BÙI ĐỨC ANH

KHOA HỌC MÁY TÍNH

**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

---

# ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC

# NGÀNH KHOA HỌC MÁY TÍNH

# PHÂN TÍCH VÀ DỰ ĐOÁN NGUY CƠ BỆNH SUY TIM BẰNG MÔ HÌNH XGBOOST


**GVHD : Th.S Nhữ Văn Kiên**

**Sinh viên : Bùi Đức Anh**

**Mã số sinh viên : 2021605978**


**Hà Nội – Năm 2025**

**BỘ CÔNG THƯƠNG**
**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

---

# ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC
# NGÀNH KHOA HỌC MÁY TÍNH

# PHÂN TÍCH VÀ DỰ ĐOÁN NGUY CƠ BỆNH SUY TIM BẰNG MÔ HÌNH XGBOOST

**GVHD : Th.S Nhữ Văn Kiên**

**Sinh viên : Bùi Đức Anh**

**Mã số sinh viên : 2021605978**

**Hà Nội – Năm 2025**

# LỜI CẢM ƠN

Trước tiên, em xin được gửi lời cảm ơn chân thành và sâu sắc nhất đến Ban giám hiệu, quý Thầy/Cô Trường Đại học Công Nghiệp Hà Nội, đặc biệt là Trường Công nghệ thông tin và Truyền thông đã tận tình giảng dạy, hướng dẫn và truyền đạt kiến thức và kinh nghiệm quý báu và tạo điều kiện để em có thể hoàn thành tốt Đồ án tốt nghiệp và cũng là hành trang cho em trong công việc sau này.

Đặc biệt, em xin được bày tỏ lòng biết ơn chân thành tới Thầy Nhữ Văn Kiên, người đã trực tiếp hướng dẫn, chỉ bảo tận tâm và đồng hành cùng em trong suốt quá trình nghiên cứu, xây dựng và hoàn thiện đề tài "Phân tích và dự đoán nguy cơ bệnh suy tim bằng mô hình XGBoost".

Em cũng xin được gửi lời cảm ơn tới gia đình, người thân và bạn bè đã luôn động viên và hỗ trợ và tiếp thêm động lực cho em trong suốt thời gian học tập và thực hiện đồ án.

Trong quá trình thực hiện đồ án, do kiến thức và thời gian còn hạn chế, dù đã rất cố gắng nhưng chắc chắn sẽ không thể tránh khỏi những thiếu sót. Em rất mong nhận được sự đóng góp ý kiến từ Quý Thầy Cô để em có thể hoàn thiện sản phẩm cũng như tích luỹ thêm kinh nghiệm cho con đường sau này.

Em xin trân trọng cảm ơn!

Sinh viên thực hiện

Bùi Đức Anh

# MỤC LỤC

# DANH MỤC HÌNH ẢNH

# DANH MỤC BẢNG BIỂU

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

| | |
|---|---|
| SVM | Support Vector Machine - Vec-tơ máy hỗ trợ |
| DT | Decision Tree - Cây quyết định |
| AI | Artificial Intelligence - Trí tuệ nhân tạo |
| ML | Machine Learning - Học máy |
| CART | Classification and Regression Trees |
| CHẠID | Chi-squared Automatic Interaction Detection |

# MỞ ĐẦU

## 1. Lý do chọn đề tài

Suy tim là một hội chứng lâm sàng nghiêm trọng, ảnh hưởng đến khoảng 64 triệu người trên toàn thế giới (theo số liệu của Global Burden of Disease 2022) và là nguyên nhân gây hơn 9 triệu ca tử vong mỗi năm [10]. Tại Việt Nam, tỷ lệ mắc suy tim ngày càng tăng, đặc biệt ở nhóm người trên 60 tuổi, chủ yếu do các bệnh lý nền như tăng huyết áp, bệnh mạch vành, đái tháo đường. Việc chẩn đoán sớm nguy cơ suy tim đóng vai trò then chốt trong giảm tỷ lệ tử vọng, giảm gánh nặng chi phí y tế và nâng cao chất lượng cuộc sống cho bệnh nhân.

Trong bối cảnh dữ liệu y tế ngày càng phong phú, các thuật toán học máy hiện đại như XGBoost nổi bật nhờ khả năng xử lý dữ liệu phi tuyến, tối ưu hóa mô hình và đạt độ chính xác cao ngay cả với tập dữ liệu có kích thước vừa và nhỏ. Việc áp dụng XGBoost trong phân tích nguy cơ suy tim không chỉ giúp xác định các yếu tố tác động quan trọng, mà còn xây dựng mô hình dự đoán hỗ trợ bác sĩ ra quyết định sớm và chính xác hơn. Chính vì vậy, em lựa chọn đề tài "Phân tích và dự đoán nguy cơ suy tim bằng mô hình XGBoost" nhằm kết hợp sức mạnh của trí tuệ nhân tạo với dữ liệu y khoa, góp phần cải thiện hiệu quả chẩn đoán và phòng ngừa bệnh.

## 2. Mục tiêu nghiên cứu

- Tìm hiểu, nghiên cứu và áp dụng triển khai xây dựng mô hình dự đoán nguy cơ mắc bệnh suy tim dựa trên các bộ dữ liệu lâm sàng, bao gồm các mô hình XGBoost, Decision Tree, SVM, …

- Đánh giá và lựa chọn các yếu tố lâm sàng có ảnh hưởng lớn đến nguy cơ suy tim.

- Đánh giá, so sánh độ chính xác, các thông số kĩ thuật, qua đó lựa chọn mô hình hiệu quả nhất với bộ dữ liệu.

- Xây dựng chương trình demo có chức năng dự đoán nguy cơ mắc bệnh suy tim dựa trên các thông số dữ liệu y tế, tự động phân tích và dự báo kết quả dựa trên mô hình học máy.

**3. Đối tượng nghiên cứu**

- Ứng dụng các mô hình học máy trong việc phân tích và dự đoán nguy cơ suy tim dựa trên dữ liệu lâm sàng.

- Tìm ra những yếu tố có ảnh hưởng lớn đến nguy cơ suy tim.

- Xây dựng chương trình demo dự đoán nguy cơ suy tim dựa trên dữ liệu lâm sàng.

**4. Phạm vi nghiên cứu**

- Đề tài tập trung vào phân tích và dự đoán nguy cơ suy tim dựa trên dữ liệu lâm sàng.

- Xây dựng chương trình demo có chức năng dự đoán nguy cơ suy tim, cho phép người dùng nhập các thông số y tế, xử lý và hiển thị kết quả.

- Đề tài góp phần nghiên cứu và ứng dụng các kỹ thuật học máy trong lĩnh vực y tế - một lĩnh vực quan trọng và đang phát triển nhanh chóng với nhiều thách thức như dữ liệu phức tạp, yêu cầu độ chính xác rất cao và có tính ứng dụng thực tiễn.

**5. Bố cục đề tài**

Bản báo cáo gồm 4 chương như sau:

- **Chương 1: Tổng quan và cơ sở lý thuyết:** Chương này giới thiệu về học máy, giới thiệu về bài toán phân tích dữ liệu và dự báo trong học máy truyền thống.

- **Chương 2: Một số thuật toán giải quyết bài toán:** Chương này giới thiệu các kỹ thuật học máy hiện đại, bao gồm mô hình XGBoost được áp dụng để giải quyết bài toán.

- **Chương 3: Kết quả thực nghiệm:** Chương này mô tả các bước triển khai thực nghiệm, bao gồm tập dữ liệu, phân tích, xử lý và trực quan hóa dữ liệu và xây dựng các mô hình, so sánh và đánh giá các mô hình.

- **Chương 4: Sản phẩm Demo:** Chương này trình bày các công cụ và các bước xây dựng chương trình demo sản phẩm.

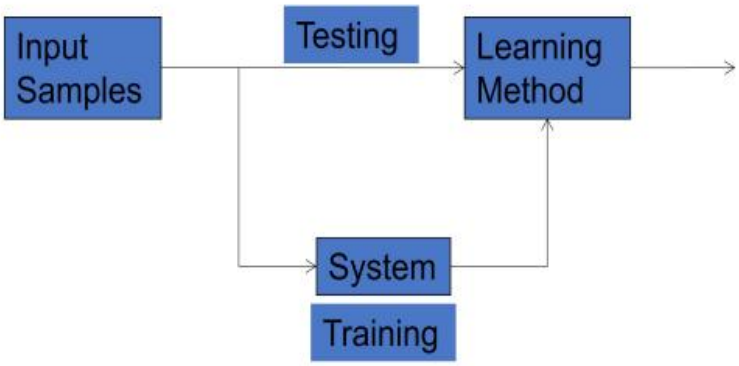
# CHƯƠNG 1: TỔNG QUAN VÀ CƠ SỞ LÝ THUYẾT

## 1.1. Tổng quan về học máy

### *1.1.1. Giới thiệu về học máy*

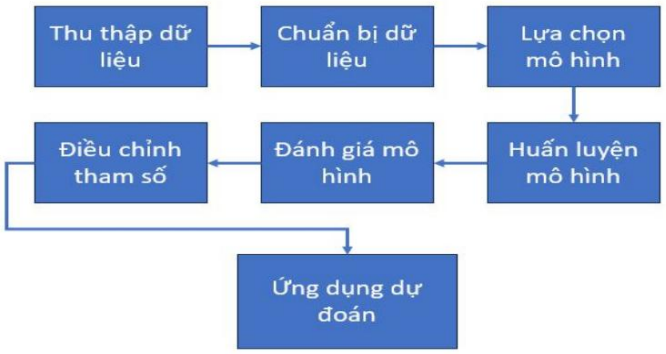
Học máy (Machine Learning - ML) là một nhánh của trí tuệ nhân tạo (Artificial Intelligence - AI), trong đó các máy tính được lập trình để có thể tự động học hỏi từ dữ liệu, thay vì phải được lập trình bằng các quy tắc cụ thể. Machine Learning cho phép máy tính tự động học hỏi và cải thiện từ kinh nghiệm mà không cần sự can thiệp trực tiếp của con người [17].

Quá trình học bao gồm việc sử dụng dữ liệu đầu vào để huấn luyện mô hình, sau đó kiểm tra độ hiệu quả của mô hình thông qua dữ liệu kiểm tra. Cuối cùng, phương pháp học được áp dụng để tối ưu hóa các mô hình học máy cho đến khi đạt được kết quả mong muốn.




*Hình 1.1. Định nghĩa học máy*

### *1.1.2. Các bước chính của mô hình học máy*




*Hình 1.2. Các bước chính của mô hình học máy*

Quá trình xây dựng, phát triển và ứng dụng mô hình học máy được thực hiện theo các bước chính sau:

- Thu thập dữ liệu: Thu thập dữ liệu từ các nguồn khác nhau như cơ sở dữ liệu, cảm biến, API, hoặc từ nguồn trực tuyến. Dữ liệu yêu cầu cần phải có kích thước đủ lớn và đa dạng để mô hình học máy có thể học được các đặc trưng cần thiết.

- Chuẩn bị dữ liệu: Sau khi thu thập, dữ liệu cần được làm sạch và tiền xử lý bao gồm: loại bỏ dữ liệu bị thiếu, khuyết hoặc trùng lặp, xử lý các giá trị ngoại lệ, chọn lọc đặc trưng có ảnh hưởng lớn tới kết quả, chuẩn hóa dữ liệu, và chia tập dữ liệu thành các tập huấn luyện và kiểm tra. Quá trình này đảm bảo dữ liệu được chuẩn bị đầy đủ cho quá trình huấn luyện.

- Lựa chọn mô hình: Lựa chọn mô hình học máy phù hợp để giải quyết yêu cầu của bài toán. Tùy thuộc vào bản chất dữ liệu và bài toán cụ thể, có thể lựa chọn các thuật toán và các mô hình học máy khác nhau.

- Huấn luyện mô hình: Sử dụng tập dữ liệu đã được chuẩn bị để đưa vào huấn luyện mô hình. Trong quá trình này, mô hình sẽ học từ dữ liệu và điều chỉnh các tham số để tối ưu hóa hiệu suất dự đoán.

- Đánh giá mô hình: Sau khi huấn luyện, mô hình cần được đánh giá thông qua tập dữ liệu kiểm tra. Các chỉ số đánh giá như độ chính xác, độ nhạy, độ đặc hiệu hoặc F1 - score được sử dụng để đo lường hiệu suất của mô hình và xác định liệu mô hình có phù hợp và đáp ứng được yêu cầu bài toán hay không.

- Điều chỉnh tham số: Nếu mô hình chưa đạt hiệu suất mong muốn, cần điều chỉnh các siêu tham số ( hyperparameters) hoặc thay đổi mô hình để cải thiện. Quá trình này có thể sử dụng các phương pháp như tìm kiếm lưới ( Grid Search), tìm kiếm ngẫu nhiên ( Random Search) để tối ưu hóa.

- Ứng dụng dự đoán: Sau khi mô hình đạt hiệu suất mong muốn, nó sẽ được triển khai vào hệ thống thực tế để thực hiện dự đoán trên dữ liệu mới.

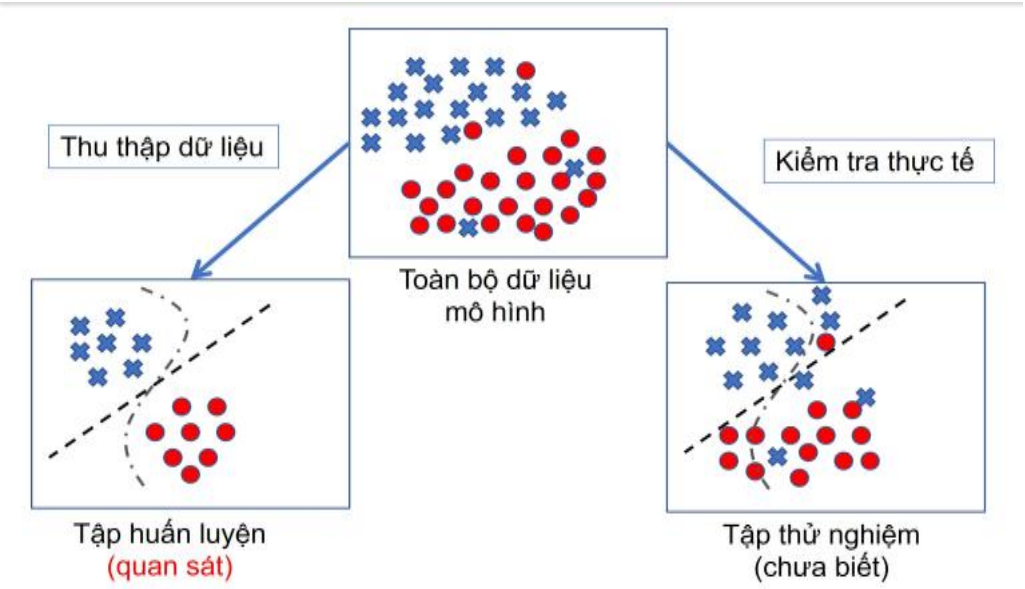
Mô hình sẽ tiếp tục được theo dõi và cập nhật khi cần thiết để duy trì hoạt động hiệu quả.

Quá trình này là một vòng lặp liên tục khép kín, với các bước điều chỉnh tham số và đánh giá diễn ra liên tục để đảm bảo mô hình luôn đáp ứng nhu cầu và cải thiện theo thời gian.

### *1.1.3. Huấn luyện và kiểm chứng*

- Huấn luyện: là quá trình làm cho máy có thể "học" từ dữ liệu.

- Kiểm chứng: Là quá trình đánh giá khả năng của mô hình sau khi huấn luyện.




*Hình 1.3. Huấn luyện và kiểm chứng*

- Tập huấn luyện (Training Set): Một phần của tập dữ liệu sẽ được sử dụng làm tập huấn luyện, mô hình sẽ học các đặc trưng và mẫu từ tập dữ liệu này. Tập huấn luyện chứa các điểm dữ liệu mà mô hình đã quan sát, cho phép mô hình học cách phân loại giữa các lớp khác nhau.

- Tập kiểm chứng (Test Set): Một phần các của tập dữ liệu, chưa được mô hình biết đến sẽ được sử dụng để kiểm tra hiệu suất của mô hình. Tập thử nghiệm cho phép đánh giá xem mô hình có thể phân loại dữ liệu mới chưa từng thấy trước đó một cách chính xác hay không.

**Định lý 1: No free-lunch**

- Tập huấn luyện (Training Set) và tập thử nghiệm (Test Set) tuân theo cùng phân bố thống kê.

- Thực tế, dữ liệu có thể chứa nhiễu hoặc tuân theo một số giả thiết nhất định.

### *1.1.4. Quá trình học*

Là việc máy tính sẽ học một hàm $f : x \rightarrow y$ để biến một đầu vào $x$ thành một đầu ra $y$. Trong đó, $x$ là các quan sát hay các kinh nghiệm trong quá khứ và $y$ là các dự đoán, kiến thức mới hay tri thức mới.

**Chú ý:**

- Hàm $f$ còn có thể gọi là một mô hình (Model).

- Đôi khi giả thuyết dữ liệu thường tuân theo hoặc được tạo ra từ một mô hình nào đó.

- Học một mô hình có nghĩa là học hay tìm kiếm những tham số của mô hình nào đó.

- Quá trình học xuất phát từ các mẫu huấn luyện (tập học, tập huấn luyện,

tập quan sát, …) $\{\{x_1, x_2, \ldots, x_n\}, \{y_1, y_2, \ldots, y_n\}\}$.

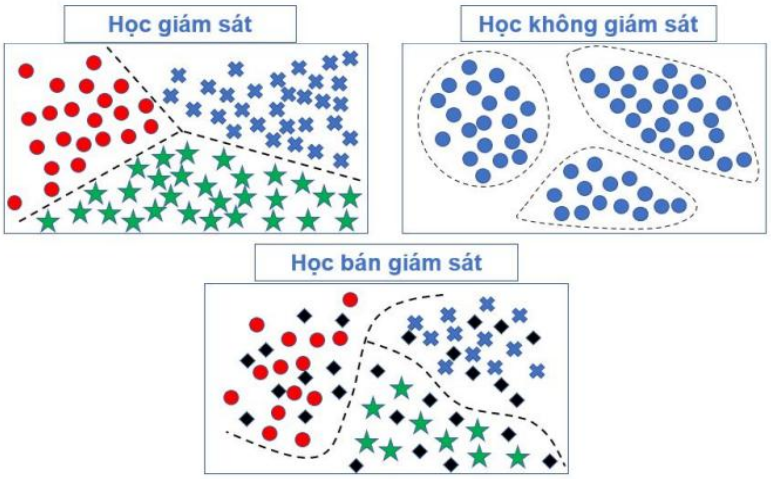
**Kết thúc quá trình học sẽ thu được:**

- Tri thức mới, kinh nghiệm mới, mô hình.

- Tạo ra hàm $y = f(x)$ là kết quả của quá trình học.

Việc xác định hàm $y = f(x)$ là một bài toán phức tạp do tính rộng lớn của không gian.

### *1.1.5. Phân nhánh học máy*




*Hình 1.4. Các phân nhánh của học máy*

*Học có giám sát (Supervised machine learning)* là mô hình học máy được huấn luyện trên tập dữ liệu đã được gán nhãn (labelled data set), bao gồm các thẻ mô tả từng loại dữ liệu. Điều này có nghĩa là trong quá trình huấn luyện, mỗi mẫu dữ liệu đầu vào (input) đã có sẵn một đầu ra mong muốn (output) tương ứng, giúp mô hình học được mối quan hệ giữa đầu vào và đầu ra.

- Học một hàm $y = f(x)$ từ tập dữ liệu

$$\{\{x_1, x_2, \ldots, x_n\}, \{y_1, y_2, \ldots, y_n\}\}$$

- Ví dụ bài toán: Nhận dạng khuôn mặt, nhận dạng biển số xe, phát hiện vật thể, nhận dạng chữ viết,…

*Học không giám sát (Unsupervised machine learning)* là mô hình học máy được huấn luyện trên tập dữ liệu không có nhãn. Điều này có nghĩa là trong quá trình huấn luyện, không có chỉ dẫn rõ ràng về kết quả đầu ra mong muốn cho mỗi mẫu dữ liệu. Mục tiêu của unsupervised learning là để phát hiện cấu trúc, mối quan hệ, hoặc các mẫu ẩn trong dữ liệu.

- Học một hàm $y = f(x)$ từ tập dữ liệu

$$\{x_1, x_2, \ldots, x_n\}$$

- $y$ có thể là một cụm dữ liệu, một cấu trúc ẩn hoặc một xu thế.

- Ví dụ phương pháp: Phân cụm (Clustering), phát hiện bất thường (Anomaly Detection), phân tích liên kết (Association Analysis),…

- Ví dụ về bài toán: Phân cụm khách hàng, phân tích hành vi,…

*Học bán giám sát (Semi-supervised machine learning)* là một phương pháp trong trí tuệ nhân tạo mà sử dụng cả dữ liệu có nhãn (labeled) và không có nhãn (unlabeled) trong quá trình huấn luyện. Điều này cho phép hệ thống học hỏi một cách hiệu quả hơn, tận dụng lượng lớn dữ liệu không có nhãn mà thường sẵn có và ít tốn kém hơn so với dữ liệu đã được gắn nhãn. Phương pháp này rất hữu ích trong các tình huống mà việc lấy nhãn cho dữ liệu là tốn kém hoặc khó khăn.

Ví dụ bài toán: Phân loại văn bản, nhận dạng giọng nói…

### *1.1.6. Ưu điểm của học máy*

- Khả năng học tập từ dữ liệu: Học máy có thể học và cải thiện từ dữ liệu mà không cần phải được lập trình cụ thể cho từng tác vụ.

- Độ chính xác cao: Các mô hình học máy, khi được huấn luyện đúng cách, có thể cung cấp kết quả với độ chính xác rất cao.

- Tính tự động hóa cao: Học máy có thể tự động hóa các tác vụ phức tạp và giảm thiểu sự can thiệp của con người.

- Xử lý dữ liệu lớn: Các thuật toán học máy có khả năng xử lý và phân tích lượng dữ liệu lớn, từ đó đưa ra những hiểu biết và thông tin quan trọng.

- Đa dạng hóa các ứng dụng: Áp dụng rộng rãi trong nhiều lĩnh vực như nhận diện hình ảnh, xử lý ngôn ngữ tự nhiên, dự đoán và phân tích dữ liệu, quản lý rủi ro, và nhiều ứng dụng khác.

### *1.1.7. Nhược điểm của học máy*

- Yêu cầu lượng dữ liệu lớn: Để đạt được độ chính xác cao, học máy thường yêu cầu lượng dữ liệu lớn để huấn luyện mô hình.

- Phụ thuộc vào chất lượng dữ liệu: Chất lượng dữ liệu đầu vào sẽ ảnh hưởng trực tiếp đến chất lượng của mô hình học máy.

- Vấn đề overfitting và underfitting: Đây là hiện tượng mô hình học máy quá phù hợp hoặc không đủ phù hợp với dữ liệu huấn luyện.

- Chi phí huấn luyện cao: Một số loại mô hình phức tạp có thể yêu cầu nhiều tài nguyên tính toán và thời gian để huấn luyện.

Tuy nhiên, với sự phát triển của công nghệ và nghiên cứu, nhiều trong số các nhược điểm này đều đang được giải quyết và cải thiện. Học máy vẫn tiếp tục là một công cụ mạnh mẽ trong nhiều lĩnh vực ứng dụng và nghiên cứu hiện đại.

### *1.1.8. Một số ứng dụng trong học máy*

- Hệ thống khuyến nghị (Đề xuất phim, nhạc hoặc các sản phẩm trên các nền tảng mạng xã hội,…).

