**A yellow oval with red text

Description automatically generated**

TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC THÀNH PHỒ HỒ CHÍ MINH

**KHOA CÔNG NGHỆ THÔNG TIN**

🙡🕮🙣

**BÁO CÁO KẾT THÚC HỌC PHẦN**

**MÁY HỌC**

**XÂY DỰNG MÔ HÌNH CHẨN ĐOÁN BỆNH VIÊM GAN C**

Mã lớp học phần: **241123018401**

Năm học: **2024 – 2025**

Học kỳ: **1**

Sinh viên thực hiện:

1. Phạm Đình Anh Phương 22DH112900
2. Mai Hòa Hảo 22DH110957
3. Vũ Xuân Dương 22DH114489

***Thành phố Hồ Chí Minh, tháng 11 năm 2024***

MỤC LỤC

[DANH MỤC HÌNH i](#_Toc182925033)

[DANH MỤC BẢNG ii](#_Toc182925034)

[CHƯƠNG 1. GIỚI THIỆU MÔ HÌNH CHẨN ĐOÁN BỆNH VIÊM GAN C 3](#_Toc182925035)

[1.1. Giới thiệu bài toán 3](#_Toc182925036)

[1.2. Các công trình liên quan 3](#_Toc182925037)

[CHƯƠNG 2. CHUẨN BỊ DỮ LIỆU 4](#_Toc182925038)

[2.1. Giới thiệu tập dữ liệu 4](#_Toc182925039)

[2.2. Tiền xử lý và trực quan hóa dữ liệu 5](#_Toc182925040)

[2.3. Trích chọn đặc trưng 7](#_Toc182925041)

[CHƯƠNG 3. XÂY DỰNG MÔ HÌNH 8](#_Toc182925042)

[3.1. Thuật Toán SVM (Support Vector Machine) 8](#_Toc182925043)

[3.1.1. **Ý tưởng chính của SVM** 8](#_Toc182925044)

[3.1.2. **Siêu phẳng và khoảng cách lề (Margin)** 8](#_Toc182925045)

[3.1.3. **Hạt nhân (Kernel Trick)** 9](#_Toc182925046)

[3.1.4. **Hàm mục tiêu của SVM** 9](#_Toc182925047)

[3.1.5. **Ưu và nhược điểm** 10](#_Toc182925048)

[3.1.6. **Ví dụ trực quan** 10](#_Toc182925049)

[3.1.7. **So sánh với thuật toán khác** 11](#_Toc182925050)

[3.1.8. **Các bước thực hiện:** 11](#_Toc182925051)

[3.2. Thuật toán Cây Quyết Định (Decision Tree) 12](#_Toc182925052)

[3.2.1. **Cấu trúc của cây quyết định** 12](#_Toc182925053)

[3.2.2. **Quy trình xây dựng cây quyết định** 13](#_Toc182925054)

[3.2.3. **Tiêu chí chọn đặc trưng** 13](#_Toc182925055)

[3.2.4. **Kỹ thuật cắt tỉa cây (Pruning)** 14](#_Toc182925056)

[3.2.5. **Ưu và nhược điểm** 14](#_Toc182925057)

[3.2.6. **Ứng dụng thực tế** 14](#_Toc182925058)

[3.2.7. **Ví dụ trực quan** 15](#_Toc182925059)

[3.2.8. **Các bước thực hiện:** 15](#_Toc182925060)

[3.3. Thuật Toán Random Forest 16](#_Toc182925061)

[3.3.1. **Ý tưởng chính của Random Forest** 17](#_Toc182925062)

[3.3.2. **Cách hoạt động của Random Forest** 17](#_Toc182925063)

[3.3.3. **Tăng cường hiệu quả với Random Forest** 17](#_Toc182925064)

[3.3.4. **Ưu và nhược điểm** 18](#_Toc182925065)

[3.3.5. Ứng dụng thực tế 18](#_Toc182925066)

[3.3.6. **Ví dụ trực quan** 19](#_Toc182925067)

[3.3.7. **Các bước thực hiện:** 19](#_Toc182925068)

[3.4. So sánh và đánh giá mô hình 20](#_Toc182925069)

[3.4.1. So sánh 20](#_Toc182925070)

[3.4.2. Đánh Giá Mô Hình 22](#_Toc182925071)

[CHƯƠNG 4. KẾT QUẢ THỰC NGHIỆM 23](#_Toc182925072)

[4.1. Đối với code 23](#_Toc182925073)

[4.2 Đối với streamlit 25](#_Toc182925074)

[CHƯƠNG 5. KẾT LUẬN 26](#_Toc182925075)

[5.1. Kết quả đạt được 26](#_Toc182925076)

[5.2. Những khó khăn, hạn chế 26](#_Toc182925077)

[TÀI LIỆU THAM KHẢO 27](#_Toc182925078)

[PHỤ LỤC 28](#_Toc182925079)

# DANH MỤC HÌNH

[Hình 4.2.1 27](#_Toc182939163)

[Hình 4.2.2 28](#_Toc182939164)

[Hình 4.2.3 29](#_Toc182939165)

[Hình 4.2 4 29](#_Toc182939166)

[Hình 4.2.5 30](#_Toc182939167)

[Hình 4.2.6 30](#_Toc182939168)

[Hình 4.2.7 31](#_Toc182939169)

[Hình 4.2.8 32](#_Toc182939170)

# DANH MỤC BẢNG

[Bảng 1.2.1 Các mô hình khác 4](#_Toc182939259)

[Bảng 3.2.7 Ví dụ trực quan 15](#_Toc182939260)

[Bảng 3.3.6 Ví dụ Trực quan 19](#_Toc182939261)

[Bảng 3.4.1 So sánh 21](#_Toc182939262)

[Bảng 4.2.5 Ví Dụ 26](#_Toc182939263)

# GIỚI THIỆU MÔ HÌNH CHẨN ĐOÁN BỆNH VIÊM GAN C

## Giới thiệu bài toán

Trong những năm gần đây, bệnh viêm gan C đã trở thành một trong những mối quan tâm hàng đầu trong lĩnh vực y tế. Đây là bệnh truyền nhiễm do virus HCV gây ra, có khả năng lây nhiễm qua đường máu và gây ra các biến chứng nghiêm trọng cho gan, như xơ gan, ung thư gan và các bệnh lý liên quan khác. Phát hiện và chẩn đoán bệnh viêm gan C từ giai đoạn sớm có ý nghĩa quan trọng, giúp gia tăng cơ hội điều trị thành công và cải thiện chất lượng cuộc sống cho bệnh nhân. [1]

**Nhiệm vụ của mô hình chẩn đoán bệnh viên gan C**: Đề tài này tập trung vào việc xây dựng một mô hình dự đoán khả năng mắc bệnh viêm gan C dựa trên dữ liệu y tế của bệnh nhân, bao gồm các đặc trưng về sức khỏe như chỉ số men gan, tình trạng nhiễm trùng và các thông tin liên quan. Mô hình dự đoán sẽ ứng dụng các thuật toán máy học nhằm phân loại bệnh nhân thành hai nhóm: mắc bệnh và không mắc bệnh, dựa trên các đặc trưng của dữ liệu đầu vào.

