TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

BỘ MÔN KỸ THUẬT DỮ LIỆU

NGUYỄN ĐỨC LINH (MSSV: 20133007)

PHAN THANH TÍN (MSSV: 20133095)

Đề tài:

ỨNG DỤNG HỌC MÁY DỰ ĐOÁN SỐ LƯỢNG SẢN PHẨM BÁN RA ĐƯỢC TRÊN TRANG THƯƠNG MẠI ĐIỆN TỬ

KHÓA LUẬN TỐT NGHIỆP KỸ SƯ KỸ THUẬT DỮ LIỆU

GIÁO VIÊN HƯỚNG DẪN

TS. TRẦN NHẬT QUANG

Tp. Hồ Chí Minh, ngày 10 tháng 7 năm 2024

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

BỘ MÔN KỸ THUẬT DỮ LIỆU

NGUYỄN ĐỨC LINH (MSSV: 20133007)

PHAN THANH TÍN (MSSV: 20133095)

Đề tài:

ỨNG DỤNG HỌC MÁY DỰ ĐOÁN SỐ LƯỢNG SẢN PHẨM BÁN RA ĐƯỢC TRÊN TRANG THƯƠNG MẠI ĐIỆN TỬ

KHÓA LUẬN TỐT NGHIỆP KỸ SƯ KỸ THUẬT DỮ LIỆU

GIÁO VIÊN HƯỚNG DẪN

TS. TRẦN NHẬT QUANG

Tp. Hồ Chí Minh, ngày 10 tháng 7 năm 2024

ĐH SƯ PHẠM KỸ THUẬT TP.HCM
KHOA CNTT
****

XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh Phúc
****

PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

Họ và tên Sinh viên 1 : Nguyễn Đức Linh MSSV 1: 20133007
Họ và tên Sinh viên 2 : Phan Thanh Tín MSSV 2: 20133095
Ngành: Công nghệ Thông tin

Tên đề tài: ỨNG DỤNG HỌC MÁY DỰ ĐOÁN SỐ LƯỢNG SẢN PHẨM BÁN RA ĐƯỢC TRÊN TRANG THƯƠNG MẠI ĐIỆN TỬ

Họ và tên Giáo viên hướng dẫn: TS. Trần Nhật Quang

NHẬN XÉT

1. Về nội dung đề tài & khối lượng thực hiện:

.....

.....

.....

2. Ưu điểm:

.....

.....

3. Khuyết điểm:

.....

.....

4. Đề nghị cho bảo vệ hay không ?
5. Đánh giá loại :
6. Điểm :

Tp. *Hồ Chí Minh, ngày tháng năm 2024*
Giáo viên hướng dẫn
(Ký & ghi rõ họ tên)

ĐH SƯ PHẠM KỸ THUẬT TP.HCM
KHOA CNTT
****

XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh Phúc
****

PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

Họ và tên Sinh viên 1 : Nguyễn Đức Linh MSSV 1: 20133007
Họ và tên Sinh viên 2 : Phan Thanh Tín MSSV 2: 20133095
Ngành: Công nghệ Thông tin

Tên đề tài: ỨNG DỤNG HỌC MÁY DỰ ĐOÁN SỐ LƯỢNG SẢN PHẨM BÁN RA ĐƯỢC TRÊN TRANG THƯƠNG MẠI ĐIỆN TỬ

Họ và tên Giáo viên phản biện: PGS.TS. Hoàng Văn Dũng

NHẬN XÉT

1. Về nội dung đề tài & khối lượng thực hiện:

.....

.....

.....

2. Ưu điểm:

.....

.....

3. Khuyết điểm:

.....

.....

4. Đề nghị cho bảo vệ hay không ?
5. Đánh giá loại :
6. Điểm :

Tp. *Hồ Chí Minh, ngày tháng năm 2024*
Giáo viên phản biện
(Ký & ghi rõ họ tên)

LỜI CAM ĐOAN

Đồ án "ỨNG DỤNG HỌC MÁY DỰ ĐOÁN SỐ LƯỢNG SẢN PHẨM BÁN RA ĐƯỢC TRÊN TRANG THƯƠNG MẠI ĐIỆN TỬ" được nhóm tác giả Nguyễn Đức Linh và Phan Thanh Tín thực hiện với mục đích học tập, nghiên cứu nhằm ôn tập và hệ thống hóa kiến thức đã học áp dụng vào bài toán thực tế.

Chúng tôi xin cam đoan, những số liệu trong luận văn là hoàn toàn trung thực, kết quả nghiên cứu do chúng tôi thực hiện dưới sự hướng dẫn của TS. Trần Nhật Quang. Những tài liệu tham khảo được sử dụng trong đồ án đã được trích dẫn và nêu rõ tại mục Tài liệu tham khảo.

Nếu những lời cam đoan trên của chúng tôi không chính xác, chúng tôi xin chịu hoàn toàn trách nhiệm và mọi hình thức kỷ luật từ khoa và nhà trường.

LỜI CẢM ƠN

Trong suốt quá trình học tập và thực hiện đề tài "ỨNG DỤNG HỌC MÁY DỰ ĐOÁN SỐ LƯỢNG SẢN PHẨM BÁN RA ĐƯỢC TRÊN TRANG THƯƠNG MẠI ĐIỆN TỬ" ngoài nỗ lực và cố gắng của bản thân, chúng tôi đã nhận được sự hướng dẫn tận tình và quý giá từ quý Thầy Cô, cùng với sự động viên từ gia đình và bạn bè. Bằng tất cả sự kính trọng, chúng tôi xin gửi lời cảm ơn sâu sắc đến Ban Giám hiệu trường Đại học Sư phạm Kỹ thuật Thành phố Hồ Chí Minh đã tạo mọi điều kiện thuận lợi giúp chúng tôi hoàn thành luận văn. Xin cảm ơn quý Thầy Cô khoa Công nghệ Thông tin đã nhiệt tình giảng dạy, giải đáp và giúp đỡ chúng tôi trên con đường học tập. Đặc biệt, xin chân thành cảm ơn TS. Trần Nhật Quang, người thầy kính yêu đã hết lòng hướng dẫn, động viên và đồng hành cùng chúng tôi trong suốt quá trình thực hiện đề tài.

Chúng tôi đã không ngừng cố gắng và nỗ lực, tuy nhiên với thời gian và kinh nghiệm nghiên cứu, ứng dụng vào dữ liệu thực tế còn hạn chế nên không thể tránh khỏi những thiếu sót. Kính mong nhận được sự góp ý của quý Thầy Cô để chúng tôi ngày một hoàn thiện hơn.

Xin chân thành cảm ơn!

ĐỀ CƯƠNG LUẬN VĂN TỐT NGHIỆP

Họ và tên Sinh viên 1 : Nguyễn Đức Linh MSSV 1: 20133007

Họ và tên Sinh viên 2 : Phan Thanh Tín MSSV 2: 20133095

Thời gian làm bài luận: Từ ngày 24/02/2024 đến ngày 10/07/2024

Chuyên ngành : Kỹ thuật dữ liệu

Tên luận văn: ỨNG DỤNG HỌC MÁY DỰ ĐOÁN SỐ LƯỢNG SẢN PHẨM BÁN RA ĐƯỢC TRÊN TRANG THƯƠNG MẠI ĐIỆN TỬ

Họ và tên Giáo viên hướng dẫn: TS. Trần Nhật Quang

Nhiệm vụ của luận văn:

1. Vận dụng kiến thức đã học về thu thập, xử lý và phân tích dữ liệu trên tập dữ liệu thực tế.
2. Xây dựng và so sánh hiệu quả dự đoán số lượng sản phẩm bán ra trên trang thương mại điện tử sử dụng các thuật toán học máy gồm hồi quy tuyến tính, cây quyết định, rừng ngẫu nhiên, XGBoost, KNN, LSTM.
3. Nghiên cứu phương pháp tích hợp phản hồi của khách hàng về sản phẩm vào mô hình để cải thiện hiệu quả dự đoán.

MỤC LỤC

2.2 Chương 2: Các thành phần và phương pháp tối ưu hóa mô hình học máy

2.3. Chương 3: Giới thiệu về tập dữ liệu

2.4. Chương 4: Kết quả thực nghiệm

3. PHẦN KẾT LUẬN

3.1. Kết quả đạt được

3.2. Hạn chế của đề tài

3.3. Hướng phát triển trong tương lai

TÀI LIỆU THAM KHẢO

KẾ HOẠCH THỰC HIỆN

STT	Thời gian	Công việc
1	24/02/2024 - 09/03/2024	Nghiên cứu về đề tài, khái quát vấn đề tổng thể Tham khảo các nghiên cứu trước đây Lên bố cục sơ bộ và kế hoạch thực hiện
2	10/03/2024 - 10/04/2024	Thu thập dữ liệu bán hàng từ Shopee Nghiên cứu lý thuyết các thuật toán được sử dụng phổ biến như hồi quy tuyến tính, cây quyết định, rừng ngẫu nhiên, XGBoost, KNN, LSTM. Tìm hiểu về phân loại cảm xúc văn bản để áp dụng tạo thêm biến cho mô hình
3	11/4/2024 - 24/04/2024	Tổng hợp dữ liệu đã thu thập, làm sạch và khai phá dữ liệu qua các biểu đồ và bảng thống kê Biến đổi dữ liệu thành định dạng phù hợp cho từng mô hình
4	25/04/2024 - 08/05/2024	Cài đặt mô hình ban đầu cho các mô hình đã tìm hiểu Chạy mô hình pre-train PhoBert để phân tích dữ liệu cảm xúc văn bản
5	09/05/2024 - 22/05/2024	Biến đổi các biến hiện có (tạo biến tổng hợp, số hóa các biến phân loại dạng chuỗi) Kết hợp kết quả phân tích PhoBert để tạo biến mới Đánh giá độ phù hợp của biến với mô hình

6	23/05/2024 - 06/06/2024	Sử dụng phương pháp cross-validation và grid-search để tìm thông số tối ưu cho mô hình Đánh giá hiệu suất các mô hình và so sánh kết quả
7	07/06/2024 - 15/06/2024	Tổng hợp nội dung báo cáo. Tổ chức lại mã nguồn.
8	16/06/2024 - 30/6/2024	Kiểm tra lần cuối, hoàn thành báo cáo. Trình bày slide thuyết trình.
9	01/07/2024 - 10/07/2024	Trao đổi với thầy hướng dẫn về nội dung cần trình bày ở buổi bảo vệ khóa luận tốt nghiệp.

Ý kiến của giáo viên hướng dẫn

(ký và ghi rõ họ tên)

TS. Trần Nhật Quang

Ngày tháng năm 2024

Người viết đề cương

Nguyễn Đức Linh

Phan Thanh Tín

MỤC LỤC

DANH MỤC TỪ VIẾT TẮT

Ký hiệu chữ viết tắt	Ký hiệu đầy đủ	Ý nghĩa
BERT	Bidirectional Encoder Representations from Transformers	Đại diện bộ mã hóa hai chiều từ máy biến áp
KNN	K-Nearest Neighbors	Giải thuật k hàng xóm gần nhất
LSTM	Long Short-Term Memory	Bộ nhớ dài-ngắn hạn
ML	Machine learning	Học máy
MLM	Masked Language Model	Mô hình ngôn ngữ đeo mặt nạ
NSP	Next Sentence Prediction	Dự đoán câu tiếp theo
RNN	Recurrent Neural Network	Mạng nơ ron truy hồi
RoBERTa	A Robustly Optimized BERT Pretraining Approach	Là một phương pháp đào tạo trước dựa trên kiến trúc BERT
XGBoost	eXtreme Gradient Boosting	Tăng cường độ dốc cực đại
XLM	Cross-lingual Language Models	Mô hình ngôn ngữ đa ngôn ngữ

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG

PHẦN 1: MỞ ĐẦU

1. Tính cấp thiết của đề tài

Trong thời đại công nghệ số hiện nay, bán hàng trên các sàn thương mại điện tử đã trở thành xu thế và đem lại nguồn doanh số quan trọng cho các doanh nghiệp. Đặc biệt, doanh thu thương mại điện tử tại Việt Nam năm 2023 đã đạt 20,5 tỷ USD, tăng trưởng 25% so với năm 2022. Với tốc độ phát triển này, Việt Nam đã vươn lên vị trí top 10 quốc gia có ngành TMĐT tăng trưởng nhanh nhất thế giới [1]. Việc tạo ra các dự báo về số lượng sản phẩm bán được trên sàn thương mại điện tử ngày càng được quan tâm. Những mô hình dự báo chính xác và đáng tin cậy tạo điều kiện thuận lợi cho việc lập kế hoạch sản xuất, quản lý kho, lập kế hoạch khuyến mãi và đưa ra giá cả cạnh tranh, dự báo doanh thu,… nhằm tối ưu hóa nguồn vốn và đem lại lợi thế cạnh tranh cho sản phẩm của doanh nghiệp.

Số lượng sản phẩm bán ra trên sàn thương mại điện tử có nhiều biến động, nguyên nhân chủ yếu do các đợt khuyến mãi từ sàn thương mại điện tử, hành vi của đối thủ cạnh tranh,… Đi kèm với đó là những khó khăn về thu thập dữ liệu như là dữ liệu không ổn định, dữ liệu trên toàn sàn rất lớn, khó thu thập đầy đủ cũng như là chính sách bảo mật dữ liệu của sàn thương mại điện tử, chính sách tính số lượng bán và phạt trừ vi phạm của sàn,… Do đó, dự đoán số lượng sản phẩm bán được gặp nhiều khó khăn thách thức.

Mặc dù dự đoán nhu cầu thị trường đã rất phổ biến và chứng minh nhận được sự quan tâm lớn qua các cuộc thi trên Kaggle, nền tảng cung cấp dữ liệu và các cuộc thi về học máy. Tuy nhiên, các bài nghiên cứu và hướng dẫn cho thị trường Việt Nam hay những quốc gia bắt đầu chuyển giao công nghệ chưa có nhiều. Đề tài đặt biệt nhắm đến việc tìm hiểu, đánh giá kết quả dự đoán trong quãng thời gian ngắn hơn so với các tập dữ liệu mẫu trong các cuộc thi để các doanh nghiệp trong nước có thể tham khảo và triển khai ứng dụng.

2. Mục đích của đề tài

Mục đích chính của đề tài là xây dựng và đánh giá các thuật toán như hồi quy tuyến tính, cây quyết định, rừng ngẫu nhiên, XGBoost, KNN, LSTM vào mô hình dự đoán số lượng sản phẩm bán được. Ngoài ra, chúng tôi còn muốn kết hợp thêm kết quả

phân tích cảm xúc bình luận của khách hàng về sản phẩm để cải thiện hiệu suất của mô hình.

Các nhiệm vụ chính:

- Hệ thống lại kiến thức đã học và tìm hiểu thêm những tài liệu mới để xây dựng cơ sở lý thuyết vững chắc trước khi bắt đầu triển khai luận án.
- Vận dụng kiến thức về thu thập, xử lý, phân tích để đưa ra cái nhìn tổng quan về dữ liệu. Thực nghiệm và đánh giá các đặc trưng sẵn có cũng như tạo thêm các đặc trưng mới trên dữ liệu thực tế để tìm ra lựa chọn phù hợp.
- Xây dựng và thử nghiệm các mô hình học máy và so sánh với kết quả khi kết hợp thêm phân tích cảm xúc bình luận về sản phẩm.
- Triển khai huấn luyện và đánh giá hiệu suất các mô hình và so sánh để tìm ra mô hình tối ưu cho bài toán.

3. Cách tiếp cận và phương pháp nghiên cứu

Đối tượng nghiên cứu:

- Dữ liệu biến động về sản phẩm trên sàn thương mại điện tử.
- Dữ liệu về bình luận tương ứng với các sản phẩm.
- Cơ sở lý thuyết về các thuật toán dự đoán và phân tích cảm xúc.
- Các đặc trưng liên quan đến số lượng sản phẩm bán ra và các chỉ số đánh giá mô hình.

Phạm vi nghiên cứu:

- Top 13 nhà bán có doanh số cao nhất lĩnh vực làm đẹp trên sàn thương mại điện tử Shopee.
- Thời gian thu thập dữ liệu trong vòng 1 tháng từ 10/3/2024 đến ngày 09/4/2024.
- Các thuật toán dự đoán: hồi quy tuyến tính, cây quyết định, rừng ngẫu nhiên, XGBoost, KNN, LSTM.
- Phân loại cảm xúc bình luận bằng mô hình ngôn ngữ lớn, cụ thể là PhoBERT.

Phương pháp nghiên cứu:

- Thu thập dữ liệu về sản phẩm, bán hàng và phản hồi của khách hàng.

- Tìm hiểu các thuật toán và các nghiên cứu liên quan.
- Xây dựng mô hình, tinh chỉnh và đánh giá.

4. Phân tích những công trình có liên quan

Vì tính cấp thiết của đề tài, rất nhiều nghiên cứu dự đoán số lượng sản phẩm bán được chuỗi thời gian đã được nghiên cứu. Trong cuộc thi MliP_M5 [2] được tổ chức trên Kaggle, các thuật toán đạt hiệu suất cao nhất là RNN và các biến thể của nó, XGBoost và LightGBM. Tuy nhiên, các phương pháp được sử dụng phổ biến hiện nay như công trình nghiên cứu của Y. Liu [3], M. Li, S. Ji và G. Liu [4], hay K. Bandara, P. Shi, C. Bergmeir, H. Hewamalage, Q. Tran, B. Seaman [5], thường sử dụng dữ liệu chuỗi thời gian lên tới vài năm. Mặc dù kết quả đạt được là rất ấn tượng, tuy nhiên, trong nhu cầu thực tế tại các doanh nghiệp Việt Nam, việc bắt đầu thu thập và chờ đợi hàng năm trời để nhận về kết quả sẽ là rào cản rất lớn để doanh nghiệp ứng dụng học máy vào kinh doanh. Hơn nữa, việc huấn luyện từng mô hình cho từng sản phẩm sẽ gây ra tốn kém tài nguyên và thời gian hơn là sử dụng mô hình cho từng nhóm sản phẩm.

Nhóm tác giả thực hiện đề tài "ỨNG DỤNG HỌC MÁY DỰ ĐOÁN SỐ LƯỢNG SẢN PHẨM BÁN RA ĐƯỢC TRÊN TRANG THƯƠNG MẠI ĐIỆN TỬ" trên tập dữ liệu với nhiều sản phẩm trong cùng ngành hàng trong một tháng để so sánh và đánh giá phương pháp phù hợp cho các doanh nghiệp mới bắt đầu khai thác thông tin từ dữ liệu. Ngoài ra, chúng tôi còn cố gắng tích hợp thêm thông tin từ phản hồi của khách hàng vào mô hình dự đoán bằng cách sử dụng mô hình ngôn ngữ lớn để phân tích cảm xúc bình luận của khách hàng về sản phẩm. Trong hàng loạt mô hình ngôn ngữ lớn, nhóm chúng tôi đã lựa chọn PhoBERT cho bài toán của mình dựa trên kết quả xử lý ngôn ngữ tiếng việt ấn tượng được công bố tại bài báo "PhoBERT: Pre-trained language models for Vietnamese" [6].

5. Kết quả dự kiến đạt được

Các kết quả dự kiến đạt được là

- Thu thập, phân tích và có hiểu biết thêm về ngành thương mại điện tử, đặc biệt là nền tảng Shopee.
- Nắm vững các kiến thức lý thuyết về các mô hình được sử dụng để dự đoán.

- Hiểu và có thể vận dụng mô hình ngôn ngữ lớn phục vụ cho tác vụ phân tích cảm xúc.
- Xây dựng mô hình và đánh giá kết quả đạt được. Rút ra được kinh nghiệm để áp dụng mô hình vào doanh nghiệp.

