

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC VÀ KỸ THUẬT THÔNG TIN



BÁO CÁO ĐỒ ÁN HỆ HỖ TRỢ QUYẾT ĐỊNH
HEART DISEASE PREDICTION

GVHD: ThS. Nguyễn Hồ Duy Trí

Lớp: IS254.O21

Sinh viên thực hiện:

21520473 - Lê Quốc Thuận

21520578 - Nguyễn Thị Vân Anh

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

....., ngày.....tháng.....năm 2024

Người nhận xét
(Ký tên và ghi rõ họ tên)

MỤC LỤC

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN	2
MỤC LỤC	3
MỤC LỤC HÌNH ẢNH	4
MỤC LỤC BẢNG	4
1. Lý do chọn đề tài	5
2. Mô tả dữ liệu	5
3. Mô tả bài toán	7
4. Các mô hình được lựa chọn	8
4.1. Logistic Regression	8
4.2. Decision Tree	10
4.3. XGBoost	11
4.4. AdaBoost	12
4.5. K-Nearest Neighbour (KNN)	15
4.6. Support Vector Machine (SVM)	17
4.7. Random Forest	18
4.8. Nhận xét:	20
5. Quy trình ra quyết định	20
6. Kết quả đạt được	24
7. Kết luận	24
8. Bảng phân công công việc của các thành viên trong nhóm	26
9. Bảng đánh giá chéo các thành viên trong nhóm (thang điểm 10)	26
TÀI LIỆU THAM KHẢO	27

MỤC LỤC HÌNH ẢNH

Figure 1 : Report of Logistic Regression	9
Figure 2 : Confusion Matrix for Logistic Regression	9
Figure 3 : Report of Decision Tree	11
Figure 4 : Confusion Matrix for Decision Tree	11
Figure 5 : Report of XGboost	12
<i>Figure 6 : Confusion Matrix for XGboost</i>	<i>12</i>
Figure 7 : Report of Adaboost	14
Figure 8 : Confusion Matrix for Adaboost	14
Figure 9 : Report of Knearest Neighbour Classificaiton	16
Figure 10 : Confusion Matrix for Knearest Neighbour Classificaiton	16
Figure 11 : Report of Support Vector Machine	17
Figure 12 : Confusion Matrix for Support Vector Machine	18
Figure 13 : Report of Random Forest	19
Figure 14 : Confusion Matrix for Random Forest	20
Figure 15 : Sơ đồ hệ thống theo cách tiếp cận thủ công	21
Figure 16 : Sơ đồ kiến trúc hệ thống	21
Figure 17 : Giao diện người dùng hệ thống dự đoán bệnh tim mạch 1	22
Figure 18 : Giao diện người dùng hệ thống dự đoán bệnh tim mạch 2	23
Figure 19 : Giao diện người dùng hệ thống dự đoán bệnh tim mạch 3	23
Figure 20 : Giao diện người dùng hệ thống dự đoán bệnh tim mạch 4	24

MỤC LỤC BẢNG

Table 1 : Cấu trúc của tập dữ liệu	7
Table 2 : Bảng phân công công việc của các thành viên trong nhóm 13	26
Table 3 : Bảng đánh giá chéo các thành viên trong nhóm 13 (thang điểm 10)	26

IS254.O21– Hệ hỗ trợ quyết định

1. Lý do chọn đề tài

Ngày nay, các bệnh về tim mạch gia tăng nhanh chóng và ngày càng trở nên phổ biến. Theo báo cáo Gánh nặng bệnh tật toàn cầu năm 2022 do Tổ chức Y tế Thế giới (WHO) công bố, bệnh tim mạch là nguyên nhân hàng đầu gây tử vong trên toàn thế giới. Mỗi năm, các bệnh tim mạch cướp đi 19,5 triệu sinh mạng, chiếm khoảng 1/3 số ca tử vong do mọi nguyên nhân. Nguyên nhân chủ yếu của bệnh tim mạch là do di truyền, tuổi tác, lối sống, thói quen sinh hoạt (ít hoạt động thể chất), chế độ ăn uống (uống nhiều rượu bia, hút thuốc), béo phì (BMI cao)...Việc tìm kiếm và ngăn ngừa các yếu tố có tác động lớn đến bệnh tim mạch là điều quan trọng trong chăm sóc sức khỏe. Vì những nguyên do trên, nhóm tin rằng việc sử dụng các mô hình học máy là cần thiết để xác định xem bệnh nhân có mắc các bệnh về tim mạch hay không và đẩy nhanh quá trình chẩn đoán và kiểm soát các bệnh tim mạch dựa trên các thông tin y tế được cung cấp về bệnh nhân đó.

2. Mô tả dữ liệu

Tập dữ liệu Heart Disease.xlsx được thu tập từ Kaggle. Tập dữ liệu gồm 319795 dòng và 18 trường dữ liệu. Tập dữ liệu chứa các thông tin y tế về bệnh nhân và kết quả chẩn đoán xem bệnh nhân có mắc các bệnh tim mạch hay không.

Cấu trúc của tập dữ liệu:

STT	Tên biến	Mô tả chi tiết	Dữ liệu mẫu
0	BMI	Body Mass Index - Chỉ số khối cơ thể	34.75; 22.67;...
1	Smoking	Bạn đã hút ít nhất 100 điếu thuốc trong đời chưa? 0 = No 1 = Yes	0; 1;...
2	AlcoholDrinking	Người nghiện rượu nặng (đàn ông trưởng thành) Hơn 14 ly mỗi tuần và hơn 7 ly mỗi tuần đối với phụ nữ trưởng thành 0 = No 1 = Yes	0; 1;...

IS254.O21– Hệ hỗ trợ quyết định

3	Stroke	0 = No (không bị đột quỵ) 1 = Yes (bị đột quỵ)	0; 1;...
4	PhysicalHealth	Sức khỏe thể chất - Bây giờ hãy nghĩ về sức khỏe thể chất của bạn, bao gồm cả bệnh tật và chấn thương thể chất. Trong 30 ngày qua, có bao nhiêu ngày sức khỏe thể chất của bạn kém? ? (0-30 ngày)	3; 28; 6;...
5	MentalHealth	Sức khỏe tinh thần - Có bao nhiêu ngày trong 30 ngày qua bạn có sức khỏe tinh thần kém? (0-30 ngày)	2; 30; 15;...
6	DiffWalking	Bạn có gặp khó khăn nghiêm trọng khi đi bộ hoặc leo cầu thang không? 0 = No 1 = Yes	0; 1;...
7	Sex	0 = Female (nữ) 1 = Male (nam)	0; 1;...
8	AgeCategory	Nhóm tuổi (65-69); (60-64); Khác	0; 7; 9;...
9	Race	Chủng tộc: White (Da trắng), Hispanic (gốc Tây Ban Nha), Khác	0; 5; 3;...
10	Diabetic	Bạn có bị tiểu đường không? Yes/ No/ Khác	0; 2;...
11	PhysicalActivity	Hoạt động thể chất-Bạn có tham gia hoạt động thể chất hoặc tập thể dục thường xuyên không? 0 = No 1 = Yes	0; 1;...
12	GenHealth	Bạn cảm thấy thế nào về sức khỏe của mình? Very good/ Good/ Khác (Fair/ Poor/Excellent...)	1; 2; 4;...
13	SleepTime	Thời gian ngủ-Trung bình, bạn khỏe mạnh như thế nào trong khoảng thời gian 24 giờ? Bạn đã ngủ được bao nhiêu giờ?	6; 8; 12;...
14	Asthma	Hen suyễn - Bạn có bị hen suyễn không? 0 = No 1 = Yes	0; 1;...
15	KidneyDisease	Bệnh thận - Bạn đã bao giờ được thông báo rằng	0; 1;...

