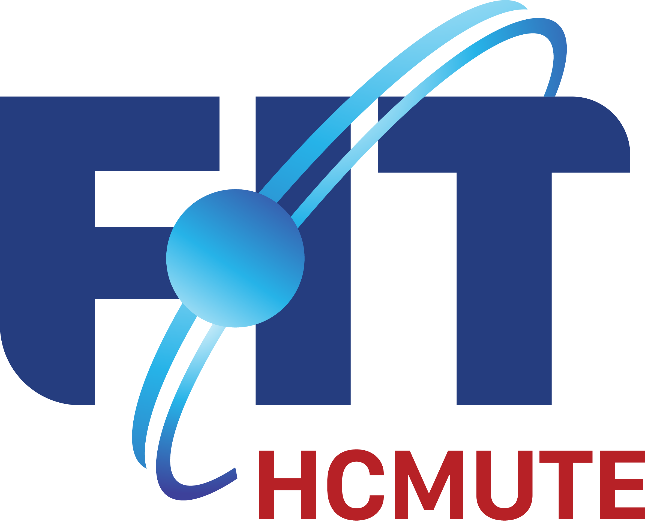
**Trường đại học sư phạm kỹ thuật TP.HCM**

**Khoa Công Nghệ Thông Tin**

**Bộ môn trí tuệ nhân tạo**



**Hồng Anh khoa – 22110351**

**trịnh tấn hào – 22110315**

**trần vũ khanh – 22110349**

Đề Tài :

**mô hình dự đoán bệnh tim sử dụng tập dữ liệu tổng hợp**

**Giáo viên hướng dẫn**

**ThS. Lê Minh Tân**

**Khóa 2022 – 2026**

|  |  |
| --- | --- |
| **ĐH SƯ PHẠM KỸ THUẬT TP.HCM**  **KHOA CNTT**  \*\*\*\*\*\*\* | **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc Độc lập – Tự do – Hạnh Phúc**  \*\*\*\*\*\*\* |

**PHIẾU NHẬN XÉT CỦA GIÁO VIÊN**

Họ và tên Sinh viên 1 : Hồng Anh Khoa MSSV: 22110351

Họ và tên Sinh viên 2 : Trịnh Tấn Hào MSSV: 22110315

Họ và tên Sinh viên 3 : Trần Vũ Khanh MSSV: 22110349

Ngành: Công nghệ Thông tin

Tên đề tài:

Họ và tên Giáo viên hướng dẫn:

**NHẬN XÉT**

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

2. Ưu điểm:

3. Khuyết điểm

4. Điểm :

Tp*. Hồ Chí Minh, ngày tháng năm 20*

Giáo viên hướng dẫn

*(Ký & ghi rõ họ tên)*

lời cảm ơn

Tôi xin gửi lời cảm ơn chân thành đến thầy Lê Minh Tân, người đã trực tiếp giảng dạy và hướng dẫn tôi trong môn Khoa học Dữ liệu. Những bài giảng của thầy/cô không chỉ cung cấp kiến thức chuyên môn mà còn giúp tôi hiểu rõ hơn về các khái niệm, phương pháp và ứng dụng thực tiễn của khoa học dữ liệu. Sự tận tâm, những ví dụ thực tế và cách truyền đạt rõ ràng của thầy/cô đã giúp tôi tiếp thu bài học một cách hiệu quả, từ đó xây dựng nền tảng vững chắc để thực hiện báo cáo cuối kỳ này.

Đặc biệt, tôi xin cảm ơn thầy vì đã dành thời gian hướng dẫn, đưa ra những góp ý chi tiết và định hướng cụ thể trong quá trình tôi chuẩn bị và hoàn thiện báo cáo. Những ý kiến quý báu của thầy/cô đã giúp tôi cải thiện nội dung, trình bày vấn đề một cách rõ ràng hơn và áp dụng kiến thức từ các bài học trên lớp vào báo cáo một cách phù hợp.

Báo cáo này là kết quả của quá trình học tập và nỗ lực áp dụng những gì tôi đã được học trong môn Nhập môn Khoa học Dữ liệu. Tuy nhiên, do hạn chế về kinh nghiệm và thời gian, báo cáo có thể vẫn còn nhiều thiếu sót. Tôi rất mong nhận được những ý kiến đóng góp và nhận xét từ thầy để có thể hoàn thiện hơn trong tương lai.

Một lần nữa, tôi xin chân thành cảm ơn thầy/cô vì sự hỗ trợ và những bài học quý giá đã giúp tôi hoàn thành báo cáo này.

**MỤC LỤC**

[PHẦN 1: MỞ ĐẦU 1](#_Toc197546669)

[1.1. Tính cấp thiết của đề tài 1](#_Toc197546670)

[1.2. Mục đích đề tài 1](#_Toc197546671)

[1.3. Cách tiếp cận và phương pháp nghiên cứu 1](#_Toc197546672)

[PHẦN 2 : NỘI DUNG 2](#_Toc197546673)

[Chương 1: Cơ sở lý thuyết và tổng quan 2](#_Toc197546674)

[1.1. Tổng quan về bệnh tim 2](#_Toc197546675)

[1.2. Tổng quan về học máy trong y khoa 2](#_Toc197546676)

[1.3. Giới thiệu bộ dữ liệu. 3](#_Toc197546677)

[1.3.1. Phân tích Bộ dữ liệu Heart Failure Prediction Dataset (Fedesoriano): 3](#_Toc197546678)

[1.3.2. Khám phá Chi tiết về Bộ dữ liệu Bệnh tim (David Lapp): 4](#_Toc197546679)

[1.3.1. Nghiên cứu về Bộ dữ liệu Dự đoán Bệnh tim (Rashad Mammadov): 5](#_Toc197546680)

[Chương 2 : Phân tích, thống kê dữ liệu sau khi xử lý. 7](#_Toc197546681)

[2.1. Thông tin cơ bản: 7](#_Toc197546682)

[2.2. Phân tích các đặc trưng: 8](#_Toc197546683)

[Chương 3: Cơ sở lý thuyết các thuật toán 15](#_Toc197546684)

[3.1. Các thuật toán học máy 15](#_Toc197546685)

[3.1.1. Multi-Layer Perceptron (MLP) 15](#_Toc197546686)

[3.1.2. Random Forest 16](#_Toc197546687)

[3.1.3. Stacking Classifier 18](#_Toc197546688)

[3.1.4. Logistic Regression 19](#_Toc197546689)

[3.2. Các kỹ thuật hỗ trợ 20](#_Toc197546690)

[3.2.1. GridSearchCV 20](#_Toc197546691)

[3.2.2. SMOTE (Synthetic Minority Oversampling Technique) 21](#_Toc197546692)

[3.2.3. Permutation Importance 22](#_Toc197546693)

[Chương 4 : Thực hiện mô hình học máy để dự đoán 23](#_Toc197546694)

[Chương 5: Đánh giá trực quan hóa cho mô hình 27](#_Toc197546695)

[5.1. Đánh giá số học 27](#_Toc197546696)

[5.1.1. F1 Score 27](#_Toc197546697)

[5.1.2. Classification Report 28](#_Toc197546698)

[5.2. Đánh giá trực quan 29](#_Toc197546699)

[5.2.1. Confusion Matrix 29](#_Toc197546700)

[5.2.2. ROC Curve 29](#_Toc197546701)

[5.2.3. Learning Curve 30](#_Toc197546702)

[5.2.4. Feature Importance 30](#_Toc197546703)

[5.3. So sánh và phân tích tổng quát 31](#_Toc197546704)

[5.4. Đánh giá ưu nhược điểm của mô hình 31](#_Toc197546705)

[5.4.1. Ưu điểm 31](#_Toc197546706)

[5.4.2. Nhược điểm 32](#_Toc197546707)

[5.5. Hướng cải thiện 34](#_Toc197546708)

[5.5.1. Giảm False Negative 34](#_Toc197546709)

[5.5.2. Tăng tính tổng quát hóa 34](#_Toc197546710)

[5.5.3. Giảm độ phức tạp và tăng tính diễn giải 34](#_Toc197546711)

[5.5.4. Xử lý dữ liệu thực tế 35](#_Toc197546712)

[5.5.5. Thử nghiệm các mô hình khác 35](#_Toc197546713)

[5.5.6. Đánh giá lâm sàng 35](#_Toc197546714)

[Phần 3 : Kết luận 36](#_Toc197546715)

[TÀI LIỆU THAM KHẢO 37](#_Toc197546716)

Phần 1: mở đầu

## Tính cấp thiết của đề tài

Bệnh tim mạch là một trong những nguyên nhân gây tử vong hàng đầu trên toàn cầu. Theo thống kê của Tổ chức Y tế Thế giới (WHO), mỗi năm có khoảng 17,9 triệu người tử vong vì các bệnh liên quan đến tim mạch, chiếm 32% tổng số ca tử vong toàn cầu. Đặc biệt, các bệnh lý tim mạch thường diễn tiến âm thầm, khó phát hiện ở giai đoạn đầu, dẫn đến việc điều trị chậm trễ và gây hậu quả nghiêm trọng.

Trong bối cảnh đó, việc phát triển các công cụ hỗ trợ chẩn đoán sớm và chính xác bệnh tim là vô cùng cần thiết. Các hệ thống dựa trên trí tuệ nhân tạo (AI) và học máy (machine learning) ngày càng chứng tỏ được tiềm năng mạnh mẽ trong việc phân tích dữ liệu y tế phức tạp, từ đó hỗ trợ đưa ra các cảnh báo hoặc dự đoán bệnh lý nhanh chóng, khách quan và đáng tin cậy. Việc áp dụng các thuật toán học máy vào dự đoán bệnh tim không chỉ giúp giảm tải cho đội ngũ y tế mà còn góp phần nâng cao hiệu quả trong công tác phòng ngừa và điều trị bệnh.

* 1. **Mục đích đề tài**
* Xây dựng mô hình dự đoán bệnh tim dựa trên dữ liệu y khoa với độ chính xác cao.
* So sánh hiệu quả của các thuật toán học máy khác nhau trong việc dự đoán bệnh tim.
* Góp phần hỗ trợ các bác sĩ trong công tác sàng lọc và chẩn đoán sớm bệnh lý tim mạch.
* Tối ưu hóa quy trình phát hiện bệnh, từ đó giảm thiểu chi phí điều trị và tỷ lệ tử vong.
  1. **Cách tiếp cận và phương pháp nghiên cứu**

**Phương pháp tiếp cận**: Sử dụng học máy để xây dựng mô hình dự đoán trên dữ liệu y tế đã được thu thập và xử lý.

**Các bước chính**:

Thu thập và phân tích bộ dữ liệu bệnh tim

Tiền xử lý dữ liệu: xử lý giá trị thiếu, chuẩn hóa dữ liệu, mã hóa thuộc tính.

Áp dụng và huấn luyện các mô hình học máy như Random Forest, Support Vector Machine (SVM), Neural Network.

Đánh giá và so sánh kết quả mô hình qua các chỉ số: Accuracy, Precision, Recall, F1-Score, ROC-AUC.

**Đối tượng nghiên cứu**: Dữ liệu gồm các thuộc tính y khoa như tuổi, giới tính, huyết áp, cholesterol, nhịp tim, đường máu, triệu chứng đau thắt ngực, v.v.

**Phạm vi nghiên cứu**: Tập trung vào việc áp dụng mô hình học máy trên tập dữ liệu có sẵn; chưa triển khai trong thực tế lâm sàng nhưng hướng tới ứng dụng thực tiễn.

# PHẦN 2 : NỘI DUNG

# Chương 1: Cơ sở lý thuyết và tổng quan

## 1.1. Tổng quan về bệnh tim

**Định nghĩa**: Bệnh tim là thuật ngữ chung chỉ các bệnh lý ảnh hưởng đến cấu trúc và chức năng của tim, bao gồm bệnh động mạch vành, suy tim, nhồi máu cơ tim, rối loạn nhịp tim, và các bệnh van tim.

**Các loại bệnh tim phổ biến**:

* Bệnh mạch vành (Coronary Artery Disease)
* Suy tim (Heart Failure)
* Loạn nhịp tim (Arrhythmia)
* Bệnh van tim (Valvular Heart Disease)

**Yếu tố nguy cơ**:

* + **Di truyền**: Tiền sử gia đình có người mắc bệnh tim.
  + **Lối sống**: Hút thuốc, ăn uống không lành mạnh, ít vận động, stress kéo dài.
  + **Bệnh lý nền**: Tăng huyết áp, đái tháo đường, béo phì, rối loạn lipid máu.

## 1.2. Tổng quan về học máy trong y khoa

**Định nghĩa học máy**: Là một nhánh của trí tuệ nhân tạo, học máy cho phép máy tính học từ dữ liệu để thực hiện các nhiệm vụ mà không cần lập trình rõ ràng.

**Ứng dụng trong y khoa**:

* Dự đoán khả năng mắc bệnh.
* Phân loại hình ảnh y khoa (X-quang, MRI).
* Phân tích dữ liệu gen và xét nghiệm.
* Hỗ trợ chẩn đoán, kê đơn và cá nhân hóa điều trị.

**Các thuật toán phổ biến trong dự đoán bệnh tim**:

* **Logistic Regression**: Phân loại nhị phân, dễ triển khai và giải thích.
* **Decision Tree / Random Forest**: Cung cấp cây quyết định dễ hiểu, hiệu quả trên dữ liệu phi tuyến tính.
* **SVM (Support Vector Machine)**: Tốt cho dữ liệu có chiều cao.
* **Neural Network**: Mạnh mẽ với dữ liệu phức tạp, khả năng học sâu (deep learning).

**Lý do áp dụng học máy cho bệnh tim**:

* Dữ liệu y tế ngày càng phong phú và đa chiều.
* Các mô hình học máy có thể phát hiện các mẫu tiềm ẩn trong dữ liệu mà con người khó nhận ra.
* Giảm sai sót trong chẩn đoán và tăng tính khách quan.

