BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM

ĐỒ ÁN MÔN HỌC

PHÂN TÍCH DỮ LIỆU Y TẾ ĐỂ DỰ ĐOÁN BỆNH TẬT (BỆNH TIM MẠCH)

Ngành: **KHOA HỌC DỮ LIỆU**

Môn học: **THU THẬP VÀ TIỀN XỬ LÝ DỮ LIỆU**

Giảng viên hướng dẫn : ThS. Hứa Thị Phượng Vân

Sinh viên thực hiện :

2286400908-Nguyễn Ngọc Quỳnh Anh

2286400043-Lê Hoàng Gia Vĩ

2286400045-Hoàng Nguyên Vũ

2386400039-Phạm Tường Phát

Lớp: 22DKHA1

TP. Hồ Chí Minh, 2025

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM

ĐỒ ÁN MÔN HỌC

PHÂN TÍCH DỮ LIỆU Y TẾ ĐỂ DỰ ĐOÁN BỆNH TẬT (BỆNH TIM MẠCH)

Ngành: **KHOA HỌC DỮ LIỆU**

Môn học: **THU THẬP VÀ TIỀN XỬ LÝ DỮ LIỆU**

Giảng viên hướng dẫn : ThS. Hứa Thị Phượng Vân

Sinh viên thực hiện :

2286400908-Nguyễn Ngọc Quỳnh Anh

2286400043-Lê Hoàng Gia Vĩ

2286400045-Hoàng Nguyên Vũ

2386400039-Phạm Tường Phát

Lớp: 22DKHA1

TP. Hồ Chí Minh, 2025

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

TPHCM, Ngày….. Tháng ….. Năm 2025

Giáo viên hướng dẫn

(Ký tên, đóng dấu)

LỜI CAM ĐOAN

Nhóm chúng tôi gồm Nguyễn Ngọc Quỳnh Anh, Lê Hoàng Gia Vĩ, Phạm Tường Phát và Hoàng Nguyên Vũ xin cam đoan rằng:

Tất cả thông tin và kết quả nghiên cứu trong bài báo cáo này đều trung thực và khách quan, được thu thập từ các nguồn đáng tin cậy, chính thống. Chúng tôi đã phân tích kỹ lưỡng các tài liệu và đảm bảo rằng mọi dữ liệu hoặc ý kiến trích dẫn đều được ghi rõ ràng nguồn gốc, tuân thủ đúng quy định về trích dẫn học thuật.

Chúng tôi cam kết không có hành vi sao chép hoặc sử dụng trái phép bất kỳ thông tin nào từ các nguồn khác. Bài báo cáo này là sản phẩm nghiên cứu độc lập của nhóm, chưa từng được công bố trước đây và tuân thủ đầy đủ các quy định của môn học. Chúng tôi sử dụng công cụ nghiên cứu một cách hợp lý và chính xác, đảm bảo tính khoa học và đạo đức trong quá trình thực hiện.

Nhóm chúng tôi hy vọng rằng bài báo cáo sẽ cung cấp cái nhìn sâu sắc về chủ đề "Phân tích dữ liệu y tế để dự đoán bệnh tật (bệnh tim mạch)" và đóng góp tích cực vào phát triển của lĩnh vực này.

TPHCM, Ngày…… Tháng …… Năm 2025

Sinh viên

Nguyễn Ngọc Quỳnh Anh

Lê Hoàng Gia Vĩ

Phạm Tường Phát

Hoàng Nguyên Vũ

DANH MỤC CÁC KÝ HIỆU, TỪ VIẾT TẮT VÀ TỪ KHOÁ

Missing Data	Khuyết thiếu dữ liệu
Outlier	Dữ liệu ngoại lai
Duplicate Data	Dữ liệu trùng lặp
Min-Max Normalization	Chuẩn hóa Min-Max
Z-Score Standardization	Chuẩn hóa Z-Score
Robust Scaling	Phương pháp chuẩn hóa mạnh mẽ
Logarithmic Transformation	Biến đổi logarithmic
Label Encoding	Mã hóa nhãn
One-hot Encoding	Mã hóa one-hot
OdinalEncode	Mã hóa thứ tự
Logistic Regression	Hồi quy logistic
Random Forest	Mô hình ensemble sử dụng nhiều cây quyết định

MỤC LỤC

DANH MỤC BẢNG

DANH MỤC HÌNH ẢNH

CHƯƠNG 1: TỔNG QUAN

1.1 Giới thiệu đề tài

Trong những năm gần đây, các bệnh mãn tính, đặc biệt là bệnh tim, đã trở thành một trong những nguyên nhân hàng đầu gây tử vong trên toàn thế giới. Sự phức tạp trong nguyên nhân và tiến trình phát triển của bệnh tim khiến việc chẩn đoán sớm trở thành một thách thức lớn đối với ngành y tế. Điều này không chỉ ảnh hưởng nghiêm trọng đến sức khỏe và chất lượng cuộc sống của người bệnh mà còn tạo áp lực đáng kể lên hệ thống chăm sóc sức khỏe toàn cầu.

Sự phát triển mạnh mẽ của công nghệ dữ liệu và trí tuệ nhân tạo đã mang lại những giải pháp tiềm năng trong việc khai thác dữ liệu y tế, từ đó nâng cao khả năng dự đoán và quản lý bệnh tật. Đề tài "Phân tích dữ liệu y tế để dự đoán bệnh tật (bệnh tim mạch)" tập trung vào việc thu thập và tiền xử lý dữ liệu y tế, nhằm tối ưu hóa độ chính xác trong dự đoán bệnh.

1.2 Nhiệm vụ của đề tài

1.2.1 Tính cấp thiết của đề tài

Bệnh tim là một vấn đề y tế nghiêm trọng với tỷ lệ mắc và tử vong ngày càng tăng. Việc chẩn đoán sớm và quản lý hiệu quả bệnh tim có thể giảm thiểu các biến chứng nguy hiểm và tối ưu hóa nguồn lực y tế. Tuy nhiên, việc dự đoán bệnh tim thường gặp khó khăn do sự phức tạp của các yếu tố nguy cơ. Ứng dụng khoa học dữ liệu trong phân tích và dự đoán bệnh tim không chỉ cấp thiết mà còn mang tính chiến lược trong việc nâng cao chất lượng chăm sóc sức khỏe.

1.2.2 Ý nghĩa khoa học và thực tiễn của đề tài

Ý nghĩa khoa học: Đề tài đóng góp vào lĩnh vực khoa học dữ liệu y tế thông qua việc áp dụng các phương pháp thu thập, tiền xử lý và phân tích

dữ liệu hiện đại, từ đó xây dựng mô hình dự đoán chính xác nguy cơ bệnh tim.

Ý nghĩa thực tiễn: Kết quả nghiên cứu có thể được áp dụng vào các hệ thống hỗ trợ quyết định lâm sàng, giúp bác sĩ chẩn đoán và quản lý bệnh tim hiệu quả hơn. Ngoài ra, đề tài còn góp phần nâng cao nhận thức cộng đồng về việc sử dụng dữ liệu y tế để bảo vệ sức khỏe.

1.3 Mục tiêu của đề tài

1.3.1 Mục tiêu tổng quát

Đề tài "Phân tích dữ liệu y tế để dự đoán bệnh tật (bệnh Tim mạch)" nhằm thu thập, xử lý và phân tích dữ liệu y tế từ nhiều nguồn khác nhau liên quan đến bệnh tim mạch, từ đó ứng dụng các phương pháp khoa học dữ liệu để xây dựng mô hình dự đoán nguy cơ mắc bệnh. Kết quả của đề tài góp phần hỗ trợ chẩn đoán sớm và quản lý hiệu quả bệnh tim mạch, đồng thời mở rộng tiềm năng ứng dụng công nghệ trong lĩnh vực chăm sóc sức khỏe.

1.3.2 Mục tiêu cụ thể

Đề tài tập trung vào việc nghiên cứu và phân tích cấu trúc dữ liệu y tế nhằm xác định các thông tin quan trọng liên quan đến bệnh tim mạch. Cụ thể, các chỉ số sinh học như huyết áp, cholesterol, nhịp tim, cùng với lịch sử bệnh lý và các yếu tố nguy cơ khác sẽ được phân tích để hiểu rõ mối quan hệ giữa chúng và nguy cơ mắc bệnh tim mạch. Các yếu tố này không chỉ bao gồm các chỉ số sinh học mà còn liên quan đến các thói quen sống, di truyền và môi trường sống của bệnh nhân. Đồng thời, đề tài cũng xây dựng một quy trình thu thập và tiền xử lý dữ liệu y tế từ nhiều nguồn đáng tin cậy, như hồ sơ bệnh án, cơ sở dữ liệu y tế công cộng, và các chỉ số sức khỏe cá nhân. Quá trình tiền xử lý này sẽ đảm bảo tính đầy đủ, chính xác và nhất quán của dữ liệu, bao gồm việc làm sạch dữ liệu, xử lý các giá trị

thiếu, chuẩn hóa và mã hóa các biến để phù hợp với các phương pháp phân tích dữ liệu tiếp theo. Điều này sẽ tạo ra một nền tảng dữ liệu đáng tin cậy, có chất lượng cao, phục vụ cho các bước phân tích và dự đoán nguy cơ mắc bệnh tim mạch, từ đó giúp nâng cao hiệu quả chẩn đoán và quản lý bệnh.

1.4 Đối tượng và phạm vi

1.4.1 Đối tượng

Đối tượng của đề tài bao gồm bộ dữ liệu y tế chứa thông tin liên quan đến bệnh tim mạch, bao gồm các chỉ số sức khỏe (huyết áp, cholesterol, nhịp tim, v.v.), các yếu tố nguy cơ (tuổi tác, giới tính, thói quen sinh hoạt, tiền sử bệnh lý), và kết quả chẩn đoán (có hoặc không mắc bệnh tim). Ngoài ra, đối tượng còn bao gồm các thuật toán và công cụ khoa học dữ liệu phục vụ cho việc phân tích và dự đoán bệnh tim mạch.

1.4.2 Phạm vi

Đề tài tập trung vào việc thu thập, làm sạch, chuẩn hóa và phân tích dữ liệu y tế để xây dựng mô hình dự đoán bệnh tim mạch. Phạm vi dữ liệu sẽ bao gồm các thông tin y tế từ các nguồn uy tín như bệnh viện, cơ sở y tế, và các nghiên cứu công bố, có liên quan đến bệnh tim và các yếu tố nguy cơ như huyết áp, cholesterol, nhịp tim, tiền sử bệnh lý, thói quen sống, và di truyền. Phạm vi ứng dụng của đề tài chỉ giới hạn trong việc phân tích và dự đoán bệnh tim mạch, không mở rộng ra các bệnh lý khác.

1.5 Những đóng góp nghiên cứu của đề tài

1.5.1 Trong lĩnh vực học thuật

Đề tài sẽ đóng góp vào việc phát triển các phương pháp phân tích dữ liệu y tế, đặc biệt trong việc ứng dụng học máy để dự đoán và chẩn đoán bệnh tim mạch. Qua việc xây dựng và đánh giá các mô hình dự đoán, nghiên cứu này sẽ giúp cải thiện hiểu biết về các yếu tố nguy cơ liên quan đến

bệnh tim và mối quan hệ giữa các chỉ số sức khỏe với khả năng mắc bệnh. Các kết quả nghiên cứu có thể đóng góp vào cơ sở lý thuyết của các phương pháp học máy trong y tế, mở ra hướng nghiên cứu mới về ứng dụng trí tuệ nhân tạo (AI) và khoa học dữ liệu trong chẩn đoán bệnh lý. Hơn nữa, việc ứng dụng các thuật toán học máy vào dữ liệu y tế sẽ tạo ra một nền tảng để phát triển các mô hình dự đoán cho các bệnh lý khác trong tương lai.

