

TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC TP. HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN

KHÓA LUẬN TỐT NGHIỆP

NGHIÊN CỨU KỸ THUẬT PHÂN TÍCH VÀ TIỀN XỬ LÝ DỮ LIỆU TRONG CÁC BÀI TOÁN MÁY HỌC VÀ KHAI THÁC DỮ LIỆU

GIẢNG VIÊN HƯỚNG DẪN : TS. Trần Minh Thái
SINH VIÊN THỰC HIỆN: Trần Ngọc Thảo Ngân – 21DH114460
Nguyễn Mai Khánh Vy – 21DH114581

TP. HỒ CHÍ MINH - THÁNG 08 - NĂM 2024

TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC TP. HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN

KHÓA LUẬN TỐT NGHIỆP

NGHIÊN CỨU KỸ THUẬT PHÂN TÍCH VÀ TIỀN XỬ LÝ DỮ LIỆU TRONG CÁC BÀI TOÁN MÁY HỌC VÀ KHAI THÁC DỮ LIỆU

GIẢNG VIÊN HƯỚNG DẪN : TS. Trần Minh Thái
SINH VIÊN THỰC HIỆN: Trần Ngọc Thảo Ngân – 21DH114460
Nguyễn Mai Khánh Vy – 21DH114581

TP. HỒ CHÍ MINH - THÁNG 08 - NĂM 2024

Lời cảm ơn

Trong quá trình thực hiện đề tài "Nghiên cứu kỹ thuật phân tích và tiền xử lý dữ liệu trong các bài toán máy học và khai thác dữ liệu" chúng em đã nhận được sự giúp đỡ, hướng dẫn, của các thầy/cô. Chúng em xin gửi lời cảm ơn chân thành và sâu sắc nhất tới tất cả các thầy/cô đã góp phần giúp chúng em hoàn thành khóa luận một cách trọn vẹn.

Trước tiên, chúng em xin được gửi lời tri ân đến thầy Trần Minh Thái, người đã trực tiếp hướng dẫn và đồng hành cùng chúng em trong suốt quá trình thực hiện đề tài. Với kiến thức, kinh nghiệm phong phú và tâm huyết truyền đạt, thầy đã giúp chúng em nhận được vô vàn bài học và kinh nghiệm quý báu, tạo tiền đề vững chắc để hoàn thành tốt đề tài. Những lời chỉ bảo ân cần, nhận xét của thầy chính là nguồn động lực vô cùng to lớn, thôi thúc chúng em nỗ lực hết mình để hoàn thành tốt đề tài này.

Chúng em cũng xin gửi lời cảm ơn chân thành tới các thầy cô trong những người đã tận tình truyền đạt kiến thức chuyên môn, rèn luyện kỹ năng và định hướng phương pháp nghiên cứu cho chúng em. Những bài giảng đầy tâm huyết và những chia sẻ kinh nghiệm quý báu đã giúp chúng em có được nền tảng vững chắc để thực hiện đề tài này.

Một lần nữa, chúng em xin gửi lời cảm ơn chân thành và sâu sắc nhất tới tất cả các thầy cô đã hướng dẫn và hỗ trợ chúng em hoàn thành tốt đề tài "Nghiên cứu kỹ thuật phân tích và tiền xử lý dữ liệu trong các bài toán máy học và khai thác dữ liệu".

Lời cam đoan

Chúng em xin cam đoan rằng báo cáo khóa luận tốt nghiệp với đề tài "Nghiên cứu kỹ thuật phân tích và tiền xử lý dữ liệu trong các bài toán máy học và khai thác dữ liệu" là công trình nghiên cứu của riêng chúng em, được thực hiện dưới sự hướng dẫn của Thầy Trần Minh Thái.

Các số liệu, kết quả nghiên cứu trong báo cáo này là trung thực và chưa từng được công bố trong bất kỳ công trình nghiên cứu nào khác.

Tất cả các tài liệu tham khảo, nguồn thông tin trích dẫn đều được ghi rõ nguồn gốc và tác giả theo đúng quy định.

Chúng em hoàn toàn chịu trách nhiệm về tính trung thực và chính xác của nội dung báo cáo này.

Chúng em xin chân thành cảm ơn!

MỤC LỤC

MỤC LỤC	III
DANH MỤC CÁC KÝ HIỆU, CHỮ VIẾT TẮT VÀ THUẬT NGỮ	VIII
DANH MỤC HÌNH ẢNH	IX
DANH MỤC BẢNG BIỂU	XI
CHƯƠNG 1. MỞ ĐẦU	1
1.1. Giới thiệu đề tài	1
1.2. Mục tiêu và nội dung của đề tài.....	1
1.3. Giới hạn đề tài	2
1.4. Cấu trúc báo cáo	2
1.5. Tổng kết chương.....	3
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT	4
2.1. Tổng quan về tiền xử lý dữ liệu.....	4
2.2. Sơ lược về máy học và khai thác dữ liệu.....	4
2.3. Các loại dữ liệu	5
2.3.1. Dữ liệu số (Numerical Data)	5
2.3.2. Dữ liệu phân loại (Categorical Data)	6
2.3.3. Dữ liệu văn bản (Textual Data).....	6
2.3.4. Dữ liệu chuỗi thời gian (Time Series Data)	6
2.4. Tổng quan về Python.....	7
2.5. Các thư viện hỗ trợ phân tích và tiền xử lý dữ liệu trong python	7
2.5.1. Pandas.....	7
2.5.2. NumPy.....	8
2.5.3. Matplotlib	8
2.5.4. Seaborn	9
2.5.5. Scikit-learn	9

2.6. Các vấn đề phổ biến trong dữ liệu.....	9
2.6.1. Dữ liệu bị thiếu.....	9
2.6.2. Dữ liệu bị trùng	10
2.6.3. Dữ liệu ngoại lệ	11
2.6.4. Dữ liệu mất cân bằng	12
2.6.5. Trích chọn đặc trưng (Feature Selection).....	12
2.6.6. Dữ liệu đa chiều	12
2.6.7. Mã hóa dữ liệu phân loại (encoding categorical data)	13
2.6.8. Feature scaling.....	13
2.7. Xử lý dữ liệu thiếu.....	13
2.7.1. Single imputation	14
2.7.2. Multiple Imputation (MI)	19
2.8. Xử lý dữ liệu trùng	20
2.8.1. Loại bỏ dữ liệu trùng	20
2.8.2. Gộp dữ liệu trùng	20
2.9. Các phương pháp trích chọn đặc trưng (Feature Selection).....	20
2.9.1. Filter method	20
2.9.2. Wrapper method	21
2.9.3. Embedded methods	21
2.10. Xử lý dữ liệu mất cân bằng	22
2.10.1. Down Sampling.....	22
2.10.2. Up Sampling.....	22
2.11. Xử lý dữ liệu ngoại lệ	25
2.11.1. Loại bỏ các ngoại lệ (Removing outliers)	25
2.11.2. Giới hạn ngoại lệ bằng IQR (Outlier Capping Using IQR)	26

2.11.3. Giới hạn ngoại lệ sử dụng giá trị trung bình và độ lệch chuẩn (Outlier Capping Using Mean and Std).....	26
2.11.4. Giới hạn ngoại lệ bằng giá trị phân vị (Outlier Capping Using Quantiles)	27
2.11.5. Giới Hạn Ngoại Lệ Bằng Giá Trị Tùy Chỉnh (Capping Outliers Using Custom Values)	28
2.12. Các kỹ thuật mã hóa dữ liệu	29
2.12.1. One-hot encoding	29
2.12.2. Label Encoding	29
2.12.3. Frequency Encoding.....	30
2.12.4. Ordinal Encoding.	30
2.12.5. Mean Encoding.	31
2.13. Chuẩn hóa dữ liệu.....	32
2.13.1. Standardization (Z-score Normalization).....	32
2.13.2. Min/Max Scaling.....	32
2.13.3. Mean Normalization.....	33
2.13.4. Maximum Absolute Scaling.....	34
2.14. Giảm số lượng thuộc tính	34
2.15. Phân tích và tiền xử lý dữ liệu văn bản (Textual Data).....	39
2.15.1. Các thư viện hỗ trợ	39
2.15.2. Chuẩn bị dữ liệu	41
2.15.3. Loại bỏ từ dừng (stop word)	41
2.15.4. Phân tích thành phần từng loại từ (POS).....	42
2.15.5. Thực hiện stemming và lemmatization	43
2.15.6. Phân tích Ngrams	44
2.15.7. Tạo word clouds	44
2.15.8. Vector hóa dữ liệu (TF-IDF).....	46

2.16. Phân tích và tiền xử lý dữ liệu chuỗi thời gian (Time-series Data).....	50
2.16.1. Các yếu tố cần lưu ý khi phân tích dữ liệu chuỗi thời gian.....	50
2.16.2. Sử dụng biểu đồ đường và biểu đồ hộp để trực quan hóa dữ liệu chuỗi thời gian	52
2.16.3. Nhận diện các mẫu trong chuỗi thời gian	54
2.16.4. Thực hiện phân tách dữ liệu chuỗi thời gian.....	55
2.16.5. Thực hiện làm mịn dữ liệu - trung bình di động (Moving Average)	57
2.16.6. Thực hiện làm mịn dữ liệu - làm mịn theo cấp số nhân (Exponential Moving Average)	59
2.16.7. Thực hiện kiểm tra tính dừng trên dữ liệu chuỗi thời gian	59
2.16.8. Tính sai phân cho dữ liệu chuỗi thời gian	60
2.16.9. Thư viện hỗ trợ.....	60
2.17. Các độ đo đánh giá mô hình	61
2.17.1. Mean Absolute Error (MAE)	61
2.17.2. Mean Squared Error (MSE)	61
2.17.3. Root Mean Square Error (RMSE).....	61
2.17.4. R-squared (R^2).....	62
2.17.5. Adjusted R-Squared	63
2.17.6. Accuracy.....	63
2.17.7. Precision	63
2.17.8. Recall.....	64
2.17.9. F1-Score	64
2.18. Tổng kết chương.....	64
CHƯƠNG 3. PHÂN TÍCH BÀI TOÁN	65
3.1. Mô tả tập dữ liệu bài toán.....	65
3.1.1. Bài toán 1: Xây dựng mô hình dự đoán giá nhà ở khu vực TP. Hồ Chí Minh. 65	

3.1.2. Bài toán 2: Xây dựng mô hình dự đoán bệnh nhân có mắc bệnh tiểu đường hay không	66
3.1.3. Bài toán 3: Phân loại thư rác (Spam Email).....	67
3.1.4. Bài toán 4: Phân tích và dự đoán giá tiền ảo Bitcoin	67
3.2. Quy trình thực hiện bài toán	68
3.2.1. Bài toán 1: Xây dựng mô hình dự đoán giá nhà ở khu vực TP. Hồ Chí Minh. 68	
3.2.2. Bài toán 2: Xây dựng mô hình dự đoán bệnh nhân có mắc bệnh tiểu đường hay không	68
3.2.3. Bài toán 3: Phân loại thư rác (Spam Email).....	68
3.2.4. Bài toán 4: Phân tích và dự đoán giá tiền ảo Bitcoin	69
3.3. Tổng kết chương	69
CHƯƠNG 4. KẾT QUẢ THỰC NGHIỆM.....	70
4.1. Yêu cầu phần cứng, phần mềm.	70
4.2. Quy trình thực hiện trên Python.	70
4.2.1. Bài toán 1: Xây dựng mô hình dự đoán giá nhà ở khu vực TP. Hồ Chí Minh. 70	
4.2.2. Bài toán 2: Xây dựng mô hình dự đoán bệnh nhân có mắc bệnh tiểu đường hay không	81
4.2.3. Bài toán 3: Phân loại thư rác (Spam email)	90
4.2.4. Bài toán 4: Phân tích và dự đoán giá tiền ảo Bitcoin	99
4.3. Tổng kết chương	109
KẾT LUẬN.....	110
Kết luận.....	110
Hướng phát triển tương lai.	110
TÀI LIỆU THAM KHẢO	111

DANH MỤC CÁC KÝ HIỆU, CHỮ VIẾT TẮT VÀ THUẬT NGỮ

STT	Từ viết tắt	Nội dung
1	KNN	K-Nearest Neighbors
2	LSTM	Long Short-Term Memory
3	MA	Moving Average
4	MACD	Moving Average Convergence Divergence
5	MAE	Mean Absolute Error
6	MNB	Multinomial Naïve Bayes
7	MSE	Mean Squared Error
8	NLP	Natural Language Processing
9	NLTK	Natural Language Toolkit
10	NSC	NUS SMS Corpus
11	NUS	National University of Singapore
12	PCA	Principal Component Analysis
13	POS	Part of Speech
14	R2	R-squared
15	RFR	Random Forest Regressor
16	RMSE	Root Mean Square Error
17	SGD	Stochastic Gradient Descent
18	SMOTE	Synthetic Minority Over-sampling Technique
19	SMS	Short Message Service
20	SVC	Support Vector Classifier
21	SVM	Support Vector Machine
22	SVR	Support Vector Regression
23	TF-IDF	Term Frequency-Inverse Document Frequency

DANH MỤC HÌNH ẢNH

Hình 2.1. Nhãn các loại từ trong tiếng anh	43
Hình 2.2. Top 20 từ có tần suất xuất hiện nhiều nhất trong đoạn văn	45
Hình 2.3. Hình ảnh Word Cloud được tạo từ đoạn văn	46
Hình 2.4. Biểu đồ đường phân tích chuỗi thời gian của lưu lượng hành khách tại San Francisco	52
Hình 2.5. Biểu đồ hộp thể hiện lưu lượng hành khách tại San Francisco qua các quý trong năm	53
Hình 2.6. Biểu đồ ví dụ Moving Average.....	58
Hình 3.1. Sơ đồ quy trình thực hiện bài toán 1	68
Hình 3.2. Sơ đồ quy trình thực hiện bài toán 2	68
Hình 3.3. Sơ đồ quy trình thực hiện bài toán 3	69
Hình 3.4. Sơ đồ quy trình thực hiện bài toán 4	69
Hình 4.1. Hiển thị 5 dòng dữ liệu đầu tiên	70
Hình 4.2. Dữ liệu mẫu sau khi biến đổi dữ liệu của 5 dòng đầu tiên.	72
Hình 4.3: Một phần các giá trị trùng trong tập dữ liệu.	73
Hình 4.4. Biểu đồ Scatter trực quan mối quan hệ giữa các biến số với giá trị “Gia”	73
Hình 4.5. Biểu đồ hộp trước khi xử lý ngoại lệ	74
Hình 4.6. Biểu đồ hộp sau khi xử lý ngoại lệ	74
Hình 4.7. Biểu đồ Scatter trực quan mối quan hệ giữa các biến số với cột giá “Gia”	75
Hình 4.8. Biểu đồ heatmap thể hiện sự tương quan.....	75
Hình 4.9: Biểu đồ cột giữa các cột dữ liệu phân loại và cột Gia.	76
Hình 4.10. Kết quả sau khi thực hiện Feature Selection.....	78
Hình 4.11. Biểu đồ scatter thể hiện mối quan hệ giữa giá trị thực tế và giá trị dự đoán của mô hình SVR và RFR	81
Hình 4.12. Hiển thị 5 dòng dữ liệu.	82
Hình 4.13. Thống kê tóm tắt dữ liệu.....	82
Hình 4.14. Các giá trị unique của cột age.....	83
Hình 4.15. Biểu đồ Histogram của các cột dữ liệu số	83
Hình 4.16: Biểu đồ trực quan Gender và Smoking History	84
Hình 4.17: Biểu đồ heatmap thể hiện sự tương quan giữa các biến.	84

Hình 4.18: Các giá trị trùng nhau.....	85
Hình 4.19. Biểu đồ hộp trước khi xử lý ngoại lệ.....	86
Hình 4.20: Kết quả sau khi Feature Scaling.....	87
Hình 4.21. Biểu đồ cột của cột Outcome.....	87
Hình 4.22. Biểu đồ cột của thuộc tính diabetes sau khi xử lý mất cân bằng.....	88
Hình 4.23: Kết quả thực hiện Feature Selection.....	88
Hình 4.24. Năm dòng đầu của tập dữ liệu	91
Hình 4.25. Dữ liệu sau khi cắt bỏ các cột không có giá trị phân tích và đổi tên cột	91
Hình 4.26. Kết quả sau khi mã hóa.....	91
Hình 4.27. Kết quả kiểm tra dữ liệu null	92
Hình 4.28. Biểu đồ tròn thể hiện phần trăm của ham và spam.....	92
Hình 4.29. Kết quả sau khi mở rộng từ viết tắt.....	93
Hình 4.30. Kết quả thực hiện chuẩn hóa dữ liệu bằng thư viện nltk	94
Hình 4.31. Kết quả chuẩn hóa dữ liệu bằng thư viện spacy	94
Hình 4.32. Dữ liệu sau khi chuẩn hóa.....	95
Hình 4.33. Word cloud của dữ liệu spam	95
Hình 4.34. Biểu đồ word cloud của ham	96
Hình 4.35. Năm dòng dữ liệu đầu tiên của tập dữ liệu ban đầu	99
Hình 4.36. Năm dòng đầu của dữ liệu btc sau khi xử lý	100
Hình 4.37. Tóm tắt thống kê cho tập dữ liệu	100
Hình 4.38. Kết quả kiểm tra dữ liệu null	101
Hình 4.39. Biểu đồ đường của giá đóng Bitcoin	101
Hình 4.40. Biểu đồ hộp thể hiện giá đóng của Bitcoin.....	102
Hình 4.41. Biểu đồ phân tích mùa vụ theo giá đóng Bitcoin.....	102
Hình 4.42. Kết quả kiểm định tính dừng	103
Hình 4.43. Kết quả sau khi thực hiện Moving Average	104
Hình 4.44. Kết quả sau khi scaling dữ liệu	104
Hình 4.45. Kiến trúc mô hình LSTM.....	106
Hình 4.46. Đồ thị hàm loss của mô hình	107
Hình 4.47. Biểu đồ so sánh kết quả dự đoán, giá trị thực tế và dự báo 30 ngày tiếp theo cho giá Bitcoin	109

DANH MỤC BẢNG BIỂU

Bảng 2.1 Trước khi Mean Imputation	14
Bảng 2.2 Sau khi thực hiện Mean Imputation	15
Bảng 2.3 Trước khi thực hiện Median Imputation	16
Bảng 2.4 Sau khi thực hiện Median Imputation	17
Bảng 2.5. Ví dụ về Mode Imputation	18
Bảng 2.6. Dữ liệu mẫu	23
Bảng 2.7. Khoảng cách Euclidean giữa các mẫu thiểu số.	24
Bảng 2.8. Sau khi thực hiện SMOTE	25
Bảng 2.9. Sau khi One-hot encoding	29
Bảng 2.10. Ví dụ về Label Encoding	29
Bảng 2.11. Ví dụ về frequency encoding.....	30
Bảng 2.12. Ordinal encoding	31
Bảng 2.13. Ví dụ Mean Encoding.....	31
Bảng 2.14. Standardization cho tập dữ liệu.	32
Bảng 2.15. Min max scaling cho tập dữ liệu.	33
Bảng 2.16. Mean normalization cho tập dữ liệu.....	33
Bảng 2.17. Maximum absolute scaling.....	34
Bảng 2.18. Dữ liệu mẫu.	36
Bảng 2.19. Sau khi chuẩn hóa.....	36
Bảng 2.20. Số lần xuất hiện của các từ trong 3 documents	47
Bảng 2.21. TF của cả 3 Document.....	48
Bảng 2.22. Kết quả tính IDF.....	48
Bảng 2.23. Kết quả tính TF-IDF.....	49
Bảng 2.24. Ví dụ Bag of Words.....	49
Bảng 2.25. Doanh số hàng tháng của một doanh nghiệp.....	56
Bảng 2.26. Trung bình động 3 tháng	56
Bảng 2.27. Biến động mùa vụ.....	56
Bảng 2.28. Dư lượng.....	57
Bảng 2.29. Số liệu bán hàng của một doanh nghiệp từ 2000 đến 2009.....	58
Bảng 3.1. Bảng mô tả dữ liệu bài toán 4.....	67

Bảng 4.1. Thông tin về tập dữ liệu.....	71
Bảng 4.2. Tóm tắt thống kê cho tập dữ liệu.....	71
Bảng 4.3. Tỷ lệ phần trăm các giá trị null.....	72
Bảng 4.4. Kết quả sau khi xử lý các giá trị null.....	73
Bảng 4.5: Kết quả sau khi mã hóa.	77
Bảng 4.6: Sau khi Feature Scaling.....	78
Bảng 4.7: Kết quả của mô hình khi sử dụng PCA và không sử dụng PCA	79
Bảng 4.8: So sánh kết quả giữa SVR và RandomForest.	80
Bảng 4.9. Thông tin về dữ liệu.	82
Bảng 4.10: Sau khi xử lý các giá trị null.	85
Bảng 4.11: Sau khi mã hóa	86
Bảng 4.12: Kết quả khi dùng LDA và không dùng LDA.....	89
Bảng 4.13: Kết quả của mô hình Logistic Regression.....	89
Bảng 4.14: Kết quả của mô hình Decision Tree.....	89
Bảng 4.15: Kết quả của mô hình Random Forest.....	90
Bảng 4.16: Tổng quan 3 mô hình.	90
Bảng 4.17. Thông tin của tập dữ liệu sau khi loại bỏ dữ liệu trùng lặp.....	92
Bảng 4.18. Kết quả đếm tần suất xuất hiện của các từ tri-ngrams	96
Bảng 4.19. Kết quả đánh giá mô hình SVC.....	98
Bảng 4.20. Kết quả đánh giá mô hình MultinomialNB	98
Bảng 4.21. Kết quả đánh giá mô hình Logistic Regression.....	98
Bảng 4.22. Kết quả đánh giá mô hình RandomForest.....	98
Bảng 4.23. Bảng so sánh hiệu suất các mô hình.....	98
Bảng 4.24. Bảng thông tin về tập dữ liệu	100
Bảng 4.25. Bảng so sánh kết quả dự đoán mô hình với các độ dài chuỗi khác nhau	105
Bảng 4.26. Kết quả dự báo trong 30 ngày tiếp theo	108

CHƯƠNG 1. MỞ ĐẦU

Chương này giới thiệu về đề tài nghiên cứu, mục tiêu và nội dung chính của đề tài, giới hạn của nghiên cứu, và cấu trúc của báo cáo. Đề tài tập trung vào các kỹ thuật phân tích và tiền xử lý dữ liệu trong các bài toán máy học và khai thác dữ liệu, nhằm giải quyết các vấn đề liên quan đến chất lượng dữ liệu và cải thiện hiệu suất của các mô hình học máy.

1.1. Giới thiệu đề tài

Trong thời đại ngày nay, dữ liệu đóng vai trò vô cùng quan trọng trong mọi lĩnh vực của đời sống. Việc khai thác và phân tích dữ liệu một cách hiệu quả là chìa khóa để đưa ra những quyết định chiến lược, cải tiến quy trình hoạt động, và tối ưu hóa nguồn lực. Tuy nhiên, dữ liệu thô thường chứa nhiều vấn đề như dữ liệu bị thiếu, dữ liệu nhiễu, dữ liệu không nhất quán, và nhiều vấn đề khác. Điều này đòi hỏi phải có những kỹ thuật phân tích và tiền xử lý dữ liệu hiệu quả để chuẩn bị dữ liệu cho các bài toán máy học và khai thác dữ liệu.

Với đề tài "Nghiên cứu kỹ thuật phân tích và tiền xử lý dữ liệu trong các bài toán máy học và khai thác dữ liệu" chúng em tập trung vào việc nghiên cứu và áp dụng các kỹ thuật để giải quyết các vấn đề liên quan đến dữ liệu bao gồm: xử lý dữ liệu bị thiếu, chuẩn hóa dữ liệu, loại bỏ dữ liệu nhiễu, và các vấn đề khác. Việc nghiên cứu này sẽ cung cấp một nền tảng vững chắc cho các bài toán máy học và khai thác dữ liệu, giúp nâng cao chất lượng và độ chính xác của các mô hình dự đoán và phân tích.

1.2. Mục tiêu và nội dung của đề tài

- Mục tiêu:

Nghiên cứu và đánh giá các kỹ thuật phân tích và tiền xử lý dữ liệu hiện đại trong các bài toán máy học và khai thác dữ liệu.

Thiết kế và triển khai các thuật toán, mô hình và công cụ để xử lý dữ liệu hiệu quả.

Đánh giá hiệu suất và tính khả thi của các kỹ thuật đề xuất thông qua các nghiên cứu trường hợp và thử nghiệm trên dữ liệu thực tế.

- Nội dung:

Phân tích và nghiên cứu các vấn đề phổ biến trong dữ liệu như dữ liệu bị thiếu, dữ liệu nhiễu, dữ liệu không nhất quán.

Nghiên cứu và áp dụng các kỹ thuật tiền xử lý dữ liệu như xử lý dữ liệu bị thiếu, chuẩn hóa dữ liệu, loại bỏ dữ liệu nhiễu,...

Phát triển các thuật toán và mô hình để phân tích và xử lý dữ liệu một cách hiệu quả. Sử dụng các công cụ hỗ trợ trong quá trình xử lý và phân tích dữ liệu.

Thực hiện thử nghiệm trên các bộ dữ liệu thực tế để kiểm chứng hiệu quả và tính khả thi của các kỹ thuật đã triển khai. Đánh giá hiệu suất của các giải pháp xử lý dữ liệu qua các nghiên cứu tình huống cụ thể và kết quả thực nghiệm.

1.3. Giới hạn đề tài

Đề tài này tập trung vào các kỹ thuật phân tích và tiền xử lý dữ liệu cho các bài toán máy học và khai thác dữ liệu. Tuy nhiên, đề tài sẽ không đi sâu vào các vấn đề liên quan đến việc lưu trữ và quản lý dữ liệu.

Ngoài ra, đề tài chủ yếu nghiên cứu và áp dụng các kỹ thuật tiền xử lý dữ liệu cho dữ liệu dữ liệu số, dữ liệu văn bản và dữ liệu chuỗi thời gian. Các kỹ thuật xử lý dữ liệu văn bản, hình ảnh, âm thanh, và video sẽ không được đề cập sâu trong phạm vi nghiên cứu này.

1.4. Cấu trúc báo cáo

Báo cáo khóa luận này bao gồm năm chương chính, mỗi chương được cấu trúc để trình bày một cách rõ ràng và chi tiết các khía cạnh của đề tài. Dưới đây là cấu trúc chi tiết của báo cáo:

Chương 1: Mở đầu

Chương này giới thiệu về đề tài, bao gồm lý do lựa chọn đề tài, mục tiêu và nội dung của nghiên cứu, giới hạn của đề tài và mô tả cấu trúc của báo cáo.

Chương 2: Tổng quan

Chương này cung cấp một cái nhìn tổng quan về các loại dữ liệu (dữ liệu số, dữ liệu văn bản và dữ liệu chuỗi thời gian).

Chương 3: Cơ sở lý thuyết

Chương này đi sâu vào các phương pháp phân tích và tiền xử lý dữ liệu. Nó mô tả các kỹ thuật cụ thể như làm sạch dữ liệu, chuyển đổi dữ liệu và tạo tính năng kỹ thuật. Đồng thời trình bày các phương pháp phân tích dữ liệu cho từng loại dữ liệu cụ thể.

Chương 4: Phân tích bài toán

Chương này mô tả chi tiết các tập dữ liệu và các bài toán nghiên cứu. Nó bao gồm các bước thực hiện và phân tích kết quả cho từng bài toán cụ thể như xây dựng mô hình dự đoán giá nhà, dự đoán bệnh nhân có mắc bệnh tiểu đường, phân loại thư rác, và phân tích dự đoán giá tiền ảo Bitcoin.

Chương 5: Kết quả thực nghiệm

Chương này trình bày các kết quả thực nghiệm, bao gồm yêu cầu phần cứng và phần mềm, quy trình thực hiện trên Python, và phân tích chi tiết kết quả của từng bài toán đã nêu trong chương 4.

Kết luận

Nội dung phần này tóm tắt các kết quả chính của nghiên cứu, nêu bật những đóng góp của đề tài và đề xuất các hướng phát triển trong tương lai. Nó kết thúc báo cáo bằng việc tổng kết những gì đã đạt được và những gì có thể cải thiện hoặc tiếp tục nghiên cứu trong lĩnh vực này.

1.5. Tổng kết chương

Nội dung của phần này tóm tắt các kết quả chính của nghiên cứu, nêu bật những đóng góp của đề tài để đề xuất các hướng phát triển trong tương lai. Nó kết thúc báo cáo bằng việc tổng kết những gì đã đạt được và những gì có thể cải thiện hoặc tiếp tục nghiên cứu trong lĩnh vực này.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

Chương này cung cấp cái nhìn tổng quan về các khái niệm cơ bản liên quan đến tiền xử lý dữ liệu, máy học, khai thác dữ liệu, và các loại dữ liệu khác nhau. Bên cạnh đó, chương này cũng giới thiệu về Python và các thư viện quan trọng hỗ trợ cho việc phân tích và tiền xử lý dữ liệu như Pandas, NumPy, Matplotlib, Seaborn, và Scikit-learn. Ngoài ra, chương còn đi sâu vào các vấn đề thường gặp trong dữ liệu, bao gồm dữ liệu bị thiếu, trùng lặp, ngoại lệ và mất cân bằng, cùng với các phương pháp xử lý tương ứng như trích chọn đặc trưng, mã hóa dữ liệu phân loại, chuẩn hóa dữ liệu và giảm số lượng thuộc tính. Cuối cùng, chương cũng đề cập đến các kỹ thuật phân tích và tiền xử lý dữ liệu văn bản, chuỗi thời gian, cùng với các độ đo đánh giá mô hình.

2.1. Tổng quan về tiền xử lý dữ liệu

Tiền xử lý dữ liệu là quá trình áp dụng các kỹ thuật nhằm nâng cao chất lượng dữ liệu và từ đó giúp nâng cao chất lượng kết quả khai phá. Dữ liệu thô thu thập ban đầu thường không hoàn chỉnh, thiếu các giá trị, thuộc tính quan trọng, hoặc chỉ chứa các dữ liệu gộp nhóm, các giá trị nhiễu, lệch quá xa ra ngoài phạm vi mong đợi, hay không nhất quán. Lý do là vì kích thước dữ liệu quá lớn và được thu thập từ nhiều nguồn khác nhau, dẫn đến chất lượng dữ liệu thấp, từ đó cho ra những kết quả khai phá không tốt. Chính vì vậy, tiền xử lý dữ liệu là một bước rất quan trọng trong việc giải quyết bất kỳ vấn đề nào trong Machine Learning, Data Mining.

2.2. Sơ lược về máy học và khai thác dữ liệu

Máy học là một lĩnh vực của trí tuệ nhân tạo (AI) tập trung vào việc phát triển các thuật toán cho phép máy tính tự động "học" từ dữ liệu mà không cần được lập trình cụ thể. Các thuật toán này có thể học các mẫu và mối quan hệ trong dữ liệu, từ đó đưa ra dự đoán hoặc quyết định cho các tình huống mới.

Khai thác dữ liệu là quá trình trích xuất thông tin có giá trị từ một tập dữ liệu lớn. Quá trình này bao gồm các bước như thu thập dữ liệu, làm sạch dữ liệu, chuẩn bị dữ liệu, phân tích dữ liệu và mô hình hóa dữ liệu. Khai thác dữ liệu có thể được sử dụng để tìm hiểu các xu hướng, dự đoán tương lai và đưa ra quyết định sáng suốt hơn.

Máy học và khai thác dữ liệu có mối quan hệ mật thiết với nhau. Khai thác dữ liệu cung cấp dữ liệu cần thiết để huấn luyện các mô hình học máy, trong khi máy học cung cấp các thuật toán mạnh mẽ để phân tích dữ liệu và trích xuất thông tin có giá trị.

Máy học và khai thác dữ liệu có nhiều ứng dụng trong các lĩnh vực: Tài chính, y tế, quản lý chuỗi cung ứng, marketing, giáo dục,...

Phân loại các bài toán máy học: gồm 2 loại.

- Học có giám sát (Supervised Learning): Huấn luyện mô hình từ dữ liệu đã được gán nhãn, để dự đoán các mục tiêu mới. Ví dụ: Phân loại, Hồi quy.
- Học không giám sát (Unsupervised Learning): Khám phá mẫu và cấu trúc ẩn trong dữ liệu chưa được gán nhãn. Ví dụ: Gom cụm.

Quy trình khai thác dữ liệu:

- + Thu thập dữ liệu.
- + Tiền xử lý và làm sạch dữ liệu.
- + Xây dựng mô hình.
- + Đánh giá mô hình.
- + Triển khai mô hình.

2.3. Các loại dữ liệu

Dữ liệu có thể có nhiều dạng, nhưng các mô hình học máy, khai phá dữ liệu dựa trên bốn kiểu dữ liệu chính. Dữ liệu trong Học máy, khai phá dữ liệu bao gồm dữ liệu số, dữ liệu phân loại, dữ liệu chuỗi thời gian và dữ liệu văn bản.

2.3.1. Dữ liệu số (Numerical Data)

Dữ liệu số là dữ liệu có thể được đo lường hoặc đếm, chẳng hạn như tuổi, chiều cao, cân nặng, thu nhập, nhiệt độ, v.v. Dữ liệu số có thể được phân loại thành dữ liệu rời rạc (discrete data) và dữ liệu liên tục (continuous data). Dữ liệu rời rạc là hữu hạn hoặc có thể đếm được, chẳng hạn như số lượng con cái, số lượng xe hơi, v.v. Dữ liệu liên tục là vô hạn hoặc có thể đo lường được, chẳng hạn như chiều cao, cân nặng, thời gian, v.v. Dữ liệu số có thể được xử lý bằng cách áp dụng các kỹ thuật khác nhau, chẳng hạn như chuẩn hóa, tiêu chuẩn hóa, phân nhóm, phát hiện giá trị ngoại lai, v.v. Những kỹ thuật này giúp giảm phạm vi, phương sai, độ lệch và nhiễu của dữ liệu số và làm cho chúng trở nên so sánh và tương thích hơn.

2.3.2. Dữ liệu phân loại (Categorical Data)

Dữ liệu phân loại là dữ liệu có thể được nhóm vào các danh mục hoặc lớp, chẳng hạn như giới tính, màu sắc, nghề nghiệp, quốc gia, v.v. Dữ liệu phân loại có thể được phân loại thành dữ liệu định danh và dữ liệu thứ tự. Dữ liệu định danh là không có thứ tự và không có thứ hạng nội tại, chẳng hạn như giới tính, màu sắc, quốc gia, v.v. Dữ liệu thứ tự là có thứ tự và có một số thứ hạng nội tại, chẳng hạn như trình độ học vấn, mức thu nhập, mức độ hài lòng, v.v. Dữ liệu phân loại có thể được xử lý bằng cách áp dụng các kỹ thuật khác nhau, chẳng hạn như mã hóa, ánh xạ, biến giả, mã hóa One-Hot, v.v. Những kỹ thuật này giúp chuyển đổi dữ liệu phân loại thành các giá trị số hoặc nhị phân có thể được sử dụng bởi các thuật toán khai phá dữ liệu.

2.3.3. Dữ liệu văn bản (Textual Data)

Dữ liệu văn bản là dữ liệu bao gồm các từ, câu, đoạn văn, tài liệu, v.v., chẳng hạn như đánh giá, tweet, email, bài báo, v.v. Dữ liệu văn bản có thể được phân loại thành dữ liệu có cấu trúc và không có cấu trúc. Dữ liệu có cấu trúc được tổ chức và có định dạng được định trước, chẳng hạn như bảng, biểu mẫu, cơ sở dữ liệu, v.v. Dữ liệu không có cấu trúc là không tổ chức và không có định dạng được định trước, chẳng hạn như tệp văn bản, hình ảnh, video, v.v. Dữ liệu văn bản có thể được xử lý bằng cách áp dụng các kỹ thuật khác nhau, chẳng hạn như tokenization, stemming, lemmatization, loại bỏ từ dừng, tần suất từ - nghịch đảo tần suất tài liệu (TF-IDF), word embedding, v.v. Những kỹ thuật này giúp trích xuất thông tin liên quan, đặc trưng và ý nghĩa từ dữ liệu văn bản và làm cho chúng trở nên phù hợp hơn cho các nhiệm vụ khai phá dữ liệu.

2.3.4. Dữ liệu chuỗi thời gian (Time Series Data)

Dữ liệu chuỗi thời gian là dữ liệu được thu thập theo thời gian và có thứ tự thời gian, chẳng hạn như giá cổ phiếu, dữ liệu thời tiết, dữ liệu bán hàng, v.v. Dữ liệu chuỗi thời gian có thể được phân loại thành dữ liệu đơn biến và dữ liệu đa biến. Dữ liệu đơn biến bao gồm một biến thay đổi theo thời gian, chẳng hạn như nhiệt độ, lượng mưa, v.v. Dữ liệu đa biến bao gồm nhiều hơn một biến thay đổi theo thời gian, chẳng hạn như nhiệt độ và độ ẩm, doanh số và lợi nhuận, v.v. Dữ liệu chuỗi thời gian có thể được xử lý bằng cách áp dụng các kỹ thuật khác nhau, chẳng hạn như làm mịn, phân tách, phân biệt, làm cho ổn định, tự tương quan, dự báo, v.v. Những kỹ thuật này giúp phân tích các mẫu, xu hướng, chu kỳ và

tính mùa vụ của dữ liệu chuỗi thời gian và làm cho chúng trở nên phù hợp hơn cho các nhiệm vụ khai phá dữ liệu.

2.4. Tổng quan về Python

Python là một ngôn ngữ lập trình được thông dịch, hướng đối tượng và cấp cao. Đây là một trong những ngôn ngữ dễ học nhất và rất hữu ích trong ngành công nghiệp phần mềm, được sử dụng rộng rãi trong lập trình cạnh tranh, phát triển web, và tạo phần mềm. Python có cú pháp đơn giản, phù hợp cho người mới bắt đầu và có nhu cầu ngày càng tăng trong các lĩnh vực công nghệ hiện đại như Khoa học dữ liệu, Học máy, và Tự động hóa.

Lịch sử: Python được Guido van Rossum tạo ra vào những năm 1980 và phiên bản đầu tiên ra mắt năm 1991. Phiên bản mới nhất là Python 3.11, phát hành năm 2022.

Đặc điểm nổi bật: Python dễ đọc, hiểu, là ngôn ngữ thông dịch, hướng đối tượng, miễn phí và mã nguồn mở, đa nền tảng, có nhiều thư viện và framework, và có cộng đồng lớn và năng động.

Ưu điểm: Dễ học, linh hoạt, mã nguồn mở, hỗ trợ thư viện phong phú, cộng đồng mạnh. **Nhược điểm:** Hạn chế trong thiết kế, không hiệu quả về bộ nhớ, yếu trong tính toán di động, tốc độ thực thi chậm.

Ứng dụng: Python được sử dụng trong phát triển web, khoa học dữ liệu, web scraping, tự động hóa, thiết kế CAD, trí tuệ nhân tạo, học máy, và phát triển game. Các framework phổ biến như Django và Flask hỗ trợ phát triển web, còn các thư viện như Pandas và TensorFlow hỗ trợ AI và ML.

2.5. Các thư viện hỗ trợ phân tích và tiền xử lý dữ liệu trong python

2.5.1. Pandas

Giới thiệu: Pandas là một thư viện mã nguồn mở cung cấp các cấu trúc dữ liệu và công cụ phân tích dữ liệu mạnh mẽ cho Python. Thư viện này đặc biệt hữu ích cho việc xử lý và phân tích dữ liệu có cấu trúc (như bảng dữ liệu).

Tính năng chính:

- **DataFrame:** Cấu trúc dữ liệu chính của Pandas, giống như một bảng tính Excel hoặc một bảng SQL, rất tiện lợi để thao tác dữ liệu. DataFrame là một cấu trúc dữ liệu dạng bảng 2 chiều có thể chứa các kiểu dữ liệu khác nhau.