**Mục tiêu cụ thể** của đề tài:

* Nghiên cứu và lựa chọn mô hình máy học phù hợp để dự đoán bệnh viêm gan C.
* Tiền xử lý dữ liệu, bao gồm làm sạch và chuẩn hóa dữ liệu.
* Đánh giá độ chính xác của mô hình, nhằm đảm bảo tính khả thi và hiệu quả khi ứng dụng trong thực tế.

## Các công trình liên quan

Nhiều nghiên cứu trước đây đã ứng dụng các thuật toán máy học vào bài toán dự đoán bệnh viêm gan C cũng như các bệnh lý khác. Dưới đây là một số mô hình phổ biến thường được sử dụng trong các công trình nghiên cứu tương tự:

* **Logistic Regression**: Một mô hình hồi quy được sử dụng phổ biến trong các bài toán phân loại nhị phân. Mô hình này dựa vào hàm logistic để dự đoán xác suất thuộc vào một trong hai nhóm.
* **Neural Network (Mạng nơ-ron)**: Mô hình học sâu (Deep Learning) có khả năng học các mẫu phức tạp trong dữ liệu, đặc biệt hiệu quả khi có lượng dữ liệu lớn.
* **K-Nearest Neighbors (KNN):** Thuật toán dựa trên việc tìm kiếm các điểm dữ liệu gần nhất (hàng xóm) để dự đoán lớp của một điểm mới. KNN hoạt động tốt với dữ liệu nhỏ và ít nhiễu nhưng có thể chậm khi dữ liệu lớn.
* **Naive Bayes:** Một phương pháp phân loại dựa trên định lý Bayes và giả định độc lập giữa các đặc trưng. Thuật toán này đặc biệt hiệu quả với dữ liệu dạng văn bản hoặc các bài toán mà mối quan hệ giữa các đặc trưng đơn giản.
* **Gradient Boosting:** Một kỹ thuật tăng cường mạnh mẽ, bao gồm các thuật toán như XGBoost và LightGBM, giúp tối ưu hóa hiệu suất dự đoán bằng cách kết hợp nhiều mô hình cây quyết định yếu (weak learners). Thuật toán này nổi bật trong việc xử lý dữ liệu phức tạp.

Bảng 1.2.1 Các mô hình khác

|  |  |  |
| --- | --- | --- |
| Mô hình | Đặc điểm | Ứng dụng trong sự đoán bệnh viêm gan C |
| Logistic  Regression | Dễ hiểu, hiệu quả với dữ liệu tuyến tính | Dự đoán nhanh, dùng làm baseline |
| Decision  Tree | Diễn giả dễ dàng, độ chính xác cao khi điều chỉnh | Thường dùng cho dữ liệu phân nhóm rõ ràng |
| Random Forest | Kết hợp nhiều cây quyết định, độ chính xác cao | Dữ đoán tốt, giảm nguy cơ overfitting |
| SVM | Phân chia tối ưu, hiệu quả với dữ liệu không gian lớn | Hiệu quả với dữ liệu có biên rõ |
| Neural  Network | Phức tạp, hiệu quả với lượng dữ liệu lớn | Khả năng dự đoán cao nhưng cần nhiều tài nguyên. |

# CHUẨN BỊ DỮ LIỆU

## Giới thiệu tập dữ liệu

Tập dữ liệu được sử dụng trong nghiên cứu này được thiết kế để phân tích và đánh giá sức khỏe gan dựa trên các chỉ số sinh hóa của bệnh nhân. Các chỉ số này bao gồm:

* **Age (Tuổi)**: Độ tuổi của bệnh nhân, một yếu tố có thể liên quan đến nguy cơ và tiến triển của các bệnh về gan.
* **Sex (Giới tính)**: Nam hoặc nữ. Giới tính có thể ảnh hưởng đến các bệnh gan do sự khác biệt trong cấu trúc sinh học và hành vi.
* **ALB (Albumin)**: Một loại protein quan trọng được sản xuất bởi gan, giúp duy trì áp lực keo của máu và vận chuyển các chất như hormone và thuốc. Mức albumin thấp có thể chỉ ra suy giảm chức năng gan.
* **ALP (Alkaline Phosphatase)**: Enzyme liên quan đến gan và xương, có thể tăng cao khi có các vấn đề về gan, đặc biệt là bệnh viêm gan hoặc tắc mật.
* **AST (Aspartate Aminotransferase)**: Enzyme chỉ thị tổn thương gan hoặc tim.
* **ALT (Alanine Aminotransferase)** và **AST (Aspartate Aminotransferase)**: Các enzyme có mặt trong gan. Khi tế bào gan bị tổn thương, các enzyme này sẽ rò rỉ vào máu, nên nồng độ cao có thể cho thấy tổn thương gan.
* **CHE (Cholinesterase)**: Enzyme do gan sản xuất, giảm khi chức năng gan suy giảm.
* **CHOL (Cholesterol)**: Mức cholesterol trong máu, liên quan đến tổng hợp tại gan.
* **BIL (Bilirubin)**: Một sản phẩm phân hủy của hemoglobin. Mức bilirubin cao trong máu có thể là dấu hiệu của tổn thương gan.
* **CREA (Creatinine)**: Đánh giá chức năng thận nhưng cũng có liên quan đến sức khỏe tổng thể.
* **GGT (Gamma-glutamyl Transferase)**: Enzyme giúp đánh giá mức độ tổn thương gan, thường được sử dụng để phát hiện các vấn đề về gan do rượu.
* **PROT (Protein tổng số)**: Tổng lượng protein trong máu, phản ánh sức khỏe gan và dinh dưỡng.

Việc thu thập và hiểu rõ về các chỉ số sinh hóa này giúp cung cấp dữ liệu quan trọng cho mô hình, từ đó hỗ trợ trong việc phát hiện và đánh giá mức độ của các bệnh về gan.

## Tiền xử lý và trực quan hóa dữ liệu

Dữ liệu thu thập từ thực tế thường chứa các giá trị bị thiếu, ngoại lai hoặc không đồng nhất. Vì vậy, bước tiền xử lý dữ liệu đóng vai trò quan trọng nhằm làm sạch và chuẩn bị dữ liệu tốt nhất cho mô hình.

**2.2.1. Làm sạch dữ liệu**

Làm sạch dữ liệu là bước đầu tiên, bao gồm việc xử lý giá trị bị thiếu và giá trị ngoại lai:

* **Giá trị thiếu**: Khi thu thập dữ liệu, một số trường thông tin có thể bị bỏ sót. Các phương pháp xử lý phổ biến bao gồm thay thế giá trị thiếu bằng trung bình, giá trị trung vị, hoặc loại bỏ bản ghi chứa giá trị thiếu.
* **Giá trị ngoại lai**: Là những giá trị nằm ngoài phạm vi thông thường của dữ liệu và có thể gây sai lệch trong mô hình. Các phương pháp phát hiện và xử lý giá trị ngoại lai bao gồm việc sử dụng phương pháp Z-score hoặc khoảng tứ phân vị (IQR).