## 1.3. Giới thiệu bộ dữ liệu.

* + 1. **Phân tích Bộ dữ liệu Heart Failure Prediction Dataset (Fedesoriano):**

**Nguồn gốc và Bối cảnh của Bộ dữ liệu:**

* **Nguồn trên Kaggle:** Bộ dữ liệu "Heart Failure Prediction Dataset" có sẵn trên Kaggle dưới tên người dùng "Fedesoriano".
* **Ngày tạo:** Bộ dữ liệu được tạo vào tháng 9 năm 2021.
* **Liên kết:** <https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction>
* **Kết hợp các Bộ dữ liệu:** Bộ dữ liệu này độc đáo vì nó kết hợp năm bộ dữ liệu bệnh tim khác nhau, trước đây độc lập (Cleveland, Hungary, Switzerland, Long Beach VA và Stalog (Heart) Data Set).
* **Số lượng Quan sát:** Bộ dữ liệu kết hợp ban đầu có 1190 quan sát, đã giảm xuống còn 918 sau khi loại bỏ 272 bản sao.
* **Biến mục tiêu:** Heart disease: 1 (bệnh tim), 0 (bình thường)
* **Kho lưu trữ Nguồn:** Các bộ dữ liệu gốc đến từ Kho lưu trữ Học máy UCI. <https://archive.ics.uci.edu/dataset/45/heart+disease>

**Mô tả Chi tiết về các Biến:**

Bộ dữ liệu chứa 11 đặc trưng (biến đầu vào) và 1 lớp đầu ra (biến mục tiêu). Các biến này đại diện cho các đặc điểm lâm sàng và nhân khẩu học khác nhau của bệnh nhân.

Dữ liệu bao gồm các biến liên quan đến bệnh tim của bệnh nhân, cụ thể như sau:

• Tuổi (Age): Biến liên tục, đo bằng năm.

• Giới tính (Sex): Biến phân loại nhị phân (M: Nam, F: Nữ).

• Loại đau ngực (ChestPainType): Biến phân loại danh nghĩa với 4 loại (TA: Đau thắt ngực điển hình(1), ATA: Không điển hình(2), NAP: Không do đau thắt ngực(3), ASY: Không triệu chứng(0)).

• Huyết áp lúc nghỉ (RestingBP): Biến liên tục, đo bằng mm Hg.

• Mức cholesterol (Cholesterol): Biến liên tục, đo bằng mg/dl.

• Mức đường huyết lúc đói (FastingBS): Biến nhị phân (1: >120 mg/dl, 0: ngược lại).

• Kết quả điện tâm đồ lúc nghỉ (RestingECG): Biến phân loại danh nghĩa với 3 loại (Normal: Bình thường, ST: Bất thường sóng ST-T, LVH: Phì đại thất trái).

• Nhịp tim tối đa đạt được (MaxHR): Biến liên tục, giá trị từ 60 đến 202.

• Đau thắt ngực do gắng sức (ExerciseAngina): Biến nhị phân (Y: Có, N: Không).

• ST chênh xuống do gắng sức (Oldpeak): Biến liên tục.

• Độ dốc đoạn ST đỉnh gắng sức (ST\_Slope): Biến phân loại thứ tự với 3 mức độ (Up: Dốc lên, Flat: Nằm ngang, Down: Dốc xuống).

• Biến mục tiêu (HeartDisease): Xác định sự hiện diện của bệnh tim (1: Bệnh, 0: Bình thường).

### 1.3.2. Khám phá Chi tiết về Bộ dữ liệu Bệnh tim (David Lapp):

**Bối cảnh Bộ dữ liệu:**

* **Nguồn trên Kaggle:** Bộ dữ liệu "Heart Disease Dataset" có sẵn trên Kaggle dưới tên người dùng "**David Lapp**".
* **Liên kết:** <https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset>
* **Nguồn gốc Lịch sử:** Bộ dữ liệu này được cập nhật trên Kaggle năm 2019 và có ý nghĩa lịch sử vì nó có từ năm 1988 và là một tập hợp dữ liệu từ bốn địa điểm địa lý khác nhau: Cleveland, Hungary, Switzerland và Long Beach V.
* **Số lượng Thuộc tính:** Bộ dữ liệu gốc chứa một tập hợp toàn diện gồm 76 thuộc tính. Tuy nhiên, do hầu hết các nghiên cứu tập trung vào một tập hợp có thể quản lý được, nên tất cả các thí nghiệm đã công bố thường đề cập đến việc sử dụng một tập con gồm 14 thuộc tính chính, bao gồm cả thuộc tính được dự đoán.
* **Biến Mục tiêu:** Trường "target" cho biết sự hiện diện của bệnh tim ở bệnh nhân 1 (bệnh tim), 0 (bình thường).
* **Nhận định:** Bộ dữ liệu này, mặc dù đã cũ, vẫn là một nguồn tài nguyên có giá trị do các nghiên cứu sâu rộng đã được thực hiện trên tập con gồm 14 thuộc tính được xác định rõ ràng của nó.

**Mô tả Chi tiết về các Biến:**

Phiên bản thường được sử dụng của bộ dữ liệu này bao gồm 14 thuộc tính, bao gồm các phép đo lâm sàng và đặc điểm bệnh nhân khác nhau, trong đó một thuộc tính là biến mục tiêu quan trọng.

Dữ liệu bao gồm các biến liên quan đến bệnh tim của bệnh nhân, cụ thể như sau:

* **Tuổi (age):** Biến liên tục, đo bằng năm.
* **Giới tính (sex):** Biến nhị phân (1: Nam, 0: Nữ).
* **Loại đau ngực (cp):** Biến phân loại với 4 mức độ.
* **Huyết áp lúc nghỉ (trestbps):** Biến liên tục, đo bằng mm Hg.
* **Mức cholesterol (chol):** Biến liên tục, đo bằng mg/dl.
* **Mức đường huyết lúc đói (fbs):** Biến nhị phân (1: >120 mg/dl, 0: ngược lại).
* **Kết quả điện tâm đồ lúc nghỉ (restecg):** Biến phân loại với 3 mức độ.
* **Nhịp tim tối đa đạt được (thalach):** Biến liên tục.
* **Đau thắt ngực do gắng sức (exang):** Biến nhị phân (1: Có, 0: Không).
* **ST chênh xuống do gắng sức (oldpeak):** Biến liên tục.
* **Độ dốc đoạn ST đỉnh gắng sức (slope):** Biến phân loại với 3 mức độ.
* **Số mạch máu chính được tô màu (ca):** Biến rời rạc (0-3).
* **Tình trạng thalassemia (thal):** Biến phân loại với 3 mức độ.
* **Biến mục tiêu (target):** Xác định bệnh tim (1: Bệnh, 0: Không bệnh).

### Nghiên cứu về Bộ dữ liệu Dự đoán Bệnh tim (Rashad Mammadov):

**Tổng quan về Bộ dữ liệu:**

* **Nguồn trên Kaggle:** Bộ dữ liệu "Heart Disease Prediction Dataset" có sẵn để sử dụng công khai trên Kaggle dưới tên người dùng "Rashad Mammadov".
* **Ngày tạo:** Bộ dữ liệu được tạo vào tháng 6 năm 2024.
* **Liên kết:** <https://www.kaggle.com/datasets/rashadrmammadov/heart-disease-prediction>
* **Mô tả Chung:** Bộ dữ liệu này cung cấp một tập hợp thông tin liên quan đến các cá nhân và các yếu tố nguy cơ khác nhau góp phần vào sự phát triển của bệnh tim. Nó bao gồm các chi tiết nhân khẩu học, tiền sử bệnh, lựa chọn lối sống và các triệu chứng được báo cáo liên quan đến bệnh tim.
* **Biến Mục tiêu:** Biến mục tiêu quan trọng trong bộ dữ liệu này được gắn nhãn "Heart Disease" và nó đóng vai trò là một chỉ số cho biết liệu một cá nhân đã được chẩn đoán mắc bệnh tim (giá trị 1) hay chưa (giá trị 0).
* **Nhận định:** Bộ dữ liệu này cung cấp một phạm vi các yếu tố nguy cơ tiềm ẩn rộng hơn, đặc biệt bao gồm các lựa chọn lối sống như hút thuốc và uống rượu, điều này có thể cung cấp các quan điểm và hiểu biết khác nhau so với hai bộ dữ liệu đầu tiên tập trung nhiều hơn vào các phép đo lâm sàng.

**Mô tả Chi tiết về các Biến:**

Bộ dữ liệu này bao gồm tổng cộng 16 biến, cung cấp một tập hợp các đặc trưng phong phú cho việc phân tích và mô hình hóa dự đoán .

Dữ liệu bao gồm các biến sức khỏe của cá nhân, cụ thể như sau:

* **Tuổi (Age):** Biến liên tục, đo bằng năm.
* **Giới tính (Gender):** Biến phân loại (Nam, Nữ).
* **Mức cholesterol (Cholesterol):** Biến liên tục, đo bằng mg/dL.
* **Huyết áp (Blood Pressure):** Biến liên tục, đo bằng mmHg.
* **Nhịp tim (Heart Rate):** Biến liên tục, đo bằng nhịp mỗi phút (bpm).
* **Tình trạng hút thuốc (Smoking):** Biến phân loại với 3 mức độ (Chưa bao giờ, Đã từng, Hiện tại).
* **Tần suất uống rượu (Alcohol Intake):** Biến phân loại với 3 mức độ (Không, Vừa phải, Nhiều).
* **Số giờ tập thể dục (Exercise Hours):** Biến liên tục, đo bằng số giờ mỗi tuần.
* **Tiền sử gia đình mắc bệnh tim (Family History):** Biến nhị phân (Có, Không).
* **Chẩn đoán tiểu đường (Diabetes):** Biến nhị phân (Có, Không).
* **Béo phì (Obesity):** Biến nhị phân (Có, Không).
* **Mức độ căng thẳng (Stress Level):** Biến số rời rạc, đo trên thang điểm từ 1 đến 10.
* **Mức đường huyết (Blood Sugar):** Biến liên tục, đo bằng mg/dL.
* **Đau thắt ngực do gắng sức (Exercise Induced Angina):** Biến nhị phân (Có, Không).
* **Loại đau ngực (Chest Pain Type):** Biến phân loại với 4 mức độ (Đau thắt ngực điển hình, Không điển hình, Không do đau thắt ngực, Không triệu chứng).
* **Bệnh tim (Heart Disease - Target):** Biến mục tiêu xác định sự hiện diện của bệnh tim (0: Không, 1: Có).

# Chương 2 : Phân tích, thống kê dữ liệu sau khi xử lý.

Bộ dữ liệu này được xử lý bằng cách xử lý các bộ dữ liệu riêng biệt về 1 mẫu chung sau đó hợp lại thành 1 bộ hoàn chỉnh.

Các bước phân tích cơ bản:  
1. Đọc dữ liệu từ các file

2. Loại bỏ 2 cột không cần thiết ‘ca’, ‘thal’ của tập “**David Lapp”.** Kết hợp 2 tập “**Fedesoriano**” và “**David Lapp**” đồng bộ tên cột theo bộ “**David Lapp”.**

3. Xử lý dữ liệu thiếu của tập “**Rashad Mammadov**” bằng phương pháp KNN Imputer

4. Chuẩn hóa và đồng bộ 3 tập dữ liệu.

5. Kết hợp tập dữ liệu “**Rashad Mammadov”** với tập đã kết hợp trước đó.

6. Kiểm tra giá trị trùng lập.

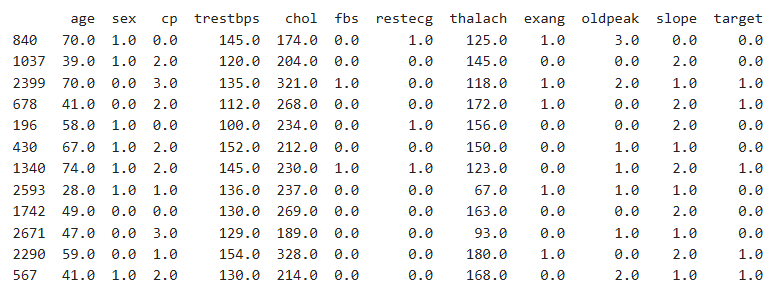
7. lấy 2000 mẫu ngẫu nhiên tạp thành dataset