1.5.2 Trong thực tiễn

Về mặt thực tiễn, đề tài sẽ có những đóng góp quan trọng trong việc cải thiện quá trình chẩn đoán và quản lý bệnh tim mạch. Các mô hình dự đoán được xây dựng sẽ giúp các bác sĩ và chuyên gia y tế có công cụ hỗ trợ chẩn đoán sớm, từ đó phát hiện nguy cơ bệnh tim mạch trước khi các triệu chứng lâm sàng xuất hiện. Điều này không chỉ giúp giảm thiểu chi phí điều trị mà còn nâng cao hiệu quả điều trị và cải thiện chất lượng cuộc sống cho bệnh nhân. Ngoài ra, kết quả của đề tài có thể được ứng dụng trong các hệ thống quản lý y tế và các chương trình chăm sóc sức khỏe cộng đồng, góp phần nâng cao khả năng dự đoán và phòng ngừa bệnh tim mạch trong các cộng đồng.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 Khuyết thiếu dữ liệu (Missing Data)

2.1.1 Định nghĩa

Tình trạng dữ liệu bị thiếu hoặc không đầy đủ trong một tập dữ liệu, còn gọi là **Missing Data**. Hiện tượng này xảy ra khi một hoặc nhiều giá trị trong tập dữ liệu không được ghi nhận, dẫn đến việc thiếu hụt thông tin ở các phần quan trọng. Nguyên nhân của dữ liệu bị thiếu có thể xuất phát từ nhiều yếu tố khác nhau, bao gồm lỗi kỹ thuật trong quá trình thu thập, lưu trữ, hoặc truyền tải dữ liệu, ...[1]

2.1.2 Các loại khuyết thiếu dữ liệu (Missing Data)

Missing Completely at Random (MCAR) Dữ liệu bị thiếu hoàn toàn ngẫu nhiên:

Như tên gọi của nó đã nói lên tất cả. Sự mất mát dữ liệu ở đây là hoàn toàn ngẫu nhiên, và không có bất kỳ một mối quan hệ hay sự liên quan nào giữa dữ liệu và bất kì dữ liệu nào, missing hoặc dữ liệu quan sát.

Ở ví dụ dưới đây chúng ta sẽ không hề tìm thấy một mối quan hệ nào giữa giá trị bị thiếu và giá trị được giữ nguyên.[2]

Complete data	
Age	IQ score
25	133
26	121
29	91
30	105
30	110
31	98
44	118
46	93
48	141
51	104
51	116
54	97

Incomplete data	
Age	IQ score
25	
26	121
29	91
30	
30	110
31	
44	118
46	93
48	
51	
51	116
54	

Hình 2.1: Ví dụ MCAR

Missing at Random (MAR) Dữ liệu khuyết ngẫu nhiên:

Sự mất mát dữ liệu ở đây là ngẫu nhiên, tuy nhiên vẫn có mối quan hệ hệ thống giữa dữ liệu bị mất và dữ liệu được quan sát. Ví dụ ở hình 2.2 những người trẻ tuổi bị khuyết dữ liệu về IQ, có nghĩa là có một mối quan hệ hệ thống giữa biến IQ và biến tuổi.[1]

Complete data	
Age	IQ score
25	133
26	121
29	91
30	105
30	110
31	98
44	118
46	93
48	141
51	104
51	116
54	97

Incomplete data	
Age	IQ score
25	
26	
29	
30	
30	
31	
44	118
46	93
48	141
51	104
51	116
54	97

Hình 2.2: Ví dụ MAR

Missing Not at Random Dữ liệu khuyết không ngẫu nhiên:

MNAR: Sự mất mát dữ liệu không phải là ngẫu nhiên mà có một mối quan hệ xu hướng giữa giá trị bị missing và giá trị không bị missing trong một biến. Ví dụ: ở hình 2.3 - những người có IQ thấp sẽ bị missing còn IQ cao thì không bị thiếu, có nghĩa là có sự liê quan giữa 2 giá trị missing và không missing trong chính biến IQ.[2]

Complete data	
Age	IQ score
25	133
26	121
29	91
30	105
30	110
31	98
44	118
46	93
48	141
51	104
51	116
54	97

Incomplete data	
Age	IQ score
25	133
26	121
29	
30	
30	110
31	
44	118
46	
48	141
51	
51	116
54	

Hình 2.3: Ví dụ MNAR

2.1.3 Nguyên nhân của Missing Data

Nguyên nhân dẫn đến Missing Data (Dữ liệu bị thiếu) có thể xuất phát từ nhiều yếu tố khác nhau trong quá trình thu thập, lưu trữ và truyền tải dữ liệu. Dưới đây là một số nguyên nhân phổ biến gây ra dữ liệu bị thiếu:

Lỗi kỹ thuật: Sự cố phần mềm, hỏng thiết bị, hoặc lỗi truyền tải dữ liệu.

Lỗi con người: Sai sót khi nhập liệu, bỏ sót câu hỏi, hoặc nhập sai định dạng.

Khảo sát không hoàn chỉnh: Người tham gia từ chối trả lời câu hỏi nhạy cảm hoặc bỏ qua vì không hiểu.

Dữ liệu nhạy cảm: Các vấn đề cá nhân như thu nhập, sức khỏe khiến người tham gia không muốn cung cấp thông tin.

Lỗi xử lý dữ liệu: Mất giá trị khi dọn dẹp hoặc chuyển đổi dữ liệu.

Sự thay đổi theo thời gian: Biến động môi trường, xã hội khiến dữ liệu không được thu thập đầy đủ.

Ngẫu nhiên: Các lỗi không dự đoán trước như trục trặc thiết bị hoặc sự cố ngoài ý muốn.[3]

2.2 Dữ liệu ngoại lai (Outlier)

2.2.1 Định nghĩa

Dữ liệu ngoại lai là những giá trị dữ liệu (records) được ghi nhận có sự khác biệt bất thường so với những giá trị dữ liệu khác, không theo một quy tắc chung nào và có thể gây ra sự sai lệch trong kết quả phân tích và việc xây dựng các thuật toán dự đoán.[4]

Các dữ liệu ngoại lai có thể làm sai lệch và tạo ra thiên kiến tiêu cực cho toàn bộ kết quả của một phân tích.

Trong một số trường hợp, các giá trị ngoại lai có thể cung cấp insight cho một kết quả phân tích.

Loại bỏ outlier trong dataset sẽ giúp kết quả phân tích của bạn chính xác và sát với thực tế hơn.

2.2.2 Xác định giá trị ngoại lai (Outlier)

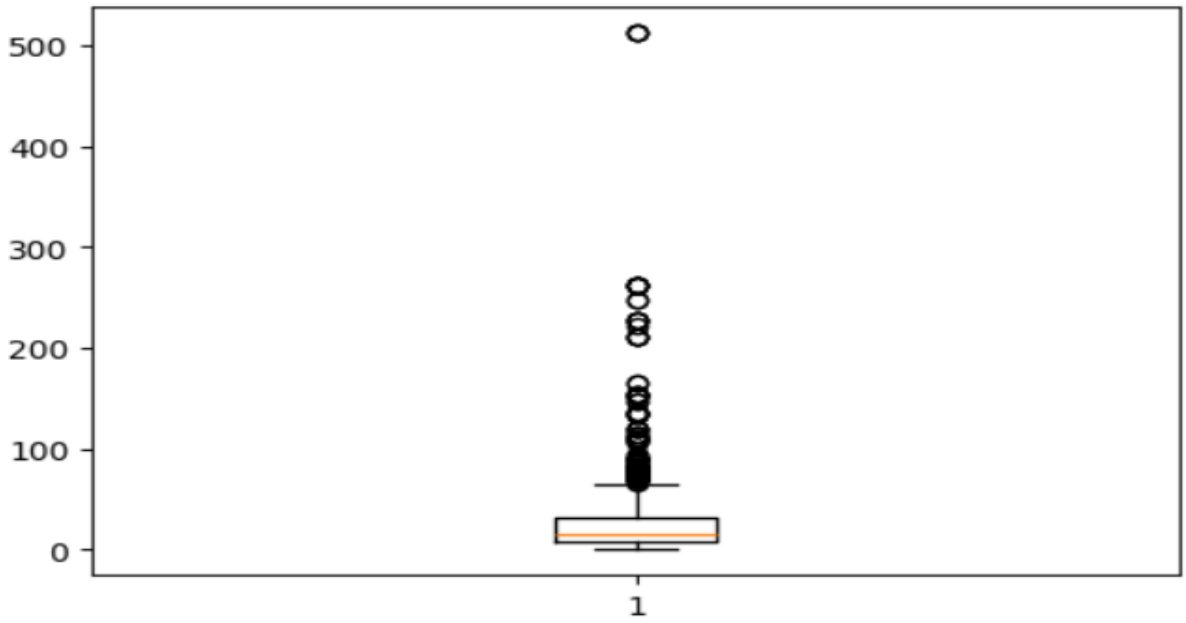
Cách đơn giản nhất để tìm ra các giá trị ngoại lai (outlier) chính là trực tiếp tìm trong bảng/trang tính của một tập dữ liệu. Có một số trường hợp các giá trị ngoại lai là do lỗi đánh máy trong lúc nhập liệu.[5]

Ví dụ cột "Tuổi" của Atony Smith có giá trị ngoại lai lên đến 470 tuổi. Giá trị chính xác có thể là 47 tuổi, 70 hoặc 40 tuổi.

Cod	Name	Age	ethnicity	Purchase value
1	Mark Grant	33	white	5500
2	Steve Manson	57	black	2500
3	Mary Jane	27	indigenous	3700
4	Antony Smith	470	white	2900
5	Aaron James	44	black	3300

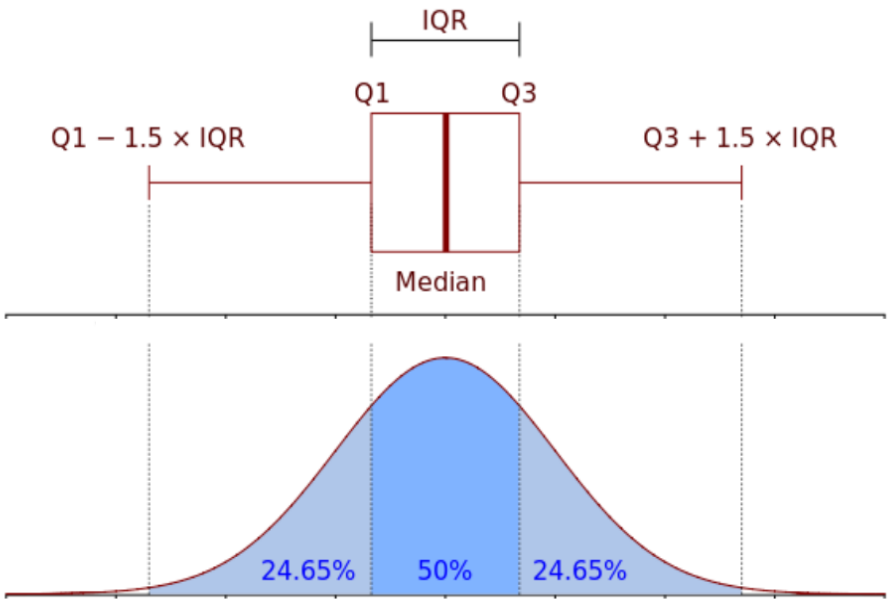
Hình 2.4: Ví dụ về outlier

Trực quan dữ liệu theo biểu đồ có thể giúp xác định một cách rõ ràng rằng có điểm dữ liệu khác biệt với những giá trị dữ liệu còn lại. Đồ thị Boxplot là phương pháp phổ biến trong nhận diện điểm dị biệt loại 2 dựa trên đặc điểm phân phối chuẩn dữ liệu trong phân tích thống kê.



Hình 2.5: Ví dụ về outlier trực quan bằng boxplot

Dựa trên khoảng tứ phân vị



Hình 2.6: Khoảng tứ phân vị

Các giá trị nằm ngoài khoảng [Q1 – 1.5 x IQR, Q1 + 1.5 x IQR] được coi là các giá trị ngoại lai.

2.3 Dữ liệu trùng lặp (Duplicate Data)

2.3.1 Định nghĩa

Dữ liệu trùng lặp (Duplicate Data) là các bản ghi hoặc mục dữ liệu xuất hiện nhiều lần trong một tập dữ liệu mà không cần thiết hoặc không mong muốn. Những bản ghi này có thể giống nhau hoàn toàn hoặc tương tự nhau ở một số thuộc tính, gây ra sự dư thừa và làm giảm chất lượng của dữ liệu.[6]

2.3.2 Nguyên nhân của dữ liệu trùng lặp

Dữ liệu trùng lặp có thể phát sinh từ nhiều nguồn, bao gồm:

- **Lỗi nhập liệu**: Người dùng nhập thông tin nhiều lần hoặc nhầm lẫn.
- **Hệ thống ghi nhận lỗi**: Một sự kiện được ghi lại nhiều lần bởi các hệ thống khác nhau.
- **Gộp dữ liệu từ nhiều nguồn**: Khi dữ liệu được hợp nhất từ nhiều cơ sở dữ liệu hoặc tập dữ liệu, các bản ghi có thể bị lặp lại.

- **Lỗi khi thu thập dữ liệu**: Lập trình không chính xác khi crawling dữ liệu từ các trang web.

