TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

LÝ TUẤN AN – 52000620

NGUYỄN THÀNH AN – 52000621

PHÙNG PHÚC HẬU – 52000443

PHÂN TÍCH VÀ HỖ TRỢ RA QUYẾT ĐỊNH TRONG DỰ ĐOÁN BỆNH TIM

BÁO CÁO CUỐI KỲ

HỆ THỐNG

THƯƠNG MẠI THÔNG MINH

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

LÝ TUẤN AN – 52000620

NGUYỄN THÀNH AN – 52000621

PHÙNG PHÚC HẬU – 52000443

PHÂN TÍCH VÀ HỖ TRỢ RA QUYẾT ĐỊNH TRONG DỰ ĐOÁN BỆNH TIM

BÁO CÁO CUỐI KỲ

HỆ THỐNG

THƯƠNG MẠI THÔNG MINH

Người hướng dẫn

ThS. Dương Hữu Phúc

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023

LỜI CẢM ƠN

Trong suốt thời gian qua, nhờ sự giảng dạy tận tâm của quý Thầy Cô khoa Công Nghệ Thông Tin, trường Đại học Tôn Đức Thắng, chúng em đã học hỏi được rất nhiều điều bổ ích và tích lũy cho mình một số kiến thức để hoàn thành bài báo cáo này. Nhóm em xin chân thành cảm ơn.

Nhóm em xin cảm ơn thầy Dương Hữu Phúc đã tận tình chỉ bảo chúng em qua những buổi học tại lớp, thầy đã chỉ nhóm em cách thức làm bài, chỉ điểm những chỗ còn sai sót chưa phù hợp cũng như phải làm sao để trình bày bố cục đẹp. Nếu không có những lời hướng dẫn, dạy bảo của thầy thì bài thu hoạch của nhóm em cũng rất khó để hoàn thiện. Một lần nữa chúng em xin chân thành cảm ơn thầy.

Bước đầu đi vào thực tế với nền kiến thức mở rộng, kiến thức của nhóm em còn hạn chế và nhiều bỡ ngỡ. Vì thế, trong quá trình biên soạn khó tránh những sai sót, nhóm em rất mong nhận được những ý kiến đóng góp quý báu của thầy và các bạn để bài báo cáo hoàn thiện hơn.

ĐỒ ÁN ĐƯỢC HOÀN THÀNH
TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Chúng tôi xin cam đoan đây là sản phẩm đồ án của riêng chúng tôi và được sự hướng dẫn của ThS Dương Hữu Phúc. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình. Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 05 tháng 01 năm 2024

Tác giả

(ký tên và ghi rõ họ tên)

An

Lý Tuấn An

An

Nguyễn Thành An

Hậu

Phùng Phúc Hậu

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

Phần xác nhận của GV hướng dẫn

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày tháng năm
(kí và ghi họ tên)

Phần đánh giá của GV chấm bài

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày tháng năm
(kí và ghi họ tên)

MỤC LỤC

DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT

ML - Machine Learning

DT - Dataset

LG - Logistic Regression

KNN - K-Nearest Neighbors

DTC - Decision Tree Classical

SVM - Support Vector Machine

DANH MỤC CÁC BẢNG BIỂU VÀ HÌNH ẢNH

CHƯƠNG I: TỔNG QUAN

1.1 Giới thiệu đề tài

- Trong số tất cả các cơ quan, trái tim là một bộ phận quan trọng của cơ thể chúng ta. Tim đập khoảng 2,5 tỷ lần trong suốt cuộc đời trung bình, đẩy hàng triệu gallon máu đến mọi bộ phận của cơ thể.
- Trong thời đại này, bệnh tim mạch ngày càng gia tăng do lối sống và thực phẩm hiện đại. Việc chẩn đoán bệnh tim là một nhiệm vụ đầy thách thức. Mô hình phân loại này sẽ dự đoán bệnh nhân có mắc bệnh tim hay không dựa trên các tình trạng/triệu chứng khác nhau của cơ thể họ.
- Đề tài sử dụng học máy để dự đoán bệnh tim mạch dựa trên các chỉ số cơ bản của bệnh nhân là một chủ đề quan trọng trong lĩnh vực y học và có thể đem lại những ứng dụng thực tiễn đáng kể. Các chỉ số cơ bản mà bạn có thể sử dụng để dự đoán bệnh tim mạch có thể bao gồm:
  - **Dữ liệu Lâm sàng cơ bản:** Như huyết áp, mức độ đường huyết, cholesterol, hàm lượng triglycerides, hành vi hút thuốc lá, BMI (Chỉ số khối cơ thể), và lịch sử bệnh lý.
  - **Chỉ số EKG (Electrocardiogram):** Các thông số từ EKG như nhịp tim, hình dạng sóng, và bất thường trong sóng điện có thể được sử dụng để dự đoán bệnh tim mạch.
  - **Thông tin về tuổi, giới tính và di truyền:** Các yếu tố này có thể đóng vai trò quan trọng trong việc dự đoán nguy cơ bệnh tim mạch.
  - **Lịch sử y tế:** Bao gồm lịch sử bệnh tim mạch trong gia đình, các vấn đề sức khỏe đã từng có và các phác đồ điều trị trước đó.
- Cách tiếp cận thông thường là thu thập dữ liệu từ các bệnh viện hoặc cơ sở y tế, sau đó sử dụng các thuật toán học máy như Logistic Regression, Random Forest, Support Vector Machines (SVM), hoặc Neural Networks để phân loại bệnh nhân thành các nhóm có nguy cơ bệnh tim mạch cao và thấp dựa trên thông tin này.
- Một số đặc điểm quan trọng sau cần lưu ý:

- **Tiền xử lý dữ liệu:** Đôi khi, dữ liệu có thể không đầy đủ hoặc có nhiễu, việc xử lý dữ liệu để loại bỏ giá trị ngoại lai (outliers), điền các giá trị thiếu (missing values), và chuẩn hóa dữ liệu có thể là cần thiết.
- **Đánh giá mô hình:** Để đảm bảo mô hình học máy là hiệu quả và có thể áp dụng trong thực tế, cần phải thực hiện các phương pháp đánh giá như cross-validation, ROC curves, và confusion matrices để đo lường hiệu suất của mô hình.
- **Phân tích diễn giải:** Việc hiểu và diễn giải các biến quan trọng trong mô hình có thể giúp cải thiện hiểu biết về mối quan hệ giữa các yếu tố và bệnh tim mạch.
- **Bảo vệ quyền riêng tư:** Trong việc sử dụng dữ liệu y tế, việc bảo vệ thông tin cá nhân và tuân thủ các quy định về quyền riêng tư là rất quan trọng.

1.2 Phương pháp nghiên cứu

- Xác định mục tiêu nghiên cứu:
  - Phân tích hành vi tiêu dùng: Thu thập dữ liệu về những chỉ số/tình trạng gặp phải trên cơ thể người, bao gồm giới tính, đường huyết, huyết áp, cholesterol, đau thắt ngực, …
  - Sử dụng Tableau để trực quan hóa dữ liệu, xác định mối liên hệ giữa các chỉ số và khả năng chuẩn đoán bệnh tim.
- Thu thập dữ liệu:
  - Lựa chọn mẫu nghiên cứu: Tập trung vào một nhóm người tham gia nghiên cứu có đủ thông tin về các chỉ số cơ thể và lịch sử bệnh lý về tim mạch.
  - Thu thập chỉ số cơ thể: Thu thập dữ liệu về huyết áp, nhịp tim, cholesterol, BMI, glucose level, và các chỉ số cơ thể khác thông qua thiết bị y tế hoặc các phương pháp đo lường chuẩn.
- Xử lý và phân tích dữ liệu:
  - Xử lý dữ liệu: Kiểm tra và xử lý dữ liệu thiếu sót, loại bỏ các giá trị ngoại lệ, chuẩn hóa dữ liệu nếu cần.
  - Trực quan hóa:
    - Import dữ liệu vào Tableau và tạo các biểu đồ trực quan hóa để khám phá mối quan hệ giữa các chỉ số cơ thể và bệnh tim.

- Tạo biểu đồ dạng scatterplot để thấy mối tương quan giữa các chỉ số.
- Áp dụng các kỹ thuật trực quan hóa như heatmap, biểu đồ đường, biểu đồ cột để so sánh sự biến đổi của các chỉ số theo thời gian hoặc theo nhóm người khác nhau.

- Phân tích và kết luận:
  - Phân tích kết quả: Phân tích biểu đồ và số liệu để nhận diện các mẫu, xu hướng, mối liên hệ giữa các chỉ số cơ thể và bệnh tim.
  - Kết luận: Đưa ra các kết luận hoặc dự đoán về khả năng chuẩn đoán bệnh tim dựa trên dữ liệu và mối liên hệ phát hiện được.
  - Lập đề xuất: Dựa trên kết quả, đề xuất các phương pháp chuẩn đoán hoặc quản lý bệnh tim dựa trên thông tin từ các chỉ số cơ thể.

1.3 Kết quả dự kiến

- Cung cấp thông tin hữu ích và nhận thức sâu sắc về mối quan hệ giữa chỉ số cơ thể và bệnh tim, hỗ trợ việc chuẩn đoán và điều trị bệnh tim.
- Đóng góp vào lĩnh vực nghiên cứu y học về khả năng sử dụng dữ liệu chỉ số cơ thể để dự đoán bệnh tim.

