

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**-----🙠🕮🙢-----**

**BÁO CÁO BTL THUỘC HỌC PHẦN**

**TRÍ TUỆ NHÂN TẠO**

**ĐỀ TÀI:**

**TÌM HIỂU CÂY QUYẾT ĐỊNH VÀ ỨNG DỤNG**

**DỰ ĐOÁN SỨC KHỎE**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn :** | **ThS Trần Thanh Huân** |
| **Lớp :** | **20242IT6094001** |
| **Nhóm :** | **2** |
| **Thành viên :** | **Trần Phương Anh – 2023603990**  **Trần Thị Chi – 2023603380**  **Đỗ Thị Lan – 2023603182**  **Nguyễn Thu Thuỷ - 2023602988** |

**Hà Nội, năm 2025**

# **LỜI NÓI ĐẦU**

Trong thời đại công nghệ 4.0, trí tuệ nhân tạo (AI - Artificial Intelligence) đang ngày càng khẳng định vai trò quan trọng trong nhiều lĩnh vực của đời sống, đặc biệt là trong y tế. Việc áp dụng các kỹ thuật AI để hỗ trợ chẩn đoán, theo dõi và dự đoán tình trạng sức khỏe đã và đang trở thành xu hướng tất yếu nhằm nâng cao chất lượng dịch vụ chăm sóc sức khỏe, tiết kiệm thời gian và chi phí cho cả bệnh nhân và đội ngũ y bác sĩ.

Trong số các kỹ thuật học máy (machine learning) được ứng dụng phổ biến, cây quyết định (decision tree) là một thuật toán đơn giản nhưng hiệu quả, dễ hiểu và dễ triển khai. Cây quyết định giúp mô hình hóa quá trình ra quyết định thông qua các quy tắc phân nhánh theo đặc trưng dữ liệu, từ đó cho phép hệ thống đưa ra dự đoán hoặc phân loại một cách trực quan và logic.

Với đề tài "Tìm hiểu cây quyết định và ứng dụng dự đoán sức khỏe", nhóm em hướng đến việc nghiên cứu lý thuyết về cây quyết định, cách thức xây dựng và vận hành mô hình, đồng thời áp dụng thực tế vào một bài toán dự đoán tình trạng sức khỏe dựa trên dữ liệu đầu vào. Qua đó, nhóm mong muốn hiểu rõ hơn về cách trí tuệ nhân tạo có thể hỗ trợ hiệu quả trong lĩnh vực y tế.

Báo cáo này được thực hiện dưới sự hướng dẫn của ThS. Trần Thanh Huân. Nhóm xin gửi lời cảm ơn chân thành đến thầy vì đã tận tình hướng dẫn, hỗ trợ và tạo điều kiện thuận lợi trong suốt quá trình thực hiện đề tài và truyền đạt kiến thức và tạo môi trường học tập tốt để nhóm có thể hoàn thành báo cáo này.

Mặc dù đã cố gắng hoàn thiện, nhưng do hạn chế về kiến thức và kinh nghiệm, báo cáo không tránh khỏi những thiếu sót. Nhóm rất mong nhận được sự góp ý của thầy cùng toàn thể bạn đọc để hoàn thiện hơn trong những lần sau.

**Nhóm em xin trân trọng cảm ơn!**

**MỤC LỤC**

[LỜI NÓI ĐẦU 1](#_Toc200383717)

[DANH MỤC HÌNH ẢNH 3](#_Toc200383718)

[1.1. Tìm hiểu về cây quyết định 4](#_Toc200383719)

[1.2. Quy trình phân lớp dữ liệu dựa trên cây quyết định 5](#_Toc200383720)

[1.3. Độ lợi thông tin 5](#_Toc200383721)

[1.4. Thuật toán ID3 6](#_Toc200383722)

[1.5. Các bước xây cây quyết định 8](#_Toc200383723)

[CHƯƠNG 2. XÂY DỰNG ỨNG DỤNG DỰ ĐOÁN SỨC KHỎE 17](#_Toc200383724)

[2.1. Mô tả bài toán 17](#_Toc200383725)

[2.2. Cài đặt bài toán 18](#_Toc200383726)

[2.2.1. Ngôn ngữ và công cụ triển khai 18](#_Toc200383727)

[2.2.1.1 Python 18](#_Toc200383728)

[2.2.1.2 PyCharm 18](#_Toc200383729)

[2.2.1.3 PyQt 18](#_Toc200383730)

[2.2.2. Data cho bài toán 19](#_Toc200383731)

[2.2.3 Các thành phần để cài đặt thuật toán 19](#_Toc200383732)

[KẾT LUẬN 30](#_Toc200383733)

[TÀI LIỆU THAM KHẢO 31](#_Toc200383734)

# **DANH MỤC HÌNH ẢNH**

Hình 1.1. Quy trình training……………………………………………………..4

Hình 1.2. Dữ liệu training……………………………………………………….8

Hình 1.3. Tìm thuộc tính tốt nhất………………………………………………..9

Hình 1.4. Tìm thuộc tính tốt nhất………………………………………………10

Hình 1.5. Tìm thuộc tính tốt nhất………………………………………………11

Hình 1.6. Tìm thuộc tính tốt nhất………………………………………………12

Hình 1.7. Dữ liệu training sau khi chia nhánh và sắp xếp……………………...13

Hình 1.9. Dữ liệu training sau khi chia nhánh và sắp xếp……………………...13

Hình 1.10. Dữ liệu training sau khi chia nhánh và sắp xếp…………………….13

Hình 1.11. Cây quyết định……………………………………………………...14

Hình 1.12. Tập luật……………………………………………………………..15

Hình 2.1. Hàm load\_data……………………………………………………….18

Hình 2.2. Hàm cal\_entropy.…………………………………………………….18

Hình 2.3. Hàm partition…………………………………………………………19

Hình 2.3. Hàm find\_best\_attribute.……………………………………………..20

Hình 2.4. Hàm calculate\_attr\_importance.……………………………………...21

Hình 2.5. Hàm find\_best\_attribute.……………………………………………..22

Hình 2.6. Hàm build\_decision\_tree.…………………………………………….23

Hình 2.7. Hàm generate\_rules.………………………………………………….24

Hình 2.8. Hàm save\_rules\_to\_csv.……………………………………………...25

Hình 2.9. Hàm load\_rules\_from\_csv.…………………………………………...26

Hình 2.10. Hàm diagnose.………………………………………………………27

Hình 2.11. Giao diện chương trình.……………………………………………..28

**CHƯƠNG 1. TỔNG QUAN VỀ CÂY QUYẾT ĐỊNH**

## **1.1. Tìm hiểu về cây quyết định**

Trong lĩnh vực [máy học](https://vi.wikipedia.org/wiki/M%C3%A1y_h%E1%BB%8Dc), cây quyết định là một kiểu mô hình dự báo (predictive model), nghĩa là một ánh xạ từ các quan sát về một sự vật/hiện tượng tới các kết luận về giá trị mục tiêu của sự vật/hiện tượng. Mỗi một nút trong (internal node) tương ứng với một biến; đường nối giữa nó với nút con của nó thể hiện một giá trị cụ thể cho biến đó. Mỗi nút lá đại diện cho giá trị dự đoán của biến mục tiêu, cho trước các giá trị của các biến được biểu diễn bởi đường đi từ nút gốc tới nút lá đó. Kỹ thuật học máy dùng trong cây quyết định được gọi là học bằng cây quyết định, hay chỉ gọi với cái tên ngắn gọn là cây quyết định.

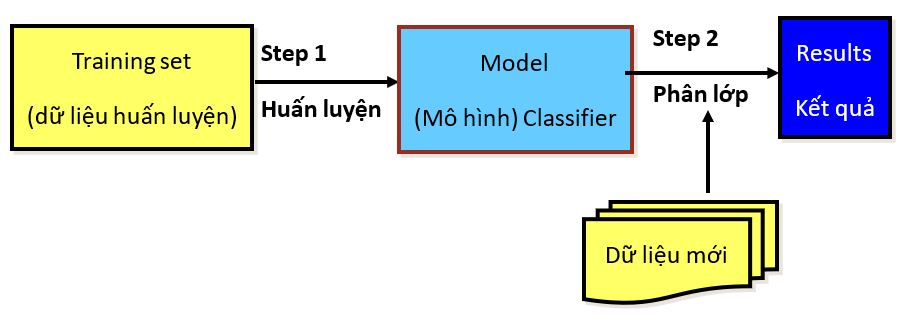
Học bằng cây quyết định cũng là một phương pháp thông dụng trong [khai phá dữ liệu](https://vi.wikipedia.org/wiki/Khai_ph%C3%A1_d%E1%BB%AF_li%E1%BB%87u). Khi đó, cây quyết định mô tả một cấu trúc cây, trong đó, các lá đại diện cho các phân loại còn cành đại diện cho các kết hợp của các thuộc tính dẫn tới phân loại đó. Một cây quyết định có thể được học bằng cách chia [tập hợp](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_h%E1%BB%A3p) nguồn thành các tập con dựa theo một kiểm tra giá trị thuộc tính. Quá trình này được lặp lại một cách đệ quy cho mỗi tập con dẫn xuất. Quá trình [đệ quy](https://vi.wikipedia.org/wiki/%C4%90%E1%BB%87_quy) hoàn thành khi không thể tiếp tục thực hiện việc chia tách được nữa, hay khi một phân loại đơn có thể áp dụng cho từng phần tử của tập con dẫn xuất. Một bộ phân loại [rừng ngẫu nhiên](https://vi.wikipedia.org/w/index.php?title=R%E1%BB%ABng_ng%E1%BA%ABu_nhi%C3%AAn&action=edit&redlink=1) (random forest) sử dụng một số cây quyết định để có thể cải thiện tỉ lệ phân loại.

