|  |  |
| --- | --- |
|  |  |
| **NÔNG CẢNH TRIỀN** | **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  **---------------------------------------** |
|  |
| **ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC** |
| **NGÀNH CÔNG NGHỆ THÔNG TIN** |
| **XÂY DỰNG HỆ THỐNG DỰ ĐOÁN GIÁ CỔ PHIẾU CHO CÔNG TY APPLE** |
|  |
|  |
| **CBHD:ThS. Nguyễn Lan Anh** |
| **CÔNG NGHỆ THÔNG TIN** | **Sinh viên: Nông Cảnh Triền** |
| **Mã số sinh viên: 2021603776** |
|  |
|  |
|  |
| **Hà Nội – Năm 2025** |
|  |

**LỜI CẢM ƠN**

Lời đầu tiên cho em xin gửi lời cảm ơn đến quý Thầy Cô đang công tác tại khoa Công nghệ thông tin trường Đại Học Công Nghiệp Hà Nội đã tận tình hướng dẫn và truyền đạt cho em những kiến thức, những kinh nghiệm nghề nghiệp để em có hành trang vững chắc bước vào đời. Không những vậy, Thầy Cô đã tạo mọi điều kiện thuận lợi nhất cho em thực hiện đồ án tốt nghiệp của mình trong suốt thời gian vừa qua.

Em xin gửi lời cảm ơn chân thành đến cô ThS. Nguyễn Lan Anh đã giúp đỡ, chỉ bảo và hỗ trợ hướng dẫn em trong suốt thời gian làm đồ án tốt nghiệp. Cô đã bỏ rất nhiều thời gian và công sức để giúp cho em có được một đồ án tốt nghiệp để em có thể hoàn thành khóa học của mình một cách tốt nhất.

Trong quá trình làm đồ án tốt nghiệp, cũng như là trong quá trình làm bài báo cáo đồ án tốt nghiệp do trình độ cũng như kinh nghiệm thực tiễn còn hạn chế nên đồ án không thể tránh khỏi những thiếu sót, em rất mong nhận được ý kiến đóng góp của Thầy Cô để em học thêm được nhiều kinh nghiệm và sẽ hoàn chỉnh hơn bài báo cáo đồ án tốt nghiệp của em.

Cuối cùng em kính chúc tất cả các Thầy Cô khoa Công nghệ thông tin của trường Đại Học Công Nghiệp Hà Nội dồi dào sức khỏe, đạt được nhiều thành công trong công việc.

Em xin chân thành cảm ơn!

Sinh viên thực hiện

*Nông Cảnh Triền*

**MỤC LỤC**

[MỤC LỤC i](#_Toc169142479)

[DANH MỤC CHỮ VIẾT TẮT iv](#_Toc169142480)

[DANH MỤC HÌNH ẢNH v](#_Toc169142481)

[DANH MỤC BẢNG BIỂU vii](#_Toc169142482)

[MỞ ĐẦU 1](#_Toc169142483)

[**1.** **Lý do chọn đề tài** 1](#_Toc169142484)

[**1.** **Đối tượng, mục tiêu và phạm vi nghiên cứu** 1](#_Toc169142485)

[**2.** **Kết quả dự kiến** 2](#_Toc169142486)

[**4. Bố cục đề tài** 3](#_Toc169142487)

[CHƯƠNG 1: TỔNG QUAN VỀ CHỨNG KHOÁN VÀ CÁC CÔNG NGHỆ SỬ DỤNG 4](#_Toc169142488)

[**1.1.** **Giới thiệu về chứng khoán** 4](#_Toc169142489)

[**1.1.1.** **Cổ phiếu là gì?** 4](#_Toc169142490)

[**1.1.2.** **Trái phiếu là gì?** 6](#_Toc169142491)

[**1.1.3.** **Chứng quyền là gì?** 7](#_Toc169142492)

[**1.1.4.** **Đặc điểm của chứng khoán** 8](#_Toc169142493)

[**1.2.** **Giới thiệu về thị trường chứng khoán** 9](#_Toc169142494)

[**1.3.** **Vai trò, chức năng của thị trường chứng khoán** 10](#_Toc169142495)

[**1.3.1.** **Huy động vốn cho nên kinh tế** 10](#_Toc169142496)

[**1.3.2.** **Cung cấp môi trường đầu tư cho công chúng** 11](#_Toc169142497)

[**1.3.3.** **Tạo tính thanh khoản cho các chứng khoán** 11](#_Toc169142498)

[**1.3.4.** **Đánh giá hoạt động của doanh nghiệp** 11](#_Toc169142499)

[**1.3.5.** **Tạo môi trường giúp chính phủ thực hiện các chính sách vĩ mô** 11](#_Toc169142500)

[**1.4.** **Phát biểu bài toán** 12](#_Toc169142501)

[**1.4.1.** **Input** 12](#_Toc169142502)

[**1.4.2.** **Output** 12](#_Toc169142503)

[**1.4.3.** **Mục tiêu** 12](#_Toc169142504)

[**1.4.4.** **Ràng buộc** 13](#_Toc169142505)

[**1.4.5.** **Cách tiếp cận** 13](#_Toc169142506)

[**1.4.6.** **Các bước thực hiện** 13](#_Toc169142507)

[**1.5. Một số phương pháp dự đoán** 14](#_Toc169142508)

[**1.5.1. Phương pháp thống kê (Statistical Methods)** 14](#_Toc169142509)

[**1.5.2. Các mô hình học máy (Machine Learning)** 14](#_Toc169142510)

[**1.5.3. Các mô hình học sâu (Deep Learning)** 15](#_Toc169142511)

[CHƯƠNG 2: MỘT SỐ THUẬT TOÁN HỌC MÁY, HỌC SÂU VÀ THƯ VIỆN 17](#_Toc169142512)

[**2.1. Mạng RNN** 17](#_Toc169142513)

[**2.1.1. Giới thiệu về RNN** 17](#_Toc169142514)

[**2.1.2. Dữ liệu dạng sequence** 17](#_Toc169142515)

[**2.1.3. Mô hình RNN** 18](#_Toc169142516)

[**2.1.4. Hạn chế của mạng nơ ron hồi quy** 20](#_Toc169142517)

[**2.1.5. Phân loại bài toán RNN** 20](#_Toc169142518)

[**2.1.6. Ứng dụng bài toán RNN** 21](#_Toc169142519)

[**2.1.7. Ưu điểm** 22](#_Toc169142520)

[**2.1.8. Nhược điểm** 22](#_Toc169142521)

[**2.2. Mạng LSTM** 23](#_Toc169142522)

[**2.2.1. Giới thiệu về LSTM** 23](#_Toc169142523)

[**2.2.3. Mạng trí nhớ ngắn hạn** 24](#_Toc169142524)

[**2.2.5. Thứ tự các bước của LSTM** 25](#_Toc169142525)

[**2.2.6. Các biển thể của LSTM** 29](#_Toc169142526)

[**2.2.7. Ưu điểm** 32](#_Toc169142527)

[**2.2.8. Nhược điểm** 32](#_Toc169142528)

[**2.3. Thuật toán KNN** 33](#_Toc169142529)

[**2.3.1. Giới thiệu về thuật toán KNN** 33](#_Toc169142530)

[**2.3.2. Cách thức hoạt động** 33](#_Toc169142531)

[**2.3.3. Ứng dụng của thuật toán** 35](#_Toc169142532)

[**2.3.4. Ưu điểm** 36](#_Toc169142533)

[**2.3.5. Nhược điểm** 36](#_Toc169142534)

[**2.4. Giới thiệu về Python** 37](#_Toc169142535)

[**2.4.1. Ưu điểm** 37](#_Toc169142536)

[**2.4.2. Nhược điểm** 38](#_Toc169142537)

[**2.5. Một số thư viện của Python** 39](#_Toc169142538)

[**2.5.1. Numpy** 39](#_Toc169142539)

[**2.5.2. Keras** 40](#_Toc169142540)

[**2.4.3. Pandas** 41](#_Toc169142541)

[**2.4.4. Scikit-learn** 42](#_Toc169142542)

[**2.4.5. PyQt** 43](#_Toc169142543)

[**2.4.6. Matplotlib** 44](#_Toc169142544)

[CHƯƠNG 3: ỨNG DỤNG LSTM TRONG DỰ ĐOÁN CHỨNG KHOÁN 46](#_Toc169142545)

[**3.1. Giới thiệu về dữ liệu** 46](#_Toc169142546)

[**3.2. Xử lý dữ liệu** 48](#_Toc169142547)

[**3.3. Huấn luyện mô hình** 50](#_Toc169142548)

[**3.4. Kết quả thực nghiệm** 54](#_Toc169142549)

[**3.5. So sánh độ chính xác của mô hình** 56](#_Toc169142550)

[**3.6. Xây dựng giao diện ứng dụng chứng khoán** 57](#_Toc169142551)

[**3.7. Kiểm thử mô hình** 61](#_Toc169142552)

[**3.7.1. Kế hoạch kiểm thử** 61](#_Toc169142553)

[**3.7.2. Kiểm thử mô hình** 62](#_Toc169142554)

[**3.7.3. Kiểm thử giao diện đăng nhập** 64](#_Toc169142555)

[**3.7.4. Kết quả kiểm thử** 65](#_Toc169142556)

[KẾT LUẬN 66](#_Toc169142557)

[TÀI LIỆU THAM KHẢO 67](#_Toc169142558)

# **DANH MỤC CHỮ VIẾT TẮT**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Ký hiệu** | **Tiếng anh** | **Nghĩa tiếng việt** |
| 1 | RNN | Recurrent Neural Network | Mạng nơ-ron hồi quy |
| 2 | LSTM | Long Short – Term Memory | Bộ nhớ dài - ngắn |
| 3 | KNN | K-Nearest Neighbors | K – hàng xóm gần nhất |
| 4 | CNN | Convolutional Neural Networks | Mạng nơ-ron nhân tạo |
| 5 | GUI | Graphical User Interface | Giao diện đồ họa người dùng |
| 6 | NYSE | New York Stock Exchange | Sở giao dịch chứng khoán New York |
| 7 | NASDAQ | National Association of Securities Dealers Automated Quotation System | Sàn giao dịch chứng khoán tại Mỹ NASDAQ |
| 8 | EDA | Exploratory Data Analysis | Phân tích Khám phá Dữ liệu |
| 9 | MNIST | Modified National Institute of Standards and Technology database | cơ sở dữ liệu lớn chứa các chữ số viết tay |
| 10 | SQL | Structured Query Language | Ngôn ngữ truy vấn có cấu trúc |

# **DANH MỤC HÌNH ẢNH**

[Hình 1. 1. Hình ảnh về chứng khoán 4](#_Toc169138756)

[Hình 1. 2. Hình ảnh thị trường chứng khoán 10](#_Toc169138757)

[Hình 2. 1. Mô hình RNN cho bài toán [4] 18](#_Toc169143569)

[Hình 2. 2. Mô hình RNN rút gọn [4] 19](#_Toc169143570)

[Hình 2. 3. Các dạng bài toán RNN [4] 21](#_Toc169143571)

[Hình 2. 4. Sự lặp lại kiến trúc module trong mạng LSTM [5] 24](#_Toc169143572)

[Hình 2. 5. Diễn giải các kí hiệu trong đồ thị mạng nơ ron [5] 25](#_Toc169143573)

[Hình 2. 6. Tầng cổng quên (forget gate layer) [5] 26](#_Toc169143574)

[Hình 2. 7. Cập nhật giá trị cho ô trạng thái [5] 27](#_Toc169143575)

[Hình 2. 8. Ô trạng thái mới [5] 28](#_Toc169143576)

[Hình 2. 9. Điều chỉnh thông tin ở đầu ra thông qua hàm tanh [5] 29](#_Toc169143577)

[Hình 2. 10. Kết nối ổng tiểu liên kết trực tiếp ô trạng thái với các cổng [5] 30](#_Toc169143578)

[Hình 2. 11. Cấu trúc điều chỉnh thêm mới và bỏ vào thông tin đồng thời [5] 31](#_Toc169143579)

[Hình 2. 12. Cấu trúc cổng truy hồi đơn vị (GRU – Gatesd Recurrent Unit) [5] 31](#_Toc169143580)

[Hình 2. 13. Thuật toán KNN [6] 35](#_Toc169143581)

[Hình 3. 1. Hình ảnh dữ liệu được lấy từ Yahoo Finance 46](#_Toc169138771)

[Hình 3. 2. Dữ liệu mã chứng khoán GOOG 48](#_Toc169138772)

[Hình 3. 3. Kết quả thông tin của dữ liệu được lấy 49](#_Toc169138773)

[Hình 3. 4. Hình ảnh kết quả dự đoán giá của mã chứng khoán 7 ngày tới 55](#_Toc169138774)

[Hình 3. 5. Biểu đồ mã chứng khoán trước khi dự đoán 55](#_Toc169138775)

[Hình 3. 6. Biểu đồ mã chứng khoán khi thêm phần dự đoán 56](#_Toc169138776)

[Hình 3. 7. Độ chính xác của mô hình LSTM 56](#_Toc169138777)

[Hình 3. 8. Độ chính xác của mô hình KNN 56](#_Toc169138778)

[Hình 3. 9. Giao diện đăng nhập 57](#_Toc169138779)

[Hình 3. 10. Giao diện đăng kí 58](#_Toc169138780)

[Hình 3. 11. Giao diện lấy lại mật khẩu 59](#_Toc169138781)

[Hình 3. 12. Giao diện ứng dụng dự đoán mã chứng khoán 60](#_Toc169138782)

[Hình 3. 13. Giao diện khi đã nhập mã chứng khoán 61](#_Toc169138783)

# **DANH MỤC BẢNG BIỂU**

[Bảng 3. 1. Bảng kiểm thử mô hình 62](#_Toc169047247)

[Bảng 3. 2. Bảng kiểm thử giao diện đăng nhập 64](#_Toc169047248)

# **MỞ ĐẦU**

1. **Lý do chọn đề tài**

Như mọi người đã biết, thị trường cổ phiếu ngày nay được xem là thời kì phát triển mạnh mẽ nhất với hàng trăm công ty được niêm yết trên sàn và giá thay đổi theo từng giây từng phút. Khi dự báo giá cổ phiếu, các nhà phân tích thường phải nghiên cứu các loại báo cáo tài chính, nguồn dữ liệu tổng hợp từ nhiều năm trước. Tuy nhiên việc đối mặt với các yếu tố khách quan như là dịch bệnh, thiên tai có thể ảnh hưởng rất lớn tới giá của một nhóm cổ phiếu củ thể.