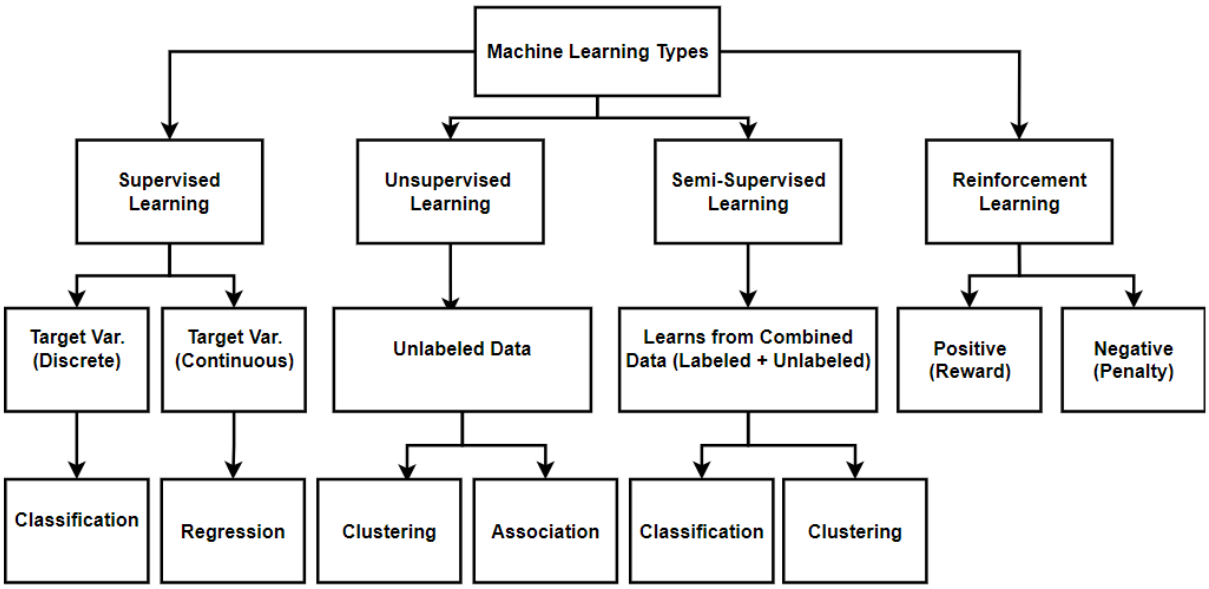
PHẦN 2: NỘI DUNG

1. CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

1.1. Giới thiệu về học máy

Học máy (ML) là một nhánh của trí tuệ nhân tạo (AI) cho phép các máy tính học hỏi từ dữ liệu mà không cần được lập trình cụ thể cho từng nhiệm vụ. Nó làm điều này bằng cách sử dụng các thuật toán để phân tích dữ liệu đầu vào để đưa ra dự đoán cho các giá trị đầu ra mới [7].

Các thuật toán học máy chủ yếu được chia thành bốn loại: học có giám sát (Supervised Learning), học không giám sát (Unsupervised Learning), học bán giám sát (Semi-Supervised Learning) và học tăng cường (Reinforcement Learning). Hình 1 minh họa các loại kỹ thuật học máy này.



Hình 1: Các loại kỹ thuật học máy khác nhau.

(Được vẽ lại dựa trên Sarker, I. H.) **[8]**

1.1.1. Học có giám sát (Supervised Learning)

Học có giám sát là loại phổ biến nhất trong học máy. Ở loại này, dữ liệu huấn luyện đã được gắn nhãn. Nhiệm vụ của học máy là tìm ra một hàm có nhiệm vụ ánh xạ (xấp xỉ) mỗi phần tử đầu vào sang một phần tử đầu ra tương ứng.

Các nhiệm vụ phổ biến nhất được thực hiện từ một tập hợp đầu vào nhất định là Classification – phân loại, và Regression - hồi quy. Ví dụ, việc phân loại cảm xúc một đánh giá sản phẩm là một ví dụ về học có giám sát.

1.1.2. Học không giám sát (Unsupervised Learning)

Học không giám sát là loại học máy được đào tạo trên tập dữ liệu không được gắn nhãn. Loại học máy này không cần sự can thiệp của con người, nghĩa là một quy trình dựa hoàn toàn trên dữ liệu. Các nhiệm vụ học không giám sát phổ biến nhất là phân cụm, giảm chiều, tìm quy tắc kết hợp, phát hiện dị thường, …

1.1.3. Học bán giám sát (Semi-Supervised Learning)

Học bán giám sát là sự kết hợp của học có giám sát và không giám sát, nó hoạt động trên tập dữ liệu bao gồm cả dữ liệu có nhãn và dữ liệu không có nhãn. Trong một số trường hợp, dữ liệu được gắn nhãn thường ít hơn rất nhiều so với dữ liệu không gắn nhãn. Một số lĩnh vực bán giám sát được sử dụng bao gồm phát hiện gian lận, dán nhãn dữ liệu,…

1.1.4. Học tăng cường (Reinforcement Learning)

Học tăng cường là bài toán mà trong đó hệ thống tự đánh giá hành vi tối ưu trong ngữ cảnh hoặc môi trường cụ thể để cải thiện hiệu quả của chính nó. Loại học này dựa trên phần thưởng hoặc hình phạt, và mục tiêu cuối cùng của nó là sử dụng những hiểu biết thu được từ các nhà hoạt động môi trường để hành động nhằm tăng phần thưởng hoặc giảm thiểu rủi ro.

Nó là một công cụ mạnh mẽ để đào tạo các mô hình trí tuệ nhân tạo có thể giúp tăng cường tự động hóa hoặc tối ưu hóa hiệu quả hoạt động của các hệ thống phức tạp như robot học, nhiệm vụ lái xe tự động, hậu cần sản xuất và chuỗi cung ứng,…

1.1.5. Ứng dụng của học máy trong dự đoán kinh doanh

Dự báo kinh doanh là quá trình đưa ra các dự đoán thông tin về các chỉ số kinh doanh nhất định, chẳng hạn như tăng trưởng doanh số, dự đoán sản phẩm bán được hoặc dự báo cho nền kinh tế nói chung. Nó liên quan đến việc sử dụng dữ liệu lịch sử, xu

hướng thị trường và kỹ thuật thống kê, học máy để dự đoán kết quả kinh doanh trong tương lai [9].

Các doanh nghiệp hiện đại ngày nay muốn phát triển thì đều đi theo hướng số hóa, điều này bao gồm cả việc sử dụng ML cho các tác vụ dự đoán kinh doanh. Một số lợi ích của học máy trong kinh doanh:

- Ra quyết định dựa trên dữ liệu. Học máy cho phép các doanh nghiệp đưa ra quyết định chính xác và sáng suốt hơn bằng cách phân tích một lượng lớn dữ liệu một cách nhanh chóng và hiệu quả và cung cấp thông tin chi tiết dựa trên nó.
- Nâng cao trải nghiệm khách hàng. Bằng cách cá nhân hóa các dịch vụ và đề xuất, học máy cải thiện đáng kể sự hài lòng và tương tác của khách hàng. Điều này dẫn đến kết quả hoạt động tốt hơn và nâng cao lòng trung thành với thương hiệu.
- Cải thiện hiệu quả hoạt động. Học máy cho doanh nghiệp tự động hóa các tác vụ thường ngày, tối ưu hóa quy trình và giảm lỗi, dẫn đến tăng năng suất và tiết kiệm chi phí. Các công ty hàng đầu báo cáo hiệu quả quy trình tăng 30% khi triển khai ML [10].
- Quản lý rủi ro tốt hơn. ML trong kinh doanh cung cấp các công cụ tiên tiến để xác định, đánh giá và giảm thiểu các rủi ro khác nhau dựa trên dữ liệu lịch sử và thực tế, tăng cường sự ổn định tài chính và hoạt động. Thống kê cho thấy việc triển khai ML có thể giúp giảm tới 20% chi phí quản lý rủi ro [11].
- Tăng cường phân tích xu hướng thị trường. Học máy giúp dự đoán chính xác xu hướng thị trường, cho phép các doanh nghiệp chủ động điều chỉnh chiến lược.
- Cải thiện bảo trì dự đoán. Các mô hình dựa trên ML có thể dự đoán lỗi thiết bị và nhu cầu bảo trì, giảm thời gian chết và chi phí vận hành trong sản xuất và các ngành công nghiệp khác.
- Học máy cho các công ty có sức mạnh biến đổi. Với sự gia tăng về số lượng và chất lượng dữ liệu kinh doanh, việc sử dụng các thuật toán thông minh có lẽ là cách duy nhất để đạt được tầm cao mới và đi trước các đối thủ cạnh tranh.

1.2. Thuật toán Hồi quy tuyến tính (Linear Regression)

1.2.1. Giới thiệu

Linear Regression là một thuật toán học có giám sát (supervised learning) để xác định mối quan hệ tuyến tính giữa một hoặc nhiều biến đầu vào (biến độc lập) với biến đầu ra (biến phụ thuộc).

Ví dụ:

- Biến độc lập: Lịch sử giá, lịch sử bán hàng, xu hướng thị trường, mức độ đánh giá của khách hàng về sản phẩm,…
- Biến phụ thuộc: Số lượng sản phẩm bán được.

Linear Regression được gọi là là hồi quy tuyến tính đơn biến (simple linear regression) khi chỉ có một biến độc lập, hoặc hồi quy tuyến tính đa biến (multiple linear regression) khi có nhiều biến độc lập. Thuật toán Linear Regression sẽ tìm ra một phương trình tuyến tính để biểu diễn mối quan hệ này.

1.2.2. Phân tích toán học

1.2.2.1. Dạng phương trình của Linear Regression

Trong trường hợp tổng quát, mô hình hồi quy tuyến tính đa biến (multiple linear regression) được biểu diễn dưới dạng toán học:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k + \varepsilon \quad (1)$$

Trong đó:

- y là biến phụ thuộc.
- β_0 là hệ số tự do.
- $\beta_1, \beta_2, \dots, \beta_k$ là hệ số hồi quy (coefficients) tương ứng với các biến độc lập $x_1, x_2, \dots, x_k$.
- ε là thành phần sai số.

Một mẫu (sample) trong (1) có thể biểu diễn là:

$$y^{(i)} = \beta_0 + \beta_1 x_1^{(i)} + \beta_2 x_2^{(i)} + \cdots + \beta_k x_k^{(i)} + \varepsilon^{(i)}, \quad 1 \le i \le n \quad (2)$$

Để thuận tiện hơn trong tính toán, chúng ta đặt

$$\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{bmatrix}, \boldsymbol{X} = \begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \cdots & x_k^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} & & x_k^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(n)} & x_2^{(n)} & \cdots & x_k^{(n)} \end{bmatrix}, \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon^{(1)} \\ \varepsilon^{(2)} \\ \vdots \\ \varepsilon^{(n)} \end{bmatrix}, p = k + 1$$

Khi đó, phương trình hồi quy tuyến tính đa biến có thể được biểu diễn dạng ma trận là:

$$\underbrace{\boldsymbol{y}}_{n\times 1} = \underbrace{\boldsymbol{X}}_{n\times p} \cdot \underbrace{\boldsymbol{\beta}}_{p\times 1} + \underbrace{\boldsymbol{\varepsilon}}_{n\times 1}$$

1.2.2.2. Hàm mất mát

Hàm mất mát (loss function) là hàm biểu thị độ sai lệch giữa giá trị dự đoán và giá trị thực tế. Hàm mất mát được sử dụng phổ biến nhất trong hồi quy tuyến tính là hàm trung bình độ lỗi bình phương (Mean Squared Error – MSE). Với $\hat{y}_i$ là giá trị dự đoán của y_i, phương trình hàm mất mát được biểu diễn dạng

$$L(\beta) = MSE(\beta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \frac{1}{n}\sum_{i=1}^{n}\varepsilon_i^2 = \boldsymbol{\varepsilon}^T\boldsymbol{\varepsilon} = (\boldsymbol{y} - \boldsymbol{X\beta})^T(\boldsymbol{y} - \boldsymbol{X\beta})$$

Sau khi có hàm mất mát, mục tiêu của bài toán hồi quy tuyến tính trở thành tìm hệ số $\boldsymbol{\beta}$ sao cho hàm mất mát nhỏ nhất. Để tìm tham số $\boldsymbol{\beta}$, thông thường ta sử dụng phương pháp đạo hàm hoặc phương pháp Gradient Descent.

1.3. Thuật toán Decision Tree

1.3.1. Giới thiệu

Cây quyết định (Decision Tree) là một thuật toán học có giám sát có thể áp dụng cho cả hai nhiệm vụ phân loại và hồi quy. Ý tưởng chính của thuật toán này là tạo ra một cây phân cấp. Trong đó, mỗi nút trên cây đại diện cho một quyết định, trong khi mỗi lá là một giá trị dự đoán. Ưu điểm lớn nhất của cây quyết định là vì nó tương tự với cách chúng ta đưa ra quyết định, có thể biểu diễn trực quan nên dễ hiểu. Trong phạm vi nghiên cứu của đề tài, chúng ta sẽ tập trung tìm hiểu về cây quyết định cho bài toán hồi quy hay còn gọi là cây hồi quy.

1.3.2. Các thuật ngữ trong cây quyết định hồi quy

- Nút gốc (Root Node): là nút đầu tiên của cây và là nút duy nhất không có nút cha. Nút này chứa toàn bộ dữ liệu và khởi tạo quá trình phân cấp.

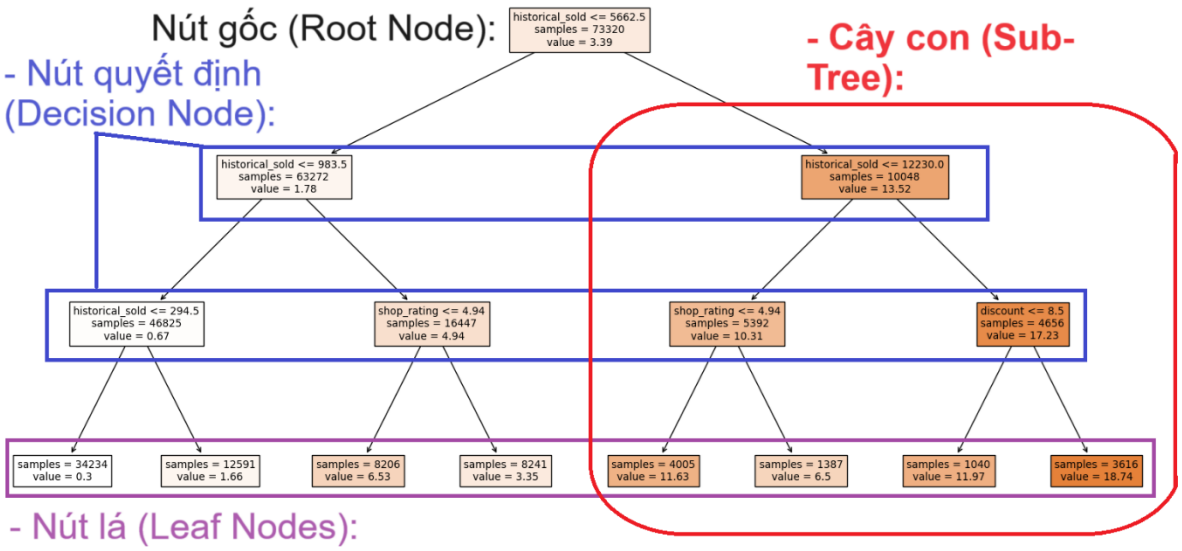
- Nút quyết định (Decision Node): Nút này đại diện cho các quyết định hoặc điều kiện trung gian trong cây.

- Nút lá (Leaf Nodes): Các nút không thể phân tách thêm, là kết quả cuối cùng của cây. Các nút lá còn được gọi là nút cuối cùng.

- Cây con (Sub-Tree): Tương tự như phần con của biểu đồ được gọi là biểu đồ con, phần con của cây quyết định được gọi là cây con. Nó đại diện cho một phần cụ thể của cây quyết định.

- Cắt tỉa (Pruning): Quá trình loại bỏ hoặc cắt giảm các nút cụ thể trong cây quyết định để ngăn chặn quá khớp (overfitting) và đơn giản hóa mô hình.

- Nút cha và con (Parent and Child Node): Một nút được chia thành các nút con được gọi là nút cha và các nút con xuất hiện từ nó được gọi là nút con. Nút cha thể hiện một quyết định hoặc điều kiện, trong khi các nút con thể hiện các kết quả tiềm năng hoặc các quyết định tiếp theo dựa trên điều kiện đó.



Hình 2: Minh họa các thuật ngữ cây quyết định

1.3.3. Xây dựng cây quyết định hồi quy

Có nhiều thuật toán được phát triển cho cây quyết định, nhưng phổ biến nhất phải nhắc đến CART (Classification And Regression Tree) có thể xử lý cho cả dữ liệu phân

loại và hồi quy, và là ý tưởng chính được sử dụng xây dựng cây hồi quy trong thư viện scikit-learn [12].

Ý tưởng xây dựng cây hồi quy dựa trên CART là thực hiện quá trình đệ quy phân chia dữ liệu thành 2 tập con nhỏ hơn, các bước chi tiết như sau:

- **Bước 1**: Xác định điểm phân chia t_k của đặc trưng k

Đề tìm điểm phân chia tối ưu, chúng ta dựa vào giá trị của hàm mất mát. Hàm mất mát phổ biến nhất là Mean Squared Error (MSE). Để xác định được t_k và k ta duyệt qua lần lượt các đặc trưng trong dữ liệu, lần lượt thử t_i là giá trị trung bình giữa 2 giá trị liên tiếp. Phân dữ liệu thành 2 phần lớn hơn và bé hơn dựa trên t_i và tính giá trị hàm mất mát. Vị trí điểm phân chia t_k của đặc trưng k có hàm mất mát nhận giá trị thấp nhất là vị trí được chọn.

- **Bước 2**: Sau khi xác định được điểm phân chia t_k của đặc trưng k. Chia tập dữ liệu thành 2 tập con và tiếp tục lặp lại cho tập con mới tạo ra cho đến khi gặp điều kiện dừng.

Điều kiện dừng là một trong các điều kiện sau (Tên các biến được lấy theo tên trong thư viện scikit-learn):

- Độ sâu của cây đạt đến giới hạn max_depth
- Không thể tìm được điểm phân chia t_k nào có giá trị hàm mất mát tốt hơn
- Nếu nút đó có số phần tử nhỏ hơn một ngưỡng min_samples_split
- Nếu số lượng phần tử trong nút phân chia nhỏ hơn ngưỡng min_samples_leaf
- Nếu tổng số nốt lá đã đạt đến giới hạn max_leaf_nodes

1.3.4. Cắt tỉa (pruning)

Để hạn chế tình trạng quá khớp và làm đơn giản hóa mô hình, ta có thể sử dụng kỹ thuật cắt tỉa (pruning). Cắt tỉa hoạt động bằng cách loại bỏ các nút không cần thiết hoặc ít thông tin, giúp mô hình trở nên tổng quát và hiệu quả hơn. Một phương pháp phổ biến trong cắt tỉa là giảm thiểu sai số cắt tỉa (reducing error pruning).

Cụ thể, quy trình thực hiện như sau và chỉ dừng lại khi độ chính xác trên tập kiểm định không còn tiếp tục được cải thiện.

Bước 1: Huấn luyện mô hình trên tập huấn luyện (training set).

Bước 2: Lần lượt thử loại bỏ các nút lá ở phía dưới và chuyển các nút cha của các nút lá đó thành nút lá.

Bước 3: Kiểm tra với tập dữ liệu kiểm định: Sau mỗi lần loại bỏ một nút lá, kiểm tra kết quả dự đoán của cây trên tập dữ liệu kiểm định (validation set).

1.4. Thuật toán Random Forest

1.4.1. Giới thiệu

Rừng ngẫu nhiên (Random Forest) được phát triển bởi Leo Breiman vào năm 2001 [13]. Ý tưởng chính của thuật toán này là xây dựng một tập hợp nhiều cây quyết định riêng lẻ và tổng hợp kết quả bằng phương pháp bỏ phiếu (voting). Vì thế, tương tự như cây quyết định, rừng ngẫu nhiên cũng là thuật toán học có giám sát và có thể áp dụng cho cả bài toán phân loại và hồi quy. Trong phạm vi nghiên cứu của đề tài, chúng ta sẽ tập trung vào phần hồi quy của thuật toán rừng ngẫu nhiên.