**2.2.2. Chuẩn hóa và Chuẩn hóa dữ liệu**

Để đưa các đặc trưng về cùng một phạm vi giá trị, quá trình chuẩn hóa hoặc chuẩn hóa dữ liệu là cần thiết:

* **Chuẩn hóa (Normalization)**: Thường được sử dụng khi dữ liệu không có phân phối chuẩn. Quá trình này chuyển đổi dữ liệu về khoảng [0, 1], giúp mô hình học máy hoạt động hiệu quả hơn.
* **Chuẩn hóa (Standardization)**: Được sử dụng khi dữ liệu có phân phối chuẩn. Quá trình này chuyển đổi dữ liệu thành phân phối chuẩn với trung bình là 0 và độ lệch chuẩn là 1, giúp tăng tính ổn định của mô hình.

**2.2.3. Trực quan hóa dữ liệu**

Trực quan hóa dữ liệu giúp phát hiện các mẫu tiềm ẩn và mối liên hệ giữa các đặc trưng. Một số công cụ phổ biến trong trực quan hóa dữ liệu:

* **Biểu đồ phân phối (Histogram)**: Giúp quan sát phân bố của từng chỉ số sinh hóa, xác định tính đối xứng và sự hiện diện của giá trị ngoại lai.
* **Biểu đồ Boxplot**: Là công cụ mạnh mẽ để phát hiện các giá trị ngoại lai trong dữ liệu.
* **Bản đồ nhiệt (Heatmap)**: Giúp phân tích tương quan giữa các đặc trưng, từ đó xác định các đặc trưng có mối quan hệ mạnh mẽ với biến mục tiêu và có thể hữu ích trong dự đoán.

## Trích chọn đặc trưng

Các bước trích chọn đặc trưng trong đoạn code của nhóm:

1. **Chuyển đổi dữ liệu định tính thành định lượng**:
   * Cột Category được mã hóa từ dạng chuỗi thành số, ví dụ: 'Blood Donor' → 0, 'Hepatitis' → 2.
   * Cột Sex được chuyển thành boolean, ví dụ: 'M' → 0, 'F' → 1.
2. **Xử lý giá trị bị thiếu (missing values)**:
   * Sử dụng giá trị trung bình của cột để điền vào các giá trị NaN (data.fillna(data.mean(), inplace=True)).
3. **Chuẩn hóa các đặc trưng**:
   * Sử dụng StandardScaler để chuẩn hóa tất cả các cột trong tập dữ liệu X, đưa các đặc trưng về cùng một thang đo có trung bình là 0 và phương sai là 1 (scaler.fit\_transform(X)).

* **Chuẩn hóa** giúp các thuật toán như Ramdom Forest, SVM và Cây Quyết Định hoạt động tốt hơn, đặc biệt khi dữ liệu có các đặc trưng có đơn vị đo lường khác nhau hoặc phân phối khác biệt.

# XÂY DỰNG MÔ HÌNH

Trong chương này, chúng ta sẽ tiến hành xây dựng và đánh giá ba mô hình phân loại sử dụng ba thuật toán học máy phổ biến: **SVM (Support Vector Machine)**, **Cây Quyết Định (Decision Tree)** và **Ramdom Forest** . Mỗi mô hình sẽ được huấn luyện trên bộ dữ liệu huấn luyện, sau đó đánh giá độ chính xác và các chỉ số khác trên bộ dữ liệu kiểm tra và huấn luyện.

## Thuật Toán SVM (****Support Vector Machine)****

Support Vector Machine (SVM) là một thuật toán học máy mạnh mẽ và hiệu quả, được sử dụng phổ biến trong các bài toán phân loại và hồi quy. Thuật toán hoạt động bằng cách tìm siêu phẳng (hyperplane) tối ưu để phân chia các lớp dữ liệu trong không gian đặc trưng. [1]

### **Ý tưởng chính của SVM**

SVM tập trung vào việc tối đa hóa khoảng cách lề (margin) giữa siêu phẳng phân chia và các điểm dữ liệu gần nhất từ mỗi lớp, gọi là Support Vectors.

💡 Ví dụ đơn giản:

- Trong không gian 2D, siêu phẳng là một đường thẳng.

- Trong không gian 3D, siêu phẳng là một mặt phẳng.

- Trong không gian nhiều chiều, siêu phẳng là một không gian con có chiều thấp hơn.

### **Siêu phẳng và khoảng cách lề (Margin)**

- Siêu phẳng (Hyperplane):Là đường/mặt phân chia dữ liệu giữa các lớp.

Công thức của siêu phẳng:



Trong đó:

- w ;Vector trọng số.

- x: Vector đặc trưng.

- b: Hệ số chặn.

- Khoảng cách lề (Margin): Là khoảng cách giữa siêu phẳng và các điểm gần nhất thuộc hai lớp (Support Vectors).

### **Hạt nhân (Kernel Trick)**

Khi dữ liệu không thể phân tách tuyến tính, SVM sử dụng Kernel Trick để ánh xạ dữ liệu từ không gian đầu vào sang không gian đặc trưng cao hơn, nơi nó có thể được phân tách tuyến tính.

Các loại kernel phổ biến:

- Linear Kernel: Dùng khi dữ liệu có thể phân tách tuyến tính.

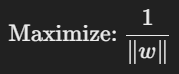
- Polynomial Kernel:Dùng để xử lý các quan hệ phi tuyến tính bậc cao.

- RBF (Radial Basis Function) Kernel: Hiệu quả với dữ liệu phi tuyến tính.

### **Hàm mục tiêu của SVM**

SVM giải bài toán tối ưu hóa để tìm siêu phẳng tối ưu:

1. Tối đa hóa lề (Margin):



2. \*\*Đảm bảo phân tách đúng các lớp:



Trong đó :

Yi là nhãn của điểm dữ liệu Xi

### **Ưu và nhược điểm**

🌟 Ưu điểm:

- Hiệu quả cao: Đặc biệt với dữ liệu nhỏ nhưng phức tạp.

- Tính tổng quát tốt: Giảm nguy cơ quá khớp (overfitting) nhờ tối đa hóa lề.

- Linh hoạt: Hỗ trợ nhiều loại kernel để xử lý dữ liệu tuyến tính và phi tuyến tính.

⚠️ Nhược điểm:

- Chi phí tính toán cao: Khi kích thước dữ liệu lớn.

- Khó chọn kernel:Kernel không phù hợp có thể làm giảm hiệu suất mô hình.

6. Ứng dụng thực tế

- Phân loại: Nhận diện khuôn mặt, phát hiện thư rác, phân tích y tế (ví dụ: chẩn đoán bệnh).

- Hồi quy: Dự đoán giá nhà, dự đoán thời gian hoàn thành dự án.