Hình 1. 1 Dự đoán bệnh tim bằng ML

CHƯƠNG II: THU THẬP VÀ XỬ LÝ DATASET

2.1 Xây dựng dataset

2.1.1 Mô tả dataset

- Bộ dữ liệu cung cấp thông tin đa dạng về chỉ số cơ thể và yếu tố rủi ro để đánh giá nguy cơ bệnh tim của một người.
- Việc sử dụng các công cụ như Tableau để trực quan hóa và phân tích dataset này có thể giúp xác định mối liên hệ giữa các chỉ số và khả năng dự đoán bệnh tim.
- Một thông tin cơ bản như: Giới tính, huyết áp, nhịp tim, mức cholesterol, chỉ số cơ thể, tiền sử bệnh án, thói quen, …

2.1.2 Đặc tả dữ liệu

STT	Tập dữ liệu	Mô tả
1	age	Độ tuổi bệnh nhân (tính bằng năm)
2	gender	Giới tính bệnh nhân: nứ (0) - nam (1)
3	cp (chest pain type)	Loại đau ngực: - typical angina – đau ngực điển hình (0): Đau ngực liên quán đến giảm cung cấp máu cho tim - atypical angina – đau ngực không điển hình (1): Đau ngực không liên quan đến tim - non-anginal pain – đau ngực không phải do tim: Thường là do co thắt dạ dày, không liên quan đến tim (2) - asymptomatic – không có triệu trứng đau ngực (3): Không có dấu hiệu của bệnh
4	resting_blood_pressure	Huyết áp nghỉ: - Là huyết áp của bệnh nhân, được đo bằng mm Hg khi nhập viện - Giá trị trên 130 – 140 thường là nguyên nhân lo lắng
5	cholesterol	Cholesterol huyết thanh: - Là lượng cholesterol trong huyết thanh, được tính bằng mg/dl - Giá trị trên 200 thường là nguyên nhân lo lắng
6	fasting_blood_sugar	Đường huyết nhanh: - Đường huyết nhanh > 120 ml/dl – Đúng (1)

		- Đường huyết nhanh <= 120 ml/dl – Sai (0) - Giá trị '> 126 m' g/dl là dấu hiệu của bệnh tiểu đường
7	resting_electrocardiogram	Điện tâm đồ nghỉ (Biến thiên của dòng điện do tim phát ra khi nghỉ ngơi): - Bình thường (0) - Biến động song ST-T – Dấu hiệu của nhịp tim không bình thường (1) - Có thể hoặc chắc chắn có phì độ thất trái – tăng thể tích tim trái theo tiêu chí của Estes (2)
8	max_heart_rate_achieved	Nhịp tim tối đa đạt được trong quá trình kiểm tra
9	exercise_induced_angina	Đau thắt ngực do tập thể dục: - yes (1) - no (0)
10	st_depression	ST depression tạo ra bởi tập thể dục so với nghỉ ngơi: - Đo lượng sự giảm độ lệch ST tạo ra bởi tập thể dục so với thời gian nghỉ. - Nhìn vào sự căng trải qua thời gian thể dục, với tim không khỏe sẽ trải qua căng càng nhiều.
11	st_slope	Độ dốc của đoạn ST khi tập luyện mạnh: - upsloping – Tăng lên: Tốt hơn với nhiệp tim khi tập thể dục – Hiếm gặp (0) - flatsloping – Bằng phẳng: Thay đổi tối thiểu – Thường là tim khỏe mạnh (1) - downsloping – Giảm xuống: Dấu hiệu của tim không khỏe mạnh (2)
12	num_major_vessels	Số lượng mạch chính (0-3) được nhuộm màu bằng fluoroscopy: - Số lượng mạch máu chính được nhuộm bằng fluoroscopy, từ 0 – 3 - Mạch máu nhuộm được xem như mạch máu đang di chuyển mà không có cục máu động
13	thalassemia	Chỉ số rối loạn máu di truyền (Kết quả thử Thallium Stress): - Fixed defect – Khuyết tật cố định: Từng có khuyết tật nhưng hiện tại đã ổn định (0)

		- Normal – Bình thường: Không có vấn đề với sự di chuyển của máu (1) - Reversible defect – Khuyết tật có thể đảo ngược: Sự di chuyển của máu không bình thường khi tập thể dục (2)
14	disease	Chuẩn đoán: - yes – Mắc bệnh tim mạch (1) - no – Không mắc bệnh (0)

Bảng 2. 1 Đặc tả dữ liệu

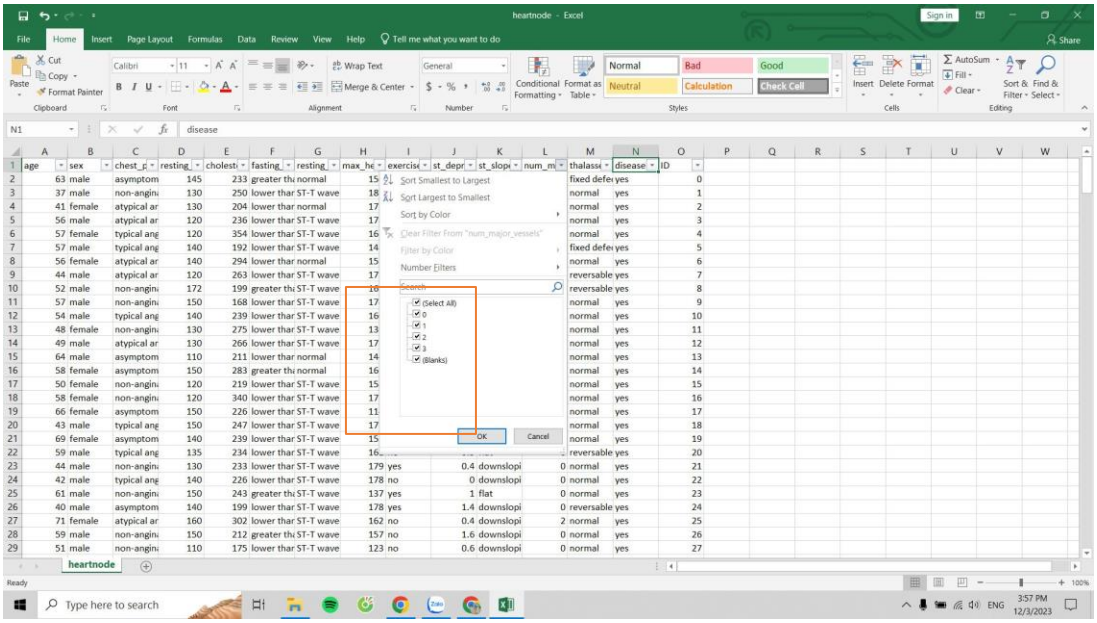
- Dưới đây là giải thích chi tiết hơn về các thuộc tính liên quan đến ST, ST-T, số mạch máu chính được nhuộm bằng fluoroscopy, và kết quả thử thallium stress:
  - ST và ST-T:
    - Trong điện tâm đồ (ECG), ST và ST-T là phần của chu kỳ điện tim được theo dõi để đánh giá hoạt động điện của tim.
    - ST Segment (Đoạn ST): Đoạn này bắt đầu từ kết thúc sóng Q và kéo dài đến đầu sóng T. Sự thay đổi của đoạn ST thường được quan sát để đánh giá sự cung cấp máu và khả năng hoạt động của tim.
    - ST-T Wave: Là sóng T kết hợp với đoạn ST. Sự thay đổi của ST-T thường được sử dụng để chẩn đoán các vấn đề về tim, đặc biệt là trong trường hợp có thể xuất hiện những biến động không bình thường.
  - Số mạch máu chính (0 - 3) được nhuộm màu bằng Fluoroscopy:
    - Đây là một ước lượng về số lượng mạch máu chính (các động mạch lớn cung cấp máu đến tim) được nhuộm bằng fluoroscopy.
    - Fluoroscopy là một phương pháp hình ảnh y khoa sử dụng tia X để theo dõi chuyển động và dòng chảy của chất nhuộm trong các mạch máu.
    - Số mạch máu nhuộm được thường được xem là một chỉ số của sự di chuyển của máu trong tim. Động mạch nhuộm được đánh giá lành mạnh nếu có nhiều mạch máu được nhuộm.

- Kết quả thử Thallim Stress:
  - Là một phương pháp thử nghiệm thực hiện trong điện tâm đồ để đánh giá cách tim hoạt động khi được đặt trong tình trạng căng thẳng.
  - Fixed Defect (Khuyết Tật Cố Định): Kết quả 1 chỉ ra một khuyết tật đã cố định, nghĩa là có một khu vực trong tim mà không có sự di chuyển của máu và vẫn giữ nguyên tình trạng này sau khi tăng cường tập thể dục.
  - Normal (Bình Thường): Kết quả 2 cho thấy không có vấn đề đặc biệt với sự di chuyển của máu trong tim khi tập thể dục.
  - Reversible Defect (Khuyết Tật Có Thể Đảo Ngược): Kết quả 3 chỉ ra sự di chuyển của máu không bình thường khi tập thể dục, và tình trạng này có thể đảo ngược sau khi dừng tập thể dục.

⇨ Những thông tin này cung cấp hiểu biết sâu sắc hơn về sự hoạt động của tim và các vấn đề tiềm ẩn mà có thể được đánh giá thông qua các kết quả điện tâm đồ và thử thallium stress.

2.1.3 Làm sạch dữ liệu

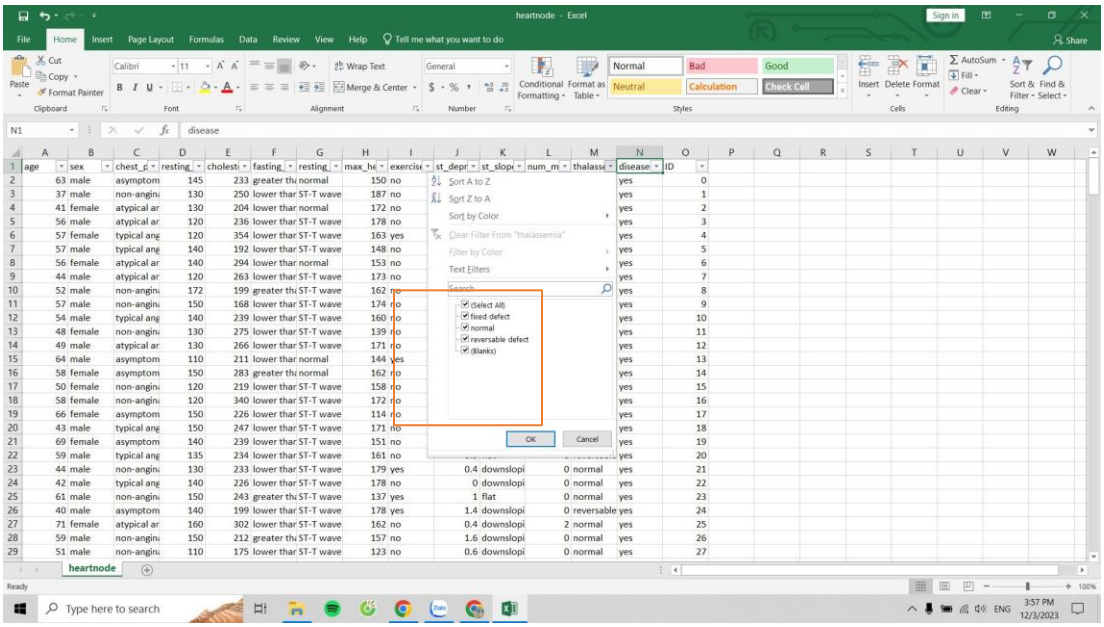
- Những điểm bất thường ở bộ dữ liệu ban đầu:
  - Phát hiện có phần tử rỗng ở cột num_major_vessels và cột thalassemia:



Hình 2. 1 Phát hiện có phần tử rỗng

age	sex	chest_p	resting_	cholest	fasting_	resting_	max_he	exercis	st_depr	st_slop	num_m	thalass	disease	ID
52	male	non-angin	138	223	lower thar	ST-T wave	169	no	0	downsloping		normal	yes	92
58	male	atypical ar	125	220	lower thar	ST-T wave	144	no	0.4	flat		reversable	yes	158
38	male	non-angin	138	175	lower thar	ST-T wave	173	no	0	downsloping		normal	yes	163
38	male	non-angin	138	175	lower thar	ST-T wave	173	no	0	downsloping		normal	yes	164
43	male	typical ang	132	247	greater tha	normal	143	yes	0.1	flat		reversable	no	251

