DỰ ĐOÁN BỆNH NHÂN MẮC BỆNH ĐỘNG MẠCH VÀNH SỬ DỤNG MÔ HÌNH MÁY HỌC

(Angiographic Coronary Heart Disease Prediction Using Machine Learning)

Le Viet Lam Quang
20520290
KHNT2020

Nguyen My Hanh
21522820
KHDL2021

Ho Thanh Tinh
20520813
KHNT2020

***Abstract*—**Trong bài viết này, chúng tôi tập trung vào nghiên cứu và so sánh một số mô hình classification phổ biến trong việc dự đoán bệnh mạch vành ở tim. Chúng tôi sử dụng bộ dữ liệu Heart Failure, một bộ dữ liệu đã từng rất phổ biến, dễ tiếp cận và có tính đại diện cho bài toán classification. Nhằm mục đích thực hiện một quy trình đầy đủ các bước từ tìm hiểu dữ liệu cho đến tối ưu hóa mô hình, thông qua việc nghiên cứu và so sánh các mô hình.

Bằng việc sử dụng các loại mô hình khác nhau Logistic Regression, Artificial Neural Network (ANN) và Adaptive Boosting (AdaBoost), chúng tôi đánh giá hiệu suất của từng mô hình dựa trên các thang điểm như F1, Accuracy, Precision và Recall trên bộ dữ liệu này. Kết quả cho thấy mô hình AdaBoost đạt hiệu suất tốt nhất trong việc dự đoán suy tim trên bộ dữ liệu Heart Failure.

Keywords—Bệnh mạch vành, máy học, classification, logistic regression, Neural Network, AdaBoosting.

I. GIỚI THIỆU

Học máy, với khả năng dự đoán và phân loại, đã trở thành một công cụ quan trọng trong lĩnh vực công nghệ thông tin. Trong ngành y tế, ứng dụng của học máy đang ngày càng phát triển, mang lại những lợi ích to lớn cho việc chuẩn đoán và điều trị các bệnh.

Trong phạm vi đề tài này, chúng tôi tập trung vào một bài toán cụ thể - dự đoán suy tim dựa trên các chỉ số chuẩn đoán y khoa của bệnh nhân. Mục tiêu là áp dụng các phương pháp học máy để xây dựng mô hình dự đoán chính xác và hỗ trợ quyết định lâm sàng trong việc đánh giá nguy cơ suy tim.

Việc xây dựng một công cụ dự đoán suy tim có thể hỗ trợ quá trình chuẩn đoán sàng lọc tại các bệnh viện. Điều này giúp cung cấp thông tin quan trọng cho các nhà điều hành y tế và bác sĩ để đưa ra quyết định chính xác về chẩn đoán và điều trị.

Input: *các chỉ số chuẩn đoán y khoa của một bệnh nhân gồm :tuổi, giới tính, lượng mỡ máu , chế độ tập luyện,…*
Output*: dự đoán bệnh nhân đó có bị suy tim hay không. Áp dụng vào thực tiễn: công cụ có áp dụng phương pháp các máy học dùng để hỗ trợ chuẩn đoán sàng lọc ở các bệnh viện*

II. DATASET

A. Giới thiệu Dataset

Dữ liệu là yếu tố quyết định và bắt buộc trong mọi giả pháp Máy Học. Dataset của chúng tôi được thu thập từ website Kaggle[1]. Dataset được xây dựng bằng cách kết hợp nhiều các nguồn dữ liệu khác sẵn có, với độ uy tín cao. Với việc tổng hợp từ 5 nguồn dữ liệu (bao gồm tập dữ liệu Heart Disease của UCI) và loại bỏ những dữ liệu trùng lặp, dataset cuối cùng sẽ có 918 dòng dữ liệu và 12 thuộc tính (bao gồm cả thuộc tính nhãn).

B. Mô tả Dataset

Bộ dữ liệu sẽ được trình bày dưới dạng bảng thống kê (file csv) và thông tin của các thuộc tính như sau:

- **Age**: Tuổi của bệnh nhân [Năm]
- **Sex:** giới tính của bệnh nhân [M: Nam, F: Nữ]
- **ChestPainType**: loại đau ngực [TA: Đau thắt ngực điển hình, ATA: Đau thắt ngực không điển hình, NAP: Không đau, ASY: Không có triệu chứng]
- **RestingBP**: huyết áp khi nghỉ ngơi [mm Hg]
- **Cholesterol**: Cholesterol trong huyết thanh [mm/dl]
- **FastingBS**: lượng đường trong máu [1: Nếu FastingBS > 120 mg/dl, 0: Còn lại]
- **RestingECG**: kết quả điện tâm đồ khi nghỉ ngơi [Bình thường: Bình thường, ST: Có sóng ST-T bất thường (sóng T đảo ngược và/hoặc ST chênh lên hoặc xuống > 0,05 mV), LVH: Cho thấy phì đại thất trái có thể xảy ra hoặc chắc chắn theo tiêu chí của Estes]
- **MaxHR**: nhịp tim tối đa đạt được [Giá trị số từ 60 đến 202]
- **ExerciseAngina**: đau thắt ngực do gắng sức [Y: Có, N: Không]
- **Oldpeak**: ST chênh xuống [Giá trị số được do gắng sức so với nghỉ ngơi]
- **ST_Slope**: độ dốc của đoạn ST trong kết quả điện tâm đồ cao nhất [Up: Dốc lên, Flat: Bằng phẳng, Down: Dốc xuống]
- **HeartDisease**: Chuẩn doán suy tim [1: Suy tim, 0: Bình thường]

