**TRƯỜNG ĐẠI HỌC MỎ - ĐỊA CHẤT**

**KHOA CÔNG NGHỆ THÔNG TIN**

**\_\_\_\_\_\_\_\_\_\_\_\_\*\*\*\_\_\_\_\_\_\_\_\_\_\_\_**

Logo

Description automatically generated

**BÁO CÁO ĐỒ ÁN MÔN HỌC**

**TÊN ĐỀ TÀI: DỰ BÁO GIÁ CỔ PHIẾU**

**Giảng viên hướng dẫn : GV.TS Dương Thị Hiền Thanh**

**Sinh viên thực hiện : Đỗ Ngọc Phi - 2221050848**

**Đào Anh Tú - 2221050231**

**Cao Ngọc Thiện - 2221050445**

**Đặng Gia Định -**

**Ngành : Công nghệ thông tin**

**Chuyên ngành : Khoa học dữ liệu**

**Lớp :DCCTKH67A**

**Khóa học 2022 – 2026**

***HÀ NỘI (12/2024)***

[Giới thiệu 4](#_Toc185708645)

[Mục tiêu chính của dự án 4](#_Toc185708646)

[Ý nghĩa của dự án 5](#_Toc185708647)

[I.Thu thập dữ liệu và phân tích dữ liệu 5](#_Toc185708648)

[1.1.Thu thập dữ liệu 5](#_Toc185708649)

[1.2.Phân tích dữ liệu 6](#_Toc185708650)

[1.3.Lựa chọn tính năng 7](#_Toc185708651)

[1.4.Điểm RSI và mối quan hệ với khối lượng 9](#_Toc185708652)

[II.Thuật toán 10](#_Toc185708653)

[2.1.Extreme Gradient Boosting (XGBoost) 10](#_Toc185708654)

[2.2.Support Vector Regression (SVR) 11](#_Toc185708655)

[2.3.Auto Regressive Integrated Moving Average (ARIMA) 12](#_Toc185708656)

[III.Điều chỉnh mô hình 13](#_Toc185708657)

[3.1.Hyperparameter Tuning 13](#_Toc185708658)

[3.2.Support Vector Regression: 13](#_Toc185708659)

[3.3.XGBoost: 14](#_Toc185708660)

[3.4.Cross Validation For Time Series 15](#_Toc185708661)

[IV.Triển khai 18](#_Toc185708662)

[4.1.Tiền xử lý dữ liệu 19](#_Toc185708663)

[4.2.Mô hình SVR 22](#_Toc185708664)

[V.Áp dụng mô hình ARIMA 25](#_Toc185708665)

[5.1.Import thư viện 25](#_Toc185708666)

[5.2.Tải dữ liệu 26](#_Toc185708667)

[5.3.Kiểm tra tính dừng của chuỗi thời gian 27](#_Toc185708668)

[5.4.Xây dựng và huấn luyện mô hình ARIMA 29](#_Toc185708669)

[5.4.1.Tạo tập dữ liệu huấn luyện và kiểm tra 29](#_Toc185708670)

[5.4.2.Huấn luyện mô hình ARIMA 31](#_Toc185708671)

[5.4.3.Dự báo 32](#_Toc185708672)

[5.5.Đánh giá mô hình 33](#_Toc185708673)

[5.5.1.Tạo dữ liệu kiểm tra dịch chuyển 34](#_Toc185708674)

[5.5.2.Đánh giá mô hình 35](#_Toc185708675)

[5.6.Biểu đồ dự báo cuối cùng 37](#_Toc185708676)

[5.6.1.Chuẩn bị dữ liệu cho biểu đồ để dự báo 38](#_Toc185708677)

[5.6.2.Vẽ biểu đồ dự báo 39](#_Toc185708678)

[5.7.Kết luận 42](#_Toc185708679)

[VI.Áp dụng mô hình SVR 43](#_Toc185708680)

[6.1.Import các thư viện cần thiết 43](#_Toc185708681)

[6.2.Tải dữ liệu 44](#_Toc185708682)

[6.3.Tạo tập dữ liệu huấn luyện và kiểm tra 45](#_Toc185708683)

[6.3.1.Chia dữ liệu thành tập huấn và kiểm tra 45](#_Toc185708684)

[6.3.2.Chuẩn hoá dữ liệu 46](#_Toc185708685)

[6.3.3.Tạo tập dữ liệu huấn luyện và kiểm tra 47](#_Toc185708686)

[6.3.4.Định hình lại dữ liệu 48](#_Toc185708687)

[6.4.Xây dựng và huấn luyện mô hình 49](#_Toc185708688)

[6.4.1.Khởi tạo mô hình SVR 50](#_Toc185708689)

[6.4.2.Huấn luyện mô hình 50](#_Toc185708690)

[6.4.3.Thực hiện dự báo 50](#_Toc185708691)

[6.5.Đánh giá mô hình 51](#_Toc185708692)

[6.5.1.Đảo ngược quá trình chuẩn hóa 51](#_Toc185708693)

[6.5.2.Tính toán các chỉ số đánh giá 52](#_Toc185708694)

[6.6.Biểu đồ dự báo 55](#_Toc185708695)

[6.6.1.Tạo DataFrame từ các giá trị dự báo và giá trị thực tế 55](#_Toc185708696)

[6.6.2.Vẽ biểu đồ so sánh giữa giá trị thực tế và giá trị dự báo 55](#_Toc185708697)

[6.7.Dự báo giá cổ phiếu trong những ngày tiếp theo 57](#_Toc185708698)

[6.7.1.Dự báo 57](#_Toc185708699)

[6.7.2.Vẽ Biểu đồ dự báo cho các ngày tiếp theo 59](#_Toc185708700)

[6.8.Kết luận 61](#_Toc185708701)

[VII.So sánh hiệu quả của 2 mô hình 62](#_Toc185708702)

[7.1. Độ phức tạp 62](#_Toc185708703)

[7.1.1.Mô hình ARIMA 62](#_Toc185708704)

[7.2. Mô hình SVR 63](#_Toc185708705)

[VIII.Độ chính xác 64](#_Toc185708706)

[8.1.Mô hình ARIMA 64](#_Toc185708707)

[8.2.Mô hình SVR 64](#_Toc185708708)

[8.3.Kết luận về độ chính xác 65](#_Toc185708709)

[IX.Đề xuất cải tiến 66](#_Toc185708710)

[X.Tổng thể 66](#_Toc185708711)

[XI.Kết luận 66](#_Toc185708712)

[XII.Tài liệu tham khảo 67](#_Toc185708713)

# Giới thiệu

Thị trường chứng khoán là một trong những kênh đầu tư hấp dẫn và quan trọng nhất trong nền kinh tế toàn cầu. Tuy nhiên, biến động giá cổ phiếu thường khó lường và chịu ảnh hưởng bởi nhiều yếu tố phức tạp, từ tình hình kinh tế vĩ mô đến tâm lý thị trường. Việc dự báo chính xác giá cổ phiếu không chỉ giúp nhà đầu tư đưa ra quyết định đúng đắn mà còn góp phần nâng cao hiệu quả hoạt động của các tổ chức tài chính.

Dự đoán giá cổ phiếu là nhiệm vụ dự báo giá cổ phiếu trong tương lai dựa trên dữ liệu lịch sử và nhiều chỉ số thị trường khác nhau. Nhiệm vụ này bao gồm việc sử dụng các mô hình thống kê và thuật toán học máy để phân tích dữ liệu tài chính và đưa ra dự đoán về hiệu suất trong tương lai của một cổ phiếu nhằm giúp các nhà đầu tư đưa ra quyết định đầu tư sáng suốt bằng cách cung cấp dự báo về giá cổ phiếu trong tương lai. Báo cáo này là giới thiệu về dự đoán giá cổ phiếu của thị trường Việt Nam được chia thành các lĩnh vực công nghiệp bằng một số kỹ thuật học máy đơn giản với ba thuật toán: ARIMA, XGBooxt và Hồi quy vectơ hỗ trợ.

Các thuật ngữ chỉ số—Thống kê, Học máy, Cổ phiếu, Dự đoán, ARIMA, XGBoost, Hồi quy vectơ hỗ trợ.

## Mục tiêu chính của dự án

Các mục tiêu chính của dự án này như sau:

• Phân tích dữ liệu thị trường chứng khoán lịch sử từ thị trường Việt Nam, bao gồm giá cổ phiếu và các chỉ số tài chính có liên quan khác.

• Xác định các xu hướng, mô hình và mối quan hệ phụ thuộc chính của thị trường

có thể hỗ trợ hiểu được động lực cơ bản của biến động giá cổ phiếu.

• Phát triển các mô hình dự đoán bằng thuật toán học máy và kỹ thuật thống kê để dự báo chính xác giá cổ phiếu trong tương lai.

• Đánh giá hiệu suất của các mô hình dự đoán dựa trên các số liệu đánh giá đã thiết lập và so sánh chúng với các phương pháp dự báo truyền thống.

Khi kết thúc dự án này, chúng tôi mong đợi đạt được những kết quả sau:

• Phân tích toàn diện dữ liệu thị trường chứng khoán lịch sử, xác định các xu hướng và mô hình cụ thể cho thị trường chứng khoán Việt Nam.

• Phát triển các mô hình dự đoán chính xác có khả năng dự báo giá cổ phiếu trong tương lai cho các cổ phiếu Việt Nam đã chọn.

• Phân tích so sánh giữa hiệu suất của các mô hình dự đoán và các phương pháp dự báo truyền thống.

• Thông tin chi tiết và khuyến nghị cho các nhà đầu tư, tổ chức tài chính và nhà hoạch định chính sách để đưa ra quyết định sáng suốt trên thị trường chứng khoán Việt Nam.

Thông qua dự án này, chúng tôi mong muốn đóng góp vào khối lượng nghiên cứu ngày càng tăng về dự báo giá cổ phiếu đồng thời cung cấp các hàm ý thực tế cho các bên liên quan trên thị trường chứng khoán Việt Nam

## Ý nghĩa của dự án

Thị trường chứng khoán Việt Nam đã nổi lên như một thành phần quan trọng của bối cảnh tài chính của đất nước, mang đến cho các nhà đầu tư tiềm năng tăng trưởng đáng kể và cơ hội tăng giá vốn. Với nền kinh tế năng động và sự tham gia ngày càng tăng của cả nhà đầu tư trong và ngoài nước, việc dự đoán chính xác giá cổ phiếu đã trở thành một nhiệm vụ quan trọng đối với những người tham gia thị trường, các tổ chức tài chính và các nhà hoạch định chính sách.

Khả năng dự đoán giá cổ phiếu có thể cung cấp những thông tin chi tiết có giá trị cho các nhà đầu tư, giúp họ đưa ra quyết định sáng suốt về việc mua, bán hoặc nắm giữ cổ phiếu. Hơn nữa, những dự đoán chính xác có thể giúp các cơ quan quản lý thị trường xác định những rủi ro tiềm ẩn và thực hiện các biện pháp thích hợp để bảo vệ sự ổn định của thị trường.

Mặc dù việc dự đoán giá cổ phiếu rất quan trọng, nhưng nhiệm vụ này vẫn rất khó khăn do bản chất phức tạp của thị trường tài chính, sự tương tác của nhiều yếu tố khác nhau và sự hiện diện của những bất ổn cố hữu. Tuy nhiên, những tiến bộ trong kỹ thuật tính toán, thuật toán học máy và khả năng tiếp cận lượng lớn dữ liệu tài chính lịch sử đã mở ra những hướng đi mới để phát triển các mô hình dự đoán mạnh mẽ.

Báo cáo dự án này nhằm mục đích khám phá lĩnh vực dự đoán giá cổ phiếu trong bối cảnh thị trường chứng khoán Việt Nam. Bằng cách sử dụng phân tích dữ liệu, mô hình thống kê và kỹ thuật học máy, chúng tôi tìm cách dự báo hiệu suất trong tương lai.

# Thu thập dữ liệu và phân tích dữ liệu

## Thu thập dữ liệu

Khi chúng tôi tìm kiếm dữ liệu công khai trên Google Dataset Search, chúng tôi thấy rằng dữ liệu chủ yếu dành cho thị trường quốc tế, chẳng hạn như NYSE hoặc NASDAQ. Tuy nhiên, vì cổ phiếu là thứ phản ánh một thị trường cụ thể và cả chính trị, chúng tôi muốn chỉ lấy dữ liệu cho Việt Nam, từ một số thị trường Việt Nam (HOSE, HNX, UPCOM, v.v.)

Sau khi phân tích bảng điều khiển cổ phiếu để thu thập dữ liệu, chúng tôi thấy rằng tại Bảng điều khiển TCBS1 của Techcombank Securities có API riêng để thu thập dữ liệu web. Chúng tôi đã nghiên cứu trong API đó, xem xét ba phương pháp thu thập dữ liệu bằng cách sử dụng request, Selenium và urllib3 và quyết định sử dụng request dựa trên khối lượng dữ liệu và tốc độ thu thập trong đó:

• ticker: Mã chứng khoán

• resolution: Bước lưu trữ (ví dụ: 1 ngày, 1 tuần, v.v.)

• from và to: Thời gian bắt đầu và thời gian kết thúc (chuyển đổi thành

thời gian tuyệt đối)

Tuy nhiên, trong khi thu thập dữ liệu, một vấn đề khác xuất hiện là độ dài dữ liệu tối đa của tin nhắn JSON chỉ là 250 bản ghi. Vì vậy, chúng tôi quyết định sử dụng vòng lặp for để lấy dữ liệu từ thời điểm công ty niêm yết trên thị trường chứng khoán cho đến ngày 1 tháng 7 năm 2023.

Tóm lại, tập dữ liệu của chúng tôi sẽ chứa dữ liệu cho đến ngày 1 tháng 7 năm 2023 với 7 thuộc tính: Ngày giao dịch, Mở cửa, Cao, Thấp, Đóng cửa, Khối lượng và Mã chứng khoán.

## Phân tích dữ liệu

Một điều nữa chúng ta phải cân nhắc là mã cổ phiếu được chọn, vì nếu chúng ta chọn một mã ngẫu nhiên, sẽ không có thông tin chi tiết nào được nêu ra. Nhiều nghiên cứu đã chứng minh rằng tác động của kinh tế vĩ mô và chính trị chia thị trường chứng khoán thành một số lĩnh vực công nghiệp: xây dựng, công nghệ, tiêu dùng (dịch vụ, F&B, v.v.), tài chính và các ngành khác. Ở đây, để tiết kiệm thời gian và phân tích dễ dàng hơn, chúng tôi quyết định lấy dữ liệu từ 4 ngành đã nêu với các mã cổ phiếu này nhằm mục đích có được cái nhìn tổng quan về thị trường chứng khoán:

• Xây dựng: – LCS: CTCP Licogi.

– PTC: CTCP Đầu tư ICapital.

• Công nghệ: – VGI: CTCP Đầu tư Toàn cầu Viettel. – ITD: CTCP Công nghệ Tiên Phong.

• Tiêu dùng: – LSS: CTCP Đường Lam Sơn. – PLX: Tập đoàn Xăng dầu Việt Nam.

• Tài chính: – TCB: Ngân hàng Thương mại Cổ phần Kỹ thương Việt Nam. – VIG: Tổng công ty Chứng khoán Đầu tư Tài chính Việt Nam.

Ở đây, tôi sẽ tập trung vào việc chỉ ra lý do tại sao tôi nói rằng thị trường chứng khoán bị ảnh hưởng bởi kinh tế vĩ mô và chính trị, cùng với một số xu hướng chung về kinh tế.

A graph of stock prices

Description automatically generated

Như chúng ta thấy, giá cổ phiếu trong giai đoạn 2008 - 2012 đã giảm xuống. Điều này là do khủng hoảng kinh tế quốc tế, cũng ảnh hưởng đến Việt Nam. Và chính phủ Việt Nam đã áp dụng một số phương pháp vào kinh tế quốc gia, do đó, giá đã tăng nhẹ vào năm 2010, nhưng tiếp tục giảm xuống trong ngày vì phương pháp áp dụng không áp dụng được trong dài hạn.

Ngoài ra, sau đại dịch COVID-19, chính phủ Việt Nam đã áp dụng một số chính sách để thúc đẩy tăng trưởng kinh tế, do đó, giá cổ phiếu đã chứng kiến ​​sự gia tăng đáng kể. Tuy nhiên, sau khi đạt đỉnh tăng trưởng, vào năm 2023 và 2024, kinh tế Việt Nam sẽ suy giảm mạnh, theo một số chuyên gia kinh tế.

## Lựa chọn tính năng

A green and white chart

Description automatically generated

Để chọn tính năng phù hợp vì mục đích tiết kiệm thời gian và độ chính xác, chúng tôi phân tích các tập dữ liệu và vẽ bản đồ nhiệt của các mối tương quan giữa các tính năng.

Như chúng ta có thể thấy, mối tương quan của Mở, Cao, Thấp, Đóng cao hơn đáng kể so với Khối lượng. Hơn nữa, điểm số cho Mở, Cao, Thấp và Đóng là như nhau và về mặt thống kê, tất cả đều là giá của cổ phiếu trong một ngày, vì vậy chúng tôi quyết định chỉ sử dụng một trong số chúng, tính năng Đóng.

Tuy nhiên, chúng tôi sẽ tiếp tục sử dụng Khối lượng trong phần khác của

dự án, đó là dự đoán khối lượng giao dịch theo điểm số RSI (điểm số chỉ số sức mạnh tương đối)

## Điểm RSI và mối quan hệ với khối lượng

Như tôi đã đề cập ở trên, chúng ta sẽ sử dụng chỉ số sức mạnh tương đối x để dự đoán khối lượng. Để bắt đầu, chúng ta sẽ xem xét RSI là gì và nó liên quan như thế nào đến cổ phiếu. Để tính RSI, chúng ta định nghĩa các chỉ báo hướng lên (U) và hướng xuống (D), đó là:

A math equations on a white background

Description automatically generated

trong đó t là ngày được tính toán và Pt là giá cổ phiếu cho ngày đó.

Sau đó upt và downt là số trung bình của các đợt tăng giá và các đợt giảm giá của giá đóng cửa trong n ngày qua:

A math equations with numbers and symbols

Description automatically generated

trong đó s là tham số trơn tru, thông thường, s = 2. Cuối cùng, chúng ta có được điểm RSI bằng cách sử dụng công thức:

A math equations on a white background

Description automatically generated

RSI có xu hướng duy trì trạng thái tĩnh hơn trong xu hướng tăng so với xu hướng giảm. Điều này có lý vì RSI đo lường mức tăng so với mức giảm. Trong xu hướng tăng, có nhiều mức tăng hơn, giúp RSI duy trì ở mức cao hơn. Mặt khác, trong xu hướng giảm, RSI có xu hướng duy trì ở mức thấp hơn. Trong xu hướng tăng, RSI có xu hướng duy trì trên 30 và thường xuyên chạm mức 70. Trong xu hướng giảm, hiếm khi thấy RSI vượt quá 70 và chỉ báo này thường xuyên chạm mức 30 hoặc giảm xuống dưới ngưỡng này. Nhữnghướng dẫn này có thể giúp xác định sức mạnh của xu hướng và phát hiện các sự đảo ngược tiềm năng. Ngược lại, xu hướng giảm thì đúng. Điều này có nghĩa là nếu xu hướng giảm không thể đạt đến 30 hoặc thấp hơn rồi tăng trên 70, thì xu hướng giảm đó được cho là yếu đi.

A graph with a line graph and arrows

Description automatically generated with medium confidence

Do đó, chúng ta có thể sử dụng RSI để dự đoán khối lượng nhằm có được thông tin chi tiết về "Có bao nhiêu cổ phiếu được trao đổi theo thời gian và xu hướng hiện tại của mã cổ phiếu là gì?" Từ đó, nhà đầu tư có thể quyết định khối lượng giao dịch của riêng họ để tối đa hóa lợi ích của họ.