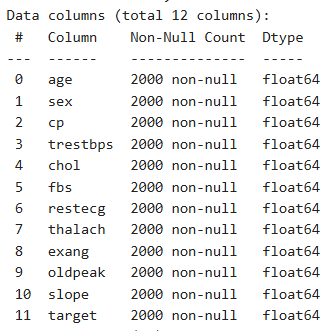
**2.1. Thông tin cơ bản:**

- Số lượng mẫu: 2000

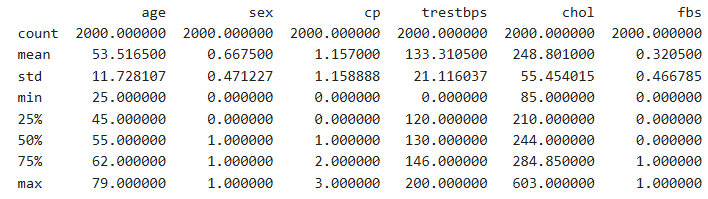
- Số đặc trưng: 12

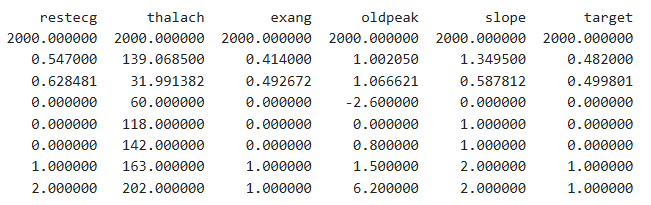
**\* Đọc 12 hàng đầu tiên:**





**\* Thống kê mô tả (đặc trưng số):**

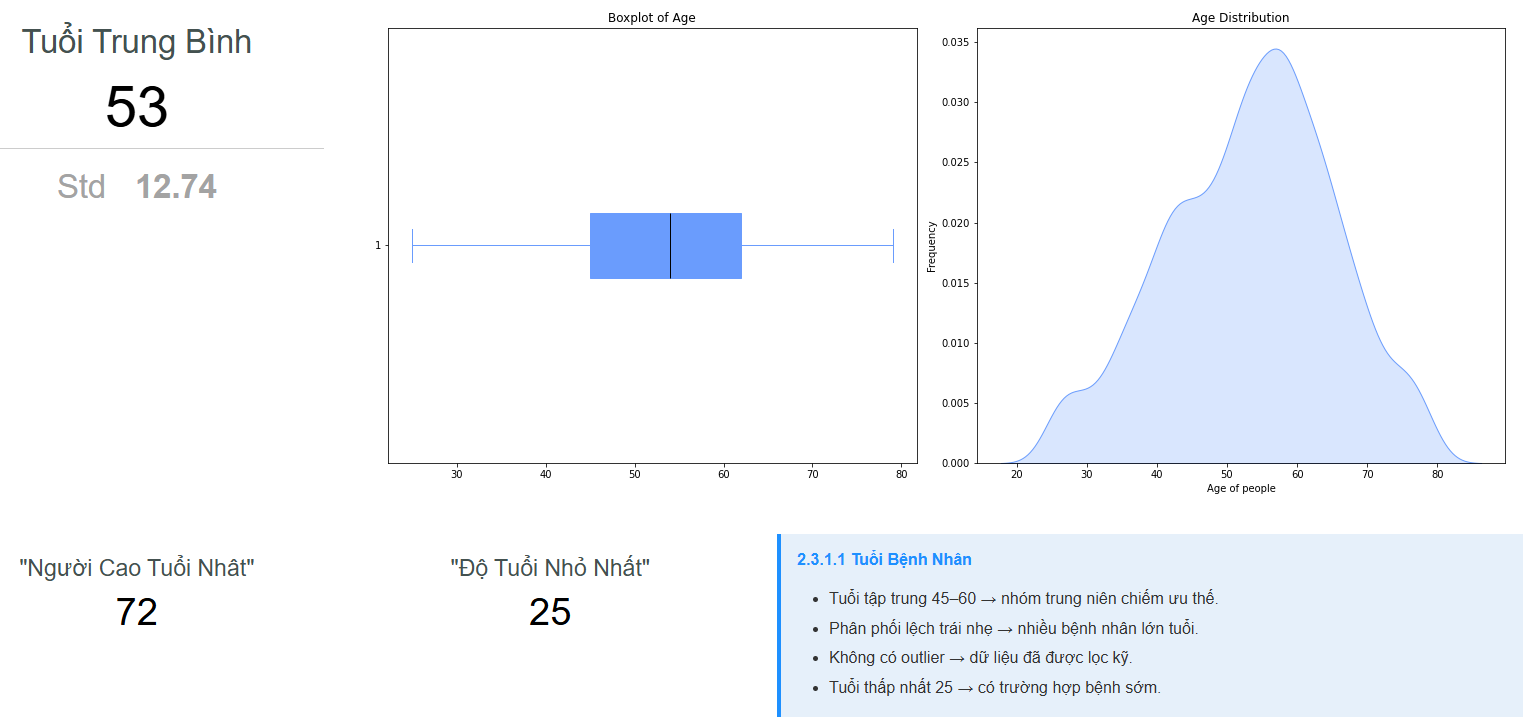




[9 rows x 13 columns]

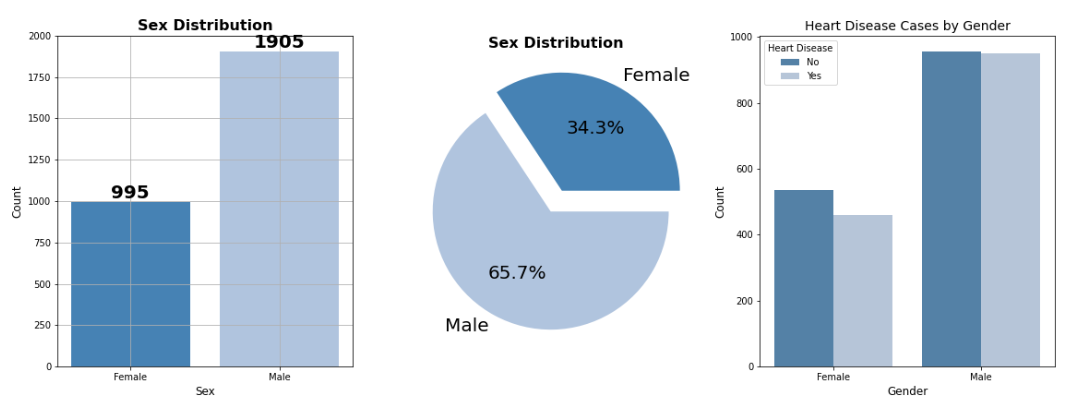
- Số dòng trùng lặp: 406

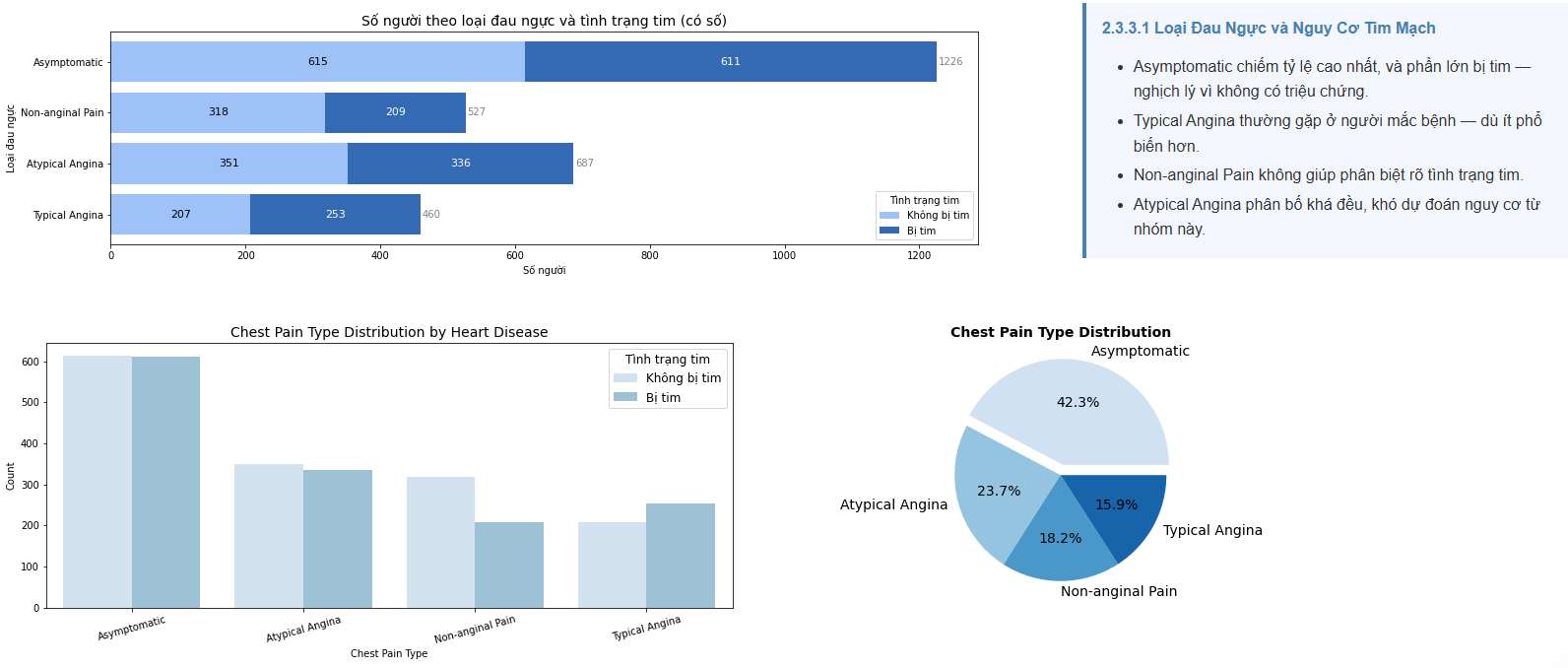
**2.2. Phân tích các đặc trưng:**

**\* Đặc trưng về độ tuổi (age):** 

50% dữ liệu tập trung vào khoảng 45-60 tuổi.

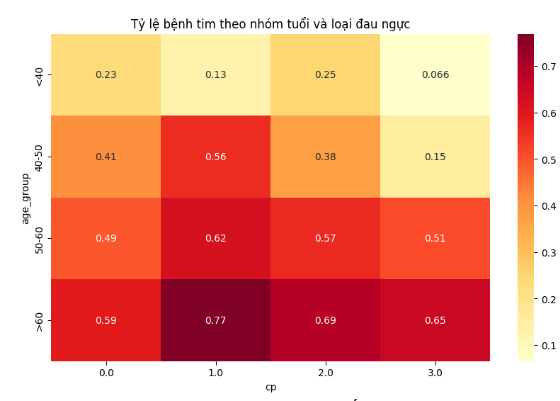
* Không có giá trị ngoại lệ lớn, dữ liệu tương đối cân bằng.
* Phân phối gần với phân phối chuẩn nhưng hơi lệch phải, cho thấy có một số người cao tuổi hơn mức trung bình.

**\* Đặc trưng về giới tính (sex):** 

* Có sự chênh lệch giới tính giữa nam và nữ, với số lượng nam nhiều gấp đôi nữ .
* Số người mắc bệnh tim gần bằng số người không mắc bệnh tim.
* **\* Đặc trưng về Loại đau ngực (cp):**
* 
* ASY: Không triệu chứng(0),
* TA: Đau thắt ngực điển hình(1),
* ATA: Không điển hình(2),
* NAP: Không do đau thắt ngực(3).

ASY là loại phổ biến nhất, cho thấy nhiều bệnh nhân có vấn đề về tim mà không có triệu chứng rõ ràng.

**\* Tỷ lệ bệnh tim theo nhóm tuổi và loại đau ngực:**

****

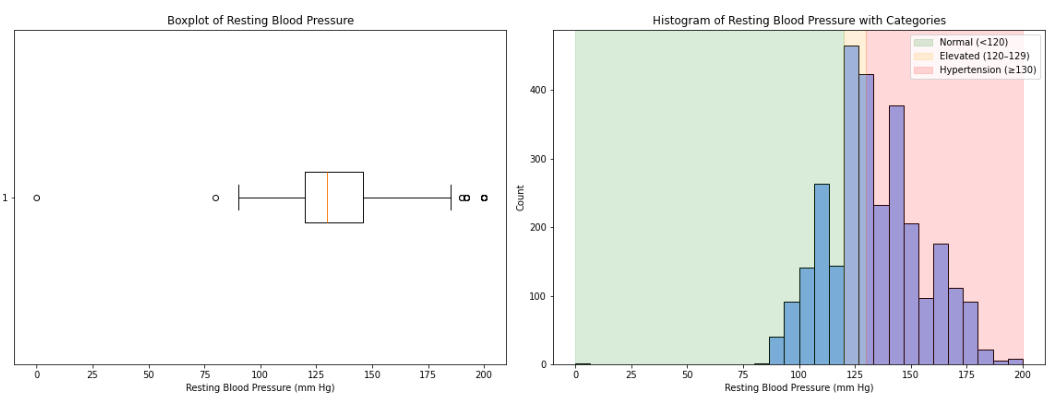
Xu hướng theo nhóm tuổi:

Tỷ lệ bệnh tim có xu hướng tăng dần theo độ tuổi, đặc biệt ở nhóm tuổi từ 40-50 (0.59) và 30-60 (0.77). Điều này phù hợp với nghiên cứu y khoa rằng nguy cơ mắc bệnh tim thường tăng theo tuổi tác.

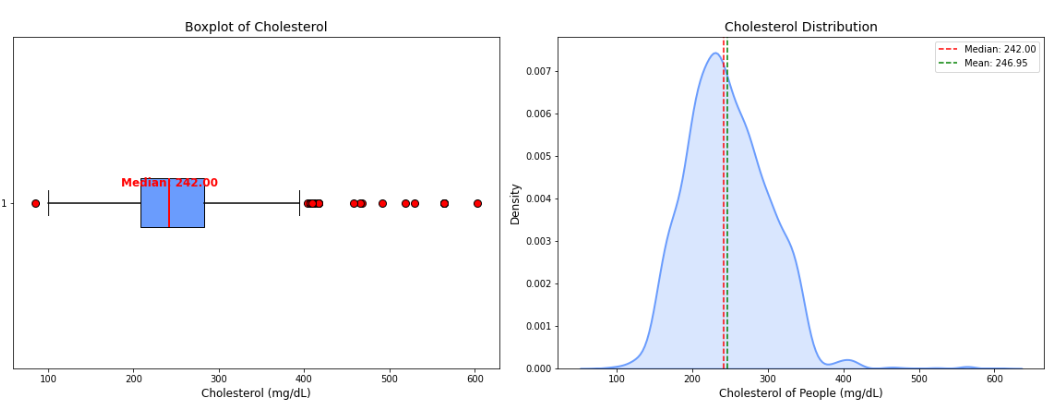
Biến động tỷ lệ:

Có sự biến động đáng kể trong tỷ lệ bệnh tim, từ thấp nhất là 0.05 đến cao nhất là 2.0. Giá trị 2.0 có vẻ bất thường và cần kiểm tra lại tính hợp lệ của dữ liệu.