Do đây cũng là một bài toán về chuẩn đoán vấn đề liên quan đến y khoa, vì vậy nguồn dữ liệu phải yêu cầu cần có độ uy tín và chuẩn xác cao. Những đặc trưng có trong bộ dữ liệu chính là những đặc trưng sẽ góp phần liên quan việc dự đoán suy tim đã qua những nghiên cứu y học chứng minh.

C. Khai phá Dataset

Áp dụng những phương pháp để phân tích bộ dữ liệu (EDA), nhóm đã rút ra được một số nhận xét sau đây: (những ảnh về số liệu cũng như biểu đồ phân tích dữ liệu tương ứng với từng nhận xét có trong phần code của đồ án)

Comment 1: Bộ dữ liệu có 918 quan sát, trong đó gồm 11 thuộc tính với 5 thuộc tính là dạng Numerical, 7 thuộc tính là Categorical vì vậy cần phải thực hiện encode để đưa các thuộc tính Categorical về dạng Numerical, các thuộc tính đó gồm:

Sex, **ChestPainType**, **FastingBS**, **RestingECG**, **ExerciseAngina**, **ST_Slope**.

Comment 2: Bộ dữ liệu không có thuộc tính nào có dữ liệu bị thiếu.

Comment 3: Phân bố của từng các đặc trưng dữ liệu:

- Các numerical features: Age, RestingPB, MaxHR có phân phối gần giông với phân phối chuẩn.
- Thuộc tính Cholesterol bị nhiễu ở giá trị Cholesterol=0 nên không tuân theo phân phối chuẩn. Có thể ta sẽ bỏ đi phần nhiễu đó để Cholesterol tuân theo phân phối chuẩn.

Comment 4: Có thể nói đây là một bộ dữ liệu không bị mất cân bằng, tỉ lệ của hai class target gần như là cân bằng, phân bố bị lệch cho nhãn 1: Có bệnh mạch vành, nhưng với tỉ lệ nhỏ (tỉ lệ 0.553:0.447).

Comment 5: Xét trong những trường hợp bị bệnh tim có:

- Hơn 90% là nam giới --> Người nam dễ bị bệnh tim mạch hơn là người nữ.
- Hơn 77% quan sát thuộc loại ASY (Không có triệu chứng) ở thuộc tính ChestPainType.
- 2/3 số quan sát có Fasting Blood Sugar level < 120 mg/dl.
- Hơn 1/2 quan sát thuộc Normal level (RestingECG). Với mỗi giá trị trong RestingECG, tỉ lệ mắc bệnh không có sự khác biệt nhiều nên không giúp phân tách các trường hợp bệnh tim mạch tốt.
- Hơn 62% quan sát có đau thắt ngực do gắng sức (Exercise Angina: Angina).
- 3/4 số quan sát có độ dốc của đoạn ST trong kết quả điện tâm đồ cao nhất là bằng phẳng (Flat). Với giá trị Down thì cũng có nhiều trường hợp bị bệnh tim, nhưng có khá ít điểm dữ liệu mang giá trị down.

Comment 6: Từ biểu đồ correlation heatmap của tất cả thuộc tính so với thuộc tính nhãn HeartDisease, ta thấy ResingBP và RestingECG có tương quan yếu với HeartDisease so với các thuộc tính khác (lần lượt là 0.11, 0.057). Nên nhóm sẽ xem xét loại bỏ 2 đặc trưng này khỏi bộ dữ liệu.

D. Tiền xử lý Dataset

Qua bước phân tích bộ dữ liệu để đưa ra các hướng giải quyết trên, chúng tôi sẽ thực hiện loại bỏ một số đặc trưng **RestingBP**, **RestingECG**, chỉ giữ lại các đặc trưng **Age**, **Sex,** **ChestPainType**, **Cholesterol**, **FastingBS**, **MaxHR**, **ExerciseAngina**, **Oldpeak**, **ST_Slope**. Ngoài ra nhóm còn cần đưa các thuộc tính *Categorical* là **Sex**, **ChestPainType**, **FastingBS**, **RestingECG**, **ExerciseAngina**, ST_Slope về *Numerical*. Công cụ hỗ trợ mà nhóm sẽ sử dụng cho bước này sẽ là *LabelEncoder* của thư viện *Sklearn*.