# Thuật toán

## Extreme Gradient Boosting (XGBoost)

Ensemble Learning là một trong những phương pháp Machine Learning kết hợp một số mô hình cơ sở để đưa ra một mô hình tối ưu. Thay vì tạo một mô hình và triển khai mô hình này với kỳ vọng sẽ đưa ra dự đoán chính xác, các phương pháp Ensemble sẽ tính đến rất nhiều mô hình và tính trung bình các mô hình đó để tạo ra kết quả cuối cùng.

Boosting là một trong ba phương pháp chính trong Ensemble Learning. Khác với bagging là đào tạo nhiều cây quyết định song song trên các mẫu khác nhau của cùng một tập dữ liệu và tạo ra dự đoán bằng cách tính trung bình, phương pháp Boosting kết hợp một tập hợp các học viên yếu thành các học viên mạnh để giảm thiểu lỗi đào tạo

Trong Boosting, một mẫu dữ liệu ngẫu nhiên được chọn, được trang bị một mô hình và sau đó đào tạo tuần tự - nghĩa là, mỗi mô hình cố gắng bù đắp cho điểm yếu của mô hình tiền nhiệm.

Extreme Gradient Boosting Machine (XGBM) là phiên bản mới nhất của máy Boosting. Trong XGBM, các cây được thêm vào tuần tự (từng cây một) để học hỏi từ các lỗi của các cây trước đó và cải thiện chúng.

A diagram of a computer algorithm

Description automatically generated

## Support Vector Regression (SVR)

Support Vector Machine (SVM) là một thuật toán học máy cho các vấn đề phân loại. Sau khi SVM được phát minh, nó đã được cập nhật để hỗ trợ hồi quy vectơ (SVR) bằng cách kết hợp một dạng hàm mất mát được gọi là hàm mất mát không nhạy cảm với ε, hàm này phạt các điểm dữ liệu miễn là chúng vượt quá ε, SVR trong không gian đặc điểm có nhiều chiều trở thành phương pháp hồi quy dựa trên hạt nhân phi tuyến tính xác định siêu phẳng hồi quy tối ưu với rủi ro cấu trúc tối thiểu.

Chức năng của phương pháp SVR là: 

trong đó φ biểu thị phép ánh xạ phi tuyến tính từ không gian đầu vào đến không gian đặc điểm, w là vectơ hệ số trọng số và b là hằng số độ lệch. w và b được ước tính bằng cách giảm thiểu vấn đề tối ưu hóa sau:

A math equations and numbers

Description automatically generated with medium confidence

## Auto Regressive Integrated Moving Average (ARIMA)

ARIMA thực chất là một lớp mô hình giải thích một chuỗi thời gian nhất định dựa trên các giá trị quá khứ của chính nó, tức là độ trễ của chính nó và các lỗi dự báo bị trễ, do đó phương trình có thể được sử dụng để dự báo các giá trị trong tương lai.

Bất kỳ chuỗi thời gian 'không theo mùa' nào thể hiện các mẫu vàkhông phải là nhiễu trắng ngẫu nhiên đều có thể được mô hình hóa bằng các mô hình ARIMA.

Một mô hình ARIMA được đặc trưng bởi 3 thuật ngữ: p, d, q, trong đó,

• p là bậc của thuật ngữ AR

• q là bậc của thuật ngữ MA

• d là số phép tính sai phân cần thiết để làm cho chuỗi thời gian dừng.

Bước đầu tiên để xây dựng mô hình ARIMA là làm cho chuỗi thời gian dừng. Bởi vì, thuật ngữ 'Tự hồi quy' trong ARIMA có nghĩa là nó là một mô hình hồi quy tuyến tính sử dụng độ trễ của chính nó làm các yếu tố dự báo. Các mô hình hồi quy tuyến tính, như bạn đã biết, hoạt động tốt nhất khi các yếu tố dự báo không tương quan và độc lập với nhau. Cách tiếp cận phổ biến nhất là lấy chênh lệch. Nghĩa là trừ giá trị trước đó khỏi giá trị hiện tại. Đôi khi, tùy thuộc vào độ phức tạp của chuỗi, có thể cần nhiều hơn một chênh lệch. Do đó, giá trị d là số chênh lệch tối thiểu cần thiết để làm cho chuỗi dừng. Và nếu chuỗi thời gian đã dừng, thì d = 0. Tiếp theo, các điều khoản p và q là gì?

p là bậc của điều khoản 'Tự hồi quy' (AR). Nó đề cập đến số độ trễ của Y được sử dụng làm yếu tố dự báo. Và q là bậc của điều khoản 'Trung bình động' (MA). Nó đề cập đến số lỗi dự báo bị trễ cần đưa vào Mô hình ARIMA.

Mô hình hồi quy tự động thuần túy (chỉ AR) là mô hình mà Yt chỉ phụ thuộc vào độ trễ của chính nó. Nghĩa là, Yt là hàm số của 'độ trễ của Yt'.



trong đó, Yt−1 là độ trễ1 của chuỗi, beta1 là hệ số của độ trễ1 mà mô hình ước tính và α là số hạng chặn, cũng được ước tính bởi mô hình.

Một mô hình Trung bình động thuần túy (chỉ MA) là mô hình mà Yt chỉ phụ thuộc vào các lỗi dự báo bị trễ.



trong đó các điều khoản lỗi là các lỗi của các mô hình hồi quy tự động l của các độ trễ tương ứng. Các lỗi Et và Et−1 là các lỗi từ các phương trình sau:

A black and white math equation

Description automatically generated with medium confidence

Mô hình ARIMA là mô hình mà chuỗi thời gian được phân biệt ít nhất một lần để làm cho nó dừng và bạn kết hợp các điều khoản AR và MA. Vì vậy, phương trình trở thành:



# Điều chỉnh mô hình

## Hyperparameter Tuning

Điều chỉnh siêu tham số chủ yếu là tìm tập hợp các giá trị tham số sao cho góp phần vào hiệu suất của mô hình, tối đa hóa hiệu quả nhưng vẫn đảm bảo tính phức tạp trong một tập dữ liệu. Quá trình này được sử dụng không chỉ để cải thiện sức mạnh dự đoán của mô hình mà còn cải thiện tốc độ, giúp mô hình chạy nhanh hơn.

**Chọn siêu tham số**: Đây có lẽ là vấn đề khó khăn nhất trong việc điều chỉnh siêu tham số. Lý do là rất khó để chọn siêu tham số quan trọng nhất để điều chỉnh và các siêu tham số có tương tác và ảnh hưởng lẫn nhau trong quá trình. Do đó, việc đánh giá tại một thời điểm không hiệu quả vì khi thay đổi sang siêu tham số khác, việc đánh giá sẽ hoàn toàn khác. Trong mô hình này, theo trực giác và thông qua thử nghiệm, chúng ta chỉ chọn siêu tham số có khả năng có tác động đáng kể nhất đến mô hình để điều chỉnh thử nghiệm.

## Support Vector Regression:

Trong SVR (Support Vector Regression), tham số chính quy hóa, được ký hiệu là C, kiểm soát sự đánh đổi giữa lỗi khớp và độ phức tạp của mô hình. Nó xác định hình phạt cho các điểm dữ liệu nằm ngoài biên độ hoặc vi phạm dung sai hồi quy. Giá trị C cao hơn trong SVR ngụ ý dung sai lỗi nhỏ hơn, khiến mô hình cố gắng khớp dữ liệu đào tạo chặt chẽ hơn. Điều này có thể dẫn đến quá khớp, khi đó mô hình trở nên quá cụ thể đối với dữ liệu đào tạo và hoạt động kém trên dữ liệu chưa biết. Mặt khác, giá trị C thấp hơn cho phép biên độ rộng hơn và cho phép nhiều lỗi hơn, tạo ra mô hình linh hoạt hơn, có thể khái quát hóa tốt hơn đối với dữ liệu chưa biết. Trong mã được cung cấp, các giá trị của C là [0,01, 0,1, 1, 100] và mô hình SVR sẽ được đào tạo và kiểm tra bằng cách sử dụng từng giá trị này để tìm giá trị tối ưu của C cho tác vụ hồi quy đã cho.

• kernel: Tham số này xác định loại hàm kernel được sử dụng trong SVR để chuyển đổi không gian đặc điểm. Có ba kernel thường được sử dụng:

– linear: Kernel tuyến tính, được sử dụng khi dữ liệu có thể được phân tách tốt bằng một siêu phẳng. – rbf: Kernel Gaussian (RBF - Radial Basis Function), là kernel thường được sử dụng nhất, được sử dụng khi dữ liệu không thể phân tách tuyến tính trong không gian ban đầu. – sigmoid: Kernel sigmoid, được sử dụng khi dữ liệu không thể phân tách tuyến tính và có các đặc điểm tương tự như hàm sigmoid trong hồi quy logistic. Mô hình sẽ được đào tạo và kiểm tra bằng từng loại kernel này.

• γ: Tham số này điều chỉnh ảnh hưởng của điểm dữ liệu đào tạo lên các điểm dữ liệu khác. Giá trị gamma cao hơn giới hạn ảnh hưởng chỉ đến các điểm gần, trong khi giá trị gamma thấp hơn sẽ lan tỏa ảnh hưởng đến cả các điểm gần và xa. Trong mã đã cho, các giá trị gamma là [0,01, 0,001, 1] và mô hình sẽ được đào tạo và thử nghiệm với mỗi giá trị này.

Các tham số này cho phép điều chỉnh mô hình SVR để tìm các siêu phẳng phân tách tối ưu trong không gian đặc điểm để có hiệu suất tốt hơn.

## XGBoost:

• n ước tính: Tham số này biểu thị số lượng cây quyết định riêng lẻ (người học yếu) sẽ được xây dựng trong mô hình XGboost. Việc tăng số lượng ước tính có thể cải thiện hiệu suất của mô hình bằng cách giảm độ lệch, nhưng cũng làm tăng thời gian tính toán. Trong mã được cung cấp các giá trị của n ước tính là [100, 500, 1000] và mô hình sẽ được đào tạo và kiểm tra với từng giá trị này.

• Độ sâu tối đa: Tham số này xác định độ sâu tối đa của mỗi cây quyết định trong mô hình GBM. Độ sâu tối đa cao hơn cho phép mô hình tìm hiểu các mối quan hệ phức tạp hơn trong dữ liệu nhưng có thể dẫn đến quá khớp. Mặt khác, độ sâu tối đa thấp hơn sẽ hạn chế độ phức tạp của cây, giảm quá khớp nhưng có khả năng hy sinh một số sức mạnh dự đoán. Trong mã được cung cấp, các giá trị của độ sâu tối đa là [3,6, 10] và mô hình sẽ được đào tạo và kiểm tra với mỗi giá trị này.

• Tốc độ học: Tham số này kiểm soát sự đóng góp của mỗi cây trong tập hợp. Tốc độ học nhỏ hơn làm mô hình bảo thủ hơn bằng cách giảm tác động của mỗi cây, dẫn đến việc học chậm hơn nhưng có khả năng tổng quát hóa tốt hơn. Ngược lại, tốc độ học cao hơn cho phép mô hình học nhanh hơn nhưng cũng có thể dẫn đến quá khớp.

Trong mã được cung cấp, các giá trị của tốc độ học là [0,01, 0,05, 0,1] và mô hình sẽ được đào tạo và kiểm tra với mỗi giá trị này.

## Cross Validation For Time Series

Khi xây dựng một mô hình, điều quan trọng là phải đánh giá hiệu suất của mô hình đó. Xác thực chéo là một kỹ thuật thống kê có thể hỗ trợ quá trình này. Một phương pháp thường được sử dụng là Xác thực chéo K-fold, trong đó tập dữ liệu được chia thành nhiều tập hợp con hoặc nếp gấp. Mô hình được đào tạo trên tất cả các nếp gấp trừ một nếp gấp và sau đó được thử nghiệm trên nếp gấp còn lại. Quá trình này được lặp lại cho từng nếp gấp, cho phép đánh giá toàn diện hiệu suất của mô hình.

Tuy nhiên, chúng ta không thể sử dụng quy trình này trong Chuỗi thời gian. Xác thực chéo trở nên khó khăn hơn trong bối cảnh dữ liệu chuỗi thời gian. Việc chỉ định ngẫu nhiên các mẫu cho các tập kiểm tra và đào tạo là không phù hợp vì nó sẽ liên quan đến việc sử dụng các giá trị trong tương lai để dự đoán các giá trị trong quá khứ, điều này là phi logic. Nói cách khác, cần tránh nhìn về tương lai trong khi đào tạo mô hình.

Xác thực chéo chuỗi thời gian: Giải pháp là một cách xác thực chéo độc đáo cho chuỗi thời gian. Xác thực chéo trên cơ sở lăn là một kỹ thuật có thể được áp dụng để xác thực chéo mô hình chuỗi thời gian. Bắt đầu với một tập hợp con nhỏ dữ liệu cho mục đích đào tạo, đưa ra dự đoán cho các điểm dữ liệu tiếp theo, sau đó đánh giá độ chính xác của các dự đoán. Tập dữ liệu đào tạo sau có cùng các điểm dữ liệu dự đoán và các điểm dữ liệu bổ sung được dự báo sau đó. Các phân chia này phải tuân theo quy tắc:

• Mỗi tập kiểm tra chứa các quan sát duy nhất

• Các quan sát từ tập huấn luyện xảy ra trước bộ kiểm tra tương ứng của chúng

Trong dự án này, chúng tôi giới thiệu hai loại Xác thực chéo lồng nhau. Chúng tôi sẽ giải quyết trường hợp cổ phiếu có dữ liệu kéo dài nhiều ngày:

• Dự đoán nửa sau

• Đào tạo chuyển tiếp ngày



**Dự đoán nửa thứ hai**: Phương pháp đầu tiên được gọi là Dự đoán nửa thứ hai, đóng vai trò là phương pháp xác thực chéo lồng nhau (CV) cơ bản. Phương pháp "Dự đoán nửa thứ hai" bao gồm một phân tách đào tạo/kiểm tra duy nhất, trong đó nửa đầu của dữ liệu được sử dụng để đào tạo và nửa sau được sử dụng để kiểm tra.

A diagram of a model

Description automatically generated

Trong phương pháp này, nửa đầu của dữ liệu được phân bổ cho tập huấn luyện và nửa sau được chỉ định là tập kiểm tra, trong đó kích thước của tập xác thực thay đổi tùy thuộc vào vấn đề cụ thể đang được giải quyết. Điều quan trọng là phải duy trì thứ tự thời gian, đảm bảo rằng tập xác thực bao gồm dữ liệu sau tập con huấn luyện.

Ưu điểm của phương pháp này là tính đơn giản khi triển khai. Tuy nhiên, nó có một hạn chế là tập kiểm tra được chọn tùy ý, điều này có thể gây ra sai lệch hoặc dẫn đến kết quả không đáng tin cậy. Để giải quyết vấn đề này, điều quan trọng là phải duy trì thứ tự thời gian của dữ liệu, đảm bảo rằng tập xác thực, được sử dụng để lựa chọn mô hình hoặc điều chỉnh siêu tham số, bao gồm dữ liệu đến sau tập con huấn luyện. Điều này giúp đánh giá hiệu suất tổng quát hóa của mô hình trên dữ liệu tương lai chưa biết.

**Chuỗi ngày chuyển tiếp**: Phương pháp xác thực chéo lồng nhau Dự đoán nửa thứ hai ở trên có một hạn chế là việc lựa chọn tùy ý tập kiểm tra giữ lại có thể dẫn đến ước tính sai lệch về lỗi dự đoán khi đánh giá mô hình trên một tập kiểm tra độc lập. Để khắc phục nhược điểm này và có được ước tính đáng tin cậy hơn về lỗi dự đoán mô hình, một cách tiếp cận phổ biến là tạo nhiều phân tách đào tạo/kiểm tra và tính trung bình các lỗi trên tất cả các phân tách. Một kỹ thuật hiệu quả để đạt được điều này được gọi là Day Forward-Chaining.

Day Forward-Chaining dựa trên các nguyên tắc của forward-chaining và rolling-origin-recalibration evaluation. Phương pháp này bao gồm việc xem xét một cách có hệ thống từng ngày như một tập kiểm tra trong khi gán tất cả dữ liệu trước đó vào tập huấn luyện. Ví dụ, nếu chúng ta có một tập dữ liệu kéo dài năm ngày, Day Forward-Chaining sẽ tạo ra ba lần phân tách huấn luyện và kiểm tra khác nhau, đảm bảo rằng có ít nhất một ngày dữ liệu huấn luyện và xác thực khả dụng.

A diagram of a model

Description automatically generated

Bằng cách sử dụng phương pháp này, chúng tôi tạo ra nhiều phân chia đào tạo/kiểm tra, cho phép chúng tôi đánh giá hiệu suất của mô hình trong các khoảng thời gian khác nhau. Mỗi phân chia đại diện cho một kịch bản đánh giá riêng biệt, trong đó mô hình được đào tạo trên dữ liệu lịch sử và được thử nghiệm vào những ngày tiếp theo. Sự đa dạng của các phân chia này giúp nắm bắt khả năng khái quát hóa và đưa ra dự đoán chính xác của mô hình trong các khoảng thời gian khác nhau.

Để tính toán ước tính mạnh mẽ về lỗi mô hình, lỗi dự đoán trên mỗi phân chia được tính trung bình. Quá trình tính trung bình này làm mịn mọi biến động hoặc thay đổi có thể xảy ra do các phân chia đào tạo/kiểm tra cụ thể và cung cấp đánh giá ổn định hơn về hiệu suất của mô hình.

**So sánh và Kết luận**: Phương pháp Dự đoán Nửa sau dựa trên một phân chia đào tạo/kiểm tra duy nhất, có thể không nắm bắt được toàn bộ tính biến động của dữ liệu hoặc cung cấp ước tính mạnh mẽ về lỗi mô hình. Phương pháp Dự đoán Nửa sau có thể không đánh giá hiệu quả khả năng khái quát hóa của mô hình đối với dữ liệu tương lai chưa biết, vì tập kiểm tra được chọn tùy ý.

Trong khi đó, đào tạo theo ngày tạo ra nhiều phân chia đào tạo/kiểm tra bằng cách coi mỗi ngày là một tập kiểm tra riêng biệt. Bằng cách tính trung bình hiệu suất của mô hình trên các phân chia này, có thể thu được ước tính mạnh mẽ hơn về lỗi của mô hình, tính đến các biến thể và dao động trong các khoảng thời gian khác nhau. Những điều này phù hợp với mục tiêu dự báo các giá trị tương lai trong phân tích chuỗi thời gian. Bằng cách đào tạo mô hình trên dữ liệu trong quá khứ và đánh giá nó vào những ngày tiếp theo, mô hình được thử nghiệm trên các điểm dữ liệu tương lai chưa thấy, cho phép đánh giá tốt hơn về khả năng khái quát hóa của nó.

Tóm lại, đào tạo theo ngày cung cấp các lợi thế so với phương pháp Dự đoán nửa sau về mặt lựa chọn tập kiểm tra, tính đến động lực thời gian, độ mạnh của đánh giá và khái quát hóa cho dữ liệu chưa thấy trong tương lai. Bằng cách xem xét một cách có hệ thống từng ngày như một tập kiểm tra, đào tạo theo ngày cung cấp đánh giá đại diện hơn về hiệu suất của mô hình và nắm bắt tốt hơn sự phụ thuộc về thời gian có trong dữ liệu chuỗi thời gian.

# Triển khai

Dữ liệu bao gồm các biến số sau:

• Open: Giá mở cửa của cổ phiếu trong ngày.

• High: Giá cao nhất của cổ phiếu trong ngày.

• Low: Giá thấp nhất của cổ phiếu trong ngày.

