ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
ĐẠI HỌC BÁCH KHOA
========== o ==========
KHOA KHOA HỌC VÀ KĨ THUẬT MÁY TÍNH

Assignment Report

Assignment:

Stock Prediction

Giảng viên hướng dẫn: TS. Phan Trọng Nhân
Sinh viên thực hiện: Du Thành Đạt – 2010206
Nguyễn Phúc Đăng – 2012968
Lê Phúc Đức – 2370116

Thành phố Hồ Chí Minh, tháng 11 năm 2023

Bảng phân công công việc:

MSSV	Tên	Đánh giá nhiệm vụ được phân công
2010206	Du Thành Đạt	(100%)
2012968	Nguyễn Phúc Đăng	(100%)
2370116	Lê Phúc Đức	(100%)

Mục lục

1 Introduction

Trong thế kỷ 21, thị trường chứng khoán trở thành một trong những lĩnh vực quan trọng và phức tạp nhất của nền kinh tế toàn cầu. Đối mặt với sự biến động ngày càng tăng và sự ảnh hưởng của nhiều yếu tố không dự đoán được, việc dự đoán giá cổ phiếu trở thành một thách thức lớn đối với các nhà đầu tư và chuyên gia tài chính.

Đề tài này tập trung vào việc phát triển một hệ thống dự đoán giá cổ phiếu sử dụng kết hợp giữa chỉ số RSI (Relative Strength Index) và các mô hình học máy như LSTM (Long Short-Term Memory), SVM (Support Vector Machine), và Decision Tree. RSI là một chỉ số phổ biến trong phân tích kỹ thuật được sử dụng để đo lường sức mạnh tương đối của xu hướng giá và xác định khi một cổ phiếu đã quá mua hoặc quá bán.

Mô hình LSTM, với khả năng học từ chuỗi dữ liệu thời gian dài, có thể giúp nắm bắt được các mô hình phức tạp của biến động giá cổ phiếu. SVM, với khả năng làm việc tốt trong không gian đa chiều, và Decision Tree, với khả năng mô phỏng quyết định dựa trên dữ liệu đa biến, đều là những công cụ hữu ích trong việc tạo ra các mô hình dự đoán chính xác.

Bằng cách kết hợp sức mạnh của RSI và đa dạng hóa của các mô hình học máy, nghiên cứu này nhằm đạt được một hệ thống dự đoán giá cổ phiếu độ chính xác cao, linh hoạt và có thể áp dụng trong nhiều điều kiện thị trường khác nhau. Kết quả của đề tài có thể cung cấp thông tin hữu ích cho các nhà đầu tư và quản lý rủi ro, giúp họ đưa ra quyết định đầu tư thông minh và hiệu quả.

2 Background

2.1 Stock data features

Dữ liệu về giá cổ phiếu thường bao gồm nhiều đặc trưng hoặc thông tin khác nhau để mô tả diễn biến của cổ phiếu trong một khoảng thời gian nhất định. Dưới đây là một số đặc trưng thông thường trong dữ liệu về giá cổ phiếu mà do VN30 cung cấp như sau:

- Ngày: Gồm ngày tháng năm cụ thể.
- Lần cuối: Hay còn gọi là "Giá đóng"(close price), có nghĩa là giá cổ phiếu của phiên giao dịch cuối của ngày đó.
- Mở: Hay còn gọi là "Giá mở"(open price), có nghĩa là giá cổ phiếu của phiên giao dịch đầu của ngày đó.
- Cao: Hay còn gọi là "Giá trần", có nghĩa là giá cổ phiếu cao nhất trong ngày.
- Thấp: Hay còn gọi là "Giá sàn", có nghĩa là giá cổ phiếu thấp nhất trong ngày.
- KL: Số lượng giao dịch trong ngày của cổ phiếu.
- Thay đổi: Phần trăm thay đổi của giá đóng của ngày hôm sau so với ngày hiện tại.

Trong đó, có những trường dữ liệu mà chúng ta có thể suy ra, bao gồm:

- **RSI** :
  - Đây là "Chỉ số sức mạnh tương đối"(Relevant Strength Index) là giá trị phản ánh tình trạng quá "mua" hoặc quá "bán" trên thị trường. Có công thức như sau:

$$RSI = 100 - [\frac{100}{1 + \frac{\text{Mức tăng trung bình}}{\text{Tổn thất trung bình}}}]$$

  - Giá trị của RSI được biểu diễn trong khoảng từ 0 đến 100. Thông thường:
    - Nếu RSI $>$ 70, thì cổ phiếu đang bị mua quá mức dẫn đến việc xu hướng tăng của giá trị cổ phiếu có khả năng bị giảm và xu hướng giảm có khả năng tăng.
    - Nếu RSI $<$ 30, thì cổ phiếu đang bị bán quá mức dẫn đến giá trị cổ phiếu gần chạm đáy và chuẩn bị tăng.
    - Nếu RSI giữa khoảng (30,70) được coi là trung tính. Ví dụ với mức RSI=50 thì xem là không có xu hướng.
  - Bên cạnh việc thể hiện tình trạng quá bán hoặc quá mua, nhà đầu tư cũng có thể sử dụng phân kỳ RSI để dự đoán xu hướng đảo chiều. Sự phân kỳ xảy ra khi giá chứng khoán di chuyển theo hướng ngược lại so với các chỉ báo kỹ thuật. Điều này cảnh báo xu hướng giá hiện tại có thể đang suy yếu và nguy cơ dẫn đến sự thay đổi hướng của giá. Trong đó có hai loại là phân kỳ âm và phân kỳ dương:
    - Phân kỳ dương: RSI tăng tạo đáy cao trong khi giá giảm tạo đáy thấp, cảnh báo đà tăng mạnh bất chấp xu hướng giá giảm.
    - Phân kỳ âm: RSI giảm và tạo đỉnh thấp trong khi giá tài sản tăng tạo đỉnh cao hơn, cảnh

báo giá có thể giảm mạnh.