Hình 2. 2 Phát hiện có phần tử rỗng ở cột num_major_vessels



Hình 2. 3 Phát hiện có phần tử rỗng

age	sex	chest_p	resting_	cholest	fasting_	resting_	max_he	exercis	st_depr	st_slop	num_m	thalass	disease	ID
53	female	non-angin	128	216	lower thar	normal	115	no	0	downslopi	0		yes	48
52	male	typical ang	128	204	greater tha	ST-T wave	156	yes	1	flat	0		no	281

Hình 2. 4 Phát hiện có phần tử rỗng ở cột thalassemia

```
data.isnull().sum()
```

```
age                         0
sex                         0
chest_pain_type             0
resting_blood_pressure      0
cholesterol                 0
fasting_blood_sugar         0
resting_electrocardiogram   0
max_heart_rate_achieved     0
exercise_induced_angina     0
st_depression               0
st_slope                    0
num_major_vessels           5
thalassemia                 2
disease                     0
ID                          0
dtype: int64
```

Hình 2. 5 Tổng kết phát hiện số lỗi ở bộ dữ liệu

- Kiểu dữ liệu ở cột num_major_vessels đang là kiểu float trong khi nó chỉ cần để kiểu int.

```
data.dtypes
```

```
age                          int64
sex                         object
chest_pain_type             object
resting_blood_pressure       int64
cholesterol                  int64
fasting_blood_sugar         object
resting_electrocardiogram   object
max_heart_rate_achieved      int64
exercise_induced_angina     object
st_depression              float64
st_slope                    object
num_major_vessels          float64
thalassemia                 object
disease                     object
ID                           int64
dtype: object
```

Hình 2. 6 Lỗi sai kiểu dữ liệu

- Làm sạch dữ liệu:
  - Loại bỏ những phần tử bị null ở 2 cột.

```
data = data.dropna(subset=['num_major_vessels'])
data = data.dropna(subset=['thalassemia'])
print(f'The length of the data now is {len(data)} instead of 303!')
```

```
The length of the data now is 296 instead of 303!
```

Hình 2. 7 Loại bỏ phần tử bị null

```
data.isnull().sum()
```

```
age                         0
sex                         0
chest_pain_type             0
resting_blood_pressure      0
cholesterol                 0
fasting_blood_sugar         0
resting_electrocardiogram   0
max_heart_rate_achieved     0
exercise_induced_angina     0
st_depression               0
st_slope                    0
num_major_vessels           0
thalassemia                 0
disease                     0
dtype: int64
```

Hình 2. 8 Tổng kết không còn xuất hiện phần tử null

- Chuyển kiểu float ở cột num_major_vessels thành kiểu int.

```
[6]  data['num_major_vessels'] = data['num_major_vessels'].astype(int)
```

```
data = data.drop('ID', axis=1)
data
```

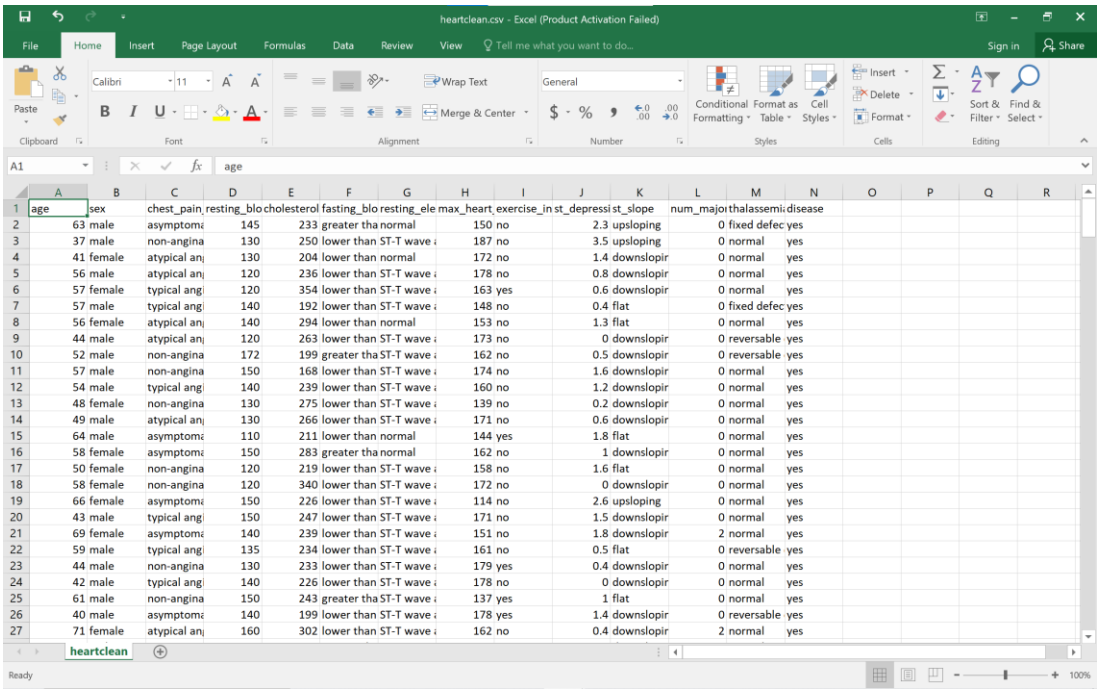
Hình 2. 9 Code chuyển kiểu float thành int

```
data.dtypes
```

```
age                        int64
sex                       object
chest_pain_type           object
resting_blood_pressure     int64
cholesterol                int64
fasting_blood_sugar       object
resting_electrocardiogram object
max_heart_rate_achieved    int64
exercise_induced_angina   object
st_depression            float64
st_slope                  object
num_major_vessels          int64
thalassemia               object
disease                   object
dtype: object
```

Hình 2. 10 Kiểu dữ liệu của num major vessels

⇨ Sau khi làm sạch dữ liệu thì ta đươc bộ dataset như sau:



	age	sex	chest_pain	resting_blo	cholesterol	fasting_blo	resting_ele	max_heart	exercise_in	st_depressi	st_slope	num_majo	thalassemi	disease
2	63	male	asymptoma	145	233	greater tha	normal	150	no	2.3	upsloping	0	fixed defec	yes
3	37	male	non-angina	130	250	lower than	ST-T wave a	187	no	3.5	upsloping	0	normal	yes
4	41	female	atypical an	130	204	lower than	normal	172	no	1.4	downslopir	0	normal	yes
5	56	male	atypical an	120	236	lower than	ST-T wave a	178	no	0.8	downslopir	0	normal	yes
6	57	female	typical angi	120	354	lower than	ST-T wave a	163	yes	0.6	downslopir	0	normal	yes
7	57	male	typical angi	140	192	lower than	ST-T wave a	148	no	0.4	flat	0	fixed defec	yes
8	56	female	atypical an	140	294	lower than	normal	153	no	1.3	flat	0	normal	yes
9	44	male	atypical an	120	263	lower than	ST-T wave a	173	no	0	downslopir	0	reversable	yes
10	52	male	non-angina	172	199	greater tha	ST-T wave a	162	no	0.5	downslopir	0	reversable	yes
11	57	male	non-angina	150	168	lower than	ST-T wave a	174	no	1.6	downslopir	0	normal	yes
12	54	male	typical angi	140	239	lower than	ST-T wave a	160	no	1.2	downslopir	0	normal	yes
13	48	female	non-angina	130	275	lower than	ST-T wave a	139	no	0.2	downslopir	0	normal	yes
14	49	male	atypical an	130	266	lower than	ST-T wave a	171	no	0.6	downslopir	0	normal	yes
15	64	male	asymptoma	110	211	lower than	normal	144	yes	1.8	flat	0	normal	yes
16	58	female	asymptoma	150	283	greater tha	normal	162	no	1	downslopir	0	normal	yes
17	50	female	non-angina	120	219	lower than	ST-T wave a	158	no	1.6	flat	0	normal	yes
18	58	female	non-angina	120	340	lower than	ST-T wave a	172	no	0	downslopir	0	normal	yes
19	66	female	asymptoma	150	226	lower than	ST-T wave a	114	no	2.6	upsloping	0	normal	yes
20	43	male	typical angi	150	247	lower than	ST-T wave a	171	no	1.5	downslopir	0	normal	yes
21	69	female	asymptoma	140	239	lower than	ST-T wave a	151	no	1.8	downslopir	2	normal	yes
22	59	male	typical angi	135	234	lower than	ST-T wave a	161	no	0.5	flat	0	reversable	yes
23	44	male	non-angina	130	233	lower than	ST-T wave a	179	yes	0.4	downslopir	0	normal	yes
24	42	male	typical angi	140	226	lower than	ST-T wave a	178	no	0	downslopir	0	normal	yes
25	61	male	non-angina	150	243	greater tha	ST-T wave a	137	yes	1	flat	0	normal	yes
26	40	male	asymptoma	140	199	lower than	ST-T wave a	178	yes	1.4	downslopir	0	reversable	yes
27	71	female	atypical an	160	302	lower than	ST-T wave a	162	no	0.4	downslopir	2	normal	yes

Hình 2. 11 Dữ liệu sau khi đã làm sạch

2.2 Trực quan hóa

- Encoder label: Để thực hiện trực quan hóa thì tiến hành thay đổi các phần tử từ kiểu chuỗi sang kiểu int.

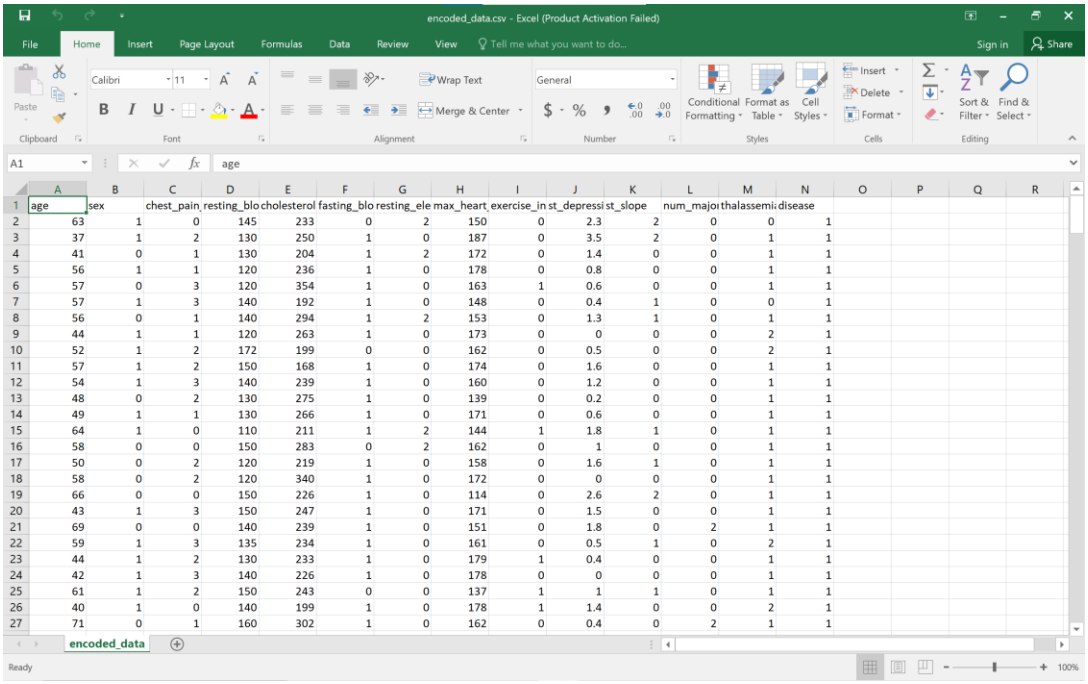
```
le=LabelEncoder()
lst=['sex','chest_pain_type','fasting_blood_sugar','resting_electrocardiogram','exercise_induced_angina','st_slope','thalassemia','disease']
label_mapping = {}
for i in lst:
  data[i]=le.fit_transform(data[i])
  label_mapping[i] = dict(zip(le.classes_, range(len(le.classes_))))

for feature, mapping in label_mapping.items():
    print(f"{feature} mapping:")
    for label_str, number in mapping.items():
        print(f"  {label_str}: {number}")
```