Cây quyết định còn có hai tên khác:

- Cây hồi quy (Regression tree) ước lượng các hàm giá có giá trị là [số thực](https://vi.wikipedia.org/wiki/S%E1%BB%91_th%E1%BB%B1c) thay vì được sử dụng cho các nhiệm vụ phân loại. (ví dụ: ước tính giá một ngôi nhà hoặc khoảng thời gian một bệnh nhân nằm viện).

- Cây phân loại (Classification tree), nếu y là một biến phân loại như: giới tính (nam hay nữ), kết quả của một trận đấu (thắng hay thua).

## **1.2. Quy trình phân lớp dữ liệu dựa trên cây quyết định**

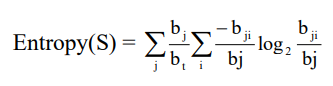


*Hình 1.1. Quy trình training*

## **1.3. Độ lợi thông tin**

Với bảng dữ liệu huấn luyện có các thuộc tính điều kiện A1, A2, …, An, vấn đề đặt ra là trong số thuộc tính này ta sẽ lựa chọn thuộc tính nào làm gốc của cây. Để lựa chọn thuộc tính làm gốc của cây ta cần đưa ra tiêu chuẩn để đánh giá các thuộc tính này, một trong các tiêu chuẩn đó là độ lợi thông tin (information gain) của thuộc tính. Các độ đo dựa trên lý thuyết thông tin đều sử dụng đến một khái niệm đó là entropy. Entropy là đại lượng đặc trưng cho mức độ hỗn loạn của các phần tử trong quần thể S.

Entropy của tập S được tính theo công thức:



Với:

S: tập dữ liệu mẫu;

bt: tổng số phần tử có trong phân hoạch;

bj: tổng số phần tử có thuộc tính A với giá trị của thuộc tính là j;

bji: tổng số phần tử của thuộc tính A có giá trị j ứng với thuộc tính mục tiêu

có giá trị i.

- Nếu Entropy(S) = 0 thì S đồng nhất (phân tách hoàn toàn)

- Nếu Entropy (S) = 1 thì số phần tử thuộc lớp ci (i = 1, 2,…, n) bằng nhau

- Nếu Entropy(S) ∈ (0;1) thì số phần tử thuộc các lớp ci khác nhau

Độ lợi thông tin (information gain) là đại lượng được dùng để chọn thuộc tính nhằm phân chia tập dữ liệu huấn luyện. Độ lợi thông tin của thuộc tính A trong tập mẫu S, ký hiệu Gain(S, A) và được xác định qua công thức:

**Gain(S, A) = Entropy(S) – Info(S, A)**

Trong đó:

Info(S, A) = ∑((|Sv| / |S|) \* Entropy(Sv))

Sv: tập con của S mà ở đó thuộc tính A có giá trị v;

|S|: số mẫu trong S.

Chẳng hạn S là tập hợp của 14 mẫu trong đó có 9 mẫu thuộc lớp Yes và 5 mẫu thuộc lớp No. Khi đó Entropy(S) = - (9/14) log2(9/14) - (5/14) log2(5/14) = 0,940 6

**🡲 Nhận xét:** Entropy(S) là hằng số đối với tất cả thuộc tính. Mức độ hỗn loạn của các phân hoạch do thuộc tính A tạo ra (Entropy (SV)) được mong đợi là có giá trị càng nhỏ càng tốt (ít hỗn loạn).

## **1.4. Thuật toán ID3**

Thuật toán ID3 (Iterative DiChaudomiser 3) được J. Ross Quinlan phát triển tại đại học Sydney năm 1975. ID3 dựa trên thuật giải CLS (Concept Learning System). ID3 là thuật giải học có giám sát (supervised learning algorithm), tạo cây quyết định dựa trên tập dữ liệu có sẵn. Cây kết quả được sử dụng để phân lớp cho mẫu mới chưa được gán nhãn.

Thuật toán ID3 tạo cây quyết định dựa trên độ lợi thông tin thu được từ tập dữ liệu huấn luyện. Thuộc tính được chọn là thuộc tính có độ lợi thông tin lớn nhất. Thuật toán ID3 thường được sử dụng với các thuộc tính định danh (nominal attributes) để phân lớp với dữ liệu đầy đủ.

Thuật toán này tạo cây quyết định theo chiến lược phân tích từ trên xuống (Top – Down), bắt đầu từ một tập các đối tượng và các thuộc tính của nó. Tại mỗi nút của cây một thuộc tính được kiểm tra, kết quả của phép kiểm tra này được sử dụng để phân chia tập đối tượng theo kết quả kiểm tra trên. Quá trình này được thực hiện một cách đệ quy cho tới khi tập đối tượng trong cây con được sinh ra thuần nhất (các mẫu có cùng giá trị của thuộc tính mục tiêu) theo một tiêu chí phân lớp nào đó.

Cho X là tập các vectơ đặc trưng (không gian đặc trưng)

X = {x1, x2,…, xn} trong đó xi = (xi1, xi2,…, xim) với i = 1, 2, …, n

C là tập các giá trị phân lớp

c: X → C thực hiện gán nhãn cho phần tử trong X D = {(x1, c(x1)), (x2, c(x2),…, (xn, c(xn)} ⊆ X × C

Nhiệm vụ: Dựa trên D, hãy xây dựng cây quyết định T xấp xỉ c.

ID3(D, Atributes, Target)

{

- Tạo nút t cho cây,

- Nếu tất cả các mẫu trong D đều thuộc lớp dương thì trả về cây có nút gốc t và được gán nhãn “+”,

- Nếu tất cả các mẫu trong D đều thuộc lớp âm thì trả về cây có nút gốc t và được gán nhãn “-”,

- Đặt t là nhãn phổ biến nhất của thuộc tính mục tiêu trong D,

- Nếu Atributes rỗng thì trả về cây có một nút gốc trỏ bởi t,

- Trường hợp khác:

+ Đặt A\* là thuộc tính phân lớp tốt nhất trong D,

+ Thuộc tính quyết định của t là A\* .

+ Với mỗi giá trị a của A\* thực hiện:

• Bổ sung nhánh mới dưới t ứng với A\* = “a”,

• Đặt D\_a là tập con của D chứa các mẫu mà A\* = “a”,

• Nếu D\_a rỗng thì dưới nhánh mới này bổ sung nút lá với nhãn phổ biến nhất trong D.

• Ngược lại, dưới nhánh mới này bổ sung cây con ID3(D\_a, Attributes − {A\*}, Target).

- Trả về t.

}

**Sau đây là mã giả cho thuật giải ID3:**

ID3(D, Attributes, Target)

{

1. t = createNode()

2. IF ∀(x, c(x)) ∈D: c(x) = 1 THEN label(t) = ‘+’ return(t) ENDIF

3. IF ∀(x, c(x)) ∈ D: c(x) = 0 THEN label(t) = ‘-’ return(t) ENDIF 8

4. label(t) = mostCommonClass(D, Target)

5. IF Attributes = ∅ THEN return(t) ENDIF

6. A\* = argmaxA∈Attributes(Gain(D, A))

7. FOR EACH a∈A\* DO

Da = {(x, c(x)) ∈ D: x| \* A = a}

IF Da = ∅ THEN

t ’ = createNode()

label(t’ ) = mostCommonClass(D, Target)

createEdge(t, a, t’ )

ELSE

createEdget(t, a, ID3(Da, Attributes – {A\*}, Target))

ENDIF

ENDDO

8. return(t)