2.3.3 Các loại dữu liệu trùng lặp (Duplicate Data)

Trùng lặp hoàn toàn: Các bản ghi giống nhau 100% trên tất cả các thuộc tính.

Trùng lặp một phần: Các bản ghi có một số thuộc tính giống nhau nhưng không hoàn toàn.

Trùng lặp ngữ nghĩa (semantic duplication): Các bản ghi mang ý nghĩa tương tự nhưng được biểu diễn khác nhau.

2.4 Làm sạch dữ liệu

Làm sạch dữ liệu (Data Cleaning) là quá trình xác định, sửa chữa hoặc loại bỏ các dữ liệu không chính xác, không đầy đủ, không phù hợp hoặc trùng lặp từ tập dữ liệu để nâng cao chất lượng và độ tin cậy của dữ liệu. Đây là bước quan trọng trong phân tích dữ liệu và chuẩn bị cho các mô hình học máy.

2.4.1 Xữ lý khuyết thiếu dữ liệu (Missing Data)

Các kỹ thuật phổ biến để xử lý mising data:

- Loại bỏ hoàn toàn hàng có dữ liệu thiếu: sử dụng khi tỷ lệ mising data nhỏ (dưới 5 – 10%). Dữ liệu bị thiếu không ảnh hướng đến phân tích.
- Điền dữ liệu thiếu bằng các giá trị thống kê (mean, median, mode): Khi dữ liệu missing không quá phức tạp và không ảnh hưởng bởi các thuộc tính khác.
- Gán nhãn dữ liệu: Khi giá trị bị thiếu mang ý nghĩa riêng hoặc cần giữ lại các hàng có missing data.
- Điền khuyết bằng giá trị lân cận: Khi giá trị bị thiếu có thể được ước tính từ các giá trị gần đó (theo thứ tự thời gian hoặc vị trí).

- Hồi quy tuyến tính (Regression Imputation): Khi giá trị bị thiếu phụ thuộc vào các biến khác trong tập dữ liệu.
- Phương pháp Multiple Imputation: Khi dữ liệu thiếu phức tạp hoặc có nhiều biến ảnh hưởng lẫn nhau.

2.4.2 Xử lý dữ liệu ngoại lai (Outlier)

Loại bỏ hoàn toàn dữ liệu ngoại lai (outlier):

- Ngoại lai là lỗi nhập liệu hoặc không có giá trị thực tiễn: Ví dụ, giá sản phẩm ghi nhầm thành 1 triệu thay vì 10 nghìn.
- Ngoại lai không đại diện cho dữ liệu thực tế: Các giá trị quá bất thường có thể không hữu ích và làm sai lệch phân tích.
- Giá trị nằm ngoài phạm vi [Q1−1.5×IQR, Q3+1.5×IQR] được coi là ngoại lai.

Giữ nguyên dữ liệu ngoại lai:

- Khi dữ liệu có ý nghĩa đặc biệt: Ví dụ, dữ liệu ngoại lai có thể đại diện cho các sự kiện hiếm gặp (như một đợt mua sắm lớn trong thương mại điện tử hoặc sự cố thời tiết bất thường). Trong trường hợp này, việc giữ nguyên dữ liệu ngoại lai là cần thiết để phân tích các hiện tượng đặc biệt.
- Dữ liệu không ảnh hưởng đến mục tiêu phân tích: Nếu ngoại lai không gây nhiễu hoặc làm sai lệch kết quả phân tích, bạn có thể giữ nguyên.
- Phân tích sâu các ngoại lệ: Đôi khi, các ngoại lệ cung cấp thông tin quan trọng mà phân tích thông thường bỏ qua. Ví dụ, trong y học, một bệnh nhân có chỉ số vượt xa bình thường có thể giúp phát hiện bệnh hiếm gặp.

2.5 Chuẩn hoá dữ liệu

Chuẩn hóa dữ liệu là quá trình biến đổi dữ liệu về một thang đo chung hoặc một dạng tiêu chuẩn để phù hợp hơn với các thuật toán phân tích hoặc học máy. Có nhiều cách để chuẩn hóa dữ liệu, tùy thuộc vào mục tiêu và đặc điểm của tập dữ liệu.

2.5.1 Min-Max Normalization

Min-Max Normalization là một phương pháp chuẩn hóa dữ liệu phổ biến nhằm chuyển đổi giá trị dữ liệu về một phạm vi cố định, thường là [0,1] hoặc [−1,1]. Phương pháp này yêu cầu chúng ta cần xác định được giá trị lớn nhất và giá trị nhỏ nhất của dữ liệu.

Công thức

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

- x: Giá trị gốc của dữ liệu.
- $\min(x)$: Giá trị nhỏ nhất trong tập dữ liệu.
- $\max(x)$: Giá trị lớn nhất trong tập dữ liệu.
- x': Giá trị sau khi chuẩn hóa.

2.5.2 Z-Score Standardization

Z-Score Standardization (còn gọi là Z-Score Normalization) là một phương pháp chuẩn hóa dữ liệu nhằm đưa các giá trị của biến số về dạng có trung bình bằng 0 và độ lệch chuẩn bằng 1. Điều này giúp các giá trị trở nên đồng nhất về thang đo và giảm ảnh hưởng của các giá trị lớn hoặc nhỏ bất thường (outliers).

Công thức Z-Score Standardization

$$z = \frac{x - \mu}{\sigma}$$

- x: Giá trị ban đầu của dữ liệu.
- μ: Giá trị trung bình của tập dữ liệu.
- σ: Độ lệch chuẩn của tập dữ liệu.
- z: Giá trị sau khi chuẩn hóa.

2.5.3 Robust Scaling

Robust Scaling là một phương pháp chuẩn hóa dữ liệu sử dụng **median (trung vị)** và **IQR (khoảng tứ phân vị)** thay vì trung bình (μ\muμ) và độ lệch chuẩn (σ\sigmaσ). Phương pháp này đặc biệt hiệu quả khi làm việc với dữ liệu chứa nhiều ngoại lai (outliers), vì trung vị và IQR ít nhạy cảm với các giá trị bất thường.

Giả sử x là giá trị ban đầu, x' là giá trị sau khi chuẩn hóa:

$$x' = \frac{x - \text{median}(x)}{\text{IQR}}$$

- **Median (trung vị)**: Giá trị trung tâm của dữ liệu khi được sắp xếp theo thứ tự tăng dần.
- **IQR (Interquartile Range - Khoảng tứ phân vị)**: Hiệu giữa phần tư thứ ba (Q3) và phần tư thứ nhất (Q1):

$$\text{IQR} = Q3 - Q1$$

2.5.4 Logarithmic Transformation (Biến đổi logarit)

Logarithmic Transformation là phương pháp biến đổi dữ liệu bằng cách áp dụng hàm logarit. Mục tiêu chính là **giảm sự chênh lệch giữa các giá trị lớn và nhỏ** trong dữ liệu, đồng thời làm mượt dữ liệu lệch (skewed) hoặc không phân phối chuẩn, giúp phân tích và mô hình hóa hiệu quả hơn.

$$x' = \log(x + c)$$

- x: Giá trị ban đầu.
- c: Hằng số để đảm bảo $x + c > 0$ (thường là 1 hoặc giá trị nhỏ hơn nếu dữ liệu không âm).
- $\log$: Logarit cơ số tự nhiên (hoặc cơ số 10 tùy ứng dụng).

2.6 Chuyển đổi dữ liệu dạng phân loại (Encoding Categorical Data)

Chuyển đổi dữ liệu dạng phân loại (categorical data) là quá trình chuyển đổi các giá trị phân loại (ví dụ như tên, loại, nhãn) thành các dạng dữ liệu số mà các thuật toán học máy có thể xử lý. Dữ liệu phân loại thường không có sự sắp xếp hoặc mối quan hệ toán học, do đó cần phải biến đổi trước khi sử dụng trong các mô hình học máy.

2.6.1 Label Encoding

Label Encoding là một kỹ thuật trong tiền xử lý dữ liệu, chủ yếu được sử dụng để chuyển đổi các giá trị phân loại (categorical values) thành các giá trị số. Phương pháp này thay thế các nhãn (labels) trong dữ liệu phân loại thành các số nguyên (integer values). Đây là một kỹ thuật đơn giản và phổ biến trong học máy, đặc biệt là khi làm việc với các thuật toán yêu cầu dữ liệu số như **Decision Trees**, **Logistic Regression**, hoặc **Random Forest**.

2.6.2 One-hot Encoding

One-hot encoding là một phương pháp phổ biến trong tiền xử lý dữ liệu, đặc biệt khi làm việc với dữ liệu phân loại (categorical data). Mục đích của One-hot Encoding là chuyển đổi các giá trị phân loại thành các vector nhị phân sao cho mỗi giá trị phân loại được biểu diễn bằng một cột mới, trong đó chỉ có một giá trị là 1 (hot) và các giá trị còn lại là 0 (cold).

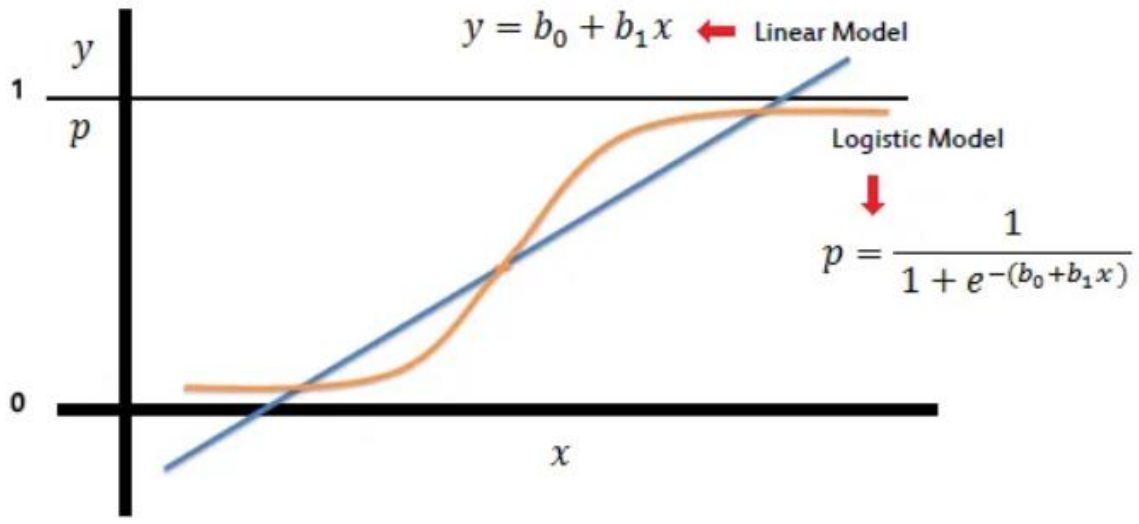
2.6.3 OdinalEncode

Ordinal Encoding là một kỹ thuật trong học máy được sử dụng để chuyển đổi các biến phân loại (categorical variables) thành các giá trị số nguyên. Phương pháp này đặc biệt hữu ích khi xử lý các biến phân loại có thứ tự (ordinal), tức là các biến phân loại có một thứ tự tự nhiên giữa các mức giá trị của nó. Các biến phân loại thứ tự thường đại diện cho những đặc tính có thể xếp theo thứ tự tăng dần hoặc giảm dần. Ví dụ điển hình bao gồm các mức độ đánh giá như "thấp", "trung bình", "cao" hoặc các mức độ hài lòng như "rất không hài lòng", "không hài lòng", "hài lòng", "rất hài lòng".

2.7 Mô hình dự đoán

2.7.1 Logistic Regression

Hồi quy Logistic là một mô hình thống kê được sử dụng để phân loại nhị phân, tức dự đoán một đối tượng thuộc vào một trong hai nhóm. Hồi quy Logistic làm việc dựa trên nguyên tắc của hàm sigmoid – một hàm phi tuyến tự chuyển đầu vào của nó thành xác suất thuộc về một trong hai lớp nhị phân.[7]



Hình 2.7: Logistic Regression sử dụng hàm phi tuyến để xác định xác suất của hai lớp 0 và 1.

Hồi quy Logistic hoạt động dựa trên hàm Sigmoid, được biểu diễn như sau:

$$S(z)=1/(1+e^{-z})$$

Hàm Sigmoid nhận đầu vào là một giá trị z bất kỳ, và trả về đầu ra là một giá trị xác suất nằm trong khoảng [0,1]. Khi áp dụng vào mô hình Hồi quy Logistic với đầu vào là ma trận dữ liệu *X* và trọng số *w*, ta có *z*=*Xw*.