• Close: Giá đóng cửa của cổ phiếu trong ngày.

• Volume: Khối lượng cổ phiếu được giao dịch trong ngày.

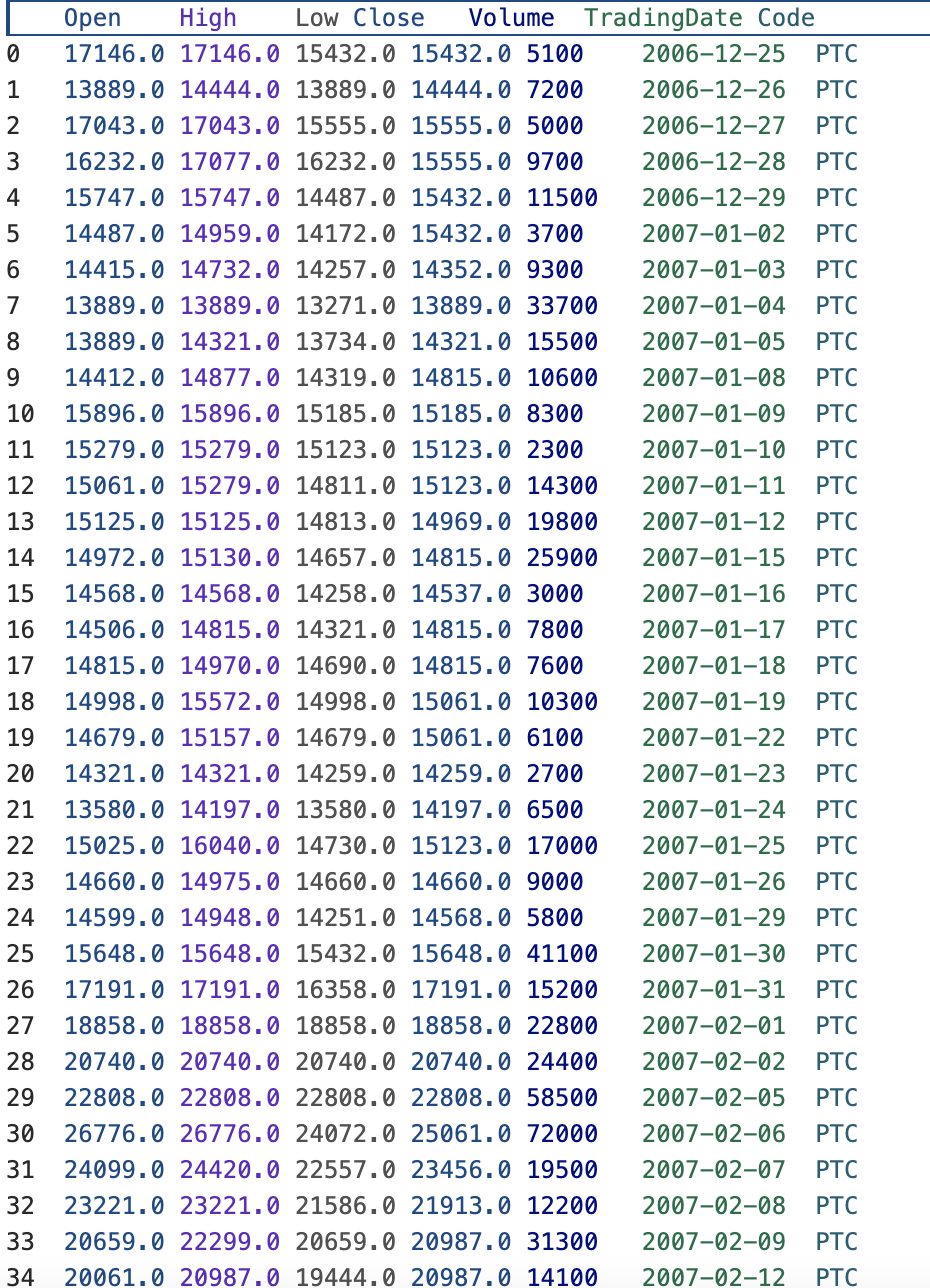
• TradingDate: Ngày giao dịch.

• Code: Mã cổ phiếu, trong trường hợp này là "PTC".

Các biến số này sẽ được sử dụng để xây dựng và huấn luyện các mô hình dự báo giá cổ phiếu như ARIMA và SVR. Cụ thể:

• Giá đóng cửa (Close) thường được sử dụng làm biến mục tiêu trong các mô hình dự báo.

• Giá mở cửa (Open), giá cao nhất (High), giá thấp nhất (Low), và khối lượng giao dịch (Volume) có thể được sử dụng làm các biến đầu vào để cải thiện độ chính xác của mô hình.



## Tiền xử lý dữ liệu

Trong ARIMA\_Model.ipynb, các bước tiền xử lý bao gồm:

* Import các thư viện cần thiết:

# 1. Thư viện xử lý dữ liệu

import numpy as np # Xử lý mảng, ma trận

import pandas as pd # Xử lý dữ liệu dạng bảng

import warnings

warnings.filterwarnings('ignore') # Ẩn các cảnh báo

# 2. Thư viện kiểm tra và xử lý chuỗi thời gian

from statsmodels.tsa.stattools import adfuller # Kiểm tra tính dừng

from statsmodels.tsa.seasonal import seasonal\_decompose # Phân rã theo mùa

from statsmodels.graphics.tsaplots import plot\_acf, plot\_pacf # Đồ thị ACF, PACF

# 3. Thư viện chuẩn hóa dữ liệu

from sklearn.preprocessing import MinMaxScaler # Scale dữ liệu về [0,1]

from sklearn.preprocessing import StandardScaler # Chuẩn hóa dữ liệu

* Load và kiểm tra dữ liệu:

# Load data

df = pd.read\_csv('/Users/dophi/Desktop/dudoangiacophieu-main/Dataset/construction/PTC.csv', sep='\t', index\_col=6)

close=df[['Close']]

close.head(10)

Kết quả:A screenshot of a computer screen

Description automatically generated

Chia tập train/test (70-30):

Trong phân tích dữ liệu chuỗi thời gian, việc chia tập dữ liệu là bước quan trọng đầu tiên. Nhóm sẽ chia theo tỷ lệ 70% cho training và 30% cho testing để đảm bảo mô hình có đủ dữ liệu để học và kiểm tra.

n = int(len(df) \* 0.7)

m = int(len(df) \* 0.7)

df\_train = close.copy()[:n][['Close']]

df\_test = close.copy()[m:][['Close']]

* Chuẩn hóa dữ liệu:

Chuẩn hóa dữ liệu giúp đưa các giá trị về cùng một thang đo, tránh việc các giá trị lớn chi phối quá trình học. sử dụng MinMaxScaler để đưa dữ liệu về khoảng [0,1].

# Scale train data

scaler = MinMaxScaler()

df\_train['Close'] = scaler.fit\_transform(df\_train)

# Scale test data

df\_test['Close'] = scaler.transform(df\_test)

* Kiểm tra tính dừng (Stationarity):

Tính dừng là yêu cầu cần có của mô hình ARIMA. Một chuỗi dừng có các đặc tính thống kê (như mean, variance) không thay đổi theo thời gian. Nhóm dùng Augmented Dickey-Fuller test để kiểm tra.

# Kiểm tra bằng Dickey-Fuller test

def check\_stationarity(ts\_data):

# Tính rolling statistics

roll\_mean = ts\_data.rolling(30, center=True, closed='both').mean()

roll\_std = ts\_data.rolling(5, center=True, closed='both').std()

# Thực hiện Dickey-Fuller test

df\_test = adfuller(ts\_data, regresults=False)

* Xử lý để đạt tính dừng:

Nếu chuỗi không dừng, lúc này cần biến đổi nó thành chuỗi dừng. Phương pháp phổ biến là lấy sai phân (differencing).

# Lấy sai phân bậc 1 để đạt tính dừng

df\_final\_diff = df\_final - df\_final.shift()

df\_final\_diff.dropna(inplace=True)

check\_stationarity(df\_final\_diff)

* Chuẩn bị dữ liệu cho dự đoán:

Bước cuối cùng là chuẩn bị dữ liệu theo định dạng phù hợp cho việc dự đoán. Đó là tạo cửa sổ dự đoán (forecast horizon) và tổ chức dữ liệu theo cấu trúc phù hợp.

# Tạo cửa sổ training

training\_window = 30

history = [x for x in train\_ts]

history = history[(-training\_window):]

# Shift dữ liệu test theo horizon

df\_test\_shifted = df\_test.copy()

for t in range(1, HORIZON):

df\_test\_shifted['Close+'+str(t)] = df\_test\_shifted['Close'].shift(-t)

Qua 5 bước xử lý này, Nhóm đã có một bộ dữ liệu sạch, chuẩn hóa và phù hợp cho việc huấn luyện mô hình ARIMA. Việc chuẩn bị dữ liệu kỹ lưỡng sẽ giúp tăng đáng kể hiệu quả của mô hình dự đoán.

Chuyển đổi timestamp: Đảm bảo index là dạng datetime

Chuẩn hóa giá trị: Đưa về khoảng [0,1] bằng MinMaxScaler

Kiểm tra và xử lý tính dừng:

• Kiểm tra bằng Augmented Dickey-Fuller test

• Lấy sai phân nếu chuỗi không dừng

* Tạo cửa sổ dự đoán:

• Dùng 30 ngày để training

• Shift dữ liệu test theo số bước dự đoán

* Đảm bảo không có missing values:

• dropna() cho dữ liệu sau khi xử lý sai phân

• Xử lý các giá trị NaN trong quá trình shift

## Mô hình SVR

Trong mô hình SVR các bước tiền xử lý dữ liệu được thực hiện trong file DataPreprocessing.py và SVR-Model.ipynb

* Trước hết cần import thư viện cần thiết:

import pandas as pd

import numpy as np

from sklearn.preprocessing import StandardScaler

* Đọc dữ liệu:

Đầu tiên, cần đọc dữ liệu từ file CSV chứa thông tin giá cổ phiếu. Việc này được thực hiện thông qua thư viện pandas - một công cụ mạnh mẽ để xử lý dữ liệu dạng bảng trong Python

df = pd.read\_csv('/Users/dophi/Desktop/dudoangiacophieu-main/Dataset/construction/PTC.csv', sep='\t', index\_col=0)

df.head()

Kết quả:

A screen shot of a black board with white text

Description automatically generated

Đoạn code này sẽ hiển thị thông tin tổng quan về dữ liệu, sẽ nắm được cấu trúc và kích thước của dataset.

* Xử lý dữ liệu thời gian:

Tiếp theo, cần chuẩn hóa định dạng thời gian và loại bỏ các giá trị null. Đây là bước quan trọng để đảm bảo tính nhất quán của dữ liệu

df['TradingDate'] = pd.to\_datetime(df['TradingDate'], format='%Y-%m-%d')

df = df.dropna()

Việc chuẩn hóa thời gian giúp các phép tính toán và phân tích chuỗi thời gian được chính xác hơn.

* Chuẩn bị dữ liệu dự đoán:

Trong giai đoạn này, cần phải tập trung vào giá đóng cửa (Close price) - biến mục tiêu cần dự đoán

# Create a new dataframe

data = df.filter(['Close'])

# Convert the dataframe to a numpy array

dataset = data.values

Chuyển đổi dữ liệu sang dạng mảng numpy giúp tối ưu hóa các phép tính toán về sau.

* Chuẩn hoá dữ liêu:

StandardScaler được sử dụng để chuẩn hóa dữ liệu về cùng một thang đo.

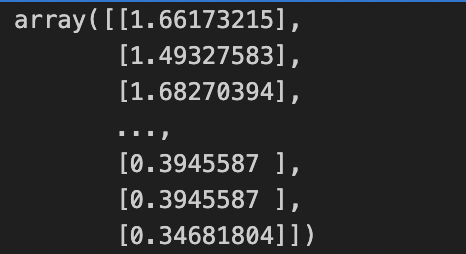
# scaler = MinMaxScaler(feature\_range=(0,1))

scaler = StandardScaler()

scaled\_data = scaler.fit\_transform(dataset)

scaled\_data

Kết quả:



Bước này không thể thiếu vì:

• Loại bỏ ảnh hưởng của đơn vị đo

• Đưa dữ liệu về phân phối chuẩn

• Cải thiện hiệu suất của mô hình SVR

* Tạo chuỗi dữ liệu theo time steps:

Đây là bước then chốt để xây dựng các mẫu dự đoán chuỗi thời gian.

def create\_dataset(dataset, time\_step=1):

dataX, dataY = [], []

for i in range(len(dataset)-time\_step-1):

a = dataset[i:(i+time\_step), 0]

dataX.append(a)

dataY.append(dataset[i + time\_step, 0])

return np.array(dataX), np.array(dataY)

time\_step = 30

x\_for\_train, y\_for\_train = create\_dataset(scaled\_data, time\_step)

Hàm này tạo ra các chuỗi 30 ngày liên tiếp, giúp mô hình học được mối quan hệ thời gian trong dữ liệu.

* Chia dữ liệu train/test:

Cuối cùng, chia dữ liệu thành hai tập

training\_size = int(len(dataset)\*0.8)

test\_size = len(dataset)-training\_size

train\_data = scaled\_data[0:training\_size]

test\_data = scaled\_data[training\_size:len(dataset)]

Kết quả:

Việc chia theo tỷ lệ 80/20 giúp:

• Có đủ dữ liệu để huấn luyện mô hình

• Đảm bảo tập test đủ lớn để đánh giá hiệu quả

• Tránh hiện tượng overfitting

# Áp dụng mô hình ARIMA

## Import thư viện

**from datetime import datetime**

**import numpy as np**

**import warnings**

**warnings.filterwarnings('ignore')**

**from statsmodels.tsa.arima.model import ARIMA**

**from statsmodels.tsa.statespace.sarimax import SARIMAX**

**from statsmodels.tsa.stattools import adfuller**

**from statsmodels.tsa.seasonal import seasonal\_decompose**

**from statsmodels.graphics.tsaplots import plot\_acf, plot\_pacf**

**from sklearn.metrics import mean\_squared\_error, r2\_score**

**from pandas.plotting import autocorrelation\_plot**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**import math**

**import mplfinance as mpf**

**from sklearn.preprocessing import MinMaxScaler, StandardScaler**

* Numpy, pandas ở đây sử dụng để xử lý và tính toán số học
* Matplotlib.pyplot chủ yếu tạo biểu đồ để trực quan hoá dữ liệu
* Statsmodels chứa các công cụ phân tích chuỗi thời gian như Arima, ADF test, vv
* Sklearn giúp đánh giá mô hình bằng các chỉ số như MSE, R2
* Mplfinance để vẽ biểu đồ tài chính
* Warnings dùng để tắt các cảnh báo không cần thiết trong bài

## 5.2. Tải dữ liệu

Trong bước này, thực hiện tải dữ liệu giá cổ phiếu PTC từ tệp CSV. Dữ liệu này chứa thông tin về giá cổ phiếu theo thời gian, bao gồm các cột như giá mở cửa, giá đóng cửa, giá cao nhất, giá thấp nhất và khối lượng giao dịch. Để đơn giản hóa, sẽ chỉ sử dụng cột giá đóng cửa (Close) cho việc phân tích và dự báo.

Đầu tiên, sử dụng thư viện pandas để đọc dữ liệu từ tệp CSV. Tệp dữ liệu được lưu trữ tại đường dẫn PTC.csv. tiếp đến sử dụng hàm read\_csv của pandas để đọc dữ liệu và chỉ định rằng cột thứ 6 sẽ được sử dụng làm chỉ số (index) cho DataFrame.

# Load data

df = pd.read\_csv('/Users/dophi/Desktop/dudoangiacophieu-main/Dataset/construction/PTC.csv', sep='\t', index\_col=6)

close = df[['Close']]

close.head(10)

Kết quả:

A screenshot of a computer

Description automatically generated

## Kiểm tra tính dừng của chuỗi thời gian

Trước khi áp dụng mô hình ARIMA, cần kiểm tra tính dừng của chuỗi thời gian. Một chuỗi thời gian được gọi là dừng nếu các thống kê của nó (như trung bình và phương sai) không thay đổi theo thời gian. Tính dừng là một giả định quan trọng trong nhiều mô hình chuỗi thời gian, bao gồm cả ARIMA, vì nó đảm bảo rằng các đặc điểm của chuỗi thời gian không thay đổi theo thời gian, giúp mô hình dự báo chính xác hơn.

Để kiểm tra tính dừng của chuỗi thời gian, sẽ phải sử dụng hai phương pháp chính:

* Tính toán và vẽ đồ thị thống kê trượt (rolling statistics): sẽ phải tính toán trung bình trượt và độ lệch chuẩn trượt của chuỗi thời gian và vẽ chúng cùng với dữ liệu gốc. Nếu chuỗi thời gian là dừng, các giá trị trung bình và độ lệch chuẩn trượt sẽ không thay đổi nhiều theo thời gian.
* Kiểm định Dickey-Fuller: Đây là một kiểm định thống kê được sử dụng để kiểm tra tính dừng của chuỗi thời gian. Kiểm định này kiểm tra giả thuyết gốc rằng chuỗi thời gian có một đơn vị gốc (tức là không dừng). Nếu giá trị p của kiểm định nhỏ hơn một mức ý nghĩa nhất định (ví dụ: 0.05), có thể bác bỏ giả thuyết gốc và kết luận rằng chuỗi thời gian là dừng.

Dưới đây là đoạn mã:

A screen shot of a computer program

Description automatically generated

df.index = pd.to\_datetime(df.index, format="%Y-%M-%d")

df\_final = pd.Series(df['Close'])

type(df\_final)

check\_stationarity(df\_final)

Và kết quả:

A screenshot of a graph

Description automatically generated

* ts\_data.rolling(30, center=True, closed='both').mean() tính toán trung bình trượt với cửa sổ 30 ngày.
* ts\_data.rolling(5, center=True, closed='both').std() tính toán độ lệch chuẩn trượt với cửa sổ 5 ngày.
* plt.plot vẽ đồ thị của dữ liệu gốc, trung bình trượt và độ lệch chuẩn trượt.
* adfuller(ts\_data, regresults=False) thực hiện kiểm định Dickey-Fuller trên chuỗi thời gian.
* pd.Series(df\_test[0:4], index=['Test Statistic','p-value','# of lags','# of obs']) tạo một Series từ kết quả kiểm định Dickey-Fuller để dễ dàng hiển thị.

Kết quả của kiểm định Dickey-Fuller sẽ cho biết liệu chuỗi thời gian có tính dừng hay không. Nếu giá trị p nhỏ hơn 0.05, thì có thể kết luận rằng chuỗi thời gian là dừng. Nếu không, có thể cần phải thực hiện các bước bổ sung, chẳng hạn như lấy sai phân của chuỗi thời gian, để đạt được tính dừng trước khi áp dụng mô hình ARIMA.

## Xây dựng và huấn luyện mô hình ARIMA

Trong bước này, nhóm sẽ xây dựng và huấn luyện mô hình ARIMA để dự báo giá cổ phiếu PTC. Mô hình ARIMA bao gồm ba tham số chính: p (bậc của phần tự hồi quy), d (bậc của sai phân), và q (bậc của phần trung bình trượt). Xác định các tham số này dựa trên dữ liệu huấn luyện và sau đó sử dụng mô hình để dự báo giá cổ phiếu trong tương lai.

### Tạo tập dữ liệu huấn luyện và kiểm tra

Để xây dựng và đánh giá mô hình, cần chia dữ liệu thành hai tập: tập huấn luyện và tập kiểm tra. Tập huấn luyện sẽ được sử dụng để huấn luyện mô hình, trong khi tập kiểm tra sẽ được sử dụng để đánh giá hiệu suất của mô hình.