- Hệ thống dự đoán lưu lượng giao thông (Hỗ trợ điều phối giao thông đô thị, dự đoán thời gian di chuyển,…).

- Xử lý ngôn ngữ tự nhiên (chatbot, trợ lý ảo như ChatGPT, dịch máy,…).

- Điều khiển và tối ưu hóa quy trình sản xuất (Giám sát dây chuyền sản xuất, dự đoán bảo trì máy móc,…).

- Chẩn đoán và phát hiện bệnh dựa trên hình ảnh y khoa (Phát hiện khối u từ ảnh MRI/CT, chẩn đoán bệnh dựa trên chỉ số y tế,…).

- Dự đoán thị trường tài chính, phân tích rủi ro tín dụng (Hỗ trợ quyết định đầu tư, phát hiện gian lận trên sàn thương mại,…).

- Phân tích dữ liệu khách hàng để đề xuất sản phẩm phù hợp (Cá nhân hóa trải nghiệm mua sắm,…).

- Nhận diện hình ảnh và video (ứng dụng trong an ninh, các dịch vụ xã hội, phân loại hình ảnh,…).

## 1.2. Giới thiệu về học máy trong y tế

### *1.2.1. Giới thiệu chung*

Trong bối cảnh y tế hiện đại, sự gia tăng nhanh chóng của dữ liệu bệnh nhân cùng với sự phát triển của công nghệ tính toán và lưu trữ đã tạo điều kiện thuận lợi cho việc ứng dụng các phương pháp học máy. Trong những năm gần đây, học máy đã trở thành một trong những công nghệ đột phá được ứng dụng mạnh mẽ trong y tế. Với khả năng phân tích dữ liệu quy mô lớn và phát hiện ra các mối quan hệ tiềm ẩn mà con người khó nhận thấy, học máy đang mở ra hướng đi mới trong chẩn đoán, điều trị và quản lý bệnh.

Một trong những giá trị cốt lõi của học máy trong y tế là khả năng học hỏi từ dữ liệu lâm sàng và dữ liệu hình ảnh y khoa. Các bệnh viện, phòng khám và trung tâm nghiên cứu y tế hiện nay tạo ra một khối lượng dữ liệu khổng lồ, bao gồm hồ sơ bệnh án điện tử (EHR), kết quả xét nghiệm, hình ảnh y học (X-quang, MRI, CT, siêu âm), dữ liệu di truyền và dữ liệu từ thiết bị theo dõi sức khỏe. Học máy có thể xử lý đồng thời nhiều nguồn dữ liệu này, từ đó hỗ trợ bác sĩ trong việc đưa ra chẩn đoán nhanh chóng và chính xác hơn.

Bên cạnh tiềm năng to lớn, việc ứng dụng học máy trong y tế vẫn đối mặt với nhiều thách thức. Đầu tiên là vấn đề chất lượng và tính toàn vẹn của dữ liệu, khi dữ liệu y tế thường thiếu đồng bộ, có sai sót hoặc bị phân tán ở nhiều nguồn khác nhau. Thứ hai là yêu cầu về tính minh bạch và khả năng giải thích: trong y học, các mô hình không chỉ cần chính xác mà còn phải đưa ra lý do rõ ràng để bác sĩ tin tưởng. Ngoài ra, các yếu tố về bảo mật và quyền riêng tư dữ liệu cũng là thách thức lớn, đòi hỏi tuân thủ các quy định nghiêm ngặt để bảo vệ thông tin bệnh nhân.

Nhìn chung, học máy đã và đang khẳng định vai trò quan trọng trong y tế, mang lại những tiến bộ rõ rệt trong chẩn đoán, điều trị và chăm sóc sức khỏe. Trong tương lai, với sự phát triển của trí tuệ nhân tạo nói chung và học máy nói riêng, lĩnh vực y tế hứa hẹn sẽ đạt được những bước tiến vượt bậc,

hướng tới một hệ thống chăm sóc sức khỏe thông minh, chính xác và cá nhân hóa hơn.

### *1.2.2. Ứng dụng học máy trong dự đoán nguy cơ bệnh suy tim*

Suy tim là một trong những bệnh lý tim mạch nguy hiểm, có tỷ lệ tử vong và tái nhập viện cao, gây ra gánh nặng lớn cho hệ thống y tế. Việc phát hiện sớm và dự đoán nguy cơ suy tim đóng vai trò quan trọng trong việc điều trị kịp thời, nâng cao chất lượng cuộc sống và giảm chi phí chăm sóc y tế cho bệnh nhân. Tuy nhiên, các phương pháp chẩn đoán truyền thống chủ yếu dựa vào kinh nghiệm lâm sàng, xét nghiệm cận lâm sàng và hình ảnh y khoa, đôi khi chưa đủ khả năng để phát hiện những dấu hiệu tiềm ẩn hoặc dự đoán nguy cơ chính xác.

Trong bối cảnh đó, học máy mang đến một cách tiếp cận mới trong phân tích và dự đoán bệnh suy tim. Với khả năng xử lý khối lượng dữ liệu lớn từ hồ sơ bệnh án điện tử, kết quả xét nghiệm sinh hóa, thông tin di truyền, tín hiệu điện tim (ECG) và dữ liệu theo dõi từ xa, các thuật toán học máy có thể phát hiện những mối quan hệ phức tạp giữa các yếu tố nguy cơ mà phương pháp truyền thống khó nhận biết. Nhờ đó, học máy hỗ trợ xác định sớm những bệnh nhân có nguy cơ cao, dự đoán khả năng tái phát, cũng như ước lượng tiên lượng sống của bệnh nhân.

Các nghiên cứu trong và ngoài nước đã cho thấy học máy có khả năng vượt trội so với phương pháp thống kê truyền thống trong việc nhận diện yếu tố nguy cơ và dự đoán kết quả điều trị. Một số mô hình còn cho phép bác sĩ theo dõi sự thay đổi rủi ro của bệnh nhân theo thời gian, từ đó đưa ra quyết định kịp thời hơn.

Tuy nhiên, việc ứng dụng học máy trong dự đoán nguy cơ suy tim vẫn đối mặt với những thách thức. Vấn đề về chất lượng và sự đồng nhất của dữ liệu còn hạn chế, trong khi đó nhu cầu về khả năng giải thích kết quả lại rất quan trọng để bác sĩ có thể tin tưởng và ứng dụng trong lâm sàng. Do vậy, các nghiên cứu hiện nay không chỉ tập trung nâng cao độ chính xác dự đoán, mà

còn hướng tới phát triển những mô hình có tính minh bạch, dễ diễn giải và phù hợp với thực tiễn y tế.

Học máy đã chứng minh được tiềm năng lớn trong dự đoán nguy cơ suy tim, mở ra triển vọng xây dựng các hệ thống cảnh báo sớm, hỗ trợ ra quyết định lâm sàng và cá nhân hóa điều trị, góp phần quan trọng vào việc nâng cao hiệu quả chăm sóc sức khỏe tim mạch.

### *1.2.3. Thuận lợi*

- Khối lượng dữ liệu y tế phong phú và đa dạng: Hồ sơ bệnh án, xét nghiệm, hình ảnh,…

- Hạ tầng tính toán mạnh mẽ: Điện toán đám mây, GPU.

- Cộng đồng nghiên cứu và y tế: quan tâm thúc đẩy phát triển các ứng dụng thông minh trong chẩn đoán và điều trị.

- Sự phát triển nhanh chóng của các công cụ, thư viện học máy hỗ trợ nghiên cứu và triển khai.

### *1.2.4. Khó khăn*

- Chất lượng dữ liệu: Dữ liệu y tế thường không đồng bộ, chứa nhiều giá trị thiếu, sai lệch hoặc phân tán…

- Quyền riêng tư và bảo mật: Việc thu thập và xử lý dữ liệu bệnh nhân đòi hỏi tuân thủ nghiêm ngặt các quy định bảo mật, gây khó khăn trong chia sẻ và khai thác dữ liệu.

- Khả năng giải thích mô hình: Các mô hình học máy hoạt động phức tạp nên khó giải thích kết quả, khó nhận được sự tin tưởng từ các bác sĩ.

- Tính khái quát hóa: Mô hình được huấn luyện trên một tập dữ liệu cụ thể có thể không hoạt động tốt trên tập dữ liệu khác.

- Ứng dụng thực tiễn: Phần lớn nghiên cứu mới chỉ dừng lại ở môi trường thử nghiệm, thiếu các hệ thống tích hợp vào quy trình lâm sàng.

- Nguồn lực triển khai: Các bệnh viện hoặc cơ sở y tế với hạ tầng hạn chế khó có khả năng triển khai và duy trì hệ thống học máy phức tạp.s

## 1.3. Giới thiệu bài toán

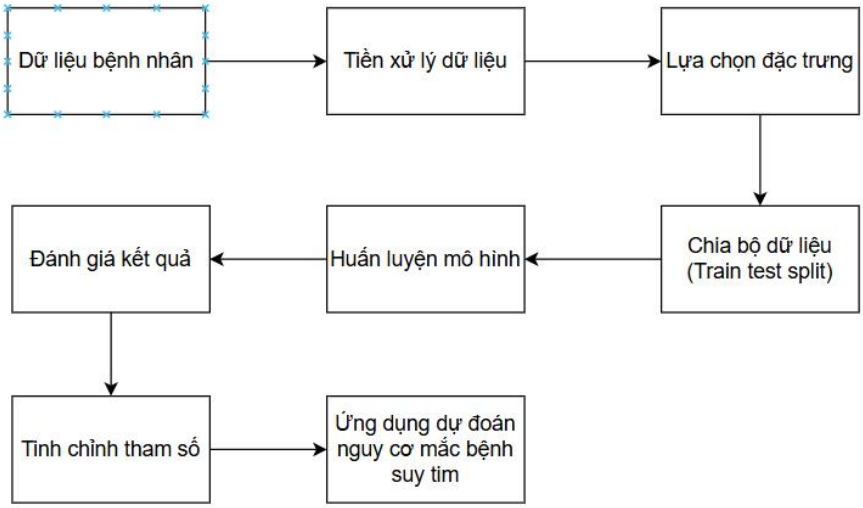
### *1.3.1. Khái quát bài toán*

Bệnh tim vẫn luôn đứng đầu danh sách các nguyên nhân gây tử vong trên toàn cầu. Theo Tổ chức Y tế Thế giới, thống kê năm 2022 cho thấy khoảng 19,8 triệu người chết vì các bệnh tim mạch và đột quỵ chiếm xấp xỉ 32% tỉ lệ tử vong toàn trên toàn thế giới, trong đó hơn 75% xảy ra ở các quốc gia có thu nhập trung bình và thấp. Đáng chú ý, đau tim và đột quỵ chiếm khoảng 85% tổng số ca tử vong do các bệnh tim mạch [9].

Việc đánh giá triệu chứng lâm sàng kết hợp với khám sức khỏe thường là bước đầu tiên để chẩn đoán bệnh tim. Một số yếu tố nguy cơ quan trọng bao gồm tuổi tác, tiền sử gia đình, hút thuốc, mức cholesterol cao, ít vận động, huyết áp cao, béo phì, tiểu đường và căng thẳng. Thay đổi thói quen sống như ngừng hút thuốc, giảm cân, tập thể dục đều đặn và quản lý căng thẳng có thể giúp giảm nguy cơ mắc bệnh.

Chẩn đoán bệnh tim hiện nay thường dựa trên tiền sử bệnh, khám lâm sàng và các xét nghiệm hình ảnh như điện tâm đồ, siêu âm tim, MRI tim, hoặc xét nghiệm máu. Việc điều trị có thể bao gồm thay đổi lối sống, sử dụng thuốc, các thủ thuật y tế như nong mạch vành, phẫu thuật bắc cầu động mạch hoặc cấy ghép thiết bị hỗ trợ tim như máy tạo nhịp hoặc máy khử rung. Với sự phát triển mạnh mẽ của các hệ thống quản lý dữ liệu y tế, việc xây dựng các mô hình dự đoán bệnh tim dựa trên lượng lớn dữ liệu bệnh nhân trở nên khả thi. Học máy, với khả năng phân loại và phân tích dữ liệu từ nhiều góc độ khác nhau, đang được ứng dụng để chuyển các tập dữ liệu y tế phức tạp thành thông tin hữu ích, hỗ trợ chẩn đoán và ra quyết định lâm sàng.

Quy trình giải quyết bài toán dựa trên các bước chính của mô hình học máy:




*Hình 1.5. Quy trình của bài toán*

### *1.3.2. Mục tiêu của bài toán*

Mục tiêu chính của bài toán dự đoán nguy cơ suy tim là xây dựng một mô hình học máy có khả năng phân tích dữ liệu lâm sàng của bệnh nhân để nhận diện những yếu tố quan trọng ảnh hưởng đến bệnh suy tim và dự đoán chính xác nguy cơ suy tim trong tương lai. Qua đó, mô hình hỗ trợ bác sĩ trong việc phát hiện sớm, đưa ra quyết định điều trị kịp thời và góp phần nâng cao hiệu quả chăm sóc sức khỏe, giảm thiểu rủi ro và chi phí y tế. Cụ thể, hệ thống cần đạt được những yêu cầu sau:

- Dự đoán chính xác: Mô hình có độ chính xác lớn hơn 90%.

- Thời gian xử lý nhanh: Đảm bảo hệ thống có khả năng xử lý và đưa ra kết quả dự đoán trong thời gian ngắn.

- Dễ sử dụng: Giao diện thân thiện, dễ sử dụng không yêu cầu người dùng có kiến thức chuyên sâu về công nghệ cũng có thể sử dụng hệ thống.

- Khả năng mở rộng: Hệ thống có khả năng cập nhật và mở rộng để nhận diện thêm các yếu tố ảnh hướng đến nguy cơ suy tim.

### *1.3.3. Mô tả dữ liệu đầu vào, đầu ra của bài toán*

Dữ liệu đầu vào của bài toán dự báo nguy cơ suy tim được lấy từ hồ sơ bệnh án và kết quả xét nghiệm lâm sàng của bệnh nhân. Các đặc trưng thường

bao gồm thông tin nhân khẩu học (tuổi, giới tính, chỉ số BMI), các chỉ số sinh học quan trọng. Đây là những yếu tố đầu vào có ảnh hưởng trực tiếp hoặc gián tiếp đến tình trạng tim mạch của bệnh nhân, bao gồm:

- age: Tuổi
- sex: Giới tính
- cp: Loại đau ngực
- trestbps: Huyết áp khi nghỉ (mm Hg)
- chol: Mức cholesterol trong huyết thanh (mg/dl)
- fbs: Đường huyết lúc đói
- restecg: Kết quả điện tâm đồ khi nghỉ
- thalach: Nhịp tim tối đa đạt được
- exang: Đau thắt ngực khi gắngức
- oldpeak: Độ chênh ST so với khi nghỉ
- slope: Độ dốc của đoạn ST trong điện tâm đồ khi gắng sức
- ca: Số lượng mạch máu lớn được nhuộm màu
- thal: Kết quả kiểm tra Thalessemia
- target: Biến mục tiêu

Dữ liệu đầu ra của mô hình là biến mục tiêu thể hiện khả năng xảy ra suy tim. Thông thường, đầu ra được thiết kế ở dạng giá trị nhị phân:

- 0: Biểu thị bệnh nhân không mắc bệnh suy tim hoặc nguy cơ suy tim thấp.
- 1: Biểu thị bệnh nhân mắc bệnh suy tim hoặc nguy cơ suy tim cao.

Đầu ra còn được biểu diễn dưới dạng xác suất, ví dụ như bệnh nhân có 70% nguy cơ suy tim, từ đó cung cấp thông tin chi tiết hơn giúp bác sĩ đưa ra quyết định điều trị và theo dõi phù hợp.

Bên cạnh các giá trị được biểu diễn dưới dạng nhị phân và xác suất là biểu đồ các đặc trưng quan trọng giúp giải thích cho kết quả dự đoán và tăng độ tin cậy.

### *1.3.4. Ứng dụng*

- Hỗ trợ chẩn đoán sớm: Giúp bác sĩ nhận diện bệnh nhân nguy cơ cao ngay cả khi chưa có triệu chứng rõ ràng, từ đó can thiệp và điều trị kịp thời.

- Cá nhân hóa điều trị: Dự báo nguy cơ theo tỷ lệ phần trăm, hỗ trợ xây dựng phác đồ theo dõi và điều trị phù hợp, nâng cao hiệu quả và giảm chi phí.

- Quản lý bệnh viện hiệu quả: Phân tầng bệnh nhân theo mức độ rủi ro, ưu tiên chăm sóc trường hợp khẩn cấp và sử dụng nguồn lực y tế hợp lý.

- Đóng góp cho nghiên cứu y học: Làm rõ mối quan hệ giữa yếu tố nguy cơ và sự tiến triển bệnh, hỗ trợ cải tiến phác đồ điều trị và nghiên cứu dịch tễ học.

- Tích hợp công nghệ y tế thông minh: Khi kết nối với hồ sơ bệnh án điện tử hoặc ứng dụng chăm sóc sức khỏe, mô hình có thể cảnh báo sớm, giúp bệnh nhân chủ động theo dõi tình trạng tim mạch.

- Ý nghĩa xã hội: Góp phần giảm gánh nặng y tế, nâng cao chất lượng cuộc sống và bảo vệ sức khỏe cộng đồng.

## Tổng kết chương 1

Chương 1 đã cung cấp một cái nhìn toàn diện về bài toán dự báo trong lĩnh vực y tế, nhằm mục đích phát hiện sớm các nguy cơ mắc bệnh để nâng cao khả năng sống sót cũng như chất lượng khám chữa bệnh. Chương này đã trình bày chi tiết về học máy, quá trình xây dựng, phát triển và ứng dụng các mô hình học máy. Bên cạnh đó, chương đã trình bày về bài toán dự báo trong lĩnh vực y tế, hiện trạng bệnh suy tim trên thế giới và giới thiệu về bài toán dự đoán nguy cơ suy tim, mục tiêu chính của bài toán, mô tả dữ liệu đầu vào, đầu ra và ứng dụng của bài toán trong thực tế.

# CHƯƠNG 2: MỘT SỐ KỸ THUẬT GIẢI QUYẾT BÀI TOÁN

## 2.1. Mô hình cây quyết định (Decision Tree)

### *2.1.1. Giới thiệu*

Cây quyết định (Decision Tree) [11] là một mô hình học máy được thiết kế để giải quyết hiệu quả các tác vụ phân loại (classification) và hồi quy (regression). Mô hình này hoạt động dựa trên cấu trúc dạng cây, trong đó các nút đại diện cho điều kiện kiểm tra, các nhánh biểu diễn kết quả của điều kiện, và các nút lá đưa ra dự đoán cuối cùng. Cây quyết định được ứng dụng rộng rãi trong nhiều lĩnh vực như y tế (hỗ trợ chẩn đoán bệnh), tài chính – ngân hàng (dự đoán rủi ro tín dụng, phát hiện gian lận), marketing (phân loại khách hàng, dự báo hành vi mua sắm) và công nghiệp (dự báo bảo trì, kiểm định chất lượng sản phẩm). Điểm nổi bật của cây quyết định là khả năng diễn giải trực quan thông qua các quy tắc "IF – THEN" dễ hiểu, giúp người dùng không chuyên vẫn có thể nắm bắt cách mô hình đưa ra quyết định. Ngoài ra, cây quyết định có thể xử lý cả dữ liệu số lẫn dữ liệu phân loại mà không yêu cầu giả định phức tạp về phân phối dữ liệu. Tuy nhiên, hạn chế của phương pháp này là dễ quá khớp (overfitting) khi cây phát triển quá sâu và nhạy cảm với dữ liệu nhiễu. Sự đơn giản, minh bạch và tính ứng dụng rộng rãi của cây quyết định cho thấy đây là một mô hình nền tảng trong học máy, đồng thời cũng là thành phần cốt lõi của nhiều thuật toán nâng cao khác như Random Forest hay XGBoost.

Cây quyết định (Decision Tree) được ứng dụng rộng rãi trong nhiều lĩnh vực khác nhau, bao gồm:

- Phân loại dữ liệu: Đây là một trong những ứng dụng chính của cây quyết định, cho phép gán nhãn các đối tượng dữ liệu vào những lớp được xác định trước, chẳng hạn như phân loại khách hàng tiềm năng hay phân loại văn bản.

- Dự đoán và hồi quy: Cây quyết định có khả năng dự đoán giá trị số liên tục, điển hình như dự đoán giá nhà đất, doanh thu, hay nhu cầu tiêu thụ sản phẩm dựa trên các yếu tố đầu vào.

- Phân tích và hỗ trợ ra quyết định: Với khả năng biểu diễn dưới dạng các quy tắc "IF – THEN" dễ hiểu, cây quyết định thường được dùng để hỗ trợ các nhà quản lý đưa ra lựa chọn trong kinh doanh, quản trị rủi ro hoặc y tế.

- Phát hiện gian lận: Trong lĩnh vực tài chính – ngân hàng, cây quyết định có thể nhận diện các mẫu hành vi bất thường, hỗ trợ hệ thống phát hiện gian lận trong giao dịch.

- Y tế và chẩn đoán: Cây quyết định được ứng dụng để hỗ trợ bác sĩ chẩn đoán bệnh hoặc dự đoán nguy cơ sức khỏe dựa trên dữ liệu hồ sơ bệnh án và chỉ số y tế.

- Marketing và phân tích khách hàng: Doanh nghiệp sử dụng cây quyết định để phân nhóm khách hàng, dự đoán hành vi mua sắm và thiết kế chiến lược tiếp thị phù hợp.

- Ứng dụng trong công nghiệp: Cây quyết định có thể dự đoán sự cố kỹ thuật, lập kế hoạch bảo trì thiết bị, hoặc phân tích chất lượng sản phẩm trong quy trình sản xuất.

- Các lĩnh vực khác: Cây quyết định còn được ứng dụng trong giáo dục (dự đoán kết quả học tập), nông nghiệp (dự đoán năng suất mùa vụ), và cả trong khoa học môi trường (dự báo ô nhiễm hoặc biến đổi khí hậu).

### *2.1.2. Kiến trúc cơ bản của mô hình cây quyết định*

Cây quyết định (Decision Tree) [11] có kiến trúc dựa trên cấu trúc phân cấp dạng cây, trong đó mỗi thành phần giữ một vai trò riêng biệt trong việc đưa ra quyết định. Các thành phần chính gồm:

- *Nút gốc (Root Node):*

+ Là điểm khởi đầu của cây, đại diện cho toàn bộ tập dữ liệu ban đầu.

+ Tại nút gốc, thuật toán sẽ lựa chọn đặc trưng tối ưu nhất để thực hiện lần phân chia đầu tiên.

- *Nút trong (Internal Nodes):*

+ Là các nút con nằm giữa gốc và lá.

+ Mỗi nút trong chứa một điều kiện kiểm tra (ví dụ: tuổi > 40, thu nhập > 10 triệu), được lựa chọn dựa trên các tiêu chí như Information Gain, Gini Index hoặc Variance Reduction.

Dữ liệu sẽ được phân chia theo kết quả của điều kiện này để đi theo các nhánh khác nhau.

