

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM  
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG  
KHOA CÔNG NGHỆ THÔNG TIN



ĐỖ MINH NHẬT – 52000697

XÂY DỰNG MÔ HÌNH CHUỖI THỜI  
GIAN DỰ ĐOÁN XU HƯỚNG TRÊN  
SÀN CHỨNG KHOÁN CỦA CÁC CÔNG  
TY TẠI VIỆT NAM

CHUYÊN ĐỀ NGHIÊN CỨU 1  
KỸ THUẬT PHẦN MỀM

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM  
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG  
KHOA CÔNG NGHỆ THÔNG TIN



ĐỖ MINH NHẬT – 52000697

XÂY DỰNG MÔ HÌNH CHUỖI THỜI  
GIAN DỰ ĐOÁN XU HƯỚNG TRÊN  
SÀN CHỨNG KHOÁN CỦA CÁC CÔNG  
TY TẠI VIỆT NAM

CHUYÊN ĐỀ NGHIÊN CỨU 1  
KỸ THUẬT PHẦN MỀM

*Người hướng dẫn:*

**PGS.TS. Lê Anh Cường**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024

## LỜI CẢM ƠN

Lời đầu tiên, nhóm chúng em xin chân thành cảm ơn đến PGS.TS. Lê Anh Cường. Trong khoảng thời gian làm bài thi nhóm chúng tôi đã gặp nhiều khó khăn trong giai đoạn xác định nội dung làm bài nhưng nhờ có thầy đã giải đáp những khuất mắt và những thứ liên quan mà nhóm chưa hiểu để có thể thực hiện đề tài nghiên cứu một cách tốt nhất.

Bên cạnh đó, nhóm chúng em cũng xin gửi lời cảm ơn sâu sắc đến Khoa Công Nghệ Thông Tin vì đã tạo điều kiện cho chúng tôi được tham gia tìm hiểu và nghiên cứu đề tài. Khoa đã luôn sẵn sàng giúp đỡ, cung cấp những tài liệu cần thiết để phục vụ trong quá trình làm bài của nhóm.

Sau cùng, tập thể nhóm chúng em xin chân thành cảm ơn rất nhiều.

*TP. Hồ Chí Minh, ngày tháng năm*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Đỗ Minh Nhật*

## CÔNG TRÌNH ĐƯỢC HOÀN THÀNH TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Chúng tôi xin cam đoan đây là công trình nghiên cứu của riêng chúng tôi và được sự hướng dẫn khoa học của thầy PGS.TS. Lê Anh Cường. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong bài nghiên cứu còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày tháng năm

Tác giả

(ký tên và ghi rõ họ tên)

Đỗ Minh Nhật

# XÂY DỰNG MÔ HÌNH CHUỖI THỜI GIAN DỰ ĐOÁN XU HƯỚNG TRÊN SÀN CHỨNG KHOÁN CỦA CÁC CÔNG TY TẠI VIỆT NAM

## TÓM TẮT

Đề tài nghiên cứu mà nhóm chúng tôi thực hiện liên quan đến việc sử dụng các kỹ thuật về học máy, trí tuệ nhân tạo cùng với kiến thức liên quan đến lĩnh vực tài chính, chứng khoán để xây dựng một mô hình chuỗi thời gian có thể dự đoán được xu hướng trên sàn chứng khoán của các công ty tại Việt Nam. Chúng tôi hướng đến việc sẽ thu thập các dữ liệu chứng khoán của 10 công ty lớn nhất tại Việt Nam và lấy mốc thời gian từ tháng 1 năm 2014 đến tháng 1 năm 2024 làm cơ sở thu thập. Bằng những dữ liệu đã thu thập được trên sàn chứng khoán, chúng tôi có thể dựa vào đó mà tái cấu trúc dữ liệu đầu vào. Chúng tôi tính toán các chỉ số, các chỉ báo đo lường trong chứng khoán để thuận tiện hơn cho việc dự đoán. Tính toán và đánh nhãn xu hướng biến đổi của thị trường chứng khoán đối với các công ty tương ứng. Tổng hợp các dữ liệu đó và thêm vào cùng với dữ liệu ban đầu. Sau đó thì chúng tôi sẽ sử dụng mô hình các mô hình như Logistic Regression và Random Forest để tiến hành huấn luyện để đưa ra dự đoán về xu hướng giá trên sàn chứng khoán. Bên cạnh đó, chúng tôi cũng sẽ thực hiện xây dựng mô hình ARIMA và mô hình LSTM để huấn luyện và dự đoán xu hướng, sau đó so sánh giữa các kết quả dự đoán với nhau và đánh giá tổng quan hiệu suất dự đoán.

# **BUILDING A TIME SERIES MODEL TO PREDICT TRENDS ON THE STOCK EXCHANGE OF COMPANIES IN VIETNAM**

## **ABSTRACT**

The research topic that our team is conducting is relate to the use of machine learning and artificial intelligence techniques along with knowledge in the field of finance and stock market to build a time series model which can predict trends on the stock exchange of companies in Vietnam. We aim to collect stock data of the 10 largest companies in Vietnam and the timeline we will take is from January 2014 to January 2024. Based on the collected data, we can restructure the input data. Then, we will calculate financial indicators and measurement indicators to facilitate prediction. We calculate and label the trend of the stock market for the corresponding companies. We finally aggregate all the data and add to the original data. We will then use models such as Logistic Regression and Random Forest to train and predict the price trend on the stock market. In addition, we will also build ARIMA and LSTM models to train and predict trends, and then compare the prediction results with each other and evaluate the overall prediction performance.

## MỤC LỤC

DANH MỤC CÁC BẢNG BIẾU, HÌNH VẼ, ĐỒ THI .....	viii
DANH MỤC CHỮ VIẾT TẮT .....	xii
CHƯƠNG 1 – GIỚI THIỆU BÀI TOÁN .....	1
1.1 Tổng quan về bài toán .....	1
1.2 Mục tiêu và phạm vi, phương pháp nghiên cứu.....	2
1.2.1 Mục tiêu nghiên cứu .....	2
1.2.2 Phạm vi và phương pháp nghiên cứu.....	3
1.3 Nội dung công việc .....	5
CHƯƠNG 2 – KIẾN THỨC NỀN TẢNG .....	7
2.1 Tổng quan về thị trường chứng khoán Việt Nam .....	7
2.1.1 Khái niệm về thị trường chứng khoán .....	7
2.1.2 Các dữ liệu của thị trường chứng khoán .....	8
2.2 Mô hình Logistic Regression .....	10
2.2.1 Khái niệm.....	10
2.2.2 Công thức toán tổng quát.....	10
2.3 Mô hình Random Forest .....	11
2.3.1 Khái niệm.....	11
2.3.2 Xây dựng thuật toán.....	12
2.4 Mô hình LSTM .....	12
2.4.1 Khái niệm.....	12
2.4.2 Giải thuật của mô hình LSTM .....	13
2.5 Mô hình ARIMA.....	15
2.5.1 Khái niệm.....	15
2.5.2 Xây dựng mô hình chuỗi thời gian bằng ARIMA .....	16
CHƯƠNG 3 – PHƯƠNG PHÁP ĐỀ XUẤT .....	17
3.1 Các nghiên cứu liên quan.....	17

3.1.1 Xây dựng mô hình dự đoán giá cổ phiếu dựa trên mô hình học sâu (Lounnapha Sayavong, 2019) .....	17
3.1.2 Nghiên cứu về dự đoán giá chứng khoán dựa trên mô hình học máy và mô hình truyền thông (Daiyou Xiao, 2022).....	17
3.2 Mô hình đề xuất .....	18
3.2.1 Mô hình Logistic Regression .....	18
3.2.2 Mô hình Random Forest .....	18
3.2.3 Mô hình ARIMA.....	18
3.2.4 Mô hình LSTM .....	19
3.3 Mô hình xử lý dữ liệu.....	19
3.3.1 Các miền dữ liệu .....	19
3.3.2 Tái cấu trúc dữ liệu của các công ty .....	26
3.3.3 Tái cấu trúc dữ liệu tổng hợp các công ty.....	27
3.3.4 Tái cấu trúc dữ liệu của các công ty theo tháng.....	27
3.3.5 Tái cấu trúc dữ liệu tổng hợp các công ty theo tháng .....	27
CHƯƠNG 4 – THỰC NGHIỆM .....	28
4.1 Xây dựng dữ liệu.....	28
4.1.1 Thu thập dữ liệu .....	28
4.1.2 Trực quan hóa dữ liệu .....	29
4.1.3 Tiền xử lý dữ liệu.....	36
4.2 Mô hình thực nghiệm .....	48
4.2.1 Mô hình Logistic Regression .....	48
4.2.2 Mô hình Random Forest .....	49
4.2.3 Mô hình LSTM .....	49
4.2.4 Mô hình ARIMA.....	50
4.3 Các tiêu chí, độ đo đánh giá mô hình.....	52
4.4 Kết quả thực nghiệm .....	53

4.4.1 Với dữ liệu của các công ty .....	53
4.4.2 Với dữ liệu tổng hợp các công ty.....	58
4.4.3 Với dữ liệu của các công ty theo tháng.....	59
4.4.4 Với dữ liệu tổng hợp các công ty theo tháng .....	64
4.4.5 Mô hình dự đoán dữ liệu mới .....	65
CHƯƠNG 5 – KẾT LUẬN.....	69
5.1 Thành quả nghiên cứu.....	69
5.2 Hạn chế và hướng phát triển tương lai.....	69
TÀI LIỆU THAM KHẢO.....	70

## **DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ**

### **DANH MỤC HÌNH**

Hình 3. 1 Mô hình LSTM .....	13
Hình 3. 2 Mô tả thành phần trong LSTM .....	14
Hình 4. 1 Cài đặt dữ liệu vnquant từ github.....	28
Hình 4. 2 Xác định các mã chứng khoán .....	29
Hình 4. 3 Tạo một tập hợp gồm dữ liệu của các công ty .....	29
Hình 4. 4 Dữ liệu sau khi được tập hợp .....	30
Hình 4. 5 Tập hợp các khoảng thời gian sẽ lấy dữ liệu.....	30
Hình 4. 6 Thông tin chi tiết của dữ liệu chứng khoán.....	31
Hình 4. 7 Mô tả cụ thể của dữ liệu chứng khoán .....	31
Hình 4. 8 Biểu đồ thể hiện độ tương quan của các cột trong dữ liệu chứng khoán .....	32
Hình 4. 9 Biểu đồ đường thể hiện giá đóng cửa của các công ty.....	32
Hình 4. 10 Biểu đồ cột thể hiện khối lượng giao dịch cao nhất của các công ty .....	33
Hình 4. 11 Biểu đồ cột thể hiện khối lượng giao thấp nhất của các công ty .....	33
Hình 4. 12 Biểu đồ nền biểu diễn các giá của các công ty.....	34
Hình 4. 13 Biểu đồ đường so sánh dữ liệu bình thường với dữ liệu có sử dụng trung bình trượt đơn giản và trung bình trượt mở rộng .....	34
Hình 4. 14 Biểu đồ đường thể hiện sự khác biệt giữa hai đường trung bình trượt mở rộng.....	35
Hình 4. 15 Biểu đồ Bollinger Bands .....	35
Hình 4. 16 Biểu đồ đường thể hiện độ mạnh hay yếu của xu hướng trên thị trường chứng khoán .....	36
Hình 4. 17 Làm mịn toàn bộ dữ liệu chứng khoán .....	36
Hình 4. 18 Tính toán phần trăm thay đổi của giá đóng cửa .....	37
Hình 4. 19 Lùi dữ liệu phần trăm thay đổi giá đóng cửa về 1 đến 5 ngày trước .....	37

Hình 4. 20 Tính tỉ lệ trung bình trượt mở rộng cho nhiều ngày.....	38
Hình 4. 21 Tính chỉ số phân tích chứng khoán RSI .....	38
Hình 4. 22 Tính chỉ số phân tích chứng khoán MACD .....	39
Hình 4. 23 Tính chỉ số phân tích chứng khoán OBV.....	39
Hình 4. 24 Tính chỉ số phân tích chứng khoán EMV .....	39
Hình 4. 25 Tính chỉ số phân tích chứng khoán MFI .....	39
Hình 4. 26 Tính chỉ số phân tích chứng khoán ROC .....	40
Hình 4. 27 Tính chỉ số phân tích chứng khoán ATR .....	40
Hình 4. 28 Tính chỉ số phân tích chứng khoán CMF.....	40
Hình 4. 29 Tính chỉ số phân tích chứng khoán STOCH .....	40
Hình 4. 30 Tính chỉ số phân tích chứng khoán CCI .....	41
Hình 4. 31 Độ dài dữ liệu của tất cả các mã chứng khoán.....	41
Hình 4. 32 Tính toán các chỉ số để cân bằng độ dài dữ liệu .....	41
Hình 4. 33 Tính trung bình giá cao nhất được tổng hợp từ các công ty .....	42
Hình 4. 34 Tính trung bình giá thấp nhất được tổng hợp từ các công ty .....	42
Hình 4. 35 Tính trung bình giá mở cửa được tổng hợp từ các công ty .....	42
Hình 4. 36 Tính trung bình giá đóng cửa được tổng hợp từ các công ty .....	42
Hình 4. 37 Tính trung bình giá trung bình được tổng hợp từ các công ty .....	43
Hình 4. 38 Tính trung bình khối lượng giao dịch được tổng hợp từ các công ty .....	43
Hình 4. 39 Chuyển các giá trị trung bình vừa tính được thành Dataframe.....	43
Hình 4. 40 Hàm tái cấu trúc lại dữ liệu .....	44
Hình 4. 41 Chuyển đổi dữ liệu các cột giá về mảng và tái cấu trúc giá đóng cửa .....	44
Hình 4. 42 Kết quả sau khi tái cấu trúc giá đóng cửa .....	45
Hình 4. 43 Tính trung bình các cột của các mã chứng khoán.....	45
Hình 4. 44 Tái cấu trúc dữ liệu khoảng 4 tuần .....	46
Hình 4. 45 Hàm tính toán sự chênh lệch giá .....	46
Hình 4. 46 Tính chênh lệch giá giữa các tuần với nhau.....	46

Hình 4. 47 Hàm tính trung bình của các chênh lệch giá .....	47
Hình 4. 48 Tính tổng trung bình của các chênh lệch giá giữa các tuần .....	47
Hình 4. 49 Đánh nhãn xu hướng dựa trên tổng trung bình của các chênh lệch giá tuần .....	47
Hình 4. 50 Xây dựng mô hình Logistic Regression.....	48
Hình 4. 51 Xây dựng mô hình Random Forest .....	49
Hình 4. 52 Xây dựng mô hình LSTM .....	50
Hình 4. 53 Tính toán bậc sai phân của chuỗi giá đóng .....	50
Hình 4. 54 Biểu đồ tự tương quan ACF .....	51
Hình 4. 55 Biểu đồ tự tương quan riêng phần PACF.....	51
Hình 4. 56 Xây dựng và huấn luyện mô hình ARIMA .....	52
Hình 4. 57 Kết quả thống kê sau khi huấn luyện mô hình .....	52
Hình 4. 58 Độ chính xác khi dự đoán bằng mô hình Logistic Regression .....	54
Hình 4. 59 Độ chính xác khi dự đoán bằng mô hình Random Forest.....	55
Hình 4. 60 Độ chính xác khi dự đoán bằng mô hình LSTM.....	56
Hình 4. 61 Độ chính xác khi dự đoán bằng mô hình ARIMA .....	57
Hình 4. 62 Kết quả dự đoán của mô hình Logistic Regression .....	58
Hình 4. 63 Kết quả dự đoán của mô hình Random Forest.....	58
Hình 4. 64 Kết quả dự đoán của mô hình LSTM.....	58
Hình 4. 65 Kết quả dự đoán mô hình ARIMA .....	59
Hình 4. 66 Kết quả dự đoán với mô hình Logistic Regression.....	60
Hình 4. 67 Kết quả dự đoán với mô hình Random Forest .....	61
Hình 4. 68 Kết quả dự đoán với mô hình LSTM .....	62
Hình 4. 69 Kết quả dự đoán với mô hình ARIMA .....	63
Hình 4. 70 Kết quả dự đoán mô hình Logistic Regression .....	64
Hình 4. 71 Kết quả dự đoán mô hình Random Forest .....	64
Hình 4. 72 Kết quả dự đoán mô hình LSTM .....	65

Hình 4. 73 Kết quả dự đoán mô hình ARIMA.....	65
Hình 4. 74 Lấy miền dữ liệu mới .....	66
Hình 4. 75 Miền dữ liệu mới được đưa vào dự đoán mô hình Logistic Regression.....	66
Hình 4. 76 Miền dữ liệu mới được đưa vào dự đoán mô hình Random Forest .....	67
Hình 4. 77 Kết quả dự đoán với mô hình Logistic Regression.....	67
Hình 4. 78 Kết quả dự đoán với mô hình Random Forest .....	68
Hình 4. 79 Kết quả dự đoán của hai mô hình Logistic Regression và Random Forest tương ứng lần lượt từ trái sang phải .....	68

## **DANH MỤC CHỮ VIẾT TẮT**

### **CÁC CHỮ VIẾT TẮT**

CNN	Convolutional Neural Network
VNX	Thị trường chứng khoán Việt Nam
MAE	Mean Absolute Error
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
VIC	Vingroup
VCB	VietcomBank
VNM	Vinamilk
GAS	PetroVietnam
HVN	Vietnam Airlines
MWG	Thế giới di động
FPT	The Corporation for Financing Promoting Technology
HPG	Hòa Phát Group
MSN	Masan group
SAB	Sabeco
LSTM	Long Short-Term Memory
ARIMA	Autoregressive Integrated Moving Average
RNN	Recurrent Neural Network
EMA	Exponential Moving Average
RSI	Relative Strength Index
MACD	Moving Average Convergence Divergence
OBV	On-Balance Volume
EMV	Ease of Movement
MFI	Money Flow Index
ADF	Argument Dickey Fuller

ROC	Rate Of Change
ATR	Average True Range
CMF	Chaikin Money Flow
STOCH	Stochastic Oscillator
CCI	Commodity Channel Index

## CHƯƠNG 1 – GIỚI THIỆU BÀI TOÁN

### 1.1 Tổng quan về bài toán

Trong đề tài nghiên cứu này, nhóm chúng tôi sẽ tập trung vào việc xây dựng mô hình chuỗi dữ liệu có khả năng dự đoán xu hướng biến động giá trên sàn chứng khoán của các công ty lớn tại Việt Nam dựa trên các giá chứng khoán của các công ty đó từ quá khứ cho đến hiện tại. Từ những dự đoán ấy mà các doanh nghiệp, các nhà đầu tư, các công ty sẽ có những chiến lược phù hợp để có thể đạt được mục tiêu lợi nhuận và giảm thiểu thiệt hại tối đa khi tham gia vào thị trường chứng khoán. Có thể nói đây là một bài toán mang tính hiện đại trong lĩnh vực khoa học máy tính vì mục tiêu của bài toán là việc dự đoán xu hướng trong tương lai của giá chứng khoán trong khi dựa vào dữ liệu trước đó. Ngoài lĩnh vực tài chính thì đề tài xây dựng mô hình chuỗi dữ liệu này còn có khả năng dự đoán trong các lĩnh vực khác như kinh doanh, quảng cáo, thời tiết, quảng cáo...

Điểm mạnh của bài toán này là việc nó có thể trở thành một công cụ đắc lực giúp hỗ trợ quyết định mua bán trong quá trình tham gia thị trường chứng khoán nếu như nó được huấn luyện với một lượng dữ liệu đủ lớn và việc dự đoán gần như sai lệch rất thấp nhưng cũng chính vì điều này cản trở nó đến với phổ biến đại đa số các nhà kinh doanh. Lý do gây trở ngại chính là nó đặt nặng việc khai thác và xử lý dữ liệu đủ tốt để có thể dự đoán chính xác xu hướng giá trên sàn chứng khoán của các công ty tại Việt Nam trong tương lai.

Sau khi trình bày điểm mạnh và nhược điểm của đề tài thì nhóm chúng tôi nhận định rằng đây là đề tài luôn nổi bật và có khả năng phát triển mạnh trong tương lai vì sự phát triển nhanh chóng của các thị trường tài chính ngày nay tại Việt Nam như thị trường chứng khoán, thị trường bitcoin và còn một số thị trường khác liên quan. Chính vì lẽ đó đã thôi thúc chúng tôi tham gia vào nghiên cứu và mong muốn có thể tạo ra một mô hình chuỗi dữ liệu đủ mạnh để có thể thực hiện dự đoán xu hướng giá trên thị trường chứng khoán của các công ty tại Việt Nam.

## **1.2 Mục tiêu và phạm vi, phương pháp nghiên cứu**

### ***1.2.1 Mục tiêu nghiên cứu***

Sau quá trình thực hiện đề tài nghiên cứu thì chúng tôi mong rằng có thể mang được những thứ giá trị đến cho người đọc báo cáo như là cách triển khai quy trình nghiên cứu, các thông tin thu thập được liên quan đến chứng khoán, giải thuật mà nhóm tìm hiểu cũng như hiện thực thành ứng dụng cho cuộc sống thực tế. Đặc biệt là phần hiện thực, chúng tôi đặt ra mục tiêu rằng có thể tạo ra được một mô hình với dữ liệu đủ tốt phục vụ cho công việc dự đoán xu hướng giá tương lai trên sàn chứng khoán của các công ty tại Việt Nam. Qua đó có thể tạo ra được một công cụ dự đoán hiệu quả để có thể hỗ trợ tốt đến với những quyết định đầu tư về chứng khoán.

Để có thể đặt được mục tiêu lớn này, chúng tôi sẽ đề ra các mục tiêu nhỏ hơn để có thể tiếp nối đến thành công tạo ra một mục tiêu lớn.

Thứ nhất, tìm hiểu và thiết kế phương pháp tái cấu trúc dữ liệu, tiến hành việc tính toán các chỉ số chứng khoán, các chỉ báo phục vụ cho việc dự đoán và thêm vào các khung dữ liệu này. Hệ thống các cơ sở lý luận về các mô hình Linear Regression, Random Forest để phục vụ cho việc huấn luyện mô hình với dữ liệu đã được tái cấu trúc. Đồng thời hệ thống cơ sở lý luận về các mô hình ARIMA và LSTM để xây dựng mô hình và phục vụ cho việc so sánh hiệu suất dự đoán.

Thứ hai, tiến hành xây dựng các mô hình đã tìm hiểu để dự đoán xu hướng biến động của thị trường chứng khoán với dữ liệu chứng khoán của 10 công ty lớn nhất tại Việt Nam.

Thứ ba, đánh giá các mô hình đã xây dựng và đánh giá hiệu suất giữa các mô hình, từ đó có thể đánh giá được hiệu suất đánh giá và có thể cho ra lời khuyên có các doanh nghiệp cũng như cá nhân tham gia đầu tư chứng khoán.

### **1.2.2 Phạm vi và phương pháp nghiên cứu**

#### **- Phạm vi nghiên cứu:**

Trong quá trình tìm hiểu để thực hiện đề tài thì có rất nhiều đối tượng dữ liệu mà nhóm chúng tôi hướng đến nhưng để không bị nhiễu loạn dữ liệu mà vẫn giữ được sự tối ưu nhất trong quá trình xây dựng mô hình chuỗi dữ liệu để dự đoán xu hướng giá trên sàn chứng khoán thì những dữ liệu mà nhóm quyết định chọn đó là các dữ liệu như giá đóng cửa, giá mở cửa, giá cao nhất, giá thấp nhất, giá trung bình, khối lượng giao dịch của các công ty tại Việt Nam trên sàn chứng khoán để phục vụ cho quá trình xử lý, trực quan hóa và dự đoán xu hướng giá trong tương lai.

Phạm vi mà nhóm chúng tôi sẽ thực hiện quá trình thu thập dữ liệu chứng khoán là từ 10 công ty lớn tại Việt Nam trên sàn chứng khoán như Vingroup (VIC), Vietcombank (VCB), Vinamilk (VNM), PetroVietnam (GAS), Vietnam Airlines (HVN), Thế giới di động (MWG), Viễn thông FPT (FPT), Hòa phát group (HPG), Masan group (MSN), Sabeco (SAB). Sau khi đã xác định được các công ty thì tiếp theo là chúng tôi sẽ xác định khoảng thời gian mà chúng tôi sẽ thu thập dữ liệu, như đã đề cập ở phần tóm tắt thì khoảng thời gian mà nhóm chọn là từ ngày 1 tháng 1 năm 2014 đến ngày 1 tháng 1 năm 2024. Chúng tôi nghĩ rằng với những phạm vi đã giới hạn này sẽ giúp chúng tôi có thể tập trung làm tốt nhất cho công việc xử lý, trực quan hóa và xây dựng mô hình một cách chuyên sâu và tốt nhất để có thể đưa ra được dự đoán cho xu hướng tương lai của giá chứng khoán.

#### **- Phương pháp nghiên cứu:**

Sau khi đã có đối tượng cũng như phạm vi nghiên cứu, nhóm chúng tôi sẽ tiến hành tìm hiểu các phương pháp giải quyết đề tài nghiên cứu và sau khi thống nhất thì chúng tôi đã đề ra được một phương pháp tiếp cận cụ thể để dựa vào nó mà thực hiện. Phương pháp áy bao gồm các bước như sau:

Bước 1 là phần thu thập dữ liệu đầu vào, nhóm chúng tôi sẽ sử dụng các dữ liệu như giá mở cửa, đóng cửa, cao nhất, thấp nhất, trung bình và khối lượng giao dịch trên

các sàn chứng khoán Việt Nam như HOSE (Sở giao dịch Chứng khoán Thành phố Hồ Chí Minh) hoặc HNX (Sở giao dịch Chứng khoán Hà Nội). Trong đó, nhóm chúng tôi sẽ đảm bảo tuân thủ phần phạm vi mà chúng tôi đã đề ra từ trước để có thể xử lý, trực quan hóa và huấn luyện mô hình một cách thuận lợi nhất.

Bước 2 là phần tiền xử lý dữ liệu, trong bước này chúng tôi sẽ xử lý các dữ liệu bị thiếu bằng cách sử dụng các kỹ thuật như điền giá trị trung bình sau khi đã tính toán các giá trị tương tự với giá trị bị thiếu hoặc sử dụng các thư viện có hỗ trợ việc tự động điền các giá trị bị thiếu. Ngoài ra, đôi khi còn có trường hợp một số dữ liệu tồn tại dấu câu trong đó thì cũng cần được xử lý bằng cách loại bỏ dấu câu đó để dữ liệu sạch sẽ và nhìn ổn hơn.

Bước 3 là phần phân tích các cột dữ liệu và trực quan hóa nó lên các biểu đồ để có thể dễ xem cũng như hiểu rõ hơn về sự tăng trưởng của nó trong khoảng thời gian đã thu thập. Ngoài ra, chúng tôi còn có thể đánh giá được xu hướng thường xuyên của một mã cổ phiếu nào đó để phục vụ cho phần thuật toán sau.