Dữ liệu sẽ chia theo tỷ lệ 70:30, trong đó 70% dữ liệu đầu tiên sẽ được sử dụng làm tập huấn luyện và 30% dữ liệu còn lại sẽ được sử dụng làm tập kiểm tra. Việc chia dữ liệu theo tỷ lệ này giúp đảm bảo rằng mô hình được huấn luyện trên một lượng dữ liệu đủ lớn và có thể được đánh giá trên một tập dữ liệu độc lập.

Dưới đây là đoạn mã:

# Chia dữ liệu thành tập huấn luyện và kiểm tra

n = int(len(df) \* 0.7)

m = int(len(df) \* 0.7)

df\_train = close.copy()[:n][['Close']]

df\_test = close.copy()[m:][['Close']]

# Hiển thị kích thước của tập huấn luyện và kiểm tra

print('Training data shape: ', df\_train.shape)

print('Test data shape: ', df\_test.shape)

# Vẽ đồ thị ACF và PACF để xác định các tham số p và q

plot\_acf(df\_train['Close'])

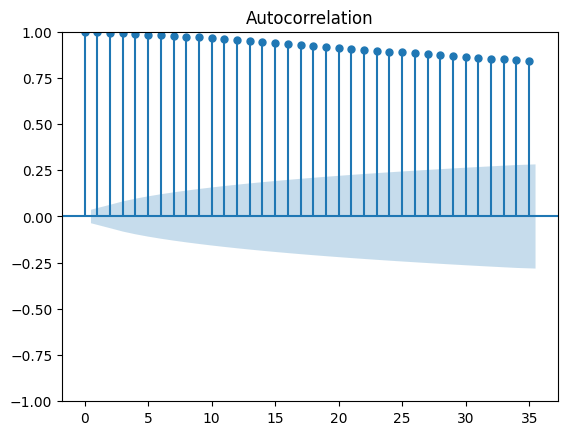
plot\_pacf(df\_train['Close'])

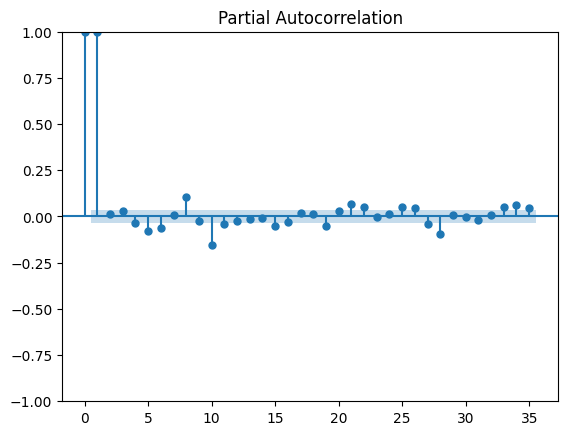
plt.show()

Và đây là kết quả:

A black screen with white text

Description automatically generated





* n = int(len(df) \* 0.7) tính toán số lượng mẫu trong tập huấn luyện, chiếm 70% tổng số mẫu.
* m = int(len(df) \* 0.7) tính toán số lượng mẫu bắt đầu của tập kiểm tra, chiếm 30% tổng số mẫu.
* df\_train = close.copy()[:n][['Close']] tạo tập huấn luyện từ 70% dữ liệu đầu tiên.
* df\_test = close.copy()[m:][['Close']] tạo tập kiểm tra từ 30% dữ liệu còn lại.
* print('Training data shape: ', df\_train.shape) và print('Test data shape: ', df\_test.shape) hiển thị kích thước của tập huấn luyện và kiểm tra để kiểm tra lại việc chia dữ liệu.
* plot\_acf(df\_train['Close']) và plot\_pacf(df\_train['Close']) vẽ đồ thị hàm tự tương quan (ACF) và hàm tự tương quan từng phần (PACF) của tập huấn luyện để xác định các tham số p và q cho mô hình ARIMA.

Việc vẽ đồ thị ACF và PACF giúp xác định các tham số p và q cho mô hình ARIMA. Đồ thị ACF cho thấy mức độ tương quan giữa các giá trị trong chuỗi thời gian với các độ trễ khác nhau, trong khi đồ thị PACF cho thấy mức độ tương quan từng phần giữa các giá trị trong chuỗi thời gian với các độ trễ khác nhau sau khi loại bỏ ảnh hưởng của các độ trễ trung gian. Dựa trên các đồ thị này, có thể xác định các giá trị thích hợp cho p và q để xây dựng mô hình ARIMA.

### Huấn luyện mô hình ARIMA

Sau khi đã tạo tập dữ liệu huấn luyện và kiểm tra, bước tiếp theo là huấn luyện mô hình ARIMA. Mô hình ARIMA được xác định bởi ba tham số chính: p (bậc của phần tự hồi quy), d (bậc của sai phân), và q (bậc của phần trung bình trượt). Trong bước này, nhóm sẽ khởi tạo mô hình ARIMA với các tham số đã xác định và huấn luyện mô hình trên tập dữ liệu huấn luyện.

Để huấn luyện mô hình ARIMA, cần sử dụng thư viện statsmodels, một thư viện phổ biến trong Python cho phân tích chuỗi thời gian và thống kê. Khởi tạo mô hình ARIMA với các tham số p, d, và q, sau đó sử dụng phương pháp fit để huấn luyện mô hình trên tập dữ liệu huấn luyện.

Dưới đây là đoạn mã:

# Định nghĩa tham số cho mô hình ARIMA(p,d,q)

order = (1,1,1) # p=1: AR(1), d=1: sai phân bậc 1, q=1: MA(1)

# Khởi tạo và huấn luyện mô hình

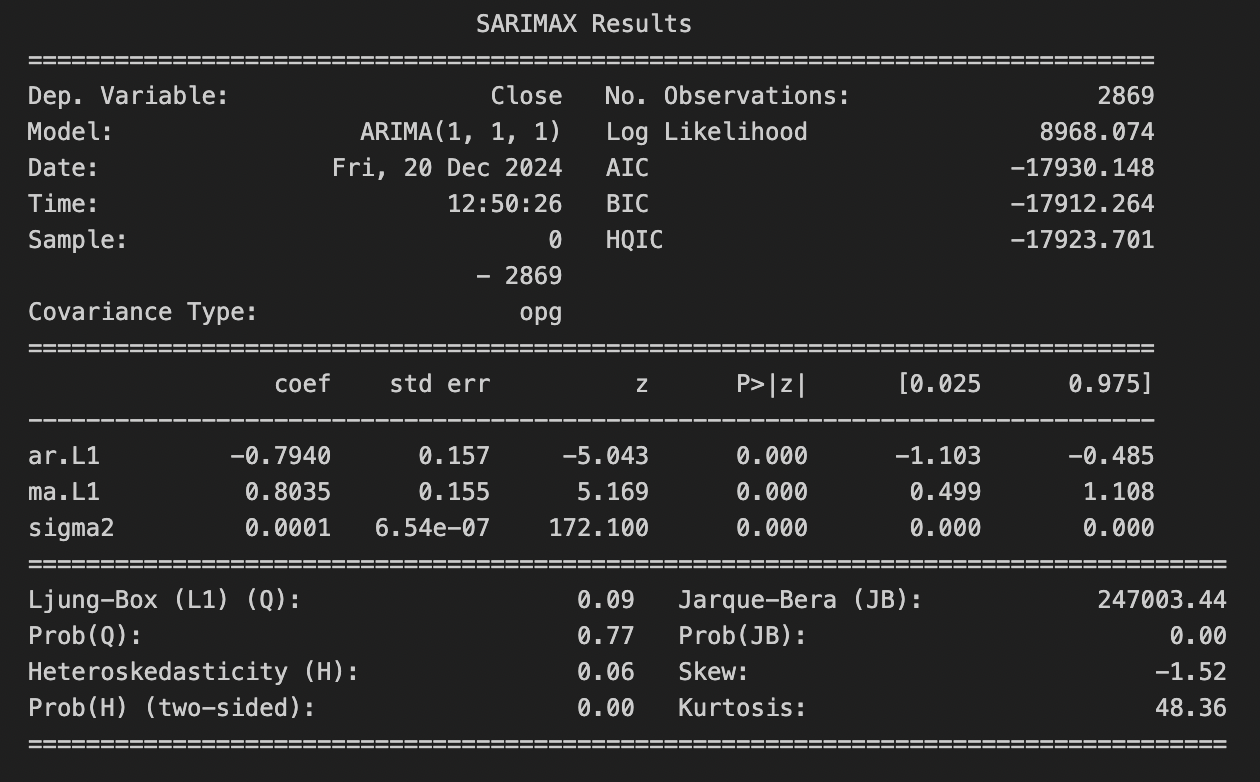
model = ARIMA(df\_train, order=order)

results = model.fit()

# In kết quả mô hình

print(results.summary())

Và kết quả là:



* order = (1,1,1) xác định các tham số p, d, và q cho mô hình ARIMA. Trong trường hợp này, chọn p=1, d=1, và q=1, nghĩa là mô hình ARIMA(1,1,1).
* model = ARIMA(df\_train, order=order) khởi tạo mô hình ARIMA với các tham số đã xác định và tập dữ liệu huấn luyện.
* results = model.fit() huấn luyện mô hình ARIMA trên tập dữ liệu huấn luyện.
* print(results.summary()) in ra kết quả của mô hình sau khi huấn luyện, bao gồm các tham số ước lượng và các chỉ số thống kê.

Kết quả của mô hình ARIMA sau khi huấn luyện sẽ bao gồm các tham số ước lượng cho phần tự hồi quy (AR), sai phân (I), và trung bình trượt (MA), cùng với các chỉ số thống kê như AIC (Akaike Information Criterion) và BIC (Bayesian Information Criterion). Các chỉ số này giúp đánh giá chất lượng của mô hình và so sánh với các mô hình khác.

Việc huấn luyện mô hình ARIMA là một trong những quá trình dự báo chuỗi thời gian, vì nó giúp ta nắm bắt được các đặc điểm tự tương quan trong dữ liệu và xây dựng một mô hình dự báo chính xác. Sau khi mô hình đã được huấn luyện, có thể sử dụng nó để dự báo giá cổ phiếu trong tương lai, sẽ được trình bày trong các bước tiếp theo.

### Dự báo

Sau khi đã huấn luyện mô hình ARIMA, bước tiếp theo là sử dụng mô hình này để dự báo giá cổ phiếu trong tương lai. Dự báo là một phần quan trọng của phân tích chuỗi thời gian, giúp dự đoán các giá trị tương lai dựa trên mô hình đã được huấn luyện từ dữ liệu lịch sử.

Trong bước này, sẽ phải sử dụng mô hình ARIMA đã được huấn luyện để dự báo giá cổ phiếu PTC cho một khoảng thời gian nhất định trong tương lai. Sau đó thiết lập khoảng thời gian dự báo và sử dụng phương pháp forecast của mô hình ARIMA để thực hiện dự báo.

Dưới đây là đoạn mã:

# Specify the number of steps to forcast ahead

HORIZON=6

print('Forescating horizon:', HORIZON, 'days')

Và kết quả như sau:



* HORIZON = 6 xác định khoảng thời gian dự báo là 6 ngày. Điều này có nghĩa là dự báo giá cổ phiếu PTC cho 6 ngày tiếp theo.
* print('Forecasting horizon:', HORIZON, 'days') in ra khoảng thời gian dự báo để xác nhận.

Kết quả của dự báo sẽ là một mảng các giá trị dự báo cho 6 ngày tiếp theo. Các giá trị này đại diện cho dự đoán của mô hình ARIMA về giá cổ phiếu PTC trong tương lai dựa trên dữ liệu lịch sử và các tham số đã được huấn luyện.

Dự báo là một phần trong quá trình phân tích chuỗi thời gian, vì nó hỗ trợ đưa ra các quyết định dựa trên các dự đoán về tương lai. Trong trường hợp này, dự báo giá cổ phiếu PTC có thể giúp các nhà đầu tư và các bên liên quan đưa ra các quyết định đầu tư và quản lý rủi ro dựa trên các dự đoán về xu hướng giá cổ phiếu trong tương lai.

Sau khi thực hiện dự báo, tiếp tục với bước đánh giá mô hình để kiểm tra độ chính xác của các dự báo và đảm bảo rằng mô hình ARIMA đã được xây dựng và huấn luyện một cách hiệu quả.

## Đánh giá mô hình

Sau khi đã thực hiện dự báo, bước tiếp theo là đánh giá hiệu suất của mô hình ARIMA. Đánh giá mô hình là một bước quan trọng để kiểm tra độ chính xác của các dự báo và đảm bảo rằng mô hình đã được xây dựng và huấn luyện một cách hiệu quả. Trong phần này, cần chuẩn bị dữ liệu kiểm tra để so sánh với các giá trị dự báo và sau đó đánh giá mô hình dựa trên các chỉ số đánh giá.

### Tạo dữ liệu kiểm tra dịch chuyển

Để đánh giá mô hình ARIMA, phải chuẩn bị dữ liệu kiểm tra sao cho có thể so sánh các giá trị dự báo với giá trị thực tế. Một cách để làm điều này là tạo ra các cột mới trong tập dữ liệu kiểm tra, mỗi cột chứa giá trị giá đóng cửa của các ngày tiếp theo. Điều này giúp dễ dàng so sánh các giá trị dự báo với giá trị thực tế trong các bước đánh giá mô hình tiếp theo.

Dưới đây là đoạn mã:

df\_test\_shifted = df\_test.copy()

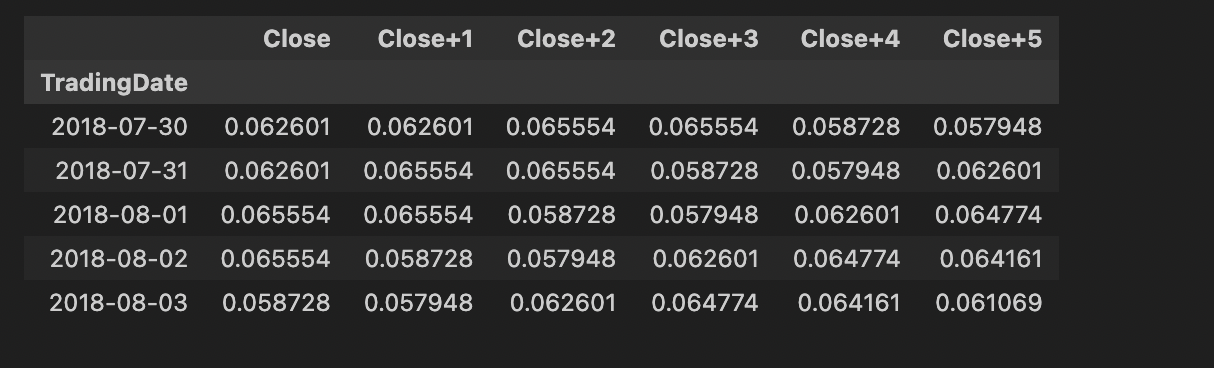
for t in range(1, HORIZON):

df\_test\_shifted['Close+'+str(t)] = df\_test\_shifted['Close'].shift(-t)

df\_test\_shifted = df\_test\_shifted.dropna(how='any')

df\_test\_shifted.head(5)

Kết quả là:



* df\_test\_shifted = df\_test.copy() tạo một bản sao của tập dữ liệu kiểm tra để không làm thay đổi dữ liệu gốc.
* Vòng lặp for t in range(1, HORIZON): thêm các cột mới vào df\_test\_shifted, mỗi cột chứa giá trị giá đóng cửa của t ngày sau đó. Ví dụ, cột Close+1 chứa giá đóng cửa của ngày tiếp theo, cột Close+2 chứa giá đóng cửa của hai ngày sau đó, v.v.
* df\_test\_shifted['Close+'+str(t)] = df\_test\_shifted['Close'].shift(-t) dịch chuyển giá trị của cột Close lên t hàng, tạo ra các cột mới chứa giá trị giá đóng cửa của các ngày tiếp theo.
* df\_test\_shifted = df\_test\_shifted.dropna(how='any') loại bỏ các hàng có giá trị NaN (không xác định) do việc dịch chuyển dữ liệu.
* df\_test\_shifted.head(5) hiển thị 5 hàng đầu tiên của df\_test\_shifted để kiểm tra kết quả.

Mục đích của đoạn mã này là chuẩn bị dữ liệu kiểm tra để so sánh với các giá trị dự báo từ mô hình ARIMA. Bằng cách tạo ra các cột chứa giá trị giá đóng cửa của các ngày tiếp theo, có thể so sánh các giá trị dự báo với giá trị thực tế trong các bước đánh giá mô hình tiếp theo.

Việc tạo dữ liệu kiểm tra dịch chuyển cần trong quá trình đánh giá mô hình. Nó giúp chuẩn bị dữ liệu kiểm tra sao cho có thể so sánh các giá trị dự báo với giá trị thực tế một cách dễ dàng và chính xác. Trong các bước tiếp theo, sử dụng dữ liệu kiểm tra đã được chuẩn bị này để đánh giá hiệu suất của mô hình ARIMA và kiểm tra độ chính xác của các dự báo.

### Đánh giá mô hình

Sau khi đã chuẩn bị dữ liệu kiểm tra dịch chuyển, bước tiếp theo là đánh giá hiệu suất của mô hình ARIMA. Đánh giá mô hình là một bước quan trọng để kiểm tra độ chính xác của các dự báo và đảm bảo rằng mô hình đã được xây dựng và huấn luyện một cách hiệu quả. Sử dụng các chỉ số đánh giá như RMSE (Root Mean Squared Error) và R² (R-squared) để đo lường độ chính xác của các dự báo so với giá trị thực tế.

Dưới đây là đoạn mã:

training\_window = 30 # dedicate 30 days for training

train\_ts = df\_train['Close']

test\_ts = df\_test\_shifted

history = [x for x in train\_ts]

history = history[(-training\_window):]

predictions = list()

for i in range(test\_ts.shape[0]):

model = ARIMA(history, order=order)

model\_fit = model.fit()

output = model\_fit.forecast(steps=HORIZON)

predictions.append(output)

obs = list(test\_ts.iloc[i])

history.append(obs[0])

history.pop(0)

# evaluate forecasts

rmse = np.sqrt(mean\_squared\_error(test\_ts, predictions))

r2\_test = r2\_score(test\_ts, predictions)

print('Test RMSE: %.3f' % rmse)

print('Test r2\_score: %.3f' % r2\_test)

Và kết quả:



* training\_window = 30 xác định cửa sổ huấn luyện là 30 ngày. Điều này có nghĩa là ta sử dụng 30 ngày gần nhất trong tập huấn luyện để huấn luyện mô hình trước khi thực hiện dự báo.
* train\_ts = df\_train['Close'] và test\_ts = df\_test\_shifted tạo các chuỗi thời gian từ tập huấn luyện và tập kiểm tra.
* history = [x for x in train\_ts] sao chép dữ liệu huấn luyện vào danh sách history.
* history = history[(-training\_window):] chỉ giữ lại 30 ngày gần nhất trong history để sử dụng cho huấn luyện mô hình.
* predictions = list() khởi tạo danh sách để lưu trữ các giá trị dự báo.

Vòng lặp for i in range(test\_ts.shape[0]): thực hiện các bước sau cho mỗi hàng trong tập kiểm tra:

* model = ARIMA(history, order=order) khởi tạo mô hình ARIMA với các tham số đã xác định và dữ liệu lịch sử.
* model\_fit = model.fit() huấn luyện mô hình ARIMA trên dữ liệu lịch sử.
* output = model\_fit.forecast(steps=HORIZON) thực hiện dự báo cho khoảng thời gian đã xác định (HORIZON).
* predictions.append(output) lưu trữ các giá trị dự báo vào danh sách predictions.
* obs = list(test\_ts.iloc[i]) lấy giá trị thực tế từ tập kiểm tra.
* history.append(obs[0]) thêm giá trị thực tế vào dữ liệu lịch sử.
* history.pop(0) loại bỏ giá trị cũ nhất trong dữ liệu lịch sử để duy trì cửa sổ huấn luyện cố định.