- **Series:** Là một mảng một chiều có thể chứa bất kỳ kiểu dữ liệu nào (giống như một cột trong bảng). Series có thể được tạo ra từ danh sách, từ điển hoặc từ một mảng NumPy.
- **Đọc và ghi dữ liệu:** Pandas hỗ trợ nhiều định dạng tệp như CSV, Excel, SQL, JSON, HDF5, và nhiều định dạng khác, giúp việc nhập và xuất dữ liệu trở nên dễ dàng.
- **Xử lý dữ liệu:** Pandas cung cấp các phương thức linh hoạt cho việc lọc, nhóm, tổng hợp, nối và biến đổi dữ liệu. Các thao tác này có thể thực hiện một cách nhanh chóng và hiệu quả trên các tập dữ liệu lớn.
- **Quản lý thời gian:** Pandas có hỗ trợ mạnh mẽ cho dữ liệu thời gian và ngày tháng, bao gồm các hàm để thao tác và phân tích dữ liệu theo chuỗi thời gian.

2.5.2. NumPy

Giới thiệu: NumPy (Numerical Python) là một thư viện cốt lõi cho tính toán khoa học trong Python. Nó cung cấp một đối tượng mảng mạnh mẽ, giúp thực hiện các thao tác toán học và logic trên các mảng lớn và đa chiều.

Tính năng chính:

- **Mảng n-dimensional:** NumPy hỗ trợ các mảng nhiều chiều với hiệu suất cao. Các mảng này có thể là một chiều, hai chiều (ma trận) hoặc nhiều chiều hơn.
- **Hàm toán học:** NumPy cung cấp nhiều hàm toán học cho các phép tính trên mảng, như phép tính đại số tuyến tính, biến đổi Fourier, thống kê và nhiều hàm toán học khác.
- **Tích hợp với các thư viện khác:** Nhiều thư viện khác như Pandas, SciPy và scikit-learn được xây dựng trên NumPy, giúp tăng cường tính tương thích và hiệu suất.

2.5.3. Matplotlib

Giới thiệu: Matplotlib là một thư viện vẽ đồ thị và trực quan hóa dữ liệu trong Python. Nó rất hữu ích cho việc tạo ra các biểu đồ từ đơn giản đến phức tạp.

Tính năng chính:

- **Biểu đồ 2D:** Matplotlib hỗ trợ nhiều loại biểu đồ như biểu đồ đường, biểu đồ cột, biểu đồ tán xạ, biểu đồ tròn và nhiều loại biểu đồ khác.
- **Tùy chỉnh cao:** Thư viện cho phép tùy chỉnh mọi yếu tố của biểu đồ, bao gồm màu sắc, nhãn trực, tiêu đề, lưới và nhiều yếu tố khác.

- **Tích hợp với Pandas:** Matplotlib có thể trực tiếp vẽ đồ thị từ DataFrame của Pandas, giúp trực quan hóa dữ liệu một cách dễ dàng và hiệu quả.

2.5.4. Seaborn

Giới thiệu: Seaborn là một thư viện xây dựng trên Matplotlib, được thiết kế để trực quan hóa dữ liệu một cách dễ dàng và đẹp mắt hơn.

Tính năng chính:

- **Biểu đồ thống kê:** Seaborn hỗ trợ các biểu đồ thống kê như biểu đồ phân phôi, biểu đồ hộp, biểu đồ nhiệt, biểu đồ đôi, và nhiều loại biểu đồ khác.
- **Dễ sử dụng:** Seaborn có cú pháp đơn giản và dễ sử dụng, cho phép tạo ra các biểu đồ phức tạp với ít dòng mã.
- **Tích hợp với Pandas:** Seaborn hoạt động tốt với DataFrame của Pandas, giúp trực quan hóa dữ liệu một cách dễ dàng và hiệu quả.

2.5.5. Scikit-learn

Giới thiệu: Scikit-learn là một thư viện mạnh mẽ và dễ sử dụng cho học máy và khai thác dữ liệu trong Python. Nó cung cấp các công cụ cho phân loại, hồi quy, clustering, và nhiều phương pháp học máy khác.

Tính năng chính:

- **Phân loại:** Scikit-learn hỗ trợ nhiều thuật toán phân loại như SVM, cây quyết định, k-NN, và nhiều thuật toán khác, giúp xây dựng các mô hình dự đoán và phân loại.
- **Hồi quy:** Thư viện cung cấp các thuật toán hồi quy như hồi quy tuyến tính, hồi quy logistic, hồi quy Ridge, và nhiều thuật toán hồi quy khác.
- **Clustering:** Hỗ trợ các thuật toán clustering như K-means, DBSCAN, và nhiều thuật toán khác, giúp nhóm các điểm dữ liệu lại với nhau dựa trên sự tương đồng.
- **Tiền xử lý:** Thư viện cung cấp các công cụ để chuẩn hóa và biến đổi dữ liệu, bao gồm các phương pháp chuẩn hóa, điều chỉnh dữ liệu và giảm chiều dữ liệu.
- **Đánh giá mô hình:** Scikit-learn cung cấp các công cụ để đánh giá mô hình, bao gồm các phương pháp chia dữ liệu thành tập huấn luyện và kiểm tra, các chỉ số đánh giá hiệu suất và các phương pháp kiểm định chéo.

2.6. Các vấn đề phổ biến trong dữ liệu

2.6.1. Dữ liệu bị thiếu

Trong phân tích dữ liệu, chúng ta thường gặp phải tình huống dữ liệu bị thiếu (missing data). Việc hiểu cơ chế gây ra dữ liệu bị thiếu là rất quan trọng để xác định cách xử lý phù hợp. Có ba loại cơ chế chính gây ra dữ liệu bị thiếu:

Missing Completely at Random (MCAR): Dữ liệu bị thiếu hoàn toàn ngẫu nhiên, không có mối liên hệ với các biến trong tập dữ liệu hoặc với giá trị của chính biến đó. Ví dụ, một số mẫu bị mất do sự cố máy móc hoặc do nhập liệu sai. MCAR là trường hợp lý tưởng nhất vì nó không gây ra bất kỳ sai lệch nào trong phân tích thống kê. Phân tích dữ liệu có thể được tiến hành một cách chính xác bằng cách loại bỏ các quan sát bị thiếu. [1] [2] [3]

Missing at Random (MAR): Dữ liệu bị thiếu có tương quan với một hoặc nhiều biến khác trong tập dữ liệu, nhưng không phụ thuộc vào chính biến bị thiếu. Ví dụ, Trong một cuộc khảo sát về thu nhập và tuổi tác, nếu những người trẻ tuổi có xu hướng không trả lời câu hỏi về thu nhập của họ. MAR là một trường hợp khá phổ biến và có thể được xử lý thông qua các phương pháp như Multiple Imputation hoặc Maximum Likelihood Estimation để dự đoán và thay thế các giá trị thiếu. [1] [2] [3]

Missing Not at Random (MNAR): Dữ liệu thiếu phụ thuộc vào chính biến bị thiếu hoặc các yếu tố không quan sát được. Ví dụ, trong một nghiên cứu về cân nặng, người béo phì có xu hướng không báo cáo cân nặng của họ. MNAR là trường hợp phức tạp nhất vì nó gây ra sai lệch nghiêm trọng trong phân tích. Để xử lý MNAR, thường cần sử dụng các phương pháp phức tạp hơn như phân tích mô hình hoặc các giả định về cơ chế thiếu dữ liệu. [1] [2] [3]

2.6.2. Dữ liệu bị trùng

Là những bản ghi trong tập dữ liệu có chứa thông tin giống nhau, một phần hoặc toàn bộ, trên hai hoặc nhiều trường. Dữ liệu trùng lặp có thể xuất hiện do nhiều nguyên nhân như lỗi nhập liệu, sao chép dữ liệu, hoặc tích hợp dữ liệu từ nhiều nguồn khác nhau.

Dữ liệu trùng lặp có thể gây ra các vấn đề như:

- Giảm chất lượng dữ liệu: Dữ liệu trùng lặp có thể làm sai lệch kết quả phân tích và báo cáo, dẫn đến quyết định sai lầm.
- Tốn thời gian và chi phí: Việc xử lý dữ liệu trùng lặp tốn nhiều thời gian và chi phí cho các tổ chức.

- Gây khó khăn trong việc quản lý dữ liệu: Dữ liệu trùng lặp có thể khiến việc quản lý dữ liệu trở nên khó khăn hơn, ảnh hưởng đến hiệu quả hoạt động của tổ chức.

2.6.3. Dữ liệu ngoại lệ

Dữ liệu ngoại lệ là những giá trị khác biệt rõ rệt so với các giá trị khác trong tập dữ liệu. Những giá trị này có thể nằm ngoài phạm vi dự kiến của các giá trị thông thường và có thể gây ảnh hưởng lớn đến các phân tích và mô hình thống kê.

Có 2 loại ngoại lệ chính: Global outlier và contextual outlier.

- **Global outlier (ngoại lệ toàn cầu):** là những điểm dữ liệu biệt lập, cách xa phần chính của dữ liệu. Chúng thường dễ dàng xác định và loại bỏ. Ví dụ: Chiều cao trung bình của người trưởng thành là khoảng từ 150 cm đến 200 cm, một người có chiều cao 250 cm sẽ được coi là một ngoại lệ toàn cầu vì chiều cao này nằm ngoài phạm vi thông thường của dữ liệu.
- **Contextual outlier(ngoại lệ theo ngữ cảnh):** là các điểm dữ liệu được coi là ngoại lệ chỉ trong một ngữ cảnh cụ thể, nhưng không phải là ngoại lệ trong các ngữ cảnh khác. Điều này có nghĩa là dữ liệu chỉ trở nên bất thường khi xét đến một số yếu tố hoặc điều kiện xung quanh. Ví dụ: Doanh số bán hàng tăng vọt vào tháng 12 cho các cửa hàng bán lẻ thường không phải là ngoại lệ vì đây là mùa mua sắm lớn nhất trong năm. Tuy nhiên, nếu doanh số của một cửa hàng vật liệu xây dựng tăng vọt vào tháng 12, điều này có thể là một ngoại lệ ngữ cảnh vì không có lý do mùa vụ rõ ràng cho việc tăng doanh số trong lĩnh vực này vào thời điểm đó.

Một số nguyên nhân dẫn đến dữ liệu ngoại lệ:

- Sai sót trong quá trình thu thập dữ liệu.
- Đặc điểm tự nhiên.
- Thay đổi môi trường.
- Gian lận hoặc tấn công.

Dữ liệu ngoại lệ có thể ảnh hưởng tiêu cực đến kết quả phân tích dữ liệu, dẫn đến sai lệch trong mô hình và sai sót trong dự đoán. Do đó, việc xác định và xử lý dữ liệu ngoại lệ là một bước quan trọng trong quá trình tiền xử lý dữ liệu. Một số phương pháp xử lý dữ liệu ngoại lệ như: loại bỏ ngoại lệ, giới hạn ngoại lệ bằng IQR, giới hạn ngoại lệ bằng cách sử dụng giá trị trung bình và độ lệch chuẩn, giới hạn ngoại lệ bằng các giá trị phân vị, giới hạn ngoại lệ bằng cách sử dụng giá trị tùy chỉnh.

2.6.4. Dữ liệu mất cân bằng

Mất cân bằng dữ liệu (hay còn gọi là thiếu cân bằng dữ liệu, chênh lệch dữ liệu) là tình trạng trong một tập dữ liệu số lượng mẫu thuộc một số lớp nhất định nhiều hơn so với các lớp khác một cách đáng kể.

Một số nguyên nhân dẫn đến mất cân bằng dữ liệu:

- Hiếm gặp tự nhiên
- Khó khăn trong việc thu thập dữ liệu
- Lỗi mất mát dữ liệu
- Lọc dữ liệu

Mất cân bằng dữ liệu có thể gây ảnh hưởng đến hiệu suất của mô hình vì vậy ta cần có các phương pháp xử lý như: Down Sampling, Up Sampling.

2.6.5. Trích chọn đặc trưng (Feature Selection)

Trong thực tế khi ta xây dựng các mô hình học máy, có rất nhiều feature trong tập dữ liệu và không phải lúc nào tất cả các feature này đều quan trọng. Việc thêm các feature không cần thiết trong khi huấn luyện mô hình sẽ làm giảm độ chính xác, tăng độ phức tạp và giảm khả năng khai quát hóa mô hình làm cho mô hình dễ bị sai lệch. Chính vì thế ta cần phải Feature Selection (Trích chọn đặc trưng).

Feature Selection là quá trình chọn ra một tập hợp con của các features từ bộ dữ liệu ban đầu, sao cho các features này có ảnh hưởng lớn nhất hoặc có giá trị nhất đối với mô hình dự đoán. Điều này giúp đơn giản hóa mô hình, cải thiện hiệu suất, giảm chi phí tính toán và giảm nguy cơ overfitting. [4] [5] [6]

Một số kỹ thuật phổ biến để Feature Selection:

- Filter methods
- Wrapper methods
- Embedded methods

2.6.6. Dữ liệu đa chiều

Dữ liệu đa chiều là dữ liệu có nhiều hơn hai chiều hoặc thuộc tính. Mỗi chiều đại diện cho một biến hoặc đặc trưng khác nhau.

Trong các bài toán học máy thì dữ liệu có kích thước rất lớn. Máy tính có thể hiểu và thực thi các thuật toán trên dữ liệu này, tuy nhiên đối với người dùng để "nhìn" dữ liệu nhiều chiều thật sự rất khó. Vì vậy, giảm chiều dữ liệu giúp ta đưa ra cái nhìn mới về dữ

liệu nhiều chiều. Ngoài để trực quan dữ liệu, các phương pháp giảm chiều dữ liệu còn giúp đưa dữ liệu về một không gian mới giúp khai phá các thuộc tính ẩn mà trong chiều dữ liệu ban đầu không thể hiện rõ, hoặc đơn giản là giảm kích thước dữ liệu để tăng tốc độ thực thi cho máy tính.

Phương pháp thường được sử dụng để giảm chiều dữ liệu là Principal Component Analysis (PCA) và Linear Discriminant Analysis (LDA).

2.6.7. Mã hóa dữ liệu phân loại (encoding categorical data)

Các mô hình thống kê và học máy yêu cầu dữ liệu đầu vào dưới dạng số học để có thể thực hiện các tính toán và phân tích. Dữ liệu phân loại như tên, nhãn hoặc các giá trị không có ý nghĩa số học, không thể được sử dụng trực tiếp. Vì vậy, ta cần chuyển đổi dữ liệu phân loại sang dạng số.

Các kỹ thuật được sử dụng để chuyển đổi dữ liệu phân loại sang dạng số được gọi là encoding categorical data. Một số phương pháp phổ biến như: One-hot encoding, label encoding, frequency encoding, ordinal encoding, mean encoding.

2.6.8. Feature scaling

Một tập dữ liệu có thể có các thuộc tính khác nhau. Các thuộc tính có thể có độ lớn, phương sai, độ lệch chuẩn, giá trị trung bình khác nhau. Ví dụ, mức lương có thể tính bằng hàng nghìn, trong khi tuổi thường là một con số có hai chữ số. Sự khác biệt về quy mô hoặc độ lớn của các thuộc tính thực sự có thể ảnh hưởng đến các mô hình thống kê. Các biến có phạm vi lớn hơn chiếm ưu thế so với các biến có phạm vi nhỏ hơn, đối với các mô hình tuyến tính. Tương tự, thuật toán gradient descent hội tụ nhanh hơn khi các biến có tỷ lệ tương tự. Vì vậy ta cần phải đưa các thuộc tính về 1 thang đo.

Feature Scaling (hay còn gọi là Normalization) là một kỹ thuật phổ biến trong tiền xử lý dữ liệu cho machine learning. Nó được sử dụng để chuẩn hóa phạm vi giá trị của các thuộc tính trong một dataset, đưa chúng về một thang đo chung, thường là khoảng từ 0 đến 1 hoặc -1 đến 1. Một số phương pháp phổ biến như: Standardization, Min/Max Scaling, Mean Normalization, Maximum Absolute Scaling.

2.7. Xử lý dữ liệu thiếu

Để xử lý dữ liệu bị thiếu, chúng ta có thể loại bỏ các hàng/cột chứa các giá trị thiếu. Đây là cách đơn giản nhất, tuy nhiên có thể dẫn đến mất đi một lượng lớn dữ liệu và làm giảm hiệu quả của mô hình. Chỉ nên dùng khi các giá trị bị thiếu hoàn toàn ngẫu nhiên và

tỷ lệ phần trăm bản ghi có giá trị bị thiếu nhỏ hơn 5%. Khi dữ liệu của một biến bị thiếu khoảng 60 ~70 % thì chúng ta nên xem xét đến việc loại bỏ hoàn toàn biến đó. Sau đây là một số cách xử lý dữ liệu bị thiếu phổ biến: [2] [1] [3]

2.7.1. Single imputation

Single imputation là một kỹ thuật thống kê được sử dụng để xử lý dữ liệu bị thiếu. Khi dữ liệu bị thiếu, thay vì loại bỏ các mẫu hoặc biến có dữ liệu thiếu, single imputation điền giá trị thay thế cho các điểm dữ liệu bị thiếu. Có một số phương pháp phổ biến của single imputation:

2.7.1.1. Mean Imputation (Điền giá trị trung bình)

Mean Imputation sử dụng giá trị trung bình của các quan sát không thiếu để thay thế cho các giá trị bị thiếu. Phương pháp này thường được sử dụng cho dữ liệu định lượng khi dữ liệu bị thiếu ngẫu nhiên và phân phối của dữ liệu gần với phân phối chuẩn (normal distribution). Tuy nhiên, cần lưu ý rằng mean imputation có thể làm biến đổi phân phối của dữ liệu nếu tỷ lệ dữ liệu bị thiếu lớn. Ví dụ (Bảng 2.1):

Bảng 2.1 Trước khi Mean Imputation.

Chỉ số IQ	Tình trạng tâm lý tốt	Hiệu suất công việc
78	13	
84	9	
84	10	
85	10	
87		
91	3	
92	12	
94	3	
94	13	
96		
99	6	7
105	12	10
105	14	11
106	10	15
108		10
112	10	10
113	14	12
115	14	14
118	12	16
134	11	12

$$\text{Mean(cột 2)} = \frac{\sum x_i}{n} = 10,4$$

$$\text{Mean(cột 3)} = \frac{\sum x_i}{n} = 11,7$$

$$\text{Variance (cột 2)} = \frac{\sum (xi - \bar{x})^2}{n-1} = 11,99$$

$$\text{Variance (cột 3)} = \frac{\sum (xi - \bar{x})^2}{n-1} = 7,34$$

Bảng 2.2 Sau khi thực hiện Mean Imputation

Chỉ số IQ	Tình trạng tâm lý tốt	Hiệu suất công việc
78	13	11,7
84	9	11,7
84	10	11,7
85	10	11,7
87	10,4	11,7
91	3	11,7
92	12	11,7
94	3	11,7
94	13	11,7
96	10,4	11,7
99	6	7
105	12	10
105	14	11
106	10	15
108	10,4	10
112	10	10
113	14	12
115	14	14
118	12	16
134	11	12

$$\text{Variance (cột 2)} = \frac{\sum (xi - \bar{x})^2}{n-1} = 10,1$$

$$\text{Variance (cột 3)} = \frac{\sum (xi - \bar{x})^2}{n-1} = 4,48$$

Sau khi thực hiện Mean Imputation (Bảng 2.2) ta nhận thấy variance của *tình trạng tâm lý* đã giảm từ 11,9 xuống còn 10,1 và *hiệu suất công việc* giảm từ 7,34 xuống còn 4,48.

Điều này cho thấy mean imputation làm giảm độ biến thiên trong dữ liệu vì nó thay thế các giá trị bị thiếu bằng một giá trị cố định, làm cho dữ liệu trở nên ít đa dạng hơn.

Nhìn chung phương pháp này có thể được thực hiện nhanh chóng mà không cần các kỹ thuật phức tạp, không làm thay đổi tổng số lượng dữ liệu, giúp duy trì tính ổn định. Tuy nhiên nó lại làm giảm độ biến thiên trong dữ liệu, dữ liệu trở nên đồng nhất hơn, làm mất đi sự đa dạng và khả năng phản ánh đúng thực tế của dữ liệu. Việc này có thể dẫn đến các kết quả phân tích thiếu chính xác và bị lệch.

2.7.1.2. Median Imputation (Thay thế bằng giá trị trung vị)

Khi dữ liệu không tuân theo phân phối chuẩn (skewed distribution), ta nên sử dụng median imputation. Median là giá trị ở giữa của dãy dữ liệu đã sắp xếp. Khi dữ liệu bị thiếu, ta thay thế giá trị bằng giá trị trung vị. Phương pháp này ít bị ảnh hưởng bởi các giá trị ngoại lệ (outliers) và không làm biến đổi phân phối của dữ liệu vì trung vị chỉ dựa trên vị trí của các giá trị trong tập dữ liệu, chứ không phải các giá trị cụ thể. Ví dụ (Bảng 2.3):

Bảng 2.3 Trước khi thực hiện Median Imputation

Chỉ số IQ	Tình trạng tâm lý tốt	Hiệu suất công việc
78	13	
84	9	
84	10	
85	10	
87		
91	3	
92	12	
94	3	
94	13	
96		
99	6	7
105	12	10
105	14	11
106	10	15
108		10
112	10	10
113	14	12
115	14	14
118	12	16
134	11	12

Median (cột 2) = 11

Median (cột 3) = 11,5

$$\text{Variance (cột 2)} = \frac{\sum (xi - \bar{x})^2}{n-1} = 11,99$$

$$\text{Variance (cột 3)} = \frac{\sum (xi - \bar{x})^2}{n-1} = 7,34$$

Bảng 2.4 Sau khi thực hiện Median Imputation

Chỉ số IQ	Tình trạng tâm lý tốt	Hiệu suất công việc
78	13	11,5
84	9	11,5
84	10	11,5
85	10	11,5
87	11	11,5
91	3	11,5
92	12	11,5
94	3	11,5
94	13	11,5
96	11	11,5
99	6	7
105	12	10
105	14	11
106	10	15
108	11	10
112	10	10
113	14	12
115	14	14
118	12	16
134	11	12

$$\text{Variance (cột 2)} = \frac{\sum (xi - \bar{x})^2}{n-1} = 10,16$$

$$\text{Variance (cột 3)} = \frac{\sum (xi - \bar{x})^2}{n-1} = 3,49$$

Tương tự như mean imputation: median imputation cũng làm giảm độ biến thiên trong dữ liệu. Nhìn chung phương pháp này đơn giản, dễ thực hiện không bị ảnh hưởng bởi các giá trị ngoại lai. Tuy nhiên, việc thay thế các giá trị thiếu bằng một giá trị trung vị cố định có thể làm giảm độ biến thiên của dữ liệu, dẫn đến tập dữ liệu ít đa dạng hơn. Đôi

với các tập dữ liệu liên tục có xu hướng thay đổi theo thời gian việc thay thế bằng giá trị trung vị có thể không phù hợp và làm mất đi xu hướng của dữ liệu.

2.7.1.3. Mode Imputation

Sử dụng giá trị xuất hiện nhiều nhất trong dữ liệu để thay thế cho các giá trị bị thiếu.

Phương pháp này thường được sử dụng cho dữ liệu phân loại. Ví dụ (Bảng 2.5):

Bảng 2.5. Ví dụ về Mode Imputation

Tên	Tuổi	Giới tính	Môn học yêu thích	Mode Imputation
An	16	Nam	Toán	Toán
Bình	17	Nữ	Văn	Văn
Cường	16	Nam	Toán	Toán
Dương	17		Toán	Toán
Hạnh		Nữ	Văn	Văn
Khang	16	Nam		Toán
Minh	17	Nữ	Văn	Văn
Huệ	17	Nữ	Toán	Toán
Kim	17	Nữ	Toán	Toán

Mode (Tuổi): 17 (có 5 giá trị 17, có 3 giá trị 16)

Mode (Giới tính): Nữ (có 3 giá trị nam, có 5 giá trị nữ)

Mode (Môn học yêu thích): Toán (có 5 giá trị toán, có 3 giá trị văn)

Mode Imputation là một phương pháp khá đơn giản và hiệu quả để xử lý dữ liệu bị thiếu đối với dữ liệu phân loại, đặc biệt trên các tập dữ liệu lớn. Tuy nhiên, nó cũng có những hạn chế như làm tăng quá mức tần suất của nhãn danh mục phổ biến nhất. Trong trường hợp có rất ít giá trị trong tập dữ liệu ban đầu, việc sử dụng nhãn danh mục xuất hiện thường xuyên nhất có thể dẫn đến tạo ra một nhãn mới chứa các giá trị hiếm gặp.

2.7.1.4. Arbitrary Value Imputation (gán giá trị tùy ý)

Arbitrary Value Imputation là kỹ thuật thay thế các giá trị thiếu (missing values) trong dữ liệu bằng một giá trị tùy ý do người phân tích lựa chọn, thường được sử dụng khi dữ liệu bị thiếu không chiếm tỷ lệ nhỏ, giá trị bị thiếu không có một mẫu cố định. Thông thường, giá trị này có thể là một số không tồn tại trong dữ liệu gốc, chẳng hạn như 99, 999. Trong trường hợp tập dữ liệu chỉ chứa giá trị dương, -1 có thể được chọn làm số tùy ý để dễ dàng phân biệt với các giá trị khác trong tập dữ liệu.

Giả sử chúng ta có một bộ dữ liệu về thu nhập hàng năm của một nhóm người, trong đó có một số giá trị bị thiếu. Chúng ta có thể chọn giá trị tùy ý là -9999 để thay thế cho các giá trị bị thiếu.

Arbitrary Value Imputation là một phương pháp đơn giản và nhanh chóng để xử lý giá trị bị thiếu trong dữ liệu. Tuy nhiên gán giá trị tùy ý có thể ảnh hưởng đến các thống kê như trung bình, độ lệch chuẩn, và có thể gây ra sai lệch trong các kết quả phân tích.

2.7.1.5. Zero Imputation

Zero Imputation được sử dụng để xử lý dữ liệu bị thiếu (missing data) bằng cách thay thế tất cả các giá trị bị thiếu bằng giá trị 0. Phương pháp này giả định rằng các giá trị bị thiếu không có ý nghĩa đặc biệt và có thể được thay thế bằng 0. Thường được sử dụng cho các biến rời rạc (discrete variables), như dữ liệu đếm hoặc dữ liệu nhị phân, khi giá trị bị thiếu có thể được giải thích là "không" hoặc "có". [1]

Giả sử chúng ta có một bộ dữ liệu về khảo sát hành vi sử dụng rượu của một nhóm người, trong đó có một số giá trị bị thiếu. Chúng ta thay thế các giá trị bị thiếu bằng 0.

Nhìn chung phương pháp này đơn giản dễ thực hiện nhưng nếu có quá nhiều giá trị bị thiếu và được thay thế bằng 0, điều này có thể làm sai lệch các thống kê mô tả và ảnh hưởng đến kết quả phân tích.

2.7.2. Multiple Imputation (MI)

Là kỹ thuật thống kê được sử dụng để xử lý dữ liệu thiếu trong các tập dữ liệu. Thay vì chỉ điền vào giá trị thiếu một lần, MI tạo ra nhiều bản sao của tập dữ liệu với các giá trị thiếu được điền vào bằng các giá trị khác nhau, dựa trên một mô hình xác suất. Sau đó, phân tích thống kê được thực hiện trên mỗi bản sao dữ liệu, và kết quả được kết hợp lại để đưa ra ước lượng cuối cùng và sai số chuẩn. Do vậy, quy trình thực hiện MI gồm 3 bước: imputation, analysis and pooling.

Imputation: Ước lượng các giá trị thiếu trong tập dữ liệu ban đầu bằng cách sử dụng các mô hình như: hồi quy tuyến tính, thay thế giá trị trung bình dự báo, hồi quy logit, hồi quy logit thứ tự (ologit), hồi quy logit đa bậc (mlogit). Quá trình này tạo ra nhiều (thường là m) tập dữ liệu hoàn chỉnh, trong đó mỗi tập dữ liệu có các giá trị thiếu được ước lượng khác nhau.

Analysis: Thực hiện các phân tích thống kê như trung bình, phương sai trên từng bộ dữ liệu hoàn chỉnh.

Pooling: m kết quả được hợp nhất thành một kết quả bằng cách tính giá trị trung bình, phương sai.

Nhìn chung phương pháp này giúp đảm bảo sự biến thiên của dữ liệu và tăng độ chính xác của kết quả phân tích bên cạnh đó quá trình tạo ra nhiều bộ dữ liệu và phân tích từng bộ dữ liệu này đòi hỏi thời gian và tài nguyên tính toán.

2.8. Xử lý dữ liệu trùng

2.8.1. Loại bỏ dữ liệu trùng

Xóa bỏ các bản ghi trùng lặp khỏi tập dữ liệu [7] là một phương pháp đơn giản và dễ thực hiện. Phương pháp này giúp loại bỏ các bản ghi trùng lặp mà không cần phân tích phức tạp. Tuy nhiên, nó có thể dẫn đến mất thông tin quan trọng nếu các bản ghi trùng lặp chứa các dữ liệu khác nhau. Do đó, phương pháp này chỉ thực sự hiệu quả khi các bản ghi trùng lặp hoàn toàn giống nhau.

2.8.2. Gộp dữ liệu trùng

Thay vì loại bỏ các bản ghi trùng, chúng được gộp lại thành một bản ghi duy nhất bằng cách kết hợp các giá trị từ các bản ghi trùng lặp [7]. Phương pháp này giúp ta kết hợp dữ liệu lại với nhau tránh mất dữ liệu nhưng nó đòi hỏi phân tích xác định cách gộp dữ liệu hợp lý để đảm bảo tính chính xác và nhất quán của dữ liệu sau khi gộp.

2.9. Các phương pháp trích chọn đặc trưng (Feature Selection)

2.9.1. Filter method

Sử dụng các kỹ thuật thống kê để dự đoán mối quan hệ giữa từng biến đầu vào độc lập và biến đầu ra. Một số kỹ thuật:

- Chi-Square Test: Được sử dụng cho các đặc trưng phân loại. Kiểm tra mức độ độc lập giữa đặc trưng và nhãn.
- Correlation Coefficient: Đánh giá mức độ tương quan giữa các đặc trưng và nhãn. Pearson's correlation coefficient thường được sử dụng cho các đặc trưng liên tục.
- Variance Threshold: Loại bỏ các đặc trưng có phương sai thấp hơn ngưỡng xác định.
- Mutual Information: Đo lường sự phụ thuộc lẫn nhau giữa các đặc trưng và nhãn.

Phương pháp này nhanh chóng và đơn giản, không đòi hỏi xây dựng mô hình. Tuy nhiên, kết quả sẽ kém nếu dữ liệu không đủ để mô hình hóa mối tương quan thống kê giữa các biến đặc trưng. [4] [5] [6]

2.9.2. Wrapper method

Tạo ra một số mô hình máy học gồm các tập hợp con chứa các biến tính năng đầu vào khác nhau. Sau đó, các tính năng được chọn sẽ tạo ra mô hình hoạt động tốt nhất theo chỉ số hiệu suất. Một số kỹ thuật thường được dùng trong phương pháp này: [4] [5] [6]

- Forward selection: là một quá trình lặp đi lặp lại, bắt đầu với một bộ tính năng trống. Sau mỗi lần lặp, nó tiếp tục bổ sung thêm một tính năng và đánh giá hiệu suất để kiểm tra xem nó có cải thiện hiệu suất hay không. Quá trình tiếp tục cho đến khi việc bổ sung một biến/tính năng mới không cải thiện được hiệu suất của mô hình.
- Backward elimination: là một cách tiếp cận lặp đi lặp lại, nhưng nó trái ngược với Forward selection. Kỹ thuật này bắt đầu quá trình bằng cách xem xét tất cả các đặc điểm và loại bỏ những đặc điểm ít quan trọng nhất. Quá trình loại bỏ này tiếp tục cho đến khi việc loại bỏ các đặc trưng không cải thiện được hiệu suất của mô hình.
- Exhaustive selection: Kỹ thuật này được coi là phương pháp tiếp cận mạnh mẽ để đánh giá các tập hợp đặc trưng. Nó tạo ra tất cả các tập hợp con có thể và xây dựng thuật toán học cho từng tập hợp con và chọn tập hợp con có hiệu suất mô hình tốt nhất.
- Recursive feature elimination (RFE): được sử dụng để loại bỏ dần các đặc trưng ít quan trọng từ tập dữ liệu. Phương pháp này dựa trên việc huấn luyện một mô hình và sau đó loại bỏ lặp lại các đặc trưng có đóng góp thấp nhất cho mô hình.

Ưu điểm của Wrapper method là có thể tìm ra tập hợp đặc trưng tối ưu cho mô hình cụ thể. Tuy nhiên, nhược điểm là tốn kém về thời gian tính toán, đặc biệt là với các tập dữ liệu lớn.

2.9.3. Embedded methods

Kết hợp các ưu điểm của cả phương filter và wrapper bằng cách xem xét sự tương tác của các tính năng cùng với chi phí tính toán thấp. Đây là những phương pháp xử lý nhanh tương tự như filter nhưng có độ chính xác cao hơn. Một số kỹ thuật thường dùng: [4] [5] [6]

- Regularization Methods: Các thuật toán như L1 hoặc Elastic Net regularization có thể ép một số trọng số về 0, do đó loại bỏ các đặc trưng ít quan trọng.

- Tree-based methods: thuật toán như Random Forest, Gradient Boosting có thể cung cấp tầm quan trọng của các đặc trưng dựa trên mức độ ảnh hưởng của chúng trong cây quyết định.

Do kết hợp quá trình lựa chọn đặc trưng và huấn luyện mô hình nên Embedded giúp tiết kiệm thời gian và tài nguyên tính toán. Tuy nhiên, chúng có thể phức tạp hơn để triển khai và yêu cầu hiểu biết sâu hơn về thuật toán sử dụng.

2.10. Xử lý dữ liệu mất cân bằng

2.10.1. Down Sampling

Giảm số lượng mẫu trong lớp đa số để cân bằng với lớp thiểu số. Kỹ thuật phổ biến được dùng là Random Under-sampling. [8]

Các bước thực hiện:

Bước 1. Chia tập dữ liệu thành hai lớp (đa số và thiểu số).

Bước 2. Xác định tỷ lệ phần trăm cần loại bỏ từ lớp đa số.

Bước 3. Chọn một mẫu ngẫu nhiên từ lớp đa số để loại bỏ.

Bước 4. Lặp lại bước 3 cho đến khi số lượng phiên bản được loại bỏ theo tỷ lệ phần trăm nhất định.

Ví dụ: Giả sử ta có một tập dữ liệu với 1000 mẫu của lớp A và 100 mẫu của lớp B. Để cân bằng dữ liệu bằng phương pháp down sampling, ta sẽ giảm số lượng mẫu của lớp A xuống còn 100 mẫu, sao cho số lượng mẫu của hai lớp bằng nhau.

Nhìn chung đây là phương pháp tuy giúp giảm thời gian và tài nguyên tính toán nhưng làm giảm kích thước dữ liệu có thể làm mất đi các thông tin quan trọng do loại bỏ một số mẫu từ lớp chiếm ưu thế.

2.10.2. Up Sampling

Ngược lại với Down Sampling phương pháp Up Sampling cân bằng dữ liệu bằng cách tăng số lượng mẫu trong lớp thiểu bằng cách sao chép các mẫu của lớp thiểu để phù hợp với số lượng mẫu trong lớp chiếm ưu thế. Phương pháp này làm tăng độ chính xác cho mô hình học máy, nhưng hiệu suất cho các bộ dữ liệu mới không tốt vì tập dữ liệu mới có thể xuất hiện các trường hợp không có thực. [8]

Các kỹ thuật dùng để Up Sampling: Random Oversampling, SMOTE.

Random Oversampling: làm tăng dữ liệu với nhiều bản sao của một số mẫu được chọn ngẫu nhiên ở lớp thiểu số.

Quy trình thực hiện

Bước 1. Chia tập dữ liệu thành hai lớp (đa số và thiểu số).

Bước 2. Tạo ra các mẫu ngẫu nhiên từ lớp thiểu số với phép lấy mẫu thay thế (có thể lặp lại các mẫu ban đầu) cho đến khi số lượng mẫu của lớp thiểu số bằng lớp chiếm ưu thế.

Bước 3. Kết hợp các mẫu từ lớp thiểu số đã được up sample với lớp chiếm ưu thế để tạo thành tập dữ liệu cân bằng.

Ví dụ: Trong dự đoán bệnh hiếm, có 1000 bệnh nhân khỏe mạnh và 50 bệnh nhân mắc bệnh. Random Oversampling sẽ sao chép ngẫu nhiên các bệnh nhân mắc bệnh để có 1000 mẫu.

Kỹ thuật này giúp ta giữ lại được toàn bộ thông tin nhưng có nhược điểm là số lượng mẫu lỗi được tăng lên gấp nhiều lần nhưng nội dung và chất lượng của mẫu không có sự thay đổi dễ dẫn đến overfitting và không làm tăng tính đa dạng của lớp thiểu số. Trong trường hợp tập dữ liệu có kích thước lớn thì chi phí thời gian và bộ nhớ cho giai đoạn phân lớp sẽ gia tăng đáng kể.

SMOTE: sử dụng các mẫu dữ liệu được tổng hợp. Dữ liệu mới này được tạo bằng cách nội suy giữa một số trường hợp ở lớp thiểu số nằm trong vùng lân cận gần nhất.

Các bước thực hiện SMOTE:

- Tìm KNN: với mỗi mẫu trong lớp thiểu số, tính khoảng cách Euclidean tới các mẫu khác trong lớp thiểu số để tìm K láng giềng gần nhất.
- Chọn ngẫu nhiên từ KNN: Chọn ngẫu nhiên N mẫu từ k-láng giềng gần nhất của mỗi mẫu trong lớp thiểu số. N được gọi là tỷ lệ lấy mẫu và quyết định số lượng mẫu mới được tạo ra cho mỗi mẫu dữ liệu gốc.
- Tạo mẫu tổng hợp mới theo công thức:

$$\text{Mẫu mới} = \text{Mẫu gốc} + r * (\text{Mẫu láng giềng} - \text{Mẫu gốc}) \quad (1)$$

Với r được chọn ngẫu nhiên nằm trong khoảng từ 0 đến 1

Ví dụ (Bảng 2.6): Giả sử chúng ta có một tập dữ liệu nhỏ với 2 đặc trưng (features) và một nhãn lớp. Lớp chiếm ưu thế (majority class): 9 mẫu. Lớp thiểu số (minority class): 4 mẫu.

Bảng 2.6. Dữ liệu mẫu

Mẫu	Feature1	Feature2	Lớp
1	1	1	Majority

2	2	2	Majority
3	3	3	Majority
4	4	4	Majority
5	5	5	Majority
6	6	6	Majority
7	7	7	Majority
8	8	8	Majority
9	9	9	Majority
10	10	10	Minority
11	11	11	Minority
12	12	12	Minority
13	13	13	Minority

Ta sẽ tạo thêm 5 mẫu từ lớp thiểu số để cân bằng với lớp chiếm ưu thế.

Bước 1: Tìm KNN. Chọn k=2

Bảng 2.7. Khoảng cách Euclidean giữa các mẫu thiểu số.

Mẫu	Mẫu 10	Mẫu 11	Mẫu 12	Mẫu 13
10	0	$\sqrt{2}$	$2\sqrt{2}$	$3\sqrt{2}$
11	$\sqrt{2}$	0	$\sqrt{2}$	$2\sqrt{2}$
12	$2\sqrt{2}$	$\sqrt{2}$	0	$\sqrt{2}$
13	$3\sqrt{2}$	$2\sqrt{2}$	$\sqrt{2}$	0

Dựa vào Bảng 2.7 ta có:

Mẫu 10: có 2 láng giềng gần nhất là mẫu 11 và mẫu 12.

Mẫu 11: có 2 láng giềng gần nhất là mẫu 10 và mẫu 12.

Mẫu 12: có 2 láng giềng gần nhất là mẫu 11 và mẫu 13.

Mẫu 13: có 2 láng giềng gần nhất là mẫu 11 và mẫu 12.

Bước 2 + 3: Chọn ngẫu nhiên 5 mẫu từ KNN của các mẫu thiểu số sau đó tạo mẫu tổng hợp.

Ta chọn mẫu 10 với láng giềng mẫu 11.

$$\text{Mẫu mới} = (10,10) + 0,6 * ((11,11) - (10,10)) = (10.6, 10.6)$$

Ta chọn mẫu 11 với láng giềng mẫu 12.

$$\text{Mẫu mới} = (11,11) + 0,3 * ((12,12) - (11,11)) = (11.3, 11.3)$$

Ta chọn mẫu 12 với láng giềng mẫu 13.

$$\text{Mẫu mới} = (12,12) + 0,5 * ((13,13) - (12,12)) = (12.5, 12.5)$$

Ta chọn mẫu 13 với láng giềng mẫu 12.