- **EMA**:
  - Đây là "Đường trung bình động luỹ thừa"(Exponent Moving Average). Nó được dùng để phản ánh sự biến động giá được tính theo cấp số nhân, qua đó giúp nhà đầu tư xác định được sự biến động giá một cách chuẩn xác hơn và giảm thiểu được tình trạng nhiễu giá so với đường trung bình động thông thường.
  - Có công thức:
$$\text{EMA}(t) = P(t) \times K + \text{EMA}(t-1) \times (1-K)$$
  - Với:
    - $P(t)$: giá đóng của cổ phiếu ở ngày t.
    - $K = \frac{2}{1+N}$: hệ số làm trơn, được tính bằng công thức đính kèm với N là chu kì (ví dụ như 5 ngày, 10 ngày,...).
    - EMA(t): giá trị EMA tại ngày t.
  - Sự di chuyển của đường EMA phản ánh xu hướng dịch chuyển của giá, thông qua đó nhà đầu tư nắm bắt tình hình xu hướng giá ngay thời điểm hiện tại.
  - Đường EMA dốc lên thể hiện giá đang đi lên, thị trường trong xu hướng tăng. Ngược lại, đường EMA dốc xuống thể hiện giá đang đi xuống, thị trường trong xu hướng giảm. Đường EMA nằm ngang đồng nghĩa với giá cũng đi ngang, thị trường trong trạng thái sideway.
  - EMA được theo 3 khung thời gian gồm có:
    - EMAF: EMA nhanh, thường tương ứng với các chu kỳ nhỏ.
    - EMAM: EMA trung bình, thường tương ứng với các chu kỳ trung bình.
    - EMAS: EMA chậm, thường tương ứng với chu kỳ lớn.
- **Volume-Weighted Average Price (VWAP)**:
  - Giá trung bình theo khối lượng (Volume-Weighted Average Price - VWAP) là một chỉ báo trong phân tích kỹ thuật và tài chính, thường được sử dụng trong giao dịch chứng khoán. VWAP tính toán giá trung bình của một tài sản tài chính (chẳng hạn như cổ phiếu) trong một khoảng thời gian cụ thể, chia cho tổng lượng giao dịch của tài sản đó trong khoảng thời gian đó.
  - Công thức tính VWAP là tổng của giá mua vào (hoặc giá bán ra) nhân với số lượng tương ứng, chia cho tổng lượng giao dịch trong khoảng thời gian đã chọn:
  - Chỉ báo VWAP góp phần xác định mức giá trung bình phù hợp nhất của các cặp tiền tệ giao dịch trong ngày đó. Thông qua VWAP, các nhà đầu tư có thể nhìn nhận sâu sắc hơn về xu hướng giá biến động như thế nào trên thị trường ở thời điểm hiện tại.
  - Công thức tính VWAP:
$$\text{VWAP} = \frac{\sum(\text{Giá} \times \text{Khối lượng})}{\sum \text{Khối lượng}}$$
  - VWAP thường được sử dụng để đánh giá giá trị trung bình mà các nhà đầu tư hoặc người

giao dịch đã mua hoặc bán một tài sản trong một khoảng thời gian nhất định. Nó cung cấp thông tin về giá trung bình mà giao dịch đã được thực hiện và có thể giúp xác định xem liệu một giao dịch cụ thể có được thực hiện với giá tốt hơn so với VWAP hay không.

- Người giao dịch thường sử dụng VWAP để so sánh giá giao dịch của họ với giá trung bình chung của thị trường trong cùng khoảng thời gian, từ đó đánh giá hiệu suất giao dịch của họ.

2.2 Prediction models

2.2.1 Linear Regression

Mô hình hồi quy tuyến tính là một trong những công cụ cơ bản và quan trọng nhất trong thống kê và học máy. Nó là một phương pháp mạnh mẽ giúp mô phỏng mối quan hệ tuyến tính giữa một biến phụ thuộc và một hoặc nhiều biến độc lập. Trong lĩnh vực phân tích dữ liệu, linear regression thường được sử dụng để hiểu và dự đoán biến động của một biến phụ thuộc dựa trên sự thay đổi của một hoặc nhiều biến độc lập.

Mô hình hồi quy tuyến tính có dạng toán học đơn giản, thường được biểu diễn dưới dạng:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n + \epsilon$$

Trong đó, Y là biến phụ thuộc, $X_1, X_2, ..., X_n$ là các biến độc lập, $\beta_0, \beta_1, \beta_2, ..., \beta_n$ là các hệ số hồi quy, và ϵ là sai số ngẫu nhiên. Mục tiêu của mô hình là điều chỉnh các hệ số này sao cho sai số ngẫu nhiên là nhỏ nhất, tạo ra một đường tuyến tính "tốt nhất" phản ánh mối quan hệ giữa biến phụ thuộc và các biến độc lập.

Linear regression không chỉ giúp ta hiểu mối quan hệ giữa các biến mà còn có thể được sử dụng để dự đoán giá trị của biến phụ thuộc khi có giá trị mới của các biến độc lập. Điều này làm cho nó trở thành một công cụ mạnh mẽ trong dự báo và quyết định dựa trên dữ liệu.

Ứng dụng của linear regression không chỉ xuất hiện trong thống kê, mà còn mở rộng sang nhiều lĩnh vực khác nhau như tài chính, kinh tế, y học, xã hội học, và nhiều lĩnh vực khác. Trong khi mô hình có những giới hạn của mình, như giả định về tuyến tính và điều kiện độc lập, sự đơn giản và tính hiệu quả của nó khiến cho linear regression vẫn là một công cụ quan trọng và không thể thiếu trong kho dữ liệu đa dạng và phức tạp ngày nay.

2.2.2 Support Vector Machine

Support Vector Machine (SVM) là một phương pháp máy học được sử dụng rộng rãi trong các bài toán phân loại và hồi quy. SVM đặc biệt hiệu quả trong việc xử lý dữ liệu không tuyến tính và có khả năng làm việc tốt trong không gian nhiều chiều.