1.4.2. Cách thức hoạt động

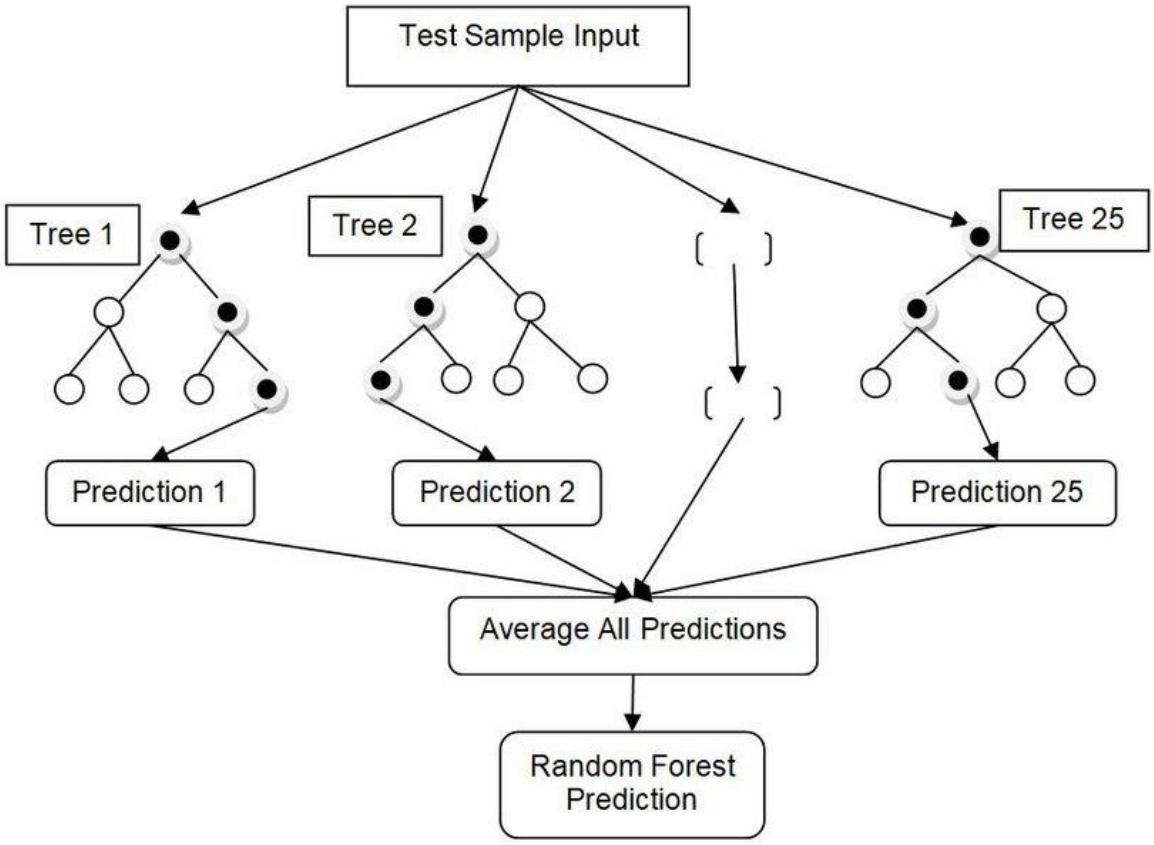
Rừng ngẫu nhiên hồi quy hoạt động theo nguyên tắc sau:

Bước 1: Tạo tập dữ liệu con ngẫu nhiên: Từ tập dữ liệu ban đầu tạo ra các tập dữ liệu con bằng phương pháp lấy mẫu có hoàn lại. Mỗi tập dữ liệu con này được sử dụng để đào tạo một cây quyết định riêng biệt.

Bước 2: Chọn ngẫu nhiên tập đặc trưng (feature): Chọn ngẫu nhiên k thuộc tính từ n thuộc tính của tập dữ liệu để huấn luyện (k<n).

Bước 3: Xây dựng các cây dựng trên các tập dữ liệu con và tập đặc trưng tương ứng.

Bước 4: Tổng hợp kết quả bằng cách tính trung bình dự đoán của các cây đã xây dựng.



Hình 3: Minh họa thuật toán Rừng ngẫu nhiên cho bài toán hồi quy

(Nguồn: Pangarkar, D. J., Sharma, R., Sharma, A., & Sharma, M., 2020) **[14]**

1.4.3. Các tham số trong rừng ngẫu nhiên hồi quy

- Số lượng cây (n_estimators): Là số lượng cây quyết định trong rừng, thông số này cao thường cho kết quả chính xác hơn nhưng khiến cho mô hình tốn nhiều thời gian tính toán hơn.
- Số lượng đặc trưng cho mỗi tập đặc trưng (max_features): Số lượng đặc trưng tối đa được xem xét khi tách nhánh cũng chính là số k trong bước 2 được mô tả trong phần 2.4.2. Thông thường người ta sẽ chọn là căn bậc hai hoặc log cơ số hai của tổng số lượng đặc trưng.
- Độ sâu tối đa của mỗi cây (max_depth): Giới hạn độ sâu của mỗi cây quyết định.

1.5. Thuật toán XGBoost

1.5.1. Giới thiệu

XGBoost (Extreme Gradient Boosting) là một thuật toán mạnh mẽ và được sử dụng nhiều trong các cuộc thi trên Kaggle đạt được nhiều thành tích xuất sắc [15]. Ý tưởng chính của thuật toán XGBoost là xây dựng một tập hợp cây quyết định yếu để cải thiện độ chính xác của mô hình. Vì thế, tương tự như cây quyết định, thuật toán XGBoost cũng vừa có thể áp dụng cho bài toán phân loại và bài toán hồi quy. Trong phạm vi đề tài, ta sẽ xét về tính hồi quy của thuật toán này.

1.5.2. Cách thức hoạt động

Bước 1: Khởi tạo giá trị dự đoán ban đầu f_0 bằng cách lấy trung bình của dữ liệu giá trị mục tiêu Y

$$f_0 = mean(Y)$$

Bước 2: Tính toán độ tương quan (Similarity Score) cho nút gốc

$$\text{Similarity Score} = \frac{[\sum(Y - f_0)]^2}{N + \lambda}$$

- N là số lượng mẫu có trong nút đó
- λ là tham số điều chuẩn (regularization parameter)

Bước 3: Chọn điều kiện phân chia cho nút gốc bằng các lấy trung bình 2 mẫu liên tiếp nhau để tính độ tương quan cho nhánh trái và nhánh phải.

$$Gain = \text{Similarity Score}_{\text{Nhánh trái}} + \text{Similarity Score}_{Nhánh\ phải} - \text{Similarity Score}_{\text{Nút gốc}}$$

Điểm phân chia có Gain lớn nhất là điểm phân chia được chọn cho nút gốc

Bước 4: Phân chia dữ liệu theo điều kiện đã chọn và lặp lại bước 2 và 3 cho tới khi đạt điều kiện:

- Đạt độ sâu tối đa của cây
- Giá trị Gain không còn đủ qua mốc lựa chọn
- Số lượng mẫu mỗi lá tối thiểu

Bước 5: Cập nhật giá trị dự đoán

$$f_t = f_{t-1} + \mu * Output$$

- $Output = \frac{[\sum(Y - f_{t-1})]^2}{N}$
- μ là hệ số học (learning rate)

Bước 6: Lặp lại các bước cho đến khi đạt điều kiện dừng.

1.5.3. Các tham số trong XGBoost

- n_estimator/ntrees: Số lượng cây xây dựng để dự đoán.
- max_depth: Độ sâu tối đa của cây.
- min_child_weight: Số lượng mẫu tối thiểu để tiếp tục phân nhỏ thành các nhánh.
- learning_rate: Tốc độ học của mô hình, chỉ số này nhỏ thì mô hình đào tạo lâu hơn nhưng nếu quá lớn có thể dẫn đến mô hình không tìm được điểm tối ưu.

1.6. Thuật toán K-Nearest Neighbors (KNN)

1.6.1. Giới thiệu

Thuật toán K-Nearest Neighbors (KNN) là một trong những thuật toán học có giám sát (Supervised Learning) đơn giản nhất. KNN có thể được áp dụng cho cả bài toán phân loại và hồi quy. Tuy nhiên, trong phạm vi nghiên cứu này, chúng ta sẽ tập trung vào ứng dụng của KNN trong hồi quy.

Ý tưởng chính của KNN trong hồi quy là dự đoán giá trị của biến đầu ra dựa trên K điểm lân cận gần nhất với dữ liệu đầu vào. Nói cách khác, thuật toán sẽ tìm K điểm dữ liệu trong tập huấn luyện gần nhất với điểm dữ liệu cần dự đoán và sử dụng giá trị trung bình (hoặc một hàm tổng hợp khác) của các điểm lân cận này để dự đoán giá trị đầu ra.

KNN được coi là một thuật toán dựa trên ví dụ (Instance-based) vì nó không xây dựng một mô hình tổng quát như các thuật toán học máy khác. Thay vào đó, nó sử dụng trực tiếp các điểm dữ liệu trong tập huấn luyện để dự đoán kết quả.

1.6.2. Cách thức hoạt động

Bước 1: Xác định tham số K láng giềng gần nhất.

Bước 2: Tính khoảng cách giữa điểm đầu vào với tất cả điểm trong tập huấn luyện. Thường là khoảng cách Euclidean.

Bước 3: Sắp xếp khoảng cách theo thứ tự và chọn ra K điểm gần nhất.

Bước 4: Lấy ra K điểm gần nhất.

Bước 5: Tính giá trị dự đoán. Với K = 1, giá trị dự đoán bằng giá trị của điểm gần với giá trị đầu vào nhất trong tập dữ liệu. Với số K khác, dữ liệu được tính dựa trên trung bình (hoặc trung vị) giá trị đầu ra của các điểm trong tập huấn luyện gần với giá trị đầu vào nhất.

1.6.3. Một số hàm tính khoảng cách trong không gian vector thông dụng

1.6.3.1. Khoảng cách Euclidean

Đây là loại khoảng cách phổ biến nhất trong không gian n chiều. Nó được định nghĩa là căn bậc hai của tổng bình phương các hiệu số giữa các tọa độ tương ứng của hai điểm. Về mặt toán học, nó được biểu diễn như sau:

Nếu $(x_1, x_2, x_3, \dots, x_n)$ và $(y_1, y_2, y_3, \dots, y_n)$ là hai điểm trong không gian n chiều, khoảng cách giữa chúng là

$$d = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

1.6.3.2. Khoảng cách Manhattan

Còn được gọi là khoảng cách L1, đây là tổng các giá trị tuyệt đối của hiệu số giữa các tọa độ tương ứng của hai điểm. Nó được sử dụng trong một số ứng dụng mà khoảng cách được đo bằng số bước cần thiết để di chuyển từ điểm này sang điểm khác.

Nếu $(x_1, x_2, x_3, \dots, x_n)$ và $(y_1, y_2, y_3, \dots, y_n)$ là hai điểm trong không gian n chiều, khoảng cách giữa chúng là

$$d = \sum_{i=1}^{n} |x_i - y_i|$$

1.6.3.3. Khoảng cách Minkowski (p = norm)

Khoảng cách Minkowski có thể coi là dạng tổng quát hóa của cả khoảng cách Euclidean và khoảng cách Manhattan.

Nếu $(x_1, x_2, x_3, \dots, x_n)$ và $(y_1, y_2, y_3, \dots, y_n)$ là hai điểm trong không gian n chiều, khoảng cách giữa chúng là

$$d = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}$$

Trong đó p là số thực dương. Khi p=1, khoảng cách Minkowski trở thành khoảng cách Manhattan. Khi p=2, khoảng cách Minkowski trở thành khoảng cách Euclidean.

1.6.3.4. Khoảng cách Chebyshev

Đây là giá trị tuyệt đối lớn nhất của hiệu số giữa tọa độ tương ứng của hai điểm. Nó được sử dụng trong các trường hợp mà sự khác biệt lớn nhất giữa hai điểm quan trọng hơn so với sự khác biệt trung bình.

Nếu $(x_1, x_2, x_3, \dots, x_n)$ và $(y_1, y_2, y_3, \dots, y_n)$ là hai điểm trong không gian n chiều, khoảng cách giữa chúng là

$$d = \max_{i} (|x_i - y_i|)$$

1.6.4. Đánh trọng số cho các điểm lân cận

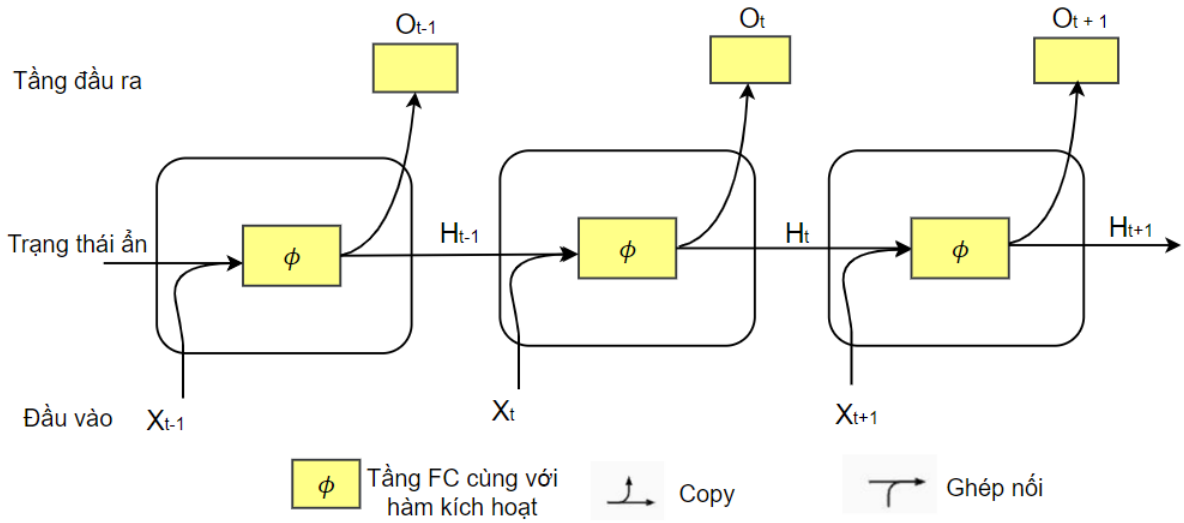
Trong nhiều trường hợp, mức độ ảnh hưởng của các điểm đến giá trị dự đoán không đồng nhất (đặc biệt là các trường hợp có nhiễu). Trong trường hợp đó, để cải thiện thêm độ chính xác cho thuật toán, chúng ta có thể nhân thêm trọng số vào các giá trị ở bước 5. Cách đơn giản nhất để tạo trọng số là chúng ta chọn chính nghịch đảo khoảng cách của điểm cần dự đoán tới điểm huấn luyện tương ứng (Nếu khoảng cách bằng 0 thì chúng ta lấy luôn giá trị đầu ra của điểm huấn luyện).

1.7. Thuật toán Long Short-Term Memory (LSTM)

1.7.1. Mạng neural hồi tiếp (RNN)

RNN được gọi là mạng neural hồi tiếp (recurrent) vì chúng thực hiện cùng một tác vụ cho tất cả các phần tử của một chuỗi với đầu ra phụ thuộc vào cả các phép tính trước đó. Mạng neural hồi tiếp được thiết kế để xử lý thông tin tuần tự, các mạng này sử dụng các biến trạng thái để lưu trữ thông tin trong quá khứ, sau đó dựa vào chúng và các đầu vào hiện tại để xác định các đầu ra hiện tại. Trên lý thuyết, RNN có thể lưu và sử dụng được thông tin của một văn bản rất dài nhờ vào các nút ẩn, tuy nhiên trên thực tế thì nó chỉ có thể nhớ được một vài bước trước đó.

Cấu trúc của 1 mạng RNN:



Hình 4: Cấu trúc của một mạng RNN với 1 trạng thái ẩn

(Được vẽ lại dựa trên Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J, 2021) **[16]**

Trong hình, ý nghĩa các ký hiệu như sau:

- Thông số X_t là đầu vào tại bước t.
- H_t là trạng thái ẩn tại bước t. Nó chính là bộ nhớ của mạng. H_t tính toán dựa trên cả các trạng thái ẩn phía trước và đầu vào tại bước thời gian đó.
- O_t là đầu ra tại bước t.
- Tầng kết nối đầy đủ (Fully Connected layer) cùng với hàm kích hoạt (ϕ) nhận đầu vào hiện tại và trạng thái ẩn từ bước trước đó để tính toán trạng thái ẩn mới. Các hàm kích hoạt phổ biến có thể là hàm sigmoid, tanh hoặc ReLU.

Các bước xử lý trong RNN:

- Tại mỗi bước thời gian t, đầu vào X_t và trạng thái ẩn từ bước trước H_{t-1} được đưa vào tầng kết nối đầy đủ cùng với hàm kích hoạt để tính toán trạng thái ẩn mới H_t.
- Trạng thái ẩn mới H_t được sử dụng để tính toán đầu ra O_t.
- Quá trình này được lặp lại cho mỗi bước thời gian, giúp mạng RNN có khả năng ghi nhớ và xử lý thông tin tuần tự.

Ưu điểm của một kiến trúc RNN:

- Khả năng xử lý đầu vào với bất kì độ dài nào.
- Kích cỡ mô hình không tăng theo kích cỡ đầu vào.
- Quá trình tính toán sử dụng các thông tin cũ.
- Trọng số được chia sẻ trong suốt thời gian.

Nhược điểm của một kiến trúc RNN:

- Tính toán chậm.
- Khó để truy cập các thông tin từ một khoảng thời gian dài trước đây.
- Không thể xem xét bất kì đầu vào sau này nào cho trạng thái hiện tại.

Đối với mạng neural hồi tiếp có trạng thái ẩn, giả sử trong một vòng lặp ta có X_t và H_t là biến ẩn tại bước thời gian t của chuỗi. Biến ẩn H_{t-1} được lưu từ bước thời gian trước đó và dùng thêm một tham số trọng số mới W_{hh} để mô tả việc sử dụng trạng thái ẩn của bước thời gian t-1 trước đó trong bước thời gian hiện tại. Trạng thái ẩn của bước thời gian hiện tại được tính bằng công thức:

$$H_t = \phi\left(X_t W_{xh} + H_{t-1} W_{hh} + b_h\right)$$

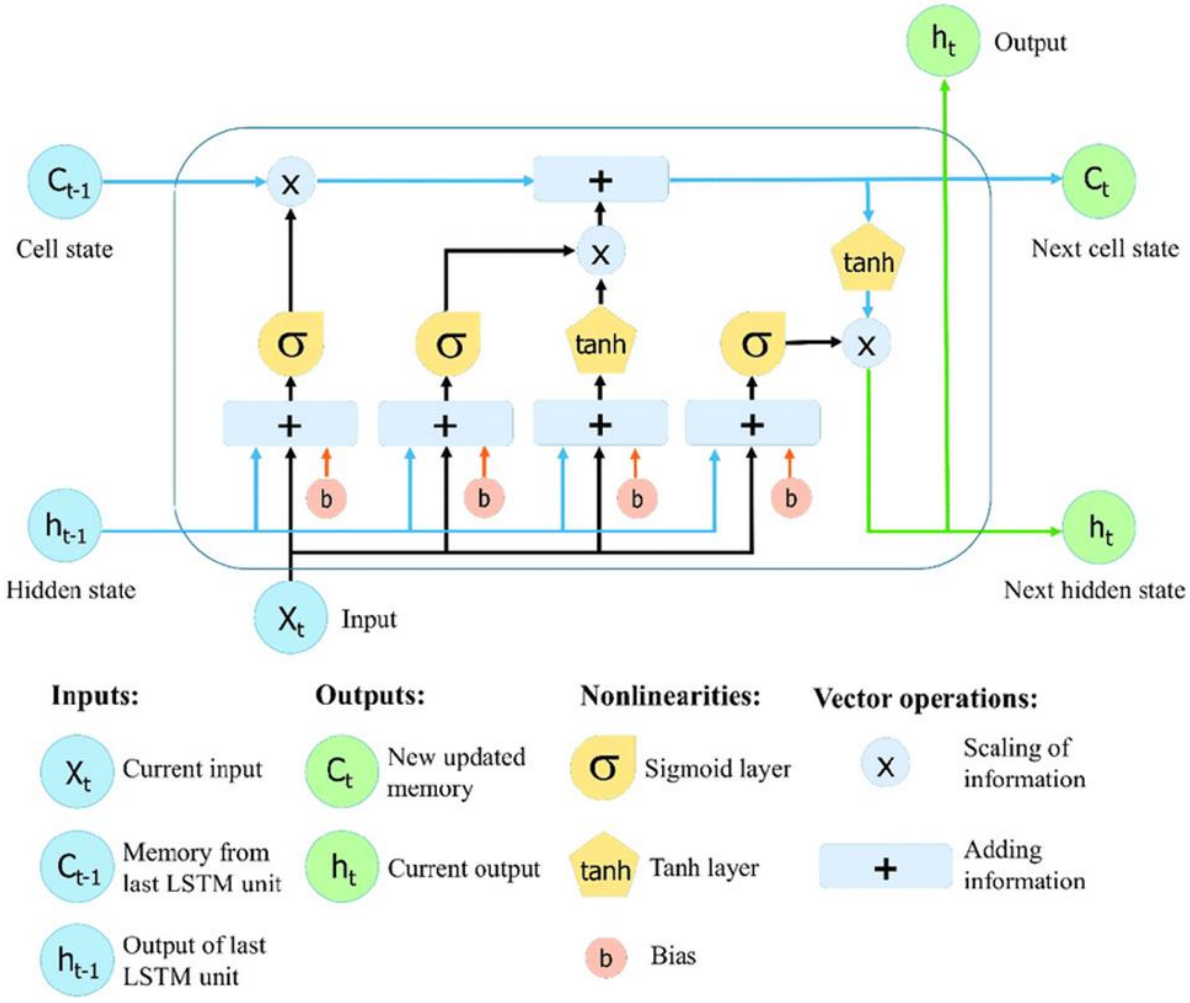
Từ công thức trên ta có thể thấy mối quan hệ giữa các biến ẩn H_t và H_{t-1}, điều đó cho thấy chúng đã lưu lại thông tin lịch sử của chuỗi cho tới bước thời gian hiện tại. Một biến ẩn còn được gọi là một trạng thái ẩn (hidden state).