Hình 2. 12 Chuyển toàn bộ dữ liệu khác sang int

```
sex mapping:
  female: 0
  male: 1
chest_pain_type mapping:
  asymptomatic: 0
  atypical angina: 1
  non-anginal pain: 2
  typical angina: 3
fasting_blood_sugar mapping:
  greater than 120mg/ml: 0
  lower than 120mg/ml: 1
resting_electrocardiogram mapping:
  ST-T wave abnormality: 0
  left ventricular hypertrophy: 1
  normal: 2
exercise_induced_angina mapping:
  no: 0
  yes: 1
st_slope mapping:
  downsloping: 0
  flat: 1
  upsloping: 2
thalassemia mapping:
  fixed defect: 0
  normal: 1
  reversable defect: 2
disease mapping:
  no: 0
  yes: 1
```

Hình 2. 13 Quy định giá trị cho từng thuộc tính ở kiểu int

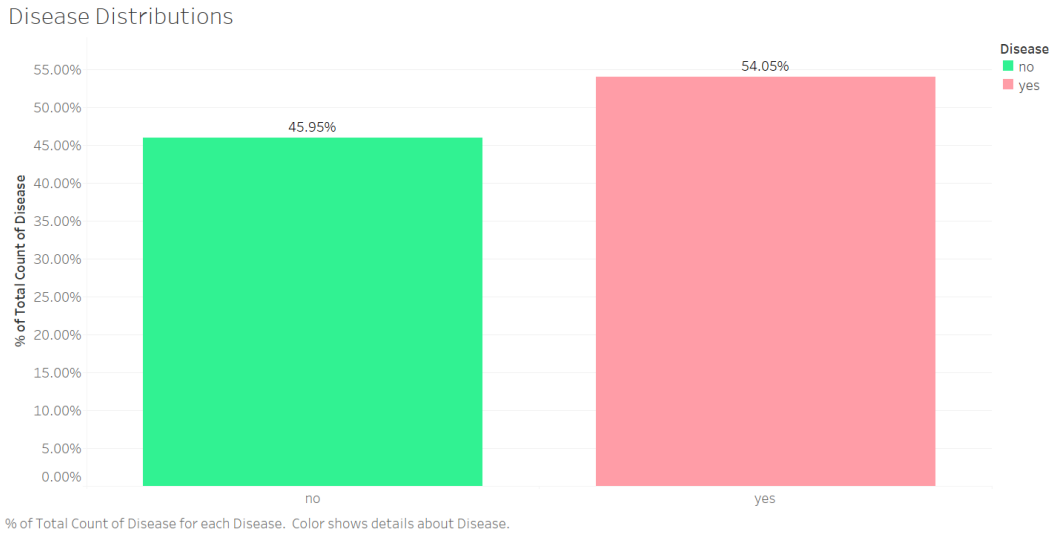
- Sau cùng thì ta được bộ dữ liệu như sau:



age	sex	chest_pain	resting_blo	cholesterol	fasting_blo	resting_ele	max_heart	exercise_in	st_depressi	st_slope	num_majo	thalassemi	disease
63	1	0	145	233	0	2	150	0	2.3	2	0	0	1
37	1	2	130	250	1	0	187	0	3.5	2	0	1	1
41	0	1	130	204	1	2	172	0	1.4	0	0	1	1
56	1	1	120	236	1	0	178	0	0.8	0	0	1	1
57	0	3	120	354	1	0	163	1	0.6	0	0	1	1
57	1	3	140	192	1	0	148	0	0.4	1	0	0	1
56	0	1	140	294	1	2	153	0	1.3	1	0	1	1
44	1	1	120	263	1	0	173	0	0	0	0	2	1
52	1	2	172	199	0	0	162	0	0.5	0	0	2	1
57	1	2	150	168	1	0	174	0	1.6	0	0	1	1
54	1	3	140	239	1	0	160	0	1.2	0	0	1	1
48	0	2	130	275	1	0	139	0	0.2	0	0	1	1
49	1	1	130	266	1	0	171	0	0.6	0	0	1	1
64	1	0	110	211	1	2	144	1	1.8	1	0	1	1
58	0	0	150	283	0	2	162	0	1	0	0	1	1
50	0	2	120	219	1	0	158	0	1.6	1	0	1	1
58	0	2	120	340	1	0	172	0	0	0	0	1	1
66	0	0	150	226	1	0	114	0	2.6	2	0	1	1
43	1	3	150	247	1	0	171	0	1.5	0	0	1	1
69	0	0	140	239	1	0	151	0	1.8	0	2	1	1
59	1	3	135	234	1	0	161	0	0.5	1	0	2	1
44	1	2	130	233	1	0	179	1	0.4	0	0	1	1
42	1	3	140	226	1	0	178	0	0	0	0	1	1
61	1	2	150	243	0	0	137	1	1	1	0	1	1
40	1	0	140	199	1	0	178	1	1.4	0	0	2	1
71	0	1	160	302	1	0	162	0	0.4	0	2	1	1

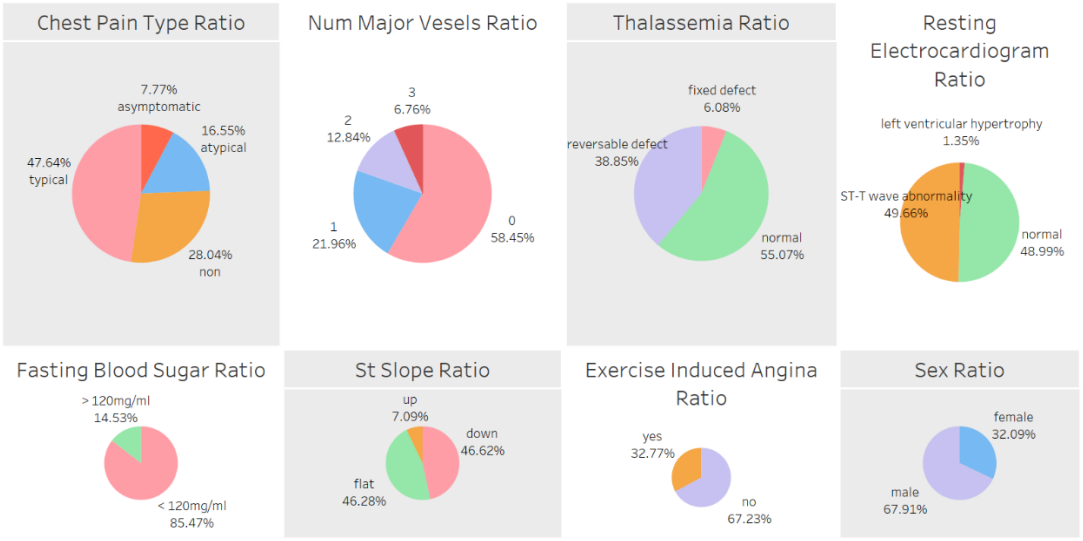
Hình 2. 14 Bộ dữ liệu hoàn chỉnh

- Disease Distributions



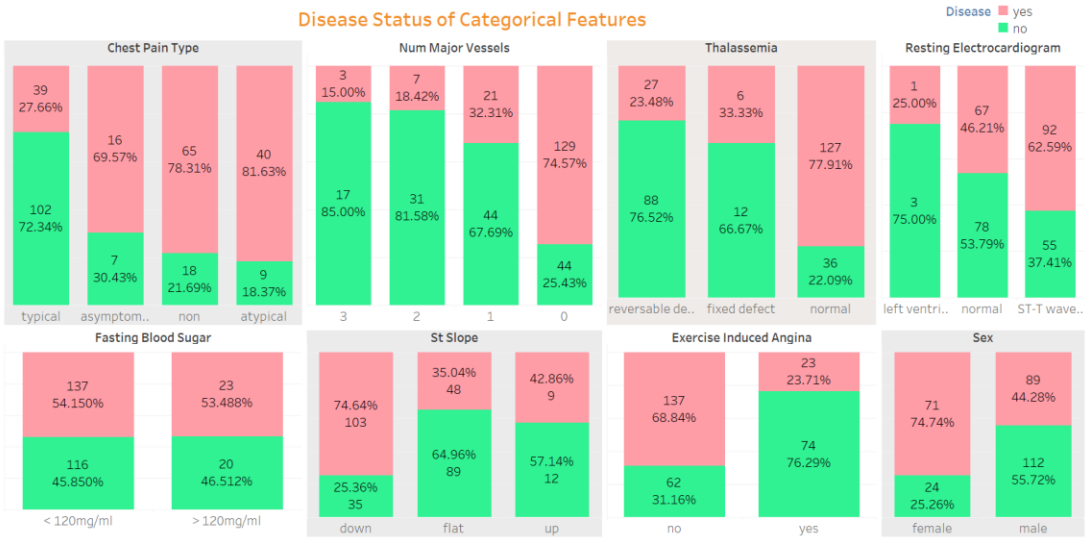
Hình 2. 15 Disease Distributions

- Categorical Feature:
  - Disease Ratio of Categorical Feature



Hình 2. 16 Disease Ratio of Categorical Feature

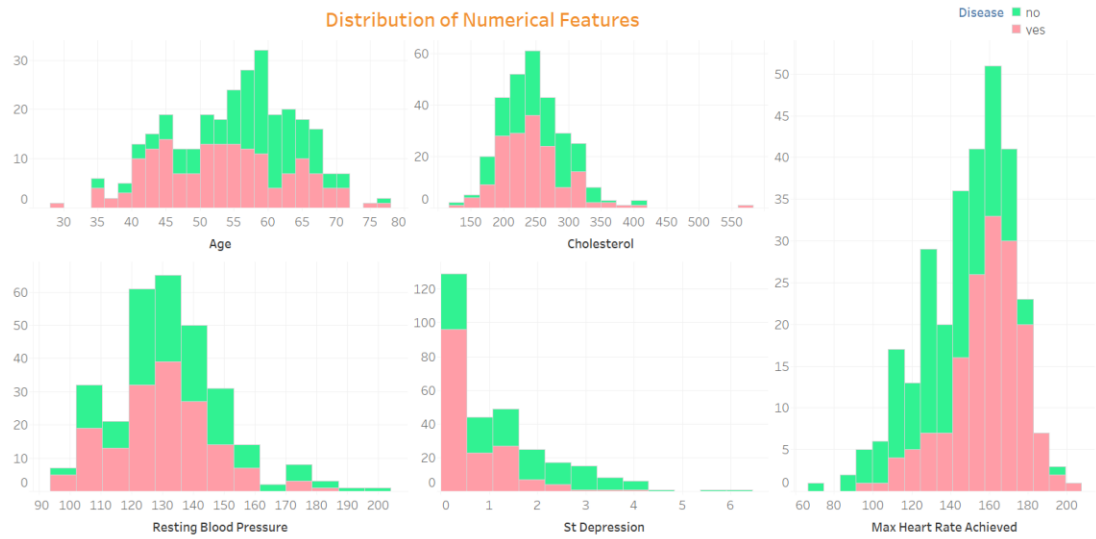
  - Disease Status of Categorical Features