SVM hoạt động bằng cách tìm ra ranh giới phân chia tốt nhất giữa các lớp dữ liệu. Trong bài toán phân loại nhị phân, SVM tìm ranh giới phân chia sao cho khoảng cách giữa các điểm dữ liệu gần nhất (Support Vectors) và ranh giới là lớn nhất. Ranh giới này được xác định bởi siêu phẳng (hyperplane) mà có thể được biểu diễn dưới dạng phương trình:

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$$

Trong đó, **w** là vector trọng số và b là độ dời (bias). Điều quan trọng là SVM cố gắng tối đa hóa khoảng cách giữa siêu phẳng và các điểm dữ liệu gần nhất. Hơn nữa, SVM có thể được mở rộng để xử lý dữ liệu không tuyến tính bằng cách sử dụng các hàm nhân (kernel functions) để ánh xạ dữ liệu vào không gian nhiều chiều.

Các biến trong phương trình:

- **x** là vector đặc trưng của điểm dữ liệu.
- **w** là vector trọng số của siêu phẳng.
- b là độ dời của siêu phẳng.

SVM là một công cụ mạnh mẽ cho việc phân loại dữ liệu và có thể được điều chỉnh linh hoạt thông qua việc chọn kernel phù hợp và các siêu tham số khác.

2.2.3 Support Vector Regression

Thuật toán Hồi quy Vector hỗ trợ (Support Vector Regression) là một loại máy học hồi quy hỗ trợ hồi quy tuyến tính và phi tuyến tính

Máy học vector hỗ trợ có thể được sử dụng cho bài toán hồi quy, giữ lại những đặc trưng chính để máy học tận dụng tối đa nhằm tạo ra vùng biên tối đa. Nhưng thuật toán SVR, dù chung một ý tưởng với thuật toán phân loại SVM, có một số đặc điểm khác biệt.

SVR cố gắng tìm ra hàm mục tiêu khớp với dữ liệu huấn luyện tốt nhất, nhưng đồng thời cũng cố giảm thiểu mất mát giữa giá trị đoán nhận và giá trị thực. Hàm mục tiêu ở đây được gọi là biên, và những điểm nằm gần nhất với nó được gọi là các vector hỗ trọ.

Ý tưởng then chốt của SVR chính là đặt ra lề cho phép mất mát ở xung quanh các các vector hỗ trợ. Nhờ đó, mô hình trở nên khái quát hơn với dữ liệu mới, giúp mô hình có thể chịu đựng được nhiễu hoặc sự biến thiên trong dữ liệu.

SVR có thể được sử dụng cho các bài toán hồi quy, như là dự đoán chứng khoán, giá nhà cửa,... SVR là công cụ mạnh mẽ cho hồi quy phi tuyến tính, vì thuật toán có thể sử dụng các hàm nhân (kernels) để biến đổi dữ liệu và chiếu dữ liệu lên các chiều cao hơn, do có thể tìm được các hàm mục tiêu phù hợp với dáng hình của dữ liệu

Đặc điểm quan trọng của SVR bao gồm:

Hàm mất mát (Loss function): SVR sử dụng hàm mất mát có mục tiêu là tối đa hoặc tối thiểu hóa sai số trong việc dự đoán. Hàm mất mát thường kết hợp việc đo khoảng cách giữa dự đoán và giá trị thực tế với việc xác định một ranh giới cho sai số được chấp nhận (epsilon).

Hàm kernel: Kernel functions trong SVR cho phép ánh xạ dữ liệu từ không gian chiều thấp lên không gian chiều cao hơn để tạo ra các đường biên phức tạp hơn, giúp mô hình mô tả được mối quan hệ phức tạp giữa biến độc lập và biến phụ thuộc.

Tham số điều chỉnh: SVR có các tham số như cost (điều chỉnh sự linh hoạt của đường biên) và epsilon (điều chỉnh khoảng cách chấp nhận được cho sai số). Việc điều chỉnh các tham số này có thể ảnh hưởng đến hiệu suất của mô hình.

2.2.4 Long Short Term Memory

Trong nhiều ứng dụng máy học, đặc biệt là trong xử lý dữ liệu chuỗi, việc duy trì và hiểu được mối quan hệ trong dữ liệu thời gian là quan trọng. Các mô hình truyền thống như mạng nơ-ron thuần (feedforward neural networks) thường không thể hiệu quả trong việc này do không giữ được thông tin liên tục qua thời gian.

LSTM là một dạng đặc biệt của mạng nơ-ron hồi quy (RNN - Recurrent Neural Network) được thiết kế để vượt qua vấn đề "mất mát thông tin" trong quá trình học dữ liệu thời gian.

2.2.4.1 Forget Gate

LSTM sử dụng cổng quên để quyết định xem thông tin nào nên được giữ lại và thông tin nào nên bị loại bỏ từ bước thời gian trước đó.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

$$C_t = f_t \odot C_{t-1}$$

2.2.4.2 Input Gate

Cổng đầu vào quyết định thông tin mới nào sẽ được thêm vào trạng thái ẩn.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

$$C_t = i_t \odot \tilde{C}_t$$

2.2.4.3 Output Gate

Cổng đầu ra quyết định thông tin nào sẽ được xuất ra từ trạng thái ẩn để làm đầu vào cho bước thời gian tiếp theo.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \odot \tanh(C_t)$$

Trong các công thức trên:

- $\mathbf{f}_t$, $\mathbf{i}_t$, $\mathbf{o}_t$ là các vectơ cổng quên, đầu vào, và đầu ra tương ứng.
- σ là hàm sigmoid, tanh là hàm tanh.

- $\mathbf{W}_f, \mathbf{W}_i, \mathbf{W}_o, \mathbf{W}_c$ là ma trận trọng số cho các cổng và trạng thái ẩn.
- $\mathbf{h}_{t-1}, \mathbf{x}_t$ là vector kết hợp giữa trạng thái ẩn từ bước thời gian trước đó $\mathbf{h}_{t-1}$ và đầu vào $\mathbf{x}_t$.
- $\mathbf{b}_f, \mathbf{b}_i, \mathbf{b}_o, \mathbf{b}_c$ là các vectơ độ lệch (bias).

