HỌC VIỆN HÀNG KHÔNG VIỆT NAM

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO TIỂU LUẬN
KHAI THÁC DỮ LIỆU

"CHẨN ĐOÁN BỆNH TIM SỬ DỤNG MÔ HÌNH PHÂN LỚP KNN, DECISION TREE VÀ RANDOM FOREST"

HỌC KỲ 2 – NĂM HỌC: 2023 - 2024

MÃ LỚP HỌC PHẦN: 010100087302

Giảng viên hướng dẫn: Th.S Trần Anh Tuấn

Nhóm sinh viên thực hiện: Lê Cao Tấn Lộc - 2154810086

Trịnh Vinh Qui - 2154810110

Phan Tường Bảo Trâm – 2154810077

TP. HCM, tháng 3 năm 2024

MỤC LỤC

MỤC LỤC HÌNH ẢNH

NHẬN XÉT CỦA GIẢNG VIÊN

ĐÁNH GIÁ:.....

.....

.....

ĐIỂM SỐ:.....

ĐIỂM CHỮ:.....

Chữ ký của giảng viên

CHƯƠNG I. GIỚI THIỆU

1. Đặt vấn đề.

Trong dự án này, chúng em đi sâu vào tập dữ liệu bao gồm nhiều số liệu sức khỏe khác nhau của bệnh nhân tim, bao gồm tuổi, huyết áp, nhịp tim, v.v. Mục tiêu của chúng em là phát triển một mô hình dự đoán có khả năng xác định chính xác những người mắc bệnh tim. Do những tác động nghiêm trọng của việc bỏ lỡ chẩn đoán dương tính, trọng tâm chính của chúng em là đảm bảo rằng mô hình xác định được tất cả các bệnh nhân tiềm năng, khiến việc thu hồi nhóm bệnh nhân dương tính với bệnh tim trở thành một thước đo quan trọng.

Một điểm đặc biệt quan trọng của dự án là sự cần thiết phải đảm bảo rằng mô hình của chúng em có khả năng xác định chính xác những người mắc bệnh tim. Sự chính xác của mô hình không chỉ giúp giảm thiểu rủi ro cho bệnh nhân mà còn giúp tối ưu hóa tài nguyên y tế bằng cách hướng dẫn quy trình chẩn đoán và điều trị. Điều này đặc biệt quan trọng trong bối cảnh tài nguyên y tế hạn chế, nơi mà việc phân loại chính xác bệnh nhân có nguy cơ mắc bệnh tim sẽ giúp tăng cường khả năng điều phối và sử dụng tài nguyên một cách hiệu quả nhất.

Do đó, mục tiêu cuối cùng của chúng em không chỉ là xây dựng một mô hình dự đoán, mà còn là tạo ra một công cụ hữu ích và đáng tin cậy cho các chuyên gia y tế, góp phần vào việc nâng cao chất lượng chăm sóc sức khỏe và cải thiện kết quả lâm sàng cho bệnh nhân.

2. Đối tượng và phạm vi nghiên cứu của đề tài.

Đối tượng nghiên cứu đề tài là các bệnh nhân có nguy cơ mắc bệnh tim đặc biệt là những người có các yếu tố nguy cơ cao như tuổi tác, huyết áp cao, nhịp tim không bình thường và các vấn đề sức khỏe liên quan khác.

Phạm vi của đề tài này là nghiên cứu về dự đoán nguy cơ mắc bệnh tim dựa trên các chỉ số sức khỏe của bệnh nhân được cung cấp trong tập dữ liệu. Các chỉ số sức khỏe bao gồm tuổi (age), giới tính (sex), loại đau ngực (cp), huyết áp tĩnh (trestbps), cholesterol huyết thanh (chol), đường huyết sau ăn (fbs), kết quả điện tâm đồ nghỉ (restecg), nhịp tim tối đa đạt được (thalach), tình trạng thể chất gắng sức gây đau ngực (exang), biến

đổi cũng như ST(Segment T) trên EKG(Electrocardiogram) dẫn đến cảm giác đau ngực (oldpeak), độ dốc của đoạn tăng ST trong thử nghiệm chức năng thể (slope), số mạch đã được tô màu từng đoạn bằng Fluoroscopy (ca), loại thalassemia (thal), và trạng thái bệnh (target).

Môi trường thực nghiệm: Google Colab: Google Colab là một môi trường phát triển dựa trên trình duyệt của Google, hỗ trợ việc viết và chia sẻ mã nguồn Python một cách thuận lợi và không đòi hỏi cài đặt phần mềm trên máy tính cá nhân. Nó cung cấp môi trường lập trình và nguồn lực máy tính mạnh mẽ như GPU và TPU, giúp thực hiện các tác vụ tính toán phức tạp mà không làm chậm máy tính cá nhân. Python, với sự đơn giản và dễ đọc, đã trở thành ngôn ngữ phổ biến trong khoa học dữ liệu, máy học và trí tuệ nhân tạo. Colab tích hợp tốt với Google Drive, cung cấp lưu trữ và chia sẻ dự án dễ dàng. Nó cũng cung cấp tài nguyên GPU và TPU miễn phí, giúp tăng tốc độ tính toán cho máy học và deep learning. Python không chỉ là ngôn ngữ lập trình phổ biến trong phân tích dữ liệu và khoa học dữ liệu, mà còn là công cụ mạnh mẽ trong nhiều lĩnh vực khác như phát triển web, machine learning và AI, nhờ vào sự dễ đọc và hiểu của nó.