Bước 4 là phần quan trọng và trọng tâm trong phần nghiên cứu của nhóm chúng tôi. Ở bước này chúng tôi sẽ thực hiện công việc tái cấu trúc dữ liệu chuỗi thời gian trên sàn chứng khoán. Chúng tôi sẽ dựa vào các dữ liệu giá của chứng khoán để tính toán các chỉ báo dữ liệu cũng như xu hướng của chứng khoán. Đồng thời, xử lý tối ưu lại dữ liệu mới tính toán và đánh nhận xu hướng biến đổi của chứng khoán tương ứng. Gộp dữ liệu mới tính toán và thêm cùng với dữ liệu ban đầu.

Bước 5 là bước chuẩn hóa dữ liệu, vì trong quá trình huấn luyện dữ liệu của các mô hình thì dữ liệu cần phải đồng nhất trước khi tiến hành đào tạo. Công việc này đa phần sẽ là cần thiết để dữ liệu được làm sạch, giảm độ biến động và tăng sự ổn định hơn cho mô hình sau khi đã huấn luyện và đưa ra dự đoán.

Bước 6 là bước chúng tôi sẽ chia dữ liệu ra thành các tập dữ liệu nhỏ hơn, trong đó có hai tập dữ liệu là tập dữ liệu huấn luyện (train) được dùng để đào tạo mô hình và

tập dữ liệu kiểm tra (test) được dùng để thực hiện dự đoán và đánh giá hiệu suất cũng như độ chính xác của mô hình.

Bước 7 là xây dựng mô hình để thực hiện việc huấn luyện và đưa ra dự đoán về xu hướng giá trên sàn chứng khoán. Các mô hình chính mà nhóm chúng tôi quyết định sử dụng là mô hình Logistic Regression và Random Forest vì nó rất phù hợp với chuỗi dữ liệu đã được chúng tôi tái cấu trúc lại như bước 4. Ngoài ra, trong suốt quá trình huấn luyện thì nhóm sẽ liên tục điều chỉnh các tham số trong mô hình huấn luyện để có thể tối ưu hóa hiệu suất và độ chính xác đầu ra của đề tài.

Bước 8 là thực hiện xây dựng thêm các mô hình ARIMA và LSTM với dữ liệu đã được tiền xử lý để dự đoán xu hướng giá trên sàn chứng khoán. Qua đó có cái nhìn tổng quan hơn và có thể đánh giá hiệu suất giữa các mô hình.

Bước 9 là bước cuối cùng đảm nhiệm công việc thực hiện dự đoán kết quả trên tập kiểm tra (test) và đánh giá độ đo chính xác (accuracy) giữa các mô hình.

### **1.3 Nội dung công việc**

Tìm hiểu về vấn đề bài toán của chúng tôi và cách thức thực hiện để giải quyết bài toán là như thế nào. Sau đó, nhóm chúng tôi tiến hành việc nghiên cứu phương pháp tái cấu trúc dữ liệu và nghiên cứu thêm nhiều miền dữ liệu giúp ích cho mô hình dự đoán của nhóm. Đồng thời thiết kế và xây dựng các mô hình mà nhóm đã giới thiệu, huấn luyện mô hình và đánh giá độ chính xác của mô hình.

Nội dung công việc của nhóm chung tôi sẽ được giới thiệu kĩ hơn thông qua nội dung các chương của bài báo cáo về đề tài nghiên cứu của nhóm chúng. Tổng quan về công việc của nhóm chúng tôi như sau.

#### **Chương 1: Giới thiệu**

- Chương này sẽ giới thiệu về vấn đề nghiên cứu của bài toán, các mục tiêu và phạm vi nghiên cứu, phương pháp nghiên cứu.

#### **Chương 2: Các kiến thức cơ bản**

- Chương này sẽ bao gồm tổng quan các lý thuyết liên quan đến vấn đề mà nhóm đang nghiên cứu cũng như phương pháp mà nhóm dự định thực hiện để xử lý bài toán.
- Ngoài ra cũng sẽ bao gồm hệ thống các lý thuyết và các mô hình mà nhóm dự định sẽ sử dụng.

### Chương 3: Phương pháp đề xuất

- Chương này sẽ bao gồm về các liên cứu nghiên quan mà chúng tôi tham khảo để có thể có cái nhìn tổng quan hơn về vấn đề bài toán và cách thức xây dựng các phương pháp hiện giờ của chúng tôi.
- Ngoài ra sẽ bao gồm đưa ra các đề xuất thiết kế xây dựng mô hình và cách thức thiết kế bộ dữ liệu được đưa vào huấn luyện cho mô hình.

### Chương 4: Thực nghiệm

- Chương này sẽ bao gồm quá trình thực nghiệm sau khi đã tìm hiểu lý thuyết và có những đề xuất phù hợp về thiết kế xây dựng và hệ thống cần xây dựng.
- Bao gồm quá trình xây dựng dữ liệu của nhóm chúng tôi. Chúng tôi sẽ làm rõ ra về các loại dữ liệu mà chúng tôi sử dụng để thực hiện việc tái cấu trúc dữ liệu. Ngoài ra cũng sẽ trình bày về cách thức tính toán cũng như ý nghĩa của độ đo đánh giá mô hình của chúng tôi.
- Qua đó cho ra các bước chạy thực nghiệm và đánh giá các kết quả đạt được với nhau.

### Chương 5: Kết luận

- Chương này sẽ bao gồm tổng quát lại các kết quả nhóm đã đạt được, đồng thời có một cái nhìn và đánh giá tổng quát nhất về giải pháp mà nhóm xây dựng để giải quyết vấn đề. Ngoài ra cũng sẽ tổng quát lại những mặt hạn chế và hướng phát triển trong tương lai.

## CHƯƠNG 2 – KIẾN THỨC NỀN TẢNG

### 2.1 Tổng quan về thị trường chứng khoán Việt Nam

#### 2.1.1 Khái niệm về thị trường chứng khoán<sup>1</sup>

Thị trường chứng khoán là nơi mà các hoạt động phát hành, giao dịch, mua bán và trao đổi các loại chứng khoán được diễn ra. Hàng hóa trên thị trường này là các sản phẩm tài chính bao gồm trái phiếu, cổ phiếu và một số công cụ tài chính trung và dài hạn khác, được phát hành để huy động vốn cho doanh nghiệp và chính phủ.

Các chủ thể tham gia vào thị trường chứng khoán bao gồm các nhà phát hành và các nhà đầu tư. Trong đó, các nhà phát hành là các tổ chức thực hiện huy động vốn thông qua thị trường chứng khoán. Các nhà phát hành này có thể là Chính phủ, Chính quyền địa phương và Công ty. Thị trường chứng khoán bao gồm các chức năng chính như tạo ra một môi trường đầu tư cho mọi người, huy động vốn đầu tư cho nền kinh tế, giúp đánh giá hoạt động của doanh nghiệp và tạo ra tính thanh khoản cho các chứng khoán trong thị trường...

Riêng về thị trường chứng khoán Việt Nam (có tên viết tắt là VNX) được thành lập vào năm 2000 và đã có những bước phát triển vượt bậc trong thời gian qua. Tính đến ngày 19/01/2024, quy mô vốn hóa của thị trường đã đạt hơn 8,7 triệu tỷ đồng, tương đương khoảng 40% GDP. Số lượng nhà đầu tư tham gia thị trường cũng tăng trưởng nhanh chóng, với hơn 4,5 triệu tài khoản giao dịch. VNX đã đóng góp vai trò quan trọng trong việc huy động vốn cho nền kinh tế. Trong năm 2023, thị trường đã huy động được hơn 700.000 tỷ đồng cho doanh nghiệp phát hành, trong đó có nhiều doanh nghiệp lớn như Vingroup, Masan Group, Vietcombank...

Thị trường chứng khoán Việt Nam được đánh giá là một thị trường tiềm năng, do đó mong muốn xây dựng được một mô hình đủ mạnh để có thể dự đoán được xu hướng biến động của thị trường tại Việt Nam sẽ giúp ích rất nhiều cho nhiều doanh nghiệp hay

---

<sup>1</sup> <https://takeprofit.vn/thi-truong-chung-khoan-la-gi/1635323133893>

cá nhân tham gia vào đầu tư, góp phần cho thị trường chứng khoán Việt Nam sẽ ngày càng phát triển hơn. Nhóm chúng tôi sẽ lựa chọn ra 10 công ty có thị trường chứng khoán lớn nhất tại Việt Nam là Vingroup (VIC), Vietcombank (VCB), Vinamilk (VNM), PetroVietnam (GAS), Vietnam Airlines (HVN), Thế giới di động (MWG), FPT (FPT), Hòa phát group (HPG), Masan group (MSN) và Sabeco (SAB).

### **2.1.2 Các dữ liệu của thị trường chứng khoán<sup>2</sup>**

Đối với thị trường chứng khoán, sẽ bao gồm các loại dữ liệu sau:

- Giá mở cửa (Open Price): là giá của một cổ phiếu tại thời điểm bắt đầu phiên giao dịch, có nghĩa là giá tại thời điểm giao dịch đầu tiên trong ngày giao dịch. Giá mở cửa là giá được xác định bởi giá mà người mua và người bán đồng ý trong giao dịch đầu tiên của ngày.
- Giá đóng cửa (Close Price): là giá của một cổ phiếu tại thời điểm kết thúc phiên giao dịch, có nghĩa là giá tại thời điểm giao dịch cuối cùng trong ngày giao dịch. Giá đóng cửa là giá được xác định bởi giá mà người mua và người bán đồng ý trong giao dịch cuối cùng của ngày.
- Giá cao nhất (High Price): là giá cao nhất mà một cổ phiếu đạt được trong suốt một khoảng thời gian nhất định, thường là trong một ngày giao dịch. Đây là giá đỉnh cao nhất mà người mua đã sẵn sàng trả trong khoảng thời gian đó.
- Giá thấp nhất (Low Price): là giá thấp nhất mà một cổ phiếu đạt được trong suốt một khoảng thời gian nhất định, thường là trong một ngày giao dịch. Đây là giá thấp nhất mà người mua đã sẵn sàng trả trong khoảng thời gian đó.
- Khối lượng giao dịch (Trading Volume): là lượng cổ phiếu được giao dịch trong một khoảng thời gian cụ thể, thường là trong một ngày giao dịch. Nó

---

<sup>2</sup> <https://topi.vn>

đo lường tổng số lượng cổ phiếu đã được chuyển đổi chủ sở hữu trong khoảng thời gian đó.

- Dữ liệu thời gian trong chứng khoán liên quan đến thông tin về giá chứng khoán, khối lượng giao dịch và các chỉ số khác, được thu thập và ghi lại dựa trên các khoảng thời gian cụ thể. Thông tin về thời gian này giúp xác định thứ tự và tính liên tục của dữ liệu, tạo nên chuỗi thời gian. Đây là một khía cạnh đặc biệt quan trọng trong phân tích kỹ thuật và dự đoán xu hướng giá chứng khoán.
- Chênh lệch giữa giá đóng cửa (Close Price) và giá mở cửa (Open Price) là một chỉ số đặc biệt quan trọng trong phân tích chứng khoán và dĩ nhiên nó có thể sẽ mang đến những thông tin quan trọng về xu hướng của thị trường chứng khoán trong tương lai. Khi giá đóng cửa cao hơn giá mở cửa thì đây là một tín hiệu tích cực cho việc tăng số lượng mua và giảm số lượng bán trong suốt phiên giao dịch. Khi giá đóng cửa thấp hơn giá mở cửa thì đây là một tín hiệu tiêu cực cho việc giảm số lượng mua và tăng số lượng bán trong suốt phiên giao dịch.
- Chênh lệch giữa giá cao nhất (High Price) và giá thấp nhất (Low Price) là một chỉ số đặc biệt quan trọng trong phân tích chứng khoán và nó thể hiện mức độ biến động của thị trường chứng khoán. Nếu hai giá này chênh lệch lớn thì ta có thể thấy một ngày giao dịch có biến động cao và ngược lại.
- Các chỉ số như EMA, MACD, CMF, EMV, RSI, Stochastic Oscillator...

Do vậy, các dữ liệu cơ bản nhất mà nhóm chúng tôi sẽ thu thập để đưa vào cho việc mô hình sẽ bao gồm:

- Các giá chứng khoán: Dữ liệu về giá mở cửa (Open), giá đóng cửa (Close), giá cao nhất (High), giá thấp nhất (Low) trong mỗi khoảng thời gian (mỗi ngày). Đây là các thông tin chính để xây dựng chuỗi thời gian.

- Khối lượng giao dịch (Volume): Dữ liệu về lượng cổ phiếu được giao dịch trong mỗi đợt giao dịch. Khối lượng giao dịch thể hiện được sự quan tâm và tương tác của nhà đầu tư.
- Dữ liệu thời gian: Thông tin về ngày giờ của mỗi điểm dữ liệu là quan trọng để xác định thứ tự của chuỗi thời gian. Điều này giúp mô hình hiểu được sự biến động theo thời gian.
- Các chỉ số kỹ thuật: Các chỉ số như Exponential Moving Average, Relative Strength Index (RSI), Stochastic Oscillator, và các chỉ số khác có thể cung cấp thông tin bổ sung về xu hướng và sức mạnh của thị trường.

## 2.2 Mô hình Logistic Regression<sup>3</sup>

### 2.2.1 Khái niệm

Mô hình Logistic Regression (hồi quy logistic) là một thuật toán học có giám sát (Supervised Learning) trong Machine Learning, nó là một phương pháp thống kê dùng để ước lượng mối quan hệ giữa các biến độc lập và biến phụ thuộc. Mô hình này là sự kết hợp giữa thuật toán Linear Regression (hồi quy tuyến tính) và thuật toán Perceptron Learning Algorithm (PLA) vì nó tổng hợp các sự tương quan giữa các biến độc lập liên quan và tìm ra hàm tuyến tính tốt nhất để biểu diễn mối liên hệ này thành những phân loại phù hợp trong đoạn từ 0 đến 1. Mặc dù có tên là Regression nhưng phương pháp này chủ yếu được sử dụng cho các bài toán phân loại (Classification).

### 2.2.2 Công thức toán tổng quát

Điểm đặc biệt của mô hình Logistic Regression là nó sử dụng hàm logistic hay hàm sigmoid để chuyển đổi giá trị đầu ra sau khi tính toán từ các giá trị đầu vào thành một giá trị xác suất nằm trong khoảng từ 0 đến 1. Hàm sigmoid được định nghĩa bởi công thức sau:

---

<sup>3</sup> <https://machinelearningcoban.com/2017/01/27/logisticregression/>

$$f(s) = \frac{1}{1 + e^{-s}} \triangleq \sigma(s)$$

Được sử dụng nhiều nhất vì nó chặn trong khoảng (0, 1) như sau:

$$\lim_{s \rightarrow -\infty} \sigma(s) = 0 \text{ & } \lim_{s \rightarrow \infty} \sigma(s) = 1$$

Ở công thức của hàm sigmoid như trên thì s là một tổ hợp tuyến tính của các biến đầu vào và các trọng số tương ứng. Cụ thể thì s được biểu diễn như sau.

$$z = b_0 + b_1x_1 + b_2x_2 + \cdots + b_nx_n$$

Trong đó:

$b_0$ : Là hệ số chêch (intercept).

$b_1, b_2, \dots, b_n$ : Là các trọng số.

$x_1, x_2, \dots, x_n$ : Là các biến đầu vào.

Sau khi tính xong giá trị s, chúng ta sử dụng hàm sigmoid để chuyển đổi nó thành giá trị xác suất. Nếu giá trị xác suất này cao hơn một ngưỡng (thường là 0.5), chúng ta sẽ dự đoán được một sự kiện xảy ra và ngược lại thì sẽ dự đoán được một sự kiện còn lại sẽ xảy ra.

Mô hình Logistic Regression thường được huấn luyện bằng cách tối ưu hàm chi phí để đạt được các trọng số b phù hợp nhất cho dữ liệu huấn luyện mô hình.

## 2.3 Mô hình Random Forest<sup>4</sup>

### 2.3.1 Khái niệm

Mô hình Random Forest là mô hình học máy thuộc loại Ensemble Learning và kết hợp với kỹ thuật lấy mẫu tái lặp lại (Boosting), nó kết hợp việc dự đoán từ nhiều mô hình nhỏ để tạo ra được một dự đoán tổng quát nhất. Nhiệm vụ chính của mô hình này là thực hiện việc phân loại và hồi quy.

Cụ thể thuật toán này sẽ tạo ra một rừng các cây quyết định, mà trong đó mỗi cây quyết định sẽ được huấn luyện dựa trên nhiều mẫu con khác nhau và cho ra các kết quả

---

<sup>4</sup> [https://phamdinhkhanh.github.io/deepai-book/ch\\_ml/RandomForest.html#mo-hinh-rung-cay](https://phamdinhkhanh.github.io/deepai-book/ch_ml/RandomForest.html#mo-hinh-rung-cay)

dự báo từ toàn bộ những cây quyết định, đây còn gọi là Voting. Một kết quả dự báo được tổng hợp từ nhiều mô hình nhỏ sẽ cho ra kết quả không bị lệch quá nhiều. Đồng thời kết hợp kết quả dự báo từ nhiều mô hình sẽ cho ra phương sai nhỏ hơn so với chỉ một mô hình. Điều này giúp cho mô hình khắc phục được hiện tượng quá khớp (Overfitting).

### 2.3.2 Xây dựng thuật toán

Giả sử chúng ta có  $n$  dữ liệu (sample) và mỗi dữ liệu có  $d$  thuộc tính (feature). Để có thể xây dựng mỗi cây quyết định thì ta sẽ có thuật toán như sau:

Lấy ngẫu nhiên  $n$  dữ liệu từ bộ dữ liệu và kết hợp với kỹ thuật bootstrapping. Tức khi mình tạo ra được 1 dữ liệu thì mình không bỏ dữ liệu đó ra mà vẫn giữ lại trong tập dữ liệu ban đầu, rồi tiếp tục tạo ra thêm dữ liệu cho tới khi đủ  $n$  dữ liệu. Khi dùng kỹ thuật này thì tập  $n$  dữ liệu mới có thể có những dữ liệu bị trùng nhau.

Sau khi đã tạo ra được  $n$  dữ liệu từ bước 1 thì ta sẽ chọn ngẫu nhiên  $k$  thuộc tính ( $k < n$ ). Sau bước này thì ta sẽ có một bộ dữ liệu mới gồm  $n$  dữ liệu và mỗi dữ liệu có  $k$  thuộc tính.

Sau đó ta xây dựng cây quyết định với bộ dữ liệu mới đã tìm được ở bước 2.

Trong quá trình xây dựng mỗi cây quyết định đều có yếu tố ngẫu nhiên (random) nên kết quả là các cây quyết định trong thuật toán Random Forest đều có thể khác nhau.

Thuật toán Random Forest sẽ bao gồm nhiều cây quyết định, mỗi cây sẽ bầu chọn cho một nhãn nhất định cho đến hết  $n$  dữ liệu. Sau đó kết quả dự đoán của thuật toán Random Forest sẽ được tổng hợp từ các cây quyết định đã bầu chọn trước đó.

## 2.4 Mô hình LSTM<sup>5</sup>

### 2.4.1 Khái niệm

Mô hình LSTM (Long Short-Term Memory) là một mô hình được xây dựng dựa trên mô hình RNNs (Recurrent Neural Network) nhưng đã được cải tiến hơn để giải quyết vấn đề về Gradient Vanishing tốt hơn. Tuy nhiên có một nhược điểm của mô hình

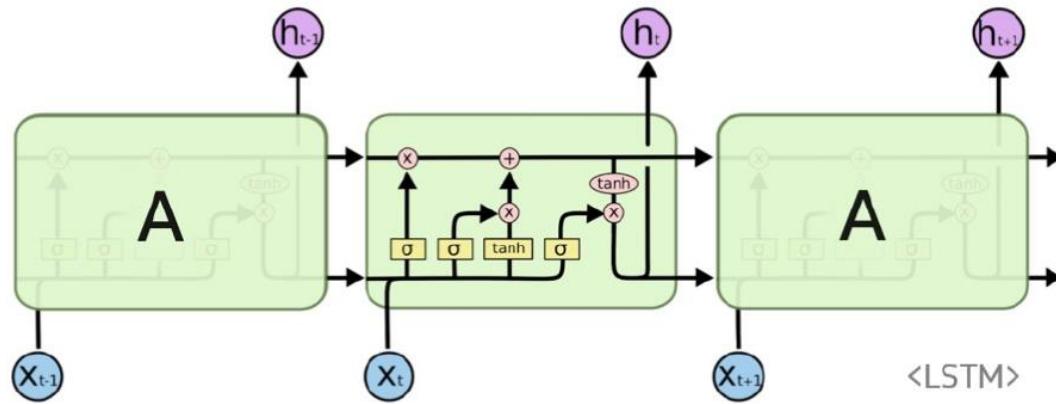
---

<sup>5</sup> <https://dominhhai.github.io/vi/2017/10/what-is-lstm/>

là sự quá phức tạp của nó nên tốc độ huấn luyện bị ảnh hưởng khá nhiều và chậm hơn so rất nhiều với mô hình RNNs.

Vấn đề Gradient Vanishing được nhắc đến bên trên là một hiện tượng mà ở đó khi số lượng các dữ liệu tăng lên dẫn đến số lượng units cũng sẽ tăng lên theo và đồng thời khi đó các gradient sẽ giảm dần ở các units cuối do công thức Đạo hàm chuỗi gây nên, chính vì điều này nên sẽ dẫn đến sự mất thông tin phụ thuộc ở xa giữa các units với nhau. Hiện tượng trên thường xảy ra trong mô hình RNNs nhưng mô hình LSTM đã giải quyết được điều đó và giúp cho mô hình học tốt hơn ở các chuỗi dữ liệu dài đầu vào, đồng thời nó cũng có thể xử lý các thông tin liên quan ở xa tốt hơn so với mô hình truyền thống.

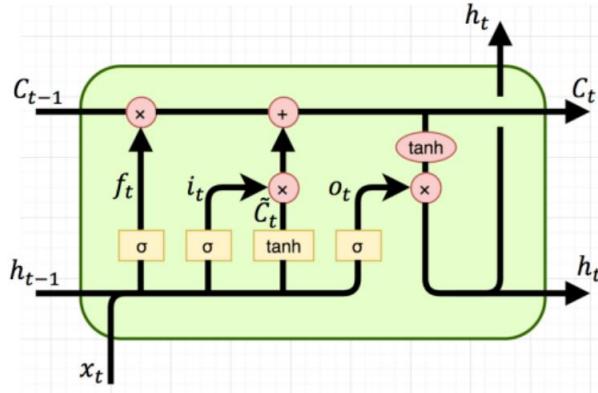
Các thành phần của mô hình LSTM bao gồm cổng quên (forget gate), cổng đầu vào (input gate), cổng đầu ra (output gate) và ô trạng thái (cell state). Trong đó các thông tin được lưu trữ ở ô trạng thái và được sử dụng để đưa ra kết quả đầu ra. Các mô hình LSTM đều có các biến thể khác nhau tùy thuộc vào bài toán là gì để xây dựng một mô hình LSTM tốt nhất



Hình 3. 1 Mô hình LSTM

#### 2.4.2 Giải thuật của mô hình LSTM

Với mô hình LSTM, các thành phần của mô hình đều có những nhiệm vụ riêng và chúng tương tác với nhau một cách hiệu quả.



Hình 3. 2 Mô tả thành phần trong LSTM

Đầu tiên, các thông tin nào sẽ bị bỏ quên và không đưa vào ô trạng thái sẽ được cổng forget gate lựa chọn. Cổng này nhận các đầu vào là  $x_t$  và trạng thái ẩn (hidden state)  $h_{t-1}$  qua thông qua hàm kích hoạt  $\sigma$  đưa về giá trị (0,1):

$$f_t = \sigma(U_f * x_t + W_f * h_{t-1} + b_f)$$

Thông tin mới sau đó được quyết định những thông tin nào sẽ được lưu trữ thông qua cổng đầu vào với 1 hàm sigmoid để quyết định giá trị sẽ được cập nhật và 1 hàm tanh tạo ra một véc-tơ mới  $C_t$  để thêm vào trạng thái.

$$i_t = \sigma(U_i * x_t + W_i * h_{t-1} + b_i)$$

$$\tilde{C}_t = \tanh(U_c * x_t + W_c * h_t + b_c)$$

Sau đó, ô mới (cell state)  $C_t$  cập nhật trạng thái từ các thông tin ở ô trạng thái trước đó, thực hiện  $f_t * C_{t-1}$  và khi đó  $i_t * \tilde{C}_t$  là các thông tin mới sẽ được cập nhật ở ô trạng thái mới:

$$C_t = \sigma(f_t * C_{t-1} + i_t * \tilde{C}_t)$$

Cuối cùng, ta sẽ quyết định xem đầu ra sẽ cần trả về những thông tin gì thông qua cổng đầu ra với hàm  $\sigma$  và chuyển các thông tin từ trạng thái mới  $C_t$  cho hàm tanh và trả về giá trị (-1, 1) rồi sau đó nhân với giá trị đầu ra của hàm  $\sigma$  trước đó.

$$o_t = \sigma(U_o * x_t + W_o * h_{t-1} + b_o)$$

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

## 2.5 Mô hình ARIMA<sup>6</sup>

### 2.5.1 Khái niệm

ARIMA là viết tắt của cụm từ Autoregressive Integrated Moving Average. Mô hình sẽ biểu diễn phương trình hồi quy tuyến tính đa biến (Multiple Linear Regression) của các biến đầu vào (hay còn gọi là biến phụ thuộc ở trong thống kê) gồm có ba thành phần chính:

- Đầu tiên là Auto regression (AR), bao gồm các tập hợp của độ trễ hiện tại với giá trị p là giá trị lùi về quá khứ p bước thời gian của chuỗi thời gian chúng ta đưa vào.

$$AR(p) = \phi_0 + \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p}$$

- Thứ hai là Moving average (MA), là quá trình biến đổi giá trị trung bình của chuỗi thời gian tương ứng. Quá trình sẽ thực hiện việc hồi quy tuyến tính với các giá trị của chuỗi thời gian hiện tại và quá khứ của sai số nhiễu trắng (White noise error term) được xem như là đại diện cho các yếu tố về giá chứng khoán ngẫu nhiên.

$$MA(q) = \mu + (1 + \theta_1 B + \cdots + \theta_q B^q) \epsilon_t$$

- Cuối cùng là Integrated (I), lấy sai phân để đánh giá và xác định chuỗi dữ liệu đưa vào có tính dừng. Đây là đây là quá trình đồng tích hợp với bậc d và quá trình này được thực hiện như sau:

- Sai phân bậc 1:  $I(1) = \Delta(x_t) = x_t - x_{t-1}$
- Sai phân bậc d:  $I(d) = \Delta^d(x_t) = \underbrace{\Delta(\Delta(\dots \Delta(x_t)))}_{d \text{ times}}$

Như vậy, về tổng quát thì ARIMA là mô hình sử dụng dữ liệu trong quá khứ để dự đoán dữ liệu trong tương lai thông qua quá trình đồng tích hợp I để xác định tính dừng và kết hợp của hai quá trình AR và MA.

$$\Delta x_t = \emptyset_1 \Delta x_{t-1} + \emptyset_2 \Delta x_{t-2} + \cdots + \emptyset_p \Delta x_{t-p} + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q}$$

---

<sup>6</sup> <https://bigdatauni.com/tin-tuc/tim-hieu-ve-time-series-p-5-so-luoc-ve-arima.html>

### **2.5.2 Xây dựng mô hình chuỗi thời gian bằng ARIMA**

Đầu tiên, sau khi đã có được chuỗi dữ liệu thời gian, ta tiến hành kiểm tra tính dừng của chuỗi dữ liệu. Với việc kiểm định ta sẽ sử dụng kiểm định nghiệm đơn vị Argument Dickey Fuller và xác định được bậc của quá trình tích hợp d.