- *Cành (Branches):*

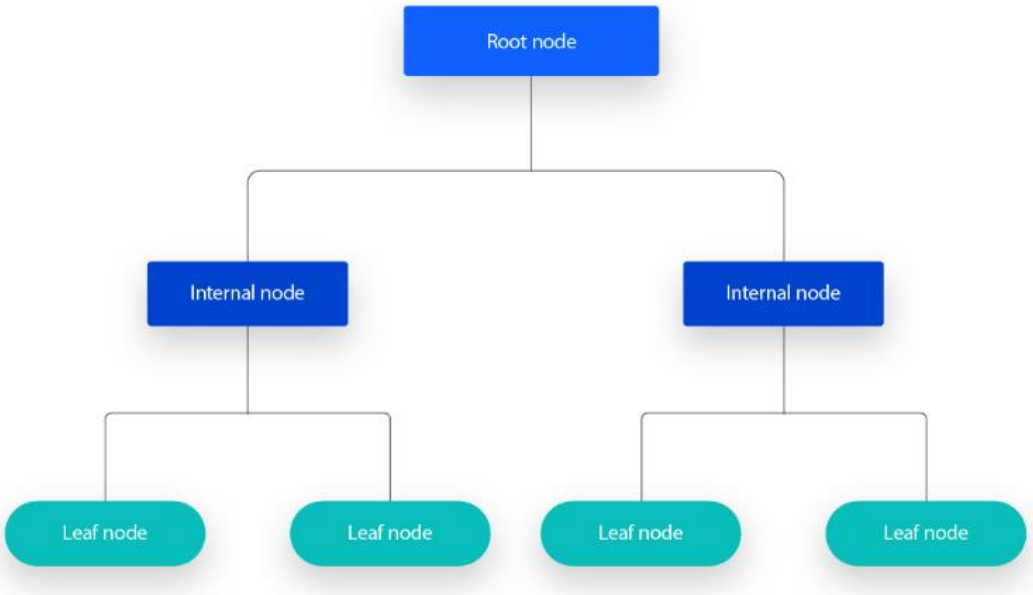
+ Biểu diễn kết quả của một điều kiện tại nút trong.

+ Mỗi cành dẫn đến một nút con khác, cho đến khi dữ liệu đến được nút lá.

- Nút lá (Leaf Nodes/Terminal Nodes):

+ Là điểm kết thúc của một đường đi trong cây.

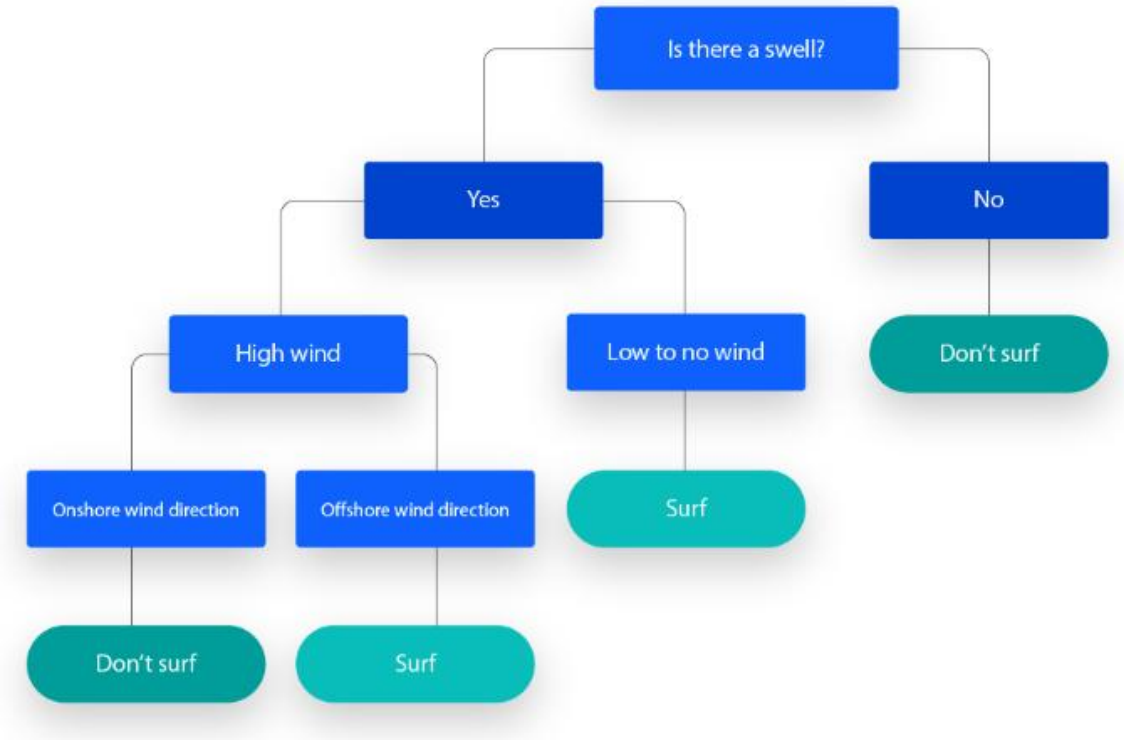
+ Nút lá chứa kết quả dự đoán cuối cùng: có thể là một nhãn lớp (trong phân loại) hoặc một giá trị số (trong hồi quy).




*Hình 2.1. Kiến trúc cơ bản của cây quyết định (Nguồn: IBM - What is a Decision Tree)*

Thuật toán cây quyết định chia nhỏ dữ liệu đầu vào thành các tập con nhỏ hơn dựa trên giá trị của thuộc tính, và tại mỗi bước chia, nó chọn thuộc tính tốt nhất để tối ưu hóa việc phân loại. Với tập dữ liệu gồm các thuộc tính và lớp (classes), cây quyết định tạo ra các quy tắc để dự đoán lớp của dữ liệu chưa biết.

Ví dụ minh họa: Hãy tưởng tượng rằng bạn đang cố gắng đánh giá xem mình có nên đi lướt sóng hay không, bạn có thể sử dụng các quy tắc quyết định sau để đưa ra lựa chọn:




*Hình 2.2. Ví dụ minh họa về cây quyết định (Nguồn: IBM - What is a Decision Tree)*

Cấu trúc sơ đồ luồng này tạo ra cách trình bày dễ hiểu về quá trình ra quyết định, cho phép các nhóm trong tổ chức hiểu rõ hơn lý do tại sao quyết định được đưa ra.

Xem Hình 2.2, ta có thể thấy nút gốc là "Có sóng lớn không". Nếu không có sóng, kết quả ngay lập tức là "Không đi lướt sóng". Nếu có sóng, tiếp tục xem xét các yếu tố: "Khi gió mạnh", hướng gió sẽ quyết định - "Gió

vào bờ" dẫn đến kết quả "Không đi lướt sóng", "Gió ra biển" dẫn đến kết quả "Đi lướt sóng". Trường hợp "Ít hoặc không có gió", kết quả cũng là "Đi lướt sóng".

Qua ví dụ minh họa, có thể thấy được cây quyết định (Decision Tree) mô phỏng quá trình phân nhánh tuần tự để đưa ra quyết định cuối cùng dựa trên các yếu tố điều kiện.

Mức độ đồng nhất của các điểm dữ liệu trong cây quyết định phụ thuộc nhiều vào độ phức tạp của cây. Với những cây nhỏ, các nút lá thường dễ đạt được trạng thái thuần túy, tức là hầu hết dữ liệu trong nút đó thuộc về một lớp duy nhất. Tuy nhiên, khi cây phát triển quá lớn, việc duy trì độ thuần túy trở nên khó khăn hơn, do dữ liệu bị chia nhỏ thành nhiều nhánh với số lượng mẫu rất ít. Hiện tượng này được gọi là phân mảnh dữ liệu và thường dẫn đến tình trạng quá khớp (overfitting).

### *2.1.3. Thuật toán xây dựng mô hình cây quyết định*

#### *2.1.3.1. Thuật toán ID3*

Thuật toán ID3 được Ross Quinlan đề xuất vào năm 1986 và được coi là một trong những thuật toán nền tảng trong xây dựng cây quyết định. Nguyên lý chính của ID3 là lựa chọn thuộc tính phân chia dựa trên Information Gain (lượng thông tin thu được), vốn được tính toán từ chỉ số Entropy – thước đo mức độ hỗn loạn hoặc độ không chắc chắn trong dữ liệu. Một thuộc tính có Information Gain càng cao thì càng được ưu tiên lựa chọn để chia tách.

- Công thức tính Entropy:

$$H(D) = -\sum_{i=1}^{n} P_i \log_2(P_i)$$

Trong đó:

+ H(D)**:** Entropy của tập dữ liệu D.

+ Pi**:** Xác suất của lớp i trong D.

Giá trị entropy có thể nằm trong khoảng từ 0 đến 1. Nếu tất cả các mẫu trong tập dữ liệu S thuộc cùng một lớp, thì entropy sẽ bằng 0. Nếu một nửa số mẫu được phân loại vào một lớp và nửa còn lại thuộc một lớp khác, entropy sẽ đạt giá trị cao nhất là 1. Để chọn đặc trưng tốt nhất để phân tách và tìm cây quyết định tối ưu, nên sử dụng thuộc tính có lượng entropy nhỏ nhất.

- Công thức Information Gain:

$$Gain(S, A) = H(S) - H(S \mid A)$$

Trong đó:

+ H(S)**:** Entropy trước phân chia.

+ H(S|A)**:** Entropy sau phân chia theo A.

Độ lợi thông tin biểu thị sự khác biệt về entropy trước và sau khi phân tách trên một thuộc tính nhất định. Thuộc tính có độ lợi thông tin cao nhất sẽ tạo ra phép phân tách tốt nhất vì nó thực hiện tốt nhất việc phân loại dữ liệu huấn luyện theo phân loại mục tiêu.

Mặc dù đơn giản và dễ triển khai, ID3 có một số hạn chế như: thiên lệch đối với các thuộc tính có nhiều giá trị phân loại, khó xử lý dữ liệu liên tục, và dễ dẫn đến hiện tượng quá khớp (overfitting) khi cây quá phức tạp.

*2.1.3.2. Thuật toán C4.5*

Được phát triển bởi Ross Quinlan năm 1993 như một sự mở rộng và cải tiến của ID3, C4.5 khắc phục các nhược điểm bằng cách sử dụng Gain Ratio thay vì Information Gain, nhằm giảm sự thiên lệch đối với các thuộc tính có nhiều giá trị. Ngoài ra, C4.5 còn cung cấp cơ chế xử lý thuộc tính liên tục bằng cách tìm ngưỡng tối ưu để chia tách dữ liệu, đồng thời hỗ trợ xử lý dữ liệu thiếu (missing values).

- Công thức Gain Ratio:

Gain Ratio(A) = Information Gain (A) / Split Information (A)

Trong đó:

+ Information Gain(A): lượng thông tin đạt được khi phân chia theo thuộc tính A.

+ Split Information(A): thước đo mức độ "phân mảnh" của dữ liệu khi chia theo thuộc tính AAA. Nó được tính bằng công thức entropy dựa trên tỷ lệ mẫu rơi vào các nhánh sau phân chia.

Đặc biệt, C4.5 đưa vào cơ chế cắt tỉa cây (pruning) để ngăn chặn việc mô hình hóa quá mức dữ liệu huấn luyện. Nhờ những cải tiến này, C4.5 đã trở thành một trong những thuật toán cây quyết định phổ biến nhất, được sử dụng rộng rãi trong nhiều lĩnh vực nghiên cứu và ứng dụng thực tế.

*2.1.3.3. Thuật toán CHAID*

CHAID được phát triển với nguyên lý lựa chọn thuộc tính dựa trên kiểm định Chi-square, nhằm tìm ra thuộc tính có mối liên hệ thống kê mạnh nhất với biến mục tiêu. Không giống như CART – vốn chỉ cho phép phân nhánh nhị phân – CHAID cho phép tạo ra nhiều nhánh cùng lúc từ một nút, phù hợp với dữ liệu có đặc tính phân loại định tính (categorical).

Do sử dụng phương pháp kiểm định thống kê, CHAID thường được ứng dụng nhiều trong các lĩnh vực xã hội học, kinh tế học, nghiên cứu thị trường và marketing, nơi dữ liệu thường có bản chất phân loại và mối quan hệ thống kê được ưu tiên phân tích.

*2.3.1.4. Thuật toán CART*

CART, do Breiman và cộng sự giới thiệu vào năm 1986, là một trong những thuật toán tiêu biểu và được ứng dụng rộng rãi nhất. Khác với ID3 và C4.5, CART có khả năng giải quyết cả bài toán phân loại và bài toán hồi quy.

Trong phân loại, CART sử dụng chỉ số Gini Index để đánh giá độ thuần khiết của nút, trong khi đó đối với hồi quy, thuật toán dựa trên tiêu chí Mean Squared Error (MSE) để chọn cách phân chia.

Một điểm đặc trưng quan trọng của CART là mọi phép chia tách đều là nhị phân, nghĩa là mỗi nút luôn được tách thành đúng hai nhánh con. Chính

đặc điểm này giúp mô hình dễ triển khai, đồng thời tạo tiền đề cho các phương pháp học máy tiên tiến hơn như Random Forest và Gradient Boosting Trees.

*2.3.1.4. Các biến thể mở rộng khác*

Ngoài bốn thuật toán chính nêu trên, còn có những phương pháp khác như MARS (Multivariate Adaptive Regression Splines) – vốn không phải cây quyết định thuần túy, nhưng dựa trên ý tưởng phân chia miền dữ liệu thành các đoạn spline để xử lý các mối quan hệ phi tuyến. Bên cạnh đó, nhiều thuật toán hiện đại hơn như Random Forest, XGBoost hay LightGBM đều được xây dựng dựa trên nền tảng CART, nhưng cải tiến thêm bằng các kỹ thuật bagging, boosting hoặc pruning hiệu quả hơn.

### *2.1.4. Ưu điểm của mô hình cây quyết định*

- Dễ hiểu và trực quan: Cây quyết định xây dựng các quy tắc đơn giản, dễ hiểu đối với người dùng.

- Linh hoạt: Có thể xử lý cả dữ liệu liên tục và dữ liệu danh mục.

- Khả năng tổng quát hóa tốt: Nắm bắt được mối quan hệ phi tuyến tính giữa các biến.

- Hiệu quả: Làm việc tốt với tập dữ liệu lớn, đặc biệt khi kết hợp với phương pháp cắt tỉa cây (pruning).

### *2.1.5. Nhược điểm của mô hình cây quyết định*

- Overfitting: Cây quyết định có xu hướng quá phù hợp với dữ liệu huấn luyện, dẫn đến khả năng dự đoán kém trên dữ liệu mới.

- Nhạy cảm với dữ liệu đầu vào: Thay đổi nhỏ trong dữ liệu có thể làm thay đổi đáng kể cấu trúc cây.

- Không ổn định: Cần kết hợp với các phương pháp khác như Random Forest để cải thiện độ ổn định.

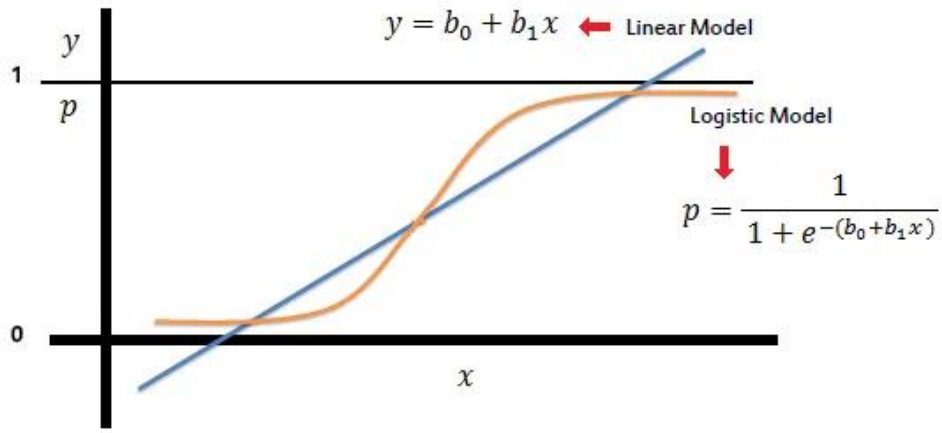
- Tính toán tốn kém: Với dữ liệu lớn hoặc nhiều thuộc tính, việc xây dựng cây có thể mất nhiều thời gian.

Thuật toán cây quyết định là một trong những phương pháp nền tảng và phổ biến nhất trong lĩnh vực học máy và khai phá dữ liệu. Với cấu trúc phân cấp trực quan, cây quyết định không chỉ cung cấp khả năng dự đoán hiệu quả mà còn mang lại tính dễ hiểu và khả năng diễn giải kết quả, điều mà nhiều mô hình học máy phức tạp khác khó đạt được. Tuy nhiên, bên cạnh ưu điểm, cây quyết định cũng tồn tại hạn chế như dễ bị quá khớp (overfitting), nhạy cảm với biến động dữ liệu, và có thể tạo ra các cây quá phức tạp nếu không được cắt tỉa hợp lý.

## 2.2. Mô hình hồi quy Logistic

### *2.2.1. Giới thiệu*

Hồi quy logistic là một thuật toán học máy được sử dụng rộng rãi để giải quyết các bài toán phân loại nhị phân. Mô hình này dự đoán xác suất một sự kiện xảy ra, thường được biểu diễn dưới dạng một nhãn nhị phân (ví dụ: 0 hoặc 1, có hoặc không). Không giống như hồi quy tuyến tính dự đoán giá trị liên tục, hồi quy logistic đưa ra dự đoán về xác suất thuộc về một lớp nhất định. [12]




*Hình 2.3. Thuật toán hồi quy Logistic (Nguồn: Medium - Logistic Regression)*

Mô hình hồi quy Logistic được đánh giá cao bởi tính đơn giản, dễ cài đặt, khả năng tính toán nhanh và kết quả dễ diễn giải. Do đó, nó thường được

xem như một mô hình nền tảng trong học máy và phân tích dữ liệu. Trong thực tế, hồi quy Logistic được ứng dụng trong nhiều lĩnh vực gồm:

- Y tế: Dự đoán khả năng mắc bệnh dựa trên dữ liệu nhân khẩu học và chỉ số y tế của bệnh nhân, ví dụ như dự đoán nguy cơ mắc bệnh tim, tiểu đường hay ung thư.

- Tài chính – ngân hàng: Đánh giá rủi ro tín dụng, xác định khả năng khách hàng không trả nợ, hoặc phát hiện giao dịch gian lận.

- Tiếp thị và kinh doanh: Phân tích hành vi khách hàng, dự đoán khả năng mua hàng hoặc rời bỏ dịch vụ (churn prediction), từ đó hỗ trợ xây dựng chiến lược chăm sóc khách hàng.

- Xử lý văn bản và NLP: Ứng dụng trong phân loại văn bản, phân tích cảm xúc, hoặc phát hiện thư rác (spam detection).

- Khoa học xã hội: Nghiên cứu mối quan hệ giữa các biến nhân khẩu học và khả năng xảy ra một sự kiện, ví dụ như tham gia bầu cử hay lựa chọn nghề nghiệp.

### *2.2.2. Đặc điểm của mô hình hồi quy Logistic*

#### *2.2.2.1. Các loại hồi quy Logistic*

Hồi quy logistic có thể được phân loại thành ba loại chính dựa trên bản chất của biến phụ thuộc:

- Hồi quy logistic nhị thức : Loại này được sử dụng khi biến phụ thuộc chỉ có hai loại. Ví dụ bao gồm Có/Không, Đạt/Không đạt hoặc 0/1. Đây là dạng hồi quy logistic phổ biến nhất và được sử dụng cho các bài toán phân loại nhị phân.

- Hồi quy logistic đa thức : Phương pháp này được sử dụng khi biến phụ thuộc có ba hoặc nhiều nhóm có thể có mà không được sắp xếp theo thứ tự. Ví dụ: phân loại động vật thành các nhóm như "mèo", "chó" hoặc "cừu". Phương pháp này mở rộng hồi quy logistic nhị phân để xử lý nhiều nhóm.

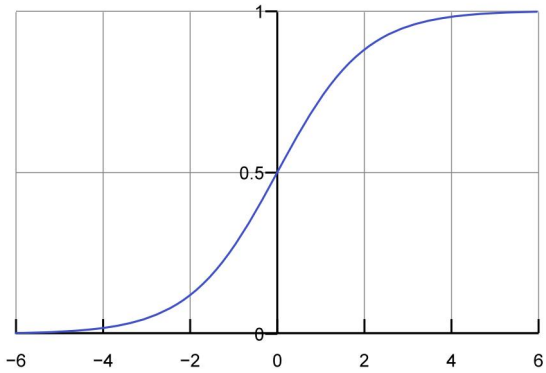
- Hồi quy Logistic Thứ tự : Loại này áp dụng khi biến phụ thuộc có ba hoặc nhiều hạng mục với thứ tự hoặc thứ hạng tự nhiên. Ví dụ bao gồm các xếp hạng như "thấp", "trung bình" và "cao". Nó tính đến thứ tự của các hạng mục khi lập mô hình.

*2.2.2.2. Giả định của hồi quy Logistic*

Việc hiểu được các giả định đằng sau hồi quy logistic là rất quan trọng để đảm bảo mô hình được áp dụng một cách chính xác nhất, các giả định chính bao gồm:

- Quan sát độc lập : Mỗi điểm dữ liệu được coi là độc lập với các điểm khác, nghĩa là không có mối tương quan hoặc phụ thuộc giữa các mẫu đầu vào.
- Biến phụ thuộc nhị phân : Giả định rằng biến phụ thuộc phải là nhị phân, nghĩa là nó chỉ có thể nhận hai giá trị. Đối với nhiều hơn hai loại, hàm SoftMax được sử dụng.
- Mối quan hệ tuyến tính giữa các biến độc lập và tỷ lệ cược logarit : Mô hình giả định mối quan hệ tuyến tính giữa các biến độc lập và tỷ lệ cược logarit của biến phụ thuộc, nghĩa là các yếu tố dự báo ảnh hưởng đến tỷ lệ cược logarit theo cách tuyến tính.
- Không có giá trị ngoại lệ : Tập dữ liệu không được chứa các giá trị ngoại lệ cực đoan vì chúng có thể làm sai lệch ước tính hệ số hồi quy logistic.
- Kích thước mẫu lớn : Cần có kích thước mẫu đủ lớn để đưa ra kết quả đáng tin cậy và ổn định.

*2.2.2.3. Hàm Sigmoid*




*Hình 2.4. Hàm Sigmoid*

- Hàm sigmoid là một phần quan trọng của hồi quy logistic được sử dụng để chuyển đổi đầu ra thô của mô hình thành giá trị xác suất từ 0 đến 1.

- Hàm này lấy bất kỳ số thực nào và ánh xạ nó vào phạm vi từ 0 đến 1, tạo thành một đường cong hình chữ "S" được gọi là đường cong sigmoid hoặc đường cong logistic. Vì xác suất phải nằm giữa 0 và 1, hàm sigmoid hoàn toàn phù hợp cho mục đích này.

- Trong hồi quy logistic, chúng ta sử dụng giá trị ngưỡng thường là 0,5 để quyết định nhãn lớp.

+ Nếu đầu ra sigmoid giống hoặc cao hơn ngưỡng, đầu vào được phân loại là Lớp 1.

+ Nếu thấp hơn ngưỡng, đầu vào được phân loại là Lớp 0.

*2.2.2.4. Ngưỡng quyết định và tối ưu hóa mô hình*

Để đưa ra dự đoán cuối cùng, ta thường đặt một ngưỡng (thường là 0.5). Nếu xác suất dự đoán lớn hơn ngưỡng, ta kết luận dữ liệu thuộc về lớp dương, ngược lại thuộc về lớp âm được gọi là ngưỡng quyết định.

Để tối ưu hóa, mô hình tìm kiếm các tham số của hàm sigmoid bằng cách tối ưu hóa một hàm mất mát, thường là hàm cross-entropy.

### *2.2.3. Nguyên lý hoạt động của mô hình hồi quy Logistic*

Mô hình hồi quy logistic chuyển đổi giá trị đầu ra liên tục của hàm hồi quy tuyến tính thành giá trị đầu ra phân loại bằng cách sử dụng hàm sigmoid ánh xạ bất kỳ tập hợp các biến độc lập có giá trị thực nào thành một giá trị từ 0 đến 1. Hàm này được gọi là hàm logistic.

