng lượng tích lúy đặt được vẫn cao. Ví dụ, ta có thể chọn K sao cho g nằm trên một khoảng nhất định, chẳng hạn 90%. Cụ thể, ta chọn K bé nhất sao cho: $\frac{g_K}{g_d} \geq 0.9$ [31].

3.3.2 Symbolic ApproXimation Aggregation (SAX)

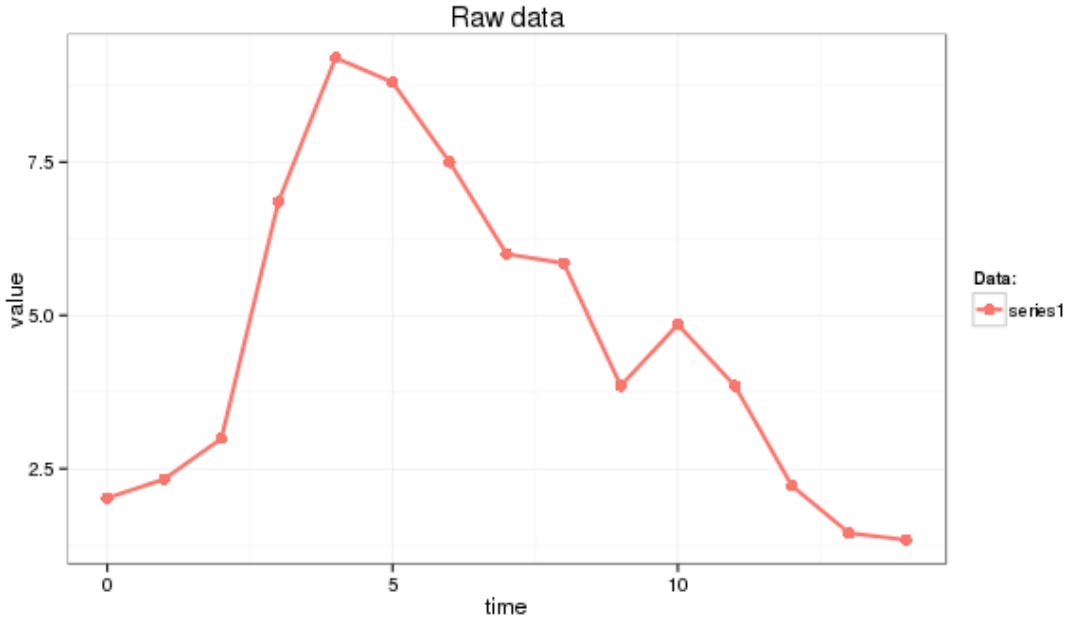
Trước hết, ta cần hiểu một khái niệm lớn là nền tảng của SAX: **PAA**.

Piecewise Aggregate Approximation (PAA)

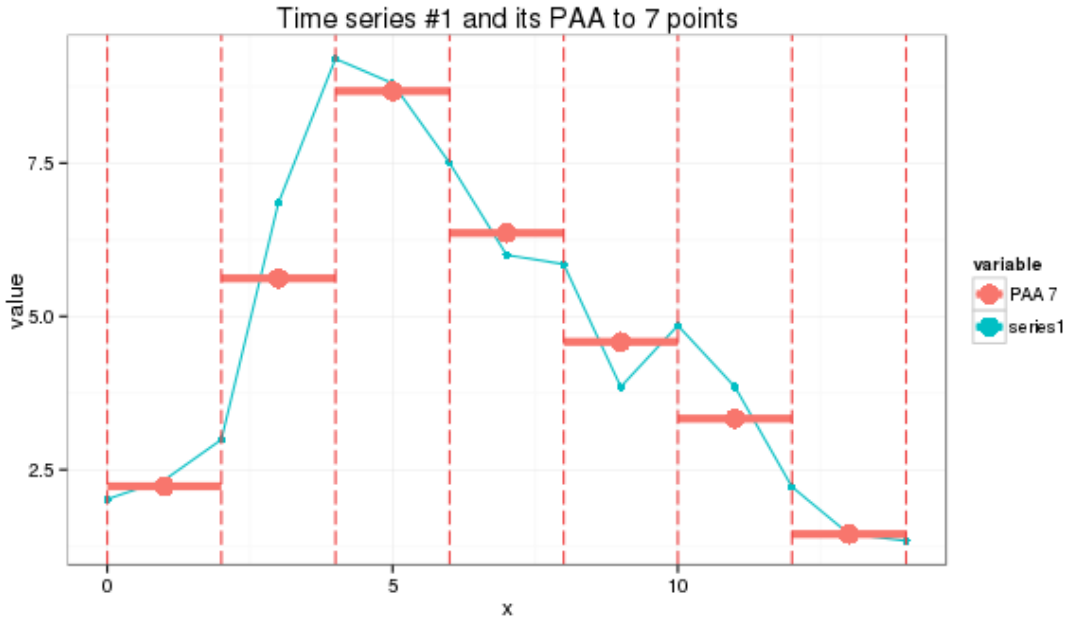
Tạm dịch là xấp xỉ tổng hợp theo từng mảnh, PAA là một phương thức giảm chiều cho chuỗi thời gian với nguyên lý sau:
PAA xấp xỉ chuỗi thời gian X có chiều dài n thành vector $\bar{X} = (\bar{x_1}, \ldots, \bar{x_M}), M \leq n$ với mỗi $\bar{x_i}$ được tính theo công thức [33]:

$$\bar{x_i} = \frac{M}{n} \sum_{j=n/M(i-1)+1}^{(n/M)i} x_j. \tag{3.43}$$

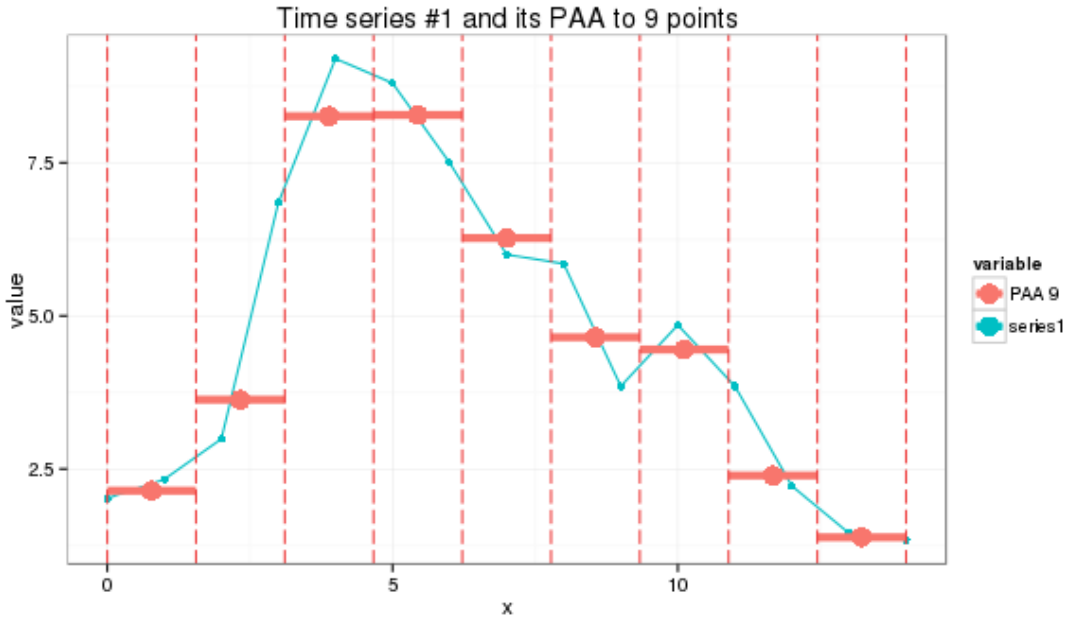
Tức là, PAA giảm chiều từ n thành M qua hai bước: một, chia chuỗi thời gian gốc thành M khung thời gian bằng nhau, và sau đó, hai, tính giá trị kỳ vọng cho từng khung.



Hình 3.19: Ví dụ minh họa chuỗi thời gian gốc. [33]



Hình 3.20: Kết quả PAA với $M = 7$. [33]



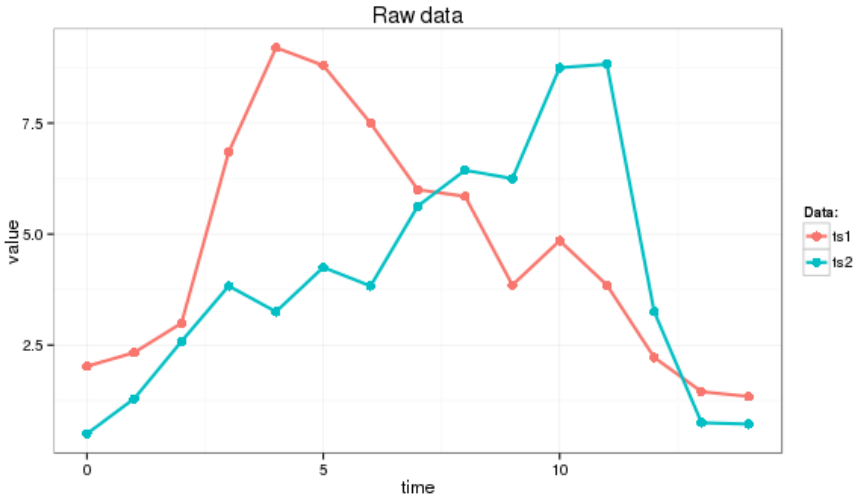
Hình 3.21: Kết quả PAA với $M = 9$. [33]

Symbolic ApproXimation Aggregation (SAX)

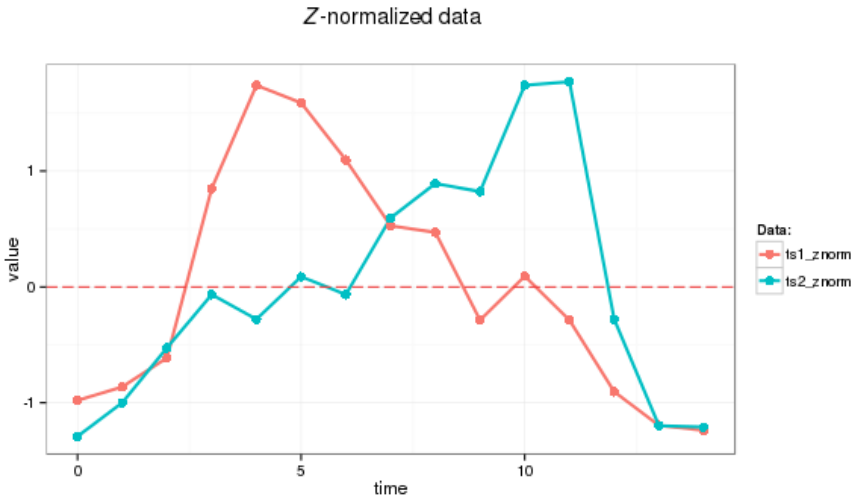
Tạm dịch là tổng hợp xấp xỉ theo ký hiệu, SAX là một kỹ thuật giảm chiều mà các giá trị được chuyển đổi thành các ký tự. Trong trường hợp chuỗi thời gian, thuật toán biến chuỗi X có chiều dài n thành chuỗi ký tự có chiều dài $\omega, \omega \leq n$, thông qua hai bước chính:

- Chuyển đổi dữ liệu gốc sử dụng PAA.
- Chuyển dữ liệu PAA thành các ký tự, thông thường bằng một bảng chữ cái có độ dài > 2.

Để đảm bảo quá trình rời rạc hóa từ PAA sang SAX của chuỗi thời gian diễn ra sao cho các ký tự được sinh ra, tương ứng với các đặc trưng của chuỗi thời gian, có xác suất bằng nhau, chuỗi thời gian ban đầu X sẽ được chuẩn hóa theo phân phối chuẩn tắc (phân phối chuẩn với kỳ vọng là 0 và độ lệch chuẩn là 1). Tức là, $Z = \dfrac{X - \mu_X}{\sigma_X} \sim N(0, 1)$.

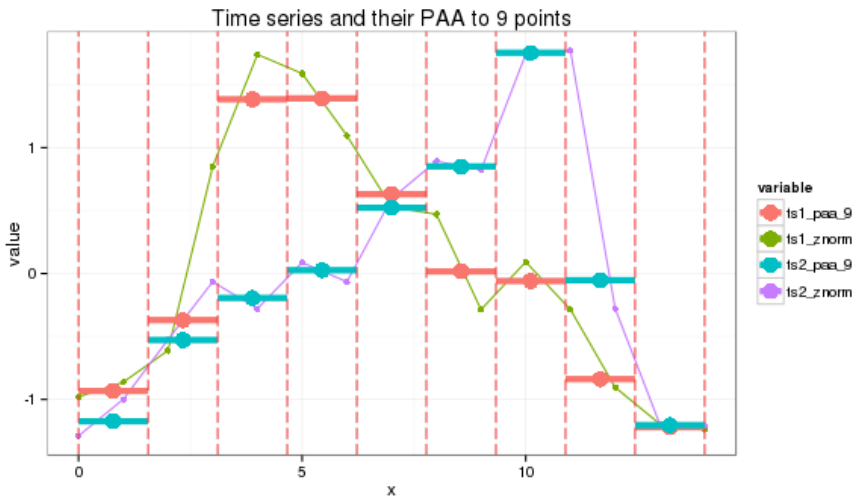


Hình 3.22: Biểu diễn hai chuỗi thời gian gốc. [34]



Hình 3.23: Chuẩn hóa hai chuỗi thời gian theo phân phối chuẩn tắc. [34]

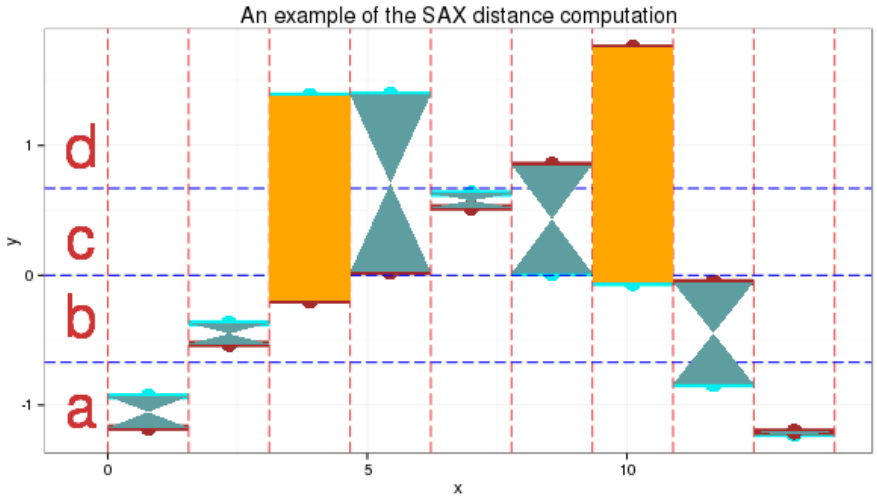
Sau đó, chuỗi thời gian được chia ra bởi các đường cắt có tên là *điểm dừng* (breakpoint) β. Danh sách các điểm dừng $B = \beta_1, \beta_2, \ldots, \beta_{a-1}$ này, với $\beta_{i-1} < \beta_i$ và $\beta_0 = -\infty, \beta_a = \infty$, cắt dữ liệu ra thành a đoạn bằng nhau.