Các biến:

- $\mathbf{C}_t$ là trạng thái ẩn (cell state) tại thời điểm t.
- $\tilde{\mathbf{C}}_t$ là thông tin mới được đề xuất cho trạng thái ẩn.
- $\mathbf{h}_t$ là trạng thái ẩn đầu ra tại thời điểm t.
- $\odot$ đại diện cho phép nhân element-wise (tích vô hướng).

3 Implementation

3.1 Data Pre-processing

Đầu tiên dữ liệu sẽ được lấy ở trang web: https://vn.investing.com/indices/vn-30-historical-data, bao gồm dữ liệu theo từng ngày của cổ phiếu tổng hợp VN30 sẽ được lưu dưới dạng CSV như sau:

```
"Date","Close","Open","High","Low","Vol","Percentage Change"
"03/10/2023","1,130.89","1,156.86","1,158.38","1,128.54","260.26K","-3.11%"
"02/10/2023","1,167.13","1,166.64","1,173.06","1,159.16","119.87K","0.07%"
"29/09/2023","1,166.26","1,169.26","1,176.77","1,165.15","144.90K","0.16%"
"28/09/2023","1,164.45","1,167.96","1,168.09","1,144.47","193.65K","-0.36%"
"27/09/2023","1,168.60","1,154.54","1,168.60","1,141.09","190.47K","1.32%"
```

Các bước xử lý dữ liệu:

- Sau khi đã tải dữ liệu xuống, tiến hành load vào jupyter notebook để phân tích, kết quả được hiển thị ở hình **3.1**

```
data = pd.read_csv('./VN30_031023-031013.csv')
data.head(5)
```

	Date	Close	Open	High	Low	Vol	Percentage Change
0	03/10/2023	1,130.89	1,156.86	1,158.38	1,128.54	260.26K	-3.11%
1	02/10/2023	1,167.13	1,166.64	1,173.06	1,159.16	119.87K	0.07%
2	29/09/2023	1,166.26	1,169.26	1,176.77	1,165.15	144.90K	0.16%
3	28/09/2023	1,164.45	1,167.96	1,168.09	1,144.47	193.65K	-0.36%
4	27/09/2023	1,168.60	1,154.54	1,168.60	1,141.09	190.47K	1.32%

Hình **3.1**: Dữ liệu được load vào Jupyter Notebook dưới dạng dataframe.

- Kiểm tra dữ liệu, ta thấy không có cột nào bị thiếu dữ liệu, kết quả được biểu diễn ở hình **3.2**.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2496 entries, 0 to 2495
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Date               2496 non-null   object
 1   Close              2496 non-null   object
 2   Open               2496 non-null   object
 3   High               2496 non-null   object
 4   Low                2496 non-null   object
 5   Vol                2496 non-null   object
 6   Percentage Change  2496 non-null   object
dtypes: object(7)
memory usage: 136.6+ KB
```

Hình **3.2**: Kiểm tra data.info()

- Các feature như Close, Open, High, Low có cùng cách biểu diễn, nhưng với Vol thì được biểu diễn theo đơn vị là K, vậy còn những đơn vị khác không, ta kiểm tra bằng cách kiểm tra ký tự cuối của mỗi phần tử trong dataframe Vol, kết quả thu được gồm 2 đơn vị là K và M(kết quả biểu diễn ở hình **3.4**).

```
Vol
K    2491
M       5
Name: count, dtype: int64
```

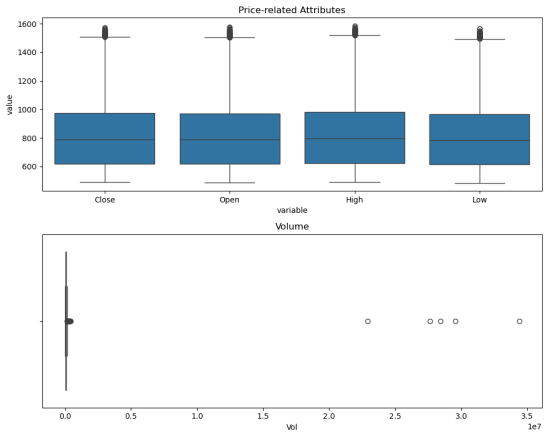
Hình **3.3**: Kết quả các đơn vị của Vol.

- Tiếp tục xử lý các dữ liệu kiểu chuỗi từ các feature như Close, Open, High, Low bằng cách loại bỏ dấu phẩy mỗi 3 chữ số và chuyển thành kiểu dữ liệu số thực. Đối với Vol thì ta sẽ nhân 1000 đối với những hàng có đơn vị là K và nhân 1000000 đối với những hàng có đơn vị là M.
- Đồng thời cũng xoá dấu "%" ở cột Percentage Change và đổi kiểu dữ liệu của cột này thành kiểu số thực.

	Date	Close	Open	High	Low	Vol	Percentage Change
0	2013-10-03	549.58	552.41	552.66	548.44	19120	-0.57
1	2013-10-04	552.07	549.74	552.27	548.63	17850	0.45
2	2013-10-07	557.54	552.55	557.95	552.30	22800	0.99
3	2013-10-08	558.55	557.83	560.36	555.02	29150	0.18
4	2013-10-09	556.72	558.24	558.99	555.25	17700	-0.33

Hình **3.4**: Kết quả data sau khi đã thực hiện việc chuyển đổi.

- Sau khi đã hoàn thành việc xử lý dữ liệu thô cơ bản, ta xem xét boxplot của từng feature trong dữ liệu(biểu diễn hình **3.5**). Ta thấy các giá phân bố khá tốt, tuy nhiên về khối lượng giao dịch của một vài ngày lên đến hàng triệu giao dịch(cụ thể là 5 ngày), đều này làm biểu đồ boxplot của khối lượng giao dịch biểu diễn khá tệ.



Hình **3.5**: Boxplot của từng feature trong dữ liệu.

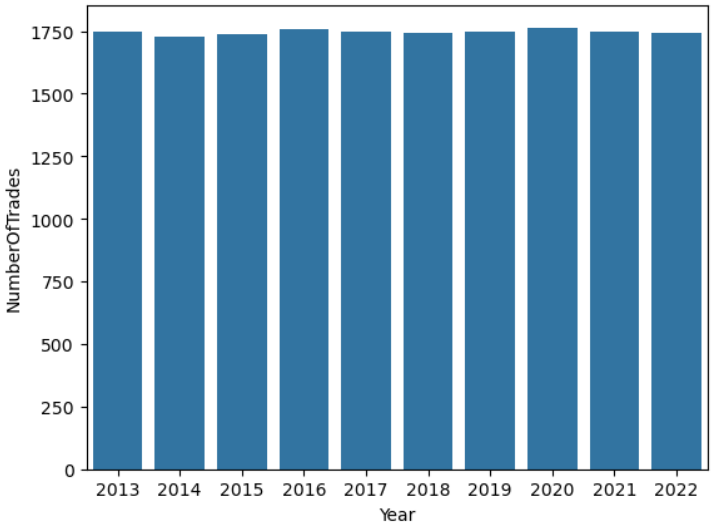
- Xem xét tổng khối lượng giao dịch trong từng ngày trong tuần. Theo kết quả biểu diễn ở hình **3.6**, với thứ 2, thứ 3, thứ 4, thứ 5, thứ 6, thứ 7, Chủ Nhật lần lượt là 0, 1, 2, 3, 4, 5, 6, ta thấy rằng dữ liệu không bao gồm thứ 7 và chủ nhật (do thứ 7 và chủ nhật, các sàn giao dịch không hoạt động). Đồng thời khối lượng giao dịch của 2 ngày đầu tuần cao hơn rất nhiều so với những ngày còn lại.

	Vol
	sum
dayofweek	
0	75548605
1	81016394
2	72851016
3	73431784
4	69654354

Hình **3.6**: Kết quả giao dịch được nhóm theo ngày trong tuần.

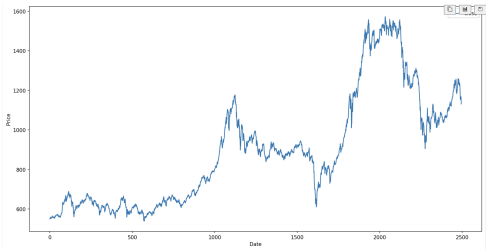
- Tổng các record trong từng năm từ ngày 01/01/2013 đến ngày 31/12/2022 được mô tả dưới đây(biểu diễn ở hình **3.7**. Ta có thể thấy, mỗi năm số lượng record sẽ khác nhau, điều này chứng tỏ, sẽ có những ngày bị thiếu liên tục, chứ không những chỉ ngày thứ 7 và chủ nhật.