Hình 1. Google Colab

3. Mục tiêu nghiên cứu.

- Các bước tiền xử lý:
  + Loại bỏ các tính năng không liên quan.
  + Giải quyết các giá trị bị thiếu.
  + Xử lý các giá trị ngoại lệ.
  + Mã hóa các biến phân loại.
  + Chuyển đổi các tính năng lệch để đạt được phân phối giống như chuẩn.

- Xây dựng mô hình: Thiết lập quy trình cho các mô hình yêu cầu chia tỷ lệ Triển khai và điều chỉnh các mô hình phân loại bao gồm KNN, Decision tree và Random forest. Nhấn mạnh đạt được khả năng thu hồi cao cho loại 1, đảm bảo nhận dạng toàn diện bệnh nhân tim.
- Đánh giá và so sánh hiệu suất của mô hình: Sử dụng precision, recall, và F1-score để đánh giá.

CHƯƠNG II. CƠ SỞ LÝ THUYẾT VÀ CÔNG NGHỆ ỨNG DỤNG

1. Thuật toán KNN.

KNN (K-Nearest Neighbors) là một trong những thuật toán supervised-learning có giám sát đơn giản nhất được sử dụng nhiều trong khai phá dữ liệu và học máy. Ý tưởng của thuật toán này là nó không học một điều gì từ tập dữ liệu học (nên KNN được xếp vào loại lazy learning), mọi tính toán được thực hiện khi nó cần dự đoán nhãn của dữ liệu mới. Lớp (nhãn) của một đối tượng dữ liệu mới có thể dự đoán từ các lớp (nhãn) của hàng xóm gần nó nhất.

KNN là một mô hình đơn giản và trực quan, nhưng vẫn rất hiệu quả vì nó không yêu cầu các tham số cụ thể và không đặt ra bất kỳ giả định nào về phân phối của dữ liệu. Nó cũng có thể áp dụng trực tiếp vào các bài toán phân loại đa lớp.

Với sự linh hoạt và tính tiện lợi của nó, thuật toán KNN đã được áp dụng rộng rãi trong ngành đầu tư. Các ứng dụng bao gồm dự đoán phá sản, dự đoán giá cổ phiếu, xếp hạng tín dụng của các doanh nghiệp, cũng như tạo ra các chỉ số vốn và trái phiếu tùy chỉnh.

Tính đa dạng của các bài toán mà KNN có thể giải quyết, cùng với khả năng sử dụng mà không cần phải điều chỉnh nhiều tham số, làm cho nó trở thành một công cụ hữu ích và phổ biến trong việc phân tích dữ liệu và ra quyết định trong lĩnh vực đầu tư.

2. Thuật toán Decision Tree.

Mô hình cây quyết định (Decision Tree) là thuật toán học máy có thể thực hiện công việc phân loại và hồi quy, thậm chí có thể thực hiện các công việc yêu cầu xuất nhiều dạng dữ liệu khác nhau. Cây quyết định có tiềm năng lớn, hiệu suất ổn định khi thực hiện học trên các tập dữ liệu phức tạp.

Mô hình cây quyết định là còn là nền tảng của một thuật toán học máy khác là Random Forests, một trong các thuật toán mạnh nhất hiện nay.

Trong học máy, mô hình cây quyết định nhận đầu ra là các giá trị rời rạc trong tập hữu hạn được gọi là cây phân loại (classification trees). Với mục đích phân loại, các nút

lá sẽ biểu diễn các kết quả phân loại của thuật toán, và các nhánh thể hiện phép giao giữa các đặc trưng dẫn tới các kết quả đoán nhận.

Những mô hình cây quyết định nhận đầu ra là các giá trị liên tục (thường là số thực) được gọi là cây hồi quy (regression trees).

Ta có thể mở rộng khái niệm cây hồi quy với những mô hình sử dụng các cặp đặc trưng không tương đồng, như các chuỗi danh mục (categorical sequences).

Cây quyết định là một trong các mô hình học máy nổi tiếng do tính đơn giản, dễ hiểu.

3. Thuật toán Random forest.

Random forest là một phương pháp thống kê mô hình hóa bằng máy (machine learning statistic) dùng để phục vụ các mục đích phân loại, tính hồi quy và các nhiệm vụ khác. Thuật toán Random Forest sẽ xây dựng nhiều cây quyết định bằng thuật toán Decision Tree trên tập dữ liệu khác nhau và dùng tập thuộc tính khác nhau. Sau đó kết quả dự đoán của thuật toán Random Forest sẽ được tổng hợp từ các cây quyết định.

Do quá trình xây dựng mỗi cây quyết định đều có yếu tố ngẫu nhiên (random) nên kết quả là các cây quyết định trong thuật toán Random Forest có thể khác nhau.

Thuật toán Random Forest sẽ xây dựng nhiều cây quyết định bằng thuật toán Decision Tree, tuy nhiên mỗi cây quyết định sẽ khác nhau (có yếu tố random). Sau đó kết quả dự đoán được tổng hợp từ các cây quyết định.

Random Forest cho thấy hiệu quả hơn so với thuật toán phân loại thường được sử dụng vì có khả năng tìm ra thuộc tính nào quan trọng hơn so với những thuộc tính khác. Trên thực tế, nó còn có thể chỉ ra rằng một số thuộc tính là không có tác dụng trong cây quyết định.

CHƯƠNG III. SẢN PHẨM ĐỒ ÁN

1. Tổng quan về tập dữ liệu.

1.1. Thông tin cơ bản về tập dữ liệu.

Mô tả bộ dữ liệu: DataFrame chứa 5035 dòng và 14 cột, với mỗi cột có thông tin về mỗi bệnh nhân.