Sau đó, ta tiến hành vẽ các biểu đồ ACF và PACF để phân tích tính tương quan giữa các dữ liệu quan sát. Xác định các giá trị p và q để đưa ra các dạng mô hình dự báo được cho là tốt.

- ACF (Autocorrelation Function) là biểu đồ tự tương quan được dùng để tìm độ trễ của quá trình trung bình trượt q thông qua công thức:

$$p(s, t) = \frac{cov(x_s, x_t)}{\sqrt{\sigma_s \sigma_t}}$$

- PACF (Partial Autocorrelation Function) biểu đồ tương quan riêng phần, cũng là chỉ số đo lường hệ số tương quan như ACF. Biểu đồ này được dùng để xác định giá trị lùi p.

$$\phi_k = corr(x_t - P_{t,k}(x_t), x_{t-k} - P_{t,k}(x_t - k))$$

Tiếp đến, ta sẽ tiến hành việc huấn luyện với các dạng mô hình được dự đoán là tốt ở trên và tiến hành dự đoán, đánh giá tính chính xác của mô hình đồng thời ta cũng sẽ có dạng mô hình tốt nhất.

## CHƯƠNG 3 – PHƯƠNG PHÁP ĐỀ XUẤT

### 3.1 Các nghiên cứu liên quan

Với đề tài là nghiên cứu của nhóm liên quan đến lĩnh vực tài chính và bài toán dự đoán nên những nghiên cứu của các tác giả khác mà chúng tôi đã tìm hiểu và sẽ đề cập dưới đây đều sẽ có liên quan đến đề tài của nhóm tuy nhiên các nghiên cứu áy sẽ tập trung vào các mô hình kiến trúc khác hoặc là mục tiêu dự đoán đầu ra cũng khác so với chúng tôi.

#### *3.1.1 Xây dựng mô hình dự đoán giá cổ phiếu dựa trên mô hình học sâu (Louannapha Sayavong, 2019)<sup>7</sup>*

##### 3.1.1.1 Mục tiêu và phạm vi nghiên cứu

Tác giả đã tập trung vào các dữ liệu trên thị trường chứng khoán Thái Lan để có thể đưa ra được các dự đoán về giá cổ phiếu trong tương lai. Mục đích bài toán của tác giả là mong muốn cải thiện hiệu suất dự đoán và giúp cho các nhà đầu tư ở Thái Lan tránh được rủi ro và đưa ra được các quyết định hợp lý trên thị trường của họ.

##### 3.1.1.2 Phương pháp giải quyết

Tác giả đã xây dựng và ứng dụng mô hình học sâu CNN. Sau đó kết hợp với dữ liệu trên thị trường chứng khoán Thái Lan cho ra được các kết quả xác định xu hướng biến động hiệu quả của giá trị cổ phiếu. Các dự đoán này được tổng hợp thành tài liệu tham khảo có giá trị về việc dự đoán giá cổ phiếu.

#### *3.1.2 Nghiên cứu về dự đoán giá chứng khoán dựa trên mô hình học máy và mô hình truyền thống (Daiyou Xiao, 2022)<sup>8</sup>*

##### 3.1.2.1 Mục tiêu và phạm vi nghiên cứu

Tác giả của nghiên cứu này đã tập trung vào thị trường chứng khoán của Mỹ cụ thể là chứng khoán của New York. Mục tiêu của tác giả là có thể thực hiện các nghiên

---

<sup>7</sup><https://ieeexplore.ieee.org/document/8920761/authors#authors>

<sup>8</sup>[https://www.researchgate.net/publication/359646743\\_Research\\_on\\_Stock\\_Price\\_Time\\_Series\\_Prediction\\_Based\\_on\\_Deep\\_Learning\\_and\\_Autoregressive\\_Integrated\\_Moving\\_Average](https://www.researchgate.net/publication/359646743_Research_on_Stock_Price_Time_Series_Prediction_Based_on_Deep_Learning_and_Autoregressive_Integrated_Moving_Average).

cứu chuyên sâu về các mô hình học máy để dự đoán và so sánh hiệu suất cũng như tính chính xác giữa các mô hình.

### 3.1.2.2 Phương pháp giải quyết

Phương pháp giải quyết của tác giả là thu thập dữ liệu chứng khoán của New York. Sau đó thực hiện nghiên cứu và xây dựng hai mô hình là ARIMA và LSTM, tối ưu hai mô hình để đạt hiệu quả tốt nhất. Sau đó, so sánh các mô hình với nhau và đưa ra kết luận cuối cùng.

## 3.2 Mô hình đề xuất

### 3.2.1 Mô hình Logistic Regression

Chúng tôi sẽ xây dựng mô hình Logistic Regression với tham số quy định hằng phạt (penalty) được sử dụng trong quá trình tối ưu hóa hàm chi phí. Regularization này giúp tránh được tình trạng Overfitting bằng cách kiểm soát hàng loạt giá trị của các trọng số. Ngoài ra còn cần có một tham số để điều chỉnh sức mạnh của Regularization này, trong đó giá trị của C càng nhỏ thì cho thấy rằng mức độ regularization càng mạnh và ngược lại. Tóm lại mô hình được xây dựng sẽ có thể huấn luyện với dữ liệu đầu vào và phải tránh được hiện tượng Overfitting nếu có.

### 3.2.2 Mô hình Random Forest

Đối với việc xây dựng mô hình Random Forest, chúng tôi sẽ chọn số lượng cây quyết định tốt nhất cho việc tối đa hóa hiệu suất mô hình và đồng thời chúng tôi cũng sẽ thiết lập một seed để đảm bảo rằng mỗi khi chạy mô hình thì sẽ cho ra kết quả giống nhau nếu nó được cùng một seed sử dụng. Việc này tương đối phù hợp với việc mô tả mô hình vì nó có thể tái tạo kết quả đầu ra và so sánh giữa các lần chạy khác nhau.

### 3.2.3 Mô hình ARIMA

Chúng tôi sẽ tiến hành việc lựa chọn các mô hình tham số cho mô hình ARIMA là các tham số ( $p, d, q$ ). Với tham số  $d$  ta sẽ tìm được thông qua kiểm tra tính dừng của chuỗi thời gian.

Tiếp đến, chúng tôi sẽ tìm kiếm tham số  $q$  là độ trễ của trung bình trượt MA bằng cách thông qua biểu đồ tự tương quan (ACF). Nếu tại một độ trễ nhỏ nhất mà đồ dài đại diện cho giá trị hệ số tự tương quan nằm ngoài khoảng tin cậy thì đó chính là độ trễ phù hợp lớn nhất nên lựa chọn cho quá trình trung bình trượt.

Cuối cùng chúng tôi sẽ tìm kiếm tham số  $p$  là hệ số bậc tự do của quá trình hồi qui AR thông qua tự tương quan riêng phần (PACF). Tương tự như ACF, thông qua biểu đồ của PACF với các giá trị hệ số tự tương quan riêng phần khác nhau ta có thể tìm ra được các tham số  $p$  thích hợp.

Thông qua các cách đã được giới thiệu ở trên, chúng tôi đã có được vô số kịch bản cho mô hình tùy thuộc vào các tham số đầu vào và có thể tiến hành xác định mô hình nào là tốt nhất.

### **3.2.4 Mô hình LSTM**

Với mô hình LSTM, chúng tôi sẽ xây dựng một mô hình tuần tự gồm 2 lớp LSTM với các tham số unit cho phù hợp với chuỗi dữ liệu đầu vào, và thêm 1 lớp Dense với các giá trị 1 để cho kết quả đầu ra phù hợp. Tại đây chúng tôi sử dụng các giá trị bước thời gian tương ứng và các cột feature là các cột giá trị miền dữ liệu chúng tôi tiền xử lý dữ liệu để dự đoán. Chúng tôi sẽ sử dụng optimizer là adam và loss function dùng để tối ưu là mean square error (MSE).

## **3.3 Mô hình xử lý dữ liệu**

### **3.3.1 Các miền dữ liệu**

Ngoài các miền dữ liệu cơ bản của chứng khoán đã được chúng tôi giới thiệu sơ lược qua ở chương phía trên. Chúng tôi cũng tiến hành nghiên cứu các chỉ số, chỉ báo chứng khoán phù hợp để có thể tăng hiệu quả dự đoán cũng như cho ra được kết quả dự đoán chính xác. Các chỉ báo, chỉ số mà chúng tôi nghiên cứu và sử dụng là EMA, RSI, MACD, ROC, ATR, OBV, CMF, EMV, STOCH, MFI, CCI.

Trong đó, EMA (Exponential Moving Average)<sup>9</sup> là một công cụ phân tích kỹ thuật được dùng trong thị trường chứng khoán. Đây là một dạng thông số trung bình di động được tính toán dựa vào giá đóng cửa của một chứng khoán trong một khoảng thời gian nhất định. Giá trị trung bình này được tính dựa vào hệ số smoothing (hay còn gọi là alpha) có giá trị từ 0 đến 1. Hệ số này quyết định độ ưu tiên của giá trị mới so với các giá trị cũ khi đang tính toán trung bình. Giá trị EMA mới được tính bằng công thức như sau:

$$EMA_t = \alpha * X_t + (1 - \alpha) * EMA_{t-1}$$

Trong đó:

$EMA_t$ : Là giá trị EMA ở thời điểm t.

$X_t$ : Là giá trị thực tế tại thời điểm t.

$EMA_{t-1}$ : Là giá trị EMA ở thời điểm t-1.

$\alpha$ : Là hệ số smoothing ( $0 < \alpha < 1$ ).

Ý nghĩa khi áp dụng chỉ số EMA này là giúp làm mịn cho dữ liệu đầu vào và điều này làm giảm ảnh hưởng của giá trị nhiễu đột ngột trong việc dự báo với dữ liệu chuỗi thời gian.

Tiếp theo, MACD (Moving Average Convergence Divergence)<sup>10</sup> là một chỉ báo kỹ thuật khá phổ biến trong phân tích kỹ thuật chứng khoán, được sử dụng cho việc đo lường sự chênh lệch giữa hai đường trung bình để dự đoán xu hướng tăng hay giảm từ dữ liệu thực tế. Cấu trúc cơ bản để tiếp cận của MACD bao gồm đường MACD với công thức tính như sau:

$$\text{MACD Line} = EMA(14) - EMA(26)$$

Đường MACD thể hiện sự chênh lệch giữa hai đường trung bình EMA có độ chậm và độ nhanh trong một khoảng thời gian được quy định. Nó được tính bằng cách trừ một đường trung bình EMA ngắn hạn với một đường trung bình EMA dài hạn.

---

<sup>9</sup> <https://vnexpress.net/duong-ema-la-gi-4493726.html>

<sup>10</sup> <https://topi.vn/duong-macd.html>

Sau khi có đường MACD thì ta có thể tính đường Signal theo công thức sau:

$$\text{Signal Line} = \text{EMA}(\text{MACD Line}, 9)$$

Đây là một đường trung bình di động dạng báo hiệu, được tính từ đường trung bình EMA của đường MACD.

Sự chênh lệch giữa MACD Line và Signal Line sẽ báo hiệu được những tín hiệu từ đó có thể dự đoán được xu hướng thị trường chứng khoán. Khi MACD Line cắt trên Signal Line thì có thể xem xét được đây là dấu hiệu quá mua thì xu hướng sẽ tăng. Ngược lại, khi MACD Line cắt xuống dưới Signal Line thì có thể đây là dấu hiệu quá bán thì xu hướng sẽ giảm.

Tiếp đến là chỉ số OBV (On-Balance Volume)<sup>11</sup>, có thể gọi là Khối Lượng Cân Đôi. Đây là một chỉ số trong phân tích kỹ thuật chứng khoán được sử dụng để đo lường áp lực mua và bán trên thị trường. Chỉ số này theo dõi được sự biến động của khối lượng giao dịch theo một khoảng thời gian nhất định. Nếu giá đóng cửa chốt phiên giao dịch cao hơn giá đóng cửa chốt phiên trước đó, thì khối lượng giao dịch sẽ được thêm vào chỉ số OBV của hôm đó. Ngược lại, thì khối lượng giao dịch sẽ được trừ đi khỏi chỉ số OBV. Còn nếu giá đóng cửa chốt không có sự thay đổi giữa hai ngày liên tiếp thì sẽ giữ nguyên giá trị OBV. Sau đây là công thức của cách tính OBV.

- Nếu giá đóng cửa chốt hiện tại lớn hơn giá đóng cửa chốt trước đó thì chỉ số OBV hiện tại sẽ bằng chỉ số OBV trước đó cộng thêm cho khối lượng giao dịch.

$$\text{OBV} = \text{OBV trước đó} + \text{Khối lượng giao dịch hiện tại}$$

- Nếu giá đóng cửa chốt hiện tại mà nhỏ hơn giá đóng cửa chốt trước đó thì chỉ số OBV hiện tại sẽ bằng chỉ số OBV trước đó trừ đi cho khối lượng giao dịch.

$$\text{OBV} = \text{OBV trước đó} - \text{Khối lượng giao dịch hiện tại}$$

---

<sup>11</sup> <https://topi.vn/obv-la-gi.html>

- Nếu giá đóng cửa chốt hiện tại mà không đổi gì so với giá đóng cửa chốt trước đó thì OBV sẽ giữ nguyên giá trị.

$$\text{OBV} = \text{OBV} \text{ trước đó}$$

Khi giá đóng cửa chốt tăng và chỉ số OBV cũng tăng thì cho thấy dấu hiệu rằng xu hướng giá đang tăng trên thị trường chứng khoán. Ngược lại, nếu giá đóng cửa chốt giảm và chỉ số OBV cũng giảm thì cho thấy rằng xu hướng giá đang giảm trên thị trường chứng khoán.

Tiếp theo là chỉ số EMV (Ease of Movement)<sup>12</sup> là một công cụ hữu ích trong công cuộc phân tích kỹ thuật chứng khoán với mục đích đo lường sự dễ dàng của giá chứng khoán trong việc di chuyển lên hay xuống. Công thức chính của chỉ số EMV là:

$$EMV = \frac{(H + L)/2 - (H' + L')/2}{((V/1000000) / (H - L))}$$

Trong đó:

H: Là giá cao nhất của phiên giao dịch hiện tại.

L: Là giá thấp nhất của phiên giao dịch hiện tại.

H': Là giá cao nhất của phiên giao dịch trước đó.

L': Là giá thấp nhất của phiên giao dịch trước đó.

V: Là khối lượng giao dịch hiện tại.

Chỉ số EMV này không hẳn là một công cụ phổ biến như các chỉ số kỹ thuật chứng khoán khác nhưng nếu giá trị EMV cao thì có thể chỉ ra được xu hướng tăng mạnh trong khi đó nếu giá trị EMV thấp thì có thể là biểu hiện của xu hướng giảm.

Kế đến chỉ số MFI (Money Flow Index)<sup>13</sup>, đây là một chỉ số kỹ trong nghiệp vụ phân tích kỹ thuật chứng khoán được dùng để đo lường áp lực mua bán trên thị trường. Chỉ số này dựa trên ý tưởng về sự cân bằng giữa dòng tiền vào và dòng tiền ra của một chứng khoán trong một khoảng thời gian cụ thể.

---

<sup>12</sup> <https://www.qmcapital.vn/bai-viet/chi-bao-ky-thuat-ease-of-movement>

<sup>13</sup> <https://www.dsc.com.vn/kien-thuc/chi-bao-dong-tien-mfi-la-gi>

Đầu tiên nó sẽ kết hợp trung bình ba loại giá trong thị trường chứng khoán là giá cao nhất, giá thấp nhất, giá đóng cửa. Sau ta sẽ tính dòng tiền tích lũy bằng cách so sánh giữa giá đóng cửa hiện tại với giá đóng cửa trước đó và nhân nó với khối lượng giao dịch.

Tiếp theo, chúng tôi sẽ tính dòng tiền khi giá tăng và dòng tiền khi giá giảm và tính tỷ lệ giữa dòng tiền của giá tăng và dòng tiền giữa giá giảm đó trước khi tính được chỉ số MFI (Money Flow Index). Các công thức như phân tích trên sẽ được thể hiện sau đây:

$$\text{Money\_Flow} = \text{Combined\_Price} * \text{Volume}$$

$$(\text{Combined\_Price} = (\text{High} + \text{Low} + \text{Close}) / 3)$$

$$\text{Positive\_Money\_Flow} = \sum \text{Money\_Flow} \text{ (Khi giá tăng)}$$

$$\text{Negative\_Money\_Flow} = \sum \text{Money\_Flow} \text{ (Khi giá giảm)}$$

$$\text{Money\_Ratio} = \text{Positive\_Money\_Flow} / \text{Negative\_Money\_Flow}$$

$$\text{MFI} = 100 - \frac{100}{1 + \text{Money\_Ratio}}$$

Khi chỉ số MFI lớn hơn 80 thì cho thấy rằng thị trường chứng khoán đang trong tình trạng quá mua, có thể đây là dấu hiệu cho thấy sự đảo chiều từ xu hướng giảm thành xu hướng tăng. Ngược lại nếu chỉ số MFI nhỏ hơn 20 thì cho thấy rằng thị trường chứng khoán đang trong tình trạng quá bán, có thể đây là dấu hiệu cho thấy sự đảo chiều từ xu hướng tăng thành xu hướng giảm.

Tiếp đến, RSI (Relative Strength Index)<sup>14</sup> là một chỉ báo động lượng dùng để xác định xu hướng của chứng khoán, là một chỉ báo thông dụng được sử dụng rộng rãi và được biểu diễn dưới dạng dao động từ 0 đến 100. Chỉ báo này được tính qua 2 bước:

$$RS(\text{Relative Strength}) = (\text{average gain}) / (\text{average loss})$$

$$RSI = 100 - 100 / (1 + RS)$$

---

<sup>14</sup> <https://www.vietcap.com.vn/kien-thuc/ung-dung-rsi-trong-dau-tu-chung-khoan-hieu-qua>

Ý nghĩa của chỉ báo này là chứng khoán sẽ được cho là quá bán khi dao động dưới vùng 30 điểm và chứng khoán được cho là quá mua khi dao động trên vùng 70 điểm.

Tiếp theo, ROC (Rate Of Change)<sup>15</sup> là một chỉ số kĩ thuật tính toán dựa vào tỷ lệ phần trăm thay đổi giá giữa giá hiện tại với giá của một khoảng thời gian trước đó để đo lường được tốc độ biến thiên của giá cả trong hai thời điểm khác nhau. Như đã giải thích thì công thức của chỉ số này sẽ có dạng như sau:

$$\begin{aligned} ROC = & [(Giá đóng cửa hiện tại - Giá đóng cửa n phiên trước) \\ & / Giá đóng cửa n phiên trước] * 100 \end{aligned}$$

Ý nghĩa của chỉ số ROC này giúp chúng ta nhận biết được giá của các công ty đang có tín hiệu tăng hay giảm. Ngoài ra, chúng ta còn biết được mức độ tăng giảm của nó ra sao nếu chúng ta hiểu rằng nếu chỉ số này tính ra dưới 0 thì sẽ dự đoán được rằng giá kì này sẽ giảm mạnh và ngược lại thì giá sẽ tăng mạnh.

Tiếp đến là ATR (Average True Range)<sup>16</sup> là một chỉ số kĩ thuật giúp xác định khoảng giao động thực tế trung bình. Chỉ số này được sử dụng để đo lường biến động giá của các công ty trong một khoảng thời gian cụ thể. Giải thích về công thức tính toán chỉ số này thì trước tiên chúng ta sẽ phải tính sự chênh lệch giữa giá cao nhất (đỉnh) và giá thấp nhất (đáy), thứ hai thì sẽ tính giá cao nhất (đỉnh) trừ đi giá đóng cửa trước đó, cuối cùng thì sẽ tính giá thấp nhất (đáy) trừ đi giá đóng cửa trước đó. Sau khi có ba giá trị đã tính toán như trên thì chúng ta tìm ra giá trị lớn nhất của ba giá trị đó để đem đi tính trung bình động với một khoảng thời gian nhất định. Kết thúc quá trình đó chúng ta sẽ có được một chỉ số kĩ thuật như trên.

Ý nghĩa của chỉ số ATR này là sự tăng lên mạnh mẽ của nó, nếu thị trường càng biến động thì chỉ số này càng cao và điều này giúp cho các nhà đầu tư đánh giá được

---

<sup>15</sup> <https://www.dNSE.com.vn/hoc/chi-bao-roc>

<sup>16</sup> <https://topi.vn/chi-bao-atr.html>

mức độ rủi ro thông qua chỉ số này và từ đó họ cũng có thể xác định hợp lý các điểm vào và điểm ra của thị trường để tránh mất mát nhiều tài sản khi tham gia.

Ké đến là chỉ số CMF (Chaikin Money Flow)<sup>17</sup>, được tạo ra để theo dõi sự tích lũy và phân phói giá trong một khoảng thời gian nhất định. Chỉ số này sẽ có giá trị giao động từ -1 đến +1. Chỉ số này được tính toán theo ba bước, đầu tiên là bước tính hệ số dòng tiền theo công thức sau:

$$\text{Hệ số dòng tiền} = ((\text{Giá đóng cửa} - \text{Giá thấp nhất}) - (\text{Giá cao nhất} - \text{Giá đóng cửa})) / (\text{Giá cao nhất} - \text{Giá thấp nhất})$$

Bước tiếp theo là bước tính khối lượng dòng tiền, ta lấy hệ số dòng tiền vừa tính được nhân với khối lượng giao dịch trong cùng thời điểm đó.

Bước cuối cùng thì ta sẽ tính giá trị CMF bằng cách chia khối lượng dòng tiền trong một khoảng thời gian nhất định với tổng khối lượng giao dịch trong cùng một khoảng thời gian.

Sau khi tính được chỉ số CMF thì chúng ta có thể dựa vào đó để xác định xu hướng là tăng nếu như giá trị của CMF được duy trì trên 0. Ngược lại, nếu có hiện tượng giá trị CMF này được duy trì dưới 0 thì ta có thể xác định rằng xu hướng là giảm.

Tiếp theo đó là chỉ số STOCH (Stochastic Oscillator)<sup>18</sup> là một chỉ báo động lượng so sánh giá đóng cửa với phạm vi giá của một mã chứng khoáng trong một khoảng thời gian nhất định. Để có thể tính toán chỉ số này thì chúng ta chỉ cần tuân thủ hai bước sau, đầu tiên chúng ta sẽ tìm giá trị thấp nhất trong tất cả giá thấp nhất của một mã chứng khoán và giá trị cao nhất trong tất cả giá cao nhất của một mã chứng khoán. Hai giá trị này được xem là phạm vi giá của một mã chứng khoán. Sau khi tìm ra được hai giá trị này thì chúng ta tính theo công thức sau:

$$STOCH = ((\text{Giá đóng cửa} - \text{Giá thấp nhất}) / (\text{Giá cao nhất} - \text{Giá thấp nhất})) * 100$$

---

<sup>17</sup> <https://ysradar.yuanta.com.vn/blog/phan-tich-ky-thuat/tong-quan-ve-chaikin-money-flow-cmf/>

<sup>18</sup> <https://topi.vn/stochastic-la-gi-cach-su-dung-chi-bao-stochastic.html>

Chỉ báo STOCH này chỉ ra được tình trạng quá mua hay quá bán bằng cách xem xét quả cuối cùng của nó. Nếu giá trị này nằm trên 80 thì thị trường đang ở trạng thái quá mua (Overbought), còn nếu giá trị này nằm dưới 20 thì thị trường đang ở tình trạng quá bán (Oversold).

Cuối cùng, chỉ số CCI (Commodity Channel Index)<sup>19</sup> là một chỉ báo kỹ thuật giúp đo lường biến động giá cả trên thị trường chứng khoán của các công ty. Để có thể tính toán được chỉ số CCI này thì chúng ta có thể tính dựa trên việc đo lượng mối liên hệ giữa các giá của một mã chứng khoán và đường trung bình cộng của chúng. Ta có thể hiểu đơn giản thì đây là chỉ số thể hiện độ lệch so với mức trung bình giá, và công thức như sau:

$$CCI = (Tổng giá - Giá trung bình) / (\text{Độ lệch trung bình} * 0.015)$$

Chỉ số CCI này giúp cho những nhà kinh doanh, nhà đầu tư xác định được xu hướng của thị trường. Nếu giá trị này chạy từ 0 đến 100 thì xu hướng giá của thị trường đang tăng và nếu giá trị này chạy từ -100 đến 0 thì xu hướng giá của thị trường đang giảm.

### **3.3.2 Tái cấu trúc dữ liệu của các công ty**

Với dữ liệu đầu vào là dữ liệu các giá chứng khoán của các công ty tại Việt Nam. Chúng tôi sẽ tiến hành làm mịn dữ liệu bằng cách thay thế bằng các giá trị trung bình và sau đó sẽ tính phần trăm thay đổi giá hằng ngày của giá đóng cửa. Khi đã có được giá trị này thì chúng tôi sẽ tiếp tục tạo ra các cột dữ liệu khác lùi lại từ 1 đến 5 ngày để làm dữ liệu đầu vào mới thay cho các cột dữ liệu ban đầu. Điều này giúp mô hình học được sự thay đổi giá giữa các ngày liên tiếp để đưa ra được dự đoán về xu hướng trên sàn chứng khoán của các công ty tại Việt Nam. Bên cạnh đó, chúng tôi cũng sẽ tính toán đến các chỉ số, chỉ báo chứng khoán cần thiết để đưa ra dự đoán như là EMA, RSI, MACD, ROC, ATR, OBV, CMF, EMV, STOCH, MFI, CCI. Sau đó chúng tôi sẽ trích chọn các đặc

---

<sup>19</sup> <https://topi.vn/chi-so-cci.html>

trung như phần trăm thay đổi các ngày được tính toán ở trên, các chỉ số, chỉ báo chứng khoán để đưa vào huấn luyện cho mô hình.