Sau khi thực hiện dự báo cho tất cả các hàng trong tập kiểm tra, nên sử dụng các chỉ số đánh giá để đo lường độ chính xác của các dự báo:

* rmse = np.sqrt(mean\_squared\_error(test\_ts, predictions)) tính toán RMSE, một chỉ số đo lường độ lệch trung bình bình phương giữa các giá trị dự báo và giá trị thực tế. RMSE càng nhỏ, mô hình càng chính xác.
* r2\_test = r2\_score(test\_ts, predictions) tính toán R², một chỉ số đo lường mức độ giải thích của mô hình đối với biến phụ thuộc. R² càng gần 1, mô hình càng tốt.
* print('Test RMSE: %.3f' % rmse) và print('Test r2\_score: %.3f' % r2\_test) in ra các chỉ số đánh giá để kiểm tra hiệu suất của mô hình.

Chỉ số RMSE = 0.033 cho thấy mô hình ARIMA đã dự báo giá cổ phiếu PTC với độ chính xác cao. Mô hình đã được xây dựng và huấn luyện một cách hiệu quả, và có thể được sử dụng để dự báo giá cổ phiếu trong tương lai với độ tin cậy cao. Tuy nhiên, để đảm bảo tính ổn định và độ chính xác của mô hình trong các điều kiện khác nhau, có thể tiếp tục thử nghiệm với các tham số khác nhau và kết hợp với các kỹ thuật dự báo khác.

## Biểu đồ dự báo cuối cùng

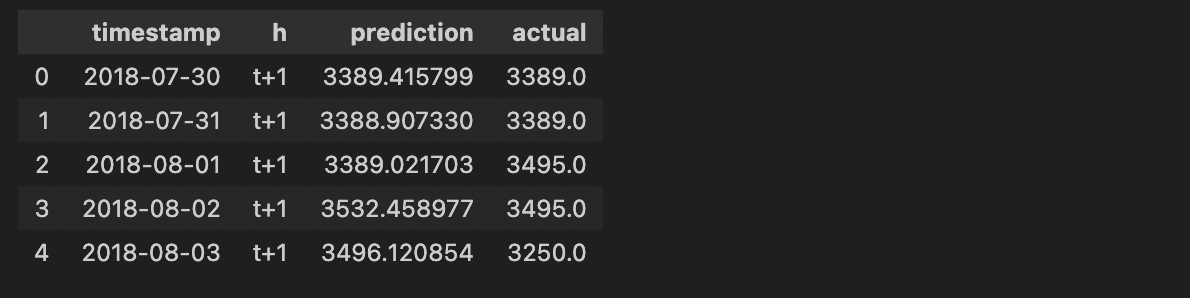
Sau khi đã đánh giá mô hình bằng các chỉ số như RMSE và R², bước tiếp theo là trực quan hóa kết quả dự báo bằng cách vẽ biểu đồ so sánh giữa giá trị thực tế và giá trị dự báo. Việc trực quan hóa này giúp nhận thấy mức độ chính xác của mô hình và kiểm tra xem mô hình có nắm bắt được các xu hướng và mẫu trong dữ liệu hay không.

### Chuẩn bị dữ liệu cho biểu đồ để dự báo

Trước khi vẽ biểu đồ, cần chuẩn bị dữ liệu dự báo và dữ liệu thực tế trong một định dạng phù hợp. Nhóm sẽ tạo một DataFrame chứa các giá trị dự báo và giá trị thực tế, sau đó chuyển đổi DataFrame này từ dạng rộng sang dạng dài để dễ dàng vẽ biểu đồ

Dưới đây là đoạn mã:A screen shot of a computer code

Description automatically generated

Và kết quả:

* eval\_df = pd.DataFrame(predictions, columns=['t+'+str(t) for t in range(1, HORIZON+1)]) tạo một DataFrame từ danh sách các giá trị dự báo, với mỗi cột tương ứng với một bước dự báo.
* eval\_df['timestamp'] = df\_test.index[0:len(df\_test.index)-HORIZON+1] thêm cột timestamp vào DataFrame để lưu trữ các mốc thời gian tương ứng với các giá trị dự báo.
* eval\_df = pd.melt(eval\_df, id\_vars='timestamp', value\_name='prediction', var\_name='h') chuyển đổi DataFrame từ dạng rộng sang dạng dài để dễ dàng vẽ biểu đồ.
* eval\_df['actual'] = np.array(np.transpose(test\_ts)).ravel() thêm cột actual vào DataFrame để lưu trữ các giá trị thực tế.
* eval\_df[['prediction', 'actual']] = scaler.inverse\_transform(eval\_df[['prediction', 'actual']]) đảo ngược quá trình chuẩn hóa để đưa các giá trị dự báo và thực tế về thang đo ban đầu.
* eval\_df.head() hiển thị 5 hàng đầu tiên của DataFrame để kiểm tra kết quả.

Ý nghĩa kết quả:

* timestamp: Cột này chứa các mốc thời gian tương ứng với các giá trị dự báo và giá trị thực tế. Mỗi hàng trong DataFrame đại diện cho một ngày cụ thể.
* h: Cột này chỉ định bước dự báo. Trong ví dụ này, tất cả các giá trị đều là t+1, có nghĩa là dự báo cho ngày tiếp theo. Nếu HORIZON lớn hơn 1, cột này sẽ chứa các giá trị như t+2, t+3, v.v., tương ứng với các bước dự báo tiếp theo.
* prediction: Cột này chứa các giá trị dự báo từ mô hình ARIMA. Đây là các giá trị mà mô hình dự đoán cho giá cổ phiếu PTC vào các ngày tương ứng.
* actual: Cột này chứa các giá trị thực tế của giá cổ phiếu PTC vào các ngày tương ứng. Đây là các giá trị thực tế để so sánh với các giá trị dự báo.

Phân tích kết quả:

* Ngày 2018-07-30: Giá dự báo là 3389.415799, rất gần với giá thực tế là 3389.0. Điều này cho thấy mô hình dự báo khá chính xác cho ngày này.
* Ngày 2018-07-31: Giá dự báo là 3388.907330, cũng rất gần với giá thực tế là 3389.0. Mô hình tiếp tục dự báo chính xác.
* Ngày 2018-08-01: Giá dự báo là 3389.021703, nhưng giá thực tế là 3495.0. Có một sự chênh lệch đáng kể giữa giá trị dự báo và giá trị thực tế, cho thấy mô hình không dự báo chính xác cho ngày này.
* Ngày 2018-08-02: Giá dự báo là 3532.458977, gần với giá thực tế là 3495.0. Mô hình dự báo khá chính xác cho ngày này.
* Ngày 2018-08-03: Giá dự báo là 3496.120854, nhưng giá thực tế là 3250.0. Có một sự chênh lệch đáng kể giữa giá trị dự báo và giá trị thực tế, cho thấy mô hình không dự báo chính xác cho ngày này.

Kết quả của phần chuẩn bị dữ liệu cho biểu đồ dự đoán cho thấy rằng mô hình ARIMA có thể dự báo khá chính xác trong một số ngày, nhưng cũng có những ngày mà dự báo không khớp với giá trị thực tế. Điều này có thể do nhiều yếu tố, bao gồm sự biến động bất thường của thị trường hoặc các sự kiện không lường trước được.

### 5.6.2.Vẽ biểu đồ dự báo

Sau khi đã chuẩn bị dữ liệu, vẽ biểu đồ để so sánh giữa giá trị thực tế và giá trị dự báo. Biểu đồ này sẽ giúp cho trực quan hóa hiệu suất của mô hình ARIMA và kiểm tra xem các dự báo có khớp với giá trị thực tế hay không

Dưới đây là đoạn mã:

if(HORIZON == 1):

## Vẽ biểu đồ dự đoán một bước

eval\_df.plot(x='timestamp', y=['actual', 'prediction'], style=['r', 'b'], figsize=(15, 8))

else:

## Vẽ biểu đồ dự đoán nhiều bước

plot\_df = eval\_df[(eval\_df.h=='t+1')][['timestamp', 'actual']]

# Chuyển đổi timestamp sang datetime

plot\_df['timestamp'] = pd.to\_datetime(plot\_df['timestamp'])

for t in range(1, HORIZON+1):

plot\_df['t+'+str(t)] = eval\_df[(eval\_df.h=='t+'+str(t))]['prediction'].values

fig = plt.figure(figsize=(15, 8))

# Vẽ đường giá thực màu đỏ

plt.plot(plot\_df['timestamp'], plot\_df['actual'], color='red', linewidth=2.0, label='Giá thực')

# Vẽ các đường dự đoán màu xanh

for t in range(1, HORIZON+1):

x = plot\_df['timestamp'][(t-1):]

y = plot\_df['t+'+str(t)][0:len(x)]

plt.plot(x, y, color='blue', linewidth=1.5, alpha=0.6, label=f'Dự đoán t+{t}')

# Định dạng trục thời gian

plt.gcf().autofmt\_xdate()

from matplotlib.dates import DateFormatter

plt.gca().xaxis.set\_major\_formatter(DateFormatter('%Y-%m-%d'))

plt.legend(loc='best')

plt.grid(True, linestyle='--', alpha=0.7)

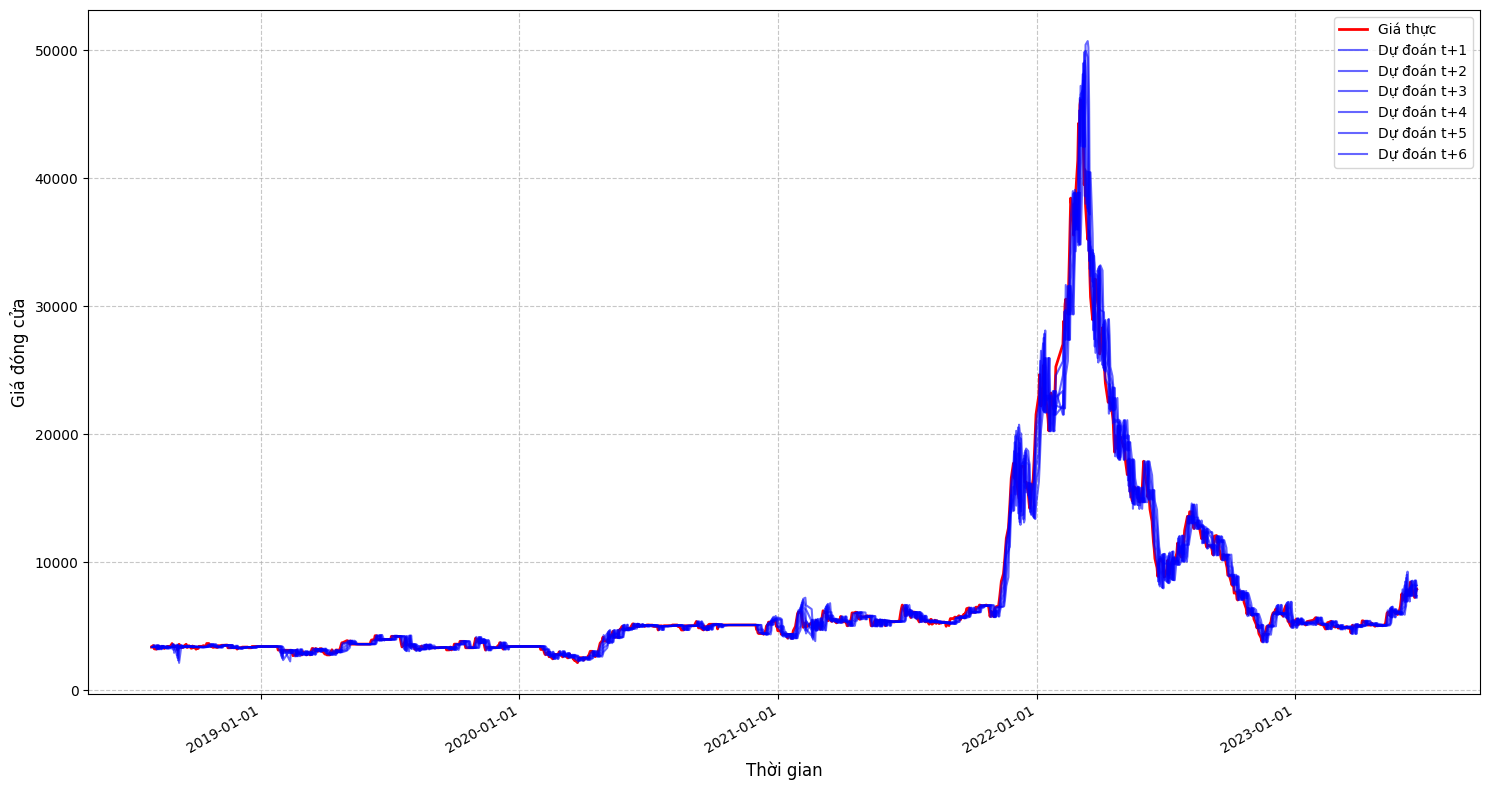
plt.xlabel('Thời gian', fontsize=12)

plt.ylabel('Giá đóng cửa', fontsize=12)

plt.tight\_layout()

plt.show()

Và kết quả:

Nếu HORIZON == 1, vẽ biểu đồ dự đoán một bước:

* eval\_df.plot(x='timestamp', y=['actual', 'prediction'], style=['r', 'b'], figsize=(15, 8)) vẽ biểu đồ so sánh giữa giá trị thực tế và giá trị dự báo.

Nếu HORIZON > 1, lúc này vẽ biểu đồ dự đoán nhiều bước:

* plot\_df = eval\_df[(eval\_df.h=='t+1')][['timestamp', 'actual']] tạo DataFrame chứa giá trị thực tế và giá trị dự báo cho bước đầu tiên.
* plot\_df['timestamp'] = pd.to\_datetime(plot\_df['timestamp']) chuyển đổi cột timestamp sang định dạng datetime.
* Vòng lặp for t in range(1, HORIZON+1): thêm các cột chứa giá trị dự báo cho các bước tiếp theo vào DataFrame plot\_df.
* plt.plot(plot\_df['timestamp'], plot\_df['actual'], color='red', linewidth=2.0, label='Giá thực') vẽ đường giá thực tế màu đỏ.
* Vòng lặp for t in range(1, HORIZON+1): vẽ các đường dự đoán màu xanh cho các bước dự báo.
* plt.gcf().autofmt\_xdate() định dạng trục thời gian.
* plt.gca().xaxis.set\_major\_formatter(DateFormatter('%Y-%m-%d')) định dạng nhãn trục thời gian.
* plt.legend(loc='best') thêm chú thích vào biểu đồ.
* plt.grid(True, linestyle='--', alpha=0.7) thêm lưới vào biểu đồ.
* plt.xlabel('Thời gian', fontsize=12) và plt.ylabel('Giá đóng cửa', fontsize=12) thêm nhãn cho các trục.
* plt.tight\_layout() điều chỉnh bố cục biểu đồ.
* plt.show() hiển thị biểu đồ.

Việc trực quan hóa kết quả dự báo bằng biểu đồ để nhận thấy các xu hướng và mẫu trong dữ liệu, cũng như kiểm tra xem mô hình có nắm bắt được các đặc điểm của chuỗi thời gian hay không. Điều này sẽ làm hiểu rõ hơn về hiệu suất của mô hình và có thể thực hiện các điều chỉnh cần thiết để cải thiện độ chính xác của dự báo.

Biểu đồ trên minh họa kết quả dự báo giá cổ phiếu bằng mô hình ARIMA, cụ thể là so sánh giữa giá thực tế (đường màu đỏ) và giá dự báo (các đường màu xanh ứng với các khoảng thời gian dự báo t+1, t+2,..., t+6).

* Đường dự báo (màu xanh) bám khá sát vào đường giá thực tế (màu đỏ), đặc biệt là ở các giai đoạn ổn định.
* Điều này cho thấy mô hình ARIMA đã hoạt động hiệu quả trong việc nắm bắt xu hướng giá cổ phiếu dựa trên dữ liệu quá khứ.

## Kết luận

Hiệu quả dự báo tại các điểm thay đổi lớn:

* Tại những giai đoạn giá cổ phiếu có biến động mạnh (ví dụ: đỉnh cao nhất vào khoảng năm 2022), giá dự báo vẫn cố gắng theo sát xu hướng nhưng có độ lệch nhất định.
* Điều này là bình thường do các mô hình ARIMA dựa trên dữ liệu quá khứ và khó dự đoán các biến động đột ngột hoặc ngoại lệ (outliers).

Mô hình ARIMA phù hợp cho việc dự báo trong ngắn hạn, nhưng khả năng xử lý biến động lớn trong dài hạn còn hạn chế.

Dự báo cho các khoảng thời gian t+1, t+2,...

* Các đường dự báo từ t+1 đến t+6 có xu hướng mở rộng và "phân kỳ" nhẹ so với giá thực tế khi thời gian tăng dần.
* Điều này phản ánh đặc điểm chung của mô hình ARIMA, khi mức độ không chắc chắn của dự báo tăng theo thời gian.

Khi áp dụng ARIMA, các dự báo ngắn hạn sẽ chính xác hơn so với dài hạn. Vì vậy, cần cẩn thận khi sử dụng mô hình này để dự đoán trong các giai đoạn xa.

Mô hình ARIMA là công cụ tốt để dự báo giá cổ phiếu trong ngắn hạn, với độ chính xác cao ở các giai đoạn ổn định và ít biến động. Tuy nhiên, để cải thiện kết quả tại các giai đoạn có biến động lớn hoặc dự báo dài hạn, cần xem xét kết hợp thêm các mô hình phi tuyến tính hoặc các yếu tố kinh tế bổ sung.

# Áp dụng mô hình SVR

Mô hình SVR (Support Vector Regression) được sử dụng để dự báo giá cổ phiếu PTC từ dữ liệu lịch sử. SVR là một kỹ thuật học máy mạnh mẽ, thường được sử dụng trong các bài toán hồi quy và dự báo chuỗi thời gian. Mô hình SVR có khả năng xử lý các dữ liệu phi tuyến tính và có thể tạo ra các dự báo chính xác.

## Import các thư viện cần thiết

**import os**

**import pandas as pd**

**import numpy as np**

**import matplotlib.pyplot as plt**

**import math**

**import datetime as dt**

**from sklearn.svm import SVR**

**from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error, r2\_score**

**from sklearn.preprocessing import MinMaxScaler, StandardScaler**

**from sklearn.model\_selection import train\_test\_split**

**import seaborn as sns**

**sns.set\_style('whitegrid')**

**plt.style.use("fivethirtyeight")**

**import warnings**

**warnings.filterwarnings("ignore")**

* Pandas được sử dụng để tải dữ liệu từ tệp CSV, xử lý dữ liệu và tạo các tập dữ liệu huấn luyện và kiểm tra.
* NumPy được sử dụng để thực hiện các phép toán trên dữ liệu và tạo các tập dữ liệu huấn luyện và kiểm tra.
* Matplotlib được sử dụng để trực quan hóa dữ liệu và kết quả dự báo, giúp nhận thấy các xu hướng và mẫu trong dữ liệu.