Mẫu mới = $(13,13) + 0,8 * ((12,12) - (13,13)) = (12.2, 12.2)$

Ta chọn mẫu 10 với láng giềng mẫu 12.

Mẫu mới = $(10,10) + 0,4 * ((12,12) - (10,10)) = (10.8, 10.8)$

Bảng 2.8. Sau khi thực hiện SMOTE

Mẫu	Feature1	Feature2	Lớp
1	1	1	Majority
2	2	2	Majority
3	3	3	Majority
4	4	4	Majority
5	5	5	Majority
6	6	6	Majority
7	7	7	Majority
8	8	8	Majority
9	9	9	Majority
10	10	10	Minority
11	11	11	Minority
12	12	12	Minority
13	13	13	Minority
14	10.6	10.6	Minority
15	11.3	11.3	Minority
16	12.5	12.5	Minority
17	12.2	12.2	Minority
18	10.8	10.8	Minority

SMOTE giúp tăng số lượng mẫu của lớp thiểu số, làm cho dữ liệu trở nên cân bằng hơn, từ đó cải thiện hiệu suất của các mô hình học máy trên các tập dữ liệu mất cân bằng. Bằng cách tạo ra các mẫu tổng hợp mới giữa các điểm hiện có, SMOTE giúp tạo ra một tập dữ liệu đa dạng hơn, có thể giúp mô hình học tốt hơn. Tuy nhiên, nó có thể tạo ra các mẫu tổng hợp không tồn tại trong thực tế, đặc biệt là khi các mẫu trong lớp thiểu số không đồng nhất hoặc bị cụm lại. Điều này có thể dẫn đến mô hình học máy bị học theo các mẫu không thực tế.

2.11. Xử lý dữ liệu ngoại lệ

2.11.1. Loại bỏ các ngoại lệ (Removing outliers)

Loại bỏ hoàn toàn các điểm dữ liệu được xem là ngoại lệ khỏi tập dữ liệu. [3]

Giả sử ta có tập dữ liệu { 5, 6, 12, 13, 15, 18, 22, 50 }.

$$Q1 = (6 + 12) / 2 = 9$$

$$Q3 = (18 + 22) / 2 = 20$$

$$IQR = Q3 - Q1 = 20 - 9 = 11.$$

$$\text{Giới hạn dưới} = 9 - 1.5 * 11 = -7,5$$

$$\text{Giới hạn trên} = 20 + 1.5 * 11 = 36,5$$

=> Loại bỏ điểm dữ liệu 50 vì nó nằm ngoài giới hạn trên.

Đây là phương pháp đơn giản dễ thực hiện giúp làm sạch dữ liệu, loại bỏ những giá trị cực đoan có thể gây ảnh hưởng xấu đến kết quả phân tích nhưng nó có thể làm mất thông tin quan trọng đặc biệt nếu số lượng ngoại lệ lớn không phù hợp khi số lượng dữ liệu ban đầu ít, vì loại bỏ nhiều điểm dữ liệu có thể làm giảm độ tin cậy của kết quả.

2.11.2. Giới hạn ngoại lệ bằng IQR (Outlier Capping Using IQR)

Là một kỹ thuật xử lý ngoại thay vì loại bỏ hoàn toàn các giá trị ngoại lệ, nó thay thế chúng bằng các giá trị hợp lý hơn. Các outlier thấp bằng giá trị Lower Bound, các outlier cao bằng giá trị Upper Bound. [3]

Giả sử ta có tập dữ liệu { 5, 6, 12, 13, 15, 18, 22, 50 }.

$$Q1 = (6 + 12) / 2 = 9$$

$$Q3 = (18 + 22) / 2 = 20$$

$$IQR = Q3 - Q1 = 20 - 9 = 11.$$

$$\text{Lower Bound} = 9 - 1.5 * 11 = -7,5$$

$$\text{Upper Bound} = 20 + 1.5 * 11 = 36,5$$

Giá trị 50 là outlier ta thay bằng 36,5.

Tập dữ liệu sau khi xử lý { 5, 6, 12, 13, 15, 18, 22, 36,5 }.

Phương pháp này giúp giảm ảnh hưởng của các giá trị ngoại lai mà không hoàn toàn loại bỏ chúng, giúp bảo toàn xu hướng và phân phối dữ liệu. Tuy nhiên nó không hiệu quả với dữ liệu có phân phối không chuẩn hoặc có nhiều ngoại lệ, có thể làm mất đi thông tin quan trọng tiềm ẩn trong các giá trị ngoại lai đó.

2.11.3. Giới hạn ngoại lệ sử dụng giá trị trung bình và độ lệch chuẩn (Outlier Capping Using Mean and Std)

Phương pháp này sử dụng giá trị trung bình (μ) và độ lệch chuẩn (σ) của tập dữ liệu để xác định và giới hạn các giá trị ngoại lệ. Để tìm Upper Bound ta cộng giá trị trung bình của dữ liệu với ba lần giá trị độ lệch chuẩn. Tương tự, để tìm Lower Bound, ta lấy giá trị trung bình trừ cho tích độ lệch chuẩn với 3. Các giá trị vượt quá Upper Bound

sẽ được gán bằng Upper Bound, và các giá trị thấp hơn Lower Bound sẽ được gán bằng Lower Bound, qua đó giảm thiểu ảnh hưởng của các giá trị cực đoan trong quá trình phân tích dữ liệu. [3]

Giả sử ta có tập dữ liệu { 5, 6, 12, 13, 15, 18, 22, 50 }.

Mean = 17,625

Std = 14,25

Lower Bound = Mean – 3Std = -25,125

Upper Bound = Mean + 3Std = 60,375

Kết quả: Dữ liệu không thay đổi vì không có điểm dữ liệu nào nằm ngoài phạm vi từ -25,125 đến 60,375

Đây là phương pháp đơn giản hiệu quả với dữ liệu phân phối chuẩn, không làm mất dữ liệu mà chỉ thay đổi giá trị của ngoại lệ. Mean và standard deviation rất nhạy cảm với outliers, nên chúng có thể bị ảnh hưởng bởi chính những giá trị mà chúng ta đang cố gắng xử lý vì thế có thể không hiệu quả với các tập dữ liệu nhỏ hoặc dữ liệu có nhiều ngoại lệ.

2.11.4. Giới hạn ngoại lệ bằng giá trị phân vị (Outlier Capping Using Quantiles)

Là phương pháp giới hạn các giá trị ngoại lệ bằng cách sử dụng các phân vị (quantiles). [3]

Giả sử: Sử dụng phân vị 5% và 95% để giới hạn các giá trị.

Nếu một giá trị nhỏ hơn phân vị 5%, nó được thay thế bằng giá trị phân vị 5%.

Nếu một giá trị lớn hơn phân vị 95%, nó được thay thế bằng giá trị phân vị 95%

$$\text{Vị trí phân vị } 5\%: V = \frac{5}{100} * (N + 1) \quad (2)$$

Với N là số lượng giá trị của tập dữ liệu.

Nếu vị trí không phải là số nguyên ta cần nội suy giữa các giá trị lân cận:

$$Q = x_1 + (x_2 - x_1) * V \quad (3)$$

Trong đó:

- Q là giá trị phân vị
- x_1 là giá trị ở vị trí thứ 1
- x_2 là giá trị ở vị trí thứ 2

Ví dụ: Giả sử có dữ liệu về thu nhập hàng tháng đã được sắp xếp thứ tự tăng dần: [2, 3, 4, 4, 5, 5, 6, 7, 8, 10, 12, 15, 20, 50, 100].

Tính phân vị 5%

$$V = \frac{5}{100} * (15 + 1) = 0,8$$

$$Q5 = 2 + (3-2)*0,8 = 2,8$$

=> Phân vị 5% là 2,8

Tính phân vị 95%

$$V = \frac{95}{100} * (15 + 1) = 15,2$$

$$Q95 = 50 + (100-100)*15,2 = 100$$

=> Phân vị 95% là 100

Dữ liệu sau khi được xử lý : [2.8, 3, 4, 4, 5, 5, 6, 7, 8, 10, 12, 15, 20, 50, 100].

Đây là phương pháp không yêu cầu dữ liệu phải tuân theo bất kỳ phân phối cụ thể nào, phù hợp với nhiều loại dữ liệu khác nhau. Phương pháp này chỉ ảnh hưởng đến các giá trị ngoại lệ và giữ nguyên các giá trị trung tâm của dữ liệu, giúp bảo tồn tính chất tổng thể của tập dữ liệu. Mặc dù không loại bỏ các giá trị ngoại lệ, việc giới hạn chúng có thể làm mất đi một số thông tin quan trọng về sự biến đổi và đặc điểm của chúng.

2.11.5. Giới Hạn Ngoại Lệ Bằng Giá Trị Tùy Chỉnh (Capping Outliers Using Custom Values)

Giới hạn ngoại lệ bằng giá trị tùy chỉnh là kỹ thuật giới hạn các giá trị ngoại lai bằng cách thay thế chúng với các giá trị tùy chỉnh, được xác định bởi người phân tích dựa trên hiểu biết sâu sắc về dữ liệu, ứng dụng, hoặc các yêu cầu cụ thể của dự án. [3]

Ví dụ: Giả sử có dữ liệu về nhiệt độ cơ thể ($^{\circ}\text{C}$) của bệnh nhân: {34.8, 36.5, 37.1, 38.2, 39.5, 41.2, 42.8, 45.0}

Phân tích: Nhiệt độ cơ thể bình thường dao động từ 36.5°C đến 37.5°C . Trên 38.5°C được coi là sốt cao, và trên 41°C có thể gây nguy hiểm.

Xác định ngưỡng:

- Ngưỡng dưới: 35°C (dưới ngưỡng này có thể là lỗi đo hoặc hạ thân nhiệt nguy hiểm)
- Ngưỡng trên: 41°C (trên ngưỡng này cần can thiệp y tế)

Dữ liệu sau khi được xử lý: {35, 36.5, 37.1, 38.2, 39.5, 41.0, 41.0, 41.0}

Đây là phương pháp linh hoạt và có ý nghĩa, đặc biệt hữu ích khi bạn có kiến thức chuyên sâu về dữ liệu, có thể giữ lại thông tin có giá trị từ outliers. Tuy nhiên quyết định các giá trị giới hạn có thể mang tính chủ quan và phụ thuộc vào quan điểm của người phân

tích, phương pháp này có thể không phù hợp với các tập dữ liệu có phân phối đặc thù hoặc chứa nhiều giá trị ngoại lệ hợp lý.

2.12. Các kỹ thuật mã hóa dữ liệu

2.12.1. One-hot encoding

Là kỹ thuật được dùng phổ biến nhất. One - hot encoding sẽ tạo một vector mới cho mỗi giá trị duy nhất trong thuộc tính phân loại. Mỗi vector có kích thước bằng số lượng giá trị duy nhất, với giá trị 1 ở vị trí tương ứng với giá trị hiện tại và 0 ở các vị trí khác. [3]

Ta có *Bảng 2.9* với thuộc tính màu sắc có các giá trị: đỏ, xanh dương, xanh lá.

Bảng 2.9. Sau khi One-hot encoding

Màu sắc	Đỏ	Xanh dương	Xanh lá
Đỏ	1	0	0
Xanh dương	0	1	0
Xanh lá	0	0	1

Sau khi thực hiện One-hot encoding:

Mẫu dữ liệu có màu “đỏ” sẽ được mã hóa thành [1, 0, 0].

Mẫu dữ liệu có màu “xanh dương” sẽ được mã hóa thành [0, 1, 0].

Mẫu dữ liệu có màu “đỏ” sẽ được mã hóa thành [0, 0, 1].

Đây là phương pháp đơn giản phù hợp với dữ liệu không có thứ tự, mỗi giá trị được đại diện riêng biệt, giảm nguy cơ mô hình học sai quan hệ giữa các danh mục. Bên cạnh đó kích thước của dữ liệu sẽ tăng theo số lượng giá trị duy nhất.

2.12.2. Label Encoding

Label encoding được sử dụng để chuyển đổi các giá trị của biến phân loại thành các giá trị số nguyên duy nhất. [3] Ví dụ (*Bảng 2.10*):

Bảng 2.10. Ví dụ về Label Encoding

Tên	Kỹ năng	Label Encoding
Hồng	SQL	0
Mai	Java	1
Đào	Python	2
Cường	Java	1
Hạnh	Python	2
Toàn	SQL	0
Tâm	SQL	0

Label encoding là một kỹ thuật đơn giản, nhanh chóng biến đổi dữ liệu phân loại. Không như One-hot encoding, label encoding không làm tăng số lượng đặc trưng, giúp giảm tài nguyên tính toán và dung lượng lưu trữ. Nhưng nhược điểm lớn của label encoding là nó có thể gây ra một mối quan hệ thứ tự giả giữa các giá trị phân loại. Ví dụ, với mã hóa [1, 2, 3] có thể khiến mô hình hiểu rằng 3 lớn hơn 2 và 2 lớn hơn 1, điều này có thể không đúng với dữ liệu phân loại.

2.12.3. Frequency Encoding

Là phương pháp mã hóa dựa trên số lần xuất hiện của từng giá trị trong dữ liệu. Phương pháp này thường hữu ích trong các tình huống khi tần suất xuất hiện của các giá trị quan trọng và có ảnh hưởng đáng kể đến biến mục tiêu. [3]

Giả sử ta có *Bảng 2.11* dữ liệu gồm 10 mẫu các loại trái cây: táo, cam chuối, nho.

Bảng 2.11. Ví dụ về frequency encoding.

Mẫu	Loại trái cây	Frequency Encoding
1	Táo	0.4
2	Cam	0.3
3	Táo	0.4
4	Chuối	0.2
5	Nho	0.1
6	Táo	0.4
7	Cam	0.3
8	Táo	0.4
9	Chuối	0.2
10	Cam	0.3

Dựa vào *Bảng 2.11* ta nhận thấy số lần xuất hiện của các loại trái cây táo, cam, chuối, nho lần lượt là 4, 3, 2, 1 tương ứng với tần suất là 0.4; 0.3; 0.2; 0.1

Frequency encoding là một phương pháp mã hóa biến phân loại hữu ích trong một số trường hợp, đặc biệt khi tần suất xuất hiện của các danh mục có ý nghĩa đối với mô hình học máy. Trong trường hợp hai giá trị phân loại khác nhau có cùng tần suất xuất hiện, chúng sẽ được mã hóa thành cùng một giá trị số, làm mất đi thông tin ban đầu.

2.12.4. Ordinal Encoding.

Là phương pháp mã hóa dựa trên một sự sắp xếp (thứ tự) cụ thể của các giá trị trong biến đó. [3] Ví dụ (*Bảng 2.12*):

Bảng 2.12. Ordinal encoding.

Trình độ học vấn	Mã hóa
Tiểu học	1
Trung học cơ sở	2
Trung học phổ thông	3
Đại học	4
Trung học phổ thông	3
Tiểu học	1
Đại học	4

Ordinal encoding là một phương pháp mã hóa đơn giản và hiệu quả đặc biệt là khi các danh mục có thứ tự tự nhiên. Đối với các biến phân loại không có thứ tự tự nhiên việc sử dụng ordinal encoding có thể tạo ra sự nhầm lẫn vì các mô hình học máy có thể hiểu rằng có một thứ tự tự nhiên giữa các số được mã hóa.

2.12.5. Mean Encoding.

Mean encoding là một kỹ thuật trong xử lý dữ liệu phân loại, trong đó các giá trị phân loại được thay thế bằng giá trị trung bình của biến mục tiêu (target variable) tương ứng với mỗi giá trị phân loại đó. Phương pháp này thường được sử dụng trong các bài toán hồi quy và phân loại để tận dụng mối quan hệ giữa biến phân loại và biến mục tiêu. [3]

Giả sử bạn có một biến phân loại "Loại trái cây" và một biến mục tiêu "Giá bán" như (Bảng 2.13):

Ta tính giá trị trung bình cho từng loại trái cây:

$$\text{Táo: } (10 + 14 + 13 + 12) / 4 = 12.25$$

$$\text{Cam : } (12 + 11 + 13) / 3 = 12$$

$$\text{Chuối: } (8 + 7)/2 = 7.5$$

$$\text{Nho} = 15$$

Bảng 2.13. Ví dụ Mean Encoding.

Loại trái cây	Giá bán	Mean Encoding
Táo	10	12.25
Cam	12	12
Táo	14	12.25
Chuối	8	7.5
Nho	15	15
Táo	13	12.25
Cam	11	12

Táo	12	12.25
Chuối	7	7.5
Cam	13	12

Nhìn chung đây là một phương pháp mã hóa mạnh mẽ khi có mối quan hệ chặt chẽ giữa biến phân loại và biến mục tiêu. Do mean encoding sử dụng trực tiếp thông tin từ biến mục tiêu nên mô hình dễ bị overfit nếu không có biện pháp kiểm soát.

2.13. Chuẩn hóa dữ liệu

2.13.1. Standardization (Z-score Normalization)

Phương pháp này điều chỉnh các đặc tính để chúng có giá trị trung bình bằng 0 và độ lệch chuẩn bằng 1. [3] Ví dụ (Bảng 2.14):

$$X' = \frac{X - \mu}{\sigma} \quad (4)$$

Trong đó μ là giá trị trung bình, σ là độ lệch chuẩn.

Bảng 2.14. Standardization cho tập dữ liệu.

Original	Standardization
1	-12.649
2	-0.6325
3	0.0000
4	0.6325
5	12.649

Phương pháp này ít nhạy cảm với các giá trị ngoại lai hơn so với Min-Max Scaling vì nó dựa vào độ lệch chuẩn, giúp giảm tác động của các giá trị cực đoan. Standardization không thay đổi phân phối của dữ liệu gốc, chỉ thay đổi thang đo. Điều này giúp giữ nguyên các đặc tính thống kê quan trọng của dữ liệu.

2.13.2. Min/Max Scaling

Phương pháp này đưa tất cả các giá trị dữ liệu vào một khoảng giá trị từ 0 đến 1. [3]

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (5)$$

Trong đó:

- x' là giá trị sau khi chuẩn hóa.
- x là giá trị ban đầu.
- x_{min} là giá trị nhỏ nhất trong tập dữ liệu.

- x_{max} là giá trị lớn nhất trong tập dữ liệu

Bảng 2.15. Min max scaling cho tập dữ liệu.

Original	Min-Max Scaled
1	0
2	0.25
3	0.5
4	0.75
5	1

Phương pháp này rất dễ thực hiện và trực quan. Các giá trị được chuyển đổi vào một khoảng xác định trước (thường là từ 0 đến 1), giữ nguyên tỷ lệ giữa các giá trị dữ liệu ban đầu sau khi chuẩn hóa giúp dễ dàng so sánh và phân tích. Min-Max Scaling không bảo toàn được các đặc tính thống kê quan trọng như trung bình và độ lệch chuẩn của dữ liệu, phân phối của dữ liệu có thể bị thay đổi sau khi chuẩn hóa, điều này có thể làm mất thông tin quan trọng trong quá trình phân tích.

2.13.3. Mean Normalization

Mean normalization sẽ đưa giá trị trung bình của mỗi đặc trưng về 0, giúp cân bằng phạm vi giá trị giữa các đặc trưng. [3]

$$x' = \frac{x - \mu}{x_{max} - x_{min}} \quad (6)$$

Trong đó:

- μ là giá trị trung bình.
- x_{max} và x_{min} lần lượt là giá trị lớn nhất và nhỏ nhất.

Bảng 2.16. Mean normalization cho tập dữ liệu.

Original	Mean Normalized
1	-0.50
2	-0.25
3	0.00
4	0.25
5	0.50

So với Min-Max Scaling, mean normalization ít bị ảnh hưởng bởi các giá trị ngoại lai vì nó không chỉ dựa vào các giá trị cực đại và cực tiểu mà còn cân nhắc giá trị trung bình. Mean normalization yêu cầu tính toán giá trị trung bình, giá trị lớn nhất và giá trị nhỏ

nhất của dữ liệu. Điều này có thể gây khó khăn khi làm việc với các tập dữ liệu lớn hoặc thay đổi theo thời gian.

2.13.4. Maximum Absolute Scaling

Phương pháp này chuẩn hóa các đặc trưng bằng cách chia cho giá trị tuyệt đối lớn nhất của đặc trưng đó. Kết quả là các giá trị của đặc trưng sẽ nằm trong khoảng [-1, 1]. [3]

$$x' = \frac{x}{|x_{max}|} \quad (7)$$

Trong đó: $|x_{max}|$ là giá trị tuyệt đối lớn nhất.

Bảng 2.17. Maximum absolute scaling.

Original	Maximum Absolute Scaled
1	0.20
2	0.40
3	0.60
4	0.80
5	1.00

Maximum Absolute Scaling giữ nguyên tỷ lệ giữa các giá trị dữ liệu ban đầu sau khi chuẩn hóa. Phương pháp này ít nhạy cảm hơn với các giá trị ngoại lai vì nó chỉ phụ thuộc vào giá trị tuyệt đối lớn nhất. Mặc dù ít bị ảnh hưởng bởi outliers hơn Min-Max scaling, nhưng nếu có một outlier cực kỳ lớn, nó sẽ làm co lại tất cả các giá trị khác, gây mất thông tin.

2.14. Giảm số lượng thuộc tính

- Principal Component Analysis (PCA)

Principal Component Analysis (PCA) là một kỹ thuật phân tích thống kê được sử dụng rộng rãi để giảm chiều dữ liệu bằng cách tìm một tập hợp các biến mới, nhỏ hơn tập hợp các biến ban đầu, giữ lại hầu hết thông tin của mẫu. [9] [10]

Các bước thực hiện PCA:

Bước 1: Chuẩn hóa dữ liệu: Để đảm bảo mỗi biến có trọng số như nhau, dữ liệu được chuẩn hóa về cùng một thang đo với giá trị trung bình = 0 và độ lệch chuẩn = 1.

Bước 2: Tính ma trận hiệp phương sai: đo cường độ biến thiên chung giữa hai hoặc nhiều biến, cho biết chúng thay đổi bao nhiêu trong mối quan hệ với nhau. Để tìm hiệp phương sai, chúng ta có thể sử dụng công thức: [11] [12]

$$\text{Covariance matrix} = \frac{1}{n-1}(X^T X) \quad (8)$$

Trong đó:

- X là ma trận dữ liệu đã chuẩn hóa.
- n là số lượng mẫu.

Bước 3: Tính toán các vectơ riêng và giá trị riêng: Từ ma trận hiệp phương sai, tính toán các vectơ riêng (eigen vectors) và giá trị riêng (eigen values). Các vectơ riêng là các hướng của các thành phần chính, còn các giá trị riêng là độ lớn của các thành phần chính đó. [13]

Các giá trị riêng là nghiệm của phương trình đặc trưng:

$$\det(\text{Covariance matrix} - \lambda I) = 0 \quad (9)$$

Trong đó:

- I là ma trận đơn vị
- λ là giá trị riêng
- Covariance matrix là ma trận hiệp phương sai

Ta tìm các vector riêng bằng cách giải hệ phương trình:

$$(\text{Covariance matrix} - \lambda I)\vec{v} = 0 \quad (10)$$

Trong đó:

- I là ma trận đơn vị
- λ là giá trị riêng
- Covariance matrix là ma trận hiệp phương sai
- \vec{v} là vector riêng

Bước 4: Sắp xếp các giá trị riêng theo thứ tự giảm dần: Chọn các vectơ riêng tương ứng với các giá trị riêng lớn nhất để tạo ra ma trận thành phần chính.

Bước 5: Biến đổi dữ liệu: Sử dụng ma trận thành phần chính để biến đổi dữ liệu gốc vào không gian các thành phần chính. [14]

Chuyển đổi dữ liệu sang không gian của các thành phần chính theo công thức:

$$Z = X * \text{Eigenvectors} \quad (11)$$

Trong đó:

- X là ma trận đã được chuẩn hóa
- Eigenvectors là vector riêng

Ví dụ (Bảng 2.18):

Bảng 2.18. Dữ liệu mẫu.

Người	Chiều cao (cm)	Cân nặng (kg)
1	150	50
2	160	55
3	170	65
4	180	70
5	190	80
6	155	52
7	165	57
8	175	68
9	185	75
10	195	85

Bước 1: Sử dụng phương pháp Standardization để chuẩn hóa dữ liệu.

Bảng 2.19. Sau khi chuẩn hóa.

Người	Chiều cao (cm)	Cân nặng (kg)
1	-1,49	-1,30
2	-0,83	-0,89
3	-0,17	-0,06
4	0,50	0,36
5	1,16	1,18
6	-1,16	-1,13
7	-0,50	-0,72
8	0,17	0,19
9	0,83	0,77
10	1,49	1,60

Bước 2: Tính ma trận hiệp phương sai:

Áp dụng công thức (8) ta được:

$$\text{Covariance matrix} = \frac{1}{9} \begin{bmatrix} 9,067 & 8,9609 \\ 8,9609 & 8,992 \end{bmatrix} = \begin{bmatrix} 1,007 & 0,996 \\ 0,996 & 0,996 \end{bmatrix}$$

Bước 3: Tính các vector riêng và giá trị riêng.

Ta tính các giá trị riêng (eigen values) và vector riêng (eigen vectors) của ma trận hiệp phương sai bằng công thức (9) và (10).

$$\det(\text{Covariance matrix} - \lambda I) = 0$$

$$\begin{vmatrix} 1,007 - \lambda & 0,996 \\ 0,996 & 0,999 - \lambda \end{vmatrix} = 0$$

$$\Leftrightarrow (1,007 - \lambda) * (0,999 - \lambda) - 0,996^2 = 0$$

Giải phương trình trên ta có được các giá trị riêng: $\lambda_1 = 1,999$, $\lambda_2 = 0,007$

Tìm các vector riêng bằng cách giải hệ phương trình:

$$(\text{Covariance matrix} - \lambda I)\vec{v} = 0$$

Ứng với $\lambda_1 = 1,999$:

$$\begin{bmatrix} 1,007 - 1,999 & 0,996 \\ 0,996 & 0,999 - 1,999 \end{bmatrix} \begin{bmatrix} v_{11} \\ v_{21} \end{bmatrix} = 0$$

$$\Leftrightarrow \begin{pmatrix} -0,992 & 0,996 \\ 0,996 & -1 \end{pmatrix} \begin{pmatrix} v_{11} \\ v_{21} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Giải hệ phương trình trên ta được: $v_{11} = 1,004v_{21}$

Để đơn giản ta cho $v_{21}=1$ để tính độ lớn của vector:

$$\|v_1\| = \sqrt{v_1^2 + v_2^2}$$

$$\|v_1\| = \sqrt{1,004^2 + 1^2} = 1,417$$

$$\text{Ta chuẩn hóa vector } V_1 = \frac{\vec{v}_1}{\|v_1\|} = V_1 = \frac{1}{1,417} \begin{bmatrix} 1,004 \\ 1 \end{bmatrix}$$

$$V_1 = \begin{bmatrix} 0,709 \\ 0,796 \end{bmatrix}$$

Ứng với $\lambda_2 = 0,007$:

$$\begin{bmatrix} 1,007 - 0,007 & 0,996 \\ 0,996 & 0,999 - 0,007 \end{bmatrix} \begin{bmatrix} v_{11} \\ v_{21} \end{bmatrix} = 0$$

$$\Leftrightarrow \begin{pmatrix} 1 & 0,996 \\ 0,996 & 0,992 \end{pmatrix} \begin{pmatrix} v_{11} \\ v_{21} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Giải hệ phương trình trên ta được: $v_{11} = -0,996v_{21}$

Để đơn giản ta cho $v_{21}=1$ để tính độ lớn của vector:

$$\|v_2\| = \sqrt{v_1^2 + v_2^2} = \sqrt{(-0,996)^2 + 1^2} = 1,411$$

$$\text{Ta chuẩn hóa vector: } V_2 = \frac{\vec{v}_2}{\|v_2\|} = V_2 = \frac{1}{1,411} \begin{bmatrix} -0,996 \\ 1 \end{bmatrix}$$

$$V_2 = \begin{bmatrix} -0,706 \\ 0,709 \end{bmatrix}$$

Bước 4: $\lambda_1 = 1,999$ là lớn nhất, nên vector riêng tương ứng sẽ là thành phần chính đầu tiên.

$$V_1 = \begin{bmatrix} 0,709 \\ 0,796 \end{bmatrix}$$

Bước 5: Chuyển đổi dữ liệu sang không gian của các thành phần chính bằng công thức (11):

$$Z = X * \text{Eigenvectors}$$

$$\text{Eigenvectors (vector riêng)} = \begin{bmatrix} 0,709 \\ 0,796 \end{bmatrix}$$

$$Z = \begin{bmatrix} -1,49 & -1,3 \\ -0,83 & -0,89 \\ -0,17 & -0,06 \\ 0,5 & 0,36 \\ 1,16 & 1,18 \\ -1,16 & -1,13 \\ -0,5 & -0,72 \\ 0,17 & 0,19 \\ 0,83 & 0,77 \\ 1,49 & 1,16 \end{bmatrix} * \begin{bmatrix} 0,709 \\ 0,796 \end{bmatrix} = \begin{bmatrix} -2,09 \\ -1,3 \\ -0,17 \\ 0,64 \\ 1,76 \\ -1,72 \\ -0,93 \\ -0,27 \\ 1,2 \\ 2,33 \end{bmatrix}$$

Z là kết quả cuối cùng là tập dữ liệu mới với chỉ một thành phần chính, giữ lại phần lớn thông tin từ dữ liệu ban đầu.

Đây là một phương pháp giúp giảm chiều của dữ liệu mà không làm mất quá nhiều thông tin quan trọng tuy nhiên với các bộ dữ liệu rất lớn, tính toán các thành phần chính có thể tốn kém về mặt thời gian và tài nguyên tính toán. Ngoài ra, do PCA hoàn toàn dựa trên các biến đổi tuyến tính nên nó không phù hợp với dữ liệu phi tuyến tính.

- **Linear Discriminant Analysis (LDA)**

LDA là một kỹ thuật thống kê được sử dụng để giảm chiều dữ liệu, chủ yếu được áp dụng trong các bài toán phân loại. LDA giúp tối ưu hóa sự tách biệt giữa các lớp khác nhau trong dữ liệu, đồng thời giảm số chiều của dữ liệu bằng cách chiếu dữ liệu ban đầu lên một không gian mới với số chiều thấp hơn. [15] [16]

Các bước thực hiện LDA:

Bước 1: Tính trung bình của các đặc trưng cho mỗi lớp trong tập dữ liệu.

Bước 2: Tìm ma trận hiệp phương sai cho từng lớp.

Bước 3: Tìm ma trận phương sai nội lớp (within class scatter matrix).

Bước 4: Tìm ma trận phương sai liên lớp (Between-Class Scatter Matrix).

Bước 5: Tính các giá trị riêng (eigen values).

Bước 6: Sắp xếp các giá trị đặc trưng và chọn k giá trị lớn nhất.

Bước 7: Tính toán các vector riêng (eigen vectors)

Bước 8: Tạo không gian LDA bằng cách nhân ma trận các vector riêng với dữ liệu gốc.

Đây là phương pháp mạnh mẽ và hiệu quả để giảm chiều dữ liệu và phân loại, đặc biệt hữu ích trong các tình huống mà số lượng đặc trưng vượt quá số lượng mẫu huấn luyện. Khả năng xử lý đa cộng tuyến (tương quan giữa các đặc trưng) của LDA làm cho nó trở thành một lựa chọn ổn định trong nhiều ứng dụng. Tuy nhiên, LDA bị hạn chế bởi một số giả định: nó yêu cầu dữ liệu phải tuân theo phân phối Gaussian, giả định ma trận hiệp phương sai giữa các lớp là bằng nhau, và hoạt động tốt nhất khi dữ liệu có thể được phân tách tuyến tính. Những hạn chế này có thể ảnh hưởng đến hiệu suất của LDA, đặc biệt là trong không gian đặc trưng có chiều cao hoặc khi phân phối dữ liệu không tuân theo các giả định này. [17]

2.15. Phân tích và tiền xử lý dữ liệu văn bản (Textual Data)

2.15.1. Các thư viện hỗ trợ

2.15.1.1. NLTK (Natural Language Toolkit)

Giới thiệu: NLTK là một trong những thư viện đầu tiên và phổ biến nhất cho xử lý ngôn ngữ tự nhiên trong Python. Nó được phát triển với mục đích giáo dục và nghiên cứu, nhưng cũng đủ mạnh để sử dụng trong các ứng dụng thực tế.

Các tính năng chính:

- **Tokenization:** NLTK cung cấp các công cụ để tách văn bản thành các từ hoặc câu. Đây là bước cơ bản trong xử lý văn bản.
- **Stemming và Lemmatization:** Thư viện cung cấp các công cụ để chuyển đổi các từ về gốc của chúng, giúp giảm bớt sự phức tạp của dữ liệu văn bản.
- **Stemming:** Là quá trình cắt bỏ phần cuối của từ để đưa về gốc. Ví dụ, "running" thành "run".
- **Lemmatization:** Là quá trình chuyển đổi từ về dạng gốc của nó dựa trên ngữ cảnh. Ví dụ, "better" thành "good".
- **POS Tagging:** Gán nhãn từ loại (danh từ, động từ, tính từ,...) cho các từ trong câu, giúp hiểu rõ hơn cấu trúc ngữ pháp của văn bản.
- **Named Entity Recognition (NER):** Nhận diện các thực thể có tên trong văn bản như tên người, địa điểm, tổ chức.

- **Parsing:** Phân tích cú pháp câu để hiểu cấu trúc ngữ pháp.
- **Corpus:** NLTK cung cấp nhiều tập dữ liệu văn bản (corpora) để thực hành và nghiên cứu.

2.15.1.2. SpaCy

Giới thiệu: spaCy là một thư viện NLP hiện đại và hiệu quả, được thiết kế để xử lý nhanh chóng và có thể mở rộng. Nó thường được sử dụng trong các ứng dụng thực tế và sản xuất nhờ tính hiệu quả và dễ sử dụng.

Các tính năng chính:

- **Tokenization:** Chia văn bản thành các từ hoặc câu một cách nhanh chóng và chính xác.
- **POS Tagging và Dependency Parsing:** Gán nhãn từ loại và phân tích quan hệ giữa các từ trong câu, giúp hiểu rõ cấu trúc ngữ pháp và ngữ nghĩa.
- **Named Entity Recognition (NER):** Nhận diện các thực thể có tên trong văn bản với độ chính xác cao.
- **Lemmatization:** Chuyển đổi các từ về gốc của chúng dựa trên ngữ cảnh, giúp giảm bớt sự phức tạp của dữ liệu văn bản.
- **Support cho nhiều ngôn ngữ:** spaCy hỗ trợ nhiều ngôn ngữ khác nhau, giúp mở rộng phạm vi ứng dụng.

2.15.1.3. TextBlob

Giới thiệu: TextBlob là một thư viện đơn giản nhưng mạnh mẽ để xử lý văn bản, xây dựng trên nền tảng của NLTK và Pattern. Nó cung cấp giao diện dễ sử dụng cho các nhiệm vụ NLP thông thường.

Các tính năng chính:

- **Sentiment Analysis:** Phân tích cảm xúc của văn bản, xác định xem văn bản mang tính tích cực, tiêu cực hay trung lập.
- **Tokenization:** Chia văn bản thành các từ hoặc câu một cách dễ dàng.
- **POS Tagging:** Gán nhãn từ loại cho các từ trong câu.
- **Translation và Language Detection:** Dịch và nhận diện ngôn ngữ văn bản, hỗ trợ nhiều ngôn ngữ khác nhau.
- **Lemmatization và Noun Phrase Extraction:** Chuyển đổi các từ về gốc của chúng và trích xuất các cụm danh từ.

2.15.2. Chuẩn bị dữ liệu

Dữ liệu văn bản thường cần trải qua một số bước xử lý trước khi có thể được phân tích một cách hiệu quả. Điều này là do dữ liệu văn bản thường rất lộn xộn. Nó có thể chứa thông tin không liên quan và đôi khi không có cấu trúc dễ dàng để phân tích. Một số bước thông thường để chuẩn bị dữ liệu văn bản bao gồm những điều sau:

- **Mở rộng các từ viết tắt:** Từ viết tắt là phiên bản rút gọn của một từ. Nó được tạo ra bằng cách loại bỏ một số chữ cái và thay thế chúng bằng dấu nháy đơn. Ví dụ, "don't" thay cho "do not" và "would've" thay cho "would have". Thông thường, khi chuẩn bị dữ liệu văn bản, tất cả các từ viết tắt nên được mở rộng về dạng ban đầu của chúng.
- **Loại bỏ dấu câu:** Dấu câu hữu ích để tách các câu, mệnh đề, hoặc cụm từ. Tuy nhiên, chúng thường không cần thiết cho phân tích văn bản vì chúng không truyền đạt ý nghĩa đáng kể.
- **Chuyển đổi về chữ thường:** Dữ liệu văn bản thường là sự kết hợp của chữ hoa và chữ thường. Tuy nhiên, điều này cần được chuẩn hóa để dễ dàng phân tích. Do đó, tất cả các chữ cái cần được chuyển về dạng chữ thường trước khi phân tích.
- **Thực hiện tách từ (Tokenization):** Tách từ liên quan đến việc chia nhỏ văn bản thành các đoạn nhỏ hơn, chẳng hạn như từ hoặc cụm từ. Việc chia nhỏ văn bản thành các đơn vị nhỏ hơn giúp chúng ta phân tích nó một cách hiệu quả hơn.
- **Loại bỏ các từ dừng (stop word):** Các từ dừng là những từ không thêm ý nghĩa đáng kể vào văn bản. Ví dụ bao gồm "the", "and", và "a". Điều này sẽ được đề cập chi tiết hơn trong công thức tiếp theo.

2.15.3. Loại bỏ từ dừng (stop word)

Từ dừng là những từ xuất hiện rất thường xuyên trong một ngôn ngữ. Chúng thường không thêm ý nghĩa đáng kể vào văn bản. Một số từ dừng phổ biến bao gồm đại từ, giới từ, liên từ và mạo từ. Trong tiếng Anh, ví dụ về từ dừng bao gồm "a", "an", "the", "and", "is", "was", "of", "for", và "not". Danh sách này có thể thay đổi dựa trên ngôn ngữ hoặc ngữ cảnh.

Trước khi phân tích văn bản, chúng ta nên loại bỏ từ dừng để có thể tập trung vào các từ có ý nghĩa hơn trong văn bản. Các từ dừng thường không có thông tin quan trọng và có

thể gây nhiễu trong bộ dữ liệu của chúng ta. Do đó, việc loại bỏ chúng giúp chúng ta dễ dàng tìm thấy những điểm chính và tập trung vào những gì quan trọng nhất.

Tuy nhiên, việc loại bỏ từ dừng phụ thuộc rất nhiều vào mục tiêu của phân tích và loại nhiệm vụ chúng ta thực hiện. Ví dụ, kết quả của nhiệm vụ phân tích cảm xúc có thể bị sai lệch do việc loại bỏ các từ dừng quan trọng. Điều này được nêu rõ ở đây:

- Câu mẫu: The food was not great.
- Câu mẫu sau khi loại bỏ từ dừng: Food great.

Câu đầu tiên là một đánh giá tiêu cực trong khi câu thứ hai sau khi loại bỏ từ dừng lại là một đánh giá tích cực. Cần phải cẩn thận khi loại bỏ từ dừng vì đôi khi việc loại bỏ từ dừng có thể làm thay đổi ý nghĩa của một câu.

Việc loại bỏ từ dừng là một trong những bước chúng ta thực hiện trong tiền xử lý văn bản, một quy trình chuẩn bị văn bản giúp chúng ta phân tích văn bản hiệu quả và chính xác hơn. [18]

2.15.4. Phân tích thành phần từng loại từ (POS)

Phân tích từ loại liên quan đến việc xác định và gán nhãn từ loại trong một văn bản cho trước. Điều này giúp hiểu cấu trúc ngữ pháp của các từ trong văn bản của chúng ta, điều này có thể rất hữu ích để trích xuất những thông tin quan trọng. Ví dụ, sử dụng phân tích từ loại, chúng ta có thể dễ dàng tìm thấy các thuật ngữ chính như danh từ, động từ, hoặc tính từ trong văn bản. Những thuật ngữ chính này có thể là chỉ dẫn đến các sự kiện chính, tên, sản phẩm, dịch vụ, địa điểm, từ mô tả, và nhiều hơn nữa.