### **3.3.3 Tái cấu trúc dữ liệu tổng hợp các công ty**

Chúng tôi cũng tiến hành việc tổng hợp lại giá chứng khoán của 10 công ty mà chúng tôi chọn lại thành một tượng trưng cho thị trường chứng khoán Việt Nam. Chúng tôi cũng theo dõi và thực hiện việc xử lý độ dài của các công ty để có thể hoàn thiện một miền dữ liệu chứng khoán tổng hợp. Sau đó, chúng tôi cũng thực hiện việc tính toán các chỉ số, chỉ báo chứng khoán để thêm vào miền dữ liệu và trích chọn các đặc trưng tương tự như phần tái cấu trúc dữ liệu của các công ty để đưa vào huấn luyện mô hình.

### **3.3.4 Tái cấu trúc dữ liệu của các công ty theo tháng**

Ở đây, dữ liệu đầu vào là dữ liệu các giá chứng khoán của các công ty tại Việt Nam, chúng tôi thực hiện việc tái cấu trúc của các công ty thành 4 tuần và thành một tháng. Ngoài ra, chúng tôi cũng có một ý tưởng rằng tiến hành việc tính toán các chênh lệch giữa các tuần và thêm vào miền dữ liệu. Qua đó ta có thể dự đoán được tốt hơn về xu hướng của chứng khoán. Chúng tôi cũng tiến hành làm mịn dữ liệu, tính toán các chỉ số, chỉ báo EMA, RSI, MACD, OBV, EMV, MFI và thêm vào miền dữ liệu. Tại đây, chúng tôi trích chọn các cột dữ liệu là các chỉ báo, chỉ số chứng khoán, các chênh lệch giữa các tuần để làm đặc trưng cho việc huấn luyện mô hình.

### **3.3.5 Tái cấu trúc dữ liệu tổng hợp các công ty theo tháng**

Tại đây, chúng tôi thực hiện việc kết hợp tái cấu trúc dữ liệu tổng hợp các công ty và tái cấu trúc dữ liệu theo tháng. Chúng tôi cũng tiến hành tổng hợp lại giá chứng khoán của 10 công ty, xây dựng lại độ dài và thực hiện tái cấu trúc thành 4 tuần cũng như một tháng. Thực hiện các tính toán các chỉ số, chỉ báo để thêm vào miền dữ liệu, trích chọn các đặc trưng cho việc huấn luyện mô hình tương tự như phần tái cấu trúc dữ liệu của các công ty theo tháng.

## CHƯƠNG 4 – THỰC NGHIỆM

### 4.1 Xây dựng dữ liệu

#### 4.1.1 Thu thập dữ liệu

Đầu tiên, chúng tôi sẽ lấy dữ liệu vnquant<sup>20</sup> từ github, đây là dữ liệu đã được thu thập trên thị trường chứng khoán của các công ty tại Việt Nam. Thao tác lấy dữ liệu như trên sẽ tương ứng với việc chúng tôi sẽ cài một thư viện vào phần lập trình để có thể lấy api của dữ liệu chứng khoán trước khi xử lý dữ liệu.

```
▼ Install needed libraries

[ ] import os
from google.colab import drive
drive.mount('/content/gdrive')

path = "/content/gdrive/MyDrive/Chuyên đề nghiên cứu 1/vnquant_package"
%cd {path}
!ls

▶ !git clone https://github.com/phamdinhkhanh/vnquant
%cd vnquant
!python setup.py install

[ ] %cd ..
!rm -rf vnquant
!ls

[ ] !pip freeze | grep vnquant
vnquant==0.1.1

[ ] import vnquant
vnquant.__version__

'0.1.1'
```

Hình 4. 1 Cài đặt dữ liệu vnquant từ github

Chúng tôi sẽ xác định một chuỗi các mã cổ phiếu của các công ty mà chúng tôi sẽ tiến hành thu thập dữ liệu chứng khoán.

---

<sup>20</sup> <https://github.com/phamdinhkhanh/vnquant>

▼ We choose 10 codes from biggest companies on stock exchange in Vietnam

```
[ ] stock_symbols = ['VIC', 'VCB', 'VNM', 'GAS', 'HVN', 'MMG', 'FPT', 'HPG', 'MSN', 'SAB']
```

Hình 4. 2 Xác định các mã chứng khoán

Sau đó, sau khi đã có được các mã chứng khoán, chúng tôi tiến hành tạo ra một tập hợp bao gồm dữ liệu chứng khoán từ tất cả các mã này với thời gian là từ ngày 1 tháng 1 năm 2014 đến 1 tháng 1 năm 2024 và nguồn dữ liệu được lấy từ VNINDEX.

▼ Prepare the dataset

```
[ ] stock = {}
for i in range(len(stock_symbols)):
    data = dt.DataLoader(symbols=stock_symbols[i],
                          start="2014-01-01",
                          end="2024-01-01",
                          data_source="VND")
    stock[stock_symbols[i]] = data.download()
```

▶ stock[stock\_symbols[0]]

	Attributes	high	low	open	close	avg	volume
Symbols	VIC	VIC	VIC	VIC	VIC	VIC	
	date						
2014-01-02	70.50	69.50	70.00	70.00	69.92	74950.0	
2014-01-03	70.50	69.50	70.00	70.50	70.00	91920.0	
2014-01-06	70.50	69.50	70.50	70.50	70.11	111760.0	
2014-01-07	70.50	70.00	70.50	70.50	70.47	294120.0	
2014-01-08	70.50	70.00	70.00	70.00	70.02	109890.0	

Hình 4. 3 Tạo một tập hợp gồm dữ liệu của các công ty

#### 4.1.2 Trực quan hóa dữ liệu

Sau khi lấy dữ liệu chứng khoán thành công thì các dữ liệu chứng khoán của các công ty tại Việt Nam được chứa trong một tập hợp chung và khi cần lấy dữ liệu thì chỉ cần gọi đến dữ liệu đó đi kèm với mã chứng khoán công ty tương ứng thì nó sẽ trả ra kết quả mà mình mong muốn.

Explore and visualize the datasets								
[12] stock_data								
{ 'VIC': Attributes high low open close avg volume								
Symbols VIC VIC VIC VIC VIC VIC								
date								
2014-01-02 70.50 69.50 70.00 70.00 69.92 74950.0								
2014-01-03 70.50 69.50 70.00 70.50 70.00 91920.0								
2014-01-06 70.50 69.50 70.50 70.50 70.11 111760.0								
2014-01-07 70.50 70.00 70.50 70.50 70.47 294120.0								
2014-01-08 70.50 70.00 70.00 70.00 70.02 109890.0								
... ... ... ... ... ... ...								
2023-12-25 43.55 43.00 43.10 43.40 43.34 3364500.0								
2023-12-26 43.75 43.35 43.40 43.55 43.55 1806700.0								
2023-12-27 43.95 43.60 43.65 43.60 43.76 1920500.0								
2023-12-28 44.60 43.60 43.60 44.45 44.32 4359700.0								
2023-12-29 44.85 44.45 44.65 44.60 44.63 2474300.0								
[2494 rows x 6 columns],								
'VCB': Attributes high low open close avg volume								
Symbols VCB VCB VCB VCB VCB VCB								
date								
2014-01-02 26.9 26.6 26.9 26.7 26.74 333900.0								
2014-01-03 27.3 26.7 26.7 27.2 27.07 813360.0								
2014-01-06 27.4 27.0 27.3 27.1 27.17 375890.0								
2014-01-07 27.9 27.4 27.5 27.7 27.68 2372530.0								
2014-01-08 28.1 27.6 27.8 28.1 27.86 1474370.0								
... ... ... ... ... ... ...								

Hình 4. 4 Dữ liệu sau khi được tập hợp

Tiếp đến chúng tôi sẽ gọi ra tập hợp khoảng thời gian mà dữ liệu chứng khoán của các công ty được gọi tới. Như đã quy định từ đầu là khoảng thời gian sẽ từ ngày 1 tháng 1 năm 2014 đến ngày 1 tháng 1 năm 2024.

[13] stock_data[stock_symbols[7]].index
DatetimeIndex(['2014-01-02', '2014-01-03', '2014-01-06', '2014-01-07', '2014-01-08', '2014-01-09', '2014-01-10', '2014-01-13', '2014-01-14', '2014-01-15', ..., '2023-12-18', '2023-12-19', '2023-12-20', '2023-12-21', '2023-12-22', '2023-12-25', '2023-12-26', '2023-12-27', '2023-12-28', '2023-12-29'], dtype='datetime64[ns]', name='date', length=2494, freq=None)

Hình 4. 5 Tập hợp các khoảng thời gian sẽ lấy dữ liệu

Ngoài ra, ta cũng có thể theo dõi dữ liệu chứng khoán bằng cách sử dụng các hàm để gọi ra được thông tin chi tiết cũng như mô tả dữ liệu để hiểu thêm về dữ liệu mà chúng ta đã lấy về.

```
▶ stock_data[stock_symbols[5]].info()

[1]: <class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 2368 entries, 2014-07-14 to 2023-12-29
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   (high, MWG)    2368 non-null   float64
 1   (low, MWG)     2368 non-null   float64
 2   (open, MWG)    2368 non-null   float64
 3   (close, MWG)   2368 non-null   float64
 4   (avg, MWG)    2368 non-null   float64
 5   (volume, MWG)  2368 non-null   float64
dtypes: float64(6)
memory usage: 129.5 KB
```

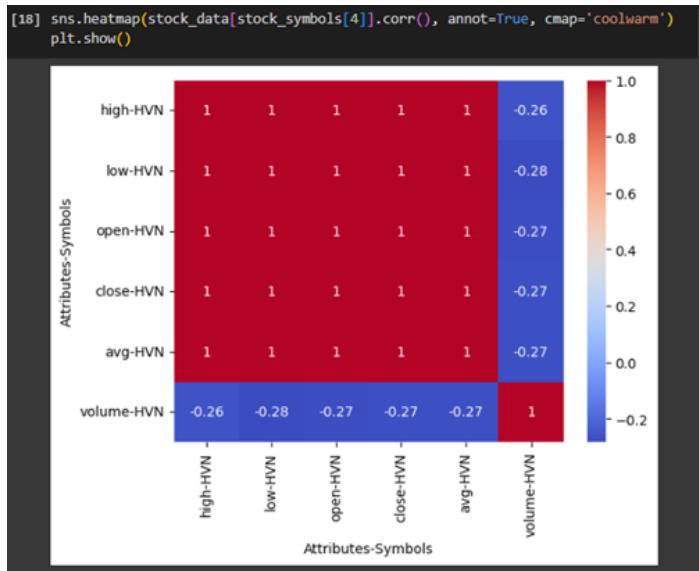
Hình 4. 6 Thông tin chi tiết của dữ liệu chứng khoán

Ngoài ra, ta cũng có thể theo dõi dữ liệu chứng khoán bằng cách sử dụng các hàm để gọi ra được thông tin chi tiết cũng như mô tả dữ liệu để hiểu thêm về dữ liệu mà chúng ta đã lấy về.

stock_data[stock_symbols[1]].describe()						
	Attributes	high	low	open	close	avg
Symbols		VCB	VCB	VCB	VCB	VCB
count	2494.000000	2494.000000	2494.000000	2494.000000	2494.000000	2.494000e+03
mean	64.697253	63.161167	63.906736	63.955994	63.927551	1.189667e+06
std	24.395285	23.858451	24.153036	24.163703	24.123859	8.201846e+05
min	24.300000	23.500000	23.800000	23.800000	23.910000	3.638000e+04
25%	40.600000	39.625000	40.000000	40.100000	40.170000	6.474900e+05
50%	65.650000	63.250000	64.550000	64.400000	64.340000	9.970150e+05
75%	86.000000	84.800000	85.400000	85.375000	85.427500	1.498182e+06
max	117.200000	115.000000	116.700000	116.400000	116.120000	7.265480e+06

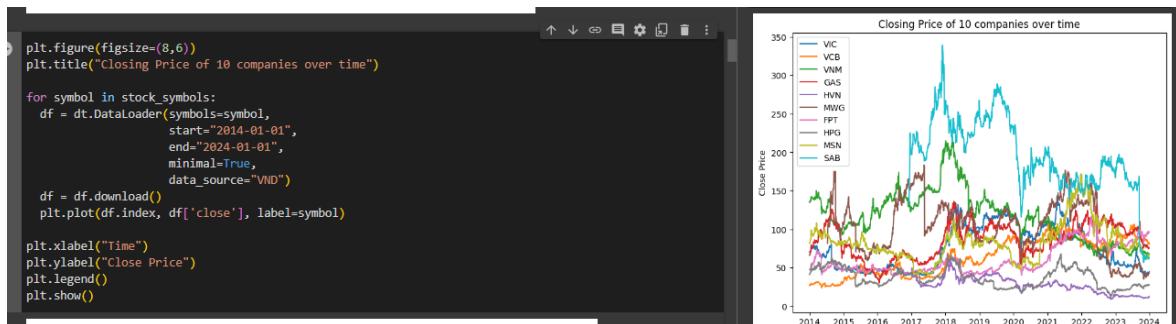
Hình 4. 7 Mô tả cụ thể của dữ liệu chứng khoán

Mức độ tương quan của các cột dữ liệu này cũng rất quan trọng vì khi ta biết được sự liên quan mật thiết của các cột dữ liệu thì ta có thể sử dụng nó để tính được các số liệu có giá trị hơn phục vụ cho việc huấn luyện mô hình để có được hiệu suất tốt nhất. Chúng tôi sẽ biểu thị độ tương quan đó thông qua hình sau.



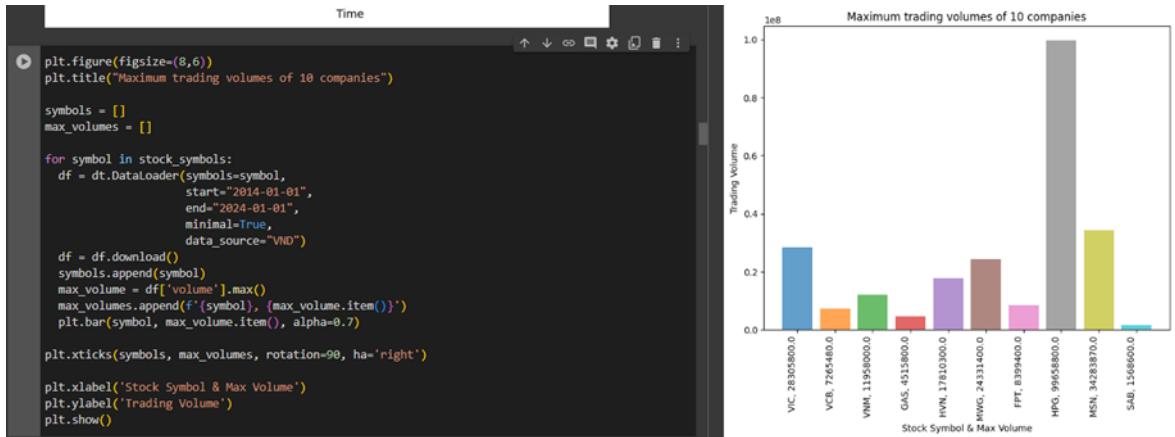
Hình 4. 8 Biểu đồ thể hiện độ tương quan của các cột trong dữ liệu chứng khoán

Hình tiếp theo sẽ thể hiện biểu đồ đường cho giá đóng cửa của các công ty tại Việt Nam qua toàn bộ khoảng thời gian từ ngày 1 tháng 1 năm 2014 đến ngày 1 tháng 1 năm 2024.

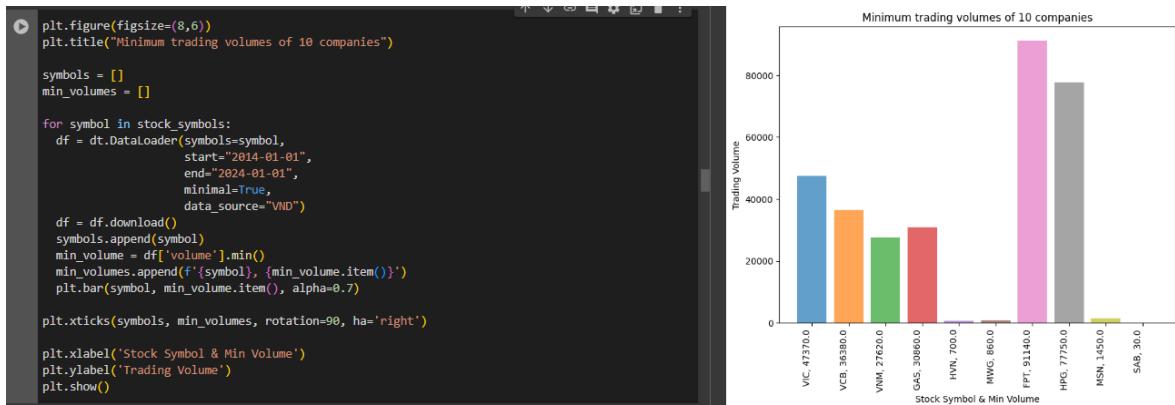


Hình 4. 9 Biểu đồ đường thể hiện giá đóng cửa của các công ty

Sau đó chúng tôi sẽ tiếp tục trực quan hóa thêm các biểu đồ về khối lượng giao dịch cao nhất và khối lượng giao dịch thấp nhất của các công ty tại Việt Nam. Những biểu đồ này để ta hiểu thêm về dữ liệu chứng khoán mà ta đã thu thập và ta cũng có thể tự đánh giá lại được mức độ phát triển của các công ty trên thị trường chứng khoán.



Hình 4. 10 Biểu đồ cột thể hiện khối lượng giao dịch cao nhất của các công ty



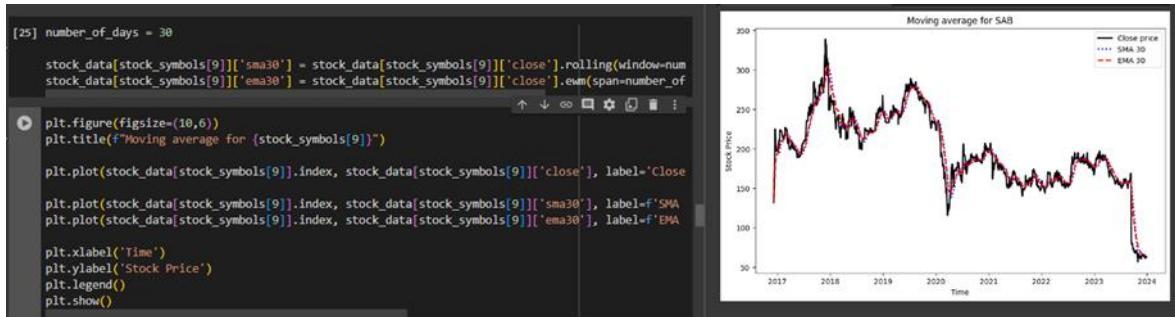
Hình 4. 11 Biểu đồ cột thể hiện khối lượng giao thấp nhất của các công ty

Kết quả là biểu đồ nến biểu diễn xu hướng tăng giảm của các công ty thông qua những thông tin chi tiết của các giá như giá đóng cửa, giá mở cửa, giá cao nhất, giá thấp nhất trong một khoảng thời gian với dữ liệu cụ thể. Ta có thể sử dụng biểu đồ này để quan sát sự biến động giá của các công ty trên thị trường chứng khoán tại Việt Nam.



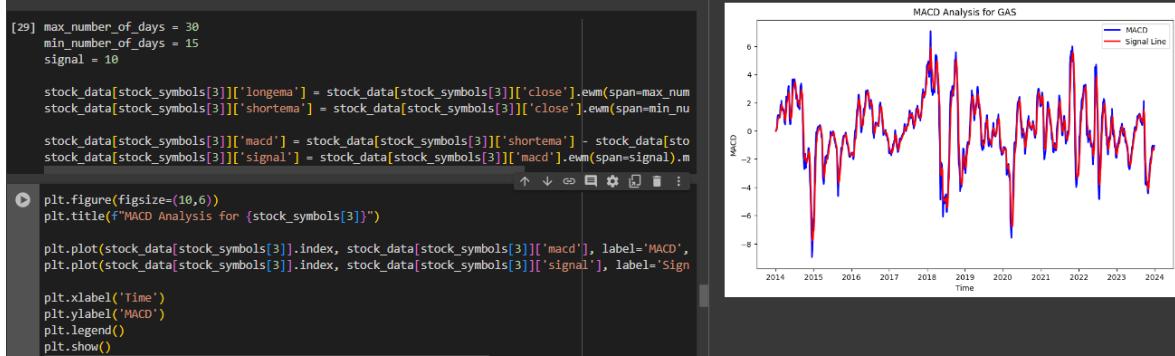
Hình 4. 12 Biểu đồ nền biểu diễn các giá của các công ty

Tiếp đến chúng tôi sẽ tính trung bình trượt đơn giản (SMA) và trung bình trượt mở rộng (EMA) của giá đóng cửa trong dữ liệu chứng khoán để có thể quan sát được mượt mà hơn so với dữ liệu khi chưa sử dụng công thức tính trung bình.



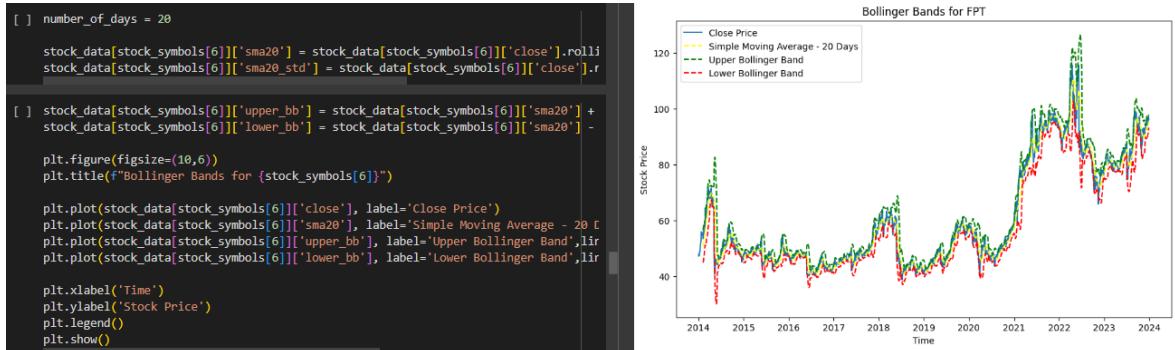
Hình 4. 13 Biểu đồ đường so sánh dữ liệu bình thường với dữ liệu có sử dụng trung bình trượt đơn giản và trung bình trượt mở rộng

Sau khi đã theo dõi được sự thay đổi của dữ liệu khi không sử dụng trung bình trượt và dữ liệu đã sử dụng trung bình trượt thì biểu đồ tiếp theo sẽ cho chúng ta biết được sự khác biệt giữa hai đường trung bình trượt mở rộng (EMA). Để dễ quan sát thì biểu đồ sẽ tính ra hai đường chính là đường MACD và đường Signal, nếu đường MACD cắt lên trên đường Signal thì đây có thể là dấu hiệu quá mua và xu hướng sẽ có thể tăng. Ngược lại, nếu đường MACD cắt dưới đường Signal thì đây có thể là dấu hiệu quá bán và xu hướng sẽ có thể giảm.



Hình 4. 14 Biểu đồ đường thể hiện sự khác biệt giữa hai đường trung bình trượt mở rộng

Tiếp theo, biểu đồ Bollinger Bands biểu diễn sự kết hợp của đường trung bình động MA (Moving Average) và độ lệch chuẩn. Biểu đồ này gồm có bốn đường cơ bản, một đường thể hiện giá đóng cửa, một nữa là đường trung bình động, và hai đường còn lại là hai đường biên trên và dưới. Ta có thể sử dụng biểu đồ này để xác định xu hướng và dự đoán xem được mã công ty đó có đang bị quá mua hay quá bán hay không.



Hình 4. 15 Biểu đồ Bollinger Bands

Cuối cùng, còn có một chỉ số trong phân tích chứng khoán cũng cực kỳ quan trọng để có thể quan sát xu hướng tăng giảm của chứng khoán trên thị trường, đó là chỉ số RSI được biểu diễn như hình dưới. Chỉ số này sẽ tính độ mạnh yếu của xu hướng, sau khi tính toán và biểu diễn chỉ số RSI lên đồ thị thì ta sẽ tiếp tục vẽ hai đường thẳng là 70 và 30. Nếu đường RSI cắt ở trên đường thẳng 70 thì ta có thể xem đó là tín hiệu của xu hướng tăng và nếu đường RSI cắt ở dưới đường thẳng 30 thì ta có thể xem đây là tín hiệu của xu hướng giảm.

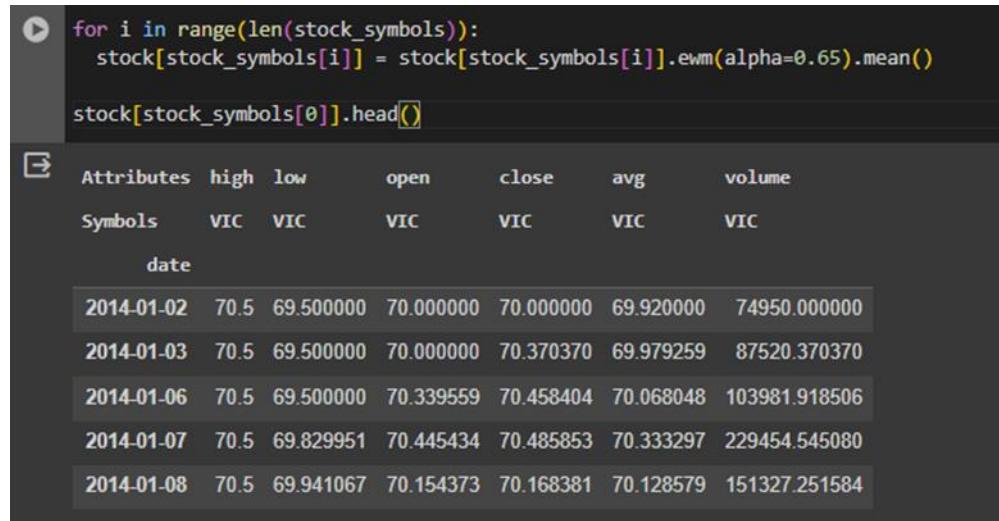


Hình 4. 16 Biểu đồ đường thể hiện độ mạnh hay yếu của xu hướng trên thị trường chứng khoán

Kết thúc quá trình trực quan hóa dữ liệu, chúng tôi đã có cái nhìn rõ ràng hơn về dữ liệu trên sàn chứng khoán và bước tiếp theo của chúng tôi là sẽ xây dựng những cột dữ liệu mới có tính chuyên sâu vào dữ liệu để phục vụ cho quá trình huấn luyện và đưa ra dự đoán về xu hướng trên sàn chứng khoán.