Trong dự án này, Nhóm sử dụng các thành phần sau của Scikit-learn:

* SVR: Lớp SVR (Support Vector Regression) được sử dụng để xây dựng mô hình hồi quy vector hỗ trợ.
* mean\_squared\_error, mean\_absolute\_error, r2\_score: Các hàm này được sử dụng để đánh giá hiệu suất của mô hình bằng cách tính toán các chỉ số đánh giá như RMSE, MAE và R².
* MinMaxScaler, StandardScaler: Các lớp này được sử dụng để chuẩn hóa dữ liệu, giúp mô hình học máy hoạt động hiệu quả hơn.
* train\_test\_split: Hàm này được sử dụng để chia dữ liệu thành tập huấn luyện và tập kiểm tra.
* Seaborn được sử dụng để thiết lập phong cách trực quan hóa dữ liệu, giúp các biểu đồ và đồ thị trở nên dễ nhìn và chuyên nghiệp hơn.
* Warnings thư viện này để tắt các cảnh báo không cần thiết, giúp mã nguồn trở nên sạch sẽ và dễ đọc hơn.

## Tải dữ liệu

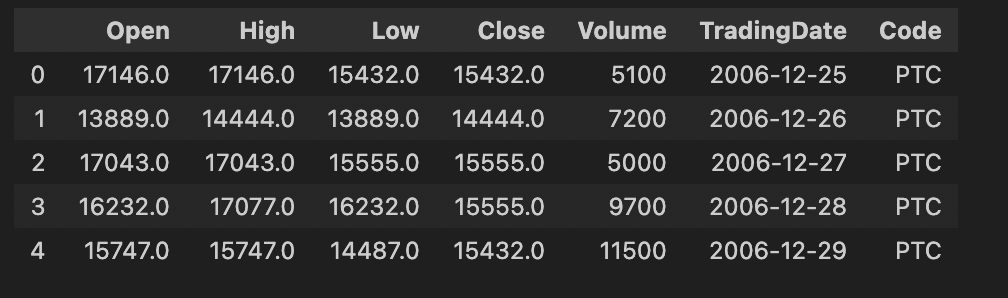
Trong bước này, cần tải dữ liệu giá cổ phiếu PTC từ tệp CSV. Dữ liệu này chứa thông tin về giá cổ phiếu theo thời gian, bao gồm các cột như giá mở cửa, giá đóng cửa, giá cao nhất, giá thấp nhất và khối lượng giao dịch. Để đơn giản hóa, sẽ chỉ sử dụng cột giá đóng cửa (Close) cho việc phân tích và dự báo.

Dưới đây là đoạn mã:

# Load data

df = pd.read\_csv('/Users/dophi/Desktop/dudoangiacophieu-main/Dataset/construction/PTC.csv', sep='\t', index\_col=0)

df.head()

Và kết quả:

* pd.read\_csv: Đây là hàm của thư viện Pandas để đọc dữ liệu từ tệp CSV.
* sep='\t': Tham số này chỉ định rằng các giá trị trong tệp CSV được phân tách bằng tab. Điều này là cần thiết nếu tệp CSV của bạn sử dụng tab làm ký tự phân tách.
* index\_col=0: Tham số này chỉ định rằng cột đầu tiên trong tệp CSV sẽ được sử dụng làm chỉ số (index) cho DataFrame. Điều này giúp truy cập và thao tác với dữ liệu theo thời gian.
* df.head(): Hàm này hiển thị 5 hàng đầu tiên của DataFrame để kiểm tra dữ liệu đã được tải đúng cách. Kiểm tra dữ liệu bằng cách hiển thị một phần nhỏ của nó đảm bảo rằng dữ liệu đã được tải đúng cách và sẵn sàng cho các bước xử lý tiếp theo.

Sau khi thực hiện đoạn mã trên, sẽ có được một DataFrame chứa giá đóng cửa của cổ phiếu PTC theo thời gian. Đây sẽ là dữ liệu đầu vào cho các bước phân tích và dự báo tiếp theo.

## Tạo tập dữ liệu huấn luyện và kiểm tra

Bước đây nhóm em đã nêu ở mục 2.2 là phần tiền xử lý dữ liệu nhưng nhóm sẽ làm rõ thêm chút nữa.

### Chia dữ liệu thành tập huấn và kiểm tra

Để xây dựng và đánh giá mô hình, cần chia dữ liệu thành hai tập: tập huấn luyện và tập kiểm tra. Tập huấn luyện sẽ được sử dụng để huấn luyện mô hình, trong khi tập kiểm tra sẽ được sử dụng để đánh giá hiệu suất của mô hình.

Dưới đây là đoạn mã:

# Chia dữ liệu thành tập huấn luyện và kiểm tra

training\_size = int(len(df) \* 0.8)

test\_size = len(df) - training\_size

train\_data, test\_data = df[0:training\_size], df[training\_size:len(df)]

print("train\_data: ", train\_data.shape)

print("test\_data: ", test\_data.shape)

Kết quả:



* training\_size = int(len(df) \* 0.8): Tính toán số lượng mẫu trong tập huấn luyện, chiếm 80% tổng số mẫu.
* test\_size = len(df) - training\_size: Tính toán số lượng mẫu trong tập kiểm tra, chiếm 20% tổng số mẫu.
* train\_data, test\_data = df[0:training\_size], df[training\_size:len(df)]: Chia dữ liệu thành tập huấn luyện và tập kiểm tra.
* print("train\_data: ", train\_data.shape) và print("test\_data: ", test\_data.shape): Hiển thị kích thước của tập huấn luyện và kiểm tra để kiểm tra lại việc chia dữ liệu.

Dữ liệu chia theo tỷ lệ 80:20, trong đó 80% dữ liệu đầu tiên sẽ được sử dụng làm tập huấn luyện và 20% dữ liệu còn lại sẽ được sử dụng làm tập kiểm tra. Việc chia dữ liệu theo tỷ lệ này giúp đảm bảo rằng mô hình được huấn luyện trên một lượng dữ liệu đủ lớn và có thể được đánh giá trên một tập dữ liệu độc lập.

### Chuẩn hoá dữ liệu

Chuẩn hóa dữ liệu là một bước quan trọng trong quá trình xây dựng mô hình học máy. Việc chuẩn hóa giúp đưa các giá trị dữ liệu về cùng một thang đo, giúp mô hình học máy hoạt động hiệu quả hơn và tránh các vấn đề liên quan đến sự khác biệt về thang đo giữa các đặc trưng.

Trong dự án này, sử dụng StandardScaler từ thư viện Scikit-learn để chuẩn hóa dữ liệu. StandardScaler chuẩn hóa dữ liệu bằng cách trừ đi giá trị trung bình và chia cho độ lệch chuẩn của từng đặc trưng.

Dưới đây là đoạn mã:

# scaler = MinMaxScaler(feature\_range=(0,1))

scaler = StandardScaler()

scaled\_data = scaler.fit\_transform(dataset)

scaled\_data

Và kết quả:



* scaler = StandardScaler(): Khởi tạo một đối tượng StandardScaler.
* scaled\_data = scaler.fit\_transform(dataset): Chuẩn hóa dữ liệu giá đóng cửa bằng cách trừ đi giá trị trung bình và chia cho độ lệch chuẩn.

Kết quả này cho thấy các giá trị giá đóng cửa của cổ phiếu PTC đã được chuẩn hóa. Mỗi giá trị trong mảng này đại diện cho giá trị giá đóng cửa của một ngày, đã được chuẩn hóa bằng cách trừ đi giá trị trung bình và chia cho độ lệch chuẩn của toàn bộ dữ liệu.

* Các giá trị dương (ví dụ: 1.66173215, 1.49327583, 1.68270394) cho thấy giá đóng cửa của cổ phiếu vào những ngày này cao hơn giá trị trung bình của toàn bộ dữ liệu.
* Các giá trị âm (nếu có) sẽ cho thấy giá đóng cửa của cổ phiếu vào những ngày này thấp hơn giá trị trung bình của toàn bộ dữ liệu.
* Các giá trị gần bằng 0 cho thấy giá đóng cửa của cổ phiếu vào những ngày này gần với giá trị trung bình của toàn bộ dữ liệu.

### Tạo tập dữ liệu huấn luyện và kiểm tra

Sau khi chuẩn hóa dữ liệu, tạo các tập dữ liệu huấn luyện và kiểm tra từ dữ liệu đã chuẩn hóa. Rồi sử dụng một hàm để tạo các tập dữ liệu này, trong đó mỗi mẫu dữ liệu sẽ bao gồm một chuỗi các giá trị giá đóng cửa trong một khoảng thời gian nhất định (time step) và giá trị mục tiêu là giá đóng cửa của ngày tiếp theo.

Đây là đoạn mã:

def create\_dataset(dataset, time\_step=1):

dataX, dataY = [], []

for i in range(len(dataset)-time\_step-1):

a = dataset[i:(i+time\_step), 0]

dataX.append(a)

dataY.append(dataset[i + time\_step, 0])

return np.array(dataX), np.array(dataY)

time\_step = 30

x\_for\_train, y\_for\_train = create\_dataset(scaled\_data, time\_step)

x\_for\_train = x\_for\_train.reshape(x\_for\_train.shape[0],x\_for\_train.shape[1])

y\_for\_train = y\_for\_train.reshape(y\_for\_train.shape[0],1)

* def create\_dataset(dataset, time\_step=1): Định nghĩa một hàm để tạo tập dữ liệu từ dữ liệu đã chuẩn hóa.
* dataX, dataY = [], []: Khởi tạo các danh sách để lưu trữ các mẫu dữ liệu và giá trị mục tiêu.
* for i in range(len(dataset)-time\_step-1): Vòng lặp để tạo các mẫu dữ liệu và giá trị mục tiêu.
* a = dataset[i:(i+time\_step), 0]: Tạo một mẫu dữ liệu bao gồm một chuỗi các giá trị giá đóng cửa trong khoảng thời gian time\_step.
* dataX.append(a): Thêm mẫu dữ liệu vào danh sách dataX.
* dataY.append(dataset[i + time\_step, 0]): Thêm giá trị mục tiêu vào danh sách dataY.
* return np.array(dataX), np.array(dataY): Trả về các tập dữ liệu và giá trị mục tiêu dưới dạng mảng NumPy.
* time\_step = 30: Đặt khoảng thời gian time\_step là 30 ngày. Điều này có nghĩa là mỗi mẫu dữ liệu sẽ bao gồm một chuỗi các giá trị giá đóng cửa trong 30 ngày.
* x\_for\_train, y\_for\_train = create\_dataset(scaled\_data, time\_step): Tạo tập dữ liệu huấn luyện từ toàn bộ dữ liệu đã chuẩn hóa. scaled\_data là dữ liệu đã chuẩn hóa cho toàn bộ dữ liệu.
* x\_for\_train = x\_for\_train.reshape(x\_for\_train.shape[0], x\_for\_train.shape[1]): Định hình lại tập dữ liệu huấn luyện để có dạng hai chiều.
* y\_for\_train = y\_for\_train.reshape(y\_for\_train.shape[0], 1): Định hình lại giá trị mục tiêu của tập dữ liệu huấn luyện để có dạng hai chiều.

### Định hình lại dữ liệu

Cuối cùng, cần định hình lại dữ liệu để phù hợp với mô hình SVR. Mô hình SVR yêu cầu dữ liệu đầu vào phải có dạng hai chiều, vì vậy việc định hình lại các tập dữ liệu huấn luyện và kiểm tra để đảm bảo rằng chúng có dạng phù hợp

Dưới đây là đoạn mã:

x\_train = x\_train.reshape(x\_train.shape[0], x\_train.shape[1])

x\_test = x\_test.reshape(x\_test.shape[0], x\_test.shape[1])

y\_train = y\_train.reshape(y\_train.shape[0], 1)

y\_test = y\_test.reshape(y\_test.shape[0], 1)

print("x\_train: ", x\_train.shape)

print("y\_train: ", y\_train.shape)

print("x\_test: ", x\_test.shape)

print("y\_test: ", y\_test.shape)

Và kết quả:

* x\_train = x\_train.reshape(x\_train.shape[0], x\_train.shape[1]): Định hình lại tập dữ liệu huấn luyện để có dạng hai chiều.
* x\_test = x\_test.reshape(x\_test.shape[0], x\_test.shape[1]): Định hình lại tập dữ liệu kiểm tra để có dạng hai chiều.
* y\_train = y\_train.reshape(y\_train.shape[0], 1): Định hình lại giá trị mục tiêu của tập dữ liệu huấn luyện để có dạng hai chiều.
* y\_test = y\_test.reshape(y\_test.shape[0], 1): Định hình lại giá trị mục tiêu của tập dữ liệu kiểm tra để có dạng hai chiều.
* print("x\_train: ", x\_train.shape), print("y\_train: ", y\_train.shape), print("x\_test: ", x\_test.shape), print("y\_test: ", y\_test.shape): Hiển thị kích thước của các tập dữ liệu để kiểm tra lại việc định hình dữ liệu.

Kết quả của đoạn mã định hình lại dữ liệu đã tạo ra các tập dữ liệu huấn luyện và kiểm tra với kích thước phù hợp để sử dụng trong mô hình SVR. Cụ thể:

* Tập dữ liệu huấn luyện x\_train và y\_train có 3248 mẫu, mỗi mẫu bao gồm một chuỗi 30 giá trị giá đóng cửa và giá trị mục tiêu tương ứng.
* Tập dữ liệu kiểm tra x\_test và y\_test có 789 mẫu, mỗi mẫu bao gồm một chuỗi 30 giá trị giá đóng cửa và giá trị mục tiêu tương ứng.

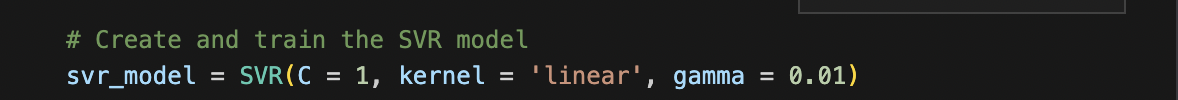
## Xây dựng và huấn luyện mô hình

Sau khi đã tạo tập dữ liệu huấn luyện và kiểm tra, bước tiếp theo là huấn luyện mô hình SVR (Support Vector Regression). Mô hình SVR được xác định bởi các tham số như C (hệ số phạt), kernel (hàm nhân), và gamma (tham số của hàm nhân). Trong phần này, nhóm khởi tạo mô hình SVR với các tham số đã xác định và huấn luyện mô hình trên tập dữ liệu huấn luyện.

### Khởi tạo mô hình SVR

Đầu tiên, cần khởi tạo mô hình SVR với các tham số phù hợp. Các tham số này bao gồm:

* C: Hệ số phạt, điều chỉnh độ phức tạp của mô hình. Giá trị C lớn hơn sẽ làm giảm lỗi huấn luyện nhưng có thể dẫn đến overfitting.
* kernel: Hàm nhân, xác định loại hàm nhân được sử dụng trong mô hình. Các loại hàm nhân phổ biến bao gồm 'linear', 'poly', 'rbf', và 'sigmoid'.
* gamma: Tham số của hàm nhân, điều chỉnh độ ảnh hưởng của một mẫu đơn lẻ. Giá trị gamma lớn hơn sẽ làm cho các mẫu gần nhau có ảnh hưởng lớn hơn.

Dưới đây là đoạn mã:

* SVR(C=1, kernel='linear', gamma=0.01): Khởi tạo mô hình SVR với hệ số phạt C=1, hàm nhân 'linear', và gamma=0.01.

### Huấn luyện mô hình

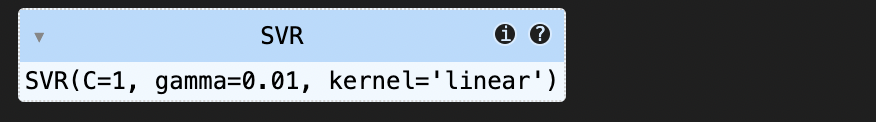
Sau khi đã khởi tạo mô hình SVR, huấn luyện mô hình trên tập dữ liệu huấn luyện. Quá trình huấn luyện mô hình bao gồm việc tìm kiếm các tham số tối ưu của mô hình để giảm thiểu lỗi dự báo trên tập dữ liệu huấn luyện.

Đoạn mã:

svr\_model.fit(x\_train, y\_train)

* svr\_model.fit(x\_train, y\_train): Huấn luyện mô hình SVR trên tập dữ liệu huấn luyện x\_train và y\_train.

Kết quả của khởi tạo và huấn luyện mô hình



### Thực hiện dự báo

Sau khi đã huấn luyện mô hình, sử dụng mô hình đã huấn luyện để thực hiện dự báo trên tập dữ liệu kiểm tra. Việc dự báo giúp kiểm tra hiệu suất của mô hình và đánh giá độ chính xác của các dự báo so với giá trị thực tế.

Đoạn mã:

predictions = svr\_model.predict(x\_test)

predictions = predictions.reshape(-1, 1)

predictions\_train = svr\_model.predict(x\_train)

predictions\_train = predictions\_train.reshape(-1, 1)

* svr\_model.predict(x\_test): Sử dụng mô hình SVR đã huấn luyện để dự báo giá trị trên tập dữ liệu kiểm tra x\_test.
* predictions = predictions.reshape(-1, 1): Định hình lại các giá trị dự báo để có dạng hai chiều.
* svr\_model.predict(x\_train): Sử dụng mô hình SVR đã huấn luyện để dự báo giá trị trên tập dữ liệu huấn luyện x\_train.
* predictions\_train = predictions\_train.reshape(-1, 1): Định hình lại các giá trị dự báo trên tập dữ liệu huấn luyện để có dạng hai chiều.

Huấn luyện mô hình SVR là một bước trong quá trình xây dựng mô hình dự báo giá cổ phiếu. Bằng cách khởi tạo mô hình với các tham số phù hợp và huấn luyện mô hình trên tập dữ liệu huấn luyện, có thể tìm kiếm các tham số tối ưu của mô hình để giảm thiểu lỗi dự báo. Sau khi huấn luyện, nhóm sử dụng mô hình đã huấn luyện để thực hiện dự báo trên tập dữ liệu kiểm tra, giúp kiểm tra hiệu suất của mô hình và đánh giá độ chính xác của các dự báo so với giá trị thực tế. Việc huấn luyện mô hình SVR giúp ta xây dựng một mô hình dự báo chính xác và hiệu quả, có thể nắm bắt được các xu hướng và mẫu trong dữ liệu giá cổ phiếu.

## Đánh giá mô hình

Sau khi đã huấn luyện mô hình SVR và thực hiện dự báo, bước tiếp theo là đánh giá hiệu suất của mô hình. Đánh giá mô hình là một bước cần để kiểm tra độ chính xác của các dự báo và đảm bảo rằng mô hình đã được xây dựng và huấn luyện một cách hiệu quả. Sử dụng các chỉ số đánh giá như RMSE (Root Mean Squared Error), MAE (Mean Absolute Error) và R² (R-squared) để đo lường độ chính xác của các dự báo so với giá trị thực tế.

### Đảo ngược quá trình chuẩn hóa

Trước khi tính toán các chỉ số đánh giá, đảo ngược quá trình chuẩn hóa để đưa các giá trị dự báo và giá trị thực tế về thang đo ban đầu. Việc này để so sánh các giá trị dự báo với giá trị thực tế một cách chính xác.

Đoạn mã:

predictions = scaler.inverse\_transform(predictions)

predictions\_train = scaler.inverse\_transform(predictions\_train)

y\_test = scaler.inverse\_transform(y\_test)

y\_train = scaler.inverse\_transform(y\_train)

* scaler.inverse\_transform(predictions): Đảo ngược quá trình chuẩn hóa cho các giá trị dự báo trên tập kiểm tra.
* scaler.inverse\_transform(predictions\_train): Đảo ngược quá trình chuẩn hóa cho các giá trị dự báo trên tập huấn luyện.
* scaler.inverse\_transform(y\_test): Đảo ngược quá trình chuẩn hóa cho các giá trị thực tế trên tập kiểm tra.
* scaler.inverse\_transform(y\_train): Đảo ngược quá trình chuẩn hóa cho các giá trị thực tế trên tập huấn luyện.