Việc xây dựng hệ thống dự đoán giá cổ phiếu cho công ty Apple được lựa chọn làm đề tài nghiên cứu vì tầm quan trọng của việc phân tích và dự báo xu hướng giá cổ phiếu trong thị trường tài chính đầy biến động. Apple là một trong những tập đoàn công nghệ hàng đầu thế giới, có ảnh hưởng lớn đến nền kinh tế toàn cầu và thu hút sự quan tâm của nhiều nhà đầu tư. Giá cổ phiếu của Apple chịu tác động từ nhiều yếu tố như báo cáo tài chính, xu hướng công nghệ, chính sách kinh tế và tâm lý thị trường, đòi hỏi các phương pháp dự báo chính xác và hiệu quả. Bằng cách ứng dụng các mô hình học máy, đặc biệt là LSTM (Long Short-Term Memory), hệ thống có thể phân tích dữ liệu lịch sử, nhận diện xu hướng và dự đoán giá cổ phiếu với độ chính xác cao. Nghiên cứu này không chỉ giúp nhà đầu tư có công cụ hỗ trợ ra quyết định mà còn góp phần nâng cao hiểu biết về ứng dụng trí tuệ nhân tạo trong lĩnh vực tài chính, mở ra cơ hội áp dụng vào các hệ thống giao dịch thông minh.

1. **Đối tượng, mục tiêu và phạm vi nghiên cứu**

**Đối tượng:**

* Đề tài tập trung nghiên cứu các phương pháp dự đoán giá cổ phiếu, đặc biệt là mô hình Long Short-Term Memory (LSTM), một dạng mạng nơ-ron hồi quy sâu (RNN) chuyên xử lý dữ liệu chuỗi thời gian. Dữ liệu nghiên cứu bao gồm giá cổ phiếu lịch sử của công ty Apple (AAPL) cùng các yếu tố ảnh hưởng như khối lượng giao dịch, chỉ số thị trường và thông tin kinh tế.

**Mục tiêu:**

* Xây dựng một hệ thống có khả năng dự đoán giá cổ phiếu Apple dựa trên dữ liệu lịch sử bằng cách ứng dụng mô hình LSTM.
* Phân tích và đánh giá hiệu suất mô hình thông qua các chỉ số đo lường như MSE, RMSE, MAE.
* So sánh mô hình LSTM với các phương pháp truyền thống khác như đường trung bình động (Moving Average) hoặc hồi quy tuyến tính.

**Phạm vi nghiên cứu:**

* Dữ liệu nghiên cứu: Dữ liệu lịch sử giá cổ phiếu Apple (AAPL) thu thập từ các nguồn tài chính uy tín như Yahoo Finance hoặc Alpha Vantage, giới hạn trong khoảng thời gian từ 5 đến 10 năm gần đây.
* Phương pháp nghiên cứu: Ứng dụng kỹ thuật học sâu (Deep Learning), đặc biệt là mô hình LSTM để xử lý và dự báo dữ liệu chuỗi thời gian.
* Công cụ và nền tảng: Sử dụng Python và các thư viện hỗ trợ như TensorFlow/Keras, Pandas, NumPy, Matplotlib để thu thập, xử lý và huấn luyện mô hình.
* Giới hạn nghiên cứu: Hệ thống chỉ tập trung vào việc dự đoán giá cổ phiếu dựa trên dữ liệu lịch sử, chưa tích hợp các yếu tố phi cấu trúc như tin tức tài chính hay phân tích cảm xúc thị trường.

1. **Kết quả dự kiến**

* Xác định được yêu cầu bài toán và phạm vi của ứng dụng.
* Phân tích thiết kế đầy đủ những tính năng cơ bản của hệ thống.
* So sánh mô hình LSTM với các phương pháp truyền thống.
* Hoàn thành giao diện và cài đặt được các chức năng và yêu cầu đã đặt.
* Báo cáo tổng hợp của đề tài.
* Thuyết minh và đĩa CD đính kèm.
* Mã nguồn được quản lý với Git và lưu trữ trên GitHub.
* Kiểm thử sản phẩm và báo cáo kết quả kiểm thử.

## **4. Bố cục đề tài**

Trong báo cáo đồ án này, bố cục được chia thành 3 chương:

* **Chương 1: Tổng quan về thị trường cổ phiếu công ty Apple và công nghệ dự báo**: Tiến hành tìm hiểu về cổ phiếu công ty Apple , các công nghệ sử dụng dự báo.
* **Chương 2: Một số thuật toán học máy, học sâu và thư viện**: Tìm hiểu mô hình LSTM và một số thuật toán học máy, học sâu và một số thư viện Python.
* **Chương 3:** **Ứng dụng LSTM trong dự đoán chứng khoán**: Trình bày quá trình ứng dụng LSTM vào bài toán, kết quả đạt được, ứng dụng và quy trình kiểm thử ứng dụng.

# **CHƯƠNG 1: TỔNG QUAN VỀ THỊ TRƯỜNG CỔ PHIẾU CÔNG TY APPLE VÀ CÔNG NGHỆ DỰ BÁO**

* 1. **Giới thiệu về thị trường cổ phiếu công ty Apple.**

Chứng khoán là một loại tài sản gồm cổ phiếu, chứng chỉ quỹ, chứng khoán phái sinh, trái phiếu, chứng quyền có bảo đảm...

Theo định nghĩa trong Luật Chứng khoán, chứng khoán là tài sản bao gồm cổ phiếu, chứng chỉ quỹ, chứng khoán phái sinh, trái phiếu, chứng quyền, chứng quyền có bảo đảm, quyền mua cổ phần, chứng chỉ lưu ký. Những loại tài sản này có điểm chung là một bằng chứng xác nhận sở hữu hợp pháp của người sở hữu (gọi chung là nhà đầu tư) với tài sản của doanh nghiệp hoặc tổ chức phát hành.



Hình .1. Hình ảnh về cổ phiếu

[*(Nguồn: Hình ảnh về chứng khoán)*](https://www.google.com/url?sa=i&url=https%3A%2F%2Fnld.com.vn%2Fkinh-te%2Fchung-khoan-ngay-8-3-tranh-chap-quanh-nguong-1500-diem-20220307193915734.htm&psig=AOvVaw01YkMnpHdeWvbn8RVikACm&ust=1718306600925000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCJiKnobl1oYDFQAAAAAdAAAAABAE)

* + 1. ***Cổ phiếu là gì?***

Cổ phiếu được xem là chứng khoán vốn bởi nó xác nhận quyền và lợi ích như như dự họp đại hội đồng cổ đông, biểu quyết, đề cử Hội đồng quản trị, hưởng cổ tức... của người sở hữu với một phần vốn của doanh nghiệp. Đây là loại chứng khoán phổ biến và được giao dịch nhiều nhất trên thị trường.

Tương tự cổ phiếu, chứng chỉ quỹ cũng là chứng khoán vốn nhưng để khẳng định quyền sở hữu của nhà đầu tư với một phần vốn góp của quỹ đầu tư chứng khoán. Bản chất đầu tư vào chứng chỉ quỹ là ủy thác vốn cho chuyên gia của các tổ chức quản lý quỹ.

* Các đặc điểm chính:
* Người phát hành: Có thể là chính phủ, các địa phương hoặc các công ty.
* Giá trị, mệnh giá: Là giá trị gốc của trái phiếu, được trả lại cho nhà đầu tư khi trái phiếu đáo hạn.
* Lãi suất cố định hoặc biến động: Trái phiếu có thể có lãi suất cố định (được xác định và không đổi suốt thời gian tồn tại của trái phiếu) hoặc lãi suất biến động (có thể thay đổi theo một chỉ số tham chiếu nhất định).
* Kỳ hạn: Độ dài thời gian từ khi phát hành trái phiếu cho đến khi trả hết nợ gốc, có thể từ vài năm đến vài chục năm.
* Lợi Ích và Rủi Ro
* Lợi ích: Trái phiếu thường an toàn hơn so với cổ phiếu, đặc biệt là trái phiếu chính phủ, và cung cấp lãi suất thu nhập cố định.
* Rủi ro: Trái phiếu có rủi ro lãi suất (giá trị của trái phiếu giảm khi lãi suất tăng) và rủi ro tín dụng (rủi ro mất vốn nếu người phát hành không thể trả nợ).
* Các Loại Trái Phiếu
* Trái phiếu chính phủ: Phát hành bởi chính phủ quốc gia (như trái phiếu kho bạc Mỹ) hoặc chính quyền địa phương (như trái phiếu thành phố hoặc quận).
* Trái phiếu doanh nghiệp: Phát hành bởi các công ty nhằm mục đích tăng vốn cho các hoạt động kinh doanh.
* Trái phiếu chuyển đổi: Có thể chuyển đổi thành một số lượng cổ phiếu xác định của công ty phát hành trong tương lai.
  + 1. ***Cổ phiếu công ty Apple.***

Apple Inc. là một trong những tập đoàn công nghệ lớn nhất thế giới, được thành lập vào năm 1976 bởi Steve Jobs, Steve Wozniak và Ronald Wayne. Công ty có trụ sở chính tại Cupertino, California, và chuyên sản xuất các sản phẩm điện tử tiêu dùng, phần mềm và dịch vụ kỹ thuật số. Những sản phẩm nổi bật của Apple bao gồm iPhone, MacBook, iPad, Apple Watch và các dịch vụ như iCloud, Apple Music, App Store. Với sự đổi mới liên tục và hệ sinh thái sản phẩm mạnh mẽ, Apple đã trở thành thương hiệu có giá trị hàng đầu thế giới.

Cổ phiếu của Apple (AAPL) được niêm yết trên sàn giao dịch NASDAQ và là một trong những mã cổ phiếu có vốn hóa lớn nhất toàn cầu. Cổ phiếu AAPL thuộc các chỉ số quan trọng như S&P 500, NASDAQ-100 và Dow Jones Industrial Average. Nhờ vào sự tăng trưởng ổn định, Apple thu hút nhiều nhà đầu tư dài hạn và có chính sách mua lại cổ phiếu để hỗ trợ giá trị cổ đông. Công ty cũng chi trả cổ tức hàng quý, mặc dù lợi suất không quá cao do tập trung vào phát triển và mở rộng thị trường.

Việc đầu tư vào cổ phiếu Apple mang lại nhiều lợi ích như doanh thu ổn định, thương hiệu mạnh và hệ sinh thái bền vững, nhưng cũng đi kèm với rủi ro như cạnh tranh khốc liệt, sự phụ thuộc vào iPhone và các yếu tố kinh tế vĩ mô. Nhà đầu tư có thể mua cổ phiếu AAPL thông qua các nền tảng giao dịch chứng khoán như Robinhood, eToro, Interactive Brokers hoặc các công ty môi giới hỗ trợ giao dịch chứng khoán Mỹ. Trước khi đầu tư, việc nghiên cứu kỹ lưỡng về tình hình tài chính và chiến lược của Apple là điều cần thiết.

* 1. **Công nghệ dự báo trong thị trường cổ phiếu Apple.**

Công nghệ dự báo đóng vai trò quan trọng trong việc phân tích và dự đoán xu hướng giá cổ phiếu Apple. Các phương pháp hiện đại như trí tuệ nhân tạo (AI), học máy (Machine Learning), và phân tích dữ liệu lớn (Big Data) giúp các nhà đầu tư nhận diện mô hình biến động giá và đưa ra quyết định chính xác hơn.

Một số kỹ thuật phổ biến được sử dụng để dự báo cổ phiếu Apple bao gồm:

* **Phân tích kỹ thuật (Technical Analysis):** Sử dụng các chỉ báo như đường trung bình động (Moving Averages), RSI, MACD để đánh giá xu hướng giá.
* **Phân tích cơ bản (Fundamental Analysis):** Xem xét doanh thu, lợi nhuận, báo cáo tài chính và chiến lược kinh doanh của Apple.
* **Mô hình học máy (Machine Learning Models):** Ứng dụng thuật toán như mạng nơ-ron nhân tạo (Neural Networks) và cây quyết định (Decision Trees) để dự đoán giá cổ phiếu dựa trên dữ liệu lịch sử.
* **Xử lý ngôn ngữ tự nhiên (NLP):** Phân tích tin tức, báo cáo tài chính và tâm lý thị trường để đánh giá ảnh hưởng đến cổ phiếu.

Sự kết hợp giữa các phương pháp truyền thống và công nghệ hiện đại giúp nâng cao độ chính xác trong dự báo giá cổ phiếu Apple, từ đó hỗ trợ nhà đầu tư đưa ra quyết định thông minh và tối ưu hóa lợi nhuận. Tuy nhiên, dù có công nghệ tiên tiến, thị trường chứng khoán vẫn chịu ảnh hưởng từ các yếu tố bất ngờ, do đó việc quản lý rủi ro luôn là yếu tố quan trọng trong đầu tư.

* 1. **Phát biểu bài toán**
     1. ***Input***

Dữ liệu lịch sử của các mã chứng khoán bao gồm các thông tin sau:

* Giá mở cửa (Open)
* Giá cao nhất (High)
* Giá thấp nhất (Low)
* Giá đóng cửa (Close)
* Khối lượng giao dịch (Volume)
* Các chỉ số kỹ thuật khác (nếu có)
  + 1. ***Output***

Dự đoán giá đóng cửa của một mã chứng khoán trong một khoảng thời gian nhất định trong tương lai (ví dụ: ngày kế tiếp, tuần kế tiếp, tháng kế tiếp).