Tác giả dữ liệu: **Farzad Nekouei** - Computer Vision Engineer | Expert in Machine Learning, Deep Learning & Data Science

Bảng mô tả dữ liệu:

Biến	**Mô tả**
age	Tuổi của bệnh nhân (năm)
sex	Giới tính bệnh nhân (0 = male, 1 = female)
cp	Loại đau ngực: 0: Đau thắt ngực điển hình 1: Đau thắt ngực không điển hình 2: Không đau thắt ngực 3: Không có triệu chứng
trestbps	Huyết áp lúc nghỉ mm Hg

chol	Cholesterol trong huyết thanh (mg/dl)
fbs	Mức đường huyết nhanh sau ăn, phân loại như sau 120 mg/dl (1 = đúng, 0 = sai)
restecg	Kết quả điện tâm đồ nghỉ: 0: Bình thường 1: Có biến đổi sóng ST-T 2: Hiển thị có thể hoặc chắc chắn là tăng cường thất trái
thalach	Tần suất tim tối đa đạt được trong khi thực hiện bài kiểm tra căng thẳng
exang	Đau ngực do vận động gây ra (1 = có, 0 = không)
oldpeak	Sự giảm ST do vận động so với nghỉ ("ST" là viết tắt của "ST segment", là một phần của đường cong điện tâm đồ (ECG) được sử dụng để đánh giá hoạt động điện của tim)

slope	slope: Góc nghiêng của đoạn ST cao điểm khi vận động: 0: Đang tăng 1: Bằng phẳng 2: Đang giảm
ca	Số mạch chính (0-4) được tô màu bằng chụp tia X
thal	Kết quả thử nghiệm căng thẳng thalium: 0: Bình thường 1: Khuyết tật cố định 2: Khuyết tật có thể đảo ngược 3: Không mô tả
target	Tình trạng bệnh tim (0 = không có bệnh, 1 = có bệnh)

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3.0	145	233	1	0.0	150	0	2.3	0	0.0	1.0	1.0
1	37	1	2.0	130	250	0	1.0	187	0	3.5	0	0.0	2.0	1.0
2	41	0	1.0	130	204	0	0.0	172	0	1.4	2	0.0	2.0	1.0
3	56	1	1.0	120	236	0	1.0	178	0	0.8	2	0.0	2.0	1.0
4	57	0	0.0	120	354	0	1.0	163	1	0.6	2	0.0	2.0	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
5030	63	0	3.0	150	407	0	1.0	154	0	4.0	2	3.0	2.0	1.0
5031	45	1	1.0	112	160	0	1.0	138	0	0.0	1	0.0	2.0	0.0
5032	54	1	0.0	120	258	0	1.0	147	1	1.6	2	0.0	2.0	1.0
5033	48	1	2.0	130	256	1	1.0	150	1	0.0	2	2.0	2.0	0.0
5034	55	1	3.0	140	217	0	1.0	111	1	5.6	0	0.0	3.0	1.0

5035 rows × 14 columns

Hình 2. Tổng quan về bộ dữ liệu

Tóm tắt dữ liệu:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5035 entries, 0 to 5034
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       5035 non-null   int64
 1   sex       5035 non-null   int64
 2   cp        5033 non-null   float64
 3   trestbps  5035 non-null   int64
 4   chol      5035 non-null   int64
 5   fbs       5035 non-null   int64
 6   restecg   5035 non-null   float64
 7   thalach   5035 non-null   int64
 8   exang     5035 non-null   int64
 9   oldpeak   5035 non-null   float64
 10  slope     5035 non-null   int64
 11  ca        5034 non-null   float64
 12  thal      5034 non-null   float64
 13  target    5033 non-null   float64
dtypes: float64(6), int64(8)
memory usage: 550.8 KB
```

Hình 3. Bảng tóm tắt dữ liệu

Hầu hết các cột có kiểu dữ liệu int64 và float64.

Dựa vào tính năng, chúng chúng em hấy rằng 9 cột (sex, cp, fbs, Restecg, exang, Slope, ca, thal và target) thực sự là số về loại dữ liệu, nhưng mang tính phân loại về mặt

ngữ nghĩa của chúng. Vì thế các tính năng này phải được chuyển đổi thành kiểu dữ liệu chuỗi (object) để phân tích và diễn giải thích hợp.