Có chín loại từ trong tiếng Anh, chúng ta sẽ tập trung vào bốn loại quan trọng nhất.

- **Danh từ (Nouns):** Được sử dụng để đặt tên và xác định người, nơi chốn, hoặc vật thể – ví dụ, con mèo, quả bóng, hoặc London.
- **Động từ (Verbs):** Được sử dụng để biểu đạt hành động hoặc sự kiện. Chúng đi cùng với danh từ – ví dụ, chạy, nhảy, hoặc ngủ.
- **Tính từ (Adjectives):** Cung cấp thông tin bổ sung về danh từ hoặc đại từ – ví dụ, con cáo nhanh nhẹn, đôi giày nâu, hoặc thành phố đẹp.
- **Trạng từ (Adverbs):** Cung cấp thông tin bổ sung về động từ – ví dụ, chạy nhanh hoặc nói rõ ràng.

Part of Speech	Tag
Noun	n
Verb	v
Adjective	a
Adverb	r

Hình 2.1. Nhãn các loại từ trong tiếng anh

Hãy xem một số ví dụ:

- The boy ran quickly (Cậu bé chạy nhanh)
- Lagos is a beautiful city (Lagos là một thành phố đẹp)

Bây giờ, hãy gán nhãn từ loại cho ví dụ trên:

- **Boy** – danh từ (noun), **ran** – động từ (verb), và **quickly** – trạng từ (adverb)
- **Lagos** – danh từ (noun), **is** – động từ (verb), **beautiful** – tính từ (adjective), và **city** – danh từ (noun)

Chúng ta sẽ khám phá việc gán nhãn từ loại bằng hàm ‘pos_tag’ trong nltk. [19]

2.15.5. Thực hiện stemming và lemmatization

Stemming (Nguồn gốc) là kỹ thuật dùng để biến đổi một từ về dạng gốc (được gọi là stem hoặc root form) bằng cách cực kỳ đơn giản là loại bỏ một số ký tự nằm ở cuối từ mà nó nghĩ rằng là biến thể của từ. Có thể lấy ví dụ đơn giản như các từ *worked*, *working*, *works* chỉ khác nhau ở những ký tự cuối cùng, bằng cách bỏ đi các hậu tố *-ed*, *-ing*, *-s*, chúng ta đều được từ nguyên gốc là *work*. Bởi vì nguyên tắc hoạt động của Stemming rất đơn giản nên tốc độ xử lý của nó rất nhanh và kết quả stem đôi khi cho ra những kết quả không mong muốn. Một ví dụ khác như từ *goes* sẽ được stem thành từ *goe* (bỏ chữ s cuối từ) trong khi đó stem của từ *go* vẫn là *go*, kết quả là 2 từ *goes* và *go* sau khi được stem thì vẫn không giống nhau. Một nhược điểm khác là nếu các từ dạng bất quy tắc như *went* hay *spoke* thì kỹ thuật Stemming sẽ không thể đưa các từ này về dạng gốc là *go* hay *speak*. Tuy có các nhược điểm như trên nhưng trong thực tế Stemming vẫn được sử dụng khá phổ biến trong NLP vì nó có tốc độ xử lý nhanh và kết quả cuối cùng nhìn chung không hề tệ khi so với Lemmatization.

Lemmatization (Bổ đề ngôn ngữ) khác với Stemming – xử lý bằng cách loại bỏ các ký tự cuối từ một cách rất “máy móc”, Lemmatization sẽ xử lý thông minh hơn bằng một bộ từ điển hoặc một bộ ontology nào đó. Điều này sẽ đảm bảo rằng các từ

như **goes**, **went** và **go** sẽ chắc chắn có kết quả trả về là như nhau. Kể cả các từ danh từ như **mouse**, **mice** cũng đều được đưa về cùng một dạng như nhau. Người ta gọi bộ xử lý Lemmatization là Lemmatizer. Nhược điểm của lemmatization là tốc độ xử lý khá chậm vì phải thực hiện tra cứu từ trong cơ sở dữ liệu. Trong các ứng dụng Xử lý ngôn ngữ tự nhiên mà cần độ chính xác cao hơn và thời gian không quan trọng, người ta có thể sử dụng Lemmatization.

Chúng ta sẽ thực hiện stemming và lemmatization bằng cách sử dụng các lớp PorterStemmer và WordNetLemmatizer trong nltk. [20]

2.15.6. Phân tích Ngrams

N-gram là một chuỗi liên tục của n phần tử trong một văn bản cho trước. Các phần tử này có thể là từ, chữ cái hoặc âm tiết. N-grams giúp chúng ta trích xuất thông tin hữu ích về sự phân bố của từ, âm tiết hoặc chữ cái trong một văn bản cho trước. N đại diện cho các giá trị số dương, bắt đầu từ 1 đến n. Các n-grams phổ biến nhất là unigram, bigram và trigram, trong đó n lần lượt là 1, 2 và 3.

Phân tích n-grams liên quan đến việc kiểm tra tần suất hoặc sự phân bố của một n-gram trong một văn bản. Chúng ta thường chia văn bản thành n-gram tương ứng và đếm tần suất của từng n-gram trong dữ liệu văn bản. Điều này sẽ giúp chúng ta xác định các từ, âm tiết hoặc cụm từ phổ biến nhất trong dữ liệu.

Ví dụ, trong câu "This is a sentence" các n-grams sẽ như sau:

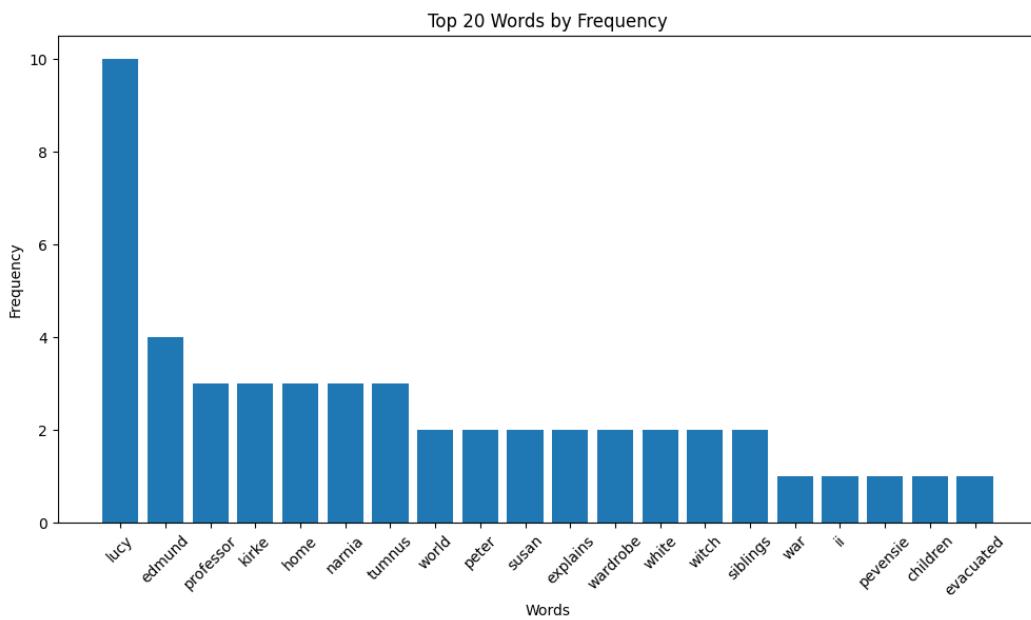
- 1-gram (hoặc unigram): ["This", "is", "a", "sentence"]
- 2-gram (hoặc bigram): ["This is", "is a", "a sentence"]
- 3-gram (hoặc trigram): ["This is a", "is a sentence"]

Chúng ta sẽ khám phá phân tích n-gram bằng hàm ngram trong nltk. [21]

2.15.7. Tạo word clouds

Đám mây từ (word cloud) là một biểu diễn trực quan của các từ phổ biến nhất trong một văn bản. Nó đo lường tần suất của mỗi từ trong văn bản và biểu diễn tần suất này bằng kích thước của từ. Các từ lớn hơn xuất hiện thường xuyên hơn trong văn bản, trong khi các từ nhỏ hơn xuất hiện ít thường xuyên hơn. Đám mây từ cung cấp một bản tóm tắt rất hữu ích về sự phân bố từ trong một văn bản. Nó cũng là một cách tuyệt vời để có được những cái nhìn nhanh về các từ nổi bật trong dữ liệu văn bản.

Ví dụ: Cho đoạn văn sau và tạo word cloud cho nó: “During World War II, the Pevensie children, Peter, Susan, Edmund and Lucy, are evacuated from a London suburb to Professor Digory Kirke’s country home. Mrs. Macready, the strict housekeeper, explains he is unaccustomed to hosting children. While the Pevensies play hide-and-seek, Lucy discovers a wardrobe and enters the fantasy world called Narnia. Seeing a lamppost, Lucy encounters a faun named Mr. Tumnus, who invites her to his home. He puts Lucy to sleep by playing a lullaby on his flute. When Lucy wakes up, she finds Tumnus grieving. He explains that Jadis, the White Witch, has cursed Narnia to a hundred years of winter. If a human is encountered, they are to be brought to her. Tumnus cannot bring himself to kidnap Lucy, so he sends her home. When she returns to Professor Kirke’s house, hardly any time has passed, and her siblings disbelieve her story. One night, Edmund follows Lucy into the wardrobe, entering Narnia. While searching for Lucy, he meets the White Witch, who claims to be queen. She offers him tea and Turkish Delight and the prospect of becoming king if he brings his siblings to her castle. After she departs, Edmund and Lucy meet again and return. Lucy tells Peter and Susan what happened, but Edmund lies. Professor Kirke suggests she is telling the truth, though they remain unconvinced.”



Hình 2.2. Top 20 từ có tần suất xuất hiện nhiều nhất trong đoạn văn

Tần suất xuất hiện của các từ:

'lucy': 10, 'edmund': 4, 'professor': 3, 'kirke': 3, 'home': 3, 'narnia': 3, 'tumnus': 3, 'world': 2, 'peter': 2, 'susan': 2, 'explains': 2, 'wardrobe': 2, 'white': 2, 'witch': 2, 'siblings': 2, 'war': 1, 'ii': 1, 'pevensie': 1, 'children': 1, 'evacuated': 1



Hình 2.3. Hình ảnh Word Cloud được tạo từ đoạn văn

Từ *Hình 2.3* được tạo cho thấy các từ có tần suất xuất hiện càng nhiều trong đoạn văn thì từ đó sẽ càng lớn khi được thể hiện trong Word Cloud.

Chúng ta sẽ khám phá cách tạo đám mây từ trong Python bằng cách sử dụng thư viện wordcloud và lớp FreqDist trong nltk. [22]

2.15.8. Vector hóa dữ liệu (TF-IDF)

Tần suất từ (TF) đo lường tần suất xuất hiện của một từ (hoặc cụm từ) cụ thể trong văn bản. Nó hiển thị số lần từ xuất hiện trong văn bản so với tổng số từ trong văn bản đó. Điều này rất hữu ích để phân tích tầm quan trọng của một từ trong văn bản. Nó được tính bằng cách chia số lần từ xuất hiện trong văn bản cho tổng số từ trong văn bản đó:

$$TF(t, d) = \frac{\text{Số lần xuất hiện của từ } t}{\text{Tổng số từ trong 1 document}} \quad (12)$$

Kết quả trên là một giá trị giữa 0 và 1, đại diện cho tần suất tương đối của từ trong văn bản. Tần suất từ thường được kết hợp với Tần suất Ngược tài liệu (IDF) để tạo ra một thước đo tốt hơn về mức độ liên quan hoặc tầm quan trọng của từ. Trước khi thảo luận về IDF, chúng ta cần nói về hai khái niệm thường được sử dụng khi phân tích văn bản – tài liệu và tập hợp tài liệu (corpora). Nói đơn giản, một corpus là một tập hợp các tài liệu. Mặt khác, tài liệu là một đơn vị văn bản thường nhỏ hơn một corpus. Nó có thể chỉ một văn bản, câu, bài viết hoặc đoạn văn. Hãy xem ví dụ sau:

- This is a document
- This is another document

- This is yet another document
- All these documents form a corpus

Trong danh sách trên, mỗi gạch đầu dòng là một tài liệu, trong khi tập hợp tất cả các gạch đầu dòng (tài liệu) là một corpus. IDF đo lường mức độ hiếm hoi của một từ trong “corpus,” trong khi tần suất từ (TF) đo lường tần suất xuất hiện của một từ trong “tài liệu”. Khi kết hợp, chúng ta có một thước đo tiên tiến về tầm quan trọng của từ gọi là Tần suất Từ – Tần suất Ngược tài liệu (TF-IDF).

TF-IDF cho một trọng số lớn cho những từ xuất hiện thường xuyên trong một tài liệu nhưng hiếm khi trong toàn bộ corpus, và một trọng số thấp cho những từ xuất hiện thường xuyên cả trong tài liệu và corpus. Những từ sau thường là các bài báo, liên từ và giới từ không có tầm quan trọng đáng kể trong tài liệu hoặc corpus. IDF được tính bằng cách chia số tài liệu trong corpus cho số tài liệu trong corpus chứa từ đó, sau đó lấy logarit của kết quả. TF-IDF được tính bằng cách nhân tần suất từ (TF) với tần suất ngược tài liệu (IDF):

$$IDF(t, D) = \log \frac{\text{Số văn bản trong tập } D}{\text{Số văn bản chứa từ } t \text{ trong tập } D} \quad (13)$$

$$TF - IDF(t, d, D) = TF(t, d) * IDF(t, D) \quad (14)$$

Ví dụ:

- Document 1: It is going to rain today.
- Document 2: Today I am not going outside.
- Document 3: I am going to watch the season premiere.

Bước 1: Làm sạch dữ liệu và tách từ

Trước tiên, chúng ta cần làm sạch dữ liệu bằng cách chuẩn hóa (standardize) dữ liệu, chuẩn hóa (normalize) dữ liệu về chữ thường, loại bỏ từ dừng (stop word) và lemmatize (chuyển từ về dạng gốc). Sau đó, tách từ (tokenize) dữ liệu để đưa về dạng các từ đơn. Tiếp theo là đếm số lần xuất hiện của các từ trong cả 3 document (Bảng 2.20).

Bảng 2.20. Số lần xuất hiện của các từ trong 3 documents

Words/Documents	Count
going	3
to	2
today	2
i	2
am	2

it	1
is	1
rain	1

Bước 2: Tìm TF (Tần suất Từ)

- Document 1: It is going to rain today.

Ví dụ: $TF(going, doc1) = \frac{1}{6} = 0.16$

Tính TF tương tự với các từ còn lại trong document 1 và các document còn lại (Bảng 2.21).

Bảng 2.21. TF của cả 3 Document

Words/Documents	Document 1	Document 2	Document 3
going	0.16	0.16	0.12
to	0.16	0	0.12
today	0.16	0.16	0
i	0	0.16	0.12
am	0	0.16	0.12
it	0.16	0	0
is	0.16	0	0
rain	0.16	0	0

Bước 3: Tìm IDF

Ví dụ: $IDF(going, D) = \log \frac{3}{3} = 0$

Bảng 2.22. Kết quả tính IDF

Words	IDF Value
going	0
to	0.41
today	0.41
i	0.41
am	0.41
it	1.09
is	1.09
rain	1.09

Bước 4: Tính TF-IDF

Ví dụ: $TF - IDF(going, doc1, D) = TF(t, d) * IDF(t, D) = 0,16 * 0 = 0$

Bảng 2.23. Kết quả tính TF-IDF

Words/Documents	going	to	today	i	am	it	is	rain
Document 1	0	0.1	0.07	0	0	0.2	0.2	0.2
Document 2	0	0	0.07	0.1	0.1	0	0	0
Document 3	0	0.1	0	0.1	0.1	0	0	0

Có thể dễ dàng thấy rằng các từ như 'it', 'is', 'rain' quan trọng đối với tài liệu 1 nhưng không quan trọng đối với tài liệu 2 và tài liệu 3, điều này có nghĩa là tài liệu 1 khác với tài liệu 2 và 3 về nội dung nói về mưa.

Cũng có thể nói rằng tài liệu 1 và 2 nói về điều gì đó 'today', và tài liệu 2 và 3 thảo luận về người viết vì có từ 'T'. Bảng 2.23 giúp bạn tìm thấy sự tương đồng và khác biệt giữa các tài liệu, từ ngữ.

Chúng ta sẽ khám phá cách thực hiện TF-IDF trong Python bằng cách sử dụng lớp TfidfVectorizer trong thư viện scikit-learn.

Ngoài ra,

Còn một khái niệm khác liên quan đến TF-IDF là **Bag of Words (BoW)**. Phương pháp BoW chuyển đổi văn bản thành các vector có độ dài cố định bằng cách đếm số lần một từ xuất hiện trong văn bản. Khi tạo BoW, chúng ta thường xác định danh sách các từ duy nhất trong tài liệu hoặc corpus. Sau đó, chúng ta đếm số lần mỗi từ xuất hiện. Điều này tạo thành các vector có độ dài cố định. Trong phương pháp BoW, thứ tự của các từ không thực sự được xem xét; thay vào đó, trọng tâm là số lần một từ xuất hiện trong tài liệu.

Ví dụ sau đây giải thích rõ hơn:

- The boy is tall
- The boy went home
- The boy kicked the ball

Danh sách các từ duy nhất trong corpus trên là {"the", "boy", "is", "tall", "went", "home", "kicked", "ball"}. Đây là danh sách gồm tám từ. Để tạo mô hình BoW, chúng ta đếm số lần mỗi từ xuất hiện, như Bảng 2.24 được hiển thị dưới đây :

Bảng 2.24. Ví dụ Bag of Words

Document	the	boy	is	tall	went	home	kicked	ball
1	1	1	1	1	0	0	0	0
2	1	1	0	0	1	1	0	0

3	1	1	0	0	0	0	1	1
---	---	---	---	---	---	---	---	---

Từ bảng trên, chúng ta có thể thấy rằng các tài liệu đã được chuyển đổi thành các vector có độ dài cố định với độ dài cố định là tám. Thông thường, lemmatization hoặc stemming nên được thực hiện trước khi tạo BoW để giảm chiều của mô hình BoW.

BoW là một kỹ thuật đơn giản chỉ đếm tần suất của các từ trong một tài liệu, trong khi TF-IDF là một kỹ thuật phức tạp hơn, xem xét cả tần suất của các từ trong một tài liệu và toàn bộ corpus. TF-IDF là một đại diện tốt hơn về sự liên quan/tầm quan trọng của các từ trong một tài liệu. [23] [24]

2.16. Phân tích và tiền xử lý dữ liệu chuỗi thời gian (Time-series Data)

2.16.1. Các yếu tố cần lưu ý khi phân tích dữ liệu chuỗi thời gian

2.16.1.1. Trend (Xu hướng)

Xu hướng trong chuỗi thời gian đề cập đến sự chuyển động lên hoặc xuống của dữ liệu trong khoảng thời gian dài. Xu hướng (Trend) đại diện cho cấu trúc cơ bản của dữ liệu, nắm bắt hướng và mức độ thay đổi trong một khoảng thời gian dài hơn. Trong phân tích chuỗi thời gian, việc mô hình hóa và loại bỏ xu hướng từ dữ liệu là một bước quan trọng để hiểu rõ hơn các mẫu cơ bản và đưa ra dự báo chính xác hơn.

Một số xu hướng phổ biến khi phân tích Time series data:

- **Upward trend:** giá trị của dữ liệu có xu hướng tăng theo thời gian
- **Downward trend:** giá trị của dữ liệu có xu hướng giảm theo thời gian
- **Horizontal trend:** giá trị của dữ liệu không có sự thay đổi đáng kể hoặc không đổi theo thời gian
- **Damped trend:** giá trị của dữ liệu giảm dần theo thời gian, nhưng càng về sau tốc độ thay đổi càng chậm lại
- **Non-linear trend:** giá trị của dữ liệu thay đổi không theo một xu hướng chung mà phức tạp hơn, có thể bao gồm tăng giảm đổi hướng hoặc thay đổi đột biến theo thời gian

Điều quan trọng cần lưu ý là dữ liệu chuỗi thời gian có thể có sự kết hợp của các loại xu hướng này hoặc nhiều xu hướng xuất hiện đồng thời. Việc xác định và mô hình hóa chính xác xu hướng là một bước quan trọng trong phân tích chuỗi thời gian, vì nó có thể ảnh hưởng đáng kể đến độ chính xác của dự báo các mẫu trong dữ liệu.

2.16.1.2. Seasonality (Tính mùa vụ)

Tính mùa vụ trong dữ liệu chuỗi thời gian đề cập đến những biến động tăng hoặc giảm lặp đi lặp lại một cách đều đặn của dữ liệu trong một khoảng thời gian

Một số tính mùa vụ phổ biến khi phân tích Time Series data:

- **Weekly Seasonality:** Sự thay đổi lặp lại trong khoảng thời gian 7 ngày. Ví dụ: số lượng vé xem phim tại các rạp tăng mạnh vào các dịp cuối tuần.
- **Monthly Seasonality:** Sự thay đổi lặp lại trong khoảng thời gian 30 hoặc 31 ngày. Ví dụ: Chi tiêu của người dùng shopee tăng vọt vào các đợt sales định kỳ hàng tháng.
- **Annual Seasonality:** Sự thay đổi lặp lại trong khoảng thời gian 365 hoặc 366 ngày. Ví dụ: Số lượng khách du lịch tăng vọt vào các đợt cao điểm tháng hè.
- **Holiday Seasonality:** Sự thay đổi này thường được gây ra bởi các sự kiện đặc biệt như ngày lễ, lễ hội, sự kiện thể thao. Ví dụ: Doanh số các hàng bán lẻ tăng mạnh vào dịp sát Tết.

2.16.1.3. Chu kỳ (Cyclicity)

Tính chu kỳ trong dữ liệu chuỗi thời gian đề cập đến những biến động lặp đi lặp lại, hoặc những thay đổi định kỳ, có thể kéo dài trong nhiều năm và chuyển từ giai đoạn này qua giai đoạn khác.

Sự khác biệt giữa tính mùa vụ (Seasonality) và tính chu kỳ (Cyclicity):

- Tính mùa vụ (Seasonality) đề cập đến sự thay đổi lặp lại, xảy ra trong một khoảng thời gian cố định (hàng ngày, hàng tuần, hàng tháng hoặc hàng năm), bị ảnh hưởng bởi nhiều yếu tố như thời tiết, các ngày lễ, hành vi người dùng,... và có thể dự đoán được.
- Mặt khác, tính chu kỳ (Cyclicity) đề cập đến các biến động lặp đi lặp lại trong một khoảng thời gian không xác định. Những biến động này thường bị ảnh hưởng bởi những yếu tố ví mô hơn như chu kỳ kinh tế, xu hướng thị trường,... Tính chu kỳ không bị giới hạn trong một khoảng thời gian cố định và có thể có diễn ra với tần suất khác nhau, nên việc dự đoán sẽ trở nên khó khăn hơn.

2.16.1.4. Irregularity (Sự bất thường):

Sự bất thường trong dữ liệu chuỗi thời gian đề cập đến những sự thay đổi bất thường của dữ liệu, xảy ra một cách ngẫu nhiên, có thể trái ngược hoàn toàn với các dữ liệu trong quá khứ, khó có thể giải thích và không dự đoán được trước.

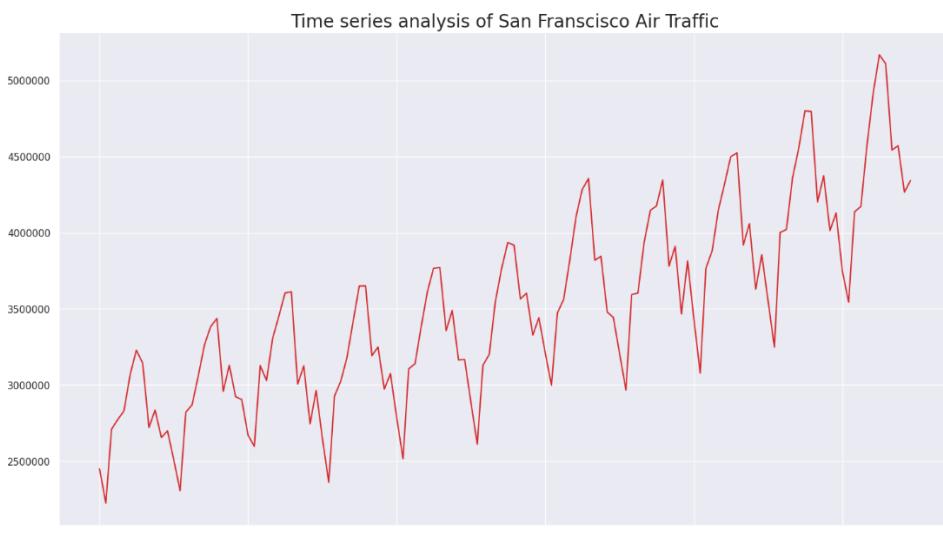
Sự bất thường này có thể do sự sai sót trong đo lường dữ liệu, hoặc những sự kiện bất ngờ diễn ra và có thể vô tình làm ảnh hưởng đến tính chính xác khi đánh giá dữ liệu chuỗi thời gian. [25]

2.16.2. Sử dụng biểu đồ đường và biểu đồ hộp để trực quan hóa dữ liệu chuỗi thời gian

Biểu đồ đường kết nối các điểm dữ liệu chuỗi thời gian thông qua một đường thẳng, hiển thị các đỉnh (điểm cao) và thung lũng (điểm thấp) trong dữ liệu. Trục x của biểu đồ đường thường đại diện cho các khoảng thời gian của chúng ta, trong khi trục y đại diện cho một biến số mà chúng ta muốn theo dõi liên quan đến thời gian. Với biểu đồ đường, rất dễ dàng để nhận ra các xu hướng hoặc thay đổi theo thời gian.

Ví dụ: Biểu đồ đường *Hình 2.4* thể hiện lưu lượng hành khách tại San Francisco qua từng năm. Trục x đại diện cho các khoảng thời gian và trục y đại diện cho số lượng hành khách.

- Biểu đồ cho thấy số lượng hành khách có xu hướng tăng dần từ năm 2006 đến năm 2017.
- Có sự biến động rõ rệt theo mùa, với các đỉnh và đáy thường xuyên xuất hiện, cho thấy có những khoảng thời gian cao điểm và thấp điểm trong năm.



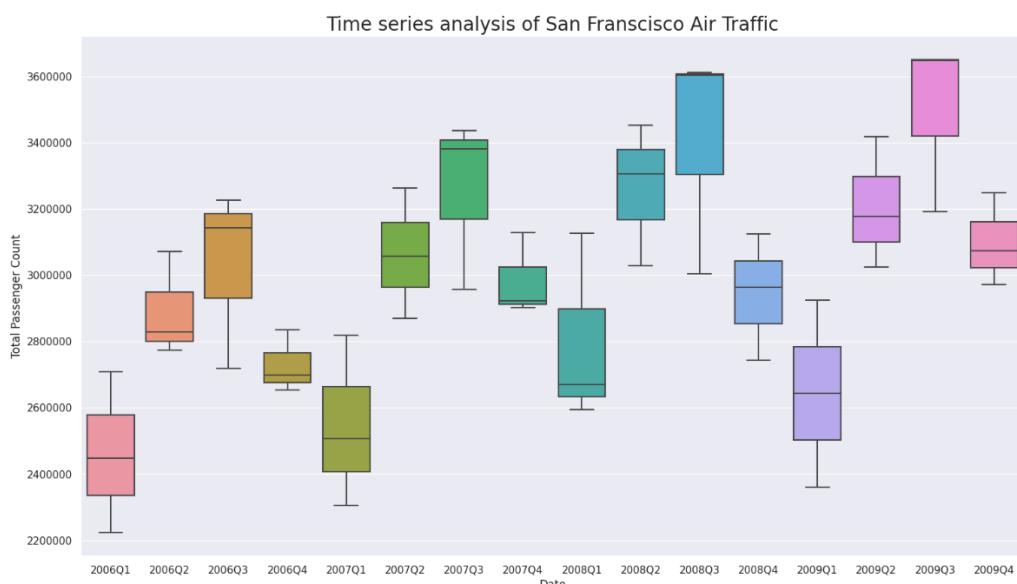
Hình 2.4. Biểu đồ đường phân tích chuỗi thời gian của lưu lượng hành khách tại San Francisco

Mặt khác, **biểu đồ hộp** cung cấp cho chúng ta một cái nhìn tổng quan về phân phối cơ bản của một tập dữ liệu thông qua năm chỉ số quan trọng. Các chỉ số bao gồm giá trị nhỏ nhất, tứ phân vị thứ nhất, trung vị, tứ phân vị thứ ba và giá trị lớn nhất. Khi sử dụng trên dữ liệu chuỗi thời gian, trực x thường đại diện cho các khoảng thời gian, trong khi trực y đại diện cho một biến số quan tâm. Các khoảng thời gian trên trực x thường được tóm tắt lại, ví dụ, giờ tóm tắt thành ngày, ngày tóm tắt thành tháng, hoặc năm.

Biểu đồ hộp cho thấy trung bình của biến số trên trực y thay đổi theo thời gian như thế nào. Nó cũng cho thấy sự phân tán của các điểm dữ liệu trong một khoảng thời gian. Các trực của biểu đồ hộp cần được tỷ lệ chính xác để tránh kết quả gây hiểu lầm. Tỷ lệ sai của trực y có thể dẫn đến nhiều ngoại lệ hơn dự kiến.

Ví dụ: Biểu đồ hộp *Hình 2.5* thể hiện lưu lượng hành khách tại San Francisco qua các quý trong năm.

- Biểu đồ hộp chia nhỏ dữ liệu theo từng quý từ năm 2006 đến năm 2009, giúp ta dễ dàng so sánh sự phân bố của số lượng hành khách theo từng khoảng thời gian.
- Có sự biến động mạnh về số lượng hành khách giữa các quý. Ví dụ, quý 2 và quý 3 năm 2008 có số lượng hành khách rất cao, trong khi quý 4 năm 2008 và quý 1 năm 2009 có số lượng hành khách thấp hơn.
- Các hộp biểu đồ cho thấy sự phân bố dữ liệu khá rộng trong một số quý, cho thấy có nhiều biến động về số lượng hành khách trong các khoảng thời gian này.



Hình 2.5. Biểu đồ hộp thể hiện lưu lượng hành khách tại San Francisco qua các quý trong năm

Để khám phá cách trực quan hóa dữ liệu chuỗi thời gian bằng biểu đồ đường và biểu đồ hộp. Chúng ta sẽ sử dụng phương pháp plot trong matplotlib và phương pháp boxplot trong seaborn để làm điều này. [10]

2.16.3. Nhận diện các mẫu trong chuỗi thời gian

Có bốn loại mẫu mà chúng ta thường tìm kiếm khi phân tích dữ liệu chuỗi thời gian. Chúng bao gồm xu hướng, biến động theo mùa, biến động theo chu kỳ, và biến động bất thường. Biểu đồ đường (line plots) là công cụ rất hữu ích để phân tích các mẫu này. Với biểu đồ đường, chúng ta có thể dễ dàng nhận diện các mẫu trong tập dữ liệu của mình.

- Khi phân tích dữ liệu để tìm xu hướng, chúng ta cố gắng xác định sự tăng hoặc giảm dài hạn trong các giá trị của chuỗi thời gian. Xu hướng là một chiều hướng dài hạn của dữ liệu, không bị ảnh hưởng bởi các yếu tố ngắn hạn.
- Khi phân tích dữ liệu để tìm biến động theo mùa, chúng ta cố gắng nhận diện các mẫu định kỳ bị ảnh hưởng bởi lịch (quý, tháng, ngày trong tuần, và v.v.). Biến động theo mùa là những thay đổi lặp đi lặp lại có thể dự đoán được trong một khoảng thời gian nhất định (ví dụ: hàng năm, hàng quý).
- Khi phân tích dữ liệu để tìm biến động theo chu kỳ, chúng ta cố gắng nhận diện các đoạn mà tại đó các điểm dữ liệu tăng và giảm với các mức độ khác nhau và trong các khoảng thời gian dài hơn, không cố định. Ví dụ: chu kỳ có thể kéo dài ít nhất hai năm, và mỗi chu kỳ có thể xảy ra trong một khoảng thời gian không cố định (như mỗi hai đến bốn năm) thay vì một khoảng thời gian cố định (như mỗi hai năm).
- Biến động bất thường là các biến động không thể dự đoán được và không tuân theo bất kỳ mẫu cụ thể nào. Chúng thường do các sự kiện ngẫu nhiên hoặc các yếu tố bên ngoài gây ra.

Chúng ta sẽ nhận diện các mẫu trong dữ liệu chuỗi thời gian bằng cách sử dụng biểu đồ đường và biểu đồ hộp (box plot). Chúng ta sẽ sử dụng phương pháp plot trong thư viện matplotlib và phương pháp boxplot trong thư viện seaborn để thực hiện điều này.

- **Biểu đồ đường (Line Plot):** Giúp hiển thị các xu hướng, biến động theo mùa và chu kỳ trong dữ liệu.
- **Biểu đồ hộp (Box Plot):** Giúp nhận diện các biến động bất thường và phân tích sự phân bố dữ liệu theo các khoảng thời gian cụ thể.

Ví dụ: Hai hình ảnh biểu đồ đường (Hình 2.4), biểu đồ hộp (Hình 2.5)

- Từ biểu đồ đường (Hình 2.4), chúng ta có thể nhận diện xu hướng tăng lên, dựa vào sự gia tăng giá trị của số lượng hành khách. Chúng ta cũng có thể nhận diện tính mùa vụ bằng các dao động nhất quán hàng năm; chúng ta nhận thấy các đỉnh đáng kể vào khoảng giữa năm. Đây có thể là mùa hè. Dữ liệu chuỗi thời gian không có các dao động theo chu kỳ.
- Từ biểu đồ hộp (Hình 2.5), chúng ta có thể nhận diện các dao động theo các quý trong năm. Quý 2 (Q2) và quý 3 (Q3) thường như có số lượng hành khách cao nhất trong các năm. [10]

2.16.4. Thực hiện phân tách dữ liệu chuỗi thời gian

Phân rã (decomposition) là quá trình tách dữ liệu chuỗi thời gian thành các thành phần riêng lẻ để hiểu rõ hơn về các mô hình tiềm ẩn. Nhìn chung, phân rã giúp chúng ta hiểu rõ hơn về các mô hình tiềm ẩn trong chuỗi thời gian. Các thành phần được định nghĩa như sau:

- **Xu hướng (Trend):** Sự tăng hoặc giảm dài hạn của các giá trị trong chuỗi thời gian.
- **Tính mùa vụ (Seasonality):** Các biến động trong chuỗi thời gian bị ảnh hưởng bởi các yếu tố mùa vụ (ví dụ: quý, tháng, tuần hoặc ngày).
- **Dư lượng (Residual):** Các mô hình còn lại sau khi đã tính đến xu hướng và tính mùa vụ. Nó cũng được coi là nhiễu (biến động ngẫu nhiên trong chuỗi thời gian).

Như bạn có thể đã nhận thấy, các biến động chu kỳ được đề cập trong các công thức trước đây không xuất hiện như một thành phần trong chuỗi thời gian phân rã. Nó thường được kết hợp với thành phần xu hướng và gọi là xu hướng.

Khi phân rã chuỗi thời gian của chúng ta, chúng ta có thể xem xét chuỗi thời gian như là sự kết hợp cộng hoặc nhân của các thành phần xu hướng, tính mùa vụ và dư lượng. Công thức cho mỗi mô hình được hiển thị như sau:

- **Mô hình cộng (Additive Model):**

$$Y=T+S+R \quad (15)$$

Trong đó, Y là chuỗi, T là xu hướng, S là tính mùa vụ, và R là thành phần dư lượng.

Ta thấy trong công thức trên thì ảnh hưởng của S lên Y có hệ số không đổi là 1 nên không tạo ra khác biệt quá lớn trong tương lai lên chuỗi. Do đó lớp mô hình này mới phù hợp với các chuỗi có biến động ổn định.

Chương 2. Cơ sở lý thuyết

Trái lại, mô hình nhân sẽ phù hợp với những chuỗi có độ biến động gia tăng theo thời gian.

- **Mô hình nhân (Multiplicative Model):**

$$Y = T \times S \times R \quad (16)$$

Sự ảnh hưởng của S lên Y đã gia tăng theo cấp số T. Cấp số này sẽ gia tăng theo thời gian nên càng xa trong tương lai thì độ biến động của chuỗi càng có xu hướng gia tăng mạnh hơn.

Ví dụ: Giả sử rằng doanh số hàng tháng của một doanh nghiệp trong một giai đoạn là (Bảng 2.25):

Bảng 2.25. Doanh số hàng tháng của một doanh nghiệp

Tháng	Doanh số (Y)
1	70
2	80
3	150
4	130
5	140
6	210

Tính T (xu hướng)

Trung bình động (Moving Average) là một phương pháp phổ biến để xác định xu hướng. Giả sử chúng ta sử dụng trung bình động 3 tháng để làm mượt dữ liệu (Bảng 2.26).

Bảng 2.26. Trung bình động 3 tháng

Tháng	Doanh số (Y)	Trung bình động 3 tháng (T)
1	70	
2	80	$(70 + 80 + 150)/3 = 100$
3	150	$(80 + 150 + 130)/3 = 120$
4	130	$(150 + 130 + 140)/3 = 140$
5	140	$(130 + 140 + 210)/3 = 160$
6	210	

Tính S (tính mùa vụ)

Bảng 2.27. Biến động mùa vụ

Tháng	Doanh số (Y)	Trung bình động 3 tháng (T)	Biến động mùa vụ (S)
1	70		
2	80	100	$80 - 100 = -20$
3	150	120	$150 - 120 = +30$
4	130	140	$130 - 140 = -10$
5	140	160	$140 - 160 = -20$

Tính R (dư lượng)

Bảng 2.28. Dư lượng

Tháng	Doanh số (Y)	Trung bình động 3 tháng (T)	Biến động mùa vụ (S)	Dư lượng (R)
1	70			
2	80	100	$80 - 100 = -20$	$80 - 100 - (-20) = 0$
3	150	120	$150 - 120 = +30$	$150 - 120 - 30 = 0$
4	130	140	$130 - 140 = -10$	$130 - 140 - (-10) = 0$
5	140	160	$140 - 160 = -20$	$140 - 160 - (-20) = 0$
6	210			

Chúng ta sẽ khám phá các mô hình phân rã cộng và nhân bằng cách sử dụng seasonal_decompose trong statsmodel. [26]

2.16.5. Thực hiện làm mịn dữ liệu - trung bình di động (Moving Average)

Làm mịn thường được thực hiện như một phần của phân tích chuỗi thời gian để giúp nhận diện các mẫu và xu hướng rõ ràng hơn. Các kỹ thuật làm mịn giúp loại bỏ nhiễu từ dữ liệu chuỗi thời gian, giảm bớt tác động của các dao động ngắn hạn và làm nổi bật các mẫu cơ bản.

Trung bình động là một kỹ thuật làm mịn phổ biến, giúp xác định xu hướng và các mẫu trong dữ liệu chuỗi thời gian. Kỹ thuật này tính trung bình của một tập hợp các điểm dữ liệu liền kề trong chuỗi thời gian, với số lượng điểm dữ liệu xác định bởi kích thước cửa sổ (window). Cửa sổ (window) là số kỳ mà chúng ta muốn tính trung bình, thường được chọn dựa trên tần suất của dữ liệu và bản chất của các mẫu mà chúng ta muốn xác định. Ví dụ, với dữ liệu giá cổ phiếu hàng ngày, chúng ta có thể tính trung bình giá trong 10 ngày qua, bao gồm cả ngày hiện tại. Trong trường hợp này, 10 ngày là kích thước của cửa sổ.

Để đạt được phần "di động", chúng ta sẽ trượt cửa sổ dọc theo chuỗi thời gian để tính giá trị trung bình. Nói cách khác, vào ngày 20 tháng 1, cửa sổ 10 ngày sẽ bao gồm từ ngày 11 tháng 1 đến ngày 20 tháng 1, trong khi vào ngày 21 tháng 1, cửa sổ sẽ bao gồm từ ngày 12 tháng 1 đến ngày 21 tháng 1.