Việc huấn luyện của mô hình là tìm ra bộ trọng số w*w* sao cho đầu ra dự đoán của hàm Sigmoid gần với kết quả thực tế nhất. Để làm được điều này, ta sử dụng hàm mất mát (Loss Function) để đánh giá hiệu năng của mô hình. Mô hình càng tốt khi hàm mất mát càng nhỏ.[8]

Hàm mất mát (Loss Function) là một hàm số được sử dụng để đo lường mức độ lỗi mà mô hình của chúng ta tạo ra khi dự đoán các kết quả từ dữ liệu đầu vào. Trong bài toán Hồi quy Logistic, chúng ta sử dụng hàm mất mát Cross-Entropy (còn gọi là Log Loss) để đánh giá hiệu năng của mô hình.

Hàm mất mát Cross-Entropy được định nghĩa như sau:

$$L(w) = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(p_i) + (1-y_i)\log(1-p_i)]$$

Trong đó:

- *n*: số lượng mẫu dữ liệu trong tập huấn luyện.
- *yi*: giá trị thực tế của đầu ra thứ i.
- *pi*: xác suất dự đoán thuộc lớp 1 của mô hình cho đầu vào thứ i.

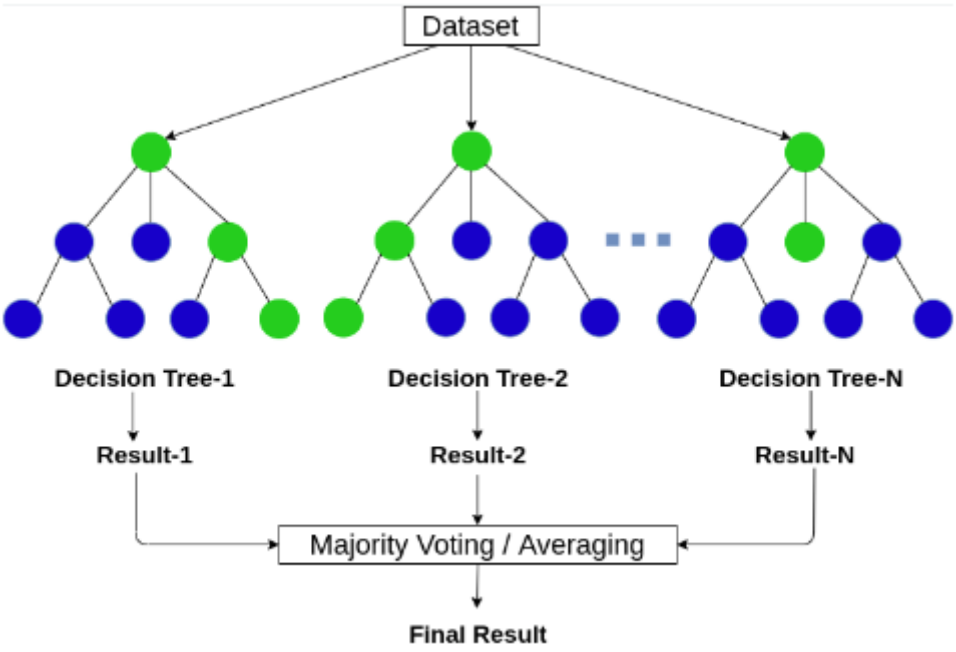
Hàm Cross-Entropy đo lường khoảng cách giữa hai phân phối xác suất *yi* và *pi*. Khi mô hình dự đoán chính xác, tức là nếu *yi* = 1 thì *pi* càng gần 1, và nếu *yi* = 0 thì *pi* càng gần 0, sau đó hàm mất mát sẽ tiến gần về 0.

Trong quá trình huấn luyện, chúng ta tìm cách cập nhật bộ trọng số w sao cho giá trị hàm mất mát Cross-Entropy đạt giá trị nhỏ nhất, dẫn đến một mô hình dự đoán tốt nhất.

Để tìm giá trị tối ưu cho bộ trọng số w, chúng ta có thể sử dụng kỹ thuật Gradient Descent. Tại mỗi bước lặp, chúng ta cập nhật w theo phương từng ứng với đạo hàm của hàm mất mát $L(w)$ theo w.

2.7.2 Random Forest

Random Forest là một trong những thuật toán học máy mạnh mẽ và phổ biến nhất hiện nay, đặc biệt trong các ứng dụng phân loại và hồi quy. Được phát triển bởi Leo Breiman vào đầu thế kỷ 21, Random Forest được xem là một kỹ thuật ensemble learning (học tổ hợp), kết hợp nhiều cây quyết định (decision trees) để đưa ra dự đoán. Random Forest có thể được sử dụng cho cả bài toán hồi quy (regression) và phân loại (classification).[9]



Hình 2.8: Mô tả Random Forest

Thành phần chính của mô hình Random Forest:

- Cây quyết định (Decision Trees): Mỗi cây quyết định trong Random Forest được xây dựng từ một tập con ngẫu nhiên của dữ liệu huấn luyện (data sampling). Tại mỗi nút (node) trong cây, chỉ một tập con

ngẫu nhiên các đặc trưng (features) được xem xét để chia nhánh (split). Điều này làm tăng tính ngẫu nhiên và giảm sự phụ thuộc giữa các cây.[10]

- Bagging (Bootstrap Aggregation): Thuật toán Random Forest sử dụng phương pháp Bagging (Bootstrap Aggregating) để xây dựng mỗi cây quyết định. Thay vì sử dụng toàn bộ tập dữ liệu huấn luyện, mỗi cây được huấn luyện trên một mẫu ngẫu nhiên từ tập dữ liệu, với việc lấy mẫu có hoàn lại (tức là một mẫu có thể được chọn nhiều lần).
- Tính ngẫu nhiên: Ngẫu nhiên hóa không chỉ ở việc chọn mẫu dữ liệu mà còn ở việc chọn các đặc trưng (features) tại mỗi lần chia nhánh. Điều này giúp giảm phương sai (variance) và tránh overfitting.

CHƯƠNG 3: THU THẬP VÀ TIỀN XỬ LÝ DỮ LIỆU

3.1 Thu thập dữ liệu

3.1.1 Giới thiệu bộ dữ liệu

Bộ dữ liệu này liên quan đến các yếu tố sức khoẻ tim mạch. Nó bao gồm 920 mẫu và 14 biến đầu vào đại diện cho kết quả xét nghiệm y khoa, cũng như các chỉ số liên quan tới sức khoẻ tim mạch.

Variable	Description
age	Age of the patient in years
sex	Gender of the patient (0 = male, 1 = female)
cp	Chest pain type: 0: Typical angina 1: Atypical angina 2: Non-anginal pain 3: Asymptomatic
trestbps	Resting blood pressure in mm Hg
chol	Serum cholesterol in mg/dl
fbs	Fasting blood sugar level, categorized as above 120 mg/dl (1 = true, 0 = false)
restecg	Resting electrocardiographic results: 0: Normal 1: Having ST-T wave abnormality 2: Showing probable or definite left ventricular hypertrophy
thalach	Maximum heart rate achieved during a stress test
exang	Exercise-induced angina (1 = yes, 0 = no)
oldpeak	ST depression induced by exercise relative to rest
slope	Slope of the peak exercise ST segment: 0: Upsloping 1: Flat 2: Downsloping
ca	Number of major vessels (0-4) colored by fluoroscopy
thal	Thalium stress test result: 0: Normal 1: Fixed defect 2: Reversible defect 3: Not described
target	Heart disease status (0 = no disease, 1 = presence of disease)

Bảng 3.1: Dataset Description

Cụ thể, các thông số bao gồm:

Thông tin cơ bản của bệnh nhân:

- age tuổi của bệnh nhân (đơn vị: năm)
- sex giới tính của bệnh nhân (0 là nam, 1 là nữ)

Các chỉ số sinh lý:

- cp (Chest pain type) các loại đau ngực mà bệnh nhân mắc phải (0: đau thắt ngực điển hình, 1: đau thắt ngực không điển hình, 2: đau không phải thắt ngực, 3: không triệu chứng)
- trestbps huyết áp lúc nghỉ ngơi của bệnh nhân (đơn vị: mmHg)
- chol mức cholesterol trong máu (đơn vị: mg/dl)
- fbs đường huyết lúc đói của bệnh nhân so với mức 120mg/dl (0: dưới hoặc bằng 120mg/dl là bình thường, 1: trên 120mg/dl là huyết áp cao)
- restecg: kết quả điện tâm đồ khi nghỉ (0: bình thường, 1: có bất thường sóng ST-T ví dụ: sóng T ngược hoặc ST chênh, 2: phì đại thất trái do tăng huyết áp hoặc các bệnh tim khác)
- thalach nhịp tim tối đa đạt được khi thực hiện bài kiểm tra gắng sức
- exang đau thắt ngực khi gắng sức (0: bệnh nhân không bị đau, 1: bệnh nhân bị đau khi gắng sức)
- oldpeak sự giảm ST so với lúc nghỉ trong bài kiểm tra gắng sức
- slope hình dạng đường cong ST (0: tăng dần - ít nguy hiểm nhất, 1: phẳng – nguy cơ trung bình, 2: giảm dần – nguy cơ cao nhất)
- ca số lượng mạch máu chính bị hẹp nhìn thấy qua soi huỳnh quang (giá trị từ 0 đến 4 số lượng càng cao nguy cơ càng lớn)
- thal kết quả kiểm tra stress thallium (0: bình thường, 1: tổn thương cố định – tổn thương không phục hồi, 2: tổn thương hồi phục – tổn thương cải thiện được, 3: không mô tả)

Biến mục tiêu

- target tình trạng bệnh của bệnh nhân (0: không mắc bệnh, 1: bệnh nhân được chuẩn đoán có tim)

3.1.2 Mục đích sử dụng bộ dữ liệu

Bộ dữ liệu này được sử dụng rộng rãi trong các nghiên cứu và ứng dụng liên quan đến chẩn đoán và dự đoán bệnh tim. Dưới đây là các mục đích cụ thể, được trình bày chi tiết và rõ ràng theo từng phần:

Dự đoán nguy cơ mắc bệnh tim

Bộ dữ liệu giúp xác định bệnh nhân có nguy cơ mắc bệnh tim hay không dựa trên các chỉ số y khoa như tuổi, giới tính, huyết áp, cholesterol, loại đau ngực, nhịp tim, v.v.

Mục tiêu: Xây dựng mô hình dự đoán để phân loại bệnh nhân thành 2 nhóm:

- 0: Không mắc bệnh tim.
- 1: Có mắc bệnh tim.

Ứng dụng: Sử dụng các thuật toán học máy (machine learning) hoặc AI để tạo ra hệ thống dự đoán tự động, giúp bác sĩ và bệnh viện phát hiện sớm nguy cơ mắc bệnh tim.

Phân tích các yếu tố ảnh hưởng đến bệnh tim

Bộ dữ liệu giúp xác định các yếu tố quan trọng và có mối liên hệ mạnh nhất với nguy cơ mắc bệnh tim.

Mục tiêu: Tìm ra các yếu tố rủi ro như:

- Độ tuổi của bệnh nhân.
- Mức độ cholesterol và huyết áp.
- Loại đau ngực và kết quả điện tâm đồ (ECG).

Ứng dụng:

- Hỗ trợ các bác sĩ và nhà nghiên cứu nhận diện các chỉ số quan trọng trong việc chẩn đoán bệnh tim.
- Đưa ra các biện pháp phòng ngừa phù hợp dựa trên yếu tố rủi ro.