1.7.2. Tổng quan về LSTM

Tiếp theo chúng tôi sẽ tìm hiểu về Long Short-Term Memory (LSTM), là một biến thể kiến trúc mạng thần kinh tái phát nhân tạo (RNN). Điểm mạnh của LSTM nằm

ở khả năng nắm bắt sự phụ thuộc thứ tự rất quan trọng để giải quyết các vấn đề phức tạp, cho phép chúng khai thác các phụ thuộc tạm thời giữa các chuỗi dữ liệu. LSTM được thiết kế để xử lý vấn đề biến mất hoặc bùng nổ độ dốc, có thể xảy ra khi đào tạo RNN truyền thống về chuỗi dữ liệu. Điều này làm cho chúng rất phù hợp cho các tác vụ liên quan đến dữ liệu tuần tự, chẳng hạn như xử lý ngôn ngữ tự nhiên (NLP), nhận dạng giọng nói và dự báo chuỗi thời gian [17].

Mạng LSTM giới thiệu các ô nhớ có khả năng lưu giữ thông tin qua các chuỗi dài. Mỗi ô nhớ có ba thành phần chính: cổng đầu vào, cổng quên và cổng đầu ra. Những cổng này giúp điều chỉnh luồng thông tin vào và ra khỏi tế bào bộ nhớ.



Hình 5: Sơ đồ trực quan về kiến trúc LSTM

(Nguồn: Le, X. H., Ho, H. V., Lee, G., & Jung, S., 2019) **[18]**

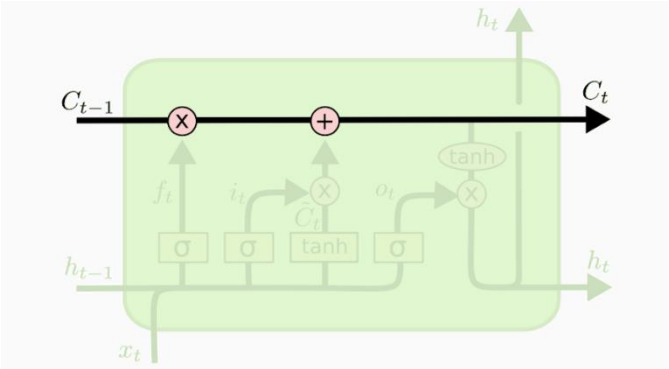
Bằng cách sử dụng các cổng, mạng LSTM có thể lưu trữ, cập nhật và truy xuất thông tin có chọn lọc qua các chuỗi dài. Điều này khiến chúng đặc biệt hiệu quả đối với các nhiệm vụ yêu cầu lập mô hình phụ thuộc lâu dài.

LSTM đã trở thành một công cụ mạnh mẽ trong trí tuệ nhân tạo và học sâu, mang lại những đột phá trong nhiều lĩnh vực khác nhau bằng cách khám phá những hiểu biết quý giá từ dữ liệu tuần tự.

1.7.3. Kiến trúc LSTM

1.7.3.1. Tổng quan kiến trúc

Ý tưởng chính của LSTM là thành phần trạng thái ô (cell state) được thể hiện qua đường chạy ngang qua đỉnh đồ thị như hình vẽ bên dưới:

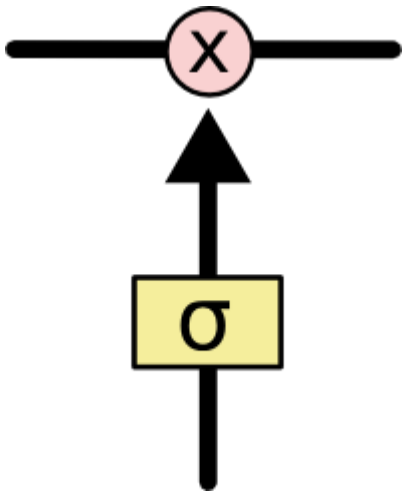


Hình 6: Đường đi của trạng thái ô (cell state) trong mạng LSTM

(Nguồn: C. Olah, 2015) **[19]**

Trạng thái ô là một dạng băng chuyền chạy thẳng xuyên suốt toàn bộ chuỗi với chỉ một vài tương tác tuyến tính nhỏ giúp cho thông tin có thể truyền dọc theo đồ thị mạng nơ ron ổn định.

LSTM có khả năng xóa và thêm thông tin vào trạng thái ô và điều chỉnh các luồng thông tin này thông qua các cấu trúc gọi là cổng. Cổng là cơ chế đặc biệt để điều chỉnh luồng thông tin đi qua. Chúng được tổng hợp bởi một tầng ẩn của hàm activation sigmoid và với một toán tử nhân như đồ thị.



Hình 7: Một cổng của hàm sigmoid trong LSTM

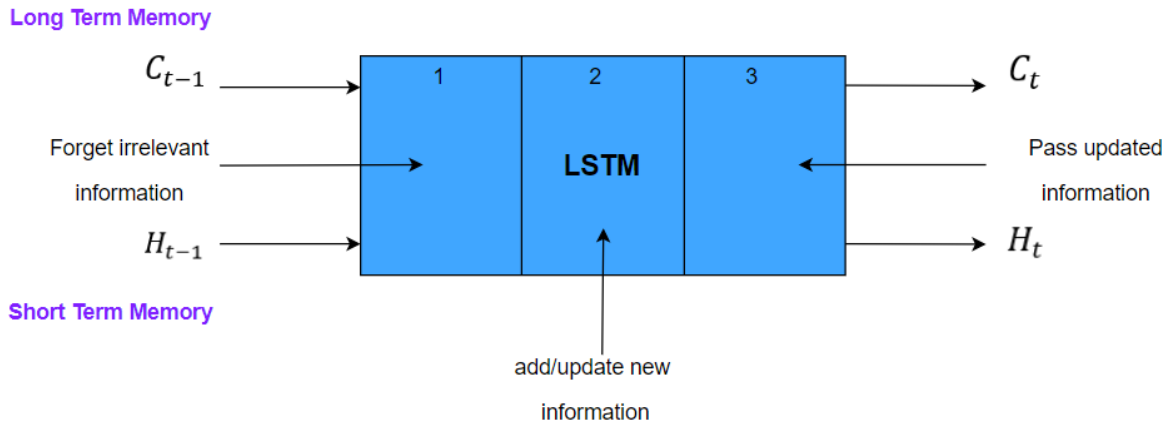
(Nguồn: C. Olah, 2015) **[19]**

Hàm sigmoid sẽ cho đầu ra là một giá trị xác suất nằm trong khoảng từ 0 đến 1, thể hiện rằng có bao nhiêu phần thông tin sẽ đi qua cổng. Giá trị bằng 0 có nghĩa rằng không cho phép thông tin nào đi qua, giá trị bằng 1 sẽ cho toàn bộ thông tin đi qua.

Một mạng LSTM sẽ có 3 cổng có kiến trúc dạng này để bảo vệ và kiểm soát các trạng thái ô. Ba phần này của đơn vị LSTM được gọi là gates (cổng). Chúng kiểm soát luồng thông tin vào và ra khỏi ô nhớ hoặc ô LSTM. Cổng đầu tiên gọi là cổng quên (Forget gate), cổng thứ hai gọi là cổng vào (Input gate) và cổng cuối cùng là cổng ra (Output gate). Một đơn vị LSTM bao gồm ba cổng này và một ô nhớ hoặc ô LSTM có thể được coi là một lớp nơ-ron trong mạng nơ-ron truyền thẳng truyền thống, với mỗi nơ-ron có một lớp ẩn và trạng thái hiện tại.

Giống như một RNN đơn giản, LSTM cũng có trạng thái ẩn trong đó H_{t-1} biểu thị trạng thái ẩn của dấu thời gian trước đó và H_t là trạng thái ẩn của dấu thời gian hiện tại. Ngoài ra, LSTM còn có trạng thái ô được biểu thị lần lượt là C_{t-1} và C_t cho dấu thời gian trước đó và hiện tại.

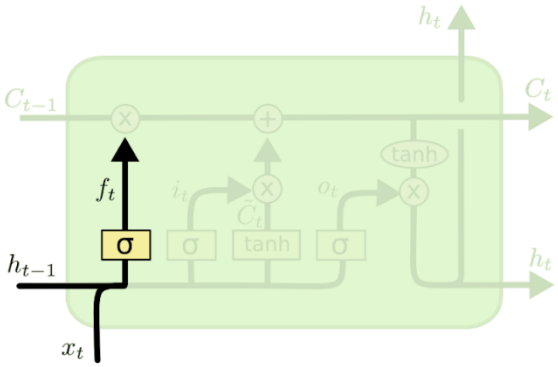
Ở đây trạng thái ẩn được gọi là Bộ nhớ ngắn hạn và trạng thái ô được gọi là Bộ nhớ dài hạn.



Hình 8: Cấu trúc 3 phần của LSTM

1.7.3.2. Tầng cổng quên (Forget gate)

Bước đầu tiên trong LSTM sẽ quyết định xem thông tin nào chúng ta sẽ cho phép đi qua trạng thái ô (cell state).



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

Hình 9: Tầng cổng quên

(Nguồn: C. Olah, 2015) **[19]**

Trong đó:

- f_t là đầu ra của cổng quên tại thời điểm t
- h_{t-1} là trạng thái ẩn của LSTM từ thời điểm trước đó (t−1)
- x_t là đầu vào tại thời điểm t
- W_f là trọng số liên quan đến cổng quên
- b_f là bias của cổng quên
- σ là hàm kích hoạt sigmoid, sử dụng để giới hạn [0, 1]

Công thức này mô tả cách tính đầu ra của cổng quên:

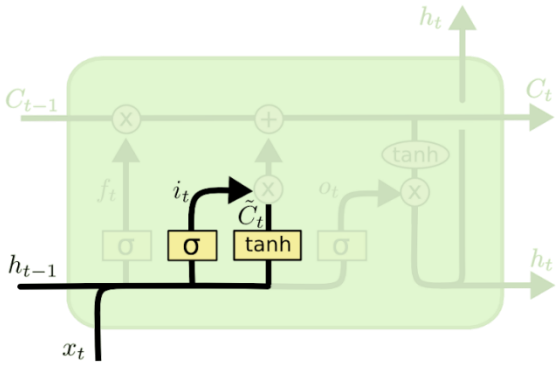
- Kết hợp h_{t-1} và x_t: Trạng thái ẩn từ thời điểm trước đó h_{t-1} và đầu vào hiện tại x_t được kết hợp lại thành một vector.
- Tích trọng số và cộng bias: Vector này sau đó được nhân với trọng số W_f và cộng thêm bias b_f.
- Áp dụng hàm sigmoid: Hàm sigmoid (σ) được áp dụng lên kết quả này để cho ra đầu ra của cổng quên f_t. Hàm sigmoid sẽ giới hạn giá trị đầu ra trong khoảng [0, 1], quyết định tỉ lệ thông tin nào từ trạng thái ô trước đó cần được quên đi.

Ý nghĩa cổng quên: Cổng quên f_t xác định thông tin nào từ trạng thái ô trước đó C_{t-1} nên được giữ lại và thông tin nào nên bị quên đi. Nếu f_t gần với 0, thông tin sẽ bị quên đi. Nếu f_t gần với 1, thông tin sẽ được giữ lại.

Ví dụ trong ngôn ngữ, ta đang cố gắng dự báo từ tiếp theo dựa trên toàn bộ những từ trước đó. Trong những trường hợp như vậy, trạng thái ô 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 ta 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 cổng quên cho phép cập nhật thông tin mới và lưu giữ giá trị của nó khi có thay đổi theo thời gian.

1.7.3.3. Cổng vào (Input gate)

Bước tiếp theo chúng ta sẽ quyết định loại thông tin nào sẽ được lưu trữ trong trạng thái ô. 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 một vectơ của một giá trị trạng thái mới $\tilde{C_t}$ 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.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

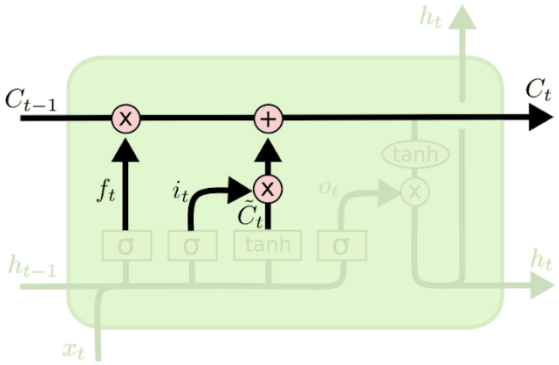
Hình 10: Cập nhật giá trị cho trạng thái ô bằng cách kết hợp 2 kết quả từ tầng cổng vào và tầng ẩn hàm tanh

(Nguồn: C. Olah, 2015) **[19]**

Trong đó,

- i_t là đầu ra của cổng tại thời điểm t.
- $C_t^{\sim}$ là giá trị mới tiềm năng của trạng thái ô tại thời điểm t.

Đây là lúc để cập nhật một trạng thái ô cũ, C_{t-1} sang một trạng thái mới C_t. Những bước trước đó đã quyết định làm cái gì, và tại bước này chỉ cần thực hiện nó. Chúng ta nhân trạng thái cũ với f_t tương ứng với việc quên những thứ quyết định được phép quên sớm. Phần tử đề cử $it * C_t^{\sim}$ là một giá trị mới được tính toán tương ứng với bao nhiêu được cập nhật vào mỗi giá trị trạng thái. Trạng thái ô mới C_t được hình thành bằng cách kết hợp hai phần trên, một phần giữ lại từ trạng thái cũ và một phần thêm thông tin mới. Sự kết hợp này giúp LSTM có thể nhớ thông tin qua nhiều bước thời gian, đồng thời linh hoạt cập nhật với thông tin mới.



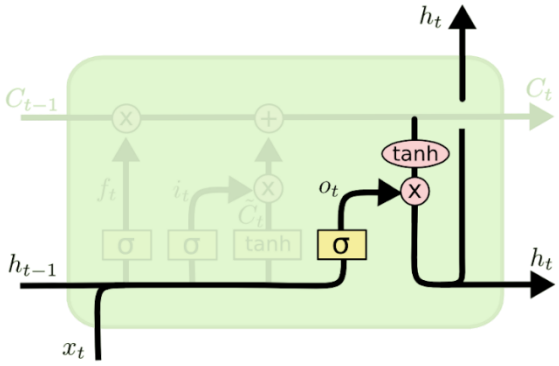
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Hình 11: Trạng thái ô mới

(Nguồn: C. Olah, 2015) **[19]**

1.7.3.4. Cổng ra (Output gate)

Cuối cùng cần quyết định xem đầu ra sẽ trả về bao nhiêu. Kết quả ở đầu ra sẽ dựa trên trạng thái ô, nhưng sẽ là một phiên bản được lọc. Đầu tiên, chúng ta chạy qua một tầng sigmoid nơi quyết định phần sẽ ở đầu ra. Sau đó, trạng thái ô 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.



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Hình 12: Điều chỉnh thông tin ở đầu ra thông qua hàm tanh

(Nguồn: C. Olah, 2015) **[19]**

1.8. Phân loại cảm xúc bằng BERT

1.8.1. Phân loại cảm xúc văn bản

Bình luận về sản phẩm là một đặc trưng có ảnh hưởng lớn đến quyết định mua sản phẩm của khách hàng, tuy nhiên, những bình luận này đang ở dạng văn bản và chưa thể đưa vào mô hình dự đoán. Nếu trước đây việc đánh giá phản ứng của thị trường với một sản phẩm thương mại là "quyền năng" của các chuyên gia trong lĩnh bán hàng, thì giờ đây, với sự phát triển của công nghệ, chúng ta hoàn toàn có thể sử dụng các mô hình học máy để chuyển những văn bản thô thành những con số phân loại cảm xúc cho văn bản. Bài toán phân loại cảm xúc văn bản có đầu vào là các văn bản bình luận, mô hình sẽ đưa ra dự đoán văn bản đó là một bình luận tích cực, tiêu cực hay trung tính.

Trong đề tài này, chúng tôi sử dụng phân tích cảm xúc để có thể chuyển những đánh giá sản phẩm dạng văn bản thành những con số có ý nghĩa phân loại để có thể đưa

vào các thuật toán chỉ có thể tính toán dựa trên các đặc trưng dạng số. Chúng tôi lựa chọn mô hình BERT cho tác vụ này vì có nhiều nghiên cứu đã chỉ ra sự vượt trội của BERT trong tác vụ phân loại cảm xúc văn bản như:

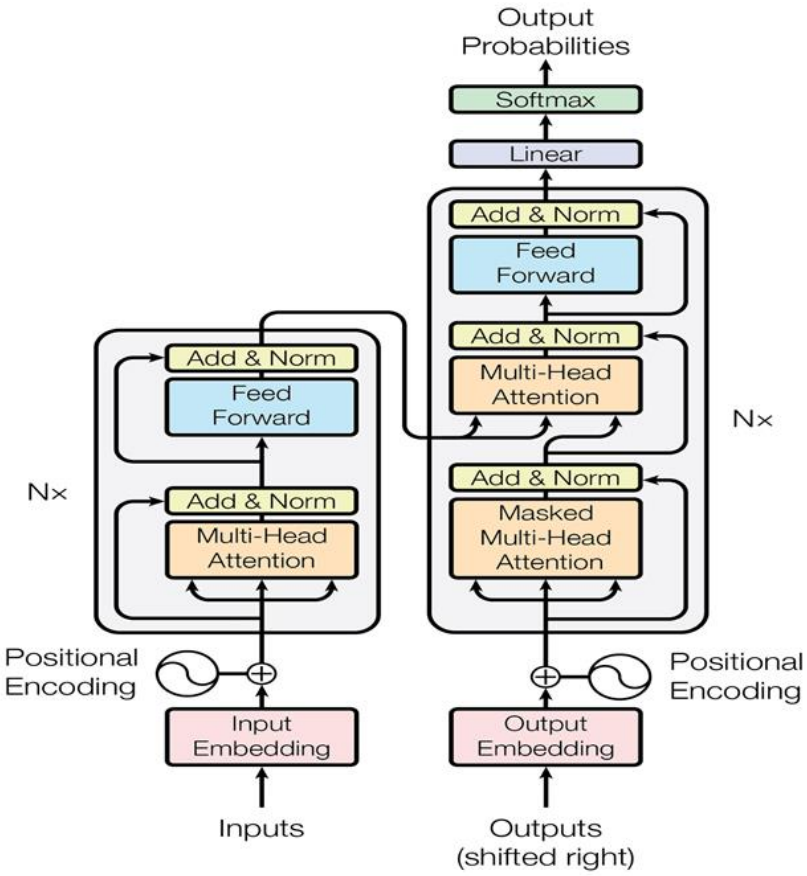
- Nghiên cứu [20] chỉ ra rằng mô hình BERT cho hiệu suất vượt trội trong tác vụ phân tích cảm xúc so với các mô hình trước đây như LSTM (Long Short-Term Memory), Naive Bayes và SVM (Support Vector Machines).
- Một nghiên cứu khác cũng cho kết quả BERT đạt độ chính xác (accuracy) 88.48% trong phân tích cảm xúc đánh giá của người tiêu dùng, vượt trội hơn các mô hình khác [21].
- Bài báo [22] đã cho ta thấy sức mạnh và độ phổ biến của BERT trong việc phát hiện cảm xúc cho văn bản qua việc đánh giá các ưu điểm và nhược điểm của các mô hình được sử dụng như Transformer-XL, XLM, BERT và GPT cùng các biến thể của nó.