Hình 2. 17 Disease Status of Categorical Features

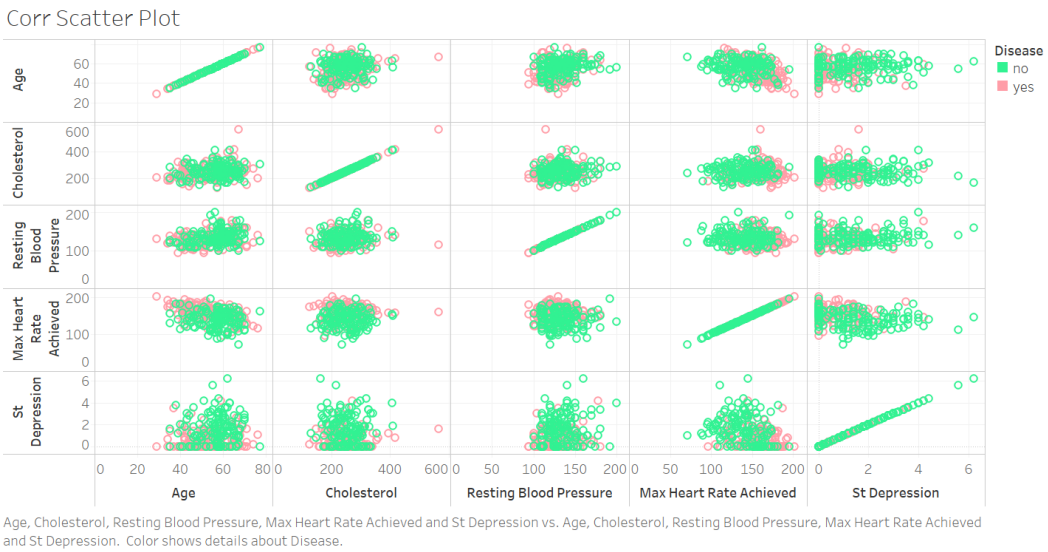
- Tổng quan:
  - Chest Pain: Hơn 75% bệnh nhân bị đau ngực điển hình (typical angina) và đau ngực không phải do tim (non-angina pain) nên họ có nguy cơ mắc bênh tim.
  - Fasting Blood Sugar: Bệnh nhân có lượng đường trong máu lúc đói thấp hơn (dưới 120 mg/ml) chiếm đa số trong tập dữ liệu của chúng tôi bao gồm ~ 85% mẫu. Lượng đường trong máu khi nghỉ ngơi thấp hơn có xu hướng làm tăng nguy cơ mắc bệnh tim (~54%).
  - ST Slope: Hầu hết bệnh nhân đều có ST-Slope trong xét nghiệm REC của họ là dốc xuống (downsloping) hoặc bằng phẳng (flat). Độ dốc ST dốc xuống (downsloping) là dấu hiệu rõ ràng cho thấy bệnh nhân có thể mắc bệnh tim.
  - Thalassemia: Hầu hết bệnh nhân có khiếm khuyết bình thường (normal) hoặc có thể đảo ngược (reversible defect). Những bệnh nhân có khuyết tật bệnh thalassemia (có thể reversible + fixed) ít có khả năng mắc bệnh tim hơn. Trong khi đó, những người khiếm khuyết bình thường (normal) có nhiều khả năng mắc bệnh tim hơn.
  - Exercise Include Angina: Hai phần ba số bệnh nhân không có biểu hiện đau thắt ngực do tập thể dục. 76% bệnh nhân bị đau thắt ngực do tập thể dục không có bệnh tim. Trong khi đó ~ 69% bệnh nhân không bị đau thắt ngực do tập thể dục được chẩn đoán có nguy cơ mắc bệnh tim.
  - Resting Electrocardiogram: Bệnh nhân bị phì độ thất trái là ít nhất (~1,4%). Phần còn lại gần như là sự phân chia 50 - 50 giữa bệnh nhân có ST-T bất thường và những người có xét nghiệm REC bình thường. Phần lớn những bệnh nhân thực hiện loại xét nghiệm này đều có nguy cơ mắc bệnh tim.
  - Sex: Đa phần bệnh nhân trong dữ liệu là nam gới. Tuy nhiên, nữ giới lại có nguy cơ mắc bệnh tim cao hơn.

- Numerical Features:
  - Distributions of Numerical Features



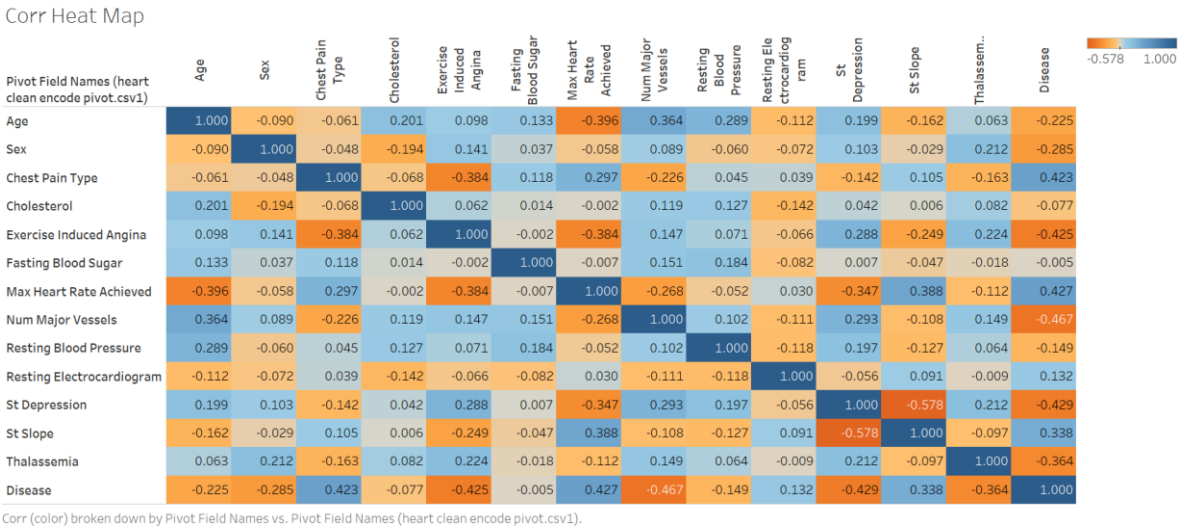
Hình 2. 18 Distributions of Numerical Features

  - Corr Scatter Plot



Hình 2. 19 Corr Scatter Plot

- Corr Heat Map



Pivot Field Names (heart clean encode pivot.csv1)	Age	Sex	Chest Pain Type	Cholesterol	Exercise Induced Angina	Fasting Blood Sugar	Max Heart Rate Achieved	Num Major Vessels	Resting Blood Pressure	Resting Electrocardiogram	St Depression	St Slope	Thalassem...	Disease
Age	1.000	-0.090	-0.061	0.201	0.098	0.133	-0.396	0.364	0.289	-0.112	0.199	-0.162	0.063	-0.225
Sex	-0.090	1.000	-0.048	-0.194	0.141	0.037	-0.058	0.089	-0.060	-0.072	0.103	-0.029	0.212	-0.285
Chest Pain Type	-0.061	-0.048	1.000	-0.068	-0.384	0.118	0.297	-0.226	0.045	0.039	-0.142	0.105	-0.163	0.423
Cholesterol	0.201	-0.194	-0.068	1.000	0.062	0.014	-0.002	0.119	0.127	-0.142	0.042	0.006	0.082	-0.077
Exercise Induced Angina	0.098	0.141	-0.384	0.062	1.000	-0.002	-0.384	0.147	0.071	-0.066	0.288	-0.249	0.224	-0.425
Fasting Blood Sugar	0.133	0.037	0.118	0.014	-0.002	1.000	-0.007	0.151	0.184	-0.082	0.007	-0.047	-0.018	-0.005
Max Heart Rate Achieved	-0.396	-0.058	0.297	-0.002	-0.384	-0.007	1.000	-0.268	-0.052	0.030	-0.347	0.388	-0.112	0.427
Num Major Vessels	0.364	0.089	-0.226	0.119	0.147	0.151	-0.268	1.000	0.102	-0.111	0.293	-0.108	0.149	-0.467
Resting Blood Pressure	0.289	-0.060	0.045	0.127	0.071	0.184	-0.052	0.102	1.000	-0.118	0.197	-0.127	0.064	-0.149
Resting Electrocardiogram	-0.112	-0.072	0.039	-0.142	-0.066	-0.082	0.030	-0.111	-0.118	1.000	-0.056	0.091	-0.009	0.132
St Depression	0.199	0.103	-0.142	0.042	0.288	0.007	-0.347	0.293	0.197	-0.056	1.000	-0.578	0.212	-0.429
St Slope	-0.162	-0.029	0.105	0.006	-0.249	-0.047	0.388	-0.108	-0.127	0.091	-0.578	1.000	-0.097	0.338
Thalassemia	0.063	0.212	-0.163	0.082	0.224	-0.018	-0.112	0.149	0.064	-0.009	0.212	-0.097	1.000	-0.364
Disease	-0.225	-0.285	0.423	-0.077	-0.425	-0.005	0.427	-0.467	-0.149	0.132	-0.429	0.338	-0.364	1.000

Hình 2. 20 Corr Heat Map

- Tổng quan:
  - Age: Độ tuổi trung bình ở trong tập dữ liệu sẽ là 54.5 tuổi, người lớn nhất là 77 tuổi và trẻ nhất là 29 tuổi.
  - Cholesterol: Mức cholesterol trung bình là 247.15, mức cao nhất là 564 và thấp nhất là 126 (Thường mức cholesterol khỏe mạnh là < 200mg/dl và ngược lại thì sẽ có liên quan đến bệnh tim)
  - Resting blood pressure: Mức huyết áp nghỉ trung bình là 132, mức cao nhất là 200 và thập nhất là 94.
  - Max heart rate achieved: Nhịp tim tối đa trung bình đạt 149.5 bpm, mức cao nhất đạt được là 200 bpm và thấp nhất đạt được là 71 bpm.
  - ST_depression: Giá trị trung bình của ST_depression là 1.06, mức tối đa đạt là 6.2 và tối thiểu là 0.