```
2013 1750
2014 1729
2015 1736
2016 1757
2017 1750
2018 1743
2019 1750
2020 1764
2021 1750
2022 1743
```

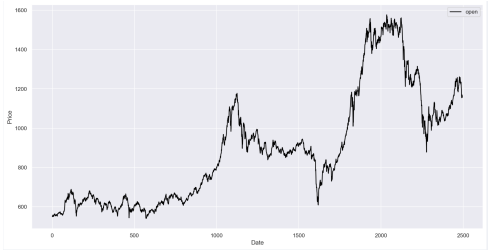


Hình **3.7**: Biểu đồ cột biểu diễn tổng giao dịch hằng năm.

- Hình **3.8**a, **3.8**b, **3.8**c, **3.8**d biểu diễn lần lượt đồ thị của từng giá Đóng, Mở, Trần, Sàn của dữ liệu theo từng ngày. Ta có thể thấy, các feature này không có một xu hướng nhất định mà tăng, giảm thất thường.

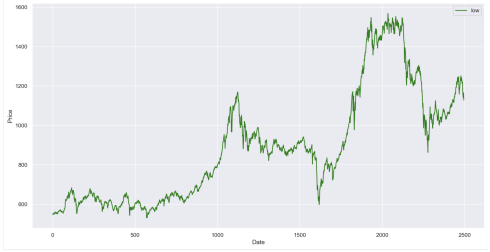


(a) Biểu đồ giá đóng



(b) Biểu đồ giá mở

(c) Biểu đồ giá trần

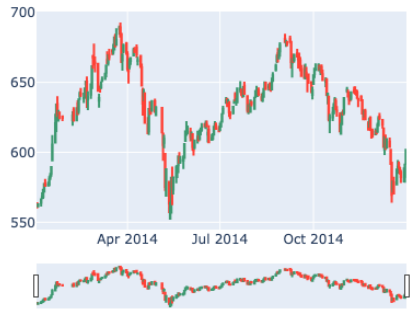


(d) Biểu đồ giá sàn

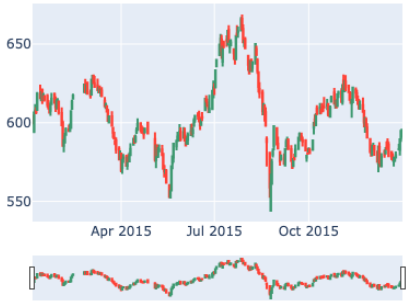
Hình **3.8**: Các biểu đồ về các giá theo ngày.

- Từ các dữ liệu giá đóng, mở, trần, sàn kể trên, ta xem thử biểu đồ đặt trưng của cổ phiếu là biểu đồ dạng nến(hình **3.9**a, **3.9**b, **3.9**c, , **3.9**d, **3.9**e, **3.9**f, **3.10**a, **3.10**b, **3.10**c và **3.10**c). Ta thấy, từ năm 2013 đến năm 2019, cổ phiếu có xu hướng tăng(từ 400 lên đến đỉnh điểm là gần 1200) tuy nhiên vào tháng 4 năm 2018 thì cổ phiếu bắt đầu giảm mạnh đến năm 2020, đây chắc hẳn là do ảnh hưởng của dịch bệnh Covid-19, tuy nhiên đến giữa năm 2020 (tháng 7) thì cổ phiếu lại tăng mạnh cho đến tháng 12 năm 2022 thì cổ phiếu vẫn đang giảm và đang duy trì mức ổn.

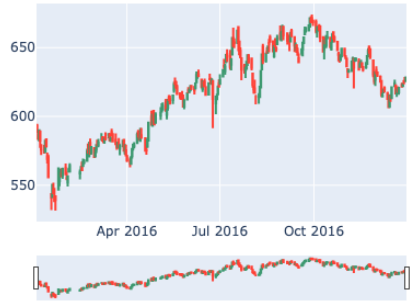
(a) Biểu đồ nến trong năm 2013



(b) Biểu đồ nến trong năm 2014



(c) Biểu đồ nến trong năm 2015



(d) Biểu đồ nến trong năm 2016

(e) Biểu đồ nến trong năm 2017

(f) Biểu đồ nến trong năm 2018

Hình **3.9**: Các biểu đồ candlestick mỗi năm - 1.

(a) Biểu đồ nến trong năm 2019

(b) Biểu đồ nến trong năm 2020

(c) Biểu đồ nến trong năm 2021

(d) Biểu đồ nến trong năm 2022

Hình **3.10**: Các biểu đồ candlestick mỗi năm - 2.

Sau khi đã phân tích được những dữ liệu có sẵn, nhóm tiến hành thêm các chỉ số mới vào(sử dụng thư viện hỗ trợ là **pandas_ta**), bao gồm:

- RSI: chỉ số này sẽ đi kèm với chu kì của nó nên ta cần chọn chu kỳ hợp lí, nhóm chọn chu kỳ dựa vào hệ số tương quan của nó đối với giá đóng và từ đó chọn ra chu kỳ hợp lí(code mô tả bên dưới). Khi này nhóm chọn ra được chu kỳ tốt nhất của RSI là 14 ngày.