3.2 Tiền xử lý dữ liệu

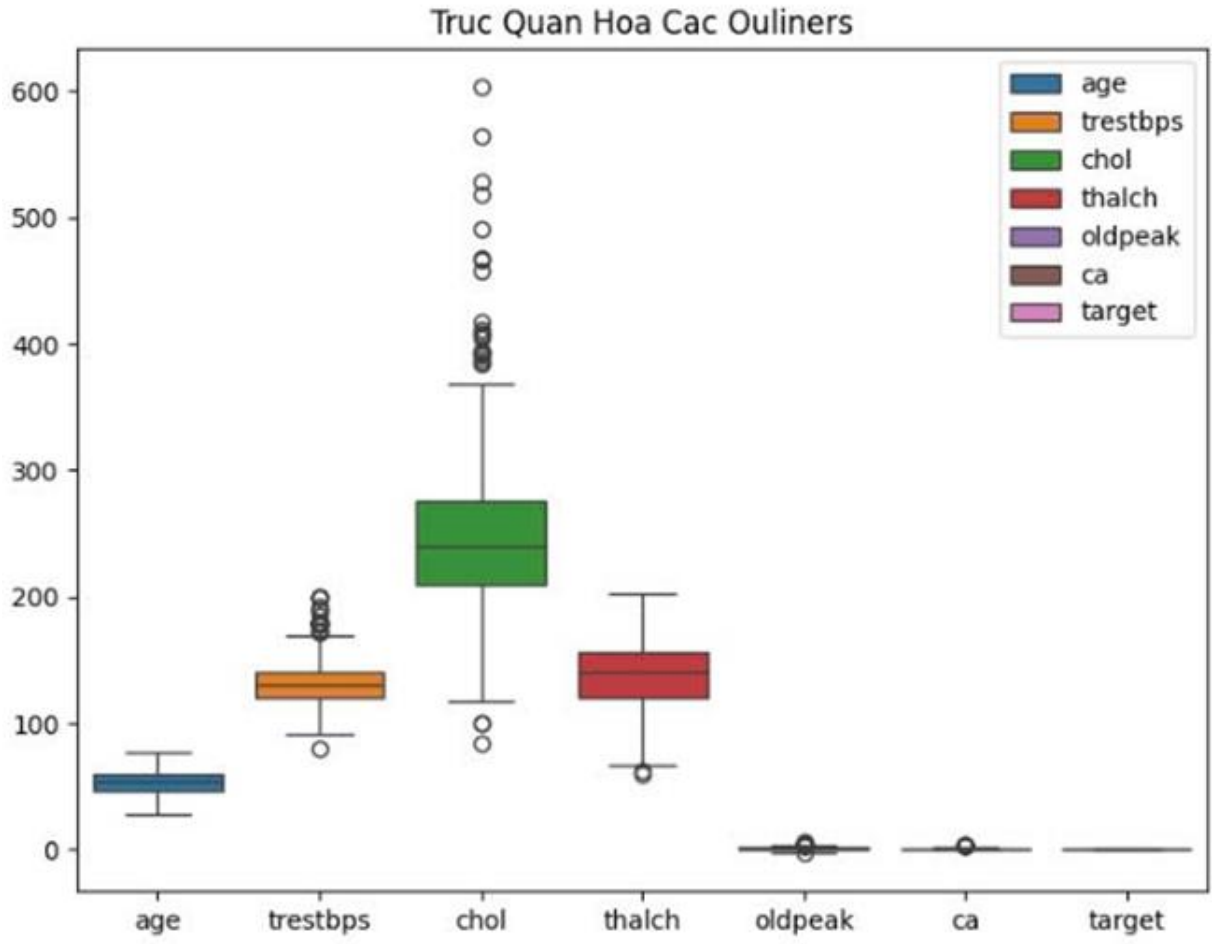
Sau khi tiến hành thu thập dữ liệu, chúng tôi đã nhận thấy một số vấn đề nổi bật cần đư`ợc xử lý để đảm bảo chất lượng cho các bước phân tích và mô hình hóa. Đầu tiên, tập dữ liệu bao gồm nhiều loại biến khác nhau, từ biến định lượng đến biến định tính, đòi hỏi các phương pháp xử lý phù hợp để chuẩn hóa và đồng nhất dữ liệu. Bên cạnh đó, có một số lượng đáng kể các giá trị bị thiếu, điều này không chỉ làm giảm tính toàn vẹn của tập dữ liệu mà còn có thể gây ảnh hưởng tiêu cực đến độ chính xác của mô hình dự đoán.

Ngoài ra, chúng tôi phát hiện nhiều giá trị ngoại lai (outliers), vốn có thể làm sai lệch kết quả phân tích nếu không được xử lý cẩn thận. Đáng chú ý, không phải tất cả các cột dữ liệu đều có ý nghĩa hoặc đóng góp đáng kể trong việc dự đoán khả năng mắc bệnh tim. Điều này đặt ra yêu cầu phải thực hiện các bước lựa chọn và đánh giá đặc trưng (feature selection) để loại bỏ những yếu tố không cần thiết, tối ưu hóa mô hình và tránh sự phức tạp không cần thiết.

3.2.1 Xử lý dữ liệu ngoại lai (Outlier)

Bộ data có dữ liệu ngoại lại như sau:

- trestbps: 27 giá trị outlier
- chol: 23 giá trị outlier
- thalch: 2 giá trị outlier
- oldpeak: 16 giá trị outlier
- ca: 20 giá trị outlier



Hình 3.1: Boxplot outliers của bộ data

Sau khi phân tích đặc điểm của các thuộc tính, chúng tôi quyết định giữ lại các giá trị ngoại lai (outliers) của các thuộc tính **trestbps**, **thalch**, **oldpeak**, và **ca**, vì các giá trị này nằm trong khoảng hợp lý và có ý nghĩa về mặt y học. Ví dụ, đối với thuộc tính huyết áp lúc nghỉ (**trestbps**), chỉ số bình thường là khoảng 120mmHg, trong khi các giá trị ngoại lai như 140 hoặc 160mmHg có thể liên quan đến các bệnh lý tim mạch như rối loạn nhịp tim hoặc các vấn đề về tim mạch khác. Tương tự, các thuộc tính **thalch**, **oldpeak**, và **ca** cũng có các giá trị ngoại lai nằm gần mức bình thường và không quá bất thường, do đó không cần loại bỏ.

Ngược lại, đối với thuộc tính **chol** (cholesterol), chúng tôi nhận thấy một số giá trị ngoại lai nằm ở mức vô lý, chẳng hạn như lượng cholesterol vượt quá 600mg/dl, điều này không phù hợp với thực tế y học. Do đó, chúng tôi

quyết định loại bỏ hoàn toàn các giá trị ngoại lai của thuộc tính này để đảm bảo tính chính xác và hợp lý của dữ liệu.

Chúng tôi sử dụng khoảng tứ phân vị để loại bỏ Outliers như sau:

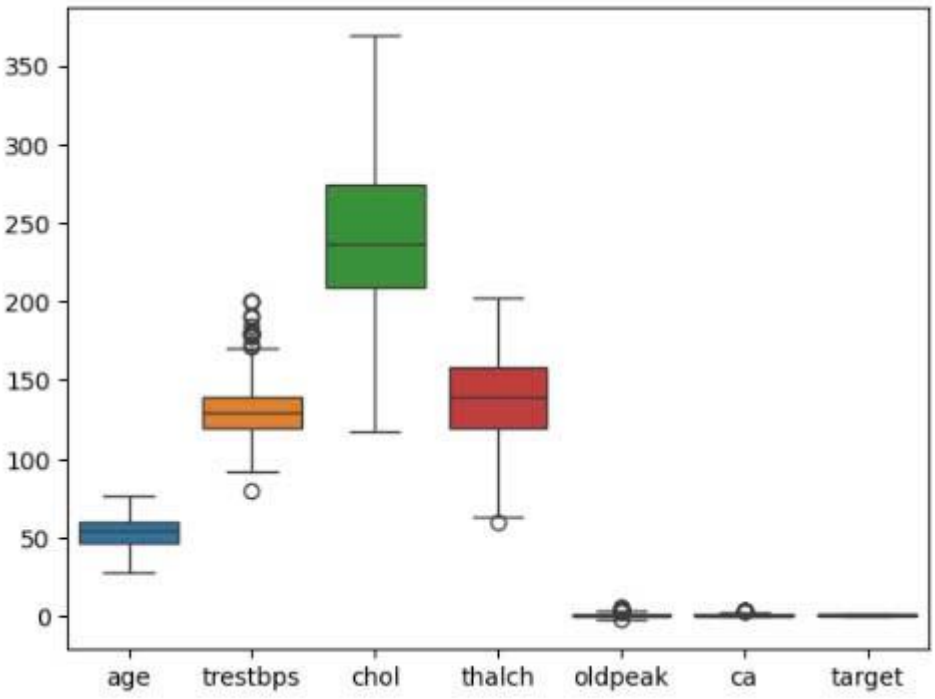
```
# loại bỏ outliers
for col in df_heart[['chol']]:
    q1 = df_heart[col].quantile(0.25)
    q3 = df_heart[col].quantile(0.75)
    iqr = q3 - q1
    lower_bound = q1 - 1.5 * iqr
    upper_bound = q3 + 1.5 * iqr

    # Lọc dữ liệu trong khoảng hợp lý
    # kết hợp giữ lại điều kiện nan và khoảng tứ phân vị
    df_heart = df_heart[(df_heart[col].isna()) | (df_heart[col] >= lower_bound) & (df_heart[col] <= upper_bound)]

df_heart
```

Hình 3.2: Hàm loại bỏ outliers

Sau khi loại bỏ outliers ta được như hình sau:



Hình 3.3: Boxplot sau khi loại bỏ outliers

3.2.2 Mã hoá các biến phân loại

Các biến phân loại (categorical variables) trong bộ data gồm các thuộc tính sau: sex, cp, fbs, restecg, exang, slope, thal.

Chúng tôi tiến hành mã hoá các biến phân loại như sau:

Bước 1: Tiến hành tách categorical variables và numeric variables

```
numeric_col = ['id', 'age', 'trestbps', 'chol', 'thalch',   'oldpeak',  'ca',   'target']
numeric_col
```

Hình 3.4: Numeric variables

```
categories_col = [column for column in df_heart.columns if column not in numeric_col]
categories_col
```

Hình 3.5: Categorical variables

Bước 2: Tiến hành mã hoá sử dụng OrdinalEncoder

```
from sklearn.preprocessing import OrdinalEncoder
import numpy as np

# Áp dụng OrdinalEncoder (giữ nguyên giá trị NaN)
ordinal_encoder = OrdinalEncoder(handle_unknown='use_encoded_value', unknown_value=-1)

# Chỉ encode các cột phân loại
df_heart[categories_col] = ordinal_encoder.fit_transform(df_heart[categories_col])

df_heart.isna().sum()
```

Hình 3.6: OrdinalEncoder

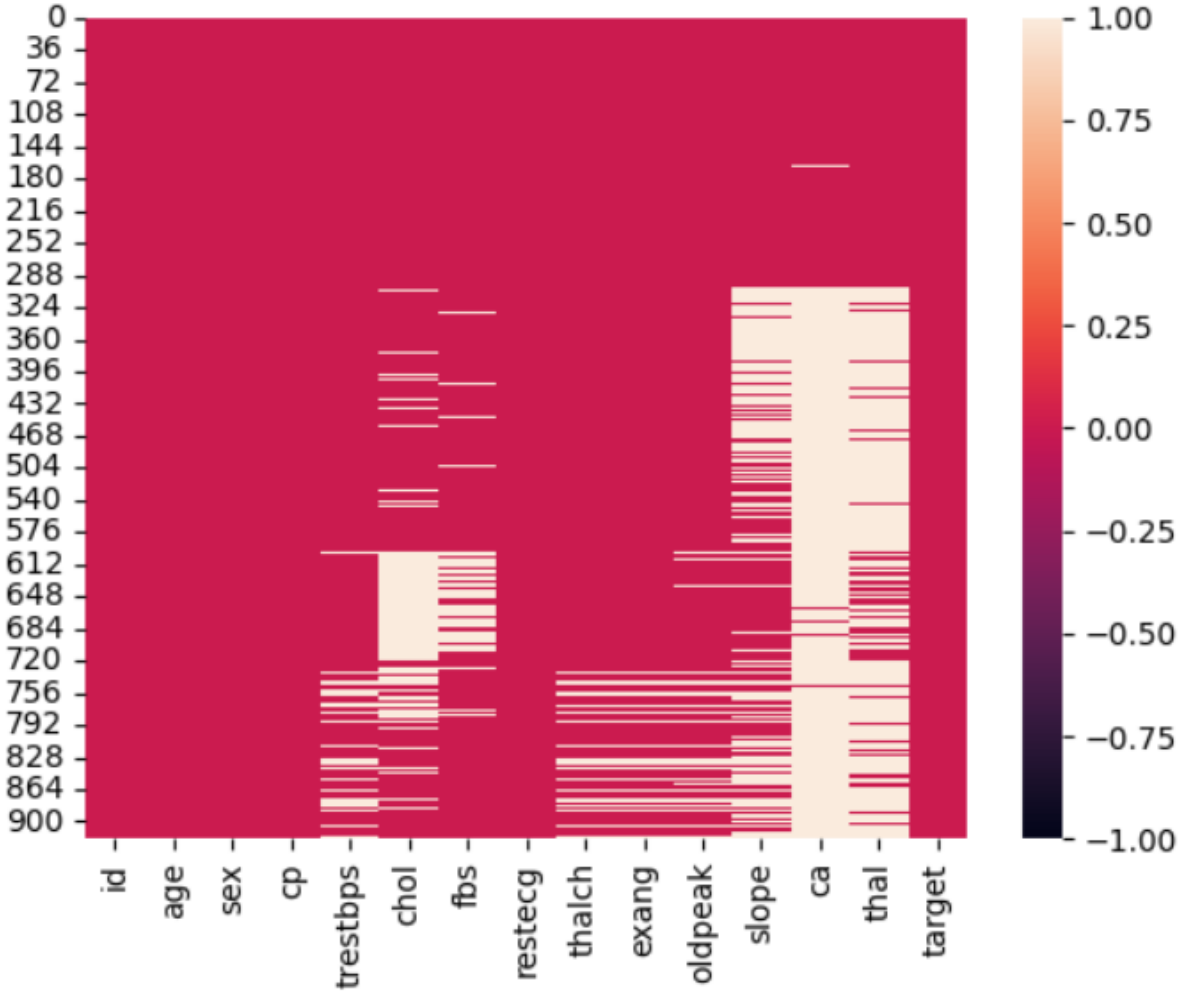
Việc chuyển đổi các biến phân loại (categorical variables) thành dạng số nguyên (integer) bằng cách ánh xạ các giá trị của chúng thành một dãy số thứ tự (ordinal) là một bước quan trọng trong xử lý dữ liệu. Điều này giúp dữ liệu phân loại trở nên dễ hiểu và sử dụng được trong các mô hình học máy (machine learning), vì hầu hết các mô hình yêu cầu đầu vào là dữ liệu số.