Đầu tiên, các đặc trưng đầu vào $[x_1, x_2, \ldots, x_n]$ được kết hợp tuyến tính với các trọng số $[w_1, w_2, \ldots, w_n]$ và hệ số chặn b, tạo thành giá trị:

$$z = w_1\, x_1, w_2\, x_2, \ldots, w_n\, x_n \; + \; b$$

Sau đó, thay vì sử dụng trực tiếp giá trị z, mô hình sẽ áp dụng hàm sigmoid:

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

Trong đó:

+ $X = (x_1, x_2, \ldots, x_n)$: vector biến độc lập (các thuộc tính).

+ $W = ([w_1, w_2, \ldots, w_n)$: là vector hệ số (trọng số).

+ b : hệ số chặn.

+ σ (z) : hàm sigmoid đảm bảo đầu ra trong khoảng (0, 1).

Kết quả của hàm sigmoid là một giá trị trong khoảng (0, 1), có thể được diễn giải như xác suất xảy ra sự kiện. Dựa trên ngưỡng quyết định mà ta đã định ra từ ban đầu.

Để huấn luyện mô hình, hàm mất mát phổ biến nhất được sử dụng là hàm Cross-Entropy, nhằm đo lường sự khác biệt giữa phân phối xác suất dự đoán và phân phối nhãn thực tế. Với dữ liệu có N mẫu, công thức được viết như sau:

$$L = -\frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{K} y_{n,i} \log \hat{y}_{n,i}$$

Trong đó:

+ $y_{n,i} \in \{0, 1\}$ là nhãn thực tế của mẫu $n_i$.

+ $\hat{y}_{n,i}$ là xác suất dự đoán của mô hình cho lớp I.

***2.2.4. Ưu điểm của mô hình hồi quy Logistic***

- Dễ hiểu: Mô hình tương đối dễ hiểu và giải thích.

- Hiệu quả: Có thể xử lý lượng lớn dữ liệu một cách hiệu quả.

- Ít nhạy cảm với dữ liệu ngoại lai: Mô hình tương đối ổn định với các giá trị ngoại lai.

- Đa dạng ứng dụng: Được sử dụng rộng rãi trong nhiều lĩnh vực như y tế, tài chính, marketing.

### *2.2.5. Nhược điểm của mô hình hồi quy Logistic*

- Giả định tuyến tính: Mô hình giả định mối quan hệ tuyến tính giữa các biến độc lập và biến phụ thuộc.

- Khó xử lý dữ liệu phi tuyến: Với dữ liệu có mối quan hệ phi tuyến phức tạp, hồi quy logistic có thể không đạt hiệu quả cao.

- Khó xử lý dữ liệu nhiều lớp: Để xử lý các bài toán phân loại đa lớp, cần sử dụng các kỹ thuật mở rộng.

Hồi quy Logistic là một trong những mô hình học máy cơ bản nhưng vô cùng quan trọng, đặc biệt trong các bài toán phân loại nhị phân và đa lớp. Với cơ sở toán học vững chắc dựa trên hàm sigmoid và softmax, mô hình này có khả năng ánh xạ xác suất cho từng lớp, giúp đưa ra dự đoán vừa trực quan vừa dễ diễn giải. Ưu điểm nổi bật của hồi quy logistic là tính đơn giản, tốc độ huấn luyện nhanh và khả năng giải thích rõ ràng thông qua hệ số hồi quy, tuy nhiên mô hình cũng có hạn chế khi xử lý dữ liệu phi tuyến hoặc có quan hệ phức tạp. Nhìn chung, hồi quy logistic vẫn là một công cụ hiệu quả, đóng vai trò nền tảng trong nhiều ứng dụng thực tế và thường được sử dụng như bước khởi đầu trong các quy trình phân tích dữ liệu và xây dựng mô hình học máy.

## 2.3. Mô hình XGBoost

### *2.3.1. Giới thiệu*

XGBoost (Extreme Gradient Boosting) là một thuật toán học máy mạnh mẽ, thuộc nhóm ensemble learning, được phát triển dựa trên kỹ thuật Gradient Boosting Decision Trees (GBDT). Thuật toán này được Tianqi Chen giới thiệu vào năm 2016 và nhanh chóng trở thành một trong những công cụ phổ biến nhất trong các cuộc thi về khoa học dữ liệu nhờ khả năng xử lý dữ liệu hiệu quả, tốc độ huấn luyện nhanh và độ chính xác cao [1].

Cốt lõi của XGBoost là việc kết hợp nhiều cây quyết định (decision trees) yếu lại với nhau theo phương pháp boosting nhằm tạo thành một mô hình mạnh. Quá trình huấn luyện diễn ra tuần tự, trong đó mỗi cây mới được

xây dựng để khắc phục các sai số còn tồn tại từ những cây trước đó. Khác với các thuật toán boosting truyền thống, XGBoost tối ưu hàm mất mát bằng phương pháp gradient descent và bổ sung thêm hệ số regularization (L1, L2) nhằm hạn chế overfitting, giúp mô hình tổng quát hóa tốt hơn trên dữ liệu mới.

Nhờ những ưu điểm vượt trội, XGBoost đã trở thành một trong những thuật toán phổ biến nhất trong các cuộc thi khoa học dữ liệu cũng như trong các ứng dụng thực tế bao gồm:

- Tài chính – Ngân hàng: XGBoost được sử dụng để dự báo rủi ro tín dụng, phát hiện gian lận giao dịch, phân tích danh mục đầu tư và đánh giá khả năng vỡ nợ của khách hàng.

- Y tế: Trong phân tích dữ liệu y tế, XGBoost hỗ trợ dự đoán nguy cơ mắc bệnh (như tim mạch, ung thư), phân loại hình ảnh y tế và phát hiện bất thường trong hồ sơ bệnh án điện tử.

- Thương mại điện tử và marketing: Thuật toán này giúp xây dựng hệ thống gợi ý sản phẩm, phân tích hành vi mua sắm, dự báo nhu cầu và tối ưu hóa chiến dịch tiếp thị.

- Khoa học dữ liệu và dự báo: XGBoost được áp dụng trong dự báo giá cổ phiếu, phân tích dữ liệu cảm biến IoT, dự đoán xu hướng thị trường hoặc dự báo thời tiết.

- Xử lý ngôn ngữ tự nhiên (NLP): XGBoost có thể được sử dụng cho các tác vụ như phân loại văn bản, phân tích cảm xúc, và phát hiện spam email.

- Cuộc thi và nghiên cứu khoa học dữ liệu: XGBoost là một trong những thuật toán được sử dụng nhiều nhất trên Kaggle và các nền tảng thi đấu dữ liệu nhờ độ chính xác cao và khả năng tùy chỉnh linh hoạt.

### *2.3.2. Kiến trúc cơ bản của mô hình XGBoost*

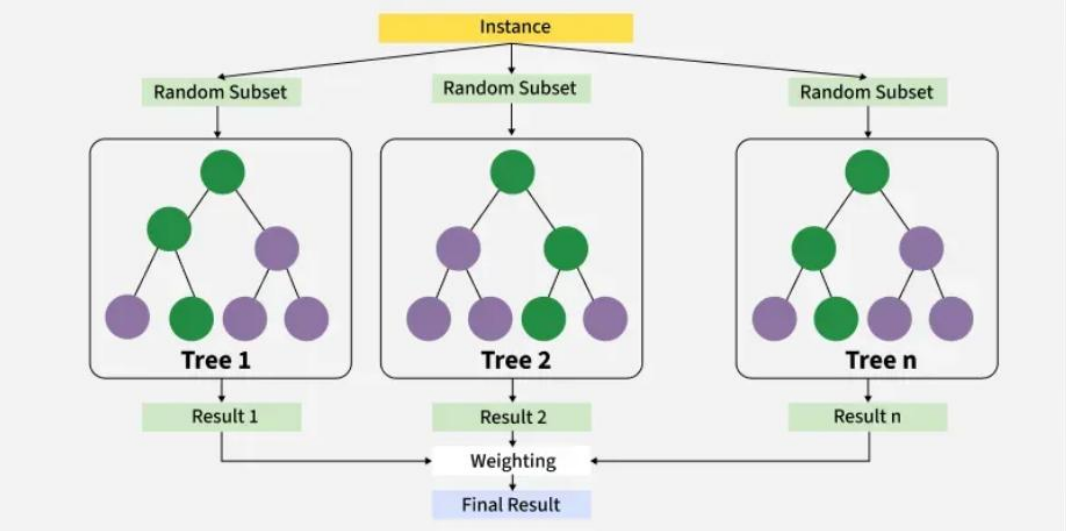
Mô hình XGBoost (Extreme Gradient Boosting) được xây dựng dựa trên ý tưởng của thuật toán Boosting, trong đó nhiều mô hình cây quyết định yếu (weak learners) được huấn luyện tuần tự và kết hợp lại thành một mô

hình mạnh (strong learner). Kiến trúc cơ bản của XGBoost có thể được khái quát qua các thành phần chính sau:

*2.3.2.1. Tập hợp các cây quyết định (Base Learners)*

- XGBoost sử dụng CART (Classification and Regression Trees) làm mô hình học yếu (weak learner).

- Mỗi cây được xây dựng một cách nối tiếp, trong đó cây mới sẽ học từ phần dư (residual errors) còn lại sau khi cộng gộp các cây trước đó.




*Hình 2.5. Sơ đồ mô hình học yếu (weak learner) (Nguồn: Geeksforgeeks - XGBoost)*

- Công thức dự đoán của mô hình XGBoost:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i)$$

Trong đó:

+ ŷ : là giá trị dự đoán cuối cùng cho điểm dữ liệu thứ I.

+ K : là tổng số cây trong mô hình học yếu.

+ $f_k$ ($x_i$): là dự đoán của cây thứ k cho dữ liệu $x_i$.

- Thay vì đưa ra dự đoán cuối cùng trực tiếp, mỗi cây trong XGBoost đóng góp một phần vào dự đoán, và kết quả cuối cùng là tổng (hoặc trung bình có trọng số) của tất cả các cây.

*2.3.2.2. Hàm mục tiêu*

Hàm mục tiêu của mô hình XGBoost bao gồm hai thành phần:

- Hàm mất mát (Loss Function): phản ánh độ sai lệch giữa giá trị dự đoán và giá trị thực tế.

+ Với hồi quy: sử dụng Mean Squared Error (MSE).

+ Với phân loại nhị phân: sử dụng Logistic Loss (Log Loss).

+ Với phân loại đa lớp: sử dụng Softmax Loss.

- Hàm chính quy (Regularization Term): điểm khác biệt quan trọng của XGBoost.

XGBoost thêm cả L1 (Lasso) và L2 (Ridge) để phạt độ phức tạp của mô hình, giúp giảm hiện tượng quá khớp (overfitting).

Biểu thức tổng quát của hàm mục tiêu

$$\mathcal{L}(\phi) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

Trong đó:

+ $l(y_i, \hat{y}_i)$ : hàm mất mát (loss function) đo sai lệch giữa giá trị thực và giá trị dự đoán ŷ.

+ Ω ($f_k$) : thành phần điều chuẩn (regularization term) giúp kiểm soát độ phức tạp của mô hình.

+ K: số lượng cây (base learners)

Công thức tính thành phần điều chuẩn (regularization term):

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

Trong đó:

+ T : là số lượng lá (leaf nodes) trong cây quyết định ($f_k$).

+ $w_j$ : trọng số gán cho lá thứ j.

+ $\gamma$ : tham số phạt độ phức tạp, kiểm soát số lượng lá (pruning).

+ $\lambda$ : hệ số điều chuẩn L2, giúp giảm overfitting bằng cách phạt các trọng số lớn trong cây.

*2.3.2.3. Cơ chế tối ưu hóa dựa trên Gradient Boosting*

- XGBoost sử dụng gradient descent để tối ưu hàm mất mát.

- Tại mỗi bước lặp, mô hình thêm một cây mới dựa trên đạo hàm bậc 1 (gradient) và bậc 2 (Hessian) của hàm mất mát, thay vì chỉ sử dụng gradient như các thuật toán boosting thông thường.

- Việc khai thác thông tin bậc hai giúp XGBoost đạt được tốc độ hội tụ nhanh hơn và dự đoán chính xác hơn.

*2.3.2.4. Cơ chế cập nhật và trọng số dữ liệu*

- Ban đầu, mọi điểm dữ liệu được gán trọng số như nhau.

- Sau mỗi vòng lặp, mô hình tính toán lỗi dự đoán và điều chỉnh trọng số, sao cho các điểm dữ liệu khó phân loại hơn được gán trọng số cao hơn.

- Nhờ vậy, XGBoost tập trung học nhiều hơn vào các mẫu khó, tăng cường khả năng phân loại.

*2.3.2.5. Cắt tỉa cây (Tree pruning) và kiểu soát độ sâu*

- XGBoost xây dựng cây theo hướng depth-first (ưu tiên mở rộng chiều sâu) và sau đó áp dụng cắt tỉa ngược (pruning) để loại bỏ các nhánh không mang lại lợi ích.

- Thay vì dừng mở rộng cây sớm như CART, XGBoost phát triển cây đầy đủ rồi mới tiến hành cắt tỉa dựa trên gain và gamma (ngưỡng phạt cho một nút mới).

- Điều này giúp cây được tối ưu tốt hơn và tránh overfitting.

*2.3.2.6. Các tối ưu hóa hiệu năng*

- Parallelization: cho phép xây dựng các nhánh cây song song.

- Handling Sparse Data: sử dụng thuật toán sparse-aware để xử lý dữ liệu khuyết thiếu hoặc ma trận thưa (sparse matrix).

- Cache Optimization: tận dụng hiệu quả bộ nhớ để tăng tốc độ huấn luyện.

- Weighted Quantile Sketch: thuật toán mới để tính toán phân chia tối ưu trong trường hợp dữ liệu có trọng số.

### *2.3.3. Ưu điểm của mô hình XGBoost*

- Có khả năng mở rộng và hiệu quả cho các tập dữ liệu lớn với hàng triệu bản ghi

- Hỗ trợ xử lý song song và tăng tốc GPU để đào tạo nhanh hơn

- Cung cấp các thông số tùy chỉnh và điều chỉnh để tinh chỉnh

- Bao gồm phân tích tầm quan trọng của tính năng để có cái nhìn sâu sắc và lựa chọn tốt hơn

- Được các nhà khoa học dữ liệu trên nhiều ngôn ngữ lập trình tin cậy

### *2.3.4. Nhược điểm của mô hình XGBoost*

- XGBoost có thể tốn nhiều tài nguyên tính toán, khiến nó không lý tưởng cho các hệ thống có tài nguyên hạn chế.

- Nó có thể nhạy cảm với dữ liệu nhiễu hoặc giá trị ngoại lệ, đòi hỏi phải xử lý dữ liệu cẩn thận.

- Dễ bị quá khớp, đặc biệt là trên các tập dữ liệu nhỏ hoặc có quá nhiều cây.

- Cung cấp tính năng quan trọng, nhưng khả năng diễn giải mô hình tổng thể bị hạn chế so với các phương pháp đơn giản hơn, đây là vấn đề trong các lĩnh vực như chăm sóc sức khỏe hoặc tài chính.

Mô hình XGBoost (Extreme Gradient Boosting) là một trong những thuật toán học máy mạnh mẽ và phổ biến nhất hiện nay, nổi bật nhờ khả năng kết hợp nhiều cây quyết định yếu thành một mô hình mạnh thông qua cơ chế boosting. XGBoost không chỉ đạt độ chính xác cao mà còn tối ưu về tốc độ

huấn luyện, khả năng xử lý dữ liệu lớn và ngăn ngừa quá khớp nhờ các cơ chế chính quy hóa. Bên cạnh đó, thuật toán còn hỗ trợ nhiều loại dữ liệu và có tính linh hoạt cao trong việc tùy chỉnh hàm mất mát, giúp nó thích hợp cho cả hồi quy, phân loại, xếp hạng và nhiều bài toán dự báo phức tạp khác. Với những ưu điểm này, XGBoost đã trở thành một công cụ không thể thiếu trong phân tích dữ liệu và các ứng dụng trí tuệ nhân tạo hiện đại.

## 2.4. Lựa chọn phương pháp giải quyết bài toán

Trong số các mô hình học máy truyền thống, Logistic Regression thường được sử dụng cho các bài toán phân loại nhị phân nhờ tính đơn giản, dễ diễn giải và tốc độ huấn luyện nhanh. Tuy nhiên, mô hình này giả định quan hệ tuyến tính giữa biến độc lập và xác suất kết quả, do đó khó nắm bắt được các mối quan hệ phi tuyến và tương tác phức tạp giữa các đặc trưng trong dữ liệu.

Trong khi đó, Decision Tree lại có khả năng mô hình hóa các quan hệ phi tuyến và trực quan nhờ cấu trúc phân nhánh. Tuy nhiên, hạn chế lớn của Decision Tree là dễ bị quá khớp (overfitting) khi cây trở nên quá sâu, đồng thời tính ổn định kém vì chỉ cần một thay đổi nhỏ trong dữ liệu huấn luyện cũng có thể dẫn đến một cây hoàn toàn khác.

XGBoost khắc phục được những nhược điểm trên nhờ việc kết hợp nhiều cây quyết định yếu (weak learners) theo phương pháp boosting, từ đó cải thiện đáng kể khả năng tổng quát hóa và giảm thiểu overfitting nhờ cơ chế regularization. Đồng thời, XGBoost có tốc độ huấn luyện nhanh nhờ tối ưu hóa tính toán song song, hỗ trợ dữ liệu thiếu và cung cấp nhiều tham số để tinh chỉnh. Chính vì vậy, so với Logistic Regression và Decision Tree, XGBoost là lựa chọn tối ưu hơn cho bài toán nghiên cứu do vừa đảm bảo độ chính xác cao, vừa duy trì khả năng mở rộng và linh hoạt.

## Tổng kết chương 2

Trong Chương 2, luận văn đã tập trung trình bày cơ sở lý thuyết và các thuật toán nền tảng được lựa chọn để giải quyết bài toán nghiên cứu. Trước hết, Logistic Regression được giới thiệu như một phương pháp kinh điển trong phân loại nhị phân, với ưu điểm đơn giản, dễ diễn giải và phù hợp khi dữ liệu có mối quan hệ tuyến tính với biến mục tiêu. Tuy nhiên, mô hình này bộc lộ hạn chế trong việc xử lý các quan hệ phi tuyến hoặc dữ liệu có nhiều tương tác phức tạp.

Tiếp theo, Decision Tree được phân tích như một phương pháp trực quan, có khả năng mô tả các mối quan hệ phi tuyến và dễ dàng diễn giải nhờ cấu trúc phân nhánh. Song song với ưu điểm đó, Decision Tree dễ gặp hiện tượng quá khớp (overfitting), dẫn đến khả năng khái quát hóa kém, đồng thời tính ổn định thấp khi dữ liệu thay đổi.

Cuối cùng, XGBoost được trình bày như một phương pháp hiện đại, khắc phục được những nhược điểm của Logistic Regression và Decision Tree nhờ cơ chế boosting. Mô hình này không chỉ nâng cao độ chính xác dự đoán mà còn kiểm soát overfitting thông qua regularization, hỗ trợ xử lý dữ liệu phức tạp và tối ưu hóa tốc độ huấn luyện. Với các đặc tính vượt trội này, XGBoost được lựa chọn là phương pháp chính để áp dụng trong nghiên cứu.

Tóm lại, ba thuật toán đã được phân tích giúp làm rõ cơ sở khoa học cho việc lựa chọn XGBoost. Logistic Regression đóng vai trò tham chiếu cơ bản, Decision Tree cung cấp nền tảng cho ý tưởng boosting, còn XGBoost đại diện cho sự phát triển nâng cao, đáp ứng tốt các yêu cầu về độ chính xác, tính tổng quát và hiệu quả thực thi trong bối cảnh bài toán đặt ra.

# CHƯƠNG 3: KẾT QUẢ THỰC NGHIỆM

## 3.1. Cài đặt thực nghiệm

### *3.1.1. Tập dữ liệu*

Bộ dữ liệu được sử dụng trong nghiên cứu là Cleveland Heart Disease Dataset, được trích xuất từ kho dữ liệu nổi tiếng UCI Machine Learning Repository. Đây là một trong những bộ dữ liệu phổ biến nhất trong lĩnh vực học máy liên quan đến y tế, đặc biệt là các nghiên cứu về dự đoán bệnh tim.

Cụ thể, bộ dữ liệu bao gồm 303 bản ghi, mỗi bản ghi đại diện cho một cá nhân được thu thập thông tin lâm sàng. Từ bộ dữ liệu gốc với 75 thuộc tính, nhóm nghiên cứu đã lựa chọn và giữ lại 14 thuộc tính quan trọng có liên quan trực tiếp đến chẩn đoán bệnh tim. Các thuộc tính này phản ánh nhiều khía cạnh sức khỏe như: tuổi, giới tính, huyết áp, cholesterol, điện tâm đồ, nhịp tim, cơn đau ngực, kết quả kiểm tra gắng sức, v.v.

Điểm đáng chú ý là bộ dữ liệu không chứa giá trị thiếu, giúp cho quá trình xử lý và phân tích dữ liệu trở nên thuận lợi và chính xác hơn.

Nhiệm vụ chính đặt ra cho bộ dữ liệu này là bài toán phân loại nhị phân: dự đoán xem một cá nhân có khả năng mắc bệnh tim hay không, với biến mục tiêu được mã hóa như sau:

0: không mắc bệnh tim (absence).

1: có mắc bệnh tim (presence).

Tên bộ dữ liệu: Heart Disease Cleveland

Nguồn: https://www.kaggle.com/datasets/ritwikb3/heart-disease-cleveland/data

| | age | sex | cp | trestbps | chol | fbs | restecg | thalach | exang | oldpeak | slope | ca | thal | target |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 63 | 1 | 0 | 145 | 233 | 1 | 2 | 150 | 0 | 2.3 | 2 | 0 | 2 | 0 |
| **1** | 67 | 1 | 3 | 160 | 286 | 0 | 2 | 108 | 1 | 1.5 | 1 | 3 | 1 | 1 |
| **2** | 67 | 1 | 3 | 120 | 229 | 0 | 2 | 129 | 1 | 2.6 | 1 | 2 | 3 | 1 |
| **3** | 37 | 1 | 2 | 130 | 250 | 0 | 0 | 187 | 0 | 3.5 | 2 | 0 | 1 | 0 |
| **4** | 41 | 0 | 1 | 130 | 204 | 0 | 2 | 172 | 0 | 1.4 | 0 | 0 | 1 | 0 |

*Hình 3.1. 5 dòng đầu của bộ dữ liệu*

*Bảng 3.1. Mô tả bộ dữ liệu*

| Tên thuộc tính | Kiểu dữ liệu | Mô tả | Giá trị / Phạm vi |
|---|---|---|---|
| Age | Số | Tuổi | 29-77 |
| Sex | Nhị phân | Giới tính | 0 = nữ, 1 = nam |
| Cp (Chest pain type) | Phân loại | Loại cơn đau | 0 = đau thắt ngực điển hình;<br>1 = đau thắt ngực không điển hình;<br>2 = đau không do thắt ngực;<br>3 = không có triệu chứng |
| trestbps (resting blood pressure) | Số | Huyết áp lúc nghỉ (mmHg) | 94-200 |
| Chol (serum cholesterol) | Số | Nồng độ cholesterol trong huyết thanh (mg/dl) | 0 = Không, 1 = Có |
| fbs (fasting blood sugar) | Nhị phân | Đường huyết lúc đói > 120 mg/dl | 0 = Không, 1 = Có |
| restecg (resting electrocardiographic results) | Phân loại | Kết quả điện tâm đồ lúc nghỉ | 0 = bình thường; 1 = ST-T bất thường; 2 = phì đại thất trái |
| thalach | Số | Nhịp tim tối đa đạt được | 71 – 202 |

| | | | |
|---|---|---|---|
| exang | Nhị phân | Đau thắt ngực khi gắng sức | 0 = Không, 1 = Có |
| oldpeak | Số | Độ chênh ST (so với lúc nghỉ) sau gắng sức | 0.0 – 6.2 |
| slope | Phân loại | Độ dốc đoạn ST khi gắng sức | 0 = dốc xuống; 1 = bằng phẳng; 2 = dốc lên |
| ca | Số | Số mạch máu chính được soi huỳnh quang (0–3) | 0 – 3 |
| thal | Phân loại | Kết quả kiểm tra thallium | 0 = không xác định;<br>1 = khiếm khuyết cố định;<br>2 = bình thường;<br>3 = khiếm khuyết có thể đảo ngược |
| target | Nhị phân | Nhãn mục tiêu: mắc bệnh tim hay không | 0 = Không mắc;<br>1 = Có mắc |

### *3.1.2. Môi trường thực nghiệm*

Google Colaboratory (Google Colab) là một môi trường thực nghiệm dựa trên nền tảng điện toán đám mây, được cung cấp miễn phí bởi Google. Colab cho phép người dùng viết và thực thi mã nguồn Python trực tiếp trên trình duyệt mà không cần cài đặt phần mềm phức tạp. Điểm mạnh nổi bật của Google Colab là hỗ trợ sẵn các thư viện phổ biến trong lĩnh vực khoa học dữ

liệu và học máy như NumPy, Pandas, Scikit-learn, TensorFlow, Keras, XGBoost, Matplotlib, Seaborn,… giúp rút ngắn đáng kể thời gian chuẩn bị môi trường.