- Hệ thống hỗ trợ quyết định: Quản lý rủi ro tài chính, tối ưu hóa vận hành.

### **Ví dụ trực quan**

Bài toán phân loại: Thư rác (Spam vs Non-Spam)

- Dữ liệu:

+Email 1: Nội dung ngắn, không chứa từ khóa nghi ngờ → Không phải spam.

+Email 2: Nội dung dài, chứa nhiều từ khóa nghi ngờ → Spam.

- SVM hoạt động như thế nào?

+Biểu diễn email dưới dạng các đặc trưng số học (ví dụ: số từ khóa nghi ngờ, độ dài email).

+ Tìm siêu phẳng tối ưu để phân chia hai lớp: Spam và Non-Spam.

### **So sánh với thuật toán khác**

- So với Decision Tree:

- SVM hiệu quả hơn trong các bài toán có biên phân chia phức tạp hoặc phi tuyến tính.

- Decision Tree dễ hiểu và trực quan hơn nhưng dễ bị overfitting nếu không được cắt tỉa đúng cách.

### **Các bước thực hiện:**

1. **Khởi tạo mô hình SVM**: Chúng ta sử dụng SVC từ thư viện sklearn.svm với kernel tuyến tính.
2. **Huấn luyện mô hình**: Dữ liệu huấn luyện sẽ được đưa vào mô hình để học các đặc trưng và phân loại dữ liệu.
3. **Dự đoán trên tập kiểm tra**: Sau khi mô hình đã được huấn luyện, chúng ta sẽ kiểm tra độ chính xác của mô hình bằng cách dự đoán trên tập kiểm tra.
4. **Đánh giá mô hình**: Các chỉ số đánh giá bao gồm độ chính xác (accuracy), ma trận nhầm lẫn (confusion matrix) và báo cáo phân loại (classification report).

**Xây dựng mô hình:**

# thuật toán SVM

from sklearn.svm import SVC

# Khởi tạo mô hình SVM

svm\_model = SVC(kernel='linear', random\_state=42)  # Sử dụng kernel tuyến tính

# Huấn luyện mô hình trên tập huấn luyện

svm\_model.fit(X\_train, y\_train)

# Dự đoán trên tập huấn luyện

y\_pred\_train\_svm = svm\_model.predict(X\_train)

# Đánh giá mô hình trên tập huấn luyện

accuracy\_train\_svm = accuracy\_score(y\_train, y\_pred\_train\_svm)

confusion\_mat\_train\_svm = confusion\_matrix(y\_train, y\_pred\_train\_svm)

classification\_rep\_train\_svm = classification\_report(y\_train, y\_pred\_train\_svm)

print("\nSVM (Train) - Accuracy (%):", accuracy\_train\_svm \* 100,"%")

print("SVM (Train) - Confusion Matrix:\n", confusion\_mat\_train\_svm)

print("SVM (Train) - Classification Report:\n",

# Huấn luyện mô hình trên các tập

svm\_model.fit(X\_train, y\_train)

# Dự đoán trên tập kiểm tra

y\_pred\_svm = svm\_model.predict(X\_test)

# Đánh giá mô hình

accuracy\_svm = accuracy\_score(y\_test, y\_pred\_svm)

confusion\_mat\_svm = confusion\_matrix(y\_test, y\_pred\_svm)

classification\_rep\_svm = classification\_report(y\_test, y\_pred\_svm)

print("\nSVM (test)- Accuracy(%):", accuracy\_svm \* 100,"%")

print("SVM (test)- Confusion Matrix:\n", confusion\_mat\_svm)

print("SVM (test)- Classification Report:\n", classification\_rep\_svm)

## Thuật toán Cây Quyết Định (Decision Tree)

Cây Quyết Định (Decision Tree) Là một thuật toán học máy dễ hiểu, mạnh mẽ, được sử dụng rộng rãi trong phân loại và hồi quy. Thuật toán hoạt động thông qua cấu trúc cây để ra quyết định dựa trên các điều kiện cụ thể. [2]

### **Cấu trúc của cây quyết định**

Một cây quyết định bao gồm:

- Nút gốc (Root Node): Điểm bắt đầu của cây, đại diện cho toàn bộ tập dữ liệu.

- Nút trung gian (Internal Node): Các điểm phân chia dữ liệu dựa trên một điều kiện.

- Nút lá (Leaf Node): Kết quả cuối cùng, chứa phân lớp hoặc giá trị dự đoán.

💡 Ví dụ:

- Nút gốc: "Thời tiết hôm nay có mưa không?"

- Nút trung gian: "Độ ẩm cao hay thấp?"

- Nút lá: "Có chơi bóng đá hay không?"

### **Quy trình xây dựng cây quyết định**

1. Chọn đặc trưng quan trọng nhất:

Sử dụng tiêu chí như Entropy, Gain thông tin, hoặc chỉ số Gini để xác định đặc trưng phân tách dữ liệu tốt nhất.

2. Phân chia dữ liệu:

Chia dữ liệu thành các nhóm con dựa trên điều kiện tại nút.

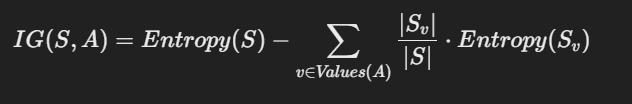
3. Tiếp tục lặp lại:

Thực hiện đệ quy trên các nhóm con, xây dựng các nhánh cho đến khi đạt điều kiện dừng (dữ liệu thuần nhất hoặc không còn đặc trưng để phân chia).

### **Tiêu chí chọn đặc trưng**

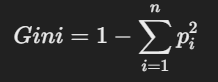
- Entropy và Gain thông tin (Information Gain):

Giảm độ hỗn loạn (mức độ không thuần nhất) trong dữ liệu.



- Chỉ số Gini (Gini Index):

Đo lường độ thuần nhất của các nhóm con.



- Giảm phương sai (Variance Reduction):

Thường áp dụng trong bài toán hồi quy để đo độ giảm phương sai.

### **Kỹ thuật cắt tỉa cây (Pruning)**

Để tránh quá khớp (overfitting):

- Cắt tỉa trước (Pre-pruning): Ngừng mở rộng cây khi đạt độ sâu tối đa hoặc số lượng dữ liệu tại nút quá nhỏ.

- Cắt tỉa sau (Post-pruning): Loại bỏ các nhánh không cần thiết sau khi cây được xây dựng hoàn chỉnh.

### **Ưu và nhược điểm**

🌟 Ưu điểm:

- Dễ hiểu: Quy trình ra quyết định đơn giản, trực quan.

- Không cần chuẩn hóa: Không yêu cầu chuẩn hóa dữ liệu hoặc xử lý giá trị thiếu.

- Linh hoạt: Hỗ trợ cả đặc trưng số và danh mục.

⚠️ Nhược điểm:

- Dễ bị overfitting: Nếu cây quá phức tạp.

- Kém ổn định: Cây có thể thay đổi mạnh nếu dữ liệu thay đổi nhỏ.