III. CÁC MÔ HÌNH VÀ PHƯƠNG PHÁP

A. *Logistic Regression:*

Giới thiệu: được phát triển bởi nhà toán học người pháp P.Laplace vào thế kỷ 19. Mô hình Logistic Regression là một kỹ thuật thống kê để xem xét mối liên hệ giữa biến độc lập (biến số hoặc biến phân loại) với biến phụ thuộc là biến nhị phân. Trong thuật toán này, biến phụ thuộc y chỉ có hai trạng thái ví dụ 1 (dương tính) hoặc 0 (âm tính). Muốn đổi ra biến số liên tục người ta tính xác suất của hai trạng thái này.

Bằng cách nói ngắn gọn về định nghĩa của mô hình này như trên, điểm đặc biệt của nó là: mô hình Logistic Regression được thiết kế đặc biệt để phân loại dữ liệu vào hai nhóm khác nhau. Ngoài ra, Mô hình Logistic Regression cung cấp khả năng dự đoán xác suất xảy ra của sự kiện nhị phân và mô tả mối quan hệ tuyến tính giữa biến độc lập và biến phụ thuộc dựa trên hàm sigmoid.

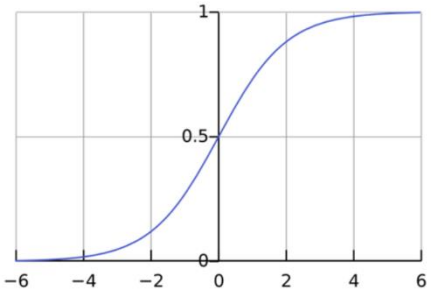
1. *Chi tiết thuật toán*

a. Xây dựng hàm Sigmoid

Đầu tiên, cần chuyển đổi giá trị được đưa vào thành một giá trị nằm trong khoảng từ 0 đến 1.

Công thức toán học của hàm Sigmoid là: $\mathbf{f(s)=\frac{1}{1+e^{-s}}}$

(s là giá trị đầu vào của hàm)



b. Xây dựng mô hình logistic

Mô hình hồi quy logistic sử dụng một hàm tuyến tính để tính toán giá trị z, tổng của các đặc trưng nhân với các tham số tương ứng.

Công thức tính z được tính như sau:

$$\boldsymbol{z = w_0 + w_1x_1 + w_2x_2 + \cdots + w_nx_n}$$

Trong đó:

- $w_0, w_1, w_2, \dots, w_n$ là các tham số mô hình cần tìm kiếm
- $x_0, x_1, x_2, \dots, x_n$ là các đặc trưng của mẫu dữ liệu

c. Xây dựng hàm mất mát

Mô hình hồi quy Logistic sử dụng hàm mất mát để đo lường sự sai lệch giữa dự đoán mô hình và giá trị thực tế. Hàm mất mát phổ biến trong mô hình này là hàm Cross-Entropy (Log Loss)

$$J(\Theta) = -\sum_{i=1}^{m}[(y^{(i)}logp^{(i)}) + (1-y^{(i)})log(1-p^{(i)})]$$

Trong đó:

- $y^{(i)}$ là nhãn thực tế của mẫu dữ liệu (0 hoặc 1)
- $p^{(i)}$ là dự đoán được tính bởi mô hình, tức là xác suất dự đoán rằng mẫu dữ liệu thuộc vào lớp dương (1).

d. Tối ưu hàm mất mát

Sử dụng phương pháp Stochastic Gradient Descent (SGD) để tối ưu hàm mất mát.

2. *Các siêu tham số của thuật toán*

- Penalty: đây là tham số điều chỉnh regularization - kiểm soát mức độc phức tạp của mô hình và giảm hiện tượng overfitting.

  • Hệ số điều chỉnh (C): đây là siêu tham số có liên quan đến regularization trong logistic regression.

Giá trị C nhỏ sẽ tạo ra môi trường có regularization mạnh và ngược lại.

• Solver: Logistic Regression sử dụng các phương pháp tối ưu để tìm ra bộ trọng số tốt nhất. Có nhiều phương pháp tối ưu như là 'newton-cg', 'lbfgs', 'liblinear', 'sag' và 'saga'.

• max_iter: số lần lặp quá trình tối ưu để tìm ra bộ trọng số tốt nhất. Nếu quá trình tối ưu không hội tụ trước khi đạt đến số lượng vòng lặp tối đa, nó sẽ dừng và trả lại kết quả tốt nhất tìm được cho tới thời điểm đó.