Ngoài ra, Colab cung cấp khả năng kết nối với GPU và TPU miễn phí, giúp tăng tốc quá trình huấn luyện mô hình máy học so với việc chỉ sử dụng CPU thông thường. Dữ liệu có thể được tải trực tiếp từ máy tính cá nhân, Google Drive hoặc các nguồn trực tuyến, tạo sự linh hoạt trong quá trình thực nghiệm. Bên cạnh đó, việc lưu trữ notebook trên Google Drive cũng giúp dễ dàng chia sẻ, cộng tác và tái sử dụng kết quả nghiên cứu.


*Hình 3.2. Google Colaboratory*

Nhờ những ưu điểm trên, Google Colab đã trở thành môi trường lý tưởng cho việc thực nghiệm, kiểm thử và so sánh các mô hình học máy trong nghiên cứu, đặc biệt là đối với các bài toán yêu cầu tài nguyên tính toán lớn nhưng người nghiên cứu không có sẵn phần cứng mạnh.

Python là ngôn ngữ lập trình bậc cao, thông dịch, nổi bật với cú pháp đơn giản, dễ đọc và dễ học. Python đặc biệt mạnh trong lĩnh vực khoa học dữ liệu và trí tuệ nhân tạo nhờ hệ sinh thái thư viện phong phú như NumPy, Pandas, Matplotlib, Scikit-learn, TensorFlow, XGBoost. Với cộng đồng phát triển lớn và khả năng mở rộng linh hoạt, Python hiện là lựa chọn phổ biến để phân tích dữ liệu, xây dựng và thử nghiệm các mô hình học máy.


*Hình 3.3. Ngôn ngữ lập trình Python*

Google Drive là dịch vụ lưu trữ và đồng bộ dữ liệu trên nền tảng đám mây do Google cung cấp. Người dùng có thể tải lên, chia sẻ và truy cập dữ liệu ở bất kỳ đâu chỉ cần có kết nối Internet. Trong nghiên cứu khoa học dữ liệu, Google Drive thường được sử dụng để lưu trữ và quản lý tập dữ liệu, đồng thời dễ dàng tích hợp với Google Colab để đọc/ghi dữ liệu trực tiếp phục vụ cho quá trình tiền xử lý và huấn luyện mô hình.


*Hình 3.4. Google Drive*

Các thư viện sử dụng trong python cho chương trình thực nghiệm này:

- Numpy: hỗ trợ tính toán đại số tuyến tính và xử lý mảng số liệu.
- Pandas: thao tác, tiền xử lý và phân tích dữ liệu dạng bảng.

- Matplotlib, Seaborn: trực quan hóa dữ liệu và kết quả mô hình.

- Scikit-learn: cung cấp các thuật toán học máy, công cụ đánh giá và tối ưu hóa mô hình.

- Plotly Express: trực quan hóa dữ liệu tương tác, hỗ trợ khám phá và phân tích trực quan hiệu quả hơn.

- SciPy.stats: cung cấp các phép kiểm định thống kê, phân phối xác suất và công cụ phân tích dữ liệu.

### *3.1.3. Các thông số cài đặt thực nghiệm*

Để đảm bảo tính công bằng trong so sánh giữa các mô hình, toàn bộ thí nghiệm được triển khai trên môi trường Google Colab, với ngôn ngữ Python 3.10.

Dữ liệu sau tiền xử lý gồm 302 mẫu và 11 thuộc tính. Các bước xử lý giống nhau cho tất cả mô hình: chuẩn hóa dữ liệu số, mã hóa biến phân loại, và chia dữ liệu theo train_test_split với tỉ lệ 80% (train) - 20% (test) và k-fold cross validation (k = 5).

Chiến lược tối ưu tham số: Cả 3 mô hình đều được tinh chỉnh bằng GridSearchCV với cùng cấu hình cv=5, scoring='accuracy', và n_jobs=-1. Điều này giúp loại bỏ yếu tố thiên lệch khi một mô hình được huấn luyện hay đánh giá khác điều kiện.

**Triển khai mô hình**

Với SVM, các tham số quan trọng được tinh chỉnh gồm: C (hệ số phạt), kernel (linear, rbf, poly) và gamma (điều chỉnh độ ảnh hưởng của một mẫu).

Với Decision Tree, tham số được tìm kiếm bao gồm: max_depth (độ sâu cây), criterion (gini, entropy), min_samples_split và min_samples_leaf nhằm kiểm soát độ phức tạp và tránh overfitting.

Đối với XGBoost, các siêu tham số được tối ưu là n_estimators (số lượng cây), learning_rate (tốc độ học), max_depth (độ sâu), subsample và colsample_bytree để cân bằng giữa độ chính xác và khả năng tổng quát hóa.

Kết quả từ quá trình tinh chỉnh cho phép xác định bộ tham số tối ưu của từng mô hình, đồng thời so sánh trực tiếp dựa trên các thước đo chung là *Accuracy, Confusion Matrix, AUC-ROC* và *Classification Report*. Qua đó, có thể đánh giá mô hình nào cho hiệu suất dự đoán tốt nhất cũng như rút ra ưu, nhược điểm trong từng trường hợp.

### 3.2. Phương pháp đánh giá

Trong bài toán này, đối với nhiệm vụ phân loại, em sử dụng các chỉ số *Recall*, *Precision*, *F1-score* và *Accuracy* để đánh giá mô hình. Trong đó, *Recall* được coi trọng nhất vì phản ánh khả năng phát hiện đúng các trường hợp bệnh nhân mắc bệnh tim, giúp hạn chế tình trạng bỏ sót ca bệnh. *Precision* và *F1-score* được dùng để cân bằng giữa việc phát hiện đúng và giảm cảnh báo giả, trong khi *Accuracy* chỉ mang tính tham khảo do bộ dữ liệu có khả năng mất cân bằng.

*Bảng 3.2. Ma trận nhầm lẫn (Confusion matrix)*

| Lớp | | Phân lớp bởi hệ thống | |
|---|---|---|---|
| | | Thật | Giả |
| Phân lớp đúng (nhãn) | Thật | FP | FN |
| | Giả | TP | TN |

Trong đó:

- TP (True Positive): Số lượng mẫu dương (positive) được dự đoán đúng.

- TN (True Negative): Số lượng mẫu âm (negative) được dự đoán đúng.

- FP (False Positive): Số lượng mẫu âm bị dự đoán nhầm thành dương.

- FN (False Negative): Số lượng mẫu dương bị dự đoán nhầm thành âm.

Chỉ số *Accuracy* là thước đo phản ánh tỷ lệ các mẫu được phân loại đúng so với tổng số mẫu.

Công thức:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100\%$$

Chỉ số *Precision* chỉ ra rằng trong số các mẫu được dự đoán là có nguy cơ suy tim, mô hình đã dự đoán đúng bao nhiêu. *Precision* được coi trọng hơn trong các bài toán mà việc nhận nhầm FP (dự đoán bệnh nhân có nguy cơ suy tim nhưng thực tế không có) mang lại hậu quả nghiêm trọng, ví dụ như gây lo lắng không cần thiết hoặc điều trị thừa cho bệnh nhân.

Công thức:

$$Precision = \frac{TP}{TP + FP} \times 100\%$$

Chỉ số *Recall* chỉ ra rằng trong số các bệnh nhân thực sự có nguy cơ suy tim, mô hình dự đoán đúng được bao nhiêu. *Recall* được coi trọng hơn khi việc nhận nhầm các nhãn *Positive* thành *False Negative* (bỏ sót bệnh nhân có nguy cơ) có thể gây hậu quả nghiêm trọng, ví dụ như bệnh nhân không được theo dõi hoặc điều trị kịp thời, làm tăng nguy cơ biến chứng hoặc tử vong.

Công thức:

$$Recall = \frac{TP}{TP + FN} \times 100\%$$

*F1-score* là trung bình điều hòa của *Precision* và *Recall*, được sử dụng khi cần cân bằng giữa hai chỉ số này, đặc biệt trong các bộ dữ liệu mất cân bằng.

Công thức:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \times 100\%$$

## 3.3. Quy trình và kết quả thực nghiệm

### *3.3.1. Quy trình thực nghiệm*

Upload file dataset archive.zip lên Google Drive, connect Google Drive với Google Colab. Giải nén file archive.zip từ Google Drive vào thư mục data trong thư mục hiện tại, tự động tạo thư mục đích nếu chưa tồn tại.

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
import zipfile
import os

zip_path = "/content/drive/MyDrive/archive.zip"
extract_path = "data"

with zipfile.ZipFile(zip_path, 'r') as zip_ref:
    zip_ref.extractall(extract_path)
```

*Hình 3.5. Kết nối Google Drive và giải nén tệp dữ liệu*

Thiết lập đường dẫn và hiển thị 5 dòng đầu của bộ dữ liệu.

```
df = pd.read_csv('data/Heart_disease_cleveland_new.csv')
df.head()
```

| | age | sex | cp | trestbps | chol | fbs | restecg | thalach | exang | oldpeak | slope | ca | thal | target |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 63 | 1 | 0 | 145 | 233 | 1 | 2 | 150 | 0 | 2.3 | 2 | 0 | 2 | 0 |
| **1** | 67 | 1 | 3 | 160 | 286 | 0 | 2 | 108 | 1 | 1.5 | 1 | 3 | 1 | 1 |
| **2** | 67 | 1 | 3 | 120 | 229 | 0 | 2 | 129 | 1 | 2.6 | 1 | 2 | 3 | 1 |
| **3** | 37 | 1 | 2 | 130 | 250 | 0 | 0 | 187 | 0 | 3.5 | 2 | 0 | 1 | 0 |
| **4** | 41 | 0 | 1 | 130 | 204 | 0 | 2 | 172 | 0 | 1.4 | 0 | 0 | 1 | 0 |

*Hình 3.6. 5 dòng đầu của bộ dữ liệu*

Hiển thị 5 dòng đầu tiên của bộ dữ liệu để quan sát và định hình bộ dữ liệu giúp ta có thể thao tác với dữ liệu hiệu quả hơn.

Xử lý dữ liệu thiếu và trùng lặp:

```
missing_vals = df.isnull().sum()
missing_pct = (missing_vals / len(df)) * 100
missing_df = pd.DataFrame({'Missing Values': missing_vals, 'Percentage': missing_pct})
print(missing_df[missing_df['Missing Values'] > 0])
```

```
Empty DataFrame
Columns: [Missing Values, Percentage]
Index: []
```

```
duplicate_count = df.duplicated().sum()
print(f"\nNumber of duplicate records: {duplicate_count}")
```

```

Number of duplicate records: 0
```

*Hình 3.7. Xử lý dữ liệu thiếu và trùng lặp*

Qua hình 3.7 ta có thể thấy rằng bộ dữ liệu hoàn toàn không có dữ liệu thiếu cũng như không có bản ghi nào bị trùng lặp nhau.

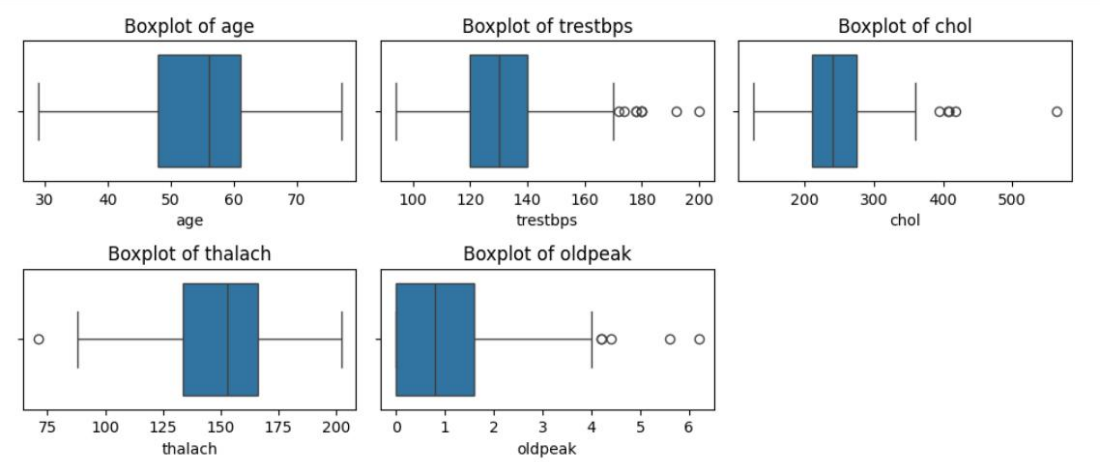
Phân loại dữ liệu dạng số và dạng phân loại:

```
numerical_cols = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']
categorical_cols = ['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca']
```

*Hình 3.8 Phân loại dữ liệu*

Thực hiện phân chia bộ dữ liệu thành dữ liệu dạng số và dữ liệu dạng phân loại để thuận tiện cho việc tiền xử lí dữ liệu.

Trực quan hóa dữ liệu ngoại lai của cột dạng số:




*Hình 3.9. Biểu đồ Boxplot của dữ liệu dạng số*

Các biểu đồ Boxplot trong hình 3.9 cho thấy phân bố dữ liệu của các thuộc tính dạng số:

- Age: phân bố chủ yếu trong khoảng từ 45 - 60 tuổi, dữ liệu khá cân đối và ổn định, không có giá trị ngoại lai rõ rệt.

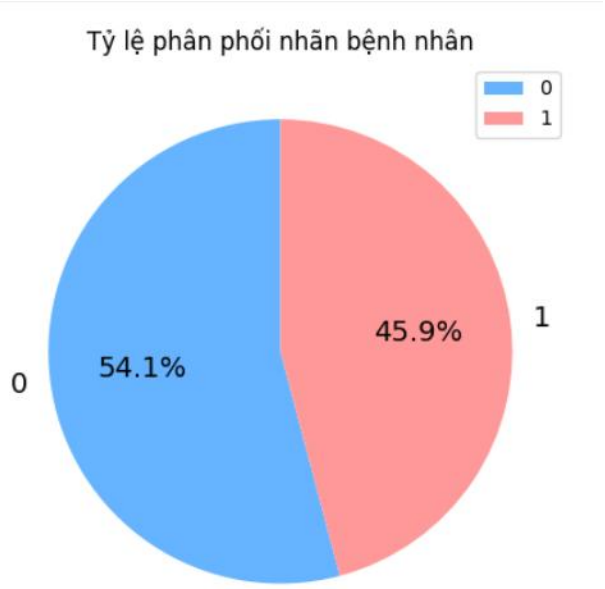
- Trestbps: đa số bệnh nhân có huyết áp trong khoảng 120 - 140 mmHg. Tuy nhiên xuất hiện nhiều giá trị ngoại lai ở phía cao, cho thấy có một số bệnh nhân có chỉ số huyết áp cao bất thường.

- Chol: Phân bố chủ yếu trong khoảng 200 - 300 mg/dl. Có nhiều ngoại lai cao trên 400 cho thấy tình trạng mỡ máu nghiêm trọng ở một số bệnh nhân.

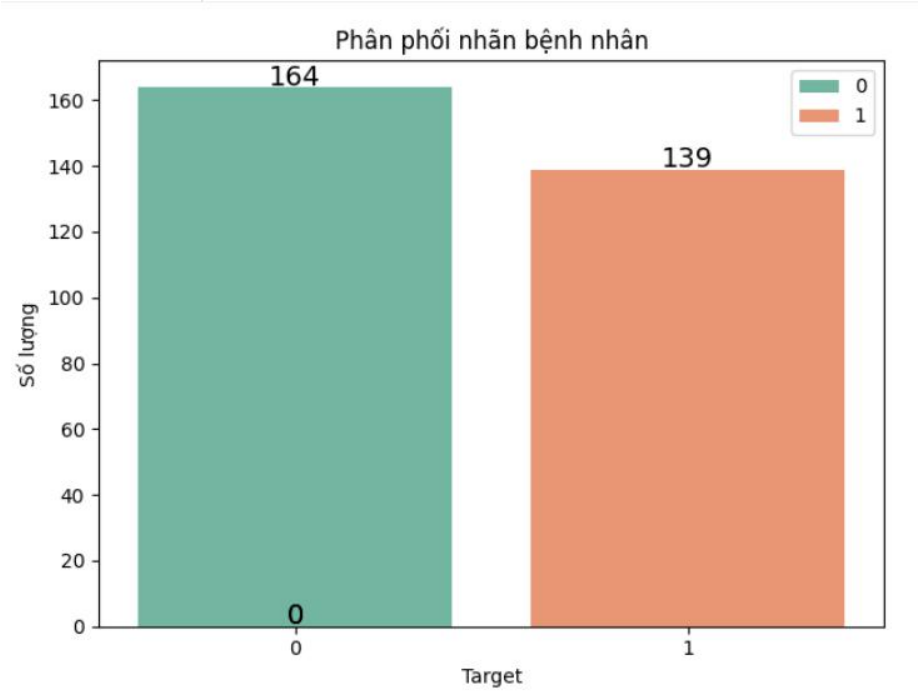
- Thalach: Tập trung trong khoảng 140 - 170 nhịp/phút. Có một vài ngoại lai thấp, nhưng dữ liệu phân bố tương đối đồng đều.

- Oldpeak: Xuất hiện nhiều giá trị ngoại lai ở phía cao, cho thấy có nhiều  bệnh nhân có chỉ số bất thường.

Tỷ lệ phân phối nhãn bệnh nhân:




*Hình 3.10. Tỷ lệ phân phối nhãn bệnh nhân*




*Hình 3.11. Phân phối nhãn bệnh nhân*

Dữ liệu bệnh nhân được phân bố tương đối cân bằng giữa hai nhóm. Cụ thể, nhóm bệnh nhân không mắc bệnh tim (Target = 0) chiếm khoảng 54,1% (tương ứng 164 trường hợp), trong khi nhóm bệnh nhân mắc bệnh tim (Target = 1) chiếm 45,9% (139 trường hợp). Sự phân bố này cho thấy tập dữ liệu không bị mất cân bằng nghiêm trọng.

Áp dụng phương pháp Anova F-test và Chi-square test để chọn lọc đặc trưng.

```
ANOVA F-test for numerical variables vs target:

age: F-statistic = 15.7696, p-value = 0.0001 (Significant)
trestbps: F-statistic = 7.0066, p-value = 0.0085 (Significant)
chol: F-statistic = 2.1991, p-value = 0.1391 (Not significant)
thalach: F-statistic = 63.4192, p-value = 0.0000 (Significant)
oldpeak: F-statistic = 66.1667, p-value = 0.0000 (Significant)
```

*Hình 3.12. Kết quả của chọn lọc đặc trưng dạng số*

Qua hình 3.11 có thể thấy được các biến số: age, trestbps, thalach, oldpeak đều có p-value rất nhỏ ( < 0.05) chứng tỏ các biến này có ý nghĩa thống kê và ảnh hưởng rõ rệt tới biến mục tiêu. Biến "chol" có giá trị p-value = 0.1391 ( > 0.05) không có ý nghĩa thống kê nên không phải yếu tố quan trọng với biến mục tiêu.