Thông thường, chúng ta sẽ hình dung trung bình động so với chuỗi thời gian gốc để nhận diện bất kỳ xu hướng hoặc mẫu nào hiện diện. Cửa sổ ngắn thường được sử dụng để

Chương 2. Cơ sở lý thuyết

năm bắt các dao động ngắn hạn hoặc thay đổi nhanh chóng trong dữ liệu, trong khi cửa sổ dài được sử dụng để nắm bắt các xu hướng dài hạn và làm mịn các dao động ngắn hạn.

Làm mịn cũng có thể được sử dụng để xác định các giá trị ngoại lai hoặc các thay đổi đột ngột trong chuỗi thời gian.

Để hiểu rõ hơn về phương pháp này, dưới đây là 1 ví dụ:

Cho số liệu bán hàng của một doanh nghiệp từ 2000 đến 2009, Moving Averages của doanh thu mỗi 3 năm sẽ được tính bằng trung bình cộng của 3 năm gần nhất trở về trước theo công thức:

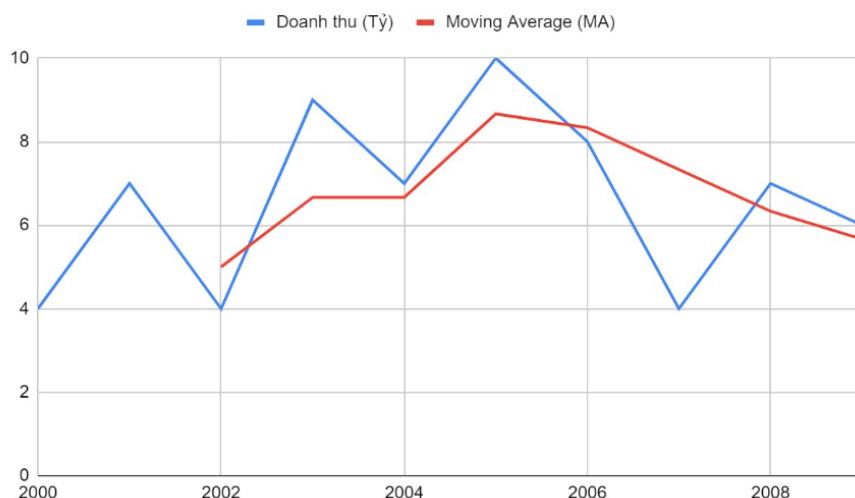
Có thể theo dõi trong *Bảng 2.29* dưới đây:

$$\begin{aligned} \text{MA}(2002) &= (\text{Doanh thu } 2000 + \text{Doanh thu } 2001 + \text{Doanh thu } 2002)/3 \\ &= (4+7+4)/3 = 5 \end{aligned}$$

Bảng 2.29. Số liệu bán hàng của một doanh nghiệp từ 2000 đến 2009

Năm	Doanh thu (Tỷ)	Moving Average (MA)
2000	4	N/A
2001	7	N/A
2002	4	5.00
2003	9	6.67
2004	7	6.67
2005	10	8.67
2006	8	8.33
2007	4	7.33
2008	7	6.33
2009	6	5.67

Khi đó đường trung bình động sẽ được biểu diễn như *Hình 2.6* dưới đây:



Hình 2.6. Biểu đồ ví dụ Moving Average

Chúng ta sẽ khám phá kỹ thuật làm mịn trung bình động trong Python, sử dụng các phương pháp ‘rolling’ và ‘mean’ trong pandas.

Ưu điểm của phương pháp Moving Average là cách tính toán đơn giản, dễ hiểu, có thể thấy các giá trị gây nhiễu đã được loại bỏ bớt, bạn có thể dễ dàng hiểu được xu hướng của dữ liệu thông qua việc phân tích các yếu tố như xu hướng, tính mùa vụ, tính chu kỳ,...

Tuy nhiên nhược điểm của phương pháp này là nó dựa hoàn toàn vào dữ liệu trong quá khứ mà không tính đến những dữ kiện tương lai. Vì vậy, để đảm bảo dự đoán chính xác hơn, nên kết hợp với nhiều phương pháp dự đoán khác. [27]

2.16.6. Thực hiện làm mịn dữ liệu - làm mịn theo cấp số nhân (Exponential Moving Average)

Một kỹ thuật làm mịn phổ biến khác là làm mịn theo cấp số nhân. Kỹ thuật này cho trọng số lớn hơn cho các quan sát gần đây và ít hơn cho các quan sát cũ hơn. Trong khi làm mịn trung bình động áp dụng trọng số bằng nhau cho các quan sát quá khứ, làm mịn theo cấp số nhân áp dụng trọng số giảm dần theo cấp số nhân khi các quan sát trở nên cũ hơn. Một lợi thế chính của làm mịn theo cấp số nhân là khả năng bắt kịp các thay đổi đột ngột trong dữ liệu hiệu quả hơn.

Ngoài phân tích thăm dò chuỗi thời gian, cả hai kỹ thuật làm mịn trung bình động và làm mịn theo cấp số nhân cũng có thể được sử dụng làm cơ sở để dự báo các giá trị tương lai trong chuỗi thời gian.

Chúng ta sẽ khám phá kỹ thuật làm mịn theo cấp số nhân trong Python bằng cách sử dụng mô-đun ExponentialSmoothing trong thư viện statsmodels. [10]

2.16.7. Thực hiện kiểm tra tính dừng trên dữ liệu chuỗi thời gian

Tính dừng là một khái niệm quan trọng trong chuỗi thời gian. Dữ liệu có tính dừng có các đặc tính thống kê như trung bình, phương sai và hiệp phương sai không thay đổi theo thời gian. Dữ liệu dừng không chứa xu hướng và tính mùa vụ; thông thường, chuỗi thời gian với các đặc tính này được gọi là không dừng. Kiểm tra tính dừng quan trọng vì dữ liệu không dừng có thể khó mô hình hóa và dự đoán.

Để kiểm tra tính dừng, chúng ta có thể sử dụng kiểm định thống kê gọi là kiểm định Dickey-Fuller. Không đi sâu vào chi tiết kỹ thuật, kiểm định Dickey-Fuller hoạt động với các giả thuyết sau:

- **Null hypothesis:** Dữ liệu chuỗi thời gian là không dừng

- **Alternative hypothesis:** Dữ liệu chuỗi thời gian là dừng

Kiểm định này tạo ra một giá trị thống kê kiểm định và các giá trị ngưỡng ở các mức ý nghĩa 1%, 5%, và 10%. Chúng ta thường so sánh giá trị thống kê kiểm định với các giá trị ngưỡng để đưa ra kết luận về tính dừng. Kết quả có thể được diễn giải như sau:

- Nếu giá trị thống kê kiểm định $<$ giá trị ngưỡng, chúng ta bác bỏ giả thuyết Null hypothesis và kết luận rằng chuỗi thời gian là dừng.
- Nếu giá trị thống kê kiểm định $>$ giá trị ngưỡng, chúng ta không bác bỏ giả thuyết Null hypothesis và kết luận rằng chuỗi thời gian là không dừng.

Kiểm định Dickey-Fuller có nhiều biến thể, trong đó một biến thể phổ biến là kiểm định Dickey-Fuller mở rộng (ADF). Kiểm định ADF thực hiện kiểm định Dickey-Fuller nhưng loại bỏ tự tương quan (autocorrelation) từ chuỗi. Chúng ta sẽ khám phá kiểm định ADF trong Python, sử dụng mô-đun adfuller trong thư viện statsmodels. [10]

2.16.8. Tính sai phân cho dữ liệu chuỗi thời gian

Khi dữ liệu chuỗi thời gian không dừng, chúng ta có thể sử dụng một kỹ thuật gọi là tính sai phân (differencing) để biến nó thành dữ liệu dừng. Tính sai phân bao gồm việc trừ đi giá trị hiện tại từ giá trị trước đó để loại bỏ các xu hướng hoặc tính mùa vụ có trong chuỗi thời gian. Sai phân bậc nhất xảy ra khi chúng ta trừ mỗi giá trị từ giá trị liền trước đó một khoảng thời gian. Sai phân có thể được thực hiện nhiều lần, và điều này được gọi là sai phân bậc cao hơn. Điều này giúp loại bỏ các mức độ cao hơn của xu hướng hoặc tính mùa vụ. Tuy nhiên, nhược điểm của việc tính sai phân quá nhiều lần là chúng ta có thể mất đi những thông tin quan trọng từ dữ liệu chuỗi thời gian ban đầu. Chúng ta sẽ khám phá kỹ thuật phân biệt trong Python bằng cách sử dụng phương thức diff trong pandas. [10]

2.16.9. Thư viện hỗ trợ

Statsmodels là một thư viện Python mạnh mẽ dành cho phân tích thống kê, mô hình hóa và kiểm định giả thuyết. Nó cung cấp các công cụ cần thiết cho việc thực hiện các phân tích số liệu, đặc biệt hữu ích trong lĩnh vực kinh tế lượng, sinh học và nghiên cứu xã hội.

Các Tính Năng Chính

- **Hồi Quy Tuyến Tính và Phi Tuyến:** Cung cấp các mô hình hồi quy tuyến tính đơn giản và phức tạp. Hỗ trợ mô hình hồi quy phi tuyến, bao gồm mô hình polynomial, spline, và các mô hình phi tuyến khác.

- Phân Tích Chuỗi Thời Gian: Hỗ trợ các mô hình chuỗi thời gian như AR, MA, ARMA, ARIMA, và SARIMA. Công cụ mạnh mẽ cho dự báo và phân tích xu hướng theo thời gian. Tính năng kiểm tra mùa vụ, tính dừng, và tự tương quan.
- Kiểm Định Giả Thuyết: Cung cấp nhiều phương pháp kiểm định giả thuyết như t-test, F-test, chi-square test, và nhiều kiểm định phi tham số khác. Các phương pháp kiểm định đa biến, kiểm định đồng nhất và kiểm định tương quan.

2.17. Các độ đo đánh giá mô hình

2.17.1. Mean Absolute Error (MAE)

MAE (Mean Absolute Error) [28] [29] được dùng cho bài toán hồi quy. MAE tính toán độ lệch trung bình giá trị tuyệt đối của sai số giữa giá trị dự đoán và giá trị thực tế. Tuy nhiên nó không đánh giá được độ lớn của các sai số.

$$MAE = \frac{1}{n} \sum_{i=1}^n |x_i - y_i| \quad (17)$$

n: số lượng quan sát trong mẫu

x_i : giá trị thực tế của biến phụ thuộc của quan sát i

y_i : giá trị dự đoán của biến phụ thuộc của quan sát i

2.17.2. Mean Squared Error (MSE)

Mean Squared Error (MSE) [28] [29] có lẽ là số liệu phổ biến nhất được sử dụng cho các bài toán hồi quy. Về cơ bản, nó tính trung bình của bình phương sai số giữa các giá trị thực tế và các giá trị dự đoán trong một tập dữ liệu. MSE là thước đo chất lượng của một công cụ ước tính, nó luôn không âm và các giá trị càng gần 0 càng tốt.

$$MSE = \frac{1}{n} \sum_{i=1}^n (x_i - y_i)^2 \quad (18)$$

n: số lượng quan sát trong mẫu

x_i : giá trị thực tế của biến phụ thuộc của quan sát i

y_i : giá trị dự đoán của biến phụ thuộc của quan sát i

2.17.3. Root Mean Square Error (RMSE)

Root Mean Squared Error (RMSE) [28] [30] là căn bậc hai của MSE. RMSE đánh giá độ lệch trung bình giữa giá trị dự đoán và giá trị thực tế được dùng trong bài toán hồi

quy, tuy nhiên nó bị ảnh hưởng bởi các giá trị nhiễu hoặc outlier trong dữ liệu. Nếu dữ liệu có các giá trị nhiễu hoặc outlier, RMSE có thể bị giảm xuống đáng kể.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (x_i - y_i)^2} \quad (19)$$

n: số lượng quan sát trong mẫu

x_i : giá trị thực tế của biến phụ thuộc của quan sát i

y_i : giá trị dự đoán của biến phụ thuộc của quan sát i

2.17.4. R-squared (R^2)

R-squared (R^2) [31] là một chỉ số thống kê được sử dụng để đánh giá mức độ phù hợp của một mô hình hồi quy tuyến tính. Nó đo lường tỷ lệ biến thiên trong biến phụ thuộc (biến kết quả) được giải thích bởi biến độc lập (biến dự đoán) trong mô hình. R^2 có giá trị từ 0 đến 1, giá trị R^2 càng cao, mô hình càng phù hợp với dữ liệu và giải thích được nhiều biến thiên hơn.

$$R^2 = 1 - \frac{RSS}{TSS} \quad (20)$$

Trong đó:

- RSS (Residual Sum of Squares): Tổng bình phương của các sai số (phản dư) giữa giá trị dự đoán và giá trị thực tế được tính theo công thức 18.
- TSS (Total Sum of Squares): Tổng bình phương của sự khác biệt giữa giá trị thực tế và giá trị trung bình của chúng được tính theo công thức 19.

$$RSS = \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (21)$$

$$TSS = \sum_{i=1}^n (y_i - \bar{y}_i)^2 \quad (22)$$

Với:

- y_i là giá trị thực tế của biến phụ thuộc
- \hat{y}_i là giá trị dự đoán từ mô hình
- \bar{y}_i là giá trị trung bình của biến phụ thuộc
- n là số lượng quan sát

2.17.5. Adjusted R-Squared

Adjusted R-Squared [31] là một phiên bản được điều chỉnh của R-squared, được sử dụng để đánh giá mức độ phù hợp của mô hình hồi quy tuyến tính. R^2 thông thường chỉ đo lường tỷ lệ biến thiên của biến phụ thuộc được giải thích bởi mô hình hồi quy. Tuy nhiên, một hạn chế lớn của R^2 là nó luôn tăng hoặc không thay đổi khi thêm biến giải thích mới vào mô hình, ngay cả khi biến đó không thực sự có ý nghĩa. Để khắc phục vấn đề này, Adjusted R-squared được sử dụng.

$$Adjusted - R^2 = 1 - \frac{(1 - R^2)(n - 1)}{n - k - 1} \quad (23)$$

n là số lượng mẫu quan sát

k là số lượng biến độc lập trong mô hình

R^2 là giá trị R-squared

2.17.6. Accuracy

Accuracy là một thước đo để đánh giá các mô hình phân loại. Accuracy là tỷ lệ giữa số lần dự đoán đúng trên tổng số lần dự đoán. [32] [33]

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (24)$$

Trong đó:

- TP là True Positive (Tổng số trường hợp dự báo khớp Positive)
- TN là True Negative (Tổng số trường hợp dự báo khớp Negative)
- FP là False Positive (Tổng số trường hợp dự báo các quan sát thuộc nhãn Negative thành Positive)
- FN là False Negative (Tổng số trường hợp dự báo các quan sát thuộc nhãn Positive thành Negative)

2.17.7. Precision

Precision cho biết tỷ lệ dự đoán dương đúng so với dự đoán dương của mô hình. [32]

$$Precision = \frac{TP}{TP + FP} \quad (25)$$

Trong đó:

- TP là True Positive (Tổng số trường hợp dự báo khớp Positive)

- FP là False Positive (Tổng số trường hợp dự báo các quan sát thuộc nhãn Negative thành Positive)

2.17.8. Recall

Recall đo lường tỷ lệ dự báo chính xác các trường hợp positive trên toàn bộ các mẫu thuộc nhóm positive. [32] [33]

$$Recall = \frac{TP}{TP + FN} \quad (26)$$

Trong đó:

- TP là True Positive (Tổng số trường hợp dự báo khớp Positive)
- FN là False Negative (Tổng số trường hợp dự báo các quan sát thuộc nhãn Positive thành Negative)

2.17.9. F1-Score

F1-Score là trung bình điều hòa giữa precision và recall. Do đó nó đại diện hơn trong việc đánh giá độ chính xác trên đồng thời precision và recall. [32] [33]

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (27)$$

2.18. Tổng kết chương

Chương 2 đã giới thiệu và giải thích chi tiết các vấn đề thường gặp trong dữ liệu, độ đo đánh giá mô hình và các phương pháp xử lý chúng. Những kiến thức này là nền tảng lý thuyết quan trọng cho các bước thực hành cụ thể trong các chương sau, giúp đảm bảo chất lượng dữ liệu và hiệu suất của các mô hình học máy.

CHƯƠNG 3. PHÂN TÍCH BÀI TOÁN

Chương này mô tả các tập dữ liệu được sử dụng trong các bài toán thực nghiệm, bao gồm dự đoán giá nhà, dự đoán bệnh tiểu đường, phân loại thư rác, và dự đoán giá tiền ảo Bitcoin. Nó cũng trình bày các bước thực hiện cho từng bài toán, từ thu thập dữ liệu, xử lý dữ liệu, đến xây dựng và đánh giá mô hình.

3.1. Mô tả tập dữ liệu bài toán.

3.1.1. Bài toán 1: Xây dựng mô hình dự đoán giá nhà ở khu vực TP. Hồ Chí Minh.

Tập dữ liệu House_price_predict được thu thập từ GitHub [34]. Tập dữ liệu mô tả giá của các căn chung cư gồm 24949 dòng với 16 cột:

- DiaChi: Mô tả địa chỉ của chung cư.
- TinhTrangBDS: Mô tả tình trạng của chung cư (Đã bàn giao hoặc chưa bàn giao cho khách)
- DienTich: Mô tả diện tích của chung cư.
- Gia/m2: Mô tả giá của 1 mét vuông.
- Phongngu: Mô tả số lượng phòng ngủ.
- TenPhanKhu: Mô tả block của căn chung cư.
- SoTang: Mô tả chung cư nằm ở tầng bao nhiêu.
- PhongTam: Số lượng nhà vệ sinh.
- Loai: Mô tả loại hình chung cư (Chung cư, căn hộ dịch vụ, penthouse, duxlex, tập thể, officetel).
- GiayTo: tình trạng pháp lý của căn nhà (đã có sổ, đang chờ sổ, giấy tờ khác).
- MaCanHo: mã căn hộ.
- TinhTrangNoiThat: nội thất cao cấp, nội thất đầy đủ, nhà thô, hoàn thiện cơ bản.
- HuongCuaChinh: hướng cửa chính của chung cư.
- HuongBanCong: hướng ban công của chung cư.
- DacDiem: căn trong góc hoặc căn chính giữa.
- Gia: giá bán của căn hộ.

3.1.2. Bài toán 2: Xây dựng mô hình dự đoán bệnh nhân có mắc bệnh tiểu đường hay không.

Tập dữ liệu Diabetes prediction dataset được thu thập từ Kaggle [35]. Tập dữ liệu cung cấp các thông tin y tế liên quan đến bệnh tiểu đường của các bệnh nhân gồm 100000 hàng và 9 cột:

- gender (giới tính) : đề cập đến giới tính sinh học của bệnh nhân, có thể ảnh hưởng đến khả năng mắc bệnh tiểu đường của họ. Có ba danh mục trong đó: Male (nam), Female (nữ) và Other (khác).
- age (tuổi): là một yếu tố quan trọng vì tiểu đường thường được chẩn đoán nhiều hơn ở người lớn tuổi. Tuổi biến động từ 0-80 trong tập dữ liệu.
- hypertension (cao huyết áp): là một tình trạng y tế trong đó áp suất máu trong các động mạch tăng liên tục. Nó có giá trị 0 hoặc 1, trong đó 0 cho biết họ không có cao huyết áp và 1 cho biết họ có cao huyết áp.
- heart_disease (Bệnh tim): là một tình trạng y tế khác liên quan đến việc tăng nguy cơ mắc bệnh tiểu đường. Nó có giá trị 0 hoặc 1, trong đó 0 cho biết họ không có bệnh tim và 1 cho biết họ có bệnh tim.
- smoking_history (Lịch sử hút thuốc): cũng được coi là một yếu tố nguy cơ cho tiểu đường và có thể làm tăng các vấn đề liên quan đến tiểu đường. Trong tập dữ liệu, có 6 danh mục: not current(không hiện tại) ,former (trước đây),No Info (không có thông tin) ,current (hiện tại), never (không bao giờ) and ever (đã từng).
- BMI (Body Mass Index) là thước đo lượng mỡ cơ thể dựa trên cân nặng và chiều cao. Chỉ số BMI cao hơn có liên quan đến nguy cơ mắc bệnh tiểu đường cao hơn. Phạm vi BMI trong tập dữ liệu là từ 10,16 đến 71,55. BMI dưới 18,5 là thiếu cân, 18,5-24,9 là bình thường, 25-29,9 là thừa cân, và 30 trở lên là béo phì.
- HbA1c_level: HbA1c (Hemoglobin A1c) là thước đo mức đường huyết trung bình của một người trong vòng 2-3 tháng qua. Các mức cao hơn chỉ ra nguy cơ cao về tiểu đường. Đa số mức HbA1c trên 6,5% chỉ ra tiểu đường.
- blood_glucose_level: Mức đường huyết đề cập đến lượng glucose trong máu tại một thời điểm nhất định. Mức đường huyết cao là dấu hiệu chính của bệnh tiểu đường.
- diabetes (tiểu đường) là biến mục tiêu đang được dự đoán, với giá trị 1 cho biết có bệnh tiểu đường và 0 cho biết không có bệnh tiểu đường.

3.1.3. Bài toán 3: Phân loại thư rác (Spam Email)

Bộ sưu tập SMS Spam là một tập hợp các tin nhắn SMS được gán nhãn để nghiên cứu về spam SMS. Nó bao gồm một tập tin nhắn SMS bằng tiếng Anh với tổng cộng 5.574 tin nhắn, được gán nhãn là ham (hợp pháp) hoặc spam.

Các tệp chứa mỗi tin nhắn trên một dòng. Mỗi dòng bao gồm hai cột: **v1** chứa nhãn (ham hoặc spam) và **v2** chứa văn bản thô.

Tập dữ liệu này được thu thập từ các nguồn miễn phí hoặc miễn phí cho nghiên cứu trên Internet:

- Một bộ sưu tập 425 tin nhắn spam SMS được trích xuất thủ công từ trang web Grumbletext. Đây là một diễn đàn ở Vương quốc Anh, nơi người dùng điện thoại di động công khai khiếu nại về tin nhắn spam SMS.
- Một tập hợp ngẫu nhiên 3.375 tin nhắn ham SMS từ NUS SMS Corpus (NSC), một tập dữ liệu chứa khoảng 10.000 tin nhắn hợp pháp thu thập tại Khoa Khoa học Máy tính, Đại học Quốc gia Singapore.
- Một danh sách 450 tin nhắn ham SMS được thu thập từ luận án Tiến sĩ của Caroline Tag.
- SMS Spam Corpus v.0.1 Big với 1.002 tin nhắn ham và 322 tin nhắn spam.

3.1.4. Bài toán 4: Phân tích và dự đoán giá tiền ảo Bitcoin

- Tên bộ dữ liệu: Every Cryptocurrency Daily Market Price
- Nguồn dữ liệu: Kaggle
- Cấu trúc gồm: 13 cột, 679185 dòng dữ liệu, chi tiết (Bảng 3.1):

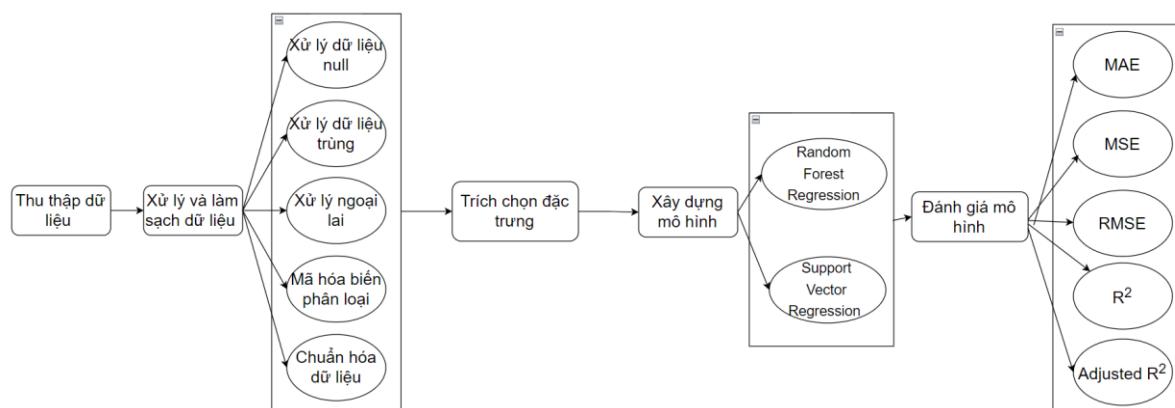
Bảng 3.1. Bảng mô tả dữ liệu bài toán 4

Thứ tự	Thuộc tính	Ý nghĩa
0	slug	Tên viết tắt của các loại tiền ảo
1	symbol	Ký hiệu của các loại tiền ảo
2	name	Tên đầy đủ của các loại tiền ảo
3	date	Ngày ghi nhận các dữ liệu thống kê về giá và khối lượng giao dịch
4	ranknow	Thứ hạng hiện tại của loại tiền ảo trong thị trường so với các loại tiền ảo khác
5	open	Giá mở của tiền ảo trong ngày
6	high	Giá cao nhất của tiền ảo trong ngày
7	low	Giá thấp nhất của tiền ảo trong ngày
8	close	Giá đóng của tiền ảo trong ngày
9	volume	Khối lượng giao dịch của tiền ảo trong ngày

10	market	Giá trị vốn hóa thị trường của tiền ảo trong ngày
11	close_ratio	Tỷ lệ giá đóng là tỷ lệ phần trăm của sự chênh lệch giữa giá đóng và giá thấp nhất trong ngày so với sự chênh lệch giữa giá cao nhất và giá thấp nhất trong cùng ngày.
12	spread	Sự chênh lệch giữa giá cao nhất và giá thấp nhất trong ngày

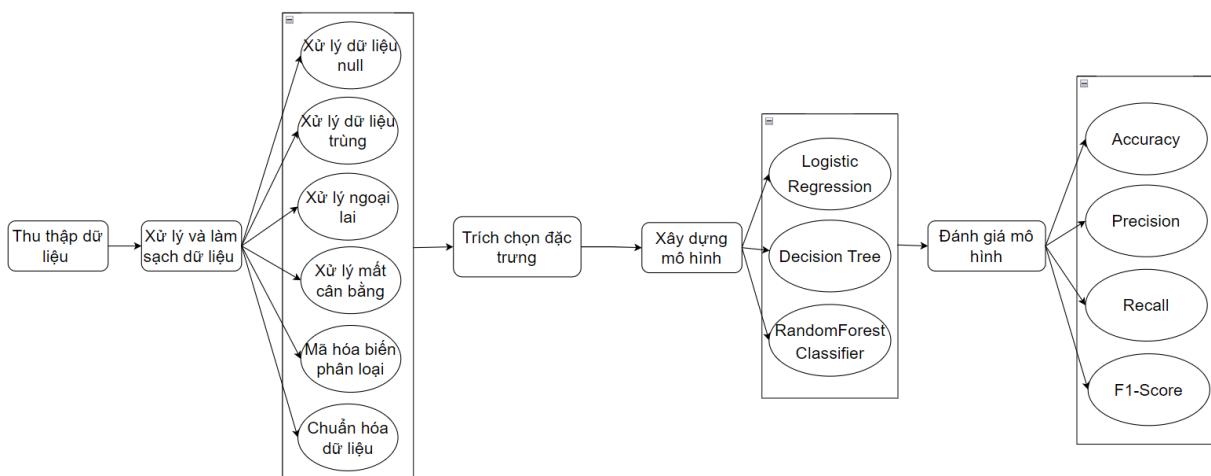
3.2. Quy trình thực hiện bài toán.

3.2.1. Bài toán 1: Xây dựng mô hình dự đoán giá nhà ở khu vực TP. Hồ Chí Minh.



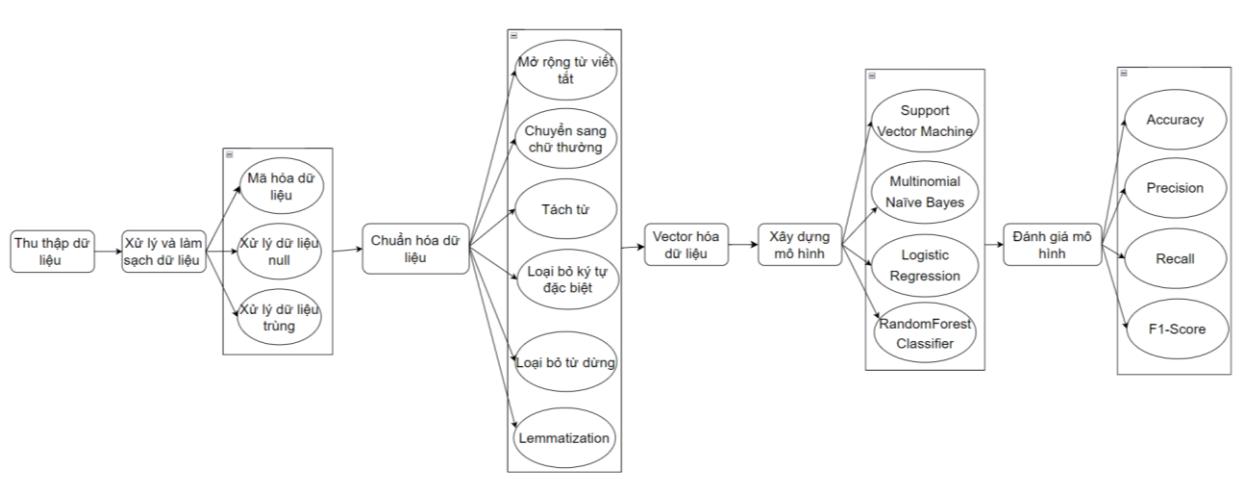
Hình 3.1. Sơ đồ quy trình thực hiện bài toán 1

3.2.2. Bài toán 2: Xây dựng mô hình dự đoán bệnh nhân có mắc bệnh tiểu đường hay không.



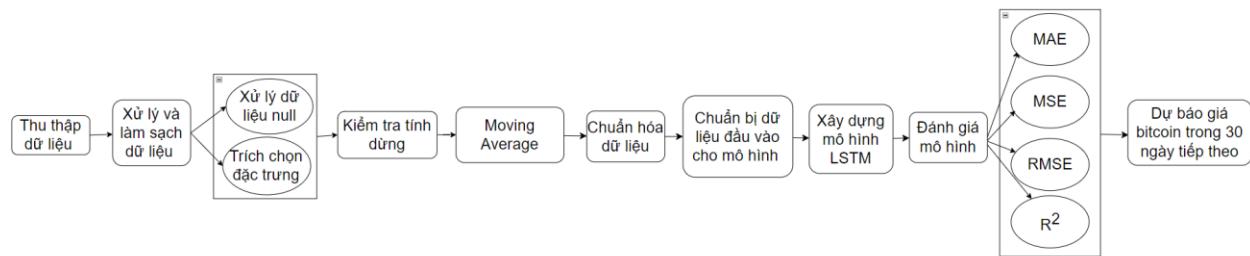
Hình 3.2. Sơ đồ quy trình thực hiện bài toán 2

3.2.3. Bài toán 3: Phân loại thư rác (Spam Email)



Hình 3.3. Sơ đồ quy trình thực hiện bài toán 3

3.2.4. Bài toán 4: Phân tích và dự đoán giá tiền ảo Bitcoin



Hình 3.4. Sơ đồ quy trình thực hiện bài toán 4

3.3. Tổng kết chương

Chương 3 đã cụ thể hóa việc áp dụng các kỹ thuật phân tích và tiền xử lý dữ liệu vào các bài toán thực tiễn. Bằng cách trình bày chi tiết từng bước thực hiện cho mỗi bài toán, chương này giúp ta hiểu rõ hơn về quy trình làm việc và cách áp dụng lý thuyết vào thực hành.

CHƯƠNG 4. KẾT QUẢ THỰC NGHIỆM

Chương này trình bày kết quả thực nghiệm cho các bài toán đã nêu ở Chương 4. Nó mô tả yêu cầu phần cứng, phần mềm, và quy trình thực hiện trên Python cho từng bài toán, từ việc xử lý dữ liệu đến xây dựng mô hình và đánh giá kết quả.

4.1. Yêu cầu phần cứng, phần mềm.

- **Phần cứng:**

Cần một máy tính có cấu hình đủ mạnh để huấn luyện các mô hình hồi quy, đặc biệt là khi áp dụng trên bộ dữ liệu có kích thước lớn. CPU: Intel Core i7 hoặc AMD Ryzen 7 trở lên. GPU: NVIDIA GeForce RTX 2070 hoặc AMD Radeon RX 5700 XT trở lên. RAM đủ lớn để chứa mô hình và dữ liệu (tối thiểu 8GB trở lên).

- **Phần mềm:**

- Môi trường lập trình: Sử dụng ngôn ngữ Python để lập trình. Môi trường cài đặt là Kaggle/ Google Colab.

- Thư viện và Gói Phần Mềm: Cài đặt các thư viện và gói phần mềm cần thiết như NumPy, Pandas, Scikit-learn,... để xử lý và tiền xử lý dữ liệu. Cài đặt thư viện đồ họa như Matplotlib, Seaborn để trực quan dữ liệu.

4.2. Quy trình thực hiện trên Python.

4.2.1. Bài toán 1: Xây dựng mô hình dự đoán giá nhà ở khu vực TP. Hồ Chí Minh.

Bước 1: Thu thập dữ liệu.

Bước 2: Xử lý và làm sạch dữ liệu.

Với dữ liệu đã được thu thập ta tiến hành xử lý và làm sạch dữ liệu. Đọc và hiển thị dữ liệu (Hình 4.1):

[2]:	DiaChi	TinhTrangBDS	DienTich	Gia/m ²	Phongngu	TenPhanKhu	SoTang	PhongTam	Loai	GiayTo	MaCanHo	TinhTrangNoiThat	HuongCuaChinh	HuongBanCong	DacDiem	Gia
0	Đường Nguyễn Văn Quý, Phường Phú Thuận, Quận 7...	Đã bàn giao	62 m ²	32,26 triệu/m ²	2 phòng		NaN	NaN	2 phòng	Chung cư	Đã có sổ	NaN	NaN	NaN	NaN	2 tỷ- 62 m ²
1	Đường Nguyễn Văn Linh, Phường Tân Thuận Tây, Q...	Đã bàn giao	95 m ²	55,79 triệu/m ²	3 phòng		NaN	NaN	2 phòng	Chung cư	Đang chờ sổ	NaN	Nội thất cao cấp	NaN	NaN	Căn góc 5,3 tỷ- 95 m ²
2	Đường Võ Văn Kiệt, Phường An Lạc, Quận Bình Tân...	Chưa bàn giao	75 m ²	34,4 triệu/m ²	2 phòng	2	5.0	2 phòng	Chung cư	Giấy tờ khác	17	NaN	Đông Nam	Đông Nam	NaN	2,58 tỷ- 75 m ²
3	108, Đường Hồng Hà, Phường 6, Quận Tân Bình, T...	Đã bàn giao	70 m ²	57,14 triệu/m ²	1 phòng	A	7.0	1 phòng	Chung cư	Đang chờ sổ	BPA - 0712	Nội thất cao cấp	Đông Nam	Tây Bắc	NaN	4 tỷ- 70 m ²
4	Đường Hậu Giang, Phường 11, Quận 6, TP Hồ Chí ...	Đã bàn giao	83 m ²	35,54 triệu/m ²	2 phòng	NaN	NaN	2 phòng	Chung cư	Đã có sổ	NaN	Nội thất cao cấp	Tây Bắc	NaN	NaN	2,95 tỷ- 83 m ²

Hình 4.1. Hiển thị 5 dòng dữ liệu đầu tiên

Chương 4. Kết quả thực nghiệm

Thông tin (`data.info()`) về số lượng hàng, cột, tên cột, số lượng giá trị không null, kiểu dữ liệu được thể hiện trong Bảng 4.1. Thống kê tóm tắt của các cột dữ liệu số (`data.describe()`) được thể hiện trong Bảng 4.2.

Bảng 4.1. Thông tin về tập dữ liệu.

RangeIndex: 24949 entries, 0 to 24948			
Data columns (total 16 columns):			
	Column	Non-Null Count	Dtype
0	DiaChi	24624 non-null	object
1	TinhTrangBDS	24924 non-null	object
2	DienTich	24917 non-null	float64
3	Gia/m2	24916 non-null	float64
4	Phongngu	24926 non-null	float64
5	TenPhanKhu	7036 non-null	object
6	SoTang	6726 non-null	float64
7	PhongTam	24388 non-null	float64
8	Loai	24926 non-null	object
9	GiayTo	18852 non-null	object
10	MaCanHo	3358 non-null	object
11	TinhTrangNoiThat	12790 non-null	object
12	HuongCuaChinh	9370 non-null	object
13	HuongBanCong	8670 non-null	object
14	DacDiem	5601 non-null	object
15	Gia	24949 non-null	float64

Bảng 4.2. Tóm tắt thống kê cho tập dữ liệu.

	DienTich	Gia/m2	Phongngu	SoTang	PhongTam	Gia
count	24917.00000	2.491600e+0	24926.00000	6726.00000	2488.00000	24949.00000
mean	193.907179	6.803146e+0	2.045134	11.482307	1.750656	2.865201
std	7465.948823	5.424467e+0	0.730980	18.142728	0.596897	7.827805
min	1.000000	0.000000e+0	1.000000	1.000000	1.000000	0.000000
25%	56.000000	1.412000e+0	2.000000	5.000000	1.000000	1.600000
50%	68.000000	2.973500e+0	2.000000	9.000000	2.000000	2.250000
75%	80.000000	4.151000e+0	2.000000	16.000000	2.000000	3.300000
max	780000.00000	5.428571e+1	10.000000	789.00000	6.000000	980.00000

Chương 4. Kết quả thực nghiệm

Do các cột 'DienTich' , 'Gia/m2', 'Phongngu' và 'PhongTam' thuộc dạng (string) bao gồm cả số và chữ nên cần chuyển chúng sang dạng số (float). Tiếp theo, cần xử lý trích lọc ra tên Quận và Huyện ở cột ‘DiaChi’ vì không cần địa chỉ chi tiết của các căn hộ. Tương tự, cột ‘Gia’ vừa chứa số và chữ nên cũng cần Convert lại cột ‘Gia’ theo đơn vị tỷ đồng. Kết quả biến đổi dữ liệu được thể hiện trong Hình 4.2.

	DiaChi	TinhTrangBDS	DienTich	Gia/m2	Phongngu	TenPhanKhu	SoTang	PhongTam	Loai	GiayTo	MaCanHo	TinhTrangNoiThat	HuongCuaChinh	HuongBanCong	DacDiem	Gia
0	Quận 7	Đã bàn giao	62.0	3226.0	2.0	NaN	NaN	2.0	Chung cư	Đã có sổ	NaN	NaN	NaN	NaN	NaN	2.00
1	Quận 7	Đã bàn giao	95.0	5579.0	3.0	NaN	NaN	2.0	Chung cư	Đang chờ sổ	NaN	Nội thất cao cấp	NaN	NaN	Căn góc	5.30
2	Quận Bình Tân	Chưa bàn giao	75.0	344.0	2.0	2	5.0	2.0	Chung cư	Giấy tờ khác	17	NaN	Đông Nam	Đông Nam	NaN	2.58
3	Quận Tân Bình	Đã bàn giao	70.0	5714.0	1.0	A	7.0	1.0	Chung cư	Đang chờ sổ	BPA - 0712	Nội thất cao cấp	Đông Nam	Tây Bắc	NaN	4.00
4	Quận 6	Đã bàn giao	83.0	3554.0	2.0	NaN	NaN	2.0	Chung cư	Đã có sổ	NaN	Nội thất cao cấp	Tây Bắc	NaN	NaN	2.95

Hình 4.2. Dữ liệu mẫu sau khi biến đổi dữ liệu của 5 dòng đầu tiên.

- Xử lý giá trị null

Thông qua phân tích, tập dữ liệu có tồn tại nhiều giá trị null (Bảng 4.3). Do vậy, cần xử lý những giá trị null này.

Bảng 4.3. Tỷ lệ phần trăm các giá trị null.