* + 1. ***Mục tiêu***
* Xây dựng mô hình dự đoán có thể sử dụng các thông tin đầu vào để dự đoán chính xác giá đóng cửa của mã chứng khoán.
* Mô hình cần phải được tối ưu hóa để giảm thiểu sai số dự đoán và nâng cao độ chính xác.
  + 1. ***Ràng buộc***
* Dữ liệu lịch sử của các mã chứng khoán phải đủ lớn và đủ chi tiết để huấn luyện mô hình.
* Mô hình dự đoán phải có khả năng xử lý các biến động mạnh và bất thường của thị trường chứng khoán.
* Thời gian tính toán và dự đoán của mô hình phải nhanh chóng để có thể áp dụng trong các tình huống thực tế.
  + 1. ***Cách tiếp cận***
* Thu thập và tiền xử lý dữ liệu lịch sử của các mã chứng khoán từ các nguồn đáng tin cậy.
* Áp dụng các kỹ thuật học máy và học sâu để xây dựng mô hình dự đoán.
* Huấn luyện và đánh giá mô hình trên tập dữ liệu kiểm tra.
* Triển khai mô hình dự đoán vào một ứng dụng demo để thử nghiệm và đánh giá hiệu suất.
  + 1. ***Các bước thực hiện***
* **Thu thập dữ liệu:** Lấy dữ liệu lịch sử của các mã chứng khoán từ các nguồn như Yahoo Finance, Alpha Vantage, Quandl.
* **Tiền xử lý dữ liệu:** Làm sạch dữ liệu, xử lý các giá trị thiếu, biến đổi dữ liệu, tạo các đặc trưng mới.
* **Lựa chọn mô hình:** Thử nghiệm với nhiều mô hình khác nhau như Hồi quy tuyến tính, Cây quyết định, Rừng ngẫu nhiên, Mạng nơ-ron hồi quy, Mạng nơ-ron tích chập, LSTM.
* **Huấn luyện và đánh giá:** Huấn luyện các mô hình trên tập dữ liệu huấn luyện và đánh giá hiệu suất trên tập dữ liệu kiểm tra.
* **Triển khai ứng dụng:** Xây dựng ứng dụng web để người dùng có thể nhập mã chứng khoán và xem dự đoán giá.

## **1.5. Một số phương pháp dự đoán**

### ***1.5.1. Phương pháp thống kê (Statistical Methods)***

Phương pháp thống kê trong dự đoán (statistical methods for prediction) bao gồm nhiều kỹ thuật được sử dụng để dự báo các giá trị tương lai dựa trên dữ liệu lịch sử. Dưới đây là một số phương pháp thông dụng:

* Hồi quy tuyến tính (Linear Regression):

Dự đoán giá trị của một biến phụ thuộc dựa trên giá trị của một hoặc nhiều biến độc lập. Mô hình hồi quy tuyến tính đơn giản sử dụng phương trình , trong đó là biến độc phụ thuộc, là biến độc lập, và là các hệ số hồi quy.

* Hồi quy bội (Multiple Regression):

Mở rộng hồi quy tuyến tính đơn để dự đoán giá trị của một biến phụ thuộc dựa trên nhiều biến độc lập. Mô hình này có dạng

* Phân tích chuỗi thời gian (Time Series Analysis):

Dự đoán các giá trị tương lai dựa trên các giá trị trong quá khứ của cùng một biến. Một số phương pháp phổ biến bao gồm:

+ ARIMA (AutoRegressive Integrated Moving Average): Kết hợp ba thành phần chính là tự hồi quy (AR), tích phân (I), và trung bình động (MA) để dự báo.

+ Exponential Smoothing: Sử dụng trọng số giảm dần theo thời gian để dự báo các giá trị tương lai.

### ***1.5.2. Các mô hình học máy (Machine Learning)***

Các mô hình học máy (machine learning) được sử dụng rộng rãi trong dự đoán vì chúng có khả năng học từ dữ liệu và cải thiện độ chính xác theo thời gian. Dưới đây là một số mô hình học máy phổ biến trong dự đoán:

* Hồi quy tuyến tính (Linear Regression):

Dự đoán một biến liên tục dựa trên giá trị của một hoặc nhiều biến độc lập. Đây là mô hình đơn giản và dễ hiểu.

* Hồi quy logistic (Logistic Regression):

Dự đoán xác suất của một biến phụ thuộc nhị phân (binary dependent variable). Thường được sử dụng trong các bài toán phân loại (classification).

* Cây quyết định (Decision Trees):

Sử dụng cấu trúc cây để phân chia dữ liệu dựa trên các điều kiện, giúp dễ dàng hiểu và diễn giải kết quả.

* Rừng ngẫu nhiên (Random Forest):

Kết hợp nhiều cây quyết định để cải thiện độ chính xác và giảm nguy cơ overfitting. Rừng ngẫu nhiên sử dụng phương pháp bagging để lấy mẫu và xây dựng nhiều cây quyết định.

* Máy vector hỗ trợ (Support Vector Machines - SVM):

Dùng để phân loại và hồi quy, SVM tìm kiếm một siêu phẳng tốt nhất để phân chia dữ liệu trong không gian cao chiều.

* K láng giềng gần nhất (K-Nearest Neighbors - KNN):

Dự đoán giá trị của một điểm dựa trên các giá trị của các điểm láng giềng gần nhất. KNN có thể được sử dụng cho cả phân loại và hồi quy.

* Mạng nơ-ron nhân tạo (Artificial Neural Networks - ANN):

Mô phỏng hoạt động của não người, mạng nơ-ron gồm nhiều tầng (layers) và có thể học các mô hình phức tạp từ dữ liệu lớn.

### ***1.5.3. Các mô hình học sâu (Deep Learning)***

Học sâu (deep learning) là một nhánh của học máy tập trung vào việc sử dụng các mạng nơ-ron nhân tạo với nhiều tầng (layers) để học từ dữ liệu phức tạp. Dưới đây là một số mô hình học sâu phổ biến được sử dụng trong dự đoán:

* Mạng Nơ-ron Tích chập (Convolutional Neural Networks - CNN):

CNN được thiết kế đặc biệt để xử lý dữ liệu dạng hình ảnh. Chúng sử dụng các lớp tích chập để tự động học các đặc trưng từ hình ảnh, và sau đó sử dụng các lớp dày đặc để dự đoán.

* Mạng Nơ-ron Hồi quy (Recurrent Neural Networks - RNN):

RNN thích hợp cho dữ liệu tuần tự hoặc chuỗi thời gian, chẳng hạn như văn bản, âm thanh hoặc dữ liệu tài chính. RNN sử dụng các nơ-ron có trạng thái ẩn để nhớ thông tin từ các bước trước đó.

* Long Short-Term Memory (LSTM):

LSTM là một biến thể của RNN được thiết kế để xử lý vấn đề vanishing gradient và có khả năng nhớ thông tin trong thời gian dài. LSTM rất hiệu quả cho các bài toán liên quan đến chuỗi thời gian dài như dịch máy, dự đoán tài chính và nhận dạng giọng nói.

# **CHƯƠNG 2: MỘT SỐ THUẬT TOÁN HỌC MÁY, HỌC SÂU VÀ THƯ VIỆN**

## **2.1. Mạng RNN**

### ***2.1.1. Giới thiệu về RNN***

Recurrent Neural Networks (RNN) là một loại mạng nơ-ron nhân tạo đặc biệt, có khả năng xử lý dữ liệu tuần tự và duy trì thông tin từ các bước trước đó thông qua việc sử dụng các vòng lặp trong cấu trúc mạng. Khác với các mạng nơ-ron truyền thống (feedforward neural networks), RNN có các kết nối ngược cho phép thông tin lưu giữ từ trạng thái trước đó để ảnh hưởng đến trạng thái hiện tại. Điều này làm cho RNN trở nên lý tưởng cho các nhiệm vụ liên quan đến dữ liệu tuần tự như xử lý ngôn ngữ tự nhiên, dịch máy, nhận dạng giọng nói và phân tích chuỗi thời gian.

### ***2.1.2. Dữ liệu dạng sequence***

Dữ liệu có thứ tự như các ảnh tách từ video ở trên được gọi là sequence, time-series data.

Trong bài toán dự đoán đột quỵ tim cho bệnh nhân bằng các dữ liệu tim mạch khám trước đó. Input là dữ liệu của những lần khám trước đó, ví dụ i1 là lần khám tháng 1, i2 là lần khám tháng 2,… i8 là lần khám tháng 8. (i1,i2,..i8) được gọi là sequence data. RNN sẽ học từ input và dự đoán xem bệnh nhân có bị đột quy tim hay không.

Ví dụ khác là trong bài toán dịch tự động với input là 1 câu, ví dụ “tôi yêu Việt Nam” thì vị trí các từ và sự xắp xếp cực kì quan trọng đến nghĩa của câu và dữ liệu input các từ [‘tôi’, ‘yêu’, ‘việt’, ‘nam’] được gọi là sequence data. Trong bài toán xử lý ngôn ngữ (NLP) thì không thể xử lý cả câu được và người ta tách ra từng từ làm input, giống như trong video người ta tách ra các ảnh (frame) làm input.

### ***2.1.3. Mô hình RNN***

Bài toán: Nhận diện hành động trong video 30s. Đây là dạng bài toán many to one trong RNN, tức nhiều input và 1 output.

Input ta sẽ tách video thành 30 ảnh ở mỗi giây. Các ảnh sẽ được cho qua model CNN để lấy ra các feature ([feature extraction](https://nttuan8.com/bai-9-transfer-learning-va-data-augmentation/#Feature_extractor)) thành các vector có kích thước n\*1. Vector tương ứng với ảnh ở giây thứ i là xi.

Output là vector có kích thước d\*1, softmax function được sử dụng như trong [bài phân loại ảnh](https://nttuan8.com/bai-7-gioi-thieu-keras-va-bai-toan-phan-loai-anh/" \l "Xay_dung_model" \t "_blank).

A diagram of a diagram

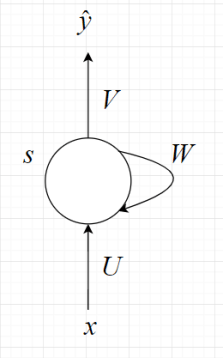
Description automatically generated

Hình 2. . Mô hình RNN cho bài toán [4]

Ta có:

* Mô hình có 30 input và 1 output, các input được cho vào model đúng với thứ tự ảnh trong video x1, x2, …x30.
* Mỗi hình tròn được gọi là 1 state, statet có input là x1 và st-1 (output của state trước); output là st = f(U\*xt + W\*st-1). f là activation funtion thường là tanh.
* Có thể thấy st mang cả thông tin từ state trước (st-1) và input của state hiện tại => giống như memory nhớ các đặc điểm của các input từ x1 đến xt.
* s0 được thêm vào chỉ cho chuẩn công thức nên thường được gán bằng 0 hoặc giá trị ngẫu nhiên. Có thể hiểu là ban đầu chưa có dữ liệu gì để học thì memory rỗng.
* Do ta chỉ có 1 output, nên sẽ được đặt ở state cuối cùng, khi đó s30 học được thông tin từ tất cả các input. . g là activation function, trong bài này là bài toán phân lại nên sẽ dùng softmax.

Ta thấy là ở mỗi state các hệ số W, U là giống nhau nên model có thể được viết lại thành:



Hình 2. . Mô hình RNN rút gọn [4]

Tóm lại:

* xi là vector có kích thước n\*1, si là vector có kích thước m\*1, yi là vector có kích thước d\*1. U là ma trận có kích thước m\*n, W là ma trận có kích thước m\*n và V là ma trận có kích thước d\*m.
* s0 =0, st = f(U\*xt + W\*st-1) với t >= 1

### ***2.1.4. Hạn chế của mạng nơ ron hồi quy***

**Vấn đề biến mất và nổ gradient (Vanishing and Exploding Gradient Problem)**:

* Trong quá trình huấn luyện, gradient của hàm mất mát (loss function) có thể trở nên rất nhỏ hoặc rất lớn. Điều này dẫn đến việc mạng không thể học được các quan hệ phụ thuộc dài hạn.
* Khi gradient biến mất, các thông tin từ quá khứ sẽ bị lãng quên nhanh chóng và mạng không thể học được các quan hệ dài hạn trong chuỗi dữ liệu.

**Khả năng nhớ hạn chế (Limited Memory)**:

* RNN có xu hướng quên thông tin khi khoảng cách giữa các thông tin đó trong chuỗi trở nên quá dài.
* Khả năng lưu trữ thông tin dài hạn của RNN là rất hạn chế, điều này làm giảm hiệu quả trong các bài toán cần nhớ các thông tin từ xa trong chuỗi dữ liệu.

**Hiệu quả kém với dữ liệu dài (Inefficiency with Long Sequences)**:

* RNN hoạt động không hiệu quả với các chuỗi dữ liệu dài do vấn đề gradient và khả năng nhớ hạn chế.
* Điều này làm giảm khả năng áp dụng của RNN trong nhiều ứng dụng thực tế yêu cầu xử lý dữ liệu dài.

***2.1.5. Phân loại bài toán RNN***

A diagram of a number of rectangular objects

Description automatically generated with medium confidence

Hình 2. . Các dạng bài toán RNN [4]

* **One to one**: mẫu bài toán cho Neural Network (NN) và Convolutional Neural Network (CNN), 1 input và 1 output, ví dụ với CNN input là ảnh và output là ảnh được segment.
* **One to many**: bài toán có 1 input nhưng nhiều output, ví dụ: bài toán caption cho ảnh, input là 1 ảnh nhưng output là nhiều chữ mô tả cho ảnh đấy, dưới dạng một câu.
* **Many to one**: bài toán có nhiều input nhưng chỉ có 1 output, ví dụ bài toán phân loại hành động trong video, input là nhiều ảnh (frame) tách ra từ video, ouptut là hành động trong video
* **Many to many**: bài toán có nhiều input và nhiều output, ví dụ bài toán dịch từ tiếng anh sang tiếng việt, input là 1 câu gồm nhiều chữ: “I love Vietnam” và output cũng là 1 câu gồm nhiều chữ “Tôi yêu Việt Nam”.