```
age           int64
sex          object
cp           object
trestbps      int64
chol          int64
fbs          object
restecg      object
thalach       int64
exang        object
oldpeak     float64
slope        object
ca           object
thal         object
target       object
dtype: object
```

Hình 4. Kiểu dữ liệu

2. Tiền xử lý dữ liệu.

2.1. Xử lý dữ liệu thiếu.

Chúng em nhận thấy có vài dòng dữ liệu bị thiếu nhưng không đáng kể. Vì thế, chúng em sẽ xoá đi những dòng chứa dữ liệu bị thiếu.

Sau khi đã xoá những dòng dữ liệu bị thiếu, bộ DataFrame còn lại chứa 5031 dòng và 14 cột, với mỗi cột có thông tin về mỗi bệnh nhân.

2.2. Chuyển đổi các biến phân loại.

Quyết định mã hóa một lần (One-Hot Encoding) là một kỹ thuật được sử dụng trong xử lý dữ liệu để biến đổi các biến phân loại (categorical variables) thành dạng số học mà các mô hình học máy có thể hiểu được.. Dưới đây là một số lý do giải thích cho quyết định đó:

- Biến danh nghĩa: Đây là các biến không có thứ tự, chẳng hạn như giới tính. Vì chúng không có thứ tự, chúng cần được chuyển đổi thành dạng nhị phân để tránh tạo ra mối quan hệ thứ tự không đúng lúc huấn luyện mô hình.

- Biến thứ tự: Trái ngược với các biến danh nghĩa, các biến này có thứ tự cụ thể, ví dụ như ca(số mạch chính). Do đó, chúng không cần phải được chuyển đổi thành biến nhị phân vì thứ tự của chúng đã cung cấp thông tin hữu ích cho mô hình.

Cần mã hóa một lần: cp, Restecg, thal.
Không cần mã hóa một lần: sex, fbs, exang,lope, ca.

	age	sex	trestbps	chol	fbs	thalach	exang	oldpeak	slope	ca	target	cp_1.0	cp_2.0	cp_3.0	cp_33.0	restecg_1.0	restecg_2.0	thal_1.0	thal_2.0	thal_3.0
0	63	1	145	233	1	150	0	2.3	0	0	1	0	0	1	0	0	0	1	0	0
1	37	1	130	250	0	187	0	3.5	0	0	1	0	1	0	0	1	0	0	1	0
2	41	0	130	204	0	172	0	1.4	2	0	1	1	0	0	0	0	0	0	1	0
3	56	1	120	236	0	178	0	0.8	2	0	1	1	0	0	0	1	0	0	1	0
4	57	0	120	354	0	163	1	0.6	2	0	1	0	0	0	0	1	0	0	1	0

Hình 5. Kết quả sau khi mã hoá 1 lần

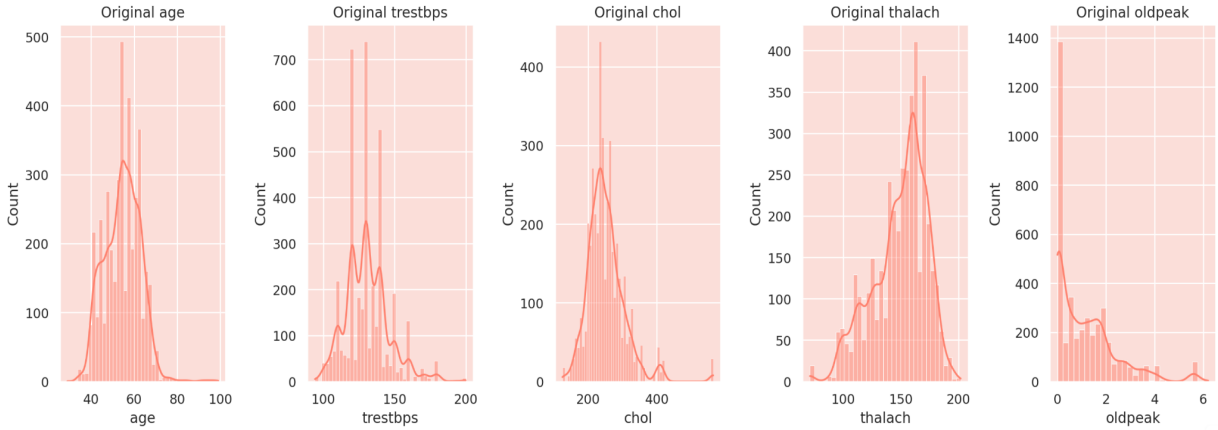
2.3. Chuyển đổi các đặc trưng bị lệch.

Phép biến đổi Box-Cox là một phương pháp mạnh mẽ để ổn định phương sai và làm cho dữ liệu giống với phân phối chuẩn hơn. Nó đặc biệt hữu ích khi không chắc chắn về bản chất chính xác của phân phối mà chúng ta đang xử lý.

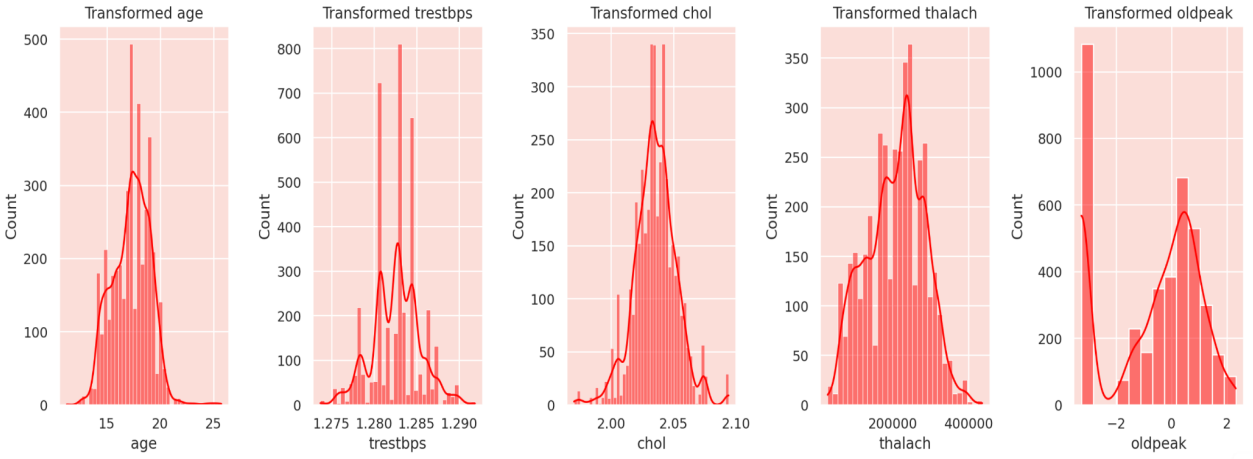
Việc chuyển đổi các tính năng trở nên giống bình thường hơn chủ yếu giúp giảm thiểu tác động của các ngoại lệ, điều này đặc biệt có lợi cho các thuật toán dựa trên khoảng cách như SVM và KNN. Bằng cách giảm ảnh hưởng của các giá trị ngoại lệ, chúng em nghĩ rằng các thuật toán này có thể tính toán khoảng cách hiệu quả hơn và tạo ra kết quả đáng tin cậy hơn.

- Trước tiên, chúng em sẽ chia tập dữ liệu của mình thành tập huấn luyện và tập thử nghiệm. Điều này đảm bảo rằng chúng em có một bộ dữ liệu riêng biệt để đánh giá hiệu suất của mô hình, không bị ảnh hưởng trong giai đoạn huấn luyện và tiền xử lý.
- Chúng em sẽ kiểm tra phân phối của các đặc trưng liên tục trong tập huấn luyện. Nếu chúng xuất hiện bị lệch, chúng em sẽ áp dụng biến đổi Box-Cox để ổn định phương sai và làm cho dữ liệu trở nên giống phân phối chuẩn hơn. Quan trọng là, chúng em sẽ xác định các thông số biến đổi Box-Cox chỉ dựa trên dữ liệu huấn luyện.

- Khi các tham số biến đổi Box-Cox đã được xác định từ tập huấn luyện, chúng em sẽ sử dụng chúng để biến đổi tập thử nghiệm. Điều này đảm bảo rằng không có thông tin từ tập kiểm tra nào rò rỉ vào quá trình.



Hình 6. Biểu đồ trước khi sử dụng phép biến đổi Box-Cox



Hình 7. Biểu đồ sau khi sử dụng phép biến đổi Box-Cox

- age: Phép biến đổi đã làm cho phân bố tuổi đối xứng hơn, đưa nó đến gần hơn với phân phối chuẩn.
- trestbps: Sự phân bố của huyết áp sau khi chuyển đổi có vẻ giống bình thường hơn, độ lệch giảm đi.
- chol: Sau khi áp dụng phép biến đổi Box-Cox, Cholesterol thể hiện hình dạng phù hợp hơn với phân bố chuẩn.
- thalach: Đặc điểm thalach trước khi biến đổi đã khá đối xứng, và sau khi biến đổi, nó tiếp tục có hình dạng tương tự, cho thấy sự phân bố ban đầu của nó gần với mức bình thường.

- oldpeak: Việc chuyển đổi đã cải thiện phân phối oldpeak, nhưng nó vẫn không hoàn toàn giống với phân phối chuẩn. Điều này có thể là do bản chất vốn có của dữ liệu hoặc sự hiện diện của các giá trị ngoại lệ.

3. Xây dựng mô hình.

3.1. Xây dựng mô hình Decision Tree (DT).

Đầu tiên, xác định mô hình Decision Tree cơ bản.