3.2.3 Xử lý dữ liệu khuyết thiếu (Missing Data)

Bộ data có dữ liệu khuyết thiếu như sau:

- trestbps: 60 dòng thiếu
- chol: 202 dòng thiếu
- fbs: 90 dòng thiếu
- restecg: 2 dòng thiếu
- thalch: 55 dòng thiếu
- exang: 55 dòng thiếu
- oldpeak: 62 dòng thiếu

- slope: 309 dòng thiếu
- ca: 611 dòng thiếu
- thal: 486 dòng thiếu



Hình 3.7: Trực quan dữ liệu khuyết thiếu

Sau khi tiến hành kiểm tra và phân tích dữ liệu, chúng tôi nhận thấy rằng một số cột trong tập dữ liệu có tỷ lệ thiếu hơn 50% tổng số giá trị. Khi tỷ lệ thiếu quá cao như vậy, việc loại bỏ các cột này có thể dẫn đến việc mất đi một lượng dữ liệu quan trọng, làm giảm tính đại diện của bộ dữ liệu và ảnh hưởng đến độ chính xác của các phân tích sau này. Bên cạnh đó, nếu áp dụng các phương pháp điền giá trị thiếu thông qua trung bình (mean), trung vị (median) hay mode, chúng tôi lo ngại rằng điều này có thể làm sai lệch kết quả dự đoán, vì những giá trị này có thể không phản ánh đúng sự phân bố tự nhiên của dữ liệu. Vì vậy, sau khi xem xét kỹ lưỡng, chúng tôi quyết

định sử dụng phương pháp MICE (Multiple Imputation by Chained Equations) để điền các giá trị thiếu. Phương pháp này cho phép tái tạo các giá trị thiếu một cách chính xác hơn, dựa trên các mối quan hệ giữa các đặc trưng trong dữ liệu, từ đó giúp duy trì tính toàn vẹn của bộ dữ liệu và cải thiện kết quả dự đoán mà không làm mất đi các thông tin quan trọng.

```
# Dung MICE dien Missing Data
from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import IterativeImputer
# Khởi tạo MICE imputer
imputer = IterativeImputer(max_iter=20, random_state=0)
df_heart = imputer.fit_transform(df_heart)
df_heart = pd.DataFrame(df_heart, columns=['id', 'age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg',
                                           'thalch', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'])
df_heart.head(50)
```

Hình 3.8: Hàm điền dữ liệu khuyết thiếu

3.2.4 Chuẩn hoá dữ liệu

Để chuẩn bị cho quá trình xây dựng mô hình dự đoán, một bước quan trọng mà chúng tôi thực hiện là chuẩn hóa dữ liệu. Mục tiêu của bước này là đưa các đặc trưng (features) trong dữ liệu về cùng một thang đo. Điều này rất quan trọng trong bối cảnh các mô hình học máy thường hoạt động dựa trên các phép tính toán trọng số hoặc khoảng cách giữa các đặc trưng. Nếu dữ liệu không được chuẩn hóa, các đặc trưng có giá trị lớn hơn sẽ vô tình được ưu tiên hơn trong quá trình học, mặc dù chúng không nhất thiết phải quan trọng hơn các đặc trưng khác.

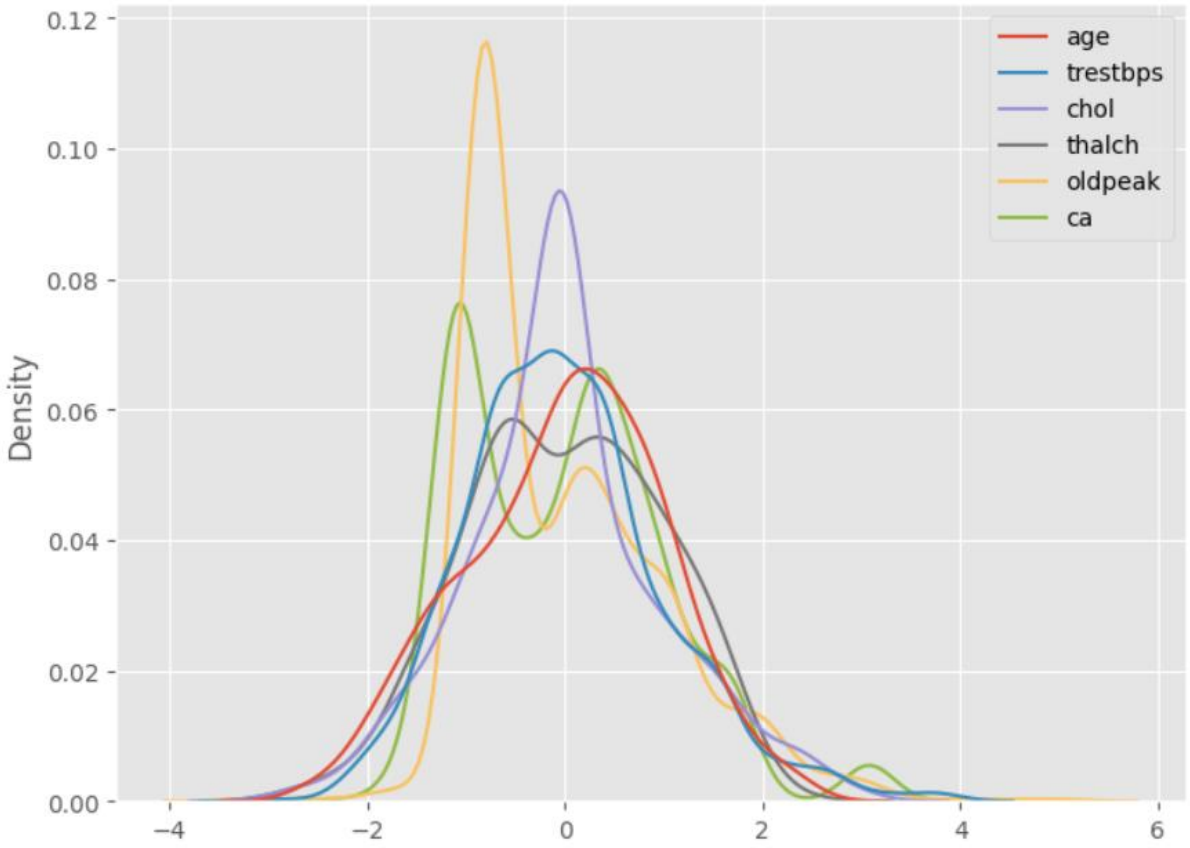
Ví dụ, trong một bộ dữ liệu với hai đặc trưng age (tuổi) và salary (lương), giá trị lương thường lớn hơn giá trị tuổi, dẫn đến việc mô hình có thể gán trọng số cao hơn cho đặc trưng salary.

Do đó chúng tôi sử dụng phương pháp Standardization để đưa các đặc trưng về cùng một thang đo.

```
X = df_heart.drop(columns=['target'])
y = df_heart['target']

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_scaled
```

Hình 3.9: Hàm chuẩn hoá StandardScaler



Hình 3.10: Biểu đồ sau khi chuẩn hoá

Việc chuẩn hóa, thông qua các phương pháp Standardization, không chỉ giúp loại bỏ sự thiên vị này mà còn cải thiện hiệu suất của mô hình. Nó đảm bảo rằng mỗi đặc trưng đóng góp vào quá trình học trên cơ sở bình đẳng, đồng thời giúp thuật toán tối ưu hội tụ nhanh hơn và cho kết quả dự đoán chính xác hơn. Vì vậy, chuẩn hóa là một bước thiết yếu trong quy trình tiền xử lý dữ liệu.

3.2.5 Giảm chiều dữ liệu

Để chuẩn bị cho việc xây dựng mô hình dự đoán, chúng tôi đã áp dụng kỹ thuật giảm chiều dữ liệu bằng Phân tích Thành phần Chính (PCA). Mục tiêu của bước này là giảm số lượng đặc trưng trong tập dữ liệu, giúp loại bỏ các thông tin dư thừa hoặc không quan trọng mà vẫn giữ được phần lớn thông tin cần thiết.

PCA hoạt động bằng cách chuyển đổi các đặc trưng ban đầu thành một tập hợp các thành phần chính mới, trong đó các thành phần này được sắp xếp theo thứ tự giảm dần về mức độ đóng góp thông tin. Chỉ một số ít thành phần đầu tiên, chứa phần lớn phương sai của dữ liệu, được giữ lại để sử dụng trong mô hình dự đoán.

Đầu tiên, PCA được áp dụng trên tập dữ liệu đã chuẩn hóa để tính toán tỷ lệ phương sai giải thích của từng thành phần. Điều này giúp chúng tôi hiểu được mức độ đóng góp thông tin của từng thành phần vào tập dữ liệu.

```
    # Ti le phuong sai
    # Khoant tin cay
    explained_var = pca.explained_variance_ratio_
    print(f'Ty le phuong sai giai thich', explained_var)
```

```
Ty le phuong sai giai thich [0.24836997 0.10275443 0.09386334 0.08157052 0.07480988 0.07405295
 0.06371643 0.05959402 0.0557424  0.04899116 0.0376435  0.03167963
 0.02721178]
```

Hình 3.11: Tỷ lệ phương sai giải thích

Sau đó, phương sai tích lũy được tính toán để xác định tổng lượng thông tin mà các thành phần chính giữ lại. Chúng tôi lựa chọn số lượng thành phần chính sao cho phương sai tích lũy đạt ít nhất 80%, đảm bảo rằng phần lớn thông tin quan trọng trong dữ liệu ban đầu được giữ lại.

```
    # chon tren phuong sai tich luy
    cumsum_explained_var = np.cumsum(pca.explained_variance_ratio_)
    print(f'Phuong sai tich luy', cumsum_explained_var)
```

```
Phuong sai tich luy [0.24836997 0.3511244  0.44498773 0.52655826 0.60136813 0.67542109
 0.73913751 0.79873153 0.85447393 0.90346509 0.94110859 0.97278822
 1.        ]
```

Hình 3.12: Phương sai tích luỹ

```
# Lựa chọn thanh phần chính
n_components = np.argmax(cumsum_explained_var >= 0.8)+1
print(f'So luong thanh phan chinh duoc chon la: {n_components}')
```

```
So luong thanh phan chinh duoc chon la: 9
```

Hình 3.13: Lựa chọn thành phần chính

Kết quả cho thấy số lượng thành phần chính được chọn là 9, tương ứng với các thành phần giữ lại ít nhất 80% phương sai. Việc lựa chọn này giúp giảm đáng kể số chiều dữ liệu, tăng hiệu quả tính toán, đồng thời giữ lại thông tin cần thiết để xây dựng mô hình dự đoán chính xác và ổn định.

Việc giảm chiều dữ liệu bằng PCA không chỉ làm giảm độ phức tạp của mô hình mà còn tăng hiệu quả tính toán và giảm nguy cơ quá khớp (overfitting). Điều này đặc biệt hữu ích khi xử lý các tập dữ liệu lớn và có nhiều đặc trưng tương quan với nhau. PCA giúp tập trung vào những thông tin quan trọng nhất, đảm bảo mô hình hoạt động hiệu quả và ổn định hơn.