3. *Đánh giá thuật toán*

Ưu điểm:

• Đơn giản và dễ hiểu
• Có thể sử dụng với nhiều loại dữ liệu và phân loại nhãn đầu ra.
• Có thể xử lí đặc trưng liên tục và rời rạc
• Không đòi hỏi tính toán tài nguyên

Nhược điểm:

• Với mối quan hệ phi tuyến, logistic regression hoạt động không hiệu quả.
• Nhạy cảm với nhiễu và giá trị cực đại.
• Khó cho kết quả tốt khi dữ liệu mất cân bằng

4. *Tổng kết*

Khi nhắc đến Logistic Regression, điều cần nhớ là:
• Nó dựa trên hàm sigmoid để biến đổi giá trị đầu vào.
• Là mô hình tuyến tính
• Dữ liệu huấn luyện được sử dụng để ước lượng các tham số thông qua phương pháp tối ưu như: Gradient Descent

B. *Neural Network:*

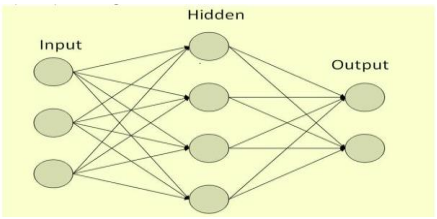
1. *Tổng quan*

a. *Giới thiệu về Neural Network*

Là một mô hình tính toán bắt chước cách thức hoạt động của các tế bào thần kinh trong não người. Mạng nơ-ron nhân tạo (ANN) sử dụng các thuật toán learning có thể thực hiện các điều chỉnh một cách độc lập – hoặc học theo một nghĩa nào đó – khi chúng nhận được giá trị input mới.
Có rất nhiều loại mạng Neural Network (NN), được sử dụng trong thực tế, ngoài mạng ANN là mạng cơ bản nhất còn có một số mạng NN phức tạp hơn như: CNN, RNN, LSTM, GAN,

Mạng Neural Network là sự kết hợp của những tầng perceptron hay còn gọi là perceptron đa tầng. Và mỗi một mạng Neural Network thường bao gồm 3 kiểu tầng là:

- Tầng input layer (tầng vào): Tầng này nằm bên trái cùng của mạng, thể hiện cho các đầu vào của mạng.
- Tầng output layer (tầng ra): Là tầng bên phải cùng và nó thể hiện cho những đầu ra của mạng.
- Tầng hidden layer (tầng ẩn): Tầng này nằm giữa tầng vào và tầng ra nó thể hiện cho quá trình suy luận logic của mạng.



Hình 37. Mô hình ANN đơn giản

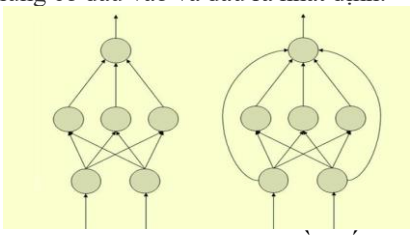
Lưu ý: Mỗi một Neural Network chỉ có duy nhất một tầng vào và 1 tầng ra nhưng lại có rất nhiều tầng ẩn.

Trong ANN, trừ input layer thì tất cả các node thuộc các layer khác đều full-connected với các node thuộc layer trước nó. Mỗi node thuộc hidden layer nhận vào ma trận đầu vào từ layer trước và kết hợp với trọng số để ra được kết quả.

2. *Các loại mạng trong Neural Network.*

Mạng lan truyền tiến (Feed Forward)

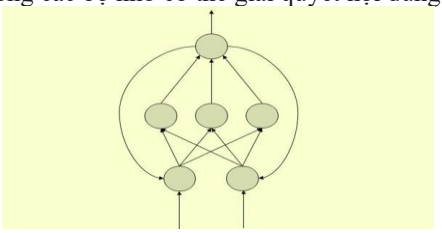
Mạng lan truyền tiến (FeedForward) có luồng thông tin 1 chiều, một đơn vị sẽ được sử dụng để gửi thông tin cho một đơn vị khác mà không nhận được bất kỳ thông tin nào. Chúng không có vòng phản hồi, thường dùng để nhận dạng một mẫu cụ thể vì chúng có đầu vào và đầu ra nhất định.



Hình 38. Mạng lan truyền tiến

Mạng lan truyền ngược (Back Forward)

Trong mạng Nơron nhân tạo này, chúng sẽ cho phép các vòng lặp phản hồi. Chúng ta thường sử dụng mô hình này trong các bộ nhớ có thể giải quyết nội dung.



Hình 39. Mô hình Feedback ANN

3. *Cách hoạt động của Neural Network*

Mạng neural nhân tạo có khả năng sử dụng được như một loại cơ chế xấp xỉ hàm tùy ý mà học được từ việc dữ liệu quan sát. Tuy nhiên, việc sử dụng chúng khá khó và cần phải có sự hiểu biết tương đối về những lý thuyết cơ bản về mạng nơron này.

Lựa chọn mô hình: Phụ thuộc vào cách trình bày dữ liệu và các ứng dụng của nó. Đây là mô hình khá phức tạp nên có thể dẫn đến nhiều thách thức cho quá trình học.

