

TRƯỜNG ĐẠI HỌC SƯ PHẠM KĨ THUẬT TP.HCM

KHOA CÔNG NGHỆ THÔNG TIN



HCMUTE

**SỬ DỤNG SUPPORT VECTOR
MACHINE CHUẨN ĐOÁN NGƯỜI MẮC
BỆNH TIM**

**BỘ MÔN: Học máy
HỌC KỲ 1 – NĂM HỌC 2023-2024
Giảng viên hướng dẫn: Từ Tuyết Hồng**

Sinh viên thực hiện:

Mã số sinh viên: 22110134

Họ và tên: Phan Phúc Hảo

Thành phố Hồ Chí Minh, Tháng 12 năm 2024

I) Giới thiệu về data	3
1) Nguồn Dataset	3
2) Nguồn code tham khảo	3
3) Nội dung data	3
3.1) Mục đích của data	3
3.2) Các thành phần trong data	3
4) Sơ lược về dữ liệu trong Dataset	5
4.1) Các loại dữ liệu trong Dataset	5
4.2) Tổng quát về dữ liệu của từng cột	5
4.2.1) Kiểu dữ liệu số	5
a) Age	5
b) RestingBP	6
c) Cholesterol	7
d) MaxHR	8
e) Oldpeak	9
4.2.2) Kiểu dữ liệu là loại	9
a) Sex	9
b) ChestPainType	10
c) FastingBS	11
d) RestingECG	11

e) ExerciseAngina	12
f) ST_Slope	13
g) HeartDisease	13
5) Mối tương quan giữa các cột so với giá trị mục tiêu	14
5.1) Dữ liệu dạng số so với giá trị mục tiêu	14
5.1.1) Age.....	14
5.1.2) RestingBP	14
5.1.3) Cholesterol.....	15
5.1.4) MaxHR.....	16
5.1.5) OldPeak.....	17
5.2) Kiểu dữ liệu là loại so với giá trị mục tiêu	18
5.2.1) Sex.....	19
5.2.2) ChestPainType	20
5.2.3) FastingBS	21
5.2.4) RestingECG	22
5.2.5) ExerciseAngina	23
5.2.6) ST_Slope	24
5.3) Tương quan giữa giá trị dạng loại và số so với giá trị mục tiêu	24
5.3.1) Sex	24
a) Age	24

b) RestingBP	25
c) Cholesterol	25
d) OldPeak	25
e) MaxHR	25
5.3.2) ChestPainType	26
a) Age	26
b) Resting BP	26
c) Cholesterol	26
d) OldPeak	27
5.3.3) FastingBS	28
a) Age	28
b) RestingBP	28
c) Cholesterol	28
d) Oldpeak	29
e) MaxHR	29
5.3.4) RestingECG	29
a) Age	29
b) RestingBP	30
c) Cholesterol	30
d) Oldpeak	30

e) MaxHR.....	31
5.3.5) ExerciseAngina	31
a) Age	31
b) RestingBP	31
c) Cholesterol	32
d) Oldpeak.....	32
e) MaxHR.....	32
5.3.6) ST_Slope.....	33
a) Age	33
b) RestingBP	33
c) Cholesterol	33
d) Oldpeak.....	34
e) MaxHR.....	34
5.4) Tương quan giữa các giá trị dạng số với nhau	34
5.4.1) Age và RestingBP	34
5.4.2) Age và Cholesterol	35
5.4.3) Age và MaxHR.....	36
5.4.4) Age và Oldpeak.....	37
5.4.5) RestingBP và Cholesterol	37
5.4.6) RestingBP và MaxHR.....	38

5.4.7) RestingBP và Oldpeak	39
5.4.8) Cholesterol và MaxHR	39
5.4.9) Cholesterol và Oldpeak	40
5.4.10) MaxHR và Oldpeak	41
6) Ma trận tương quan	41
6.1) Đồi với giá trị số	43
6.2) Đồi với giá trị loại	43
7) Kết luận về Dataset	45
II) Tiền xử lý dữ liệu	46
1) Xử lý các giá trị Null	46
2) Data Scaling	46
2.1) Normalization là gì	47
2.2) Standardization là gì	47
2.3) Chọn phương pháp nào tốt hơn?	47
2.4) Dữ liệu sau khi chuẩn hóa	48
3) Xử lý dữ liệu dạng cột	48
3.1) Đồi với các cột có giá trị là nhị phân (Yes hoặc No, Male hoặc Female, ...)	48
3.2) Đồi với các cột có số giá trị lớn hơn ba	48
4) Chia tách dữ liệu	49
4.1) K-fold cross-validation	49

4.2) Stratified k-fold cross-validation	49
4.3) Cái nào phù hợp hơn	49
III) Thuật toán Support vector machine	50
1) Giới thiệu về các thuật ngữ, đại lượng dùng để xác định tính chính xác của mô hình	50
2) Giới thiệu thuật toán SVM	52
2.1) Support Vector Machine là gì	52
2.2) Giải thích sơ lược về thuật toán	52
3) Thuật toán SVM	52
3.1) SVM phân loại dữ liệu tuyến tính	52
3.1.1) Dữ liệu tuyến tính là gì	52
3.1.2) Hard margin	53
a) Hard margin là gì?	53
b) Cơ sở lý thuyết	53
c) Áp dụng thuật toán vào Dataset	56
d) Đánh giá thuật toán	56
3.1.3) Soft Margin	57
a) Soft Margin là gì	57
b) Cơ sở lý thuyết	57
c) Áp dụng thuật toán vào Dataset	60
2.2) SVM phân loại dữ liệu phi tuyến tính	65

2.3 Kernel	66
2.3.1) Kernel là gì	66
2.3.2) Cơ sở lý thuyết	67
2.3.3) Tính chất của các hàm Kernel	68
2.3.4) Các hàm kernel thông dụng	69
2.4) Áp dụng các kernel vào SVM	69
2.4.1) Polynomial Kernel	69
a) Áp dụng thuật toán	70
b) Đánh giá thuật toán	73
2.4.2) Gaussian RBF Kernel	75
a) Áp dụng thuật toán	75
b) Đánh giá thuật toán	79

LỜI CẢM ƠN

_Với lòng kính trọng và biết ơn sâu sắc, em xin gửi lời cảm ơn chân thành tới cô Từ Tuyết Hồng người đã giảng dạy kiến thức về môn Học máy giúp em hoàn thành được báo cáo này.

_Mặc dù đã rất nỗ lực, song do bản thân còn nhiều hạn chế về kiến thức nên khó tránh khỏi có những thiếu sót trong bài làm. Em mong nhận được sự đóng góp ý kiến của cô để hiểu rõ hơn môn học.

LÝ DO CHỌN ĐỀ TÀI

_ Khi trí tuệ nhân tạo ngày càng phát triển người ta muốn áp dụng các công nghệ đó vào y học để có thể chẩn đoán bệnh dựa trên các dữ liệu có sẵn như (nhịp tim, cholesterol trong máu, huyết áp...) hay đưa ra các lời khuyên để có sức khỏe tốt hơn như điều chỉnh lại thói quen ăn uống, điều chỉnh giờ giấc sinh hoạt...

_ Và tất cả các điều trên đều dựa trên sự phát triển của học máy (Machine Learning). Thông qua các mô hình học máy, chúng ta có thể phân tích xem một bệnh nhân dựa trên các dữ liệu có sẵn để chẩn đoán xem bệnh nhân đó có mắc bệnh hay không dựa trên các dữ liệu của bệnh nhân so với các bệnh nhân đã mắc bệnh đó, nếu họ có các dấu hiệu giống với người đã mắc bệnh ta có thể chẩn đoán bệnh cho bệnh nhân.

_ Đề tài "Chuẩn đoán bệnh tim cho bệnh nhân" không chỉ mang ý nghĩa học thuật mà còn có giá trị thực tiễn cao, nó là tiền đề để chúng ta có thể đi sâu hơn vào việc áp dụng lĩnh vực học máy vào y học. Ở nơi mà kiến thức cần thiết để có thể trở thành bác sĩ là rất nhiều. Tuy nhiên với công nghệ hiện giờ nhiều người vẫn chưa tin rằng việc sử dụng AI để chẩn đoán bệnh là chính xác và còn nhiều vấn đề liên quan đến trách nhiệm và đạo đức. Nhưng trong tương lai khi mà AI đã phát triển hơn thì việc ứng dụng học máy (ML) vào y học sẽ trở thông dụng hơn và được nhiều người đón nhận hơn.

Chương 1: Giới thiệu về data

1) Nguồn Dataset

<https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction/data>

2) Nguồn code tham khảo

<https://www.kaggle.com/code/durgancegaur/a-guide-to-any-classification-problem>

<https://www.kaggle.com/code/peterhu2022/heart-support-vector-machine/notebook>

<https://www.kaggle.com/code/tanmay111999/heart-failure-prediction-cv-score-90-5-models>

3) Nội dung data

3.1) Mục đích của data

_Dataset này sẽ phân tích xem một người với các thông số như: tuổi, giới tính, các loại cơn đau ở ngực, huyết áp tâm thu, cholesterol, ... Để chuẩn đoán xem một người có bị bệnh tim hay không.

3.2) Các thành phần trong data

Age	int64
Sex	string[python]
ChestPainType	string[python]
RestingBP	int64
Cholesterol	int64
FastingBS	int64
RestingECG	string[python]
MaxHR	int64
ExerciseAngina	string[python]
Oldpeak	float64
ST_Slope	string[python]
HeartDisease	int64
dtype:	object

Tổng quan dữ liệu của các cột có dữ liệu là số

	count	mean	std	min	25%	50%	75%	max
Age	918.0	53.510893	9.432617	28.0	47.00	54.0	60.0	77.0
RestingBP	918.0	132.396514	18.514154	0.0	120.00	130.0	140.0	200.0
Cholesterol	918.0	198.799564	109.384145	0.0	173.25	223.0	267.0	603.0
FastingBS	918.0	0.233115	0.423046	0.0	0.00	0.0	0.0	1.0
MaxHR	918.0	136.809368	25.460334	60.0	120.00	138.0	156.0	202.0
Oldpeak	918.0	0.887364	1.066570	-2.6	0.00	0.6	1.5	6.2
HeartDisease	918.0	0.553377	0.497414	0.0	0.00	1.0	1.0	1.0

Vậy dữ liệu của Dataset bao gồm:

- + Age (Tuổi): [Year]
- + Sex (Giới tính): [**M**: Male, **F**: Female]
- + ChestPainType (Loại đau ở ngực): [**TA**: Typical Angina, **ATA**: Atypical Angina, **NAP**: Non-Anginal Pain, **ASY**: Asymptomatic]
 - + RestingBP (Huyết áp tâm thu hay còn gọi là huyết áp tối đa): có giá trị trong khoảng từ 0 đến 200 [mm/dl]
 - + Cholesterol: (Mỡ trong máu) [mm/dl]
 - + FastingBS (Lượng đường trong máu): [**1**: nếu FastingBS > 120 mg/dl, **0**: còn lại]
 - + RestingECG: Kết quả đo điện tâm đồ lúc nghỉ ngơi [**Normal**: Bình thường, **ST**: có bất thường ở sóng ST-T (sóng T đảo ngược và/hoặc ST chênh lên hoặc chênh xuống > 0,05 mV), **LVH**: cho thấy có khả năng hoặc chắc chắn phì đại thất trái theo tiêu chuẩn của Estes]
 - + MaxHR: Nhịp tim tối đa đạt được (Giá trị trong khoảng 60 đến 190)
 - + ExerciseAngina: Đau thắt ngực do gắng sức (**Y**: Yes, **N**: No)
 - + Oldpeak: Chênh lệch đoạn ST khi tập thể dục so với lúc nghỉ
 - + ST_Slope: Độ dốc tại đỉnh của đoạn ST khi tập thể dục [**Up**: upsloping, **Flat**: flat, **Down**: downsloping]
 - + HeartDisease: Kết quả chẩn đoán: [**1**: Heart disease, **0**: Normal]

4) Sơ lược về dữ liệu trong Dataset

4.1) Các loại dữ liệu trong Dataset

Có hai loại bao gồm:

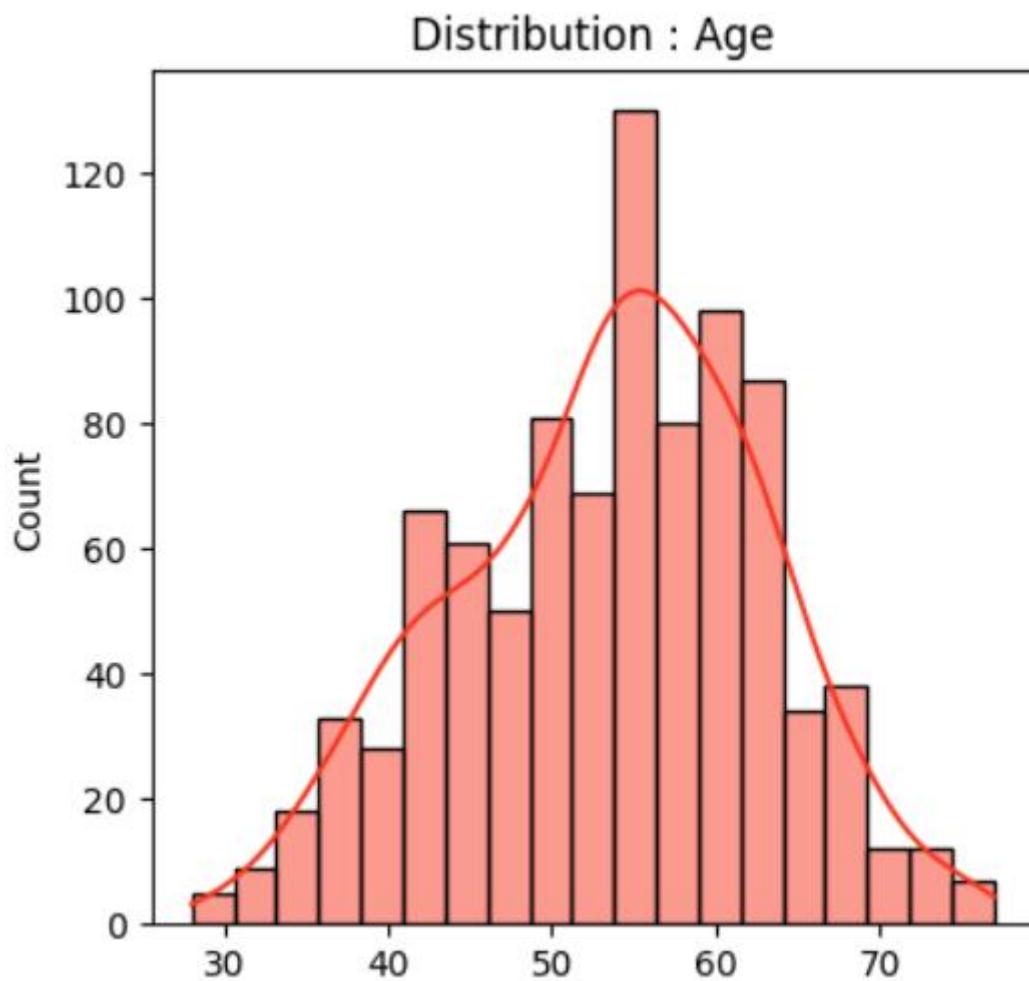
+ Dữ liệu kiểu số: Age, RestingBP, Cholesterol, FastingBS, MaxHR, Oldpeak

+ Dữ liệu kiểu object: Sex, ChestPainType, FastingBS, RestingECG, ExerciseAngina, ST_Slope, HeartDisease.

4.2) Tổng quát về dữ liệu của từng cột

4.2.1) Kiểu dữ liệu số

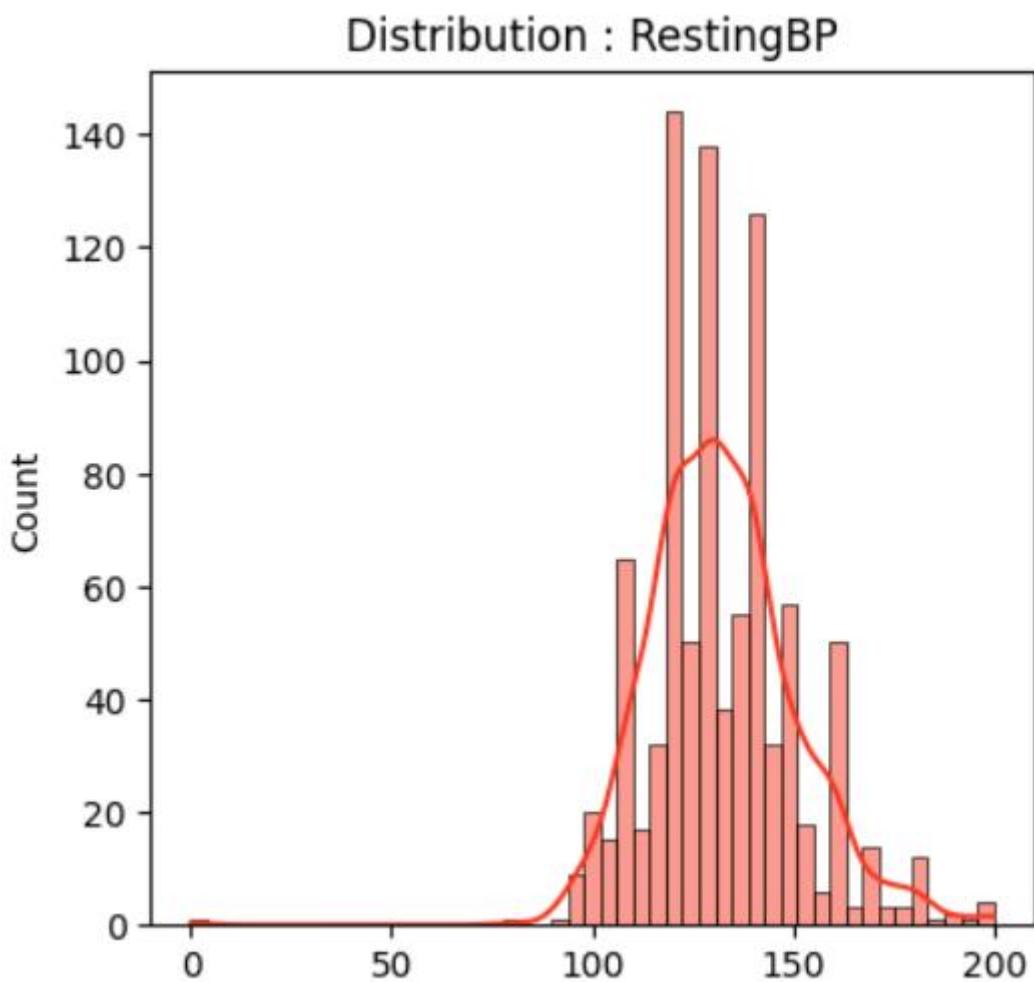
a) Age



_Qua biểu đồ ta thấy được dữ liệu phân phối khá đều từ trước 30 tuổi đến sau 70 tuổi.

_Số lượng bệnh nhân từ 50 đến 60 tuổi chiếm nhiều nhất và giảm dần ở các mức tuổi cao hơn hoặc thấp hơn.

b) RestingBP

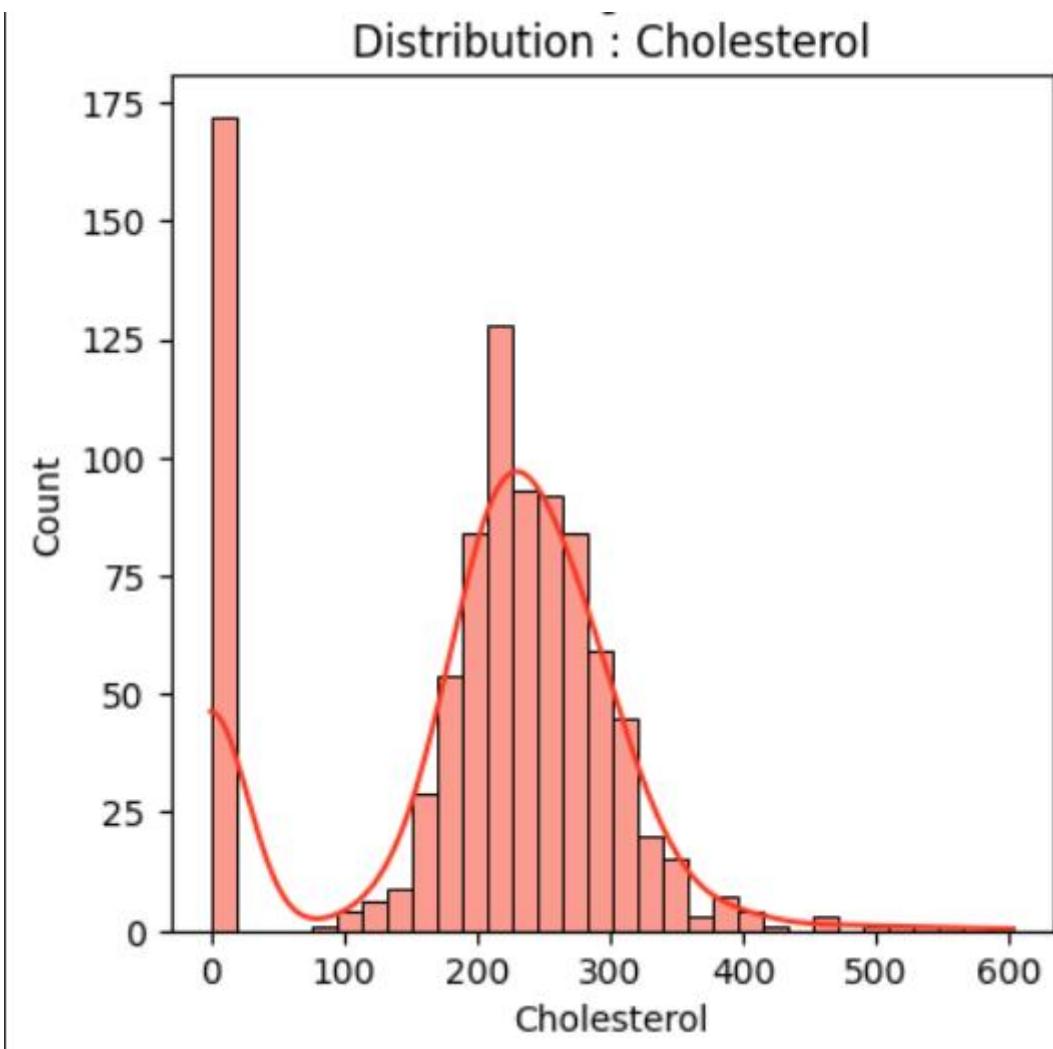


_RestingBP (huyết áp lúc nghỉ) có phân phối nghiêng về phía bên phải, với đa số các giá trị tập trung quanh khoảng 100-150 mm/dl.

_Một số giá trị bất thường được quan sát thấy ở mức thấp hơn (dưới 50 mm/dl) và cao hơn (gần 200 mm/dl), có thể là các ngoại lệ.

_Đa phần bệnh nhân có huyết áp trong ngưỡng bình thường đến cao nhẹ.