1.8.2. Tìm hiểu về BERT

BERT (Bidirectional Encoder Representations from Transformers) là một kỹ thuật học máy dựa trên Transformer để đào tạo trước cho tác vụ xử lý ngôn ngữ tự nhiên (NLP) do Google phát triển. BERT được tạo và xuất bản vào năm 2018 bởi Jacob Devlin và các đồng nghiệp của ông tại Google. BERT là một mô hình ngôn ngữ dựa trên kiến trúc Transformer, nổi tiếng với khả năng cải thiện đáng kể so với các mô hình tiên tiến trước đó. Nó đã đạt được kết quả cao trên nhiều nhiệm vụ xử lý ngôn ngữ tự nhiên, bao gồm việc đưa điểm GLUE (General Language Understanding Evaluation) đạt 80.5%, độ chính xác của MultiNLI (Multi-Genre Natural Language Inference) đạt 86.7%, và độ chính xác của SQuAD (Stanford Question Answering Dataset) v1.1 đạt 93.2 và SQuAD v2.0 đạt 83.1 [23].

Cốt lõi của BERT là mô hình Transformer, bao gồm nhiều lớp mã hóa và đầu tự chú ý. Nói một cách đơn giản, Transformer có hai cơ chế riêng biệt - một bộ mã hóa (encoder) đọc văn bản đầu vào và một bộ giải mã (decoder), tạo ra các dự đoán cho các tác vụ khác nhau. Bộ mã hóa xử lý chuỗi đầu vào của các từ hoặc mã thông báo (token), mỗi từ hoặc token được biểu diễn dưới dạng vectơ, sử dụng các lớp self-attention và

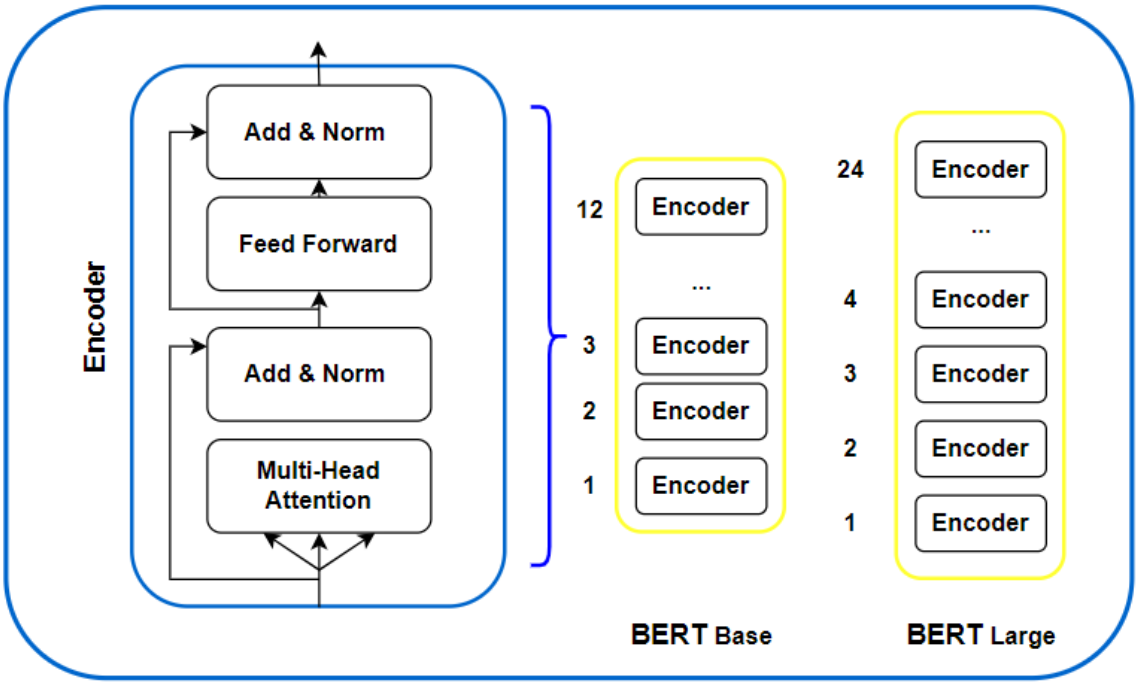
mạng neural feed-forward để xử lý chuỗi đầu vào. Quá trình này lặp lại qua nhiều lớp để tạo ra các biểu diễn đa chiều của đầu vào. Bộ giải mã , nhận đầu vào từ bộ mã hóa và dự đoán trình tự đầu ra. Nó sử dụng các lớp self-attention và lớp attention đa hướng để xử lý đầu vào. Ở mỗi bước giải mã, mô hình tạo ra một biểu diễn tạm thời cho từ hiện tại, sử dụng nó để dự đoán từ tiếp theo trong chuỗi đầu ra. Google đã trình bày chi tiết các hoạt động phức tạp của Transformer trong bài báo "Attention Is All You Need".



Hình 13: Kiến trúc Transformer

(Nguồn: Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, 2017) **[24]**

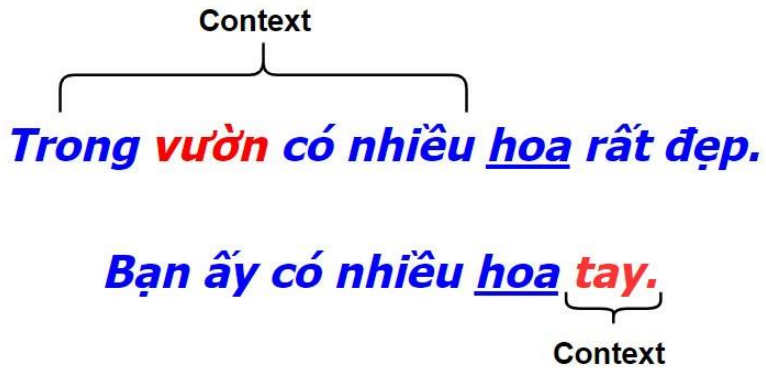
Transformer có kiến trúc ngăn xếp bộ mã hóa và bộ giải mã, trong khi BERT chỉ là ngăn xếp bộ mã hóa của kiến trúc transformer. Có hai biến thể, BERT-base và BERT-large , khác nhau về độ phức tạp của kiến trúc. Mô hình BERT-base có 12 lớp trong bộ mã hóa trong khi mô hình BERT-large có 24 lớp.



Hình 14: BERT và biến thể

(Vẽ lại dựa trên Jana, S., Biswas, R., Pal, K., Biswas, S., & Roy, K., 2024) **[25]**

BERT được thiết kế để tối ưu cho các tác vụ hiểu ngôn ngữ tự nhiên như phân loại câu, nhận dạng thực thể có tên, trả lời câu hỏi và các tác vụ tương tự. Một trong những điểm mạnh chính của BERT là khả năng xem xét ngữ cảnh từ cả hai chiều (trái và phải). Bộ mã hóa trong mô hình transformer của BERT sử dụng các tầng Multi-Head Attention để xử lý đồng thời tất cả các từ trong câu, giúp mô hình nắm bắt được ngữ cảnh đầy đủ và hai chiều của mỗi từ. Tính hai chiều của một mô hình rất quan trọng để thực sự hiểu được ý nghĩa của một ngôn ngữ. Ví dụ để minh họa điều này, có hai câu trong ví dụ và cả hai đều liên quan đến từ “bank”:



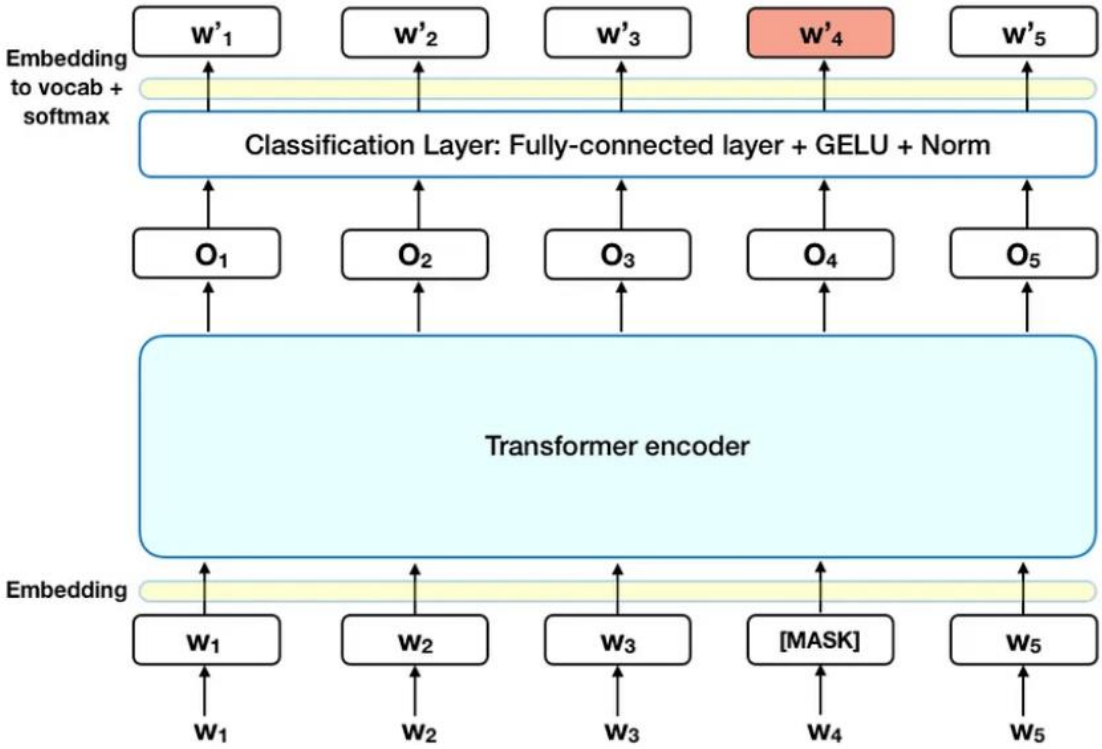
Hình 15: Ví dụ tính hiểu ngôn ngữ hai chiều

Nếu chúng ta cố gắng dự đoán bản chất của từ "hoa" bằng cách chỉ sử dụng ngữ cảnh bên trái hoặc bên phải, thì chúng ta sẽ mắc lỗi ít nhất một trong hai ví dụ trên. Một cách để giải quyết vấn đề này là xem xét cả bối cảnh bên trái và bên phải trước khi đưa ra dự đoán. Đó chính xác là những gì BERT làm.

1.8.3. BERT hoạt động như thế nào?

Có hai bước trong khung BERT: đào tạo trước và tinh chỉnh. Quá trình đào tạo trước sử dụng hai nhiệm vụ tự giám sát: mô hình ngôn ngữ đeo mặt nạ (Masked Language Model - MLM) và dự đoán câu tiếp theo (Next Sentence Prediction - NSP).

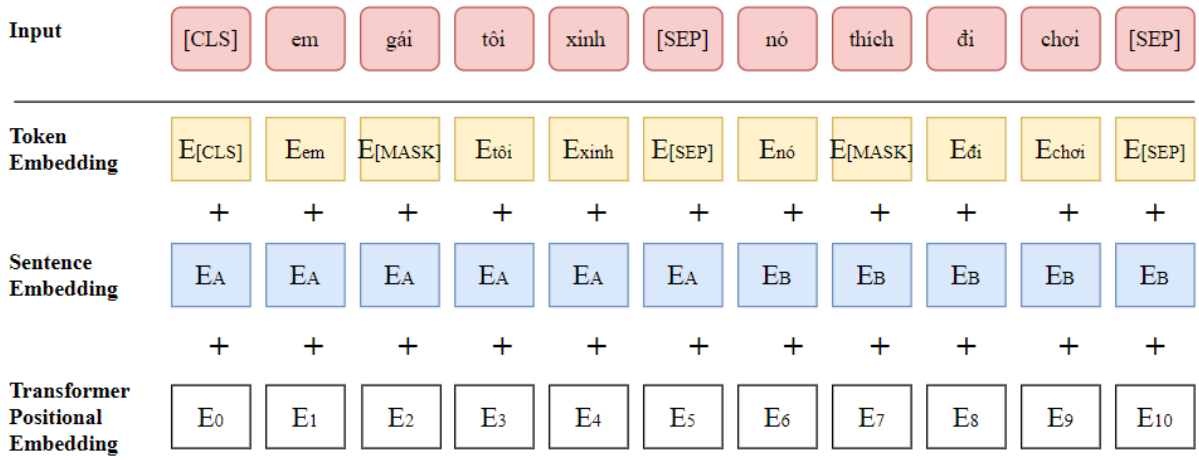
Mô hình ngôn ngữ đeo mặt nạ (MLM): Theo cách tiếp cận này, 15% từ trong đầu vào được che giấu ngẫu nhiên và mô hình được đào tạo để dự đoán những từ bị che giấu này dựa trên ngữ cảnh xung quanh. Điều này cho phép mô hình tìm hiểu các cách trình bày hai chiều của văn bản, xem xét đồng thời cả ngữ cảnh bên trái và bên phải. Cách tiếp cận hai chiều này cho phép BERT phát triển sự hiểu biết sâu sắc hơn về ngữ cảnh và ý nghĩa ngôn ngữ.



Hình 16: Mô tả MLM trong BERT

(Nguồn: Athinaios, K., Chalkidis, I., Pantazi, D. A., & Papaloukas, C., 2020) **[26]**

Trong dự đoán câu tiếp theo (NSP): mô hình nhận các cặp câu làm đầu vào và học cách dự đoán xem câu thứ hai trong cặp có phải là câu tiếp theo trong tài liệu gốc hay không. Trong quá trình đào tạo, 50% đầu vào là một cặp với câu thứ hai là câu tiếp theo trong tài liệu gốc, trong khi 50% còn lại, một câu ngẫu nhiên từ kho ngữ liệu được chọn làm câu thứ hai. Giả sử câu ngẫu nhiên sẽ bị ngắt kết nối với câu đầu tiên. Ví dụ: với "[CLS] con chó của tôi dễ thương [SEP] nó thích chơi", mô hình sẽ dự đoán [IsNext]. Với "[CLS] con chó của tôi rất dễ thương [SEP] nam châm hoạt động như thế nào" mô hình sẽ dự đoán [NotNext].

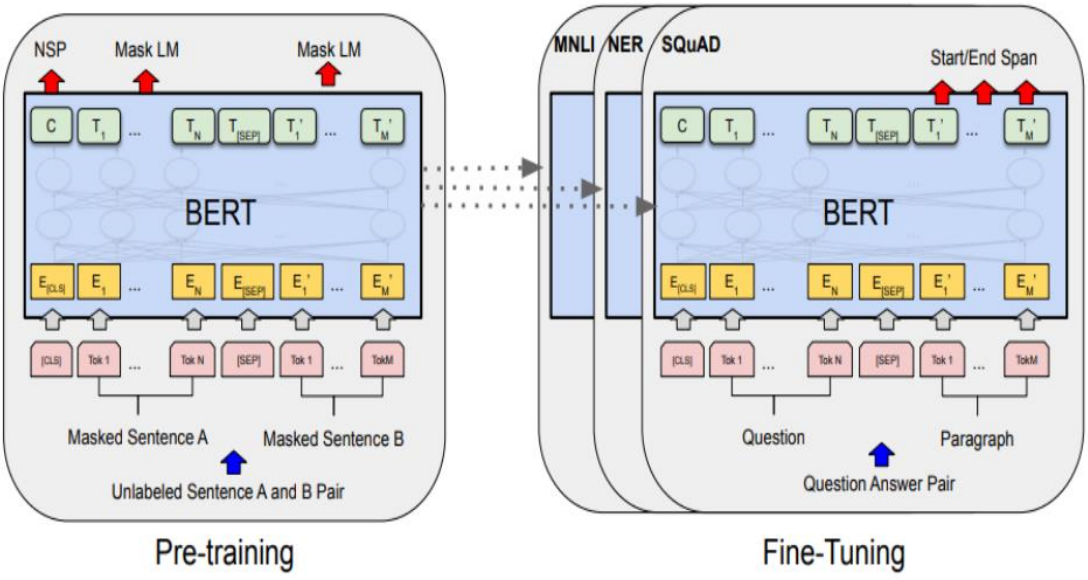


Hình 17: Tiến trình pre-training trong BERT

(Vẽ lại dựa trên Devlin, J., Chang, M. W., Lee, K., & Toutanova, K., 2018) **[23]**

Khi đào tạo mô hình BERT, MLM và NSP được đào tạo cùng nhau, với mục tiêu giảm thiểu hàm mất mát kết hợp từ hai chiến lược.

Kết quả của quá trình đào tạo này là BERT học được cách biểu đạt tiềm ẩn của từ và câu trong ngữ cảnh. Sau khi đào tạo trước, BERT có thể được tinh chỉnh với ít tài nguyên hơn trên các tập dữ liệu nhỏ hơn có gán nhãn để tối ưu hóa hiệu suất của nó đối với các tác vụ cụ thể như tác vụ NLP (suy luận ngôn ngữ, phân loại văn bản, trả lời câu hỏi,..). Giai đoạn tiền huấn luyện tốn kém hơn đáng kể về mặt tính toán so với giai đoạn tinh chỉnh.



Hình 18: Tổng quan quá trình pre-training và fine-tuning của BERT

(Nguồn: Shreyashree, S., Sunagar, P., Rajarajeswari, S., & Kanavalli, A., 2022) **[27]**

Tinh chỉnh BERT cho các nhiệm vụ cụ thể sau:

- Masked LM và NSP: Tiếp tục tinh chỉnh trên các tác vụ này với dữ liệu cụ thể để cải thiện độ chính xác.
- Tác vụ cụ thể: Tinh chỉnh mô hình cho các tác vụ như phân loại cảm xúc, xác định mối quan hệ giữa hai câu (MGNL - Multi-Genre Natural Language Inference), nhận diện thực thể có tên (NER - Named Entity Recognition) hoặc trả lời câu hỏi (như trong bộ dữ liệu SQuAD - Stanford Question Answering Dataset).

Ví dụ với các tác vụ phân loại như phân tích cảm xúc được thực hiện tương tự như phân loại Câu tiếp theo NSP, bằng cách thêm một lớp phân loại lên trên đầu ra Transformer cho mã thông báo [CLS].

1.8.4. Mô hình PhoBERT và tinh chỉnh cho tác vụ phân tích cảm xúc

Cho đến nay, có rất nhiều mô hình ngôn ngữ được đào tạo trước với các cài đặt khác nhau cho tiếng Việt bao gồm cả mô hình đơn và đa ngôn ngữ. Trong đó, PhoBERT là mô hình ngôn ngữ được đào tạo trước từ cấp độ từ (word-level) đầu tiên cho tiếng Việt [28]. PhoBERT với hai phiên bản PhoBERT-base và PhoBERT-large. Phương pháp đào tạo trước PhoBERT dựa trên RoBERTa (A Robustly Optimized BERT Pretraining

Approach) giúp tối ưu hóa quy trình đào tạo trước BERT để có hiệu suất mạnh mẽ hơn. Theo bài báo “PhoBERT: Pre-trained language models for Vietnamese” của tác giả Nguyễn Quốc Đạt và Nguyễn Tuấn Anh kết quả thử nghiệm cho thấy PhoBERT hoạt động tốt hơn mô hình XLM-R (XLM-RoBERTa là mô hình đa ngôn ngữ, các mẫu văn bản được lấy từ tất cả các ngôn ngữ và được đào tạo để dự đoán mã thông báo ẩn trong đầu vào - mô hình do Conneau cùng cộng sự phát triển vào 2020) và cải thiện tính năng tiên tiến trong nhiều nhiệm vụ NLP dành riêng cho tiếng Việt bao gồm gắn thẻ Phần lời nói , nhận dạng thực thể được đặt tên, suy luận ngôn ngữ tự nhiên [28]. Một ví dụ khác là các nhà nghiên cứu đã sử dụng PhoBERT để phân loại tâm lý trong tin tức chứng khoán Việt Nam, đạt độ chính xác ấn tượng 93,12%. Hiệu quả của mô hình thể hiện tính hiệu quả trong việc hiểu cảm xúc trong văn bản tiếng Việt [29].

PhoBERT được phát hành để tạo điều kiện thuận lợi cho việc nghiên cứu và ứng dụng tiếp theo trong tương lai cho NLP Việt Nam. Trong đề tài của chúng tôi, PhoBERT sử dụng để chuyển những bình luận đánh giá là văn bản của người dùng mà trước giờ thường bị bỏ qua thành các đặc trưng để cải thiện thêm độ chính xác cho mô hình.