2.3 Ứng dụng Machine Learning

2.3.1 Đặc tả mô hình Machine Learning

- Bộ dữ liệu về bệnh tim là một bộ dữ liệu được sử dụng để giải quyết bài toán phân loại.
- Mục tiêu chính của bài toán này là dự đoán xem một người có rủi ro cao mắc bệnh tim hay không dựa trên các đặc trưng y tế và lâm sàng của họ. Vì nó là bài toán phân loại nên ta sẽ chọn các mô hình ML sau để triển khai:
  - LogisticRegression:
    - Lý do chọn: Logistic Regression là một lựa chọn phổ biến cho bài toán phân loại nhị phân. Nó dễ hiểu, có thể giải thích được và thường hiệu quả trong các trường hợp khi có mối quan hệ tuyến tính giữa đặc trưng và kết quả.
    - Ưu điểm:
      - Dễ hiểu và dễ triển khai: Logistic Regression là một mô hình đơn giản và dễ hiểu. Nó không yêu cầu nhiều tài nguyên tính toán so với các mô hình phức tạp hơn như Neural Networks.
      - Tính diễn giải cao: Kết quả của mô hình Logistic Regression có thể dễ dàng diễn giải. Bạn có thể hiểu được tác động của từng đặc trưng đến dự đoán xác suất.
      - Ít yêu cầu về dữ liệu: Không cần nhiều điều kiện cho dữ liệu. Logistic Regression hoạt động tốt trong những tình huống dữ liệu ít và các đặc trưng không phức tạp.
    - Nhược điểm:
      - Giả định tuyến tính: Logistic Regression giả định mối quan hệ giữa biến độc lập và xác suất là tuyến tính. Điều này có thể là hạn chế nếu mối quan hệ thực sự không phải là tuyến tính.
      - Dễ bị ảnh hưởng bởi nhiễu: Nếu dữ liệu chứa nhiều nhiễu hoặc outliers, mô hình Logistic Regression có thể bị ảnh hưởng và không đưa ra dự đoán chính xác.

- Không xử lý được các tương tác phức tạp giữa đặc trứng: Logis Regression không thể mô hình hóa tương tác phức tạp giữa các đặc trưng.

- KNeighborsClassifier:
  - Lý do chọn: KNN là một mô hình không parametric đơn giản và có thể hoạt động tốt với dữ liệu có cấu trúc đơn giản. việc sử dụng KNN có thể đem lại kết quả tốt nếu dữ liệu không quá lớn và không có quá nhiều nhiễu.
  - Ưu điểm:
    - Dễ hiểu và triển khai: KNN là một trong những mô hình đơn giản và dễ hiểu nhất trong Machine Learning. Nó không yêu cầu giả định về phân phối của dữ liệu và có thể được triển khai dễ dàng.
    - Khả năng áp dụng vào dữ liệu phức tạp: KNN hoạt động tốt trên dữ liệu có cấu trúc phức tạp và có thể xử lý các đặc trưng không tuyến tính.
    - Khả năng dự đoán linh hoạt: KNN không huấn luyện một mô hình cụ thể, mà chỉ lưu trữ dữ liệu huấn luyện. Điều này có nghĩa là khi có dữ liệu mới, việc dự đoán có thể được thực hiện một cách linh hoạt và không cần phải huấn luyện lại mô hình.
  - Nhược điểm:
    - Yêu cầu nhiều tài nguyên khi dữ liệu lớn: KNN cần lưu trữ toàn bộ tập dữ liệu huấn luyện trong bộ nhớ. Điều này có thể trở nên rất tốn kém về tài nguyên nếu tập dữ liệu lớn.
    - Nhảy cảm với dữ liệu nhiễu và không đồng nhất: KNN có thể không hiệu quả khi dữ liệu chứa nhiễu hoặc không đồng nhất. Các điểm dữ liệu "gần" nhau thực tế có thể không phản ánh chính xác về tính chất của dữ liệu.
    - Yêu cầu quá trình chuẩn hóa dữ liệu: Do KNN dựa vào khoảng cách giữa các điểm dữ liệu, việc chuẩn hóa dữ liệu (đảm bảo các đặc trưng có cùng phạm vi giá trị) có thể cần thiết để mô hình hoạt động tốt.

- DecisionTreeClassifier:
  - Lý do chọn: Decision Tree có thể xử lý tốt cả với dữ liệu có mối quan hệ tuyến tính và không tuyến tính. Ngoài ra, chúng cũng có khả năng xử lý tương tác giữa các đặc trưng.
  - Ưu điểm:
    - Dễ hiểu và diễn giải: Decision Tree rất dễ diễn giải, có thể được trực quan hóa. Bạn có thể dễ dàng giải thích cho người không chuyên về machine learning về cách mô hình ra quyết định dựa trên các thuộc tính.
    - Xử lý dữ liệu không chuẩn: Mô hình này có khả năng xử lý dữ liệu không đồng nhất hoặc thiếu sót, không cần quá nhiều tiền xử lý dữ liệu trước khi huấn luyện.
    - Phát hiện mẫu: Decision Tree có khả năng phát hiện mẫu tự nhiên trong dữ liệu. Nó có thể dễ dàng xác định các mẫu tương đối phức tạp trong tập dữ liệu.
    - Áp dụng cho cả dữ liệu dạng số và phân loại: Mô hình này có thể xử lý cả dữ liệu dạng số và dạng phân loại, giúp bạn tính toán được các biến số liên tục cũng như các biến phân loại.
  - Nhược điểm:
    - Dễ bị overfitting: Decision Tree có thể dễ bị overfitting (quá khớp) trên tập dữ liệu huấn luyện, đặc biệt khi không có các biện pháp điều chuẩn như pruning hoặc sử dụng các biến thay thế.
    - Không ổn định với sự biến đổi nhỏ: Các thay đổi nhỏ trong dữ liệu đầu vào có thể dẫn đến sự thay đổi lớn trong cấu trúc của cây quyết định, làm cho mô hình không ổn định.
    - Có thể tạo ra cây quyết định phức tạp: Nếu không được kiểm soát, cây quyết định có thể trở nên rất phức tạp và khó diễn giải, dẫn đến hiện tượng quá phù hợp với dữ liệu huấn luyện.

- SVM
  - Lý do chọn: SVM là một mô hình mạnh mẽ có thể tạo ra các biên quyết định phức tạp. Nó có thể hoạt động tốt với dữ liệu có cấu trúc phức tạp và có khả năng xử lý tốt với các đặc trưng không tuyến tính.
  - Ưu điểm:
    - Hiệu suất tốt trong không gian chiều cao: SVM hoạt động tốt khi số lượng đặc trưng lớn hơn số lượng mẫu, đặc biệt trong các không gian chiều cao. Trong trường hợp dự đoán bệnh tim, thông thường có nhiều đặc trưng y tế.
    - Hiệu suất tốt khi có biên phân chia rõ ràng: SVM tốt khi có một ranh giới phân chia rõ ràng giữa các lớp dữ liệu. Nếu có tính chất phân tách tốt giữa những người có rủi ro cao và thấp về bệnh tim, SVM có thể hoạt động hiệu quả.
    - Khả năng xử lý dữ liệu lớn: SVM có thể xử lý tốt dữ liệu lớn mà không gây overfitting.
    - Hỗ trợ việc sử dụng các hàm kernel: SVM có khả năng ánh xạ dữ liệu vào các không gian chiều cao hơn thông qua việc sử dụng các hàm kernel. Điều này giúp tăng khả năng phân loại khi dữ liệu không phân tách tuyến tính.
  - Nhược điểm:
    - Đòi hỏi lựa chọn kernel phù hợp: Việc chọn kernel phù hợp có thể ảnh hưởng đến hiệu suất của SVM. Đôi khi, việc lựa chọn kernel không phù hợp có thể dẫn đến hiện tượng overfitting hoặc underfitting.
    - Khó mở rộng cho dữ liệu lớn: Trong một số trường hợp, SVM có thể tốn nhiều tài nguyên tính toán và bộ nhớ, đặc biệt khi xử lý dữ liệu lớn.
    - Khả năng xử lý nhiễu: SVM có thể nhạy cảm với nhiễu trong dữ liệu, có thể dẫn đến việc không tốt trong việc dự đoán.
    - Đòi hỏi tối ưu tham số: Để đạt hiệu suất tốt nhất, SVM yêu cầu tinh chỉnh tham số cẩn thận, điều này có thể là một quá trình tốn thời gian.

- NaiveBayes
  - Lý do chọn: Naive Bayes là một mô hình phân loại đơn giản và nhanh chóng. Nó thường là một lựa chọn tốt khi dữ liệu có nhiều đặc trưng và các đặc trưng này độc lập với nhau (giả định "ngây thơ" của Naive Bayes).
  - Ưu điểm:
    - Đơn giản và nhanh chóng: Naive Bayes (NB) là một mô hình đơn giản, dễ hiểu và dễ triển khai. Thuật toán này có thể được huấn luyện nhanh chóng trên các tập dữ liệu lớn.
    - Xử lý tốt với dữ liệu lớn: Dù đơn giản, NB vẫn có thể hoạt động tốt trên các tập dữ liệu lớn mà không gặp phải vấn đề về hiệu suất.
    - Hiệu suất tốt với các đặc trưng độc lập: Mô hình này hoạt động tốt khi các đặc trưng độc lập với nhau, điều này cũng phù hợp với một số trường hợp trong y học.
    - Độ chính xác cao ban đầu: Trong một số trường hợp, NB có thể đạt được độ chính xác tốt ngay cả khi giả định về sự độc lập không hoàn toàn chính xác.
  - Nhược điểm:
    - Giả định về độc lập: Giả định rằng các đặc trưng là độc lập với nhau có thể không phù hợp với thực tế trong nhiều trường hợp y tế, vì nhiều biến có thể ảnh hưởng lẫn nhau.
    - Đòi hỏi dữ liệu lớn cho độ chính xác cao: Mặc dù NB hoạt động tốt với dữ liệu lớn, để đạt được độ chính xác cao, nó vẫn cần một lượng dữ liệu đủ lớn và đủ đa dạng.
    - Không xử lý được mối quan hệ phức tạp: Mô hình Naive Bayes không thể hiện mối quan hệ phức tạp giữa các đặc trưng, điều này có thể làm giảm khả năng dự đoán chính xác trong một số trường hợp.