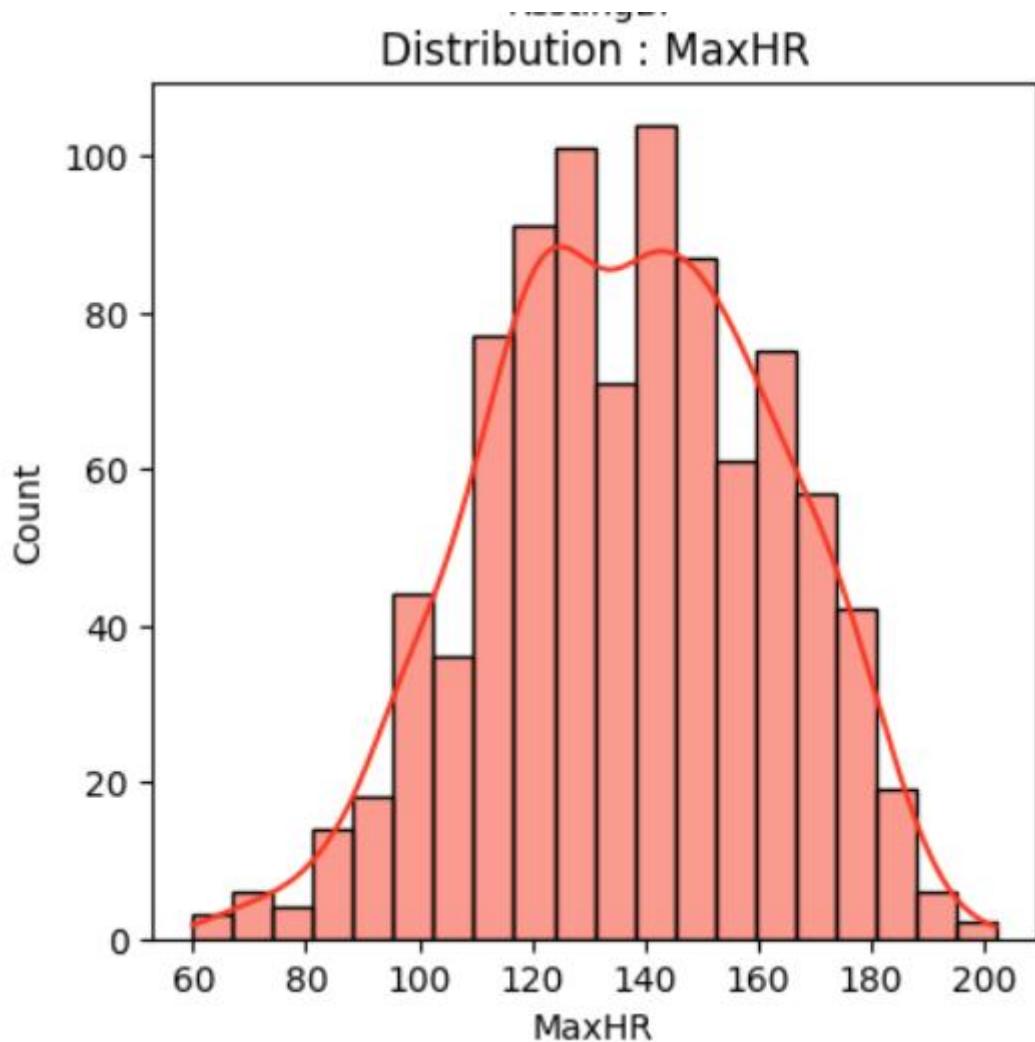
c) Cholesterol



_Phân phối cholesterol có hai nhóm rõ rệt: một nhóm ở mức 0 và một nhóm với phân phối tương đối chuẩn, tập trung trong khoảng 100-300 mm/dl.

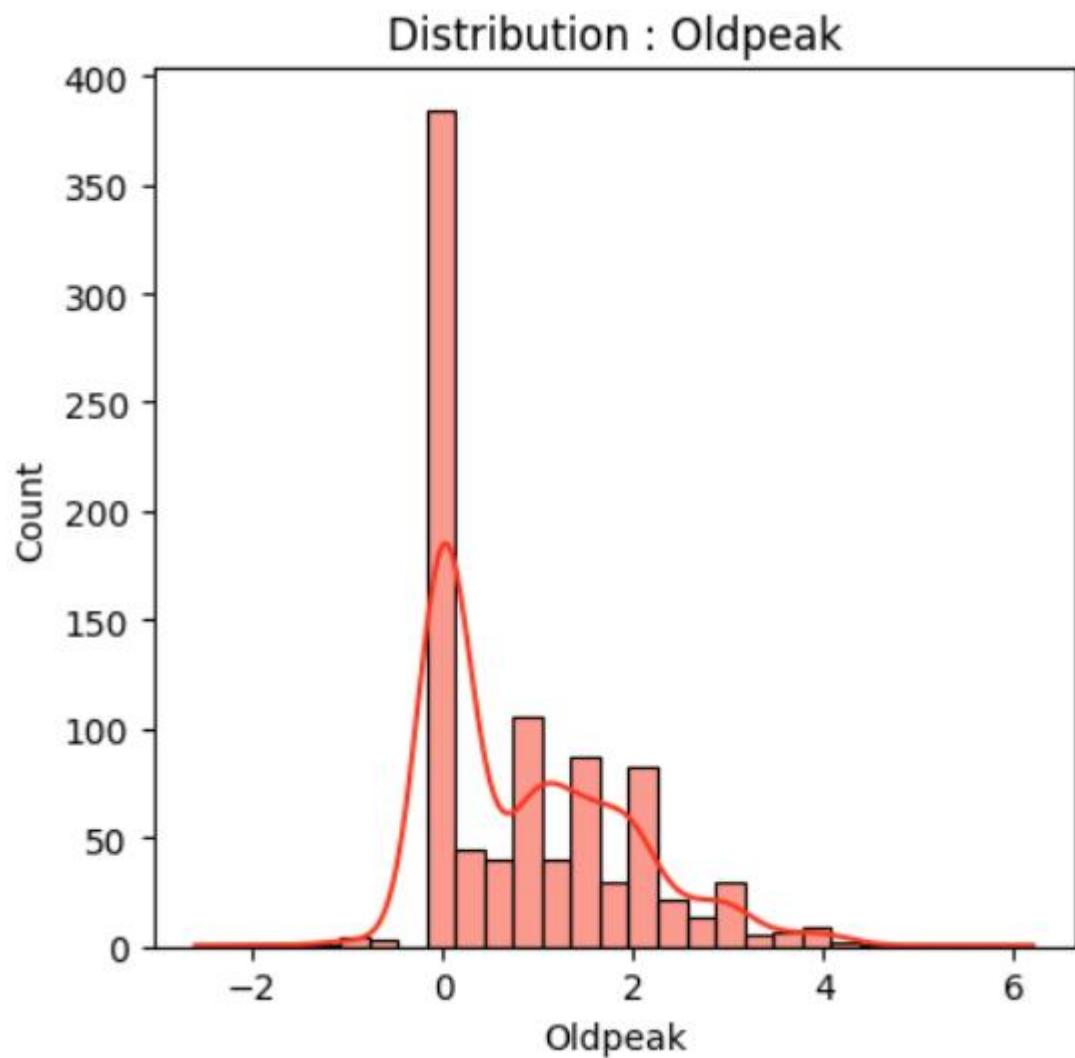
_Nhóm cholesterol có số lượng người nhiều nhất tập trung quanh 200-250 mm/dl, cho thấy nhiều bệnh nhân có nguy cơ cao về vấn đề mỡ trong máu.

d) MaxHR



_MaxHR (nhịp tim tối đa) được phân phối khá đều trong khoảng từ 100 đến 180.

e) Oldpeak



_Oldpeak có 1 đỉnh 0 cao nhất so với phần còn lại của dữ liệu.

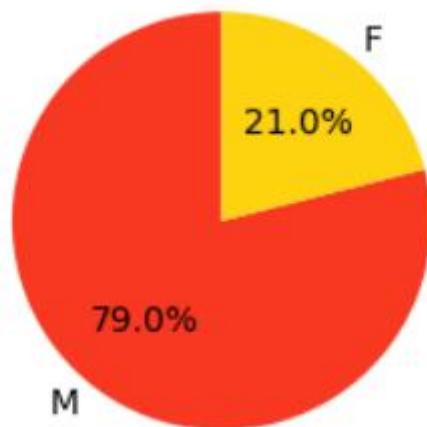
_Từ -2 đến trước 0 và từ lớn hơn 2 đến 4 có số lượng rất ít có thể là các giá trị ngoại lệ.

4.2.2) Kiểu dữ liệu là loại

a) Sex

_Sex: [Male: 1, Female: 0]

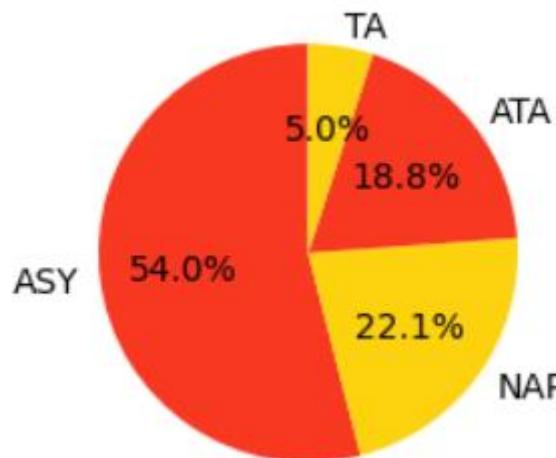
Distribution: Sex



_Dữ liệu cho thấy số người nam chiếm 79% dữ liệu khảo sát nữ chỉ có 21%. Sự mất cân bằng này có thể ảnh hưởng đến khả năng đánh giá bệnh nhân.

b) ChestPainType

Distribution: ChestPainType



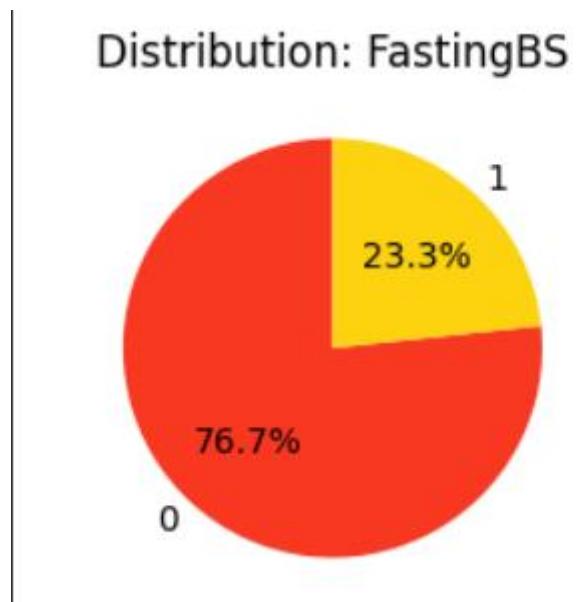
_Dữ liệu cho thấy đa số người tham gia có giá trị ChestPainType là ASY tức là không có triệu chứng đau ngực chiếm 54%.

_Giá trị loại ATA là loại đau ngực không bình thường và NAP cho loại đau ngực không do đau thắt ngực khá bằng nhau với tỉ lệ lần lượt là 18.8% và 22.1%.

_Giá trị cho loại TA cho cơn đau ngực thông thường khá thấp với tỉ lệ chỉ có 5%.

c) FastingBS

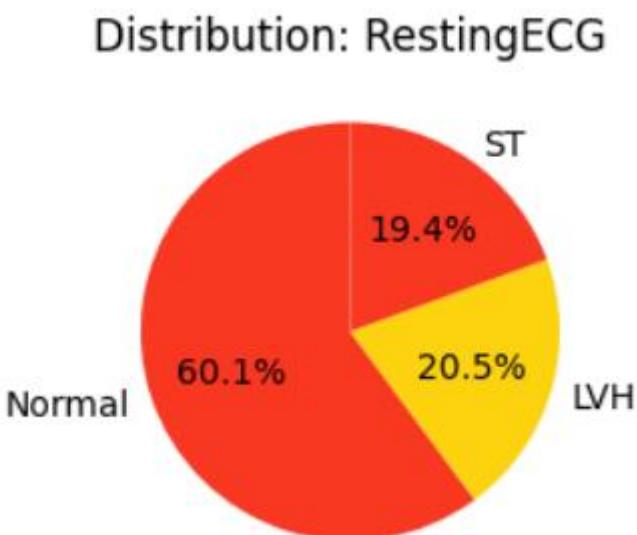
[**1**: nếu FastingBS > 120 mg/dl, **0**: còn lại]



_ Thông qua biểu đồ ta thấy số người có FastingBS (chỉ số đường huyết) cao hơn 120mg/dl ít hơn rất nhiều so với số người có FastingBS thấp hơn 120mg/dl.

_ Chúng ta 76.7% bệnh nhân có lượng đường huyết bình thường và có khoảng 23.3% bệnh nhân có dấu hiệu của bệnh tiểu đường.

d) RestingECG

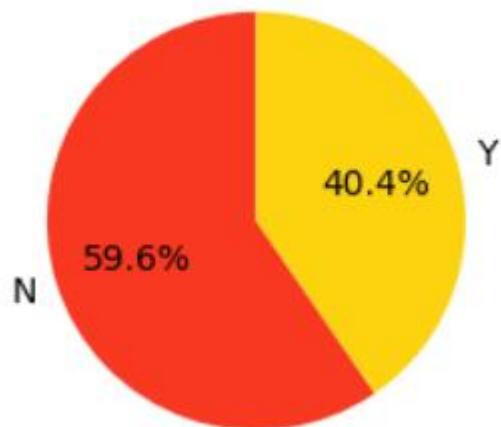


_ Ta thấy chỉ số RestingECG có giá trị là Normal chiếm phần lớn trong biểu đồ là 60.1%.