DiaChi	1.302657
TinhTrangBDS	0.100204
DienTich	0.128262
Gia/m2	0.132270
Phongngu	0.092188
TenPhanKhu	71.798469
SoTang	73.041004
PhongTam	2.248587
Loai	0.092188
GiayTo	24.437853
MaCanHo	86.540543
TinhTrangNoiThat	48.735420
HuongCuaChinh	62.443385
HuongBanCong	65.249108
DacDiem	77.550202
Gia	0.000000
dtype:	float64

Các cột TenPhanKhu, SoTang, MaCanHo, HuongCuaChinh, HuongBanCong, DacDiem cần phải xóa vì các cột này có các giá trị missing rất lớn (hơn 50%). Sau đó, sử dụng ‘dropna’ để loại bỏ các giá trị null cho các cột DiaChi, TinhTrangBDS, DienTich, Gia/m2, Phongngu, PhongTam, Loai vì các cột này có số lượng các giá trị bị thiếu nhỏ hơn 5%. Đối với cột ‘GiayTo’ và ‘TinhTrangNoiThat’ có phần trăm giá trị null lần lượt là 24.43% và 48.73% thì ta tiến hành xử lý bằng cách thay thế giá trị null bằng giá trị mode

Chương 4. Kết quả thực nghiệm

vì kiểu dữ liệu của 2 cột này là object. Bảng 4.4 thể hiện kết quả sau khi xử lý những giá trị null.

Bảng 4.4. Kết quả sau khi xử lý các giá trị null.

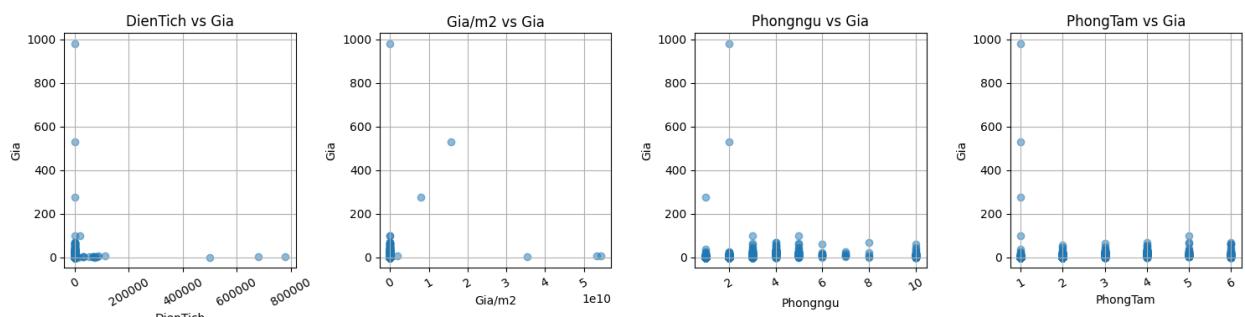
DiaChi	0
TinhTrangBDS	0
DienTich	0
Gia/m2	0
Phongngu	0
PhongTam	0
Loai	0
GiayTo	0
TinhTrangNoiThat	0
Gia	0
dtype:	int64

Ngoài ra, dữ liệu trùng cũng cần được loại bỏ. Hình 4.3 trình bày thông tin những dữ liệu bị trùng nhau (7,358 dòng bị trùng).

Number of duplicate rows: 7358											
	DiaChi	TinhTrangBDS	DienTich	Gia/m2	Phongngu	PhongTam	Loai	GiayTo	TinhTrangNoiThat	Gia	
25	Quận 6	Đã bàn giao	83.0	3554.0	2.0	2.0	Chung cư	Đã có sổ	Nội thất cao cấp	2.95	
29	Quận 7	Đã bàn giao	95.0	5579.0	3.0	2.0	Chung cư	Đang chờ sổ	Nội thất cao cấp	5.30	
51	Quận 6	Đã bàn giao	83.0	3554.0	2.0	2.0	Chung cư	Đã có sổ	Nội thất cao cấp	2.95	
53	Quận 7	Đã bàn giao	62.0	3226.0	2.0	2.0	Chung cư	Đã có sổ	Hoàn thiện cơ bản	2.00	
75	Quận 8	Đã bàn giao	65.0	2769.0	2.0	1.0	Chung cư	Đã có sổ	Hoàn thiện cơ bản	1.80	

Hình 4.3: Một phần các giá trị trùng trong tập dữ liệu.

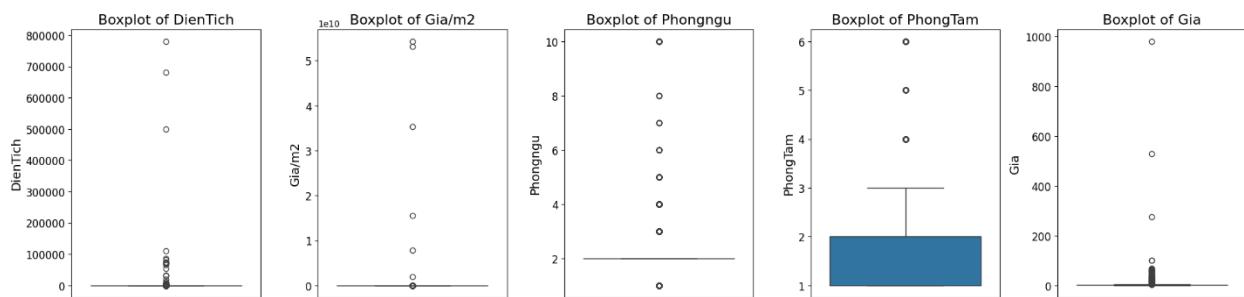
Sau đó chúng tôi phân tích mối tương quan giữa các biến dạng số với giá trị “Gia” thông qua biểu đồ Scatter (Hình 4.4).



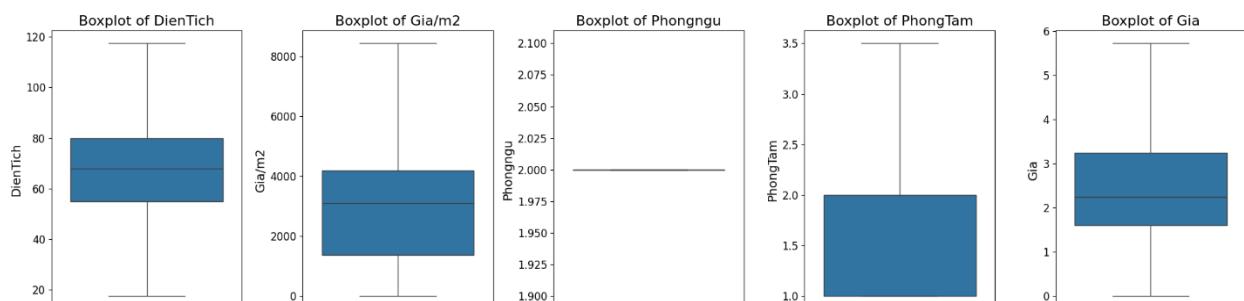
Hình 4.4. Biểu đồ Scatter trực quan mối quan hệ giữa các biến số với giá trị “Gia”

- Các điểm dữ liệu tập trung nhiều ở các giá trị thấp của DienTich và Giá, cho thấy hầu hết các bất động sản có diện tích nhỏ thì giá thấp. Có một vài điểm ngoại lệ với giá trị rất cao.
- Hầu hết các điểm dữ liệu tập trung ở giá trị thấp của Gia/m² và Giá, cho thấy các bất động sản có giá cao trên mỗi mét vuông ít phổ biến hơn.
- Các điểm dữ liệu tập trung đa số ở các bất động sản có 2-4 phòng ngủ, và giá cả tương đối thấp. Có một số điểm ngoại lệ với giá cao hơn.
- Hầu hết các điểm dữ liệu tập trung ở các bất động sản có 1-3 phòng tắm, và giá cả tương đối thấp. Có một vài điểm ngoại lệ với nhiều phòng tắm hơn và giá cao hơn.

Xử lý ngoại lệ



Hình 4.5. Biểu đồ hộp trước khi xử lý ngoại lệ



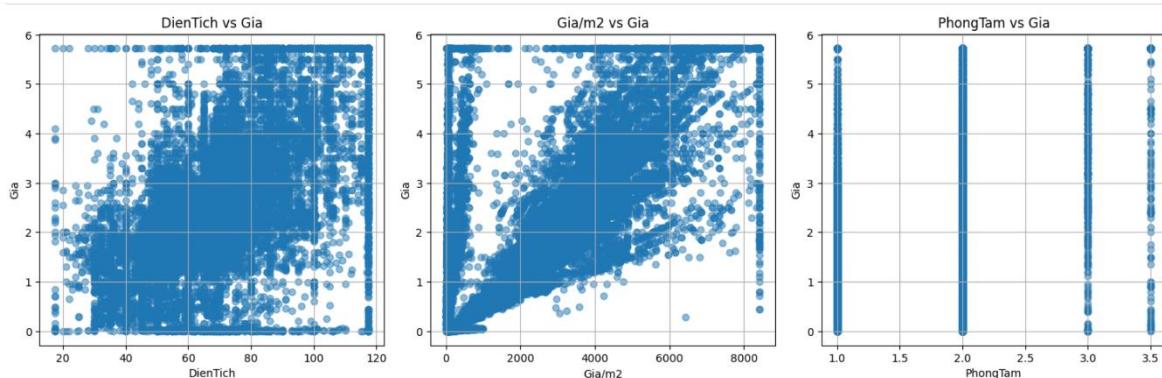
Hình 4.6. Biểu đồ hộp sau khi xử lý ngoại lệ

Sử dụng biểu đồ boxplot (Hình 4.5) cho các cột 'DienTich', 'Gia/m²', 'Phongngu', 'PhongTam', 'Gia' để xác định outliers và tiến hành xử lý trên những ngoại lệ này. Biểu đồ trong Hình 4.5 cho thấy sự hiện diện của nhiều giá trị ngoại lệ, đặc biệt là ở các biến DienTich, Gia/m² và Gia. Phần hộp chính của tất cả các biến số đều rất nhỏ, chứng tỏ sự tập trung cao độ của dữ liệu ở phạm vi thấp của các biến số. Xử lý các giá trị ngoại lệ bằng phương pháp IQR. Thay thế các giá trị lớn hơn upper_limit bằng upper_limit và các giá trị nhỏ hơn lower_limit bằng lower_limit. Kết quả xử lý ngoại lệ được thể hiện trong Hình 4.6. Sau khi xử lý outlier, biến "Phongngu" phân phối đồng nhất cho thấy sự thiếu đa dạng

Chương 4. Kết quả thực nghiệm

về số phòng ngủ trong tập dữ liệu, điều này có thể ảnh hưởng đến khả năng phân tích và dự báo vì vậy biến này cần loại bỏ.

Hình 4.7 và Hình 4.8 thể hiện biểu đồ Scatter và biểu đồ Heatmap trực quan mối quan hệ giữa các biến dạng số với giá trị “Gia”.



Hình 4.7. Biểu đồ Scatter trực quan mối quan hệ giữa các biến số với cột giá “Gia”

Biểu đồ Hình 4.7 cho thấy sự phân tán dữ liệu rộng, đặc biệt ở các biến DienTich và Gia/m2, cho thấy rằng các yếu tố này có thể có ảnh hưởng đáng kể đến giá.

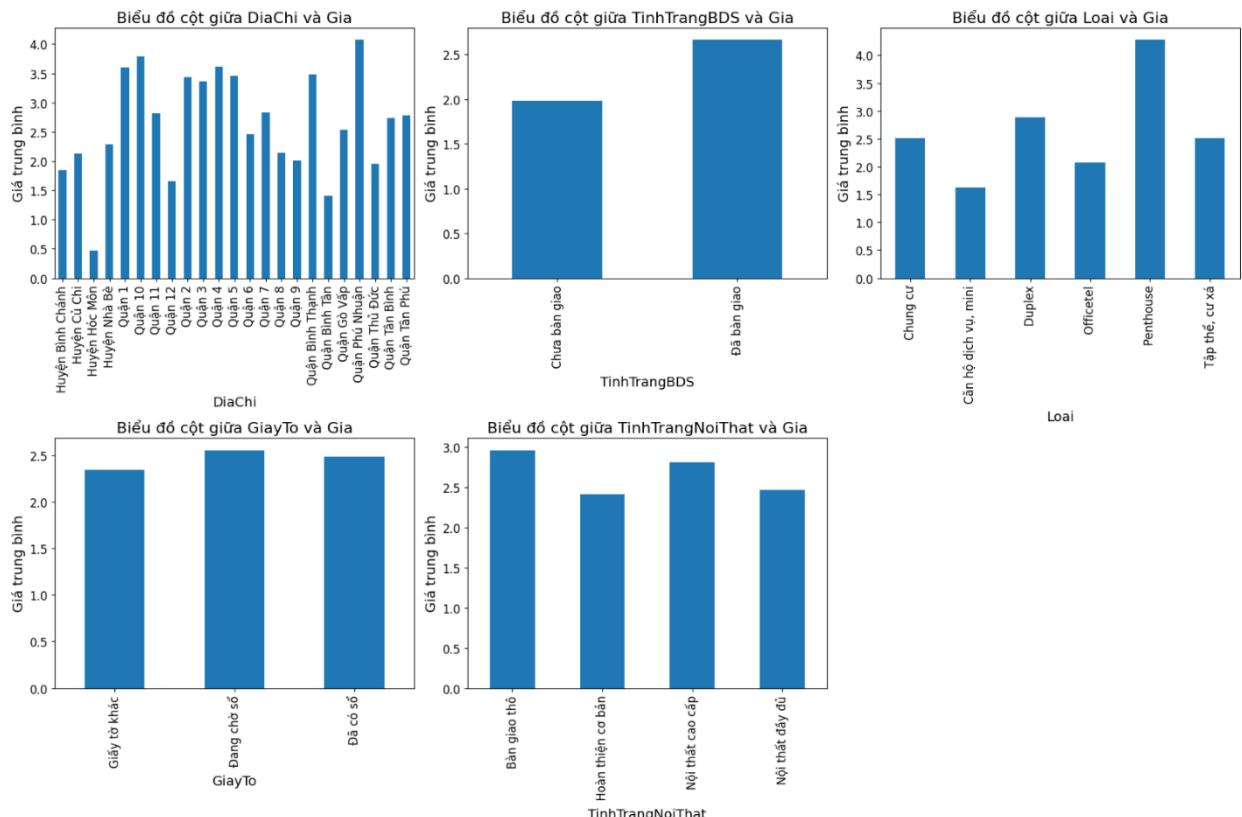


Hình 4.8. Biểu đồ heatmap thể hiện sự tương quan

Biểu đồ Hình 4.8 cho thấy DienTich và Gia/m2 thể hiện mối quan hệ khá mạnh với Gia, trong khi số lượng PhongTam lại có mối liên hệ yếu hơn. Mối tương quan giữa DienTich và Gia, cũng như giữa Gia/m2 và Gia, mang tính tuyến tính tương đối, nhưng chưa thực sự rõ nét. Ngược lại, mối quan hệ giữa số PhongTam và Gia thể hiện tính tuyến tính có phần hạn chế hơn.

- **Trực quan hóa dữ liệu**

Chương 4. Kết quả thực nghiệm



Hình 4.9: Biểu đồ cột giữa các cột dữ liệu phân loại và cột Gia.

Hình 4.9 trực quan hóa mối quan hệ giữa các cột dữ liệu phân loại và giá trị trung bình của giá nhà để xác định mức độ ảnh hưởng của các biến phân loại này. Biểu đồ cột giữa DiaChi và Gia thể hiện các quận trung tâm thường có giá trung bình cao hơn các huyện ngoại thành. Đối với Biểu đồ cột giữa TinhTrangBDS và Gia thể hiện tình trạng của bất động sản cũng ảnh hưởng đến giá. Loại hình bất động sản cũng ảnh hưởng lớn đến giá (Biểu đồ cột giữa Loai và Gia), với penthouse và duplex có giá cao nhất. Riêng Biểu đồ cột giữa GiayTo và Gia, loại giấy tờ không ảnh hưởng nhiều đến giá trung bình. Cuối cùng, là Biểu đồ cột giữa TinhTrangNoiThat và Gia thể hiện tình trạng nội thất có ảnh hưởng đáng kể đến giá trị của bất động sản. Các căn hộ "Bàn giao thô" và "Nội thất cao cấp" có giá trung bình cao hơn so với các căn hộ "Hoàn thiện cơ bản" và "Nội thất đầy đủ".

Vì các mô hình thống kê và học máy yêu cầu dữ liệu đầu vào dưới dạng số học để có thể thực hiện các tính toán và phân tích vì thế ta cần phải mã hóa các biến phân loại.

- Mã hóa dữ liệu

Dùng One-hot Encoder mã hóa biến 'DiaChi', 'TinhTrangBDS', 'Loai', 'GiayTo', 'TinhTrangNoiThat' thành số vì dữ liệu giá trị phân loại không mang ý nghĩa thứ tự. Kết quả: (Bảng 4.5).

Bảng 4.5: Kết quả sau khi mã hóa.

RangeIndex: 16728 entries, 0 to 16727			
Data columns (total 41 columns):			
	Column	Non-Null Count	Dtype
0	DienTich	16728 non-null	float64
1	Gia/m2	16728 non-null	float64
2	PhongTam	16728 non-null	float64
3	DiaChi_Huyện Bình Chánh	16728 non-null	float64
4	DiaChi_Huyện Củ Chi	16728 non-null	float64
5	DiaChi_Huyện Hóc Môn	16728 non-null	float64
6	DiaChi_Huyện Nhà Bè	16728 non-null	float64
7	DiaChi_Quận 1	16728 non-null	float64
8	DiaChi_Quận 10	16728 non-null	float64
9	DiaChi_Quận 11	16728 non-null	float64
10	DiaChi_Quận 12	16728 non-null	float64
11	DiaChi_Quận 2	16728 non-null	float64
12	DiaChi_Quận 3	16728 non-null	float64
13	DiaChi_Quận 4	16728 non-null	float64
14	DiaChi_Quận 5	16728 non-null	float64
15	DiaChi_Quận 6	16728 non-null	float64
16	DiaChi_Quận 7	16728 non-null	float64
17	DiaChi_Quận 8	16728 non-null	float64
18	DiaChi_Quận 9	16728 non-null	float64
19	DiaChi_Quận Bình Thạnh	16728 non-null	float64
20	DiaChi_Quận Bình Tân	16728 non-null	float64
21	DiaChi_Quận Gò Vấp	16728 non-null	float64
22	DiaChi_Quận Phú Nhuận	16728 non-null	float64
23	DiaChi_Quận Thủ Đức	16728 non-null	float64
24	DiaChi_Quận Tân Bình	16728 non-null	float64
25	DiaChi_Quận Tân Phú	16728 non-null	float64
26	TinhTrangBDS_Chưa bàn giao	16728 non-null	float64
27	TinhTrangBDS_Dã bàn giao	16728 non-null	float64
28	Loai_Chung cư	16728 non-null	float64
29	Loai_Căn hộ dịch vụ, mini	16728 non-null	float64
30	Loai_Duplex	16728 non-null	float64
31	Loai_Officetel	16728 non-null	float64
32	Loai_Penthouse	16728 non-null	float64
33	Loai_Tập thể, cư xá	16728 non-null	float64
34	GiayTo_Giấy tờ khác	16728 non-null	float64
35	GiayTo Đang chờ sổ	16728 non-null	float64
36	GiayTo Đã có sổ	16728 non-null	float64
37	TinhTrangNoiThat_Bàn giao thô	16728 non-null	float64
38	TinhTrangNoiThat_Hoàn thiện cơ bản	16728 non-null	float64
39	TinhTrangNoiThat_Nội thất cao cấp	16728 non-null	float64

Chương 4. Kết quả thực nghiệm

40	TinhTrangNoiThat_ Nội thất đầy đủ	16728 non-null	float64
dtypes:	float64(41)		

Sử dụng công cụ StandardScaler được dùng để chuẩn hóa dữ liệu nhằm đưa giá trị các cột dữ liệu số về một thang đo với giá trị trung bình là 0 và độ lệch chuẩn là 1. Bảng 4.6 thể hiện kết quả chuẩn hóa cho 5 mẫu dữ liệu đầu tiên.

Bảng 4.6: Sau khi Feature Scaling.

	DienTich	Gia/m2	PhongTam
0	-0.337101	0.107491	0.497391
1	1.207092	1.255721	0.497391
2	0.271218	-1.298883	0.497391
3	0.037249	1.321599	-1.278945
4	0.645568	0.267550	0.497391

5 rows \times 41 columns

Bước 3: Trích chọn đặc trưng và giảm chiều dữ liệu.

- Feature Selection.

Sử dụng phương pháp wrapper với kỹ thuật Recursive feature elimination (RFE) để trích chọn đặc trưng. Tạo mô hình LinearRegression để làm mô hình hồi quy. Tạo đối tượng RFECV với estimator=model: Sử dụng mô hình hồi quy tuyến tính để đánh giá các tập hợp đặc trưng, step=1: mỗi lần loại bỏ một đặc trưng kém nhất, cv=KFold(5): sử dụng K-Fold Cross-Validation với 5 fold để đánh giá mô hình, scoring='r2': sử dụng R-squared làm thước đo hiệu suất mô hình. Sau khi thực hiện Feature Selection ta giảm được số chiều dữ liệu từ 41 xuống còn 35 (Hình 4.10).

```
Optimal number of features: 35
Selected Features: Index(['DienTich', 'Gia/m2', 'PhongTam', 'DiaChi_Huyện Bình Chánh',
   'DiaChi_Huyện Củ Chi', 'DiaChi_Huyện Hóc Môn', 'DiaChi_Huyện Nhà Bè',
   'DiaChi_Quận 1', 'DiaChi_Quận 10', 'DiaChi_Quận 12', 'DiaChi_Quận 2',
   'DiaChi_Quận 3', 'DiaChi_Quận 4', 'DiaChi_Quận 5', 'DiaChi_Quận 6',
   'DiaChi_Quận 8', 'DiaChi_Quận 9', 'DiaChi_Quận Bình Thạnh',
   'DiaChi_Quận Bình Tân', 'DiaChi_Quận Gò Vấp', 'DiaChi_Quận Phú Nhuận',
   'DiaChi_Quận Thủ Đức', 'DiaChi_Quận Tân Bình',
   'TinhTrangBDS_Chưa bàn giao', 'TinhTrangBDS_Đã bàn giao',
   'Loai_Chung cư', 'Loai_Căn hộ dịch vụ, mini', 'Loai_Officetel',
   'Loai_Penthouse', 'Loai_Tập thể, cư xá', 'GiayTo_Đã có sổ',
   'TinhTrangNoiThat_Bàn giao thô', 'TinhTrangNoiThat_Hoàn thiện cơ bản',
   'TinhTrangNoiThat_Nội thất cao cấp',
   'TinhTrangNoiThat_Nội thất đầy đủ'],
  dtype='object')
Feature Ranking: [1 1 1 1 1 1 1 1 1 6 1 1 1 1 1 1 1 4 1 1 1 1 1 1 1 1 1 1 5 1 1 1 1 1 2 1 1 1 3 7 1
  1 1 1 1]
```

Hình 4.10. Kết quả sau khi thực hiện Feature Selection

- Giảm chiều dữ liệu bằng PCA

Với tập dữ liệu này chúng ta không thực hiện giảm số chiều bằng PCA vì kết quả dự đoán của mô hình sau khi thực hiện giảm số chiều bằng PCA kém hơn so với khi sử dụng dữ liệu ban đầu. Lý do dẫn đến khi dùng PCA kết quả dự đoán có kém hơn là do PCA không phù hợp với dữ liệu phi tuyến tính đã được nhận xét Hình 4.7 và Hình 4.8. Kết quả so sánh hiệu quả của các mô hình được biểu hiện ở *Bảng 4.7*.

Bảng 4.7: Kết quả của mô hình khi sử dụng PCA và không sử dụng PCA

Độ đo	SVM		Random Forest	
	Không dùng PCA	Dùng PCA	Không dùng PCA	Dùng PCA
MAE	0,29	0,31	0,16	0,27
MSE	0,43	0,46	0,28	0,41
RMSE	0,65	0,68	0,53	0,64
R ²	0,79	0,77	0,86	0,79
Adjusted R ²	0,79	0,77	0,86	0,79
Thời gian thực thi	97,07s	87,40s	212,14s	798,74s

Bước 4: Xây dựng mô hình.

Trong bài toán này, chúng tôi sử dụng phương pháp RandomizedSearchCV để xác định các siêu tham số phù hợp cho hai mô hình SVR và RFR [36]. RandomizedSearchCV thực hiện tìm kiếm ngẫu nhiên các tổ hợp siêu tham số từ các lưới tham số đã định nghĩa và đánh giá mô hình trên tập dữ liệu sử dụng phương pháp cross-validation. RandomizedSearchCV được chọn thay vì GridSearchCV, do nó hiệu quả hơn về mặt thời gian khi làm việc với không gian tham số lớn, đồng thời vẫn cung cấp kết quả tối ưu gần như tương đương.

Cả hai quá trình tìm kiếm siêu tham số tối ưu và đánh giá mô hình đều sử dụng K-Fold Cross-Validation với k = 5. Kỹ thuật này chia dữ liệu thành 5 phần, mỗi phần lần lượt được sử dụng làm tập kiểm tra trong khi các phần còn lại được sử dụng làm tập huấn luyện. Việc sử dụng kỹ thuật này giúp giảm thiểu hiện tượng quá khớp và đảm bảo mô hình có tính tổng quát tốt hơn. Tiêu chí đánh giá (scoring) là 'neg_root_mean_squared_error', nhằm tìm kiếm mô hình có khả năng dự đoán chính xác nhất bằng cách tối thiểu hóa sai số RMSE.

Đối với mô hình SVR, chúng tôi thiết lập không gian tìm kiếm với các giá trị C (0,1, 1, 10, 100) và các loại kernel ('rbf', 'linear', 'poly', 'sigmoid'), thực hiện 16 lần lặp ngẫu nhiên. Kết quả cho thấy siêu tham số tốt nhất cho SVR là kernel 'rbf' và C = 1. Hàm kernel 'rbf' giúp biến đổi không gian đặc trưng để mô hình có thể tìm ra các quan hệ phi tuyến giữa các đặc trưng, trong khi tham số C điều chỉnh mức độ phạt áp dụng cho các lỗi trong

dữ liệu huấn luyện, giúp đạt được sự cân bằng giữa độ phức tạp của mô hình và khả năng khái quát hóa.

Đối với mô hình RFR, chúng tôi thiết lập không gian tìm kiếm với các giá trị n_estimators (số lượng cây) từ 50 đến 300 với bước nhảy 50, max_depth (độ sâu tối đa của cây) từ 10 đến 30 với bước nhảy 1 và thêm giá trị None, min_samples_split (số mẫu tối thiểu để phân tách) từ 2 đến 10 với bước nhảy 2, và min_samples_leaf (số mẫu tối thiểu ở nút lá) từ 1 đến 4 với bước nhảy 1. Quá trình tìm kiếm này thực hiện 50 lần lặp ngẫu nhiên. Kết quả các siêu tham số tối ưu cho RFR là n_estimators = 300, min_samples_split = 4, min_samples_leaf = 1, và max_depth = 22. Chúng tôi sử dụng random_state = 42 cho cả RandomizedSearchCV và mô hình RFR để đảm bảo tính tái lập của kết quả.

Bước 5: Đánh giá mô hình.

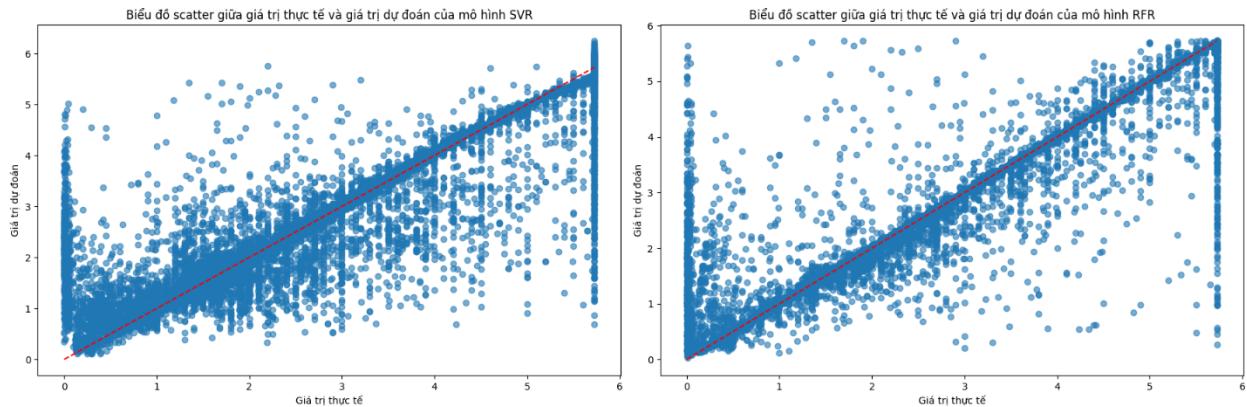
Đánh giá hiệu suất của mô hình Random Forest Regressor qua ba chỉ số đánh giá phổ biến: Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), R-squared (R^2), và R-squared hiệu chỉnh (Adjusted R^2) (Bảng 4.8).

Bảng 4.8: So sánh kết quả giữa SVR và RandomForest.

Độ đo	SVR	RFR
MAE	0,2978	0,1617
MSE	0,4278	0,2817
RMSE	0,6541	0,5307
R^2	0,7892	0,8611
Adjusted R^2	0,7887	0,8608
Thời gian thực thi	97,07 s	212,14 s

Random Forest Regressor (RFR) vượt trội hơn Support Vector Regression (SVR) ở hầu hết các chỉ số đánh giá. Cụ thể, RFR có các giá trị MAE, MSE và RMSE thấp hơn, đồng nghĩa với việc có độ chính xác cao hơn và ít lỗi hơn trong dự đoán. Bên cạnh đó, giá trị R^2 và R^2 hiệu chỉnh của RFR cũng cao hơn, cho thấy khả năng giải thích biến thiên của dữ liệu tốt hơn. Mặc dù RFR có thời gian thực thi dài hơn so với SVR, tuy nhiên, sự chênh lệch này là không đáng kể so với lợi ích mà RFR mang lại về mặt độ chính xác và khả năng giải thích. Nhìn chung, mô hình RFR là lựa chọn tốt hơn cho bài toán dự đoán giá nhà trong trường hợp này.

Chương 4. Kết quả thực nghiệm



Hình 4.11. Biểu đồ scatter thể hiện mối quan hệ giữa giá trị thực tế và giá trị dự đoán của mô hình SVR và RFR

Hình 4.11 thể hiện mối quan hệ giữa giá trị thực tế và giá trị dự đoán của hai mô hình dự đoán giá nhà: Support Vector Regressor (SVR) và Random Forest Regressor. Biểu đồ của SVR cho thấy các điểm dữ liệu nằm rải rác và có độ phân tán lớn xung quanh đường lý tưởng (đường màu đỏ), đặc biệt là ở các giá trị thấp và cao. Điều này cho thấy mô hình SVR có sự bất ổn định và độ chính xác không cao trong việc dự đoán giá trị thực tế. Trong khi đó, biểu đồ của Random Forest Regressor cho thấy các điểm dữ liệu tập trung gần đường lý tưởng hơn, cho thấy mô hình này có khả năng dự đoán chính xác và ổn định hơn. Các điểm dữ liệu ít bị phân tán hơn, đặc biệt là ở các giá trị thấp và cao, chứng tỏ mô hình Random Forest Regressor có độ chính xác cao hơn trong việc dự đoán giá trị thực tế. Tổng quan, Random Forest Regressor cho thấy sự ưu việt hơn so với SVR trong việc dự đoán giá trị thực tế, với độ chính xác cao hơn và sự ổn định tốt hơn. Điều này cho thấy Random Forest Regressor là một mô hình dự đoán hiệu quả và đáng tin cậy hơn so với SVR trong bài toán dự đoán giá nhà.

4.2.2. Bài toán 2: Xây dựng mô hình dự đoán bệnh nhân có mắc bệnh tiểu đường hay không.

Bước 1: Thu thập dữ liệu.

Bước 2: Xử lý và làm sạch dữ liệu.

Đọc và hiển thị 5 dòng dữ liệu (Hình 4.12).

Chương 4. Kết quả thực nghiệm

	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_level	diabetes
0	Female	80.0	0	1	never	25.19	6.6	140	0
1	Female	54.0	0	0	No Info	27.32	6.6	80	0
2	Male	28.0	0	0	never	27.32	5.7	158	0
3	Female	36.0	0	0	current	23.45	5.0	155	0
4	Male	76.0	1	1	current	20.14	4.8	155	0

Hình 4.12. Hiển thị 5 dòng dữ liệu.

Hiển thị các thông tin về số lượng hàng, cột, tên cột, số lượng giá trị không null, kiểu dữ liệu (Bảng 4.9) và hiển thị thống kê tóm tắt dữ liệu (Hình 4.13).

Bảng 4.9. Thông tin về dữ liệu.

RangeIndex: 100000 entries, 0 to 99999			
Data columns (total 9 columns):			
#	Column	Non-Null Count	Dtype
0	gender	100000 non-null	object
1	age	100000 non-null	float64
2	hypertension	100000 non-null	int64
3	heart_disease	100000 non-null	int64
4	smoking_history	100000 non-null	object
5	bmi	100000 non-null	float64
6	HbA1c_level	100000 non-null	float64
7	blood_glucose_level	100000 non-null	int64
8	diabetes	100000 non-null	int64
dtypes: float64(3), int64(4), object(2)			
memory usage: 6.9+ MB			

	age	hypertension	heart_disease	bmi	HbA1c_level	blood_glucose_level	diabetes
count	96146.000000	96146.000000	96146.000000	96146.000000	96146.000000	96146.000000	96146.000000
mean	41.794326	0.077601	0.040803	27.321461	5.532609	138.218231	0.088220
std	22.462948	0.267544	0.197833	6.767716	1.073232	40.909771	0.283616
min	0.080000	0.000000	0.000000	10.010000	3.500000	80.000000	0.000000
25%	24.000000	0.000000	0.000000	23.400000	4.800000	100.000000	0.000000
50%	43.000000	0.000000	0.000000	27.320000	5.800000	140.000000	0.000000
75%	59.000000	0.000000	0.000000	29.860000	6.200000	159.000000	0.000000
max	80.000000	1.000000	1.000000	95.690000	9.000000	300.000000	1.000000

Hình 4.13. Thống kê tóm tắt dữ liệu.

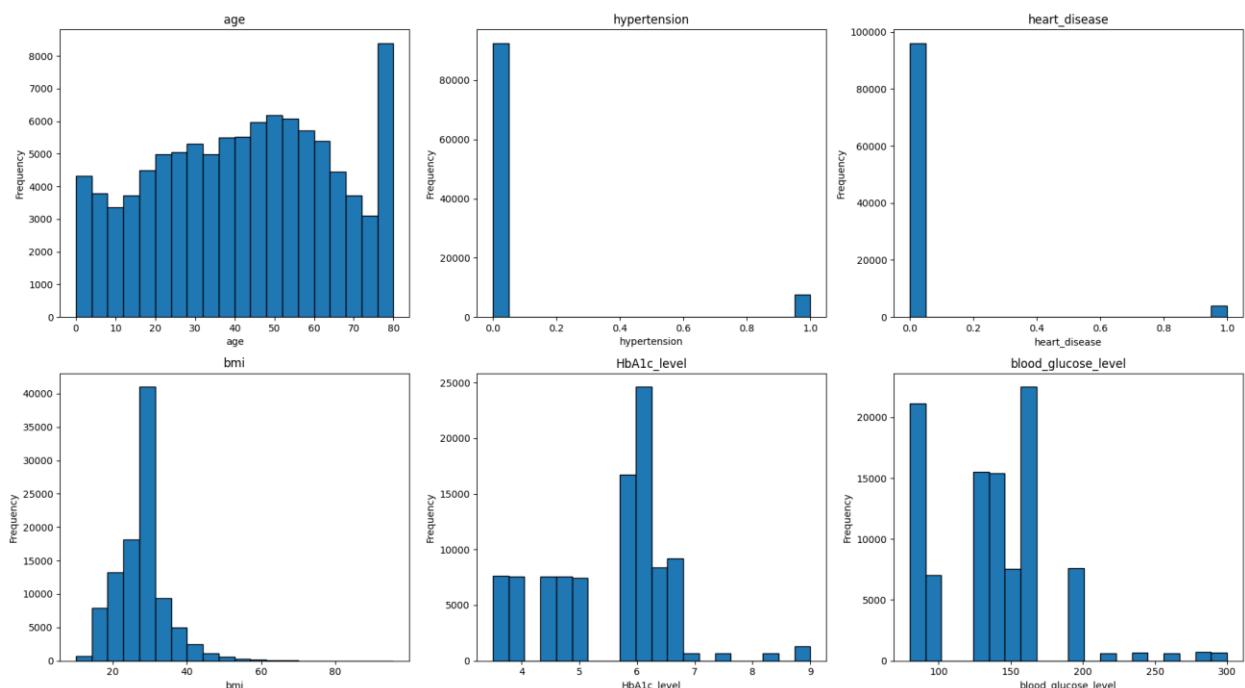
Chương 4. Kết quả thực nghiệm

```
data['age'].unique()  
  
array([80., 54., 28., 36., 76., 20., 44., 79., 42.,  
       32., 53., 78., 67., 15., 37., 40., 5., 69.,  
       72., 4., 30., 45., 43., 50., 41., 26., 34.,  
       73., 77., 66., 29., 59., 60., 38., 3., 57., 74.,  
       19., 46., 21., 59., 27., 13., 56., 2., 7.,  
       11., 6., 55., 9., 62., 47., 12., 68., 75.,  
       22., 58., 18., 24., 17., 25., 0.08, 33., 16.,  
       61., 31., 8., 49., 39., 65., 14., 70., 0.56,  
       48., 51., 71., 0.88, 64., 63., 52., 0.16, 10.,  
       35., 23., 0.64, 1.16, 1.64, 0.72, 1.88, 1.32, 0.8,  
       1.24, 1., 1.8., 0.48, 1.56, 1.08, 0.24, 1.4, 0.4,  
       0.32, 1.72, 1.48])
```

Hình 4.14. Các giá trị unique của cột age.

Hình 4.14 cho thấy các giá trị duy nhất của cột ‘age’ có các giá trị không hợp lý như 0.08, 0.56, 1.56,... Nên ta sẽ chuyển đổi kiểu dữ liệu của biến age thành int.

Vẽ biểu đồ histogram của các cột dữ liệu số (Hình 4.15).



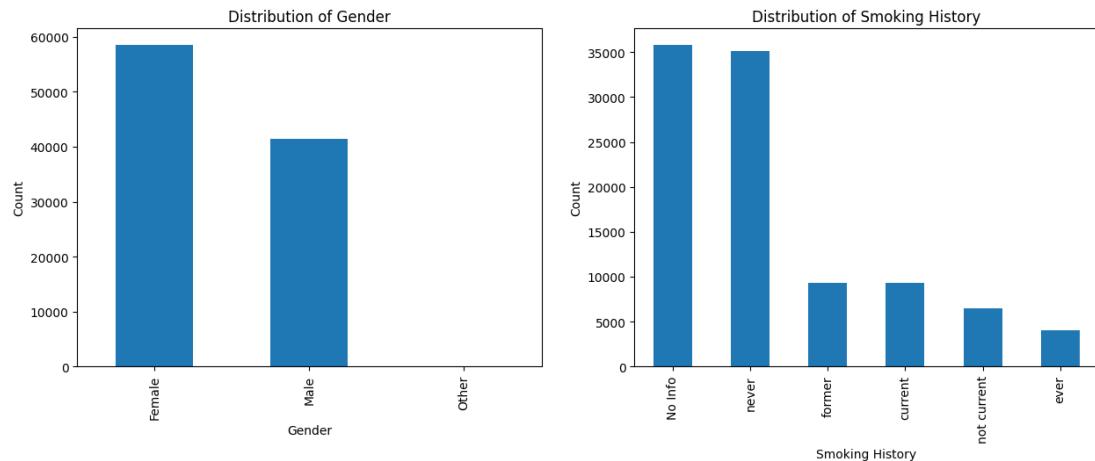
Hình 4.15. Biểu đồ Histogram của các cột dữ liệu số

- Độ tuổi của các bệnh nhân phân bố khá đồng đều và đáng chú ý ở độ tuổi từ 75-80.
- Phần lớn các bệnh nhân đều không bị tăng huyết áp và không mắc bệnh tim.
- Phân bố BMI bị lệch phải, BMI của các bệnh nhân chủ yếu nằm trong khoảng từ 15-40, với đỉnh cao nhất xung quanh mức 25-35. Điều này cho thấy BMI của các bệnh nhân nằm trong khoảng bình thường hoặc thừa cân nhẹ.
- Mức HbA1c được phân bố không đều với nhiều đỉnh, tập trung nhiều ở mức 5.5-6.5.