IS254.O21– Hệ hỗ trợ quyết định

		mình mắc bệnh thận không? 0 = No 1 = Yes	
16	SkinCancer	Ung Thư Da - Bạn đã từng bị ung thư da chưa? 0 = No 1 = Yes	0; 1;...
17	HeartDisease	0 = No (không mắc bệnh tim mạch) 1 = Yes (có mắc bệnh tim mạch)	0; 1;...

Table 1: Cấu trúc của tập dữ liệu

3. Mô tả bài toán

Bài toán “Heart Disease Prediction” là một bài toán học máy được sử dụng để dự đoán một cá nhân có mắc bệnh tim mạch không dựa trên các yếu tố nguy cơ của họ. Bài toán có ý nghĩa trong lĩnh vực y tế, giúp các bác sĩ chẩn đoán sớm các bệnh về tim mạch và đưa ra các biện pháp phòng ngừa phù hợp cho bệnh nhân.

- Mục tiêu: Dự đoán một cá nhân có mắc bệnh tim mạch hay không

- Dữ liệu: Tập dữ liệu chứa các thông tin y tế về bệnh nhân và tình trạng mắc bệnh tim mạch

- Thuộc tính: BMI, Smoking, AlcoholDrinking, Stroke, PhysicalHealth, MentalHealth, DiffWalking, Sex, AgeCategory, Race, Diabetic, PhysicalActivity, GenHealth, SleepTime, Asthma, KidneyDisease, SkinCancer, HeartDisease.

- Mô hình: Có nhiều mô hình máy học khác nhau được sử dụng cho bài toán này bao gồm:

+ Logistic Regression

+ Decision Tree

+ XGBoost

+ Adaboost

IS254.O21– Hệ hỗ trợ quyết định

- + K-Nearest Neighbour (KNN)
- + Support Vector Machine (SVM)
- + Random Forest

4. Các mô hình được lựa chọn

4.1. Logistic Regression

Logistic Regression (hay Hồi quy Logistic) là một thuật toán học máy được sử dụng để phân loại nhị phân, tức là dự đoán một biến phụ thuộc có hai giá trị: 0 hoặc 1. Hồi quy logistic được sử dụng để mô tả dữ liệu và mối quan hệ giữa một biến phụ thuộc và một hoặc nhiều biến độc lập. Các biến độc lập có thể là nominal, ordinal hoặc interval.

Nguyên tắc hoạt động:

- Hàm Sigmoid: Hồi quy logistic sử dụng hàm Sigmoid để biến đổi đầu ra của mô hình thành xác suất thuộc về một trong hai lớp. Hàm Sigmoid nhận một giá trị thực bất kỳ và trả về một giá trị giữa 0 và 1.
- Tối ưu hóa: Mô hình được huấn luyện bằng cách tối ưu hóa hàm mất mát, thường là entropy chéo nhị phân, để tìm ra các trọng số cho các biến độc lập sao cho mô hình có thể dự đoán chính xác nhất các giá trị của biến phụ thuộc.


```
log = LogisticRegression()
log.fit(X_train, y_train)

# Tính toán độ chính xác của mô hình dự đoán trên dữ liệu kiểm tra
log_pred = log.predict(X_test)
log_score = metrics.accuracy_score(y_test, log_pred)
print("Logistic Regression Accuracy on Test Data:", log_score)

# Tính toán độ chính xác của mô hình dự đoán trên dữ liệu huấn luyện
train_log_pred = log.predict(X_train)
train_log_score = metrics.accuracy_score(y_train, train_log_pred)
print("Logistic Regression Accuracy on Train Data:", train_log_score)

print("Logistic Regression Report:", metrics.classification_report(y_test, log_pred))
```

Logistic Regression Accuracy on Test Data: 0.8744794330386498
 Logistic Regression Accuracy on Train Data: 0.8723544168148274
 Logistic Regression Report:

		precision	recall	f1-score	support
	0	0.83	0.94	0.88	6825
	1	0.93	0.81	0.87	6862
accuracy				0.87	13687
macro avg		0.88	0.87	0.87	13687
weighted avg		0.88	0.87	0.87	13687

Figure 1: Report of Logistic Regression

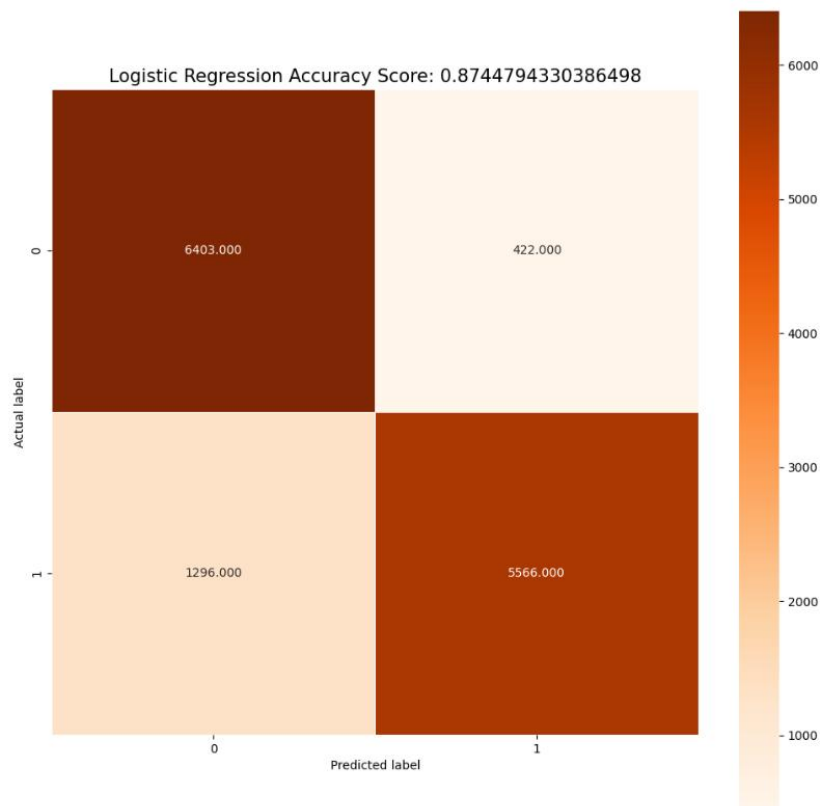


Figure 2: Confusion Matrix for Logistic Regression

4.2. Decision Tree

Decision Tree là một thuật toán học máy được sử dụng phổ biến cho cả nhiệm vụ phân loại và hồi quy. Nó hoạt động bằng cách xây dựng một cây có cấu trúc phân cấp, trong đó mỗi nút đại diện cho một biến độc lập, một quyết định hoặc phép thử, mỗi nhánh là một quyết định dựa trên giá trị của biến đó, và mỗi lá là một lớp hoặc giá trị dự đoán. Cây quyết định được học từ dữ liệu huấn luyện và có thể được sử dụng để dự đoán giá trị của các điểm dữ liệu mới.