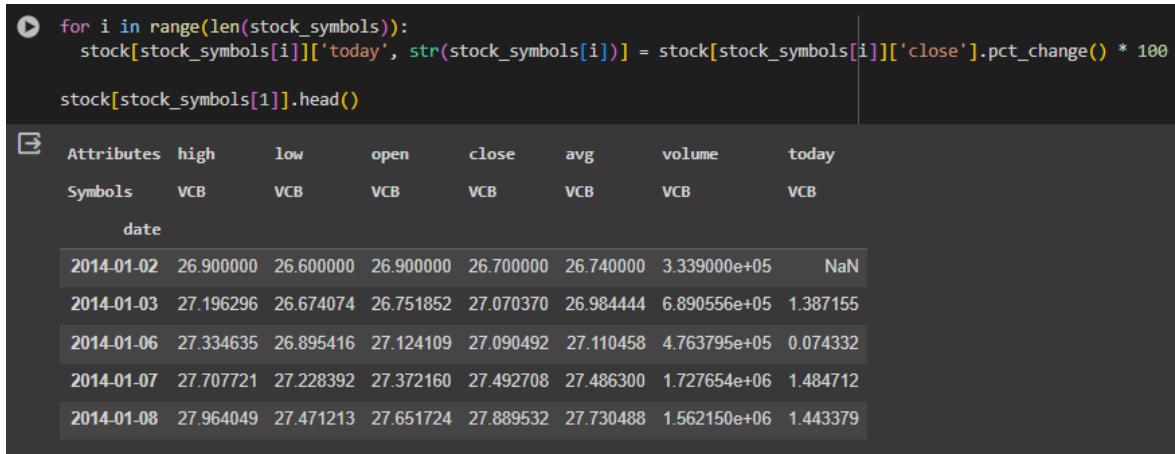
### 4.1.3 Tiết xử lý dữ liệu

#### 4.1.3.1 Tái cấu trúc dữ liệu của các công ty



Hình 4. 17 Làm mịn toàn bộ dữ liệu chứng khoán

Dòng lệnh trên giúp chúng tôi làm mịn dữ liệu đầu bằng cách tính trung bình trượt mở rộng với hệ số alpha là 0.65, nó thực hiện việc tăng cường trọng số của các giá trị gần nhất để khi tính toán hay khi đưa trên đồ thị thì sẽ có được sự mượt mà nhất định cho các cột dữ liệu chứng khoán.



```

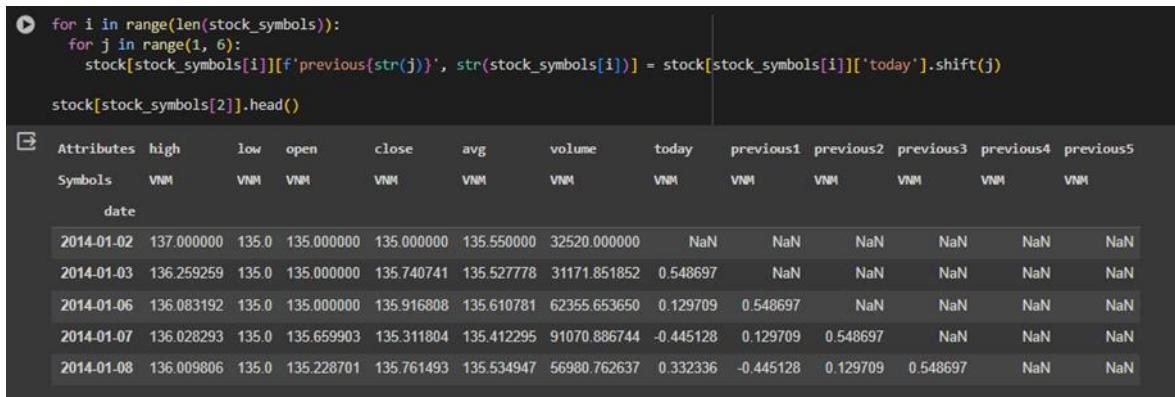
for i in range(len(stock_symbols)):
    stock[stock_symbols[i]]['today', str(stock_symbols[i])] = stock[stock_symbols[i]]['close'].pct_change() * 100
    stock[stock_symbols[1]].head()

```

	Attributes	high	low	open	close	avg	volume	today
Symbols	VCB	VCB	VCB	VCB	VCB	VCB	VCB	VCB
	date							
2014-01-02	26.900000	26.600000	26.900000	26.700000	26.740000	3.339000e+05	Nan	
2014-01-03	27.196296	26.674074	26.751852	27.070370	26.984444	6.890556e+05	1.387155	
2014-01-06	27.334635	26.895416	27.124109	27.090492	27.110458	4.763795e+05	0.074332	
2014-01-07	27.707721	27.228392	27.372160	27.492708	27.486300	1.727654e+06	1.484712	
2014-01-08	27.964049	27.471213	27.651724	27.889532	27.730488	1.562150e+06	1.443379	

Hình 4. 18 Tính toán phần trăm thay đổi của giá đóng cửa

Dòng lệnh này cho phép chúng tôi tính toán được phần trăm thay đổi của giá đóng cửa giữa hai phiên giao dịch liên tiếp nhau. Ở đây câu lệnh pct\_change() sẽ giúp ta tính được sự thay đổi của giá đóng cửa và bằng việc nhân với 100 giúp cho dữ liệu biểu diễn ở dạng số thực để khi đưa vào mô hình cho dễ huấn luyện.



```

for i in range(len(stock_symbols)):
    for j in range(1, 6):
        stock[stock_symbols[i]][f'previous{str(j)}', str(stock_symbols[i])] = stock[stock_symbols[i]]['today'].shift(j)
    stock[stock_symbols[2]].head()

```

	Attributes	high	low	open	close	avg	volume	today	previous1	previous2	previous3	previous4	previous5	
Symbols	VNM	VNM	VNM	VNM	VNM	VNM	VNM	VNM	VNM	VNM	VNM	VNM	VNM	
	date													
2014-01-02	137.000000	135.0	135.000000	135.000000	135.550000	32520.000000	Nan							
2014-01-03	136.259259	135.0	135.000000	135.740741	135.527778	31171.851852	0.548697	Nan	Nan	Nan	Nan	Nan	Nan	
2014-01-06	136.083192	135.0	135.000000	135.916808	135.610781	62355.653650	0.129709	0.548697	Nan	Nan	Nan	Nan	Nan	
2014-01-07	136.028293	135.0	135.659903	135.311804	135.412295	91070.886744	-0.445128	0.129709	0.548697	Nan	Nan	Nan	Nan	
2014-01-08	136.009806	135.0	135.228701	135.761493	135.534947	56980.762637	0.332336	-0.445128	0.129709	0.548697	Nan	Nan	Nan	

Hình 4. 19 Lùi dữ liệu phần trăm thay đổi giá đóng cửa về 1 đến 5 ngày trước

Ở dòng lệnh trên chúng tôi sẽ lùi dữ liệu phần trăm thay đổi giá đóng cửa về 1 đến 5 ngày trước đó để khi đưa lên mô hình huấn luyện thì nó có thể nhận ra được sự

thay đổi dữ liệu liên tục giữa các ngày liên tiếp mà đưa ra dự đoán về xu hướng tăng giảm phù hợp với dữ liệu thực tế.

stock[stock_symbols[3]].head()																
Attributes	high	low	open	close	avg	volume	today	previous1	previous2	previous3	previous4	previous5	ema50	ema21	ema14	ema5
Symbols	GAS	GAS	GAS	GAS	GAS	GAS	GAS	GAS	GAS	GAS	GAS	GAS	GAS	GAS	GAS	GAS
date																
2014-01-02	66.500000	65.500000	66.500000	66.500000	66.080000	229950.000000	NaN	NaN	NaN	NaN	NaN	NaN	1.000000	1.000000	1.000000	1.000000
2014-01-03	66.500000	65.500000	66.500000	66.129630	65.976296	249964.814815	-0.556948	NaN	NaN	NaN	NaN	NaN	0.997235	0.997272	0.997304	0.997461
2014-01-06	66.839559	65.839559	66.160441	66.720713	66.270832	200155.093379	0.893825	-0.556948	NaN	NaN	NaN	NaN	1.004050	1.004019	1.003992	1.003844
2014-01-07	67.275386	66.275386	66.711468	66.905015	66.705819	169367.272952	0.276229	0.893825	-0.556948	NaN	NaN	NaN	1.005058	1.004965	1.004886	1.004477
2014-01-08	67.422156	66.422156	67.227760	67.293797	67.061404	177190.635299	0.581096	0.276229	0.893825	-0.556948	NaN	NaN	1.008612	1.008418	1.008254	1.007419

Hình 4. 20 Tính tỉ lệ trung bình trượt mở rộng cho nhiều ngày

Dòng lệnh trên tính toán tỉ lệ giá trị trung bình trượt mở rộng của 50 ngày, 21 ngày, 14 ngày và 5 ngày. Cách tính này giúp chúng tôi quan sát được liên tục tỉ lệ giá trung bình trượt để có thể nhận ra được sự thay đổi giữa các giá trị trung bình trượt dựa trên cột dữ liệu giá đóng cửa của các công ty trên sàn chứng tại Việt Nam.

Tiếp theo, chúng tôi tiến hành tính các chỉ số, chỉ báo chứng khoán và thêm vào miền dữ liệu để hoàn thiện hơn về bộ dữ liệu đưa vào huấn luyện.

stock[stock_symbols[4]].head()						
previous4	previous5	ema50	ema21	ema14	emas	rsi
/N	HVN	HVN	HVN	HVN	HVN	HVN
NaN	NaN	1.000000	1.000000	1.000000	1.000000	NaN
NaN	NaN	1.051412	1.050683	1.050072	1.047008	100.000000
NaN	NaN	1.092220	1.090380	1.088840	1.081195	100.000000
NaN	NaN	1.029390	1.027717	1.026338	1.019810	78.407660
NaN	NaN	0.983469	0.982441	0.981627	0.978254	64.824189

Hình 4. 21 Tính chỉ số phân tích chứng khoán RSI

```

[ ] def macd(X, short_window=12, long_window=29, signal=9):
    short_ema = X.ewm(span=short_window, adjust=False).mean()
    long_ema = X.ewm(span=long_window, adjust=False).mean()
    macd = short_ema - long_ema
    signal = macd.ewm(span=signal, adjust=False).mean()
    return signal

for i in range(len(stock_symbols)):
    stock[stock_symbols[i]]['macd', str(stock_symbols[i])] = macd(stock[stock_symbols[i]]['close'])
stock[stock_symbols[5]].head()

```

	us4	previous5	ema50	ema21	ema14	ema5	rsi	macd
	MWG	MNG	MWG	MNG	MNG	MNG	MNG	MNG
NaN	NaN	1.000000	1.000000	1.000000	1.000000	NaN	0.000000	
NaN	NaN	1.024138	1.023804	1.023524	1.022119	100.0	0.071035	
NaN	NaN	1.054212	1.053216	1.052380	1.048204	100.0	0.271165	
NaN	NaN	1.088113	1.086065	1.084349	1.075834	100.0	0.644448	
NaN	NaN	1.121361	1.117879	1.114968	1.100672	100.0	1.215521	

Hình 4. 22 Tính chỉ số phân tích chứng khoán MACD

```

[ ] def obv(X):
    obv = pd.Series(index=X.index)
    obv.iloc[0] = 0

    for i in range(1, len(X)):
        if (X['close'].values > X['close'].iloc[i-1].values):
            obv.iloc[i] = obv.iloc[i-1] + X['volume'].iloc[i].values
        elif (X['close'].values < X['close'].iloc[i-1].values):
            obv.iloc[i] = obv.iloc[i-1] - X['volume'].iloc[i].values
        else:
            obv.iloc[i] = obv.iloc[i-1]

    return obv

for i in range(len(stock_symbols)):
    stock[stock_symbols[i]]['obv', str(stock_symbols[i])] = obv(stock[stock_symbols[i]])
stock[stock_symbols[6]].head()

```

	0	ema21	ema14	ema5	rsi	macd	obv
	FPT	FPT	FPT	FPT	FPT	FPT	FPT
1000	1.000000	1.000000	1.000000	NaN	0.000000	0.000000	
3226	0.999236	0.999245	0.999289	0.000000	-0.001292	-176092.222222	
5851	1.005777	1.005715	1.005393	86.072358	0.004650	85282.022260	
1417	1.001344	1.001283	1.000986	63.662614	0.011589	-195635.738195	
3336	1.003232	1.003146	1.002725	69.333942	0.021105	-13593.836586	

Hình 4. 23 Tính chỉ số phân tích chứng khoán OBV

```

[ ] def emv(X):
    emv = pd.Series(index=X.index)
    emv.iloc[0] = np.nan

    for i in range(1, len(X)):
        dm = 0.5 * ((X['high'].iloc[i].values + X['low'].iloc[i].values) - (X['high'].iloc[i-1].values + X['low'].iloc[i-1].values))
        br = X['volume'].iloc[i].values / (1000000 * (X['high'].iloc[i].values - X['low'].iloc[i].values))
        emv.iloc[i] = dm / br if br != 0 else 0

    return emv

for i in range(len(stock_symbols)):
    stock[stock_symbols[i]]['emv', str(stock_symbols[i])] = emv(stock[stock_symbols[i]])
stock[stock_symbols[7]].head()

```

	emv	emv21	ema14	ema5	rsi	macd	obv
	HPG	HPG	HPG	HPG	HPG	HPG	HPG
1000	1.000000	1.000000	1.000000	NaN	0.000000	0.000000e+00	NaN
3226	1.004381	1.004330	1.004076	100.0	0.006458	4.248648e+05	0.050044
5851	1.011663	1.011492	1.010634	100.0	0.026481	8.687213e+05	0.086837
1417	1.010434	1.010185	1.008961	100.0	0.055928	1.205920e+06	0.994812
3336	1.012467	1.012122	1.010445	100.0	0.093794	1.556489e+06	0.068984

Hình 4. 24 Tính chỉ số phân tích chứng khoán EMV

```

[ ] def mfi(X, window=14):
    combine_price = (X['high'] + X['low'] + X['close']) / 3
    raw_money_flow = combine_price * X['volume']

    flow_direction = (combine_price.diff() > 0).astype(int)

    positive_money_flow = flow_direction * raw_money_flow
    negative_money_flow = (1 - flow_direction) * raw_money_flow

    positive = positive_money_flow.rolling(window=window, min_periods=1).sum()
    negative = negative_money_flow.rolling(window=window, min_periods=1).sum()

    mf = positive / negative
    mfi = 100 - (100 / (1 + mf))

    return mfi

for i in range(len(stock_symbols)):
    stock[stock_symbols[i]]['mfi', str(stock_symbols[i])] = mfi(stock[stock_symbols[i]])
stock[stock_symbols[8]].head()

```

	mfi	ema5	rsi	macd	obv	emv	mfi
	MSN	MSN	MSN	MSN	MSN	MSN	MSN
100	1.000000	NaN	0.000000	0.000000	NaN	0.000000	
314	0.997942	0.000000	-0.006458	-59922.962963	-3.490993	0.000000	
372	1.000614	40.444894	-0.012272	40466.578633	-0.456699	39.671831	
318	1.003794	64.051241	-0.010242	154588.997176	5.667450	58.494597	
314	1.015355	85.033504	0.021847	294337.436332	8.272094	70.060592	

Hình 4. 25 Tính chỉ số phân tích chứng khoán MFI

```
window = 6
for i in range(len(stock_symbols)):
    stock[stock_symbols[i]]['roc', str(stock_symbols[i])] = ((stock[stock_symbols[i]]['close'] - stock[stock_symbols[i]].shift(window)) / stock[stock_symbols[i]])
stock[stock_symbols[6]].head()

   open      close      avg      volume      today  previous1  previous2  previous3  previous4  previous5  ema50  ema21  ema14  ema5      rsi      macd      roc
PT      FPT      FPT
7.200000 47.600000 47.400000 47.550000 207470.000000      NaN      NaN      NaN      NaN      NaN      1.000000 1.000000 1.000000 1.000000      NaN      0.000000  NaN
7.051852 47.377778 47.325926 47.305566 176092.22222 -0.156274      NaN      NaN      NaN      NaN      NaN      0.999226 0.999236 0.999245 0.999289 0.000000 -0.001292  NaN
7.016638 47.324958 47.783701 47.342530 261374.24482 0.967282 -0.156274      NaN      NaN      NaN      NaN      NaN      1.005851 1.005777 1.005715 1.005393 86.072358 0.004650  NaN
7.335610 47.770420 47.596486 47.637817 280917.760455 -0.391797 0.967282 -0.156274      NaN      NaN      NaN      1.001417 1.001344 1.001283 1.000986 63.662614 0.011589  NaN
7.377684 47.528376 47.729469 47.743792 182041.901609 0.279396 -0.391797 0.967282 -0.156274      NaN      NaN      1.003336 1.003232 1.003146 1.002725 69.333942 0.021105  NaN
4
```

Hình 4. 26 Tính chỉ số phân tích chứng khoán ROC

```
[ ] window = 16
for i in range(len(stock_symbols)):
    stock[stock_symbols[i]]['high-low', str(stock_symbols[i])] = stock[stock_symbols[i]]['high', str(stock_symbols[i])] - stock[stock_symbols[i]]['high-preclose', str(stock_symbols[i])] = abs(stock[stock_symbols[i]]['high', str(stock_symbols[i])]- stock[stock_symbols[i]]['low', str(stock_symbols[i])])
    stock[stock_symbols[i]]['trn', str(stock_symbols[i])] = stock[stock_symbols[i]][['high-low', str(stock_symbols[i])], ('high-preclose', str(stock_symbols[i]))]
    stock[stock_symbols[i]]['atr', str(stock_symbols[i])] = stock[stock_symbols[i]]['trn', str(stock_symbols[i])].sum() / window
stock[stock_symbols[7]].head()
```

Hình 4. 27 Tính chỉ số phân tích chứng khoán ATR

```
[ ] def cmf(X, window=16):
    money_flow_multiplier = ((X['close'] - X['low']) - (X['high'] - X['close']))/(X['high']-X['low'])
    money_flow_volume = money_flow_multiplier * X['volume']
    cmf = money_flow_volume.rolling(window=window).sum() / X['volume'].rolling(window=window).sum()
    return cmf
for i in range(len(stock_symbols)):
    stock[stock_symbols[i]]['cmf', str(stock_symbols[i])] = cmf(stock[stock_symbols[i]])
stock[stock_symbols[9]].head()
```

Hình 4. 28 Tính chỉ số phân tích chứng khoán CMF

```
5 rows × 23 columns
[ ] window = 16
for i in range(len(stock_symbols)):
    stock[stock_symbols[i]]['minimum_low', str(stock_symbols[i])] = stock[stock_symbols[i]]['low', str(stock_symbols[i])].rolling(window=window).min()
    stock[stock_symbols[i]]['maximum_high', str(stock_symbols[i])] = stock[stock_symbols[i]]['high', str(stock_symbols[i])].rolling(window=window).max()
    stock[stock_symbols[i]]['stoch', str(stock_symbols[i])] = ((stock[stock_symbols[i]]['close'], str(stock_symbols[i])) - (stock[stock_symbols[i]]['minimum_low', str(stock_symbols[i])], ('maximum_high', str(stock_symbols[i])))) / (stock[stock_symbols[i]]['maximum_high', str(stock_symbols[i])] - stock[stock_symbols[i]]['minimum_low', str(stock_symbols[i])])
for i in range(len(stock_symbols)):
    stock[stock_symbols[i]] = stock[stock_symbols[i]].drop(['minimum_low', str(stock_symbols[i])], ('maximum_high', str(stock_symbols[i])), ('stoch', str(stock_symbols[i])))
stock[stock_symbols[7]].head()
```

Hình 4. 29 Tính chỉ số phân tích chứng khoán STOCH

```

window = 21
for i in range(len(stock_symbols)):
    stock[stock_symbols[i]]['combine_price', str(stock_symbols[i])] = (stock[stock_symbols[i]]['high', s
    stock[stock_symbols[i]]['sma_combine_price', str(stock_symbols[i])] = stock[stock_symbols[i]]['combi
    stock[stock_symbols[i]]['mean_deviation', str(stock_symbols[i])] = stock[stock_symbols[i]]['combine_
    stock[stock_symbols[i]]['cci', str(stock_symbols[i])] = (stock[stock_symbols[i]]['combine_price', st
    for i in range(len(stock_symbols)):
        stock[stock_symbols[i]] = stock[stock_symbols[i]].drop([('combine_price', str(stock_symbols[i])), ('
    stock[stock_symbols[5]].head()

```

The screenshot shows a Jupyter Notebook cell with Python code for calculating the Commodity Channel Index (CCI) for a list of stock symbols. The code uses a window size of 21. It calculates the high price, simple moving average, mean deviation, and CCI for each symbol. The resulting DataFrame is displayed in the output pane.

Hình 4. 30 Tính chỉ số phân tích chứng khoán CCI

Vậy là chúng tôi đã thực nghiệm tái cấu trúc dữ liệu của các công ty, tiếp theo đây chúng tôi sẽ thực nghiệm các phương pháp tái cấu trúc dữ liệu tiếp theo.

#### 4.1.3.2 Tái cấu trúc dữ liệu tổng hợp các công ty

Ở phương pháp này, chúng tôi sẽ thực hiện thêm các cách xử lý dữ liệu được trình bày như dưới.

```

for i in range(len(stock_symbols)):
    print(f"Length of {stock_symbols[i]} with index {i}: ", len(stock[stock_symbols[i]]))

```

The screenshot shows a Jupyter Notebook cell with a for loop that prints the length of each stock symbol's data frame, indexed by its symbol name. The output shows the length for each of the 10 symbols: VIC, VCB, VNM, GAS, HVN, MNG, FPT, HPG, MSN, and SAB.

Hình 4. 31 Độ dài dữ liệu của tất cả các mã chứng khoán

Sau khi lấy dữ liệu thành công thì chúng ta sẽ theo dõi xem độ dài dữ liệu của tất cả các mã chứng khoán để phục vụ cho quá trình tổng hợp tất cả dữ liệu của tất cả các công ty.

```

[ ] changed_start_index5 = len(stock[stock_symbols[5]]) - len(stock[stock_symbols[4]])
stock[stock_symbols[5]] = stock[stock_symbols[5]][changed_start_index5:]

[ ] changed_start_index9 = len(stock[stock_symbols[9]]) - len(stock[stock_symbols[4]])
stock[stock_symbols[9]] = stock[stock_symbols[9]][changed_start_index9:]

[ ] changed_start_other_index = len(stock[stock_symbols[0]]) - len(stock[stock_symbols[4]])
for i in range(len(stock_symbols)):
    if ((i == 4) or (i == 5) or (i == 9)):
        pass
    else:
        stock[stock_symbols[i]] = stock[stock_symbols[i]][changed_start_other_index:]

```

The screenshot shows a Jupyter Notebook cell with Python code for calculating start indices for concatenating data frames. It handles four specific symbols (5, 9, and others) by calculating the difference in lengths between their current state and the state before symbol 4, and then applying that offset to the start of the respective data frame.

Hình 4. 32 Tính toán các chỉ số để cân bằng độ dài dữ liệu

Sau khi theo dõi được độ dài của tất cả các dữ liệu thì ta có thể thấy mã HVN, MWG, SAB là các công ty có độ dài dữ liệu thấp hơn các công ty còn lại. Chính vì lí do nêu trên nên chúng tôi sẽ tính toán lại các chỉ số của các công ty để dữ liệu được cân bằng trước khi tổng hợp chúng lại với nhau.

```

combined_high = (stock[stock_symbols[0]]['high'].values + stock[stock_symbols[1]]['high'].values + stock[stock_symbols[2]]['high'].values +
                  stock[stock_symbols[3]]['high'].values + stock[stock_symbols[4]]['high'].values + stock[stock_symbols[5]]['high'].values +
                  stock[stock_symbols[6]]['high'].values + stock[stock_symbols[7]]['high'].values + stock[stock_symbols[8]]['high'].values +
                  stock[stock_symbols[9]]['high'].values) / len(stock_symbols)

combined_high
array([[82.885],
       [83.95 ],
       [85.11 ],
       ...,
       [58.545],
       [58.44 ],
       [58.635]])

```

Hình 4. 33 Tính trung bình giá cao nhất được tổng hợp từ các công ty

```

combined_low = (stock[stock_symbols[0]]['low'].values + stock[stock_symbols[1]]['low'].values + stock[stock_symbols[2]]['low'].values +
                  stock[stock_symbols[3]]['low'].values + stock[stock_symbols[4]]['low'].values + stock[stock_symbols[5]]['low'].values +
                  stock[stock_symbols[6]]['low'].values + stock[stock_symbols[7]]['low'].values + stock[stock_symbols[8]]['low'].values +
                  stock[stock_symbols[9]]['low'].values) / len(stock_symbols)

combined_low
array([[81.455],
       [82.445],
       [83.565],
       ...,
       [57.755],
       [57.685],
       [57.625]])

```

Hình 4. 34 Tính trung bình giá thấp nhất được tổng hợp từ các công ty

```

combined_open = (stock[stock_symbols[0]]['open'].values + stock[stock_symbols[1]]['open'].values + stock[stock_symbols[2]]['open'].values +
                  stock[stock_symbols[3]]['open'].values + stock[stock_symbols[4]]['open'].values + stock[stock_symbols[5]]['open'].values +
                  stock[stock_symbols[6]]['open'].values + stock[stock_symbols[7]]['open'].values + stock[stock_symbols[8]]['open'].values +
                  stock[stock_symbols[9]]['open'].values) / len(stock_symbols)

combined_open
array([[82.255],
       [82.795],
       [84.76 ],
       ...,
       [58.055],
       [58.01 ],
       [58.325]])

```

Hình 4. 35 Tính trung bình giá mở cửa được tổng hợp từ các công ty

```

combined_close = (stock[stock_symbols[0]]['close'].values + stock[stock_symbols[1]]['close'].values + stock[stock_symbols[2]]['close'].values +
                  stock[stock_symbols[3]]['close'].values + stock[stock_symbols[4]]['close'].values + stock[stock_symbols[5]]['close'].values +
                  stock[stock_symbols[6]]['close'].values + stock[stock_symbols[7]]['close'].values + stock[stock_symbols[8]]['close'].values +
                  stock[stock_symbols[9]]['close'].values) / len(stock_symbols)

combined_close
array([[82.185],
       [83.725],
       [84.06 ],
       ...,
       [57.93 ],
       [58.09 ],
       [57.71 ]])

```

Hình 4. 36 Tính trung bình giá đóng cửa được tổng hợp từ các công ty

```
[ ] combined_avg = (stock[stock_symbols[0]]['avg'].values + stock[stock_symbols[1]]['avg'].values + stock[stock_symbols[2]]['avg'].values +
stock[stock_symbols[3]]['avg'].values + stock[stock_symbols[4]]['avg'].values + stock[stock_symbols[5]]['avg'].values +
stock[stock_symbols[6]]['avg'].values + stock[stock_symbols[7]]['avg'].values + stock[stock_symbols[8]]['avg'].values +
stock[stock_symbols[9]]['avg'].values) / len(stock_symbols)
combined_avg
array([[82.107 ],
   [83.28 ],
   [84.3326],
   ...,
   [58.148 ],
   [58.099 ],
   [58.183 ]])
```

Hình 4. 37 Tính trung bình giá trung bình được tổng hợp từ các công ty

```
[ ] combined_volume = (stock[stock_symbols[0]]['volume'].values + stock[stock_symbols[1]]['volume'].values + stock[stock_symbols[2]]['volume'].values +
stock[stock_symbols[3]]['volume'].values + stock[stock_symbols[4]]['volume'].values + stock[stock_symbols[5]]['volume'].values +
stock[stock_symbols[6]]['volume'].values + stock[stock_symbols[7]]['volume'].values + stock[stock_symbols[8]]['volume'].values +
stock[stock_symbols[9]]['volume'].values) / len(stock_symbols)
combined_volume
array([[ 747272. ],
   [ 605305.8],
   [ 643985. ],
   ...,
   [4714230. ],
   [5987070. ],
   [5748880.1]])
```

Hình 4. 38 Tính trung bình khối lượng giao dịch được tổng hợp từ các công ty

Sau khi cân bằng được độ dài dữ liệu của tất cả các công ty thì chúng tôi sẽ tính trung bình tất cả các giá như giá cao nhất, giá thấp nhất, giá mở cửa, giá đóng cửa, giá trung bình, khối lượng giao dịch của tất cả các công ty lại thành những mảng lưu trữ các giá trị trung bình đó.

```
[ ] df = df.rename(columns={0: 'high'})
df.head()
df = pd.DataFrame(combined_high)
df.head()
df['low'] = combined_low
df['open'] = combined_open
df['close'] = combined_close
df['avg'] = combined_avg
df['volume'] = combined_volume
df.head()

high    low   open  close    avg  volume
0  82.885  81.455  82.255  82.185  82.1070  747272.0
1  83.950  82.445  82.795  83.725  83.2800  605305.8
2  85.110  83.565  84.760  84.060  84.3326  643985.0
3  84.750  83.060  84.335  83.335  83.6911  727448.6
4  83.620  82.000  83.465  82.270  82.5046  722580.3
```

Hình 4. 39 Chuyển các giá trị trung bình vừa tính được thành Dataframe

Như vậy là chúng tôi đã thực hiện được tái cấu trúc dữ liệu thành miền dữ liệu của tất cả các công ty lại thành một, các bước tính toán các chỉ báo, chỉ số chứng khoán được thực hiện tương tự như phần tái cấu trúc dữ liệu của các công ty.