```
Chi-square test for categorical variables:

sex: p-value = 0.0000 (Significant)
cp: p-value = 0.0000 (Significant)
fbs: p-value = 0.7813 (Not significant)
restecg: p-value = 0.0066 (Significant)
exang: p-value = 0.0000 (Significant)
slope: p-value = 0.0000 (Significant)
thal: p-value = 0.0000 (Significant)
ca: p-value = 0.0000 (Significant)
```

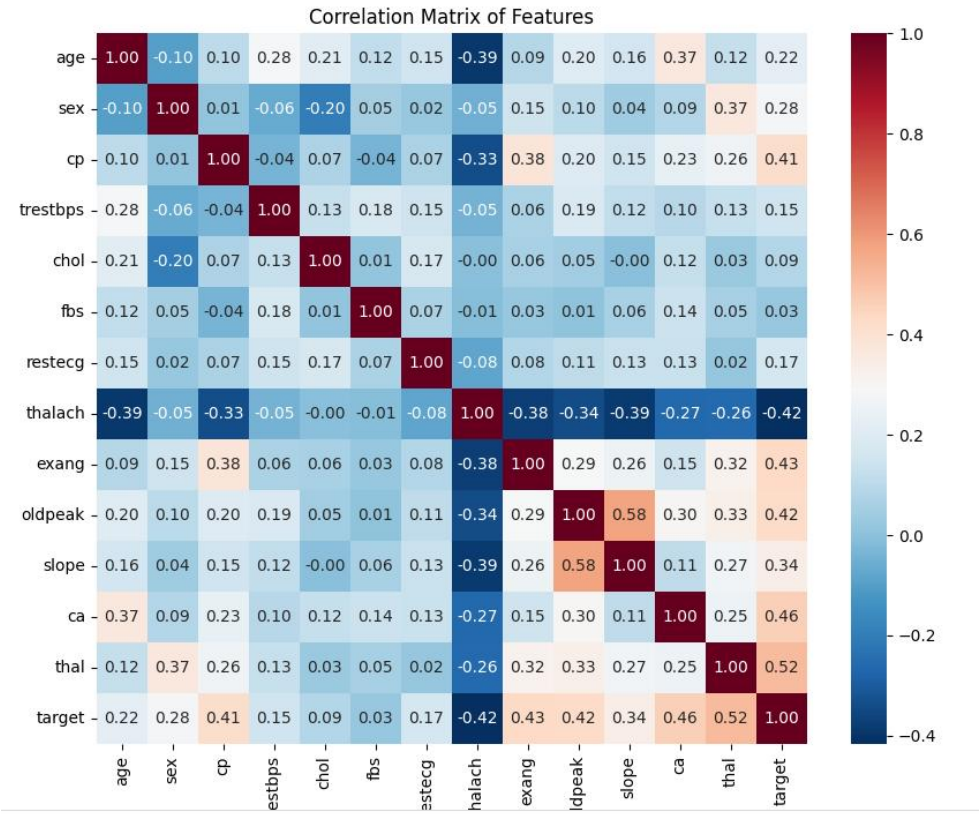
*Hình 3.13. Kết quả chọn lọc đặc trưng dạng phân loại.*

Các biến phân loại: sex, cp, restecg, exang, slope, thal, ca đều có giá trị p-value rất nhỏ ( < 0.05) có nghĩa là các biến này có ý nghĩa thống kê và có ảnh hưởng rõ rệt tới biến mục tiêu. Biến "fbs" có p-value = 0.7813 ( > 0.05) không có ý nghĩa thống kê nên không ảnh hưởng rõ rệt tới biến mục tiêu.

Anova F-test (Analysis of Variance F-test) là một phương pháp thống kê được sử dụng để kiểm tra sự khác biệt về giá trị trung bình của một biến số giữa nhiều nhóm phân loại. Trong bối cảnh học máy, Anova F-test thường được áp dụng như một kỹ thuật để chọn lọc đặc trưng với dữ liệu đầu vào là các biến độc lập dạng số và biến mục tiêu ở dạng phân loại. Giá trị F càng lớn và giá trị p-value càng nhỏ chứng tỏ mức độ khác biệt càng có ý nghĩa thống kê. Nhờ đó giúp loại bỏ các đặc trưng liên quan, giảm độ nhiễu và nâng cao hiệu quả huấn luyện mô hình.

Chi-square test là một phương pháp thống kê dùng để kiểm tra mối quan hệ phụ thuộc giữa 2 biến phân loại. Trong học máy, kiểm định này thường được sử dụng để chọn lọc đặc trưng khi biến độc lập và biến mục tiêu đều ở dạng phân loại. Nguyên lý của phương pháp dựa trên việc so sánh sự khác biệt giữa tần suất quan sát thực tế và tần suất kỳ vọng. Nếu giá trị kiểm định lớn và p-value nhỏ, điều đó cho thấy biến có liên quan đáng kể đến nhãn phân loại.

Ma trận tương quan của các đặc trưng:




*Hình 3.14. Ma trận tương quan của các đặc trưng*

Một số biến có tương quan mạnh với biến mục tiêu “target” bao gồm: thalach, cp, exang, oldpeak, slope, ca, thal với hệ số khoảng (0.3 - 0.5). Các biến có độ tương quan trung bình bao gồm: age, sex, trestbps, restecg với hệ số trong khoảng (0.1 - 0.28). Còn lại biến “chol” và “fbs” với hệ số lần lượt là 0.09 và 0.03 gần như không có mối tương quan với biến “target”. Từ ma trận tương quan có thể cân nhắc loại bỏ hoặc giảm trọng số các biến ít tương quan khi huấn luyện mô hình.

Phân chia bộ dữ liệu thành tập train và test:

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

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

*Hình 3.15. Phân chia bộ dữ liệu*

Thực hiện phân chia bộ dữ liệu thành 2 phần với 80% dữ liệu huấn luyện và 20% dữ liệu kiểm tra với random_state = 42 để đảm bảo tính tái lập và stratify=y để giữ tỷ lệ phân bố lớp cân bằng giữa tập huấn luyện và tập kiểm tra.

Chuẩn hóa dữ liệu với StandardScaler:

```
num_cols = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']
```

```
scaler = StandardScaler()

X_train_scaled = X_train.copy()
X_test_scaled = X_test.copy()

X_train_scaled[num_cols] = scaler.fit_transform(X_train[num_cols])
X_test_scaled[num_cols] = scaler.transform(X_test[num_cols])
```

*Hình 3.16. Chuẩn hóa dữ liệu*

Tiền hành chuẩn hóa dữ liệu với StandardScaler để giúp các mô hình học máy hoạt động tốt hơn và tránh việc một đặc trưng có giá trị lớn chi phối các đặc trưng khác.

Cân bằng nhãn với SMOTE:

```
from imblearn.over_sampling import SMOTE

smote = SMOTE(random_state=42, k_neighbors=5)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train_scaled, y_train)

print("Phân bố nhãn ban đầu: ", y_train.value_counts())
print("Phân bố nhãn sau SMOTE: ", y_train_resampled.value_counts())
```

```
Phân bố nhãn ban đầu:  target
0    131
1    111
Name: count, dtype: int64
Phân bố nhãn sau SMOTE:  target
1    131
0    131
Name: count, dtype: int64
```

*Hình 3.17. Cân bằng nhãn với SMOTE*

Do dữ liệu ban đầu có sự mất cân bằng giữa nhãn 0 và 1 nên thực hiện cân bằng nhãn với SMOTE sẽ giúp cả hai lớp có số lượng bằng nhau giúp cho mô hình tránh được thiên lệch trong dự đoán.

**Quá trình huấn luyện**

Bộ dữ liệu sau khi được chọn lọc đặc trưng được chia thành 2 tập với tỉ lệ 80:20 trong đó 80% dùng để huấn luyện mô hình và 20% còn lại dùng để kiểm tra, đánh giá và so sánh các mô hình. Tiếp theo, tập huấn luyện được áp dụng phương pháp Cross Validation với k=5, nghĩa là chia thành 5 phần có kích thước tương đương, để đảm bảo tính khách quan và giảm hiện tượng overfitting.

Cụ thể, đề tài sẽ huấn luyện 3 mô hình bao gồm Decision Tree, Support Vector Machine, Decision Tree và XGBoost. Bộ dữ liệu được sử dụng là Cleveland Heart Disease.


*Hình 3.18. Phân chia bộ dữ liệu*

Xây dựng mô hình huấn luyện Decision Tree:

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.calibration import CalibratedClassifierCV
from sklearn.metrics import classification_report

# GridSearch cho Decision Tree
param_grid_dt = {
    'max_depth': [3, 5, 7, None],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'criterion': ['gini', 'entropy']
}

# Khởi tạo mô hình DT
dt = DecisionTreeClassifier(random_state=42)

# GridSearchCV
grid_dt = GridSearchCV(dt, param_grid_dt, cv=cv, scoring='accuracy', n_jobs=-1)
grid_dt.fit(X_train, y_train)

# CalibratedClassifierCV (để tính xác suất chuẩn hơn cho ROC curve)
cal_dt = CalibratedClassifierCV(grid_dt.best_estimator_, cv=cv)
cal_dt.fit(X_train, y_train)

# Dự đoán
y_pred_dt = cal_dt.predict(X_test)

# Kết quả
print("Best Decision Tree Params:", grid_dt.best_params_)
print(classification_report(y_test, y_pred_dt))
```

*Hình 3.19. Khởi tạo mô hình Decision Tree*

Xây dựng mô hình huấn luyện SVM:

```
from sklearn.svm import SVC
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV
from sklearn. (module) inspection libratedClassifierCV
from sklearn.inspection import permutation_importance
from sklearn.metrics import classification_report
from sklearn.preprocessing import StandardScaler

svm_pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('svm', SVC(probability=True, class_weight='balanced'))
])

param_grid_svm = {
    'svm__kernel': ['linear', 'rbf', 'poly'],
    'svm__C': [0.1, 1, 10],
    'svm__gamma': ['scale', 'auto'],
    'svm__degree': [2, 3]
}

grid_svm = GridSearchCV(
    svm_pipeline, param_grid=param_grid_svm,
    cv=cv, scoring='accuracy', n_jobs=-1
)
grid_svm.fit(X_train, y_train)

cal_svm = CalibratedClassifierCV(estimator=grid_svm.best_estimator_, cv=cv)
cal_svm.fit(X_train, y_train)

y_pred_svm = cal_svm.predict(X_test)
print("Best SVM Params:", grid_svm.best_params_)
print(classification_report(y_test, y_pred_svm))
```

*Hình 3.20. Khởi tạo mô hình SVM*

Xây dựng mô hình huấn luyện XGBoost:

```
from xgboost import XGBClassifier, plot_importance
from sklearn.model_selection import GridSearchCV
from sklearn.calibration import CalibratedClassifierCV
from sklearn.metrics import classification_report

param_grid_xgb = {
    'n_estimators': [100, 200],
    'learning_rate': [0.05, 0.1],
    'max_depth': [3, 5, 7],
    'subsample': [0.8, 1.0]
}

xgb = XGBClassifier(use_label_encoder=False, eval_metric='logloss', random_state=42)

grid_xgb = GridSearchCV(xgb, param_grid_xgb, cv=cv, scoring='accuracy', n_jobs=-1)
grid_xgb.fit(X_train, y_train)

cal_xgb = CalibratedClassifierCV(grid_xgb.best_estimator_, cv=cv)
cal_xgb.fit(X_train, y_train)

y_pred_xgb = cal_xgb.predict(X_test)
print("Best XGBoost Params:", grid_xgb.best_params_)
print(classification_report(y_test, y_pred_xgb))
```

*Hình 3.21. Khởi tạo mô hình XGBoost*

### *3.3.2. Kết quả thực nghiệm*

Trong báo cáo này, em đã triển khai các mô hình trên bộ dữ liệu "Cleveland Heart Disease". Các mô hình được đánh giá và so sánh dựa trên các chỉ số *Accuracy*, *Precision*, *chỉ số AUC* và *biểu đồ ROC*. Thông qua những chỉ số này có thể đánh giá hiệu suất của mô hình trong việc phân biệt nhãn (có bệnh hoặc không có bệnh). Đồng thời để phân tích chi tiết hơn hiệu quả dự đoán, em đã xây dựng ma trận nhầm lẫn (*Confusion Matrix*) cho từng mô hình, cho phép quan sát rõ ràng số lượng mẫu được phân loại đúng và sai. Giúp cho việc đánh giá trở nên dễ dàng hơn.

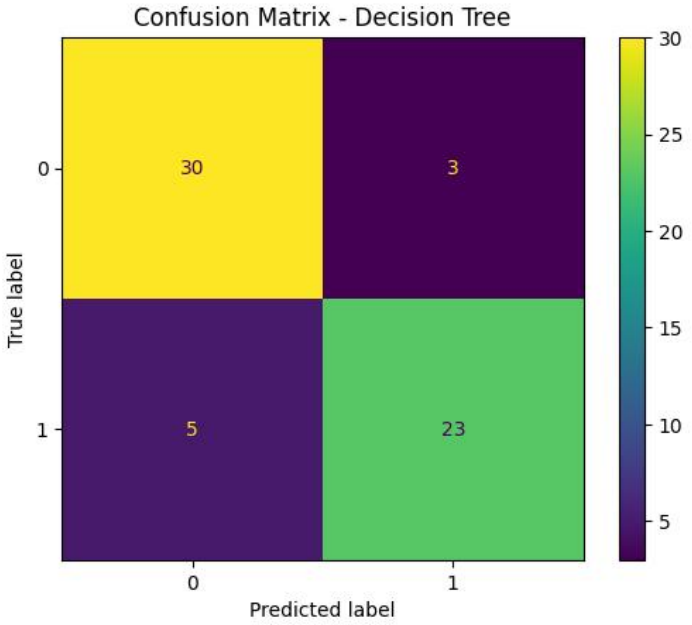
Ngoài ra để kết quả có ý nghĩa lâm sàng thực tế hơn, em đã sử dụng các chỉ số và khoảng tin cậy tại ngưỡng lâm sàng như: *Sensivity, Specifility, PPV/NPV, F1, 95% CI*. Các chỉ số này rất quen thuộc với các bác sĩ trong lĩnh vực y tế, giúp người sử dụng dễ dàng diễn giải hơn. Biểu đồ độ tin cậy cho xác suất các dự đoán của mô hình là *Reliability curve* để đánh giá độ ổn định cũng như độ tin cậy của mô hình.

*Bảng 3.3. Kết quả của các mô hình*

| | Accuracy | Precision | recall | F1 |
|---|---|---|---|---|
| Decision Tree | 0.868 | 0.88 | 0.82 | 0.85 |
| Support Vector Machine | 0.868 | 0.83 | 0.89 | 0.86 |
| XGBoost | 0.918 | 0.89 | 0.928 | 0.91 |

Trên cơ sở kết quả thực nghiệm, mô hình XGBoost cho thấy hiệu suất vượt trội so với các mô hình còn lại. Cụ thể, XGBoost đạt *Accuracy* = 0.918, *Precision* = 0.89, *Recall* = 0.928 và *F1-score* = 0.91. Các chỉ số này chứng minh rằng mô hình không chỉ dự đoán chính xác ở mức tổng thể mà còn duy trì sự cân bằng tốt giữa việc phát hiện đúng các trường hợp dương tính (*Recall* cao) và hạn chế báo sai (*Precision* hợp lý).

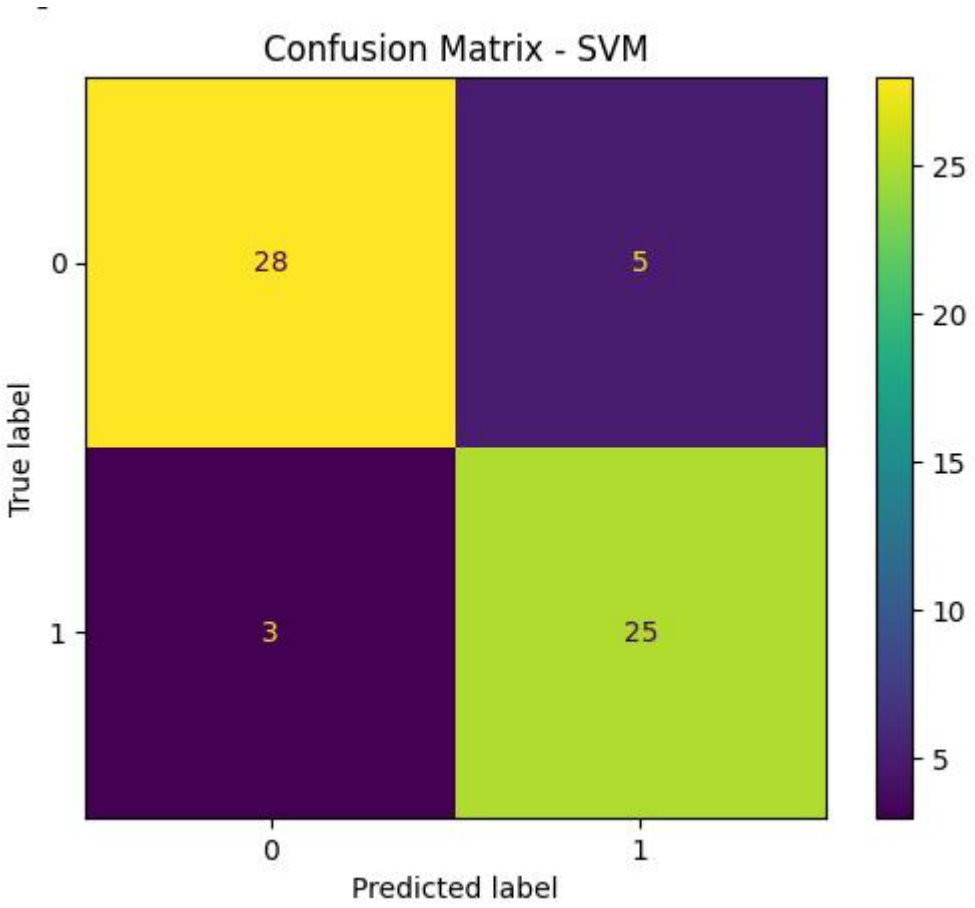
Ma trận nhầm lẫn của mô hình Decision Tree (DT).




*Hình 3.22. Ma trận nhầm lẫn của mô hình DT*

Kết quả ma trận nhầm lẫn của mô hình Decision Tree cho thấy có 30 mẫu âm tính và 23 mẫu dương tính được dự đoán chính xác. Tuy nhiên, mô hình vẫn mắc phải 3 trường hợp âm tính bị dự đoán nhầm thành dương tính (False Positive) và 5 trường hợp dương tính bị dự đoán thành âm tính (False Negative). Điều này phản ánh rằng Decision Tree có khả năng phân loại tương đối tốt, nhưng vẫn tồn tại hạn chế trong việc cân bằng giữa hai loại sai số, đặc biệt cần cải thiện ở khả năng nhận diện đúng các ca dương tính.

Ma trận nhầm lẫn (Confusion Matrix) của mô hình SVM.

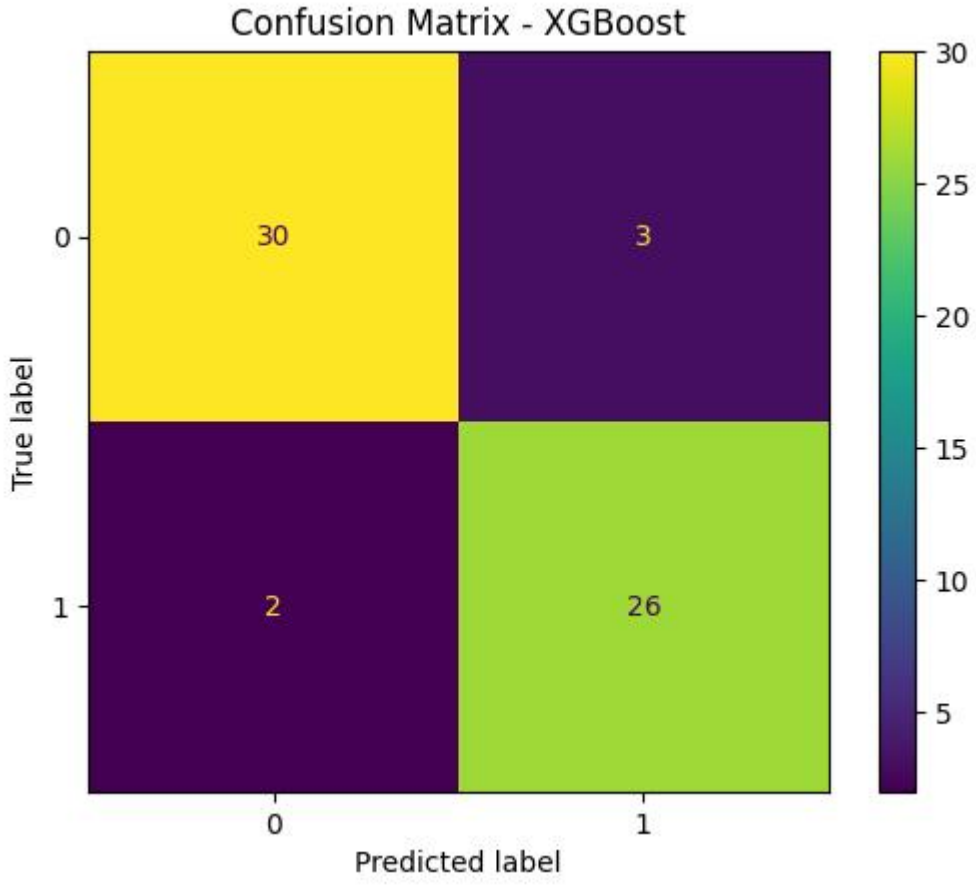



*Hình 3.23. Ma trận nhầm lẫn của mô hình DT*

Kết quả ma trận nhầm lẫn của mô hình SVM cho thấy mô hình dự đoán chính xác 28 mẫu âm tính và 25 mẫu dương tính. Tuy nhiên, vẫn tồn tại 5 trường hợp âm tính bị dự đoán sai thành dương tính (False Positive) và 3 trường hợp dương tính bị dự đoán sai thành âm tính (False Negative). So với Decision Tree, SVM giảm được số lượng False Negative, nghĩa là mô hình có

khả năng phát hiện các ca dương tính tốt hơn, mặc dù số lượng False Positive vẫn giữ nguyên. Điều này chứng tỏ SVM cân bằng tốt hơn giữa hai loại sai số và có độ tin cậy cao hơn trong việc phân loại bệnh nhân dương tính.

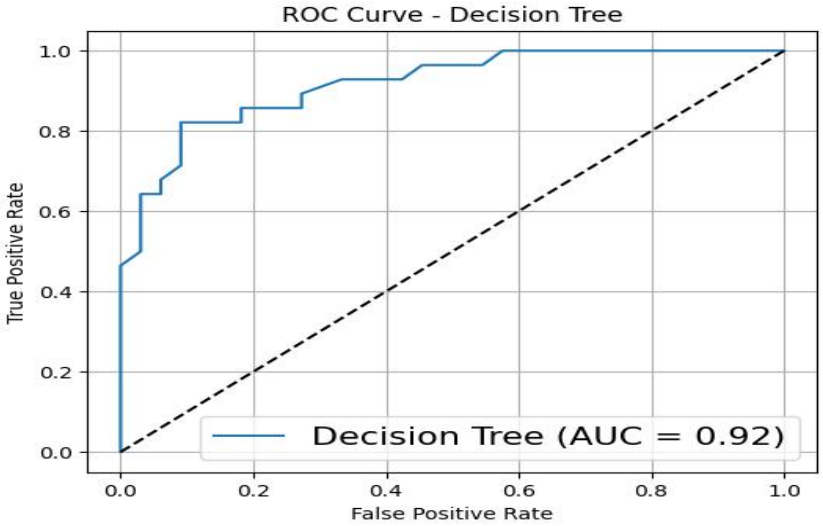
Ma trận nhầm lẫn (Confusion Matrix) của mô hình XGBoost.




*Hình 3.24. Ma trận nhầm lẫn của mô hình XGBoost*

Ma trận nhầm lẫn của mô hình XGBoost cho thấy kết quả vượt trội so với Decision Tree và SVM. Cụ thể, XGBoost dự đoán đúng 30 mẫu âm tính và 26 mẫu dương tính, trong khi số trường hợp dự đoán sai chỉ còn 3 mẫu False Positive và 2 mẫu False Negative – thấp nhất trong ba mô hình. Điều này chứng tỏ XGBoost vừa giảm thiểu được sai sót trong nhận diện bệnh nhân âm tính, vừa nâng cao độ chính xác trong việc phát hiện bệnh nhân dương tính. Nhờ vậy, XGBoost đạt được sự cân bằng tốt giữa các chỉ số Accuracy, Precision, Recall và đặc biệt phù hợp với các bài toán y tế yêu cầu độ tin cậy cao.

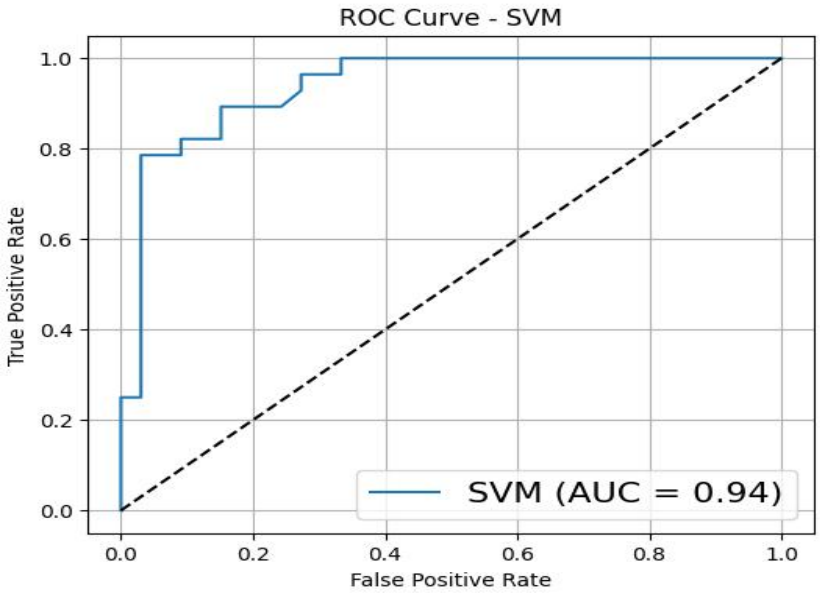
Biểu đồ ROC của mô hình Decision Tree (DT).