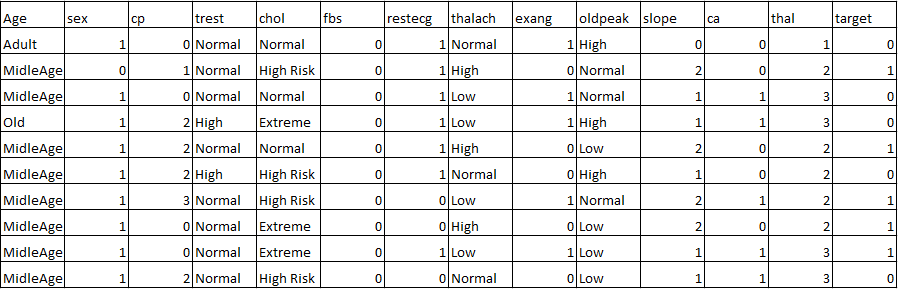
}

## **1.5. Các bước xây cây quyết định**

1.5.1 Step1: Tạo cây quyết định từ dữ liệu huấn luyện

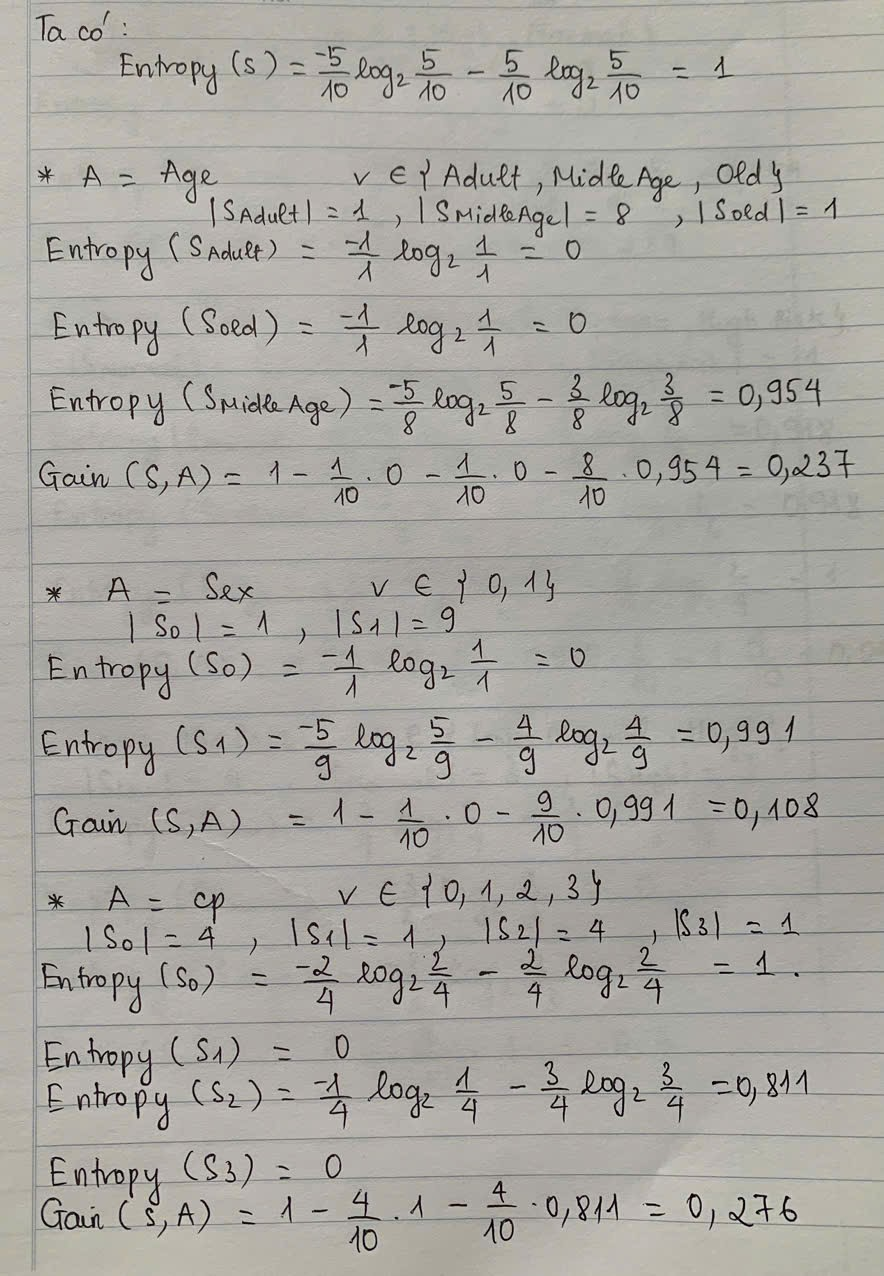
1. Chọn thuộc tính “tốt nhất” theo một độ đo lựa chọn cho trước
2. Mỗi giá trị trên thuộc tính được chọn sẽ thành một nhánh
3. Sắp các mẫu dữ liệu tương ứng vào các nhánh
4. Nếu các mẫu được phân lớp rõ, tạo một nút lá; ngược lại thì lặp từ 1->4
5. Tỉa thưa: tỉa bỏ các nút không ổn định

Ví dụ với 10 dữ liệu đầu tiên của tập dữ liệu

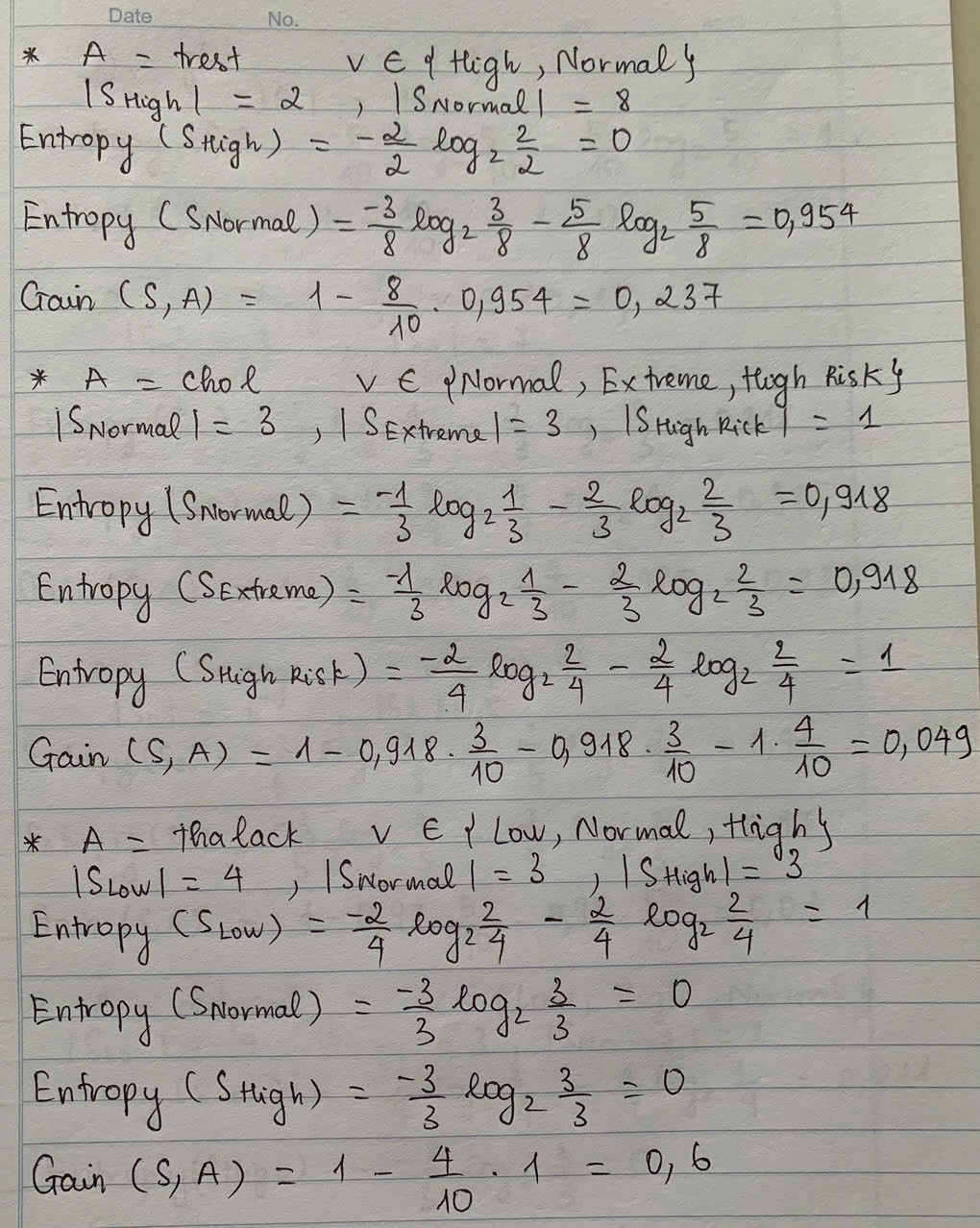


*Hình 1.2. Dữ liệu training*

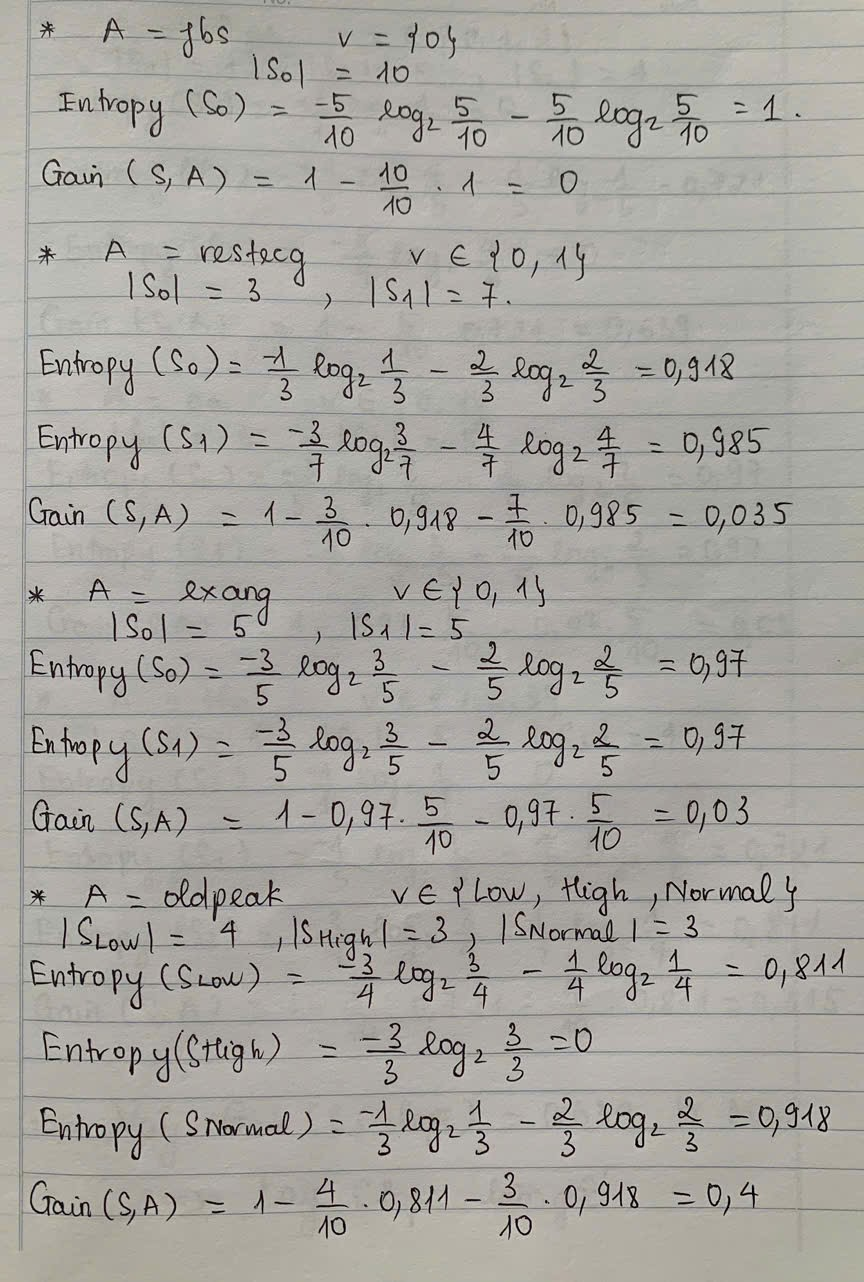
Ta sẽ chọn thuộc tính “tốt nhất” theo độ quan trọng (Information Gain) trước đó ta đi tìm IG của từng thuộc tính