#### 4.1.3.3 Tái cấu trúc dữ liệu của các công ty theo tháng

Ở đây, chúng tôi sẽ trình bày cách thức chúng tôi thực hiện để tái cấu trúc dữ liệu của các công ty từ ngày thành các tuần và từ đó có được dữ liệu từng tháng một tương ứng.

```
[ ] def reconstruct_data(list_price):
    week = []
    combine = []
    i = 0
    while (i < len(list_price)):
        if (len(week) != 4):
            week.append(np.round(np.mean(list_price[i:i+7]), 2))
            i = i + 7
        else:
            combine.append(week)
            week = []
    return combine
```

Hình 4. 40 Hàm tái cấu trúc lại dữ liệu

Trong hàm tái cấu trúc dữ liệu này thì chúng tôi sẽ tính trung bình dữ liệu theo bảy ngày trong một tuần và dữ liệu đó sẽ được chia làm 4 tuần để trở thành một hàng dữ liệu và lưu lại trong một mảng tổng hợp để làm đầu ra sau khi đã tái cấu trúc dữ liệu.

```
[ ] close = stock[stock_symbols[0]]['close'].values
open = stock[stock_symbols[0]]['open'].values
high = stock[stock_symbols[0]]['high'].values
low = stock[stock_symbols[0]]['low'].values
avg = stock[stock_symbols[0]]['avg'].values
volume = stock[stock_symbols[0]]['volume'].values

[ ] reconstruct_close = reconstruct_data(close)
```

Hình 4. 41 Chuyển đổi dữ liệu các cột giá về mảng và tái cấu trúc giá đóng cửa

	(close_week1, VIC)	(close_week2, VIC)	(close_week3, VIC)	(close_week4, VIC)
0	70.43	72.93	75.93	77.43
1	77.71	74.64	77.14	75.50
2	72.36	66.79	65.93	63.43
3	66.07	67.21	65.21	64.07
4	64.14	69.21	69.86	72.21

Hình 4. 42 Kết quả sau khi tái cấu trúc giá đóng cửa

```

▶ mean_close = []
mean_open = []
mean_high = []
mean_low = []
mean_avg = []
mean_volume = []

i = 0
while (i < len(close)):
    mean_close.append(np.round(np.mean(close[i:i+28]), 2))
    mean_open.append(np.round(np.mean(open[i:i+28]), 2))
    mean_high.append(np.round(np.mean(high[i:i+28]), 2))
    mean_low.append(np.round(np.mean(low[i:i+28]), 2))
    mean_avg.append(np.round(np.mean(avg[i:i+28]), 2))
    mean_volume.append(np.round(np.mean(volume[i:i+28]), 2))
    i = i + 28

[ ] df_vic[['close_total4week', f'{stock_symbols[0]}']] = mean_close[:len(df_vic)]
df_vic[['open_total4week', f'{stock_symbols[0]}']] = mean_open[:len(df_vic)]
df_vic[['high_total4week', f'{stock_symbols[0]}']] = mean_high[:len(df_vic)]
df_vic[['low_total4week', f'{stock_symbols[0]}']] = mean_low[:len(df_vic)]
df_vic[['avg_total4week', f'{stock_symbols[0]}']] = mean_avg[:len(df_vic)]
df_vic[['volume_total4week', f'{stock_symbols[0]}']] = mean_volume[:len(df_vic)]

```

Hình 4. 43 Tính trung bình các cột của các mã chứng khoán

Sau khi tái cấu trúc dữ liệu giá đóng cửa thì chúng tôi sẽ tính trung bình của tất cả các cột của các mã chứng khoán với điều kiện số ngày sẽ tương đồng với số lượng ngày đã tái cấu trúc dữ liệu để được tương đồng các cột. Chúng tôi đã tái cấu trúc dữ liệu khoảng 4 tuần tương đương với 28 ngày. Chính vì lí do nêu trên nên chúng tôi sẽ tính trung bình với số lượng dữ liệu là 28 với tất cả các cột dữ liệu.

	(close_week1, VIC)	(close_week2, VIC)	(close_week3, VIC)	(close_week4, VIC)	(close_total4week, VIC)	(open_total4week, VIC)	(high_total4week, VIC)	(low_total4week, VIC)	(avg_total4week, VIC)	(volume_total4week, VIC)
0	70.43	72.93	75.93	77.43	74.18	74.12	74.82	73.20	74.03	259468.93
1	77.71	74.64	77.14	75.50	76.25	76.62	77.11	75.61	76.32	401583.57
2	72.36	66.79	65.93	63.43	67.12	67.21	68.00	66.20	67.00	669534.64
3	66.07	67.21	65.21	64.07	65.64	65.70	66.29	64.96	65.61	247855.36
4	64.14	69.21	69.86	72.21	68.86	68.93	69.57	68.21	68.87	429015.71
...	...	...	...	...	...	...	...	...	...	...
84	53.06	52.00	50.80	52.00	51.96	52.16	52.52	51.67	52.02	2993792.86
85	52.04	62.04	71.27	64.44	62.45	63.04	63.80	61.65	62.67	13572447.36
86	61.00	55.26	47.25	45.59	52.28	53.05	53.66	51.69	52.57	14991744.62
87	44.86	42.80	41.59	44.69	43.48	43.54	43.98	42.68	43.32	6670760.71
88	42.03	42.54	43.69	43.34	42.90	42.81	43.23	42.42	42.83	3607896.43

Hình 4. 44 Tái cấu trúc dữ liệu khoảng 4 tuần

```
[ ] def closediff(x, y):
    res = []

    for i in range(len(x)):
        res.append((y[i]-x[i]) / x[i]*100)

    return res
```

Hình 4. 45 Hàm tính toán sự chênh lệch giá

```
close_week1 = df_sab[('close_week1', f'{stock_symbols[9]}')].values
close_week2 = df_sab[('close_week2', f'{stock_symbols[9]}')].values
close_week3 = df_sab[('close_week3', f'{stock_symbols[9]}')].values
close_week4 = df_sab[('close_week4', f'{stock_symbols[9]}')].values

df_sab[('closediff_week2_1', f'{stock_symbols[9]}')] = closediff(close_week1, close_week2)
df_sab[('closediff_week3_1', f'{stock_symbols[9]}')] = closediff(close_week1, close_week3)
df_sab[('closediff_week4_1', f'{stock_symbols[9]}')] = closediff(close_week1, close_week4)
df_sab[('closediff_week3_2', f'{stock_symbols[9]}')] = closediff(close_week2, close_week3)
df_sab[('closediff_week4_2', f'{stock_symbols[9]}')] = closediff(close_week2, close_week4)
df_sab[('closediff_week4_3', f'{stock_symbols[9]}')] = closediff(close_week3, close_week4)
```

Hình 4. 46 Tính chênh lệch giá giữa các tuần với nhau

Sau khi tính toán các chỉ số kĩ thuật thì chúng tôi có một ý tưởng là sẽ tính sự chênh lệch giữa tuần sau và tuần trước đó để biểu diễn thêm các cột dữ liệu mới. Các cột dữ liệu này sẽ cho chúng ta góc nhìn về sự chênh lệch về giá giữa các tuần với nhau và từ đó có thể đưa ra được dự đoán về xu hướng của tháng đó.

```

❶ def total_closediff(a, b, c, d, e, f):
    res = []

    for i in range(len(a)):
        res.append(((a[i]+b[i]+c[i]+d[i]+e[i]+f[i])/6))

    return res

```

Hình 4. 47 Hàm tính trung bình của các chênh lệch giá

```

❶ df_sab[('total_closediff', f'{stock_symbols[9]}')] = total_closediff(df_sab[['closediff_week2_1', f'{stock_symbols[9]}']], df_sab[['closediff_week3_1', f'{stock_symbols[9]}']],
df_sab[['closediff_week4_1', f'{stock_symbols[9]}']], df_sab[['closediff_week3_2', f'{stock_symbols[9]}']],
df_sab[['closediff_week4_2', f'{stock_symbols[9]}']], df_sab[['closediff_week4_3', f'{stock_symbols[9]}']])

```

Hình 4. 48 Tính tổng trung bình của các chênh lệch giá giữa các tuần

Với ý tưởng chính ở đây là chúng tôi sẽ tính tổng trung bình của tất cả các chênh lệch giá giữa các tuần để làm tiền đề cho quá trình dự đoán xu hướng của chúng tôi. Nếu như tổng trung bình trên lớn hơn 0 thì chúng tôi sẽ đánh nhãn dữ liệu là 1 thể hiện cho xu hướng tăng và nếu tổng trung bình nhỏ hơn 0 thì chúng tôi sẽ đánh nhãn dữ liệu là 0 thể hiện cho xu hướng giảm.

v, VIC	(emv, VIC)	(mfi, VIC)	(closediff_week2_1, VIC)	(closediff_week3_1, VIC)	(closediff_week4_1, VIC)	(closediff_week3_2, VIC)	(closediff_week4_2, VIC)	(closediff_week4_3, VIC)	(total_closediff, VIC)	(trend, VIC)
390e+05	7.307327	58.973335	-2.144382	1.323753	0.234465	3.544134	2.430977	-1.075056	0.718982	1
919e+05	-17.609446	31.828188	-5.858651	-5.505093	-8.176645	0.375560	-2.462249	-2.827192	-4.075712	0
965e+05	-11.443835	24.930677	-1.037558	-2.833028	-4.903153	-1.814295	-3.906123	-2.130482	-2.770773	0
910e+05	4.100213	39.901517	4.668414	4.665287	6.254013	-0.002988	1.514877	1.517911	3.102919	1

Hình 4. 49 Dánh nhãn xu hướng dựa trên tổng trung bình của các chênh lệch giá tuần

Về các bước tính toán về các chỉ số, các chỉ báo để đưa vào miền dữ liệu thì được thực hiện tương tự như phần tái cấu trúc dữ liệu của các công ty.

#### 4.1.3.4 Tái cấu trúc dữ liệu tổng hợp các công ty theo tháng

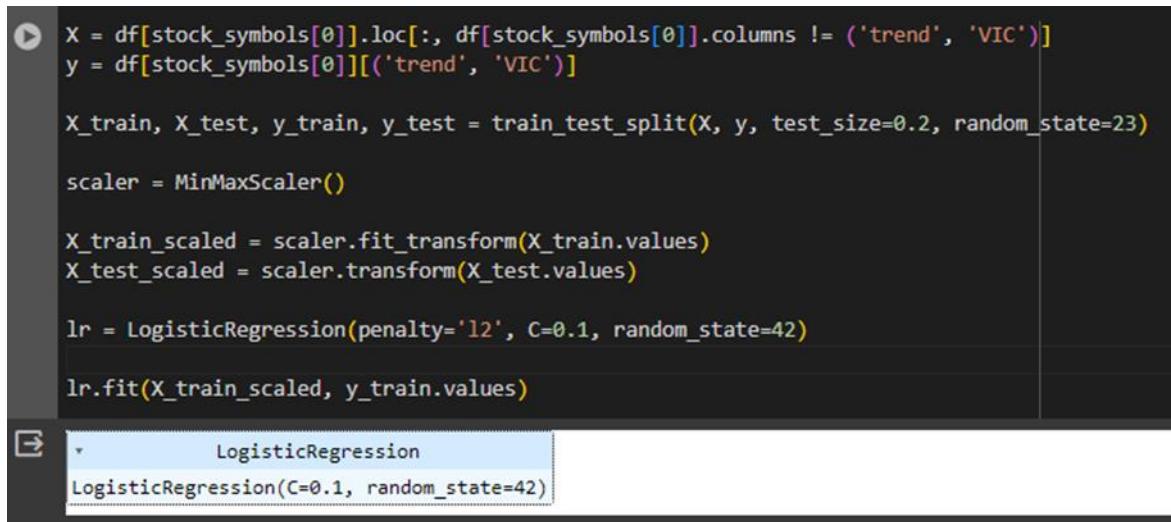
Đối với phần thực hiện việc tái cấu trúc dữ liệu tổng hợp các công ty theo tháng, chúng tôi thực hiện phần tái cấu trúc dữ liệu tổng hợp các công ty trước. Như vậy là chúng tôi đã có được miền dữ liệu tổng hợp từ tất cả các công ty, sau đó chúng tôi tiếp tục thực hiện phần tái cấu trúc dữ liệu của công ty theo tháng. Như vậy, chúng tôi đã có

được miền dữ liệu tổng hợp của tất cả các công ty theo từng các tuần và tháng tương ứng. Sau đó việc tính toán các chỉ số, chỉ báo chứng khoán chúng tôi thực hiện tương tự như phân tái cấu trúc dữ liệu của các công ty.

## 4.2 Mô hình thực nghiệm

### 4.2.1 Mô hình Logistic Regression

Với mô hình Logistic Regression, dữ liệu huấn luyện cho mô hình này sẽ là tất cả các cột dữ liệu trừ cột “trend” là đầu ra của bộ dữ liệu. Sau đó chúng tôi sẽ dùng hàm để chia ra thành tập huấn luyện và tập kiểm thử với tỉ lệ 80-20, trong đó với 80 là dữ liệu huấn luyện được dùng làm dữ liệu đầu vào của mô hình để thực hiện huấn luyện còn với 20 là dữ liệu kiểm thử dùng để dự đoán và đánh giá hiệu suất của mô hình. Chúng tôi sẽ áp dụng loại hồi quy L2 để giúp giảm Overfitting vì quá nhiều cột dữ liệu. Ngoài ra, chúng tôi còn áp dụng một đối số với loại hồi quy L2 và sử dụng là C bằng 0.1 để tăng cường hơn cho sức mạnh của loại hồi quy này.



```

X = df[stock_symbols[0]].loc[:, df[stock_symbols[0]].columns != ('trend', 'VIC')]
y = df[stock_symbols[0]][('trend', 'VIC')]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=23)

scaler = MinMaxScaler()

X_train_scaled = scaler.fit_transform(X_train.values)
X_test_scaled = scaler.transform(X_test.values)

lr = LogisticRegression(penalty='l2', C=0.1, random_state=42)

lr.fit(X_train_scaled, y_train.values)

```

LogisticRegression  
LogisticRegression(C=0.1, random\_state=42)

Hình 4. 50 Xây dựng mô hình Logistic Regression

Với cách thức mô hình này chúng tôi sẽ áp dụng cho mọi miền dữ liệu mà chúng tôi đã thực hiện việc tái cấu trúc ở phía trên.

#### 4.2.2 Mô hình Random Forest

Chúng tôi thực hiện phân chia dữ liệu bằng cách lấy các cột chỉ số phân tích chứng khoán và cột dữ liệu khối lượng giao dịch làm dữ liệu đầu vào. Dữ liệu đầu ra vẫn là cột dữ liệu nhãn thể hiện xu hướng tăng giảm trên thị trường chứng khoán. Sau đó, chúng tôi thực hiện chia dữ liệu ra làm hai tập là tập huấn luyện và tập kiểm thử để thực hiện huấn luyện, dự đoán và đánh giá hiệu suất của mô hình. Đối với các thông số trong mô hình Random Forest thì chúng tôi sử dụng số lượng cây quyết định là 110 vì sau khi thử rất nhiều các thông số thì chúng tôi đánh giá rằng đây là thông số tốt nhất, ngoài thông số này thì chúng tôi cũng tạo ra các seed để lặp lại dữ liệu trong quá trình bầu chọn của thuật toán để so sánh và đưa ra trọng số tốt nhất để sau khi huấn luyện thì mô hình có thể cho ra được đầu ra tốt nhất.

```
[ ] X = df[stock_symbols[0]][[('volume', stock_symbols[0]), ('ema50', stock_symbols[0]), ('ema21', stock_symbols[0]),
                           ('emad4', stock_symbols[0]), ('emas', stock_symbols[0]), ('rsi', stock_symbols[0]),
                           ('macd', stock_symbols[0]), ('obv', stock_symbols[0]), ('emv', stock_symbols[0]), ('mfi', stock_symbols[0])]]
y = df[stock_symbols[0]][('trend', stock_symbols[0])]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, shuffle=False)

rf = RandomForestClassifier(n_estimators=110, random_state=21)
rf.fit(X_train.values, y_train.values)
```

Hình 4. 51 Xây dựng mô hình Random Forest

Với cách thức mô hình này chúng tôi cũng sẽ áp dụng cho mọi miền dữ liệu mà chúng tôi đã thực hiện việc tái cấu trúc ở phía trên.

#### 4.2.3 Mô hình LSTM

Chúng tôi cũng thực hiện việc phân chia dữ liệu huấn luyện và dữ liệu kiểm tra lần lượt là 80 và 20. Dữ liệu huấn luyện cũng sẽ là các cột dữ liệu trừ cột “trend” là cột đầu ra. Với bước nhảy chúng tôi sẽ cho các giá trị là 30, 15 hoặc là 10 cho từng miền dữ liệu khác nhau. Các đặc trưng được đưa vào sẽ là tổng số các cột dữ liệu. Chúng tôi đầu tiên thiết lập một lớp Sequence, sau đó chúng tôi sẽ cho hai lớp LSTM với unit lần lượt là 128 và 64. Thiết lập return\_sequences cho lớp đầu tiên để trả về cho lớp thứ hai. Cuối

cùng sẽ có một lớp Dense với hàm activation là “sigmoid”. Chúng tôi lựa chọn optimizer cho mô hình là “adam” và hàm loss là “mse”.

```

▶ lstm_model = Sequential()
lstm_model.add(LSTM(128, return_sequences=True, input_shape=(timesteps, feature)))
lstm_model.add(LSTM(64, return_sequences=False))
lstm_model.add(Dense(1, activation='sigmoid'))
lstm_model.compile(optimizer='adam', loss='mse')

[109] lstm_model.summary()

Model: "sequential_9"
-----  

Layer (type)          Output Shape         Param #
-----  

lstm_18 (LSTM)        (None, 10, 128)      76800  

lstm_19 (LSTM)        (None, 64)           49408  

dense_9 (Dense)       (None, 1)            65  

-----  

Total params: 126273 (493.25 KB)
Trainable params: 126273 (493.25 KB)
Non-trainable params: 0 (0.00 Byte)
-----
```

Hình 4. 52 Xây dựng mô hình LSTM

Chúng tôi áp dụng các thức xây mô hình tương tự cho các miền dữ liệu khác nhau.

#### 4.2.4 Mô hình ARIMA

Với mô hình ARIMA, chúng tôi sẽ áp dụng với chỉ giá đóng cửa của miền dữ liệu. Đầu tiên chúng tôi sẽ kiểm định sai phân của chuỗi cũng như xác định chuỗi có tính dừng ở bậc sai phân nào.

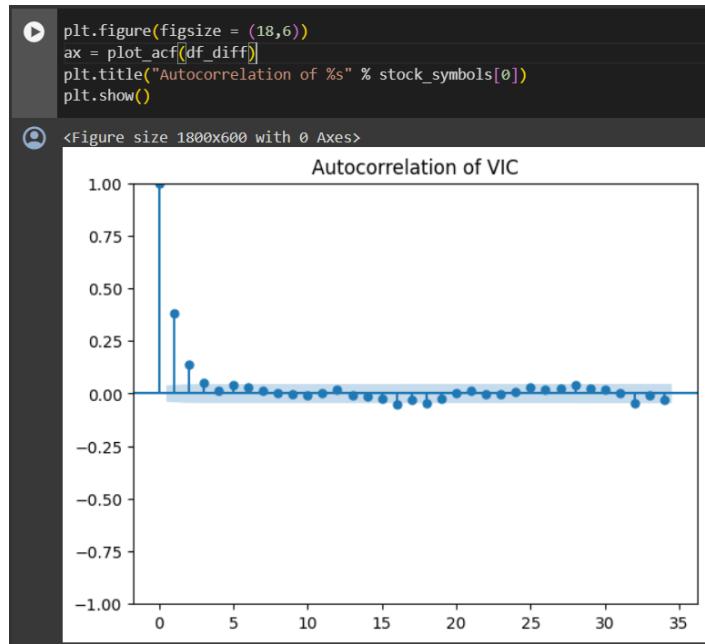
```

▶ df = stock_data[stock_symbols[0]]['close']
trend_data = stock_data[stock_symbols[0]]['trend']
d = ndiffs(df, test="adf")
print("Tính toán bậc sai phân của chuỗi giá đóng:", d)

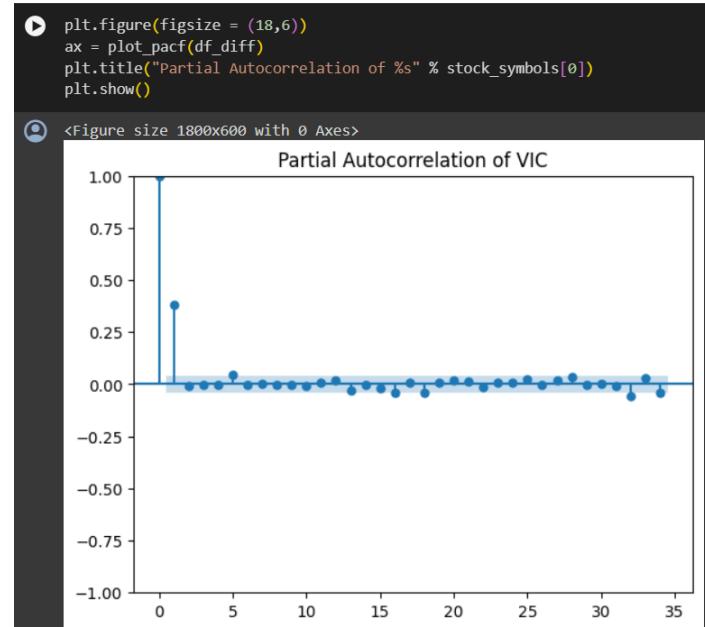
@ Tính toán bậc sai phân của chuỗi giá đóng: 1
```

Hình 4. 53 Tính toán bậc sai phân của chuỗi giá đóng

Tiếp đến, chúng tôi sẽ thực hiện việc vẽ các biểu đồ ACF và PACF tương ứng lần lượt có thể xác định giá trị  $q$  của quá trình trung bình trượt và tham số  $p$  của quá trình hồi qui tuyến tính.



Hình 4. 54 Biểu đồ tự tương quan ACF



Hình 4. 55 Biểu đồ tự tương quan riêng phần PACF

Sau khi đã có được một danh sách các tham số có triển vọng từ việc lựa chọn từ các biểu đồ trên. Chúng tôi cũng thực hiện việc chia tập dữ liệu thành các tập huấn luyện và tập kiểm thử lần lượt là 80 và 20.

Chúng tôi tiến hành thực hiện việc xây dựng các mô hình từ các tham số triển vọng đó, huấn luyện và kiểm định tính chính xác để có một mô hình tốt nhất.

```
▶ model = ARIMA(train_data, order=(4,1,7))
result = model.fit()
print(result.summary())
```

Hình 4. 56 Xây dựng và huấn luyện mô hình ARIMA

SARIMAX Results						
Dep. Variable:	VIC	No. Observations:	1995			
Model:	ARIMA(4, 1, 7)	Log Likelihood	-3003.215			
Date:	Tue, 12 Mar 2024	AIC	6006.431			
Time:	17:28:15	BIC	6097.606			
Sample:	0 - 1995	HQIC	6055.100			
Covariance Type:	opg					
	coef	std err	z	P> z	[0.025	0.975]
ar.L1	0.0634	0.020	3.226	0.001	0.025	0.102
ar.L2	1.4019	0.018	79.050	0.000	1.367	1.437
ar.L3	0.0946	0.019	5.016	0.000	0.058	0.132
ar.L4	-0.9700	0.020	-48.820	0.000	-1.009	-0.931
ma.L1	0.3345	0.026	12.884	0.000	0.284	0.385
ma.L2	-1.2926	0.030	-42.873	0.000	-1.352	-1.233
ma.L3	-0.5983	0.028	-21.012	0.000	-0.654	-0.543
ma.L4	0.7458	0.033	22.791	0.000	0.682	0.810
ma.L5	0.3052	0.024	12.789	0.000	0.258	0.352
ma.L6	0.1320	0.018	7.542	0.000	0.098	0.166
ma.L7	0.0471	0.014	3.371	0.001	0.020	0.075
sigma2	1.2021	0.013	93.811	0.000	1.177	1.227
Ljung-Box (L1) (Q):	0.44	Jarque-Bera (JB):	68383.68			
Prob(Q):	0.51	Prob(JB):	0.00			
Heteroskedasticity (H):	2.03	Skew:	-2.07			
Prob(H) (two-sided):	0.00	Kurtosis:	31.39			

Hình 4. 57 Kết quả thống kê sau khi huấn luyện mô hình

Sau đó, chúng tôi tiến hành việc tính toán phần trăm thay đổi lần lượt của hôm trước và hôm sau để đánh nhận xu hướng cho tập dự đoán và tiến hành việc so sánh tính chính xác.