Hình 3.24: Kết quả sau khi sử dụng PAA để chia thành 9 đoạn. [34]

Bằng cách gán một ký tự chữ cái α_j tương ứng với từng nửa khoảng $[\beta_{j-1}, \beta_j)$, ta có thể công thức hóa việc chuyển đổi của vector PAA sang chuỗi ký tự như sau [34]:

$$\bar{x} \in [\beta_{j-1}, \beta_j) \Rightarrow \hat{x}_i = \alpha_j. \tag{3.44}$$



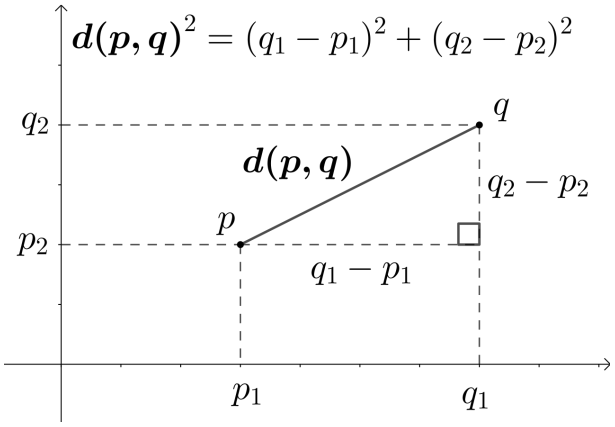
Hình 3.25: Ví dụ kết quả của SAX chuyển chuỗi thời gian thành chuỗi ký tự. [34]

Hình 3.25 cho biết kết quả của SAX chuyển đổi cho chuỗi thời gian *ts1* là "abddccbaa", và cho *ts2* là "abbccddba". Ngoài ra, trong hình còn có các vùng được tô đậm giữa hai giá trị tương ứng của hai chuỗi thời gian trên mỗi đoạn được chia, các vùng này biểu diễn "khoảng cách" giữa các ký tự. Độ đo khoảng cách này sẽ được đề cập sau.

3.4 Giới thiệu các thuật toán tính độ tương đồng

3.4.1 Khoảng cách Euclid

Trong Toán học, khoảng cách Euclid (Euclidean distance) giữa hai điểm chính là độ dài đoạn thẳng được nối bởi hai điểm đó trong một chiều không gian (hay còn gọi là *không gian Euclid*). Khoảng cách này có thể được tính bằng tọa độ của mỗi điểm trong hệ trục tương ứng bằng định lý Pythago. Do đó, khoảng cách Euclid còn có tên gọi khác là khoảng cách Pythago [35].



Hình 3.26: Sử dụng định lý Pythago, tính khoảng cách Euclid của hai điểm p, q trên mặt phẳng 2 chiều. [35]

Công thức tính khoảng cách như sau, với hai điểm p, q [35]:

- Ở không gian 1 chiều, do khoảng cách giữa hai điểm bất kỳ trên trục số thực chính là trị tuyệt đối của hiệu tọa độ hai điểm đó nên, nếu p, q là hai điểm nằm trên trục số thực, thì khoảng cách của chúng sẽ là:

$$d(p,q) = |p - q| = \sqrt{(p-q)^2} \tag{3.45}$$

- Ở không gian 2 chiều, ta đặt tọa độ của hai điểm p, q lần lượt là (p_1, p_2) và (q_1, q_2). Sử dụng định lý Pythago, ta có khoảng cách Euclid của p, q là:

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2} \tag{3.46}$$

- Ở không gian n chiều, từ hai công thức trên, ta có thể tổng quát hóa công thức tính khoảng cách Euclid thành:

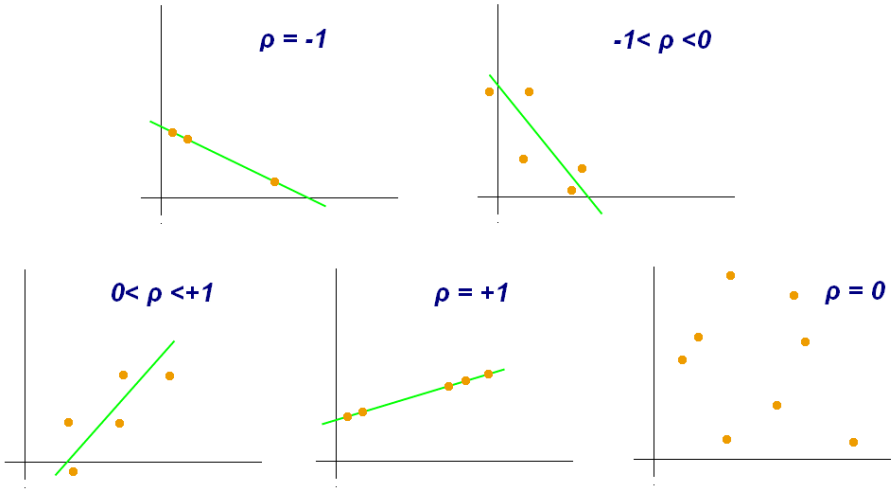
$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2} \tag{3.47}$$

Nếu ta xem xét một chuỗi thời gian $X = (x_1, x_2, \ldots, x_m)$ nào đó với m số lần quan sát là một điểm trong không gian, với tập tọa độ có giá

trị tương ứng, thì công thức tính khoảng cách Euclid (3.47) sẽ được sử dụng để tính **"độ tương đồng"** (hay thực chất là khoảng cách ở trường hợp này) giữa hai chuỗi thời gian. Ta có thể áp dụng nguyên tắc này để sử dụng các thuật toán tính khoảng cách thông thường khác hoặc tính độ tương quan của hai mẫu trong việc tính toán độ tương đồng.

3.4.2 Hệ số tương quan Pearson

Trong Thống kê, hệ số tương quan Pearson (Pearson correlation coefficient, hay còn có tên khác là Pearson's R, ký hiệu tương ứng là r) là đại lượng tính độ tương quan tuyến tính giữa hai tập dữ liệu. Đại lượng này được tính bằng cách lấy thương của hiệp phương sai của hai tập, chia cho *tích Descartes* độ lệch chuẩn của chúng, trong đó, tích Descartes của hai tập A, B là $A \times B = \{(a, b)|a \in A, b \in B\}$. Vì thế, giá trị của hệ số luôn thuộc đoạn $[-1, 1]$ do phép tính tương đương với việc chuẩn hóa hiệp phương sai [36].



Hình 3.27: Một số ví dụ của giá trị hệ số tương quan (ρ) biểu diễn trên biểu đồ phân tán. [36]

Xét hai chuỗi thời gian $X = (x_1, x_2, \ldots, x_n)$ và $Y = (y_1, y_2, \ldots, y_n)$, ta có bộ dữ liệu tương ứng là $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$. Dựa theo định

nghĩa trên, ta có hệ số tương quan Pearson của X và Y là [36]:

$$r_{XY} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \tag{3.48}$$

Mà: $\begin{cases} cov(X,Y) = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) \\ \sigma_X = \sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2} \\ \sigma_Y = \sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2} \end{cases}$

Từ đó, công thức (3.48) trở thành:

$$r_{XY} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{3.49}$$

với $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ là kỳ vọng của X, tương tự với $\bar{y}$.

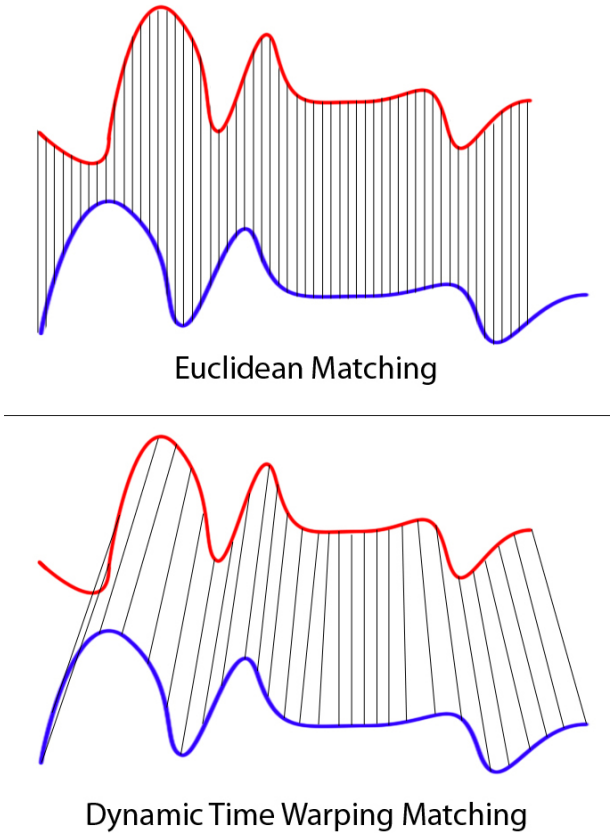
Một độ đo có tên là *khoảng cách Pearson* (Pearson's distance) dùng để đo khoảng cách giữa hai bộ X, Y có công thức như sau [36]:

$$d_{XY} = 1 - r_{XY}.$$

, với $d \in [0, 2]$ do $r \in [-1, 1]$.

3.4.3 Dynamic Time Warping

Dynamic time warping (DTW) là một thuật toán phổ biến dùng để tìm mối liên kết tối ưu giữa hai dãy thời gian. Trong lĩnh vực phân tích chuỗi thời gian, DTW được sử dụng để tính độ tương đồng giữa hai chuỗi. Ngoài ra, DTW còn có ứng dụng trong các mảng như nhận diện tiếng nói (automatic speech recognition) hay nhận diện chữ ký online.



Hình 3.28: Minh họa so sánh liên kết giữa hai chuỗi thời gian bằng khoảng cách Euclid và DTW. [37]

Theo nghiên cứu của Meinard Müller [38], với hai chuỗi thời gian $X = (x_1, x_2, \ldots, x_n)$ và $Y = (y_1, y_2, \ldots, y_m)$, ta có một *không gian đặc trưng* (feature space) ký hiệu là $\mathcal{F}$ sao cho $x_i, y_j \in \mathcal{F}$ với $i \in [1, n], j \in [1, m]$. Để so sánh hai đặc trưng $x, y \in \mathcal{F}$ bất kỳ, ta cần một hàm c để tính *chi phí* (cost), được định nghĩa như sau:

$$c : \mathcal{F} \times \mathcal{F} \to \mathbb{R}_{\geq 0}. \tag{3.50}$$

Chi phí này sẽ được tính trên từng cặp thuộc X, Y, và hình thành một *ma trận chi phí* (cost matrix) $C \in \mathbb{R}^{n \times m}$, với $C(i, j) = c(x_i, y_j)$. Từ đây, mục tiêu của ta là tìm một liên kết giữa X và Y sao cho nó có chi phí **nhỏ nhất**. Liên kết này chính là liên kết tối ưu cần tìm, và nó sẽ chạy dọc theo một "thung lũng" gồm các chi phí nhỏ thuộc ma trận C. Người ta đặt tên cho nó là "warping path" (tạm dịch là đường cong vênh).

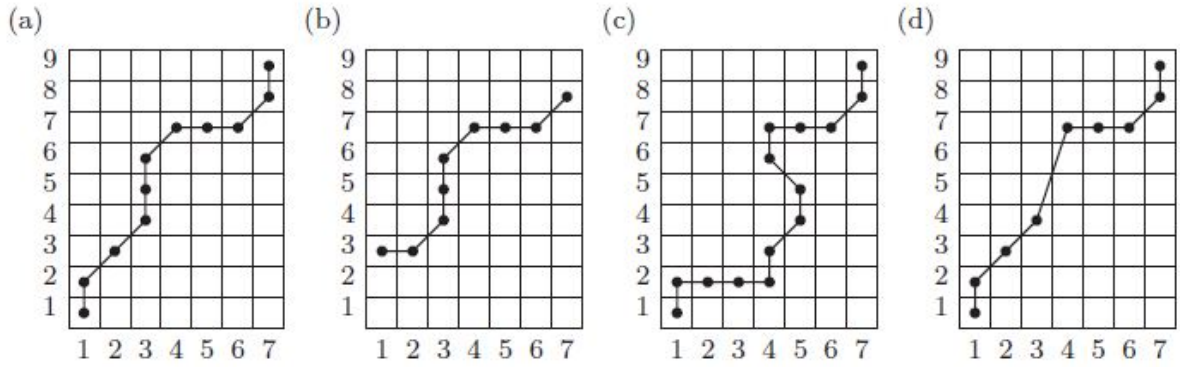
Đường này có thể được định nghĩa hoàn thiện như sau: Một đường

cong vênh là một dãy $p = (p_1, p_2, \ldots, p_L)$ với $p_l = (i_l, j_l) \in [1, n] \times [1, m]$ và $l \in [1, L]$ và phải thỏa các điều kiện sau:

- (i) Điều kiện ranh giới (Boundary condition): $p_1 = (1, 1)$ và $p_L = (n, m)$.
  Điều kiện ranh giới bảo đảm đường cong vênh bắt đầu ở điểm đầu ma trận, và tương ứng kết thúc ở điểm cuối.

- (ii) Điều kiện đơn điệu (Monotonicity condition): $i_1 \leq i_2 \leq \cdots \leq i_L$ và $j_1 \leq j_2 \leq \cdots \leq j_L$.
  Điều kiện này bảo toàn thứ tự thời gian theo chiều đi tới.

- (iii) Điều kiện liên tục (Step size condition): với $l \in [1, L-1]$, $p_{l+1} - p_l \in \{(0, 1), (1, 0), (1, 1)\}$
  hay: $i_{l+1} - i_l \leq 1$ và $j_{l+1} - j_l \leq 1$.
  Điều kiện này hạn chế bước chuyển tiếp của đường cong vênh chỉ sang các ô liền kề (không nhảy vượt thời gian).