Thuật toán học: Thường sẽ có rất nhiều thỏa thuận giữa các thuật toán học. Và hầu hết, chúng sẽ làm việc tốt với những tham số đúng nhằm huấn luyện trên dữ liệu mà không nhìn thấy yêu cầu một số lượng đáng kể các thử nghiệm.

Mạnh mẽ: Nếu như các mô hình, thuật toán học và hàm chi phí được lựa chọn một cách thích hợp thì Neural Network có thể cho ra kết quả vô cùng hợp lý.

4. *Ưu và nhược điểm của Neural Nerwork*

Ưu điểm:

- Phân tích dữ liệu một cách trực quan hiệu quả: Mạng nơ ron nhân tạo tương tự như mạng thần kinh của con người nên nó có thể xử lý một số các tác vụ phức tạp hơn so với các phương pháp khác. Mạng neural có thể phân tích các dữ liệu một cách hiệu qủa và tách chúng thành nhiều loại khác nhau.
- Xử lý hiệu quả các dữ liệu không có cấu trúc.
- Cấu trúc có tính thích ứng: bất kỳ mục đích nào mà ANN được áp dụng, nó sẽ thay đổi tiến trình cấu trúc của nó theo mục đích đó.

Nhược điểm:

- Yêu cầu phần cứng.
- Cấu trúc phức tạp và khó triển khai.
- Cần nhiều dữ liệu: Dữ liệu sẽ tỉ lệ thuận với độ chính xác.

5. Ứng dụng của Neural Network

Mạng ANN hiện nay được ứng dụng rất nhiều trong rất nhiều ngành, nó có thể làm những việc mà con người gặp khó khăn một cách rất dễ dàng. Một số lĩnh vực sử dụng mạng nơron nhân tạo như: Hàng không vũ trụ, phần mềm, giao thông vận tải, viễn thông.

C. Adaptive Boosting:

1. Giới thiệu

AdaBoost (Adaptive Boosting) là một thuật toán Machine Learning thuộc nhóm bài toán Boosting, nằm trong nhóm các phương pháp Ensemble Learning, nghĩa là kết hợp nhiều mô hình học máy độc lập với nhau để cho ra kết quả cuối cùng. Thuật toán này có thể sử dụng cho cả hai bài toán Classification và Regression, nhưng thường là Classification, dùng để giải quyết các bài toán thuộc nhóm "Học có giám sát" (Supervised Learning).

Điểm đặc biệt của thuật toán này đó là nó tập trung vào những điểm dữ liệu khó (bị dự đoán sai) từ mô hình trước để cải tiến mô hình sau đó nhằm dự đoán được những điểm dữ liệu này). Việc học tuần tự qua nhiều mô hình cơ sở (base model hoặc weak-learner) có độ quan trọng khác nhau sẽ tạo thành một mô hình dự đoán tốt (strong-learner) cuối cùng.

2. Chi tiết thuật toán

Nguyên lý hoạt động của AdaBoost sẽ gồm các bước như sau:

a. Gán trọng số cho tất cả điểm dữ liệu

Đầu tiên, thuật toán sẽ gán cho mỗi điểm dữ liệu trong bộ dữ liệu một trọng số (gọi là sample weights) với giá trị độ lớn như nhau theo công thức:

$$w_i = \frac{1}{N}, \forall i = 1, N$$

trong đó: N là số điểm dữ liệu

Với công thức trên thì giá trị trọng số của mỗi sample sẽ nằm trong khoảng (0, 1) và tổng độ lớn của chúng luôn bằng 1.

b. Khởi tạo bộ dự đoán (mô hình) để dự đoán dữ liệu

Đối với thuật toán AdaBoost, mỗi mô hình nhỏ (weak-learner) thường sẽ là những mô hình đơn giản (nhưng vẫn có thể dự đoán tốt hơn dự đoán ngẫu nhiên) và độc lập nhau như: Linear Regression, Decision Tree, … Nhưng thường dùng nhất vẫn là Decision Tree với chỉ 1 node gốc và 2 node lá (độ sâu của cây là 1) được gọi là stump (tạm dịch: gốc). Lý do vì sao lại chọn cấu trúc như này sẽ được giải thích ở phần cuối.

Để chọn được giá trị và thuộc tính phân chia trong stump, tương tự như trong mô hình Decision Tree, ta có thể dùng các giá trị Entropy – Information Gain hoặc Gini Index để chọn được thuộc tính phân chia dữ liệu tốt nhất. Nếu sử dụng Gini Index thì ta sẽ chọn thuộc tính có Gini nhỏ nhất để làm thuộc tính phân loại.

c. Tính độ lỗi (total_error hay r) của mô hình:

Giá trị độ lỗi của mô hình chính là tổng trọng số (sample weights) của những điểm dữ liệu bị dự đoán sai.

d. Tính trọng số cho mô hình (Amount of Say)

Trọng số của mô hình(nhỏ) là giá trị quyết định độ quan trọng của mô hình này đối với mô hình lớn cuối cùng, được tính theo công thức sau:

$$\alpha = \eta * \log \frac{1-r}{r}$$

Trong đó:

- α: Trọng số của mô hình
- η: Tốc độ học (learning rate, thuật toán AdaBoost gốc sử dụng giá trị $\frac{1}{2}$).
- r: Độ lỗi của mô hình

Với trường hợp r = 0, dựa theo công thức ta sẽ không tính toán được, nên công thức thực tế thường được cộng thêm một giá trị để tránh trường hợp chia số 0.

e. Cập nhật trọng số sample weights cho các điểm dữ liệu

Do ta muốn mô hình sau tập trung vào những điểm bị dự đo án sai ở mô hình này, nên ta cần trọng số của những điểm sai sẽ lớn, còn những điểm đúng sẽ nhỏ hơn. Vì vậy ta sử dụng công thức:

$$w_i = \begin{cases} w_i * e^{-\alpha}, khi\ điểm\ dữ\ liệu\ được\ dự\ đoán\ đúng \\ w_i * e^{\alpha}, khi\ điểm\ dữ\ liệu\ được\ dự\ đoán\ sai \end{cases}$$

Nếu tổng của các trọng số chưa bằng 1, ta cần bước chuẩn hóa các trọng số theo công thức:

$$w_i = \frac{w_i}{\sum_{i=1}^{N} w_i}$$

f. Khởi tạo mô hình tiếp theo dựa trên các trọng số từ mô hình trước

Mô hình sau sẽ tập trung hơn vào các điểm dữ liệu sai (trọng số lớn) trong mô hình trước bằng 2 cách:

- Một là giữ nguyên bộ dữ liệu, nhưng thay vì tính Gini Index để chọn thuộc tính phân loại cho stump, ta sẽ tính Weighted Gini Index, nghĩa là điểm Gini có kèm trọng số từ mô hình trước.
- Hai là tạo bộ dataset mới (bootstrap dataset) chứa phần lớn là các điểm dữ liệu sai nhờ vào các trọng số.

Cả 2 cách này đều giúp mô hình sau tập trung dự đoán đúng những điểm dữ liệu bị dự đoán sai từ mô hình trước. Và ta sẽ lặp lại tất cả các bước trên cho đến khi không có dòng dữ liệu nào bị dự đoán sai hoặc số mô hình được tạo ra đạt một mốc nhất định.

g. Cách thuật toán AdaBoost dự đoán nhãn một sample mới

Khi dự đoán, dữ liệu sẽ được đưa qua tất cả các mô hình con trong thuật toán và ghi lại kết quả của từng mô hình (ở ví dụ này là có bệnh hoặc không, 0 hoặc 1). Sau đó ta tính tổng trọng số của các mô hình ứng với kết quả mà nó dự đoán, rồi chọn tổng nào lớn hơn thì kết quả sẽ là nhãn đó.

$$P = argmax_k \sum_{\substack{j=1 \\ Predict\ (x)_j = k}}^{M} \alpha_j,$$

trong đó M là số mô hình

3. *Các siêu tham số trong mô hình:*

- **base_estimator**: chính là các mô hình nhỏ mà ta chọn để huấn luyện mô hình chung. Các mô hình thường được sử dụng là Decision Tree, Linear Regression, SVM, …
- **n_estimators**: số lượng mô hình con được tạo ra và sử dụng trong AdaBoost.
- **learning_rate**: tốc độ học của từng mô hình con. Nếu learning_rate nhỏ thì n_estimators lớn và ngược lại.

Lưu ý rằng ở mỗi mô hình con cũng sẽ có các siêu tham số tương ứng ta cần chọn. Để tránh trường hợp quá khớp (overfitting), người ta thường chọn các mô hình con có cấu trúc đơn giản, hoặc điều chỉnh cấu trúc nếu mô hình con quá phức tạp, hoặc giảm số lượng mô hình cơ sở trong quá trình huấn luyện.

4. *Đánh giá thuật toán:*

Ưu điểm:

- Dễ sử dụng và ít điều chỉnh tham số.
- Sử dụng base model là Decision Tree (độ sâu = 1) sẽ mô hình ít bị overfitting hơn do mỗi mô hình chỉ có 1 node gốc và 2 node lá (lí giải lý do chọn cấu trúc này).
- Cho ra kết quả tốt hơn các mô hình cơ bản, riêng lẻ như Linear, Logistic Regression, Decision Tree, … do đặc tính cải thiện khuyết điểm qua từng mô hình.
- Hiện nay đã được phát triển để phân loại hình ảnh và văn bản.