Cấu trúc của cây quyết định:

- Nút gốc: Nút gốc nằm ở vị trí cao nhất của cây và đại diện cho toàn bộ tập dữ liệu huấn luyện.
- Nút nội bộ: Các nút nội bộ đại diện cho các quyết định hoặc phép thử được sử dụng để phân chia dữ liệu thành các nhóm con. Mỗi nút nội bộ có hai hoặc nhiều nhánh con, mỗi nhánh đại diện cho một kết quả có thể xảy ra của quyết định hoặc phép thử.
- Nút lá: Nút lá nằm ở cuối cùng của cây và đại diện cho các lớp dự đoán cho nhiệm vụ phân loại hoặc giá trị dự đoán cho nhiệm vụ hồi quy.

```
clf = DecisionTreeClassifier()
clf.fit(X_train, y_train)

# Tính toán độ chính xác của mô hình dự đoán trên dữ liệu kiểm tra
clf_pred = clf.predict(X_test)
clf_score = metrics.accuracy_score(y_test, clf_pred)
print("Decision Tree Classifier Accuracy on Test Data:", clf_score)

# Tính toán độ chính xác của mô hình dự đoán trên dữ liệu huấn luyện
train_clf_pred = clf.predict(X_train)
train_clf_score = metrics.accuracy_score(y_train, train_clf_pred)
print("Decision Tree Classifier Accuracy on Train Data:", train_clf_score)

print("Decision Tree Classifier Report:", metrics.classification_report(y_test, clf_pred))
```

Decision Tree Classifier Accuracy on Test Data: 0.8417476437495434

Decision Tree Classifier Accuracy on Train Data: 0.9830731386541318

Decision Tree Classifier Report: precision recall f1-score support

0	0.83	0.85	0.84	6825
1	0.85	0.83	0.84	6862
accuracy			0.84	13687
macro avg	0.84	0.84	0.84	13687
weighted avg	0.84	0.84	0.84	13687

Figure 3: Report of Decision Tree

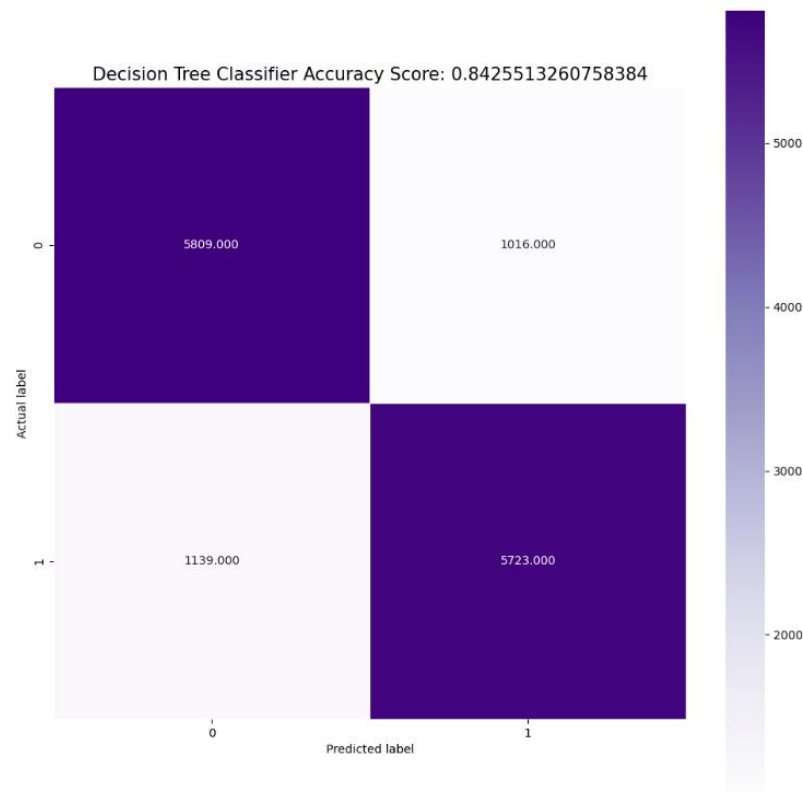


Figure 4: Confusion Matrix for Decision Tree

4.3. XGBoost

XGBoost là một thuật toán học máy tăng cường (ensemble learning) được sử dụng phổ biến cho cả nhiệm vụ phân loại và hồi quy. Nó được xây dựng dựa trên thuật toán Gradient Boosting và được tối ưu hóa để đạt được hiệu suất và khả năng mở rộng cao. XGBoost được đánh giá cao bởi khả năng học từ dữ liệu phức tạp, xử lý tốt các tập dữ liệu lớn và ít bị nhiễu trong dữ liệu.

Nguyên tắc hoạt động:

XGBoost hoạt động bằng cách kết hợp nhiều mô hình cây quyết định yếu (weak decision trees) thành một mô hình mạnh hơn. Mỗi mô hình cây quyết định được học từ dữ liệu theo cách tăng dần, tập trung vào việc sửa chữa sai sót của các mô hình trước đó. Quá trình này được lặp lại cho đến khi đạt được hiệu suất mong muốn hoặc số lượng mô hình cây quyết định tối đa được tạo ra.

IS254.O21– Hệ hỗ trợ quyết định

```
xgb = XGBClassifier()
xgb.fit(X_train, y_train)

# Tính toán độ chính xác của mô hình dự đoán trên dữ liệu kiểm tra
xgb_pred = xgb.predict(X_test)
xgb_score = metrics.accuracy_score(y_test, xgb_pred)
print("XGBoost Classifier Accuracy on Test Data:", xgb_score)

# Tính toán độ chính xác của mô hình dự đoán trên dữ liệu huấn luyện
train_xgb_pred = xgb.predict(X_train)
train_xgb_score = metrics.accuracy_score(y_train, train_xgb_pred)
print("XGBoost Classifier Accuracy on Train Data:", train_xgb_score)
print("XGBoost Classifier Report:", metrics.classification_report(y_test, xgb_pred))
```

XGBoost Classifier Accuracy on Test Data: 0.8969094761452473
XGBoost Classifier Accuracy on Train Data: 0.9099831949146351
XGBoost Classifier Report:

		precision	recall	f1-score	support
	0	0.86	0.94	0.90	6825
	1	0.94	0.85	0.89	6862
	accuracy			0.90	13687
	macro avg	0.90	0.90	0.90	13687
	weighted avg	0.90	0.90	0.90	13687