Chúng tôi áp dụng các thức xây dựng mô hình ARIMA này cho toàn bộ miền dữ liệu khác nhau.

### 4.3 Các tiêu chí, độ đo đánh giá mô hình

Đối với bài nghiên cứu của nhóm chúng tôi, chúng tôi tập trung vào xu hướng biến đổi của thị trường chứng hay nói cách khác là xác định xu hướng tăng hoặc giảm của từng mốc thời điểm. Chúng tôi sử dụng tính toán sự thay đổi của dữ liệu trước so với dữ liệu sau để xác định và đánh nhận. Sau đó, chúng tôi sẽ tiến hành so sánh từ tập

nhân xu hướng dự đoán so với tập nhãn xu hướng kiểm thử. Từ đó, chúng tôi có thể xác định phần trăm chính xác hay nói cách khác là đánh giá được accuracy của các loại mô hình.

Và như đã nói từ trước, nhóm chúng tôi sử dụng độ đo accuracy là độ đo đánh giá hiệu quả dự đoán mô hình là chủ yếu. Cách đánh giá này đơn giản là đếm số lượng đúng giữa hai tập dự đoán và kiểm thử, từ đó tính phần trăm với tổng độ dài của tập để cho ra được độ đo accuracy.

## 4.4 Kết quả thực nghiệm

### 4.4.1 Với dữ liệu của các công ty

- Với mô hình Logistic Regression, chúng tôi thu được độ chính xác của mô hình là:

VIC	VCB
<pre> [ ] predictions = lr.predict(X_test_scaled) predictions[:21]  [ ] array([0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1])  [ ] y_test.values[:21] array([0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1])  [ ] accuracy = accuracy_score(y_test.values, predictions) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.69 </pre>	<pre> [ ] predictions = lr.predict(X_test_scaled) predictions[:21]  [ ] array([1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1])  [ ] y_test.values[:21] array([0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1])  [ ] accuracy = accuracy_score(y_test.values, predictions) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.66 </pre>
VNM	GAS
<pre> [ ] predictions = lr.predict(X_test_scaled) predictions[:21]  [ ] array([1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0])  [ ] y_test.values[:21] array([1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1])  [ ] accuracy = accuracy_score(y_test.values, predictions) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.73 </pre>	<pre> [ ] predictions = lr.predict(X_test_scaled) predictions[:21]  [ ] array([1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0])  [ ] y_test.values[:21] array([1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1])  [ ] accuracy = accuracy_score(y_test.values, predictions) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.78 </pre>
HVN	MWG
<pre> [ ] predictions = lr.predict(X_test_scaled) predictions[:21]  [ ] array([0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])  [ ] y_test.values[:21] array([0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])  [ ] accuracy = accuracy_score(y_test.values, predictions) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.73 </pre>	<pre> [ ] predictions = lr.predict(X_test_scaled) predictions[:21]  [ ] array([0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1])  [ ] y_test.values[:21] array([1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1])  [ ] accuracy = accuracy_score(y_test.values, predictions) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.66 </pre>

<p><b>FPT</b></p> <pre> ❶ predictions = lr.predict(X_test_scaled) predictions[:21]  ❷ array([1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0]) [ ] y_test.values[:21]  array([1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1])  [ ] accuracy = accuracy_score(y_test.values, predictions) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.65 </pre>	<p><b>HPG</b></p> <pre> ❶ predictions = lr.predict(X_test_scaled) predictions[:21]  ❷ array([1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1]) [ ] y_test.values[:21]  array([0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1])  [ ] accuracy = accuracy_score(y_test.values, predictions) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.72 </pre>
<p><b>MSN</b></p> <pre> ❶ predictions = lr.predict(X_test_scaled) predictions[:21]  ❷ array([0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1]) [ ] y_test.values[:21]  array([1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1])  [ ] accuracy = accuracy_score(y_test.values, predictions) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.70 </pre>	<p><b>SAB</b></p> <pre> ❶ predictions = lr.predict(X_test_scaled) predictions[:21]  ❷ array([0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0]) [ ] y_test.values[:21]  array([0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0])  [ ] accuracy = accuracy_score(y_test.values, predictions) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.69 </pre>

Hình 4. 58 Độ chính xác khi dự đoán bằng mô hình Logistic Regression

- Với mô hình Random Forest, chúng tôi thu được độ chính xác của mô hình là:

VIC	VCB
<pre>y_pred = rf.predict(X_test.values) y_pred[:21]  array([1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1])  [ ] y_test.values[:21] array([1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1])  [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {accuracy:.2f}")  Accuracy: 0.79</pre>	<pre>y_pred = rf.predict(X_test.values) y_pred[:21]  array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1])  [ ] y_test.values[:21] array([0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1])  [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {accuracy:.2f}")  Accuracy: 0.76</pre>
VNM	GAS
<pre>y_pred = rf.predict(X_test.values) y_pred[:21]  array([0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0])  [ ] y_test.values[:21] array([1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0])  [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {accuracy:.2f}")  Accuracy: 0.79</pre>	<pre>y_pred = rf.predict(X_test.values) y_pred[:21]  array([0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0])  [ ] y_test.values[:21] array([1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1])  [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {accuracy:.2f}")  Accuracy: 0.77</pre>
HVN	MWG
<pre>y_pred = rf.predict(X_test.values) y_pred[:21]  array([0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0])  [ ] y_test.values[:21] array([0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0])  [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {accuracy:.2f}")  Accuracy: 0.78</pre>	<pre>y_pred = rf.predict(X_test.values) y_pred[:21]  array([1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0])  [ ] y_test.values[:21] array([1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0])  [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {accuracy:.2f}")  Accuracy: 0.80</pre>
FPT	HPG
<pre>y_pred = rf.predict(X_test.values) y_pred[:21]  array([0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1])  [ ] y_test.values[:21] array([0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0])  [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {accuracy:.2f}")  Accuracy: 0.77</pre>	<pre>y_pred = rf.predict(X_test.values) y_pred[:21]  array([1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0])  [ ] y_test.values[:21] array([1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0])  [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {accuracy:.2f}")  Accuracy: 0.78</pre>
MSN	SAB
<pre>y_pred = rf.predict(X_test.values) y_pred[:21]  array([1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0])  [ ] y_test.values[:21] array([0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0])  [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {accuracy:.2f}")  Accuracy: 0.78</pre>	<pre>y_pred = rf.predict(X_test.values) y_pred[:21]  array([1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0])  [ ] y_test.values[:21] array([1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1])  [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {accuracy:.2f}")  Accuracy: 0.79</pre>

Hình 4. 59 Độ chính xác khi dự đoán bằng mô hình Random Forest

- Với mô hình LSTM, chúng tôi thu được độ chính xác của mô hình là:

VIC	VCB																																																																						
<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.74</td><td>0.65</td><td>0.69</td><td>307</td></tr> <tr> <td>1</td><td>0.51</td><td>0.61</td><td>0.56</td><td>184</td></tr> <tr> <td>accuracy</td><td></td><td></td><td>0.64</td><td>491</td></tr> <tr> <td>macro avg</td><td>0.62</td><td>0.63</td><td>0.62</td><td>491</td></tr> <tr> <td>weighted avg</td><td>0.65</td><td>0.64</td><td>0.64</td><td>491</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.74	0.65	0.69	307	1	0.51	0.61	0.56	184	accuracy			0.64	491	macro avg	0.62	0.63	0.62	491	weighted avg	0.65	0.64	0.64	491	<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.67</td><td>0.61</td><td>0.64</td><td>273</td></tr> <tr> <td>1</td><td>0.57</td><td>0.63</td><td>0.60</td><td>222</td></tr> <tr> <td>accuracy</td><td></td><td></td><td>0.62</td><td>495</td></tr> <tr> <td>macro avg</td><td>0.62</td><td>0.62</td><td>0.62</td><td>495</td></tr> <tr> <td>weighted avg</td><td>0.62</td><td>0.62</td><td>0.62</td><td>495</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.67	0.61	0.64	273	1	0.57	0.63	0.60	222	accuracy			0.62	495	macro avg	0.62	0.62	0.62	495	weighted avg	0.62	0.62	0.62	495
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.74	0.65	0.69	307																																																																			
1	0.51	0.61	0.56	184																																																																			
accuracy			0.64	491																																																																			
macro avg	0.62	0.63	0.62	491																																																																			
weighted avg	0.65	0.64	0.64	491																																																																			
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.67	0.61	0.64	273																																																																			
1	0.57	0.63	0.60	222																																																																			
accuracy			0.62	495																																																																			
macro avg	0.62	0.62	0.62	495																																																																			
weighted avg	0.62	0.62	0.62	495																																																																			
VNM	GAS																																																																						
<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.66</td><td>0.66</td><td>0.66</td><td>274</td></tr> <tr> <td>1</td><td>0.58</td><td>0.58</td><td>0.58</td><td>221</td></tr> <tr> <td>accuracy</td><td></td><td></td><td>0.62</td><td>495</td></tr> <tr> <td>macro avg</td><td>0.62</td><td>0.62</td><td>0.62</td><td>495</td></tr> <tr> <td>weighted avg</td><td>0.62</td><td>0.62</td><td>0.62</td><td>495</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.66	0.66	0.66	274	1	0.58	0.58	0.58	221	accuracy			0.62	495	macro avg	0.62	0.62	0.62	495	weighted avg	0.62	0.62	0.62	495	<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.57</td><td>0.64</td><td>0.60</td><td>236</td></tr> <tr> <td>1</td><td>0.63</td><td>0.56</td><td>0.59</td><td>259</td></tr> <tr> <td>accuracy</td><td></td><td></td><td>0.60</td><td>495</td></tr> <tr> <td>macro avg</td><td>0.60</td><td>0.60</td><td>0.60</td><td>495</td></tr> <tr> <td>weighted avg</td><td>0.60</td><td>0.60</td><td>0.60</td><td>495</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.57	0.64	0.60	236	1	0.63	0.56	0.59	259	accuracy			0.60	495	macro avg	0.60	0.60	0.60	495	weighted avg	0.60	0.60	0.60	495
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.66	0.66	0.66	274																																																																			
1	0.58	0.58	0.58	221																																																																			
accuracy			0.62	495																																																																			
macro avg	0.62	0.62	0.62	495																																																																			
weighted avg	0.62	0.62	0.62	495																																																																			
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.57	0.64	0.60	236																																																																			
1	0.63	0.56	0.59	259																																																																			
accuracy			0.60	495																																																																			
macro avg	0.60	0.60	0.60	495																																																																			
weighted avg	0.60	0.60	0.60	495																																																																			
HVN	MWG																																																																						
<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.52</td><td>0.65</td><td>0.58</td><td>144</td></tr> <tr> <td>1</td><td>0.70</td><td>0.56</td><td>0.62</td><td>282</td></tr> <tr> <td>accuracy</td><td></td><td></td><td>0.60</td><td>346</td></tr> <tr> <td>macro avg</td><td>0.61</td><td>0.61</td><td>0.60</td><td>346</td></tr> <tr> <td>weighted avg</td><td>0.62</td><td>0.60</td><td>0.60</td><td>346</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.52	0.65	0.58	144	1	0.70	0.56	0.62	282	accuracy			0.60	346	macro avg	0.61	0.61	0.60	346	weighted avg	0.62	0.60	0.60	346	<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.48</td><td>0.60</td><td>0.53</td><td>184</td></tr> <tr> <td>1</td><td>0.72</td><td>0.61</td><td>0.66</td><td>312</td></tr> <tr> <td>accuracy</td><td></td><td></td><td>0.61</td><td>496</td></tr> <tr> <td>macro avg</td><td>0.60</td><td>0.61</td><td>0.60</td><td>496</td></tr> <tr> <td>weighted avg</td><td>0.63</td><td>0.61</td><td>0.61</td><td>496</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.48	0.60	0.53	184	1	0.72	0.61	0.66	312	accuracy			0.61	496	macro avg	0.60	0.61	0.60	496	weighted avg	0.63	0.61	0.61	496
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.52	0.65	0.58	144																																																																			
1	0.70	0.56	0.62	282																																																																			
accuracy			0.60	346																																																																			
macro avg	0.61	0.61	0.60	346																																																																			
weighted avg	0.62	0.60	0.60	346																																																																			
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.48	0.60	0.53	184																																																																			
1	0.72	0.61	0.66	312																																																																			
accuracy			0.61	496																																																																			
macro avg	0.60	0.61	0.60	496																																																																			
weighted avg	0.63	0.61	0.61	496																																																																			
FPT	HPG																																																																						
<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.49</td><td>0.60</td><td>0.54</td><td>191</td></tr> <tr> <td>1</td><td>0.71</td><td>0.61</td><td>0.65</td><td>305</td></tr> <tr> <td>accuracy</td><td></td><td></td><td>0.60</td><td>496</td></tr> <tr> <td>macro avg</td><td>0.60</td><td>0.60</td><td>0.60</td><td>496</td></tr> <tr> <td>weighted avg</td><td>0.62</td><td>0.60</td><td>0.61</td><td>496</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.49	0.60	0.54	191	1	0.71	0.61	0.65	305	accuracy			0.60	496	macro avg	0.60	0.60	0.60	496	weighted avg	0.62	0.60	0.61	496	<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.49</td><td>0.67</td><td>0.57</td><td>189</td></tr> <tr> <td>1</td><td>0.74</td><td>0.58</td><td>0.65</td><td>309</td></tr> <tr> <td>accuracy</td><td></td><td></td><td>0.61</td><td>498</td></tr> <tr> <td>macro avg</td><td>0.62</td><td>0.62</td><td>0.61</td><td>498</td></tr> <tr> <td>weighted avg</td><td>0.65</td><td>0.61</td><td>0.62</td><td>498</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.49	0.67	0.57	189	1	0.74	0.58	0.65	309	accuracy			0.61	498	macro avg	0.62	0.62	0.61	498	weighted avg	0.65	0.61	0.62	498
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.49	0.60	0.54	191																																																																			
1	0.71	0.61	0.65	305																																																																			
accuracy			0.60	496																																																																			
macro avg	0.60	0.60	0.60	496																																																																			
weighted avg	0.62	0.60	0.61	496																																																																			
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.49	0.67	0.57	189																																																																			
1	0.74	0.58	0.65	309																																																																			
accuracy			0.61	498																																																																			
macro avg	0.62	0.62	0.61	498																																																																			
weighted avg	0.65	0.61	0.62	498																																																																			
MSN	SAB																																																																						
<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.62</td><td>0.64</td><td>0.63</td><td>250</td></tr> <tr> <td>1</td><td>0.62</td><td>0.60</td><td>0.61</td><td>244</td></tr> <tr> <td>accuracy</td><td></td><td></td><td>0.62</td><td>494</td></tr> <tr> <td>macro avg</td><td>0.62</td><td>0.62</td><td>0.62</td><td>494</td></tr> <tr> <td>weighted avg</td><td>0.62</td><td>0.62</td><td>0.62</td><td>494</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.62	0.64	0.63	250	1	0.62	0.60	0.61	244	accuracy			0.62	494	macro avg	0.62	0.62	0.62	494	weighted avg	0.62	0.62	0.62	494	<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.90</td><td>0.61</td><td>0.73</td><td>270</td></tr> <tr> <td>1</td><td>0.34</td><td>0.74</td><td>0.47</td><td>74</td></tr> <tr> <td>accuracy</td><td></td><td></td><td>0.64</td><td>344</td></tr> <tr> <td>macro avg</td><td>0.62</td><td>0.68</td><td>0.60</td><td>344</td></tr> <tr> <td>weighted avg</td><td>0.78</td><td>0.64</td><td>0.67</td><td>344</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.90	0.61	0.73	270	1	0.34	0.74	0.47	74	accuracy			0.64	344	macro avg	0.62	0.68	0.60	344	weighted avg	0.78	0.64	0.67	344
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.62	0.64	0.63	250																																																																			
1	0.62	0.60	0.61	244																																																																			
accuracy			0.62	494																																																																			
macro avg	0.62	0.62	0.62	494																																																																			
weighted avg	0.62	0.62	0.62	494																																																																			
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.90	0.61	0.73	270																																																																			
1	0.34	0.74	0.47	74																																																																			
accuracy			0.64	344																																																																			
macro avg	0.62	0.68	0.60	344																																																																			
weighted avg	0.78	0.64	0.67	344																																																																			

Hình 4. 60 Độ chính xác khi dự đoán bằng mô hình LSTM

- Với mô hình ARIMA, chúng tôi thu được độ chính xác của mô hình là:

<pre> a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1].values, index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:] &gt; 0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}") </pre> <p>Accuracy: 0.51</p>	<pre> a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1].values, index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:] &gt; 0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}") </pre> <p>Accuracy: 0.53</p>
<p><b>VIC</b></p>	<p><b>VCB</b></p>
<pre> a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1].values, index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:] &gt; 0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}") </pre> <p>Accuracy: 0.54</p>	<pre> a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1].values, index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:] &gt; 0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}") </pre> <p>Accuracy: 0.54</p>
<p><b>VNM</b></p>	<p><b>GAS</b></p>
<pre> a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1].values, index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:] &gt; 0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}") </pre> <p>Accuracy: 0.54</p>	<pre> a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1].values, index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:] &gt; 0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}") </pre> <p>Accuracy: 0.54</p>
<p><b>HVN</b></p>	<p><b>MWG</b></p>
<pre> a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1].values, index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:] &gt; 0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}") </pre> <p>Accuracy: 0.54</p>	<pre> a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1].values, index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:] &gt; 0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}") </pre> <p>Accuracy: 0.52</p>
<p><b>FPT</b></p>	<p><b>HPG</b></p>
<pre> a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1].values, index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:] &gt; 0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}") </pre> <p>Accuracy: 0.53</p>	<pre> a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1].values, index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:] &gt; 0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}") </pre> <p>Accuracy: 0.52</p>
<p><b>MSN</b></p>	<p><b>SAB</b></p>
<pre> a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1].values, index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:] &gt; 0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}") </pre> <p>Accuracy: 0.54</p>	<pre> a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1].values, index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:] &gt; 0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}") </pre> <p>Accuracy: 0.56</p>

Hình 4. 61 Độ chính xác khi dự đoán bằng mô hình ARIMA

Qua các kết quả từ các mô hình, chúng tôi nhận thấy rằng mô hình Random Forest cho ra nhiều kết quả dự đoán với độ chính xác khá cao, tiếp đến là mô hình Logistic Regression cũng cho ra kết quả dự đoán tốt, kế đến là mô hình LSTM và cuối cùng là mô hình ARIMA cho ra kết quả dự đoán thấp hơn những mô hình còn lại.

#### 4.4.2 Với dữ liệu tổng hợp các công ty

- Với mô hình Logistic Regression, chúng tôi thu được kết quả dự đoán như sau:

```

▶ predictions = lr.predict(x_test_scaled)
predictions

@ array([1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1])

[ ] y_test.values
array([0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1])

[ ] accuracy = accuracy_score(y_test.values, predictions)
print(f"Accuracy: {accuracy:.2f}")

Accuracy: 0.79

```

Hình 4. 62 Kết quả dự đoán của mô hình Logistic Regression

- Với mô hình Random Forest, chúng tôi thu được kết quả dự đoán như sau:

```

▶ y_pred = rf.predict(x_test.values)
y_pred

@ array([0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1])

[ ] y_test.values
array([0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1])

[ ] accuracy = accuracy_score(y_test.values, y_pred)
print(f"Accuracy: {accuracy:.2f}")

Accuracy: 0.74

```

Hình 4. 63 Kết quả dự đoán của mô hình Random Forest

- Với mô hình LSTM, chúng tôi thu được kết quả dự đoán như sau:

```

▶ accuracy = accuracy_score(prediction, y_test)
classification_rep = classification_report(prediction, y_test)
print("Accuracy: ", accuracy)
print("Classification Report:\n", classification_rep)

@ Accuracy: 0.75
Classification Report:
precision    recall   f1-score   support
          0       0.00     0.00      0.00        1
          1       0.75     1.00      0.86        3

           accuracy            0.75
          macro avg       0.38     0.50      0.43        4
      weighted avg       0.56     0.75      0.64        4

```

Hình 4. 64 Kết quả dự đoán của mô hình LSTM

- Với mô hình ARIMA, chúng tôi thu được kết quả dự đoán như sau:

```

▶ a = predict_series
    index = pd.to_datetime(train_data.index.values[-1])
    b = pd.Series(train_data.iloc[-1], index=[index])
    predict = pd.concat([b,a])
    predict = predict[1:]
    trend_predict = predict.pct_change()*100 > 0
    trend_predict = (trend_predict.iloc[:] > 0).astype(int).values
    accuracy = accuracy_score(trend_predict, trend_test)
    print(f"Accuracy: {accuracy: .2f}")

└ Accuracy: 0.56

```

Hình 4. 65 Kết quả dự đoán mô hình ARIMA

Qua kết quả dự đoán ở phía trên, chúng tôi thấy được rằng mô hình Logistic Regression cho ra được kết quả với độ chính xác cao nhất với kết quả khá cao, thứ hai là mô hình LSTM cho ra kết quả độ chính xác cao xấp xỉ, kế đến là mô hình Random Forest và cuối cùng là mô hình ARIMA.

#### **4.4.3 Với dữ liệu của các công ty theo tháng**

- Với mô hình Logistic Regression, chúng tôi thu được kết quả như sau:

<p><b>VIC</b></p> <pre> ● predictions = lr.predict(X_test_scaled) predictions array([1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1])  [ ] y_test.values array([1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1])  [ ] accuracy = accuracy_score(y_test.values, predictions) print("Accuracy: {:.2f}")  Accuracy: 0.77 </pre>	<p><b>VCB</b></p> <pre> ● predictions = lr.predict(X_test_scaled) predictions array([1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1])  [ ] y_test.values array([1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1])  [ ] accuracy = accuracy_score(y_test.values, predictions) print("Accuracy: {:.2f}")  Accuracy: 0.77 </pre>
<p><b>VNM</b></p> <pre> ● predictions = lr.predict(X_test_scaled) predictions array([1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1])  [ ] y_test.values array([0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1])  [ ] accuracy = accuracy_score(y_test.values, predictions) print("Accuracy: {:.2f}")  Accuracy: 0.82 </pre>	<p><b>GAS</b></p> <pre> ● predictions = lr.predict(X_test_scaled) predictions array([1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1])  [ ] y_test.values array([1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0])  [ ] accuracy = accuracy_score(y_test.values, predictions) print("Accuracy: {:.2f}")  Accuracy: 0.77 </pre>
<p><b>HVN</b></p> <pre> ● predictions = lr.predict(X_test_scaled) predictions array([0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0])  [ ] y_test.values array([0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0])  [ ] accuracy = accuracy_score(y_test.values, predictions) print("Accuracy: {:.2f}")  Accuracy: 0.79 </pre>	<p><b>MWG</b></p> <pre> ● predictions = lr.predict(X_test_scaled) predictions array([1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0])  [ ] y_test.values array([1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0])  [ ] accuracy = accuracy_score(y_test.values, predictions) print("Accuracy: {:.2f}")  Accuracy: 0.81 </pre>
<p><b>FPT</b></p> <pre> ● predictions = lr.predict(X_test_scaled) predictions array([1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1])  [ ] y_test.values array([1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0])  [ ] accuracy = accuracy_score(y_test.values, predictions) print("Accuracy: {:.2f}")  Accuracy: 0.73 </pre>	<p><b>HPG</b></p> <pre> ● predictions = lr.predict(X_test_scaled) predictions array([0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1])  [ ] y_test.values array([0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0])  [ ] accuracy = accuracy_score(y_test.values, predictions) print("Accuracy: {:.2f}")  Accuracy: 0.73 </pre>
<p><b>MSN</b></p> <pre> ● predictions = lr.predict(X_test_scaled) predictions array([1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1])  [ ] y_test.values array([0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1])  [ ] accuracy = accuracy_score(y_test.values, predictions) print("Accuracy: {:.2f}")  Accuracy: 0.77 </pre>	<p><b>SAB</b></p> <pre> ● predictions = lr.predict(X_test_scaled) predictions array([0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0])  [ ] y_test.values array([0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1])  [ ] accuracy = accuracy_score(y_test.values, predictions) print("Accuracy: {:.2f}")  Accuracy: 0.69 </pre>

Hình 4. 66 Kết quả dự đoán với mô hình Logistic Regression

- Với mô hình Random Forest, nhóm chúng tôi thu được kết quả như sau:

<p><b>VIC</b></p> <pre> y_pred = rf.predict(X_test.values) y_pred array([0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0]) [ ] y_test.values array([0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0]) [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {:.2f}") Accuracy: 0.73 </pre>	<p><b>VCB</b></p> <pre> y_pred = rf.predict(X_test.values) y_pred array([1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0]) [ ] y_test.values array([1, 0, 0, 0, 0]) [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {:.2f}") Accuracy: 0.81 </pre>
<p><b>VNM</b></p> <pre> y_pred = rf.predict(X_test.values) y_pred array([0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1]) [ ] y_test.values array([0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0]) [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {:.2f}") Accuracy: 0.72 </pre>	<p><b>GAS</b></p> <pre> y_pred = rf.predict(X_test.values) y_pred array([1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0]) [ ] y_test.values array([1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0]) [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {:.2f}") Accuracy: 0.68 </pre>
<p><b>HVN</b></p> <pre> y_pred = rf.predict(X_test.values) y_pred array([1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]) [ ] y_test.values array([0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0]) [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {:.2f}") Accuracy: 0.59 </pre>	<p><b>MWG</b></p> <pre> y_pred = rf.predict(X_test.values) y_pred array([1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1]) [ ] y_test.values array([1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0]) [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {:.2f}") Accuracy: 0.82 </pre>
<p><b>FPT</b></p> <pre> y_pred = rf.predict(X_test.values) y_pred array([1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1]) [ ] y_test.values array([1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1]) [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {:.2f}") Accuracy: 0.73 </pre>	<p><b>HPG</b></p> <pre> y_pred = rf.predict(X_test.values) y_pred array([0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0]) [ ] y_test.values array([0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1]) [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {:.2f}") Accuracy: 0.61 </pre>
<p><b>MSN</b></p> <pre> y_pred = rf.predict(X_test.values) y_pred array([1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0]) [ ] y_test.values array([1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0]) [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {:.2f}") Accuracy: 0.74 </pre>	<p><b>SAB</b></p> <pre> y_pred = rf.predict(X_test.values) y_pred array([1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1]) [ ] y_test.values array([1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0]) [ ] accuracy = accuracy_score(y_test.values, y_pred) print("Accuracy: {:.2f}") Accuracy: 0.69 </pre>

