BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC PHENIKAA**

**KHOA CÔNG NGHỆ THÔNG TIN**

**------------oOo-----------**

**Ảnh có chứa Phông chữ, biểu tượng, Đồ họa, văn bản

Mô tả được tạo tự động**

**Đề tài:**

**“Phân tích thị trường chứng khoán”**

**Giảng viên: ThS. Nguyễn Văn Thiệu**

**Nhóm thực hiện: Nhóm 13**

**Thành viên nhóm:**

**Nguyễn Thanh Quang – 21011417 – K15-CNTTVJ1**

**Nguyễn Phát – 21011414 – K15-CNTTVJ1**

**Học phần: Lập trình phân tích dữ liệu với python**

**Hà Nội, 05/2023**

|  |  |  |  |
| --- | --- | --- | --- |
| **Họ và tên** | **MSSV** | **Lớp** | **Phân công** |
| Nguyễn Phát | **21011414** | K15-CNTTVJ1 | 1) Thu thập dữ liệu  2) Làm sạch và xử lí trước dữ liệu  3) Khám phá và trực quan hóa dữ liệu |
| Nguyễn Thanh Quang | **21011417** | K15-CNTTVJ1 | 1) Phát triển các mô hình học máy  2) Diễn giải kết quả  3) Làm Slide |

Mục lục

**I. Giới thiệu và tóm tắt ý tưởng4**

**II. Làm sạch và xử lí trước dữ liệu5**

1. Tiến hành lấy dữ liệu5

2. Tiền xư lí dữ liệu6

**III. Khám phá và trực quan hóa dữ liệu9**

**IV. Phát triển các mô hình học máy12**

1. Phân tích và xử lí dữ liệu đầu vào12

1.1. Tổng quan về bộ dữ liệu12

1.2. Vấn đề về chuỗi không dừng13

1.3. Chọn bước thời gian cho huấn luyện mô hình17

1.4. Chia tập dữ liệu18

2. Xây dựng mô hình19

2.1. CNN19

2.2. RNN24

3. Kết quả dự đoán trên tập test27

3.1. CNN27

3.2. RNN28

4. Đánh giá mô hình trên tập test28

4.1. Đánh giá mô hình CNN29

4.2. Đánh giá mô hình RNN29

4.3. Nhận xét30

5. Dự đoán kết quả cho tuần tới32

**V. Diễn giải kết quả thu được32**

# I. Giới thiệu và tóm tắt ý tưởng

Thị trường chứng khoán là một trong những thị trường tài chính phức tạp nhất, nơi hàng triệu giao dịch được thực hiện mỗi ngày. Để phân tích và dự đoán các xu hướng trong thị trường chứng khoán, các nhà đầu tư và nhà phân tích thị trường đã sử dụng nhiều phương pháp khác nhau, bao gồm cả các mô hình học máy như RNN và CNN.

Trong mục này, chúng tôi sẽ giới thiệu về hai kiến trúc mạng nơ-ron tích chập (Convolutional Neural Network - CNN) và mạng nơ-ron hồi quy (Recurrent Neural Network - RNN) và khám phá cách chúng được áp dụng trong phân tích thị trường chứng khoán.

CNN là một kiến trúc mạng nơ-ron thường được sử dụng trong xử lý ảnh và nhận dạng đối tượng. CNN sử dụng các lớp tích chập để tìm kiếm các đặc trưng không gian trong dữ liệu. Trong phân tích thị trường chứng khoán, CNN có thể được áp dụng để tự động học các mẫu và quy luật không gian trong dữ liệu giá cổ phiếu hoặc biểu đồ giá.

RNN là một kiến trúc mạng nơ-ron phổ biến trong việc xử lý dữ liệu chuỗi hoặc dữ liệu có mối quan hệ thời gian. RNN có khả năng lưu trữ thông tin trạng thái trước đó và sử dụng nó để dự đoán thông tin tiếp theo trong chuỗi. Trong phân tích thị trường chứng khoán, RNN có thể được sử dụng để mô hình hóa sự phụ thuộc thời gian trong dữ liệu giá cổ phiếu và dự đoán xu hướng giá trong tương lai.

Sử dụng RNN và CNN để phân tích thị trường chứng khoán có nhiều lợi ích như sau:

1. Xử lý dữ liệu chuỗi thời gian: Thị trường chứng khoán là một loại dữ liệu chuỗi thời gian, trong đó các giá trị của cổ phiếu, chỉ số và yếu tố kinh tế được ghi lại theo thời gian. RNN là một mô hình phù hợp cho việc phân tích các dữ liệu chuỗi thời gian, bởi vì nó có khả năng lưu trữ thông tin của các trạng thái trước đó và sử dụng chúng để dự đoán trạng thái tiếp theo.

2. Phân tích biểu đồ giá cổ phiếu: CNN là một mô hình phù hợp cho việc phân tích hình ảnh và biểu đồ. Với thị trường chứng khoán, biểu đồ giá cổ phiếu cũng được coi như một loại hình ảnh có thể được xử lý bởi CNN để phát hiện các đường xu hướng và các mô hình giá.

3. Dự đoán xu hướng thị trường: Sử dụng RNN và CNN có thể giúp các nhà đầu tư và nhà phân tích thị trường phân tích và dự đoán xu hướng thị trường. Bằng cách sử dụng các mô hình này, có thể phát hiện được các mô hình, xu hướng và biến động của thị trường chứng khoán, từ đó giúp đưa ra các quyết định đầu tư thông minh hơn.

4. Giảm thời gian phân tích: Sử dụng RNN và CNN có thể giúp giảm thời gian phân tích dữ liệu và dự đoán xu hướng thị trường. Các mô hình này có thể xử lý dữ liệu lớn và tự động hóa phân tích thị trường, giúp giảm thiểu sự phụ thuộc vào nhận định chủ quan của con người.

Tóm lại, sử dụng RNN và CNN để phân tích thị trường chứng khoán có thể giúp đưa ra những quyết định đầu tư thông minh và giảm thiểu sự phụ thuộc vào con người. Tuy nhiên, việc áp dụng các mô hình này cần có dữ liệu đầy đủ và chính xác, và kết quả dự đoán cần được xem xét cẩn thận trước khi đưa ra quyết định đầu tư.

# II. Làm sạch và xử lí trước dữ liệu

1. Tiến hành lấy dữ liệu

Lấy dữ liệu từ trang Alpha Vantage. Alpha Vantage là một dịch vụ cung cấp dữ liệu và công cụ phân tích tài chính, bao gồm dữ liệu chứng khoán, thông tin tài chính, chỉ số kỹ thuật và nhiều hơn nữa. Dịch vụ này cung cấp API miễn phí để truy cập dữ liệu tài chính qua mạng.

Để sử dụng Alpha Vantage, chúng tôi cần làm các bước sau đây:

Đăng ký tài khoản: Truy cập trang web Alpha Vantage (https://www.alphavantage.co/) và đăng ký một tài khoản miễn phí. Sau khi đăng kí thành công sẽ nhận được một API key để lấy dữ liệu.

Lấy API key: Sau khi đăng nhập, chúng tôi dùng API key đã được đăng kí trong phần "My Account". API key này sẽ được sử dụng để xác thực và truy cập dữ liệu từ Alpha Vantage.