Hình 3.29: Minh họa một số *đường cong vênh* trên ma trận chi phí DTW. [38]

Hình 3.29 gồm 4 ví dụ của đường cong vênh. Tuy nhiên, ta nhận thấy ví dụ (b) vi phạm điều kiện (i), ví dụ (c) vi phạm điều kiện (ii) và ví dụ (d) vi phạm điều kiện (iii). Còn lại, ví dụ (a) là một trường hợp đúng (thỏa toàn bộ điều kiện) của một đường cong vênh.

Dựa theo lý thuyết, thuật toán DTW có thể được cài đặt như sau [39]:

```python
import numpy as np


def dtw(X, Y):
    n, m = len(X), len(Y)

    # Tao ma tran chi phi
    dtw_matrix = np.zeros((n + 1, m + 1))
    for i in range(n + 1):
        for j in range(m + 1):
            dtw_matrix[i, j] = np.inf
            dtw_matrix[0, 0] = 0

    # Tinh chi phi cho tung o cua ma tran
    for i in range(1, n + 1):
        for j in range(1, m + 1):
            cost = abs(X[i - 1] - Y[j - 1])
            step = np.min([dtw_matrix[i - 1, j], dtw_matrix[i, j - 1],
    dtw_matrix[i - 1, j - 1]])
            dtw_matrix[i, j] = cost + step

    return dtw_matrix
```

Trong đó:

- **cost** là hàm chi phí c nói trên, và được tính bằng trị tuyệt đối hiệu (hay khoảng cách Euclid ở không gian 1 chiều) giữa 2 điểm quan sát tương ứng của X và Y.
- **step** xét chi phí ở 3 ô liền kề (ràng buộc bởi điều kiện (ii) và (iii)) và tìm ô có chi phí nhỏ nhất.
- **dtw_matrix[i , j]** là tổng của hai giá trị trên.

Các bước này sẽ thực hiện tuần tự tới hết ma trận. Kết quả thu được ma trận chi phí C, có điểm kết thúc chứa tổng chi phí của tất cả các bước đi mang chi phí thấp nhất. Giá trị này được gọi là khoảng cách DTW (DTW distance), và cũng được xem xét để đánh giá độ tương đồng giữa hai chuỗi thời gian sử dụng thuật toán DTW.

3.4.4 SAX MINDIST

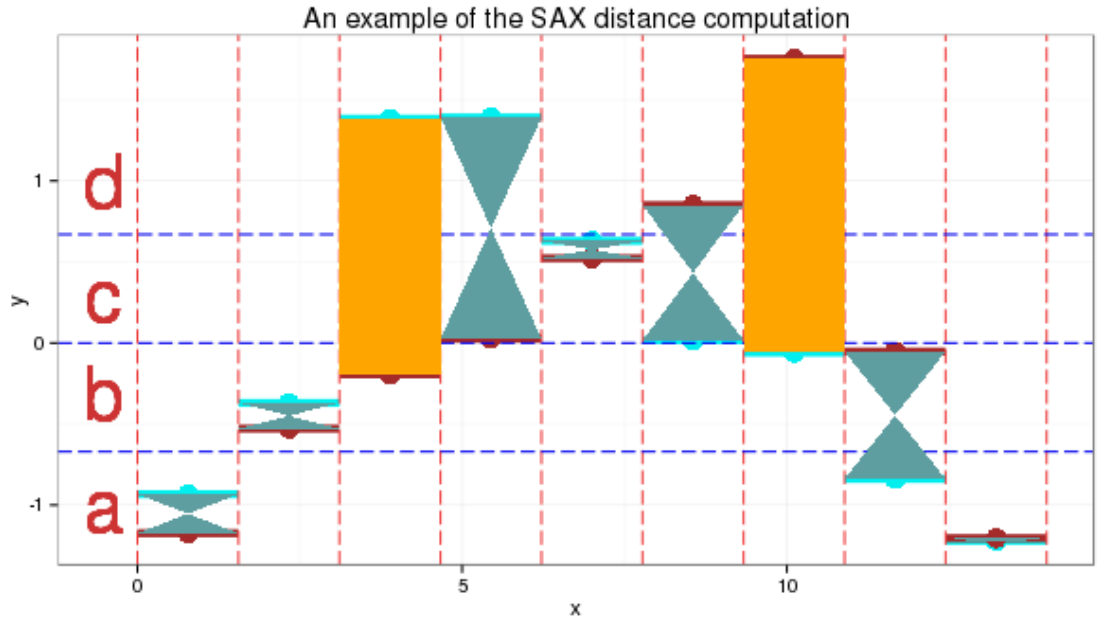
SAX MINDIST là một độ đo thuộc SAX (xem lại 3.3.2) dùng để tính khoảng cách giữa hai ký tự sau khi chuyển đổi. Kết quả sẽ là khoảng cách **nhỏ nhất** giữa hai chuỗi ký tự sau khi áp dụng SAX của hai chuỗi thời gian. MINDIST có công thức như sau [34]:

$$MINDIST(X^{SAX}, Y^{SAX}) = \sqrt{\frac{n}{\omega}}\sqrt{\sum_{i=1}^{\omega}(dist(x_i^{SAX}, y_i^{SAX}))^2}. \qquad (3.51)$$

Trong đó:

- X, Y là hai chuỗi thời gian và X^{SAX}, Y^{SAX} tương ứng là kết quả SAX của chúng.
- n là độ dài của X, Y và ω là độ dài của X^{SAX}, Y^{SAX}.
- $dist(x_i^{SAX}, y_i^{SAX})$ là khoảng cách giữa từng cặp ký tự tương ứng thuộc X^{SAX}, Y^{SAX}. *dist* sẽ có kết quả dựa theo một *bảng tra cứu* (lookup table). Bảng này có liên hệ đến phân phối chuẩn tắc và sẽ thay đổi tùy vào **kích thước của bảng chữ cái**, tuy nhiên, ta có thể tính giá trị mỗi *ô* (cell) dòng r, cột c theo công thức như sau:
  - $cell(r, c) = 0$, nếu $|r - c| \leq 1$ (tức là, khoảng cách bằng 0 nếu hai ký tự giống nhau hoặc liền kề nhau).
  - $cell(r, c) = \beta_{max(r,c)-1} - \beta_{min(r,c)-1}$, với β là *điểm dừng* (xem lại 3.3.2).

Ở ví dụ hình 3.25, phần 3.3.2, các vùng được tô đậm giữa hai giá trị tương ứng của hai chuỗi thời gian trên mỗi đoạn được chia như sau:



Kết quả SAX của hai chuỗi thời gian *ts1* và *ts2* trong ví dụ trên lần lượt là "abddccbaa" và "abbccddba", với $n = 15, \omega = 9$ và bảng chữ cái tương ứng là $\{a, b, c, d\}$.

Bảng tra cứu khoảng cách cho bảng chữ cái với kích thước = 4 là:

	a	b	c	d
a	0	0	0.67	1.34
b	0	0	0	0.67
c	0.67	0	0	0
d	1.34	0.67	0	0

Trong 9 đoạn được chia, có các đoạn được tô đậm bằng hai tam giác đối nhau (vị trí số 1, 2, 4, 5, 6, 8 và 9), khoảng cách giữa hai ký tự tại các đoạn này bằng 0. Vì dựa theo công thức, các ký tự trùng nhau hoặc kề nhau trong bảng chữ cái sẽ có $dist = 0$. Vậy ở cả hai đoạn còn lại, ta có: $dist(b, d) = cell(1, 3) = 0.67$.

Như thế, kết quả MINDIST của ví dụ phần SAX, mục 3.3.2 là:

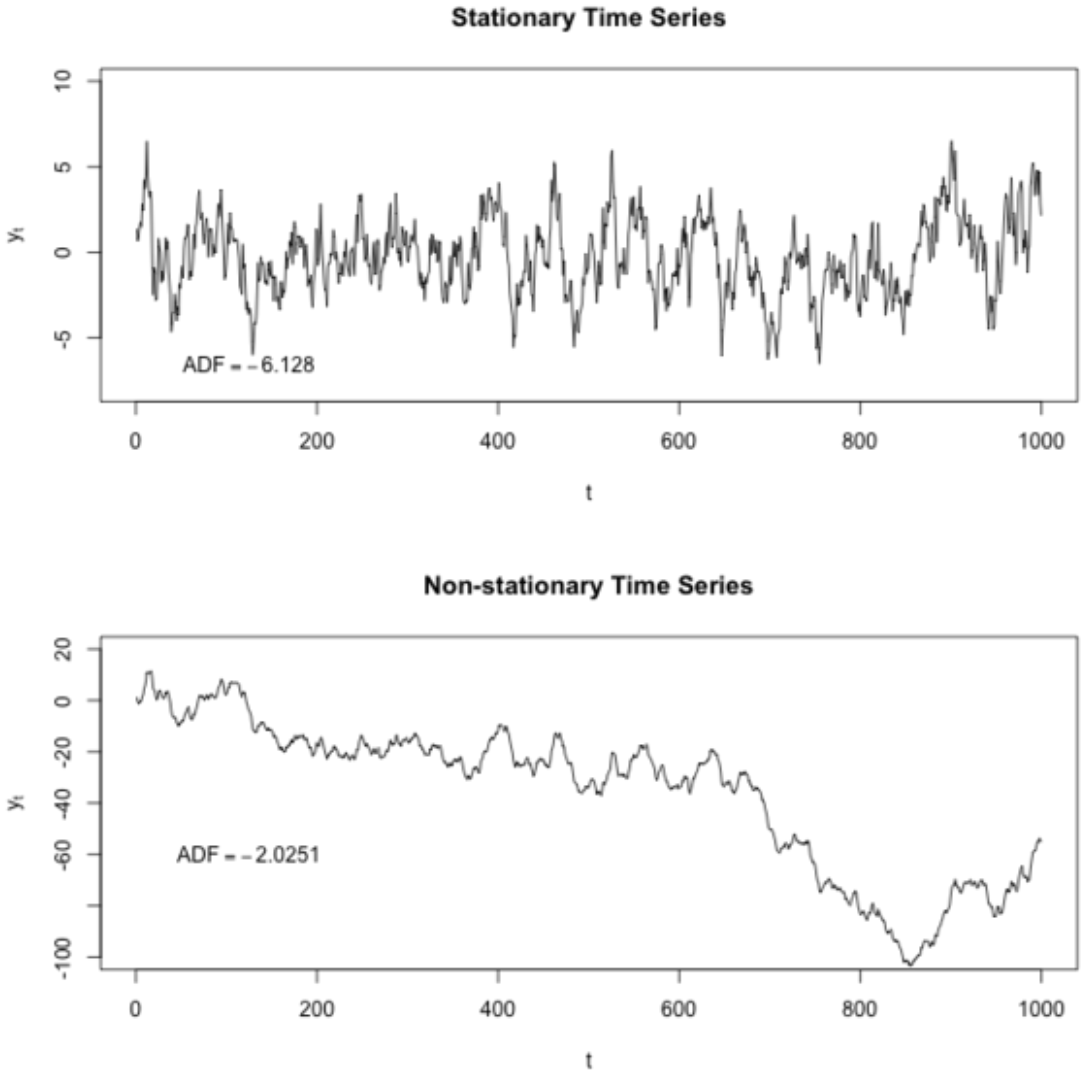
$$MINDIST(ts1^{SAX}, ts2^{SAX}) = \sqrt{\frac{15}{9}}\sqrt{0.67^2 + 0.67^2} \approx 1.2232.$$

3.4.5 Cointegration

Đồng liên kết (thường được biết đến qua tên tiếng Anh là Cointegration) là một thuộc tính thống kê, hay trong lĩnh vực kinh tế, là một phương pháp xác định mối quan hệ dài hạn, của một tập hợp các biến chuỗi thời gian $(X_1, X_2, \ldots, X_k)$.

Trước khi tìm hiểu sau về Cointegration, ta cần nắm sơ một vài khái niệm:

- Tính dừng của chuỗi thời gian:
  Một *quá trình có tính dừng nghiêm ngặt* (strict/strictly stationary process hoặc strong/strongly stationary process) nếu phân phối xác suất của nó không thay đổi theo thời gian. Hệ quả thu được các tham số như kỳ vọng hay phương sai đều không thay đổi theo thời gian.
  Một ví dụ đơn giản của quá trình dừng nghiêm ngặt là *"White noise"*. [40].



Hình 3.30: Minh họa chuỗi thời gian có tính dừng. [41]

Trong thực nghiệm, tính dừng của chuỗi được xem xét ở mức nhẹ hơn, có tên là *dừng nhẹ* (weak-sense/wide-sense stationary - WSS) hay tên khác là *dừng hiệp phương sai* (covariance stationary). Một chuỗi X_t có tính dừng hiệp phương sai nếu:

- Tồn tại $E(X_t) = \mu, Var(X_t) = \sigma^2$ xác định, tức là độc lập theo thời gian.
- Giá trị hiệp phương sai phụ thuộc vào thời điểm quan sát và độc lập theo thời gian, tức là: $Cov(X_t, X_{t-s}) = Cov(X_{t+k}, X_{t-s+k})$.

- Thứ tự liên kết:
  Trong thống kê, thứ tự liên kết (Order of Integration, ký hiệu $I(d)$) của một chuỗi thời gian là số lần lấy hiệu nhỏ nhất cần thiết để đạt trạng thái dừng hiệp phương sai. Vậy theo định nghĩa, các chuỗi dừng nghiêm ngặt luôn có $I(0)$ (tuy nhiên, các chuỗi có $I(0)$ không hoàn toàn dừng nghiêm ngặt).
  Cụ thể, với chuỗi X_t, ta lấy hiệu giữa các điểm quan sát và xem đây là chuỗi ΔX_t. Nếu ΔX_t đạt tính dừng hiệp phương sai, ta gọi X_t có $I(1)$. Nếu không, ta tiếp tục lấy hiệu $\Delta(\Delta X_t)$ và kiểm tra tương tự để kết luận cho $I(2)$. Trong thực tế, các chuỗi thời gian rất ít khi có thứ tự liên kết $I(d)$ với $d > 2$.
  Trong thực nghiệm, để xác định thứ tự liên kết của chuỗi thời gian, người ta thường sử dụng hai phương pháp là kiểm tra Dickey–Fuller và kiểm tra Dickey–Fuller mở rộng (xem [42] và [43]). Các phương pháp kiểm tra này được đặt tên là *phép kiểm gốc đơn vị* (unit root test). Ta cũng có thể gọi các chuỗi hay quá trình có tính dừng là *biến gốc đơn vị* (unit root variable).