Nhận thấy sự phụ hợp về mục tiêu dự đoán và tập dữ liệu sử dụng về đánh giá sản phẩm của khách hàng, chúng tôi đã sử dụng mô hình phobert-base-vi-sentiment-analysis để phân tích tâm lý, cảm xúc của văn bản tiếng Việt do Đặng Việt Dũng tinh chỉnh và phát triển dựa trên mô hình PhoBERT-Base của VinAI. Nó phân loại văn bản thành các cảm xúc "Tích cực", "Tiêu cực" hoặc "Trung lập". Mô hình này đào tạo trên tập dữ liệu gồm 31.436 đánh giá sản phẩm được lấy từ Kaggle. Sau đây là vài dòng về dữ liệu mẫu của tập dữ liệu đã sử dụng để tinh chỉnh mô hình PhoBERT:

comment	label	rate
Chất lượng sản ph... 2% Đóng gói sản phẩm... 1% Other (30635) 97%	POS 64% NEG 21% Other (4698) 15%	 1 5
Áo bao đẹp ạ!!	POS	5
Tuyệt vời !	POS	5
2day ao khong giong trong.	NEG	1
Mùi thơm,bôi lên da mềm da.	POS	5
Vải đẹp, dày dặn.	POS	5
Hàng rất đẹp, rất chi là ưng ý.	POS	5
Chất lượng sản phẩm tốt, date dài.	POS	5

Hình 19: Tập dữ liệu sử dụng để tinh chỉnh mô hình cho tác vụ phân tích cảm xúc

Các ví dụ đánh giá cho thấy độ tin cậy cao trong dự đoán đối với nhiều cảm xúc khác nhau. Mô hình này là nguồn mở và có thể được sử dụng tự do và được phát hành trên trang web chính thức của Hugging Face.

2. CHƯƠNG 2: CÁC THÀNH PHẦN VÀ PHƯƠNG PHÁP TỐI ƯU HÓA MÔ HÌNH HỌC MÁY

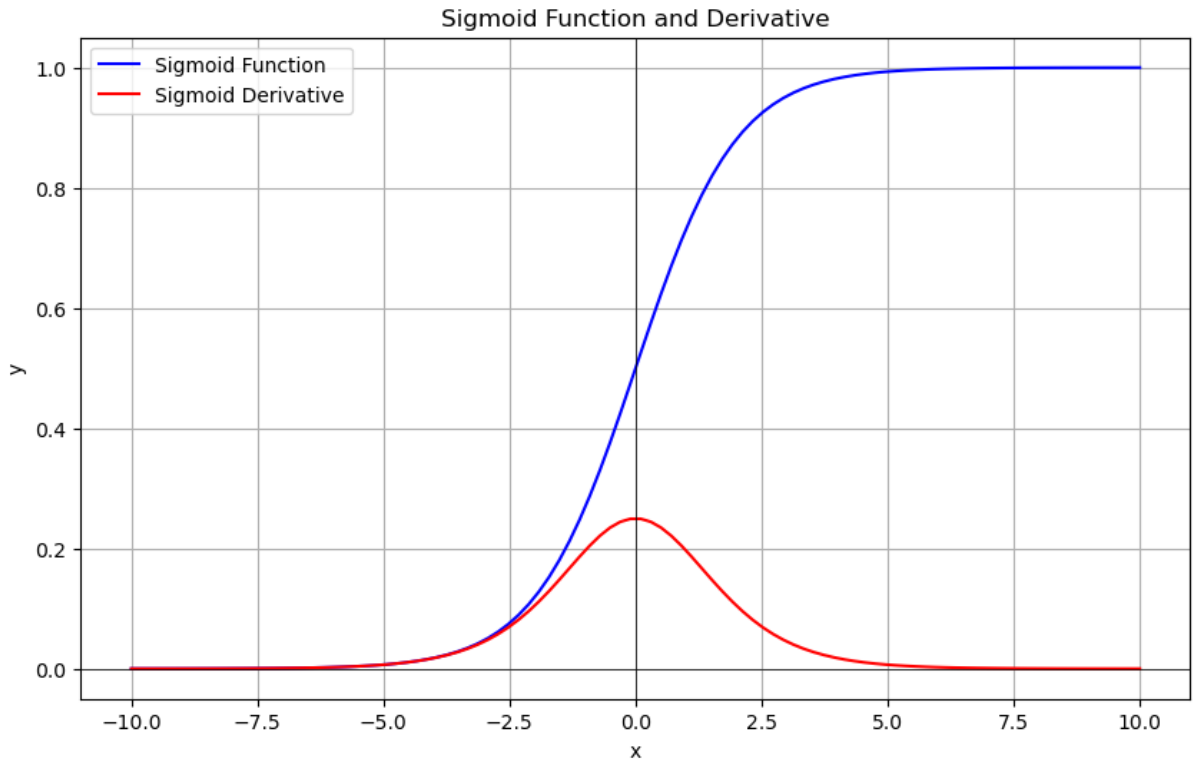
2.1. Hàm kích hoạt

Hàm kích hoạt là một khái niệm quan trọng trong học máy và đặc biệt là trong các mạng nơ-ron. Chúng là một hàm số nhận vào một giá trị đầu vào và cho kết quả là một giá trị nằm trên khoảng giá trị nào đó. Một số hàm kích hoạt phổ biến phải kể đến đó là Sigmoid, Tanh và ReLU.

2.1.1. Hàm sigmoid

Hàm sigmoid nhận giá trị đầu vào trong khoảng $(-\infty, \infty)$ và có giá trị đầu ra trong khoảng $(0,1)$. Nó thường được sử dụng trong các mô hình phân loại nhị phân, đặc biệt là trong lớp đầu ra của mạng nơ-ron.

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



Hình 20: Đồ thị hàm sigmoid và đạo hàm của nó

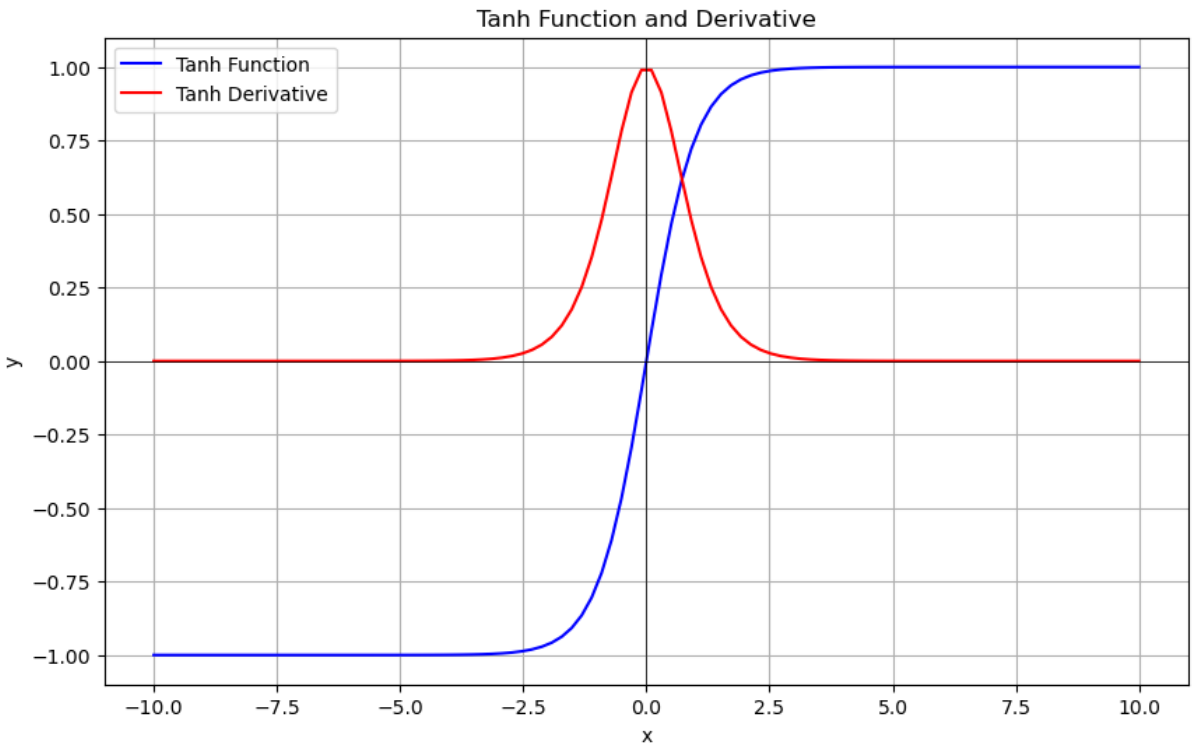
Hàm sigmoid vẫn là hàm phổ biến để biểu diễn kết quả như là xác suất nhị phân. Tuy nhiên, hàm sigmoid có một số vấn đề như:

- Việc tính toán tốn kém tài nguyên vì có phép tính mũ.
- Hàm Sigmoid có gradient nhỏ khi giá trị đầu vào của nó rất lớn hoặc rất nhỏ (đầu ra gần 0 hoặc 1), dẫn đến gradient gần bằng 0. Điều này được gọi là hiện tượng biến mất gradient làm cho việc cập nhật trọng số trong quá trình huấn luyện trở nên rất chậm, đặc biệt là trong các mạng sâu.

2.1.2. Hàm Tanh

Tương tự như hàm sigmoid, hàm tanh (Hyperbolic Tangent) cũng nhận giá trị đầu vào trong khoảng $(-\infty, \infty)$ nhưng có giá trị đầu ra trong khoảng $(-1,1)$. Công thức hàm tanh được biểu diễn như sau:

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



Hình 21: Đồ thị hàm tanh và đạo hàm của nó

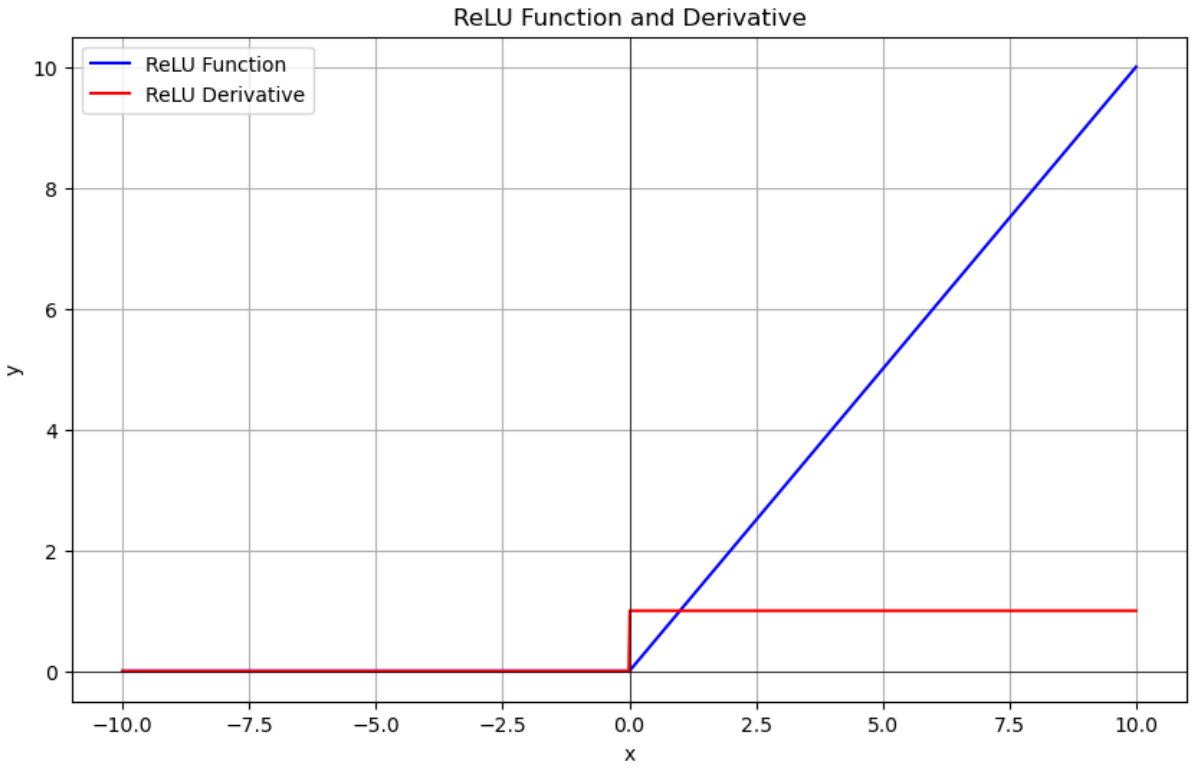
Hình dạng đồ thị của hàm tanh nhìn qua khá tương đồng với hàm sigmoid. Tuy nhiên, giá trị của hàm tanh có một phép biến đổi từ âm qua dương, và đồ thị có tính chất đối xứng qua tâm O.

Mặc dù đã cải thiện so với hàm sigmoid nhưng hàm tanh vẫn gặp vấn đề biến mất gradient ở 2 đầu, có thể dẫn đến quá trình học của mạng trở nên chậm hơn.

2.1.3. Hàm ReLU

Hàm ReLU (Rectified Linear Unit) là một hàm đơn giản và được sử dụng phổ biến. Giá trị của hàm ReLU nằm trong nửa khoảng $[0, \infty)$. ReLU đơn giản là chuyển các giá trị nhỏ hơn 0 về bằng 0. Công thức được định nghĩa là

$$ReLU(x) = \max(0, x)$$



Hình 22: Đồ thị hàm ReLU và đạo hàm của nó

Ưu điểm của hàm ReLU nằm ở sự đơn giản của nó, rất dễ cài đặt hay tính toán nhưng lại tránh được vấn đề gradient biến mất. Tuy nhiên, hàm ReLU cũng có nhược điểm của nó được gọi là "Dying ReLU". Khi giá trị đầu vào của ReLU liên tục nhỏ hơn

0, nó sẽ luôn cho đầu ra là 0, dẫn đến việc các nơ-ron có thể "chết" và không đóng góp gì vào quá trình học.

2.2. Thuật toán tối ưu hóa

Thuật toán tối ưu hóa (Optimizer) là thuật toán được sử dụng để điều chỉnh các tham số trong mô hình nhằm tối thiểu hàm mất mát.

2.2.1. Gradient Descent

Gradient Descent là phương pháp tìm cực trị (tối thiểu hoặc tối đa) của một hàm số bằng cách lặp đi lặp lại, di chuyển theo hướng ngược với gradient của hàm số đó

$$\theta_{t+1} = \theta_t - \mu \nabla_\theta f(\theta)$$

Trong đó:

- θ: là vector tham số cần tối ưu hóa
- μ: là learning rate (tốc độ học)
- $\nabla_\theta f(\theta)$: là gradient của hàm số tại điểm θ

Trong trường hợp lý tưởng, công thức cập nhật này sẽ lặp lại cho tới khi tìm được thông số θ tối ưu. Tuy nhiên, chúng ta sẽ không biết khi nào sẽ điểm tối ưu, nên chúng ta đặt ra những điều kiện dừng như:

- Gradient đủ nhỏ
- Số vòng lặp cực đại
- Thay đổi hàm mất mát đủ nhỏ
- Dừng sớm (Early Stopping): là kỹ thuật dừng quá trình huấn luyện mô hình khi hiệu suất trên tập kiểm tra không cải thiện sau một số lượng epoch nhất định.

2.2.2. Stochastic Gradient Descent (SGD)

Stochastic Gradient Descent (SGD) là một biến thể của Gradient Descent, trong đó chỉ sử dụng một mẫu dữ liệu ngẫu nhiên để tính toán gradient tại mỗi lần lặp thay vì toàn bộ tập dữ liệu. SGD thường được sử dụng khi tập dữ liệu lớn.

2.2.3. Momentum-based Gradient Descent

Momentum-based Gradient Descent (Giảm dần độ dốc dựa trên động lượng) là một biến thể tối ưu hóa kết hợp một momentum term để cập nhật tốc độ học. Momentum term được tính là trung bình động của các độ dốc trong quá khứ, được kiểm soát bởi một siêu tham số gọi là Beta. Cách tiếp cận này làm giảm các mẫu hình zigzag và tạo điều kiện cho sự hội tụ nhanh hơn đến mức tối thiểu toàn cục.

2.2.4. AdaGrad

AdaGrad (Adaptive Gradient Algorithm) là một thuật toán tối ưu hóa được thiết kế để điều chỉnh tốc độ học tập cho mỗi tham số một cách riêng biệt.

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\mu}{\sqrt{G_{t,ii} + \varepsilon}} g_{t,i}$$

Trong đó:

- μ: là tốc độ học
- $G_{t,ii}$ là tổng các bình phương của các gradient cho đến $\theta_{t,i}$
- $g_{t,i}$ là gradient của hàm mục tiêu với tham số $\theta_{t,i}$
- ε là một hằng số nhỏ được thêm vào để ổn định số học

2.2.5. RMSprop

RMSprop (Root Mean Square Propagation) là một thuật toán tối ưu hóa tốc độ học tập thích ứng. Đây là phần mở rộng của thuật toán AdaGrad và giảm hiệu quả công sức tính toán trong quá trình đào tạo. RMSprop làm giảm tốc độ học tập theo cấp số nhân khi gradient bình phương giảm xuống dưới một ngưỡng nhất định.

Thay vì dùng $G_{t,ii}$ như AdaGrad, RMSprop dùng giá trị trung bình động của các bình phương gradient

$$E[g^2]_{t,i} = \gamma E[g^2]_{t-1,i} + (1-\gamma) g_{t,i}^2$$

Với $E[g^2]_{t,i}$ là giá trị trung bình động của các bình phương gradient tại $\theta_{t,i}$ và γ là hệ số suy giảm.

2.2.6. ADAM

ADAM sử dụng các phương trình toán học để cập nhật các tham số trong quá trình tối ưu hóa. Nó kết hợp phương trình tốc độ cập nhật dựa trên Momentum với phương trình cập nhật RMSprop. Các giá trị của siêu tham số, chẳng hạn như β_1 (đặt thành 0,9), β_2 (đặt thành 0,999) và ε (đặt thành 10^{-8}).

ADAM đã trở thành thuật toán tối ưu hóa được ưa chuộng để đào tạo Mạng nơ-ron sâu (Deep Neural Network - DNN). Trong nghiên cứu của Kingma và Ba [30] các nhà nghiên cứu đã phát hiện ra rằng ADAM vượt trội hơn các trình tối ưu hóa khác, bao gồm AdaGrad, Gradient Descent và RMSprop về tốc độ và hiệu suất thử nghiệm.

2.3. Các chỉ số đánh giá mô hình

Đánh giá hiệu quả của một mô hình học máy là một bước quan trọng trong quy trình phát triển mô hình. Các chỉ số đánh giá giúp hiểu rõ hiệu suất, mức độ chính xác và khả năng dự đoán của mô hình. Trong phạm vi đề tài chúng ta sẽ tìm hiểu về một số chỉ số cơ bản trong đánh giá mô hình hồi quy.

2.3.1. MAE

MAE là viết tắt của Mean Absolute Error, nghĩa là lỗi trung bình tuyệt đối. Chỉ số này đo lường mức độ sai lệch trung bình giữa các giá trị dự đoán của mô hình và các giá trị thực tế. Nó tính toán bằng cách lấy trung bình của các giá trị tuyệt đối của sự khác biệt giữa giá trị thực tế và giá trị dự đoán.

Công thức:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y_i}|$$

Trong đó:

- y_i là giá trị thực tế.
- $\hat{y_i}$ là giá trị dự đoán.
- n là số lượng điểm dữ liệu.

2.3.2. RMSE

RMSE là viết tắt của Root Mean Squared Error, nghĩa là căn bậc hai của lỗi trung bình bình phương. Chỉ số này đo lường mức độ sai lệch trung bình giữa các giá trị dự đoán của mô hình và các giá trị thực tế, nhưng có sự khác biệt so với MAE là RMSE nhấn mạnh các sai lệch lớn hơn do việc sử dụng bình phương.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

2.3.3. R-squared

R-squared (R^2) còn được gọi là hệ số xác định (coefficient of determination). Chỉ số này đo lường mức độ mô hình dự đoán có thể giải thích được sự biến thiên của dữ liệu thực tế. Nói cách khác, nó cho biết tỷ lệ phần trăm biến thiên của giá trị thực tế có thể được giải thích bằng giá trị dự đoán của mô hình. R-squared có giá trị từ 0 đến 1, trong đó giá trị càng gần 1 thì mô hình càng tốt trong việc giải thích biến thiên của dữ liệu.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Trong đó $\bar{y}$ là giá trị trung bình của các giá trị thực tế.