Xác thực API key: Trước khi sử dụng API key của bạn, chúng tôi cần thêm nó vào các yêu cầu API của mình. Điều này có thể được thực hiện bằng cách bổ sung tham số apikey trong URL của yêu cầu như sau:

* Để truy cập vào dữ liệu, đầu tiên chúng tôi lấy API key đã được cấp trong tài khoản. APIkey lần này dùng như sau :

API\_KEY = 'WGTR4V192TYX8F33'

* Sau đó chúng tôi dùng đoạn mã sau để yêu cầu dữ liệu được chuyển về:

# Định dạng URL API với ngày hiện tại

url = 'https://www.alphavantage.co/query?function=FX\_WEEKLY&from\_symbol=EUR&to\_symbol=USD&apikey=WGTR4V192TYX8F33'

* Chọn chức năng và yêu cầu dữ liệu: Alpha Vantage cung cấp nhiều chức năng khác nhau để truy cập dữ liệu tài chính. Ví dụ: để lấy dữ liệu lịch sử giá cổ phiếu, bạn có thể sử dụng chức năng "Time Series Daily" bằng cách gửi yêu cầu API tới URL

# Gửi yêu cầu HTTP GET để lấy dữ liệu từ API

r = requests.get(url)

data = r.json()

“function=FX\_WEEKLY&from\_symbol=EUR&to\_symbol=USD&apikey=WGTR4V192TYX8F33” là dữ liệu tỷ giá hối đoái hàng tuần giữa EUR (Euro) và USD (Đô la Mỹ) bằng cách sử dụng chức năng "FX\_WEEKLY" với API key được xác nhận là WGTR4V192TYX8F33.

Xử lý và hiển thị dữ liệu: Khi nhận được dữ liệu từ Alpha Vantage, chúng tôi xử lý nó theo ý muốn của mình . Cụ thể là dữ liệu được trả về dưới dạng JSON sau đó chúng tôi chuyển dữ liệu về CSV để tiện xử lí, sau đó sử dụng các thư viện phân tích dữ liệu như pandas trong Python để xử lý và hiển thị dữ liệu một cách thuận tiện.

2. Tiền xử lí dữ liệu

Sau khi lấy dữ liệu về chúng tôi chuyển dữ liệu thô từ web về 1 file dạng DataFrame để xử lí. Đầu tiên là mở dữ liệu để xem qua:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

Tiếp theo chúng tôi xem kiểu dữ liệu , kiểm tra việc thiếu sót dữ liệu:

Ảnh có chứa văn bản, biên lai, Phông chữ, đại số

Mô tả được tạo tự động

Dữ liệu trên sau khi kiểm tra gồm có:

DataFrame có tổng cộng 999 dòng (entries) và 4 cột (Data columns).

Các cột được đặt tên lần lượt là '1. open', '2. high', '3. low', và '4. close'.

Tất cả các cột đều có 999 giá trị không bị thiếu (non-null count).

Kiểu dữ liệu (Dtype) của các cột là object.

DataFrame này sử dụng khoảng 39.0 KB bộ nhớ.

Tuy nhiên, kiểu dữ liệu của các cột hiện tại đều là object cần chuyển đổi sang kiểu dữ liệu số (float hoặc int, ở đây chúng tôi chuyển dữ liệu về 4 cột là float) để thực hiện các phép tính và phân tích tài chính.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

Sau khi chuyển dữ liệu về dạng số, tiếp theo kiểm tra bằng phương thức describe để xem dữ liệu tóm tắt các thống kê:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

Ta có thể rút ra một số nhận xét về dữ liệu tài chính như sau:

* Số lượng và đầy đủ: Có 999 mẫu dữ liệu cho mỗi cột, điều này cho thấy có một tập dữ liệu đầy đủ.
* Giá trị trung bình: Giá trị trung bình của cột Open, High, Low, và Close lần lượt là 1.242351, 1.254489, 1.230349, và 1.242254. Điều này cho thấy giá trị trung bình của các cột khá gần nhau.
* Độ lệch chuẩn: Độ lệch chuẩn của cột Open, High, Low, và Close lần lượt là 0.127560, 0.129291, 0.125391, và 0.127652. Điều này cho thấy mức độ phân tán của dữ liệu khá nhỏ, tức là giá trị tập trung xung quanh giá trị trung bình.
* Giá trị tối thiểu và tối đa: Giá trị tối thiểu và tối đa của các cột cho thấy phạm vi giá trị trong tập dữ liệu , giá trị tối thiểu của cột Open là 0.967390 và giá trị tối đa là 1.589400.
* Phân vị (Percentiles): Phân vị 25%, 50% (median), và 75% cung cấp thông tin về phân bố của các cột. Phân vị 25% (Q1) cho cột Open là 1.131500, tức là 25% giá trị thấp hơn nằm trong khoảng từ 1.131500 đến giá trị tối thiểu.

III. Khám phá và trực quan hóa dữ liệu

Biểu đồ của SMA giá đóng và MACD:

Ảnh có chứa văn bản, Sơ đồ, hàng, ảnh chụp màn hình

Mô tả được tạo tự động

Ảnh có chứa văn bản, hàng, Sơ đồ, biểu đồ

Mô tả được tạo tự động

Đầu tiên là SMA, SMA (Simple Moving Average) là một chỉ báo kỹ thuật phổ biến được sử dụng để xác định xu hướng giá trên thị trường tài chính. Nó tính toán giá trung bình của một tập hợp các giá trong một khoảng thời gian nhất định.

Cách tính SMA:

Chọn một khoảng thời gian cố định, ví dụ: 10 ngày, 20 ngày, 50 ngày.

Tính giá trung bình của các giá trong khoảng thời gian đó bằng cách cộng tất cả các giá lại và chia cho số lượng giá.

Ví dụ, để tính SMA 10 ngày, ta lấy tổng các giá trong 10 ngày gần nhất và chia cho 10. Kết quả là giá trung bình của 10 ngày đó.

Công dụng của SMA:

Xác định xu hướng: SMA được sử dụng để đánh giá xu hướng giá. Nếu giá đang ở trên SMA, thì xu hướng được coi là tăng. Ngược lại, nếu giá đang ở dưới SMA, thì xu hướng được coi là giảm.

Hỗ trợ và kháng cự: SMA cung cấp các mức hỗ trợ và kháng cự trong biểu đồ giá. Khi giá cắt qua SMA từ dưới lên trên, SMA có thể trở thành mức hỗ trợ. Ngược lại, khi giá cắt qua SMA từ trên xuống dưới, SMA có thể trở thành mức kháng cự.

Tín hiệu giao dịch: SMA cung cấp tín hiệu mua và bán dựa trên sự cắt nhau giữa các đường SMA khác nhau hoặc giữa SMA và giá. Ví dụ, khi đường SMA ngắn cắt qua đường SMA dài từ dưới lên trên, đây có thể là tín hiệu mua. Ngược lại, khi đường SMA ngắn cắt qua đường SMA dài từ trên xuống dưới, đây có thể là tín hiệu bán.

SMA thường được sử dụng cùng với các chỉ báo và công cụ kỹ thuật khác để xác định điểm vào và điểm ra trong giao dịch trên thị trường tài chính.Tiếp theo, MACD (Moving Average Convergence Divergence) là một chỉ báo kỹ thuật phổ biến được sử dụng để xác định xu hướng và tín hiệu giao dịch trên thị trường tài chính. Nó được tính toán bằng cách sử dụng ba thành phần chính:

Đường MACD (MACD Line): Đây là sự khác biệt giữa hai đường trung bình động, thường là sự khác biệt giữa EMA (Exponential Moving Average) của giá đóng cửa trong một khoảng thời gian ngắn và EMA của giá đóng cửa trong một khoảng thời gian dài hơn. Đường MACD được sử dụng để xác định sự tăng giảm của xu hướng.