### ***2.1.6. Ứng dụng bài toán RNN***

Về cơ bản nếu bạn thấy sequence data hay time-series data và bạn muốn áp dụng deep learning thì bạn nghĩ ngay đến RNN. Dưới đây là một số ứng dụng của RNN:

* Speech to text: Chuyển giọng nói sang text.
* Sentiment classification: Phân loại số sao cho các bình luận, ví dụ: input: “ứng dụng tốt”, output: 4 sao.
* Machine translation: Bài toán dịch tự động giữa các ngôn ngữ.
* Video recognition: Nhận diện hành động trong video.
* Heart attack: Dự đoán đột quỵ tim.
* ….

### ***2.1.7. Ưu điểm***

* Xử lý dữ liệu tuần tự: RNN có khả năng xử lý dữ liệu tuần tự, như chuỗi thời gian hoặc văn bản, nhờ vào cấu trúc có các kết nối ngược, cho phép thông tin từ các bước trước ảnh hưởng đến các bước sau.
* Bộ nhớ ngắn hạn: RNN có khả năng duy trì bộ nhớ ngắn hạn về thông tin trước đó, giúp ích trong các ứng dụng cần nhận dạng các mẫu liên quan trong chuỗi dữ liệu.
* Ứng dụng rộng rãi: RNN được ứng dụng rộng rãi trong nhiều lĩnh vực như xử lý ngôn ngữ tự nhiên, nhận dạng giọng nói, phân tích chuỗi thời gian, và nhiều ứng dụng khác.

## **2.2. Mạng LSTM**

### ***2.1.8. Nhược điểm***

* Vanishing/Exploding Gradient Problem: Khi độ dài của chuỗi dữ liệu tăng lên, RNN có thể gặp phải vấn đề gradient biến mất hoặc gradient bùng nổ trong quá trình học, làm giảm hiệu quả của việc học và khiến mô hình khó cập nhật trọng số một cách chính xác.
* Bộ nhớ hạn chế: RNN thông thường có khả năng lưu trữ và truy xuất thông tin từ các bước thời gian trước đó chỉ trong khoảng ngắn, hạn chế trong việc xử lý các chuỗi dài hoặc thông tin phụ thuộc xa.
* Tính toán phức tạp: Việc tính toán gradient cho từng bước thời gian thông qua quá trình lan truyền ngược qua thời gian (BPTT) có thể rất phức tạp và tốn nhiều tài nguyên tính toán.
* Hiệu suất không cao với dữ liệu dài hạn: Đối với các tác vụ yêu cầu duy trì thông tin dài hạn, RNN thông thường không hoạt động hiệu quả, và cần sử dụng các biến thể như LSTM hoặc GRU để cải thiện.

### ***2.2.1. Giới thiệu về LSTM***

Long Short – Term Memory (LSTM) là một cải tiến của mạng RNN cho phép học các mối tương quan dài mà mạng nơ-ron hồi quy bị hạn chế. Mối tương quan dài được hiều là sự ảnh hưởng của các giá trị bỏ vào ở những thời điểm bắt đầu và các giá trị bỏ ra ở những thời điểm sau.

RNN (Recurrent Neural Network) có thể xử lý thông tin dạng chuỗi (Sequence/ Time-series). RNN có thể mang thông tin của frame (ảnh) từ state trước tới các state sau, rồi ở state cuối là sự kết hợp của tất cả các ảnh để dự đoán hành động.

Ví dụ bài toán là dự đoán từ tiếp theo trong đoạn văn. Đoạn đầu tiên “Mặt trời mọc ở hướng …”, ta có thể chỉ sử dụng các từ trước trong câu để đoán là đông. Tuy nhiên, với đoạn, “Tôi là người Việt Nam. Tôi đang sống ở nước ngoài. Tôi có thể nói trôi chảy tiếng …” thì rõ ràng là chỉ sử dụng từ trong câu đấy hoặc câu trước là không thể dự đoán được từ cần điền là Việt. Ta cần các thông tin từ state ở trước đó rất xa => cần long term memory điều mà RNN không làm được => Cần một mô hình mới để giải quyết vấn đề này => Long short term memory (LSTM) ra đời.

### ***2.2.3. Mạng trí nhớ ngắn hạn***

Mạng trí nhớ ngắn hạn định hướng dài hạn còn được viết tắt là LSTM làm một kiến trúc đặc biệt của RNN (Recurrent Neural Networ) có khả năng học được sự phụ thuộc trong dài hạn (long-term dependencies). Kiến trúc này đã được phổ biến và sử dụng rộng rãi cho tới ngày nay. LSTM đã tỏ ra khắc phục được rất nhiều những hạn chế của RNN trước đây về triệt tiêu đạo hàm. Tuy nhiên cấu trúc của chúng có phần phức tạp hơn mặc dù vẫn giữ được tư tưởng chính của RNN là sự sao chép các kiến trúc theo dạng chuỗi.

LSTM cũng có một chuỗi dạng như thế nhưng phần kiến trúc lặp lại có cấu trúc khác biệt hơn. Thay vì chỉ có một tầng đơn, chúng có tới 4 tầng ẩn (3 tầng sigmoid và 1 tanh) tương tác với nhau theo cấu trúc đặc biệt.

A screenshot of a computer

Description automatically generated

Hình 2. . Sự lặp lại kiến trúc module trong mạng LSTM [5]

Các kí hiệu có thể diễn giải như sau:

A black arrow pointing to the right

Description automatically generated

Hình 2. . Diễn giải các kí hiệu trong đồ thị mạng nơ ron [5]

Trong sơ đồ tính toán trên, mỗi một phép tính sẽ triển khai trên một véc tơ. Trong đó hình tròn màu hồng biển diễn một toán tử đối với véc tơ như phép cộng véc tơ, phép nhân vô hướng các véc tơ. Màu vàng thể hiện hàm activation mà mạng nơ ron sử dụng để học trong tầng ẩn, thông thường là các hàm phi tuyến sigmoid và tanh. Kí hiệu 2 đường thẳng nhập vào thể hiện phép chập kết quả trong khi kí hiệu 2 đường thẳng rẽ nhánh thể hiện cho nội dung véc tơ trước đó được sao chép để đi tới một phần khác của mạng nơ ron.

### ***2.2.5. Thứ tự các bước của LSTM***

Bước đầu tiên trong LSTM sẽ quyết định xem thông tin nào chúng ta sẽ cho phép đi qua ô trạng thái (cell state). Nó được kiểm soát bởi hàm sigmoid trong một tầng gọi là tầng quên (forget gate layer). Đầu tiên nó nhận đầu vào là 2 giá trị và và trả về một giá trị nằm trong khoảng 0 và 1 cho mỗi giá trị của ô trạng thái . Nếu giá trị bằng 1 thể hiện ‘giữ toàn bộ thông tin’ và bằng 0 thể hiện ‘bỏ qua toàn bộ chúng’.

Trở lại ví dụ về ngôn ngữ, chúng ta đang cố gắng dự báo từ tiếp theo dựa trên toàn bộ những từ trước đó. Trong những bài toán như vậy, ô trạng thái có thể bao gồm loại của chủ ngữ hiện tại, để cho đại từ ở câu tiếp theo được sử dụng chính xác. Chẳng hạn như chúng đang mô tả về một người bạn là con trai thì các đại từ nhân xưng ở tiếp theo phải là anh, thằng, hắn thay vì cô, con ấy. Tuy nhiên chủ ngữ không phải khi nào cũng cố định. Khi chúng ta nhìn thấy một chủ ngữ mới, chúng ta muốn quên đi loại của một chủ ngữ cũ. Do đó tầng quên cho phép cập nhật thông tin mới và lưu trữ giá trị của nó khi có thay đổi theo thời gian.

A diagram of a rectangular object with arrows pointing to the side

Description automatically generated

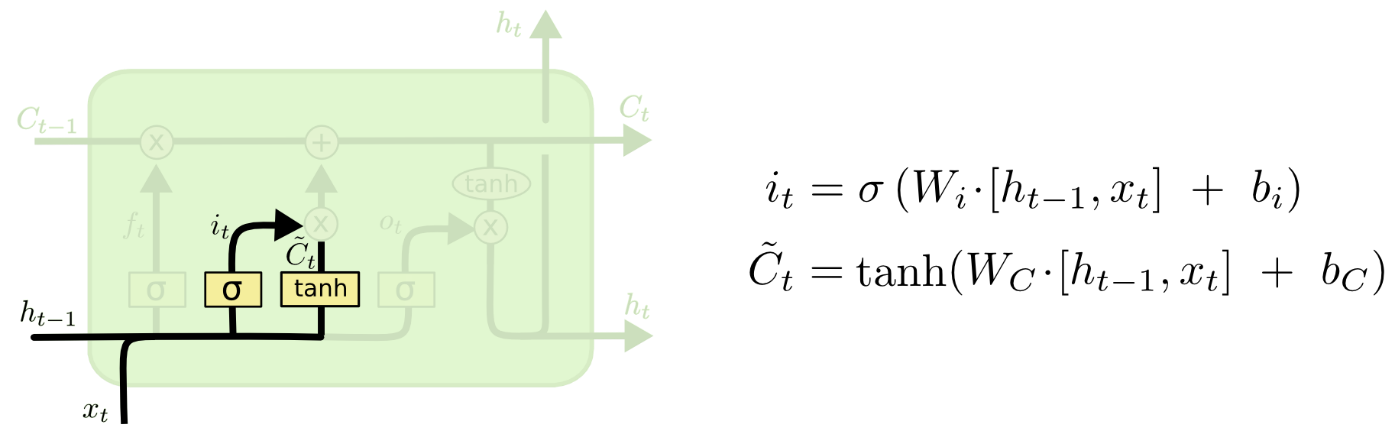
Hình 2. . Tầng cổng quên (forget gate layer) [5]

Trong đó:

* *, giá trị từ 0 đến 1, quyết định mức độ quên thông tin cũ.*
* *.*
* *.*
* *.*
* *.*
* *.*

Bước tiếp theo chúng sẽ quyết định loại thông tin nào sẽ được lưu trữ trong ô trạng thái. Bước này bao gồm 2 phần. Phần đầu tiên là một tầng ẩn của hàm sigmoid được gọi là tầng cổng vào (input gate layer) quyết định giá trị bao nhiêu sẽ được cập nhật. Tiếp theo, tầng ẩn hàm tanh sẽ tạo ra véc tơ của một giá trị trạng thái mới mà có thể được thêm vào trạng thái. Tiếp theo kết hợp kết quả của 2 tầng này để tạo thành một cập nhật cho trạng thái.

Trong ví dụ của mô hình ngôn ngữ, chúng ta muốn thêm loại của một chủ ngữ mới vào ô trạng thái để thay thế phần trạng thái cũ muốn quên đi.



Hình 2. . Cập nhật giá trị cho ô trạng thái [5]

Trong đó:

* *, quyết định thông tin mới sẽ được thêm vào bao nhiêu*

* *, một giá trị tiềm năng để thêm vào trạng thái tế bào*

Đây là thời điểm để cập nhật một ô trạng thái cũ, sang một trạng thái mới . Những bước trước đó đã quyết định làm cái gì, và tại bước này chỉ cần thực hiện nó.

Chúng ta nhận trạng thái cũ với tương ứng với việc quên những thứ quyết định được phép quên sớm. Phần tử đề cử là một giá trị mới được tính toán tương ứng với bao nhiêu được cập nhật vào mỗi giá trị trạng thái.

A diagram of a rectangular object with arrows pointing up

Description automatically generated

Hình 2. . Ô trạng thái mới [5]

Trong đó:

Cuối cùng cần quyết định xem đầu ra sẽ trả về bao nhiêu. Kết quả ở đầu ra sẽ dựa trên ô trạng thái, nhưng sẽ là một phiên bản được lọc. Đầu tiên, chúng ta chạy qua một tầng sigmoid nơi quyết định phần nào của ô trạng thái sẽ ở đầu ra. Sau đó, ô trạng thái được đưa qua hàm tanh (để chuyển giá trị về khoảng -1 và 1) và nhân nó với đầu ra của một cổng sigmoid, do đó chỉ trả ra phần mà chúng ta quyết định.

A diagram of a graphing diagram

Description automatically generated with medium confidence

Hình 2. . Điều chỉnh thông tin ở đầu ra thông qua hàm tanh [5]

Trong đó:

### ***2.2.6. Các biển thể của LSTM***

Những gì mà chúng ta vừa mổ tả cho đến giờ là một mạng LSTM rất thông thường. Nhưng không phải toàn bộ LSTM đều tương tự như trên. Trên thực tế, có vẻ như hầu hết mọi bài báo liên quan đến LSTM đều sử dụng những version khác nhau đôi chút. Sự khác biệt là rất nhỏ nhưng rất đáng để đề cập một ít trong số nhứng kiến trúc này.

Một trong những biến thể nối tiếng nhất của LSTM được giới thiệu bởi Gers & Schmidhuber (2000) thêm một kết nối ống tiểu (peehole connection) để các cổng có thể kết nối trực tiếp đến các ô trạng thái.

A rectangular object with arrows pointing to the center

Description automatically generated

Hình 2. . Kết nối ổng tiểu liên kết trực tiếp ô trạng thái với các cổng [5]

Trong đó:

* **:** Quyết định thông tin nào từ trạng thái tế bào cũ () cần được quên
* **:** Quyết định thông tin nào sẽ được thêm vào trạng thái tế bào hiện tại
* **:** Quyết định phần nào của trạng thái tế bào sẽ được sử dụng để tính toán đầu ra ẩn hiện tại

Một biến thể khác là sử dụng cặp đôi cổng vào và cổng ra. Thay vì quyết định riêng rẽ bỏ qua thông tin nào và thêm mới thông tin nào, chúng ta sẽ quyết định chúng đồng thời. Các thông tin chỉ bị quên khi chúng ta muốn cập nhập vào một vài thông tin mới.