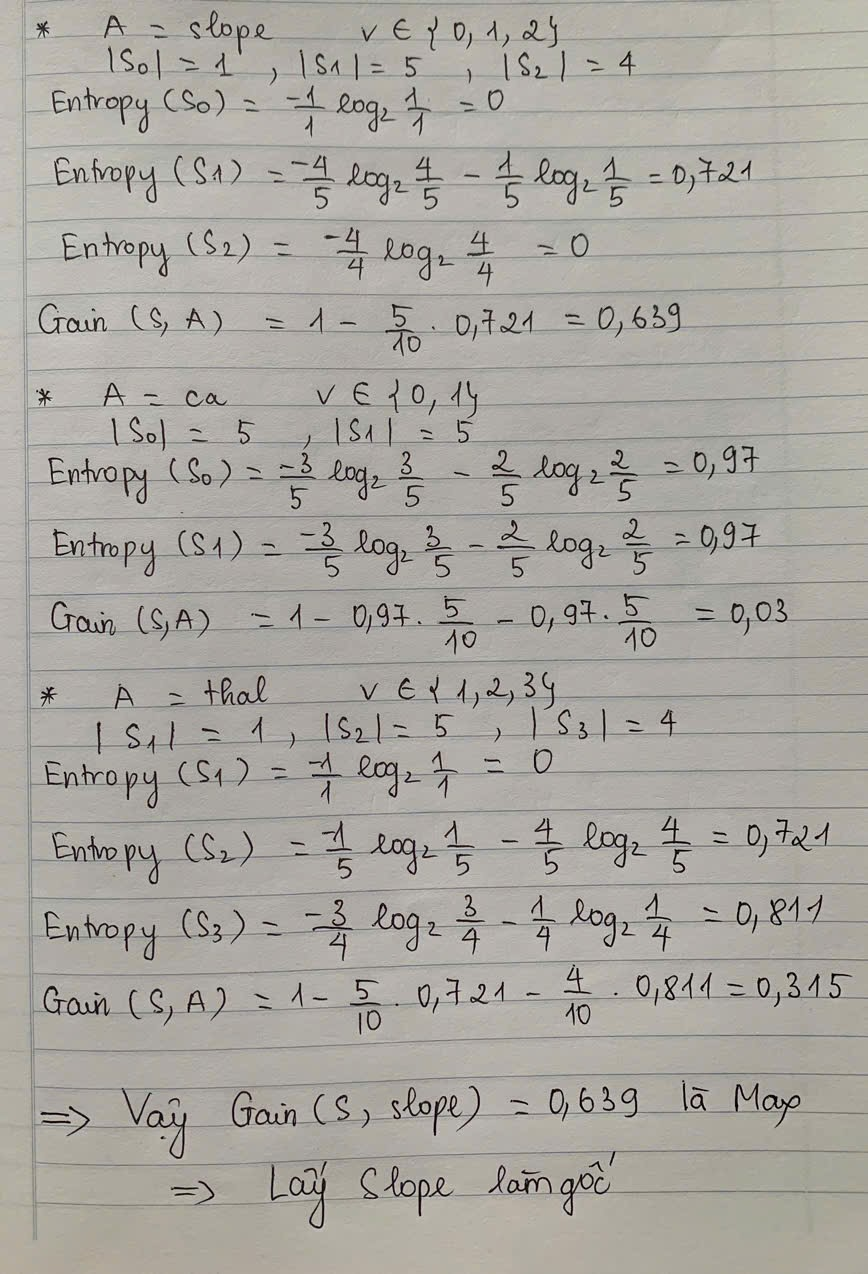
*Hình 1.3. Tìm thuộc tính tốt nhất*



*Hình 1.4. Tìm thuộc tính tốt nhất*



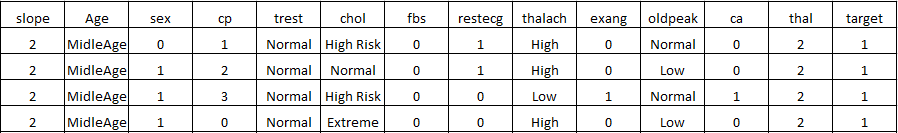
*Hình 1.5. Tìm thuộc tính tốt nhất*

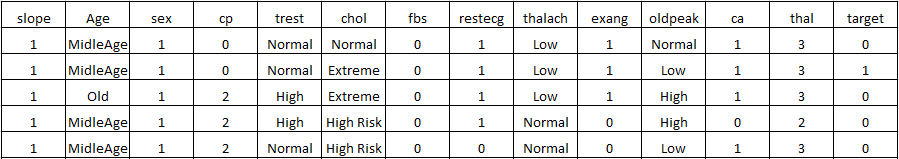


*Hình 1.6. Tìm thuộc tính tốt nhất*

* Chia nhánh và sắp xếp mẫu dữ liệu

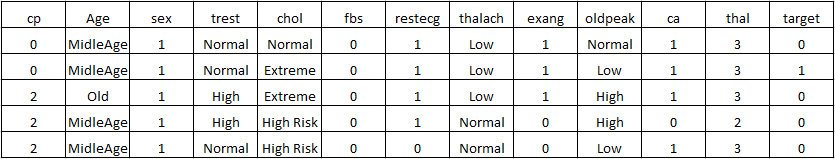






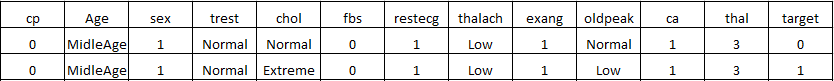
*Hình 1.7. Dữ liệu training sau khi chia nhánh và sắp xếp*

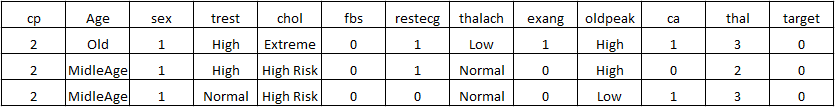
* Tạo nút lá với slope = 2; slope = 0 và lặp lại các bước từ 1 tới 4 với bảng chứa slope = 1. Tính lại Gain với bảng và tìm ra Gain(S,cp) là max



*Hình 1.8. Dữ liệu training sau khi chia nhánh và sắp xếp*

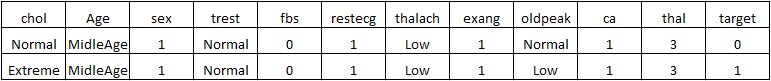
* Tiếp tục chia nhánh và sắp xếp mẫu dữ liệu





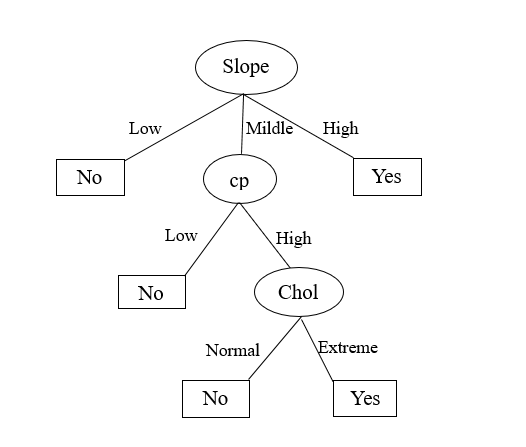
*Hình 1.9. Dữ liệu training sau khi chia nhánh và sắp xếp*

* Tạo nút lá với cp = 2 và lặp lại các bước từ 1 tới 4 với bảng chứa cp = 0. Tính lại Gain với bảng và tìm ra Gain(S,chol) là max



*Hình 1.10. Dữ liệu training sau khi chia nhánh và sắp xếp*

* Tạo nút lá với chol = Normal và chol = Extreme.
* Theo các bước đx liệt kê từ tập ví dụ với 10 dữ liệu đầu vào ta được cây quyết định có dạng như sau:



*Hình 1.11. Cây quyết định*

1.5.2 Step 2: Xây dựng tập luật

1. Duyệt cây quyết định: Bắt đầu từ gốc của cây quyết định, duyệt qua các nút và nhánh để xác định các điều kiện và quyết định được thực hiện tại mỗi nút
2. Xây dựng luật: Đối với mỗi nhánh, xây dựng một luật dựa trên điều kiện và quyết định ở nút đó. Mỗi luật thường có định dạng Nếu [điều kiện] thì [quyết định]
3. Đệ quy: Nếu một nhánh của cây dẫn đến một nút lá( nút không có nhánh con), thì đó là một luật đơn giản mà bạn có thể thêm vào tập luật.
4. Xử lý các điều kiện phức tạp: Nếu cây quyết định có điều kiện phức tạp, bạn có thể cần biến đổi chúng thành các điều kiện đơn giản hơn trong tập luật
5. Lặp lại cho tất cả các nhánh: Lặp lại quá trình trên cho tất cả các nhánh của cây để tạo ra một tập luật đầy đủ.