Nhược điểm:

- Phụ thuộc nhiều vào bộ dữ liệu, đòi hỏi bộ dữ liệu phải sạch và ít điểm gây nhiễu vì ý tưởng chính của AdaBoost là khớp từng điểm dữ liệu một cách hoàn hảo.
- Do sử dụng phương pháp học tuần tự (sequencial), nên ta chỉ có thể huấn luyện bộ dự đoán mới chỉ khi đã hoàn thành huấn luyện và đánh giá bộ dự đoán trước đó. Vì vậy, nhìn chung kĩ thuật này sẽ không mở rộng tốt bằng các thuật toán Ensemble khác (bagging, pasting).
- Đây là thuật toán cơ bản và đầu tiên trong nhánh Boosting nên ngoài nó còn có các thuật toán khác có hiệu quả và tốc độ xử lý nhanh hơn như GradientBoosting hay XGBoost, …

5. *Tổng kết*

Khi nhắc đến thuật toán Adaptive Boosting (AdaBoost), có 3 đặc điểm chính ta cần nhớ:

- AdaBoost tổng hợp nhiều mô hình con (nhỏ, yếu) lại với nhau để đưa ra kết quả dự đoán. Những mô hình con sẽ có cấu trúc rất đơn giản, thường dùng nhất là DecisionTree với cấu trúc "stump" (gốc cây) với 1 node gốc và 2 node lá.
- Mỗi mô hình con trong Adaboost sẽ có độ ảnh hưởng (Amount of Say) khác nhau đến kết quả cuối cùng.
- Các mô hình con được tạo ra lần lượt, nối tiếp nhau, với mô hình sau sẽ tập trung hơn vào lỗi từ mô hình trước.

IV. EVALUATION:

Trong bài toán phân loại bệnh này, nhóm sử dụng những độ đo thông dụng cho các bài toán Classification, cụ thể gồm: **Accuracy**, **Precision**, **Recall** và **F1-score**. Những metrics này đều rất hữu dụng trong việc đánh giá độ hiệu quả của mô hình, góp phần giải quyết bài toán.

Ngoài ra, do bộ dữ liệu nhỏ nên nhóm dùng K-Fold Cross Validation, với k = 5 để cải thiện kết quả của các mô hình.

A. *Logistic Regression:*

Trong mô hình Logistic Regression, chúng ta có thể điều chỉnh hiệu suất và độ chính xác của mô hình bằng cách áp dụng các siêu tham số phù hợp.

Kết quả thực nghiệm: (đơn vị %)

C	Penalty	Solver	ACC	Precision	Recall	F1_score
0.01	**L1**	**saga**	**84.06**	**84.47**	**83.55**	**83.78**
0.01	L1	liblinear	78.62	78.53	78.69	78.56
0.1	L1	liblinear	83.33	83.22	83.34	83.26
0.1	L2	liblinear	82.61	82.50	82.50	82.50
0.01	L2	lbfgs	81.16	81.03	81.10	81.06

Nhận xét:

Bộ mô hình này đạt độ chính xác (ACC) cao nhất là 84.06%, cho thấy việc sử dụng mức độ regularization cao (Regularization L1) và thuật toán tối ưu saga có thể cải thiện hiệu suất của mô hình Logistic Regression trong bài toán được đề cập.

B. *Neural Network:*

Giới thiệu thư viện và tham số: Xem thêm về tài liệu tại đây.

Trong Keras, mô hình **Sequential** cho phép người dùng để xây dựng các mạng neural network có kiến trúc đơn giản, liên tiếp các layer với nhau theo thứ tự.

Sử dụng các phương thức thao tác cơ bản như add, compile, fit, predict.

Kết quả thực nghiệm: (đơn vị %)

Số hidden layer	Số node mỗi hidden layer	Hàm kích hoạt của hidden layer	ACC	Precision	Recall	F1_score
1	(6)	Relu	85.06	83.75	82.99	83.33
2	**(6,6)**	**Relu**	**86.70**	**84.99**	**85.51**	**85.19**
3	(6,6)	Sigmoid	84.30	83.14	82.03	82.45
4	(6,10)	Relu	84.08	83.47	81.68	82.28

5	(6,10)	Sigmoid	84.30	83.14	82.03	82.45
6	(6,15)	Relu	85.17	84.31	82.78	83.41
7	(6,6,10)	Relu	83.43	81.75	81.37	81.43
8	(6,10,10)	Relu	85.50	83.74	83.95	83.82
9	(6,10,10,10)	Relu	86.37	84.94	84.84	84.82

Nhận xét :

- Ta thấy với bộ dữ liệu heart.csv, mô hình có độ chính xác khi có 2 tầng ẩn với số node là 6, 6.
- Ta thấy mô hình có xu hướng tăng độ chính xác khi tăng số node hoặc số tầng ẩn.