```
# Xác định mô hình DT cơ bản
dt_base = DecisionTreeClassifier(random_state=0)
```

Hình 8. Xác định mô hình DT cơ bản

Trong các tình huống y tế, đặc biệt là trong bối cảnh chẩn đoán bệnh, điều quan trọng hơn là phải có mức thu hồi (độ nhạy) cao đối với nhóm dương tính (bệnh nhân mắc bệnh). Tỷ lệ thu hồi cao đảm bảo rằng hầu hết các trường hợp dương tính thực tế đều được xác định chính xác, ngay cả khi điều đó có nghĩa là một số trường hợp dương tính giả (trường hợp những người khỏe mạnh bị phân loại sai là mắc bệnh này). Lý do là nói chung thà có một vài cảnh báo sai còn hơn là bỏ lỡ việc chẩn đoán bệnh nhân có nguy cơ mắc bệnh.

Chính vì thế chúng em đang thiết lập một hàm để xác định tập hợp siêu tham số tối ưu mang lại khả năng thu hồi cao nhất cho mô hình. Cách tiếp cận này đảm bảo một khuôn khổ có thể tái sử dụng để điều chỉnh siêu tham số của các mô hình tiếp theo.

```python
def tune_clf_hyperparameters(clf, param_grid, X_train, y_train, scoring='recall', n_splits=3):
    '''
    Hàm này tối ưu hóa các siêu tham số cho bộ phân loại bằng cách tìm kiếm trên lưới siêu tham số được chỉ định.
     Nó sử dụng GridSearchCV và xác thực chéo (StratifiedKFold) để đánh giá các kết hợp siêu tham số khác nhau.
     Sự kết hợp có khả năng thu hồi cao nhất cho loại 1 được chọn làm chỉ số tính điểm mặc định.
     Hàm trả về bộ phân loại với các siêu tham số tối ưu.
    '''

    # Tạo đối tượng xác thực chéo bằng StratifiedKFold để đảm bảo phân phối lớp giống nhau trên tất cả các nếp gấp
    cv = StratifiedKFold(n_splits=n_splits, shuffle=True, random_state=0)

    # Tạo đối tượng GridSearchCV
    clf_grid = GridSearchCV(clf, param_grid, cv=cv, scoring=scoring, n_jobs=-1)

    # Điều chỉnh đối tượng GridSearchCV cho dữ liệu huấn luyện
    clf_grid.fit(X_train, y_train)

    # Nhận siêu tham số tốt nhất
    best_hyperparameters = clf_grid.best_params_

    # Trả về thuộc tính best_estimator_ cung cấp cho chúng ta mô hình tốt nhất phù hợp với dữ liệu huấn luyện
    return clf_grid.best_estimator_, best_hyperparameters
```

Hình 9. Thiết lập một hàm để xác định tập hợp siêu tham số

Thiết lập lưới siêu tham số và sử dụng hàm tune_clf_hyperparameters để xác định các siêu tham số tối ưu cho mô hình Decision Tree.