Dựa trên cây quyết định vừa tạo, ta có tập luật:

A table with black text

Description automatically generated

*Hình 1.12. Tập luật*

+ Với slope = 0 ta sẽ dự đoán được 0 (điều đó ứng với việc bệnh nhân không bị bệnh)

+ Với slope = 2 ta sẽ dự đoán được 1 (điều đó ứng với việc bệnh nhân bị bệnh)

+ Với slope = 1 và cp = 2 ta sẽ dự đoán được 0 (điều đó ứng với việc bệnh nhân không bị bệnh)

+ Với slope = 1, cp = 0, chol = Normal ta sẽ dự đoán được 0 (điều đó ứng với việc bệnh nhân không bị bệnh)

+ Với slope =1, cp = 0, chol = Extreme ta sẽ dự đoán được 1 (điều đó ứng với việc bệnh nhân bị bệnh

# **CHƯƠNG 2. XÂY DỰNG ỨNG DỤNG DỰ ĐOÁN SỨC KHỎE**

## **2.1. Mô tả bài toán**

* Đầu vào: Gồm những đặc trưng liên quan như: Age, Sex, Chest Pain Type, Resting Blood Presure, Serum Cholesterol, Fasting Blood Sugar, Maximum Heart Rate, Exercise Induced Anginan, ST depression, Number of major vessels, Slope of peak, Thalium Stress Type, Resting Electrocardiographic Results

Trong đó:

* Age đặc trưng cho độ tuổi. Từ 25 - 45 tuổi là Adult, từ 46 – 65 là MiddleAge, từ 66 trở lên là Old
* Sex đặc trưng cho giới tính. Với 1 là Nam, 0 là Nữ
* Chest Pain Type đặc trưng cho loại đau ngực. Với 0 là đau thắt ngực điển hình, 1 là đau thắt ngực không điển hình, 2 là không đau thắt ngực, 3 là tiệm cận
* Resting Blood Presure đặc trưng cho áp lực máu khi nghỉ ngơi
* Serum Cholesterol đặc trưng cho lượng mỡ máu
* Fasting Blood Sugar đặc trưng cho lượng đường trong máu khi đói. Với 0 là lượng nhỏ hơn hoặc bằng 120, 1 là lượng lớn hơn 120
* Resting Electrocardiographic Results đặc trưng cho kết quả kiểm tra điện tâm đồ. Với 0 là bình thường, 1 là có sóng ST-T bất thường, 2 là phì đại thất trái
* Maximum Heart Rate đặc trưng cho tần số tim cao nhất
* Exercise Induced Anginan đặc trưng cho đau ngực khi vận động mạnh. Với 0 là không đau khi vận động, 1 là có đau khi vận động
* ST depression đặc trưng cho sự giảm ST được gây ra bởi vận động so với trạng thái nghỉ
* Slope of peak đặc trưng cho độ dốc của đoạn ST trong thời gian tập luyện cao điểm. Với 0 là dốc lên, 1 là phẳng, 2 là dốc xuống
* Number of major vessels đặc trưng cho số mạch máu lớn phát hiện
* Thalium Stress Type đặc trưng cho khả năng lưu thông máu đến tim. Với 1 là bình thường, 2 là khuyết điểm đã hồi phục, 3 là khuyết điểm có thể đảo ngược/ có thể hồi phục được
* Quy trình thuật toán: Sử dụng tập dữ liệu đã được cung cấp từ trước và áp dụng thuật toán cây quyết định thông qua các hàm.
* Đầu ra: In ra việc bị bệnh hay không bệnh với tập luật đã được áp dụng.

## **2.2. Cài đặt bài toán**

### **2.2.1. Ngôn ngữ và công cụ triển khai**

#### **2.2.1.1 Python**

Ngôn ngữ Python là một ngôn ngữ lập trình thông dịch, dễ đọc và dễ hiểu, được sử dụng rộng rãi trong nhiều lĩnh vực khác nhau. Nó có cú pháp đơn giản và hỗ trợ nhiều thư viện mạnh mẽ, giúp người phát triển dễ dàng triển khai các dự án phức tạp. Python cũng là ngôn ngữ rất phổ biến trong lĩnh vực trí tuệ nhân tạo và học máy, nhờ vào các thư viện như NumPy, Pandas, Scikit-learn và TensorFlow.

#### **2.2.1.2 PyCharm**

Công cụ PyCharm là một môi trường phát triển tích hợp (IDE) cho Python, được phát triển bởi JetBrains. PyCharm cung cấp nhiều tính năng hữu ích như gợi ý mã, kiểm tra lỗi, và tích hợp với các công cụ quản lý mã nguồn như Git. Nó cung cấp một giao diện sử dụng thân thiện, giúp người phát triển dễ dàng theo dõi và quản lý mã nguồn của mình.

#### **2.2.1.3 PyQt**

PyQt là một bộ công cụ phát triển GUI (Graphical User Interface) cho Python, được xây dựng dựa trên Qt, một toolkit và framework cho việc xây dựng giao diện người dùng đồ họa. PyQt giúp người phát triển tạo ra ứng dụng desktop đa nền tảng với giao diện đẹp mắt và chức năng phong phú.

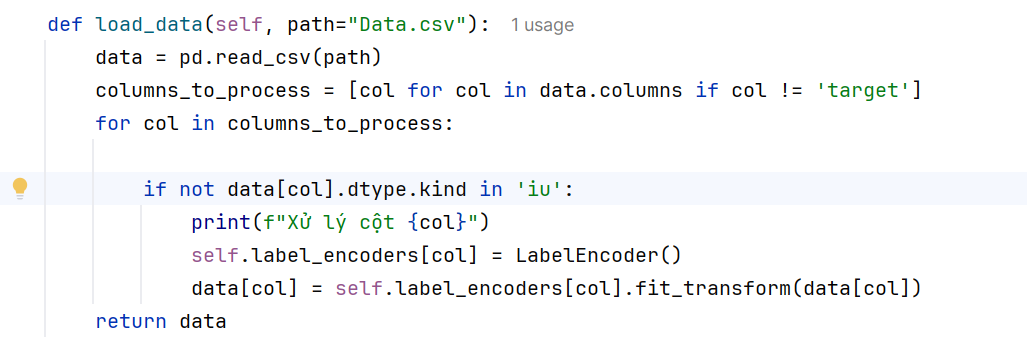
#### **2.2.2. Data cho bài toán**

Link của tập dữ liệu:

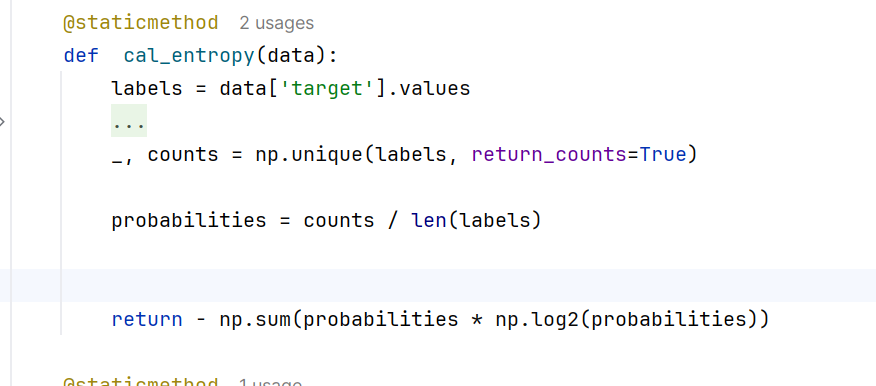
<https://drive.google.com/drive/folders/11jYeJ2Nh3gctLrBOhyp-CHcznmi0JQhv>

### **2.2.3 Các thành phần để cài đặt thuật toán**

* **Hàm đọc dữ liệu từ file CSV sử dụng pandas**



*Hình 2.1. Hàm load\_data*

* **Hàm tính độ tinh khiết của một tập dữ liệu**

*Hình 2.2. Hàm cal\_entropy*

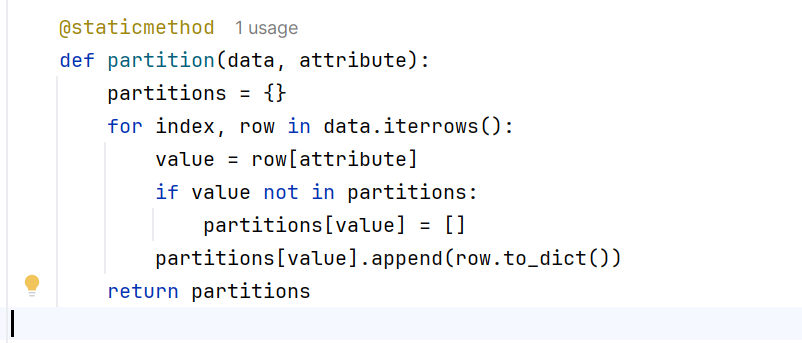
* Để tính entropy của tập dữ liệu với đầu vào là data, đầu ra là giá trị entropy
* Hàm bắt đầu bằng cách trích xuất cột nhãn target từ dữ liệu
* Sử dụng Counter để đếm số lần xuất hiện của mỗi nhãn trong cột nhãn
* Khởi tạo giá trị entropy ban đầu là 0.0
* Tính tổng số mẫu trong tập dữ liệu
* Tính xác suất của mỗi lớp.
* Duyệt qua từng lớp nhãn và tính entropy cho mỗi lớp
* Sử dụng công thức entropy:

A black symbols with letters and numbers

Description automatically generated with medium confidence

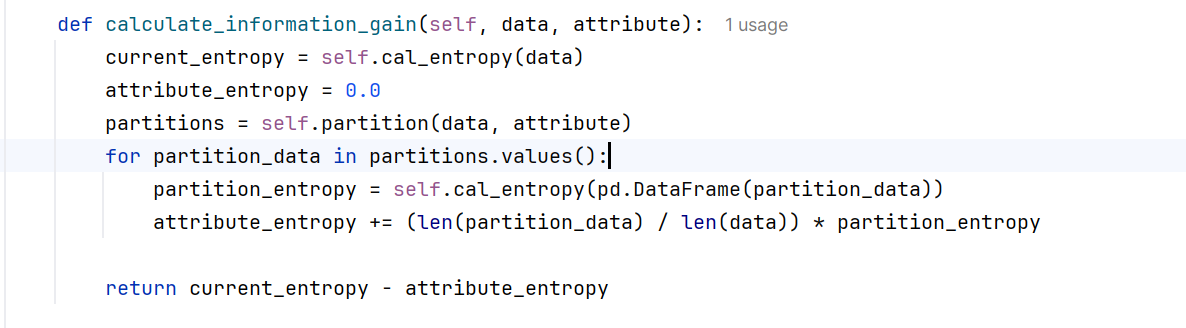
(với p­i là xác xuất của mỗi lớp)

* Sau đó tổng hợp entropy của tất cả các lớp để có giá trị entropy cuối cùng
* **Hàm chia dữ liệu thành các tập con dựa trên giá trị của thuộc tính**



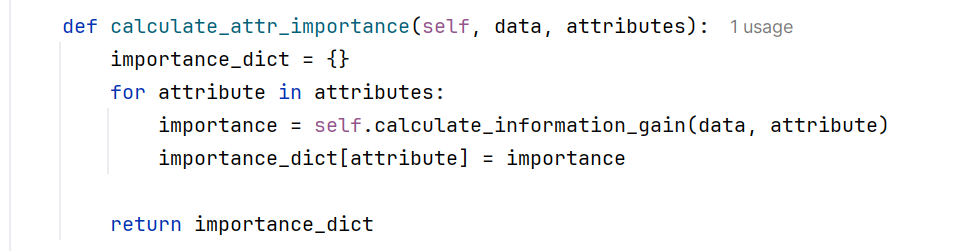
*Hình 2.3. Hàm partition*

* Dùng để phân đoạn thuộc tính với đầu vào là data và thuộc tính để phân đoạn, đầu ra là một từ điển với các giá trị thuộc tính làm khóa và các phân đoạn dữ liệu tương ứng làm giá trị.
* Hàm bắt đầu bằng cách khởi tạo một từ điển rỗng có tên là partitions để lưu trữ các phân đoạn dữ liệu dựa trên giá trị của thuộc tính.
* Dùng vòng lặp for để duyệt qua từng hàng trong DataFrame data sử dụng iterrows(). Với index là chỉ số của hàng, và row là một Series (dạng key-value) của Pandas đại diện cho giá trị trong hàng đó
* Trích xuất giá trị của thuộc tính từ hàng hiện tại
* Nếu giá trị của thuộc tính không phải là một khóa trong partitions, tạo một khóa mới trong từ điển
* Thêm hàng vào phân đoạn tương ứng với giá trị của thuộc tính
* Sau khi duyệt qua tất cả các hàng, hàm trả về từ điển partitions với các phân đoạn được tạo ra
* **Hàm tìm thuộc tính tốt nhất để chia dữ liệu (calculate\_information\_gain)**



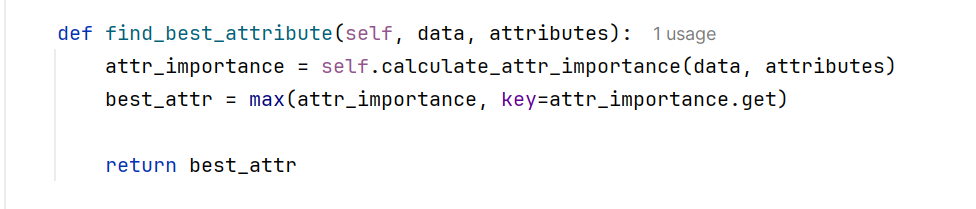
*Hình 2.3. Hàm find\_best\_attribute*

* Chức năng: Hàm này tính toán lượng thông tin thu được (Information Gain) khi phân chia dữ liệu theo một thuộc tính cụ thể. Đây là một chỉ số quan trọng trong thuật toán Cây Quyết Định (Decision Tree), giúp xác định mức độ quan trọng của thuộc tính đối với việc phân loại dữ liệu.
* Cách hoạt động:
* Bắt đầu bằng việc tính toán entropy hiện tại của toàn bộ dữ liệu (current\_entropy)
* Entropy là một thước đo mức độ hỗn loạn (không đồng nhất) của dữ liệu.
* Hàm self.cal\_entropy(data) tính entropy của tập dữ liệu đầu vào.
* Khởi tạo giá trị attribute\_entropy = 0.0
* Đây sẽ là tổng entropy có điều kiện sau khi dữ liệu được chia nhỏ theo thuộc tính.
* Gọi hàm partition(data, attribute) để phân chia dữ liệu theo thuộc tính
* partitions là một từ điển, trong đó mỗi khóa là một giá trị của thuộc tính, còn giá trị là danh sách các hàng dữ liệu tương ứng.
* Duyệt qua từng phân đoạn dữ liệu trong partitions.values()
* Với mỗi nhóm dữ liệu partition\_data, ta tính entropy của nó.
* Dữ liệu trong partition\_data được chuyển thành DataFrame để đảm bảo tính toán đúng.
* Cập nhật attribute\_entropy
* Nhân entropy của từng phân đoạn với tỷ lệ kích thước của nó so với toàn bộ tập dữ liệu.
* Điều này giúp xác định mức độ ảnh hưởng của mỗi phân đoạn đến tổng entropy.
* Tính toán và trả về Information Gain: Giá trị Information Gain được tính bằng cách lấy entropy ban đầu trừ đi entropy có điều kiện của thuộc tính.
* Kết quả này cho biết mức độ giảm độ hỗn loạn (entropy) khi phân tách theo thuộc tính.
* **Hàm tính tầm quan trọng của các thuộc tính(calculate\_attr\_importance)**



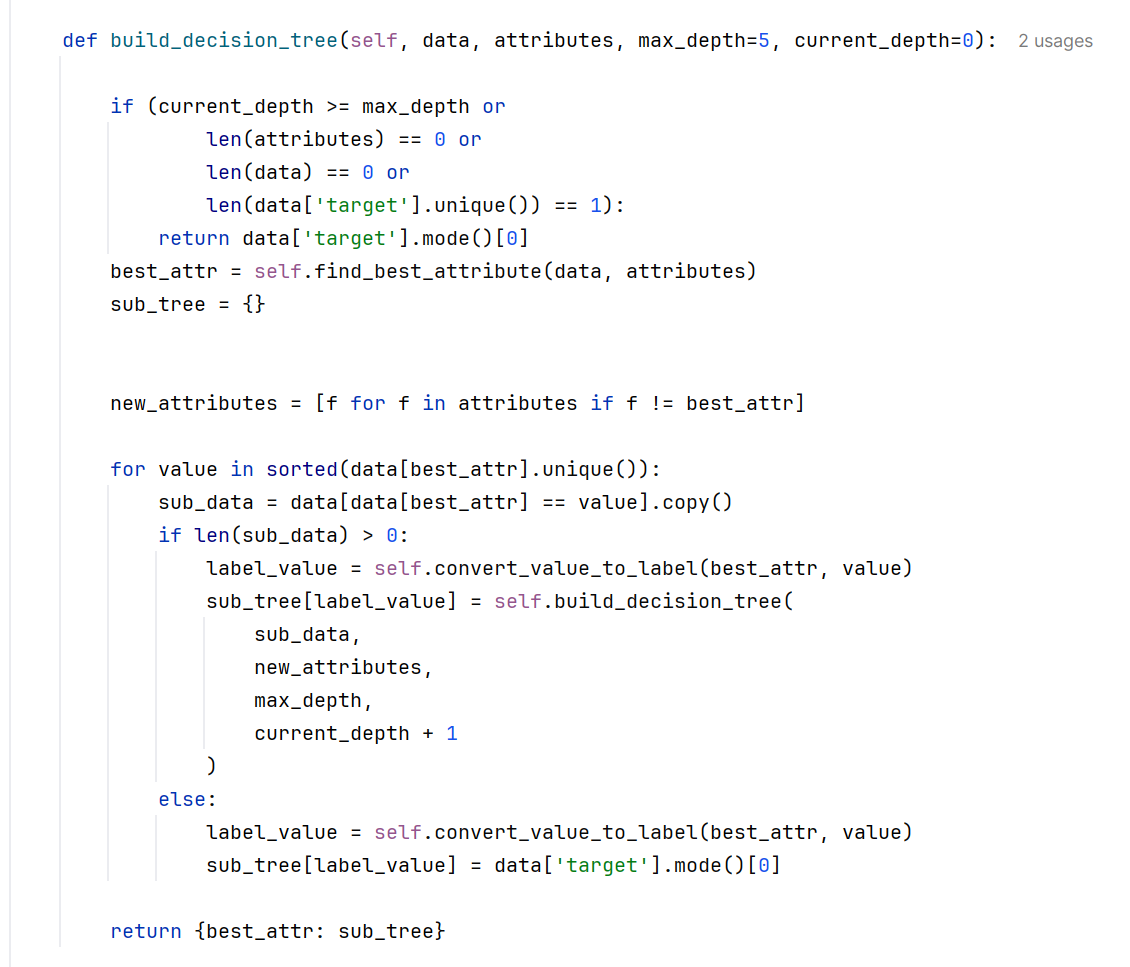
*Hình 2.4. Hàm calculate\_attr\_importance*