Hình 4. 67 Kết quả dự đoán với mô hình Random Forest

- Với mô hình LSTM, nhóm chúng tôi thu được kết quả như sau:

VIC	VCB																																																																						
<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.53</td><td>1.00</td><td>0.70</td><td>8</td></tr> <tr> <td>1</td><td>1.00</td><td>0.30</td><td>0.46</td><td>10</td></tr> <tr> <td>accuracy</td><td>0.77</td><td>0.65</td><td>0.58</td><td>18</td></tr> <tr> <td>macro avg</td><td>0.79</td><td>0.61</td><td>0.57</td><td>18</td></tr> <tr> <td>weighted avg</td><td>0.79</td><td>0.61</td><td>0.57</td><td>18</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.53	1.00	0.70	8	1	1.00	0.30	0.46	10	accuracy	0.77	0.65	0.58	18	macro avg	0.79	0.61	0.57	18	weighted avg	0.79	0.61	0.57	18	<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.30</td><td>1.00</td><td>0.46</td><td>3</td></tr> <tr> <td>1</td><td>1.00</td><td>0.53</td><td>0.70</td><td>15</td></tr> <tr> <td>accuracy</td><td>0.61</td><td>0.61</td><td>0.58</td><td>18</td></tr> <tr> <td>macro avg</td><td>0.65</td><td>0.77</td><td>0.58</td><td>18</td></tr> <tr> <td>weighted avg</td><td>0.68</td><td>0.61</td><td>0.66</td><td>18</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.30	1.00	0.46	3	1	1.00	0.53	0.70	15	accuracy	0.61	0.61	0.58	18	macro avg	0.65	0.77	0.58	18	weighted avg	0.68	0.61	0.66	18
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.53	1.00	0.70	8																																																																			
1	1.00	0.30	0.46	10																																																																			
accuracy	0.77	0.65	0.58	18																																																																			
macro avg	0.79	0.61	0.57	18																																																																			
weighted avg	0.79	0.61	0.57	18																																																																			
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.30	1.00	0.46	3																																																																			
1	1.00	0.53	0.70	15																																																																			
accuracy	0.61	0.61	0.58	18																																																																			
macro avg	0.65	0.77	0.58	18																																																																			
weighted avg	0.68	0.61	0.66	18																																																																			
<b>VNM</b>	<b>GAS</b>																																																																						
<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>1.00</td><td>0.67</td><td>0.80</td><td>18</td></tr> <tr> <td>1</td><td>0.00</td><td>0.00</td><td>0.00</td><td>0</td></tr> <tr> <td>accuracy</td><td>0.50</td><td>0.33</td><td>0.40</td><td>18</td></tr> <tr> <td>macro avg</td><td>1.00</td><td>0.67</td><td>0.80</td><td>18</td></tr> <tr> <td>weighted avg</td><td>1.00</td><td>0.67</td><td>0.80</td><td>18</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	1.00	0.67	0.80	18	1	0.00	0.00	0.00	0	accuracy	0.50	0.33	0.40	18	macro avg	1.00	0.67	0.80	18	weighted avg	1.00	0.67	0.80	18	<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.40</td><td>0.80</td><td>0.53</td><td>5</td></tr> <tr> <td>1</td><td>0.88</td><td>0.54</td><td>0.67</td><td>13</td></tr> <tr> <td>accuracy</td><td>0.64</td><td>0.67</td><td>0.60</td><td>18</td></tr> <tr> <td>macro avg</td><td>0.74</td><td>0.61</td><td>0.63</td><td>18</td></tr> <tr> <td>weighted avg</td><td>0.74</td><td>0.61</td><td>0.63</td><td>18</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.40	0.80	0.53	5	1	0.88	0.54	0.67	13	accuracy	0.64	0.67	0.60	18	macro avg	0.74	0.61	0.63	18	weighted avg	0.74	0.61	0.63	18
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	1.00	0.67	0.80	18																																																																			
1	0.00	0.00	0.00	0																																																																			
accuracy	0.50	0.33	0.40	18																																																																			
macro avg	1.00	0.67	0.80	18																																																																			
weighted avg	1.00	0.67	0.80	18																																																																			
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.40	0.80	0.53	5																																																																			
1	0.88	0.54	0.67	13																																																																			
accuracy	0.64	0.67	0.60	18																																																																			
macro avg	0.74	0.61	0.63	18																																																																			
weighted avg	0.74	0.61	0.63	18																																																																			
<b>HVN</b>	<b>MWG</b>																																																																						
<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.83</td><td>0.83</td><td>0.83</td><td>6</td></tr> <tr> <td>1</td><td>0.83</td><td>0.83</td><td>0.83</td><td>6</td></tr> <tr> <td>accuracy</td><td>0.83</td><td>0.83</td><td>0.83</td><td>12</td></tr> <tr> <td>macro avg</td><td>0.83</td><td>0.83</td><td>0.83</td><td>12</td></tr> <tr> <td>weighted avg</td><td>0.83</td><td>0.83</td><td>0.83</td><td>12</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.83	0.83	0.83	6	1	0.83	0.83	0.83	6	accuracy	0.83	0.83	0.83	12	macro avg	0.83	0.83	0.83	12	weighted avg	0.83	0.83	0.83	12	<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.50</td><td>1.00</td><td>0.67</td><td>6</td></tr> <tr> <td>1</td><td>1.00</td><td>0.45</td><td>0.62</td><td>11</td></tr> <tr> <td>accuracy</td><td>0.75</td><td>0.73</td><td>0.65</td><td>17</td></tr> <tr> <td>macro avg</td><td>0.82</td><td>0.65</td><td>0.64</td><td>17</td></tr> <tr> <td>weighted avg</td><td>0.82</td><td>0.65</td><td>0.64</td><td>17</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.50	1.00	0.67	6	1	1.00	0.45	0.62	11	accuracy	0.75	0.73	0.65	17	macro avg	0.82	0.65	0.64	17	weighted avg	0.82	0.65	0.64	17
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.83	0.83	0.83	6																																																																			
1	0.83	0.83	0.83	6																																																																			
accuracy	0.83	0.83	0.83	12																																																																			
macro avg	0.83	0.83	0.83	12																																																																			
weighted avg	0.83	0.83	0.83	12																																																																			
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.50	1.00	0.67	6																																																																			
1	1.00	0.45	0.62	11																																																																			
accuracy	0.75	0.73	0.65	17																																																																			
macro avg	0.82	0.65	0.64	17																																																																			
weighted avg	0.82	0.65	0.64	17																																																																			
<b>FPT</b>	<b>HPG</b>																																																																						
<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.70</td><td>0.64</td><td>0.67</td><td>11</td></tr> <tr> <td>1</td><td>0.50</td><td>0.57</td><td>0.53</td><td>7</td></tr> <tr> <td>accuracy</td><td>0.60</td><td>0.60</td><td>0.60</td><td>18</td></tr> <tr> <td>macro avg</td><td>0.62</td><td>0.61</td><td>0.61</td><td>18</td></tr> <tr> <td>weighted avg</td><td>0.62</td><td>0.61</td><td>0.61</td><td>18</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.70	0.64	0.67	11	1	0.50	0.57	0.53	7	accuracy	0.60	0.60	0.60	18	macro avg	0.62	0.61	0.61	18	weighted avg	0.62	0.61	0.61	18	<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.70</td><td>1.00</td><td>0.82</td><td>7</td></tr> <tr> <td>1</td><td>1.00</td><td>0.73</td><td>0.84</td><td>11</td></tr> <tr> <td>accuracy</td><td>0.85</td><td>0.86</td><td>0.83</td><td>18</td></tr> <tr> <td>macro avg</td><td>0.88</td><td>0.83</td><td>0.83</td><td>18</td></tr> <tr> <td>weighted avg</td><td>0.88</td><td>0.83</td><td>0.83</td><td>18</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.70	1.00	0.82	7	1	1.00	0.73	0.84	11	accuracy	0.85	0.86	0.83	18	macro avg	0.88	0.83	0.83	18	weighted avg	0.88	0.83	0.83	18
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.70	0.64	0.67	11																																																																			
1	0.50	0.57	0.53	7																																																																			
accuracy	0.60	0.60	0.60	18																																																																			
macro avg	0.62	0.61	0.61	18																																																																			
weighted avg	0.62	0.61	0.61	18																																																																			
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.70	1.00	0.82	7																																																																			
1	1.00	0.73	0.84	11																																																																			
accuracy	0.85	0.86	0.83	18																																																																			
macro avg	0.88	0.83	0.83	18																																																																			
weighted avg	0.88	0.83	0.83	18																																																																			
<b>MSN</b>	<b>SAB</b>																																																																						
<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>0.86</td><td>0.75</td><td>0.80</td><td>16</td></tr> <tr> <td>1</td><td>0.00</td><td>0.00</td><td>0.00</td><td>2</td></tr> <tr> <td>accuracy</td><td>0.43</td><td>0.38</td><td>0.40</td><td>18</td></tr> <tr> <td>macro avg</td><td>0.76</td><td>0.67</td><td>0.71</td><td>18</td></tr> <tr> <td>weighted avg</td><td>0.76</td><td>0.67</td><td>0.71</td><td>18</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	0.86	0.75	0.80	16	1	0.00	0.00	0.00	2	accuracy	0.43	0.38	0.40	18	macro avg	0.76	0.67	0.71	18	weighted avg	0.76	0.67	0.71	18	<pre>accuracy = accuracy_score(prediction, y_test) classification_rep = classification_report(prediction, y_test) print("Accuracy: ", accuracy) print("Classification Report:\n", classification_rep)</pre> <table border="1"> <thead> <tr> <th colspan="5">Classification Report:</th> </tr> <tr> <th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr> </thead> <tbody> <tr> <td>0</td><td>1.00</td><td>0.62</td><td>0.76</td><td>13</td></tr> <tr> <td>1</td><td>0.00</td><td>0.00</td><td>0.00</td><td>0</td></tr> <tr> <td>accuracy</td><td>0.50</td><td>0.31</td><td>0.38</td><td>13</td></tr> <tr> <td>macro avg</td><td>1.00</td><td>0.62</td><td>0.76</td><td>13</td></tr> <tr> <td>weighted avg</td><td>1.00</td><td>0.62</td><td>0.76</td><td>13</td></tr> </tbody> </table>	Classification Report:						precision	recall	f1-score	support	0	1.00	0.62	0.76	13	1	0.00	0.00	0.00	0	accuracy	0.50	0.31	0.38	13	macro avg	1.00	0.62	0.76	13	weighted avg	1.00	0.62	0.76	13
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	0.86	0.75	0.80	16																																																																			
1	0.00	0.00	0.00	2																																																																			
accuracy	0.43	0.38	0.40	18																																																																			
macro avg	0.76	0.67	0.71	18																																																																			
weighted avg	0.76	0.67	0.71	18																																																																			
Classification Report:																																																																							
	precision	recall	f1-score	support																																																																			
0	1.00	0.62	0.76	13																																																																			
1	0.00	0.00	0.00	0																																																																			
accuracy	0.50	0.31	0.38	13																																																																			
macro avg	1.00	0.62	0.76	13																																																																			
weighted avg	1.00	0.62	0.76	13																																																																			

Hình 4. 68 Kết quả dự đoán với mô hình LSTM

- Với mô hình ARIMA, chúng tôi thu được kết quả dự đoán như sau:

<p><b>VIC</b></p> <pre>a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1], index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:]&gt;0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}")  </pre> <p>Accuracy: 0.61</p>	<p><b>VCB</b></p> <pre>a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1], index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:]&gt;0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}")  </pre> <p>Accuracy: 0.56</p>
<p><b>VNM</b></p> <pre>a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1], index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:]&gt;0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}")  </pre> <p>Accuracy: 0.67</p>	<p><b>GAS</b></p> <pre>a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1], index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:]&gt;0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}")  </pre> <p>Accuracy: 0.61</p>
<p><b>HVN</b></p> <pre>a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1], index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:]&gt;0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}")  </pre> <p>Accuracy: 0.62</p>	<p><b>MWG</b></p> <pre>a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1], index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:]&gt;0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}")  </pre> <p>Accuracy: 0.65</p>
<p><b>FPT</b></p> <pre>a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1], index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:]&gt;0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}")  </pre> <p>Accuracy: 0.72</p>	<p><b>HPG</b></p> <pre>a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1], index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:]&gt;0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}")  </pre> <p>Accuracy: 0.67</p>
<p><b>MSN</b></p> <pre>a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1], index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:]&gt;0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}")  </pre> <p>Accuracy: 0.61</p>	<p><b>SAB</b></p> <pre>a = predict_series index = pd.to_datetime(train_data.index.values[-1]) b = pd.Series(train_data.iloc[-1], index=[index]) predict = pd.concat([b,a]) predict = predict[1:] trend_predict = predict.pct_change()*100 &gt; 0 trend_predict = (trend_predict.iloc[:]&gt;0).astype(int).values accuracy = accuracy_score(trend_predict, trend_test) print(f"Accuracy: {accuracy: .2f}")  </pre> <p>Accuracy: 0.69</p>

Hình 4. 69 Kết quả dự đoán với mô hình ARIMA

Qua các kết quả dự đoán của các mô hình, nhìn chung thì chúng tôi nhận thấy rằng mô hình logistic regression cho ra được nhiều kết quả dự đoán cao nên mô hình này có hiệu suất tốt nhất, đến tiếp theo là mô hình Random Forest cũng cho ra những dự đoán với độ chính xác tốt. Lần này thì mô hình LSTM chỉ nhỉnh hơn mô hình ARIMA một chút tuy vậy cả hai mô hình đều đã cho ra dự đoán tốt.

#### **4.4.4 Với dữ liệu tổng hợp các công ty theo tháng**

- Với mô hình Logistic Regression, chúng tôi thu được kết quả như sau:

```

▶ predictions = lr.predict(x_test_scaled)
predictions
array([1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1])

[ ] y_test.values
array([0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0])

[ ] accuracy = accuracy_score(y_test.values, predictions)
print(f"Accuracy: {accuracy:.2f}")

Accuracy: 0.80

```

Hình 4. 70 Kết quả dự đoán mô hình Logistic Regression

- Với mô hình Random Forest, chúng tôi thu được kết quả như sau:

```

▶ y_pred = rf.predict(x_test.values)
y_pred
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0])

[ ] y_test.values
array([0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0])

[ ] accuracy = accuracy_score(y_test.values, y_pred)
print(f"Accuracy: {accuracy:.2f}")

Accuracy: 0.73

```

Hình 4. 71 Kết quả dự đoán mô hình Random Forest

- Với mô hình LSTM, chúng tôi thu được kết quả như sau:

```

● accuracy = accuracy_score(prediction, y_test)
classification_rep = classification_report(prediction, y_test)
print("Accuracy: ", accuracy)
print("Classification Report:\n", classification_rep)

☒ Accuracy: 0.75
Classification Report:
precision    recall   f1-score   support
          0       0.89      0.80      0.84      10
          1       0.33      0.50      0.40       2

           accuracy
macro avg       0.61      0.65      0.62      12
weighted avg     0.80      0.75      0.77      12

```

Hình 4. 72 Kết quả dự đoán mô hình LSTM

- Với mô hình ARIMA, chúng tôi thu được kết quả như sau:

```

● a = predict_series
index = pd.to_datetime(train_data.index.values[-1])
b = pd.Series(train_data.iloc[-1], index=[index])
predict = pd.concat([b,a])
predict = predict[1:]
trend_predict = predict.pct_change()*100 > 0
trend_predict = (trend_predict.iloc[:]>0).astype(int).values
accuracy = accuracy_score(trend_predict, trend_test)
print(f"Accuracy: {accuracy: .2f}")

☒ Accuracy: 0.62

```

Hình 4. 73 Kết quả dự đoán mô hình ARIMA

Thông qua các kết quả thu được, chúng tôi nhận thấy rằng kết quả tốt nhất đến từ mô hình Logistic Regression với kết quả khá cao, tiếp đến là đến mô hình LSTM với kết quả dự đoán tốt, kế đến là mô hình Random Forest với kết quả dự đoán tốt. Mô hình ARIMA đã cải thiện về độ chính xác hơn nhưng tập dữ liệu trước.

#### **4.4.5 Mô hình dự đoán dữ liệu mới**

Thông qua các kết quả dự đoán của nhiều miền dữ liệu khác nhau và sự so sánh về độ chính xác, chúng tôi có thể khẳng định rằng hai mô hình Logistic Regression và mô hình Random Forest đều đã đạt tối ưu tốt và cho ra những dự đoán tốt với nhiều giá trị chính xác cao. Chúng tôi sẽ đưa ra một miền dữ liệu mới là dữ liệu chứng khoán từ đầu tháng 1 đến đầu tháng 3 năm 2024 và cho hai mô hình dự đoán để kiểm tra độ chính cũng như khẳng định lần nữa về ý kiến của chúng tôi.

#### 4.4.5.1 Xử lý dữ liệu mới

Trước tiên, chúng tôi sẽ lấy miền dữ liệu mới theo dự định thông qua dữ liệu vnquant đã được thực hiện trước đó.

```
[ ] new_stock = {}
for i in range(len(stock_symbols)):
    data = dt.DataLoader(symbols=stock_symbols[i],
                          start="2024-01-01",
                          end="2024-03-01",
                          data_source="VNDS")
    new_stock[stock_symbols[i]] = data.download()

[ ] new_stock[stock_symbols[0]].head()

Attributes  high   low  open  close  avg  volume
Symbols      VIC    VIC    VIC    VIC    VIC    VIC
date
2024-01-02  44.95  44.0  44.95  44.00  44.35  2324300.0
2024-01-03  44.15  43.5  43.50  44.15  43.73  2347100.0
2024-01-04  44.40  43.8  44.15  44.15  44.14  2380800.0
2024-01-05  44.20  43.9  44.15  44.10  44.03  1553600.0
2024-01-08  44.75  44.1  44.45  44.35  44.43  2577400.0
```

Hình 4. 74 Lấy miền dữ liệu mới

Sau khi đã có được dữ liệu mới, chúng tôi tiến hành tiền xử lý dữ liệu như đã phân tích ở hai phần đầu của mô hình xử lý dữ liệu đã trình bày ở trên. Thực hiện việc tính toán các chỉ số, chỉ báo và thêm vào miền dữ liệu tương tự như các phần trước đó.

Tuy nhiên, lần này nhóm chúng tôi sẽ sử dụng miền dữ liệu trước đó sử dụng để làm dữ liệu huấn luyện cho mô hình. Sau đó, cho mô hình dự đoán xu hướng với tập kiểm thử chính là miền dữ lợi mới này. Tiến hành việc kiểm tra độ chính xác của mô hình để rút ra được kết luận.

```
❶ X_train = df[stock_symbols[0]].loc[:, df[stock_symbols[0]].columns != ('trend', stock_symbols[0])]
y_train = df[stock_symbols[0]][('trend', stock_symbols[0])]

X_test = df_new_stock[stock_symbols[0]].loc[:, df_new_stock[stock_symbols[0]].columns != ('trend', stock_symbols[0])]
y_test = df_new_stock[stock_symbols[0]][('trend', stock_symbols[0])]

scaler = MinMaxScaler()
X_train_scaled = scaler.fit_transform(X_train.values)
X_test_scaled = scaler.transform(X_test.values)

lr = LogisticRegression(penalty='l2', C=0.2, random_state=42)
lr.fit(X_train_scaled, y_train.values)
```

Hình 4. 75 Miền dữ liệu mới được đưa vào dự đoán mô hình Logistic Regression

```

❷ X_train = df[stock_symbols[0]][[('volume', stock_symbols[0]), ('ema50', stock_symbols[0]), ('ema21', stock_symbols[0]),
   ('ema14', stock_symbols[0]), ('ema5', stock_symbols[0]), ('rsi', stock_symbols[0]),
   ('macd', stock_symbols[0]), ('roc', stock_symbols[0]), ('obv', stock_symbols[0]),
   ('atr', stock_symbols[0]), ('cmf', stock_symbols[0]), ('emv', stock_symbols[0]),
   ('stoch', stock_symbols[0]), ('cci', stock_symbols[0]), ('mfi', stock_symbols[0])]]
y_train = df[stock_symbols[0]][['trend', stock_symbols[0]]]

X_test = df_new_stock[stock_symbols[0]][[('volume', stock_symbols[0]), ('ema50', stock_symbols[0]), ('ema21', stock_symbols[0]),
   ('ema14', stock_symbols[0]), ('ema5', stock_symbols[0]), ('rsi', stock_symbols[0]),
   ('macd', stock_symbols[0]), ('roc', stock_symbols[0]), ('obv', stock_symbols[0]),
   ('atr', stock_symbols[0]), ('cmf', stock_symbols[0]), ('emv', stock_symbols[0]),
   ('stoch', stock_symbols[0]), ('cci', stock_symbols[0]), ('mfi', stock_symbols[0])]]
y_test = df_new_stock[stock_symbols[0]][['trend', stock_symbols[0]]]

rf = RandomForestClassifier(n_estimators=110, random_state=21)
rf.fit(X_train.values, y_train.values)

```

RandomForestClassifier  
RandomForestClassifier(n\_estimators=110, random\_state=21)

Hình 4. 76 Miền dữ liệu mới được đưa vào dự đoán mô hình Random Forest

#### 4.4.5.2 Sử dụng mô hình để dự đoán

- Với miền dữ liệu mới được tái cấu trúc theo phương pháp tái cấu trúc dữ liệu của các công ty, thực hiện kết quả dự đoán lần lượt với mô hình Logistic Regression và mô hình Random Forest và cho ra kết quả như sau:



Hình 4. 77 Kết quả dự đoán với mô hình Logistic Regression

VIC	VNM	VCB	GAS
[ ] accuracy = accuracy_score(y_test.values, y_pred) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.79	[ ] accuracy = accuracy_score(y_test.values, y_pred) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.68	[ ] accuracy = accuracy_score(y_test.values, y_pred) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.95	[ ] accuracy = accuracy_score(y_test.values, y_pred) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.95
HVN	FPT	MWG	HPG
[ ] accuracy = accuracy_score(y_test.values, y_pred) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.63	[ ] accuracy = accuracy_score(y_test.values, y_pred) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.84	[ ] accuracy = accuracy_score(y_test.values, y_pred) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.84	[ ] accuracy = accuracy_score(y_test.values, y_pred) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.95
MSN	SAB		
[ ] accuracy = accuracy_score(y_test.values, y_pred) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.68	[ ] accuracy = accuracy_score(y_test.values, y_pred) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.68		

Hình 4. 78 Kết quả dự đoán với mô hình Random Forest

- Với miền dữ liệu mới được tái cấu trúc theo phương pháp tái cấu trúc dữ liệu tổng hợp các công ty, thực hiện kết quả dự đoán lần lượt với mô hình Logistic Regression và mô hình Random Forest và cho ra kết quả như sau:

▶ predictions = lr.predict(X_test_scaled) predictions  [ ] array([1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1])  [ ] y_test.values  array([0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1])  [ ] accuracy = accuracy_score(y_test.values, predictions) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.79	▶ y_pred = rf.predict(X_test.values) y_pred  [ ] array([0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1])  [ ] y_test.values  array([0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1])  [ ] accuracy = accuracy_score(y_test.values, y_pred) print(f"Accuracy: {accuracy:.2f}")  Accuracy: 0.74
---	---

Hình 4. 79 Kết quả dự đoán của hai mô hình Logistic Regression và Random Forest tương ứng lần lượt từ trái sang phải

Như vậy, chúng tôi thấy rằng cả hai mô hình khi dự đoán hai miền dữ liệu khác nhau đều cho ra các kết quả dự đoán tốt và có độ chính xác cao. Đây là kết quả đạt mong đợi của nhóm chúng tôi trong bài nghiên cứu này.

## **CHƯƠNG 5 – KẾT LUẬN**

### **5.1 Thành quả nghiên cứu**

Thông qua bài nghiên cứu này, nhóm chúng tôi đã có xây dựng được những mô hình mục tiêu mà chúng tôi đề ra. Chúng tôi đã nghiên cứu, tính toán được các chỉ báo, chỉ số kinh tế, chứng khoán và ngoài ra tìm được thêm so với dự định ban đầu. Ngoài ra về phần trực quan hóa dữ liệu chúng tôi cũng đã hoàn thành tốt và đạt được mục tiêu so với dự định ban đầu. Về phương pháp tái cấu trúc chúng tôi cũng đã hiện thực được phương pháp một cách tốt nhất, bộ dữ liệu từ ban đầu đã được tái cấu trúc lại và huấn luyện với mô hình thì đạt được nhiều kết quả chính xác cao.

Với thành quả của nghiên cứu này, nhóm đã đạt được mục tiêu đặt ra ban đầu. Nghiên cứu của nhóm chúng tôi có thể giúp cho các nhà đầu tư chứng khoán sẽ đầu tư đúng đắn hơn, có cái nhìn trực quan hơn về thị trường chứng khoán.

### **5.2 Hạn chế và hướng phát triển tương lai**

Với bài nghiên cứu này, mặt hạn chế của nhóm chúng tôi vẫn còn tồn đọng vì nhóm chúng tôi chưa thể xử lý cũng như tối ưu tốt mô hình ARIMA. Chúng tôi vẫn còn thiếu kiến thức cũng như hiểu thêm về thống kê nên chưa thể làm tốt với ARIMA. Nhóm chúng tôi mong muốn có thể xây dựng được mô hình ARIMA một cách tốt nhất và mong muốn phát triển mô hình để dự đoán thêm nhiều lĩnh vực khác như dự đoán giá vàng, bitcoin...

## TÀI LIỆU THAM KHẢO

- [1] Lounnapha Sayavong, Zhongdong Wu, Sookasame Chalita, [2019], *Research on Stock Price Prediction Method Based on Convolutional Neural Network*,
- [2] Daiyou Xiao, Jinxia Su, [2022], *Research on Stock Price Time Series Prediction Based on Deep Learning and Autoregressive Integrated Moving Average*
- [3] ThS. Phạm Thị Mỹ Châu, [2020], *Hiệu quả của phân tích kỹ thuật trong đầu tư trên thị trường chứng khoán Việt Nam*
- [4] BigData, [2022], *Tìm hiểu về Time series (P.5): So lược về ARIMA*
- [5] Hai's Blog, [2017], *[RNN] LSTM là gì*
- [6] Machine Learning cơ bản, [2017], *Logistic Regression*
- [7] Pham Dinh Khanh, [2021], *Giới thiệu về mô hình rừng cây (Random forest)*