A diagram of a rectangular object with arrows

Description automatically generated

Hình 2. . Cấu trúc điều chỉnh thêm mới và bỏ vào thông tin đồng thời [5]

Trong đó:  **:** Trạng thái tế bào mới được tạo ra từ trạng thái cũ và thông tin mới.

Một dạng biến thể khá mạnh khác của LSTM là cổng truy hồi đơn vị (Gated Recurrent Unit - GRU) được giới thiệu bởi Cho, et al. (2014). Nó kết hợp cổng quên và cổng vào thành một cổng đơn gọi là cập nhật (update gate). Nó cũng nhập các ô trạng thái và trạng thái ẩn và thực hiện một số thay đổi khác. Kết quả của mô hình đơn giản hơn nhiều so với mô hình LSTM chuẩn, và đã trở nên khá phổ biến.

A close-up of a sign

Description automatically generated

Hình 2. . Cấu trúc cổng truy hồi đơn vị (GRU – Gatesd Recurrent Unit) [5]

Trong đó:

* **:** Quyết định mức độ thông tin từ trạng thái ẩn trước đó sẽ được chuyển tiếp sang trạng thái ẩn hiện tại.
* **:** Quyết định mức đọ quên thông tin từ trạng thái ẩn trước đó.
* Tạo trạng thái ẩn mới dựa trên đầu vào hiện tại và trạng thái trước đó đã được điều chỉnh bởi cổng reset.
* **:** Kết hợp giữa trạng thái ẩn cũ và trạng thái ẩn ứng viên để tạo ra trạng thái hiện tại.

### ***2.2.7. Ưu điểm***

* Giữ thông tin lâu dài: LSTM có khả năng lưu trữ thông tin trong thời gian dài mà không bị biến mất, nhờ vào cơ chế của các cổng (gate) như cổng quên (forget gate), cổng vào (input gate) và cổng ra (output gate).
* Khả năng xử lý chuỗi dài: So với RNN thông thường, LSTM có khả năng xử lý chuỗi dài mà không gặp phải vấn đề gradient biến mất (vanishing gradient problem).
* Hiệu suất trong việc học dài hạn: Nhờ vào khả năng giữ thông tin qua nhiều bước thời gian, LSTM thích hợp cho các nhiệm vụ cần xử lý dữ liệu dài hạn và nhận dạng mẫu phức tạp.
* Đa dạng hóa cấu trúc: LSTM cho phép cấu hình nhiều lớp, các cổng và kích thước của mỗi lớp, từ đó có thể tạo ra các kiến trúc mô hình phức tạp và phù hợp với nhiều loại dữ liệu.

### ***2.2.8. Nhược điểm***

* Tính tính toán phức tạp: LSTM có thể đòi hỏi nhiều tài nguyên tính toán hơn so với các mô hình nơ-ron truyền thống, đặc biệt là khi mạng trở nên rất sâu và phức tạp.
* Khó điều chỉnh tham số: Với các mô hình LSTM có kích thước lớn và phức tạp, việc điều chỉnh các tham số trở nên khó khăn và tốn kém.
* Có thể dễ dàng bị overfitting: Như các mô hình nơ-ron khác, LSTM cũng dễ bị overfitting nếu không được điều chỉnh và kiểm soát đúng cách, đặc biệt là khi dữ liệu huấn luyện có kích thước nhỏ.
* Khó hiểu: Với những người mới bắt đầu, LSTM có thể khó hiểu do có nhiều cơ chế và cổng cần phải hiểu rõ để áp dụng hiệu quả.

## **2.3. Thuật toán KNN**

### ***2.3.1. Giới thiệu về thuật toán KNN***

K-Nearest Neighbors (KNN) là một thuật toán học máy linh hoạt và được sử dụng rộng rãi, chủ yếu được sử dụng vì tính đơn giản và dễ thực hiện của nó. Nó không yêu cầu bất kỳ giả định nào về phân phối dữ liệu cơ bản. Nó cũng có thể xử lý cả dữ liệu số và dữ liệu phân loại, khiến nó trở thành lựa chọn linh hoạt cho nhiều loại tập dữ liệu khác nhau trong các nhiệm vụ phân loại và hồi quy. Đây là một phương pháp phi tham số đưa ra dự đoán dựa trên sự giống nhau của các điểm dữ liệu trong một tập dữ liệu nhất định. KNN ít nhạy cảm hơn với các ngoại lệ so với các thuật toán khác.

Thuật toán KNN hoạt động bằng cách tìm K lân cận gần nhất với một điểm dữ liệu nhất định dựa trên thước đo khoảng cách, chẳng hạn như khoảng cách Euclide. Lớp hoặc giá trị của điểm dữ liệu sau đó được xác định bằng đa số phiếu bầu hoặc mức trung bình của K hàng xóm. Cách tiếp cận này cho phép thuật toán thích ứng với các mẫu khác nhau và đưa ra dự đoán dựa trên cấu trúc cục bộ của dữ liệu.

### ***2.3.2. Cách thức hoạt động***

Thuật toán K-Nearest Neighbors (KNN) hoạt động theo nguyên tắc tương tự, trong đó nó dự đoán nhãn hoặc giá trị của điểm dữ liệu mới bằng cách xem xét nhãn hoặc giá trị của K lân cận gần nhất của nó trong tập dữ liệu huấn luyện.

Để đưa ra dự đoán, thuật toán tính toán khoảng cách giữa mỗi điểm dữ liệu mới trong tập dữ liệu thử nghiệm và tất cả các điểm dữ liệu trong tập dữ liệu huấn luyện. Khoảng cách Euclide là thước đo khoảng cách thường được sử dụng trong KNN, nhưng các thước đo khoảng cách khác, chẳng hạn như khoảng cách Manhattan hoặc khoảng cách Minkowski, cũng có thể được sử dụng tùy thuộc vào vấn đề và dữ liệu. Khi khoảng cách giữa điểm dữ liệu mới và tất cả các điểm dữ liệu trong tập dữ liệu huấn luyện được tính toán, thuật toán sẽ tiến hành tìm K hàng xóm gần nhất dựa trên những khoảng cách này. Phương pháp cụ thể để chọn các lân cận gần nhất có thể khác nhau, nhưng cách tiếp cận phổ biến là sắp xếp khoảng cách theo thứ tự tăng dần và chọn K điểm dữ liệu có khoảng cách ngắn nhất.

Sau khi xác định K hàng xóm gần nhất, thuật toán đưa ra dự đoán dựa trên nhãn hoặc giá trị liên quan đến những hàng xóm này. Đối với các nhiệm vụ phân loại, lớp đa số trong số K hàng xóm được gán làm nhãn dự đoán cho điểm dữ liệu mới. Đối với các tác vụ hồi quy, giá trị trung bình hoặc trung bình có trọng số của các giá trị K lân cận được gán làm giá trị dự đoán.

Giả sử X là tập dữ liệu huấn luyện có n điểm dữ liệu, trong đó mỗi điểm dữ liệu được biểu thị bằng vectơ đặc trưng d chiều X\_i và Y là nhãn hoặc giá trị tương ứng cho từng điểm dữ liệu trong X. Cho một điểm dữ liệu mới x, thuật toán sẽ tính toán khoảng cách giữa x và mỗi điểm dữ liệu X\_i trong X bằng cách sử dụng thước đo khoảng cách, chẳng hạn như khoảng cách Euclide:

Thuật toán chọn K điểm dữ liệu từ X có khoảng cách ngắn nhất đến x. Đối với các nhiệm vụ phân loại, thuật toán gán nhãn y thường xuyên nhất trong số K lân cận gần nhất với x. Đối với các tác vụ hồi quy, thuật toán tính trung bình hoặc trung bình có trọng số của các giá trị y của K lân cận gần nhất và gán nó làm giá trị dự đoán cho x.

A black and white image of a circle and a circle with small black triangles

Description automatically generated with medium confidence

Hình 2. . Thuật toán KNN [6]

### ***2.3.3. Ứng dụng của thuật toán***

Xử lý trước dữ liệu - Trong khi xử lý bất kỳ vấn đề nào về Machine Learning, trước tiên chúng tôi thực hiện phần EDA , trong đó nếu chúng tôi thấy rằng dữ liệu chứa các giá trị bị thiếu thì cũng có nhiều phương pháp tính toán khả dụng. Một trong những phương pháp như vậy là KNN Imputer , một phương pháp quảng cáo khá hiệu quả thường được sử dụng cho các phương pháp tính toán phức tạp.

Nhận dạng mẫu: Thuật toán KNN hoạt động rất tốt nếu bạn đã huấn luyện thuật toán KNN bằng cách sử dụng bộ dữ liệu MNIST và sau đó thực hiện quy trình đánh giá thì chắc hẳn bạn đã thấy rằng độ chính xác quá cao.

Công cụ đề xuất: Nhiệm vụ chính được thực hiện bởi thuật toán KNN là gán một điểm truy vấn mới cho một nhóm tồn tại trước đã được tạo bằng cách sử dụng một kho dữ liệu khổng lồ. Đây chính xác là những gì được yêu cầu trong hệ thống gợi ý để chỉ định mỗi người dùng vào một nhóm cụ thể và sau đó cung cấp cho họ các đề xuất dựa trên sở thích của nhóm đó.

### ***2.3.4. Ưu điểm***

* Đơn giản và dễ hiểu: Thuật toán KNN rất dễ hiểu và triển khai. Nó không yêu cầu các giả định phức tạp về dữ liệu.
* Không yêu cầu huấn luyện mô hình: KNN là thuật toán không cần huấn luyện trước (lazy learning). Điều này có nghĩa là không cần giai đoạn huấn luyện phức tạp như các thuật toán khác.
* Linh hoạt: KNN có thể được sử dụng cho cả bài toán phân loại và hồi quy. Nó cũng hoạt động tốt trên cả dữ liệu số và dữ liệu phân loại.
* Không yêu cầu giả thiết về phân phối dữ liệu: KNN không yêu cầu dữ liệu phải tuân theo bất kỳ phân phối nào, do đó nó linh hoạt trong nhiều loại dữ liệu khác nhau.
* Khả năng mở rộng: KNN có thể dễ dàng điều chỉnh độ phức tạp của mô hình thông qua tham số K. Điều này cho phép tìm được cân bằng giữa độ chính xác và độ phức tạp.

### ***2.3.5. Nhược điểm***

* Tốn kém về tính toán: KNN yêu cầu tính toán khoảng cách từ điểm cần dự đoán tới tất cả các điểm trong tập dữ liệu huấn luyện, điều này có thể rất tốn kém về mặt tính toán đối với các tập dữ liệu lớn.
* Nhạy cảm với dữ liệu nhiễu và ngoại lai: KNN có thể bị ảnh hưởng mạnh bởi các điểm dữ liệu ngoại lai hoặc nhiễu trong tập dữ liệu, điều này có thể dẫn đến dự đoán không chính xác.
* Hiệu suất kém trên dữ liệu nhiều chiều: Khi số chiều của dữ liệu tăng cao, khoảng cách giữa các điểm dữ liệu trở nên ít phân biệt hơn, điều này có thể làm giảm hiệu suất của KNN (curse of dimensionality).
* Yêu cầu bộ nhớ lớn: Vì KNN cần lưu trữ toàn bộ tập dữ liệu huấn luyện để tính toán khoảng cách, nó có thể yêu cầu một lượng lớn bộ nhớ, đặc biệt với các tập dữ liệu lớn.
* Khó khăn trong việc chọn giá trị K: Việc chọn giá trị K phù hợp là một vấn đề quan trọng. Giá trị K quá nhỏ có thể dẫn đến mô hình bị overfitting, trong khi giá trị K quá lớn có thể dẫn đến mô hình bị underfitting.
* Không cung cấp thông tin về tầm quan trọng của đặc trưng: KNN không có cơ chế để đánh giá tầm quan trọng của các đặc trưng khác nhau trong dữ liệu. Điều này có thể là hạn chế khi làm việc với dữ liệu có nhiều đặc trưng.

## **2.4. Giới thiệu về Python**

Python là một ngôn ngữ lập trình bậc cao, thông dịch và đa mục đích, được phát triển bởi Guido van Rossum và ra mắt lần đầu vào năm 1991. Python nổi tiếng với cú pháp rõ ràng và dễ đọc, điều này giúp các lập trình viên dễ dàng học và sử dụng.

### ***2.4.1. Ưu điểm***

* Cú pháp đơn giản và dễ đọc: Python có cú pháp gần gũi với ngôn ngữ tự nhiên, giúp các lập trình viên dễ dàng học và viết mã. Điều này cũng giúp giảm thời gian phát triển và bảo trì mã nguồn.
* Thư viện và framework phong phú: Python có một thư viện tiêu chuẩn rất mạnh mẽ và phong phú. Ngoài ra, có rất nhiều thư viện bên thứ ba hỗ trợ các lĩnh vực khác nhau như khoa học dữ liệu (NumPy, Pandas), học máy (TensorFlow, Scikit-learn), và phát triển web (Django, Flask).
* Đa nền tảng: Python có thể chạy trên nhiều hệ điều hành khác nhau như Windows, macOS, Linux, và Unix. Giúp các lập trình viên phát triển ứng dụng một cách linh hoạt và dễ dàng triển khai trên nhiều môi trường.
* Hỗ trợ nhiều phong cách lập trình: Python hỗ trợ lập trình thủ tục, lập trình hướng đối tượng, và lập trình hàm, cung cấp sự linh hoạt trong việc thiết kế và triển khai ứng dụng.
* Cộng đồng lớn và tích cực: Python có một cộng đồng người dùng rất lớn, cung cấp rất nhiều tài liệu, hướng dẫn, và hỗ trợ trực tuyến. Các hội nghị và sự kiện như PyCon tạo cơ hội cho các lập trình viên trao đổi kiến thức và kinh nghiệm.
* Tương tác với các ngôn ngữ khác: Python có thể dễ dàng tích hợp và tương tác với các ngôn ngữ lập trình khác như C, C++, và Java, giúp mở rộng khả năng sử dụng.