```
for i in range(5,21):
    df = data.copy()
    df['RSI'] = ta.rsi(df.Close, length=i)
    df.dropna(inplace=True)

    print(i,':',df['RSI'].corr(data['Close']))

####### OUTPUT #########
# 5 : 0.030753081691752542
# 6 : 0.032020993706694215
# 7 : 0.033073507407670066
# 8 : 0.033976963455208016
# 9 : 0.03474006362265739
# 10 : 0.035353352795538286
# 11 : 0.03580749273764777
# 12 : 0.03609928131412639
# 13 : 0.03623212186602929
# 14 : 0.036214528255693046
# 15 : 0.03605827356301644
# 16 : 0.03577678861366704
# 17 : 0.03538396834057648
# 18 : 0.03489336609940319
# 19 : 0.03431769940973615
# 20 : 0.03366858415289855
```

- EMA: Như đã mô tả ở phần Data feature, nhóm đã chọn:
  - EMAF: EMA nhanh với chu kỳ 20 ngày.
  - EMAM: EMA trung bình với chu kỳ 50 ngày.
  - EMAS: EMA chậm với chu kỳ 100 ngày.

```
data['EMAF'] = ta.ema(data.Close, length=20)
data['EMAM'] = ta.ema(data.Close, length=50)
data['EMAS'] = ta.ema(data.Close, length=100)
```

- VWAP: Đầu tiên ta cần tính trung bình giá trong 1 ngày bằng cách cộng các giá đóng, trần và sàn, sau đó chia 3. Sau đó dựa vào giá trung bình và khối lượng giao dịch ta sẽ tính được VWAP.

```
data['average'] = (data['High'] + data['Low'] + data['Close'])/3
data['vwap'] = (data['average'] * data['Vol'])/ data['Vol']
```

Sau khi đã có hết tất cả dữ liệu cần thiết, nhóm tiến hành scale lại dữ liệu để phục vụ mục đích

training và testing model hiệu quả hơn, đồng thời cũng bỏ các cột dữ liệu không cần thiết như Date, index, average,... Nhóm sử dụng Standardize Scaler của thư viện scikit-learn.

3.2 Models Implementation

Sau khi đã có tập dữ liệu, nhóm tiến hành thêm các target label để phục vụ cho việc dự đoán.

- Đối với mô hình hồi quy, LSTM thì ta cần dự đoán đường đi từ hôm nay đến ngày tiếp theo, vì vậy ta cần target label là giá đóng của ngày hôm sau.
- Đối với các mô hình phân loại thì ta cần dự đoán xem có nên mua cổ phiếu vào ngày mai hay không. Vì vậy target label sẽ là True nếu như giá cổ phiếu ngày mai lớn hơn giá cổ phiếu hôm nay, là False nếu ngược lại.

Khi có được target label cần thiết thì ta tiến hành phân chia dữ liệu để training và testing, tỉ lệ phân chia là 80-20. 80% dữ liệu cho training và 20% dữ liệu cho testing. Đối với dữ liệu dùng để làm input đầu vào bao gồm các feature:

- Close
- Open
- High
- Low
- Vol
- RSI
- EMAF
- EMAM
- EMAS
- vwap
- vwap_pct_ret

3.2.1 Linear Regression

Sử dụng mô hình LinearRegression có sẵn trong module sklearn.linear_model cung cấp. Sau đó ta chỉ cần fit dữ liệu train và test để xây dựng model.

```
linear_reg = LinearRegression().fit(X_train_regression, y_train_regression)
```

3.2.2 SVC

Sử dụng mô hình SVC có sẵn trong module sklearn.svm cung cấp. Sau đó ta chỉ cần fit dữ liệu train và test để xây dựng model. Nhóm sử dụng 3 loại kernel:

- Tuyến tính.
- rbf.
- poly bậc 2.

```
# Support vector classifier
cls_rbf = SVC().fit(X_train_classification, y_train_classification)

cls_linear = SVC(kernel = 'linear').fit(X_train_classification, y_train_classification)

cls_poly = SVC(kernel = 'poly', degree = 2 ).fit(X_train_classification, y_train_classification)
```

3.2.3 SVR

Tương tự SVM, SVR có sẵn trong module sklearn.svm cung cấp. Sau đó ta chỉ cần fit dữ liệu train và test để xây dựng model. Tuy nhiên nhóm sử dụng các kernel khác nhau để thực hiện mục đích so sánh.