### **Ứng dụng thực tế**

- Phân loại: Chẩn đoán y tế, nhận diện khách hàng trung thành.

- Hồi quy: Dự đoán giá bất động sản, dự đoán doanh thu.

- Ra quyết định: Tối ưu hóa quy trình quản lý, hỗ trợ marketing.

### **Ví dụ trực quan**

Bài toán: Có nên chơi bóng đá?

Dựa trên các điều kiện thời tiết:

Bảng 3.2.7 Ví dụ trực quan

|  |  |  |
| --- | --- | --- |
| Thời tiết | Độ ẩm | Kết quả |
| Có mưa | - | Không chơi |
| Không mưa | Cao | Không chơi |
| Không mưa | Thấp | Chơi |

Cây quyết định:

- Nút gốc: Thời tiết có mưa không?

- Nếu Có mưa: Không chơi.

- Nếu Không mưa:

- Độ ẩm cao: Không chơi.

- Độ ẩm thấp: Chơi.

### **Các bước thực hiện:**

1. **Khởi tạo mô hình Cây Quyết Định**: Sử dụng DecisionTreeClassifier từ thư viện sklearn.tree.
2. **Huấn luyện mô hình**: Dữ liệu huấn luyện sẽ được sử dụng để xây dựng cây quyết định.
3. **Dự đoán trên tập kiểm tra**: Dự đoán sẽ được thực hiện và đánh giá trên tập kiểm tra.
4. **Đánh giá mô hình**: Các chỉ số đánh giá tương tự như SVM.

# Khởi tạo mô hình cây quyết định

decision\_tree\_model = DecisionTreeClassifier(random\_state=42)

# Huấn luyện mô hình trên tập huấn luyện

decision\_tree\_model.fit(X\_train, y\_train)

# Dự đoán trên tập huấn luyện

y\_pred\_train\_dt = decision\_tree\_model.predict(X\_train)

# Đánh giá mô hình trên tập huấn luyện

accuracy\_train\_dt = accuracy\_score(y\_train, y\_pred\_train\_dt)

confusion\_mat\_train\_dt = confusion\_matrix(y\_train, y\_pred\_train\_dt)

classification\_rep\_train\_dt = classification\_report(y\_train, y\_pred\_train\_dt)

print("\nCây Quyết Định (Train) - Accuracy (%):", accuracy\_train\_dt \* 100,"%")

print("Cây Quyết Định (Train) - Confusion Matrix:\n", confusion\_mat\_train\_dt)

print("Cây Quyết Định (Train) - Classification Report:\n", classification\_rep\_train\_dt)

# Dự đoán trên tập kiểm tra

y\_pred\_dt = decision\_tree\_model.predict(X\_test)

# Đánh giá mô hình

accuracy\_dt = accuracy\_score(y\_test, y\_pred\_dt)

confusion\_mat\_dt = confusion\_matrix(y\_test, y\_pred\_dt)

classification\_rep\_dt = classification\_report(y\_test, y\_pred\_dt)

print("\nCây Quyết Định (test)- Accuracy(%):", accuracy\_dt \* 100,"%")

print("Cây Quyết Định (test) - Confusion Matrix:\n", confusion\_mat\_dt)

print("Cây Quyết Định (test) - Classification Report:\n", classification\_rep\_dt)

## Thuật Toán Random Forest

Random Forest là một thuật toán học máy mạnh mẽ dựa trên ensemble learning, được sử dụng phổ biến trong cả phân loại và hồi quy. Thuật toán kết hợp nhiều Cây Quyết Định (Decision Trees) để đưa ra dự đoán chính xác và ổn định hơn.

### **Ý tưởng chính của Random Forest**

- Random Forest xây dựng một tập hợp các cây quyết định (Decision Trees) độc lập.

- Mỗi cây được huấn luyện trên một tập con dữ liệu khác nhau và đưa ra dự đoán riêng.

- Kết quả cuối cùng được tính bằng:

- Phân loại: Đa số phiếu (Majority Vote).

- Hồi quy: Trung bình của các dự đoán từ các cây.

💡 Ví dụ trực quan:

- Trong một cuộc bầu cử, mỗi cử tri là một cây quyết định. Kết quả bầu cử là ý kiến của đa số cử tri (majority vote).

### **Cách hoạt động của Random Forest**

Bước 1: Tạo tập dữ liệu con (Bootstrap Sampling)

- Chọn ngẫu nhiên các mẫu từ tập dữ liệu gốc (có thể lặp lại).

- Mỗi cây quyết định được huấn luyện trên một tập dữ liệu con khác nhau.

Bước 2: Xây dựng cây quyết định

- Tại mỗi nút, chọn một tập hợp ngẫu nhiên các đặc trưng.

- Chỉ các đặc trưng này mới được xem xét để phân tách dữ liệu (giảm tương quan giữa các cây).

Bước 3: Tổng hợp kết quả

- Phân loại: Kết quả được xác định bởi lớp được dự đoán nhiều nhất (đa số phiếu).

- Hồi quy: Kết quả là trung bình các giá trị dự đoán từ các cây.

### **Tăng cường hiệu quả với Random Forest**

- Ngẫu nhiên hóa trong Random Forest:

1. Bootstrap Sampling: Giúp các cây quyết định không bị quá khớp vào toàn bộ dữ liệu.

2. Random Feature Selection: Tạo sự đa dạng giữa các cây bằng cách sử dụng các tập đặc trưng khác nhau.

- Overfitting giảm nhờ đa số phiếu:

+ Một cây quyết định có thể bị overfitting, nhưng kết hợp nhiều cây giúp giảm thiểu lỗi.

### **Ưu và nhược điểm**

🌟 Ưu điểm:

- Độ chính xác cao:Hoạt động tốt trên cả dữ liệu tuyến tính và phi tuyến tính.

- Giảm overfitting: Nhờ cơ chế bootstrap và random feature selection.

- Linh hoạt: Ứng dụng được cho cả phân loại và hồi quy.

⚠️ Nhược điểm:

- Tốn tài nguyên: Yêu cầu nhiều bộ nhớ và thời gian tính toán khi số lượng cây lớn.

- Khó giải thích:Không trực quan như một cây quyết định đơn lẻ.

### Ứng dụng thực tế

- Phân loại:

+ Chẩn đoán y tế (ví dụ: phát hiện ung thư).

+ Phát hiện gian lận trong giao dịch tài chính.

- Hồi quy:

+ Dự đoán giá bất động sản.

+ Phân tích rủi ro bảo hiểm.

### **Ví dụ trực quan**

Bài toán: Có nên đi du lịch không?

Dữ liệu:

Bảng 3.3.6 Ví dụ Trực quan

|  |  |  |  |
| --- | --- | --- | --- |
| Thời tiết | Ngân sách | Khoảng cách | Quyết định |
| Nắng | Cao | Xa | Đi |
| Mưa | Thấp | Gần | Không |
| Nắng | Trung bình | Gần | Đi |

Cách Random Forest hoạt động:

1. Mỗi cây quyết định được xây dựng trên các tập con ngẫu nhiên của dữ liệu.

2. Mỗi cây sẽ đưa ra một dự đoán riêng:

- Cây 1: "Đi".