_ Số người có chỉ số RestingECG là LVH và ST thấp hơn nhưng cũng chiếm lần lượt khoảng 20.5% và 19.4%.

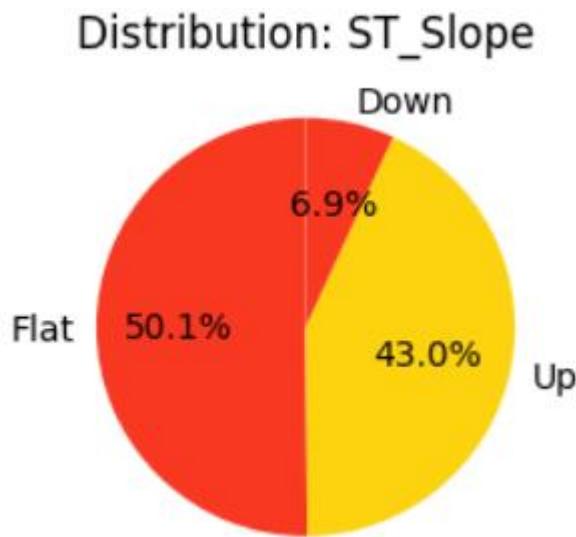
e) ExerciseAngina

Distribution: ExerciseAngina



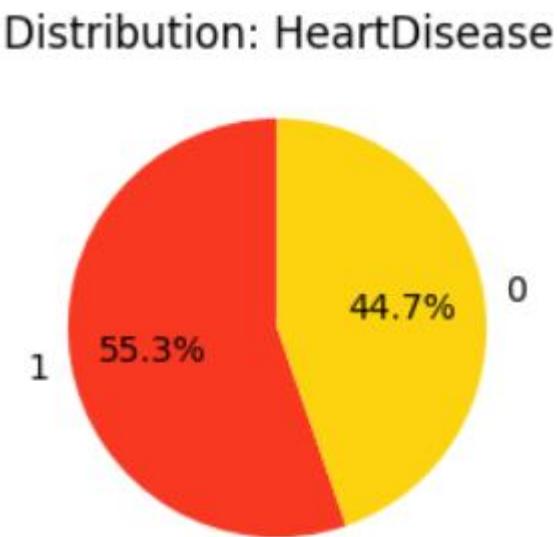
_ Biểu đồ cho ta thấy tỉ lệ giữa số bệnh nhân đau thắt ngực do gắng sức (ExerciseAngina =Y) ít hơn so với bệnh nhân không bị (ExerciseAngina=N) với tỉ lệ lần lượt là khoảng 40.4% và 59.6%.

f) ST_Slope



_Qua biểu đồ ta thấy ST_Slope có giá trị chủ yếu là Flat 50.1% và Up 43% và giá trị Down thì rất thấp chỉ có 6.9%.

g) HeartDisease

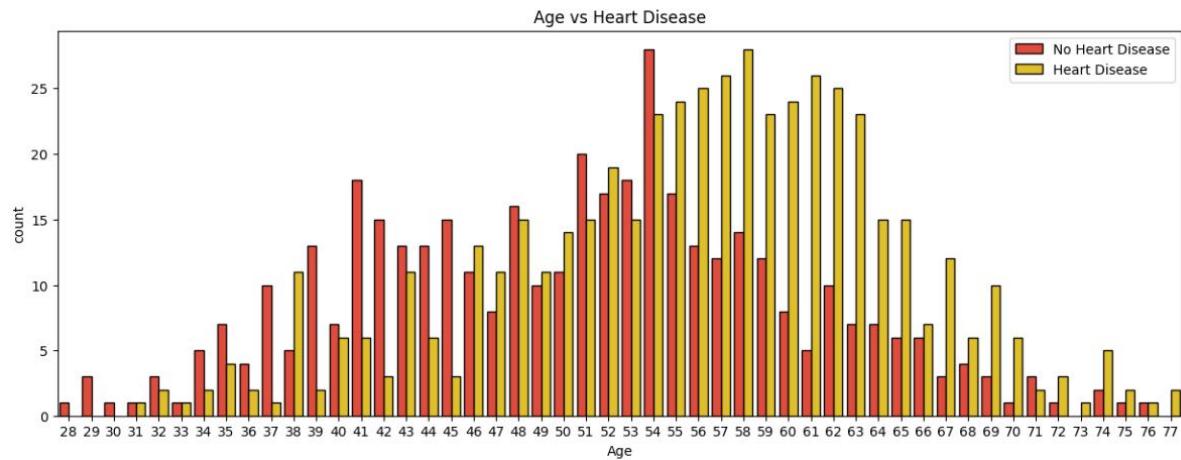


_Biểu đồ cho thấy số người bị bệnh tim và không bị bệnh tim khá là tương xứng nhau với tỉ lệ lần lượt là 57.9% và 42.1%. Cho thấy kết quả của dữ liệu khá cân bằng.

5) Mối tương quan giữa các cột so với giá trị mục tiêu

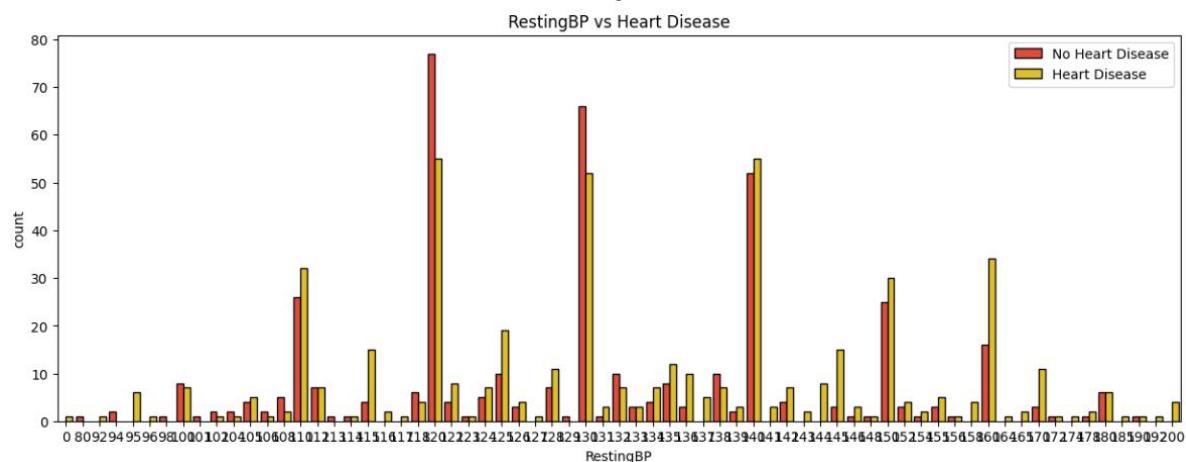
5.1) Dữ liệu dạng số so với giá trị mục tiêu

5.1.1) Age



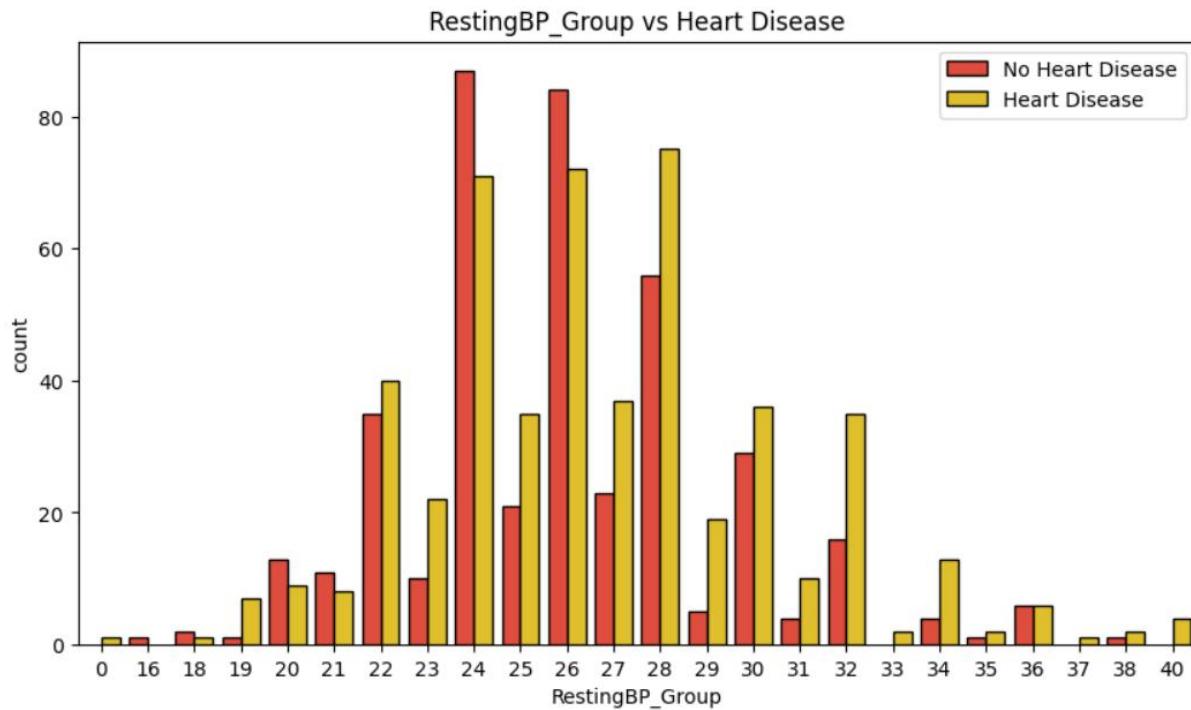
_Ta thấy khi mà tuổi từ 50 tuổi đến khoảng 70 tuổi thì tỉ lệ người bị bệnh tim tăng cao.
Qua đó ta thấy khi tuổi càng tăng thì xác suất bị bệnh tim càng tăng.

5.1.2) RestingBP



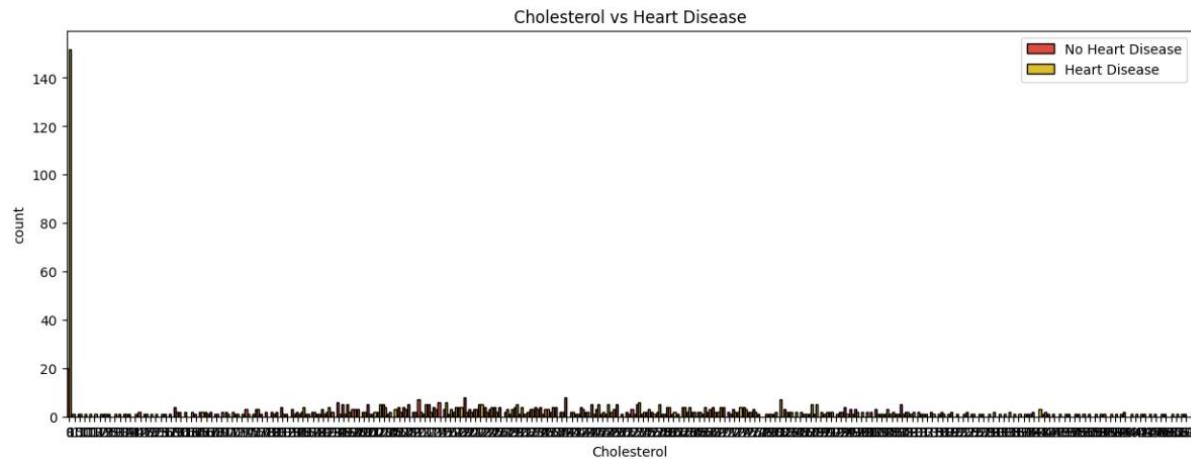
_Do dữ liệu có quá nhiều giá trị nên ta sẽ khó mà tìm ra được thông tin từ đó. Nên ta sẽ chuyển các dữ liệu này sang kiểu dữ loại dạng loại để dễ dàng hình dung. Bằng phương pháp HashMap ta sẽ chia cho 5 để gom nhóm dữ liệu cho dễ dàng hình dung.

```
df1['RestingBP_Group'] = [int(i / 5) for i in df1['RestingBP']]
```