Cointegration sẽ phân tích hai tập hợp chuỗi thời gian hoặc quá trình không có tính dừng và kiểm tra xem bộ kết hợp tuyến tính của chúng có thứ tự liên kết thấp hơn không. Nếu có, kết luận rằng chúng đồng liên kết (cointegrated). Ví dụ, đồng liên kết xảy ra nếu một tập hợp biến $I(1)$ có thể được kết hợp tuyến tính thành $I(0)$.

Cointegration bao gồm nhiều các phương thức khác nhau, trong đó, phép kiểm định Engle-Granger là phổ biến nhất. Phương pháp này xem xét trường hợp tồn tại duy nhất một vector đồng liên kết, với hai giả thuyết là:

- H_0 : Không đồng liên kết.

Giả thuyết không (null hypothesis).

- H_1 : Tồn tại đồng liên kết.

Giả thuyết khác (alternative hypothesis).

Nguyên lý chính của phép kiểm định Engle-Granger là nếu các biến đồng liên kết, thì phần dư của quá trình hồi quy đồng liên kết phải có tính dừng. Phần dư này phụ thuộc vào vector đồng liên kết có sẵn hay phải được ước lượng:

- Nếu vector đồng liên kết có sẵn, phần dư có thể được tính trực tiếp bằng công thức: $r_t = \beta X_t$.
- Nếu vector đồng liên kết chưa có sẵn, phần dư sẽ được xác định thông qua mô hình hồi quy OLS (Ordinary least squares, tạm dịch là bình phương nhỏ nhất thông thường, xem tại [44]):

$$\hat{r_t} = x_{1t} - \hat{\beta}_1 - \hat{\beta}_2 y_{2t}$$

Phần dư sẽ được kiểm tra tính dừng bằng bất kỳ phương pháp kiểm gốc đơn vị chuẩn nào (như ADF hoặc DF ở trên).

Trong việc nhận xét độ tương đồng của hai chuỗi thời gian, ta sẽ sử dụng tham số *p-value* của phép kiểm định conintegration bằng phương pháp Engle-Granger. Trong thống kế, nếu p-value càng bé thì giả thuyết khác (H_1) càng thiên về hướng là giả thuyết đúng. Nói cách khác, ta sử dụng p-value với nguyên tắc nếu p-value bé hơn thì hai chuỗi thời gian được xét có độ tương đồng cao hơn.

Chương 4

Phương pháp đề xuất: Áp dụng tương đồng trên chuỗi thời gian trong huấn luyện mô hình máy học

4.1 Giới thiệu: Kết hợp các chuỗi tương đồng trong huấn luyện

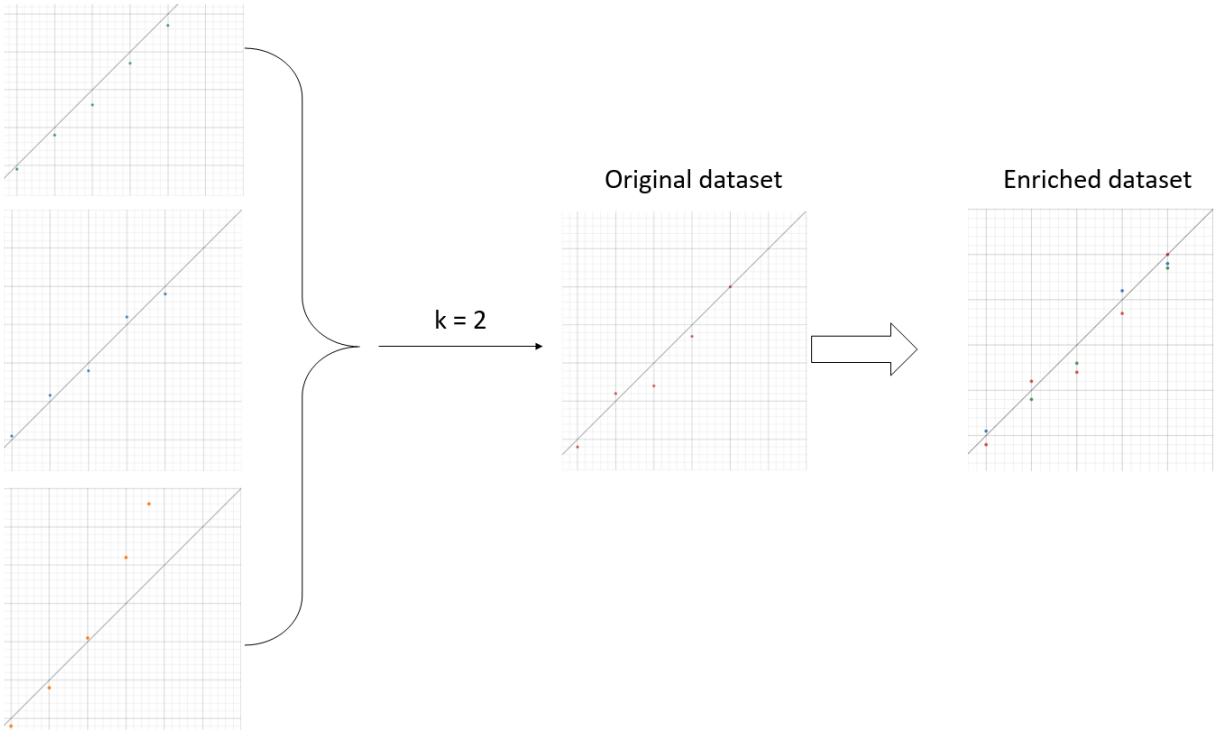
Từ những cơ sở lý thuyết trên, việc kết hợp nhiều chuỗi thời gian tương đồng vào dữ liệu huấn luyện được thực hiện như sau:

Với thuật toán xử lí độ dài 2 chuỗi LF và thuật toán tương đồng S,

- Độ dài 2 chuỗi thời gian được cố định bằng thuật toán L, sau đó độ tương đồng của 2 chuỗi sẽ được tính bằng thuật toán S.
- Thực hiện tính tương đồng giữa chuỗi thời gian của cổ phiếu cần dự đoán với các cổ phiếu còn lại trong tập dữ liệu.

- Chọn k cổ phiếu có độ tương đồng cao nhất, các điểm dữ liệu (đã tiền xử lí) của các cổ phiếu này được thêm vào dữ liệu huấn luyện của cổ phiếu cần dự đoán bằng phương pháp lấy mẫu có trọng số (weighted sampling).

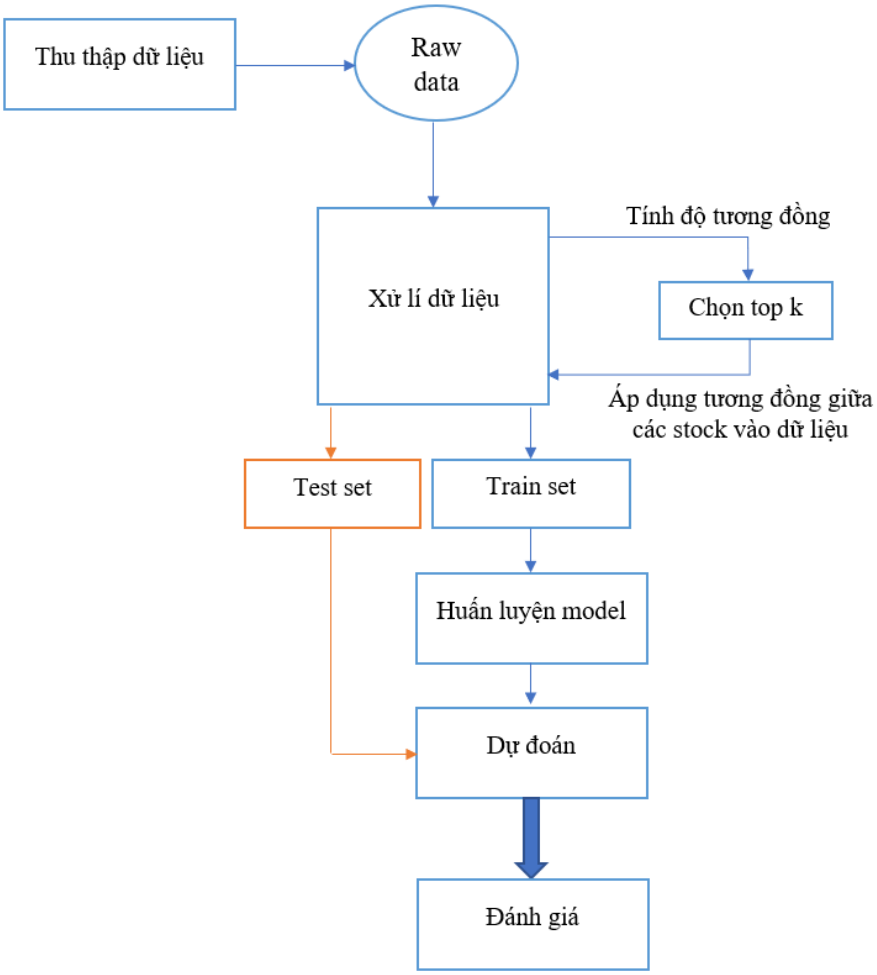
Vậy, tập huấn luyện cuối cùng là tập huấn luyện được kết hợp các chuỗi tương đồng sử dụng cặp thuật toán S-LF.



Hình 4.1: Minh họa: Kết hợp nhiều chuỗi thời gian trong huấn luyện

4.2 Tổng quan quy trình chương trình

Nhóm thiết kế quy trình thực hiện các bước thu thập, tiền xử lí, tính độ tương đồng, tạo tập huấn luyện áp dụng k cổ phiếu tương đồng nhất, huấn luyện, dự đoán và đánh giá mô hình.



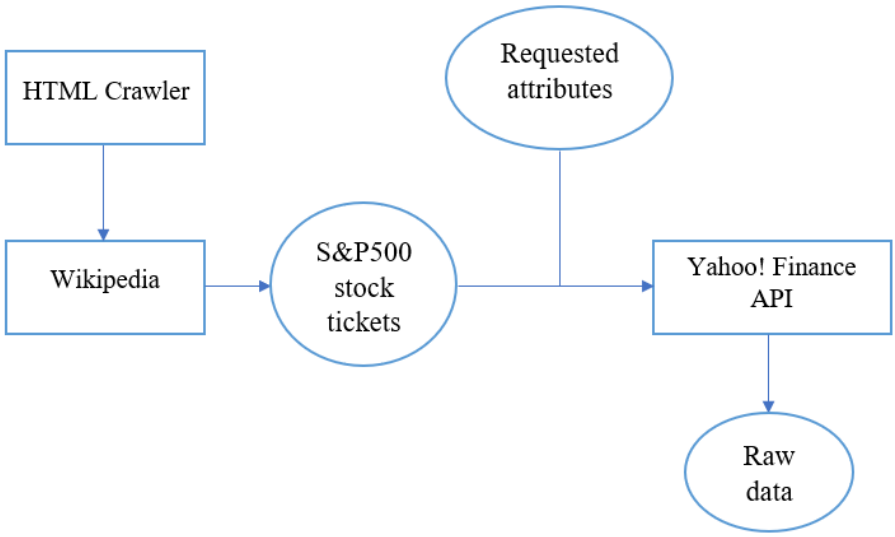
Hình 4.2: Tổng quan workflow

Quy trình này thực hiện trên 5 đoạn của dữ liệu đầu vào (5 folds) và kết quả đánh giá là kết quả trung bình của 5 đoạn này.

4.3 Thu thập dữ liệu

Tập dữ liệu 5 năm - mỗi điểm dữ liệu cách nhau 1 ngày được sử dụng để so sánh quy trình thực hiện của nhóm với quy trình trước đó là tập dữ liệu thu thập trong paper gốc [1]

Tập dữ liệu còn lại, 1 năm - mỗi điểm dữ liệu cách nhau 1h, nhóm sử dụng được thu thập qua quá trình sau:



Hình 4.3: Thu thập dữ liệu

Cả 2 tập dữ liệu có cùng các cột dữ liệu:

- Date: thời điểm dữ liệu được thu thập.
- Open, High, Low, Close: lần lượt là giá mở cửa, giá cao nhất, giá thấp nhất và giá đóng cửa.
- Volume: khối lượng giao dịch.

Định nghĩa của bộ giá Open High Low Close (OHLC) và Volume đã được trình bày rõ ở mục 2.1.3.