2.4. Phương pháp lựa chọn tham số

Phương pháp lựa chọn tham số (hyperparameter tuning) là quá trình tìm kiếm các giá trị tối ưu cho các tham số không được học trực tiếp từ dữ liệu trong quá trình huấn luyện mô hình. Những tham số này có thể bao gồm learning rate, số lượng cây, chiều sâu tối đa của cây, số hàng xóm,…

2.4.1. Grid Search

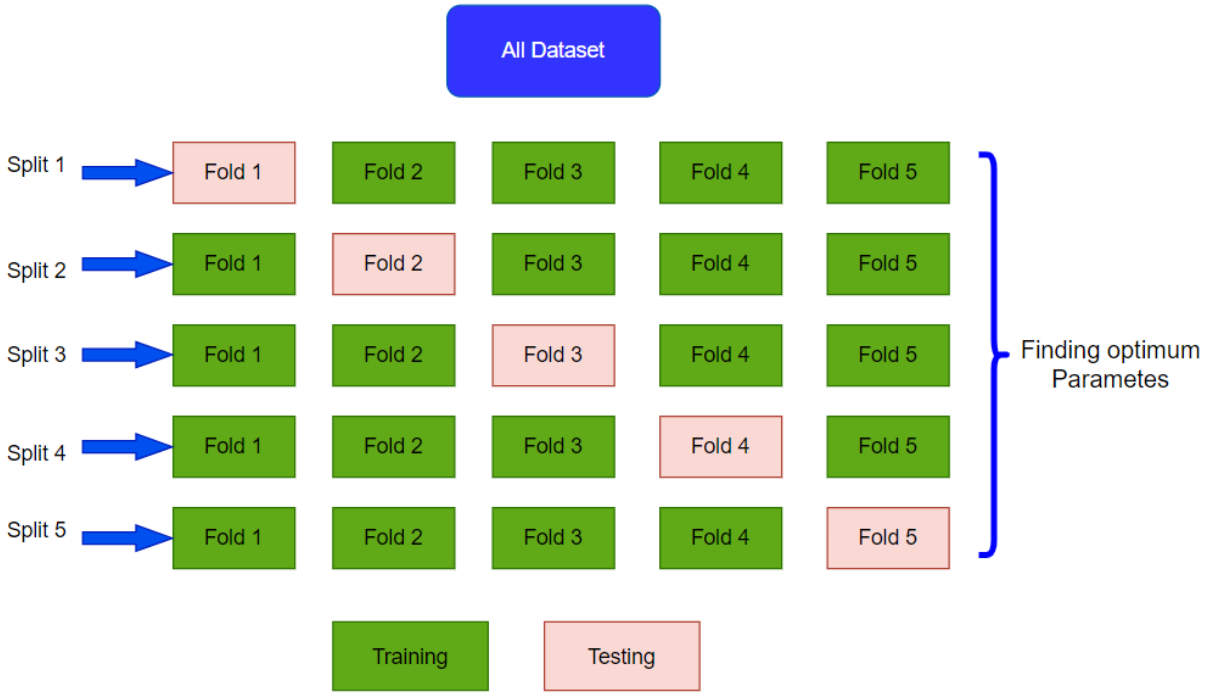
Grid Search là một phương pháp thử tất cả các kết hợp có thể của các giá trị tham số trong một phạm vi xác định trước. Mặc dù phương pháp này đơn giản và dễ hiểu, nhưng nó có thể rất tốn kém về mặt tính toán, đặc biệt là khi số lượng tham số lớn.

Ví dụ: Nếu ta có hai tham số với mỗi tham số có ba giá trị khả dĩ, Grid Search sẽ kiểm tra tất cả 3×3=9 kết hợp.

2.4.2. Cross-Validation

Cross-Validation là kỹ thuật đánh giá mô hình bằng cách chia dữ liệu thành nhiều phần và huấn luyện mô hình trên một phần, sau đó kiểm tra trên phần còn lại. Quá trình này lặp lại nhiều lần với các phần dữ liệu khác nhau để đảm bảo mô hình không bị overfitting và giúp lựa chọn tham số một cách chính xác hơn.

Ví dụ: 5-Fold Cross-Validation chia dữ liệu thành 5 phần, huấn luyện mô hình trên 4 phần và kiểm tra trên phần còn lại. Quá trình này lặp lại 5 lần và lấy trung bình các kết quả.



Hình 23: Ví dụ minh họa Cross-Validation

3. CHƯƠNG 3: GIỚI THIỆU VỀ TẬP DỮ LIỆU

3.1. Tổng quan về tập dữ liệu

Tập dữ liệu bao gồm thông tin về sản phẩm của 13 nhà bán phổ biến nhất trong lĩnh vực Sắc đẹp trên sàn thương mại điện tử Shopee. Nguyên nhân nhóm lựa chọn tập dữ liệu này vì theo thống kê của Metric thì Shopee luôn là sàn thương mại điện tử có doanh thu lớn nhất trong những năm vừa qua. Bên cạnh đó, cũng theo Metric, ngành hàng Sắc đẹp cũng là ngành hàng có doanh thu lớn nhất.

Dữ liệu sản phẩm được nhóm thu thập vào mỗi 23:30 pm hàng ngày trong khoảng thời gian từ ngày 10/3/2024 đến ngày 9/4/2024. Thông tin thu thập bao gồm:

Bảng 1: Mô tả dữ liệu sản phẩm

Tên trường	Mô tả
itemid	Mã định danh của sản phẩm
shopid	Mã định danh của cửa hàng
liked_count	Số lượng thích sản phẩm
discount	Phần trăm giảm giá của sản phẩm
shop_location	Địa chỉ kho hàng của sản phẩm
shop_rating	Trung bình đánh giá của cửa hàng
name	Tên sản phẩm
historical_sold	Số lượng sản phẩm đã bán
price	Giá sản phẩm
rating_star	Trung bình đánh giá sao
rating_count	Số lượng đánh giá sao
rcount_with_context	Số lượng đánh giá có bình luận
date	Ngày thu thập

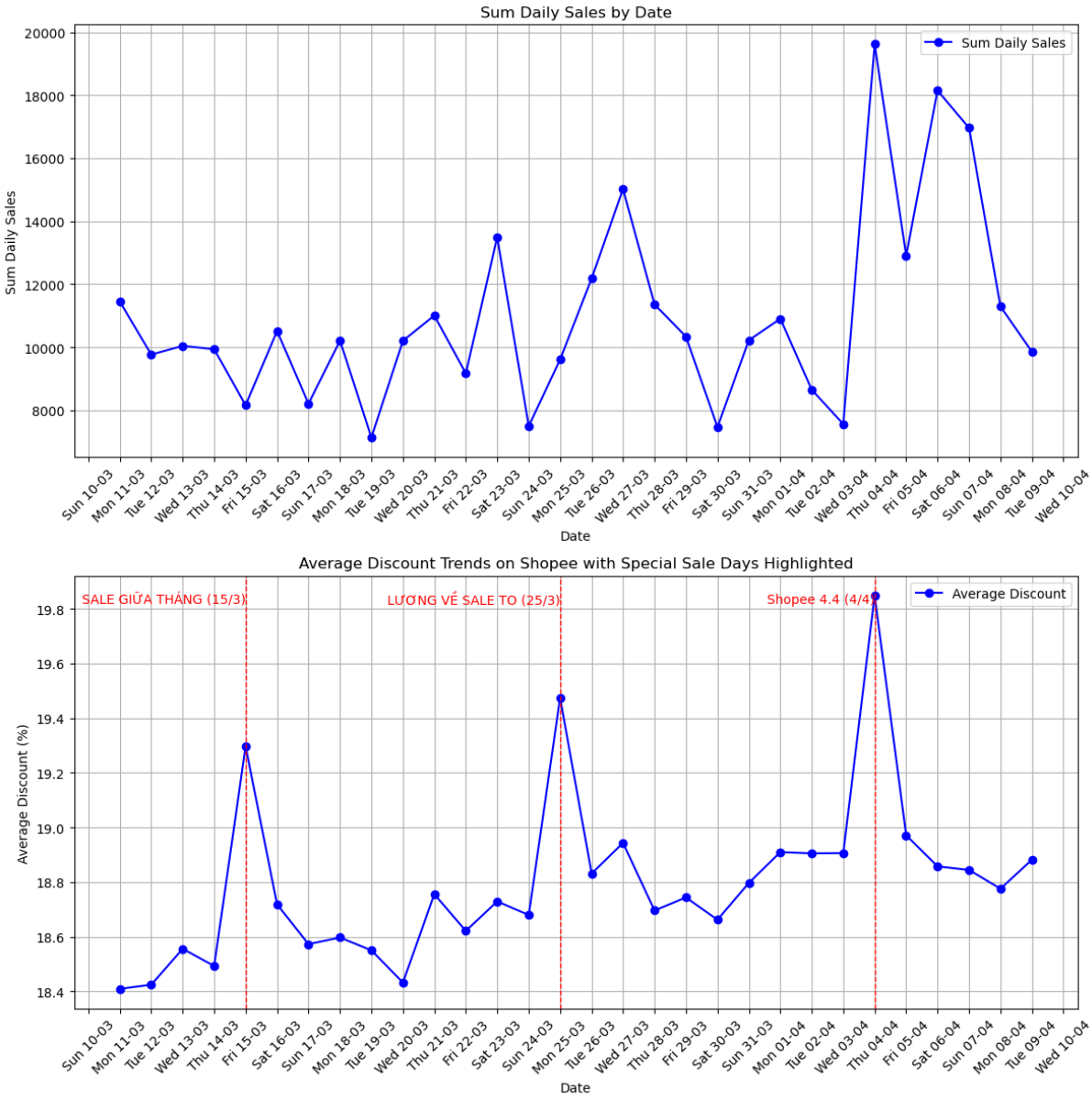
Ngoài dữ liệu về thông tin sản phẩm theo ngày, nhóm còn thu thập thêm chi tiết các bình luận về sản phẩm để sử dụng mô hình phân tích cảm xúc nhằm cải thiện hiệu quả của mô hình. Dữ liệu bình luận chi tiết được thu thập gồm các cột:

Bảng 2: Mô tả dữ liệu bình luận về sản phẩm

Tên trường	Mô tả
comment	Bình luận về sản phẩm
itemid	Mã định danh của sản phẩm được bình luận
date	Ngày bình luận
cmtid	Mã định danh của bình luận
rating_star	Đánh giá sao tương ứng với bình luận

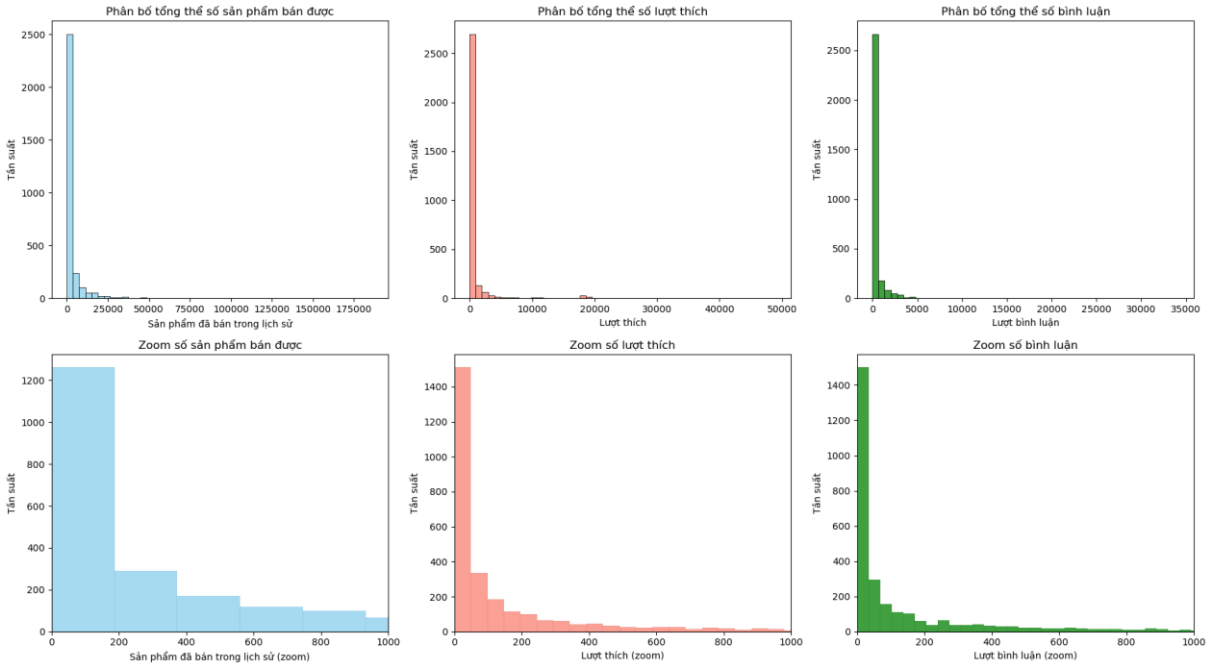
3.2. Phân tích khám phá sơ bộ về tập dữ liệu

Số lượng sản phẩm bán được hằng ngày được tính bằng số lượng sản phẩm bán được trong ngày hôm sau trừ số lượng sản phẩm bán được trong ngày hôm trước.



Hình 24: Biểu đồ biến động số lượng bán và trung bình tỉ lệ giảm giá hàng ngày

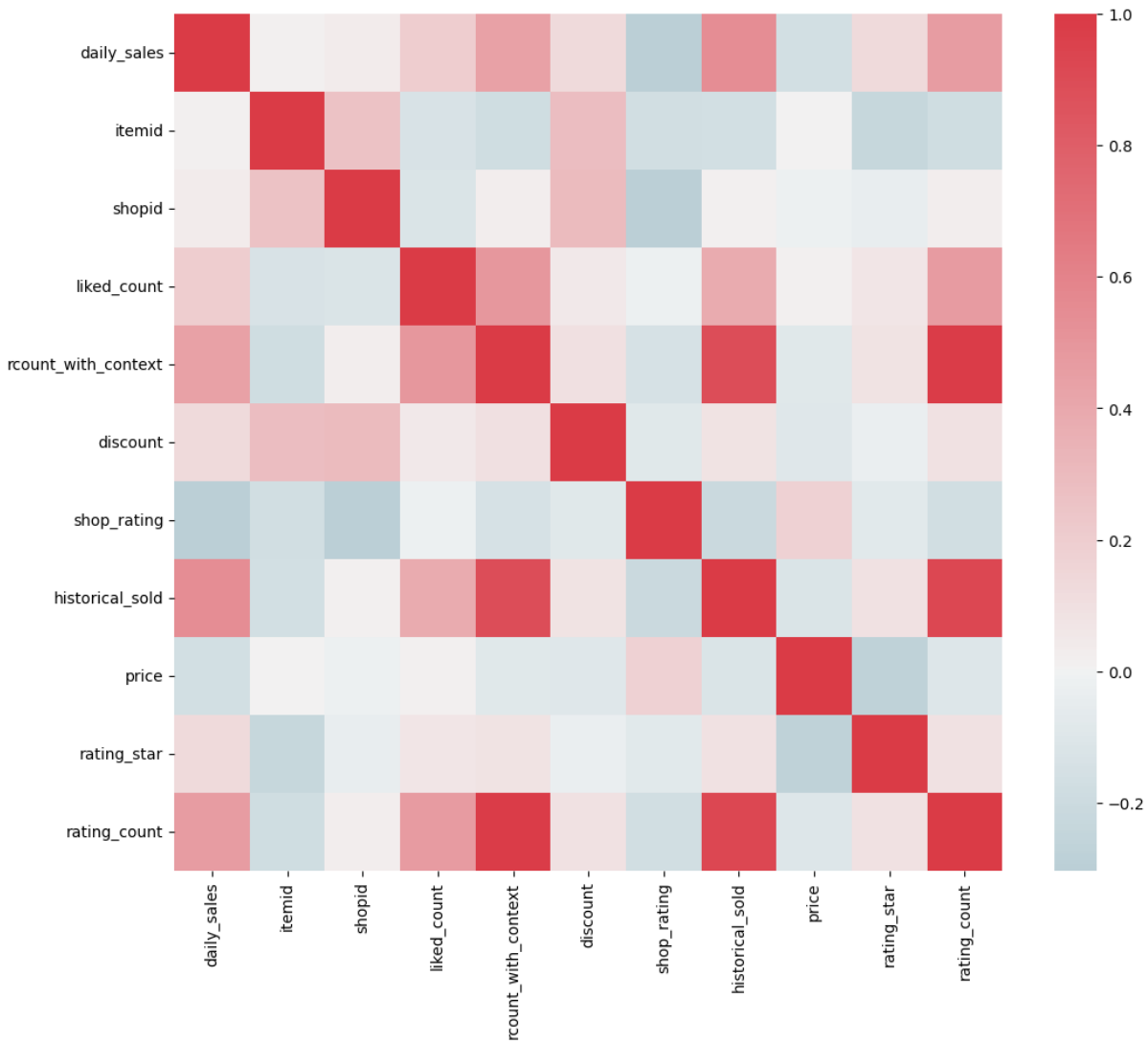
Qua biểu đồ, ta thấy mặc dù ngày 15/3 và 25/3 là 2 ngày giảm giá lớn của Shopee, nhưng lượng sản phẩm bán được cũng không có sự biến đổi lớn. Trong khi đó, ngày 06/4 và 07/4 không phải là ngày giảm giá lớn nhưng lại có lượng sản phẩm bán được khá cao. Chỉ có ngày 04/4 là ngày giảm giá nhiều nhất và bán được nhiều nhất là có thể giải thích được.



Hình 25: Biểu đồ phân bố số sản phẩm bán được, số lượt thích và số bình luận

Nhận xét:

- Cả 3 biểu đồ có phân phối lệch phải mạnh. Nghĩa là phần lớn sản phẩm có lịch sử bán không quá cao, tương tự lượt thích và bình luận cũng không nhiều.
- Biểu đồ chi tiết cho thấy sự tương đồng về hình dạng của số lượt thích và số lượng bình luận.



Hình 26: Biểu đồ tương quan các biến

Nhận xét:

- Số lượng sản phẩm bán được hằng ngày có tương quan đến số lượng đánh giá sao, số lượng bán trong lịch sử, số lượng bình luận, và số lượng thích.
- Số lượng bình luận có tương quan đến số lượng đánh giá sao và số lượng bán trong lịch sử.
- Mã định danh sản phẩm và mã định danh cửa hàng không có tương quan cao đến các đặc trưng khác.

3.3. Những khó khăn gặp phải trong quá trình trích xuất đặc trưng

Trích xuất đặc trưng (Feature Engineering) là một quá trình quan trọng trong xây dựng mô hình học máy. Trong quá trình này, nhóm tác giả đã gặp nhiều trường hợp kết quả thực tế khác với dự kiến. Những kinh nghiệm này sẽ giúp những ai muốn xây dựng hệ thống tương tự tiết kiệm được thời gian và công sức.

3.3.1. Những dữ liệu là quà tặng

Trong danh sách sản phẩm được thu thập, chúng tôi phát hiện một số lượng sản phẩm chỉ dùng cho mục đích quà tặng, không phải sản phẩm bán ra. Những sản phẩm này nếu không được phát hiện và loại bỏ sẽ trở thành những dữ liệu nhiễu và gây ra kết quả dự đoán không tốt. Sau khi nhận ra điều này, chúng tôi đã loại bỏ những sản phẩm này dựa vào tên của sản phẩm.

```
keywords_pattern = r'\[HB GIFT\]|\[Quà tặng\]|\[Gift\]|\[HC Gift\]|không bán|\[Tặng kèm\]'
df = df[~df['name'].str.contains(keywords_pattern, case=False, regex=True)]
print("Kích thước DataFrame sau khi lọc quà tặng:", df.shape)

✓ 0.1s
Kích thước DataFrame sau khi lọc quà tặng: (137969, 14)
```

Hình 27: Minh họa các lọc bỏ sản phẩm quà tặng

3.3.2. Tạo thêm các biến cửa sổ trượt

Các giá trị như thông tin về giá, số lượng bán trong lịch sử cũng là những yếu tố quan trọng cho mô hình dự đoán, các giá trị như này chúng tôi tạm gọi là các biến cửa sổ trượt. Để có được các biến này chúng ta phải chọn số ngày trượt phù hợp, nếu chọn số này quá lớn, dữ liệu chúng ta sẽ bị thu nhỏ lại (dữ liệu những ngày đầu, không có đủ số ngày trước nó sẽ bị thiếu giá trị này). Do tập dữ liệu trong đề tài chúng tôi chỉ thu thập trong vòng 1 tháng, nên phải lưu ý chọn số ngày trượt không quá cao. Ngoài ra, nếu số ngày quá nhỏ thì sẽ dẫn đến có ít thông tin được học khiến cho mô hình gặp nhiều sai số hơn. Trong trường hợp dữ liệu thu thập trong 30 ngày như đề tài, chúng tôi qua thử nghiệm đã sử dụng cửa sổ trượt trong 7 ngày.