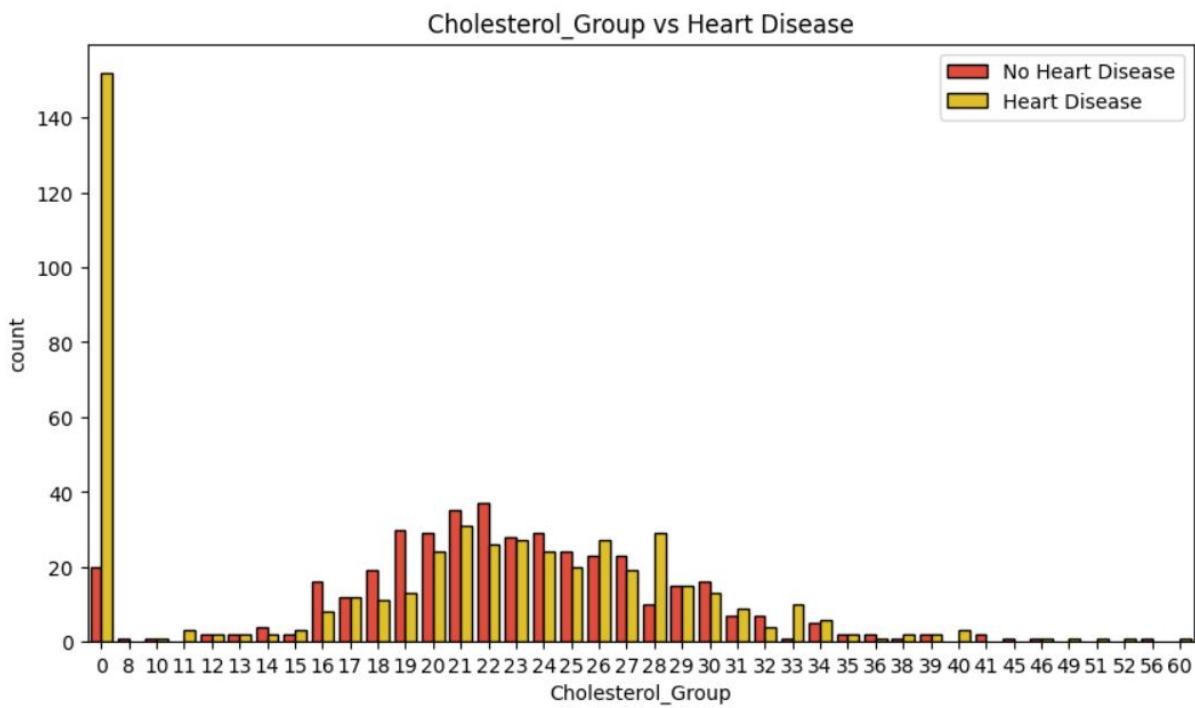
_Vậy dựa vào biểu đồ ta thấy dữ liệu có giá trị nhóm từ khoảng 19 ($19*5 = 95$ trong dữ liệu gốc) đến 34 ($34*5 = 170$ trong dữ liệu gốc) sẽ có số người mắc bệnh tim cao.

5.1.3) Cholesterol



_Cũng tương tự như RestingBP nhưng ở đây chúng ta nên chia cho 10 do dữ liệu nhiều hơn so với RestingBP.

```
df1['Cholesterol_Group'] = [ int(i / 10) for i in df1['Cholesterol']]
```

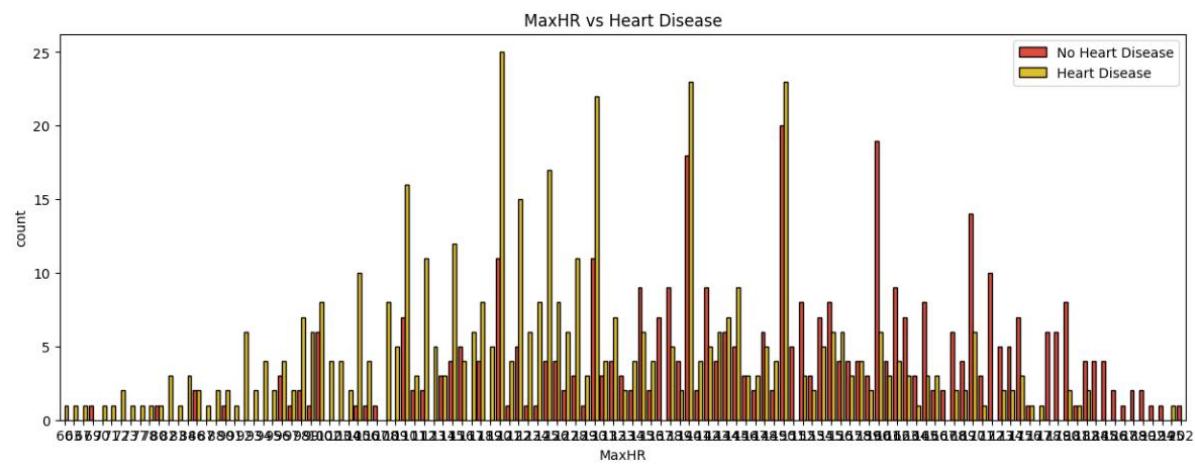


_Ở đây ta thấy rằng nhóm có giá trị 0 (0 trong giá trị gốc) có tỉ lệ bệnh tim khá cao. Nhưng đây có thể là số người có Cholesterol thấp bát thường (bé hơn 10) hoặc thiếu dữ liệu nên được để mặc định. Do đó, sự hiện diện của nhóm này không phản ánh tình trạng sức khỏe một cách chính xác mà có thể là một lỗi trong dữ liệu.

_Còn có nhóm có giá trị từ 16 (16*10 trong giá trị gốc) đến 34 (34*10 trong giá trị gốc) là thể hiện rõ tỉ lệ người bị bệnh tim hay không.

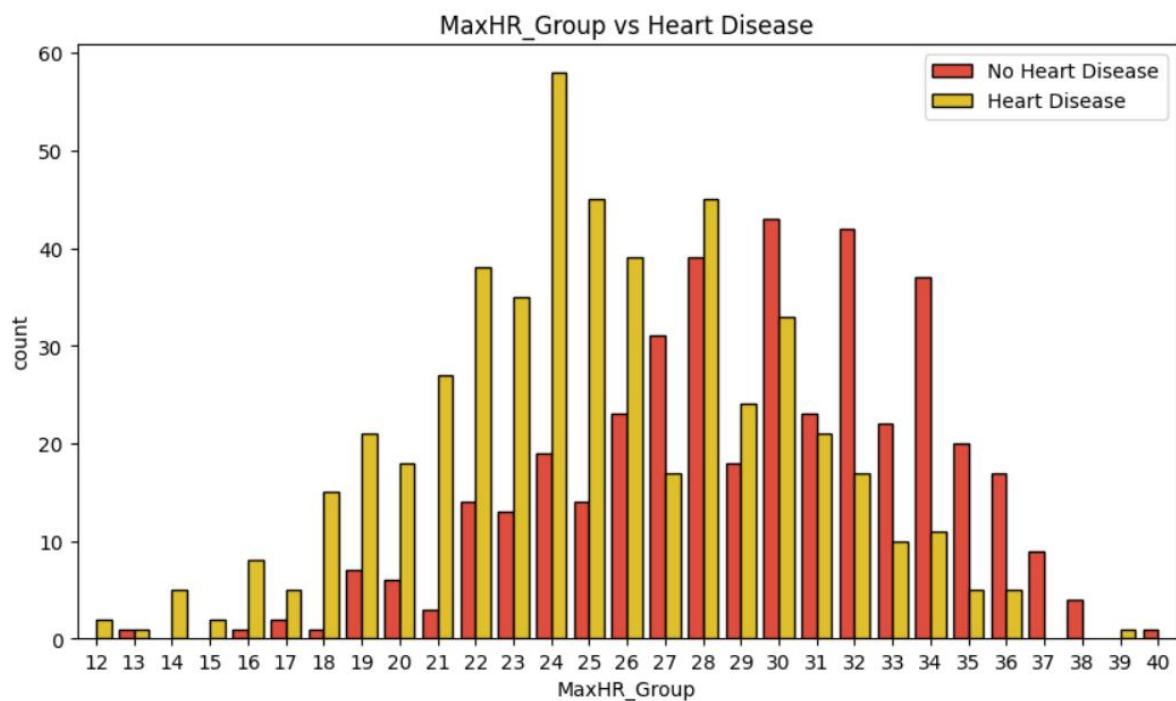
_Và khi giá trị Cholesterol càng tăng ta nhận thấy tỉ lệ của người mắc bệnh tim ở nhóm giá trị đó cũng tăng. Nhưng nó vẫn không cao hơn so với người không mắc bệnh tim.

5.1.4) MaxHR



_ Ở đây giá trị vẫn quá nhiều nên ta vẫn sẽ băm nó ra bằng cách chia 5. Để gom nhóm lại cho dễ dàng hình dung.

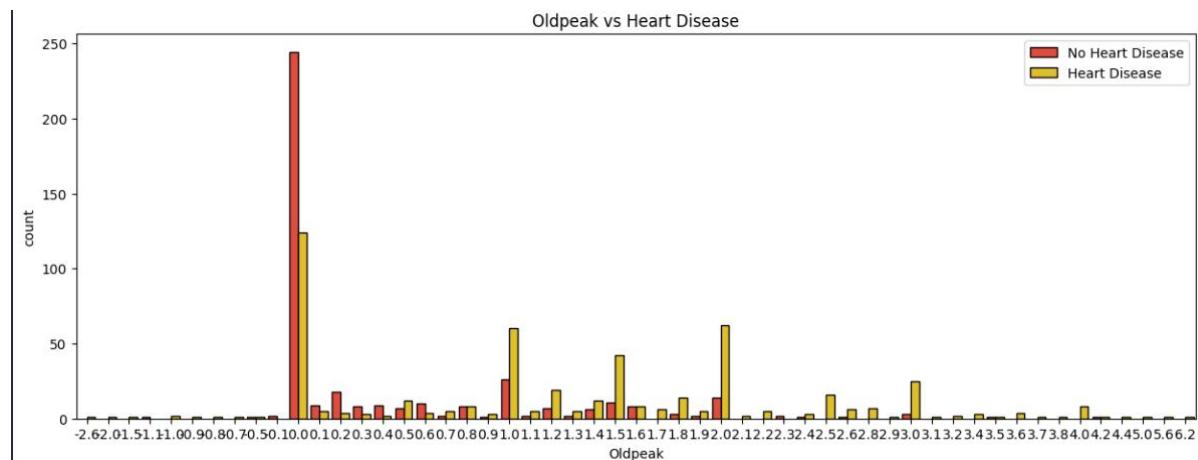
```
df1['MaxHR_Group'] = [ int(i / 5) for i in df1['MaxHR']]
```



_ Ta thấy dữ liệu cho thấy tỉ lệ bệnh tim xuất hiện nhiều bắt đầu từ 14 ($14*5=70$ trong dữ liệu gốc) đến 36 ($36*5=180$ trong dữ liệu gốc)

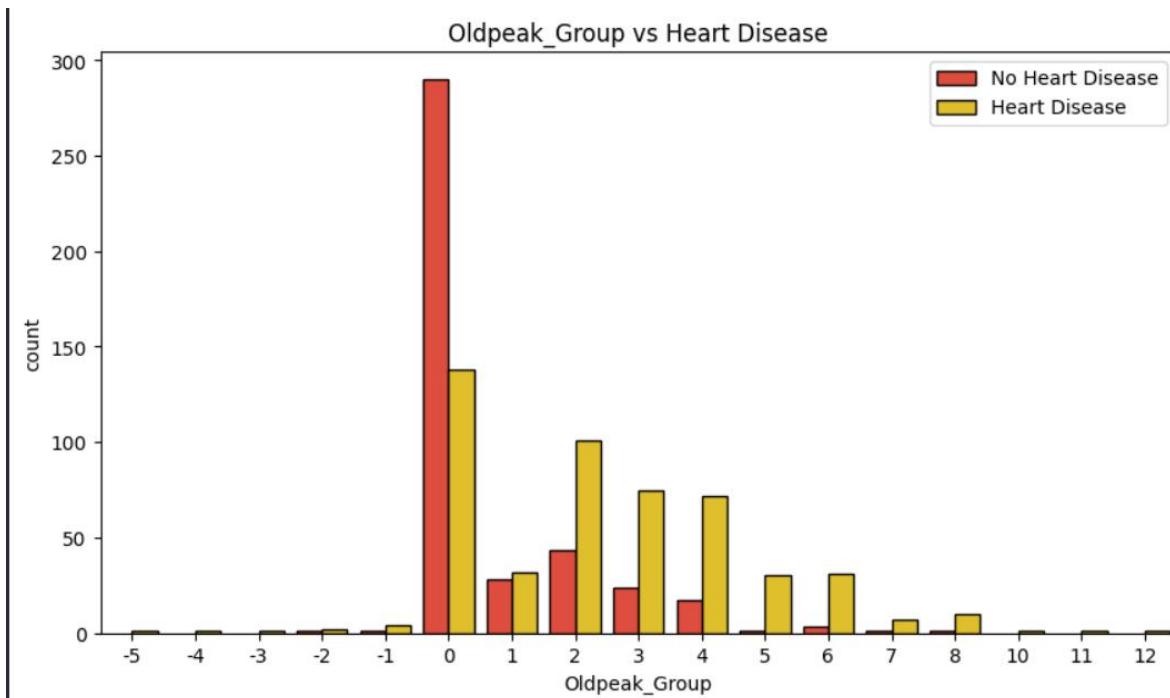
_ Vậy ta thấy tỉ lệ người mắc bệnh tim so với người không mắc bệnh tim sẽ tăng từ nhóm có giá trị là 14 đến nhóm 26 và sau đó giảm dần đến nhóm 36.