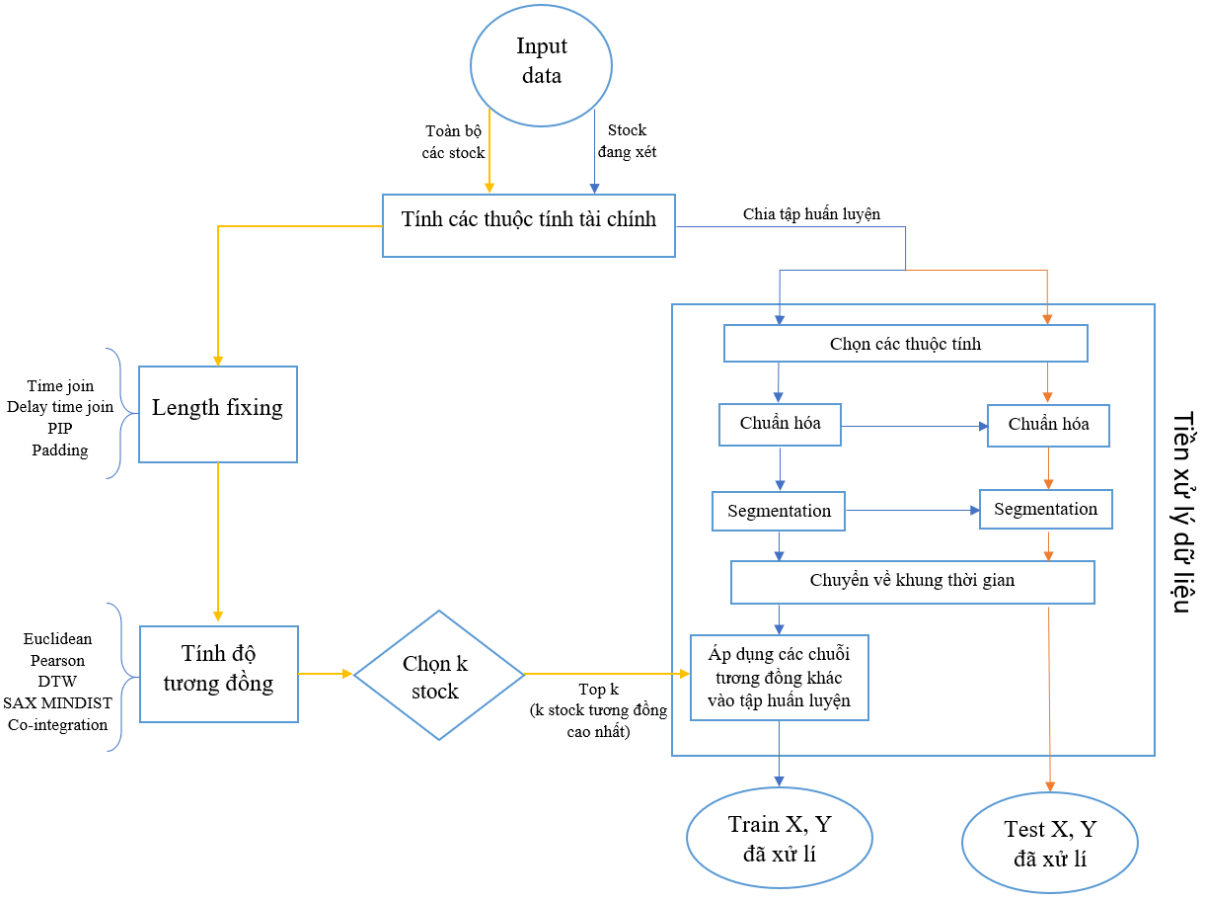
Cổ phiếu chính được dự đoán ở cả hai tập dữ liệu này là cổ phiếu Google (GOOGL)

Date	Open	High	Low	Close	Volume	Name
13-08-12	92.29	92.59	91.74	92.4	2075391	MMM
14-08-12	92.36	92.5	92.01	92.3	1843476	MMM
15-08-12	92	92.74	91.94	92.54	1983395	MMM
16-08-12	92.75	93.87	92.21	93.74	3395145	MMM

Hình 4.4: Thu thập dữ liệu

4.4 Xử lí dữ liệu và áp dụng tương đồng trong huấn luyện

Đoạn dữ liệu đầu vào sẽ được đưa qua pipeline tiền xử lí (hình 4.5)



Hình 4.5: Xử lí dữ liệu

Cụ thể từng bước:

- Tính các thuộc tính tài chính: Từ dữ liệu ban đầu tính các thuộc

tính, chỉ số tài chính khác có thể sử dụng làm thuộc tính huấn luyện mô hình. Các giá trị được tính gồm:

- Các chỉ báo kỹ thuật (đã được định nghĩa trong mục 2.1.4) như:
  - Price rate of change (PROC): hay còn gọi là ROC, sẽ tính phần trăm thay đổi của giá hiện tại với giá trước cách một khoảng thời gian nào đó. Công thức như sau:

$$\text{PROC} = \frac{\text{Price}_t - \text{Price}_{t-n}}{\text{Price}_{t-n}}.$$

  - Moving Average (MA): hay trung bình động, và thường được tính dưới dạng là Simple Moving Average (SMA), sẽ tính trung bình giá trong một khoảng thời gian n nào đó:

$$\text{SMA}_n = \frac{1}{n} \sum_{i=m-k+1}^{m} \text{Price}_i.$$

    , với m tổng số các điểm dữ liệu giá.
  - Exponential Moving Average (EMA): là một loại trung bình động gán cho các giá gần hiện tại *weight* cao hơn so với các giá cũ:

$$\text{EMA}_n = \frac{\text{WeightedSum}_n}{\text{WeightedCount}_n}.$$

    Trong đó:

$$\begin{cases} \text{WeightedSum}_0 = \text{WeightedCount}_0 = 0 \\ \text{WeightedSum}_n = \text{Price}_n + (1-\alpha)\text{WeightedSum}_{n-1} \\ \text{WeightedCount}_n = 1 + (1-\alpha)\text{WeightedCount}_{n-1} \end{cases}$$

    và α (tùy ý) là độ giảm của các *weight*. Trong thực tế, α thường sẽ có giá trị là $\frac{2}{n+1}$.
  - Moving Average Convergence Divergence (MACD): tìm ra động lượng của giá và xác định xu hướng, và có công thức

là:

$$\text{MACD} = \text{EMA}_{12} - \text{EMA}_{26}.$$

* RSI: kiểm tra diễn biến hoạt động của một cổ phiếu như thế nào trong một khoảng thời gian và sẽ có giá trị từ 0 đến 100. Công thức như sau:

$$\text{RSI} = 100 - \frac{100}{1 + \text{RS}}.$$

Trong đó, $\text{RS} = \dfrac{\text{AverageGain}}{\text{AverageLoss}}$, với AverageGain và AverageLoss lần lượt là trung bình giá tăng và giảm trong n khoảng thời gian trước. Thông thường, RSI sẽ được tính với $n = 14$.

Các chỉ báo kỹ thuật trên sẽ được tính bằng giá đóng cửa Close.

- Chêch lệch giá mở - đóng cửa: là hiệu Open - Close.
- Chêch lệch giá cao - thấp: là hiệu High - Low.

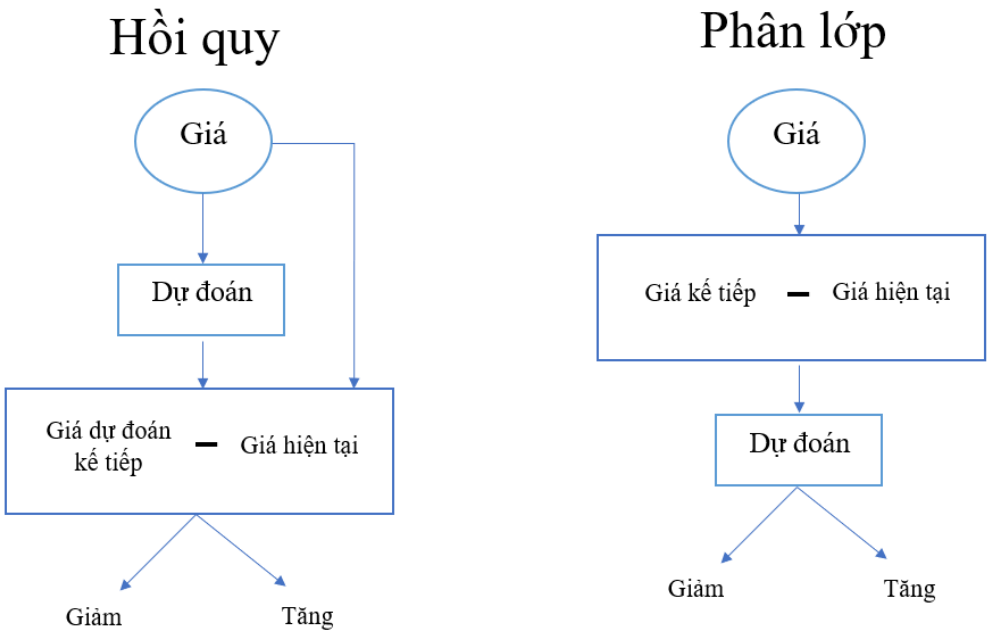
- Chia tập huấn luyện - kiểm nghiệm: Dữ liệu được chia thành 2 tập huấn luyện, kiểm nghiệm theo tỉ lệ 75% - 25%
- Lựa chọn các thuộc tính: Nhóm lựa chọn một hoặc nhiều tổ hợp khác nhau từ các thuộc tính trên cho việc huấn luyện.
- Chuẩn hóa: Dữ liệu được chuẩn hóa sử dụng Standard Scaler của thư viện sklearn. Scaler này chỉ được fit trên tập huấn luyện và dùng cho cả tập huấn luyện và kiểm nghiệm.
- Segmentation: Dữ liệu được áp dụng giảm chiều bằng PCA hoặc ... bằng SAX hoặc bỏ qua (không áp dụng bất kì hình thức segmentation nào).
- Chuyển dữ liệu về khung thời gian (Phương pháp cửa sổ trượt - Sliding window): Cuối cùng, dữ liệu được chuyển về dạng cửa sổ

trượt. Mỗi điểm dữ liệu chỉ dùng thông tin của d ngày trước đó trong huấn luyện.

- Áp dụng tương đồng vào huấn luyện: Dữ liệu của k cổ phiếu tương đồng cao nhất qua các bước xử lí trên sẽ được thêm vào dữ liệu huấn luyện sử dụng phương pháp weighted sampling như đã nêu ở trên.

4.5 Huấn luyện và dự đoán

Nhóm sử dụng các mô hình máy học Random Forest, Gradient Boosting, XGBoost (Regressor và classifier) và LSTM (Regressor) cho việc huấn luyện. Tuy có sử dụng các mô hình hồi quy, kết quả dự đoán được nhóm chuyển về dạng phân lớp, dự đoán xu hướng tăng/giảm.



Hình 4.6: Dự đoán xu hướng

Thông tin các mô hình sử dụng:

- Random Forest: Nhóm sử dụng cài đặt của thư viện sklearn, siêu tham số n_estimator = 100

- Gradient Boosting: Nhóm sử dụng cài đặt của thư viện sklearn, siêu tham số n_estimator = 100, learning rate = 0.02

- XGBoost: Nhóm sử dụng cài đặt từ tác giả (package xgboost [20]), siêu tham số n_estimator = 100, learning rate = 0.01 cho phân lớp và _estimator = 300, learning rate = 0.02 cho hồi quy.

- LSTM: Nhóm sử dụng API mạng neural cao cấp Keras và thư viện Tensorflow. Cấu trúc mạng LSTM gồm 2 lớp LSTM 50 neural, 1 lớp Dropout với tỉ lệ 0.2, 1 lớp Dense 50 neural và 1 lớp Dense output. Mạng LSTM này sử dụng optimizer Adam với learning rate 0.02, hàm loss MSE, chạy tối đa 100 epochs, validation_split = 0.2, có áp dụng callback model checkpoint của keras lưu lại weight của model có hàm loss tối thiểu.

4.6 Đánh giá

Nhóm sử dụng độ chính xác, điểm F1 và lợi nhuận để đánh giá mô hình. Lợi nhuận của mô hình được tính bằng cách sử dụng kết quả xu hướng dự đoán và dữ liệu giá cổ phiếu kết hợp với một chiến lược giao dịch qua các điểm thời gian. Nhóm áp dụng 2 chiến lược giao dịch sau:

- Long profits (Mua thấp bán cao): Chiến lược đơn giản này dựa vào dự đoán xu hướng tăng thì sẽ mua một lượng cổ phiếu và giữ cho đến khi dự đoán xu hướng giảm thì bán ra. Chênh lệch giữa điểm mua và bán sẽ là lợi nhuận của chiến lược này. Trong khoảng thời gian xu hướng dự đoán giảm liên tục thì chiến lược này sẽ không thực hiện giao dịch.

- Long and short profits (Mua thấp - Bán khống): Áp dụng như chiến lược trên, tuy nhiên trong khoảng thời gian xu hướng dự đoán giảm, chiến lược này sẽ bán khống một số lượng cổ phiếu và mua lại số cổ phiếu này khi xu hướng được dự đoán tăng trở lại. Số lượng cổ phiếu

trước và sau khoảng thời gian bán khống không thay đổi và chênh lệch giữa điểm bán và mua lại sẽ là lợi nhuận của việc bán khống. Chiến lược này áp dụng cả việc mua thấp bán cao và bán khống nên có thể cho lợi nhuận cao hơn nếu dự đoán tốt, nhưng cũng kèm theo rủi ro cao do lỗ của việc bán khống có thể là vô tận.

4.7 Các thiết lập thí nghiệm

Mô hình thử nghiệm trên mọi tổ hợp các tham số ở các bước ở quy trình đã nêu trên. Cụ thể:

- Thuộc tính huấn luyện: ('Close_norm'), ('Close_proc'), ('Close_norm', 'Close_proc', 'rsi_norm', 'MACD_norm', 'Open_Close_diff_norm', 'High_Low_diff_norm', 'Volume_norm'), ('Close_norm', 'rsi_norm', 'MA_norm')
- Khung thời gian: 5, 10, 15 ngày
- Chuẩn hóa: StandardScaler
- Segmentation: PCA(n_components=3), SAX, không sử dụng
- Thuật toán xử lí độ dài chuỗi: 'padding', 'time join', 'delay time join', 'pip'
- Thuật toán tương đồng: 'euclidean', 'pearson', 'co-integration', 'sax', 'dtw'
- Top k: 0, 10, 25, 50
- Mô hình: 'RandomForestRegressor', 'GradientBoostingRegressor', 'XGBRegressor', 'LSTM', 'XGBClassifier', 'GradientBoostingClassifier', 'RandomForestClassifier'

- Giá trị dự đoán: Giá đóng cửa đã chuẩn hóa (Close_norm), tỉ lệ thay đổi giá qua ngày (Close_proc)

Trong các tổ hợp này, các tổ hợp có thuộc tính là ('Close_proc') và giá trị dự đoán ('Close_norm') bị loại bỏ. Vì dự đoán giá đóng cửa chỉ từ các tỉ lệ là vô lý. Sau khi đã loại bỏ các cấu hình này, mỗi mô hình máy học có khoảng 2880 tổ hợp cấu hình riêng, tổng cộng khoảng 20.000 cấu hình riêng biệt cho toàn bộ thí nghiệm trên một tập dữ liệu.

4.8 Những cải tiến đã áp dụng

Trong quy trình này, một số kỹ thuật được áp dụng để khắc phục các hạn chế của bài báo cơ sở (nhắc đến ở 2.4) và một số cải tiến được thêm vào như sau:

- Khi cấu hình thuộc tính chỉ là (Close_proc) và có sử dụng biến đổi SAX, dữ liệu Close_proc được biến đổi tuyến tính để khắc phục chuỗi SAX cho được chỉ có một ký tự lặp.

- Quy trình sử dụng thêm các mô hình hiện đại như XGBoost, LSTM để tận dụng tập huấn luyện lớn sau khi đã áp dụng tương đồng.

- Ở từng fold dữ liệu có áp dụng một phương pháp phát hiện quá khớp - chưa khớp đơn giản. Các cấu hình bị phát hiện quá khớp - chưa khớp ở ít nhất 1 fold sẽ bị loại.

- Quy trình không giới hạn tổ hợp thuộc tính khi sử dụng với khung thời gian nhiều ngày.

- Quy trình không áp dụng thử nghiệm 2 bước như bài báo gốc, do việc lựa chọn cấu hình dự trên hiệu suất tổng thể có thể gây tác dụng ngoài ý muốn như miêu tả trước đó. Thay vào đó quy trình này sẽ chạy mọi tổ hợp cấu hình có thể để tìm ra cấu hình tốt nhất.