C. Adaptive Boosting:

Thư viện Sklearn hỗ trợ đầy đủ cho thuật toán AdaBoost với hàm AdaBoostClassifier, những mô hình con sử dụng trong bài toán này để thực nghiệm sẽ là **Decision Tree** với max_depth của Tree sẽ là 1 nghĩa là chỉ có 1 node gốc và 2 node lá, và **Support Vector Machine,** sử dụng mô hình với các siêu tham số mặc định,

Kết quả thực nghiệm: (đơn vị %)

Decision Tree:

Số model	Accuracy	Precision	Recall	F1_score
1	92.93	92.07	94.35	92.70
10	92.39	91.57	93.91	92.15
50	90.22	89.31	90.14	89.68
100	88.59	87.91	87.68	87.79
500	87.50	86.74	86.52	86.63

SVM:

Số model	Accuracy	Precision	Recall	F1_score
1	84.24	84.01	81.88	82.69
10	87.50	86.74	86.52	86.63
50	88.59	87.58	88.55	87.99
100	89.13	88.12	89.28	88.59
500	37.50	18.75	50.00	27.27

Nhận xét:

- Với mô hình con là DT, kết quả đạt được tốt hơn hẳn so với mô hình con là SVM.
- Kết quả tốt nhất của DT là với chỉ 1 mô hình con, accu đạt 92.93% và f1 đạt 92,70%. Còn kết quả tốt nhất của SVM là với 100 mô hình con, với accu là 89.13% và f1 là 88.59%

V. ĐÁNH GIÁ TỔNG QUAN VÀ HƯỚNG PHÁT TRIỂN

Từ kết quả thực nghiệm của 3 thuật toán Máy học trên, nhóm nhận thấy thuật toán AdaBoost đạt kết quả cao nhất, với chỉ duy nhất 1 mô hình con là Decision Tree với cấu trúc 1 node gốc 2 node lá. Vấn đề là vì sao một mô hình đơn giản lại đạt kết quả tốt hơn nhiều so với những mô hình phức tạp khác (nhiều mô hình con hoặc mô hình ANN). Từ đây nhóm rút ra được những lý do sau:

1) Bộ dữ liệu nhỏ, đơn giản (918 dòng dữ liệu, không có dữ liệu thiếu).

2) Trong dữ liệu có 1 thuộc tính có tính phân loại rất tốt, nghĩa là chỉ cần dựa vào 1 thuộc tính này là ta đã có thể phân loại bệnh nhân có bệnh hay không với độ chính xác cao.

Với những lý do này, thì hướng phát triển nhóm đề ra sẽ là dùng những thuật toán này trên các bộ dữ liệu khác lớn, phức tạp hơn, liên quan đến bệnh mạch vành để cải thiện khả năng tổng quát hóa dữ liệu mới của mô hình, phát triển thành một công cụ dự đoán bệnh lý này tốt hơn.

VI. TỔNG KẾT

Chúng tôi đã thực hiện một dự án nhằm tìm hiểu và làm quen với việc sử dụng máy học trong phân loại bệnh tim. Bằng cách sử dụng kết quả chuẩn đoán y khoa và áp dụng phương pháp máy học, chúng tôi đã xây dựng thành công một hệ thống dự đoán suy tim. Đây là một bước quan trọng và đưa ra cái nhìn tổng quan về tiềm năng của máy học trong y tế. Tuy nhiên, còn rất nhiều công việc nghiên cứu và cải tiến phải được thực hiện để áp dụng bài toán này vào thực tế. Chúng tôi xin gửi lời cảm ơn đến các thành viên trong nhóm và viện nghiên cứu vì sự đóng góp và hỗ trợ trong quá trình thực hiện dự án.

- Lê Viết Lâm Quang (Nhóm trưởng): Mô hình AdaBoost
- Hạnh: Mô hình Logistic Regression
- Hồ Thanh Tịnh: Mô hình Neural Network

Link tổng hợp source code:

https://drive.google.com/drive/folders/

I. TRÍCH DẪN

[1] Heart Failure Prediction Dataset | Kaggle
[2] Machine Learning cơ bản (machinelearningcoban.com)
[3] Types of Regularization in Machine Learning | by Aqeel Anwar | Towards Data Science
[4] Implement Logistic Regression with L2 Regularization from scratch in Python | by Tulrose Deori | Towards Data Science
[5] What is Logistic regression? | IBM
[6] sklearn.linear_model.LogisticRegression — scikit-learn 1.2.0 documentation
[7] Machine Learning cơ bản (machinelearningcoban.com) https://blog.paperspace.com/adaboost-optimizer/.
[8] https://viblo.asia/p/tong-quan-ve-artificial-neural-network-1VgZvwYrlAw
[9] https://viettelidc.com.vn/tin-tuc/cam-nang-ai-artificial-neural-network-la-gi-cau-truc-cach-hoat-dong-va-ung-dung-cua-mo-hinh-nay
[10] https://www.mygreatlearning.com/blog/adaboostalgorithm/#:~:text=AdaBoost%20algorithm%2C%20short%20for%20Adaptive,assigned%20to%20incorrectly%20classified%20instances.
[11] https://www.analyticsvidhya.com/blog/2021/09/adaboost-algorithm-a-complete-guide-for-beginners/
[12] https://viblo.asia/p/adaboost-buoc-di-dau-cua-boosting-gAm5yrGwKdb
[13] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html