Ngoài những giá trị cửa sổ trượt, chúng tôi còn sử dụng thêm những giá trị thống kê di động trong 7 ngày để cải thiện mô hình.

- rolling_min: giá trị tối thiểu trong 7 ngày
- rolling_max: giá trị tối đa trong 7 ngày
- rolling_mean: giá trị trung bình trong 7 ngày
- rolling_std: độ lệch chuẩn trong 7 ngày

3.3.3. Kết hợp dữ liệu bình luận

Ban đầu chúng tôi kỳ vọng sử dụng phân tích cảm xúc cho bình luận về sản phẩm sẽ giúp cải thiện mô hình đáng kể. Tuy nhiên, sau khi thử nghiệm, chúng tôi nhận ra những kết quả từ phân tích cảm xúc khá trùng lặp với đánh giá trung bình của sản phẩm, vì thế không mang lại sự cải thiện đáng kể cho mô hình. Các phương pháp đã được thử nghiệm bao gồm:

- Chỉ sử dụng tổng số bình luận tích cực.
- Chỉ sử dụng tổng số bình luận tiêu cực.
- Tính tổng bình luận tích cực, tiêu cực, trung tính.
- Tính tỉ lệ bình luận tích cực, tiêu cực, trung tính.
- Tính tổng bình luận tích cực, tiêu cực, trung tính trong 30 ngày gần nhất.
- Tính tỉ lệ bình luận tích cực, tiêu cực, trung tính trong 30 ngày gần nhất.
- Tính tổng bình luận tích cực, tiêu cực, trung tính trong 15 bình luận mới nhất.
- Tính tỉ lệ bình luận tích cực, tiêu cực, trung tính trong 15 bình luận mới nhất.

Qua quá trình thử nghiệm, kết quả cho thấy phương pháp thêm vào đặc trưng duy nhất là tổng số bình luận tích cực đem lại hiệu quả hơn những hướng tiếp cận khác. Chi tiết số liệu sẽ được trình bày ở phần kết quả thực nghiệm.

3.3.4. Tạo thêm biến phân nhóm

Kasun Bandara và các cộng sự trong bài báo "Sales Demand Forecast in E-commerce using a Long Short-Term Memory Neural Network Methodology" [5] đã đưa ra một cách tiếp cận để phân nhóm cho dữ liệu. Cụ thể, nhóm đầu tiên là nhóm những sản phẩm bán được ít hàng và có mật độ số ngày không bán được sản phẩm nào cao. Nhóm thứ 2 là những sản phẩm bán được nhiều hàng và có mật độ số ngày không bán được sản phẩm nào thấp. Nhóm thứ 3 là những sản phẩm còn lại.

Bảng 3: Ngưỡng phân nhóm sản phẩm

Nhóm	Xếp hạng số lượng bán	Xếp hạng mật độ không bán được hàng
1	Sales.quantile <= 0.33	Zero.sales.percentage.quantile >= 0.67
2	Sales.quantile >= 0.67	Zero.sales.percentage.quantile <= 0.33
3	Còn lại	

Để hiện thực cách phân loại này chúng tôi đã sử dụng mã nguồn như sau.

```python
# Xác định các ngưỡng cố định cho tổng số lượng bán hàng
grouped_df['sales_quantile'] = pd.qcut(grouped_df['sum_daily_sales'], q =3, labels=[1, 2, 3])


# Merge tỷ lệ không bán hàng vào grouped_df
grouped_df = grouped_df.merge(aggregated_in_train, on='itemid')

# Tính quantile cho tỷ lệ không bán hàng
grouped_df['no_sales_ratio_quantile'] = pd.qcut(grouped_df['no_sales_ratio_in_train'], q=3, labels=[1, 2, 3])


# Xác định Group-ID dựa trên điều kiện đã cho
def determine_group_id(row):
    if row['sales_quantile'] == 1 and row['no_sales_ratio_quantile'] == 3:
        return 1
    elif row['sales_quantile'] == 3 and row['no_sales_ratio_quantile'] == 1:
        return 2
    else:
        return 3

grouped_df['Group-ID'] = grouped_df.apply(determine_group_id, axis=1)
```

Python

Hình 28: Minh họa mã nguồn phân nhóm sản phẩm

Kết quả sau khi thực hiện phân nhóm, dữ liệu sẽ nhìn như sau.

```
grouped_df
```

Python

	itemid	sum_daily_sales	sales_quantile	no_sales_ratio_in_train	no_sales_ratio_quantile	Group-ID
0	188787501	90.0	3	0.083333	1	2
1	188787515	4.0	1	0.833333	2	3
2	188787770	13.0	2	0.625000	2	3
3	243602468	0.0	1	1.000000	3	1
4	243606506	1.0	1	0.958333	3	1
...	...	...	...	...	...	...
3050	25916972620	1.0	1	0.958333	3	1
3051	25950232881	9.0	2	0.708333	2	3
3052	25964607231	0.0	1	1.000000	3	1
3053	25966815409	0.0	1	1.000000	3	1
3054	25967215443	7.0	2	0.791667	2	3

3055 rows × 6 columns

Hình 29: Kết quả sau khi thực hiện phân nhóm sản phẩm

4. CHƯƠNG 4: KẾT QUẢ THỰC NGHIỆM

4.1. Kết quả kết hợp thêm phân tích cảm xúc

Sau khi phân loại cảm xúc văn bản bình luận về sản phẩm chúng tôi đã thử sử dụng thông tin này bằng nhiều cách. Và cách hiệu quả nhất trên mô hình chúng tôi thu được là sử dụng thêm duy nhất cột tổng số bình luận tích cực.

Kết quả cho thấy mô hình KNN được cải thiện rõ rệt nhất với RMSE giảm 0.1396 (tương đương 2.5%) và MAE giảm 0.0489 (2%). Trong khi đó, các mô hình liên quan đến cây quyết định đạt hiệu quả cải thiện không cao. Cụ thể, mô hình Cây quyết định có MAE gần như không đổi (chỉ cải thiện 0.000026) và RMSE chỉ giảm 0.0188 (0.33%). Mô hình Rừng ngẫu nhiên có RMSE giảm 0.0024 (0.04%) và MAE giảm 0.0024 (0,1%). Tương tự, mô hình XGBoost có RMSE giảm 0.0068 (0.13%) và MAE giảm 0.0016 (0.07%).

Nhìn chung, việc kết hợp phân loại cảm xúc vào mô hình dự đoán đã giúp cải thiện độ chính xác của các mô hình. Tuy nhiên, mức độ cải thiện này chưa thực sự đáng kể. Do đó, cần phải xem xét cân nhắc giữa hiệu suất cải thiện và chi phí phát sinh khi triển khai mô hình trong thực tế.

Bảng 4: Kết quả mô hình kết hợp phân tích cảm xúc

Thuật toán	RMSE			MAE		
	Các đặc trưng ban đầu	Thêm đặc trưng cảm xúc	Giảm RMSE	Các đặc trưng ban đầu	Thêm đặc trưng cảm xúc	Giảm MAE
Hồi quy tuyến tính	5.1038	5.082	0.0218	2.2598	2.231	0.0288
Cây quyết định	5.6191	5.6003	0.0188	2.370685	2.370659	0.000026
Rừng ngẫu nhiên	5.2051	5.2027	0.0024	2.2452	2.2428	0.0024
XGBoost	5.2082	5.2014	0.0068	2.2586	2.257	0.0016
KNN	5.6623	5.5227	0.1396	2.4313	2.3824	0.0489
LSTM	5.1714	5.1055	0.0659	2.2587	2.2422	0.0165

4.2. Kết quả cuối của mô hình

Trong quá trình thử nghiệm đào tạo và đánh giá mô hình, chúng tôi đã sử dụng phương pháp GridSearchCV để tìm kiếm thông số tối ưu cho từng mô hình. Ngoài thông số mô hình, chúng tôi còn so sánh trên những tập dữ liệu được xử lý khác nhau để tìm ra phương pháp phù hợp.

Sau khi đánh giá, chúng tôi đã xác định được tập dữ liệu quan trọng bao gồm các đặc trưng sau, tạm gọi là tập A trong bảng kết quả:

- historical_sold
- price
- rating_count
- rcount_with_context
- daily_sales
- daily_sales_lag_1d
- price_lag_1d
- daily_sales_lag_2d
- price_lag_2d
- daily_sales_lag_3d
- price_lag_3d
- daily_sales_lag_4d
- price_lag_4d
- daily_sales_lag_5d
- price_lag_5d
- daily_sales_lag_6d
- price_lag_6d
- daily_sales_lag_7d
- price_lag_7d
- rolling_mean_7d
- label_1_count

Ngoài ra, chúng tôi cũng tiến hành đào tạo và đánh giá trên tập dữ liệu đầy đủ bao gồm nhiều đặc trưng hơn để kiểm tra xem liệu việc thêm vào các yếu tố khác có cải

thiện được hiệu suất của mô hình hay không. Tập dữ liệu đầy đủ bao gồm các đặc trưng tạm gọi là B trong bảng kết quả:

- itemid
- shopid
- liked_count
- discount
- shop_location
- shop_rating
- historical_sold
- price
- rating_star
- rating_count
- rcount_with_context
- date_of_week
- day
- is_special_day
- future_special_day
- daily_sales
- daily_sales_lag_1d
- price_lag_1d
- daily_sales_lag_2d
- price_lag_2d
- daily_sales_lag_3d
- price_lag_3d
- daily_sales_lag_4d
- price_lag_4d
- daily_sales_lag_5d
- price_lag_5d
- daily_sales_lag_6d
- price_lag_6d

- daily_sales_lag_7d
- price_lag_7d
- rolling_min_7d
- rolling_max_7d
- rolling_mean_7d
- rolling_std_7d
- label_0_count
- label_1_count
- label_2_count
- enhanced_popularity_score
- min_discount_in_train
- discount_to_min_discount
- sales_quantile
- no_sales_ratio_quantile
- Group-ID

Ngoài ra chúng tôi cũng thử nghiệm các phương pháp chuẩn hóa cho tập dữ liệu xem liệu có nên chuẩn hóa không.

Kết quả cuối cùng chúng tôi đã xây dựng mô hình tốt nhất như bên dưới dựa trên đánh giá RMSE.

Bảng 5: Kết quả mô hình tốt nhất mỗi thuật toán

Thuật toán	Tập dữ liệu	Chuẩn hóa biến X	Chuẩn hóa biến Y	Thông số mô hình tối ưu	RMSE	MAE	R^2
Hồi quy tuyến tính	A	MinMax Scale	Không	{}	5.082	2.231	0.73
Cây quyết định	A	Không	Không	{'max_depth': 10, 'min_samples_leaf': 4, 'min_samples_split' : 10}	5.6	2.37	0.67
Rừng ngẫu nhiên	A	MinMax Scale	Không	{'max_depth': 10, 'min_samples_split' : 2, 'n_estimators': 100}	5.203	2.243	0.72
XGBoost	A	Không Scale và MinMax Scale bằng nhau	Không	{'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 50}	5.201	2.257	0.72
KNN	A	MinMax Scale	Không	{'algorithm': 'auto', 'metric': 'euclidean', 'n_neighbors': 3, 'weights': 'uniform'}	5.523	2.382	0.68
LSTM	A	MinMax Scale	MinMax Scale	{'units_1': 32, 'units_2': 64, 'units_3': 64, 'dropout': 0.2, 'loss': MSE 'optimizer':Adam}	5.106	2.242	0.726

Về thông số

Việc trích xuất đặc trưng (Feature engineering) đóng vai trò quan trọng trong xây dựng mô hình. Như Andrew Ng từng nhận định "Applied machine learning is basically feature engineering" (Tạm dịch "Học máy về cơ bản là trích xuất đặc trưng"). Trong đề tài này cũng vậy:

- Tất cả mô hình tốt nhất đều chỉ sử dụng những đặc trưng quan trọng để huấn luyện chứ không phải là toàn bộ đặc trưng.
- Phần lớn mô hình cho kết quả tốt khi sử dụng MinMaxScale cho đầu vào X. Một số cho kết quả tốt khi không sử dụng chuẩn hóa đầu vào X.
- Đa phần mô hình không chuẩn hóa đầu ra Y ngoại trừ thuật toán LSTM.

Về mô hình

- Hiệu suất các mô hình nếu xếp theo RMSE lần lượt được xếp hạng là cây quyết định, KNN, rừng ngẫu nhiên, XGBoost, LSTM và hồi quy tuyến tính.
- Xếp hạng thuật toán lấy ý tưởng từ cây tuân theo đúng sự phát triển của thuật toán, thuật toán đơn giản như cây quyết định, rồi tới rừng ngẫu nhiên và tốt nhất là XGBoost.
- Mô hình tốt nhất là LSTM và Hồi quy tuyến tính. Việc LSTM đưa lại hiệu suất cao đã được chứng minh trong nhiều nghiên cứu trước đây. Tuy nhiên, hồi quy tuyến tính mới là mô hình có hiệu quả nhất trong đề tài nghiên cứu của chúng tôi. Đây có thể là một cách tiếp cận đơn giản mà hiệu quả cho các bài toán dự đoán số lượng sản phẩm bán ra với dữ liệu có nhiều sản phẩm trong ngắn ngày.

PHẦN 3: KẾT LUẬN

1. Kết quả đạt được

- Hệ thống hóa lại kiến thức đã học và tìm hiểu thêm để nắm chắc nền tảng vững chắc lý thuyết về Học máy.
- Tìm hiểu và vận dụng mô hình ngôn ngữ lớn cho mục đích phân loại cảm xúc cải thiện hiệu quả mô hình dự đoán số lượng sản phẩm bán được.
- Tích lũy kinh nghiệm làm việc với tập dữ liệu thực tế, xây dựng mô hình và so sánh hiệu năng các mô hình.
- Rèn luyện nhiều kỹ năng quan trọng như: làm việc nhóm, quản lý thời gian và đặc biệt là kỹ năng nghiên cứu và đọc các bài báo tiếng Anh.

2. Hạn chế của đề tài

- Trong quá trình so sánh đánh giá đề tài, chúng tôi đã bỏ quên một chỉ số quan trọng đó là thời gian học và thời gian dự đoán của mô hình.
- Chưa có sự so sánh đánh giá giữa có mô hình phân loại cảm xúc. Chúng tôi sử dụng PhoBert vì tính phổ biến và những kết quả ấn tượng trong các nghiên cứu trước đây.

3. Hướng phát triển trong tương lai

- Thử nghiệm thêm nhiều mô hình phân loại cảm xúc để lựa chọn thuật toán phù hợp.
- Xây dựng và đánh giá các mô hình trên cùng một cấu hình máy để có thể đánh giá về hiệu suất tốc độ của các mô hình
- Tìm hiểu và xây dựng thêm các thuật toán dự đoán khác, thử nghiệm xây dựng mô hình học máy kết hợp (stacking ensemble).
- Xây dựng hệ thống hoàn chỉnh có thể tự thu thập dữ liệu và huấn luyện các mô hình có thể học tiệm tiến (Incremental learning) để tránh mô hình bị lỗi thời.

TÀI LIỆU THAM KHẢO

[1] Văn, L. B. (2024). Tác động của rủi ro bất lợi cảm nhận và lợi ích được cảm nhận đến ý định tiếp tục mua sắm trực tuyến của khách hàng trên sàn thương mại điện tử tại Việt Nam.

[2] Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2022). M5 accuracy competition: Results, findings, and conclusions. International Journal of Forecasting, 38(4), 1346-1364.

[3] Liu, Y. (2024). Sales Forecasting Based on Transformer-LSTM Model. Highlights in Science, Engineering and Technology, 85, 776-782.

[4] Li, M., Ji, S., & Liu, G. (2018). Forecasting of Chinese E-Commerce Sales: An Empirical Comparison of ARIMA, Nonlinear Autoregressive Neural Network, and a Combined ARIMA-NARNN Model. Mathematical Problems in Engineering, 2018(1), 6924960.

[5] Bandara, K., Shi, P., Bergmeir, C., Hewamalage, H., Tran, Q., & Seaman, B. (2019). Sales demand forecast in e-commerce using a long short-term memory neural network methodology. In Neural Information Processing: 26th International Conference, ICONIP 2019, Sydney, NSW, Australia, December 12–15, 2019, Proceedings, Part III 26 (pp. 462-474). Springer International Publishing.

[6] Nguyen, D. Q., & Nguyen, A. T. (2020). PhoBERT: Pre-trained language models for Vietnamese. arXiv preprint arXiv:2003.00744.

[7] Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. Science, 349(6245), 255-260.

[8] Sarker, I. H. (2021). Machine learning: Algorithms, real-world applications and research directions. SN computer science, 2(3), 160.

[9] Beattie, A. (2020). Business forecasting: Understanding the basics. Dosegljivo, 22(5), 2020.

[10] Panikkar, R., Saleh, T., Szybowski, M., & Whiteman, R. (2021). Operationalizing machine learning in processes. McKinsey, September.

[11] Bevan, O., Freiman, M., Pasricha, K., Samandari, H., & White, O. (2019). Transforming risk efficiency and effectiveness. McKinsey & Company.

[12] Latifah, R., Wulandari, E. S., & Kreshna, P. E. (2019). Model Decision Tree Untuk Prediksi Jadwal Kerja Menggunakan Scikit-Learn. Prosiding Semnastek.

[13] Breiman, L. (2001). Random forests. Machine learning, 45, 5-32.

[14] Pangarkar, D. J., Sharma, R., Sharma, A., & Sharma, M. (2020). Assessment of the different machine learning models for prediction of cluster bean (Cyamopsis tetragonoloba L. Taub.) yield. Advances in Research, 21(9), 98-105.

[15] Song, C. E., Li, Y., Ramnani, A., Agrawal, P., Agrawal, P., Jang, S. J., ... & Kang, M. (2024, April). 52.5 TOPS/W 1.7 GHz Reconfigurable XGBoost Inference Accelerator Based on Modular-Unit-Tree with Dynamic Data and Compute Gating. In 2024 IEEE Custom Integrated Circuits Conference (CICC) (pp. 1-2). IEEE.

[16] Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2021). Dive into deep learning. arXiv preprint arXiv:2106.11342.

[17] Hamad, Z., & Abdulrahman, I. (2022). Deep learning-based load forecasting considering data reshaping using MATLAB\Simulink. International Journal of Energy and Environmental Engineering, 13(2), 853-869.

[18] Le, X. H., Ho, H. V., Lee, G., & Jung, S. (2019). Application of long short-term memory (LSTM) neural network for flood forecasting. Water, 11(7), 1387.

[19] Olah, C. (2015). Understanding lstm networks.

[20] Sayeed, M. S., Mohan, V., & Muthu, K. S. (2023). BERT: A Review of Applications in Sentiment Analysis. HighTech and Innovation Journal, 4(2), 453-462.

[21] Geetha, M. P., & Renuka, D. K. (2021). Improving the performance of aspect based sentiment analysis using fine-tuned Bert Base Uncased model. International Journal of Intelligent Networks, 2, 64-69.

[22] Acheampong, F. A., Nunoo-Mensah, H., & Chen, W. (2021). Transformer models for text-based emotion detection: a review of BERT-based approaches. Artificial Intelligence Review, 54(8), 5789-5829.

[23] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[24] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.

[25] Jana, S., Biswas, R., Pal, K., Biswas, S., & Roy, K. (2024). The evolution and impact of large language model systems: A comprehensive analysis.

[26] Athinaios, K., Chalkidis, I., Pantazi, D. A., & Papaloukas, C. (2020). Named entity recognition using a novel linguistic model for greek legal corpora based on BERT model (Doctoral dissertation, BS Thesis, School of Science, Department of Informatics and Telecommunications).

[27] Shreyashree, S., Sunagar, P., Rajarajeswari, S., & Kanavalli, A. (2022). A literature review on bidirectional encoder representations from transformers. Inventive Computation and Information Technologies: Proceedings of ICICIT 2021, 305-320.

[28] Nguyen, D. Q., & Nguyen, A. T. (2020). PhoBERT: Pre-trained language models for Vietnamese. arXiv preprint arXiv:2003.00744.

[29] Tun, N. S., Long, N. N., Tran, T., Thao, N. T., Phuong, D. T., & Nguyen, T. (2021). Stock article title sentiment-based classification using PhoBERT. In CEUR Workshop Proceedings (Vol. 3026, pp. 225-233).

[30] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.