4.9 Câu hỏi nghiên cứu

Trong bài khóa luận này, nhóm quan tâm đến việc đưa ra kết quả có thể trả lời cho các câu hỏi

- Liệu quy trình áp dụng các cổ phiếu tương đồng khác trong huấn luyện có nâng cao kết quả của mô hình?
- Cấu hình quy trình nào sẽ cho ra được kết quả tốt nhất (ở từng tập dữ liệu)?
- Liệu kết quả của quy trình này có là cải tiến của bài báo cơ sở gốc?

Chương 5

Kết quả

5.1 Kết quả trên bộ dữ liệu 5 năm - khoảng cách 1 ngày

selected_features	transform	window_len	model	target_col	mean_accuracy / std_accuracy	mean_f1 / std_f1	mean_LongShort_profit	mean_Long_profit
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	None	15	**GradientBoostingClassifier**	Close_norm	0.6062 *0.0453*	0.6009 *0.0474*	67.454 *11.23%*	47.302 *8.352%*
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	SAX	5	**GradientBoostingRegressor**	Close_proc	0.5799 *0.0329*	0.5393 *0.0605*	35.128 *5.221%*	36.85 *5.975%*
['Close_norm']	None	5	**RandomForestClassifier**	Close_norm	0.5196 *0.056*	0.4842 *0.0841*	41.072 *5.907%*	41.298 *6.495%*
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	None	5	**RandomForestRegressor**	Close_proc	0.5579 *0.0673*	0.533 *0.0769*	30.208 *3.291%*	34.39 *5.01%*
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	None	15	**LSTM**	Close_proc	0.5401 *0.0394*	0.505 *0.0519*	38.582 *7.471%*	32.866 *6.473%*
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	SAX	5	**XGBClassifier**	Close_norm	0.5617 *0.0273*	0.5531 *0.0298*	53.744 *8.587%*	46.158 *7.658%*
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	SAX	15	**XGBRegressor**	Close_proc	0.5445 *0.037*	0.5234 *0.0714*	41.226 *8.685%*	34.188 *7.08%*

Hình 5.1: Kết quả tốt nhất từng model không áp dụng tương đồng

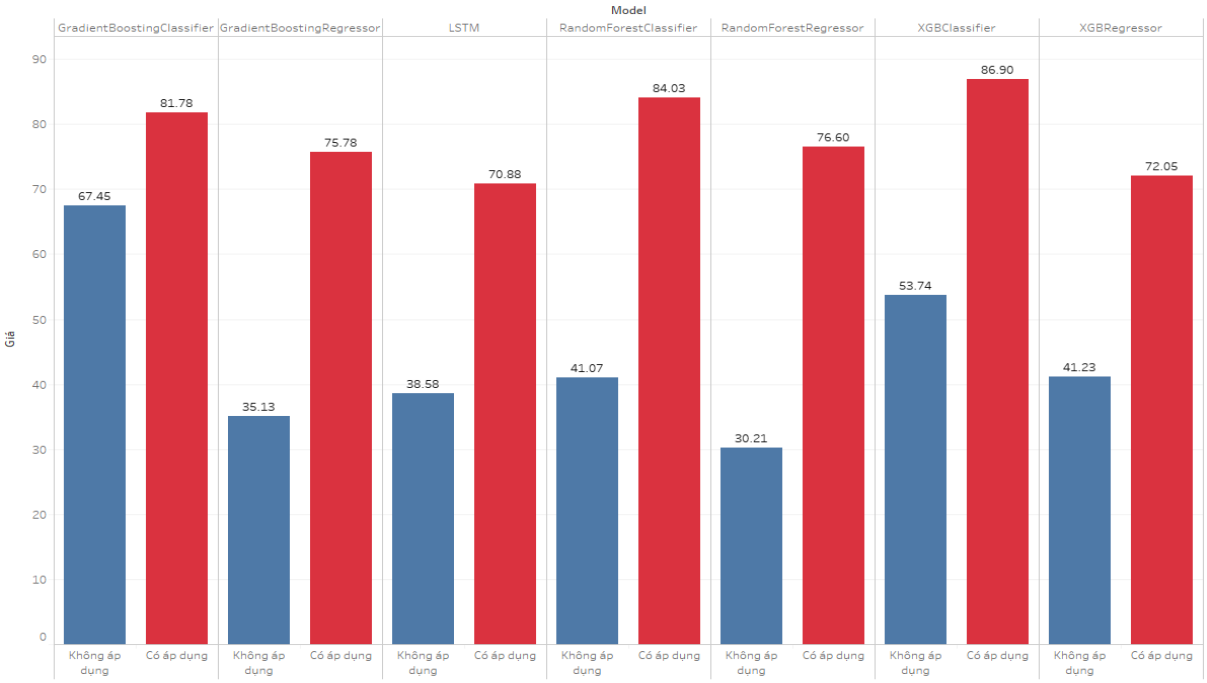
Hình 5.1 cho thấy kết quả dự đoán xu hướng tốt nhất của từng mô hình khi không áp dụng kỹ thuật tương đồng. Ở đây ta thấy được các mô hình đã cho được lợi nhuận dương với giá trị đạt được lớn nhất đạt được là 67.454$, 11.23% vốn.

selected_features	transformer	sim_func / fix_len_func	k stock	window_len	model	target_col	mean_accuracy / std_accuracy	mean_f1 / std_f1	mean_LS_profit	mean_L_profit
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	SAX	sax time_join	25	10	GradientBoostingClassifier	Close_norm	0.5979 *0.0712*	0.5593 0.0912	81.784 14.673%	54.016 9.941%
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	SAX	sax delay_time_join	25	10	GradientBoostingRegressor	Close_proc	0.5539 *0.0176*	0.4829 0.0745	75.78 12.877%	51.014 9.043%
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	SAX	co-integration pip	50	10	RandomForestClassifier	Close_norm	0.5739 *0.0389*	0.5331 0.0607	84.028 14.251%	55.138 9.73%
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	None	dtw time_join	10	5	RandomForestRegressor	Close_proc	0.5577 *0.0589*	0.5108 0.0795	76.596 11.206%	57.584 8.968%
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	None	pearson pip	50	10	LSTM	Close_proc	0.554 *0.0572*	0.4945 0.0898	70.876 11.057%	48.562 8.133%
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	SAX	co-integration delay_time_join	10	10	XGBClassifier	Close_norm	0.5539 *0.055*	0.5382 0.0635	86.9 14.785%	56.574 9.997%
['Close_norm'; 'rsi_norm'; 'MA_norm']	SAX	co-integration delay_time_join	50	5	XGBRegressor	Close_proc	0.555 *0.0676*	0.5168 0.0952	72.052 13.256%	56.788 10.17%

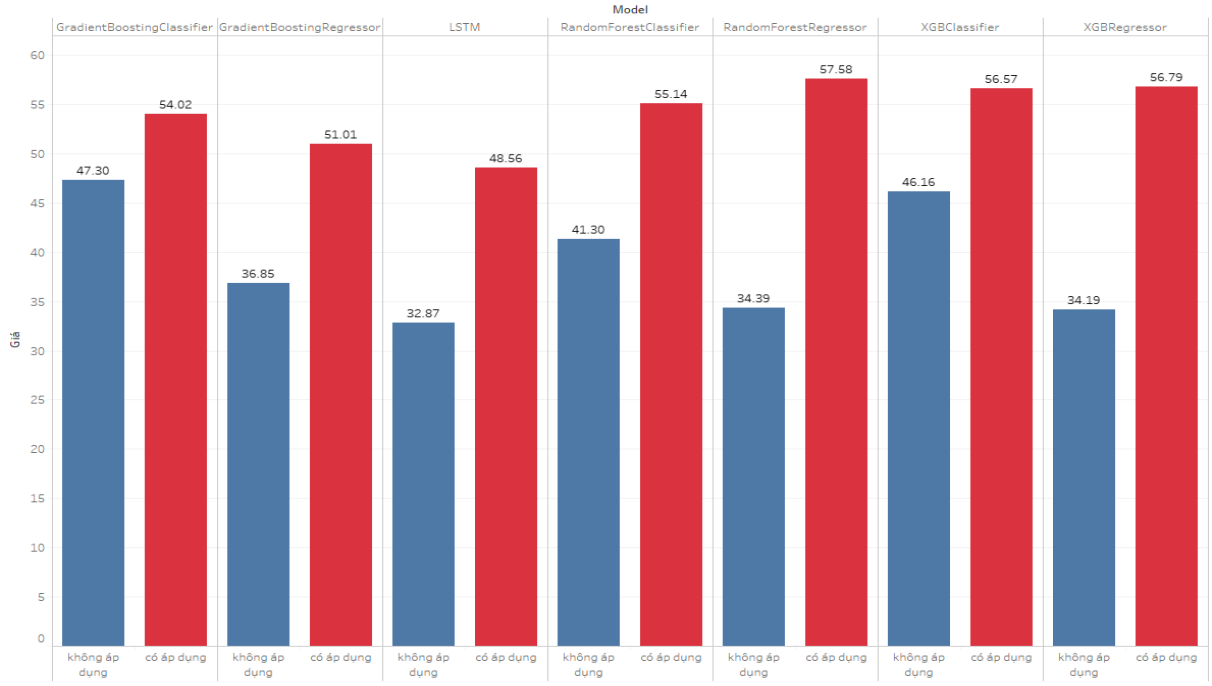
Hình 5.2: Kết quả tốt nhất từng model sau khi áp dụng tương đồng

Hình 5.2 cho thấy kết quả dự đoán tốt nhất của các mô hình sau khi đã áp dụng tương đồng trong huấn luyện, với lợi nhuận đạt được là 86.9$, 14.785% vốn.

Hình 5.4 và 5.3 cho thấy việc áp dụng tương đồng trong huấn luyện mang lại kết quả tốt hơn trên mọi mô hình, với cả hai chiến lược giao dịch.



Hình 5.3: So sánh lợi nhuận Long Short giữa có và không áp dụng tương đồng trong huấn luyện



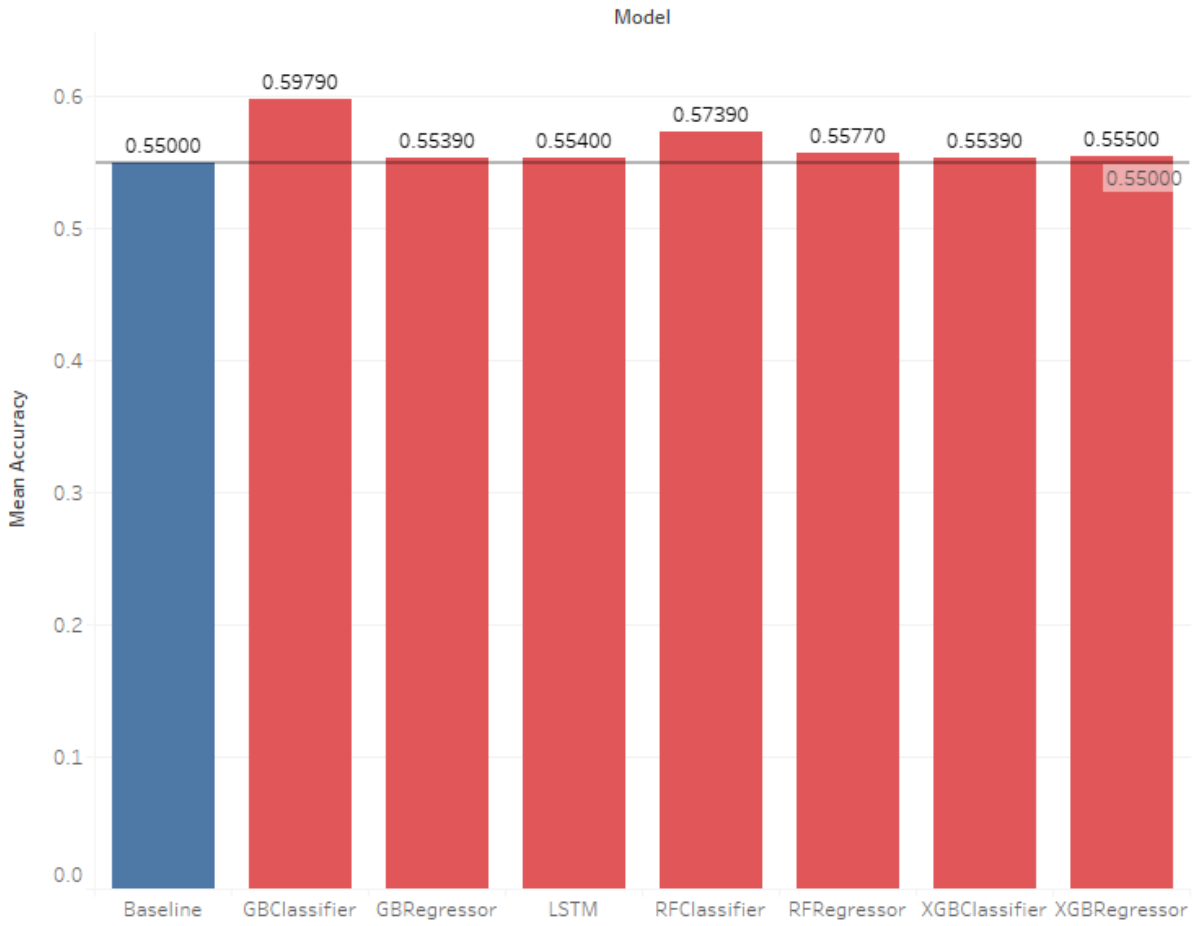
Hình 5.4: So sánh lợi nhuận Long giữa có và không áp dụng tương đồng trong huấn luyện

Với kết quả đạt được trên, việc áp dụng tương đồng trong huấn luyện đã chứng minh được hiệu quả của nó. Việc còn lại là so sánh những kết quả này với kết quả tốt nhất được báo cáo trong bài báo cơ sở được dựa vào để xét quy trình mới này có cải tiến các kết quả cũ hay không. Kết quả tốt nhất của bài báo cơ sở này (hình 5.5 và trong mục 5.2.2 [1]) đạt được độ chính xác trung bình 0.55 và điểm F1 trung bình 0.495.