- Cây 2: "Không đi".

- Cây 3: "Đi".

3. Kết quả cuối cùng: "Đi" (đa số phiếu).

### **Các bước thực hiện:**

1. **Khởi tạo mô hình Random Forest:** Sử dụng RandomForestClassifier từ thư viện sklearn.ensemble.
2. **Huấn luyện mô hình:** Mô hình được huấn luyện bằng cách sử dụng tập dữ liệu huấn luyện để tạo ra nhiều cây quyết định.
3. **Dự đoán trên tập kiểm tra:** Mô hình sử dụng kết quả tổng hợp (bằng trung bình hoặc bỏ phiếu đa số) để dự đoán trên tập kiểm tra.
4. **Đánh giá mô hình:** Các chỉ số đánh giá tương tự như SVM.

# Khởi tạo mô hình Random Forest

random\_forest\_model = RandomForestClassifier(random\_state=42)

# Huấn luyện mô hình trên tập huấn luyện

random\_forest\_model.fit(X\_train, y\_train)

# Dự đoán trên tập huấn luyện

y\_pred\_train\_rf = random\_forest\_model.predict(X\_train)

# Đánh giá mô hình trên tập huấn luyện

accuracy\_train\_rf = accuracy\_score(y\_train, y\_pred\_train\_rf)

confusion\_mat\_train\_rf = confusion\_matrix(y\_train, y\_pred\_train\_rf)

classification\_rep\_train\_rf = classification\_report(y\_train, y\_pred\_train\_rf)

print("\nRandom Forest (Train) - Accuracy (%):", accuracy\_train\_rf \* 100,"%")

print("Random Forest (Train) - Confusion Matrix:\n", confusion\_mat\_train\_rf)

print("Random Forest (Train) - Classification Report:\n", classification\_rep\_train\_rf)

# Dự đoán trên tập kiểm tra

y\_pred\_rf = random\_forest\_model.predict(X\_test)

# Đánh giá mô hình

accuracy\_rf = accuracy\_score(y\_test, y\_pred\_rf)

confusion\_mat\_rf = confusion\_matrix(y\_test, y\_pred\_rf)

classification\_rep\_rf = classification\_report(y\_test, y\_pred\_rf)

print("\nRandom Forest (test)- Accuracy(%):", accuracy\_rf \* 100,"%")

print("Random Forest (test) - Confusion Matrix:\n", confusion\_mat\_rf)

print("Random Forest (test) - Classification Report:\n", classification\_rep\_rf)

## So sánh và đánh giá mô hình

### So sánh

Bảng 3.4.1 So sánh

|  |  |  |  |
| --- | --- | --- | --- |
| Tiêu chí | Decision Tree | Random Forest | SVM (Support Vector Machine) |
| Dữ liệu yêu cầu | Dữ liệu không cần chuẩn hóa, trực quan. | Cần nhiều dữ liệu để xây dựng tập hợp cây mạnh mẽ. | Dữ liệu cần được chuẩn hóa để tăng hiệu quả. |
| Tốc độ tính toán | Nhanh với dữ liệu nhỏ, nhưng chậm khi cây sâu hoặc phức tạp. | Chậm hơn do xây dựng nhiều cây, nhưng song song hóa tốt. | Chậm khi làm việc với dữ liệu lớn hoặc số chiều cao. |
| Khả năng xử lý phi tuyến | Hạn chế, dễ bị overfitting nếu không giới hạn độ sâu | Tốt nhờ sự kết hợp từ nhiều cây ngẫu nhiên. | Xuất sắc nhờ kernel trick. |
| Khả năng tổng quát hóa | Dễ bị overfitting với dữ liệu phức tạp. | Tổng quát hóa tốt, giảm overfitting nhờ cơ chế bootstrap và random feature selection. | Tổng quát hóa tốt với siêu phẳng tối ưu. |
| Ưu điểm nổi bật | - Đơn giản, dễ hiểu, trực quan.  - Không yêu cầu chuẩn hóa dữ liệu. | - Giảm overfitting.  - Hiệu quả trên cả phân loại và hồi quy. | - Hiệu quả với dữ liệu có biên phức tạp.  -Tốt với không gian nhiều chiều. |
| Nhược điểm | - Dễ overfitting nếu không kiểm soát độ sâu.  - Nhạy cảm với dữ liệu nhiễu. | - Tốn tài nguyên (bộ nhớ và thời gian).  - Khó giải thích hơn. | - Khó chọn kernel phù hợp.  - Tốn thời gian tính toán cho dữ liệu lớn. |
| Ứng dụng chính | - Phân loại đơn giản.  - Phân tích ra quyết định. | - Phân loại phức tạp (như y tế, tài chính).  - Hồi quy (dự đoán giá, phân tích rủi ro). | - Nhận diện hình ảnh.  - Phân tích văn bản.  - Chẩn đoán y tế. |

* + - **Kết luận**
* **Decision Tree:** Dễ hiểu, phù hợp cho bài toán đơn giản hoặc cần trực quan.
* **Random Forest:** Lựa chọn tốt cho các bài toán thực tế, cần độ chính xác cao và khả năng tổng quát hóa tốt.
* **SVM:** Hiệu quả trong các bài toán phức tạp, đặc biệt là khi cần phân chia phi tuyến hoặc không gian nhiều chiều.

### Đánh Giá Mô Hình

Sau khi xây dựng và đánh giá các mô hình trên, chúng ta sẽ so sánh độ chính xác, ma trận nhầm lẫn và các chỉ số khác để xác định mô hình phân loại tốt nhất cho bài toán của mình.

* **Độ chính xác**: Độ chính xác cho biết mô hình phân loại có thể dự đoán chính xác bao nhiêu phần trăm trên tập dữ liệu kiểm tra.
* **Ma trận nhầm lẫn**: Chỉ ra số lượng dự đoán đúng và sai cho mỗi lớp.
* **Báo cáo phân loại**: Cung cấp thông tin chi tiết về độ chính xác (Precision), độ phủ (Recall) và điểm F1 cho mỗi lớp.