### ***2.4.2. Nhược điểm***

* Tốc độ thực thi: Python là ngôn ngữ thông dịch, do đó, tốc độ thực thi có thể chậm hơn so với các ngôn ngữ biên dịch như C++ hay Java. Điều này có thể trở thành vấn đề đối với các ứng dụng yêu cầu hiệu suất cao.
* Sử dụng bộ nhớ: Python sử dụng nhiều bộ nhớ hơn do bản chất của nó là ngôn ngữ cấp cao và thông dịch. Điều này có thể trở thành hạn chế khi phát triển các ứng dụng lớn hoặc trong các môi trường hạn chế tài nguyên.
* Thiếu kiểm tra kiểu tĩnh: Python là ngôn ngữ động, không kiểm tra kiểu tĩnh, điều này có thể dẫn đến lỗi thời gian chạy. Việc thiếu kiểm tra kiểu tĩnh cũng có thể làm cho việc duy trì mã nguồn lớn trở nên khó khăn hơn.
* Ứng dụng trong các tác vụ đa luồng: GIL (Global Interpreter Lock) của Python làm hạn chế khả năng thực hiện các tác vụ đa luồng hiệu quả trong Python. Điều này có thể là một vấn đề đối với các ứng dụng cần thực hiện nhiều tác vụ đồng thời.
* Không thích hợp cho một số loại ứng dụng: Python không phải là lựa chọn tốt nhất cho các ứng dụng di động hoặc các ứng dụng có yêu cầu thời gian thực cao do hạn chế về hiệu suất và quản lý bộ nhớ.

## **2.5. Một số thư viện của Python**

### ***2.5.1. Numpy***

NumPy (Numerical Python) là một thư viện cơ bản cho tính toán khoa học trong Python. Nó cung cấp hỗ trợ cho các mảng đa chiều (arrays), cùng với một tập hợp lớn các hàm toán học và logic để thao tác với các mảng này một cách hiệu quả.

* Ưu điểm:
* NumPy cung cấp hỗ trợ mạnh mẽ cho các mảng đa chiều, cho phép bạn thao tác với dữ liệu dưới dạng ma trận, tensor, v.v.
* NumPy tích hợp tốt với các thư viện khác như SciPy, Pandas, Matplotlib, và scikit-learn, giúp bạn xây dựng các ứng dụng phân tích dữ liệu và học máy dễ dàng hơn.
* NumPy cung cấp nhiều hàm toán học phức tạp như Fourier Transform, Linear Algebra, và Random Number Generation.
* Nhược điểm:
* Với những người mới bắt đầu học lập trình Python, việc làm quen với các khái niệm của NumPy và cách sử dụng các hàm của nó có thể khá khó khăn.
* Mặc dù NumPy mảng mạnh mẽ và hiệu quả, nhưng chúng ít linh hoạt hơn so với danh sách Python khi cần lưu trữ các kiểu dữ liệu hỗn hợp hoặc thực hiện các thao tác phức tạp không được hỗ trợ trực tiếp.
* Đối với các phép tính rất phức tạp hoặc không tiêu chuẩn, bạn có thể phải viết thêm mã C hoặc sử dụng các thư viện khác để mở rộng khả năng của NumPy.

### ***2.5.2. Keras***

Keras là một thư viện mã nguồn mở dành cho việc xây dựng và huấn luyện các mô hình học sâu (deep learning) một cách dễ dàng và hiệu quả. Nó được thiết kế để giúp người dùng phát triển nhanh chóng các mô hình học sâu mà không cần phải xử lý nhiều chi tiết phức tạp của các framework học sâu cơ bản. Keras có thể chạy trên các backend khác nhau như TensorFlow, Theano, và Microsoft Cognitive Toolkit .

* Ưu điểm:
* Keras được thiết kế với trọng tâm là khả năng sử dụng và khả năng mở rộng, với một API trực quan và đơn giản giúp người dùng dễ dàng bắt đầu và xây dựng các mô hình phức tạp.
* Keras là một phần của TensorFlow và được tích hợp chặt chẽ, cho phép người dùng tận dụng các tính năng tiên tiến của TensorFlow như phân tán và tối ưu hóa hiệu suất.
* Keras có một cộng đồng lớn và nhiều tài liệu, hướng dẫn, và ví dụ thực tế, giúp người dùng học hỏi và giải quyết các vấn đề một cách nhanh chóng.
* Nhược điểm:
* Mặc dù Keras rất tiện dụng, nhưng đôi khi nó có thể chậm hơn so với việc sử dụng trực tiếp các framework backend như TensorFlow, vì nó có thể tạo ra thêm lớp trừu tượng và do đó tăng thêm chi phí tính toán.
* Keras thiết kế để đơn giản hóa quá trình phát triển mô hình, điều này cũng có nghĩa là người dùng sẽ có ít kiểm soát chi tiết hơn so với việc sử dụng trực tiếp các framework backend. Điều này có thể là một hạn chế đối với các mô hình yêu cầu tùy chỉnh cao.
* Do Keras là một lớp trừu tượng trên các framework backend, việc khắc phục sự cố có thể trở nên phức tạp hơn khi bạn phải tìm hiểu qua nhiều lớp mã để tìm ra nguyên nhân của vấn đề.

### ***2.4.3. Pandas***

Pandas là một thư viện mã nguồn mở trong Python, được thiết kế để hỗ trợ thao tác và phân tích dữ liệu. Pandas cung cấp các cấu trúc dữ liệu và hàm thao tác với bảng số liệu và chuỗi thời gian (time series) một cách dễ dàng và trực quan. Hai cấu trúc dữ liệu chính của Pandas là DataFrame (bảng dữ liệu 2 chiều) và Series (dữ liệu 1 chiều).

* Ưu điểm
* Pandas cung cấp các hàm thao tác dữ liệu rất mạnh mẽ và dễ sử dụng, giúp bạn dễ dàng làm sạch, biến đổi và phân tích dữ liệu.
* Pandas được xây dựng trên nền tảng NumPy, do đó thừa hưởng hiệu suất cao và khả năng xử lý hiệu quả của NumPy.
* Pandas có thể xử lý nhiều loại dữ liệu khác nhau, bao gồm dữ liệu từ CSV, Excel, SQL, và nhiều định dạng khác.
* Pandas cung cấp nhiều công cụ và hàm mạnh mẽ để phân tích dữ liệu, bao gồm lọc dữ liệu, nhóm dữ liệu (grouping), ghép dữ liệu (merging), và xử lý dữ liệu chuỗi thời gian.
* Nhược điểm
* Khi xử lý các bộ dữ liệu rất lớn, Pandas có thể tiêu tốn nhiều bộ nhớ, gây khó khăn cho việc làm việc với các bộ dữ liệu không vừa bộ nhớ (memory).
* Mặc dù Pandas rất nhanh cho các bộ dữ liệu vừa và nhỏ, nhưng khi làm việc với các bộ dữ liệu rất lớn, nó có thể trở nên chậm chạp và kém hiệu quả.
* Đối với những người mới bắt đầu, cú pháp và các hàm của Pandas có thể hơi phức tạp và khó nắm bắt.

### ***2.4.4. Scikit-learn***

Scikit-learn là một thư viện Python được sử dụng rộng rãi cho học máy và phân tích dữ liệu. Đây là một công cụ mạnh mẽ và linh hoạt, bao gồm nhiều thuật toán học máy, từ học giám sát (như phân loại và hồi quy) đến học không giám sát (như phân cụm và giảm chiều). Scikit-learn được thiết kế để tích hợp dễ dàng với các thư viện khoa học số liệu của Python như NumPy và SciPy.

* Ưu điểm:
* Dễ sử dụng: Scikit-learn có API đơn giản và nhất quán, làm cho quá trình triển khai và thử nghiệm các mô hình học máy trở nên dễ dàng hơn.
* Tài liệu hướng dẫn tốt: Thư viện có một lượng tài liệu rất lớn và hữu ích, bao gồm nhiều ví dụ và hướng dẫn cụ thể, giúp người mới bắt đầu có thể dễ dàng tiếp cận.
* Cộng đồng mạnh mẽ và hỗ trợ rộng rãi: Với cộng đồng lớn và năng động, các vấn đề và cập nhật thường được giải quyết nhanh chóng.
* Hiệu suất tối ưu: Scikit-learn được tối ưu hóa cho hiệu suất, với một số thành phần được viết bằng Python để tăng tốc.
* Hỗ trợ nhiều thuật toán: Bao gồm một loạt các thuật toán cho cả học có giám sát và không giám sát, cũng như các công cụ cho tiền xử lý dữ liệu, chọn mô hình và đánh giá mô hình.
* Nhược điểm:
* Không tốt cho quy mô lớn: Scikit-learn không được thiết kế cho dữ liệu cực kỳ lớn hoặc tính toán phân tán, điều này có thể hạn chế khi làm việc với bộ dữ liệu cỡ lớn hay trong môi trường phân tán.
* Không hỗ trợ mô hình học sâu: Không có hỗ trợ bản địa cho các mô hình học sâu như mạng nơ-ron, người dùng cần sử dụng các thư viện khác như TensorFlow hoặc PyTorch cho những nhu cầu đó.
* Thiếu tính linh hoạt: Mặc dù Scikit-learn cung cấp nhiều thuật toán, nhưng nó không cho phép tùy chỉnh sâu các thuật toán đó. Điều này có thể không đáp ứng được cho những nhu cầu rất cụ thể hoặc nâng cao.
* Tối ưu hóa tham số có hạn: Mặc dù có các công cụ như GridSearchCV và RandomizedSearchCV cho tìm kiếm tham số, nhưng quá trình tìm kiếm này có thể không hiệu quả về mặt tính toán đối với các không gian tham số lớn.

***2.4.5. PyQt***

PyQt là một thư viện GUI (Graphical User Interface) tiêu chuẩn cho Python. Nó cho phép tạo ra các ứng dụng đồ họa một cách đơn giản và nhanh chóng.

* **Ưu điểm**
* **Dễ học và sử dụng**: PyQt có hỗ trợ phần mềm kéo thả nhanh chóng, PyQt là một lựa chọn tuyệt vời cho người mới bắt đầu học lập trình GUI.
* **Đa nền tảng**: PyQt hoạt động trên nhiều hệ điều hành khác nhau như Windows, macOS và Linux mà không cần thay đổi mã nguồn.
* **Tích hợp tốt với Python**: PyQt là thư viện GUI chính thức cho Python, do đó, nó tích hợp rất tốt với ngôn ngữ này và có thể tận dụng tất cả các tính năng của Python.
* **Nhược điểm**
* **Giao diện đơn giản**: PyQt cung cấp các widget cơ bản và có giao diện khá cổ điển, không hỗ trợ các tính năng hiện đại và tinh vi như một số thư viện GUI khác.
* **Hiệu suất thấp**: Đối với các ứng dụng phức tạp và yêu cầu cao về hiệu suất, PyQt có thể không phải là lựa chọn tốt nhất do hạn chế về khả năng xử lý và tốc độ.
* **Hạn chế về tùy chỉnh giao diện**: Khả năng tùy chỉnh giao diện của PyQt không mạnh mẽ như một số framework khác như Tkinter hoặc Kivy.
* **Cập nhật chậm**: PyQt không được cập nhật thường xuyên như một số thư viện khác, do đó, có thể thiếu các tính năng mới và cải tiến.

***2.4.6. Matplotlib***

Matplotlib là một trong những thư viện đồ họa phổ biến nhất trong Python, đặc biệt hữu ích trong lĩnh vực khoa học dữ liệu và học máy.

* **Ưu điểm**
* **Đa dạng về loại biểu đồ**: Matplotlib cung cấp một loạt các loại biểu đồ như biểu đồ đường, biểu đồ thanh, biểu đồ phân tán, biểu đồ hộp, biểu đồ tròn, và nhiều loại biểu đồ khác.
* **Tùy chỉnh mạnh mẽ**: Bạn có thể tùy chỉnh hầu hết mọi khía cạnh của biểu đồ, từ màu sắc, kiểu đường, đánh dấu, nhãn trục, đến các chú thích và tiêu đề.
* **Tích hợp tốt với các thư viện khác**: Matplotlib tích hợp tốt với các thư viện như NumPy, Pandas, và Seaborn. Điều này giúp bạn dễ dàng trực quan hóa dữ liệu từ các DataFrame của Pandas hoặc các mảng của NumPy.
* **Khả năng tạo biểu đồ chất lượng cao**: Matplotlib có thể tạo ra các biểu đồ chất lượng cao phù hợp cho các ấn phẩm khoa học và báo cáo.
* **Nhược điểm**
* **Cú pháp phức tạp**: Cú pháp của Matplotlib có thể khá phức tạp và khó nhớ, đặc biệt khi bạn cần tùy chỉnh nhiều khía cạnh của biểu đồ. Điều này có thể dẫn đến mã nguồn dài và khó đọc.
* **Thiếu tính hiện đại và tương tác**: So với các thư viện hiện đại như Plotly hoặc Bokeh, Matplotlib thiếu các tính năng tương tác và trực quan hóa hiện đại. Các biểu đồ thường tĩnh và không hỗ trợ tương tác như zoom, pan, hay cập nhật dữ liệu động.
* **Hiệu suất không cao**: Matplotlib có thể chậm khi xử lý các bộ dữ liệu lớn hoặc khi vẽ nhiều biểu đồ phức tạp cùng một lúc. Các thư viện khác như Plotly hoặc Bokeh có thể cung cấp hiệu suất tốt hơn trong các trường hợp này.
* **Giao diện không nhất quán**: Matplotlib có nhiều cách để làm cùng một việc, dẫn đến sự không nhất quán trong mã nguồn và làm cho việc học và sử dụng thư viện này trở nên khó khăn hơn.