Đường trung bình kỹ thuật (Signal Line): Đây là một EMA của đường MACD. Nó giúp xác định tín hiệu mua và bán khi đường MACD cắt qua đường Signal Line.

Histogram (Đồ Histogram): Đồ histogram đại diện cho sự khác biệt giữa đường MACD và đường Signal Line. Nó cung cấp thông tin về sự gia tăng hoặc giảm sút của độ mạnh của xu hướng hiện tại.

Cách sử dụng MACD:

- Tín hiệu mua: Khi đường MACD cắt lên trên đường Signal Line, đồ histogram chuyển từ âm sang dương.

- Tín hiệu bán: Khi đường MACD cắt xuống dưới đường Signal Line, đồ histogram chuyển từ dương sang âm.

MACD cung cấp tín hiệu mua và bán dựa trên sự hội tụ và rời rạc giữa hai đường trung bình động. Nó thường được sử dụng như một công cụ kỹ thuật bổ trợ để xác định điểm vào và điểm ra trong giao dịch trên thị trường tài chính.

Qua 2 biểu đồ này chúng ta có thể nhận ra xu hướng của tuần tiếp theo sẽ tăng vì đường xu hướng đã chạm đáy đường quá mua nên đự đoán tương lai tiềm năng là sẽ tăng trưởng .

Để có thể phân tích sơ bộ chắc chắn hơn chúng tôi có thêm biểu đồi RSI để đánh giá:

Ảnh có chứa văn bản, hàng, Sơ đồ, biểu đồ

Mô tả được tạo tự động

RSI (Relative Strength Index) là một chỉ báo kỹ thuật được sử dụng phổ biến trong phân tích kỹ thuật để đo lường sự mạnh yếu của một tài sản tài chính và xác định điểm mua và bán tiềm năng trên thị trường.

Cách tính RSI:

1. Xác định khoảng thời gian tính RSI, thường là 14 ngày.

2. Tính toán giá trị RSI dựa trên biến động của giá trong khoảng thời gian đó.

Công thức tính RSI:

- Đầu tiên, tính toán giá trị của RSI là tỉ lệ giữa số lượng phiên tăng giá và số lượng phiên giảm giá trong khoảng thời gian tính RSI.

- Sau đó, chuyển đổi giá trị RSI thành một đoạn giá trị từ 0 đến 100 để dễ đọc và hiểu.

Ý nghĩa của RSI:

- Giá trị RSI từ 0 đến 30: Thường được xem là khu vực quá bán, cho thấy tài sản có thể bị quá bán và có khả năng tăng giá trong tương lai gần.

- Giá trị RSI từ 70 đến 100: Thường được xem là khu vực quá mua, cho thấy tài sản có thể bị quá mua và có khả năng giảm giá trong tương lai gần.

- Giá trị RSI từ 30 đến 70: Thường được xem là khu vực trung lập, cho thấy tài sản không có xu hướng rõ ràng và có thể di chuyển theo các tín hiệu khác.

Cách sử dụng RSI:

- Tín hiệu mua: Khi RSI tăng lên trên mức quá bán (30) và cắt qua mức 30, đây có thể là tín hiệu mua.

- Tín hiệu bán: Khi RSI giảm xuống dưới mức quá mua (70) và cắt qua mức 70, đây có thể là tín hiệu bán.

RSI có thể được sử dụng để xác định sự mạnh yếu của xu hướng hiện tại và cung cấp tín hiệu mua và bán trong giao dịch trên thị trường tài chính. Tuy nhiên, như với bất kỳ công cụ phân tích kỹ thuật nào, RSI cần được sử dụng kết hợp với các công cụ và phương pháp phân tích khác để đưa ra quyết định giao dịch chính xác.

Như vậy đường quá bán được thiết lập sớm để tạo đáy của 1 chu kì mới nên việc giá đóng sẽ tăng của tuần sau sẽ càng chắc chắn.

IV. Phát triển các mô hình máy học

Bài toán dự báo tài chính, thị trường chứng khoán với bộ dữ liệu time series có rất nhiều mô hình học máy. Cụ thể áp dụng vào bài toán dự đoán thị trường chứng khoán, chúng tôi quyết định sẽ dùng CNN và RNN để dự báo tài chính cho tuần tiếp theo dựa trên bộ dữ liệu về biến động giá chứng khoán theo tuần.

1. Phân tích và xử lí dữ liệu đầu vào

1.1. Tổng quan về bộ dữ liệu

Với bộ dữ liệu về giá đồng EURO trong giai đoạn từ 2004-02-27 đến 2023-04-19 với chu kì theo tuần. Bảng dữ liệu giá đồng EURO gồm có 4 cột và 1000 dòng dữ liệu, 5 dòng đầu tiên:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

Để thuận tiện việc huấn luyện mô hình có hiện quả tốt nhất, tôi sẽ sắp sếp lại bảng với ngày tháng tăng dần, 5 dòng đầu sau khi sắp xếp.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

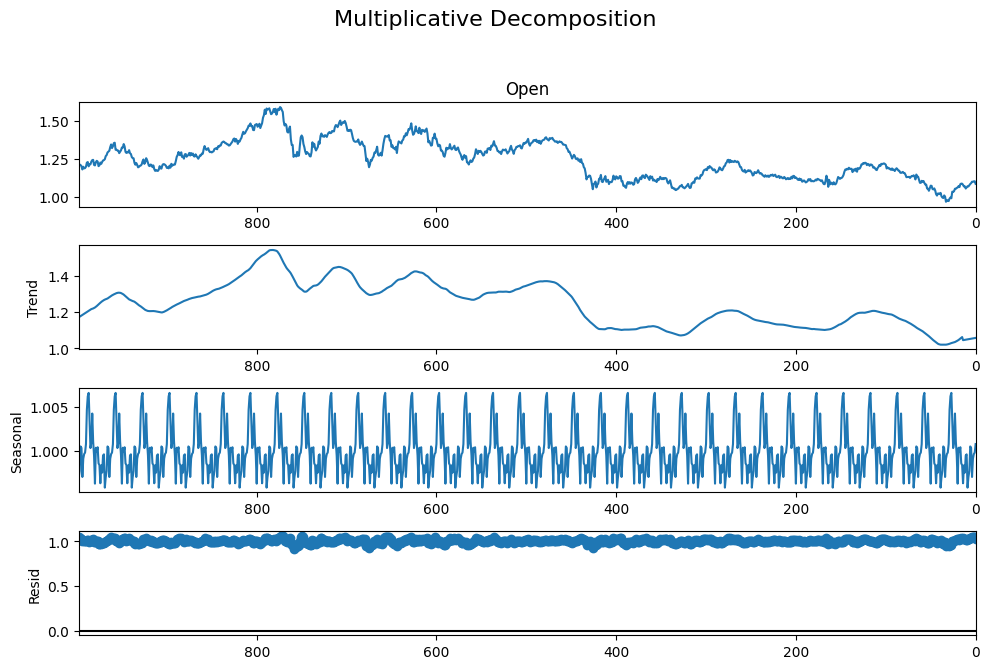
Mô tả được tạo tự động

Bài toán đặt ra sẽ là dự đoán giá đóng của và giá mở cửa trong tương lai. Như vậy, từ bộ dữ liệu tôi lấy hai cột “Open” và “Close” để làm dữ liệu đầu vào cho việc huấn luyện.