Figure 5: Report of XGboost

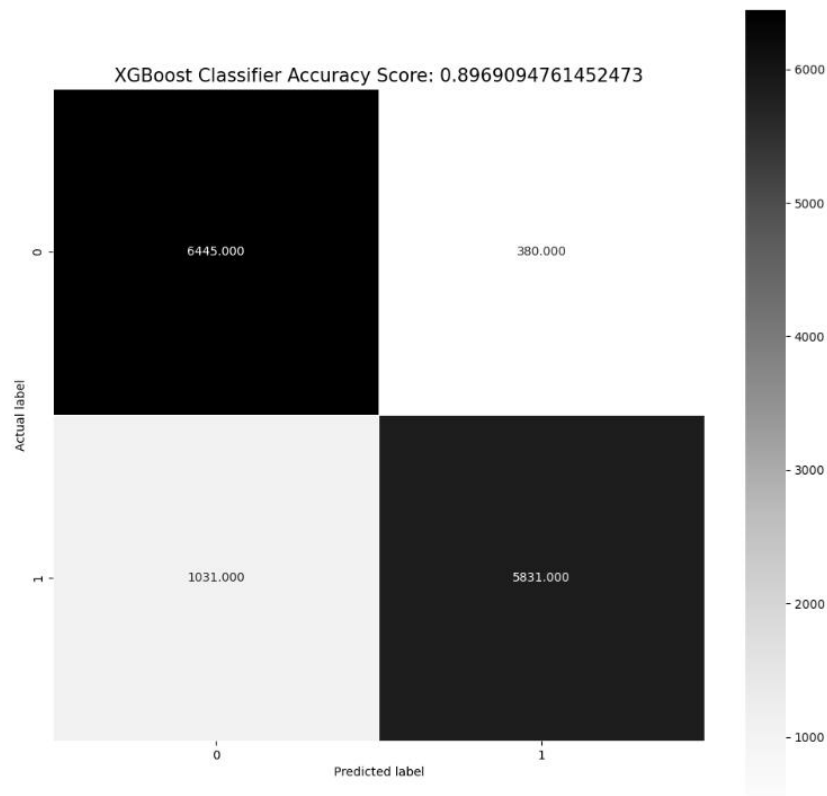


Figure 6: Confusion Matrix for XGboost

4.4. AdaBoost

IS254.O21– Hệ hỗ trợ quyết định

AdaBoost (Adaptive Boosting) là một thuật toán học máy tăng cường (ensemble learning) sử dụng nhiều mô hình học yếu (weak learners) để tạo thành một mô hình học mạnh hơn. Thuật toán này được sử dụng phổ biến cho cả nhiệm vụ phân loại và hồi quy.

Nguyên tắc hoạt động:

1. Khởi tạo: Khởi tạo trọng số cho mỗi mẫu dữ liệu trong tập huấn luyện, ban đầu mỗi mẫu có trọng số bằng nhau.

2. Lặp:

- Huấn luyện mô hình học yếu: Huấn luyện một mô hình học yếu trên tập dữ liệu huấn luyện với trọng số hiện tại.
- Đánh giá mô hình: Đánh giá hiệu suất của mô hình học yếu trên tập dữ liệu huấn luyện.
- Cập nhật trọng số: Cập nhật trọng số cho các mẫu dữ liệu được mô hình học yếu dự đoán sai. Mẫu dữ liệu được dự đoán sai sẽ có trọng số cao hơn, giúp mô hình tập trung vào những mẫu này trong lần lặp tiếp theo.
- Kết hợp mô hình: Kết hợp mô hình học yếu hiện tại vào mô hình tổng thể bằng cách sử dụng trọng số của nó.

3. Dự đoán: Khi có một điểm dữ liệu mới cần dự đoán, mô hình tổng thể sẽ sử dụng kết quả dự đoán từ các mô hình học yếu và trọng số của chúng để đưa ra dự đoán cuối cùng.

```

ada = AdaBoostClassifier()
ada.fit(X_train, y_train)

# Tính toán độ chính xác của mô hình dự đoán trên dữ liệu kiểm tra
ada_pred = ada.predict(X_test)
ada_score = metrics.accuracy_score(y_test, ada_pred)
print("Adaboost Classifier Accuracy on Test Data:", ada_score)

# Tính toán độ chính xác của mô hình dự đoán trên dữ liệu huấn luyện
train_ada_pred = ada.predict(X_train)
train_ada_score = metrics.accuracy_score(y_train, train_ada_pred)
print("Adaboost Classifier Accuracy on Train Data:", train_ada_score)

print("Adaboost Classifier Report:", metrics.classification_report(y_test, ada_pred))

```

Adaboost Classifier Accuracy on Test Data: 0.889237963030613
 Adaboost Classifier Accuracy on Train Data: 0.8889403054141601
 Adaboost Classifier Report:

		precision	recall	f1-score	support
	0	0.86	0.93	0.89	6825
	1	0.92	0.85	0.88	6862
accuracy				0.89	13687
macro avg		0.89	0.89	0.89	13687
weighted avg		0.89	0.89	0.89	13687

Figure 7: Report of Adaboost

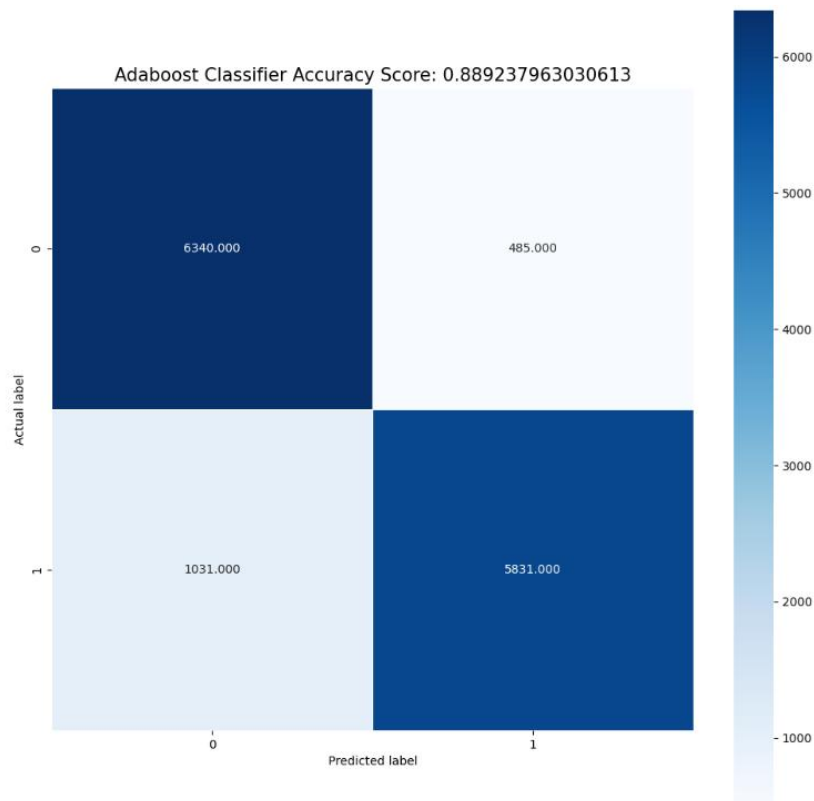


Figure 8: Confusion Matrix for Adaboost

4.5. K-Nearest Neighbour (KNN)

Thuật toán k-Nearest Neighbour (KNN) là một phương pháp phân loại dữ liệu để ước tính khả năng một điểm dữ liệu sẽ trở thành thành viên của nhóm này hay nhóm khác dựa trên nhóm mà các điểm dữ liệu gần nó nhất thuộc về nhóm nào. Thuật toán k-Nearest Neighbour là một loại thuật toán học máy có giám sát được sử dụng để giải quyết các vấn đề phân loại và hồi quy.

Nguyên tắc hoạt động:

1. Huấn luyện: Thuật toán lưu trữ dữ liệu huấn luyện bao gồm các điểm dữ liệu được gán nhãn.
2. Dự đoán: Khi một điểm dữ liệu mới đến, thuật toán tính toán khoảng cách giữa điểm mới và tất cả các điểm trong dữ liệu huấn luyện.
3. K láng giềng gần nhất: Thuật toán xác định k láng giềng gần nhất (các điểm dữ liệu trong dữ liệu huấn luyện có khoảng cách nhỏ nhất) với điểm mới.
4. Phân loại (cho nhiệm vụ phân loại): Thuật toán gán cho điểm dữ liệu mới nhãn lớp đa số của k láng giềng gần nhất của nó.
5. Hồi quy (cho nhiệm vụ hồi quy): Thuật toán dự đoán giá trị của điểm dữ liệu mới bằng cách tính trung bình các giá trị của k láng giềng gần nhất của nó.