2.3.2 Cách triển khai mô hình Machine Learning

- Khai báo các hàm cần thiết.

```
[ ]  from sklearn.linear_model import LogisticRegression
     from sklearn.neighbors import KNeighborsClassifier
     from sklearn.tree import DecisionTreeClassifier
     from sklearn.svm import SVC
     from sklearn.naive_bayes import GaussianNB

     from sklearn.model_selection import train_test_split, cross_val_score
     from sklearn.model_selection import RandomizedSearchCV, GridSearchCV
     from sklearn.metrics import confusion_matrix, classification_report
     from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score
```

Hình 2. 21 Khai báo hàm sử dụng

- Tạo bộ dữ liệu train, test và bộ dữ liệu dùng để dự đoán.

```
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5, random_state=42)

     y = df['target']
[ ]  X = df.drop('target', axis=1)
```

Hình 2. 22 Tạo dữ liệu train và test

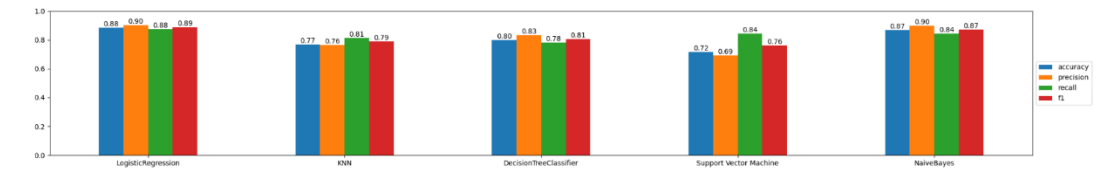
- Tính các chỉ số accuracy, precision, recall, f1-score cho các mô hình.

```
[ ]  models = {
         'LogisticRegression': LogisticRegression(max_iter=10000),
         'KNN': KNeighborsClassifier(),
         'DecisionTreeClassifier': DecisionTreeClassifier(),
         'Support Vector Machine': SVC(),
         'NaiveBayes': GaussianNB()
     }

     def fit_and_score(models, X_train, X_test, y_train, y_test):
         np.random.seed(42)

         model_scores = {}

         for model_name, model in models.items():
             model.fit(X_train, y_train)
             y_pred = model.predict(X_test)

             accuracy = accuracy_score(y_test, y_pred)
             precision = precision_score(y_test, y_pred)
             recall = recall_score(y_test, y_pred)
             f1 = f1_score(y_test, y_pred)

             model_scores[model_name] = {
                 'accuracy': accuracy,
                 'precision': precision,
                 'recall': recall,
                 'f1': f1,
             }

         return model_scores
```

Hình 2. 23 Code tính các chỉ số accuracy, precision, recall, f1-score cho các mô hình

- Vẽ biểu đồ hiển thị các chỉ số trên.

```
[ ] import matplotlib.pyplot as plt

    model_compare = pd.DataFrame(model_scores)

    plt.rcParams['figure.figsize'] = (20.5, 3.5)

    ax = model_compare.T.plot(kind='bar', rot=0)

    for p in ax.patches:
        ax.annotate(f'{p.get_height():.2f}', (p.get_x() + p.get_width() / 2., p.get_height()),
                    ha='center', va='center', xytext=(0, 5), textcoords='offset points')

    ax.set_ylim(0, 1)

    plt.tight_layout()

    plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))

    plt.show()
```



Hình 2. 24 Biểu đồ hiển thị

⇨ Nhìn vào biểu đồ cho thấy model Logistic Regression là mô hình chạy hiệu quả nhất trên bộ dữ liệu đang làm.

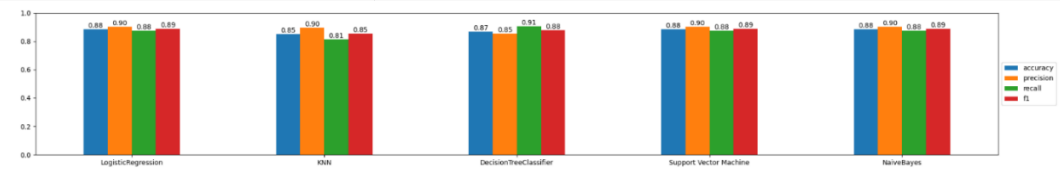
- Sau đó ta StandardScaler và SDA bộ dữ liệu để model SVM và KNN chạy tốt và hiệu quả hơn.

```
[ ] from sklearn.preprocessing import StandardScaler
    sc = StandardScaler()
    X_train = sc.fit_transform(X_train)
    X_test = sc.transform(X_test)
```

```
[ ] print("Shape of X before Dimensionlity Reduction: ", X_train.shape)

    from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
    lda = LDA()
    X_train = lda.fit_transform(X_train, y_train)
    X_test = lda.transform(X_test)

    print("Shape of X after Dimensionlity Reduction: ", X_train.shape)

    Shape of X before Dimensionlity Reduction:  (236, 13)
    Shape of X after Dimensionlity Reduction:  (236, 1)
```

Hình 2. 25 StandardScaler và SDA bộ dữ liệu để model SVM và KNN

- Tiếp đến, ta tính lại các chỉ số như bước ở trên và đo lại các chỉ số, hiển thị các chỉ số lên biểu đồ.



Hình 2. 26 Biểu đồ hiển thị sau khi cải thiện

- Kết quả cho thấy, sau khi SDA và StandardScaler thì 2 mô hình KNN và SVM đã hoạt động tối ưu và hiệu quả hơn trước.
- Vì mô hình Logistic Regression là mô hình chạy hiệu quả nhất nên sẽ sử dụng mô hình này để mô tả cách triển khai của nó:
  - Sử dụng GridSearchCV để tìm ra Hyperparameters tốt nhất.

```
def train_and_evaluate_logistic_regression(X_train, y_train, X_test, y_test):
    # Logistic Regression model
    lr_model = LogisticRegression()
    lr_model.fit(X_train, y_train)

    # Parameters to search
    param_grid = [{'penalty': ['l1', 'l2'], 'C': np.logspace(-4, 4, 20), 'solver': ['liblinear']},
         {'penalty': ['l2', None], 'C': np.logspace(-4, 4, 20), 'solver': ['lbfgs']}]

    # Grid Search
    grid_search_lr = GridSearchCV(estimator=lr_model, param_grid=param_grid, scoring='accuracy', cv=10, n_jobs = -1)
    grid_search_lr.fit(X_train, y_train)

    # Best parameters and model
    best_params_lr = grid_search_lr.best_params_
    best_lr_model = grid_search_lr.best_estimator_

    # Predict and evaluate the best model
    y_pred_lr = best_lr_model.predict(X_test)
    accuracy_lr = accuracy_score(y_test, y_pred_lr)

    return best_params_lr, accuracy_lr

# Example usage:

# Assuming X_train, y_train, X_test, y_test are defined earlier
best_params_lr, accuracy_lr = train_and_evaluate_logistic_regression(X_train, y_train, X_test, y_test)

# Display results
print('Best Hyperparameters for Logistic Regression:')
print(best_params_lr)
print(f'Accuracy for Logistic Regression with Best Hyperparameters: {accuracy_lr:.2f}')
```

```
Best Hyperparameters for Logistic Regression:
{'C': 0.0006951927961775605, 'penalty': 'l2', 'solver': 'liblinear'}
```

Hình 2. 27 Sử dụng GridSearchCV để tìm ra Hyperparameters tốt nhất.

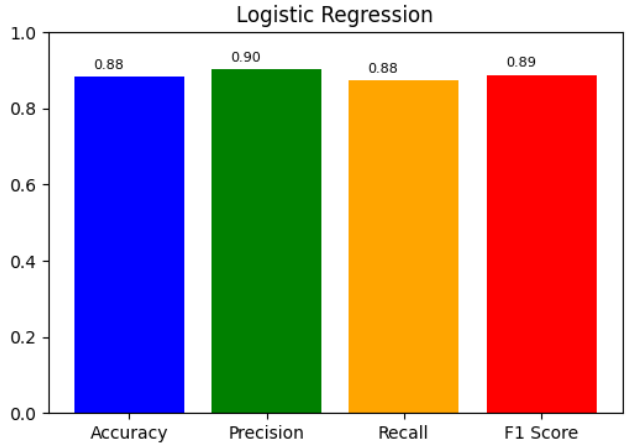
  - Dùng Hyperparameters tốt nhất để tính các chỉ số trong model.

```
def evaluate_classifier_best(X, y, C, penalty, solver):
    # Create a new classifier with current best parameters
    lr_classifier = LogisticRegression(C=C, penalty=penalty, solver=solver)
    lr_classifier.fit(X_train, y_train)

    y_pred_lr = lr_classifier.predict(X_test)

    accuracy_lr = accuracy_score(y_test, y_pred_lr)
    precision_lr = precision_score(y_test, y_pred_lr)
    recall_lr = recall_score(y_test, y_pred_lr)
    f1_lr = f1_score(y_test, y_pred_lr)

    # Plot the results with small-sized bars and values displayed on top of each bar
    labels = ['Accuracy', 'Precision', 'Recall', 'F1 Score']
    values = [accuracy_lr, precision_lr, recall_lr, f1_lr]

    fig, ax = plt.subplots(figsize=(6, 4))  # Adjust the figure size
    bars = ax.bar(labels, values, color=['blue', 'green', 'orange', 'red'])
    plt.ylim([0, 1])  # Set y-axis range between 0 and 1 for better visualization
    plt.title('Logistic Regression')

    # Display values on top of each bar
    for bar, value in zip(bars, values):
        plt.text(bar.get_x() + bar.get_width() / 2 - 0.15, bar.get_height() + 0.02, f'{value:.2f}', ha='center', fontsize=8)

    plt.show()
```

```
evaluate_classifier_best(X, y, C=0.0006951927961775605, penalty='l2', solver='liblinear')
```

Hình 2. 28 Dùng Hyperparameters tốt nhất để tính các chỉ số trong model

- Kết quả sau khi tính được các chỉ số và biểu diễn nó lên biểu đồ:



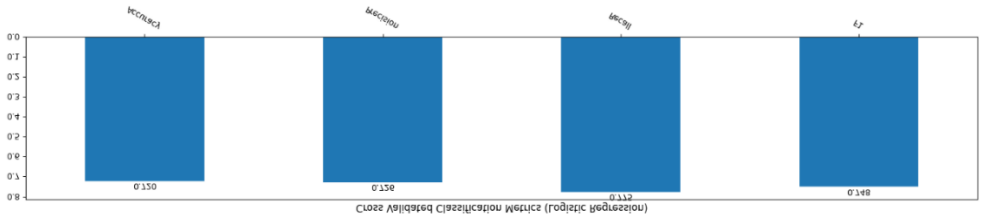
Hình 2. 29 Biểu diễn biểu đồ với mô hình LR