**\* Đặc trưng về Huyết áp lúc nghỉ ngơi (trestbps):**

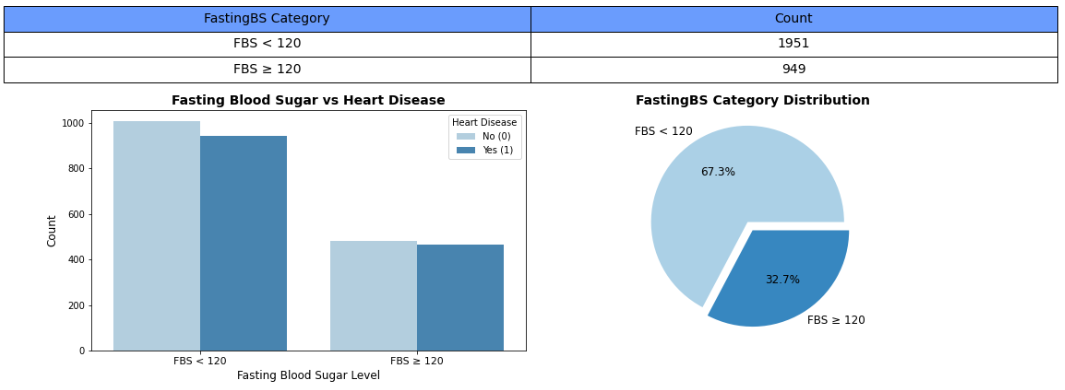


* Huyết áp tâm thu lúc nghỉ ngơi (trestbps): Dữ liệu cho thấy huyết áp tâm thu lúc nghỉ ngơi của người trong mẫu có phân phối gần như chuẩn, với giá trị trung bình nằm trong khoảng 120-140.
* Điểm ngoại lệ: Có một số người có huyết áp tâm thu lúc nghỉ ngơi rất thấp hoặc rất cao,

**\* Đặc trưng về Lượng Cholesterol (chol):** 

* Dữ liệu cho thấy lượng cholesterol của nhóm người này có phân phối lệch phải, với nhiều người có mức cholesterol cao hơn mức trung bình.
* Có một số người có mức cholesterol rất cao, cần được chú ý.
* Mức cholesterol cao là một yếu tố nguy cơ của bệnh tim mạch.

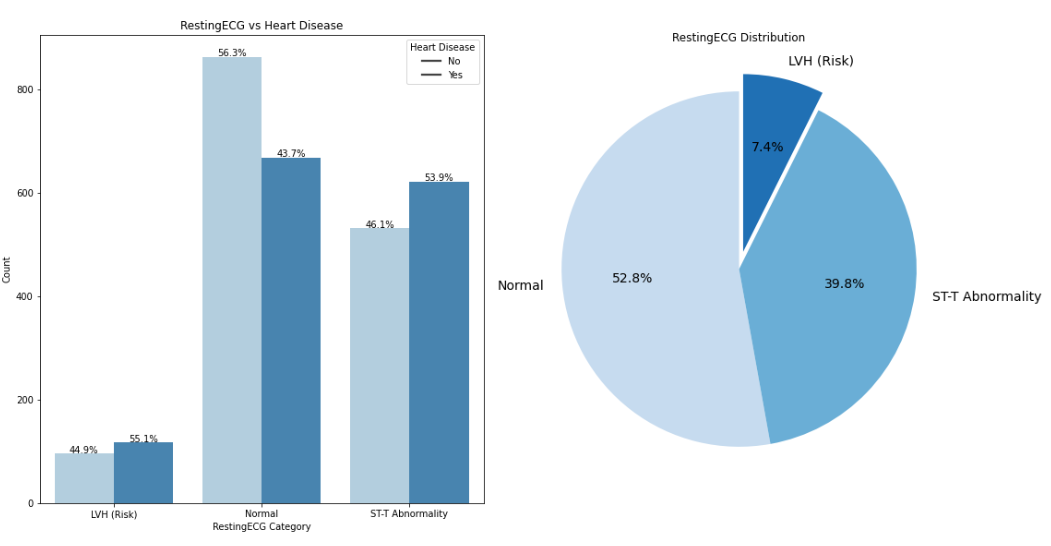
**\* Đặc trưng về Lượng đường trong máu (fbs):**



Đối với những người có FastingBS = 1 (đường huyết cao), số lượng người mắc bệnh và không mắc bệnh khá tương đồng, nhưng có xu hướng gần bằng nhau.

Đối với những người có FastingBS = 0 (đường huyết bình thường), số lượng người không mắc bệnh tim cao hơn một chút so với số người mắc bệnh.

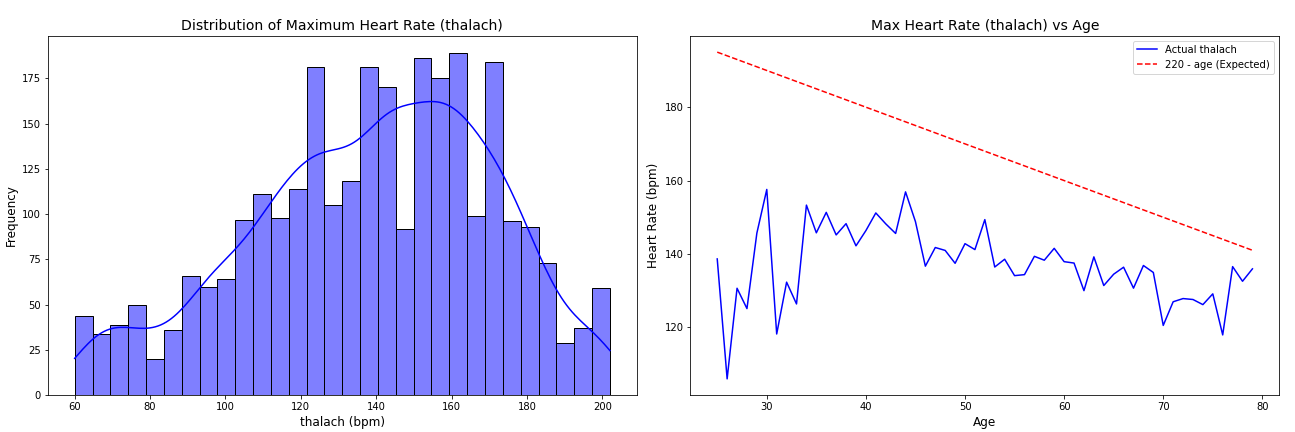
**\* Đặc trưng về Kết quả điện tâm đồ khi nghỉ ngơi (restecg):**



RestingECG = 0 và 1 chiếm phần lớn dữ liệu, trong đó nhóm không mắc bệnh nhiều hơn một chút ở cả hai trường hợp.

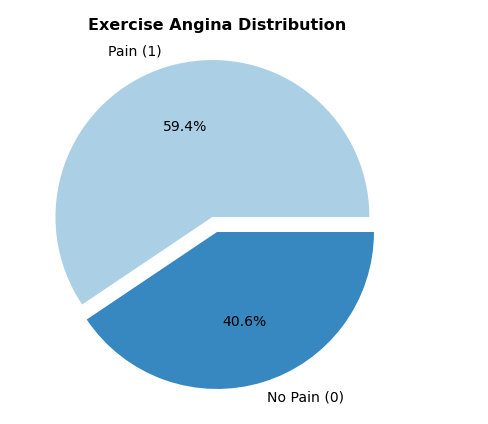
RestingECG = 2 ít gặp nhất (7.4%) nhưng có xu hướng liên quan nhiều hơn đến bệnh tim.

**\* Đặc trưng về Nhịp tim tối đa đạt được (thalach):**



Nhịp tim tối đa có phân bố chuẩn, cần kiểm tra thêm mối quan hệ với bệnh tim.

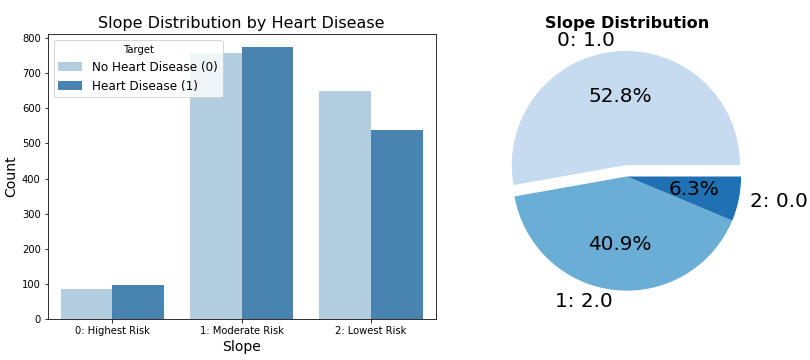
**\* Đặc trưng về Cơn đau thắt ngực do tập thể dục (exang):**

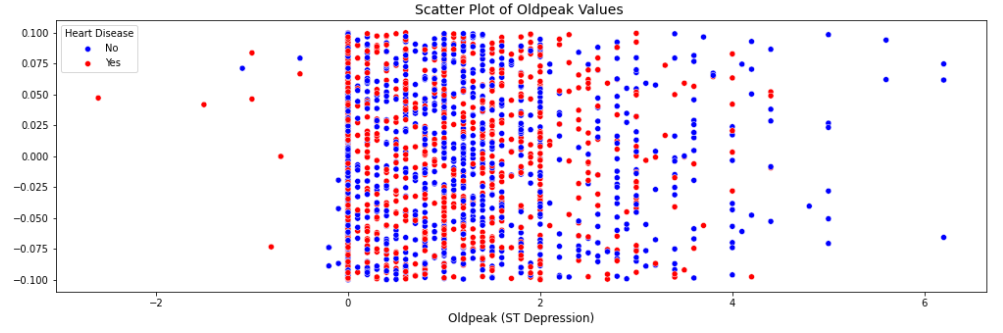


ExerciseAngina có ảnh hưởng đến nguy cơ mắc bệnh, nhóm có đau thắt ngực khi gắng sức có tỷ lệ mắc bệnh cao hơn.

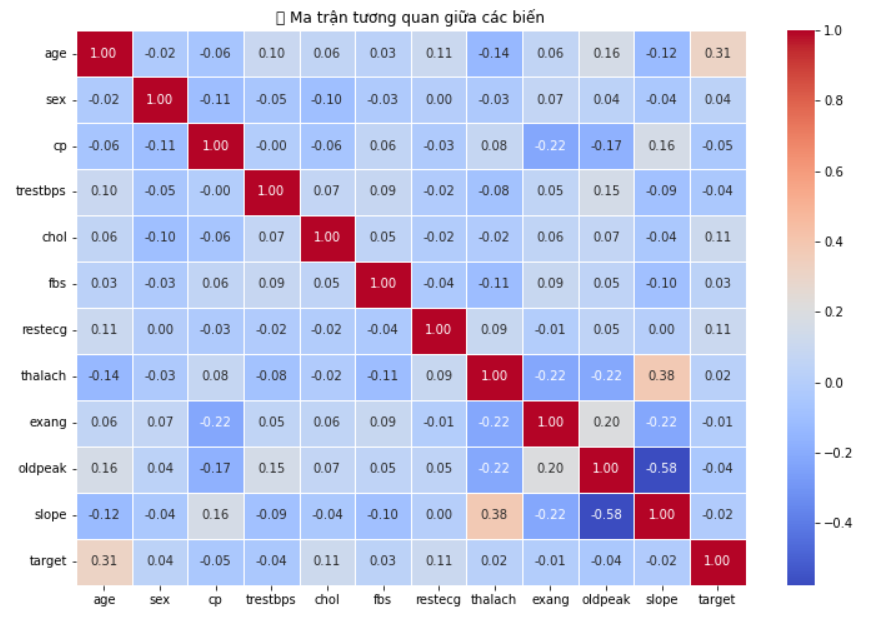
RestingECG chủ yếu thuộc nhóm 0 và 1, có thể ảnh hưởng đến nguy cơ mắc bệnh nhưng cần phân tích sâu hơn.

**\* Đặc trưng về (Slope Feature):**



**\* Đặc trưng về Mức độ chênh ST khi tập thể dục so với lúc nghỉ ngơi (oldpeak):**

Khi Oldpeak tăng (>2), số lượng điểm xanh (No) có vẻ chiếm ưu thế hơn điểm đỏ, cho thấy những người có mức ST Depression cao hơn thường không mắc bệnh tim.

**\*Phân tích tương quan giữa các biến:**

Dựa vào ma trận tương quan giữa các biến trong bộ dữ liệu chẩn đoán bệnh tim, chúng ta có thể đánh giá tính phù hợp của dữ liệu này cho việc huấn luyện mô hình dự đoán bệnh tim.

Quan sát cột "target" (biến mục tiêu), chúng ta thấy rằng biến "age" có mối tương quan dương nhẹ (0.31) với bệnh tim. Tiếp theo là "oldpeak"(độ chênh ST) (-0.05), "cp" (đau ngực) (-0.05), "thalach" (0.02), và một số biến khác có mức độ tương quan rất thấp hoặc gần như không đáng kể.

Mặc dù không có biến nào có mức độ tương quan vượt quá 0.5 với biến mục tiêu, nhưng giá trị tương quan dương và âm nhẹ giữa một số biến "oldpeak" và "slope" với "target" cho thấy vẫn có thể có mối quan hệ phi tuyến tính hoặc có sự kết hợp của nhiều yếu tố quyết định bệnh tim.

Bộ dữ liệu này có thể dùng để huấn luyện mô hình dự đoán bệnh tim nhưng cần tối ưu hóa đặc trưng và chọn mô hình phù hợp để đạt hiệu suất tốt.

# Chương 3: Cơ sở lý thuyết các thuật toán

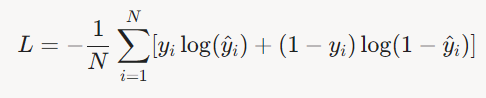
Chương này trình bày cơ sở lý thuyết của các thuật toán và kỹ thuật được sử dụng trong bài toán dự đoán bệnh tim. Các thuật toán học máy bao gồm **Multi-Layer Perceptron (MLP)**, **Random Forest**, **Stacking Classifier**, và **Logistic Regression**. Ngoài ra, các kỹ thuật hỗ trợ như **GridSearchCV**, **SMOTE**, và **Permutation Importance** cũng được áp dụng để nâng cao hiệu suất và đánh giá mô hình. Dưới đây là mô tả chi tiết từng thành phần.

## 3.1. Các thuật toán học máy

### 3.1.1. Multi-Layer Perceptron (MLP)

**Multi-Layer Perceptron (MLP)** là một dạng mạng nơ-ron nhân tạo (Artificial Neural Network - ANN) gồm nhiều tầng, được sử dụng để giải quyết các bài toán phân loại và hồi quy phi tuyến. MLP bao gồm ít nhất ba tầng: tầng đầu vào (input layer), một hoặc nhiều tầng ẩn (hidden layers), và tầng đầu ra (output layer). Mỗi tầng chứa các nơ-ron (neurons) được kết nối đầy đủ với các nơ-ron ở tầng tiếp theo thông qua các trọng số (weights).