# **CHƯƠNG 3: ỨNG DỤNG LSTM TRONG DỰ ĐOÁN CHỨNG KHOÁN**

## **3.1. Giới thiệu về dữ liệu**

Dữ liệu giá cổ phiểu Apple, được lấy trực tiếp từ Kaggle. Kaggle là một trong những nguồn cung cấp dữ liệu lớn, bao gồm thông tin về giá cổ phiếu, các tệp dữ liệu ảnh, và các dữ liệu tài chính khác, Kaggle là nơi chứa những tệp dữ liệu phổ biến khi huấn luyện mô hình. Chính vì thế tôi đã sử dụng bộ dữ liệu trên Kaggle.

Dữ liệu gồm các thông tin sau:

A table with numbers and letters

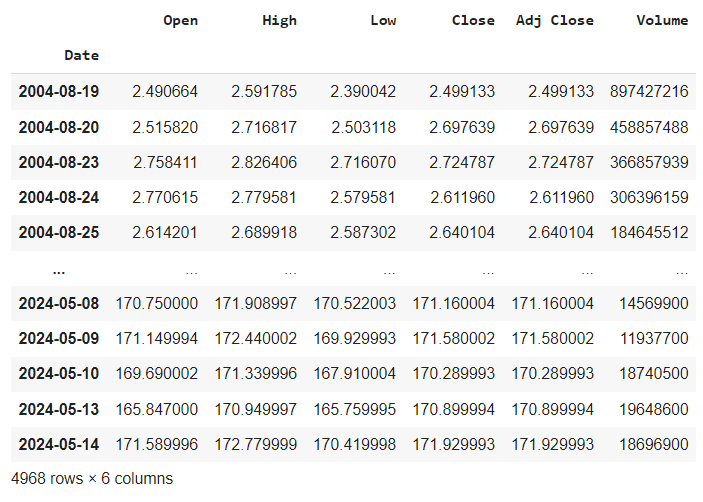
Description automatically generated

Hình 3. . Hình ảnh dữ liệu được lấy từ Yahoo Finance

* Date: Ngày xác định dữ liệu mà nó được ghi nhận.
* Open: Đại diện cho giá mở cửa của chứng khoán vào đầu phiên giao dịch của ngày củ thể đó. Đây là giá mà chứng khoán bắt đầu được giao dịch khi thị trường mở cửa vào buổi sáng. Giá mở cửa là một chỉ số quan trọng vì nó có thể phản ánh phản ứng của thị trường đối với các tin tức hoặc sự kiện xảy ra ngoài giờ giao dịch.
* High: đại diện cho giá cao nhất mà cổ phiếu hoặc tài sản đạt được trong phiên giao dịch của ngày cụ thể đó. Đây là mức giá cao nhất mà các nhà giao dịch đã sẵn sàng trả cho cổ phiếu trong suốt khoảng thời gian giao dịch của ngày đó.
* Low: đại diện cho giá thấp nhất mà cổ phiếu hoặc tài sản đạt được trong phiên giao dịch của ngày cụ thể đó. Đây là mức giá thấp nhất mà các nhà giao dịch đã sẵn sàng bán cổ phiếu trong suốt khoảng thời gian giao dịch của ngày đó.
* Close: đại diện cho giá đóng cửa của cổ phiếu hoặc tài sản vào cuối phiên giao dịch của ngày cụ thể đó. Đây là mức giá cuối cùng mà cổ phiếu được giao dịch trước khi thị trường đóng cửa. Giá đóng cửa là một trong những chỉ số quan trọng nhất vì nó thường được sử dụng để đánh giá hiệu suất hàng ngày của cổ phiếu và là cơ sở cho nhiều phân tích kỹ thuật và tài chính.
* Adj Close: đại diện cho giá đóng cửa được điều chỉnh để phản ánh các sự kiện như chia cổ tức, chia tách cổ phiếu (stock split), và các thay đổi về vốn khác. Giá này cung cấp một cái nhìn chính xác hơn về giá trị thực sự của cổ phiếu sau khi các điều chỉnh này được tính đến.
* Volume: đại diện cho khối lượng giao dịch, tức là tổng số lượng cổ phiếu hoặc tài sản được mua và bán trong một phiên giao dịch cụ thể. Khối lượng giao dịch là một chỉ số quan trọng vì nó cung cấp thông tin về mức độ hoạt động và sự quan tâm của thị trường đối với một cổ phiếu cụ thể.

## **3.2. Xử lý dữ liệu //**

Lấy dữ liệu 20 năm của mã chứng khoán ‘AAPL’ (mã chứng khoán của Tập đoàn Alphabet), ta được một bảng dữ liệu gồm hơn 10000 hàng và 6 cột như hình dưới:



Hình 3. . Dữ liệu mã chứng khoán AAPL

Tiến hành phân tích dữ liệu thấy dữ liệu sạch không có ô nào khuyết.

A screenshot of a computer code

Description automatically generated

Hình 3. . Kết quả thông tin của dữ liệu được lấy

* Lấy tập dữ liệu:

Adj\_close\_price = google\_data[['Adj Close']]

* Tiến hành lấy cột dữ liệu ‘Adj Close’ của tập dữ liệu đã lấy ở trên để chuẩn bị chuẩn hóa và chia tập dữ liệu thành train model và test model train.
* Chuẩn hóa dữ liệu:

from sklearn.preprocessing import MinMaxScaler

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

scaled\_data = scaler.fit\_transform(Adj\_close\_price)

scaled\_data

* MinMaxScaler: Được sử dụng để chuẩn hóa dữ liệu sao cho tất cả các giá trị nằm trong khoảng từ 0 đến 1.
* fit\_transform: Hàm này sẽ học các giá trị cực đại và cực tiểu từ dữ liệu đầu vào Adj\_close\_price và sau đó chuyển đổi dữ liệu này thành khoảng từ 0 đến 1.

## **3.3. Huấn luyện mô hình**

* Tạo tập dữ liệu huấn luyện và kiểm tra

x\_data = []

y\_data = []

for i in range(60, len(scaled\_data)):

    x\_data.append(scaled\_data[i-60:i])

    y\_data.append(scaled\_data[i])

import numpy as np

x\_data, y\_data = np.array(x\_data), np.array(y\_data)

* x\_data: Chứa các tập con của dữ liệu với độ dài là 60 (sử dụng 60 bước thời gian để dự đoán bước thời gian tiếp theo).
* y\_data: Chứa giá trị tiếp theo cần dự đoán từ x\_data.
* np.array: Chuyển đổi danh sách x\_data và y\_data thành mảng numpy để dễ dàng xử lý trong mô hình.
* Chia tập dữ liệu thành huấn luyện và kiểm tra

splitting\_len = int(len(x\_data)\*0.7)

x\_train = x\_data[:splitting\_len]

y\_train = y\_data[:splitting\_len]

x\_test = x\_data[splitting\_len:]

y\_test = y\_data[splitting\_len:]

* splitting\_len: Tính toán số lượng mẫu dữ liệu thuộc tập huấn luyện (70% của tổng dữ liệu).
* x\_train, y\_train: Tập huấn luyện chứa các mẫu dữ liệu và nhãn tương ứng.
* x\_test, y\_test: Tập kiểm tra chứa các mẫu dữ liệu và nhãn tương ứng.
* Khởi tạo lớp LSTM:

class LSTM:

    def \_\_init\_\_(self, input\_dim, hidden\_dim):

        self.input\_dim = input\_dim

        self.hidden\_dim = hidden\_dim

        # Initialize weights

        self.Wf = np.random.randn(hidden\_dim, hidden\_dim + input\_dim)

        self.Wi = np.random.randn(hidden\_dim, hidden\_dim + input\_dim)

        self.Wo = np.random.randn(hidden\_dim, hidden\_dim + input\_dim)

        self.Wc = np.random.randn(hidden\_dim, hidden\_dim + input\_dim)

        self.bf = np.zeros((hidden\_dim, 1))

        self.bi = np.zeros((hidden\_dim, 1))

        self.bo = np.zeros((hidden\_dim, 1))

        self.bc = np.zeros((hidden\_dim, 1))

* \_\_init\_\_ method: Phương thức khởi tạo cho lớp LSTM.
* input\_dim: Kích thước của đầu vào tại mỗi bước thời gian.
* hidden\_dim: Kích thước của trạng thái ẩn.
* Wf, Wi, Wo, Wc: Ma trận trọng số tương ứng cho cổng quên, cổng đầu vào, cổng đầu ra và trạng thái ứng cử viên.
* bf, bi, bo, bc: Vector bias tương ứng cho các cổng và trạng thái ứng cử viên.
* Các hàm kích hoạt:

  def sigmoid(self, x):

        return 1 / (1 + np.exp(-x))

    def tanh(self, x):

        return np.tanh(x)

* Hàm sigmoid: Được sử dụng để tính toán đầu ra của các cổng quên, cổng đầu vào và cổng đầu ra. Hàm sigmoid sẽ nén đầu vào vào khoảng (0,1).
* Hàm tanh: Được sử dụng để tính toán trạng thái ứng cử viên và trạng thái ẩn. Hàm tanh sẽ nén đầu vào vào khoảng (-1, 1).
* Phương thức forward:

def forward(self, x):

        h, c = np.zeros((self.hidden\_dim, 1)), np.zeros((self.hidden\_dim, 1))

        for t in range(x.shape[0]):

            x\_t = x[t].reshape(-1, 1)

            concat = np.vstack((h, x\_t))

            f = self.sigmoid(np.dot(self.Wf, concat) + self.bf)

            i = self.sigmoid(np.dot(self.Wi, concat) + self.bi)

            o = self.sigmoid(np.dot(self.Wo, concat) + self.bo)

            c\_hat = self.tanh(np.dot(self.Wc, concat) + self.bc)

            c = f \* c + i \* c\_hat

            h = o \* self.tanh(c)

        return h, c

* forward method: Phương thức tính toán trạng thái ẩn và trạng thái ô qua các bước thời gian.
* h, c: Trạng thái ẩn và trạng thái ô được khởi tạo là các vector không.
* for t in range(x.shape[0]): Vòng lặp qua các bước thời gian của đầu vào x.
* x\_t = x[t].reshape(-1, 1): Lấy đầu vào tại thời điểm t và chuyển thành vector cột.
* concat = np.vstack((h, x\_t)): Kết hợp trạng thái ẩn hiện tại và đầu vào tại thời điểm t.
* f, i, o, c\_hat: Tính toán đầu ra của các cổng quên, đầu vào, đầu ra và trạng thái ứng cử viên sử dụng các trọng số và bias tương ứng.
* c = f \* c + i \* c\_hat: Cập nhật trạng thái ô.
* h = o \* self.tanh(c): Cập nhật trạng thái ẩn.
* Định nghĩa và huấn luyện mô hình:

from keras.models import Sequential

from keras.layers import Dense

+ Nhập các lớp cần thiết từ Keras:

* Sequential: Lớp để tạo mô hình tuần tự.
* Dense, LSTM: Các lớp Dense và LSTM để xây dựng mô hình

model = Sequential()

model.add(LSTM(128, return\_sequences=True, input\_shape=(x\_train.shape[1],1)))

model.add(LSTM(64,return\_sequences=False))

model.add(Dense(25))

model.add(Dense(1))

+ Tạo mô hình tuần tự và thêm các lớp vào mô hình:

* Lớp LSTM đầu tiên với 128 đơn vị và trả về toàn bộ chuỗi đầu ra (return\_sequences=True). Kích thước đầu vào là (x\_train.shape[1], 1).
* Lớp LSTM thứ hai với 64 đơn vị và không trả về toàn bộ chuỗi đầu ra (return\_sequences=False).
* Lớp Dense với 25 đơn vị.
* Lớp Dense đầu ra với 1 đơn vị để dự đoán giá trị cuối cùng.

model.compile(optimizer='adam', loss='mean\_squared\_error')

* Sử dụng bộ tối ưu hóa Adam và hàm mất mát mean\_squared\_error.

model.fit(x\_train, y\_train, batch\_size=1, epochs = 2)

* Huấn luyện mô hình với dữ liệu x\_train và y\_train, kích thước batch là 1 và số epoch là 2.

model.summary()

* Hiển thị thông tin chi tiết về các lớp và số lượng tham số trong mô hình.

## **3.4. Kết quả thực nghiệm**

Dự đoán giá trong tương lai dựa trên dữ liệu 60 ngày gần nhất bằng cách sử dụng một mô hình đã huấn luyện (chẳng hạn như LSTM) và bộ điều chỉnh tỷ lệ (scaler).

import numpy as np

def predict\_future\_days(model, scaler, last\_60days\_sc, days\_to\_predict=7):

    future\_predictions = []

    current\_input = last\_60days\_sc.copy()  # Sao chép dữ liệu 60 ngày cuối cùng

    for \_ in range(days\_to\_predict):

        # Chuẩn bị dữ liệu để dự đoán

        X\_test = []

        X\_test.append(current\_input)

        X\_test = np.array(X\_test)

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

        # Dự đoán giá

        pred\_price = model.predict(X\_test)

        pred\_price\_inverse = scaler.inverse\_transform(pred\_price)  # Chuyển đổi về giá trị gốc

        future\_predictions.append(pred\_price\_inverse[0][0])

        # Cập nhật đầu vào cho lần dự đoán tiếp theo

        new\_input = np.append(current\_input[1:], pred\_price)  # Thêm giá dự đoán vào chuỗi dữ liệu, loại bỏ giá trị đầu tiên

        current\_input = new\_input.reshape(-1, 1)

    return future\_predictions

# Sử dụng hàm để dự đoán 7 ngày tới

predicted\_prices = predict\_future\_days(model, scaler, last\_60days\_sc, days\_to\_predict=7)

print("Dự đoán giá đóng cửa điều chỉnh 7 ngày tới: ", predicted\_prices)

Kết quả:

A close-up of a number

Description automatically generated

Hình 3. . Hình ảnh kết quả dự đoán giá của mã chứng khoán 7 ngày tới

Biểu đồ được vẽ trước khi dự đoán:

A graph showing a line

Description automatically generated with medium confidence

Hình 3. . Biểu đồ mã chứng khoán trước khi dự đoán

Biểu đồ sau khi dự đoán:

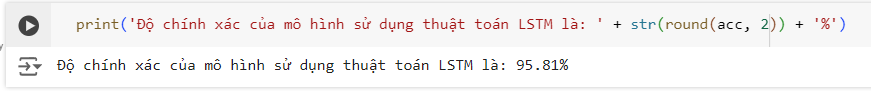
**A graph with blue lines

Description automatically generated**

Hình 3. . Biểu đồ mã chứng khoán khi thêm phần dự đoán

## **3.5. So sánh độ chính xác của mô hình**

Độ chính xác của mô hình sử dụng thuật toán LSTM:



Hình 3. . Độ chính xác của mô hình LSTM

Độ chính xác của mô hình sử dụng thuật toán KNN:

A close up of text

Description automatically generated

Hình 3. . Độ chính xác của mô hình KNN

Từ kết quả độ chính xác của 2 mô hình ta thấy mô hình LSTM có độ chính xác cao hơn mô hình KNN.