# KẾT QUẢ THỰC NGHIỆM

## Đối với code

* Trường hợp 1 Support Vector Machine (SVM):

Đối với tập Train

SVM (Train) - Accuracy (%): 98.57723577235772 %

SVM (Train) - Confusion Matrix:

[[437 0 0 0 0]

[ 0 4 0 0 0]

[ 0 0 12 3 0]

[ 0 0 2 13 0]

[ 0 0 0 2 19]]

SVM (Train) - Classification Report:

precision recall f1-score support

0 1.00 1.00 1.00 437

1 1.00 1.00 1.00 4

2 0.86 0.80 0.83 15

3 0.72 0.87 0.79 15

4 1.00 0.90 0.95 21

accuracy 0.99 492

macro avg 0.92 0.91 0.91 492

weighted avg 0.99 0.99 0.99 492

Đối với tập Test

SVM (test)- Accuracy(%): 90.2439024390244 %

SVM (test)- Confusion Matrix:

[[96 0 0 0 0]

[ 0 2 0 1 0]

[ 0 0 6 3 0]

[ 1 0 1 4 0]

[ 1 1 3 1 3]]

SVM (test)- Classification Report:

precision recall f1-score support

0 0.98 1.00 0.99 96

1 0.67 0.67 0.67 3

2 0.60 0.67 0.63 9

3 0.44 0.67 0.53 6

4 1.00 0.33 0.50 9

accuracy 0.90 123

macro avg 0.74 0.67 0.66 123

weighted avg 0.92 0.90 0.90 123

* Trường hợp 2 DecisionTree:

Đối với tâp Train

Cây Quyết Định (Train) - Accuracy (%): 100.0 %

Cây Quyết Định (Train) - Confusion Matrix:

[[437 0 0 0 0]

[ 0 4 0 0 0]

[ 0 0 15 0 0]

[ 0 0 0 15 0]

[ 0 0 0 0 21]]

Cây Quyết Định (Train) - Classification Report:

precision recall f1-score support

0 1.00 1.00 1.00 437

1 1.00 1.00 1.00 4

2 1.00 1.00 1.00 15

3 1.00 1.00 1.00 15

4 1.00 1.00 1.00 21

accuracy 1.00 492

macro avg 1.00 1.00 1.00 492

weighted avg 1.00 1.00 1.00 492

Đối với Test

Cây Quyết Định (test)- Accuracy(%): 99.1869918699187 %

Cây Quyết Định (test) - Confusion Matrix:

[[96 0 0 0 0]

[ 0 3 0 0 0]

[ 0 0 9 0 0]

[ 0 0 0 6 0]

[ 0 0 0 1 8]]

Cây Quyết Định (test) - Classification Report:

precision recall f1-score support

0 1.00 1.00 1.00 96

1 1.00 1.00 1.00 3

2 1.00 1.00 1.00 9

3 0.86 1.00 0.92 6

4 1.00 0.89 0.94 9

accuracy 0.99 123

macro avg 0.97 0.98 0.97 123

weighted avg 0.99 0.99 0.99 123

* Trường hợp 3 Random Forest

Đối với tâp train

Random Forest (Train) - Accuracy (%): 100.0 %

Random Forest (Train) - Confusion Matrix:

[[437 0 0 0 0]

[ 0 4 0 0 0]

[ 0 0 15 0 0]

[ 0 0 0 15 0]

[ 0 0 0 0 21]]

Random Forest (Train) - Classification Report:

precision recall f1-score support

0 1.00 1.00 1.00 437

1 1.00 1.00 1.00 4

2 1.00 1.00 1.00 15

3 1.00 1.00 1.00 15

4 1.00 1.00 1.00 21

accuracy 1.00 492

macro avg 1.00 1.00 1.00 492

weighted avg 1.00 1.00 1.00 492

Đối với tập test

Random Forest (test)- Accuracy(%): 95.1219512195122 %

Random Forest (test) - Confusion Matrix:

[[96 0 0 0 0]

[ 1 1 1 0 0]

[ 0 0 8 1 0]

[ 0 0 1 5 0]

[ 0 0 0 2 7]]

Random Forest (test) - Classification Report:

precision recall f1-score support

0 0.99 1.00 0.99 96

1 1.00 0.33 0.50 3

2 0.80 0.89 0.84 9

3 0.62 0.83 0.71 6

4 1.00 0.78 0.88 9

accuracy 0.95 123

macro avg 0.88 0.77 0.79 123

weighted avg 0.96 0.95 0.95 123

4.2 Đối với streamlit

Ta sử dụng 1 vài dữ liệu bất kì trong Dataset

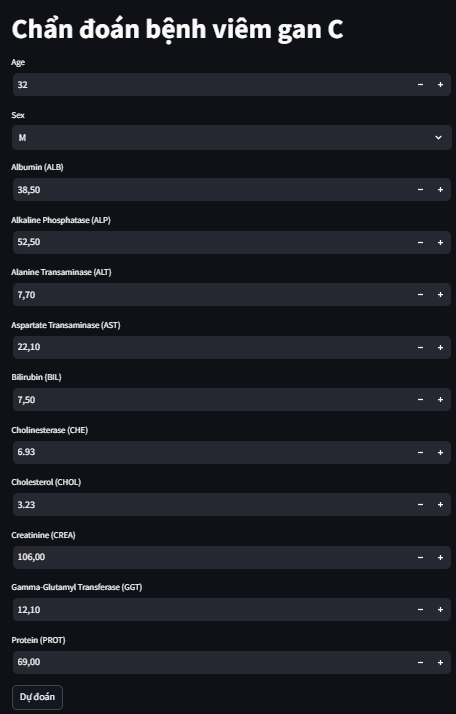
Ví dụ :

Bảng 4.2.5 Ví Dụ

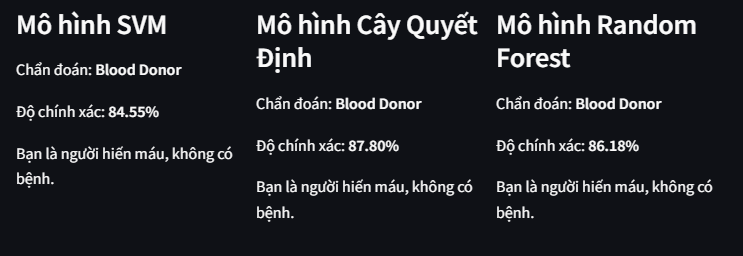
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Category | Age | Sex | ALB | ALP | ALT | AST | BIL | CHE | CHOL | CREA | GGT | PROT |
| 1 | Blood Donor | 32 | m | 38.5 | 52.5 | 7.7 | 22.1 | 7.5 | 6.93 | 3.23 | 106 | 12.1 | 69 |
| 563 | Hepatitis | 50 | f | 40 | 32.7 | 9 | 46 | 10 | 7.51 | 4.67 | 56.6 | 22.3 | 70.1 |
| 605 | Cirrhosis | 74 | m | 23 | 34.1 | 2.1 | 90.4 | 22 | 2.5 | 3.29 | 51 | 46.8 | 57.1 |
| 585 | Fibrosis | 75 | f | 36 | NA | 114 | 125 | 14 | 6.65 | NA | 57 | 177 | 72 |

Kết quả dự đoán với Streamlit:

Với dữ liệu thứ nhất cho ra được:

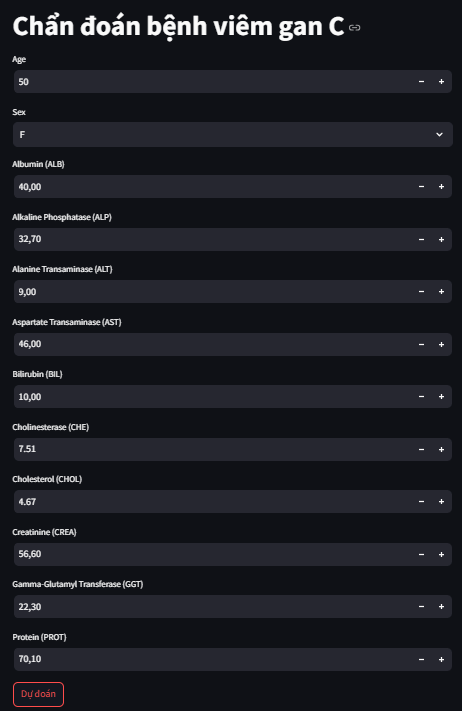


Hình 4.2.1

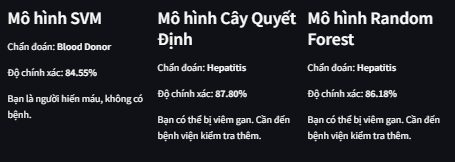


Hình 4.2.2

Với dữ liệu thứ 563 cho ra được:

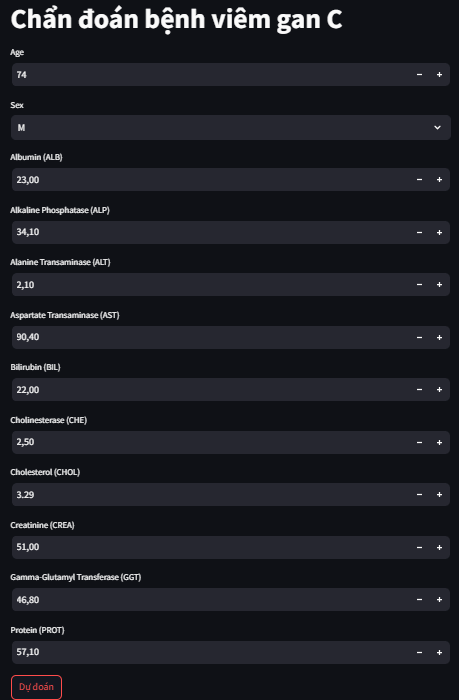


Hình 4.2.3

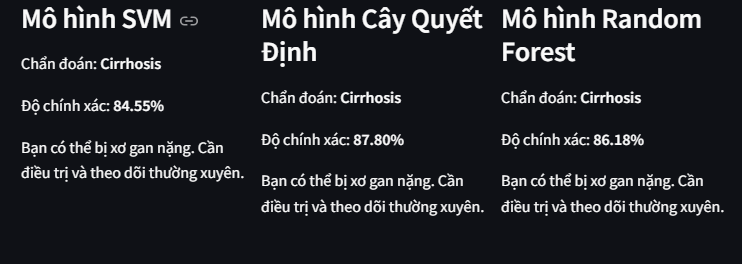


Hình 4.2 4

Với dữ liệu thứ 605 cho ra được:

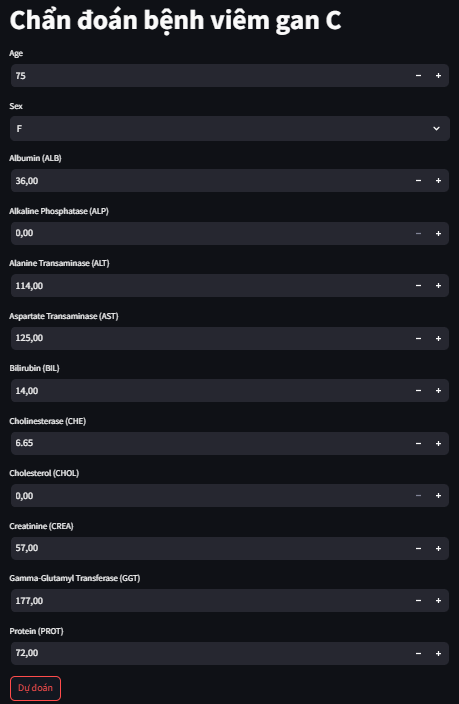


Hình 4.2.5

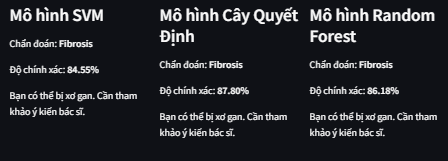


Hình 4.2.6

Với dữ liệu thứ 585 cho ra được:



Hình 4.2.7



Hình 4.2.8

# KẾT LUẬN

## Kết quả đạt được

 **Mục tiêu đã đạt được**: xử lý được bộ dữ liệu ban đầu đã thu thập được ,chia tập dữ liệu thành công với tập train và tập test (80 / 20) và chạy hiệu quả với ba thuật toán khác nhau đã xây dựng được trên bộ dữ liệu đó với độ chính xác lý tưởng gần tương đương trên cả hai tập dữ liệu này, đảm bảo tính tổng quát hóa tốt khi áp dụng trên dữ liệu huấn luyện và thực tế.

## Những khó khăn, hạn chế

 **Khó khăn gặp phải trong quá trình thực hiện**:

* **Kỹ thuật**: Các vấn đề liên quan đến dữ liệu (thiếu, không đồng nhất, nhiều nhiễu,dự liệu bị khuyết), hạn chế trong xử lý thuật toán hoặc hiệu suất mô hình đã khiến nhóm làm lại mô hình rất liều nhần trước đó,
* **Thời gian**: Trong khuân khổ của môn học nhóm vẩn chưa tạo ra 1 mô hình đủ lớn cũng như mong đợi ban đầu

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | “Giới thiệu về Support Vector Machine (SVM),” Viblo, [Trực tuyến]. Available: https://viblo.asia/p/gioi-thieu-ve-support-vector-machine-svm-6J3ZgPVElmB. |
| [2] | “Kaggle,” [Trực tuyến]. Available: https://www.kaggle.com/code/kerneler/starter-hcv-data-data-set-5c94f003-e/notebook. |
| [3] | “DEEP AI,” [Trực tuyến]. Available: https://phamdinhkhanh.github.io/deepai-book/ch\_ml/DecisionTree.html. |
| [4] | “Mechine Learning cơ bản,” [Trực tuyến]. Available: https://machinelearningcoban.com/2016/12/26/introduce/ . |

# PHỤ LỤC

**Bảng phân công công việc của các thành viên trong nhóm**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** |  | **Nội dung công việc** | 22DH112900  Phạm Đình Anh Phương | 21DH110957  Mai Hòa Hảo | 22DH114489  Vũ Xuân Dương |
| 1 | 10% | Làm Báo Cáo | 35% | 25% | 40% |
| 2 | 15% | Xử Lý Dữ Liệu | 40% | 30% | 20% |
| 3 | 45% | Code chính | 45% | 27,5% | 27,5% |
| 4 | 10% | Tìm bộ dữ liệu | 20% | 40% | 30% |
| 5 | 10% | Xây dựng thuật toán | 34% | 33% | 33% |