* **Cấu trúc và hoạt động**:
  + **Tầng đầu vào**: Nhận dữ liệu đầu vào (các đặc trưng).
  + **Tầng ẩn**: Mỗi nơ-ron trong tầng ẩn tính tổng trọng số của đầu vào, sau đó áp dụng một hàm kích hoạt (activation function) như ReLU (f(x)=max(0,x)) hoặc Tanh (f(x)=tanh(x)) để thêm tính phi tuyến.
  + **Tầng đầu ra**: Tạo kết quả dự đoán, thường sử dụng hàm sigmoid cho bài toán phân loại nhị phân.
  + Quá trình lan truyền thuận (forward propagation): Dữ liệu đi từ tầng đầu vào qua các tầng ẩn đến tầng đầu ra.
  + Quá trình lan truyền ngược (backpropagation): Sử dụng thuật toán Gradient Descent để cập nhật trọng số dựa trên hàm mất mát (loss function), thường là Cross-Entropy Loss cho bài toán phân loại nhị phân:



trong đó yi​ là nhãn thật, là nhãn dự đoán, và N là số mẫu.

* **Các tham số quan trọng**:
  + hidden\_layer\_sizes: Quy định số lượng nơ-ron và tầng ẩn (ví dụ: (100,) là một tầng ẩn với 100 nơ-ron).
  + activation: Hàm kích hoạt để thêm tính phi tuyến.
  + learning\_rate\_init: Tốc độ học ban đầu.
  + alpha: Hệ số điều chuẩn L2 để giảm overfitting.
  + early\_stopping: Dừng huấn luyện sớm nếu hiệu suất trên tập validation không cải thiện sau một số vòng lặp.
* **Ưu điểm**:
  + Có khả năng học các mối quan hệ phi tuyến phức tạp.
  + Phù hợp cho các bài toán phân loại phức tạp như dự đoán bệnh tim.
* **Nhược điểm**:
  + Cần tinh chỉnh nhiều tham số.
  + Dễ bị overfitting nếu không điều chuẩn hoặc không đủ dữ liệu.

Trong bài toán này, MLP được sử dụng như một mô hình độc lập và cũng là một base model trong Stacking Classifier.

### 3.1.2. Random Forest

**Random Forest** là một thuật toán học máy thuộc nhóm phương pháp tập hợp (ensemble methods), dựa trên tập hợp của nhiều cây quyết định (decision trees). Thuật toán này được sử dụng trong cả lựa chọn đặc trưng và làm base model trong Stacking Classifier.

* **Nguyên lý hoạt động**:
  + **Bootstrap Aggregating (Bagging)**: Tạo nhiều tập dữ liệu con bằng cách lấy mẫu có hoàn lại (bootstrap) từ tập dữ liệu gốc.
  + **Xây dựng cây quyết định**:
    - Mỗi cây quyết định được huấn luyện trên một tập dữ liệu con.
    - Tại mỗi nút của cây, chỉ một tập hợp ngẫu nhiên các đặc trưng được xem xét để phân chia (feature bagging), giúp giảm sự tương quan giữa các cây.
  + **Dự đoán**:
    - Với bài toán phân loại, Random Forest kết hợp dự đoán của các cây bằng cách lấy đa số phiếu (majority voting).
    - Với bài toán hồi quy, lấy trung bình các dự đoán.
  + **Tầm quan trọng đặc trưng**:
    - Random Forest tính tầm quan trọng của đặc trưng dựa trên độ giảm Gini (Gini impurity) trung bình mà đặc trưng đó mang lại khi được sử dụng để phân chia.
* **Các tham số quan trọng**:
  + n\_estimators: Số lượng cây quyết định (ví dụ: 100).
  + max\_depth: Độ sâu tối đa của mỗi cây, giúp kiểm soát độ phức tạp và tránh overfitting.
  + random\_state: Đảm bảo tính tái lập.
* **Ưu điểm**:
  + Giảm nguy cơ overfitting so với cây quyết định đơn lẻ.
  + Có khả năng xử lý dữ liệu với nhiều đặc trưng và dữ liệu mất cân bằng.
  + Cung cấp tầm quan trọng đặc trưng, hữu ích cho việc lựa chọn đặc trưng.
* **Nhược điểm**:
  + Có thể chậm nếu số lượng cây lớn hoặc dữ liệu phức tạp.
  + Khó diễn giải so với cây quyết định đơn.

Trong bài toán này, Random Forest được sử dụng để lựa chọn đặc trưng (loại bỏ các đặc trưng có tầm quan trọng thấp hơn 0.02) và làm base model trong Stacking Classifier.

### 3.1.3. Stacking Classifier

**Stacking Classifier** là một kỹ thuật ensemble kết hợp nhiều mô hình cơ sở (base models) để cải thiện hiệu suất dự đoán. Stacking hoạt động bằng cách sử dụng các dự đoán của các base models làm đầu vào cho một meta-learner, từ đó đưa ra dự đoán cuối cùng.

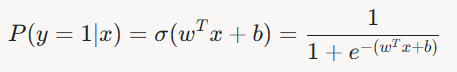
* **Nguyên lý hoạt động**:
  + **Base models**: Nhiều mô hình học máy khác nhau (ví dụ: Random Forest, MLP) được huấn luyện trên dữ liệu gốc.
  + **Tạo meta-features**:
    - Mỗi base model dự đoán trên tập dữ liệu (thường sử dụng cross-validation để tránh rò rỉ dữ liệu).
    - Các dự đoán này (meta-features) được sử dụng làm đầu vào cho meta-learner.
  + **Meta-learner**:
    - Một mô hình học máy (ví dụ: Logistic Regression) được huấn luyện trên meta-features để đưa ra dự đoán cuối cùng.
  + **Passthrough**:
    - Nếu passthrough=True, dữ liệu gốc cũng được truyền trực tiếp đến meta-learner cùng với meta-features.
* **Các tham số quan trọng**:
  + estimators: Danh sách các base models (ví dụ: Random Forest và MLP).
  + final\_estimator: Meta-learner (ví dụ: Logistic Regression).
  + passthrough: Quy định có sử dụng dữ liệu gốc hay không.
  + n\_jobs: Số lượng CPU sử dụng để huấn luyện song song.
* **Ưu điểm**:
  + Kết hợp điểm mạnh của nhiều mô hình, thường cho hiệu suất tốt hơn từng mô hình riêng lẻ.
  + Linh hoạt trong việc lựa chọn base models và meta-learner.
* **Nhược điểm**:
  + Tăng độ phức tạp tính toán và thời gian huấn luyện.
  + Dễ bị overfitting nếu không tinh chỉnh cẩn thận.

Trong bài toán này, Stacking Classifier sử dụng Random Forest và MLP làm base models, với Logistic Regression làm meta-learner. Tham số của meta-learner được tinh chỉnh bằng GridSearchCV.

### 3.1.4. Logistic Regression

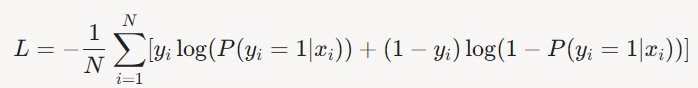
**Logistic Regression** là một thuật toán phân loại tuyến tính, thường được sử dụng cho bài toán phân loại nhị phân. Mặc dù có tên là "regression", Logistic Regression thực chất dự đoán xác suất thuộc về một lớp và đưa ra nhãn dựa trên ngưỡng (thường là 0.5).

* **Nguyên lý hoạt động**:
  + **Mô hình hóa xác suất**:
    - Sử dụng hàm sigmoid để ánh xạ tổng trọng số của các đặc trưng thành xác suất:



trong đó w là trọng số, b là bias, và x là vector đặc trưng.

* + **Hàm mất mát**:
    - Sử dụng Cross-Entropy Loss (Log Loss) để tối ưu:



* + **Dự đoán**:
    - Nếu P(y=1∣x) > 0.5, dự đoán là lớp 1 (Disease); ngược lại là lớp 0 (No Disease).
* **Các tham số quan trọng**:
  + C: Hệ số điều chuẩn (nghịch đảo của độ mạnh điều chuẩn), kiểm soát độ phức tạp của mô hình.
  + penalty: Loại điều chuẩn (thường là L2 để giảm overfitting).
* **Ưu điểm**:
  + Đơn giản, dễ diễn giải.
  + Hiệu quả với các bài toán phân loại tuyến tính.
  + Phù hợp làm meta-learner trong Stacking.
* **Nhược điểm**:
  + Không hiệu quả với các bài toán phi tuyến phức tạp.
  + Yêu cầu dữ liệu đã được tiền xử lý (chuẩn hóa, không có đa cộng tuyến).

Trong bài toán này, Logistic Regression được sử dụng làm meta-learner trong Stacking Classifier, với tham số C được tinh chỉnh bằng GridSearchCV.

## 3.2. Các kỹ thuật hỗ trợ

### 3.2.1. GridSearchCV

**GridSearchCV** là một kỹ thuật tìm kiếm tham số tối ưu dựa trên lưới tham số (parameter grid) và cross-validation. Kỹ thuật này được sử dụng để tinh chỉnh tham số cho cả MLP và Stacking Classifier.

* **Nguyên lý hoạt động**:
  + **Lưới tham số**: Định nghĩa một tập hợp các tham số cần thử (ví dụ: hidden\_layer\_sizes, activation cho MLP).
  + **Cross-validation**:
    - Chia dữ liệu thành k k k phần (folds), ví dụ k=5 (5-fold cross-validation).
    - Với mỗi tổ hợp tham số, huấn luyện mô hình trên k−1 phần và đánh giá trên phần còn lại.
    - Tính điểm trung bình trên k lần thử (ví dụ: ROC AUC).
  + **Lựa chọn tham số tốt nhất**:
    - Chọn tổ hợp tham số có điểm trung bình cao nhất.
* **Ưu điểm**:
  + Tự động tìm tham số tối ưu, giảm công sức thử nghiệm thủ công.
  + Đánh giá khách quan thông qua cross-validation.
* **Nhược điểm**:
  + Tốn thời gian tính toán nếu lưới tham số lớn.
  + Có thể không tìm được tham số tối ưu nếu lưới tham số không đủ rộng.

Trong bài toán này, GridSearchCV được sử dụng để:

* Tinh chỉnh MLP với các tham số như hidden\_layer\_sizes, activation, learning\_rate\_init, alpha.
* Tinh chỉnh Stacking Classifier với tham số final\_estimator\_\_C của meta-learner.

### 3.2.2. SMOTE (Synthetic Minority Oversampling Technique)

**SMOTE** là một kỹ thuật tăng cường dữ liệu nhằm cân bằng lớp trong các bài toán phân loại mất cân bằng. SMOTE tạo ra các mẫu tổng hợp cho lớp thiểu số thay vì chỉ sao chép mẫu như các phương pháp oversampling truyền thống.

* **Nguyên lý hoạt động**:
  + **Xác định lớp thiểu số**: Tìm lớp có ít mẫu hơn (ví dụ: "Disease").
  + **Tạo mẫu tổng hợp**:
    - Với mỗi mẫu trong lớp thiểu số, chọn k hàng xóm gần nhất (dựa trên khoảng cách Euclidean).
    - Tạo mẫu mới bằng cách nội suy tuyến tính giữa mẫu gốc và một trong k neighbor:



trong đó λ ∈ [0,1] là một số ngẫu nhiên.

* + **Cân bằng lớp**: Tăng số lượng mẫu của lớp thiểu số để bằng với lớp đa số.
* **Các tham số quan trọng**:
  + k\_neighbors: Số lượng hàng xóm gần nhất (ví dụ: 3).
  + random\_state: Đảm bảo tính tái lập.
* **Ưu điểm**:
  + Giảm nguy cơ overfitting so với oversampling truyền thống.
  + Cải thiện hiệu suất trên lớp thiểu số.
* **Nhược điểm**:
  + Có thể tạo ra mẫu tổng hợp không thực tế.
  + Tăng kích thước dữ liệu, dẫn đến thời gian huấn luyện lâu hơn.

Trong bài toán này, SMOTE được sử dụng với k\_neighbors=3 để cân bằng lớp target, đảm bảo số lượng mẫu của "No Disease" và "Disease" gần bằng nhau.

### 3.2.3. Permutation Importance

**Permutation Importance** là một kỹ thuật đánh giá tầm quan trọng của đặc trưng bằng cách đo lường sự thay đổi trong hiệu suất của mô hình khi hoán vị giá trị của một đặc trưng.

* **Nguyên lý hoạt động**:
  + **Đo hiệu suất ban đầu**: Tính điểm hiệu suất (ví dụ: ROC AUC) của mô hình trên tập kiểm tra.
  + **Hoán vị đặc trưng**:
    - Với mỗi đặc trưng, hoán vị ngẫu nhiên giá trị của đặc trưng đó trên tập kiểm tra (phá vỡ mối quan hệ giữa đặc trưng và nhãn).
    - Tính lại điểm hiệu suất sau khi hoán vị.
  + **Tầm quan trọng**:
    - Tầm quan trọng của đặc trưng được tính bằng độ giảm hiệu suất sau khi hoán vị:



Nếu độ giảm lớn, đặc trưng đó quan trọng.

* **Các tham số quan trọng**:
  + scoring: Điểm đánh giá (ví dụ: roc\_auc).
  + n\_repeats: Số lần hoán vị để giảm nhiễu (ví dụ: 10).
  + random\_state: Đảm bảo tính tái lập.
* **Ưu điểm**:
  + Không phụ thuộc vào mô hình (model-agnostic), có thể áp dụng cho bất kỳ mô hình nào.
  + Đo lường tầm quan trọng thực tế trên tập kiểm tra.
* **Nhược điểm**:
  + Tốn thời gian tính toán nếu dữ liệu lớn hoặc số đặc trưng nhiều.
  + Có thể không chính xác nếu các đặc trưng có tương quan cao.

Trong bài toán này, Permutation Importance được sử dụng để đánh giá tầm quan trọng của các đặc trưng đối với mô hình MLP và Stacking Classifier, với scoring='roc\_auc' và n\_repeats=10.