CHƯƠNG 4: XÂY DỰNG MÔ HÌNH DỰ ĐOÁN

4.1 Mô hình Logistic Regression

4.1.1 Bài toán đặt ra

Bài toán yêu cầu xây dựng một mô hình dự đoán nhị phân để phân loại dữ liệu thành hai nhóm dựa trên các đặc trưng đầu vào. Mục tiêu của bài toán là tối ưu hóa độ chính xác của mô hình trên tập kiểm tra, qua đó đánh giá khả năng dự đoán của mô hình khi áp dụng vào các trường hợp thực tế. Để đạt được mục tiêu này, dữ liệu được xử lý trước, chia tách hợp lý và sử dụng thuật toán Logistic Regression để huấn luyện mô hình.

4.1.2 Lý do chọn Logistic Regression

Logistic Regression được chọn vì sự đơn giản và hiệu quả của nó trong các bài toán phân loại nhị phân. Thuật toán này không chỉ dễ triển khai mà còn mang lại hiệu quả cao trên các tập dữ liệu có kích thước nhỏ hoặc trung bình. Một điểm mạnh khác của Logistic Regression là khả năng giải thích rõ ràng: hệ số hồi quy giúp chúng ta hiểu được mức độ ảnh hưởng của từng đặc trưng đến kết quả dự đoán. Bên cạnh đó, Logistic Regression có thời gian huấn luyện nhanh, phù hợp với các bước phân tích dữ liệu ban đầu hoặc khi cần mô hình hóa nhanh chóng.

4.1.3 Quy trình xây dựng mô hình

Quy trình xây dựng mô hình bao gồm các bước chính như sau:

Xử lý dữ liệu: Đầu tiên, các đặc trưng đầu vào đã được giảm chiều bằng phương pháp PCA nhằm giảm thiểu hiện tượng quá khớp và tối ưu hóa tốc độ huấn luyện.

```
pca= PCA(n_components=n_components)
X_pca_reduced = pca.fit_transform(X_scaled)
X_pca_reduced
```

Hình 4.1: Giảm chiều dữ liệu PCA

Sau đó, dữ liệu được chia thành hai tập: tập huấn luyện chiếm 80% để đào tạo mô hình, và tập kiểm tra chiếm 20% để đánh giá hiệu năng.

```
# Chia dữ liệu thành Train và Test
X_train, X_test, y_train, y_test = train_test_split(X_pca_reduced, y, test_size=0.2, random_state=42)
```

Hình 4.2: Chia tập train, test

Huấn luyện mô hình: Logistic Regression được sử dụng làm thuật toán dự đoán. Mô hình được huấn luyện trên tập Train và lưu lại các tham số quan trọng như hệ số chặn (intercept) và các hệ số hồi quy của đặc trưng (coefficients).

```
# Huấn luyện Logistic Regression
model = LogisticRegression()
model.fit(X_train, y_train)

# Intercept và Coefficients
intercept = model.intercept_
coefficients = model.coef_
print(f'He so chan Intercept: {intercept}')
print(f'He So Hoi Quy Ung Voi Tung Dac Trung Coefficients: {coefficients}')
print('--==================================--')
```

Hình 4.3: Huấn luyện mô hình

Đánh giá hiệu năng: Sau khi huấn luyện, mô hình được đánh giá dựa trên tập Test. Các chỉ số như độ chính xác (Accuracy), Precision, Recall và F1-Score được tính toán để đo lường chất lượng dự đoán của mô hình.

```
# Dự đoán trên tập Test
y_pred = model.predict(X_test)

# Đánh giá mô hình
print("Accuracy:", accuracy_score(y_test, y_pred))
print('--================================--')
print("Classification Report:\n", classification_report(y_test, y_pred))
```

Hình 4.4: Đánh giá hiệu năng

4.1.4 Kết quả và nhận xét

Mô hình Logistic Regression đã được triển khai và đánh giá với các kết quả cụ thể như sau:

Hệ số chặn (Intercept): 0.63324623.

Hệ số hồi quy (Coefficients): Các giá trị hệ số hồi quy lần lượt là: [1.95775961 -0.68703695 -0.65640338 -1.16877642 1.44326084 0.82079419 -1.03617265 0.59308204]. Những hệ số này thể hiện mức độ đóng góp của từng đặc trưng vào dự đoán kết quả.

Độ chính xác (Accuracy): 83%.

Báo cáo phân loại (Classification Report):

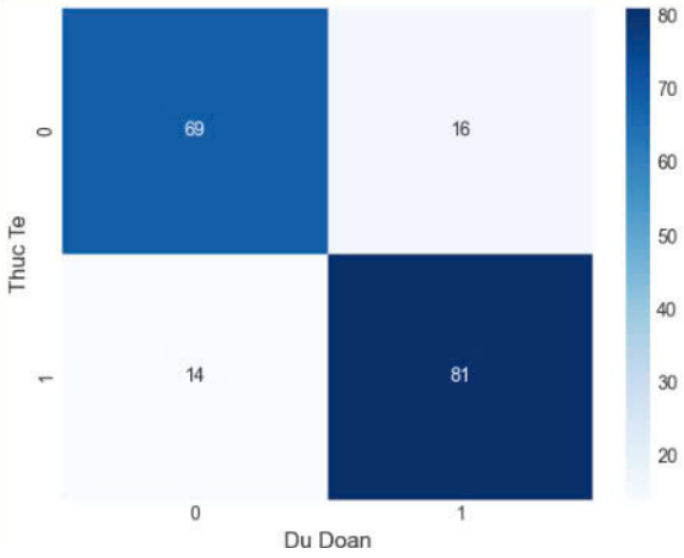
- Đối với nhãn **0.0**: Precision = 0.83, Recall = 0.81, F1-Score = 0.82 với 85 mẫu.
- Đối với nhãn **1.0**: Precision = 0.84, Recall = 0.85, F1-Score = 0.84 với 95 mẫu.
- Độ chính xác trung bình (Accuracy) là 81%, Macro Average và Weighted Average đều đạt giá trị tương tự, cho thấy sự cân bằng trong dự đoán của mô hình.

```
He so chan Intercept: [0.63324623]
He So Hoi Quy Ung Voi Tung Dac Trung Coefficients: [[ 1.95775961 -0.68703695 -0.65640338 -1.16877642  1.44326084  0.82079419
  -1.03617265  0.59308204]]
----------------------------------
Accuracy: 0.8333333333333334
----------------------------------
Classification Report:
              precision    recall  f1-score   support

         0.0       0.83      0.81      0.82        85
         1.0       0.84      0.85      0.84        95

    accuracy                           0.83       180
   macro avg       0.83      0.83      0.83       180
weighted avg       0.83      0.83      0.83       180
```

Hình 4.5: Kết quả



Hình 4.6: Matrix confusion

Mô hình Logistic Regression đã được huấn luyện và đánh giá trên tập dữ liệu đã chuẩn bị. Kết quả đạt được cho thấy độ chính xác (Accuracy) của mô hình là **83%**, một mức hiệu suất tốt cho bài toán phân loại nhị phân. Báo cáo phân loại (Classification Report) chỉ ra rằng các chỉ số Precision, Recall, và F1-Score đều ở mức cân bằng giữa hai lớp (0.0 và 1.0), lần lượt là 0.83 và 0.84. Điều này chứng minh rằng mô hình có khả năng dự đoán hiệu quả và đồng đều giữa các nhãn.

Hệ số hồi quy của mô hình cho thấy tầm quan trọng của các đặc trưng đầu vào trong việc dự đoán nhãn. Đặc trưng có hệ số tuyệt đối lớn đóng góp nhiều hơn vào quyết định của mô hình. Kết quả này cũng chỉ ra rằng mô hình Logistic Regression phù hợp với dữ liệu đã giảm chiều và không gặp phải hiện tượng quá khớp.

Tuy nhiên, mô hình vẫn tồn tại một số hạn chế. Logistic Regression chỉ mô tả được mối quan hệ tuyến tính giữa các đặc trưng và nhãn, do đó, hiệu quả có thể giảm nếu dữ liệu có quan hệ phi tuyến tính. Ngoài ra, độ chính xác 83% cho thấy vẫn còn không gian để cải thiện. Đề xuất bao gồm thử nghiệm các mô hình phức tạp hơn hoặc tối ưu hóa thêm dữ liệu và tham số.

4.2 Mô hình Random Forest

4.2.1 Bài toán đặt ra

Dự án đặt ra bài toán dự đoán biến mục tiêu target dựa trên các đặc trưng được cung cấp trong bộ dữ liệu df_heart. Đây có thể là một bài toán phân loại như chẩn đoán bệnh, phân loại khách hàng, hoặc một tình huống tương tự. Mục tiêu là xây dựng một mô hình dự đoán có độ chính xác cao, hỗ trợ việc ra quyết định hiệu quả và đáng tin cậy.

4.2.2 Lý do chọn Random Forest

Mô hình Random Forest được lựa chọn do những ưu điểm vượt trội của nó. Thứ nhất, Random Forest hoạt động hiệu quả với các dữ liệu phức tạp, có mối quan hệ phi tuyến tính giữa các biến. Thứ hai, mô hình có khả năng giảm thiểu hiện tượng quá khớp (overfitting) nhờ sử dụng nhiều cây quyết định và trung bình hóa kết quả. Cuối cùng, Random Forest có tính ổn định cao và cung cấp công cụ đánh giá tầm quan trọng của từng đặc trưng, giúp tăng khả năng giải thích mô hình.

4.2.3 Quy trình xây dựng mô hình

Xử lý dữ liệu: Đầu tiên, dữ liệu được chuẩn bị bằng cách tách biến mục tiêu (target) ra khỏi tập đặc trưng đầu vào. Các đặc trưng độc lập (X) được tách riêng và dữ liệu được chia thành hai tập:

Tập huấn luyện (80%): Sử dụng để huấn luyện mô hình.

Tập kiểm tra (20%): Dùng để đánh giá hiệu năng của mô hình.

Phương pháp chia tách được thực hiện bằng train_test_split với tham số random_state=42 để đảm bảo khả năng tái lập.

```
X = df_heart.drop(columns='target', axis=1)
y = df_heart['target']
```

Hình 4.7: Tách biến

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Hình 4.8: Chia tập train, test

Huấn luyện mô hình: Thuật toán Random Forest Classifier được sử dụng để xây dựng mô hình dự đoán. Mô hình được huấn luyện trên tập huấn luyện (X_train, y_train) bằng phương pháp fit. Sau khi huấn luyện, mô hình đã sẵn sàng để thực hiện dự đoán.

```
# Huan luyen Model
model = RandomForestClassifier()
model.fit(X_train, y_train)
```

Hình 4.9: Huấn luyện mô hình

Đánh giá hiệu năng: Mô hình được đánh giá trên tập kiểm tra (X_test) thông qua các dự đoán (pred_y). Hiệu năng của mô hình được đo lường dựa trên các chỉ số sau: Các chỉ số như độ chính xác (Accuracy), Precision, Recall và F1-Score được tính toán để đo lường chất lượng dự đoán của mô hình.

```
# Du doan tren tap test
pred_y = model.predict(X_test)

acc = accuracy_score(y_test, pred_y)
# Danh gia mo hinh
print(f'Accuracy: {acc:.2f}')

print(f'Classification Report:\n', classification_report(y_test, pred_y))
```

Hình 4.10: Đánh giá hiệu năng

4.2.4 Kết quả và nhận xét

Mô hình Random Forest Classifier đã được triển khai và đánh giá với các kết quả cụ thể như sau:

Độ chính xác (Accuracy): 89.00%.