5.1.5) OldPeak



_ Vẫn có khá nhiều giá trị duy nhất nên ta chia hình dung rõ được nên ta vẫn sẽ gom nhóm nó lại nhưng lần hơi khác một chút đó là ta không thể chia ngay cho 5 hoặc 10 do giá trị của dữ liệu bé hơn nên ta sẽ ($\text{data} * 10 / 5$) (với data là dữ liệu của OldPeak).

```
df1['Oldpeak_Group'] = [ int( (i*10) / 5) for i in df1['Oldpeak']]
```



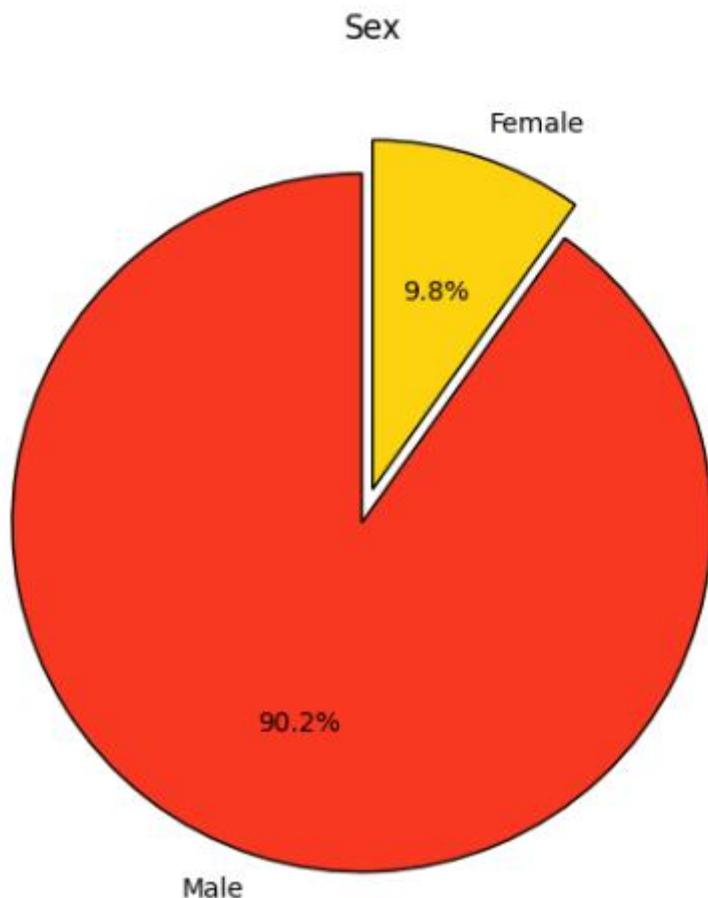
_ Dựa vào đồ thị ta thấy được rõ tỉ lệ giữa người mắc bệnh tim và không mắc bệnh tim là khoảng nhóm 0 (là nhóm có giá trị khoảng 0 đến 0.4 trong dữ liệu gốc) đến nhóm 8 ($8 * 5 / 10 = 4$ trong giá trị gốc).

_ Vậy ta thấy rằng ngoại trừ nhóm 0 ra thì khi ta tăng OldPeak thì tỉ lệ người bệnh tim cũng sẽ tăng.

5.2) Kiểu dữ liệu là loại so với giá trị mục tiêu

_ Ở các kiểu dữ liệu dạng loại ta sẽ xét mối tương quan giữa cột đối với giá trị 1 của cột mục tiêu, tức là ta sẽ xét ứng với mỗi tập giá trị của cột đó thì ta sẽ có số người bị mắc bệnh tim là bao nhiêu.

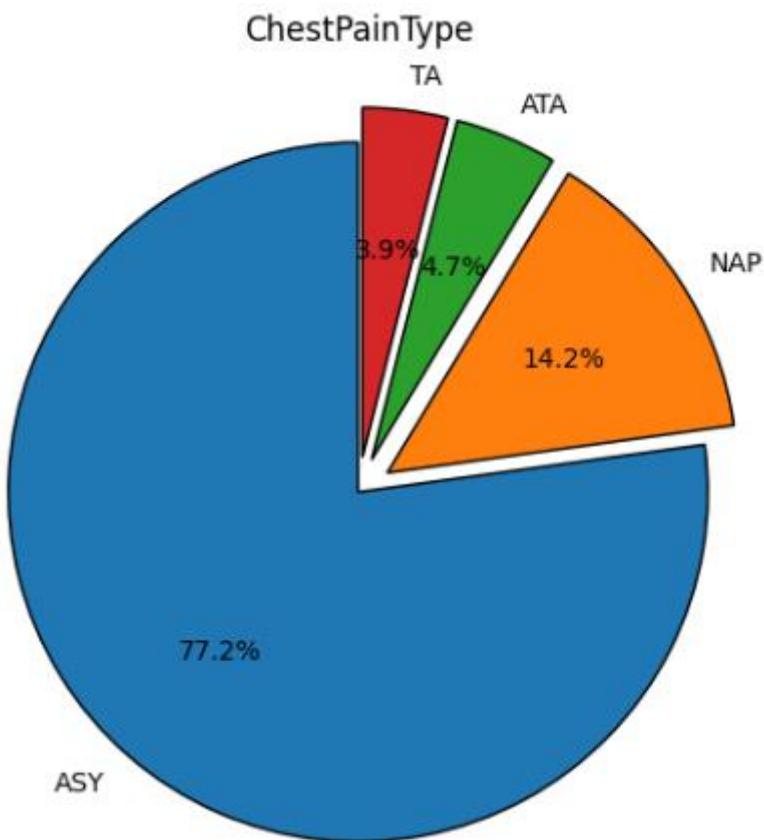
5.2.1) Sex



_Ta thấy có đến 90.2% bệnh nhân là nam sẽ mắc bệnh tim rất cao so với nữ chỉ có 9.8%.

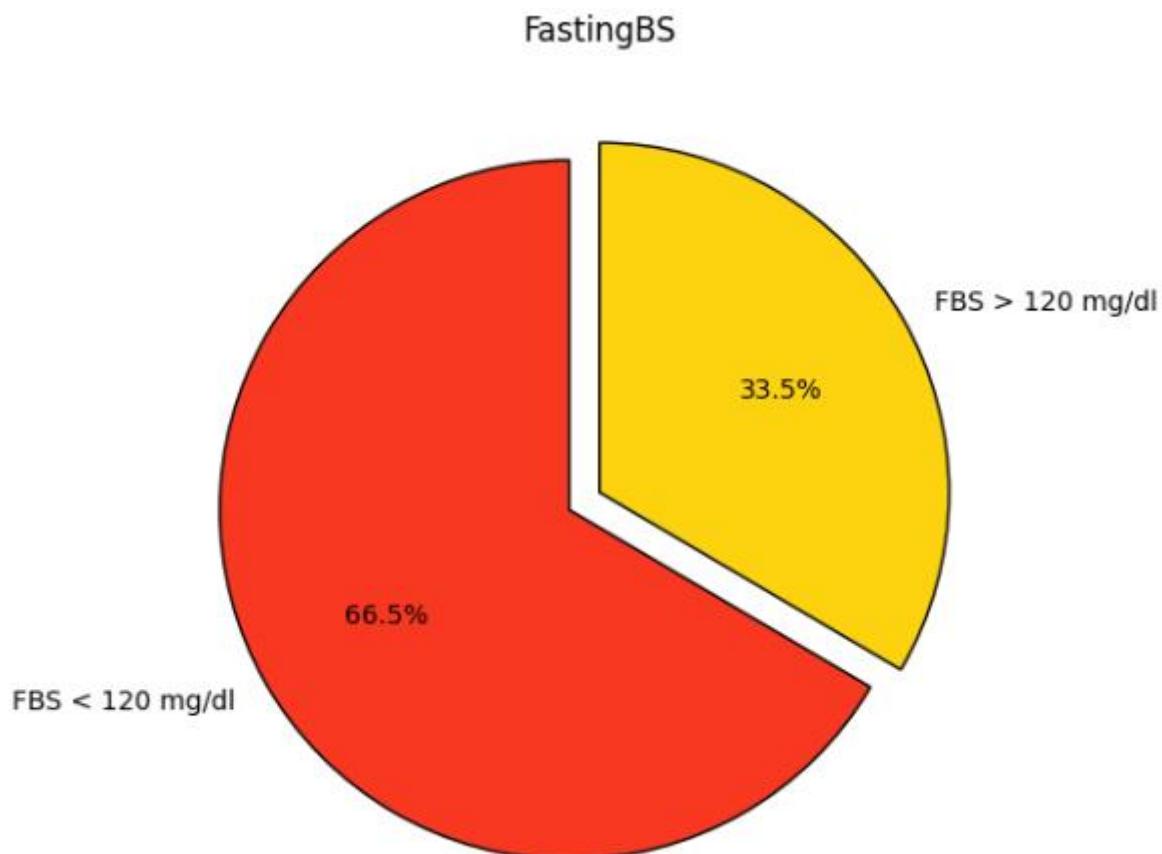
_Chứng tỏ bệnh nhân là nam sẽ có tỉ lệ mắc bệnh tim cao hơn nữ.

5.2.2) ChestPainType



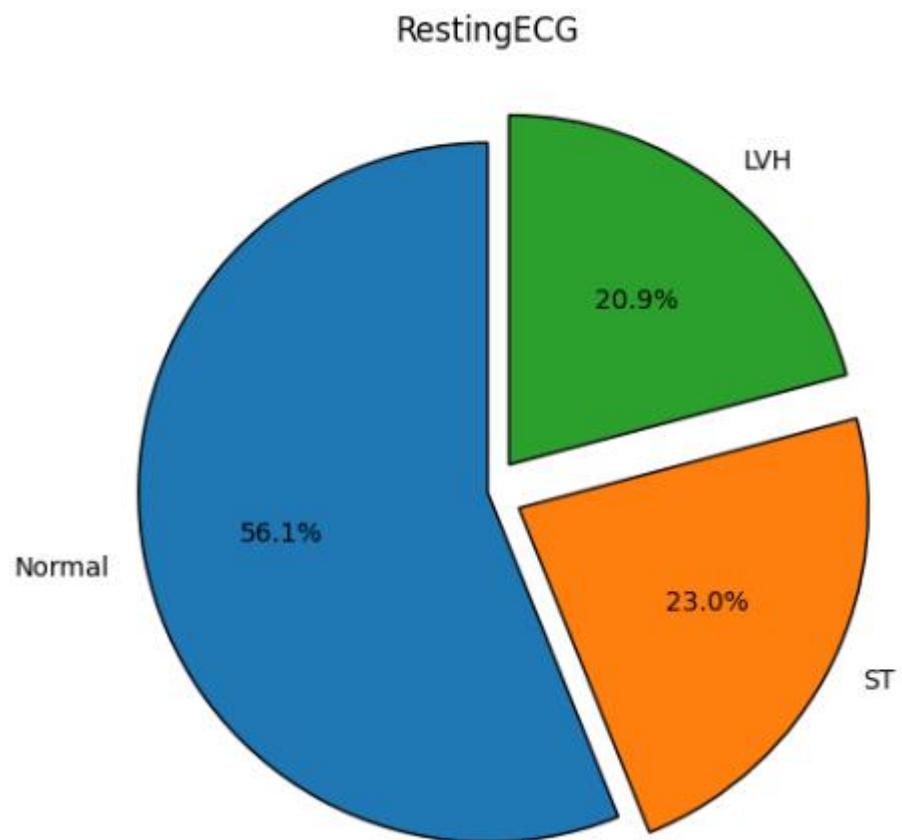
Ở đây ta thấy tỉ lệ mắc bệnh tim theo thứ của từng loại chứng đau ngực lần lượt là ASY (không có triệu chứng đau ngực), NAP (đau ngực không do đau thắt ngực), ATA (đau thắt ngực không điển hình), và cuối là TA (đau thắt ngực điển hình).