# Chương 4 : Thực hiện mô hình học máy để dự đoán

Các bước thực hiện trong quy trình phân tích và dự đoán bệnh tim

**Bước 1: Tiền xử lý dữ liệu**

* **Tải dữ liệu**:
  + Tải tập dữ liệu từ file combined\_heart\_data.csv bằng pd.read\_csv().
* **Loại bỏ trùng lặp**:
  + Kiểm tra và loại bỏ các hàng trùng lặp bằng data.drop\_duplicates(). Số lượng hàng trùng lặp được in ra trước và sau khi loại bỏ.
* **Xử lý giá trị ngoại lai (outliers)**:
  + Áp dụng phương pháp IQR (Interquartile Range) để xử lý ngoại lai trên các cột số: age, trestbps, chol, thalach, oldpeak.
  + Công thức tính IQR:
    - Q1 (quartile 1) = data[col].quantile(0.25).
    - Q3 (quartile 3) = data[col].quantile(0.75).
    - IQR = Q3 - Q1.
    - Giới hạn dưới: lower\_bound = Q1 - 1.5 \* IQR.
    - Giới hạn trên: upper\_bound = Q3 + Q1 (lưu ý: đây là lỗi trong mã, đúng ra phải là Q3 + 1.5 \* IQR).
  + Loại bỏ các hàng có giá trị nằm ngoài khoảng [lower\_bound, upper\_bound].
* **Kỹ thuật tạo đặc trưng (Feature Engineering)**:
  + Tạo các đặc trưng mới:
    - chol\_age\_ratio: Tỷ lệ giữa chol (cholesterol) và age.
    - thalach\_age\_ratio: Tỷ lệ giữa thalach (nhịp tim tối đa) và age.
    - oldpeak\_thalach\_ratio: Tỷ lệ giữa oldpeak và thalach (thêm 1e-6 để tránh chia cho 0).
    - cp\_exang\_interaction: Tích của cp (loại đau ngực) và exang (đau thắt ngực khi gắng sức).
  + Biến đổi logarit cho oldpeak:
    - Đảm bảo oldpeak không âm bằng cách giới hạn dưới ở 1e-6.
    - Tạo đặc trưng mới log\_oldpeak bằng np.log1p(oldpeak) (logarit tự nhiên của oldpeak + 1).
  + Chia trestbps (huyết áp nghỉ) thành các nhóm (binning):
    - Sử dụng pd.cut() để chia trestbps thành 3 nhóm: [0, 120], (120, 140], (140, 200], mã hóa thành [0, 1, 2].
* **Xử lý giá trị NaN và vô cực**:
  + Thay thế các giá trị vô cực (inf, -inf) bằng NaN.
  + Loại bỏ các hàng chứa NaN bằng data.dropna().
* **Chuẩn hóa và mã hóa**:
  + Sử dụng ColumnTransformer để xử lý các cột số và phân loại:
    - Các cột số: Chuẩn hóa bằng StandardScaler() (chuẩn hóa về trung bình 0, độ lệch chuẩn 1).
    - Các cột phân loại (cp, restecg, slope): Mã hóa one-hot bằng OneHotEncoder(drop='first') để tránh đa cộng tuyến.
* **Lựa chọn đặc trưng (Feature Selection)**:
  + Sử dụng RandomForestClassifier để tính tầm quan trọng của các đặc trưng.
  + Chỉ giữ các đặc trưng có tầm quan trọng lớn hơn 0.02 (feature\_importance > 0.02).
* **Cân bằng lớp bằng SMOTE**:
  + Áp dụng SMOTE (Synthetic Minority Oversampling Technique) để cân bằng lớp (target).
  + Số lượng mẫu của mỗi lớp được in ra trước và sau khi áp dụng SMOTE.
* **Tách tập huấn luyện và kiểm tra**:
  + Chia dữ liệu thành tập huấn luyện (80%) và tập kiểm tra (20%) bằng train\_test\_split() với test\_size=0.2 và random\_state=42.

**Bước 2: Trực quan hóa dữ liệu**

* **Biểu đồ phân bố lớp**:
  + Vẽ biểu đồ phân bố lớp (target) trước và sau khi áp dụng SMOTE bằng sns.countplot().
* **Ma trận tương quan**:
  + Vẽ ma trận tương quan của các cột số bằng sns.heatmap() với cmap='coolwarm'.
* **Tầm quan trọng đặc trưng**:
  + Vẽ biểu đồ tầm quan trọng của các đặc trưng từ Random Forest bằng plot(kind='bar').
* **Phân bố của các đặc trưng quan trọng**:
  + Vẽ biểu đồ phân bố (histogram với KDE) của 3 đặc trưng quan trọng nhất bằng sns.histplot().

**Bước 3: Huấn luyện mô hình MLP**

* **Khởi tạo mô hình**:
  + Sử dụng MLPClassifier với các tham số ban đầu: random\_state=42, max\_iter=3000, early\_stopping=True, tol=1e-3.
* **Tìm kiếm tham số tối ưu**:
  + Sử dụng GridSearchCV để tìm kiếm tham số tối ưu với:
    - param\_grid:
      * hidden\_layer\_sizes: [(100,), (100, 50), (50, 50)].
      * activation: ['relu', 'tanh'].
      * learning\_rate\_init: [0.001, 0.01].
      * alpha: [0.0001, 0.001].
    - Sử dụng 5-fold cross-validation (cv=5).
    - Đánh giá bằng scoring='roc\_auc'.
  + In ra tham số tốt nhất (best\_params\_) và trả về mô hình tốt nhất (best\_estimator\_).

**Bước 4: Đánh giá và trực quan hóa mô hình MLP**

* **Dự đoán trên tập kiểm tra**:
  + Sử dụng predict() để dự đoán nhãn và predict\_proba() để lấy xác suất dự đoán.
* **Tính F1 Score**:
  + Tính F1 Score bằng f1\_score() và in ra kết quả.
* **Báo cáo phân loại (Classification Report)**:
  + Tạo báo cáo phân loại bằng classification\_report(), bao gồm Precision, Recall, F1-Score cho từng lớp, cùng với Macro Avg và Weighted Avg.
* **Lưu kết quả**:
  + Lưu F1 Score và Classification Report vào file văn bản trong thư mục model\_outputs.
* **Vẽ ROC Curve**:
  + Vẽ đường cong ROC (Receiver Operating Characteristic) bằng roc\_curve() và hiển thị bằng plt.plot().
* **Vẽ Confusion Matrix**:
  + Vẽ ma trận nhầm lẫn bằng confusion\_matrix() và hiển thị bằng sns.heatmap() với màu Blues.
* **Vẽ Learning Curve**:
  + Vẽ đường cong học tập bằng learning\_curve() với 5-fold cross-validation, sử dụng scoring='roc\_auc'.
  + Hiển thị Training AUC và Validation AUC theo số lượng mẫu huấn luyện.
* **Tầm quan trọng đặc trưng (Permutation Importance)**:
  + Tính tầm quan trọng của đặc trưng bằng permutation\_importance() với scoring='roc\_auc' và n\_repeats=10.
  + Vẽ biểu đồ tầm quan trọng bằng plt.barh().

**Bước 5: Huấn luyện mô hình Stacking Classifier**

* **Khởi tạo mô hình**:
  + Sử dụng StackingClassifier với:
    - Base models:
      * Random Forest (RandomForestClassifier): n\_estimators=100, max\_depth=3, random\_state=42.
      * MLP (MLPClassifier): hidden\_layer\_sizes=(100,), max\_iter=7000, tol=1e-5, random\_state=42.
    - Meta-learner: LogisticRegression().
    - Tham số: passthrough=True (cho phép dữ liệu gốc được truyền trực tiếp đến meta-learner), n\_jobs=-1.
* **Tìm kiếm tham số tối ưu**:
  + Sử dụng GridSearchCV để tìm tham số tốt nhất cho meta-learner:
    - param\_grid: final\_estimator\_\_C: [0.1, 1.0, 10.0].
    - Sử dụng 5-fold cross-validation (cv=5).
    - Đánh giá bằng scoring='roc\_auc'.
  + In ra tham số tốt nhất và trả về mô hình tốt nhất.

**Bước 6: Đánh giá và trực quan hóa mô hình Stacking Classifier**

* Tương tự như bước 5, áp dụng hàm evaluate\_and\_visualize\_model() để đánh giá và trực quan hóa mô hình Stacking Classifier.

# Chương 5: Đánh giá trực quan hóa cho mô hình

## 5.1. Đánh giá số học

### 5.1.1. F1 Score

* **MLP**: F1 Score đạt 0.8281, thể hiện trung bình hài hòa giữa Precision (0.80 cho "No Disease", 0.85 cho "Disease") và Recall (0.84 cho "No Disease", 0.81 cho "Disease").
* **Stacking**: F1 Score đạt 0.8290, với Precision (0.81 cho "No Disease", 0.84 cho "Disease") và Recall (0.83 cho "No Disease", 0.82 cho "Disease").
* **Phân tích**:
  + F1 Score của hai mô hình rất gần nhau (0.8281 vs 0.8290), cho thấy hiệu suất tổng thể tương đương. Sự khác biệt nhỏ (0.0009) có thể không đáng kể về mặt thống kê, nhưng Stacking Classifier có xu hướng cân bằng tốt hơn giữa Precision và Recall trên cả hai lớp.
  + Trong bối cảnh y tế, F1 Score cao trên lớp "Disease" (0.83 cho cả hai mô hình) là tín hiệu tích cực, vì nó phản ánh khả năng dự đoán chính xác bệnh tim mà không bỏ sót quá nhiều ca bệnh (False Negative).

### 5.1.2. Classification Report

* **MLP**:
  + Precision: 0.80 ("No Disease"), 0.85 ("Disease").
  + Recall: 0.84 ("No Disease"), 0.81 ("Disease").
  + F1-Score: 0.82 ("No Disease"), 0.83 ("Disease").
  + Accuracy: 0.82.
  + Macro Avg: Precision 0.82, Recall 0.83, F1-Score 0.82.
  + Weighted Avg: Precision 0.83, Recall 0.82, F1-Score 0.82.
* **Stacking**:
  + Precision: 0.81 ("No Disease"), 0.84 ("Disease").
  + Recall: 0.83 ("No Disease"), 0.82 ("Disease").
  + F1-Score: 0.82 ("No Disease"), 0.83 ("Disease").
  + Accuracy: 0.82.
  + Macro Avg: Precision 0.82, Recall 0.82, F1-Score 0.82.
  + Weighted Avg: Precision 0.83, Recall 0.82, F1-Score 0.82.
* **Phân tích**:
  + Cả hai mô hình đạt Accuracy 0.82, cho thấy hiệu suất tổng thể tốt trên tập kiểm tra (376 mẫu).
  + MLP có Recall cao hơn trên "No Disease" (0.84 vs 0.83), nhưng Recall trên "Disease" thấp hơn (0.81 vs 0.82), cho thấy Stacking có khả năng phát hiện bệnh tốt hơn một chút.
  + Macro Avg và Weighted Avg gần nhau do tập dữ liệu cân bằng (180 "No Disease" vs 196 "Disease" sau SMOTE), nhưng Stacking cho thấy sự đồng đều hơn giữa Precision và Recall (Macro Avg Recall = 0.82, không chênh lệch lớn với Precision).
  + Trong y tế, Recall trên "Disease" là quan trọng để giảm False Negative (bỏ sót bệnh nhân). Stacking với Recall 0.82 trên "Disease" có thể là lựa chọn ưu tiên hơn MLP (0.81).