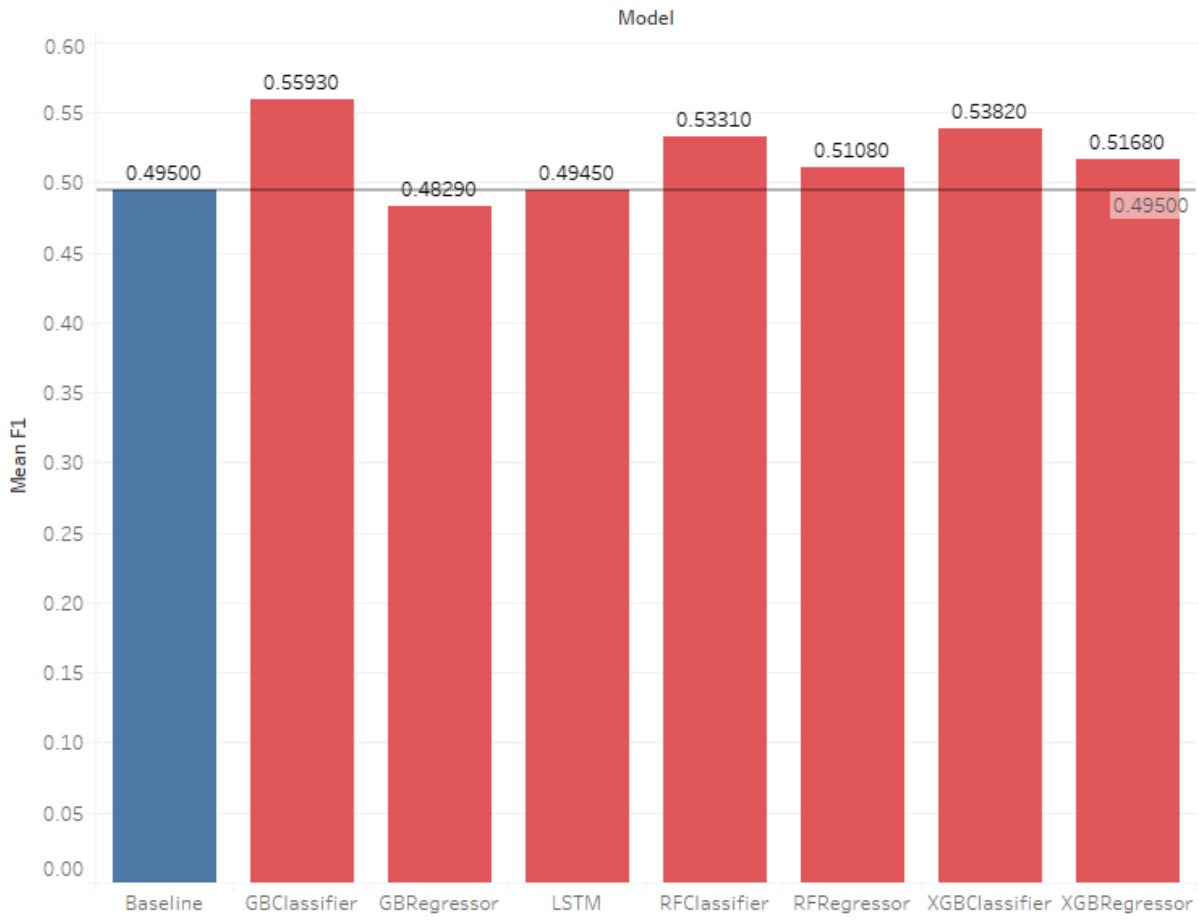
Similarity function	Model	Horizon	Mean Accuracy	Std Accuracy	Mean F1	Std F1	Mean Sharp Ratio (risk)	Mean Profit
Co-integration	GradientBoostingRegressor	Next day	0.550	0.050	0.459	0.096	1.296	19.875
		Next 3 days	0.546	0.075	0.448	0.098	1.228	11.487
		Next week	0.542	0.065	0.455	0.091	1.239	15.645

Hình 5.5: Kết quả tốt nhất của bài báo làm cơ sở [1] - Mục 7 APPENDIX A - Bảng (c)

Hình 5.6 cho thấy toàn bộ mô hình qua quy trình đều cho kết quả có độ chính xác trung bình tốt hơn kết quả cơ sở, với giá trị cao nhất đạt 0.5979. Hình 5.7 cho thấy đa số mô hình qua quy trình đều đạt kết quả có điểm F1 trung bình lớn hơn ở kết quả cơ sở, với giá trị duy nhất nhỏ hơn là 0.4829 (so với cơ sở 0.495) ở mô hình Gradient Boosting Classifier và giá trị lớn nhất đạt được 0.5593.



Hình 5.6: So sánh độ chính xác giữa các mô hình đạt được và baseline



Hình 5.7: So sánh độ điểm F1 giữa các mô hình đạt được và baseline

5.2 Kết quả trên bộ dữ liệu 1 năm - khoảng cách 1 giờ

selected_features	transformer	window_len	model	target_col	mean_accuracy / std_accuracy	mean_f1 / std_f1	mean_LS_profit	mean_L_profit
['Close_proc']	SAX	15	GradientBoostingClassifier	Close_proc	0.5491 0.0252	0.5054 0.0339	130.033 7.713%	127.299 7.514%
['Close_proc']	SAX	5	GradientBoostingRegressor	Close_proc	0.5405 0.0264	0.4493 0.0515	110.408 6.627%	112.527 6.747%
['Close_proc']	None	10	RandomForestClassifier	Close_proc	0.5581 0.0852	0.5529 0.0836	109.726 7.419%	117.284 7.439%
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	None	5	RandomForestRegressor	Close_norm	0.5201 0.0299	0.5092 0.0314	113.635 6.904%	114.14 6.885%
['Close_norm'; 'rsi_norm'; 'MA_norm']	None	10	LSTM	Close_proc	0.5162 0.0801	0.4879 0.0981	72.685 5.037%	97.777 6.177%
['Close_proc']	None	5	XGBClassifier	Close_proc	0.5303 0.0581	0.5227 0.0589	104.926 6.27%	109.786 6.568%
['Close_norm']	None	10	XGBRegressor	Close_proc	0.5426 0.0291	0.4835 0.0479	127.532 7.757%	125.2 7.537%

Hình 5.8: Kết quả tốt nhất từng model không áp dụng tương đồng

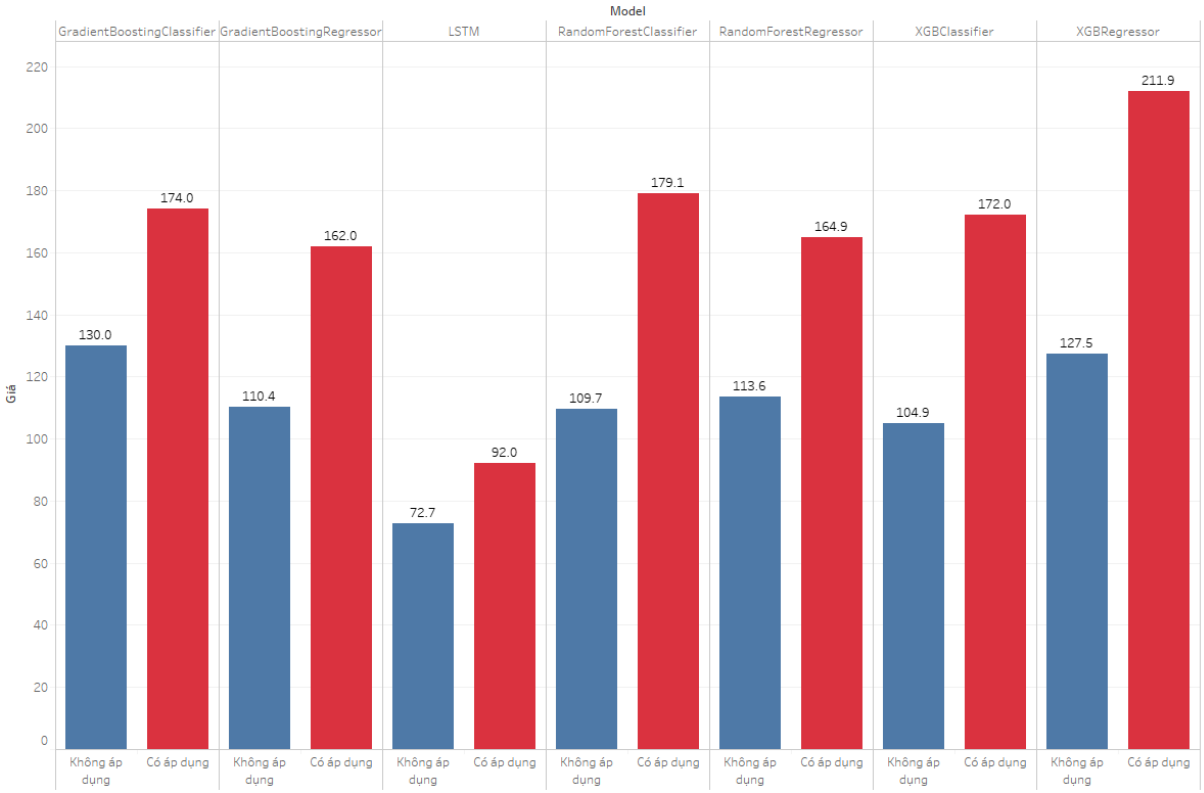
Hình 5.8 cho thấy kết quả dự đoán xu hướng tốt nhất của từng mô hình khi không áp dụng kỹ thuật tương đồng. Ở đây ta thấy được các mô hình đã cho được lợi nhuận dương với giá trị đạt được lớn nhất đạt được là 130.033$, 7.713% vốn.

selected_features	transformer	sim_func / fix_len_func	k stock	window_len	model	target_col	mean_accuracy / std_accuracy	mean_f1 / std_f1	mean_LS_profit	mean_L_profit
['Close_proc']	SAX	co-integration pip	10	15	GradientBoostingClassifier	Close_proc	0.5551 0.0404	0.4837 0.0514	174.037 10.485%	149.301 8.9%
['Close_norm']	None	co-integration padding	10	5	GradientBoostingRegressor	Close_proc	0.5535 0.0338	0.538 0.029	162.038 10.005%	146.549 8.972%
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	None	dtw delay_time_join	50	10	RandomForestClassifier	Close_norm	0.5472 0.0436	0.531 0.0504	179.116 10.575%	151.979 9.017%
['Close_norm']	None	co-integration delay_time_join	10	5	RandomForestRegressor	Close_norm	0.5485 0.0639	0.546 0.0641	164.856 9.583%	147.958 8.761%
['Close_norm'; 'rsi_norm'; 'MA_norm']	SAX	sax time_join	50	5	LSTM	Close_norm	0.546 0.0516	0.5167 0.0501	92.043 5.878%	111.552 6.908%
['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	PCA	sax time_join	10	10	XGBClassifier	Close_norm	0.5554 0.0383	0.5256 0.0459	171.989 10.203%	148.415 8.831%
['Close_norm']	None	sax pip	50	15	XGBRegressor	Close_proc	0.5786 0.0447	0.497 0.0851	211.899 13.061%	170.447 10.333%

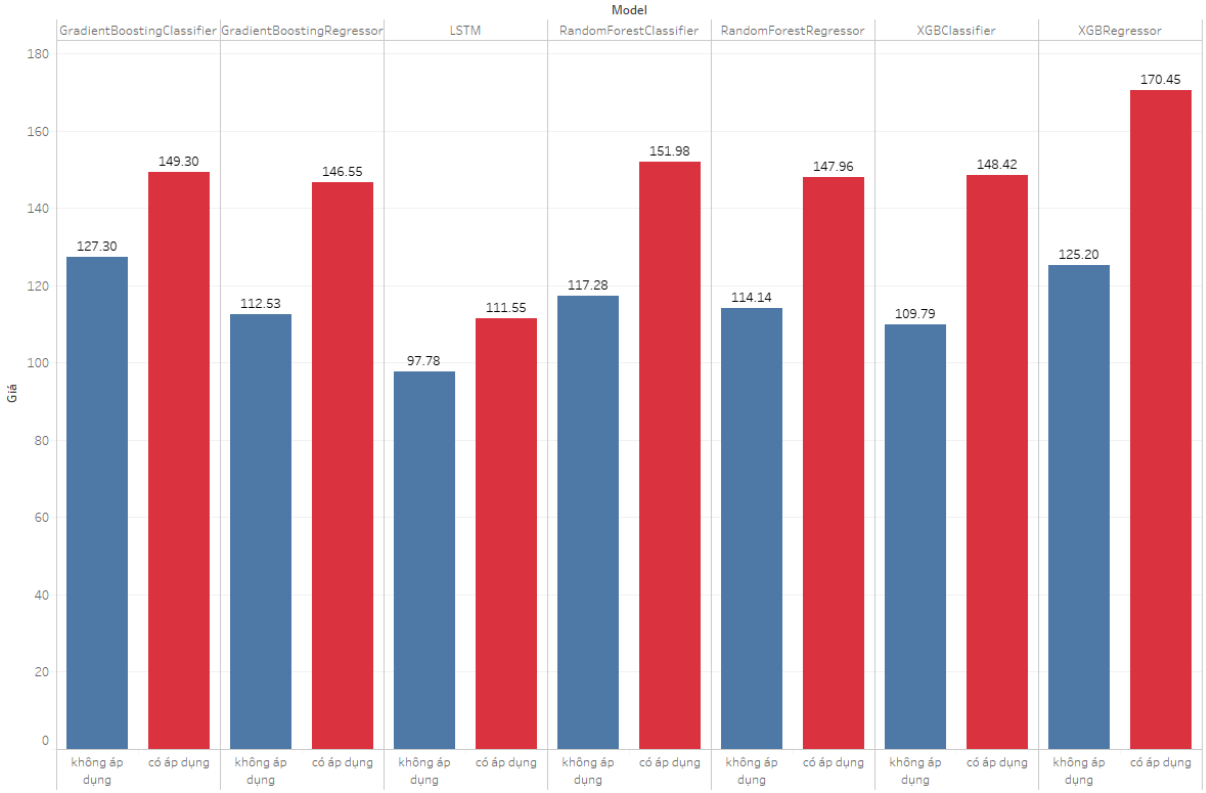
Hình 5.9: Kết quả tốt nhất từng model sau khi áp dụng tương đồng

Hình 5.9 cho thấy kết quả dự đoán tốt nhất của các mô hình sau khi đã áp dụng tương đồng trong huấn luyện, với lợi nhuận đạt được là 211.899$, 13.061% vốn.

Hình 5.11 và 5.10 cho thấy việc áp dụng tương đồng trong huấn luyện mang lại kết quả tốt hơn trên mọi mô hình, với cả hai chiến lược giao dịch.