```python
# Lưới siêu tham số cho DT
param_grid_dt = {
    'criterion': ['gini', 'entropy'],
    'max_depth': [2,3],
    'min_samples_split': [2, 3, 4],
    'min_samples_leaf': [1, 2]
}
```

```python
# Gọi hàm điều chỉnh siêu tham số
best_dt, best_dt_hyperparams = tune_clf_hyperparameters(dt_base, param_grid_dt, X_train, y_train)
```

Hình 10. Xác định các siêu tham số tối ưu cho mô hình Decision Tree

- **Đánh giá mô hình Decision Tree.**

Đánh giá hiệu suất mô hình Decision Tree trên tập dữ liệu thử nghiệm.

```
# Đánh giá mô hình tối ưu trên dữ liệu thử nghiệm
print(classification_report(y_test, best_dt.predict(X_test)))
```

```
              precision    recall  f1-score   support

           0       0.63      0.59      0.61       381
           1       0.76      0.79      0.78       626

    accuracy                           0.71      1007
   macro avg       0.70      0.69      0.69      1007
weighted avg       0.71      0.71      0.71      1007
```

Hình 11. Đánh giá hiệu suất mô hình Decision Tree

0: âm tính nghĩa là không có bệnh.

1: dương tính nghĩa là có bệnh.

Precision (Độ chính xác): Độ chính xác của lớp 0 là 0.63 và của lớp 1 là 0.76. Điều này có nghĩa là trong số các dự đoán được đưa ra bởi mô hình, có 63% là chính xác cho âm tính và 76% là chính xác cho dương tính.

Recall (Độ nhớ): Độ nhớ của lớp 0 là 0.59 và của lớp 1 là 0.79. Điều này có nghĩa là mô hình có khả năng nhớ được 59% trường hợp thực sự thuộc âm tính và 79% trường hợp thực sự thuộc dương tính.

F1-score (Điểm F1): Điểm F1 là một số đo kết hợp giữa độ chính xác và độ nhớ của mô hình. Điểm F1 cho lớp 0 là 0.61 và cho lớp 1 là 0.78. Điểm F1 càng cao thì mô hình càng tốt, vì nó kết hợp cả độ chính xác và độ nhớ.

Accuracy (Độ chính xác tổng thể): Là tỷ lệ phần trăm các dự đoán đúng trên tổng số dự đoán, tỷ lệ này là 0.71, tức là mô hình dự đoán đúng khoảng 71% trên tất cả các mẫu trong tập dữ liệu.

Chúng em chủ yếu dựa trên Recall (Độ nhớ) của mô hình để đánh giá độ hiệu quả. Và kết quả cho thất mô hình không quá hiệu quả khi mức thu hồi chỉ 79% cho loại 1(dương tính).

3.2. Xây dựng mô hình Random Forest (RF).

Đầu tiên, xác định mô hình Random Forest cơ bản.

```
# xác ddinhj mô hình RF cơ bản
rf_base = RandomForestClassifier(random_state=0)
```

Hình 12. Xác định mô hình Random Forest cơ bản

Thiết lập lưới siêu tham số và sử dụng hàm tune_clf_hyperparameters để xác định chính xác các siêu tham số tối ưu cho mô hình.

```
param_grid_rf = {
    'n_estimators': [10, 30, 50, 70, 100],
    'criterion': ['gini', 'entropy'],
    'max_depth': [2, 3, 4],
    'min_samples_split': [2, 3, 4, 5],
    'min_samples_leaf': [1, 2, 3],
    'bootstrap': [True, False]
}
```

```
# Sử dụng hàm tune_clf_hyperparameters để có công cụ ước tính tốt nhất
best_rf, best_rf_hyperparams = tune_clf_hyperparameters(rf_base, param_grid_rf, X_train, y_train)
print('RF Optimal Hyperparameters: \n', best_rf_hyperparams)
```

```
RF Optimal Hyperparameters:
 {'bootstrap': True, 'criterion': 'gini', 'max_depth': 2, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 10}
```

Hình 13. Xác định các siêu tham số tối ưu cho mô hình RF

- **Đánh giá mô hình Random Forest.**

Đánh giá hiệu suất mô hình Random Forest trên tập dữ liệu thử nghiệm.

```
# Đánh giá mô hình tối ưu trên dữ liệu thử nghiệm
print(classification_report(y_test, best_rf.predict(X_test)))
```

```
              precision    recall  f1-score   support

           0       0.68      0.49      0.57       381
           1       0.73      0.86      0.79       626

    accuracy                           0.72      1007
   macro avg       0.71      0.67      0.68      1007
weighted avg       0.71      0.72      0.71      1007
```

Hình 14. Đánh giá hiệu suất mô hình Random Forest

Hiệu suất tương tự của mô hình Random Forest trên tập dữ liệu cho thấy nó không quá phù hợp. Song, thuật toán Random Forest tốt hơn thuật toán Decision Tree với mức thu hồi 86% cho loại 1.

3.3. Xây dựng mô hình KNN.

Đầu tiên, xác định mô hình KNN cơ bản và thiết lập quy trình theo tỷ lệ.