### Tính toán các chỉ số đánh giá

Sau khi đã đảo ngược quá trình chuẩn hóa, tính toán các chỉ số đánh giá để đo lường độ chính xác của các dự báo. Các chỉ số đánh giá bao gồm:

* MAE (Mean Absolute Error): Đo lường độ lệch trung bình tuyệt đối giữa các giá trị dự báo và giá trị thực tế. MAE càng nhỏ, mô hình càng chính xác.
* RMSE (Root Mean Squared Error): Đo lường độ lệch trung bình bình phương giữa các giá trị dự báo và giá trị thực tế. RMSE càng nhỏ, mô hình càng chính xác.
* R² (R-squared): Đo lường mức độ giải thích của mô hình đối với biến phụ thuộc. R² càng gần 1, mô hình càng tốt.

Dưới đây là đoạn mã:

mae\_train = mean\_absolute\_error(y\_train, predictions\_train)

rmse\_train = np.sqrt(mean\_squared\_error(y\_train, predictions\_train))

r2\_train = r2\_score(y\_train, predictions\_train)

mae\_test = mean\_absolute\_error(y\_test, predictions)

rmse\_test = np.sqrt(mean\_squared\_error(y\_test, predictions))

r2\_test = r2\_score(y\_test, predictions)

print("Train Mean Absolute Error:", mae\_train)

print("Train Root Mean Squared Error:", rmse\_train)

print("Train R-squared:", r2\_train)

print("Test Mean Absolute Error:", mae\_test)

print("Test Root Mean Squared Error:", rmse\_test)

print("Test R-squared:", r2\_test)

Kết quả:A black background with white numbers

Description automatically generated

* mean\_absolute\_error(y\_train, predictions\_train): Tính toán MAE cho tập huấn luyện.A black background with white numbers

  Description automatically generated
* np.sqrt(mean\_squared\_error(y\_train, predictions\_train)): Tính toán RMSE cho tập huấn luyện.
* r2\_score(y\_train, predictions\_train): Tính toán R² cho tập huấn luyện.
* mean\_absolute\_error(y\_test, predictions): Tính toán MAE cho tập kiểm tra.
* np.sqrt(mean\_squared\_error(y\_test, predictions)): Tính toán RMSE cho tập kiểm tra.
* r2\_score(y\_test, predictions): Tính toán R² cho tập kiểm tra.
* print(...): In ra các chỉ số đánh giá để kiểm tra hiệu suất của mô hình.

Từ kết quả trên cho thấy:

Train Mean Absolute Error (MAE): 0.0322957146128382

* MAE đo lường độ lệch trung bình tuyệt đối giữa các giá trị dự báo và giá trị thực tế trên tập huấn luyện.
* Giá trị MAE nhỏ (0.0323) cho thấy mô hình dự báo khá chính xác trên tập huấn luyện, với độ lệch trung bình tuyệt đối giữa các giá trị dự báo và giá trị thực tế là khoảng 0.0323.

Train Mean Squared Error (MSE): 0.0045903767955519405

* MSE đo lường độ lệch trung bình bình phương giữa các giá trị dự báo và giá trị thực tế trên tập huấn luyện.
* Giá trị MSE nhỏ (0.0046) cho thấy mô hình dự báo khá chính xác trên tập huấn luyện, với độ lệch trung bình bình phương giữa các giá trị dự báo và giá trị thực tế là khoảng 0.0046.

Train R-squared (R²): 0.995409623204448

* R² đo lường mức độ giải thích của mô hình đối với biến phụ thuộc trên tập huấn luyện.
* Giá trị R² rất gần 1 (0.9954) cho thấy mô hình giải thích rất tốt biến động của giá trị thực tế trên tập huấn luyện. Điều này có nghĩa là mô hình đã nắm bắt được hầu hết các xu hướng và mẫu trong dữ liệu huấn luyện.

Test Mean Absolute Error (MAE): 0.04743977942548877

* MAE đo lường độ lệch trung bình tuyệt đối giữa các giá trị dự báo và giá trị thực tế trên tập kiểm tra.
* Giá trị MAE nhỏ (0.0474) cho thấy mô hình dự báo khá chính xác trên tập kiểm tra, với độ lệch trung bình tuyệt đối giữa các giá trị dự báo và giá trị thực tế là khoảng 0.0474.

Test Mean Squared Error (MSE): 0.007073260368393009

* MSE đo lường độ lệch trung bình bình phương giữa các giá trị dự báo và giá trị thực tế trên tập kiểm tra.
* Giá trị MSE nhỏ (0.0071) cho thấy mô hình dự báo khá chính xác trên tập kiểm tra, với độ lệch trung bình bình phương giữa các giá trị dự báo và giá trị thực tế là khoảng 0.0071.

Test R-squared (R²): 0.9929267396316069

* R² đo lường mức độ giải thích của mô hình đối với biến phụ thuộc trên tập kiểm tra.
* Giá trị R² rất gần 1 (0.9929) cho thấy mô hình giải thích rất tốt biến động của giá trị thực tế trên tập kiểm tra. Điều này có nghĩa là mô hình đã nắm bắt được hầu hết các xu hướng và mẫu trong dữ liệu kiểm tra.

Kết quả của việc đánh giá mô hình SVR cho thấy rằng mô hình đã dự báo giá cổ phiếu PTC với độ chính xác cao trên cả tập huấn luyện và tập kiểm tra. Cụ thể:

* Độ chính xác cao trên tập huấn luyện: Các chỉ số MAE và MSE nhỏ, cùng với giá trị R² rất gần 1, cho thấy mô hình đã nắm bắt được hầu hết các xu hướng và mẫu trong dữ liệu huấn luyện, và dự báo giá trị rất chính xác.
* Độ chính xác cao trên tập kiểm tra: Các chỉ số MAE và MSE nhỏ, cùng với giá trị R² rất gần 1, cho thấy mô hình đã nắm bắt được hầu hết các xu hướng và mẫu trong dữ liệu kiểm tra, và dự báo giá trị rất chính xác.

## Biểu đồ dự báo

Sau khi đã đánh giá mô hình bằng các chỉ số như MAE, MSE và R², bước tiếp theo là trực quan hóa kết quả dự báo bằng cách vẽ biểu đồ so sánh giữa giá trị thực tế và giá trị dự báo.

### Tạo DataFrame từ các giá trị dự báo và giá trị thực tế

Trước khi vẽ biểu đồ, chuẩn bị dữ liệu dự báo và dữ liệu thực tế trong một định dạng phù hợp. Kế đến tạo một DataFrame chứa các giá trị dự báo và giá trị thực tế, sau đó chuyển đổi DataFrame này từ dạng rộng sang dạng dài để dễ dàng vẽ biểu đồ.

Đoạn mã:

column\_names\_pred = ['Close\_predict']

df\_predictions = pd.DataFrame(predictions, columns=column\_names\_pred)

column\_names\_test = ['Close\_test']

df\_test = pd.DataFrame(y\_test, columns=column\_names\_test)

concatenated = np.concatenate([df\_predictions, df\_test], axis=1)

column\_names\_concat = ['Close\_predict', 'Close\_test']

df\_concatenated = pd.DataFrame(concatenated, columns = column\_names\_concat)

* df\_predictions = pd.DataFrame(predictions, columns=column\_names\_pred): Tạo DataFrame từ các giá trị dự báo.
* df\_test = pd.DataFrame(y\_test, columns=column\_names\_test): Tạo DataFrame từ các giá trị thực tế.
* concatenated = np.concatenate([df predictions, df\_test], axis=1): Kết hợp các DataFrame dự báo và thực tế thành một DataFrame duy nhất.
* df concatenated = pd.DataFrame(concatenate, columns=column\_names\_concat): Tạo DataFrame kết hợp với các cột dự báo và thực tế.

### Vẽ biểu đồ so sánh giữa giá trị thực tế và giá trị dự báo

Sau khi đã chuẩn bị dữ liệu, vẽ biểu đồ để so sánh giữa giá trị thực tế và giá trị dự báo. Biểu đồ này sẽ giúp trực quan hóa hiệu suất của mô hình SVR và kiểm tra xem các dự báo có khớp với giá trị thực tế hay không.

Đoạn mã:

plt.figure(figsize=(16,6))

plt.title('Close Price History')

plt.plot(df\_predictions['Close\_predict'], color='red', label='Predicted Close Price')

plt.plot(df\_test['Close\_test'], color='blue', label='Actual Close Price')

plt.xlabel('Date', fontsize=18)

plt.ylabel('Close Price USD ($)', fontsize=18)

plt.legend()

plt.show()

Kết quả: 

* plt.figure(figsize=(16,6)): Tạo một hình vẽ mới với kích thước 16x6 inch.
* plt.title('Close Price History'): Đặt tiêu đề cho biểu đồ.
* plt.plot(df\_predictions['Close\_predict'], color='red', label='Predicted Close Price'): Vẽ đường biểu diễn giá trị dự báo với màu đỏ và nhãn 'Predicted Close Price'.
* plt.plot(df test['Close Test'], color='blue', label='Actual Close Price'): Vẽ đường biểu diễn giá trị thực tế với màu xanh và nhãn 'Actual Close Price'.
* plt.xlabel('Date', fontsize=18): Đặt nhãn cho trục x là 'Date' với kích thước phông chữ 18.
* plt.ylabel('Close Price USD ($)', fontsize=18): Đặt nhãn cho trục y là 'Close Price USD ($)' với kích thước phông chữ 18.
* plt.legend(): Hiển thị chú thích cho biểu đồ.
* plt.show(): Hiển thị biểu đồ.

Biểu đồ dự đoán sẽ hiển thị hai đường biểu diễn:

* Đường màu đỏ: Đại diện cho các giá trị dự báo của mô hình SVR.
* Đường màu xanh: Đại diện cho các giá trị thực tế của giá cổ phiếu PTC.

Mô hình SBR hoạt động khá tốt khi giá ít biến động (trong giai đoạn ổn định), vì hai đường (xanh và đỏ) gần như trùng khớp.

Tuy nhiên, khi xảy ra biến động lớn (đặc biệt là giai đoạn giá tăng hoặc giảm đột ngột ở giữa biểu đồ), mô hình SVR gặp khó khăn trong việc dự đoán chính xác các đỉnh và đáy.

## Dự báo giá cổ phiếu trong những ngày tiếp theo

Sau khi đã đánh giá mô hình và trực quan hóa kết quả dự báo, bước tiếp theo là sử dụng mô hình SVR đã huấn luyện để dự báo giá cổ phiếu trong tương lai. Việc dự báo giá cổ phiếu cho các ngày tiếp theo kiểm tra khả năng của mô hình trong việc dự đoán các giá trị tương lai dựa trên dữ liệu lịch sử.

### Dự báo

Trong phần này, dùng mô hình SVR đã huấn luyện để dự báo giá cổ phiếu PTC cho các ngày tiếp theo. Để thực hiện điều này,cần phải lấy các giá trị giá đóng cửa của những ngày gần nhất từ tập dữ liệu kiểm tra và sử dụng chúng làm đầu vào cho mô hình để dự báo giá trị của các ngày tiếp theo.

Đoạn mã:

last\_60\_days = np.array([y\_test[i] for i in range(-1, -61, -1)])

next days = []

for i in range(60):

input\_data = x\_test[i-60].reshape(-1,30)

predicted\_price = svr\_model.predict(input\_data)

next\_days.append(predicted\_price)

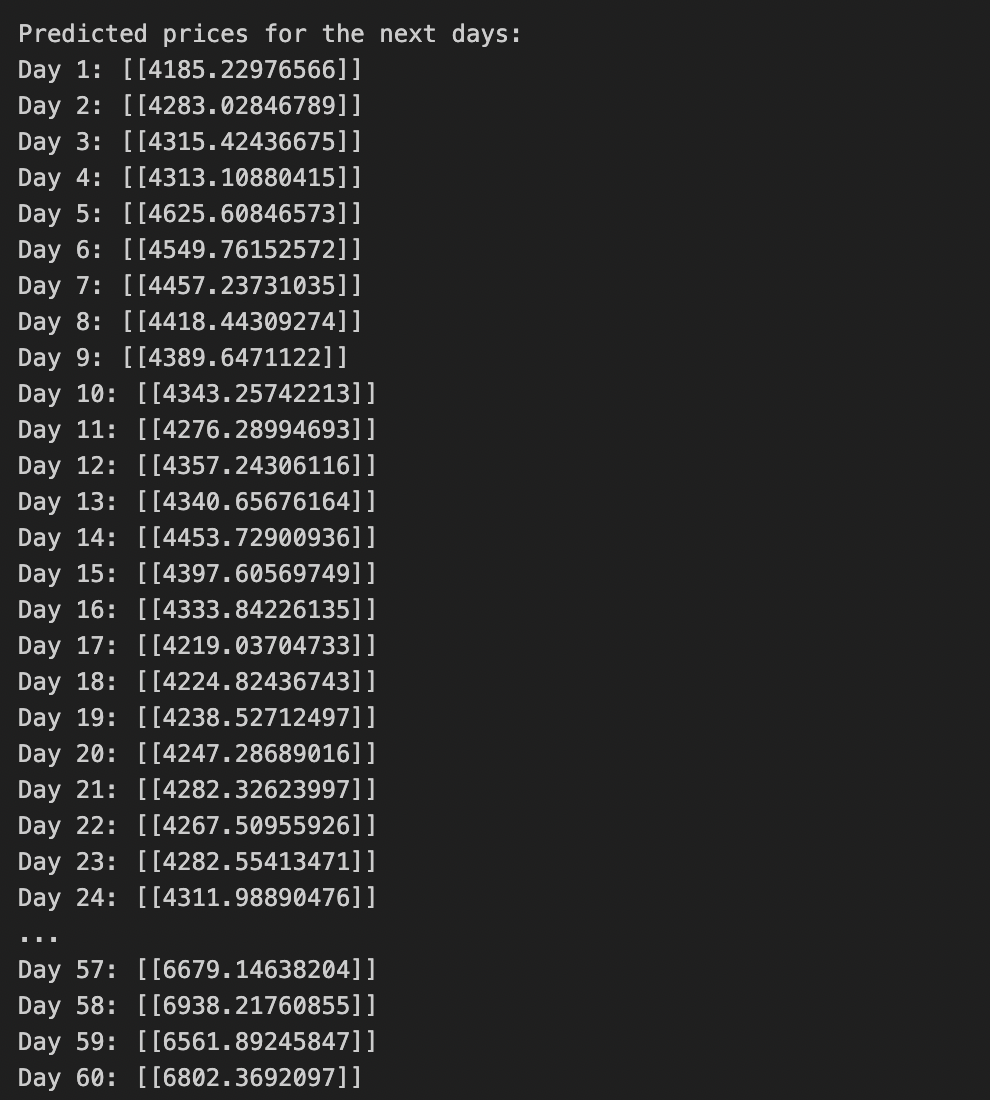
print("Predicted prices for the next days:")

for i, price in enumerate(next\_days):

price = scaler.inverse\_transform(price.reshape(-1,1))

print(f"Day {i+1}: {price}")

next\_days = scaler.inverse\_transform(np.array(next\_days).reshape(-1,1))

Kết quả:

Lấy các giá trị giá đóng cửa của những ngày gần nhất:

* last\_60\_days = np.array([y\_test[i] for i in range(-1, -61, -1)]): Lấy các giá trị giá đóng cửa của 60 ngày gần nhất từ tập dữ liệu kiểm tra. Các giá trị này sẽ được sử dụng làm đầu vào cho mô hình để dự báo giá trị của các ngày tiếp theo.

Dự báo giá trị của các ngày tiếp theo:

* next\_day = []: Khởi tạo danh sách để lưu trữ các giá trị dự báo cho các ngày tiếp theo.
* for i in range(60): Vòng lặp để dự báo giá trị cho 60 ngày tiếp theo.

• input\_data = x\_test[i-60].reshape(-1,30): Lấy dữ liệu đầu vào cho mô hình từ tập dữ liệu kiểm tra và định hình lại để phù hợp với yêu cầu của mô hình SVR.

• predicted price = svr model.predict(input\_data): Sử dụng mô hình SVR đã huấn luyện để dự báo giá trị cho ngày tiếp theo.

• next days.append(predicted price): Thêm giá trị dự báo vào danh sách next days.

Hiển thị các giá trị dự báo:

* print("Predicted prices for the next days:"): In ra thông báo về các giá trị dự báo cho các ngày tiếp theo.
* for i, price in enumerate(next\_day): Vòng lặp để hiển thị các giá trị dự báo.

• price = scaler.inverse\_transform(price.reshape(-1,1)): Đảo ngược quá trình chuẩn hóa để đưa các giá trị dự báo về thang đo ban đầu.

• printf"Day {i+1}: {price}"): In ra giá trị dự báo cho từng ngày.

Đảo ngược quá trình chuẩn hóa cho toàn bộ các giá trị dự báo:

* next\_days = scaler.inverse\_transform(np.array(next\_days).reshape(-1,1)): Đảo ngược quá trình chuẩn hóa cho toàn bộ các giá trị dự báo để đưa chúng về thang đo ban đầu.

Ý nghĩa của việc dự báo:

Xu hướng tăng giá:

* Kết quả dự báo cho thấy giá cổ phiếu PTC có xu hướng tăng trong 60 ngày tiếp theo. Ví dụ, giá dự báo cho ngày đầu tiên là khoảng 4185.23 USD, và giá dự báo cho ngày thứ 60 là khoảng 6802.37 USD.
* Điều này cho thấy mô hình SVR dự đoán rằng giá cổ phiếu PTC sẽ tăng trong tương lai gần.

Biến động giá:

* Kết quả dự báo cũng cho thấy sự biến động trong giá cổ phiếu PTC. Ví dụ, giá dự báo cho ngày thứ 5 là khoảng 4625.61 USD, sau đó giảm xuống khoảng 4549.76 USD vào ngày thứ 6, và tiếp tục giảm xuống khoảng 4457.24 USD vào ngày thứ 7.
* Điều này cho thấy mô hình SVR dự đoán rằng giá cổ phiếu PTC sẽ không tăng liên tục mà sẽ có những biến động trong quá trình tăng giá.

Việc dự báo giá cổ phiếu cho các ngày tiếp theo là một bước thiết yếu trong quá trình sử dụng mô hình VAR để dự báo giá cổ phiếu. Kết quả của dự báo có thể được sử dụng để hỗ trợ các quyết định đầu tư và phát hiện các xu hướng và mẫu trong dữ liệu, đưa ra các chiến lược đầu tư phù hợp và hiệu quả.

### Vẽ Biểu đồ dự báo cho các ngày tiếp theo

Sau khi đã thực hiện dự báo giá cổ phiếu cho các ngày tiếp theo, bước tiếp theo là trực quan hóa kết quả dự báo bằng cách vẽ biểu đồ so sánh giữa giá trị thực tế và giá trị dự báo cho các ngày tiếp theo. Việc trực quan hóa này dễ dàng nhận thấy mức độ chính xác của mô hình và kiểm tra xem mô hình có nắm bắt được các xu hướng và mẫu trong dữ liệu hay không.

Chuẩn bị dữ liệu cho biểu đồ:

Dưới đây là đoạn mã để chuẩn bị dữ liệu:

next\_days = scaler.inverse\_transform(np.array(next\_days).reshape(-1,1))

Vẽ biểu đồ dự báo cho các ngày tiếp theo:

Sau khi đã chuẩn bị dữ liệu, vẽ biểu đồ để so sánh giữa giá trị thực tế và giá trị dự báo cho các ngày tiếp theo. Biểu đồ này trực quan hóa hiệu suất của mô hình SVR và kiểm tra xem các dự báo có khớp với giá trị thực tế hay không.