Hình 5.10: So sánh lợi nhuận Long Short giữa có và không áp dụng tương đồng trong huấn luyện



Hình 5.11: So sánh lợi nhuận Long giữa có và không áp dụng tương đồng trong huấn luyện

5.3 Thảo luận

Các kết quả ở cả hai tập dữ liệu đều cho thấy rằng bằng cách áp dụng dữ liệu các cổ phiếu khác có độ tương đồng cao với cổ phiếu đang xét, ta có thể nâng cao kết quả đạt được của các mô hình máy học. Đây là câu trả lời tích cực cho câu hỏi nghiên cứu (4.9) đầu tiên.

Hình 5.12 trả lời cho câu hỏi nghiên cứu thứu hai (4.9), miêu tả cấu hình tốt nhất của cho từng tập dữ liệu. Cả hai cấu hình này đề sử dụng mô hình XGBoost, một mô hình hiện đại giải quyết bài toán học có giám sát và cho được độ chính xác bên cạnh với các mô hình học sâu. Việc sử dụng mô hình này để tận dụng tối đa tập dữ liệu huấn luyện lớn sau khi áp dụng tương đồng của nhóm là một lựa chọn chính xác.

Dataset	selected_features	transformer	sim_func / fix_len_func	k stock	window_len	model	target_col	mean_accuracy / std_accuracy	mean_f1 / std_f1	mean_LS_profit	mean_L_profit
5yr	['Close_norm'; 'Close_proc'; 'rsi_norm'; 'MACD_norm'; 'Open_Close_diff_norm'; 'High_Low_diff_norm'; 'Volume_norm']	SAX	co-integration delay_time_join	10	10	XGBClassifier	Close_norm	0.5539 *0.055*	0.5382 0.0635	86.9 14.785%	56.574 9.997%
1yr	['Close_norm']	None	sax pip	50	15	XGBRegressor	Close_proc	0.5786 0.0447	0.497 0.0851	211.899 13.061%	170.447 10.333%

Hình 5.12: Cấu hình tốt nhất trong từng tập dữ liệu

Câu hỏi nghiên cứu thứ ba (4.9) đã được trả lời ở trên, dựa vào hình 5.6 và 5.7 cho thấy các mô hình tốt nhất được huấn luyện qua quy trình này đều đạt độ chính xác vượt trội so với nghiên cứu cơ sở.

Chương 6

Kết luận và hướng phát triển

6.1 Kết luận

Trong bài khóa luận này, sử dụng nghiên cứu của Lior Sidi[1] làm cơ sở, nhóm đã xây dựng và cải tiến quy trình áp dụng các loại cổ phiếu có độ tương đồng cao vào huấn luyện cho các mô hình máy học, với mục tiêu quy trình cải tiến này có thể đưa ra kết quả tốt hơn nghiên cứu cơ sở và ứng dụng được kỹ thuật mới này trong dự đoán giá cổ phiếu cho các tập dữ liệu khác nhau.

Nghiên cứu được thực hiện trên một tập dữ liệu 5 năm, các điểm dữ liệu cách nhau 1 ngày, có sẵn để có thể so sánh với nghiên cứu cơ sở, và một tập dữ liệu 1 năm, các điểm dữ liệu cách nhau 1 giờ, được thu thập từ ngày 06/04/2020 đến ngày 06/04/2021. Các tập dữ liệu này được chia thành 5 đoạn, từng đoạn được đưa qua một quy trình xử lí và áp dụng tương đồng để tạo ra các tập huấn luyện được tăng cường nhiều cổ phiếu gần giống với cổ phiếu đang xét và tập kiểm nghiệm tương ứng. Áp dụng học và dự đoán xu hướng trên tập dữ liệu huấn luyện/kiểm nghiệm này sử dụng các mô hình máy học và các chiến lược giao dịch, ta sẽ nhận được lợi nhuận dự kiến của mô hình với từng chiến lược giao dịch. Các giá trị lợi

nhuận và độ chính xác mô hình được tính trung bình giữa 5 đoạn (folds) này để có được kết quả khách quan.

Nhóm thực hiện các tổ hợp thí nghiệm, thực hiện sử dụng và không sử dụng việc áp dụng tương đồng trong huấn luyện, trên 2 tập dữ liệu để so sánh kết quả giữa hai cách thức thực hiện. Hình 5.3, 5.4, 5.10 5.11 cho thấy rõ việc áp dụng tương đồng tăng cường cho tập huấn luyện mang lại kết quả vượt trội trên mọi mô hình ở cả hai tập dữ liệu. Đối với kết quả trên bộ dữ liệu 5 năm cơ sở, nhóm thực hiện so sánh với kết quả báo cáo trong nghiên cứu cơ sở và từ hình 5.6 và 5.7 cho thấy kết quả của quy trình thực hiện này đạt được cao hơn so với kết quả cơ sở, cho thấy quy trình này đã thực hiện cải tiến thành công.

Trong bài khóa luận này, nhóm tập trung sâu vào việc xử lí dữ liệu và áp dụng tương đồng cho tập huấn luyện nhưng có hạn chế trong việc tinh chỉnh (fine tunning) các mô hình. Hạn chế này thấy rõ nhất ở mô hình LSTM, đây là mô hình học sâu phức tạp nhưng do nhóm không thể tinh chỉnh nhiều cho mô hình này nên dù vẫn cho thấy được việc áp dụng tương đồng giúp nâng cao kết quả, giá trị lợi nhuận của mô hình này tổng quan thấp hơn so với cả các mô hình đơn giản hơn như Random Forest và Gradient Boost. Tuy nhiên, với sự thành công của việc áp dụng tương đồng trong huấn luyện và áp dụng thành công mô hình XGBoost trong bài toán này, nhóm rất hài lòng về các kết quả đạt được.

6.2 Hướng phát triển

Nhóm mong muốn việc áp dụng dữ liệu các loại cổ phiếu tương đồng cao vào dữ liệu huấn luyện, qua các kết quả chứng minh trong bài khóa luận này, sẽ trở thành một phương pháp tiền xử lý, tăng cường mới trong bài toán dự đoán giá chứng khoán. Nhóm đề xuất phương hướng phát triển cho bản thân và các nghiên cứu khác dựa vào phương pháp này như sau: Thực hiện thêm tinh chỉnh và cải tiến mô hình khi chưa áp dụng tương đồng. Sau khi áp dụng và tìm được một số thuật toán tương đồng,

thuật toán xử lý chuỗi mang lại kết quả tốt, tiếp tục tinh chỉnh mô hình trên tập dữ liệu huấn luyện tăng cường mới này.

Các quá trình tinh chỉnh này hi vọng sẽ khắc phục được hạn chế của bài khóa luận này đối với các mô hình phức tạp. Áp dụng các mô hình phức tạp như mạng học sâu,... để tận dụng dữ liệu huấn luyện đã tăng cường lớn; áp dụng các phương thức tính độ tương đồng khác như sử dụng mô hình ensemble để tính độ tương đồng,... cũng là một số phương pháp có thể cải tiến cho quy trình thực hiện này.

Tài liệu tham khảo

References

[1] Sidi, Lior. "Improving S&P stock prediction with time series stock similarity". In: *arXiv preprint arXiv:2002.05784* (2020).

[2] Online. *Chứng khoán*. URL: https://vi.wikipedia.org/wiki/Chòệẩng_khoÕán (visited on 05/31/2021).

[3] Smigel, Leo. "What is Open High Low Close in Stocks?" In: (). URL: https://analyzingalpha.com/open-high-low-close-stocks.

[4] Ho, Yu-Chi and Pepyne, David L. "Simple explanation of the no-free-lunch theorem and its implications". In: *Journal of optimization theory and applications* 115.3 (2002), pp. 549–570.

[5] Raschka, Sebastian. *EnsembleVoteClassifier*. URL: http://rasbt.github.io/mlxtend/user_guide/classifier/EnsembleVoteClassifier (visited on 05/08/2021).

[6] Efron, Bradley. "Bootstrap methods: another look at the jackknife". In: *Breakthroughs in statistics*. Springer, 1992.

[7] Breiman, Leo. "Random forests". In: *Machine learning* 45.1 (2001).

[8] Breiman, Leo. *Classification and regression trees*. Wadsworth International Group, 1984.

[9] Quiroz, Juan et al. “Fault detection of broken rotor bar in LS-PMSM using random forests”. In: *Measurement* 116 (Nov. 2017), pp. 273–280. DOI: 10.1016/j.measurement.2017.11.004.

[10] Freund, Yoav and Schapire, Robert E. “A decision-theoretic generalization of on-line learning and an application to boosting”. In: *Journal of computer and system sciences* 55.1 (1997), pp. 119–139.

[11] Freund, Yoav, Schapire, Robert, and Abe, Naoki. “A short introduction to boosting”. In: *Journal-Japanese Society For Artificial Intelligence* 14.771-780 (1999), p. 1612.

[12] Wang, Zhuo, Zhang, Jintao, and Verma, Naveen. “Realizing Low-Energy Classification Systems by Implementing Matrix Multiplication Directly Within an ADC”. In: *IEEE Transactions on Biomedical Circuits and Systems* 9 (Dec. 2015), pp. 1–1. DOI: 10.1109/TBCAS.2015.2500101.

[13] Breiman, Leo. *Arcing the edge*. Tech. rep. Technical Report 486, Statistics Department, University of California at ..., 1997.

[14] Friedman, Jerome H. “Greedy function approximation: a gradient boosting machine”. In: *Annals of statistics* (2001), pp. 1189–1232.

[15] Friedman, Jerome H. “Stochastic gradient boosting”. In: *Computational statistics & data analysis* 38.4 (2002), pp. 367–378.

[16] nikki2398@nikki2398. *ML - Gradient Boosting*. 2020. URL: https://www.geeksforgeeks.org/ml-gradient-boosting/ (visited on 05/09/2021).

[17] Starmer, Josh. *Gradient Boost Part 2 (of 4): Regression Details*. 2019. URL: https://www.youtube.com/watch?v=2xudPOBz-vs& (visited on 05/09/2021).

[18] Starmer, Josh. *Gradient Boost Part 4 (of 4): Classification Details*. 2019. URL: https://www.youtube.com/watch?v=StWY5QWMXCw (visited on 05/09/2021).

[19] Huel, Wanda. *Gradient Boosting Trees for Classification: A Beginner's Guide*. URL: https://morioh.com/p/e108a4521555.

[20] Chen, Tianqi and Guestrin, Carlos. "Xgboost: A scalable tree boosting system". In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016.

[21] developers, xgboost. *Introduction to Boosted Trees¶*. URL: https://xgboost.readthedocs.io/en/latest/tutorials/model.html (visited on 05/10/2021).

[22] Starmer, Josh. *XGBoost Part 3 (of 4): Mathematical Details*. 2020. URL: https://www.youtube.com/watch?v=ZVFeW798-2I (visited on 05/10/2021).

[23] *Taylor's theorem*. 2021. URL: https://en.wikipedia.org/wiki/Taylor\%27s_theorem (visited on 05/10/2021).

[24] Burnett, Colin M.L. *Artificial neural network*. 2006. URL: https://commons.wikimedia.org/wiki/File:Artificial_neural_network.svg.

[25] Vieira, Sandra, Pinaya, Walter, and Mechelli, Andrea. "Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders: Methods and applications". In: *Neuroscience Biobehavioral Reviews* 74 (Jan. 2017). DOI: 10.1016/j.neubiorev.2017.01.002.

[26] Ixnay. *Recurrent neural network unfold*. 2017. URL: https://en.wikipedia.org/wiki/File:Recurrent_neural_network_unfold.svg.

[27] Yuan, Xiaofeng, Li, Lin, and Wang, Yalin. "Nonlinear Dynamic Soft Sensor Modeling With Supervised Long Short-Term Memory Network". In: *IEEE Transactions on Industrial Informatics* PP (Feb. 2019), pp. 1–1. DOI: 10.1109/TII.2019.2902129.

[28] Benhe84. *Inner JOIN*. 2020. URL: https://commons.m.wikimedia.org/wiki/File:Inner_JOIN.png.

[29] Zaib, G., Ahmed, U., and Ali, A. "Computational Finance and its Applications". In: 2004, pp. 255–256. URL: https://www.witpress.com/Secure/elibrary/papers/CF04/CF04024FU.pdf.

[30] Hung, Ying Kit. "Perceptually important point identification for big data analytics : performance analysis and applications". In: (2017). URL: https://ows.lib.polyu.edu.hk/files/original/5fb3a56310e79284pdf.

[31] Online. *Principal component analysis*. URL: https://en.wikipedia.org/wiki/Principal_component_analysis (visited on 05/10/2021).

[32] Brems, Matt. "A One-Stop Shop for Principal Component Analysis". In: (2017). URL: https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c.

[33] Senin, Pavel. "Piecewise Aggregate Approximation of time series". In: (2016). URL: https://jmotif.github.io/sax-vsm_site/morea/algorithm/PAA.html.

[34] Senin, Pavel. "Symbolic Aggregate approXimation". In: (2016). URL: https://jmotif.github.io/sax-vsm_site/morea/algorithm/SAX.html.

[35] Online. *Euclidean distance*. URL: https://en.wikipedia.org/wiki/Euclidean_distance (visited on 05/09/2021).

[36] Online. *Pearson correlation coefficient*. URL: https://en.wikipedia.org/wiki/Pearson_correlation_coefficient (visited on 05/09/2021).

[37] XantaCross. *Euclidean vs DTW*. 2011. URL: https://commons.wikimedia.org/wiki/File:Euclidean_vs_DTW.jpg.

[38] Müller, Meinard. “Information Retrieval for Music and Motion”. In: 1. Springer-Verlag Berlin Heidelberg, 2007, pp. 69–70. ISBN: 978-3-540-74048-3. DOI: 10.1007/978-3-540-74048-3.

[39] Online. *Dynamic time warping*. URL: https://en.wikipedia.org/wiki/Dynamic_time_warping (visited on 05/09/2021).

[40] Online. *White noise*. URL: https://en.wikipedia.org/wiki/White_noise (visited on 05/22/2021).

[41] Online. *Stationary process*. URL: https://en.wikipedia.org/wiki/Stationary_process (visited on 05/22/2021).

[42] Online. *Dickey-Fuller test*. URL: https://en.wikipedia.org/wiki/DickeyóÀÊFuller_test (visited on 05/22/2021).

[43] Online. *Augmented Dickey-Fuller test*. URL: https://en.wikipedia.org/wiki/Augmented_DickeyóÀÊFuller_test (visited on 05/22/2021).

[44] Online. *Ordinary least squares*. URL: https://en.wikipedia.org/wiki/Ordinary_least_squares (visited on 05/22/2021).