- Tính các chỉ số trong model dựa theo hàm cross_val_score, hàm sẽ giúp chúng ta đánh giá hiệu quả của mô hình trên nhiều bộ train và test khác nhau:

```
def evaluate_classifier(X, y, C, penalty, solver):
    # Create a new classifier with current best parameters
    lr = LogisticRegression(C=C, penalty=penalty, solver=solver)

    # Cross Validated Accuracy
    lr_accuracy = cross_val_score(lr, X, y, scoring='accuracy', cv=5)
    lr_accuracy_mean = np.mean(lr_accuracy)

    # Cross Validated Precision
    lr_precision = cross_val_score(lr, X, y, scoring='precision', cv=5)
    lr_precision_mean = np.mean(lr_precision)

    # Cross Validated Recall
    lr_recall = cross_val_score(lr, X, y, scoring='recall', cv=5)
    lr_recall_mean = np.mean(lr_recall)

    # Cross Validated F1
    lr_f1 = cross_val_score(lr, X, y, scoring='f1', cv=5)
    lr_f1_mean = np.mean(lr_f1)

    # Visualize cross-validated metrics
    lr_metrics = pd.DataFrame({
        'Accuracy': [lr_accuracy_mean],
        'Precision': [lr_precision_mean],
        'Recall': [lr_recall_mean],
        'F1': [lr_f1_mean]
    })

    # Plot the metrics with values on top of each bar
    ax = lr_metrics.T.plot.bar(legend=False)
    plt.title('Cross Validated Classification Metrics (Logistic Regression)')
    plt.xticks(rotation=30)

    # Add values on top of each bar
    for i, v in enumerate(lr_metrics.values.flatten()):
        ax.text(i, v + 0.005, f'{v:.3f}', ha='center', va='bottom')

    plt.show()
```

Hình 2. 30 Tính các chỉ số trong model dựa theo hàm cross_val_score

- Kết quả:



Hình 2. 31 Kết quả

- Hiển thị bảng classification_report.

```
[ ] model_LogisticRegression= LogisticRegression(C=0.0006951927961775605, penalty='l2', solver='liblinear')
    model_LogisticRegression.fit(X_train, y_train)
```

```
LogisticRegression(C=0.0006951927961775605, solver='liblinear')
```

```
# print prediction results
predictions = model_LogisticRegression.predict(X_test)
print(classification_report(y_test, predictions))
```

```
              precision    recall  f1-score   support

           0       0.86      0.89      0.88        28
           1       0.90      0.88      0.89        32

    accuracy                           0.88        60
   macro avg       0.88      0.88      0.88        60
weighted avg       0.88      0.88      0.88        60
```

Hình 2. 32 Hiển thị bảng classification_report

- Hiển thị confusion_matrix

```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test, predictions)
```

```
array([[25,  3],
       [ 4, 28]])
```

Hình 2. 33 Hiển thị confusion_matrix

- Cuối cùng thì ta sẽ có một vài hàm dự đoán của các model:
  - Decision Tree

```
[ ] from sklearn.tree import DecisionTreeClassifier

    def train_decision_tree(age, sex, chest_pain_type, resting_blood_pressure, cholesterol, fasting_blood_sugar, resting_electrocardiogram, max_heart_rate_achieved,
                            exercise_induced_angina, st_depression, st_slope, num_major_vessels, thalassemia, inputs, target, max_depth= 5, min_samples_leaf= 4, min_samples_split= 2):
        # Tạo và đào tạo mô hình Decision Tree với các hyperparameters được cung cấp
        dt_model = DecisionTreeClassifier(max_depth=max_depth, min_samples_split=min_samples_split, min_samples_leaf=min_samples_leaf)
        dt_model.fit(inputs, target)

        # Dự đoán trên tập kiểm thử
        y=dt_model.predict([[age, sex, chest_pain_type, resting_blood_pressure, cholesterol, fasting_blood_sugar, resting_electrocardiogram,
                             max_heart_rate_achieved, exercise_induced_angina, st_depression, st_slope, num_major_vessels, thalassemia]])

        if int(y)==0:
          y=0
        else:
          y=1

        return y
```

Hình 2. 34 Code dự đoán của model Decision Tree

  - Logistic Regression

```
[ ] from sklearn.linear_model import LogisticRegression

    #Lưu ý:
    #solver='lbfgs' dùng cho penalty='l2' or none
    #solver='liblinear' dùng cho penalty='l1'và 'l2'

    def train_logistic_regression(age, sex, chest_pain_type, resting_blood_pressure, cholesterol, fasting_blood_sugar, resting_electrocardiogram, max_heart_rate_achieved,
                                  exercise_induced_angina, st_depression, st_slope, num_major_vessels, thalassemia, inputs, target, C=10, penalty='l1', solver='liblinear'):
        # Tạo và đào tạo mô hình Decision Tree với các hyperparameters được cung cấp
        lr_model = LogisticRegression(C=C, penalty=penalty, solver=solver)
        lr_model.fit(inputs, target)

        # Dự đoán trên tập kiểm thử
        y=lr_model.predict([[age, sex, chest_pain_type, resting_blood_pressure, cholesterol, fasting_blood_sugar, resting_electrocardiogram, max_heart_rate_achieved,
                             exercise_induced_angina, st_depression, st_slope, num_major_vessels, thalassemia]])
        if int(y)==0:
          y=0
        else:
          y=1

        return y
```

Hình 2. 35 Code dự đoán của model Logistic Regresssion

  - KNN

```
[ ] from sklearn.neighbors import KNeighborsClassifier

    def predictions_knn(age, sex, chest_pain_type, resting_blood_pressure, cholesterol, fasting_blood_sugar, resting_electrocardiogram, max_heart_rate_achieved,
                        exercise_induced_angina, st_depression, st_slope, num_major_vessels, thalassemia, inputs, target, algorithm='auto', n_neighbors=5, weights='uniform'):
        # Tạo và đào tạo mô hình Decision Tree với các hyperparameters được cung cấp
        knn_model = KNeighborsClassifier(algorithm=algorithm, n_neighbors=n_neighbors, weights=weights)
        knn_model.fit(inputs, target)

        # Dự đoán trên tập kiểm thử
        y=knn_model.predict([[age, sex, chest_pain_type, resting_blood_pressure, cholesterol, fasting_blood_sugar, resting_electrocardiogram, max_heart_rate_achieved,
                              exercise_induced_angina, st_depression, st_slope, num_major_vessels, thalassemia]])

        if int(y)==0:
          y=0
        else:
          y=1

        return y
```

Hình 2. 36 Code dự đoán của model KNN

- SVC

```
[ ]  from sklearn.svm import SVC

     def predictions_SVC(age, sex, chest_pain_type, resting_blood_pressure, cholesterol, fasting_blood_sugar, resting_electrocardiogram, max_heart_rate_achieved,
                         exercise_induced_angina, st_depression, st_slope, num_major_vessels, thalassemia, inputs, target, C=0.1, gamma= 0.1, kernel='linear'):
         # Tạo và đào tạo mô hình Decision Tree với các hyperparameters được cung cấp
         SVC_model = SVC(C=C, gamma=gamma, kernel=kernel)
         SVC_model.fit(inputs, target)

         # Dự đoán trên tập kiểm thử
         y=SVC_model.predict([[age, sex, chest_pain_type, resting_blood_pressure, cholesterol, fasting_blood_sugar, resting_electrocardiogram, max_heart_rate_achieved,
                               exercise_induced_angina, st_depression, st_slope, num_major_vessels, thalassemia]])

         if int(y)==0:
           y=0
         else:
           y=1

         return y
```

Hình 2. 37 Code dự đoán của model SVC

- Naive Bayes

```
[ ]  from sklearn.naive_bayes import GaussianNB

     def predictions_naive_bayes(age, sex, chest_pain_type, resting_blood_pressure, cholesterol, fasting_blood_sugar, resting_electrocardiogram, max_heart_rate_achieved,
                                 exercise_induced_angina, st_depression, st_slope, num_major_vessels, thalassemia, inputs, target):
         # Tạo và đào tạo mô hình Decision Tree với các hyperparameters được cung cấp
         naive_bayes_model = GaussianNB()
         naive_bayes_model.fit(inputs, target)

         # Dự đoán trên tập kiểm thử
         y=naive_bayes_model.predict([[age, sex, chest_pain_type, resting_blood_pressure, cholesterol, fasting_blood_sugar, resting_electrocardiogram, max_heart_rate_achieved,
                                       exercise_induced_angina, st_depression, st_slope, num_major_vessels, thalassemia]])

         if int(y)==0:
           y=0
         else:
           y=1

         return y
```

Hình 2. 38 Code dự đoán của model Naïve Bayes

- Test dự đoán với 1 trường hợp bất kì chung cho tất cả các model:

```
[ ]  # Đầu vào
     from sklearn.utils.validation import check_array
     inputs = data.drop('disease', axis='columns')
     inputs = check_array(inputs, ensure_2d=True)
     target = data['disease']

     predictions_decision_tree = train_decision_tree(59, 0, 1, 130, 236, 1, 2, 174 ,0, 0.0, 1, 1 ,1, inputs, target, max_depth= 5, min_samples_leaf= 4, min_samples_split= 2)
     predictions_logistic_regression = train_logistic_regression(59, 0, 1, 130, 236, 1, 2, 174 ,0, 0.0, 1, 1 ,1, inputs, target, C=0.0006951927961775605, penalty='l2', solver='liblinear')
     predictions_knn = predictions_knn(59, 0, 1, 130, 236, 1, 2, 174 ,0, 0.0, 1, 1 ,1, inputs, target, algorithm='auto', n_neighbors=5, weights='uniform')
     predictions_SVC = predictions_SVC(59, 0, 1, 130, 236, 1, 2, 174 ,0, 0.0, 1, 1 ,1, inputs, target, C=0.1, gamma= 0.1, kernel='linear')
     predictions_naive_bayes = predictions_naive_bayes(59, 0, 1, 130, 236, 1, 2, 174 ,0, 0.0, 1, 1 ,1, inputs, target)
     # In kết quả dự đoán
     print(predictions_decision_tree)
     print(predictions_logistic_regression)
     print(predictions_knn)
     print(predictions_SVC)
     print(predictions_naive_bayes)
```

```
1
1
1
1
1
```

Hình 2. 39 Test dự đoán đối với từng mô hình ML

⇨ Dự đoán trường hợp trên qua các model đều cho ra kết quả là 1, tức là người có các chỉ số điều kiện trên được dự đoán là bị mắc bệnh tim mạch.

TÀI LIỆU THAM KHẢO

[1]: https://drive.google.com/file/d/1fJyPy0mqglkQ_eSqaoRa8zJbZ9lFAP-b/view

[2]: https://drive.google.com/file/d/1xyJZKh6eX2KwSwDh3FpU4MOTMY8PNCBJ/view

[3]: https://drive.google.com/file/d/1-ZnbuiqPYihiDPk-_OG77iAfyVvDWOAp/view

[4]: https://drive.google.com/file/d/1f7QWTxF5KP1bRfr-4hp0ZkAUD-3aJ7Mv/view

[6]: https://machinelearningcoban.com/2017/01/27/logisticregression/

[7]: https://machinelearningcoban.com/2017/01/08/knn/

[8]: https://machinelearningcoban.com/tabml_book/ch_model/decision_tree.html

[9]: https://www.geeksforgeeks.org/classifying-data-using-support-vector-machinessvms-in-python/

[10]: https://www.geeksforgeeks.org/ml-naive-bayes-scratch-implementation-using-python/