5.2.3) FastingBS



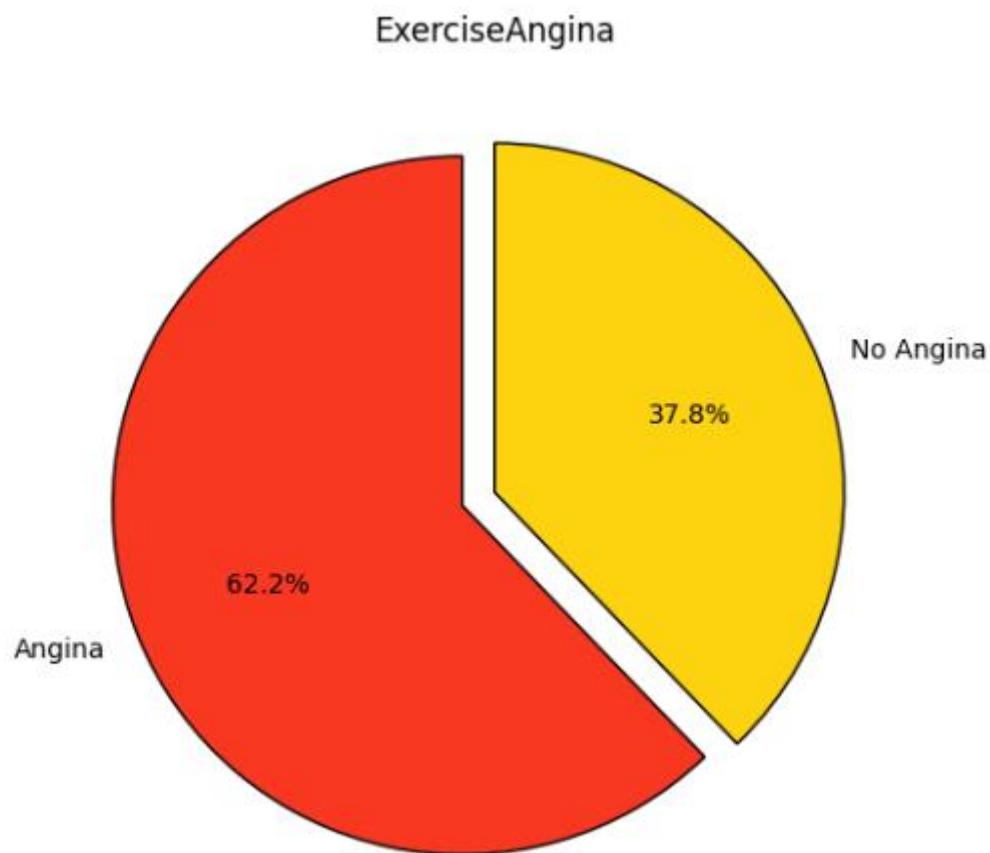
_Có khoảng 66.5% người có FastingBS <120mg/dl sẽ mắc bệnh tim cao hơn gần gấp đôi so với FastingBS >120 mg/dl (người có dấu hiệu của bệnh tiểu đường) sẽ mắc bệnh tim.

5.2.4) RestingECG



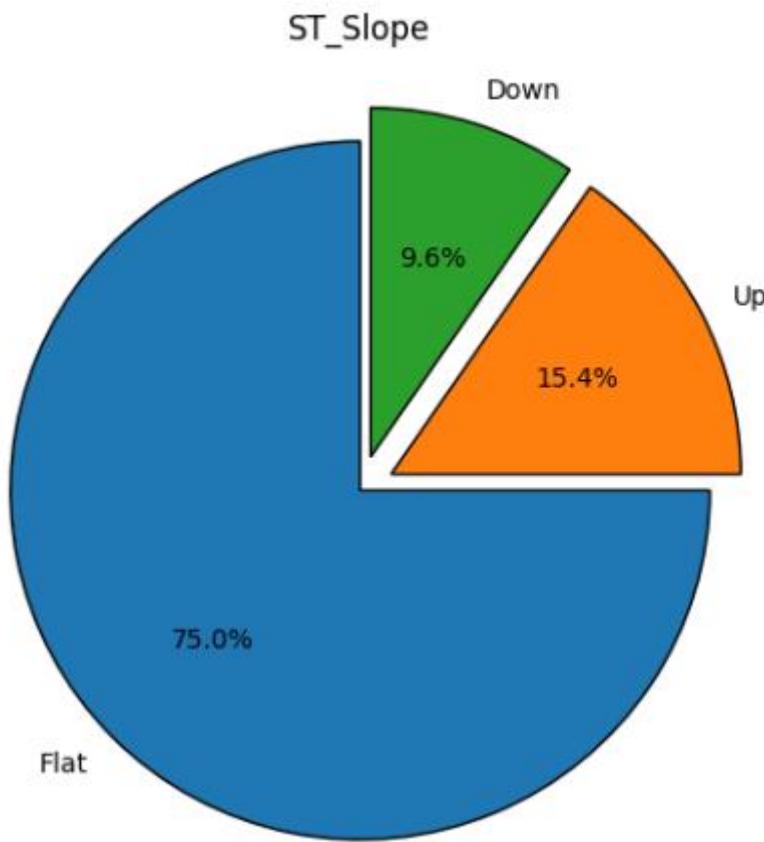
_ Có đến 56.1% người có RestingECG ở loại Normal là mắc bệnh tim trong khi ở loại ST và LVH thì thấp hơn lần lượt là 23.0% và 20.9%.

5.2.5) ExerciseAngina



_Biểu đồ cho thấy khi ExerciseAngina = 1 (đau thắt ngực do gắng sức) có đến 62.2% số bệnh nhân là mắc bệnh tim cao hơn khá nhiều so với ExerciseAngina=0 (không đau thắt ngực do gắng sức) chỉ có 37.8%.

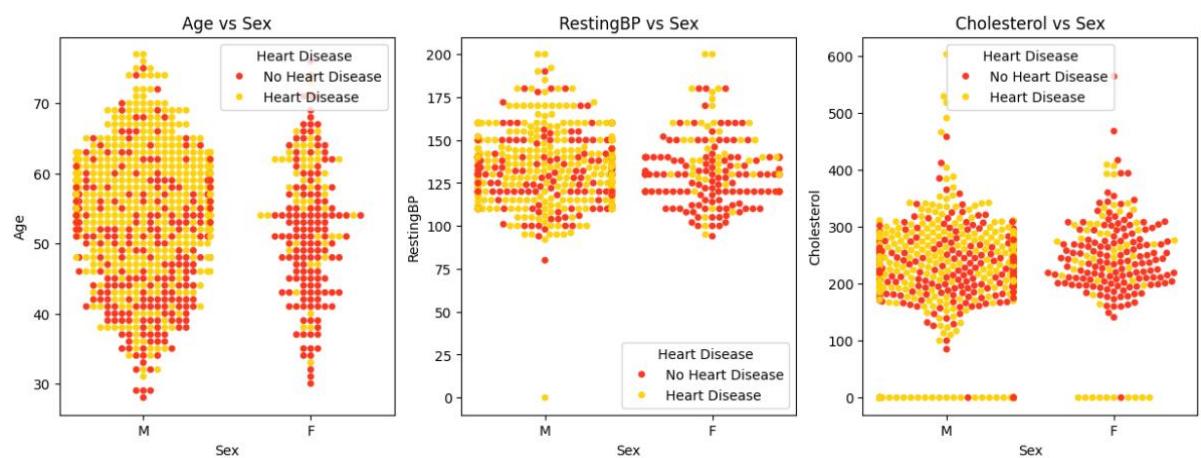
5.2.6) ST_Slope



_Biểu đồ cho thấy khi ST_Slope (độ dốc của đỉnh ST khi tập thể dục) là Flat thì xác suất người bệnh tim rất cao đến 75% trong khi Up và Down chỉ có 9.6% và 15.4%.

5.3) Tương quan giữa giá trị dạng loại và số so với giá trị mục tiêu

5.3.1) Sex



a) Age

_ Nam giới có số người bị bệnh tim tăng dần khi tuổi bắt từ khoảng 40 trở đi.

b) RestingBP

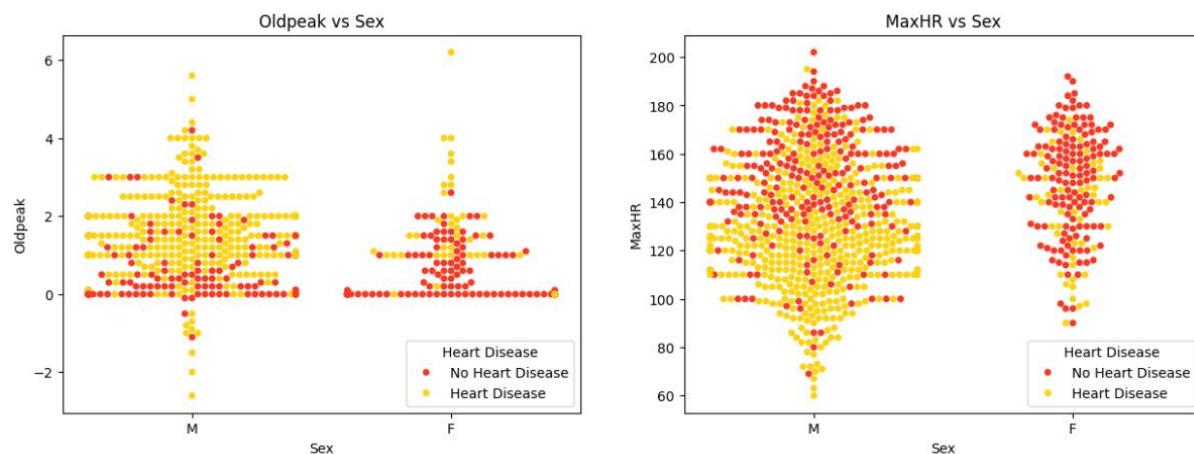
_ Ở Nam giới khi RestingBP từ 100 trở đi thì tỉ lệ người bệnh tim đã tăng khá nhiều, trong khi đó đối với F (Nữ) thì tỉ lệ đó không quá nhiều.

c) Cholesterol

_ Ở giá trị là 0 ở cả nam và nữ ta sẽ xem nó là những giá trị đặc biệt do có thể dữ liệu đã được đặt mặc định.

_ Ở những giá trị khác ta thấy ở nam giới khi Cholesterol từ khoảng 200 trở đi số lượng người mắc bệnh tim khá nhiều nhưng cũng có những chỗ có nhiều người không mắc bệnh tim nên khó có thể kết luận.

_ Đối với nữ chỉ khi Cholesterol phải gần 300 ta mới thấy được rõ ràng tỉ lệ người bị bệnh tim nhưng nó vẫn thấp hơn so với người không mắc bệnh.



d) OldPeak

_ Nhìn thấy rõ ràng khi giá trị OldPeak lớn hơn 0 thì số người bệnh tim của 2 cả nam lẫn nữ đều tăng đặc biệt ở nam thì số người bệnh tim tăng thấy rõ khi Oldpeak tăng dần.

e) MaxHR

_ Nam giới có số người mắc bệnh tim khá cao khi MaxHR nằm trong khoảng từ 60 đến dưới 140. Trong khi nữ giới thì số người bị bệnh tim bắt đầu từ 100 trở đi.