*Hình 3.25. Biểu đồ ROC của mô hình DT*

Biểu đồ ROC của mô hình Decision Tree cho thấy đường cong nằm cao hơn rõ rệt so với đường chéo ngẫu nhiên, với chỉ số AUC = 0.94. Điều này chứng tỏ mô hình có khả năng phân biệt tốt giữa hai lớp (dương tính và âm tính), đồng thời đạt được sự cân bằng hiệu quả giữa True Positive Rate (TPR) và False Positive Rate (FPR). Với AUC cao như vậy, Decision Tree cho thấy hiệu suất khá tốt, mặc dù trong thực tế kết quả này vẫn có thể bị hạn chế bởi hiện tượng overfitting.

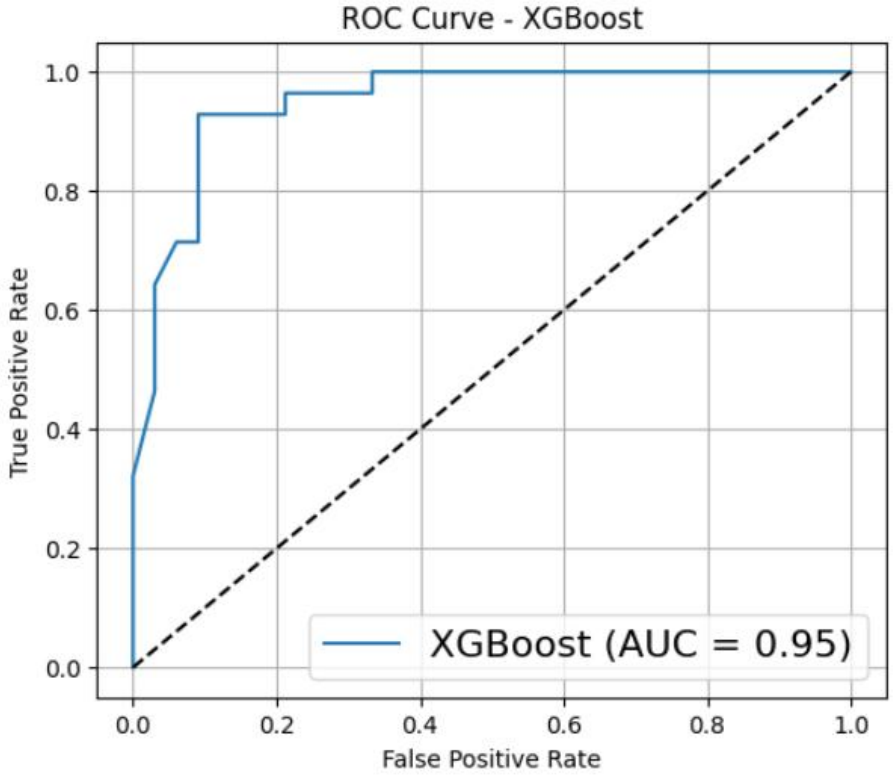
Biểu đồ ROC của mô hình SVM.




*Hình 3.26. Biểu đồ ROC của mô hình SVM*

Biểu đồ ROC của mô hình SVM cho thấy đường cong ôm sát trục tung và nằm cao hơn so với đường chéo ngẫu nhiên, với chỉ số AUC = 0.95. Kết quả này chứng tỏ SVM có khả năng phân loại hai lớp rất tốt, đồng thời giảm thiểu sai số loại I (False Positive). So với Decision Tree, SVM cải thiện một chút về khả năng dự đoán tổng thể, cho thấy tính ổn định và độ chính xác cao hơn.

Biểu đồ ROC của mô hình XGBoost.




*Hình 3.27. Biểu đồ ROC của mô hình XGBoost*

Biểu đồ ROC của mô hình XGBoost cho thấy đường cong ôm sát trục tung và nằm cao hơn so với đường chéo ngẫu nhiên, với chỉ số AUC = 0.95. Kết quả này chứng tỏ XGBoost có khả năng phân loại hai lớp rất tốt, đồng thời giảm thiểu sai số loại I (False Positive). So với Decision Tree, XGBoost cải thiện rõ rệt về khả năng dự đoán tổng thể, cho thấy tính ổn định và độ chính xác cao hơn.

*Bảng 3.4. Kết quả mô hình theo ý nghĩa lâm sàng*

| | Sensivity | Specifility | PPV | NPV | F1 | 95% CI |
|---|---|---|---|---|---|---|
| DT | 0.82 | 0.90 | 0.88 | 0.85 | 0.85 | 0.66 - 0.95 |
| SVM | 0.89 | 0.84 | 0.83 | 0.90 | 0.86 | 0.75 - 1.0 |
| XGBoost | 0.92 | 0.90 | 0.89 | 0.93 | 0.91 | 0.82 - 1.0 |

Qua bảng 3.4 ta có thể thấy rằng mô hình XGBoost đạt được kết quả tốt nhất với Sensivity = 0.92 và NPV = 0.93 cho thấy mô hình có khả năng phát hiện ca dương tính tốt và ít bỏ sót ca bệnh. Chỉ số Specifility = 0.90 và F1 = 0.91 cũng rất cao cho thấy mô hình có khả năng loại bỏ các ca dương tính giả rất tốt. Khoảng tin cậy 95% CI (0.82 - 1.0) rất nhỏ cho thấy mô hình dự đoán có hiệu suất tốt và cho độ tin cậy cao.

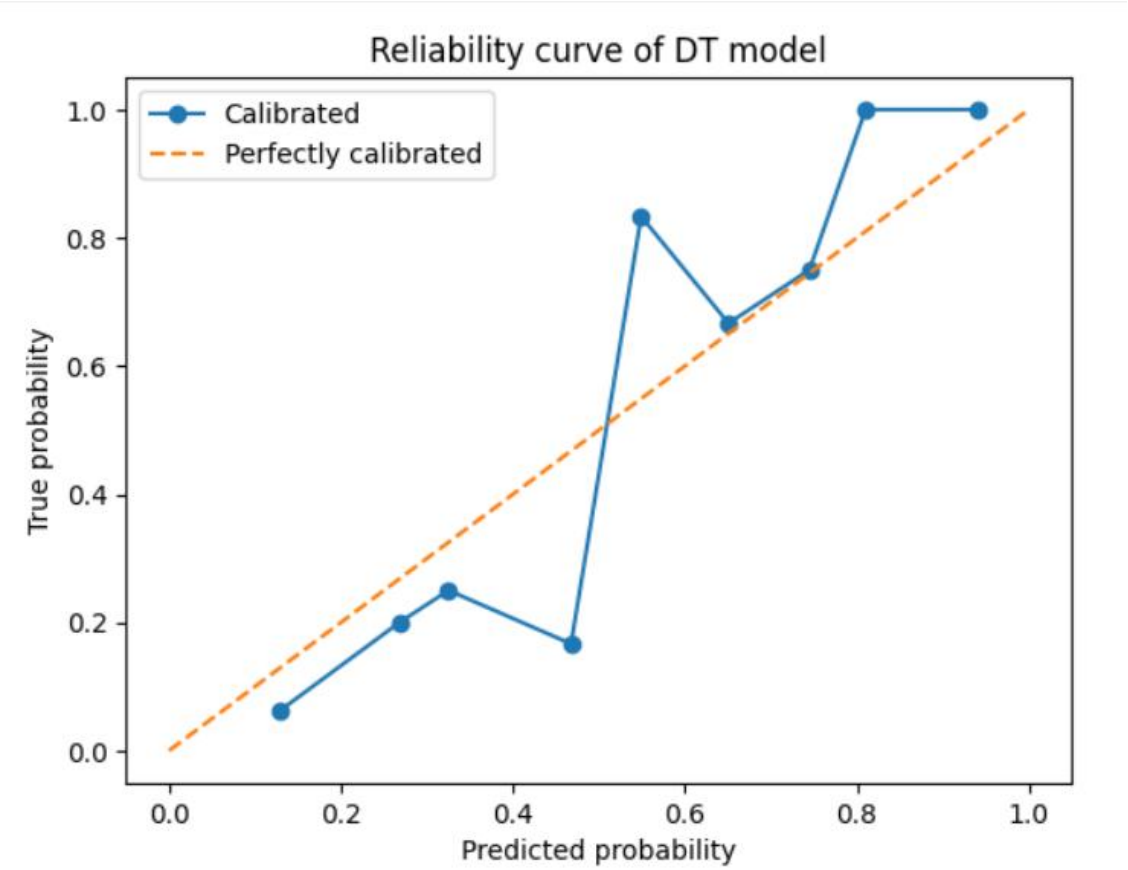
Bên cạnh đó, mô hình DT có chỉ số Sensivity = 0.82 và NPV = 0.85 còn khá thấp cho thấy mô hình nhận diện các ca dương tính kém và khả năng bỏ sót các ca mắc bệnh cao hơn, tuy nhiên chỉ số Specifility = 0.90 khá tốt cho thấy mô hình có khả năng loại bỏ các ca âm tính giả tốt. Khoảng tin cậy 95% CI (0.66 - 0.95) tương đối rộng thể hiện hiệu suất và độ tin cậy của mô hình chưa được cao.

Mô hình SVM có chỉ số Sensivity = 0.89 và NPV = 0.90 rất tốt cho thấy khả năng phát hiện ca dương tính và tỉ lệ bỏ sót ca mắc bệnh ít (gần bằng XGboost). Tuy nhiên chỉ số Specifility = 0.84 và F1 = 0.86 còn khá thấp thể hiện mô hình vẫn còn hạn chế với việc phát hiện ca dương tính giả. Khoảng tin cậy (0.75 - 1.0) tương đối rộng thể hiện hiệu suất của mô hình chưa tối ưu và độ tin cậy chưa cao.

Như vậy, trong số các mô hình được so sánh, XGBoost thể hiện hiệu quả vượt trội với độ nhạy và độ đặc hiệu đều cao, cùng khoảng tin cậy hẹp, chứng tỏ khả năng dự đoán ổn định và đáng tin cậy. Trong khi đó, SVM cũng

cho kết quả khá tốt nhưng vẫn còn hạn chế ở chỉ số đặc hiệu, còn Decision Tree có hiệu suất thấp hơn và độ tin cậy chưa cao, do đó chưa phù hợp để áp dụng trong thực tiễn bằng các mô hình còn lại.

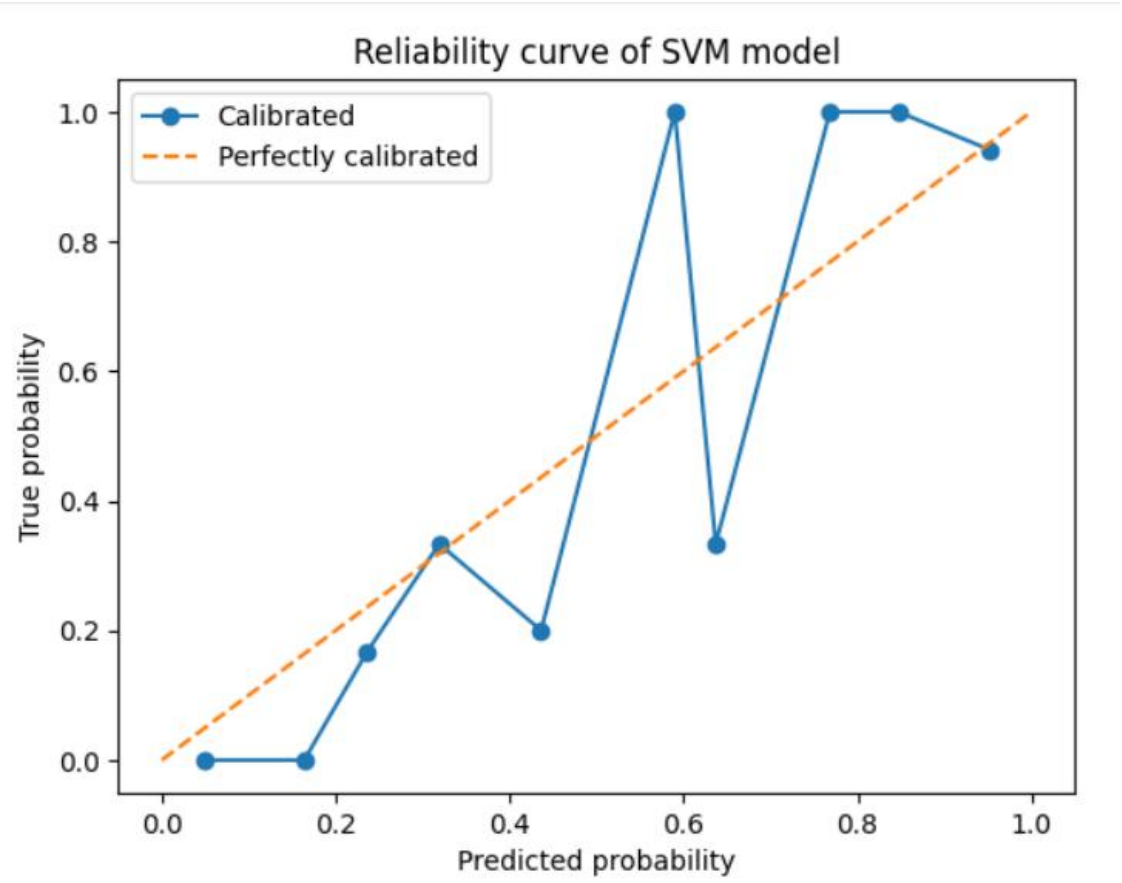
Biểu đồ độ tin cậy của mô hình Decision Tree:




*Hình 3.28. Biểu đồ độ tin cậy của mô hình DT*

Qua hình 3.28, ta có thể thấy đường hiệu chỉnh (Calibrated) nằm khá gần với đường chuẩn (Perfectly calibrated) ở nhiều mức xác suất khác nhau chứng tỏ rằng mô hình có khả năng dự báo xác suất tương đối phù hợp với thực tế, đặc biệt ở các khoảng trung bình và cao. Tuy nhiên, đường dự báo ở các mức xác suất thấp còn dao động cho thấy mô hình có xu hướng không ổn định khi xác suất dự đoán nhỏ.

Biểu đồ độ tin cậy của mô hình SVM:

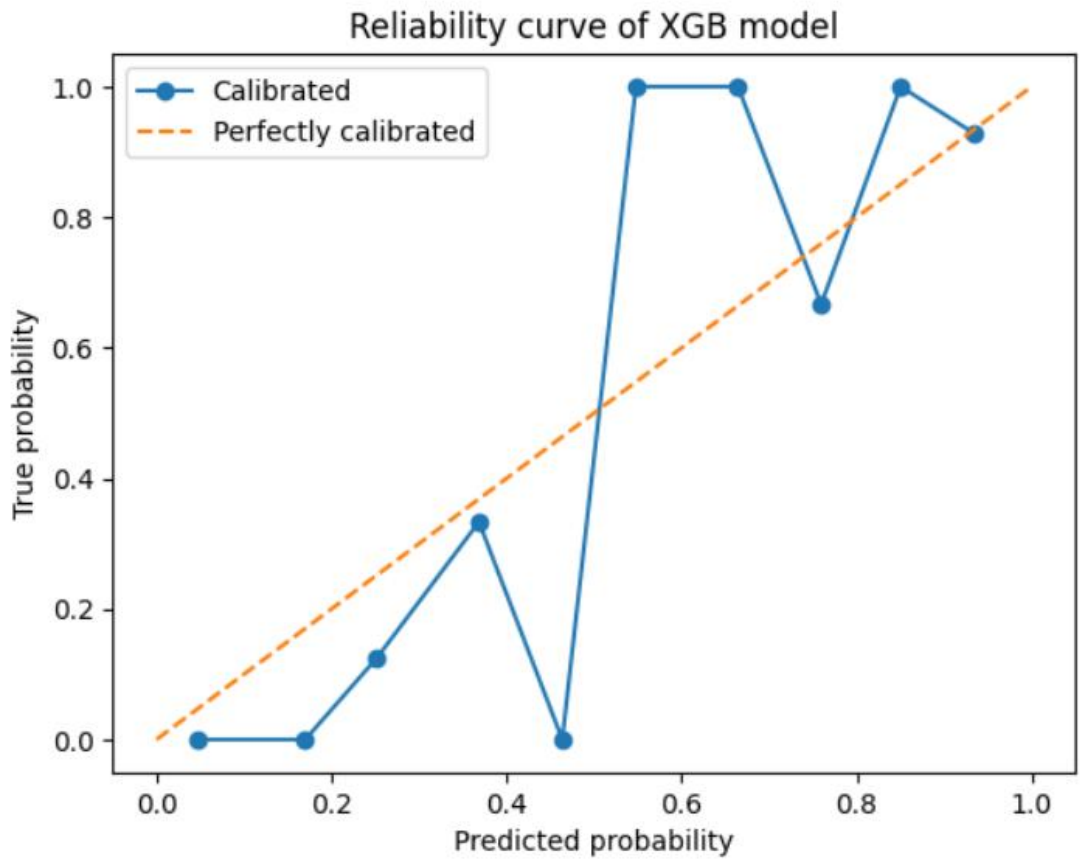



*Hình 3.29. Biểu đồ độ tin cậy của mô hình SVM*

Qua hình 3.29 ta có thể thấy rằng đường hiệu chỉnh (calibrated) có nhiều đoạn dao động xa so với đường chuẩn (Perfectly calibrated). Ở mức xác suất thấp và trung bình, mô hình dự báo chưa sát với giá trị thực tế cho thấy hiệu suất mô hình chưa được tốt. Tuy nhiên, ở các mức xác suất cao, đường hiệu chỉnh tiến gần đến đường chuẩn cho thấy dự đoán của mô hình với mức xác suất cao rất tin cậy.

Biểu đồ độ tin cậy của mô hình XGBoost:

Từ hình 3.30 ta có thể thấy rằng đường hiệu chỉnh (calibrated) bám khá sát với đường chuẩn (Perfectly calibrated), đặc biệt ở các mức xác suất trung bình và cao, mô hình cho thấy sự phù hợp tốt giữa xác suất dự báo và thực tế. Mặc dù vẫn có những điểm dao động nhẹ nhưng so với DT và SVM, XGBoost cho thấy khả năng hiệu chỉnh tốt hơn và rất đáng tin cậy.




*Hình 3.30. Biểu đồ độ tin cậy của mô hình XGBoost*

Qua quá trình thực nghiệm và so sánh, có thể thấy rằng cả ba mô hình Decision Tree, SVM và XGBoost đều cho kết quả khả quan trên bộ dữ liệu Cleveland Heart Disease. Mô hình Decision Tree mặc dù đạt AUC = 0.94 nhưng còn hạn chế về độ chính xác và F1-score, cho thấy khả năng khái quát hóa chưa cao. Mô hình SVM cải thiện đáng kể với AUC = 0.95, đồng thời thể hiện sự cân bằng tốt giữa Precision và Recall, giúp nâng cao độ tin cậy trong dự đoán. Tuy nhiên, mô hình XGBoost cho thấy hiệu quả vượt trội nhất với Accuracy = 88.5%, Precision = 83.9%, Recall = 92.9%, F1-score = 88.1% và AUC = 0.96, chứng minh khả năng dự đoán chính xác và ổn định.

Bên cạnh đó, XGBoost còn cho thấy hiệu năng vượt trội với các chỉ số lâm sàng như Sensitivity, Specificity, PPV, NPV và F1 đều cao, khoảng tin cậy nhỏ chứng tỏ mô hình có tính ổn định. Biểu đồ hiệu chỉnh cũng cho thấy XGBoost có khả năng dự đoán xác suất phù hợp với tần suất thực tế, đảm bảo

độ tin cậy khi ứng dụng trong bối cảnh lâm sàng. Trong khi đó, SVM có Sensivity khá tốt nhưng khả năng hiệu chỉnh xác suất còn hạn chế, dễ dẫn đến dự đoán thiếu chính xác ở ngưỡng trung gian. Decision Tree tuy đơn giản, dễ triển khai nhưng hiệu năng tổng thể và tính ổn định thấp hơn. Do đó, XGBoost được đánh giá là mô hình tối ưu trong nghiên cứu này và sẽ được áp dụng vào xây dựng sản phẩm demo.

## Tổng kết chương 3

Chương 3 đã trình bày về tập dữ liệu Cleveland Heart Disease, bao gồm số lượng mẫu, cách chia dữ liệu, môi trường thực nghiệm, các thông số cài đặt cũng như các phương pháp đánh giá như Accuracy, Precision, Recall, F1-score, Confusion Matrix và ROC-AUC, Sensivity, Specifility, PPV, NPV, F1 và biểu đồ Reliability Curve. Đây là những phương pháp đánh giá phù hợp để kiểm định hiệu quả của các mô hình phân loại trong y sinh học.

Bên cạnh đó, chương cũng đã trình bày về quá trình huấn luyện và kết quả của từng mô hình đối với tập dữ liệu. Mô hình XGBoost đạt được các chỉ số Accuracy, Precision, Recall và F1-score và các chỉ số lâm sàng Sensivity, Specifility, PPV, NPV, F1 cao nhất cũng như biểu đồ độ tin cậy ổn định nhất, thể hiện khả năng học tốt và độ tin cậy cao hơn so với các mô hình còn lại. Mô hình SVM cũng cho kết quả khá tốt, đặc biệt thể hiện hiệu quả trong việc xử lý dữ liệu có kích thước nhỏ và biên phân tách phức tạp, tuy nhiên còn hạn chế khi dự đoán xác suất ở mức trung bình. Trong khi đó, Decision Tree có ưu điểm trực quan và dễ hiểu, tuy nhiên độ chính xác, các chỉ số lâm sàng và độ tin cậy thấp hơn so với XGBoost và SVM.

Nhìn chung, các kết quả thực nghiệm cho thấy XGBoost là mô hình tiềm năng nhất để áp dụng vào bài toán dự đoán nguy cơ bệnh tim, trong khi SVM và Decision Tree vẫn có thể đóng vai trò hỗ trợ trong các tình huống yêu cầu sự cân bằng giữa độ chính xác và khả năng giải thích.

# CHƯƠNG 4: SẢN PHẨM DEMO

## 4.1. Giới thiệu về Framework sử dụng (Framework Streamlit)

### *4.1.1. Giới thiệu về Streamlit*

Streamlit là một framework mã nguồn mở (open-source) được viết bằng Python, chuyên dụng cho việc xây dựng và triển khai các ứng dụng web cho các dự án Machine Learning và Data Science. Điểm nổi bật của Streamlit nằm ở sự đơn giản và trực quan, cho phép người dùng tạo ra các ứng dụng web tương tác chỉ với vài dòng code Python mà không cần kiến thức chuyên sâu về phát triển web front-end (HTML, CSS, JavaScript).

Streamlit hoạt động bằng cách chạy một script Python và tự động tạo ra một ứng dụng web tương ứng. Mỗi khi script được thay đổi và lưu lại, ứng dụng web sẽ tự động cập nhật theo thời gian thực (hot-reloading), giúp quá trình phát triển và thử nghiệm diễn ra nhanh chóng và hiệu quả.

### *4.1.2. Vai trò của Streamlit*

Streamlit đóng vai trò như một cầu nối giữa code Python và giao diện người dùng web, giúp các nhà khoa học dữ liệu và kỹ sư machine learning dễ dàng chia sẻ công việc của mình với người khác. Một số trường hợp sử dụng phổ biến của Streamlit bao gồm:

- Prototype nhanh: Xây dựng nhanh chóng các bản demo và prototype cho các mô hình Machine Learning và ứng dụng Data Science.

- Chia sẻ kết quả phân tích dữ liệu: Trình bày kết quả phân tích dữ liệu một cách trực quan và tương tác thông qua các biểu đồ, bảng biểu và widget.