## **3.6. Xây dựng giao diện ứng dụng chứng khoán**

Giao diện đăng nhập của ứng dụng:

**A screenshot of a login page

Description automatically generated**

Hình 3. . Giao diện đăng nhập

Giao diện đăng kí của ứng dụng:

A screenshot of a computer

Description automatically generated

Hình 3. . Giao diện đăng kí

Giao diện lấy lại mật khẩu:

A screenshot of a computer screen

Description automatically generated

Hình 3. . Giao diện lấy lại mật khẩu

Sau khi đăng nhập thành công, giao diện ứng dụng xuất hiện gồm có:

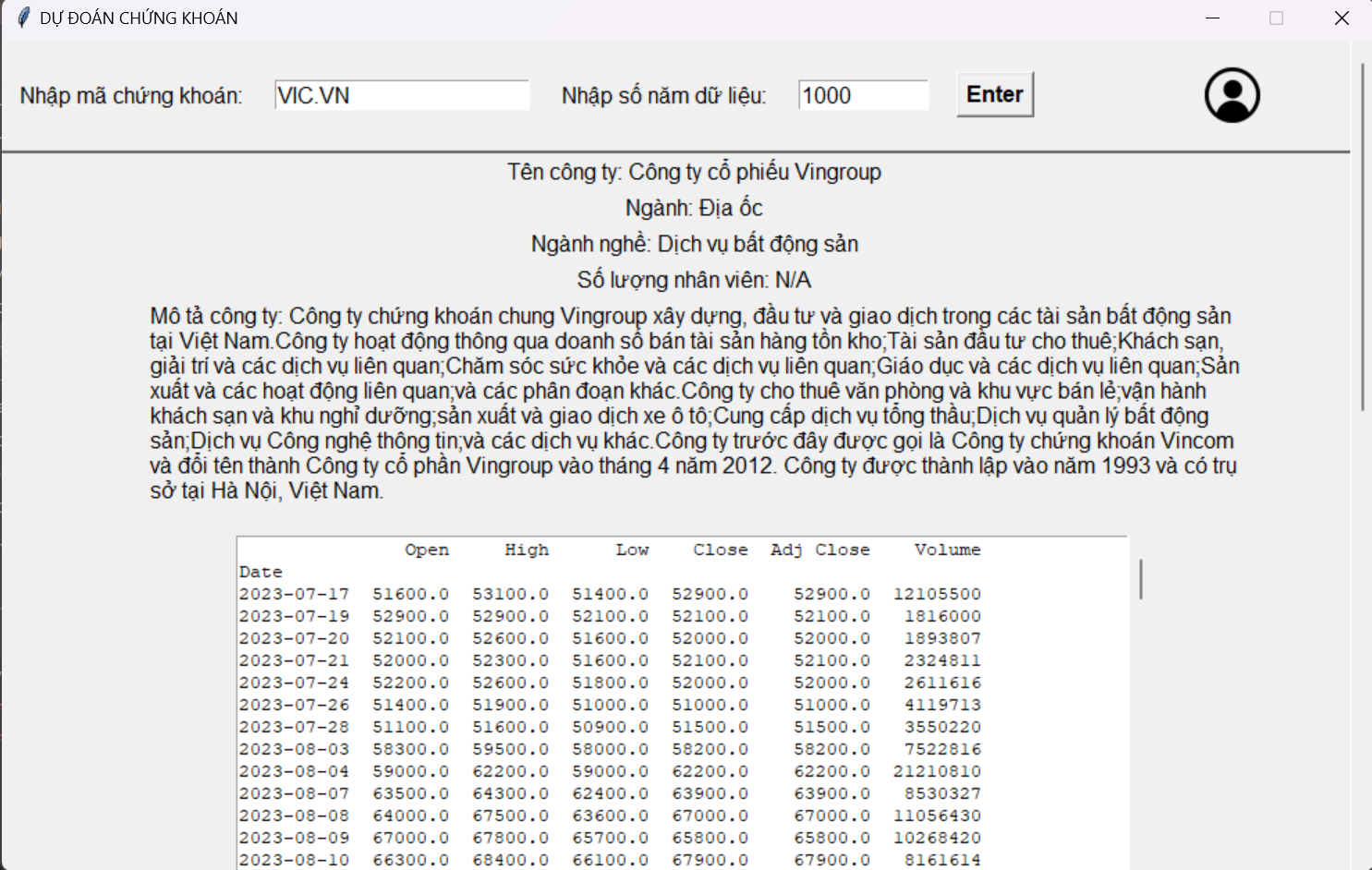
* Nhập mã chứng khoán: Nơi nhập mã chứng khoán mà mình muốn dự đoán
* Nhập số năm dữ liệu: Nơi nhập số năm dữ liệu mà mình muốn hiện thị ra (tính từ ngày hiện tại mình nhập đến ngày cách đó số năm mình đã nhập)

A screenshot of a computer

Description automatically generated

Hình 3. . Giao diện ứng dụng dự đoán mã chứng khoán

Sau khi nhập đủ các thông tin như: mã chứng khoán và số năm, nhấn ‘Enter’ ứng dụng sẽ xuất hiện các thông tin về mã chứng khoán, biểu đồ, giá dự đoán và biểu đồ dự đoán



Hình 3. . Giao diện khi đã nhập mã chứng khoán

## **3.7. Kiểm thử mô hình**

### ***3.7.1. Kế hoạch kiểm thử***

* Kiểm thử mô hình:
* Kiểm thử tải dữ liệu: trả về DataFrame chứa dữ liệu tải về từ yf.download.
* Kiểm thử chuẩn bị dữ liệu: trả về các giá trị đúng cho new\_df, last\_60days\_sc, và scaler.
* Kiểm thử dự đoán dữ liệu: trả về các giá trị dự đoán chính xác dựa trên model và dữ liệu đầu vào.
* Kiểm thử vẽ đồ thị: đảm bảo không gây ra lỗi khi vẽ đồ thị.
* Kiểm thử giao diện đăng nhập:
* Kiểm thử giao diện: hiện thị đầy đủ các thành phần của giao diện
* Kiểm thử chức năng đăng nhập: đăng nhập được vào ứng dụng
* Kiểm thử chức năng đăng kí: đăng kí được tài khoản mới
* Kiểm thử chức năng quên mật khẩu: hiện thị cửa sổ quên mật khẩu
* Kiểm thử chức năng ẩn và hiện mật khẩu: ẩn và hiện mật khẩu hoạt động đúng

### ***3.7.2. Kiểm thử mô hình***

Bảng 3. . Bảng kiểm thử mô hình

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Case | Đầu vào | Đầu ra mong muốn | Kết quả |
| 1 | Kiểm thử hàm load\_stock\_data | * stock: Mã cổ phiếu (string). * start: Ngày bắt đầu (datetime). * end: Ngày kết thúc (datetime). | DataFrame chứa dữ liệu về giá cổ phiếu, được tải về từ yfinance. | Pass |
| 2 | Kiểm thử hàm prepare\_data | * data: DataFrame chứa dữ liệu về giá cổ phiếu. * days: Số ngày cuối cùng được sử dụng để chuẩn bị dữ liệu (int). | * new\_df: DataFrame chứa dữ liệu giá cổ phiếu đã lọc. * last\_60days\_sc: Mảng NumPy chứa giá cổ phiếu chuẩn hóa của 60 ngày cuối cùng. * scaler: MinMaxScaler đã được fit với dữ liệu. | Pass |
| 3 | Kiểm thử hàm predict\_future\_days | * model: Mô hình dự đoán giá cổ phiếu. * scaler: MinMaxScaler đã được fit với dữ liệu. * last\_60days\_sc: Mảng NumPy chứa giá cổ phiếu chuẩn hóa của 60 ngày cuối cùng. * days\_to\_predict: Số ngày dự đoán tiếp theo (int). | Mảng NumPy chứa giá cổ phiếu dự đoán cho số ngày tiếp theo. | Pass |
| 4 | Kiểm thử hàm plot\_predictions | * original\_data: DataFrame chứa dữ liệu giá cổ phiếu gốc. * predicted\_data: Mảng NumPy chứa giá cổ phiếu dự đoán cho các ngày tiếp theo.   - stock: Mã cổ phiếu (string). | Biểu đồ matplotlib hiển thị dữ liệu giá cổ phiếu gốc và dự đoán. | Pass |

### ***3.7.3. Kiểm thử giao diện đăng nhập***

Bảng 3. . Bảng kiểm thử giao diện đăng nhập

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Case | Đầu vào | Đầu ra mong muốn | Kết quả |
| 1 | Giao diện người dùng | Kiểm tra xem các thành phần giao diện như cửa sổ đăng nhập, các label, các entry field, nút đăng nhập, nút đăng ký, nút quên mật khẩu có hiển thị đúng không. | Giao diện người dùng được hiển thị đúng và đầy đủ, không có lỗi hiển thị. | Pass |
| 2 | Đăng nhập | Người dùng nhập tên người dùng và mật khẩu đúng. | Hiển thị thông báo "Đăng nhập thành công" và chuyển đến trang chính của ứng dụng. | Pass |
| Người dùng nhập tên người dùng hoặc mật khẩu không chính xác. | Hiển thị thông báo lỗi "Tài khoản không tồn tại hoặc mật khẩu không đúng". | Pass |
| 3 | Đăng kí | Người dùng nhập đầy đủ thông tin và mật khẩu | Hiển thị thông báo "Đăng ký thành công". | Pass |
| Người dùng nhập tên người dùng đã tồn tại, email đã được sử dụng | Hiển thị thông báo lỗi tương ứng. | Pass |
| 4 | Quên mật khẩu | Người dùng click vào nút quên mật khẩu. | Hiển thị cửa sổ quên mật khẩu và đóng cửa sổ đăng nhập. | Pass |
| 5 | Ẩn mật khẩu | Người dùng click vào nút ẩn mật khẩu. | Mật khẩu được ẩn và nút chuyển thành nút hiển thị mật khẩu. | Pass |
| 6 | Hiện thị mật khẩu | Người dùng click vào nút hiển thị mật khẩu. | Mật khẩu được hiển thị và nút chuyển thành nút ẩn mật khẩu. | Pass |

### ***3.7.4. Kết quả kiểm thử***

* Tỉ lệ test case đạt (Passed): 100%

# **KẾT LUẬN**

Đề tài “Xây dựng ứng dụng dự đoán sự biến thiên của các mã chứng khoán trong thị trường chứng khoán bằng Python” xuất phát từ những nhu cầu thực tế mà ngày nay trong lĩnh vực tài chính – chứng khoán cần có. Qua quá trình thực hiện đề tài, đã tích lũy và vận dụng được những kiến thức công nghệ được học từ trường, từ nơi thực tập và hoàn thành được những phần sau đây:

Nắm vững quy trình sử dụng thuật toán học máy, học sâu để huấn luyện mô hình và dự đoán ra kết quả.

Có kế hoạch và đánh giá kiểm thử chặt chẽ.

Xây dựng ra giao diện ứng dụng có các chức năng như: đăng kí, đăng nhập, quên mật khẩu, đổi mật khẩu, dự đoán chứng khoán.

**Bên cạnh đó còn có những hạn chế**:

Giao diện dự đoán chưa được bắt mắt, còn nhiều thiếu sót.

Ứng dụng chưa tuân thủ tiêu chuẩn bảo mật.

**Hướng phát triển**

* Tiếp tục hoàn thiện các tính năng mở rộng phần mềm nhằm cải tiến và nâng cấp chương trình.
* Từng bước xây dựng ứng dụng đa nền tảng
* Hoàn thiện chương trình, sửa chữa các lỗi, cải thiện tính bảo mật.
* Tích hợp đăng nhập qua Facebook, Google.

# **TÀI LIỆU THAM KHẢO**

**Tiếng việt**

[1] Trần Hùng Cường, Trần Thanh Hùng, *Giáo trình khai phá dữ liệu*, NXB Đại học Công nghiệp Hà Nội.

[2] Nguyễn Phương Nga, Trần Hùng Cường, *Giáo trình trí tuệ nhân tạo*

[3] Trang tài liệu về chứng khoán

[https://chung-khoan-la-gi-phan-biet-cac-loai-chung-khoan.html](https://vnexpress.net/chung-khoan-la-gi-phan-biet-cac-loai-chung-khoan-4568366.html)

[4] Trang tài liệu về mô hình mạng RNN

<https://nttuan8.com/bai-13-recurrent-neural-network/>

[5] Trang tài liệu về mô hình mạng LSTM

[https://Ly\_thuyet\_ve\_mang\_LSTM.html](https://phamdinhkhanh.github.io/2019/04/22/Ly_thuyet_ve_mang_LSTM.html)

[6] Trang tài liệu về thuật toán KNN

[https://vietnambiz.vn/thuat-toan-k-lang-gieng-gan-nhat.html](https://vietnambiz.vn/thuat-toan-k-lang-gieng-gan-nhat-k-nearest-neighbor-knn-la-gi-2020022911113334.htm)

**Tiếng Anh**

[7] Machine Learning with Python for Everyone (Addison-Wesley Data & Analytics Series) – Softcover

[8] L. F. Sanchez, “Automatic Number Plate Recognition System Using Machine Learning Techniques,” no. August, pp. 2017–2018, 2018.

[9] Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2