Báo cáo phân loại (Classification Report):

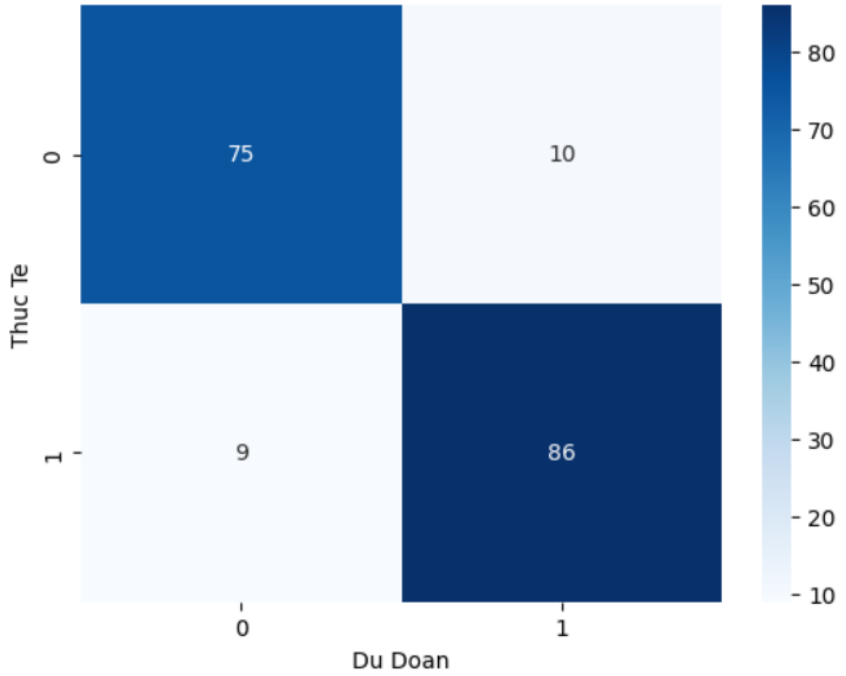
- Đối với nhãn 0.0: Precision = 0.89, Recall = 0.88, F1-Score = 0.89 với 85 mẫu.
- Đối với nhãn 1.0: Precision = 0.90, Recall = 0.91, F1-Score = 0.90 với 95 mẫu.
- Độ chính xác trung bình (Accuracy) là 89%, với Macro Average và Weighted Average đều đạt giá trị tương tự, cho thấy mô hình có hiệu năng đồng đều giữa các lớp.

```
Accuracy: 0.89
Classification Report:
              precision    recall  f1-score   support

         0.0       0.89      0.88      0.89        85
         1.0       0.90      0.91      0.90        95

    accuracy                           0.89       180
   macro avg       0.89      0.89      0.89       180
weighted avg       0.89      0.89      0.89       180
```

Hình 4.11: Kết quả



Hình 4.12: Matrix confusion

Mô hình Random Forest Classifier đã được huấn luyện và đánh giá trên tập dữ liệu đã chuẩn bị. Kết quả đạt được cho thấy độ chính xác (Accuracy) của mô hình là 89.00%, một mức hiệu suất cao cho bài toán phân loại nhị phân. Báo cáo phân loại (Classification Report) chỉ ra rằng các chỉ số Precision, Recall và F1-Score đều ở mức cân bằng giữa hai lớp (0.0 và 1.0), lần lượt là 0.89 và 0.90. Điều này chứng minh rằng mô hình có khả năng dự đoán hiệu quả và đồng đều giữa các nhãn.

Cấu trúc của mô hình Random Forest cho phép nó kết hợp thông tin từ nhiều cây quyết định, giúp giảm thiểu hiện tượng quá khớp và cải thiện độ chính xác tổng thể. Kết quả cũng cho thấy mô hình hoạt động tốt trong việc phân biệt giữa hai lớp với tỷ lệ dự đoán chính xác cao.

Tuy nhiên, mô hình vẫn tồn tại một số hạn chế. Mặc dù độ chính xác đạt 89.00%, vẫn còn không gian để cải thiện, đặc biệt khi ứng dụng trong các hệ thống yêu cầu hiệu suất cao hơn. Đề xuất bao gồm việc tinh chỉnh siêu tham số của Random Forest, tăng cường dữ liệu huấn luyện, hoặc thử nghiệm các mô hình khác như Gradient Boosting hoặc XGBoost để so sánh và tối ưu hóa kết quả.

4.3 Phân tích và đánh giá hai mô hình: Logistic Regression vs Random Forest

Tiêu chí	Logistic Regression	Random Forest Classifier
Loại mô hình	Hồi quy tuyến tính	Mô hình ensemble sử dụng nhiều cây quyết định
Độ chính xác (Accuracy)	83%	89.00%
Khả năng xử lý dữ liệu phi tuyến	Hạn chế, chỉ mô tả được mối quan hệ tuyến tính giữa đặc trưng và nhãn	Tốt, có khả năng mô hình hóa mối quan hệ phi tuyến tính phức tạp
Hiệu năng giữa các lớp	Cân bằng với Precision, Recall, F1-Score ở mức 0.83 - 0.84	Cân bằng với Precision, Recall, F1-Score ở mức 0.89 - 0.90
Ưu điểm	Dễ triển khai, giải thích đơn giản	Hiệu năng cao hơn, giảm thiểu quá khớp nhờ kết

	Nhanh khi xử lý dữ liệu lớn	hợp nhiều cây Hỗ trợ xử lý mối quan hệ phi tuyến giữa các đặc trưng
Hạn chế	Không phù hợp với dữ liệu phi tuyến tính Nhạy cảm với đa cộng tuyến	Yêu cầu tài nguyên tính toán lớn hơn Khó giải thích hơn so với Logistic Regression
Tính mở rộng	Tốt, ít phụ thuộc vào tài nguyên	Trung bình, phụ thuộc vào số lượng cây và độ sâu
Ứng dụng phù hợp	Dữ liệu nhỏ hoặc có mối quan hệ tuyến tính rõ ràng	Dữ liệu lớn, phức tạp hoặc có mối quan hệ phi tuyến tính

Bảng 4.1: Bảng phân tích và đánh giá mô hình.

Nhận xét:

- **Hiệu năng:** Mô hình Random Forest Classifier đạt độ chính xác cao hơn Logistic Regression (89.00% so với 83%) và có khả năng xử lý tốt hơn khi dữ liệu có mối quan hệ phi tuyến. Các chỉ số Precision, Recall và F1-Score của Random Forest cũng vượt trội hơn, cho thấy hiệu suất dự đoán đồng đều và ổn định giữa các lớp.
- **Tính đơn giản và giải thích:** Logistic Regression có ưu điểm về mặt đơn giản và dễ giải thích, phù hợp cho các bài toán cần diễn giải mối quan hệ giữa đặc trưng và nhãn mục tiêu. Tuy nhiên, hiệu năng của Logistic Regression bị giới hạn bởi giả định tuyến tính giữa các đặc trưng và nhãn.

- **Tài nguyên:** Random Forest yêu cầu nhiều tài nguyên tính toán hơn và khó giải thích hơn so với Logistic Regression, nhưng hiệu năng vượt trội khiến nó trở thành lựa chọn tốt hơn cho dữ liệu phức tạp.

Đề xuất:

- **Chọn Random Forest Classifier** làm mô hình chính để giải quyết bài toán hiện tại, vì nó đạt hiệu năng cao và có khả năng xử lý dữ liệu phi tuyến tính.
- **Tối ưu hóa Random Forest:** Sử dụng các kỹ thuật như GridSearchCV hoặc RandomizedSearchCV để tinh chỉnh siêu tham số (số lượng cây, độ sâu tối đa, v.v.), nhằm đạt hiệu suất tốt nhất.
- **Logistic Regression như mô hình nền tảng:** Nếu cần một mô hình đơn giản và dễ giải thích hơn, Logistic Regression vẫn là một lựa chọn thay thế phù hợp, đặc biệt khi yêu cầu tài nguyên hạn chế.
- **Phân tích thêm dữ liệu:** Cân nhắc kiểm tra và cải thiện chất lượng dữ liệu, thử nghiệm thêm các đặc trưng hoặc tăng cường dữ liệu để cải thiện hiệu năng của cả hai mô hình.

CHƯƠNG 5 KẾT LUẬN

Nghiên cứu này tập trung vào việc tiền xử lý dữ liệu và xây dựng hai mô hình phân loại nhị phân: Logistic Regression và Random Forest Classifier. Đầu tiên, quá trình tiền xử lý dữ liệu đã được thực hiện để chuẩn bị dữ liệu sẵn sàng cho mô hình, bao gồm việc xử lý các giá trị thiếu, mã hóa các biến phân loại, và chuẩn hóa các đặc trưng. Sau khi dữ liệu được tiền xử lý, chúng tôi tiến hành xây dựng và huấn luyện hai mô hình phân loại. Kết quả thực nghiệm cho thấy mô hình **Random Forest Classifier** đạt độ chính xác cao hơn, với 89.00%, cùng các chỉ số Precision, Recall và F1-Score ổn định giữa các lớp. Ngược lại, **Logistic Regression** đạt độ chính xác 83%, cho thấy mô hình này bị giới hạn khi xử lý dữ liệu phi tuyến tính.

Mô hình Random Forest Classifier, nhờ vào khả năng xử lý các mối quan hệ phi tuyến và sử dụng kết hợp nhiều cây quyết định, cho thấy hiệu suất vượt trội. Trong khi đó, Logistic Regression, mặc dù đơn giản và dễ giải thích, lại có sự hạn chế trong việc dự đoán các mối quan hệ phức tạp giữa các đặc trưng. Quá trình tiền xử lý dữ liệu đã giúp chuẩn hóa và làm sạch dữ liệu, tạo điều kiện tốt cho việc huấn luyện mô hình.

Đóng góp của nghiên cứu là xây dựng và triển khai thành công hai mô hình phân loại nhị phân, đồng thời thực hiện một quy trình tiền xử lý dữ liệu chi tiết và có hệ thống. Dựa trên kết quả đánh giá, mô hình Random Forest Classifier được xác định là phù hợp hơn cho bài toán phân loại này. Tuy nhiên, một số hạn chế của nghiên cứu vẫn tồn tại, bao gồm việc dữ liệu có thể chưa đủ lớn để đảm bảo tính tổng quát và một số yếu tố trong quá trình tiền xử lý chưa được tối ưu hoàn toàn.

Trong tương lai, nghiên cứu có thể mở rộng bằng cách thử nghiệm với các thuật toán phân loại khác như Gradient Boosting hoặc XGBoost, đồng thời tối ưu hóa thêm các bước tiền xử lý và mô hình. Việc thu thập và sử

dụng dữ liệu đa dạng hơn sẽ giúp nâng cao khả năng tổng quát của mô hình và ứng dụng trong các tình huống thực tế.

TÀI LIỆU THAM KHẢO

[1]"Hiểu và cách xử lý dữ liệu bị thiếu," unitrain.edu.vn. Accessed: Dec. 12, 2024. [Online]. Available: https://unitrain.edu.vn/hieu-va-cach-xu-ly-du-lieu-bi-thieu/

[2]"Xử lý missing data trong Data analysis." Accessed: Dec. 12, 2024. [Online]. Available: https://viblo.asia/p/xu-ly-missing-data-trong-data-analysis-maGK7qaAlj2

[3]"Nguyên nhân gây mất dữ liệu và khả năng khôi phục dữ liệu trong từng trường hợp," IT-CARE. Accessed: Dec. 12, 2024. [Online]. Available: https://edata.it-care.vn/nguyen-nhan-gay-mat-du-lieu-va-kha-nang-khoi-phuc-du-lieu-trong-tung-truong-hop-45

[4]"Dữ liệu ngoại lai (outlier) là gì và xử lý những dữ liệu này như nào trong phân tích?," Tomorrow Marketers. Accessed: Dec. 12, 2024. [Online]. Available: https://blog.tomorrowmarketers.org/du-lieu-ngoai-lai-outlier/

[5]"Xử lý các giá trị ngoại lệ — Machine Learning cho dữ liệu dạng bảng." Accessed: Dec. 12, 2024. [Online]. Available: https://machinelearningcoban.com/tabml_book/ch_data_processing/process_outliers.html

[6]"Data Cleaning là gì? Các lợi ích khi sử dụng data cleaning." Accessed: Dec. 30, 2024. [Online]. Available: https://mcivietnam.com/blog-detail/data-cleaning-la-gi-cac-loi-ich-khi-su-dung-data-cleaning-X5X77V/

[7]trituenhantao.io, "Bài 6: Logistic Regression (Hồi quy Logistic)," Trí tuệ nhân tạo. Accessed: Dec. 30, 2024. [Online]. Available: https://trituenhantao.io/machine-learning-co-ban/bai-6-logistic-regression-hoi-quy-logistic/

[8]admin01, "Logistic Regression là gì? Ví dụ bài toán Logistic Regression in Python," Trang thông tin chuyên sâu về chuyển đổi số. Accessed: Dec. 30, 2024. [Online]. Available: https://cole.edu.vn/logistic-regression/

[9]admin, "Thuật toán Random Forest: Giải thích chi tiết và ứng dụng," AIcandy. Accessed: Dec. 30, 2024. [Online]. Available: https://aicandy.vn/thuat-toan-random-forest-giai-thich-chi-tiet-va-ung-dung/

[10] "Random Forest algorithm — Machine Learning cho dữ liệu dạng bảng." Accessed: Dec. 30, 2024. [Online]. Available:

https://machinelearningcoban.com/tabml_book/ch_model/random_forest.html