- Xây dựng dashboard: Tạo các dashboard theo dõi dữ liệu và hiệu suất, hiển thị các chỉ số quan trọng một cách dễ hiểu.

- Giảng dạy và học tập: Minh họa các khái niệm Machine Learning và Data Science thông qua các ứng dụng web tương tác.

- Xây dựng các ứng dụng web nhỏ và vừa: Phát triển các ứng dụng web tập trung vào xử lý và hiển thị dữ liệu.

### *4.1.3. Ưu điểm*

- Dễ sử dụng: Cú pháp đơn giản, dễ học và dễ sử dụng, ngay cả với người mới bắt đầu.

- Nhanh chóng: Phát triển ứng dụng nhanh chóng nhờ tính năng hot-reloading và các component có sẵn.

- Tương tác: Cung cấp nhiều widget tương tác giúp người dùng dễ dàng thao tác với ứng dụng.

- Triển khai dễ dàng: Hỗ trợ triển khai lên Streamlit Cloud và các nền tảng khác một cách thuận tiện.

- Miễn phí và mã nguồn mở: Có thể sử dụng miễn phí cho cả dự án cá nhân và thương mại.

- Cộng đồng hỗ trợ mạnh mẽ: Cộng đồng người dùng đông đảo và nhiệt tình.

### *4.1.4. Nhược điểm*

- Khả năng tùy chỉnh giao diện hạn chế: So với các framework web truyền thống, khả năng tùy chỉnh giao diện của Streamlit còn hạn chế.

- Phụ thuộc vào server: Ứng dụng Streamlit cần một server để chạy.

- Chưa phù hợp cho các ứng dụng web phức tạp: Streamlit phù hợp hơn cho các ứng dụng web nhỏ và vừa, không phải là lựa chọn tốt nhất cho các ứng dụng web quy mô lớn và phức tạp.

- Hạn chế về xử lý logic phía server: Streamlit tập trung vào việc hiển thị và tương tác, việc xử lý logic phức tạp phía server có thể gặp khó khăn.

### *4.1.5. Ứng dụng của Streamlit*

- Streamlit được ứng dụng rộng rãi trong nhiều lĩnh vực, đặc biệt là trong Machine Learning và Data Science. Dưới đây là một số ví dụ cụ thể:

- Phân tích dữ liệu thăm dò (Exploratory Data Analysis - EDA): Xây dựng các ứng dụng web tương tác để khám phá và phân tích dữ liệu.

- Trực quan hóa dữ liệu: Tạo các biểu đồ và đồ thị tương tác để hiển thị dữ liệu một cách trực quan.

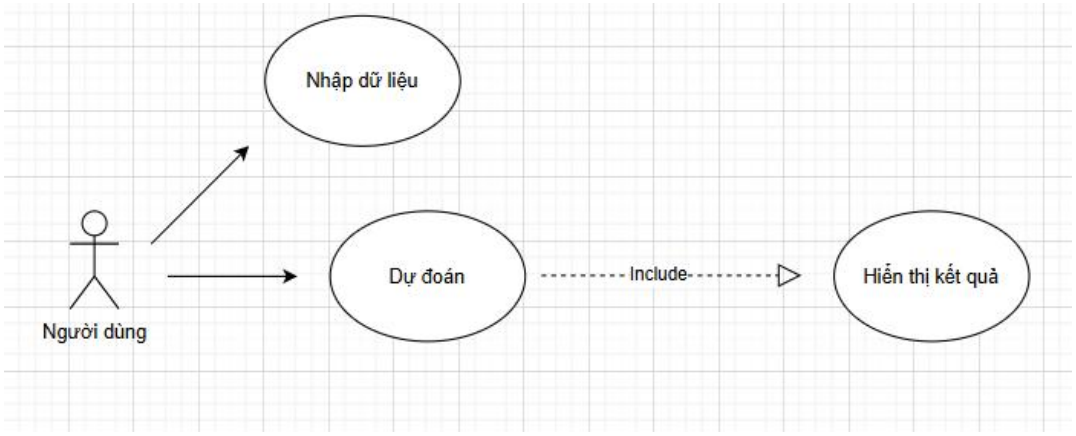
- Xây dựng mô hình Machine Learning: Huấn luyện và đánh giá các mô hình Machine Learning, đồng thời triển khai chúng dưới dạng ứng dụng web.

- Theo dõi và giám sát mô hình: Xây dựng dashboard để theo dõi hiệu suất của mô hình Machine Learning trong thời gian thực.

## 4.2. Phân tích thiết kế

Trang web của bài toán có chức năng chính là dự đoán nguy cơ mắc bệnh. Dựa trên các thông tin y tế của người dùng nhập vào để đưa ra kết quả dự đoán và xác suất dựa trên độ chính xác của mô hình.

### *4.2.1. Biểu đồ use case tổng quát*




*Hình 4.1. Biểu đồ use case tổng quát*

### *4.2.2. Mô tả chi tiết use case*

**Use case Nhập dữ liệu**

*Bảng 4.1. Mô tả chi tiết use case Nhập dữ liệu*

| | |
|---|---|
| Tên use case | Nhập dữ liệu |
| Mô tả tóm tắt | Nhập dữ liệu dự đoán nguy cơ bệnh tim |
| Luồng sự kiện | - Người dùng nhập vào các thông tin y tế cơ bản. |
| Luồng thay thế | - Người dùng nhập sai định dạng, hệ thống thông báo lỗi và yêu cầu nhập lại. |

| | |
|---|---|
| | - Người dùng có thể nhấn nút reset để nhập lại từ đầu. |
| Các điều kiện đặc biệt | Không có |
| Tiền điều kiện | Không có |
| Hậu điều kiện | Không có |
| Điểm mở rộng | Không có |

**Use case Dự đoán:**

*Bảng 4.2. Mô tả chi tiết use case Dự đoán*

| Tên use case | Dự đoán |
|---|---|
| Mô tả tóm tắt | Use case cho phép người dùng dự bắt đầu quá trình dự đoán |
| Luồng sự kiện | - Người dùng nhấn nút dự đoán trong giao diện trang web, sau đó kết quả dự đoán sẽ hiển thị |
| Luồng thay thế | Không có |
| Các điều kiện đặc biệt | Không có |
| Tiền điều kiện | Không có |
| Hậu điều kiện | Không có |
| Điểm mở rộng | Không có |

*Bảng 4.3. Mô tả chi tiết use case Tải PDF*

| Tên use case | Tải PDF |
|---|---|
| Mô tả tóm tắt | Use case cho phép người dùng có thể tải xuống kết quả dự đoán dưới dạng PDF từ hệ thống |
| Luồng sự kiện | - Người dùng nhấn nút Xuất kết quả ra PDF trong giao diện trang web, sau đó nhấn nút |

| | |
|---|---|
| | Tải PDF để tải xuống. |
| Luồng thay thế | Không có |
| Các điều kiện đặc biệt | Không có |
| Tiền điều kiện | Không có |
| Hậu điều kiện | Không có |
| Điểm mở rộng | Không có |

## 4.3. Giao diện màn hình

**Giao diện màn hình ban đầu:**


*Hình 4.2. Giao diện màn hình ban đầu*

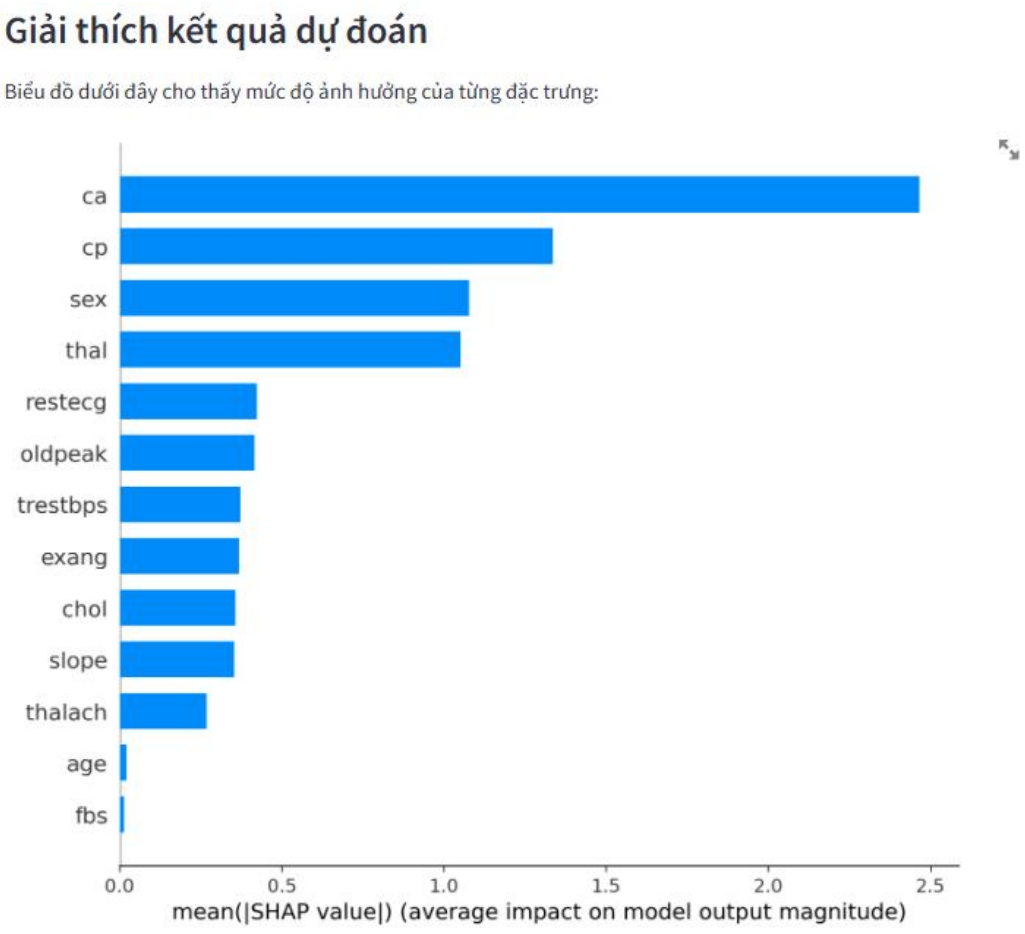
Người dùng sẽ điền các thông tin y tế vào đúng theo tên từng mục.

**Giao diện màn hình dự đoán:**


*Hình 4.3. Giao diện màn hình dự đoán*

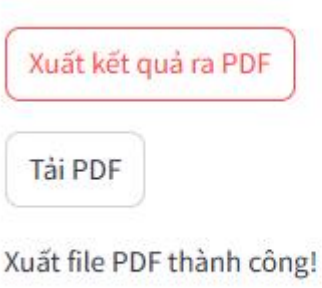
Người dùng nhấn nút "Dự đoán" để thực hiện dự đoán.

**Giao diện màn hình biểu đồ đặc trưng quan trọng:**



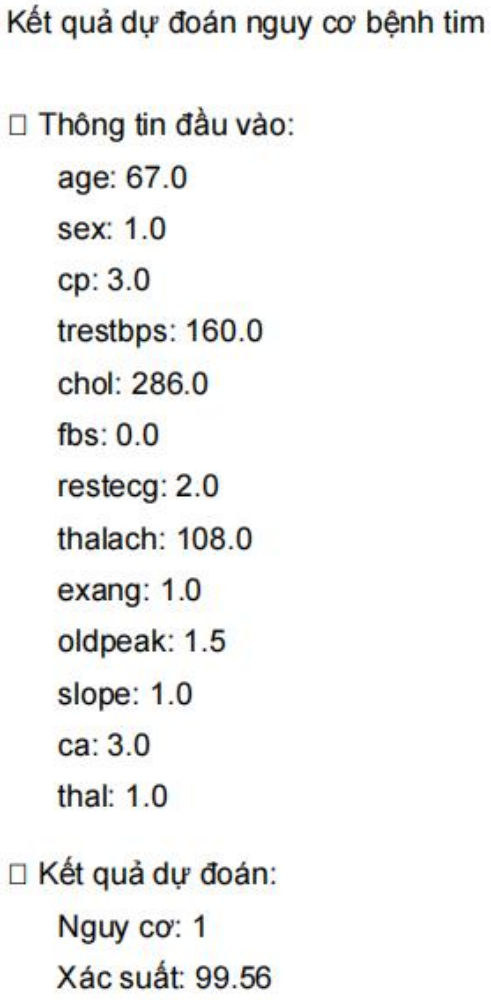

*Hình 4.4. Giao diện màn hình biểu đồ đặc trưng quan trọng*

**Giao diện màn hình xuất file PDF kết quả dự đoán:**

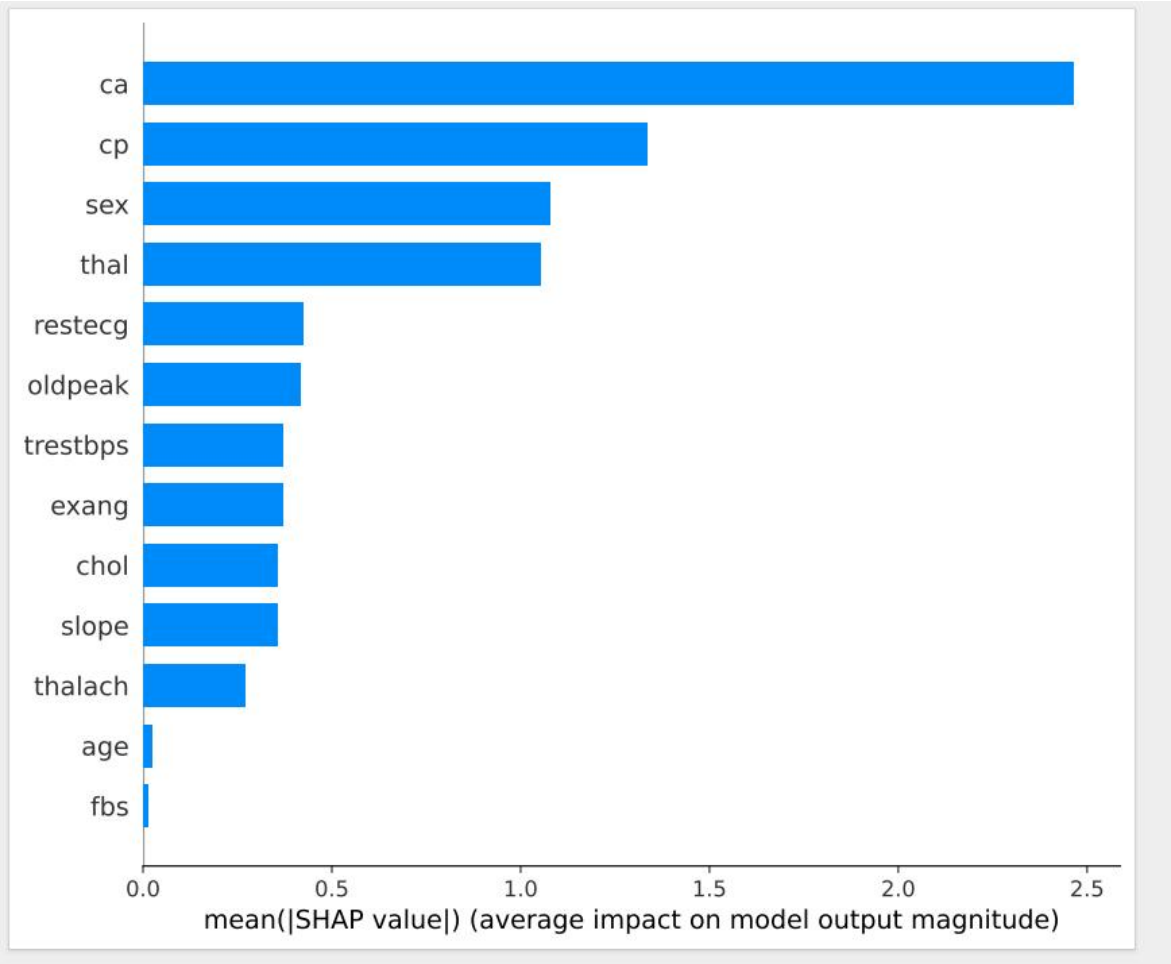



*Hình 4.5. Giao diện màn hình xuất file kết quả dự đoán*

**Giao diện màn hình file PDF kết quả dự đoán:**




*Hình 4.6. Giao diện màn hình file PDF kết quả dự đoán*




*Hình 4.7. Giao diện màn hình file PDF kết quả dự đoán*

## Tổng kết chương 4

Chương 4 đã trình bày quá trình xây dựng sản phẩm demo dựa trên mô hình XGBoost. Nội dung bao gồm phân tích và thiết kế hệ thống thông qua biểu đồ use case tổng quát, bảng mô tả chi tiết các chức năng cũng như kiến trúc xử lý dữ liệu. Phần front-end được phát triển bằng Streamlit, hỗ trợ giao diện thân thiện cho phép người dùng nhập các thông tin lâm sàng liên quan đến bệnh tim, xem kết quả dự đoán theo xác suất, trực quan hóa các đặc trưng ảnh hưởng quan trọng qua các biểu đồ và tải về file PDF kết quả dự đoán. Phần back-end đảm nhận việc tải và sử dụng mô hình XGBoost đã được huấn luyện, xử lý dữ liệu đầu vào và trả về kết quả dự đoán. Sản phẩm đã đáp ứng được mục tiêu đề ra, minh họa rõ ràng khả năng ứng dụng mô hình học máy trong việc hỗ trợ dự đoán nguy cơ mắc bệnh tim.

# KẾT LUẬN

## Kết luận

Đề tài "Phân tích và dự đoán nguy cơ suy tim bằng mô hình XGBoost" đã hoàn thành các mục tiêu đặt ra, bao gồm xây dựng và huấn luyện mô hình học máy nhằm hỗ trợ phân tích dữ liệu y tế và dự đoán nguy cơ suy tim. Ứng dụng thuật toán XGBoost, hệ thống không chỉ đạt được độ chính xác cao mà còn đảm bảo tính ổn định trong việc phân loại bệnh nhân theo mức độ rủi ro, từ đó có thể hỗ trợ bác sĩ trong việc chẩn đoán và đưa ra phác đồ điều trị kịp thời. Cụ thể:

- Mô hình XGBoost đạt giá trị AUC = 0.95, thể hiện khả năng phân biệt rõ ràng giữa bệnh nhân có nguy cơ suy tim và không có nguy cơ.

- Các chỉ số Accuracy, Precision, Recall và F1-score đều đạt mức cao, chứng tỏ mô hình cân bằng tốt giữa khả năng phát hiện bệnh nhân thật sự mắc bệnh và hạn chế báo động giả.

- Thực nghiệm trên tập dữ liệu cho thấy mô hình XGBoost có tính ổn định và hiệu quả hơn so với một số mô hình truyền thống khác.

Kết quả đạt được chứng minh rằng việc áp dụng các kỹ thuật học máy tiên tiến như XGBoost có tiềm năng lớn trong việc dự đoán bệnh lý tim mạch, đặc biệt là suy tim – một căn bệnh nguy hiểm nhưng có thể can thiệp kịp thời nếu được phát hiện sớm.

## Hướng phát triển:

Mặc dù đề tài đã đạt được những kết quả khả quan, tuy nhiên để nâng cao tính ứng dụng thực tiễn, một số hướng phát triển tiếp theo được đề xuất như sau:

- Mở rộng tập dữ liệu huấn luyện: Thu thập thêm dữ liệu đa dạng từ nhiều nguồn, nhiều đặc điểm nhân khẩu học khác nhau để cải thiện tính khái quát hóa của mô hình.

- Kết hợp nhiều mô hình học máy: Nghiên cứu các phương pháp ensemble hoặc hybrid để tối ưu hơn nữa độ chính xác dự đoán.

- Tích hợp vào hệ thống hỗ trợ bác sĩ: Phát triển giao diện trực quan, cho phép nhập dữ liệu lâm sàng và nhận kết quả dự đoán tức thì.
- Đảm bảo bảo mật dữ liệu y tế: Tuân thủ các tiêu chuẩn về an toàn và quyền riêng tư khi triển khai trên dữ liệu bệnh nhân thực tế.
- Hợp tác với cơ sở y tế: Thử nghiệm mô hình tại bệnh viện hoặc phòng khám để đánh giá hiệu quả lâm sàng và hiệu chỉnh mô hình phù hợp với thực tiễn.

# TÀI LIỆU THAM KHẢO

## Tài liệu tiếng Anh

[1]. Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785–794. URL: https://doi.org/10.1145/2939672.2939785.

[2]. Ahmad, T., Munir, A., Bhatti, S. H., Aftab, M., & Raza, M. A. (2017). Survival analysis of heart failure patients: A case study. PLOS ONE, 12(7), e0181001. URL: https://doi.org/10.1371/journal.pone.0181001

[3]. Choi, E., Schuetz, A., Stewart, W. F., & Sun, J. (2017). Using recurrent neural network models for early detection of heart failure onset. Journal of the American Medical Informatics Association, 24(2), 361–370. URL: https://doi.org/10.1093/jamia/ocw112

[5]. Kaggle. (2020). Heart Failure Clinical Records Dataset. https://www.kaggle.com/datasets/andrewmvd/heart-failure-clinical-data

[6]. C. M. Bhatt, P. Patel, T. Ghetia, và P. L. Mazzeo, "Effective Heart Disease Prediction Using Machine Learning Techniques," Algorithms, vol. 16, no. 2, art. no. 88, 2023, doi: 10.3390/a16020088.

[7]. Shorewala, V. Early detection of coronary heart disease using ensemble techniques. Inform. Med. Unlocked 2021, 26, 100655. URL: https://doi.org/10.1016/j.imu.2021.100655

[8]. Waigi, R.; Choudhary, S.; Fulzele, P.; Mishra, G. Predicting the risk of heart disease using advanced machine learning approach. Eur. J. Mol. Clin. Med. 2020, 7, 1638–1645.

[9]. World Health Organization, "Cardiovascular diseases (CVDs) Fact Sheet," WHO, 31 July 2025. URL: https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds).

[10]. Bragazzi, N. L., Zhong, W., Shu, J., Abu Much, A., Lotan, D., Grupper, A., Younis, A., & Mahaffey, K. W. (2022). Global burden of heart failure: A comprehensive and updated review of epidemiology. Cardiovascular Research, 118(17), 3272–3287. URL: https://doi.org/10.1093/cvr/cvac013

[11]. IBM, "What is a Decision Tree?", IBM Think, [Online]. URL: https://www.ibm.com/think/topics/decision-trees

[12]. IBM, "What is Logistic Regression?", IBM Think, [Online]. URL: https://www.ibm.com/think/topics/logistic-regression

**Tài liệu tiếng Việt**

[13]. "Ứng dụng kỹ thuật máy học vào phân loại bệnh tim," Tạp chí Khoa học Công nghệ Thực phẩm – HUFI, 2022.

[14]. Nguyễn Đức Dũng, Nguyễn Quang Huy, Phạm Thị Bích Ngọc (2021). Ứng dụng học máy trong dự đoán bệnh tim mạch. Tạp chí Khoa học và Công nghệ, Đại học Đà Nẵng.

[15]. Nguyễn Hữu Đức, Lê Văn Bình (2022). Mô hình học máy trong dự báo nguy cơ suy tim: Nghiên cứu thử nghiệm trên bộ dữ liệu công khai. Hội thảo Quốc gia về Công nghệ Thông tin và Truyền thông (ICT).

[16]. Nguyễn Văn Tuấn (2019). Ứng dụng thuật toán học máy trong phân loại và dự đoán bệnh lý y khoa. Tạp chí Tin học và Điều khiển học, Viện Hàn lâm KH&CN Việt Nam.

[17]. Tổng quan về học máy. URL:

https://vi.wikipedia.org/wiki/H%E1%BB%8Dc_m%C3%A1y .