1.2. Vấn đề về chuỗi không dừng

Một chuỗi không dừng là một chuỗi gồm có yếu tố xu hướng và tính mùa vụ. Điều này dẫn tới trung bình, phương sai và dộ tự tương quan của chuỗi không dừng phụ thuộc theo thời gian. Như vậy, bộ dữ liệu time series là bộ dữ liệu mà giá trị của nó tuân theo hàm thời gian. Trong khi đó, mô hình học máy tuyến tính lại cần một bộ dữ liệu với các điểm dữ liệu độc lập và có vai trò giống nhau. Tuy nhiên, chuỗi không dừng lại thường có tự tương quan, có nghĩa là chúng không độc lập với nhau. Ngoài ra, nó còn có tính mùa vụ, trend nên có thể sẽ không ổn định và các thuộc tính thống kê có thể thay đổi theo thời gian.Vì vậy, chuỗi không dừng không phù hợp để làm dữ liệu đầu vào cho các mô hình hồi quy tuyến tính. Do đó, nó ảnh hưởng rất nhiều đến kết quả huấn luyện và có thể mô hình tuyến tính sẽ không học được quy luật của chuỗi.

Trực quan hóa về xu hướng, tính mùa vụ, resid của chuỗi “Open”:



Ảnh có chứa văn bản, hàng, Sơ đồ, Phông chữ

Mô tả được tạo tự động

Kiểm tra chuỗi time series “Open” có phải chuỗi dừng bằng ADF Test:

Ảnh có chứa văn bản, Phông chữ, hàng, ảnh chụp màn hình

Mô tả được tạo tự động

Như vậy, chuỗi “Open” đang là một chuỗi không dừng. Nó vẫn có những yếu tố như xu hướng và tính mùa vụ.

Trực quan hóa về xu hướng, tính mùa vụ, resid của chuỗi “Close”:

Ảnh có chứa văn bản, hàng, Sơ đồ, ảnh chụp màn hình

Mô tả được tạo tự động

Ảnh có chứa văn bản, hàng, Sơ đồ, Phông chữ

Mô tả được tạo tự động

Kiểm tra chuỗi time series “Close” có phải chuỗi dừng bằng ADF Test:

Ảnh có chứa văn bản, hàng, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động

Như vậy, chuỗi “Close” đang là một chuỗi không dừng. Nó vẫn có những yếu tố như xu hường và tính mùa vụ.

Sau khi kiểm thử, tôi thấy rằng giá trị p-value của cả hai chuỗi “Open” và “Close” đền lớn hơn giá trị 0.05 và những trực quan hóa cụ thể bên trên. Hai chuỗi time series “Open” và “Close” đều là các chuỗi không dừng.

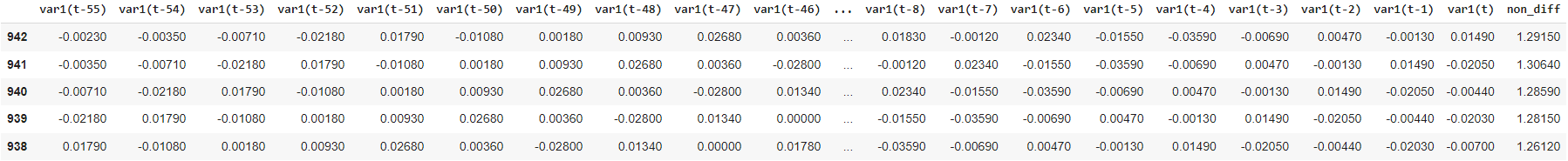
Như vậy, mô hình học máy tuyến tính bình thường có thể sẽ không hiệu quả trong việc tìm ra đặc trưng hay quy luật của hai chuỗi này trên một khoảng thời gian dài hạn như bộ dữ liệu của tôi thu thập được. Như vậy, tôi cần có những mô hình tuyến và một số kĩ thuật nâng cao khác để xử lí chuỗi không dừng. Tuy nhiên, các mô hình học sâu như CNN và RNN là những mô hình phi tuyến tính. Vì vậy, chúng có thể khắc phục những hạn chế này và tìm ra các mẫu phức tạp trong chuỗi không dừng.

1.3. Chọn bước thời gian cho huấn luyện mô hình

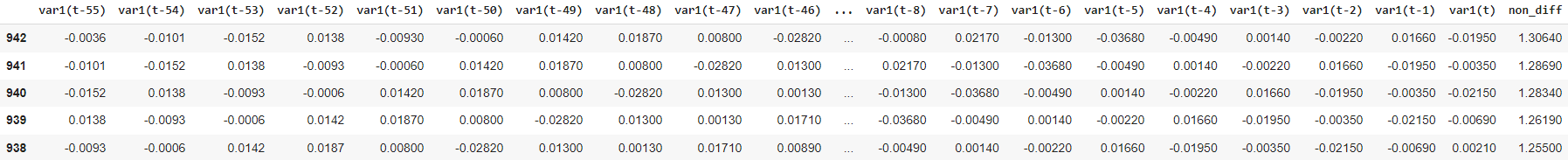
Sau khi có bộ dữ liệu, tôi cần phải xác định được trường dữ liệu đầu vào và đầu ra cho mô hình. Như đã biết, chúng tôi có hai cột cần quan tâm là “Open” và “Close”. Tuy nhiên, tôi lại chỉ đang có đầu ra là hai cột này và chúng tôi cần phải có đầu vào thì mô hình mới có thể dự đoán được. Như vậy, chúng tôi cần có một phương pháp để có thể tạo ra bộ dữ liệu đầu vào mà cụ thể là sẽ lấy lag của hai cột ban đầu. Ở trong bài này, chúng tôi sẽ lấy là 55 tuần trước để dự đoán cho tuần sau. Như đã nói, phương pháp này có thể tạo ra cho chúng tôi bộ dữ liệu đầu vào nhưng chúng tôi cũng cần lưu ý xóa những dòng chứa giá trị rỗng sau khi lấy lag.

Như vậy, bộ dữ liệu mà tôi có được sau khi áp dụng kĩ thuật lấy lag này sẽ là một ma trận 3 chiều với kích thước (944, 56, 2). Trong đó, chiều đầu tiên cho tôi thấy bộ dữ liệu gồm 944 mẫu (chiều thứ nhất là hàng). Chiều thứ hai biểu thị 55 lag và cột còn lại là lag 0 hay là giá trị đầu ra cần dự đoán (chiều thứ hai là cột). Chiều thứ ba là chiều biểu thị cho bảng dữ liệu huấn luyện để dự đoán giá mở cửa và bảng dữ liệu huấn luyện để dự đoán giá đóng cửa (chiều thứ ba là chiều phân biệt bộ dữ liệu của giá mở cửa và giá đóng cửa).

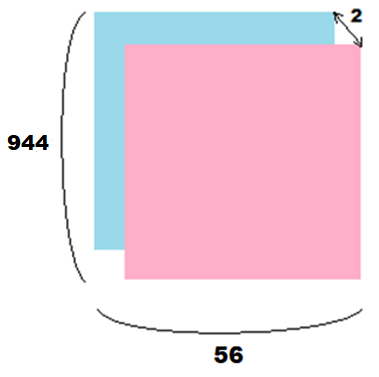
5 dòng đầu bảng dữ liệu huấn luyện để dự đoán giá mở cửa:



5 dòng đầu bảng dữ liệu huấn luyện để dự đoán giá đóng cửa:

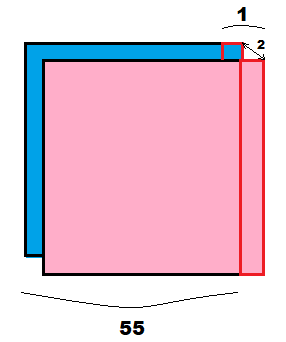


Hai bảng này xếp trồng lên nhau và tôi có chiều thứ ba:

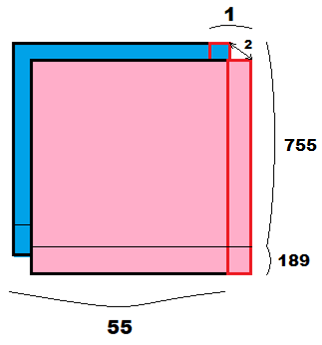


1.4. Chia tập dữ liệu

Khi đã có bộ dữ liệu bằng cách lấy lag, tôi cần xác định đầu vào, đầu ra của mô hình. Theo cấu trúc bộ dữ liệu mà chúng tôi đã tạo, chúng tôi sẽ lấy 55 cột đầu tiên làm đầu vào, cột cuối cùng sẽ là cột chứa thông tin đầu ra của mô hình.