* Chức năng: Hàm này đánh giá mức độ quan trọng của từng thuộc tính trong danh sách bằng cách sử dụng Information Gain.
* Cách hoạt động:
* Khởi tạo từ điển rỗng importance\_dict
* Từ điển này sẽ lưu giá trị Information Gain của từng thuộc tính.
* Duyệt qua từng thuộc tính trong danh sách attributes
* Với mỗi thuộc tính, gọi hàm calculate\_information\_gain(data, attribute) để tính giá trị Information Gain.
* Lưu kết quả vào importance\_dict với khóa là tên thuộc tính và giá trị là mức độ quan trọng của nó.
* Trả về importance\_dict
* Kết quả là một từ điển chứa tất cả các thuộc tính cùng với mức độ quan trọng tương ứng của chúng.
* **Hàm tìm thuộc tính có giá trị quan trọng nhất**



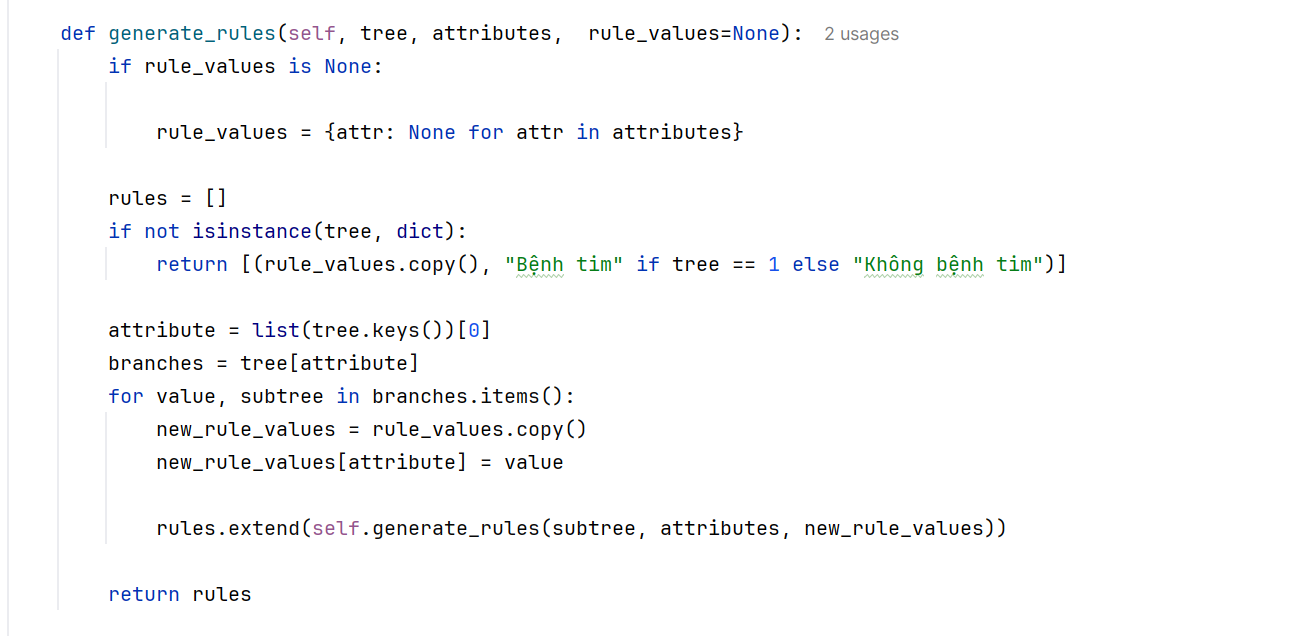
*Hình 2.5. Hàm find\_best\_attribute*

* Hàm find\_best\_attribute hoạt động như sau:
* Hàm bắt đầu bằng việc tính toán tầm quan trọng của từng thuộc tính bằng cách sử dụng phương thức acculation\_attr\_importance, trả về một dictionary (attr\_importance) chứa mức độ quan trọng của từng thuộc tính.
* Sau đó, sử dụng hàm max() để tìm thuộc tính có giá trị quan trọng nhất trong dictionary bằng cách so sánh các giá trị của nó.
* Trả về thuộc tính có tầm quan trọng cao nhất làm best\_attr.
* Hàm này hoạt động dựa trên việc tách riêng phần tính toán mức độ quan trọng của thuộc tính vào một hàm khác (acculation\_attr\_importance), giúp code gọn hơn và dễ bảo trì hơn.
* **Hàm xây dựng cây quyết định**



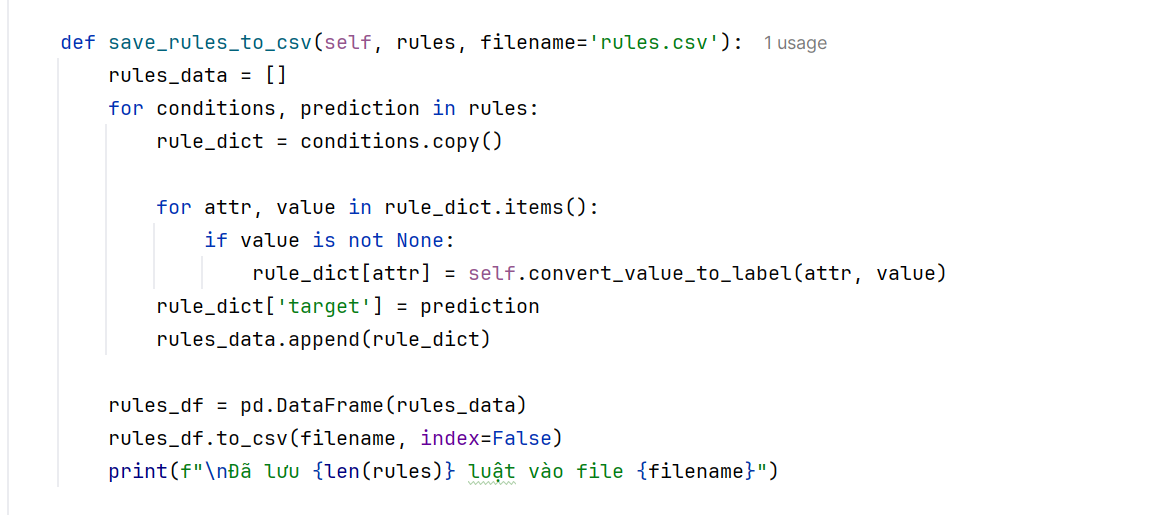
*Hình 2.6. Hàm build\_decision\_tree*

* Dùng để xây dựng cây quyết định theo thuật toán ID3 với đầu vào là data, danh sách attributes, và giới hạn độ sâu max\_depth, đầu ra là cây quyết định được biểu diễn dưới dạng từ điển.
* Đầu tiên, hàm sẽ kiểm tra nếu đạt một trong các điều kiện dừng:
* Cây đã đạt độ sâu tối đa (current\_depth >= max\_depth).
* Không còn thuộc tính nào để xem xét (len(attributes) == 0).
* Không còn dữ liệu để phân chia (len(data) == 0).
* Tất cả các mẫu thuộc cùng một lớp (len(data['target'].unique()) == 1).
* Nếu một trong các điều kiện trên xảy ra, hàm trả về nhãn xuất hiện nhiều nhất trong tập dữ liệu (data['target'].mode()[0]).
* Nếu các điều kiện trên không được đáp ứng, tìm thuộc tính tốt nhất để chia bằng cách sử dụng hàm find\_best\_attribute.
* Tạo một cây quyết định với nút gốc là thuộc tính tốt nhất và các nút con là các giá trị có thể có của thuộc tính đó.
* Loại bỏ thuộc tính vừa chọn khỏi danh sách attributes.
* Dùng vòng lặp duyệt qua từng giá trị của thuộc tính tốt nhất:
* Lọc dữ liệu để lấy tập con tương ứng với giá trị đó.
* Nếu tập con không rỗng, chuyển đổi giá trị thuộc tính sang dạng nhãn (sử dụng convert\_value\_to\_label) và tiếp tục đệ quy để xây dựng cây con với current\_depth + 1.
* Nếu tập con rỗng, gán cho nhánh con nhãn phổ biến nhất trong data['target'].
* Cuối cùng trả về cây quyết định được xây dựng.
* **Hàm sinh ra tập luật từ cây quyết định**