```
# Xác định mô hình KNN cơ sở và thiết lập quy trình theo tỷ lệ
knn_pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('knn', KNeighborsClassifier())
])
```

Hình 15. Xác định mô hình KNN cơ bản

Thiết lập lưới siêu tham số và sử dụng hàm tune_clf_hyperparameters để xác định chính xác các siêu tham số tối ưu cho quy trình KNN.

```
# Lưới siêu tham số cho KNN
knn_param_grid = {
    'knn__n_neighbors': list(range(1, 12)),
    'knn__weights': ['uniform', 'distance'],
    'knn__p': [1, 2]  # 1: Manhattan distance, 2: Euclidean distance
}
```

```
# Điều chỉnh siêu tham số cho KNN
best_knn, best_knn_hyperparams = tune_clf_hyperparameters(knn_pipeline, knn_param_grid, X_train, y_train)
```

Hình 16. Xác định các siêu tham số tối ưu cho mô hình KNN

- **Đánh giá mô hình KNN.**

Đánh giá hiệu suất của mô hình KNN trên tập dữ liệu thử nghiệm.

```
# Đánh giá mô hình tối ưu trên dữ liệu thử nghiệm
print(classification_report(y_test, best_knn.predict(X_test)))
```

```
              precision    recall  f1-score   support

           0       0.78      0.60      0.68       381
           1       0.79      0.90      0.84       626