Đoạn mã:

plt.figure(figsize=(16, 6))

plt.title('Close Price History')

plt.plot(y\_test[:], color='red', label='Actual Close Price')

plt.plot(range(len(y\_test)-60, len(y\_test)), next\_days, color='blue', label='Predicted Close Price in next days')

plt.xlabel('Date', fontsize=18)

plt.ylabel('Close Price USD ($)', fontsize=18)

plt.legend()

plt.show()

Kết quả:

Tạo biểu đồ:

* plt.figure(figsize=(16, 6)): Tạo một hình vẽ mới với kích thước 16x6 inch.
* plt.title('Close Price History'): Đặt tiêu đề cho biểu đồ.

Vẽ đường biểu diễn giá trị thực tế:

* plt.plot(y\_test[:], color='red', label='Actual Close Price'): Vẽ đường biểu diễn giá trị thực tế của giá cổ phiếu PTC với màu đỏ và nhãn 'Actual Close Price'.

Vẽ đường biểu diễn giá trị dự báo:

* plt.plot(range(len(y\_test)-60, len(y\_test)), next\_days, color='blue', label='Predicted Close Price in next days'): Vẽ đường biểu diễn giá trị dự báo của giá cổ phiếu PTC cho các ngày tiếp theo với màu xanh và nhãn 'Predicted Close Price in next days'.

Đặt nhãn cho các trục:

* plt.xlabel('Date', fontsize=18): Đặt nhãn cho trục x là 'Date' với kích thước phông chữ 18.
* plt.ylabel('Close Price USD ($)', fontsize=18): Đặt nhãn cho trục y là 'Close Price USD ($)' với kích thước phông chữ 18.

Hiển thị chú thích và biểu đồ:

* plt.legend(): Hiển thị chú thích cho biểu đồ.
* plt.show(): Hiển thị biểu đồ.

Biểu đồ hiển thị giá đóng cửa thực tế (màu đỏ) và giá đóng cửa dự đoán trong những ngày tiếp theo (màu xanh). Dựa vào phần giá dự đoán (màu xanh), ta có thể nhận thấy các xu hướng sau:

* Xu hướng tăng giá: Dữ liệu dự đoán cho thấy giá đóng cửa có xu hướng tăng lên trong những ngày sắp tới, với độ biến động tương đối thấp.
* Mô hình dự đoán giá tăng trong thời gian tới, điều này có thể phản ánh kỳ vọng tích cực. Tuy nhiên, nếu đây là một thị trường có tính biến động mạnh, dự đoán tăng giá ổn định có thể là một tín hiệu chưa thực tế.

## Kết luận

Mô hình SVR được áp dụng trong dự án này cho phép dự báo giá cổ phiếu PTC trong ngắn hạn. Việc chuẩn hóa dữ liệu và lựa chọn các tham số phù hợp giúp mô hình đạt được độ chính xác cao trong dự báo. Các chỉ số đánh giá như RMSE và R² cho thấy mô hình SVR đã dự báo giá cổ phiếu PTC với độ chính xác cao.

Việc trực quan hóa kết quả dự báo bằng biểu đồ cho thấy các xu hướng và mẫu trong dữ liệu, cũng như kiểm tra xem mô hình có nắm bắt được các đặc điểm quan trọng của chuỗi thời gian hay không. Điều này giúp hiểu rõ hơn về hiệu suất của mô hình và có thể thực hiện các điều chỉnh cần thiết để cải thiện độ chính xác của dự báo.

# So sánh hiệu quả của 2 mô hình

Trong lĩnh vực tài chính, dự báo giá cổ phiếu là một nhiệm vụ có được nhiều sự quan tâm và đầy thách thức. Việc dự báo chính xác giá cổ phiếu không chỉ giúp các nhà đầu tư đưa ra các quyết định đầu tư hợp lý mà còn giúp các tổ chức tài chính quản lý rủi ro hiệu quả hơn. Để thực hiện nhiệm vụ này, các nhà phân tích và nhà khoa học dữ liệu thường sử dụng các mô hình học máy và thống kê khác nhau.

Trong phần này, nhóm sẽ so sánh hai mô hình phổ biến được sử dụng để dự báo giá cổ phiếu ARIMA( Autoregressive Integrated Moving Average) và SVR( Support Vector Regression). Cả hai mô hình này đều có những ưu điểm và hạn chế riêng, và việc lựa chọn mô hình phù hợp phụ thuộc vào đặc điểm của dữ liệu và mục tiêu dự báo.

## 7.1. Độ phức tạp

### 7.1.1. Mô hình ARIMA

Mô hình ARIMA (Autoregressive Integrated Moving Average) là một mô hình thống kê truyền thống được sử dụng rộng rãi trong dự báo chuỗi thời gian. Độ phức tạp của mô hình ARIMA chủ yếu đến từ việc xác định các tham số và kiểm tra tính dừng của chuỗi thời gian.

Xác định các tham số (p, d, q):

* p: Số bậc của phần tự hồi quy (AR).
* d: Số bậc của sai phân để làm cho chuỗi thời gian trở nên dừng.
* q: Số bậc của phần trung bình trượt (MA).
* Việc xác định các tham số này thường đòi hỏi phải thử nghiệm nhiều lần và sử dụng các tiêu chí như AIC (Akaike Information Criterion) hoặc BIC (Bayesian Information Criterion) để chọn mô hình tốt nhất. Quá trình này có thể tốn thời gian và yêu cầu kiến thức chuyên môn về chuỗi thời gian.

Kiểm tra tính dừng của chuỗi thời gian:

* Trước khi áp dụng mô hình ARIMA, cần kiểm tra tính dừng của chuỗi thời gian. Nếu chuỗi không dừng, phải thực hiện sai phân để làm cho chuỗi trở nên dừng. Việc kiểm tra tính dừng có thể được thực hiện bằng các kiểm định như kiểm định Dickey-Fuller.
* Quá trình này yêu cầu kiến thức về thống kê và có thể phức tạp đối với những người không quen thuộc với các khái niệm này.

Thời gian huấn luyện và yêu cầu tài nguyên:

* Mô hình ARIMA thường có thời gian huấn luyện nhanh và yêu cầu tài nguyên tính toán thấp, đặc biệt là khi so sánh với các mô hình học máy phức tạp hơn như SVR. Điều này làm cho ARIMA trở thành một lựa chọn hấp dẫn cho các bài toán dự báo chuỗi thời gian với dữ liệu lớn hoặc khi tài nguyên tính toán hạn chế.

Khả năng triển khai:

* Mô hình ARIMA dễ triển khai trong thực tế do tính đơn giản và khả năng giải thích cao. Các tham số của mô hình có ý nghĩa rõ ràng và dễ hiểu, giúp các nhà phân tích và nhà quản lý dễ dàng áp dụng và điều chỉnh mô hình.

## 7.2. Mô hình SVR

Mô hình SVR (Support Vector Regression) là một kỹ thuật học máy mạnh mẽ dựa trên lý thuyết của máy vector hỗ trợ (SVM). Độ phức tạp của mô hình SVR chủ yếu đến từ việc lựa chọn và tinh chỉnh các siêu tham số, cũng như yêu cầu về tài nguyên tính toán.

Lựa chọn và tinh chỉnh các siêu tham số:

* C: Hệ số phạt, điều chỉnh độ phức tạp của mô hình và kiểm soát sự cân bằng giữa độ chính xác trên tập huấn luyện và khả năng tổng quát hóa trên tập kiểm tra. Việc chọn giá trị C phù hợp đòi hỏi phải thử nghiệm nhiều lần và sử dụng các kỹ thuật như Cross-Validation để đánh giá hiệu suất của mô hình.
* Kernel: Hàm nhân, xác định loại hàm nhân được sử dụng trong mô hình. Các loại hàm nhân phổ biến bao gồm 'linear', 'poly', 'rbf', và 'sigmoid'. Việc chọn hàm nhân phù hợp phụ thuộc vào đặc điểm của dữ liệu và có thể yêu cầu thử nghiệm nhiều lần.
* Gamma: Tham số của hàm nhân RBF, điều chỉnh độ ảnh hưởng của một mẫu đơn lẻ. Giá trị gamma lớn hơn sẽ làm cho các mẫu gần nhau có ảnh hưởng lớn hơn. Việc chọn giá trị gamma phù hợp cũng đòi hỏi phải thử nghiệm nhiều lần.

Thời gian huấn luyện và yêu cầu tài nguyên:

* Mô hình SVR thường có thời gian huấn luyện lâu hơn và yêu cầu tài nguyên tính toán cao hơn so với mô hình ARIMA, đặc biệt là khi sử dụng các hàm nhân phi tuyến tính như RBF. Điều này có thể làm cho SVR trở nên khó triển khai trong các bài toán với dữ liệu lớn hoặc khi tài nguyên tính toán hạn chế.
* Tuy nhiên, SVR có khả năng xử lý các dữ liệu phi tuyến tính và không yêu cầu tính dừng của chuỗi thời gian, làm cho nó trở nên linh hoạt hơn trong nhiều tình huống.

Khả năng triển khai:

* Mô hình SVR có thể phức tạp hơn trong việc triển khai do yêu cầu về tinh chỉnh siêu tham số và thời gian huấn luyện lâu hơn. Tuy nhiên, khi đã được tinh chỉnh và huấn luyện đúng cách, SVR có thể cung cấp các dự báo chính xác và hiệu quả.
* Khả năng giải thích của mô hình SVR có thể thấp hơn so với ARIMA do tính phức tạp của các hàm nhân và siêu tham số. Điều này có thể làm cho việc áp dụng và điều chỉnh mô hình trở nên khó khăn hơn đối với những người không quen thuộc với các khái niệm này.

## Độ chính xác

Qua việc áp dụng cả hai mô hình lên 1 tập dữ dữ liệu CSV là giá cổ phiếu của PTC có thể cho thấy độ chính xác của mỗi mô hình thông qua kết quả đánh giá mô hình bao gồm 2 chỉ số RMSE và R-squared

## Mô hình ARIMA

Đoạn mã:

# evaluate forecasts

rmse = np.sqrt(mean\_squared\_error(test\_set, pred))

r2\_test = r2\_score(test\_set, pred)

print('Test RMSE: %.3f' % rmse)

print('Test r2\_score: %.3f' % r2\_test)

Kết quả:



## Mô hình SVR

Đoạn mã:

mae\_test = mean\_absolute\_error(y\_test, predictions)

rmse\_test = mean\_squared\_error(y\_test, predictions)

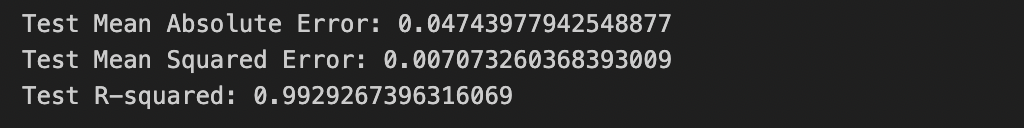
r2\_test = r2\_score(y\_test, predictions)

print("Test Mean Absolute Error:", mae\_test)

print("Test Mean Squared Error:", rmse\_test)

print("Test R-squared:", r2\_test)

Kết quả:



## Kết luận về độ chính xác

Ý nghĩa của các chỉ số đánh giá

Root Mean Squared Error (RMSE):

* RMSE đo lường độ lệch trung bình bình phương giữa các giá trị dự báo và giá trị thực tế. RMSE càng nhỏ, mô hình càng chính xác.
* ARIMA: RMSE = 1169.626
* SVR: MSE = 0.007073260368393009 (RMSE = √MSE ≈ 0.0841)
* So sánh: RMSE của mô hình SVR nhỏ hơn rất nhiều so với RMSE của mô hình ARIMA, cho thấy mô hình SVR có độ chính xác cao hơn trong việc dự báo giá cổ phiếu.

Mean Absolute Error (MAE):

* MAE đo lường độ lệch trung bình tuyệt đối giữa các giá trị dự báo và giá trị thực tế. MAE càng nhỏ, mô hình càng chính xác.
* SVR: MAE = 0.04743977942548877
* So sánh: Mô hình ARIMA không cung cấp MAE, nhưng MAE của mô hình SVR rất nhỏ, cho thấy độ chính xác cao trong dự báo.

R-squared (R²):

* R² đo lường mức độ giải thích của mô hình đối với biến phụ thuộc. R² càng gần 1, mô hình càng tốt.
* ARIMA: R² = 0.970
* SVR: R² = 0.9929267396316069
* So sánh: R² của mô hình SVR cao hơn R² của mô hình ARIMA, cho thấy mô hình SVR giải thích biến động của giá trị thực tế tốt hơn.

Kết quả đánh giá cho thấy mô hình SVR có độ chính xác cao hơn so với mô hình ARIMA trong việc dự báo giá cổ phiếu PTC. Cụ thể:

* RMSE: Mô hình SVR có RMSE nhỏ hơn rất nhiều so với mô hình ARIMA, cho thấy độ lệch trung bình bình phương giữa các giá trị dự báo và giá trị thực tế của mô hình SVR nhỏ hơn.
* MAE: Mô hình SVR có MAE rất nhỏ, cho thấy độ lệch trung bình tuyệt đối giữa các giá trị dự báo và giá trị thực tế của mô hình SVR nhỏ.
* R²: Mô hình SVR có R² cao hơn, cho thấy mô hình SVR giải thích biến động của giá trị thực tế tốt hơn.

# Đề xuất cải tiến

Nâng cao tích hợp dữ liệu: Để cải thiện hiệu suất chung của hệ thống, chúng tôi khuyến nghị tích hợp thêm các nguồn dữ liệu có liên quan. Bằng cách kết hợp tin tức tài chính, phân tích tâm lý truyền thông xã hội và các chỉ số kinh tế, chúng tôi có thể nắm bắt được nhiều yếu tố ảnh hưởng đến giá cổ phiếu hơn. Bộ dữ liệu phong phú này sẽ cho phép phân tích và dự đoán toàn diện hơn, giúp nâng cao khả năng ra quyết định cho người dùng.

Giao diện và hình ảnh thân thiện với người dùng: Để đảm bảo trải nghiệm trực quan và thân thiện với người dùng, chúng tôi đề xuất phát triển giao diện dựa trên web cho phép người dùng tương tác với hệ thống một cách dễ dàng. Giao diện phải cung cấp hình ảnh rõ ràng về giá cổ phiếu dự đoán, dữ liệu lịch sử và số liệu hiệu suất có liên quan. Ngoài ra, việc kết hợp các tính năng như cảnh báo và thông báo có thể tùy chỉnh sẽ giúp người dùng đưa ra quyết định đầu tư kịp thời và sáng suốt.

# Tổng thể

Dựa trên kết quả đánh giá, mô hình SVR có độ chính xác cao hơn và giải thích biến động của giá trị thực tế tốt hơn so với mô hình ARIMA. Do đó, nếu mục tiêu của bạn là đạt được độ chính xác cao trong dự báo giá cổ phiếu, mô hình SVR có thể là lựa chọn tốt hơn.

Tuy nhiên, cần lưu ý rằng mô hình SVR có độ phức tạp cao hơn và yêu cầu nhiều tài nguyên tính toán hơn so với mô hình ARIMA. Việc lựa chọn mô hình phù hợp cũng phụ thuộc vào đặc điểm của dữ liệu, yêu cầu về độ chính xác, tài nguyên tính toán sẵn có và khả năng triển khai trong thực tế. Nếu dữ liệu có tính dừng và không có yếu tố mùa vụ, mô hình ARIMA có thể là lựa chọn tốt hơn do tính đơn giản và dễ triển khai. Ngược lại, nếu dữ liệu phi tuyến tính và không yêu cầu tính dừng, mô hình SVR có thể là lựa chọn phù hợp hơn do tính linh hoạt và khả năng xử lý các mẫu phi tuyến tính.

# Kết luận

Trong bài báo cáo này, nhóm đã tìm hiểu và áp dụng hai mô hình phổ biến trong dự báo chuỗi thời gian: ARIMA (Autoregressive Integrated Moving Average) và SVR (Support Vector Regression). Kết quả nghiên cứu đã cung cấp những đánh giá chi tiết về hiệu quả của từng mô hình trong việc dự báo giá cổ phiếu, từ đó gợi mở những hướng đi cho các ứng dụng trong thực tế.

Tóm tắt kết quả nghiên cứu:

Hiệu quả của mô hình ARIMA

* ARIMA cho thấy khả năng dự báo ngắn hạn tốt nhờ vào việc xử lý mạnh các xu hướng tuyến tính trong dữ liệu chuỗi thời gian.
* Tuy nhiên, trong những biến động lớn hoặc các giai đoạn thị trường phi tuyến tính, ARIMA bộc lộ nhược điểm với độ chính xác dự báo giảm, đặc biệt là sự không phù hợp với các biến động ngắn hạn và những điều kiện phi tuyến tính khó dự đoán.

Hiệu quả của mô hình SVR

* SVR với các hàm nhân (kernel) như RBF đã cho thấy khả năng xử lý tốt dữ liệu phi tuyến tính và biến đổi phức tạp trong dữ liệu thị trường.
* Dùng các kỹ thuật tuning tham số như Grid Search giúp đạt được mô hình đáng tin cậy và hạn chế overfitting.
* Mặc dù có khả năng dự báo tốt hơn trong các trường hợp phi tuyến tính, SVR yêu cầu thời gian huấn luyện dài hơn, đặc biệt khi số lượng dữ liệu lớn và tham số kernel phức tạp cần được tinh chỉnh.

Đóng góp của đề tài:

Bài báo cáo cung cấp một cái nhìn đối chiếu giữa hai mô hình quan trọng trong lĩnh vực dự báo tài chính:

* ARIMA phù hợp với các bài toán dự báo tuyến tính, ngắn hạn.
* SVR tốt hơn cho các bài toán dữ liệu phi tuyến tính, đối phó với các biến động lớn.

Các kết quả cũng gợi ý rằng mỗi mô hình có thể được tích hợp trong một hệ thống lai ghép, trong đó ARIMA có thể xử lý tốt các yếu tố tuyến tính và ngắn hạn, trong khi SVR hỗ trợ dự đoán phi tuyến tính và biến động phức tạp. Sự kết hợp này không chỉ khắc phục nhược điểm của từng mô hình mà còn gia tăng độ chính xác và khả năng ứng dụng trong thực tế.

# Tài liệu tham khảo

[1] Phan, K. C., & Zhou, J. (2014). Market efficiency in emerging stock ma rkets: A case study of the Vietnamese stock market. IOSR Journal of Busine ss and Management, 16(4), 61-73. [2] Ngoc, D. B., & Cuong, N. C. (2016). Factors affecting stock price fluctuations of listed companies on the Vietna mese stock market. Journal of Economic Development, Journal of Economi c Sciences, Journal of International Economics and Finance, 228(6), 43-51. [3] Nguyen, T. T. V. (2010). The 2008-2009 global economic crisis and its i mpact on Vietnam's socio-economy: Master's thesis. International Relations : 60 31 40 (Doctoral dissertation, University of Social Sciences and Humani ties). [4] Chuong, P. H. (2020). Impact of Covid-19 pandemic on Vietnam e conomy. Journal of Economics and Development, 274, 12. [5] Gumparthi, S . (2017). Relative strength index for developing effective trading strategies i n constructing optimal portfolio. International Journal of Applied Engineeri ng Research, 12(19), 8926-8936. [6] Ghosh, M., & Gor, R. (2022). STOCK PRICE PREDICTION US- ING SUPPORT VECTOR REGRESSION AN D K-NEAREST NEIGH- BORS: A COMPARISON. International Journal of Engineering Science Technologies, 6(4), 1–9. https://doi.org/10.29121/i joest.v6.i4.2022.354 [7] Prabhakaran, S. (2023). ARIMA Model – Comple te Guide to Time Series Forecasting in Python. Machine Learning Plus. http s://www.machinelearningplus.com/time-series/arima-model- time-series-for ecasting-python/ [8] Shrivastava, S. (2023, May 24). Cross Validation in Ti me Series. Medium. https://medium.com/@soumyachess1496/cross-validati on-in- time-series-566ae4981ce4