Sau khi có bảng dữ liệu đầu vào và đầu ra, tôi chia nó thành tập huấn luyện và tập test. Trong đó tập huấn luyện chiếm 0.8 còn test là 0.2 và giữ nguyên thứ tự của chuỗi. Như vậy tập huấn luyện sẽ có 755 mẫu và tập test sẽ gồm có 189 mẫu.



2. Xây dựng mô hình

Mô hình chúng tôi sẽ đi xây dựng ở bài này sẽ vừa có thể dự đoán giá đóng và giá mở của sàn chứng khoán. Vì vậy, một mô hình gồm một đầu vào và hai đầu ra sẽ phù hợp cho bài toán này. Sau khi xây dựng cấu trúc của mạng, việc tiếp theo tôi cần tối ưu hóa siêu tham số. Thuật toán được dừng trong bài này sẽ là thuật toán Hyperband để tối ưu hóa siêu tham số. Và để sử dụng được hàm tối ưu siêu tham số cũng như xây dựng và huấn luyện mô hình, chúng tôi cần có các thư viện cần thiết là tensorflow và keras\_tuner. Vậy việc đầu tiên tôi cần làm là import thư viện:

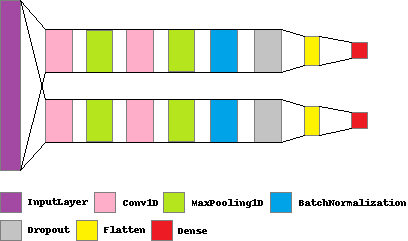
import keras\_tuner as kt

import tensorflow as tf

Trong đó, tensorflow là thư viện được sử dụng để xây dựng các mô hình học máy. Còn keras\_tuner, thư viện này sẽ cung cấp các thuật toán tối ưu hóa siêu tham số.

2.1. CNN

Việc đầu tiên, chúng ta cần xác định cấu trúc mạng khi xây dựng mô hình. Đối với mô hình CNN cho bài toán này, tôi xác định một cấu trúc mạng mà tôi muốn sẽ như sau:



Bây giờ, chúng tôi đã có cấu trúc của mạng và bây giờ tôi sẽ đi tối ưu hóa siêu tham số của mạng để có một mạng tốt nhất. Lớp đầu vào và lớp đầu ra sẽ không có siêu tham số nên chúng tôi sẽ tập chung vào những lớp ẩn của mạng.

Với các lớp Conv1D, chúng tôi có các siêu tham số như: số filter, kích thước kernel, hàm kích hoạt,… . Ở mô hình hiện tại, chúng tôi sẽ cùng đặt kích thước filter sẽ là 1x2, hàm kích hoạt sẽ là Relu. Siêu tham số chúng tôi muốn tối ưu hóa ở đây sẽ là số filter của lớp Conv1D đó và các siêu tham số khác sẽ dùng mặc định theo như thư viện tensorflow. Cụ thể:

tf.keras.layers.Conv1D(X, kernel\_size = 2, activation='relu')

Với các lớp MaxPooling1D, BatchNormalization, Flatten, chúng tôi sẽ dùng dùng mặc định theo như thư viện tensorflow. Cụ thể:

tf.keras.layers.MaxPooling1D()

tf.keras.layers.BatchNormalization()

tf.keras.layers.Flatten()

Còn đối với lớp Dropout, chúng tôi sẽ đặt tỉ lệ là 0.5. Cụ thể:

tf.keras.layers.Dropout(0.5)

Mô hình CNN của chúng tôi sẽ sử dụng hàm tối ưu hóa trọng số Adam và các siêu tham số của nó sẽ như sau:

Adam(learning\_rate = 0.0001 ,beta\_1 = 0.9,beta\_2 = 0.999,epsilon = 1e-7)

Về hàm loss, chúng tôi sẽ dùng mse (mean square error) để phục vụ việc tối ưu hóa trọng số trong mạng.

Như vậy, tôi đã có thông tin về các siêu tham số mà chúng tôi muốn tối ưu hóa trước khi đưa mô hình vào huấn luyện. Vì thế, chúng tôi sẽ định nghĩa một hàm sử dụng tuner để tối ưu hóa các siêu tham số:

ef get\_model(hp):

  X1 = hp.Int("filters1\_op", min\_value = 32, max\_value = 512, step = 32)

  X2 = hp.Int("filters2\_op", min\_value = 32, max\_value = 512, step = 32)

  X3 = hp.Int("filters1\_cl", min\_value = 32, max\_value = 512, step = 32)

  X4 = hp.Int("filters2\_cl", min\_value = 32, max\_value = 512, step = 32)

  input = tf.keras.layers.Input(shape = (55, 2))

  open = tf.keras.layers.Conv1D(X1,kernel\_size=2,activation='relu')(input)

  open = tf.keras.layers.MaxPooling1D()(open)

  open = tf.keras.layers.Conv1D(X2,kernel\_size=2,activation='relu')(open)

  open = tf.keras.layers.MaxPooling1D()(open)

  open = tf.keras.layers.BatchNormalization()(open)

  open = tf.keras.layers.Dropout(0.5)(open)

  open = tf.keras.layers.Flatten()(open)

  close =tf.keras.layers.Conv1D(X3,kernel\_size=2,activation='relu')(input)

  close = tf.keras.layers.MaxPooling1D()(close)

  close =tf.keras.layers.Conv1D(X4,kernel\_size=2,activation='relu')(close)

  close = tf.keras.layers.MaxPooling1D()(close)

  close = tf.keras.layers.BatchNormalization()(close)

  close = tf.keras.layers.Dropout(0.5)(close)

  close = tf.keras.layers.Flatten()(close)

  open = tf.keras.layers.Dense(1)(open)

  close = tf.keras.layers.Dense(1)(close)

  model = tf.keras.Model(inputs=input, outputs=[open, close])

  optimizer = tf.keras.optimizers.Adam(learning\_rate=0.0001,beta\_1=0.9,beta\_2 = 0.999, epsilon=1e-7)

  model.compile(optimizer=optimizer, loss='mse', metrics=['mse'])

  return model

Bước tiếp theo, chúng tôi sẽ chạy thuật toán Hyperband để tìm ra mô hình có siêu tham số tối ưu:

tuner = kt.Hyperband(get\_model,objective='val\_loss',directory='/content', project\_name = 'final')

stop\_early=tf.keras.callbacks.EarlyStopping(monitor='val\_loss',patience=5)

tuner.search(X\_train,y\_train,validation\_data=(X\_test,y\_test),callbacks=[stop\_early])

Sau khi chạy xong thuật toán, kết quả cuối cùng chúng tôi nhận được khi chạy thuật toán sẽ như sau:

Trial 254 Complete [00h 00m 09s]

val\_loss: 0.019938407465815544

Best val\_loss So Far: 0.0013773699756711721

Total elapsed time: 00h 29m 26s

filters1\_op: 192

filters2\_op: 480

filters1\_cl: 64

filters2\_cl: 448

Nhìn vào kết quả tối ưu mà ta thu được, ta có thể thấy hai lớp Conv1D của nhánh open sẽ có số filter lần lượt là 480 và 352, hai lớp Conv1D của nhánh close sẽ có số filter lần lượt là 320 và 512. Bây giờ ta đã có được kết quả tối ưu của các siêu tham số, chúng ta sẽ lấy mô hình chứa các thông số này để thực hiện huấn luyện:

model = tuner.get\_best\_models()[0]

model.build(X\_train.shape)

model.summary()

Đến bước này, ta đã xây dựng được mô hình CNN để sẵn sàng huấn luyện mô hình cho dự đoán giá chứng khoán. Chúng ta hãy cùng xem lại chi tiết các thông số của mạng CNN.