Chương 4. Kết quả thực nghiệm

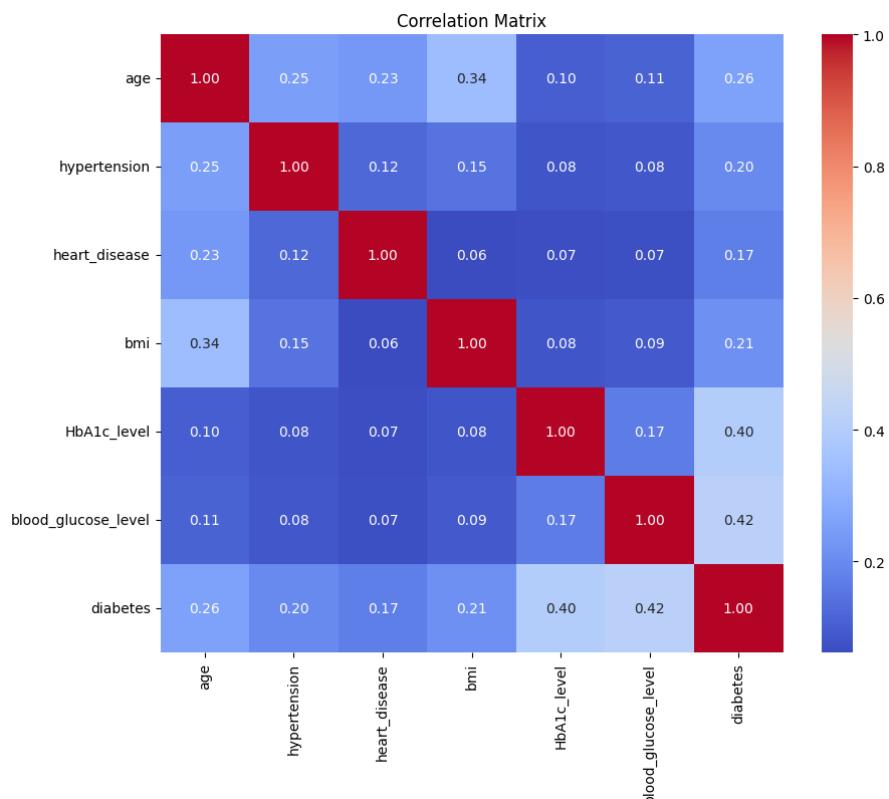
- Blood Glucose Level phân bố không đều với nhiều đỉnh, Đa số giá trị nằm trong khoảng 80-200. Bên cạnh đó có một số giá trị cao bất thường.

Trực quan dữ liệu cột ‘gender’ và ‘smoking_history’(Hình 4.16)



Hình 4.16: Biểu đồ trực quan Gender và Smoking History

Số lượng bệnh nhân là nữ cao hơn nam bên cạnh đó có phần phần rất nhỏ về giới tính khác. Phần lớn không có thông tin về lịch sử hút thuốc và lượng lớn các bệnh nhân không hút thuốc. Có một số lượng đáng kể người đã từng hút và đang hút thuốc..



Hình 4.17: Biểu đồ heatmap thể hiện sự tương quan giữa các biến.

Hầu hết các mối tương quan giữa các biến với biến diabetes đều ở mức thấp đến trung bình, cho thấy không có mối quan hệ tuyến tính mạnh giữa đa số các biến. Blood_glucose_level và HbA1c_level có mức độ tương quan cao nhất đối với biến diabetes lần lượt là 0,42 và 0,4. Điều này cho thấy rằng Blood_glucose_level và HbA1c_level có khả năng là đặc trưng dự đoán quan trọng nhất trong tập dữ liệu này. Các đặc trưng còn lại có mối tương quan thấp hơn nhưng vẫn có ý nghĩa, có thể ảnh hưởng đến khả năng dự đoán nhưng với mức độ thấp hơn.

- **Xử lý giá trị trùng.**

Tìm và hiển thị các giá trị trùng (Hình 4.18).

Number of duplicate rows: 3888										
	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_level	diabetes	
2756	Male	80	0	0	No Info	27.32	6.6	159	0	
3272	Female	80	0	0	No Info	27.32	3.5	80	0	
3418	Female	19	0	0	No Info	27.32	6.5	100	0	
3939	Female	78	1	0	former	27.32	3.5	130	0	
3960	Male	47	0	0	No Info	27.32	6.0	200	0	

Hình 4.18: Các giá trị trùng nhau.

Dựa vào kết quả trên cho thấy tập dữ liệu có 3888 giá trị trùng nhau. Ta tiến hành loại bỏ các giá trị trùng nhau.

- **Xử lý giá trị null.**

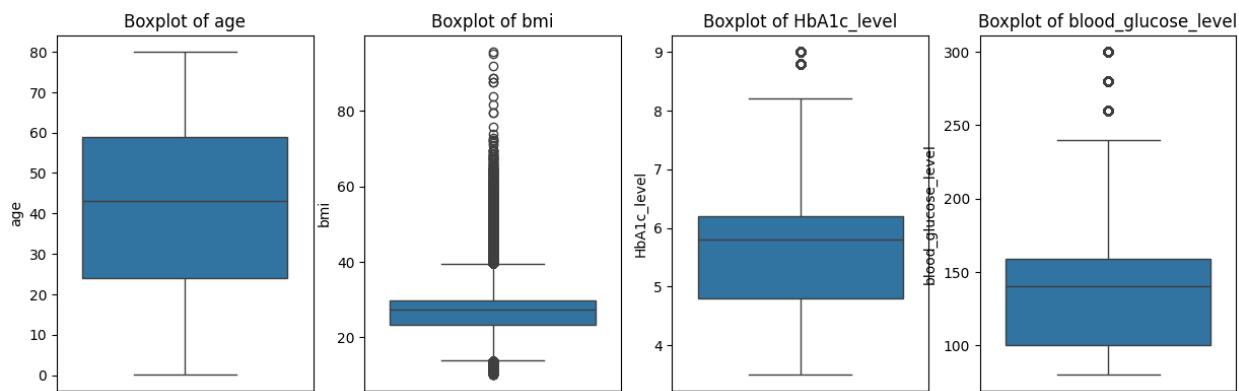
Sau khi kiểm tra ta nhận thấy tập dữ liệu không có giá trị null (Bảng 4.10).

Bảng 4.10: Sau khi xử lý các giá trị null.

Features	Phần trăm null
gender	0.0
age	0.0
hypertension	0.0
heart_disease	0.0
smoking_history	0.0
bmi	0.0
HbA1c_level	0.0
blood_glucose_level	0.0

- **Xử lý giá trị ngoại lệ.**

Vẽ biểu đồ boxplot cho các cột 'age', 'bmi', 'HbA1c_level', 'blood_glucose_level' để xác định outliers (Hình 4.19).



Hình 4.19. Biểu đồ hộp trước khi xử lý ngoại lệ.

Đối với tập dữ liệu này ta sẽ không xử lý các dữ liệu ngoại lai vì các lý do sau:

- age: Các giá trị gần 0 có thể là hợp lệ, chẳng hạn như trẻ sơ sinh hoặc trẻ nhỏ có thể bị tiểu đường do yếu tố di truyền.
- bmi: Những giá trị BMI rất cao có thể đại diện cho các tình trạng y tế đặc biệt hoặc nhóm đối tượng cụ thể có ý nghĩa trong bối cảnh nghiên cứu y tế. Việc loại bỏ những giá trị này có thể làm mất thông tin quan trọng cho việc dự đoán bệnh tiểu đường.
- HbA1c_level: Mức HbA1c cao có thể cho thấy tình trạng kiểm soát đường huyết kém, và có ý nghĩa y tế.
- blood_glucose_level: Mức đường huyết cao thường xuất hiện ở những người có vấn đề với đường huyết hoặc đã mắc bệnh tiểu đường.

Loại bỏ những giá trị này có thể làm mất các chỉ số quan trọng cho phân tích.

• Mã hóa và chuẩn hóa dữ liệu

Dùng One-hot Encoder mã hóa biến ‘gender’, ‘smoking_history’ thành số vì dữ liệu giá trị phân loại không mang ý nghĩa thứ tự. Kết quả sau khi thực hiện (Bảng 4.11):

Bảng 4.11: Sau khi mã hóa

#	Column	Non-Null Count	Dtype
0	age	96112 non-null	float64
1	hypertension	96112 non-null	float64
2	heart_disease	96112 non-null	float64
3	bmi	96112 non-null	float64
4	HbA1c_level	96112 non-null	float64
5	blood_glucose_level	96112 non-null	float64
6	gender_Female	96112 non-null	float64
7	gender_Male	96112 non-null	float64
8	gender_Other	96112 non-null	float64

Chương 4. Kết quả thực nghiệm

9	smoking_history_No Info	96112 non-null	float64
10	smoking_history_current	96112 non-null	float64
11	smoking_history_ever	96112 non-null	float64
12	smoking_history_former	96112 non-null	float64
13	smoking_history_never	96112 non-null	float64
14	smoking_history_not current	96112 non-null	float64
dtypes: float64(15)			
memory usage: 11.0 MB			

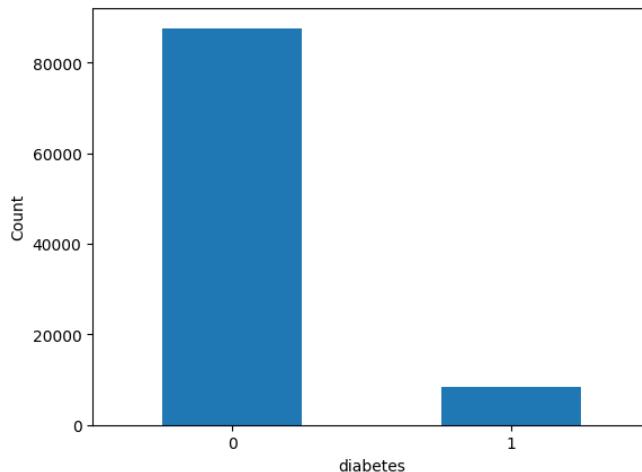
Sử dụng StandardScaler để chuẩn hóa dữ liệu đưa các đặc trưng về 1 thang đo với giá trị trung bình là 0 và độ lệch chuẩn là 1. Sau đó chuyển đổi dữ liệu đã được chuẩn hóa thành Dataframe (Hình 4.20).

	age	hypertension	heart_disease	bmi	HbA1c_level	blood_glucose_level	gender_Female	gender_Male	gender_Other	smoking_history_No Info	smoking_history_current	smoking_history_ever	smoking_history_former	smoking_history_never	smoking_history_not current
0	1.699910	-0.290106	4.847640	-0.314977	0.994503	0.043484	0.843768	-0.843443	-0.013686	-0.720653	-0.325294	-0.208333	-0.327285	1.339446	-0.266356
1	0.542955	-0.290106	-0.206286	-0.000284	0.994503	-1.423111	0.843768	-0.843443	-0.013686	1.387630	-0.325294	-0.208333	-0.327285	-0.746577	-0.266356
2	-0.614000	-0.290106	-0.206286	-0.000284	0.155953	0.483463	-1.185160	1.185617	-0.013686	-0.720653	-0.325294	-0.208333	-0.327285	1.339446	-0.266356
3	-0.258014	-0.290106	-0.206286	-0.572050	-0.496252	0.410133	0.843768	-0.843443	-0.013686	-0.720653	3.074144	-0.208333	-0.327285	-0.746577	-0.266356
4	1.521917	3.447016	4.847640	-1.061081	-0.682596	0.410133	-1.185160	1.185617	-0.013686	-0.720653	3.074144	-0.208333	-0.327285	-0.746577	-0.266356

Hình 4.20: Kết quả sau khi Feature Scaling.

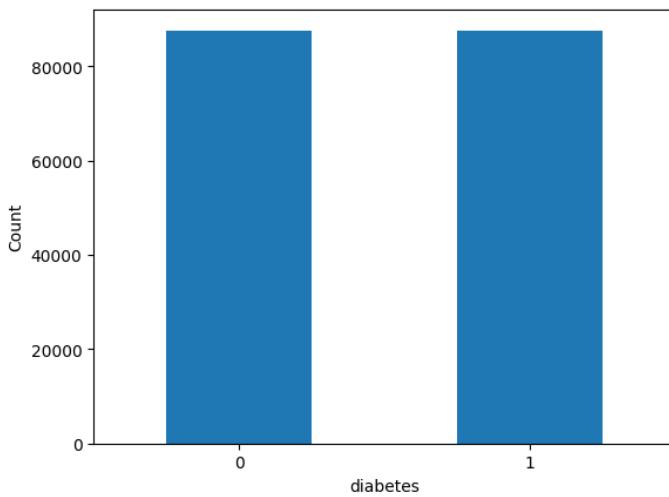
- **Xử lý mất cân bằng dữ liệu.**

Vẽ biểu đồ plot hiển thị số lượng mỗi giá trị của cột ‘diabetes’ (Hình 4.21).



Hình 4.21. Biểu đồ cột của cột Outcome

Dựa vào biểu đồ trên ta thấy sự phân bố giá trị của cột Outcome không đồng đều giữa giá trị 0 và giá trị 1. Nên ta dùng SMOTE để xử lý mất cân bằng dữ liệu.



Hình 4.22. Biểu đồ cột của thuộc tính diabetes sau khi xử lý mất cân bằng.

Bước 3: Trích chọn đặc trưng và giảm chiều dữ liệu.

- **Feature Selection.**

Sử dụng phương pháp wrapper với kỹ thuật Recursive feature elimination (RFE) để trích chọn đặc trưng. Tạo mô hình LogisticRegression để làm mô hình hồi quy. Tạo đối tượng RFECV với estimator=model: Sử dụng mô hình LogisticRegression để đánh giá các tập hợp đặc trưng, step=1: mỗi lần loại bỏ một đặc trưng kém nhất, cv=KFold(5): sử dụng K-Fold Cross-Validation với 5 fold để đánh giá mô hình, scoring='r2': sử dụng R-squared làm thước đo hiệu suất mô hình.

```
Optimal number of features: 10
Selected Features: Index(['age', 'hypertension', 'heart_disease', 'bmi', 'HbA1c_level',
   'blood_glucose_level', 'gender_Female', 'gender_Male', 'gender_Other',
   'smoking_history_No Info'],
   dtype='object')
Feature Ranking: [1 1 1 1 1 1 1 1 1 2 5 3 4 6]
```

Hình 4.23: Kết quả thực hiện Feature Selection.

Sau khi thực hiện Feature Selection (Hình 4.23) ta sẽ chọn ra 10 đặc trưng từ 15 đặc trưng ban đầu là: ‘age’, ‘hypertension’, ‘heart_disease’, ‘bmi’, ‘HbA1c_level’, ‘blood_glucose_level’, ‘gender_Female’, ‘gender_Male’, ‘gender_Other’, ‘smoking_history_No Info’.

- **Giảm chiều dữ liệu.**

Với tập dữ liệu này chúng ta không thực hiện giảm số chiều bằng LDA vì kết quả dự đoán của mô hình sau khi thực hiện giảm số chiều bằng LDA kém hơn so với khi sử dụng

Chương 4. Kết quả thực nghiệm

dữ liệu ban đầu có thể là do LDA không phù hợp với dữ liệu phi tuyến tính đã được nhận xét Hình 4.17. Kết quả so sánh hiệu quả của các mô hình được biểu hiện ở Bảng 4.12:

Bảng 4.12: Kết quả khi dùng LDA và không dùng LDA.

Độ đo	Logistic Regression		Decision Tree		RandomForest	
	Không dùng LDA	Dùng LDA	Không dùng LDA	Dùng LDA	Không dùng LDA	Dùng LDA
Accuracy	0,8848	0,8858	0,9626	0,8465	0,9740	0,8465
Precision	0,8863	0,8865	0,9620	0,8492	0,9736	0,8486
Recall	0,8829	0,8849	0,9632	0,8429	0,9744	0,8433
F1-score	0,8846	0,8857	0,9626	0,8460	0,9740	0,8459
Thời gian thực thi	0.3186s	0.1878s	0.5889s	1.0965s	33.9707s	47.4012s

Bước 4+5: Xây dựng và đánh giá mô hình.

Chia tập dữ liệu thành tập huấn luyện và kiểm tra. Với 20% cho tập test và 80% cho train. Stratify = y_resampled đảm bảo rằng phân phối của nhãn trong tập huấn luyện và tập kiểm tra giữ nguyên tỉ lệ như nhau. Khởi tạo và huấn luyện mô hình các mô hình Logistic Regression, Decision Tree, RandomForest và dự đoán mô hình trên tập dữ liệu kiểm tra. Hiển thị độ chính xác và báo cáo phân loại chi tiết về hiệu suất của mô hình trên tập dữ liệu kiểm tra. Sau khi phân tích, xử lý và xây dựng mô hình Logistic Regression, Decision Tree, và Random Forest ta có được kết quả sau:

Bảng 4.13: Kết quả của mô hình Logistic Regression.

Logistic Regression				
Accuracy: 0.8847997261211914				
Classification Report:				
	precision	recall	f1-score	support
0	0.88	0.89	0.89	17526
1	0.89	0.88	0.88	17526
accuracy			0.88	35052
macro avg	0.88	0.88	0.88	35052
weighted avg	0.88	0.88	0.88	35052

Bảng 4.14: Kết quả của mô hình Decision Tree.

Decision Tree				
Accuracy: 0.9620849024306745				
Classification Report:				
	precision	recall	f1-score	support
0	0.96	0.96	0.96	17526
1	0.96	0.96	0.96	17526

accuracy			0.96	35052
macro avg	0.96	0.96	0.96	35052
weighted avg	0.96	0.96	0.96	35052

Bảng 4.15: Kết quả của mô hình Random Forest.

Random Forest				
Accuracy: 0.9740671003081136				
Classification Report:				
	precision	recall	f1-score	support
0	0.97	0.97	0.97	17526
1	0.97	0.97	0.97	17526
accuracy			0.97	35052
macro avg	0.97	0.97	0.97	35052
weighted avg	0.97	0.97	0.97	35052

Bảng 4.16: Tổng quan 3 mô hình.

Độ đo	Logistic Regression	Decision Tree	RandomForest
Accuracy	0,8848	0,9626	0,9740
Precision	0,8863	0,9620	0,9736
Recall	0,8829	0,9632	0,9744
F1-score	0,8846	0,9626	0,9740
Thời gian thực thi	0.3186s	0.5889s	33.9707s

Dựa vào bảng kết quả đánh giá các mô hình, ta thấy là sau khi huấn luyện các mô hình đều có độ chính xác khá cao. Đặc biệt là các mô hình Decision Tree, Random Forest đều có độ chính xác 96% - 97%. Đồng thời thì các thông số khác như độ chuẩn xác (precision), độ bao phủ (recall), F1-score đều cao. Trong cả 3 mô hình được train thì mô hình Random Forest có độ chính xác cao nhất là 0.974 tuy nhiên thời gian huấn luyện tương đối lâu 33.9707 giây. Khi so sánh mô hình Random Forest với mô hình độ chính xác cao thứ 2 là Decision Tree (0.9626) thì sự chênh lệch về độ chính xác của 2 mô hình là rất nhỏ khoảng 0.0114, trong khi đó thì mô hình Decision Tree lại có thời gian huấn luyện tương đối ngắn chỉ 0.5889 giây. Vì vậy mà mô hình tốt nhất được lựa chọn để triển khai ở đây là Decision Tree.

4.2.3. Bài toán 3: Phân loại thư rác (Spam email)

Bước 1: Thu thập dữ liệu

Import các thư viện cần thiết và đọc dữ liệu vào (Hình 4.24).

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...		NaN	NaN
1	ham	Ok lar... Joking wif u oni...		NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...		NaN	NaN
3	ham	U dun say so early hor... U c already then say...		NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...		NaN	NaN

Hình 4.24. Năm dòng đầu của tập dữ liệu

Bước 2: Xử lý và làm sạch dữ liệu

Cắt bỏ các cột không có giá trị phân tích: ‘Unnamed: 2’, ‘Unnamed: 3’, ‘Unnamed:4’, đổi tên các cột lại cho dễ nhận biết v1→target , v2→text (Hình 4.25).

	target	text
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

Hình 4.25. Dữ liệu sau khi cắt bỏ các cột không có giá trị phân tích và đổi tên cột

- Mã hóa dữ liệu

Thực hiện mã hóa biến mục tiêu (target) bằng phương pháp LabelEncoder từ thư viện scikit-learn. Kết quả (Hình 4.26) là các giá trị chuỗi trong cột target đã được thay thế bằng các giá trị số, với "ham" được mã hóa thành 0 và "spam" được mã hóa thành 1.

	target	text
0	0	Go until jurong point, crazy.. Available only ...
1	0	Ok lar... Joking wif u oni...
2	1	Free entry in 2 a wkly comp to win FA Cup fina...
3	0	U dun say so early hor... U c already then say...
4	0	Nah I don't think he goes to usf, he lives aro...

Hình 4.26. Kết quả sau khi mã hóa

- Xử lý dữ liệu null

Kết quả cho thấy tập dữ liệu không có dữ liệu null (Hình 4.27).

```
df.isnull().sum()
```

```
          0  
target  0  
text    0
```

```
dtype: int64
```

Hình 4.27. Kết quả kiểm tra dữ liệu null

- Xử lý dữ liệu trùng lặp

Sử dụng phương thức duplicated() và sum() để xác định và đếm số lượng dòng trùng lặp trong DataFrame. Kết quả cho thấy có 403 dòng trùng lặp. Sau đó chúng tôi sử dụng phương thức drop_duplicates(keep='first') để loại bỏ các dòng trùng lặp. Kết quả cho thấy DataFrame giảm từ 5572 dòng dữ liệu xuống còn 5169 dòng dữ liệu (Bảng 4.17).

Bảng 4.17. Thông tin của tập dữ liệu sau khi loại bỏ dữ liệu trùng lặp

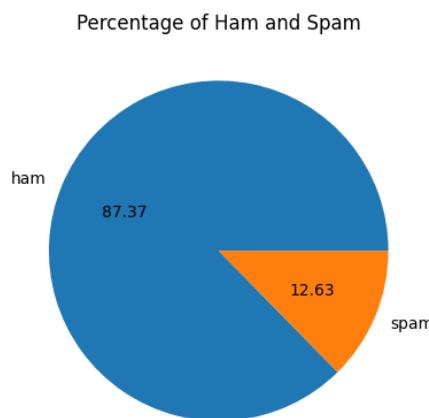
Index: 5169 entries, 0 to 5571

Data columns (total 2 columns):

	Column	Non-Null Count	Dtype
0	Target	5169 non-null	Int64
1	Text	5169 non-null	object

dtypes: int64(1), object(1)
memory usage: 121.1+ KB

Tính tỷ lệ nhãn Ham và Spam trong cột target, đồng thời trực quan thành biểu đồ tròn để dễ quan sát.



Hình 4.28. Biểu đồ tròn thể hiện phần trăm của ham và spam

Như bạn thấy trên biểu đồ Hình 4.28, tỷ lệ ham quá cao (87,37%) so với tỷ lệ tin nhắn rác (ham) cho thấy dữ liệu mất cân bằng. Ở bài toán này dù dữ liệu bị mất cân bằng, nhưng chúng tôi đã cân nhắc không áp dụng các kỹ thuật xử lý mất cân bằng dữ liệu vì nó không phù hợp với tính chất rời rạc và đa chiều của dữ liệu văn bản, có thể tạo ra các mẫu tổng hợp không có ý nghĩa, làm mất đi cấu trúc ngôn ngữ quan trọng. Mà ở đây chúng tôi đã sử dụng thuật toán SVM, Random Forest, với khả năng điều chỉnh trọng số của các lớp để đạt được sự cân bằng mà không cần can thiệp trực tiếp vào dữ liệu gốc.

Bước 3: Chuẩn hóa dữ liệu

- Mở rộng từ viết tắt

Mở rộng các dạng viết tắt phổ biến trong tiếng Anh thành dạng đầy đủ trong cột text của DataFrame và lưu kết quả vào cột mới transformed_text. Cụ thể, các bước thực hiện như sau:

- Tạo từ điển viết tắt (contractions_dict): Từ điển này chứa các dạng viết tắt làm khóa và các dạng đầy đủ tương ứng làm giá trị.
- Biểu thức chính quy (contractions_re): Tạo một biểu thức chính quy để nhận diện các từ viết tắt có trong từ điển.
- Hàm mở rộng viết tắt (expand_contractions): Hàm này nhận một chuỗi văn bản và thay thế các từ viết tắt bằng dạng đầy đủ của chúng.
- Sử dụng hàm apply của pandas để áp dụng hàm expand_contractions cho từng giá trị trong cột text, và lưu kết quả vào cột mới transformed_text.

```
contractions_re=re.compile('(%s)' % '|'.join(contractions_dict.keys()))

def expand_contractions(text,contractions_dict=contractions_dict):
    def replace(match):
        return contractions_dict[match.group(0)]
    return contractions_re.sub(replace, text)

df['transformed_text']=df['text'].apply(lambda x:expand_contractions(x))
```

target	text	num_characters	num_words	num_sentence	transformed_text
0	Go until jurong point, crazy.. Available only ...	111	24	2	Go until jurong point, crazy.. Available only ...
1	Ok lar... Joking wif u oni...	29	8	2	Ok lar... Joking wif u oni...
2	Free entry in 2 a wkly comp to win FA Cup fina...	155	37	2	Free entry in 2 a wkly comp to win FA Cup fina...
3	U dun say so early hor... U c already then say...	49	13	1	U dun say so early hor... U c already then say...
4	Nah I don't think he goes to usf, he lives aro...	61	15	1	Nah I do not think he goes to usf, he lives ar...

Hình 4.29. Kết quả sau khi mở rộng từ viết tắt

- Thực hiện chuyển văn bản sang chữ thường, tách từ (tokenization), loại bỏ ký tự đặc biệt, đánh nhãn loại từ, loại bỏ từ dừng và dấu câu, thực hiện lemmatization chuyển về dạng văn bản gốc.

Đầu tiên ta sẽ thực hiện việc chuẩn hóa trên bằng thư viện nltk. Xây dựng hàm get_wordnet_pos để xác định loại từ (part of speech) cho mỗi từ. Hàm chính transform_text thực hiện các bước xử lý văn bản theo thứ tự: chuyển đổi thành chữ thường, tách từ (tokenization), loại bỏ ký tự đặc biệt và số, loại bỏ stopwords và dấu câu, và cuối cùng là lemmatization để đưa các từ về dạng gốc.

```
print(transform_text('Ok lar... Joking wif u oni...123'))  
  
Word: ok, POS: n  
Word: lar, POS: n  
Word: joking, POS: n  
Word: wif, POS: n  
Word: u, POS: n  
Word: oni, POS: n  
Word: 123, POS: n  
ok lar joking wif u oni 123
```

Hình 4.30. Kết quả thực hiện chuẩn hóa dữ liệu bằng thư viện nltk

Kết quả (Hình 4.30) nhận được khi chạy 1 ví dụ mail thử thì kết quả cho ra không được chính xác lắm, ở đây từ ‘joking’ phải là có từ loại là ‘v’ tuy nhiên, POS Tagging trong thư viện nltk lại phân loại thành ‘n’. Vì thế lúc này ta sẽ sử dụng một thư viện có khả năng chuẩn hóa dữ liệu text chuẩn xác hơn là thư viện ‘spacy’.

Thực hiện quá trình xử lý và chuẩn hóa văn bản tiếng Anh sử dụng thư viện SpaCy. Hàm transform_text thực hiện các bước xử lý văn bản theo trình tự: chuyển đổi thành chữ thường, loại bỏ các ký tự đặc biệt và loại bỏ khoảng trắng thừa. Sau đó, văn bản được xử lý bởi mô hình SpaCy để thực hiện tokenization và gán nhãn từ loại (POS tagging). Tiếp theo, hàm loại bỏ các stopwords và thực hiện lemmatization để đưa các từ về dạng gốc. Cuối cùng, các token đã xử lý được ghép lại thành một chuỗi.

```
print(transform_text('Ok lar... Joking wif u oni...123'))  
  
ok lar joke wif u oni 123
```

Hình 4.31. Kết quả chuẩn hóa dữ liệu bằng thư viện spacy

Kết quả (Hình 4.31) nhận được lúc này đã chính xác hơn khi ‘joking’ được chuyển thành từ gốc ‘joke’. Sau đó chúng ta sẽ tiến hành chuẩn hóa dữ liệu cho cả tập dữ liệu (Hình 4.32).

Chương 4. Kết quả thực nghiệm

target		text	num_characters	num_words	num_sentence	transformed_text
0	0	Go until jurong point, crazy.. Available only ...	111	24	2	jurong point crazy available bugis n great wor...
1	0	Ok lar... Joking wif u oni...	29	8	2	ok lar joke wif u oni
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37	2	free entry 2 wkly comp win fa cup final tkts 2...
3	0	U dun say so early hor... U c already then say...	49	13	1	u dun early hor u c
4	0	Nah I don't think he goes to usf, he lives aro...	61	15	1	nah think usf live

Hình 4.32. Dữ liệu sau khi chuẩn hóa

Phân tích dữ liệu thông qua:

- Word cloud

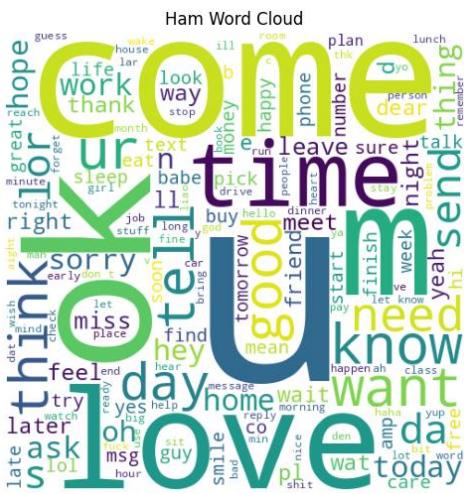
Tạo ra một đám mây từ (word cloud) từ các tin nhắn được gắn nhãn là spam.



Hình 4.33. Word cloud của dữ liệu spam

Thông qua biểu đồ (Hình 4.33) ta thấy là có những từ xuất hiện nhiều trong văn bản spam là free, txt, text, mobile. Điều này cho thấy các tin nhắn spam thường sử dụng các từ khóa liên quan đến quảng cáo, khuyến mãi và dịch vụ di động để thu hút sự chú ý và thúc đẩy người nhận tham gia hoặc mua sản phẩm/dịch vụ.

Tạo ra một đám mây từ (word cloud) từ các tin nhắn được gắn nhãn là ham.



Hình 4.34. Biểu đồ word cloud của ham

Thông qua biểu đồ (Hình 4.34) ta thấy là có những từ xuất hiện nhiều trong văn bản ham là u, come, ok, love, time. Điều này cho thấy các tin nhắn ham tập trung vào giao tiếp hàng ngày và thể hiện mối quan hệ cá nhân, với các từ khóa phản ánh cuộc trò chuyện thân mật và sự tương tác giữa các cá nhân.

- N-grams

Sử dụng hàm word_tokenize từ thư viện NLTK để tách các câu trong cột transformed_text thành các từ riêng lẻ và lưu kết quả vào cột mới cleaned_token. Sau đó, thực hiện hàm extract_ngrams để tạo ra các n-grams với giá trị $n = 3$ để tạo ra các trigrams. Tiếp theo, hàm combine_words kết hợp tất cả các danh sách từ trong cleaned_token thành một danh sách duy nhất. Sau đó, hàm count_topwords đếm số lần xuất hiện của mỗi từ.

Bảng 4.18. Kết quả đếm tần suất xuất hiện của các từ tri-ngrams

	words	word_count
5194	u wan 2	21
4626	happy new year	17
3254	2003 account statement	13
3253	private 2003 account	13
150	t c www	13
2634	land line claim	12
3261	s m point	12
2214	suite342 2land row	12
3258	800 un redeem	12
3259	un redeem s	12

Kết quả đếm tần suất xuất hiện của các từ tri-ngrams (Bảng 4.18) cho thấy một số cụm từ phổ biến trong dữ liệu. Cụm từ ‘u wan 2’ (21 lần) và ‘happy new year’ (17 lần) cho thấy các mẫu câu giao tiếp thường ngày và lời chúc mừng. Điều này có thể chỉ ra rằng dữ liệu chứa nhiều nội dung liên quan đến cuộc trò chuyện thường ngày. ‘2003 account statement’ (13 lần) và ‘private 2003 account’ (13 lần) là các cụm từ liên quan đến thông tin tài chính. Điều này cho thấy có sự xuất hiện của các dữ liệu về tài khoản và báo cáo tài chính. ‘t c www’ (13 lần) có thể đại diện cho các ký tự hoặc từ viết tắt không mang ý nghĩa rõ ràng. Cụm từ này có thể là một phần của địa chỉ trang web.

Bước 4: Vector hóa dữ liệu

Chuyển đổi văn bản sau khi xử lý thành các đặc trưng số bằng kỹ thuật TF-IDF với tham số max_features = 3000, tức là chỉ giữ lại 3000 từ có giá trị TF-IDF cao nhất. Sau đó, TfidfVectorizer được áp dụng lên cột 'transformed_text' để tạo các vector TF-IDF và chuyển đổi thành mảng numpy. Tiếp theo, cột 'target' được trích xuất và chuyển thành mảng numpy. Kết quả sau khi chuyển đổi là X có kích thước (5169, 3000), y có kích thước (5169,).

Bước 5: Xây dựng mô hình

- **SVC (Support Vector Classification):** Sử dụng lớp SVC() với tham số kernel = ‘linear’, class_weight = ‘balanced’.
- **MultinomialNB:** Sử dụng lớp MultinomialNB() để khởi tạo mô hình Naive Bayes đa thức.
- **LogisticRegression:** Sử dụng lớp LogisticRegression() để khởi tạo một mô hình Hồi quy Logistic với random_state = 42, class_weight = ‘balanced’.
- **RandomForestClassifier:** Sử dụng lớp RandomForestClassifier() với 50 cây quyết định, class_weight = ‘balanced’ và random_state bằng 42.

Bước 6: Đánh giá mô hình.

Để đánh giá mô hình chúng tôi sử dụng phương pháp kiểm tra chéo (cross-validation) với 5 folds để đánh giá hiệu suất của các mô hình học máy. Việc sử dụng kỹ thuật này giúp đánh giá hiệu suất mô hình một cách ổn định, có tính tổng quát hơn và giảm thiểu nguy cơ quá khít (overfitting). Phương pháp này sử dụng hiệu quả dữ liệu, đặc biệt hữu ích khi dữ liệu bị hạn chế hoặc mất cân đối. Kết quả thu được sau khi đánh giá mô hình:

- Mô hình SVC (Bảng 4.19):

Bảng 4.19. Kết quả đánh giá mô hình SVC

Class	Precision	Recall	F1-score	Support
0	0.98	0.99	0.99	4516
1	0.93	0.87	0.9	653
Accuracy			0.98	5169
Macro avg	0.96	0.93	0.94	5169
Weighted avg	0.98	0.98	0.98	5169

- Mô hình MultinomialNB (Bảng 4.20):

Bảng 4.20. Kết quả đánh giá mô hình MultinomialNB

Class	Precision	Recall	F1-score	Support
0	0.97	1	0.99	4516
1	0.99	0.81	0.89	653
Accuracy			0.98	5169
Macro avg	0.98	0.91	0.94	5169
Weighted avg	0.98	0.98	0.97	5169

- Mô hình Logistic Regression (Bảng 4.21):

Bảng 4.21. Kết quả đánh giá mô hình Logistic Regression

Class	Precision	Recall	F1-score	Support
0	0.98	0.99	0.99	4516
1	0.91	0.89	0.9	653
Accuracy			0.98	5169
Macro avg	0.95	0.94	0.94	5169
Weighted avg	0.98	0.98	0.98	5169

- Mô hình RandomForest (Bảng 4.22):

Bảng 4.22. Kết quả đánh giá mô hình RandomForest

Class	Precision	Recall	F1-score	Support
0	0.98	1	0.99	4516
1	0.98	0.82	0.9	653
Accuracy			0.98	5169
Macro avg	0.98	0.91	0.94	5169
Weighted avg	0.98	0.98	0.97	5169

Bảng 4.23. Bảng so sánh hiệu suất các mô hình

Mô hình	Thời gian chạy (giây)	Accuracy	Precision (\pm độ lệch)	Recall (\pm độ lệch)	F1 Score (\pm độ lệch)
SVC	233.32	0.9760 \pm 0.0015	0.9320 \pm 0.0158	0.8745 \pm 0.0212	0.9020 \pm 0.0070

MultinomialNB	1.99	0.9752 ± 0.0050	0.9908 ± 0.0142	0.8117 ± 0.0367	0.8919 ± 0.0231
Logistic Regression	14.09	0.9754 ± 0.0020	0.9156 ± 0.0226	0.8882 ± 0.0141	0.9014 ± 0.0072
RandomForestClassifier	101.63	0.9758 ± 0.0032	0.9817 ± 0.0081	0.8239 ± 0.0218	0.8958 ± 0.0144

Trong các mô hình đã thử nghiệm, cả bốn mô hình (SVC, MultinomialNB, Logistic Regression, và Random Forest) đều đạt độ chính xác cao, khoảng 97%. SVC có độ chính xác và F1 score tốt, nhưng mất thời gian huấn luyện lâu nhất. MultinomialNB tuy có thời gian huấn luyện nhanh nhưng có recall thấp hơn đối với lớp thư rác. Logistic Regression đạt cân bằng tốt giữa precision và recall, với thời gian huấn luyện tương đối ngắn. Random Forest có độ chính xác cao nhưng recall thấp hơn cho lớp thư rác so với SVC và Logistic Regression. Nhìn chung, Logistic Regression có hiệu suất tổng thể tốt nhất.

4.2.4. Bài toán 4: Phân tích và dự đoán giá tiền ảo Bitcoin

Bước 1: Thu thập dữ liệu

Bước 2: Xử lý và làm sạch dữ liệu

- Trích chọn đặc trưng những đặc trưng cần thiết cho yêu cầu bài toán

Import các thư viện cần thiết và đọc dữ liệu vào (Hình 4.35)

	slug	symbol	name	date	ranknow	open	high	low	close	volume	market	close_ratio	spread
0	bitcoin	BTC	Bitcoin	2013-04-28	1	135.30	135.98	132.10	134.21	0	1500520000	0.5438	3.88
1	bitcoin	BTC	Bitcoin	2013-04-29	1	134.44	147.49	134.00	144.54	0	1491160000	0.7813	13.49
2	bitcoin	BTC	Bitcoin	2013-04-30	1	144.00	146.93	134.05	139.00	0	1597780000	0.3843	12.88
3	bitcoin	BTC	Bitcoin	2013-05-01	1	139.00	139.89	107.72	116.99	0	1542820000	0.2882	32.17
4	bitcoin	BTC	Bitcoin	2013-05-02	1	116.38	125.60	92.28	105.21	0	1292190000	0.3881	33.32

Hình 4.35. Năm dòng dữ liệu đầu tiên của tập dữ liệu ban đầu

Trong tập dữ liệu có rất nhiều loại tiền ảo khác nhau, ta chỉ giữ lại các dòng dữ liệu liên quan đến Bitcoin từ tập dữ liệu ban đầu. Sau đó, chuyển đổi cột ‘date’ thành kiểu dữ liệu datetime bằng phương thức ‘pd.to_datetime’ và đặt cột này làm chỉ mục (index) của DataFrame. Tiếp theo, chọn ra các cột quan trọng bao gồm ‘close’, ‘open’, ‘high’, và ‘low’ để giữ lại trong DataFrame btc. Kết quả (Hình 4.36):

	close	open	high	low
date				
2013-04-28	134.21	135.30	135.98	132.10
2013-04-29	144.54	134.44	147.49	134.00
2013-04-30	139.00	144.00	146.93	134.05
2013-05-01	116.99	139.00	139.89	107.72
2013-05-02	105.21	116.38	125.60	92.28

Hình 4.36. Năm dòng đầu của dữ liệu btc sau khi xử lý

Thông tin (`data.info()`) về số lượng hàng, cột, tên cột, số lượng giá trị không null, kiểu dữ liệu được thể hiện trong Bảng 4.24. Thông kê tóm tắt của các cột dữ liệu số (`data.describe()`) được thể hiện trong Hình 4.37.