*Hình 2.7. Hàm generate\_rules*

* Hàm sẽ bắt đầu bằng việc khởi tạo một dictionary rule\_values chứa tất cả các thuộc tính với giá trị mặc định là None, nếu chưa được cung cấp.
* Nếu nút hiện tại là nút lá (tức là giá trị dự đoán), trả về danh sách chứa một tuple gồm:
* Bản sao của rule\_values với các giá trị đã cập nhật.
* Nhãn phân loại tương ứng: "Bệnh tim" nếu giá trị là 1, ngược lại là "Không bệnh tim".
* Nếu nút hiện tại không phải là nút lá, thực hiện các bước sau:
* Lấy tên thuộc tính gốc của cây và danh sách các nhánh con tương ứng.
* Duyệt qua từng nhánh của thuộc tính:
* Sao chép rule\_values để cập nhật giá trị thuộc tính hiện tại.
* Gọi đệ quy để tiếp tục tạo quy tắc cho cây con.
* Kết quả từ các lời gọi đệ quy được thêm vào danh sách rules.
* Cuối cùng, hàm trả về danh sách các quy tắc được tạo ra.
* **Hàm lưu trữ tập luật vào file CSV**



*Hình 2.8. Hàm save\_rules\_to\_csv*

* Cách hoạt động:
* Hàm bắt đầu với việc tạo một danh sách rules\_data, trong đó mỗi phần tử là một dictionary chứa điều kiện của một quy tắc.
* Duyệt qua danh sách rules, trong đó mỗi quy tắc gồm:
* conditions: Tập điều kiện dưới dạng dictionary.
* prediction: Nhãn dự đoán tương ứng.
* Đối với mỗi quy tắc:
* Sao chép conditions vào một dictionary rule\_dict.
* Duyệt qua từng thuộc tính trong rule\_dict:
* Nếu giá trị không phải None, gọi phương thức convert\_value\_to\_label để chuyển đổi giá trị số thành nhãn tương ứng.
* Thêm nhãn dự đoán (prediction) vào rule\_dict với key 'target'.
* Thêm rule\_dict vào danh sách rules\_data.
* Sau khi xử lý tất cả các quy tắc:
* Chuyển danh sách rules\_data thành một DataFrame.
* Lưu DataFrame vào file CSV với tên được chỉ định (filename), không bao gồm cột chỉ mục (index=False).
* In ra thông báo về số lượng quy tắc đã được lưu.
* **Hàm đọc tập luật từ file CSV**



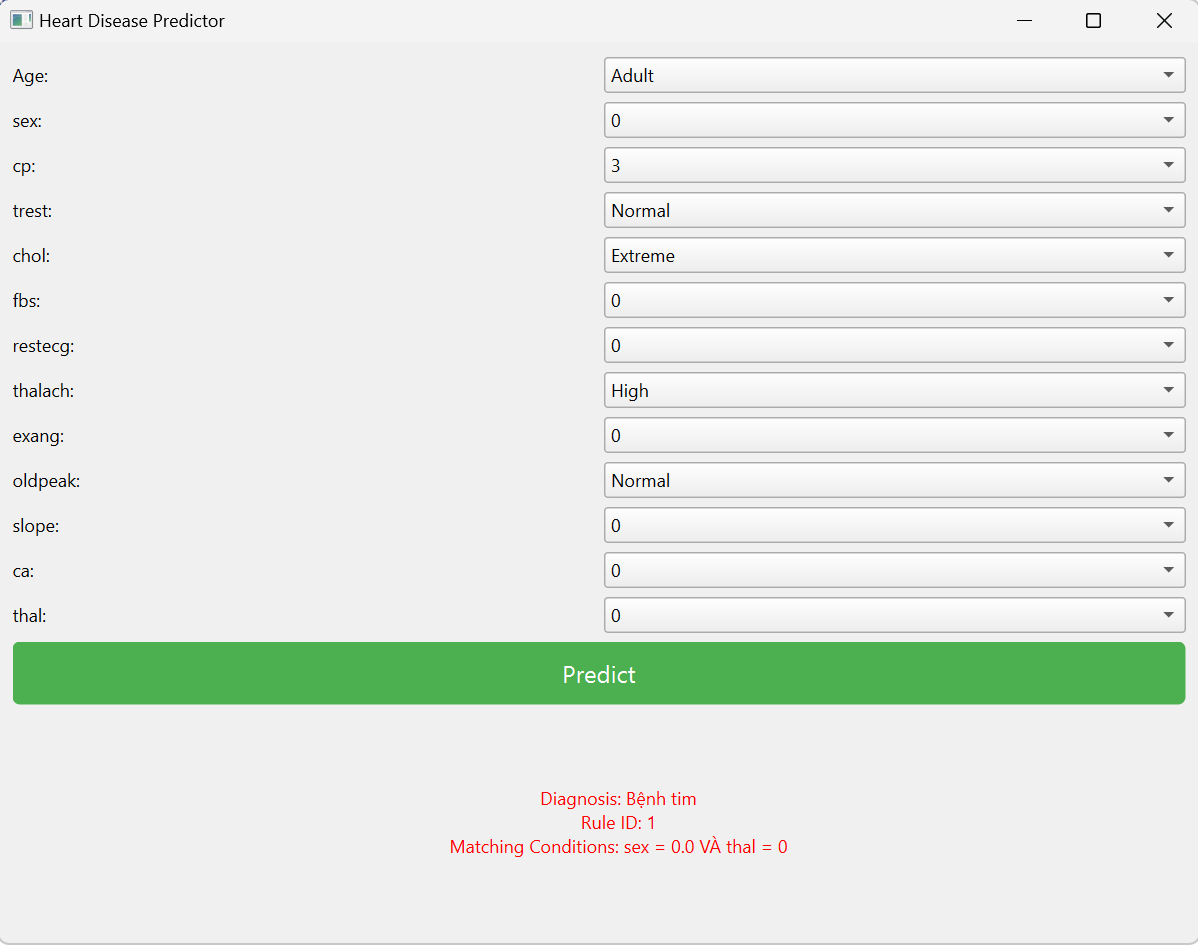
*Hình 2.9. Hàm load\_rules\_from\_csv*

* Cách hoạt động:
* Hàm bắt đầu với việc cố gắng đọc tệp CSV chứa các quy tắc từ đường dẫn được chỉ định (filename).
* Nếu tệp tồn tại: Đọc dữ liệu vào rules\_df.
* Nếu tệp không tồn tại (FileNotFoundError):
* In ra thông báo rằng không tìm thấy file quy tắc và sẽ tạo mới.
* Gọi load\_data() để tải dữ liệu gốc.
* Gọi load\_full\_attributes(data) để lấy danh sách đầy đủ các thuộc tính của dữ liệu.
* Lưu dữ liệu đã xử lý vào file result/process\_data.csv.
* Lưu bộ mã hóa nhãn vào file result/label\_encoders.json.
* Xây dựng cây quyết định bằng phương thức build\_decision\_tree(data, attributes).
* Tạo tập quy tắc từ cây quyết định bằng generate\_rules(tree, attributes).
* Lưu tập quy tắc vào file CSV bằng save\_rules\_to\_csv(rules, filename).
* Đọc lại tệp quy tắc vừa tạo bằng cách gọi load\_rules\_from\_csv(filename).
* In tập quy tắc vừa đọc được.
* Sau khi hoàn tất, in ra số lượng quy tắc đã đọc từ tệp và trả về rules\_df.
* **Hàm dự đoán từ tập luật**



*Hình 2.10. Hàm diagnose*

❖ **Giao diện chương trình**



*Hình 2.11. Giao diện chương trình*

# **KẾT LUẬN**

Trong đề tài này, chúng em đã tìm hiểu được về cây quyết định và ứng dụng của nó trong dự đoán sức khỏe thông qua việc sử dụng thuật toán ID3. Cây quyết định là một công cụ mạnh mẽ cho việc ra quyết định dựa trên các điều kiện và quy tắc quyết định. Thuật toán ID3, một trong những thuật toán quan trọng xây dựng cây quyết định, đã được chúng em phân tích chi tiết. Từ những lý thuyết tìm hiểu và phân tích được, nhóm chúng em đã tiến hành áp dụng thuật toán vào ứng dụng cho bài toán dự đoán sức khỏe. Thông qua đó, chúng em đã thực hiện viết một chương trình đơn giản giúp dự đoán người dùng có bị bệnh tim hay không thông qua những thông tin mà họ cung cấp ở trên giao diện.

Qua quá trình nghiên cứu, chúng em nhận thấy rằng ứng dụng trong dự đoán sức khỏe mang lại nhiều lợi ích. Cây quyết định giúp chúng ta hiểu rõ hơn về mối quan hệ giữa các yếu tố và dự đoán kết quả một cách chính xác. Điều này có thể hỗ trợ các chuyên gia y tế trong việc đưa ra các quyết định chuẩn đoán và điều trị. Tuy nhiên, cũng cần lưu ý rằng việc xây dựng mô hình cây quyết định đòi hỏi ta sự cân nhắc kỹ lưỡng về việc chọn lọc đặc trưng và thu thập dữ liệu đủ lớn, chính xác.

# **TÀI LIỆU THAM KHẢO**

1. Trần Hùng Cường, Nguyễn Phương Nga, Giáo trình Trí tuệ nhân tạo, 2014
2. Bài giảng học phần Trí tuệ nhân tạo, Hệ thống học kết hợp, Đại học Công nghiệp Hà Nội.
3. <https://websitehcm.com/tim-hieu-ve-decision-tree-cay-quyet-dinh/>
4. <https://media.neliti.com/media/publications/448545-applying-decision-tree-technique-to-buil-e403a0ab.pdf>
5. <https://machinelearningcoban.com/2018/01/14/id3/>
6. <https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset/data>