Model: "model"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param # Connected to

==================================================================================================

input\_1 (InputLayer) [(None, 55, 2)] 0 []

conv1d (Conv1D) (None, 54, 192) 960 ['input\_1[0][0]']

conv1d\_2 (Conv1D) (None, 54, 64) 320 ['input\_1[0][0]']

max\_pooling1d (MaxPooling1D) (None, 27, 192) 0 ['conv1d[0][0]']

max\_pooling1d\_2 (MaxPooling1D) (None, 27, 64) 0 ['conv1d\_2[0][0]']

conv1d\_1 (Conv1D) (None, 26, 480) 184800 ['max\_pooling1d[0][0]']

conv1d\_3 (Conv1D) (None, 26, 448) 57792 ['max\_pooling1d\_2[0][0]']

max\_pooling1d\_1 (MaxPooling1D) (None, 13, 480) 0 ['conv1d\_1[0][0]']

max\_pooling1d\_3 (MaxPooling1D) (None, 13, 448) 0 ['conv1d\_3[0][0]']

batch\_normalization (BatchNorm (None, 13, 480) 1920 ['max\_pooling1d\_1[0][0]']

alization)

batch\_normalization\_1 (BatchNo (None, 13, 448) 1792 ['max\_pooling1d\_3[0][0]']

rmalization)

dropout (Dropout) (None, 13, 480) 0 ['batch\_normalization[0][0]']

dropout\_1 (Dropout) (None, 13, 448) 0 ['batch\_normalization\_1[0][0]']

flatten (Flatten) (None, 6240) 0 ['dropout[0][0]']

flatten\_1 (Flatten) (None, 5824) 0 ['dropout\_1[0][0]']

dense (Dense) (None, 1) 6241 ['flatten[0][0]']

dense\_1 (Dense) (None, 1) 5825 ['flatten\_1[0][0]']

==================================================================================================

Total params: 259,650

Trainable params: 257,794

Non-trainable params: 1,856

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Để dễ dàng hình dung hơn, chúng ta sẽ trực quan hóa cấu trúc mạng bằng một biểu đồ dễ hiểu hơn.

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình, Song song

Mô tả được tạo tự động

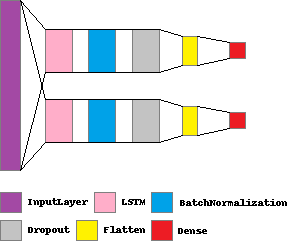
Total params: 259,650

Trainable params: 257,794

Non-trainable params: 1,856

2.2. RNN

Giống như CNN, việc đầu tiên chúng tôi cần làm là xây dựng cấu trúc mạng chúng tôi mong muốn. Và ở bài này, tôi cũng sẽ định hình cấu trúc mạng RNN của tôi sẽ như sau:



Giống như xây dựng mô hình CNN, bước tiếp theo chúng tôi cần làm là tối ưu siêu tham số của mô hình để có mô hình tốt nhất. Lớp đầu vào và lớp đầu ra sẽ không có siêu tham số nên chúng tôi sẽ tập chung vào những lớp ẩn của mạng.

Với hai lớp LSTM, chúng tôi có các siêu tham số như: số units, hàm kích hoạt,… . Với mô hình này, chúng tôi sẽ đặt hàm kích hoạt là hàm tanh. Trong khi đó, số units sẽ là siêu tham số cần được tối ưu hóa. Cụ thể:

tf.keras.layers.LSTM(units=X, return\_sequences=True, activation='tanh')

Với các lớp BatchNormalization, Flatten, chúng tôi sẽ dùng dùng mặc định theo như thư viện tensorflow. Cụ thể:

tf.keras.layers.BatchNormalization()

tf.keras.layers.Flatten()

Còn đối với lớp Dropout, tôi sẽ đặt tỉ lệ là 0.5. Cụ thể:

tf.keras.layers.Dropout(0.5)

Mô hình CNN của chúng tôi sẽ sử dụng hàm tối ưu hóa trọng số Adam và các siêu tham số của nó sẽ như sau:

Adam(learning\_rate = 0.0001 ,beta\_1 = 0.9,beta\_2 = 0.999,epsilon = 1e-7)

Về hàm loss, chúng tôisẽ dùng mse (mean square error) để phục vụ việc tối ưu hóa trọng số trong mạng.

Như vậy, tôi đã có thông tin về các siêu tham số mà chúng tôi muốn tối ưu hóa trước khi đưa mô hình vào huấn luyện. Vì thế, chúng tôi sẽ định nghĩa một hàm sử dụng tuner để tối ưu hóa các siêu tham số:

def get\_model(hp):

  X1 = hp.Int("units1\_op", min\_value = 32, max\_value = 512, step = 32)

  X2 = hp.Int("units1\_cl", min\_value = 32, max\_value = 512, step = 32)

  input = tf.keras.layers.Input(shape = (55, 2))

  open = tf.keras.layers.LSTM(units=X1, return\_sequences=True, activation='tanh')(input)

  open = tf.keras.layers.BatchNormalization()(open)

  open = tf.keras.layers.Dropout(0.5)(open)

  open = tf.keras.layers.Flatten()(open)

  close = tf.keras.layers.LSTM(units=X2, return\_sequences=True, activation='tanh')(input)

  close = tf.keras.layers.BatchNormalization()(close)

  close = tf.keras.layers.Dropout(0.5)(close)

  close = tf.keras.layers.Flatten()(close)

  open = tf.keras.layers.Dense(1)(open)

  close = tf.keras.layers.Dense(1)(close)

  model = tf.keras.Model(inputs=input, outputs=[open, close])

  optimizer = tf.keras.optimizers.Adam(learning\_rate = 0.0001, beta\_1 = 0.9, beta\_2 = 0.999, epsilon=1e-7)

  model.compile(optimizer=optimizer, loss='mse', metrics=['mse'])

  return model

Bước tiếp theo, chúng tôi sẽ chạy thuật toán Hyperband để tìm ra mô hình có siêu tham số tối ưu:

tuner = kt.Hyperband(get\_model,objective='val\_loss',directory='/content', project\_name = 'final')

stop\_early=tf.keras.callbacks.EarlyStopping(monitor='val\_loss',patience=5)

tuner.search(X\_train,y\_train,validation\_data=(X\_test,y\_test),callbacks=[stop\_early])

Sau khi chạy xong thuật toán, kết quả cuối cùng chúng ta nhận được khi chạy thuật toán sẽ như sau:

Trial 254 Complete [00h 00m 39s]

val\_loss: 0.0009988980600610375

Best val\_loss So Far: 0.0005213012918829918

Total elapsed time: 00h 51m 41s

units1\_op: 352

units1\_cl: 480

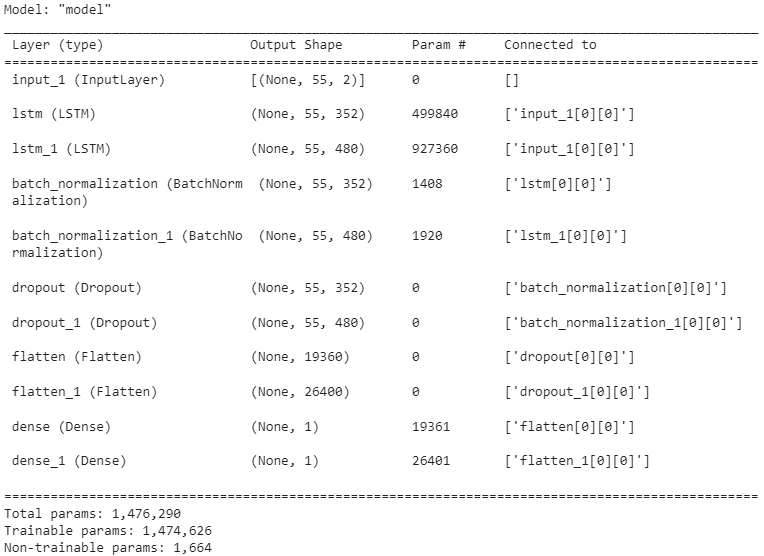
Nhìn vào kết quả tối ưu mà ta thu được, ta có thể thấy hai lớp LSTM của hai nhánh đều có số units giống nhau là 448. Bây giờ ta đã có được kết quả tối ưu của các siêu tham số, chúng ta sẽ lấy mô hình chứa các thông số này để thực hiện huấn luyện:

model = tuner.get\_best\_models()[0]

model.build(X\_train.shape)

model.summary()

Như vậy, chúng ta đã có mô hình có cấu trúc mạng RNN để dự đoán thị trường chứng khoán phù hợp. Chúng ta sẽ cùng xem các thông số của mạng RNN.



Để dễ dàng hình dung hơn, chúng ta sẽ trực quan hóa cấu trúc mạng bằng một biểu đồ dễ hiểu hơn.

Ảnh có chứa văn bản, biểu đồ, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động

Total params: 1,476,290

Trainable params: 1,474,626

Non-trainable params: 1,664

3. Kết quả dự đoán trên tập test

3.1. CNN

* Open



* Close

Ảnh có chứa hàng, Sơ đồ, biểu đồ, văn bản

Mô tả được tạo tự động

3.2. RNN

* Open

Ảnh có chứa hàng, Sơ đồ, biểu đồ, văn bản

Mô tả được tạo tự động

* Close

Ảnh có chứa hàng, Sơ đồ, biểu đồ, văn bản

Mô tả được tạo tự động

4. Đánh giá mô hình trên tập test

Phương pháp đánh giá sử dụng cho hai mô hình trong chủ đề này sẽ là phương pháp đánh giá mô hình hồi quy là RMSE (Root Mean Square Error), MAE (Mean Absolute Error), R và NSE (Nash-Sutcliffe Efficiency).

4.1. Đánh giá mô hình CNN

* Open

RMSE: 0.0044246755413716735

MAE: 0.003397018351681017

R: 1.003108727417564

NSE: 0.9952000362511806

* Close

RMSE: 0.01318037110233416

MAE: 0.009669274806168974

R: 0.9837356147564544

NSE: 0.9573382649981259

Nhìn vào các thông số đánh giá với mô hình CNN, chúng thể hiện cho ta thấy mô hình đang khá tốt khi dự báo giá chứng khoán trên tập test.

Về giá mở cửa, chúng ta nhận được giá trị RMSE là 4.4e-3 và MAE là 3.3e-3. Hai thông số này cho ta thấy giá trị dự đoán mà mô hình dự đoán đang bám khá sát với giá trị thực tế hay độ chênh lệch giữ giá trị dự đoán và thực tế rất nhỏ. Trong khi đó, thông số R là 1 chứng tỏ các giá trị dự đoán và giá trị thực tế đang có quan hệ mạnh. Bên cạnh đó, thông số NSE ở ngưỡng 0.99 cũng cho ta thấy mô hình rất phù hợp với tập dữ liệu và có thể dự đoán tốt giá mở cửa.

Tương tự với giá đóng cửa, thông số RMSE là 1.3e-2 và MAE là 9.6e-3 cũng cho ta thấy giá trị dự đoán cũng đang bám khá sát với giá trị thực tế. Thông số của R là 0.98 cho thấy mối quan hệ của giạ trị dự đoán và thực tế khá mạnh và NSE ở ngưỡng 0.95 cũng cho ta thấy mô hình khá phù hợp với tập dữ liệu và có thể dự đoán giá đóng cửa ở mức khá.

Chúng ta có thể thấy rõ, mô hình CNN đang dự báo giá mở cửa tốt hơn giá đóng cửa. Như vậy, ta có thể sẽ cần nhiều thông tin hơn để mô hình có thể dự đoán được chính xác hơn giá đóng cửa.

4.2. Đánh giá mô hình RNN

Tương tự như CNN, RNN cũng sẽ được đánh giá qua những thông số được đề cập ở trên.

* Open

RMSE: 0.002255032724973566

MAE: 0.0014038952973784544

R: 1.00470458958253

NSE: 0.9987532459273574

* Close

RMSE: 0.013477002294628311

MAE: 0.010024247951507565

R: 0.9828818555497802

NSE: 0.955396407260834

Khi nhìn vào các thông số, chúng ta cũng thấy rằng mô hình sử dụng RNN cũng đang dự đoán khá tốt về giá thị trường chứng khoán.

Với giá mở cửa, thông số RMSE và MAE cũng cho ta thấy khả năng dự đoán của mô hình cũng đang khá tốt với các dữ liệu trên tập test với giá trị lần lượt là 2.2e-3 và 1.4e-3. Trong khi đó, thông số R là 1 cũng đang cho thấy mối quan hệ rất mạnh giữa các giá trị dự đoán và các giá trị thực tế. Thông số NSE là 0.99 cũng cho thấy độ phù hợp của mô hình với tập dữ liệu và khả năng dự đoán rất tốt.

Bên dự đoán giá đóng cửa cũng không hề kém cạnh khi các thông số RMSE và MAE lần lượt là 1.3e-2 và 1e-2 cho thấy mô hình dự đoán cũng khá sát với thực tế. Bên cạnh đó với R là 0.98 cũng cho thấy quan hệ mạnh giữa các giá trị dự đoán và thực tế và NSE ở ngưỡng 0.95 cũng cho thấy mô hình khá phù hợp và có khả năng dự đoán tốt.

Giống như mô hình CNN, mô hình RNN đang dự báo giá mở cửa tốt hơn giá đóng cửa. Như vậy, ta có thể sẽ cần nhiều thông tin hơn để mô hình có thể dự đoán được chính xác hơn giá đóng cửa.

4.3. Nhận xét

Nhìn chung, hai mô hình đều đang có các thông số đánh giá khá tốt trên tập test và cả hai đều đang làm khá tốt nhiệm vụ dự đoán giá mở cửa và giá đóng cửa trong bộ dữ liệu này. Tuy nhiên, so sánh các thông số đánh giá của hai mạng, ta thấy độ chính xác của hai mạng có chút khác biệt.