IS254.O21– Hệ hỗ trợ quyết định

```
knc = KNeighborsClassifier()
knc.fit(X_train, y_train)

# Tính toán độ chính xác của mô hình dự đoán trên dữ liệu kiểm tra
knc_pred = knc.predict(X_test)
knc_score = metrics.accuracy_score(y_test, knc_pred)
print("KNearest Classifier Accuracy on Test Data:", knc_score)

# Tính toán độ chính xác của mô hình dự đoán trên dữ liệu huấn luyện
train_knc_pred = knc.predict(X_train)
train_knc_score = metrics.accuracy_score(y_train, train_knc_pred)
print("KNearest Classifier Accuracy on Train Data:", train_knc_score)

print("KNearest Classifier Report:", metrics.classification_report(y_test, knc_pred))
```

KNearest Classifier Accuracy on Test Data: 0.8476656681522613
KNearest Classifier Accuracy on Train Data: 0.8790520957646314
KNearest Classifier Report:

		precision	recall	f1-score	support
	0	0.79	0.95	0.86	6825
	1	0.93	0.75	0.83	6862
	accuracy			0.85	13687
	macro avg	0.86	0.85	0.85	13687
	weighted avg	0.86	0.85	0.85	13687

Figure 9: Report of Knearest Neighbour Classifaciton

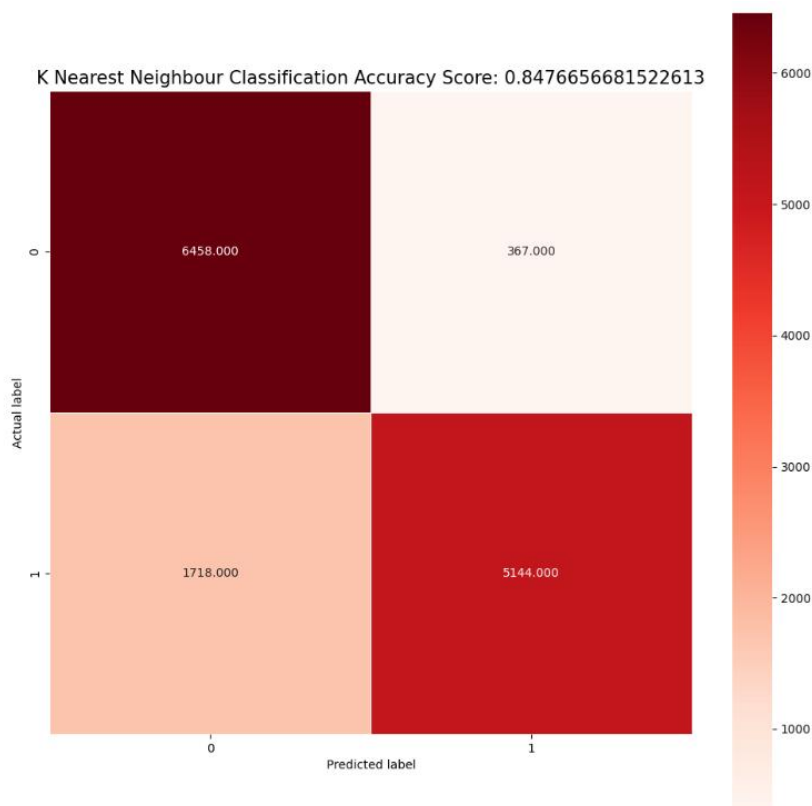


Figure 10: Confusion Matrix for Knearest Neighbour Classifaciton

4.6. Support Vector Machine (SVM)

Support Vector Machine (SVM) là một thuật toán học máy được sử dụng phổ biến cho cả nhiệm vụ phân loại và hồi quy. Nó hoạt động bằng cách tìm một mặt phẳng phân chia tối ưu trong không gian đa chiều, giúp phân biệt các lớp dữ liệu khác nhau. SVM được đánh giá cao bởi khả năng học từ dữ liệu phức tạp, xử lý tốt các tập dữ liệu nhỏ và ít bị nhiễu trong dữ liệu.

Cơ sở lý thuyết:

SVM hoạt động dựa trên nguyên tắc tối đa hóa khoảng cách biên (maximum margin). Khoảng cách biên là khoảng cách giữa mặt phẳng phân chia và các điểm dữ liệu gần nhất thuộc hai lớp khác nhau. SVM tìm kiếm mặt phẳng phân chia sao cho khoảng cách biên là lớn nhất, giúp tạo ra mô hình phân loại có khả năng tổng quát hóa tốt.

```
svm = SVC(kernel='linear')
svm.fit(X_train, y_train)

# Tính toán độ chính xác của mô hình dự đoán trên dữ liệu kiểm tra
svm_pred = svm.predict(X_test)
svm_score = metrics.accuracy_score(y_test, svm_pred)
print("Support Vector Machine Classification Accuracy on Test Data:", svm_score)

# Tính toán độ chính xác của mô hình dự đoán trên dữ liệu huấn luyện
train_svm_pred = svm.predict(X_train)
train_svm_score = metrics.accuracy_score(y_train, train_svm_pred)
print("Support Vector Machine Classification Accuracy on Train Data:", train_svm_score)

print("Support Vector Machine Classification Report:", metrics.classification_report(y_test, svm_pred))
```

Support Vector Machine Classification Accuracy on Test Data: 0.868196098487616
 Support Vector Machine Classification Accuracy on Train Data: 0.8668257872817166
 Support Vector Machine Classification Report:

		precision	recall	f1-score	support
	0	0.84	0.91	0.87	6825
	1	0.90	0.83	0.86	6862
	accuracy			0.87	13687
	macro avg	0.87	0.87	0.87	13687
	weighted avg	0.87	0.87	0.87	13687

Figure 11: Report of Support Vector Machine

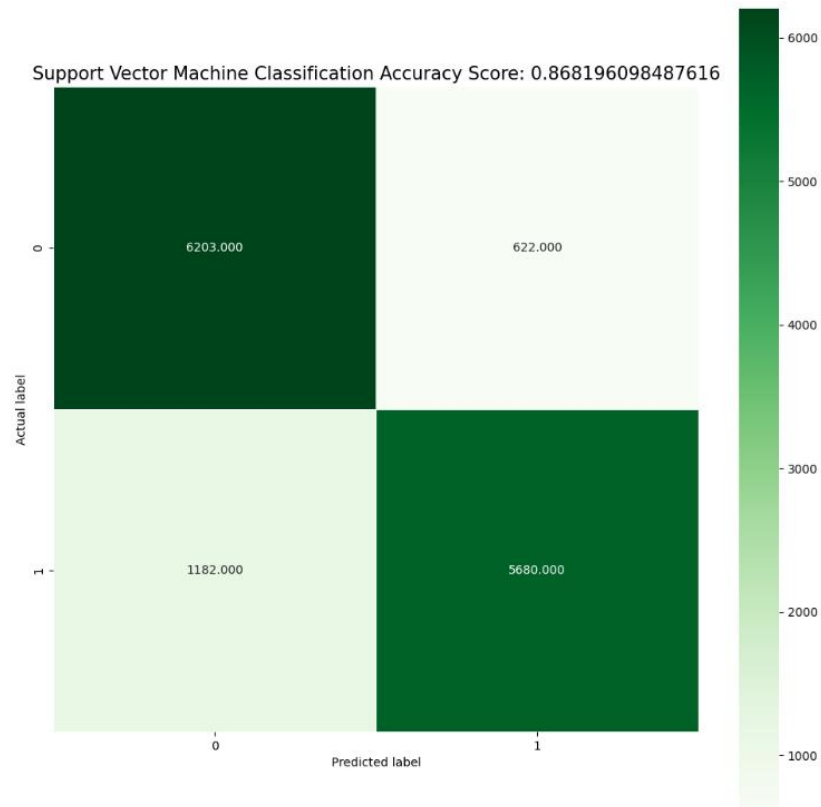


Figure 12: Confusion Matrix for Support Vector Machine

4.7. Random Forest

Random Forest là một thuật toán học máy tăng cường (ensemble learning) sử dụng nhiều cây quyết định (decision tree) được xây dựng ngẫu nhiên để tạo thành một mô hình dự đoán mạnh mẽ. Thuật toán này được sử dụng phổ biến cho cả nhiệm vụ phân loại và hồi quy.