```
#rgb
svm_reg_rgb = SVR().fit(X_train_regression, y_train_regression)
#linear
svm_reg_linear = SVR(kernel='linear').fit(X_train_regression, y_train_regression)
#poly
svm_reg_poly = SVR(kernel='poly').fit(X_train_regression, y_train_regression)
```

3.2.4 LSTM

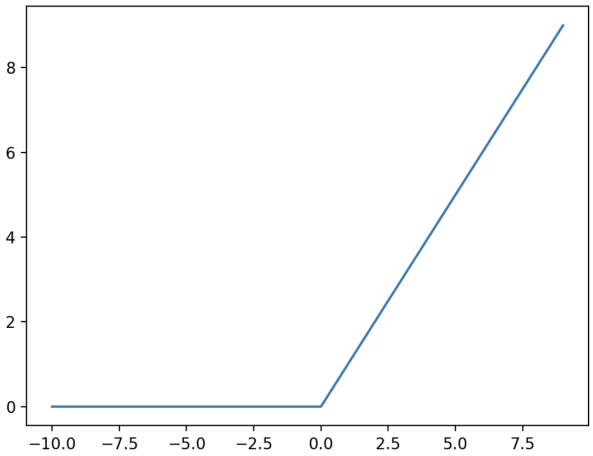
Mô hình LSTM thì việc xây dựng model khá phức tạp hơn những model đã liệt kê ở trên.

- Đầu tiên ta cần phải chọn số lượng backcandles(lượng data trước đó dùng để suy ra được data tiếp theo). Nhóm chọn backcandles dựa trên việc xây dựng nhiều model lstm trên nhiều backcandles rồi lấy mô hình có MSE(mean square error) nhỏ nhất.
- Mô hình LSTM bao gồm 2 lớp, với hàm kích hoạt(Activation function) là 'ReLU':
  - LSTM với 150 node.
  - Dense Layer.

```
def create_lstm_model(backcandles, X_train_lstm, y_train_lstm):
    lstm_input = Input(shape=(backcandles, 10), name='lstm_input')
    inputs = LSTM(150, name='first_layer')(lstm_input)
    inputs = Dense(1, name='dense_layer')(inputs)
    output = Activation('relu', name='output')(inputs)
    lstm_model = Model(inputs=lstm_input, outputs=output)
    adam = optimizers.Adam()
    lstm_model.compile(optimizer=adam, loss='mse')
    lstm_model.fit(x=X_train_lstm, y=y_train_lstm, batch_size=20, epochs=40,
        shuffle=True, validation_split = 0.1)
    return lstm_model
```

- Dữ liệu được train sẽ được nhóm theo từng bó các backcandles để có thể đưa vào mô hình, vì vậy để đưa dữ liệu vào mô hình ta cần nhóm theo từng bó các backcandles.
- Như đã đề cập ở phần trên, do dữ liệu đã được chuẩn hoá (Standardize) về khoảng (-1,1) vì vậy sẽ gặp vấn đề với hàm kích hoạt ReLU do hàm ReLU có dạng như hình **3.11**. Vì vậy khi target label

là giá đóng của ngày tiếp theo được chuẩn hoá thì, tiến hành cộng giá trị đó cho 2 để tránh những giá trị âm và làm ảnh hưởng đến việc dự đoán.



Hình **3.11**: Hàm ReLU.

- Sau khi đã chọn được backcandles tốt nhất thì đồng thời ta cũng xây dựng được mô hình LSTM phù hợp.

4 Result

4.1 Linear Regression

So sánh giữa tập kết quả dự đoán và kết quả thực tế được biểu diễn ở hình **4.12** với kết quả dự đoán là đường màu xanh và kết quả thực tế là đường màu đỏ.

Hình **4.12**: Kết quả dự đoán sử dụng Linear Regression

Mean Square Error khá thấp: 0.0046932074618599336

4.2 SVC

Đối với bài toán phân loại thì ta cần quan tâm đến accuracy score:

```
print(accuracy_score(y_pred_svm, y_test_classification))

#Output linear: 0.5459098497495827
#Output rbf: 0.5536534446764092
#Output poly: 0.5465553235908142
```

Kết quả khá thấp, tuy nhiên ta có thể vẽ xu hướng bằng cách:

- Tạo cột Predict_Signal là cột kết quả mà ta đã dự đoán.
- Cột Return sẽ là % thay đổi của giá đóng so với ngày trước (là cột Percentage change cũ).
- Ta sẽ tính cột Stragy_Return bằng cách lấy cột Return nhân với Predict_Signal.
- Tính tổng lợi nhuận thực tế bằng cách tính tổng dồn các giá trị Return trước đó.

```
data_svm['Cum_Ret'] = data_svm['Return'].cumsum()
```

- Tính tổng lợi nhuận chiến lược bằng cách tính tổng dồn các giá trị Stragy_Return trước đó:

```
data_svm['Cum_Strategy'] = data_svm['Strategy_Return'].cumsum()
```

- Kết quả so sánh 2 feature Cum_Strategy và Cum_Ret biểu diễn ở hình **4.13** đối với kernel rbf, **4.14** đối với kernel linear, **4.15** đối với kernel poly bậc 2, với Cum_Strategy được biểu diễn màu xanh còn Cum_Ret được biểu diễn màu đỏ.

Hình **4.13**: Kết quả dự đoán sử dụng SVC bằng rbf kernel

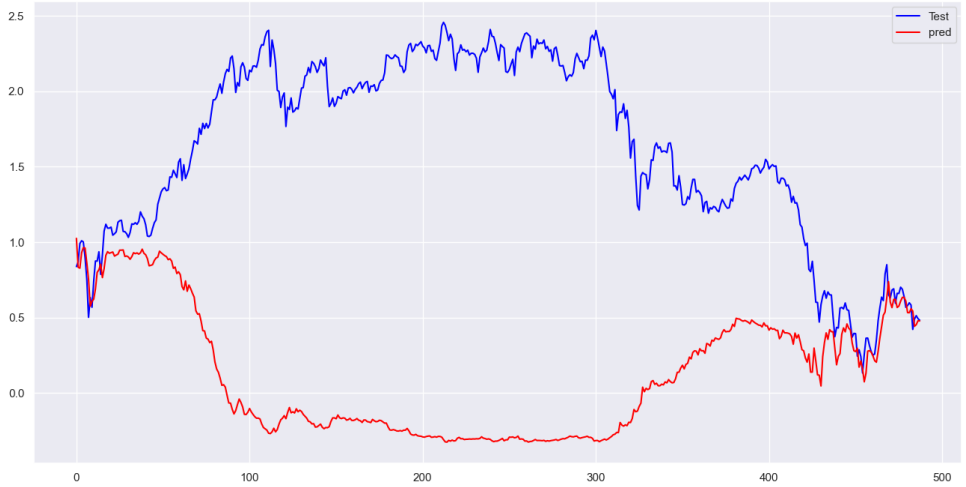
Hình **4.14**: Kết quả dự đoán sử dụng SVC bằng linear kernel

Hình **4.15**: Kết quả dự đoán sử dụng SVC bằng poly kernel

4.3 SVR

Kết quả của mô hình SVR bao gồm 3 kernel liệt kê bên dưới với tập kết quả dự đoán và kết quả thực tế được biểu diễn ở các biểu đồ kèm theo với kết quả dự đoán là đường màu xanh và kết quả thực tế là đường màu đỏ.

- rbf với Mean Square Error: 3.193165293082233

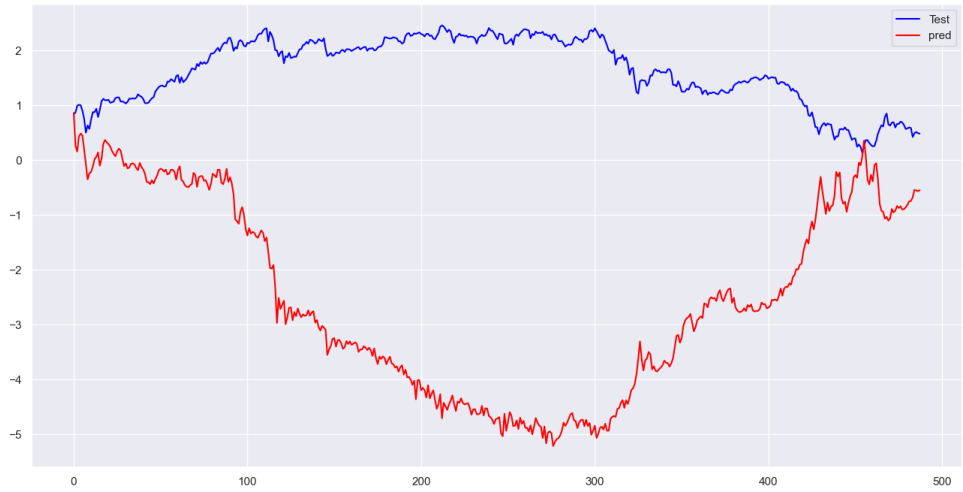


Hình **4.16**: Kết quả dự đoán sử dụng SVR kernel rbf

- Linear với Mean Square Error: 0.0056477085456062455

- Poly bậc 2 với Mean Square Error: 22.0981658659009

Hình **4.17**: Kết quả dự đoán sử dụng SVR kernel linear



Hình **4.18**: Kết quả dự đoán sử dụng SVR kernel poly

4.4 LSTM

So sánh giữa tập kết quả dự đoán và kết quả thực tế được biểu diễn ở hình **4.19** với kết quả dự đoán là đường màu xanh lá và kết quả thực tế là đường màu đen.

Hình **4.19**: Kết quả dự đoán sử dụng LSTM

Kết quả Mean Square Error: 0.06502993339174792

5 Conclusion

Trong đề tài dự đoán giá cổ phiếu sử dụng kết hợp giữa chỉ số RSI và các mô hình học máy như LSTM, SVM, và SVR, chúng ta đã tiến hành những phương pháp phổ biến để nghiên cứu và áp dụng vào lĩnh vực tài chính.

Chỉ số RSI, với khả năng đo lường sức mạnh tương đối của xu hướng giá, đã giúp chúng ta nhìn nhận và hiểu rõ hơn về biến động của thị trường. Qua đó, chúng ta đã tích hợp RSI vào các mô hình học máy như LSTM, SVM và SVR để tận dụng sức mạnh của cả hai phương pháp.

Mô hình LSTM, với khả năng học từ chuỗi dữ liệu thời gian dài, đã giúp chúng ta nắm bắt được các mô hình phức tạp của biến động giá cổ phiếu. SVM, với khả năng làm việc tốt trong không gian đa chiều, và Decision Tree, với khả năng mô phỏng quyết định dựa trên dữ liệu đa biến, đã cung cấp những góc nhìn khác nhau nhưng quan trọng đối với dự đoán giá cổ phiếu.

Kết quả của đề tài không chỉ là việc xây dựng một hệ thống dự đoán chính xác mà còn là sự kết hợp linh hoạt giữa phân tích kỹ thuật truyền thống và học máy. Điều này có thể giúp nhà đầu tư và quản lý rủi ro đưa ra quyết định đầu tư thông minh, dựa trên những thông tin hữu ích từ cả thế giới tài chính truyền thống và xu hướng mới của công nghệ máy học.

Tuy nhiên, như mọi nghiên cứu, đề tài cũng mang theo những hạn chế và thách thức. Đối diện với sự biến động không dự đoán được của thị trường, việc cải tiến và mở rộng hệ thống là một hành trình liên tục. Hy vọng rằng đề tài này sẽ đóng góp một phần nhỏ vào sự hiểu biết và phát triển trong lĩnh vực dự đoán giá cổ phiếu.

Hạn chế

- Nhóm vẫn chưa thể dự đoán được 1 khoảng thời gian rộng trong tương lai, cụ thể vẫn chỉ dự đoán được giá đóng của ngày kế tiếp, chứ chưa dự đoán được giá của tuần kế hay tháng kế.
- Chưa kiểm thử tính đúng đắng của mô hình với dữ liệu gần đây.