Đối với giá mở cửa, mô hình RNN đang làm tốt hơn CNN một chút. Trong đó, thông số đánh giá RMSE và MAE của mô hình CNN lớn hơn so với RNN, điều này chứng tỏ các giá trị dự đoán về tổng quan của RNN trên tập test đang bám sát với giá thực tế hơn CNN. Hơn nữa, thông số đánh giá R và NSE của RNN đang tốt hơn so với CNN. Chúng cho thấy các giá trị dự đoán so với thực tế của RNN đang làm tốt hơn CNN và mô hình RNN phù hợp để dự đoán giá mở cửa hơn là mô hình CNN.

Mặt khác, với giá đóng cửa, các thông số đánh giá cho thấy CNN lại đang làm tốt hơn RNN. Các thông số như RMSE và MAE của mô hình CNN đang tốt hơn so với RNN một chút, điều này cho ta biết rằng các giá trị dự đoán về tổng quan của CNN trên tập test đang bám sát với giá thực tế hơn là RNN. Bên cạnh đó, thông số đánh giá R và NSE của CNN cũng đang tốt hơn so với RNN trong việc dự đoán. Các thông số này cho thấy các giá trị dự đoán của CNN đang có mối quan hệ với các giá trị thực tế mạnh hơn RNN và mô hình CNN phù hợp để dự đoán giá đóng của hơn là mô hình RNN.

Tuy nhiên, khi nhìn vào biểu đồ “Open – Close” của cả hai mô hình, ta thấy mô hình RNN đang dự đoán song song cả hai loại giá.

CNN

Ảnh có chứa Sơ đồ, hàng, văn bản, biểu đồ

Mô tả được tạo tự động

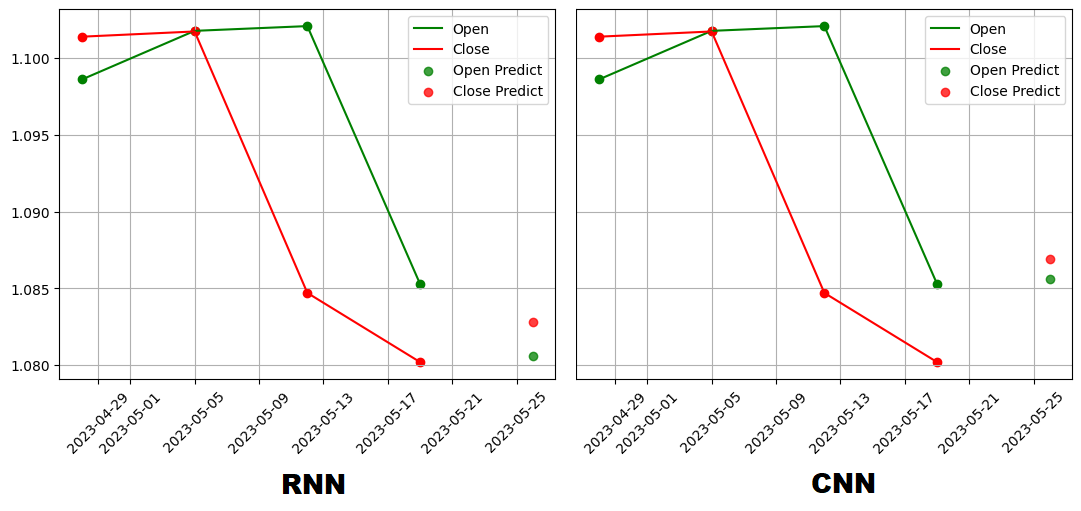
RNN

Ảnh có chứa Sơ đồ, hàng, văn bản, ảnh chụp màn hình

Mô tả được tạo tự động

Biểu đồ của mô hình RNN cho ta thấy hiệu giữa giá đóng cửa và giá mở cửa của các giá trị dự đoán đang chi tiết sự biến động và chính xác hơn khi so với biểu đồ của CNN. Trong khi đó, biểu đồ CNN lại cho ta thấy trong nhiều giai đoạn việc dự đoán song song của nó không hiệu quả và không bắt được chi tiết sự biến động.

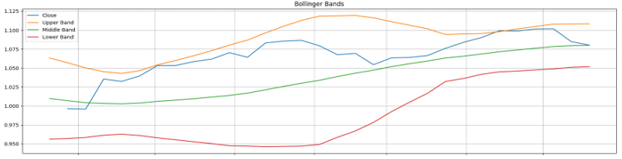
5. Kết quả dự đoán cho tuần tới



Nhìn vào biểu đồ có thể thấy rằng, giá mở của tuần tới đang được dự báo là sẽ tăng trở lại sau đợt giảm giá trị vào tuần sau. Cả hai mô hình đều đang dự báo về giá đóng của tuần tuấn sẽ lớn hơn một chút so với giá mở cửa và dự báo về một cây nến dương trong biểu đồ nến thường dùng trong thị trường chứng khoán. Nhìn vào hai biểu đồ, ta thấy hai mô hình dự đoán về giá đóng mở khá gần nhau và hai giá mở cửa được dự đoán hơi lệch nhau một chút.

V. Diễn giải kết quả thu được

Như đã biết, kết quả của dự đoán của cả hai mô hình về giá đóng cửa tuần sau đều có dấu hiệu tăng nhẹ. Kết hợp với biểu đồ bollinger band và RSI trong 30 ngày gần nhất.



Ảnh có chứa hàng, Sơ đồ, biểu đồ, sườn dốc

Mô tả được tạo tự động

Ta thấy dải bollinger band đang bị bó hẹp. Điều này báo hiệu trong giai đoạn này biến động của giá thị trường có các biến động không đáng kể. Nhưng điều này cũng là một dấu hiệu cho thấy sắp xảy ra một sự biến động giá đột biến. Còn khi nhìn vào biểu đồ RSI, ta thấy giá trị của nó đang ổn định trong khoảng 60. Nó có nghĩa rằng giá thị tường sau đó sẽ có xu hướng tăng và rất có thể sẽ có sự tăng trưởng mạnh mẽ trong thời gian sắp tới.

Ta có thể thấy rằng, dự đoán của hai mô hình CNN và RNN đang khá đúng với xu hướng của thị trường. Như vậy, mô hình RNN và CNN có thể áp dụng tốt trong những bài toán kinh tế và chúng có thể dự đoán tốt được giá của thị trường. Tuy nhiên, mô hình học máy chỉ nên là công cụ tham khảo, bạn cần phải tìm hiểu thật kĩ từ nhiều nguồn, nhiều dấu hiệu khác nhau trước khi quyết định đầu tư vào một thị trường nào đó. Giá thị trường có thể sẽ biến động khác thường mà không theo một quy luật cụ thể và mô hình lúc này sẽ không có đủ khả năng để nhìn ra và dự đoán chính xác được.

Link code:

Khai phá và trực quan hóa:

* <https://colab.research.google.com/drive/1JvSk2scpsMfIZnf0yHAb9_UEtz9K5pjj?usp=sharing>
* <https://colab.research.google.com/drive/19IkimnvEQrcSfi2HAYCf4V9ikf8ZSi_E?usp=sharing>

Xây dựng mô hình:

* CNN: https://colab.research.google.com/drive/1ekn6LorRkJoHb7CJZUCDPHxfivAxhA6J?usp=sharing
* RNN: https://colab.research.google.com/drive/1hHC0nf8FqOI3yDodKVbYJ\_OGtpiPbwvn?usp=sharing