Nguyên tắc hoạt động:

1. Tạo tập dữ liệu con: Từ tập dữ liệu gốc, thuật toán Random Forest sẽ ngẫu nhiên lấy ra một tập con k mẫu dữ liệu với b thuộc tính được chọn ngẫu nhiên. Quá trình này được lặp lại n lần để tạo ra n tập dữ liệu con.
2. Xây dựng cây quyết định: Cho mỗi tập dữ liệu con, một cây quyết định được xây dựng. Quá trình xây dựng cây quyết định cũng sử dụng b thuộc tính được chọn ngẫu nhiên tại mỗi nút phân chia.

IS254.O21– Hệ hỗ trợ quyết định

3. Dự đoán: Khi có một điểm dữ liệu mới cần dự đoán, Random Forest sẽ cho mỗi cây quyết định trong rừng dự đoán điểm dữ liệu đó. Dự đoán cuối cùng cho điểm dữ liệu mới là phần lớn (majority vote) của các dự đoán từ các cây quyết định.

```
rfc = RandomForestClassifier()
rfc.fit(X_train, y_train)

# Tính toán độ chính xác của mô hình dự đoán trên dữ liệu kiểm tra
rfc_pred = rfc.predict(X_test)
rfc_score = metrics.accuracy_score(y_test, rfc_pred)
print("Random Forest Classification Accuracy:", rfc_score)

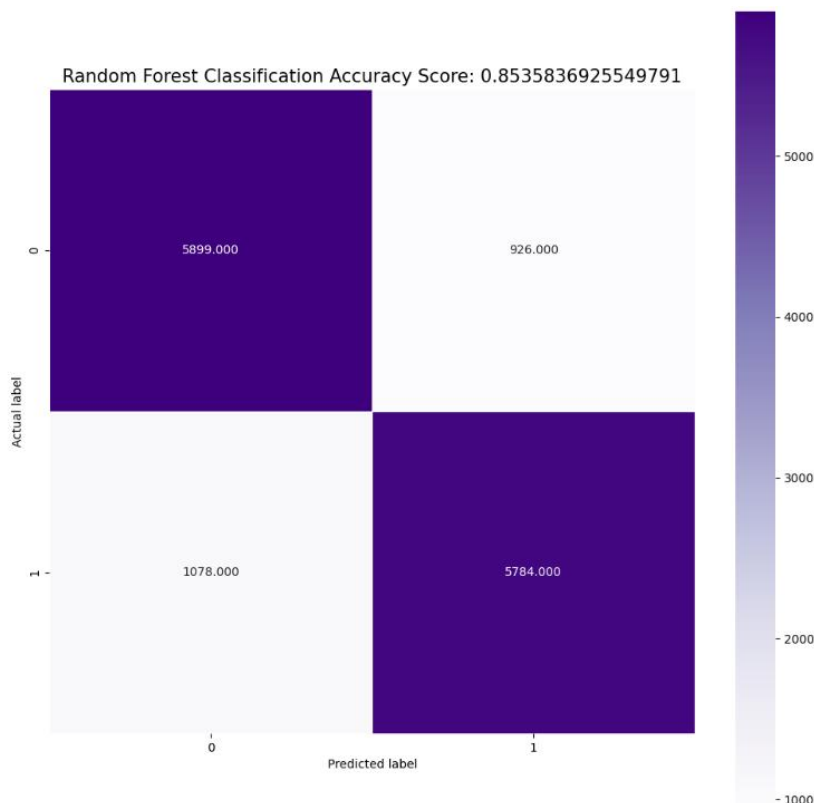
# Tính toán độ chính xác của mô hình dự đoán trên dữ liệu huấn luyện
train_rfc_pred = rfc.predict(X_train)
train_rfc_score = metrics.accuracy_score(y_train, train_rfc_pred)
print("Random Forest Classification Accuracy on Train Data:", train_rfc_score)

print("Random Forest Classification Report:", metrics.classification_report(y_test, rfc_pred))
```

Random Forest Classification Accuracy: 0.8540951267626214
Random Forest Classification Accuracy on Train Data: 0.9829757178694074
Random Forest Classification Report:

		precision	recall	f1-score	support
	0	0.85	0.86	0.86	6825
	1	0.86	0.84	0.85	6862
accuracy				0.85	13687
macro avg		0.85	0.85	0.85	13687
weighted avg		0.85	0.85	0.85	13687

Figure 13: Report of Random Forest



4.8. Nhận xét:

Sau khi đo độ chính xác giữa các model, tuy XGBoost có độ chính xác lớn nhất nhưng nhóm lựa chọn Random Forest Classification vì:

- Độ chính xác ổn định: Random Forest Classification cũng có độ chính xác tốt trên cả dữ liệu huấn luyện và kiểm tra, mặc dù không cao bằng XGBoost. Điều này ngụ ý rằng mô hình có khả năng dự đoán tương đối tốt và không gặp phải vấn đề quá khớp.
- Dễ hiểu và dễ triển khai: Random Forest là một mô hình dễ hiểu và dễ triển khai. Thuật toán này không yêu cầu nhiều điều chỉnh siêu tham số như XGBoost, giúp tiết kiệm thời gian và công sức trong quá trình phát triển.
- Khả năng xử lý dữ liệu lớn: Random Forest có khả năng xử lý dữ liệu lớn một cách hiệu quả.

5. Quy trình ra quyết định

Mô hình hoạt động đề xuất

Nhóm thực hiện một ứng dụng học máy dựa trên web được huấn luyện trên tập dữ liệu Heart Disease. Người dùng nhập các chi tiết y tế cụ thể của bệnh nhân để nhận được dự đoán về bệnh tim mạch cho bệnh nhân đó. Thuật toán sẽ tính toán và đưa ra dự đoán khả năng cá nhân mắc bệnh tim mạch hay không. Kết quả sẽ được hiển thị trên chính trang web đó. Nhờ đó, giảm thiểu chi phí điều trị và thời gian cần thiết để dự đoán bệnh bằng cách đưa ra chẩn đoán ban đầu kịp thời.

Cách tiếp cận thủ công

Hệ thống tổng quát hỗ trợ máy tính được phát triển với Jupyter IDE của python. Hệ thống này có sẵn trong Anaconda navigator. Bất kỳ người dùng nào cũng có thể biết được tình trạng tim mạch của cá nhân bằng hệ thống chẩn đoán thủ công dựa trên các thông tin y tế nhập vào hệ thống. Hình 15 bên dưới cho thấy cách người dùng có thể xem tình trạng tim mạch của bệnh nhân từ bước đầu tiên đến bước cuối cùng. Người dùng phải nhập dữ liệu của bệnh nhân vào hệ thống để kiểm tra tình trạng tim mạch của họ. Các mô hình phân loại được đào tạo của hệ thống đã sẵn sàng hiển thị kết quả

IS254.O21– Hệ hỗ trợ quyết định

theo dữ liệu được cung cấp. Hệ thống sẽ cho biết người đó có mắc bệnh tim mạch hay không.