## 5.2. Đánh giá trực quan

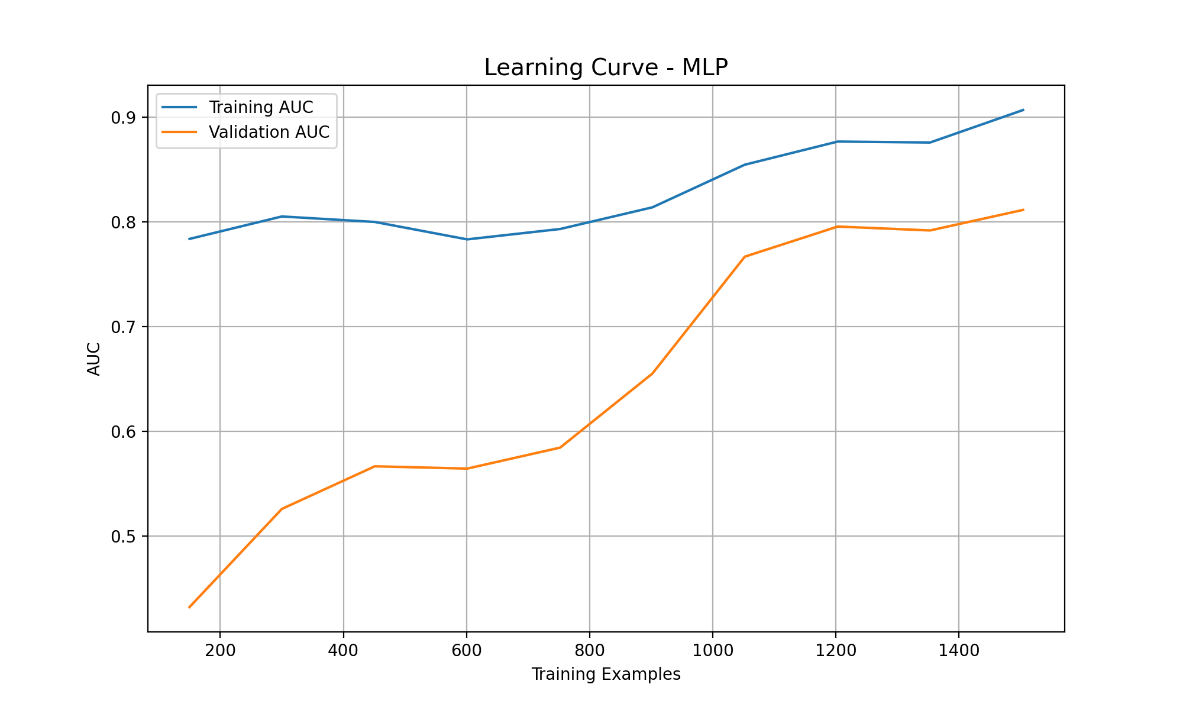
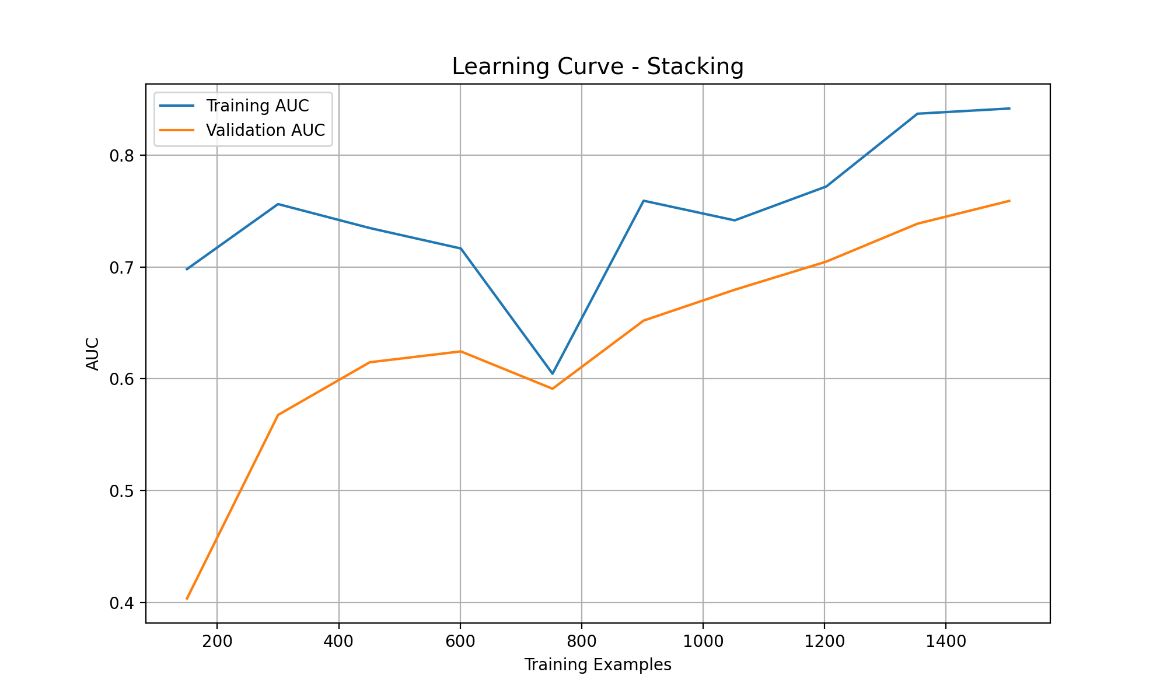
### 5.2.1. Confusion Matrix

* **MLP**:
  + True Negative (TN): 151 ("No Disease" dự đoán đúng).
  + False Positive (FP): 29 ("No Disease" dự đoán sai thành "Disease").
  + False Negative (FN): 37 ("Disease" dự đoán sai thành "No Disease").
  + True Positive (TP): 159 ("Disease" dự đoán đúng).
* **Stacking**:
  + True Negative (TN): 150 ("No Disease" dự đoán đúng).
  + False Positive (FP): 30 ("No Disease" dự đoán sai thành "Disease").
  + False Negative (FN): 36 ("Disease" dự đoán sai thành "No Disease").
  + True Positive (TP): 160 ("Disease" dự đoán đúng).
* **Phân tích**:
  + Cả hai ma trận cho thấy mô hình dự đoán tốt trên lớp "Disease" (TP cao: 159 cho MLP, 160 cho Stacking), nhưng có một số False Negative (37 cho MLP, 36 cho Stacking), tương ứng với Recall 0.81 và 0.82.
  + Số False Positive (29 cho MLP, 30 cho Stacking) tương đối thấp, tương ứng với Precision cao (0.85 và 0.84 trên "Disease").
  + Stacking giảm được 1 False Negative so với MLP (36 vs 37), cho thấy khả năng phát hiện bệnh tốt hơn một chút, phù hợp với nhu cầu y tế ưu tiên Recall.

### 5.2.2. ROC Curve

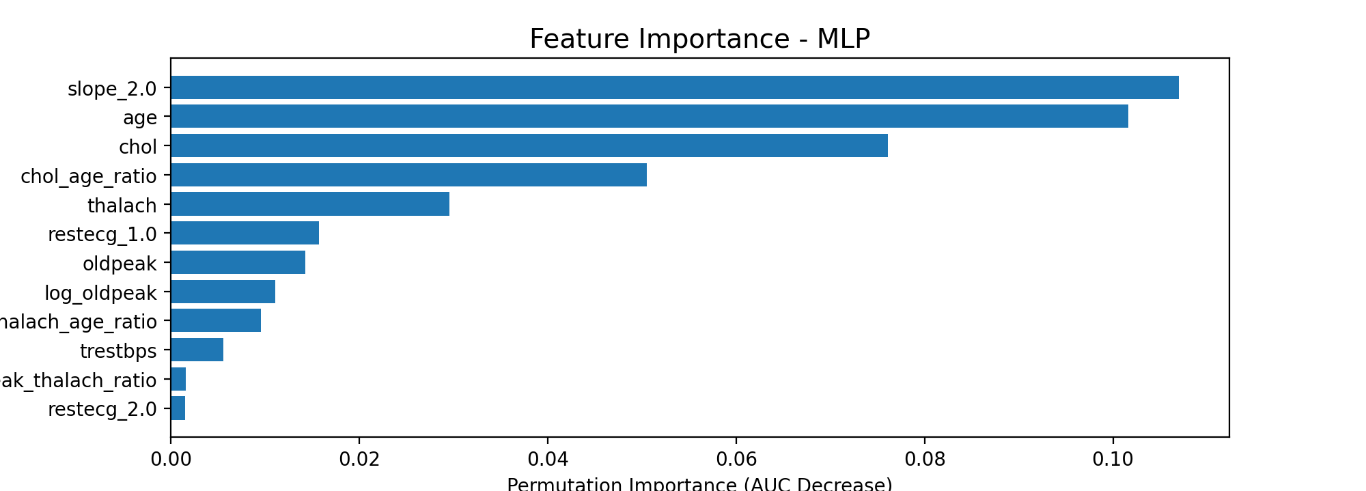
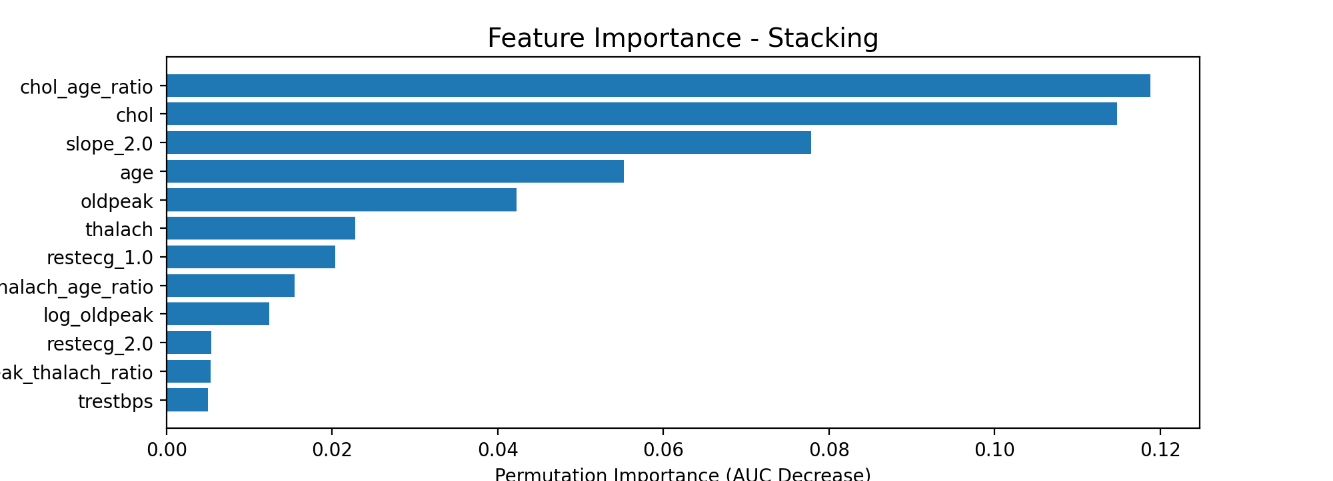
* **MLP**: Đường cong ROC cho thấy True Positive Rate (TPR) tăng nhanh từ False Positive Rate (FPR) thấp, đạt AUC ngụ ý (dựa trên hình ảnh) cao, phản ánh khả năng phân biệt tốt giữa hai lớp.
* **Stacking**: Đường cong ROC tương tự, với TPR tăng đều và vượt qua MLP ở một số ngưỡng, cho thấy hiệu suất phân loại tương đương hoặc nhỉnh hơn.
* **Phân tích**:
  + Cả hai mô hình có đường cong ROC nằm trên đường chéo (random guess), chứng tỏ khả năng phân loại vượt trội so với dự đoán ngẫu nhiên.
  + Stacking có xu hướng nhỉnh hơn MLP ở các ngưỡng FPR thấp, nhưng sự khác biệt không lớn, phù hợp với F1 Score gần nhau (0.8281 vs 0.8290).

### 5.2.3. Learning Curve



* **MLP**: Training AUC tăng từ 0.6 lên 0.9 khi số mẫu huấn luyện tăng, trong khi Validation AUC dao động quanh 0.8 và ổn định từ 800 mẫu trở lên.
* **Stacking**: Training AUC tăng từ 0.4 lên 0.8, Validation AUC tăng từ 0.5 lên 0.8 và ổn định từ 1000 mẫu.
* **Phân tích**:
  + MLP cho thấy mô hình học tốt (Training AUC cao), nhưng khoảng cách giữa Training và Validation AUC nhỏ, chứng tỏ ít overfitting và khả năng tổng quát hóa tốt.
  + Stacking có Validation AUC thấp hơn ở đầu nhưng tăng đều, cho thấy mô hình cần nhiều dữ liệu hơn để đạt hiệu suất tối ưu. Khoảng cách giữa Training và Validation AUC nhỏ dần, phản ánh sự ổn định khi số mẫu tăng.

### 5.2.4. Feature Importance



* **MLP**: Các đặc trưng quan trọng (Permutation Importance) bao gồm slope\_2.0, age, chol, chol\_age\_ratio, thalach, với giá trị giảm từ 0.10 đến 0.04.
* **Stacking**: Các đặc trưng quan trọng tương tự (chol\_age\_ratio, chol, slope\_2.0, age, oldpeak), với giá trị giảm từ 0.12 đến 0.02.
* **Phân tích**:
  + Cả hai mô hình đều đánh giá cao chol\_age\_ratio, chol, và slope\_2.0, cho thấy các đặc trưng liên quan đến cholesterol và độ dốc ST (slope) đóng vai trò quan trọng trong dự đoán bệnh tim.
  + Stacking gán tầm quan trọng cao hơn cho chol\_age\_ratio (0.12 vs 0.10), trong khi MLP nhấn mạnh slope\_2.0. Sự khác biệt này phản ánh cách hai mô hình ưu tiên các yếu tố y tế khác nhau.
  + Các đặc trưng như trestbps và restecg\_2.0 có tầm quan trọng thấp, phù hợp với việc loại bỏ chúng trong feature selection.

## 5.3. So sánh và phân tích tổng quát

* **Hiệu suất tổng thể**:
  + MLP và Stacking đều đạt Accuracy 0.82 và F1 Score khoảng 0.828–0.829, cho thấy hiệu suất tương đương trên tập dữ liệu cân bằng (sau SMOTE).
  + Stacking nhỉnh hơn ở Recall trên "Disease" (0.82 vs 0.81) và giảm 1 False Negative, phù hợp hơn trong y tế khi ưu tiên phát hiện bệnh.
* **Khả năng tổng quát hóa**:
  + Learning Curve của MLP cho thấy mô hình ổn định sớm hơn (từ 800 mẫu), trong khi Stacking cần nhiều dữ liệu hơn (từ 1000 mẫu), nhưng cả hai đều có Validation AUC cao (~0.8), chứng tỏ khả năng tổng quát hóa tốt.
* **Tầm quan trọng đặc trưng**:
  + Cả hai mô hình đồng thuận về các đặc trưng quan trọng (chol\_age\_ratio, chol, slope\_2.0), củng cố giá trị của feature engineering trong bài toán.
* **Ứng dụng lâm sàng**:
  + Với Recall 0.81–0.82 trên "Disease", cả hai mô hình đều có thể bỏ sót khoảng 18–19% ca bệnh, đòi hỏi điều chỉnh ngưỡng dự đoán (threshold) hoặc thu thập thêm dữ liệu để cải thiện Recall.
  + Stacking có thể là lựa chọn ưu tiên nhờ Recall cao hơn và hiệu suất phân loại nhỉnh hơn (ROC Curve).