    accuracy                           0.78      1007
   macro avg       0.78      0.75      0.76      1007
weighted avg       0.78      0.78      0.78      1007
```

Hình 17. Đánh giá hiệu suất của mô hình KNN

Hiệu suất tương tự của mô hình KNN trên tập dữ liệu cho thấy thuật toán khá hiệu quả. Thuật toán KNN với mức thu hồi 90% cho loại 1. Tốt nhất trong 3 mô hình.

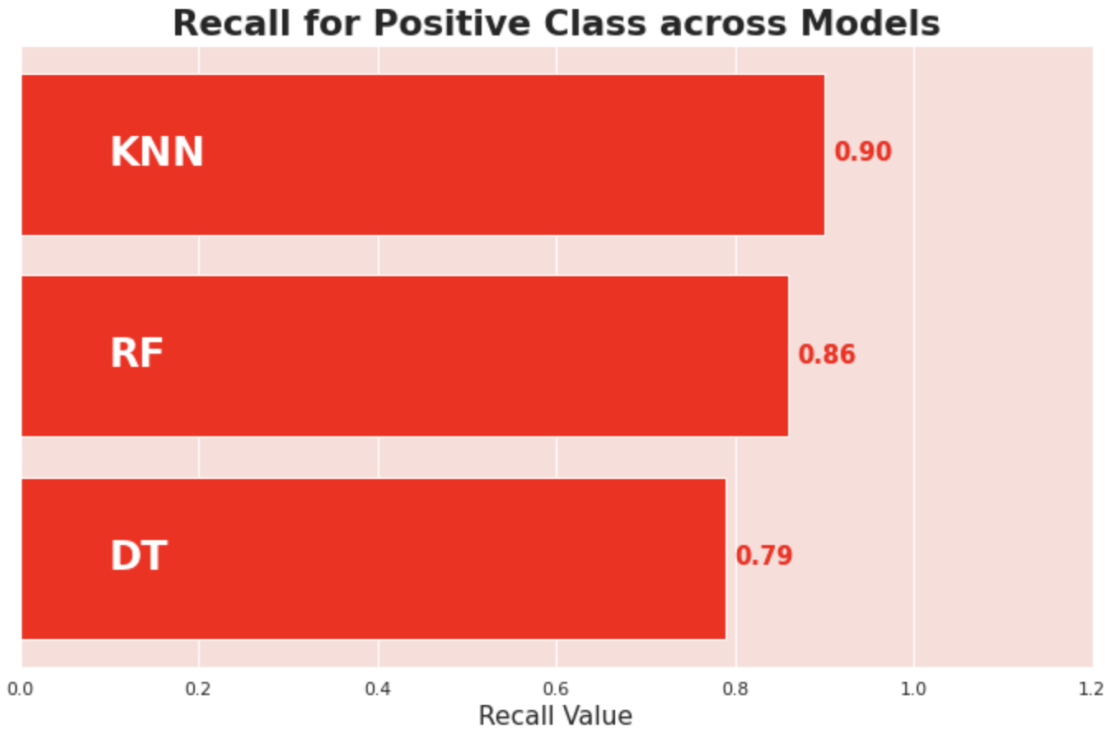
4. So sánh và đánh giá sự chính xác của 3 mô hình.

Với bối cảnh quan trọng của việc chẩn đoán bệnh tim, mục tiêu chính của việc phân tích này là đảm bảo tỷ lệ thu hồi cao đối với nhóm dương tính. Điều bắt buộc là phải xác định chính xác mọi trường hợp bệnh tim tiềm ẩn, vì ngay cả một chẩn đoán bị bỏ sót cũng có thể gây ra hậu quả nghiêm trọng. Tuy nhiên, trong khi cố gắng đạt được mức thu hồi cao, điều cần thiết là phải duy trì hiệu suất cân bằng để tránh các can thiệp y tế không cần thiết đối với những người khỏe mạnh. Bây giờ chúng em sẽ đánh giá các mô hình của mình dựa trên các tiêu chuẩn y tế quan trọng này.

	precision_0	precision_1	recall_0	recall_1	f1_0	f1_1	macro_avg_precision	macro_avg_recall	macro_avg_f1	accuracy
KNN	0.78	0.79	0.60	0.90	0.68	0.84	0.78	0.75	0.76	0.78
RF	0.68	0.73	0.49	0.86	0.57	0.79	0.71	0.67	0.68	0.72
DT	0.63	0.76	0.59	0.79	0.61	0.78	0.70	0.69	0.69	0.71

Hình 18. Bảng so sánh và đánh giá sự chính xác của 3 mô hình

Biểu đồ so sánh mức độ thu hồi đối với lớp bệnh nhân dương tính.



Hình 19. Biểu đồ so sánh 3 mô hình

=> Mô hình KNN thể hiện khả năng rất tốt trong việc nhận biết các bệnh nhân tim tiềm năng. Với mức thu hồi 0,9 cho loại 1, rõ ràng là hầu hết tất cả bệnh nhân mắc bệnh tim đều được xác định chính xác. Đây là điều hết sức quan trọng trong môi trường y tế. Tuy nhiên, hiệu suất cân bằng của mô hình đảm bảo rằng trong khi hướng đến khả năng thu hồi cao, nó không ảnh hưởng đến độ chính xác, do đó không làm hệ thống bị quá tải với các cảnh báo không cần thiết.

Trên thực tế, bệnh tim có rất nhiều loại và mỗi loại sẽ có nhiều yếu tố để xác định bệnh nhân có mắc bệnh hay không. Quá trình chẩn đoán bệnh tim thường là một quá trình tổ hợp, kết hợp nhiều thông tin từ các yếu tố khác nhau để đưa ra một kết luận chính xác. Thông thường, bác sĩ sẽ chẩn đoán bệnh tim mạch dựa trên tiểu sử bệnh của gia đình; các yếu tố nguy cơ như hút thuốc, tiểu đường, béo phì, căng thẳng...; xét nghiệm thể chất, xét nghiệm máu, chụp X-quang, siêu âm tim… thì mới có thể xác định được bệnh nhân có mắc bệnh tim hay không và đó là loại bệnh tim nào.

Trong mô hình chẩn đoán bệnh tim của chúng em, chỉ đơn giản là chẩn đoán xem bệnh nhân có mắc bệnh tim hay không, mà sẽ không đi sâu vào chuyên môn từng loại bệnh tim. Và điện tâm đồ, hay còn gọi là đo điện tim (Electrocardiogram, viết tắt ECG) là một xét nghiệm ghi lại hoạt động điện học của tim dưới dạng đồ thị. Các xung điện tự nhiên điều phối sự co bóp của tim để giữ máu tuần hoàn. Điện tâm đồ ghi lại những xung điện này. Sự thay đổi của xung điện được phát hiện qua điện tâm đồ có thể là dấu hiệu của nhiều bệnh lý liên quan đến tim. Vì thế, chúng em sẽ chủ yếu dựa vào điện tâm đồ để xác định và đó là yếu tố cốt lõi trong mô hình chẩn đoán bệnh tim của chúng em.

Trong những trường dữ liệu trong tập dữ liệu mà chúng em đưa ra, thì trường dữ liệu restecg, oldpeak và slope và những trường có liên quan đến kết quả của điện tâm đồ. Và đó chính là những trường dữ liệu cốt lõi, quan trọng nhất trong mô hình chẩn đoán bệnh tim của chúng em.

CHƯƠNG IV. KẾT LUẬN

Tóm lại, qua quá trình nghiên cứu và thực hành, nhóm chúng em áp dụng các kiến thức đã học như tiền xử lý dữ liệu, chuyển đổi kiểu dữ liệu, xây dựng các mô hình phân tích kết quả dựa vào dữ liệu sau khi đã tiền xử lý, sau đó tiến hành đánh giá mô hình rồi so sánh với các mô hình đã thực hiện trước đó để so sánh kết quả. Thông qua quá trình thực hiện, nhóm chúng em đưa ra kết luận đối với dữ liệu chuẩn đoán bệnh tim, mô hình SVM có khả năng khá tốt trong việc nhận diện các bệnh nhân có nguy cơ mắc bệnh tim từ đó đưa ra tỷ lệ thu hồi đối với các bệnh nhân này. Từ đó đưa ra mô hình thử nghiệm đánh giá bệnh nhân mắc bệnh tim có thể sử dụng để học hỏi, trao đổi và ứng dụng vào nghiên cứu khoa học.

TÀI LIỆU THAM KHẢO

[1] Hop, N. T. (2019, July 16). *KNN (K-Nearest Neighbors) #1*. Viblo. https://viblo.asia/p/knn-k-nearest-neighbors-1-djeZ14ejKWz

[2] *Tổng Quan Về Thuật Toán Cây Quyết Định*. (n.d.). Khoá Học. Retrieved March 22, 2024, from https://tek4.vn/khoa-hoc/machine-learning-co-ban/tong-quan-ve-thuat-toan-cay-quyet-dinh

[3] *Random Forest algorithm — Machine Learning cho dữ liệu dạng bảng*. (n.d.). Machinelearningcoban.com. Retrieved March 22, 2024, from https://machinelearningcoban.com/tabml_book/ch_model/random_forest.html

[4] *Bộ dữ liệu dự đoán bệnh tim*. (n.d.). Bộ dữ liệu dự đoán bệnh tim. Retrieved March 22, 2024, from https://openscience.vn/chi-tiet-du-lieu/bo-du-lieu-du-doan-benh-timsjyef-11310

[5] *Heart disease prediction*. (2023, August 11). Kaggle.com; Kaggle. https://www.kaggle.com/code/farzadnekouei/heart-disease-prediction?fbclid=IwAR1FPrpyMGPu49YBCTrfL6Ru9El44d4lUOgOCk0QcQUTKnO4rwGz5wMdNYM