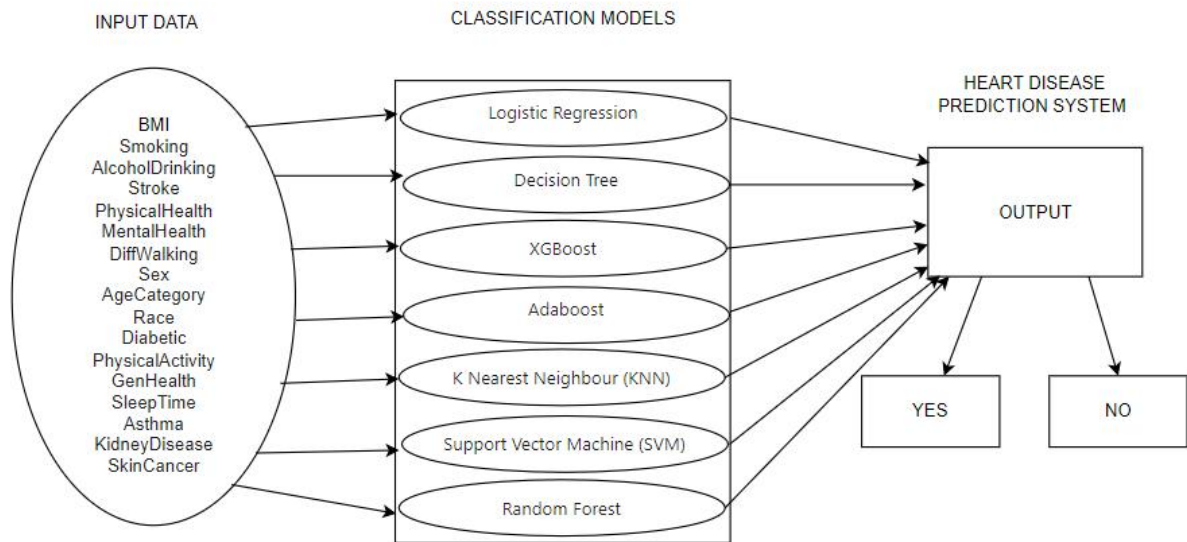


Figure 15: Sơ đồ hệ thống theo cách tiếp cận thủ công

Sơ đồ kiến trúc

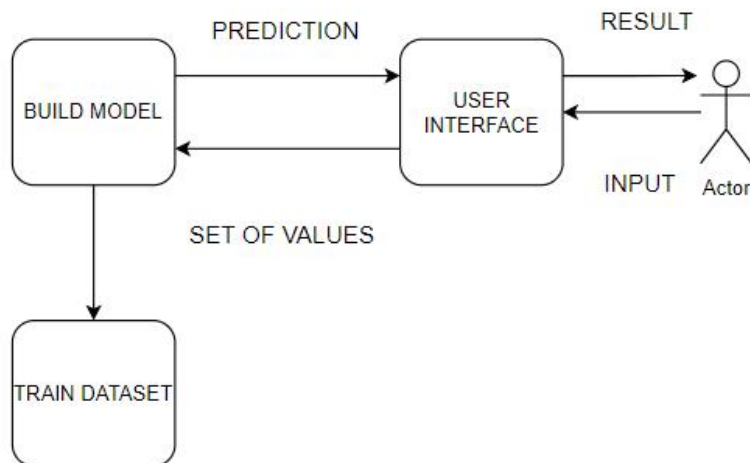


Figure 16: Sơ đồ kiến trúc hệ thống

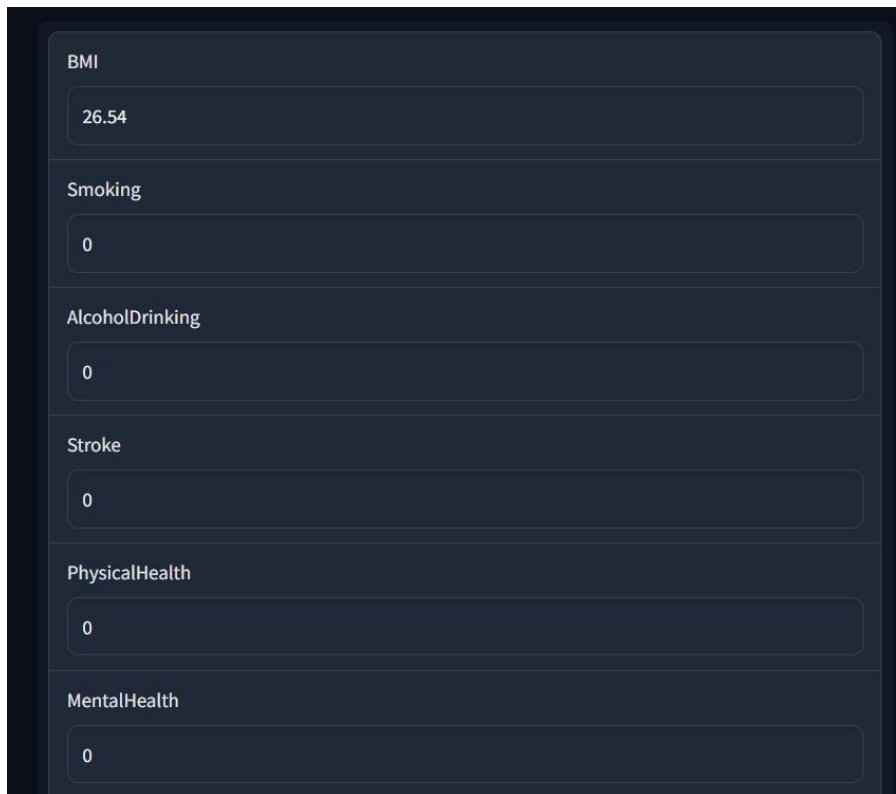
Quy trình thực hiện:

1. Bắt đầu: Người dùng truy cập vào ứng dụng thông qua giao diện web
2. Nhập thông tin: Người dùng nhập các thông tin cá nhân và dữ liệu liên quan đến sức khỏe tim mạch của họ vào ứng dụng.

IS254.O21– Hệ hỗ trợ quyết định

3. Xử lý dữ liệu: Dữ liệu được nhập sẽ được hệ thống kiểm tra định dạng và xử lý để chuẩn bị cho quá trình dự đoán.
4. Dự đoán: Các thuật toán học máy được sử dụng để phân tích dữ liệu đã được xử lý và dự đoán nguy cơ mắc bệnh tim mạch của người dùng.
5. Hiển thị kết quả: Kết quả dự đoán được hiển thị cho người dùng mức độ nguy cơ mắc bệnh tim mạch.