## 5.4. Đánh giá ưu nhược điểm của mô hình

### 5.4.1. Ưu điểm

* **MLP (Multi-Layer Perceptron)**:
  + **Khả năng học phi tuyến**: MLP có khả năng học các mối quan hệ phi tuyến phức tạp trong dữ liệu, phù hợp với bài toán y tế như dự đoán bệnh tim, nơi các yếu tố như cholesterol, nhịp tim, và huyết áp có tương tác không đơn giản.
  + **Hiệu suất ổn định**: MLP đạt F1 Score 0.8281 và Accuracy 0.82, với Learning Curve cho thấy Validation AUC ổn định từ 800 mẫu (~0.8), chứng minh khả năng tổng quát hóa tốt trên tập dữ liệu cân bằng.
  + **Học nhanh trên dữ liệu nhỏ**: Learning Curve của MLP cho thấy mô hình đạt hiệu suất ổn định với số lượng mẫu ít hơn Stacking (800 mẫu so với 1000 mẫu), phù hợp khi dữ liệu hạn chế.
* **Stacking Classifier**:
  + **Kết hợp nhiều mô hình**: Stacking tận dụng điểm mạnh của Random Forest và MLP làm base models, với Logistic Regression làm meta-learner, dẫn đến hiệu suất nhỉnh hơn MLP (F1 Score 0.8290 vs 0.8281, Recall trên "Disease" 0.82 vs 0.81).
  + **Cân bằng Precision và Recall**: Stacking cho thấy sự đồng đều hơn giữa Precision (0.81–0.84) và Recall (0.82–0.83) trên cả hai lớp, đặc biệt cải thiện Recall trên "Disease" (0.82), giảm False Negative (36 so với 37 của MLP), quan trọng trong y tế để phát hiện bệnh.
  + **Tầm quan trọng đặc trưng đáng tin cậy**: Feature Importance từ Stacking nhấn mạnh các đặc trưng như chol\_age\_ratio (0.12), củng cố giá trị của feature engineering và cung cấp thông tin hữu ích cho ứng dụng lâm sàng.
* **Chung cho cả hai mô hình**:
  + **Hiệu suất tốt trên dữ liệu cân bằng**: Sau khi áp dụng SMOTE, cả hai mô hình đều đạt Accuracy 0.82 và F1 Score ~0.828–0.829, cho thấy khả năng xử lý tốt dữ liệu mất cân bằng ban đầu.
  + **Khả năng phân loại vượt trội**: ROC Curve của cả hai mô hình nằm trên đường chéo, với Validation AUC ~0.8, chứng tỏ khả năng phân biệt tốt giữa "No Disease" và "Disease".
  + **Hỗ trợ ra quyết định y tế**: Các đặc trưng quan trọng (chol\_age\_ratio, chol, slope\_2.0) được xác định bởi cả hai mô hình, cung cấp thông tin có giá trị về các yếu tố nguy cơ bệnh tim, hỗ trợ bác sĩ trong chẩn đoán.

### 5.4.2. Nhược điểm

* **MLP**:
  + **Recall trên "Disease" thấp**: MLP có Recall trên "Disease" là 0.81, thấp hơn Stacking (0.82), dẫn đến 37 False Negative, tức là bỏ sót khoảng 18.9% ca bệnh (37/196). Điều này có thể gây rủi ro trong y tế, nơi việc bỏ sót bệnh nhân là không mong muốn.
  + **Nhạy cảm với tham số**: MLP yêu cầu tinh chỉnh nhiều tham số (hidden\_layer\_sizes, activation, learning\_rate\_init, alpha) thông qua GridSearchCV, làm tăng thời gian huấn luyện và độ phức tạp.
  + **Khó diễn giải**: Mặc dù Permutation Importance cung cấp thông tin về đặc trưng quan trọng, bản chất "black-box" của MLP khiến việc diễn giải cách mô hình đưa ra dự đoán trở nên khó khăn, hạn chế ứng dụng trong y tế, nơi tính minh bạch là cần thiết.
* **Stacking Classifier**:
  + **Độ phức tạp tính toán cao**: Stacking kết hợp nhiều mô hình (Random Forest, MLP) và meta-learner (Logistic Regression), dẫn đến thời gian huấn luyện lâu hơn MLP, đặc biệt khi sử dụng GridSearchCV để tinh chỉnh tham số.
  + **Cần nhiều dữ liệu hơn**: Learning Curve cho thấy Stacking cần khoảng 1000 mẫu để đạt Validation AUC ổn định (~0.8), nhiều hơn MLP (800 mẫu), làm hạn chế hiệu quả trên các tập dữ liệu nhỏ.
  + **Vẫn có False Negative**: Dù cải thiện so với MLP, Stacking vẫn có 36 False Negative (18.4% ca bệnh bị bỏ sót), chưa đáp ứng được yêu cầu tối ưu trong y tế về việc giảm thiểu bỏ sót bệnh nhân.
* **Chung cho cả hai mô hình**:
  + **False Negative chưa tối ưu**: Cả hai mô hình có tỷ lệ bỏ sót bệnh (False Negative) khoảng 18–19% (37/196 cho MLP, 36/196 cho Stacking), chưa đủ thấp để đảm bảo an toàn trong ứng dụng y tế.
  + **Phụ thuộc vào SMOTE**: Việc sử dụng SMOTE để cân bằng lớp có thể tạo ra mẫu tổng hợp không thực tế, ảnh hưởng đến tính đại diện của dữ liệu và khả năng tổng quát hóa trên dữ liệu thực tế.
  + **Hạn chế trên dữ liệu thực tế**: Dữ liệu được tiền xử lý kỹ lưỡng (loại bỏ ngoại lai, NaN), nhưng trong thực tế, dữ liệu y tế thường chứa nhiều nhiễu, giá trị thiếu, hoặc không cân bằng nghiêm trọng, có thể làm giảm hiệu suất của cả hai mô hình.

## 5.5. Hướng cải thiện

Dựa trên các nhược điểm của mô hình, dưới đây là các hướng cải thiện để nâng cao hiệu suất và khả năng ứng dụng thực tế:

### 5.5.1. Giảm False Negative

* **Điều chỉnh ngưỡng dự đoán (threshold)**: Hiện tại, cả hai mô hình sử dụng ngưỡng mặc định 0.5 để phân loại. Có thể giảm ngưỡng (ví dụ: 0.4 hoặc 0.3) để tăng Recall trên lớp "Disease", từ đó giảm False Negative, dù có thể làm tăng False Positive. Điều này cần được cân nhắc dựa trên yêu cầu y tế cụ thể.
* **Sử dụng chi phí bất đối xứng**: Áp dụng chi phí cao hơn cho False Negative trong hàm mất mát (loss function), ví dụ thông qua tham số class\_weight trong MLP hoặc Logistic Regression, để ưu tiên phát hiện bệnh.

### 5.5.2. Tăng tính tổng quát hóa

* **Thu thập thêm dữ liệu thực tế**: Dữ liệu hiện tại (376 mẫu sau tiền xử lý) tương đối nhỏ. Thu thập thêm dữ liệu y tế thực tế, đặc biệt từ các bệnh viện, sẽ giúp mô hình học tốt hơn và giảm phụ thuộc vào SMOTE.
* **Kiểm tra trên dữ liệu không cân bằng**: Thử nghiệm hiệu suất của mô hình trên tập dữ liệu thực tế (không áp dụng SMOTE) để đánh giá khả năng tổng quát hóa trong điều kiện thực. Có thể sử dụng các kỹ thuật khác như undersampling hoặc kết hợp SMOTE với Tomek Links để cải thiện chất lượng dữ liệu.

### 5.5.3. Giảm độ phức tạp và tăng tính diễn giải

* **Sử dụng mô hình đơn giản hơn trong Stacking**: Thay vì sử dụng MLP làm base model trong Stacking, có thể thay bằng một mô hình đơn giản hơn như Decision Tree hoặc Gradient Boosting (LightGBM), để giảm thời gian huấn luyện mà vẫn duy trì hiệu suất.
* **Áp dụng kỹ thuật diễn giải**: Sử dụng các phương pháp như SHAP (SHapley Additive exPlanations) thay vì Permutation Importance để diễn giải chi tiết hơn cách các đặc trưng ảnh hưởng đến dự đoán, tăng tính minh bạch và niềm tin từ bác sĩ.
* **Giảm số lượng tham số cần tinh chỉnh**: Với MLP, có thể giới hạn lưới tham số trong GridSearchCV (ví dụ: chỉ tinh chỉnh hidden\_layer\_sizes và learning\_rate\_init), giảm thời gian huấn luyện mà vẫn đảm bảo hiệu suất.

### 5.5.4. Xử lý dữ liệu thực tế

* **Xử lý giá trị thiếu và nhiễu**: Trong thực tế, dữ liệu y tế thường có giá trị thiếu (missing values) và nhiễu. Có thể áp dụng các kỹ thuật như điền giá trị thiếu bằng trung vị (median imputation) hoặc sử dụng mô hình dự đoán giá trị thiếu (KNN Imputation) để cải thiện chất lượng dữ liệu.
* **Feature engineering nâng cao**: Ngoài các đặc trưng đã tạo (chol\_age\_ratio, thalach\_age\_ratio), có thể thử nghiệm thêm các đặc trưng mới, ví dụ: tương tác giữa huyết áp và nhịp tim (trestbps\_thalach\_ratio), hoặc sử dụng kỹ thuật nhúng (embedding) cho các cột phân loại để tăng thông tin.

### 5.5.5. Thử nghiệm các mô hình khác

* **Mô hình Gradient Boosting**: Các mô hình như XGBoost, LightGBM, hoặc CatBoost thường hiệu quả trong bài toán y tế nhờ khả năng xử lý dữ liệu không đồng nhất và tự động học tầm quan trọng đặc trưng. Có thể thử thay thế Stacking Classifier hoặc sử dụng chúng làm base model.
* **Mạng nơ-ron sâu (Deep Neural Networks)**: Nếu có thêm dữ liệu, có thể thử nghiệm các mạng nơ-ron sâu với kiến trúc phức tạp hơn (nhiều tầng ẩn, dropout) để cải thiện hiệu suất, đặc biệt trên các bài toán phi tuyến.

### 5.5.6. Đánh giá lâm sàng

* **Hợp tác với bác sĩ**: Đưa mô hình vào thử nghiệm thực tế với dữ liệu lâm sàng mới và nhận phản hồi từ bác sĩ để đánh giá tính ứng dụng. Ví dụ, kiểm tra xem các đặc trưng quan trọng (chol\_age\_ratio, slope\_2.0) có ý nghĩa lâm sàng hay không.
* **Tích hợp vào hệ thống hỗ trợ quyết định**: Tích hợp mô hình vào một hệ thống hỗ trợ quyết định (Clinical Decision Support System) với giao diện thân thiện, cho phép bác sĩ điều chỉnh ngưỡng dự đoán và xem xét các yếu tố nguy cơ cụ thể.

# Phần 3 : Kết luận

Bài báo cáo đã trình bày một cách chi tiết quá trình xây dựng và đánh giá hai mô hình học máy, **Multi-Layer Perceptron (MLP)** và **Stacking Classifier**, để dự đoán bệnh tim dựa trên tập dữ liệu y tế đã được tiền xử lý. Các bước thực hiện bao gồm tải dữ liệu, tiền xử lý (loại bỏ trùng lặp, xử lý ngoại lai, feature engineering, cân bằng lớp bằng SMOTE), huấn luyện mô hình, và trực quan hóa kết quả. Cơ sở lý thuyết đã làm rõ nguyên lý hoạt động của các thuật toán (MLP, Random Forest, Stacking, Logistic Regression) và kỹ thuật hỗ trợ (GridSearchCV, SMOTE, Permutation Importance), tạo nền tảng vững chắc cho quá trình phân tích.

Kết quả đánh giá cho thấy cả hai mô hình đều đạt hiệu suất cao, với Accuracy 0.82 và F1 Score dao động từ 0.8281 (MLP) đến 0.8290 (Stacking). Stacking Classifier nhỉnh hơn một chút nhờ Recall cao hơn trên lớp "Disease" (0.82 so với 0.81 của MLP), giảm thiểu False Negative (36 so với 37), điều quan trọng trong y tế để tránh bỏ sót bệnh nhân. Các biểu đồ trực quan (Confusion Matrix, ROC Curve, Learning Curve, Feature Importance) xác nhận khả năng phân loại tốt (AUC ~0.8) và làm nổi bật vai trò của các đặc trưng như chol\_age\_ratio, chol, và slope\_2.0. Tuy nhiên, cả hai mô hình vẫn còn hạn chế, bao gồm tỷ lệ False Negative cao (18–19%), độ phức tạp tính toán (đặc biệt với Stacking), và khó diễn giải (đặc biệt với MLP).

Các ưu điểm của mô hình, như khả năng học phi tuyến, hiệu suất ổn định, và hỗ trợ ra quyết định y tế, đã được khẳng định. Tuy nhiên, nhược điểm như phụ thuộc vào dữ liệu cân bằng, thời gian huấn luyện lâu, và tính minh bạch hạn chế cần được khắc phục. Các hướng cải thiện được đề xuất bao gồm điều chỉnh ngưỡng dự đoán, thu thập thêm dữ liệu thực tế, thử nghiệm các mô hình như Gradient Boosting, và tích hợp mô hình vào hệ thống hỗ trợ quyết định lâm sàng. Những cải tiến này không chỉ nâng cao hiệu suất mà còn tăng tính ứng dụng thực tế của mô hình trong chẩn đoán bệnh tim.

Tóm lại, nghiên cứu này đã thành công trong việc xây dựng và đánh giá hai mô hình học máy hiệu quả để dự đoán bệnh tim, với Stacking Classifier là lựa chọn tiềm năng hơn nhờ Recall cao hơn. Tuy nhiên, để áp dụng vào thực tế, cần tiếp tục cải tiến mô hình và hợp tác với chuyên gia y tế để đảm bảo tính chính xác và an toàn. Nghiên cứu mở ra hướng đi mới cho việc sử dụng trí tuệ nhân tạo trong y học, đặc biệt trong việc hỗ trợ chẩn đoán sớm các bệnh lý tim mạch.

# TÀI LIỆU THAM KHẢO

1. **Tài liệu tham khảo**
2. Fedesoriano. (2021). Heart Failure Prediction Dataset. Kaggle. Retrieved from <https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction>
3. Johnsmith88. (2022). Heart Disease Dataset. Kaggle. Retrieved from <https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset>
4. Mammadov, R. (2023). Heart Disease Prediction. Kaggle. Retrieved from <https://www.kaggle.com/datasets/rashadrmammadov/heart-disease-prediction>
5. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
6. Fisher, A., Rudin, C., & Dominici, F. (2019). All Models are Wrong, but Many are Useful: Learning a Variable's Importance with Random Forests. *Annals of Applied Statistics*, 13(3), 1630–1656.
7. Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (2nd ed.). Springer.
8. World Health Organization. (2021). Cardiovascular Diseases (CVDs). Retrieved from <https://www.who.int/health-topics/cardiovascular-diseases>
9. Scikit-learn Documentation. (2023). Machine Learning with Scikit-learn. Retrieved from <https://scikit-learn.org/stable/>