Bảng 4.24. Bảng thông tin về tập dữ liệu

DatetimeIndex: 1745 entries, 2013-04-28 to 2018-02-05

Data columns (total 5 columns):

	Column	Non-Null Count	Dtype
0	close	1745 non-null	float64
1	open	24917 non-null	float64
2	high	24916 non-null	float64
3	low	24926 non-null	float64

dtypes: float64(4)
memory usage: 68.2 KB

	close	open	high	low
count	1745.000000	1745.000000	1745.000000	1745.000000
mean	1417.806115	1414.091759	1467.029616	1354.985954
std	2877.000892	2874.565707	3010.144643	2704.371429
min	68.430000	68.500000	74.560000	65.530000
25%	274.020000	273.500000	278.340000	267.090000
50%	477.760000	477.760000	489.830000	465.130000
75%	842.720000	841.470000	870.960000	820.270000
max	19497.400000	19475.800000	20089.000000	18974.100000

Hình 4.37. Tóm tắt thống kê cho tập dữ liệu

- Kiểm tra dữ liệu null

Kết quả cho thấy tập dữ liệu không có giá trị null (Hình 4.38).

```
btc.isnull().sum()
```

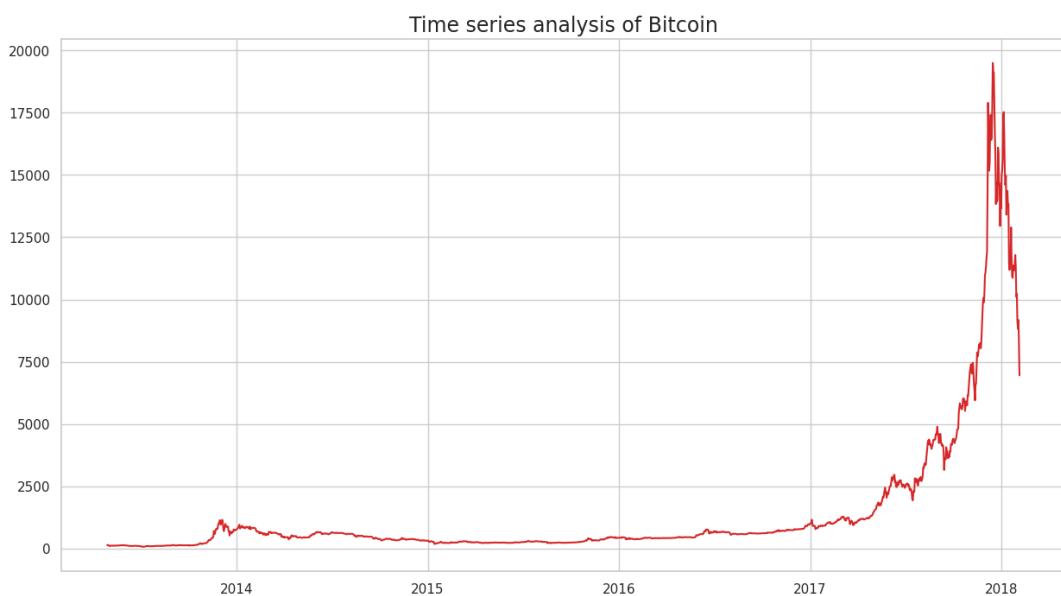
```
0  
close 0  
open 0  
high 0  
low 0
```

```
dtype: int64
```

Hình 4.38. Kết quả kiểm tra dữ liệu null

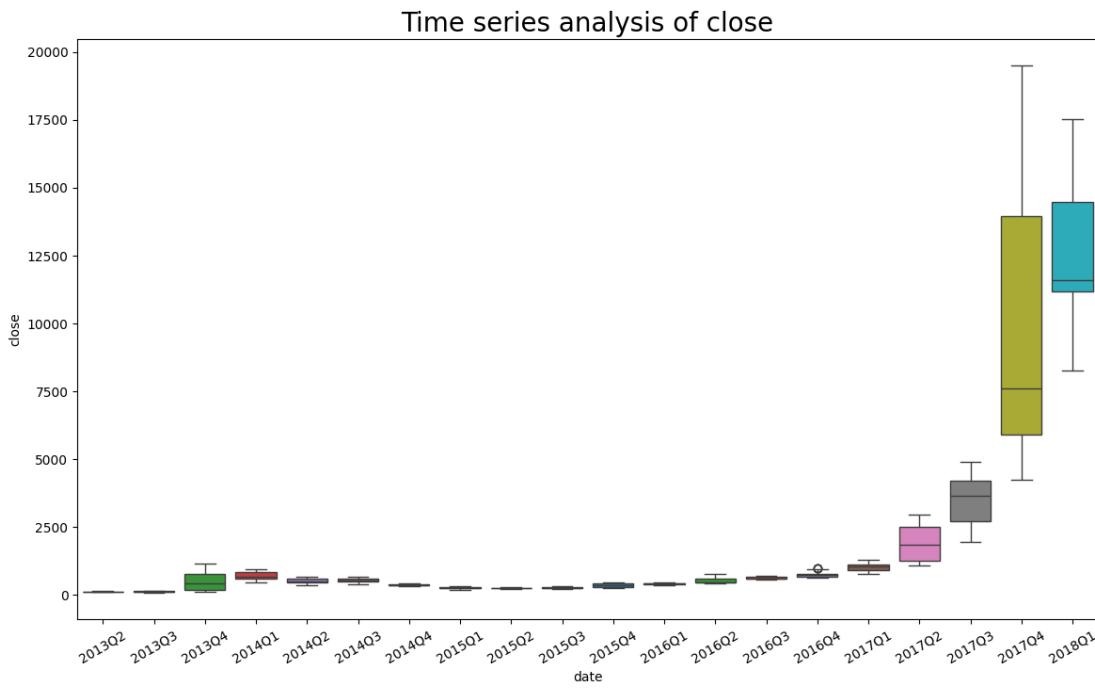
Phân tích dữ liệu thông qua:

- **Biểu đồ đường, biểu đồ hộp**



Hình 4.39. Biểu đồ đường của giá đóng Bitcoin

Thông qua biểu đồ đường (Hình 4.39), ta thấy rằng trong khoảng thời gian từ 28/4/2013 đến 5/2/2018 giá Bitcoin biến đổi chủ đạo theo xu hướng tăng. Có những biến đổi nổi bật vào các thời điểm: Vào cuối năm 2013, giá Bitcoin tăng mạnh. Cuối tháng 2/2014, giá Bitcoin bắt đầu giảm mạnh cho đến các tháng về sau. Từ đầu năm 2017, giá Bitcoin bắt đầu tăng mạnh. Từ những ngày cuối cùng năm 2017, đầu năm 2018 trở về sau, giá Bitcoin giảm mạnh.

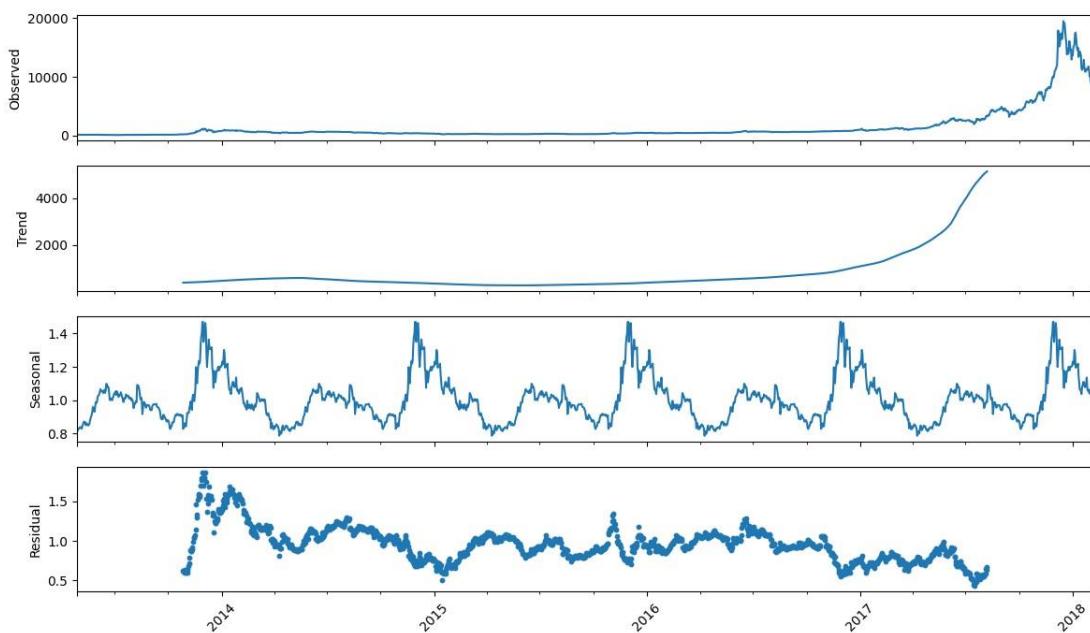


Hình 4.40. Biểu đồ hộp thể hiện giá đóng của Bitcoin

Qua các quý thì xu hướng chung của giá đóng Bitcoin là tăng theo thời gian. Biểu đồ hộp ở Hình 4.40 cũng cho thấy sự khác biệt lớn về sự biến động giữa các quý. Một số quý có biến động rất lớn (ví dụ: Q4 2017, Q1 2018), trong khi một số quý có biến động rất ít (ví dụ: Q2 2015, Q1 2016).

- **Phát hiện xu hướng, mùa vụ, nhiễu**

Vì chuỗi có xu hướng tăng dần nên trong phân phát hiện xu hướng, mùa vụ, nhiễu sẽ sử dụng mô hình nhân và với chu kỳ là 365 ngày.



Hình 4.41. Biểu đồ phân tích mùa vụ theo giá đóng Bitcoin

Dựa trên phân tích biểu đồ phân tích mùa vụ theo giá đóng Bitcoin (Hình 4.41), có thể nhận thấy sự hiện diện rõ ràng của yếu tố mùa vụ trong tập dữ liệu. Biểu đồ cho thấy mùa vụ tăng giảm giá theo từng tháng trong năm. Mùa tăng giá thường bắt đầu từ đầu tháng 11, kéo dài đến đầu tháng 12 và chậm định. Sau đó, giá Bitcoin bắt đầu giảm dần trong các tháng đầu năm và thấp nhất vào tháng 4. Vào tháng 6,7, giá Bitcoin có xu hướng tăng nhẹ, tuy nhiên, giá Bitcoin lại giảm nhẹ trong tháng 9.

Bước 3: Kiểm tra tính dừng của dữ liệu

Sử dụng hàm adfuller để thực hiện kiểm định ADF, cột giá đóng cửa của Bitcoin là chuỗi thời gian cần kiểm định, autolag='AIC' chỉ định rằng độ trễ (lag) tối ưu sẽ được chọn dựa trên tiêu chí AIC (Akaike Information Criterion).

```
ADF Test Statistic: -4.193708
p-value: 0.000675
Critical Values:
{'1%': -3.4341598265624627, '5%': -2.863222825253062, '10%': -2.567665889522738}
Reject Null Hypothesis - Time Series is Stationary
```

Hình 4.42. Kết quả kiểm định tính dừng

Bởi vì giá trị thống kê kiểm tra ADF (-4.193708) nhỏ hơn giá trị tối hạn tại mức ý nghĩa 5% (-2.863222825253062), chúng ta bác bỏ giả thuyết không (null hypothesis) rằng chuỗi thời gian không dừng (non-stationary). Điều này có nghĩa chuỗi thời gian trên là dừng.

Bước 4: Moving Average

Thực hiện tạo cột mới 'MA10', tính trung bình động (Moving Average) với cửa sổ 10 ngày cho cột 'close' bằng cách sử dụng 'rolling()' với tham số window=10 để tạo ra một cửa sổ trượt gồm 10 giá trị liên tiếp trên cột 'close', sau đó áp dụng hàm 'mean()' để tính giá trị trung bình của mỗi cửa sổ. Kết quả là một chuỗi giá trị MA10, trong đó mỗi giá trị đại diện cho trung bình của 10 giá đóng cửa gần nhất, giúp làm mịn dữ liệu và xác định xu hướng ngắn hạn của giá Bitcoin. 9 giá trị đầu tiên trong cột MA10 là NaN do không đủ dữ liệu để tính trung bình cho cửa sổ 10 ngày. Sau đó ta sử dụng dropna() để loại bỏ các dòng chứa giá trị NaN. Kết quả sau khi thực hiện Moving Average (Hình 4.43).

	close	open	high	low	MA10
date					
2013-05-07	111.50	112.25	113.44	97.70	118.991
2013-05-08	113.57	109.60	115.78	109.60	116.927
2013-05-09	112.67	113.20	113.46	109.26	113.740
2013-05-10	117.20	112.80	122.00	111.55	111.560
2013-05-11	115.24	117.70	118.68	113.01	111.385

Hình 4.43. Kết quả sau khi thực hiện Moving Average

Bước 5: Feature Scaling

Thực hiện chuẩn hóa chuỗi thời gian giá đóng cửa của Bitcoin bằng cách sử dụng MinMaxScaler từ thư viện `sklearn.preprocessing`. Đầu tiên, đối tượng `MinMaxScaler` được khởi tạo để chuẩn hóa dữ liệu đầu vào cho mô hình với các cột được chọn là ‘close’ và ‘MA10’. Kết quả (Hình 4.44) là một mảng dữ liệu chuẩn hóa ‘`data_scaled`’, giúp tăng tốc quá trình huấn luyện và cải thiện hiệu suất của các mô hình học máy.

```
array([[0.00221679, 0.00227007],
       [0.00232333, 0.00215306],
       [0.00227701, 0.00197239],
       ...,
       [0.46870627, 0.58770381],
       [0.42249177, 0.57079641],
       [0.35446243, 0.54689484]])
```

Hình 4.44. Kết quả sau khi scaling dữ liệu

Bước 6: Chuẩn bị dữ liệu đầu vào cho mô hình LSTM

Thực hiện định nghĩa hàm `create_sequences` để tạo ra các chuỗi dữ liệu đầu vào và đầu ra cho mô hình LSTM (Long Short-Term Memory). Hàm này nhận vào một tập dữ liệu và độ dài chuỗi mong muốn, sau đó tạo ra các chuỗi con liên tiếp từ dữ liệu gốc. Mỗi chuỗi con X có độ dài `seq_length` và giá trị y tương ứng là giá trị tiếp theo sau chuỗi đó. Việc này giúp mô hình LSTM học được mối quan hệ giữa các điểm dữ liệu liên tiếp trong chuỗi thời gian. Bằng cách sử dụng chuỗi dữ liệu này, mô hình có thể dự đoán giá trị tiếp theo dựa trên các giá trị trước đó. Việc tạo chuỗi dữ liệu là cần thiết vì LSTM được thiết kế để xử lý dữ liệu tuần tự và cần được huấn luyện trên các chuỗi có độ dài cố định.

```
def create_sequences(data, seq_length):
    X, y = [], []
    for i in range(len(data) - seq_length):
        X.append(data[i:(i + seq_length), :])
        y.append(data[i + seq_length, 0])
```

```
return np.array(X), np.array(y)
```

Trong bài toán này, chúng tôi thực hiện việc lặp qua các độ dài chuỗi khác nhau [7, 10, 14, 21, 30, 60, 90] để tìm ra độ dài chuỗi tối ưu nhất. Tiếp theo, thực hiện chia tập dữ liệu theo tỷ lệ 80% cho tập train và 20% cho tập test. Sau khi duyệt qua các độ dài chuỗi khác nhau, kết quả (Bảng 4.25) cho thấy với độ dài chuỗi là 7 ngày thì mô hình đạt hiệu suất dự đoán tốt nhất.

Bảng 4.25. Bảng so sánh kết quả dự đoán mô hình với các độ dài chuỗi khác nhau

Sequence Length	MSE	RMSE	R2	MAE
7	535660.8	731.89	0.98	374.1
10	541885.6	736.13	0.98	381.69
14	707320	841.02	0.97	451.28
21	711434.1	843.47	0.97	471.87
30	689435.9	830.32	0.97	453.15
60	708779.6	841.89	0.97	453.44
90	626523.5	791.53	0.97	429.03

Bước 7: Xây dựng mô hình LSTM

Mô hình (Hình 4.45) được khởi tạo bằng cách sử dụng Sequential(), cho phép xây dựng mô hình lớp chồng lớp một cách tuần tự.

- Định nghĩa dữ liệu đầu vào (Input) có dạng (7, 1), nghĩa là mô hình sẽ nhận đầu vào là các chuỗi có chiều dài 7 và 2 đặc trưng.
- Layer 1: Lớp LSTM đầu tiên (LSTM) có 50 đơn vị (units=50) và được thiết lập để trả về toàn bộ chuỗi đầu ra (return_sequences=True). Tham số huấn luyện của lớp này là 10,600.
- Layer 2: Lớp Dropout (Dropout) với tỷ lệ 20% (rate=0.2) được thêm vào sau lớp LSTM đầu tiên để giảm thiểu việc quá khớp (overfitting) bằng cách ngẫu nhiên bỏ qua 20% các đơn vị trong quá trình huấn luyện.
- Layer 3: Lớp LSTM thứ hai có 50 đơn vị và được thiết lập để không trả về toàn bộ chuỗi đầu ra (return_sequences=False), nghĩa là nó chỉ trả về đầu ra cuối cùng của chuỗi, có dạng (50). Tham số huấn luyện của lớp này là 20,200.
- Layer 4: Lớp Dropout thứ hai với tỷ lệ 20% được thêm vào sau lớp LSTM thứ hai để tiếp tục giảm thiểu việc quá khớp.

- Layer 5: Lớp Dense đầu tiên (Dense) có 25 đơn vị. Lớp này là lớp fully connected layer, nghĩa là mỗi đơn vị trong lớp này kết nối với tất cả các đơn vị trong lớp trước đó. Tham số huấn luyện của lớp này là 1,275.
- Layer 6: Lớp Dense cuối cùng (Dense) có 1 đơn vị, chịu trách nhiệm dự đoán giá trị đầu ra cuối cùng (giá trị đóng cửa dự đoán). Tham số huấn luyện của lớp này là 26.

Tổng số tham số huấn luyện của mô hình là 32,101.

Model: "sequential"

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 7, 50)	10,600
dropout (Dropout)	(None, 7, 50)	0
lstm_1 (LSTM)	(None, 50)	20,200
dropout_1 (Dropout)	(None, 50)	0
dense (Dense)	(None, 25)	1,275
dense_1 (Dense)	(None, 1)	26

Total params: 32,101 (125.39 KB)
Trainable params: 32,101 (125.39 KB)
Non-trainable params: 0 (0.00 B)

Hình 4.45. Kiến trúc mô hình LSTM

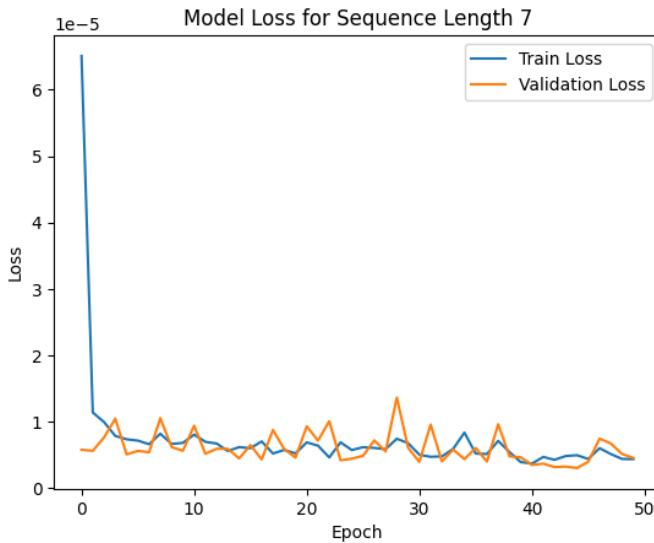
Sử dụng lớp ModelCheckpoint, một callback được tạo trong quá trình huấn luyện mô hình LSTM để lưu lại mô hình với tên 'best_model.h5.keras' khi chỉ số 'val_loss' (giá trị mất mát trên tập test) cải thiện. Tham số save_best_only=True đảm bảo chỉ lưu lại mô hình tốt nhất. Tham số mode='min' chỉ định rằng một giá trị 'val_loss' nhỏ hơn là tốt hơn. Callback này giúp lưu giữ phiên bản tốt nhất của mô hình trong suốt quá trình huấn luyện.

Mô hình sử dụng bộ tối ưu hóa là 'adam' và hàm mất mát là 'mean_squared_error'. Sau đó, mô hình được huấn luyện trên tập dữ liệu huấn luyện X_train và y_train, với tỷ lệ chia tập dữ liệu kiểm thử là 20% (validation_split=0.2). Quá trình huấn luyện được thực hiện trong 50 epoch với kích thước batch là 32. Callback checkpoint đảm bảo phiên bản tốt nhất của mô hình được lưu lại dựa trên val_loss.

Bước 8: Đánh giá mô hình.

Dựa vào đồ thị loss của mô hình có độ dài chuỗi là 7 ngày (Hình 4.46), ta thấy mô hình này thể hiện hiệu suất tốt và ổn định. Quá trình huấn luyện cho thấy sự hội tụ nhanh chóng trong những epoch đầu tiên. Điều đáng chú ý là không có dấu hiệu rõ ràng của

overfitting, vì đường validation loss không tăng đáng kể so với training loss trong suốt quá trình.



Hình 4.46. Đồ thị hàm loss của mô hình

- Đánh giá hiệu suất của mô hình

Để đánh giá hiệu suất của mô hình, các chỉ số đánh giá Mean Squared Error (MSE), Root Mean Squared Error (RMSE), R-squared (R2), và Mean Absolute Error (MAE) được tính toán dựa trên các giá trị dự đoán và thực tế của tập test (20%). Kết quả lần lượt là:

- **R2 (0.98):** Với giá trị R2 là 0.98, mô hình thể hiện khả năng giải thích rất tốt cho biến động của giá Bitcoin. Điều này có nghĩa là 98% sự biến thiên trong giá Bitcoin được giải thích bởi mô hình, cho thấy mức độ phù hợp cao giữa dự đoán và giá trị thực tế.
- **RMSE (705.14):** Giá trị RMSE là 705.14 cho biết sai số trung bình của mô hình khoảng 705 USD. Đối với một tài sản biến động như Bitcoin, đây là mức độ chính xác khá tốt.
- **MSE (497222.87):** Giá trị MSE là 497222.87, tương đương với bình phương của RMSE. Nó nhấn mạnh các sai số lớn và cho thấy mức độ phân tán của dự đoán.
- **MAE (360.03):** Giá trị MAE là 360.03 cho thấy sai số tuyệt đối trung bình thấp hơn RMSE, điều này cho rằng có một số điểm dự đoán có sai số lớn làm tăng RMSE.

Bước 9: Dự báo giá bitcoin trong ngày tiếp theo

Thực hiện dự báo giá Bitcoin cho 30 ngày tiếp theo dựa trên dữ liệu hiện tại và mô hình LSTM. Đầu tiên, lấy dãy dữ liệu cuối cùng (last_sequence) từ tập dữ liệu đã chuẩn hóa (data_scaled) làm điểm bắt đầu cho dự báo. Một danh sách trống (next_30_days) được tạo ra để lưu trữ các dự báo cho 30 ngày tiếp theo.

Trong vòng lặp 30 lần, mô hình dự đoán giá cho ngày tiếp theo (next_pred) dựa trên dãy dữ liệu hiện tại (last_sequence). Giá trị dự đoán được thêm vào danh sách next_30_days. Sau đó, dãy dữ liệu được cập nhật bằng cách xoay các phần tử của last_sequence sang trái một vị trí và gán giá trị dự đoán mới nhất vào vị trí cuối cùng.

Sau khi hoàn thành vòng lặp, danh sách next_30_days chứa các giá trị dự đoán cho 30 ngày tiếp theo. Các giá trị này được chuyển đổi về thang đo ban đầu.

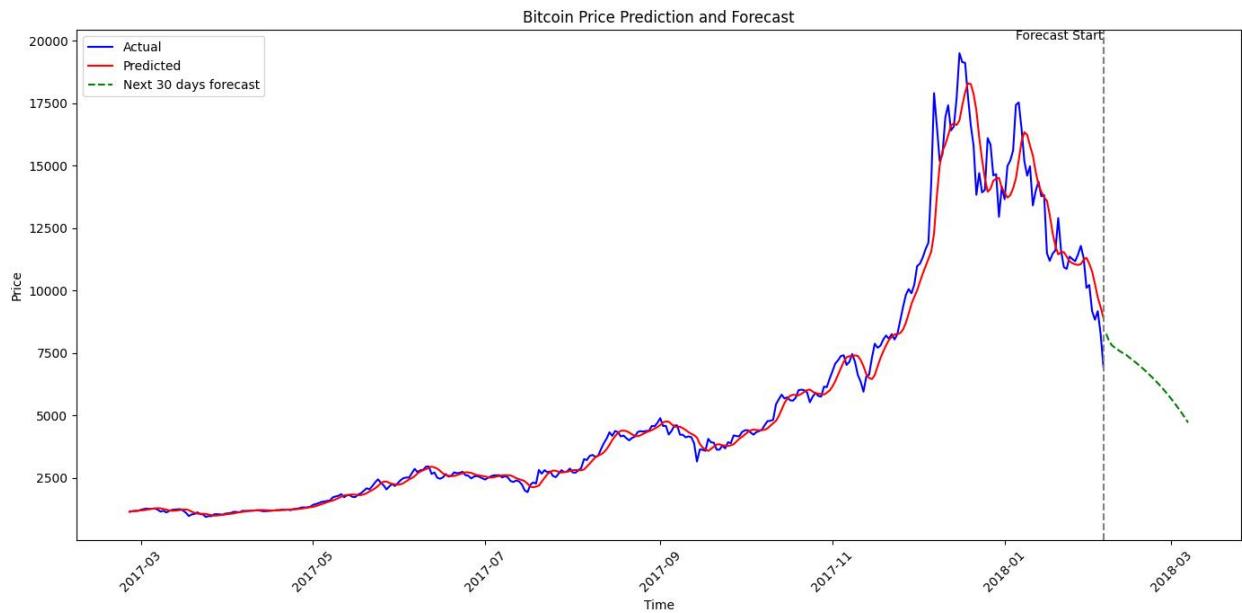
```
last_sequence = data_scaled[-seq_length:]
next_30_days = []
for _ in range(30):
    next_pred = best_model.predict(last_sequence.reshape(1, seq_length,
len(features)))
    next_30_days.append(next_pred[0])
    last_sequence = np.roll(last_sequence, -1, axis=0)
    last_sequence[-1] = np.hstack([next_pred[0], last_sequence[-2, 1:]])
next_30_days_prices =
scaler.inverse_transform(np.hstack([np.array(next_30_days).reshape(-1, 1),
np.zeros((30, len(features)-1))]))[:, 0]
```

Kết quả dự báo trong 30 ngày tiếp theo (Bảng 4.26):

Bảng 4.26. Kết quả dự báo trong 30 ngày tiếp theo

Date	Price	Date	Price	Date	Price
2018-02-06	\$8,266.48	2018-02-16	\$7,158.71	2018-02-26	\$6,076.07
2018-02-07	\$7,976.86	2018-02-17	\$7,067.34	2018-02-27	\$5,945.19
2018-02-08	\$7,810.10	2018-02-18	\$6,972.39	2018-02-28	\$5,809.61
2018-02-09	\$7,714.31	2018-02-19	\$6,873.91	2018-03-01	\$5,669.16
2018-02-10	\$7,637.53	2018-02-20	\$6,771.57	2018-03-02	\$5,523.66
2018-02-11	\$7,557.42	2018-02-21	\$6,665.31	2018-03-03	\$5,372.92
2018-02-12	\$7,491.80	2018-02-22	\$6,555.65	2018-03-04	\$5,216.77
2018-02-13	\$7,431.39	2018-02-23	\$6,442.07	2018-03-05	\$5,055.00
2018-02-14	\$7,335.06	2018-02-24	\$6,324.36	2018-03-06	\$4,887.43
2018-02-15	\$7,247.01	2018-02-25	\$6,202.41	2018-03-07	\$4,713.85

Vẽ biểu đồ so sánh kết quả dự đoán, thực tế của tập test (20%) và dự báo 30 ngày tiếp theo (Hình 4.47).



Hình 4.47. Biểu đồ so sánh kết quả dự đoán, giá trị thực tế và dự báo 30 ngày tiếp theo cho giá Bitcoin

Đường màu xanh biểu thị giá trị thực tế của giá Bitcoin. Đường màu đỏ biểu thị giá trị dự đoán của mô hình. Đường màu xanh lá cây (nét đứt) biểu thị dự báo giá Bitcoin cho 30 ngày tiếp theo. Dự báo cho 30 ngày tiếp theo cho thấy xu hướng giảm, điều này có thể hữu ích cho việc ra quyết định của các nhà đầu tư. Tuy nhiên, luôn cần lưu ý rằng dự báo dựa trên mô hình có thể không phản ánh chính xác mọi yếu tố thực tế và cần được kiểm chứng thêm.

4.3. Tổng kết chương.

Chương 5 đã cung cấp các kết quả cụ thể từ quá trình thực nghiệm, minh chứng cho hiệu quả của các kỹ thuật phân tích và tiền xử lý dữ liệu đã được trình bày trong các chương trước. Những kết quả này không chỉ chứng minh tính khả thi của các phương pháp mà còn cung cấp thông tin hữu ích cho các nghiên cứu và ứng dụng tiếp theo.

KẾT LUẬN

Kết luận

Trong nghiên cứu này, chúng em đã tập trung vào các kỹ thuật phân tích và tiền xử lý dữ liệu hiện đại trong các bài toán máy học và khai thác dữ liệu. Sau khi tìm hiểu và phân tích chúng em đã triển khai các mô hình và công cụ để xử lý dữ liệu hiệu quả, đồng thời đánh giá hiệu suất và tính khả thi của các kỹ thuật đề xuất thông qua thử nghiệm trên dữ liệu thực tế.

Các kết quả chính của nghiên cứu bao gồm:

1. Phân tích và nghiên cứu các vấn đề phổ biến trong dữ liệu như dữ liệu bị thiếu, dữ liệu trùng, dữ liệu nhiễu, dữ liệu không nhất quán.
2. Áp dụng các kỹ thuật tiền xử lý dữ liệu như xử lý dữ liệu bị thiếu, dữ liệu trùng, dữ liệu ngoại lai, mã hóa dữ liệu, chuẩn hóa dữ liệu, xử lý mất cân bằng dữ liệu, trích chọn đặc trưng, giảm chiều dữ liệu,...
3. Thiết kế và triển khai các thuật toán mô hình và công cụ để phân tích và xử lý dữ liệu hiệu quả.
4. Đánh giá hiệu suất và tính khả thi của các kỹ thuật đề xuất thông qua thử nghiệm trên dữ liệu thực tế.

Hướng phát triển tương lai.

Mở rộng phạm vi dữ liệu: Ngoài dữ liệu định lượng và định tính, nghiên cứu có thể mở rộng sang các loại dữ liệu khác như văn bản, hình ảnh, âm thanh và video. Điều này sẽ giúp đánh giá hiệu suất của các kỹ thuật tiền xử lý và mô hình học máy trong các bối cảnh dữ liệu phức tạp.

Tối ưu hóa các thuật toán: Nghiên cứu sâu hơn về các thuật toán tối ưu hóa để cải thiện hiệu suất và độ chính xác của các mô hình, đặc biệt là trong các tình huống dữ liệu lớn và phức tạp.

Ứng dụng thực tiễn: Áp dụng các kỹ thuật và mô hình đã nghiên cứu vào các bài toán thực tiễn trong các lĩnh vực như y tế, tài chính, marketing, và các ngành công nghiệp khác. Điều này sẽ giúp kiểm chứng tính khả thi và hiệu quả của các phương pháp trong môi trường thực tế.

TÀI LIỆU THAM KHẢO

- [1] K. P. M., "Các phương pháp imputation đơn giản cho missing values.," 21 7 2020. [Online]. Available: <https://bigdatauni.com/tin-tuc/cac-phuong-phap-imputation-don-gian-cho-missing-values.html>.
- [2] K. P.M, "Data cleaning – làm sạch dữ liệu: Xử lý missing values (P1)," 5 7 2020. [Online]. Available: <https://bigdatauni.com/tin-tuc/data-cleaning-lam-sach-du-lieu-xu-ly-missing-values-p1.html>. [Accessed 2024].
- [3] A. Publishing, Data Preprocessing with Python for Absolute Beginners, 2020.
- [4] A. & A. A. Arif, "Feature Selection Techniques in Machine Learning [2023 Edition] - Dataaspirant.," 6 12 2023. [Online]. Available: <https://www.javatpoint.com/feature-selection-techniques-in-machine-learning>. [Accessed 2024].
- [5] "Feature selection techniques in Machine Learning - JavatPoint.," [Online]. Available: <https://www.javatpoint.com/feature-selection-techniques-in-machine-learning>. [Accessed 6 2024].
- [6] GeeksforGeeks, "Feature selection techniques in machine learning," 19 3 2024. [Online]. Available: <https://www.geeksforgeeks.org/feature-selection-techniques-in-machine-learning/>. [Accessed 2024].
- [7] "Xử lý các giá trị trùng lặp từ bộ dữ liệu trong python.," [Online]. Available: <https://vi.linux-console.net/?p=27705>. [Accessed 6 2024].
- [8] N. T. Binh, "BÁO CÁO TỔNG KẾT ĐỀ TÀI KHOA HỌC VÀ CÔNG NGHỆ CẤP CƠ SỞ," 2021.
- [9] T. N. Huy, "ML From Scratch: Thuật toán giảm chiều dữ liệu PCA - Viblo," 2024. [Online]. Available: <https://viblo.asia/p/ml-from-scratch-thuat-toan-giam-chieu-du-lieu-pca-7ymJXKMa4kq>.
- [10] A. OLULEYE, Exploratory Data Analysis with Python Cookbook, 2023.
- [11] N. M., "Cách tính và ý nghĩa ma trận hiệp phương sai (covariance matrix)," 6 11 2019. [Online]. Available: <https://minhng.info/toan-hoc/ma-tran-hiep-phuong-sai-covariance-matrix.html>. [Accessed 8 2024].
- [12] N. T., "Trị riêng và vector riêng của ma trận. Minh Nguyen," 6 10 2017. [Online]. Available: <https://minhng.info/toan-hoc/ma-tran-tri-rieng-vector-rieng.html>. [Accessed 8 2024].
- [13] P. D. Khanh, "Phương pháp phân tích suy biến — Deep AI KhanhBlog," [Online]. Available: https://phamdinhkhanh.github.io/deepai-book/ch_ml/PCA.html. [Accessed 8 2024].
- [14] W. contributors, "Principal component analysis," 3 07 2024. [Online]. Available: https://en.wikipedia.org/wiki/Principal_component_analysis#Derivation_using_the_covariance_method. [Accessed 14 08 2024].
- [15] T. Vu, "Bài 29: Linear Discriminant Analysis," 30 6 2017. [Online]. Available: <https://machinelearningcoban.com/2017/06/30/lda/>. [Accessed 8 2024].

- [16] M. Huddar, "LDA Solved Example | Linear Discriminant Analysis | Fisher Discriminant Analysis by Mahesh Huddar," 19 1 2024. [Online]. Available: <https://www.youtube.com/watch?v=txgqfG4rfos>. [Accessed 8 2024].
- [17] GeeksforGeeks, "Linear Discriminant analysis in machine learning," 20 3 2024. [Online]. Available: <https://www.geeksforgeeks.org/ml-linear-discriminant-analysis/>. [Accessed 14 8 2024].
- [18] "Removing stop words with NLTK in Python," [Online]. Available: <https://www.geeksforgeeks.org/removing-stop-words-nltk-python/>.
- [19] P.-O.-S. T. i. NLP. [Online]. Available: <https://www.geeksforgeeks.org/nlp-part-of-speech-default-tagging/>.
- [20] "Xử lý ngôn ngữ tự nhiên (NLP) là gì? 10 thuật toán NLP bạn cần biết," [Online]. Available: <https://mctt.vn/xu-ly-ngon-ngu-tu-nhien-la-gi>.
- [21] A. Jain, "N-grams in NLP," [Online]. Available: <https://medium.com/@abhishhekjainindore24/n-grams-in-nlp-a7c05c1aff12>.
- [22] H. Suresh, "WordClouds: Basics of NLP," [Online]. Available: <https://medium.com/@harinisureshla/wordclouds-basics-of-nlp-5b60be226414>.
- [23] M. Dutta, "Bag-of-words vs TFIDF vectorization –A Hands-on Tutorial," [Online]. Available: <https://www.analyticsvidhya.com/blog/2021/07/bag-of-words-vs-tfidf-vectorization-a-hands-on-tutorial/>.
- [24] R. Madan, "TF-IDF/Term Frequency Technique: Easiest explanation for Text classification in NLP using Python (Chatbot training on words)," [Online]. Available: <https://medium.com/analytics-vidhya/tf-idf-term-frequency-technique-easiest-explanation-for-text-classification-in-nlp-with-code-8ca3912e58c3>.
- [25] G. f. Geeks, "Components of Time Series Data," [Online]. Available: <https://www.geeksforgeeks.org/components-of-time-series-data/>.
- [26] f. intuition, "Time-series analysis- calculating the seasonality and trend," [Online]. Available: <https://www.firstintuition.co.uk/fihub/time-series-analysis/>.
- [27] T. Marketers, "Phân tích dữ liệu chuỗi thời gian như thế nào?," [Online]. Available: <https://blog.tomorrowmarketers.org/time-series-analysis/>.
- [28] nandakishorereddy, "Regression Metrics," [Online]. Available: <https://www.geeksforgeeks.org/regression-metrics/>.
- [29] N. T. Thinh, "Đánh giá model trong Machine Learing," 29 05 2022. [Online]. Available: <https://viblo.asia/p/danh-gia-model-trong-machine-learing-RnB5pAq7KPG>. [Accessed 2024].
- [30] ChungToi, "9 Chỉ Tiêu đánh Giá độ Chính Xác Mô Hình Hồi Quy - Chạy định Lượng," 10 3 2023. [Online]. Available: <https://chaydinhluong.com/9-chi-tieu-danh-gia-do-chinh-xac-mo-hinh-hoi-quy/>. [Accessed 2024].
- [31] D. S. Sahin Ahmed, "ML Series 5: Understanding R-squared in Regression Analysis," 14 2 2024. [Online]. Available: <https://medium.com/@sahin.samia/understanding-r-squared-in-regression-analysis-2d8246a63dbb>.
- [32] S. K. Agrawal, "Metrics to Evaluate your Classification Model to take the Right Decisions," 5 6 2024. [Online]. Available:

TÀI LIỆU THAM KHẢO

- [https://www.analyticsvidhya.com/blog/2021/07/metrics-to-evaluate-your-classification-model-to-take-the-right-decisions/.](https://www.analyticsvidhya.com/blog/2021/07/metrics-to-evaluate-your-classification-model-to-take-the-right-decisions/)
- [33] P. Đ. Khanh, "Khoa học dữ liệu," 13 8 2020. [Online]. [Accessed https://phamdinhkhanh.github.io/2020/08/13/ModelMetric.html 2024].
- [34] HungTrinhIT, "FinalProject-Datascience," 2020. [Online]. Available: <https://github.com/HungTrinhIT/FinalProject-Datascience/blob/main/Chotot/rawdata.csv>. [Accessed 2024].
- [35] M. MUSTAFA, "Diabetes prediction dataset," 3 2024. [Online]. Available: <https://www.kaggle.com/datasets/iammustafatz/diabetes-prediction-dataset/code>.
- [36] swapnilvishwakarma7, "Comparing Randomized Search and Grid Search for Hyperparameter Estimation in Scikit Learn," 30 12 2022. [Online]. Available: <https://www.geeksforgeeks.org/comparing-randomized-search-and-grid-search-for-hyperparameter-estimation-in-scikit-learn/>.