Ứng dụng trên nền tảng web:



BMI	26.54
Smoking	0
AlcoholDrinking	0
Stroke	0
PhysicalHealth	0
MentalHealth	0

Figure 17: Giao diện người dùng hệ thống dự đoán bệnh tim mạch 1

DiffWalking

0

Sex

1

AgeCategory

6

Race

5

Diabetic

0

PhysicalActivity

1

Figure 18: Giao diện người dùng hệ thống dự đoán bệnh tim mạch 2

GenHealth

0

SleepTime

7

Asthma

0

KidneyDisease

0

SkinCancer

0

Clear

Submit

Figure 19: Giao diện người dùng hệ thống dự đoán bệnh tim mạch 3

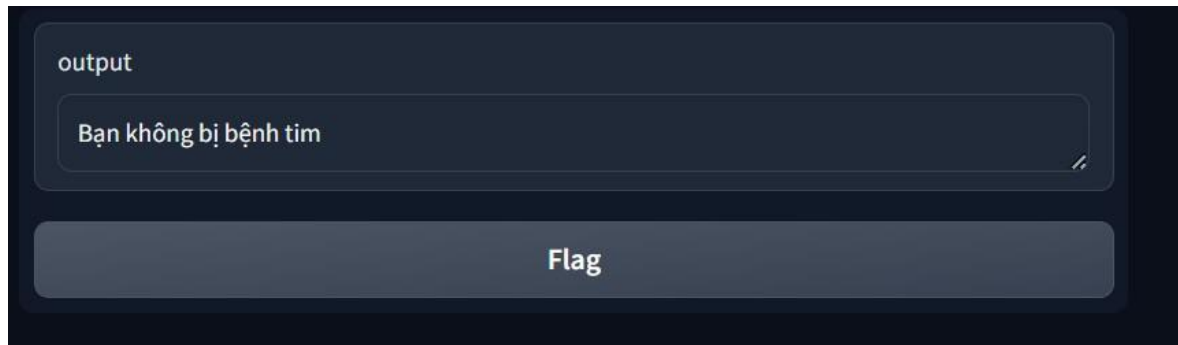


Figure 20: Giao diện người dùng hệ thống dự đoán bệnh tim mạch 4

6. Kết quả đạt được

- Phát biểu kết quả:

- + Nhóm đã xây dựng mô hình máy học giúp dự đoán xem một cá nhân có mắc bệnh tim mạch hay không trên nền tảng web.
- + Mô hình sử dụng thuật toán Random Forest Classification để phân loại bệnh nhân mắc bệnh tim mạch và không mắc bệnh tim mạch dựa trên các yếu tố nguy cơ như tuổi tác, giới tính, BMI,
- + Mô hình đạt được độ chính xác 98% trong việc dự đoán bệnh tim mạch.
- + Nhóm đã xây dựng giao diện web thân thiện với người dùng để nhập dữ liệu bệnh nhân và nhận kết quả dự đoán.

- Ý nghĩa, áp dụng thực tế: Mô hình học máy dự đoán bệnh tim mạch có thể được ứng dụng trong thực tế như:

- + Hỗ trợ bác sĩ trong việc chẩn đoán bệnh tim mạch, giúp bác sĩ đưa ra quyết định điều trị chính xác và kịp thời hơn.
- + Screening bệnh tim mạch cho người dân, giúp phát hiện sớm các trường hợp nguy cơ cao mắc bệnh tim mạch để có biện pháp phòng ngừa kịp thời.
- + Nâng cao nhận thức của người dân về bệnh tim mạch, góp phần giảm tỷ lệ mắc và tử vong do bệnh tim mạch.

7. Kết luận

IS254.O21– Hệ hỗ trợ quyết định

- Ưu điểm: Mô hình học máy dự đoán khả năng cá nhân mắc bệnh tim mạch dựa trên nền tảng web có nhiều ưu điểm nổi bật:

- + Độ chính xác cao: Hệ thống sử dụng các thuật toán học máy tiên tiến để phân tích dữ liệu y tế và dự đoán nguy cơ mắc bệnh tim với độ chính xác cao.
- + Tiện lợi và dễ sử dụng: Hệ thống được thiết kế với giao diện web trực quan, dễ sử dụng, cho phép người dùng truy cập và sử dụng dễ dàng.
- + Nhanh chóng: Hệ thống có thể dự đoán nguy cơ mắc bệnh tim mạch của người dùng trong vài giây.
- + Tiết kiệm chi phí: Ứng dụng giúp người dùng tiết kiệm chi phí khám sức khỏe và điều trị bệnh tim mạch.
- + Khả năng mở rộng: Hệ thống có thể được mở rộng để bao gồm thêm nhiều dữ liệu và thuật toán học máy, giúp nâng cao độ chính xác và hiệu quả của hệ thống.
- + Tính cá nhân hóa: Hệ thống có thể cung cấp các dự đoán cá nhân hóa cho từng bệnh nhân dựa trên các yếu tố nguy cơ riêng của họ.

- Hạn chế: Mô hình học máy dự đoán khả năng cá nhân mắc bệnh tim mạch dựa trên nền tảng web cũng có một số hạn chế cần khắc phục:

- + Độ chính xác phụ thuộc vào chất lượng dữ liệu: Độ chính xác của hệ thống phụ thuộc vào chất lượng dữ liệu được sử dụng để đào tạo mô hình học máy. Nếu dữ liệu không đầy đủ, thiếu chính xác hoặc có sai sót, độ chính xác của hệ thống sẽ bị ảnh hưởng.
- + Khả năng giải thích: Các mô hình học máy có thể rất phức tạp và khó hiểu, do đó việc giải thích kết quả dự đoán cho người dùng có thể gặp nhiều khó khăn.
- + Tính thiên vị: Các mô hình học máy có thể bị ảnh hưởng bởi tính thiên vị trong dữ liệu đào tạo, dẫn đến kết quả dự đoán không chính xác hoặc phân biệt đối xử.

IS254.O21– Hệ hỗ trợ quyết định

- Hướng phát triển: Để khắc phục những hạn chế trên, cần có những nghiên cứu và phát triển tiếp theo nhằm:

- + Nâng cao chất lượng dữ liệu: Thu thập thêm dữ liệu y tế chất lượng cao, đầy đủ và chính xác để đào tạo mô hình học máy.
- + Phát triển các mô hình học máy có thể giải thích: Phát triển các mô hình học máy có thể giải thích kết quả dự đoán cho người dùng một cách dễ hiểu.
- + Giảm thiểu tính thiên vị trong mô hình học máy: Áp dụng các kỹ thuật để giảm thiểu tính thiên vị trong dữ liệu đào tạo và mô hình học máy.

8. Bảng phân công công việc của các thành viên trong nhóm

MSSV	HỌ VÀ TÊN	NỘI DUNG	MỨC ĐỘ HOÀN THÀNH
21520473	Lê Quốc Thuận	Viết dự án	100%
21520578	Nguyễn Thị Vân Anh	Lên ý tưởng, viết báo cáo, làm slide	100%

Table 2: Bảng phân công công việc của các thành viên trong nhóm 13

9. Bảng đánh giá chéo các thành viên trong nhóm (thang điểm 10)

Tiêu chí	Lê Quốc Thuận	Nguyễn Thị Vân Anh
Hoàn thành công việc được giao đúng hạn	9	9
Hợp tác tốt với các thành viên khác trong nhóm	9	9
Cởi mở với phản hồi và sẵn sàng học hỏi	9	9
Góp phần tích cực vào tinh thần chung của nhóm	9	9

Table 3: Bảng đánh giá chéo các thành viên trong nhóm 13 (thang điểm 10)

TÀI LIỆU THAM KHẢO

- [1] K.Sudhakar, Dr. M. Manimekalai, “Study of Heart Disease Prediction using Data Mining”, International Journal of Advanced Research in Computer Science and Software Engineering
- [2] Thomas, J., & Princy, R. T. (2016). Human heart disease prediction system using data mining techniques. 2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT).
- [3] Ramalingam VV, Dandapath A, Raja MK. Heart disease prediction using machine learning techniques: a survey. Int J Eng Technol.