5.3.2) ChestPainType



a) Age

_ Khi ChestPainType là ATA thì xác suất người bị bệnh tim rất thấp ở phần lớn độ tuổi nhưng khi đến gần 60 tuổi trở đi thì mới xuất hiện nhiều người bệnh tim.

_ Đối với ChestPainType là NAP thì số người mắc bệnh tim bắt đầu xuất hiện sớm hơn so với ATA là từ 50 tuổi trở đi và khi gần 60 thì tỉ lệ này tăng rõ ràng hơn.

_ Ta thấy rằng khi ChestPainType là ASY thì đa số các độ tuổi đều xuất hiện rất nhiều người mắc bệnh tim, nhưng đặc biệt cao khi gần 40 tuổi trở đi.

_ Còn đối với ChestPainType là TA có số lượng rất ít, và các dữ liệu cũng rải rác khá đều nên có thể TA và Age không thể hiện rõ mối liên hệ đến tỉ lệ bệnh tim.

b) Resting BP

_ Dấu hiệu bị bệnh tim của ChestPainType là ATA bắt đầu xuất hiện từ khoảng RestingBP 125 trở lên, và có rất ít người bị bệnh tim khi RestingBP bé hơn 125.

_ Đối với ChestPainType là NAP thì tỉ lệ người mắc bệnh tim bắt đầu sớm hơn khi mà Resting BP khoảng 100 trở lên đã bắt đầu xuất hiện người bị bệnh tim.

_ ChestPainType là ASY Cũng tương tự như NAP nhưng tỉ lệ người bị bệnh tim lại chiếm rất nhiều so với người không bệnh tim và trải dài từ 100 đến 200.

_ Đối với TA ta chỉ thấy số người bị bệnh tim khi RestingBP tăng khi từ 125 trở đi, nhưng đến 150 thì lại giảm.

c) Cholesterol

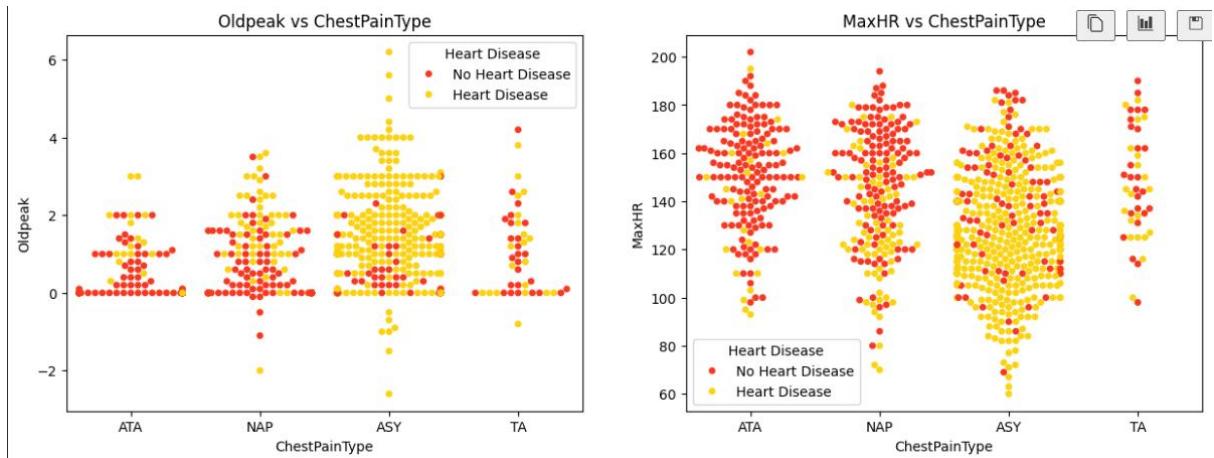
_ Ta vẫn sẽ bỏ qua giá trị 0 của Cholesterol và không xét đến nó.

_ Ta thấy khi ChestPainType là ATA thì bệnh nhân mắc bệnh tim chỉ xuất hiện ở giá trị lớn hơn 200 và gần với 300 (khoảng 280). Còn dưới giá trị đó thì rất hầu như không có bệnh nhân mắc bệnh tim.

_ Khi ChestPainType là NAP thì số lượng người bị bệnh tim nhiều hơn và xuất hiện sớm hơn khi Cholesterol khoảng 200 trở đi.

_ ChestPainType là ASY thì số người bị bệnh tim được trải đều và chiếm tỉ lệ rất cao và bắt đầu có dấu hiệu xuất hiện khi Cholesterol là 100 trở đi.

_ Đối với ChestPainType là TA số người bị bệnh tim xuất hiện khi Cholesterol khoảng 150 trở đi nhưng tỉ lệ đặc biệt tăng cao khi gần với 300.



d) OldPeak

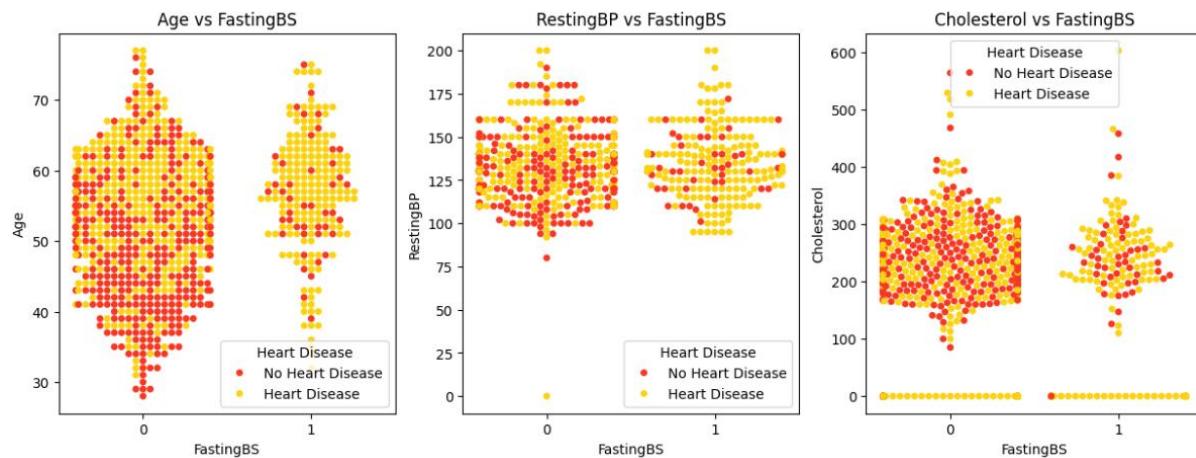
_ Ở ChestPainType là ATA thì chỉ khi Oldpeak lớn hơn 0 gần với 1 thì mới có xuất hiện bệnh nhân mắc bệnh tim.

_ Ở ChestPainType là NAP thì Oldpeak cũng tương tự như ATA nhưng khi từ 1 trở đi thì số lượng tăng người bị bệnh tim tăng thấy rõ.

_ Ở ChestPainType là ASY thì khi Oldpeak= 0 thì đã xuất hiện người có triệu chứng bệnh tim và số lượng cũng tăng dần theo giá trị của Oldpeak.

_ Ở ChestPainType là TA cũng tương tự như ở ATA.

5.3.3) FastingBS



a) Age

_ Đối với FastingBS =0 thì ta thấy xác suất mắc bệnh tim tăng dần theo độ tuổi.

_ Đối với FastingBS =1 thì gần như ở mọi lứa tuổi thì tất cả đều mắc bệnh tim trừ một vài trường hợp.

b) RestingBP

_ FastingBS =0 thì bắt đầu xuất hiện người mắc bệnh tim khi RestingBP > 100 gần với 125.

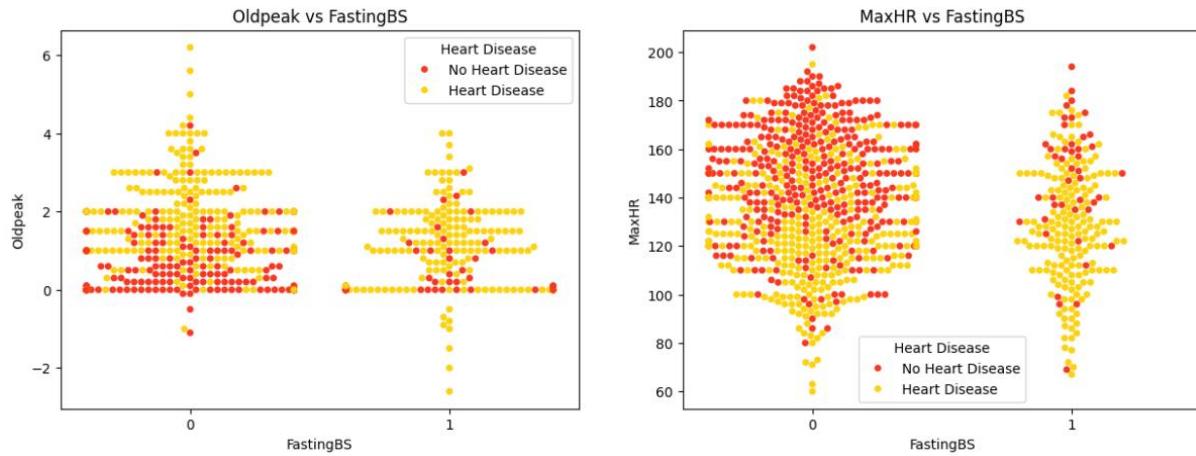
_ Fasting BS=1 thì gần như là sẽ mắc bệnh tim nhưng cũng có một vài trường hợp ngoại lệ.

c) Cholesterol

_ Vẫn là bỏ qua giá trị 0 để khảo sát.

_ FastingBS =0 thì khi Cholesterol tầm 200 trở đi thì tỉ lệ mắc bệnh tim có thể nói là xấp xỉ với người không mắc bệnh tim. Nhưng khi Cholesterol tăng đến 300 thì tỉ lệ người mắc bệnh tim lại cao hơn.

_ FastingBS =1 thì chỉ có Cholesterol < 200 thì xác suất giữa bệnh tim và không mắc bệnh xấp xỉ bằng nhau nhưng khi cao hơn thì hầu như mọi người đều mắc bệnh tim.



d) Oldpeak

_ FastingBS =0 thì khi Oldpeak từ 1 trở đi số người bị bệnh tim tăng rất nhiều.

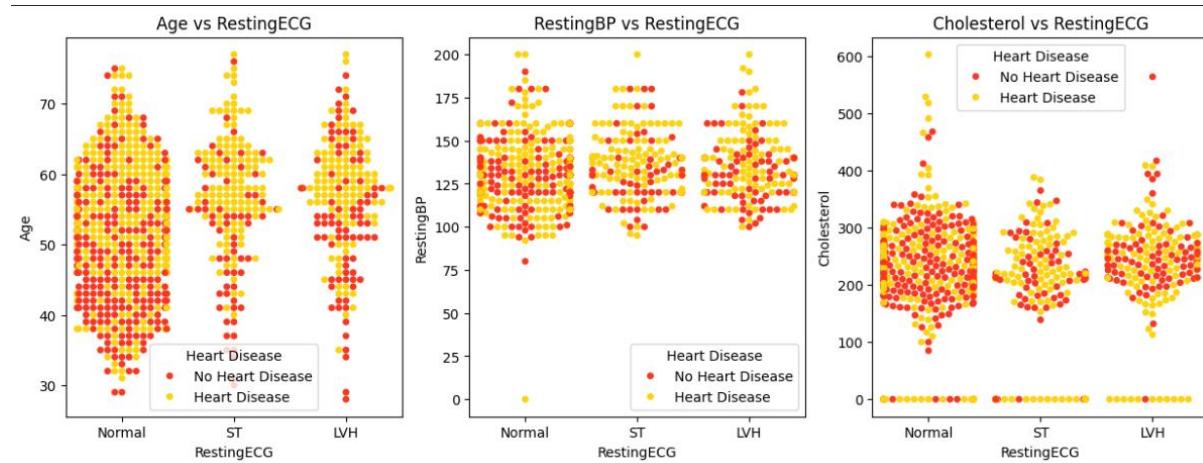
_ FastingBS =1 thì xác suất người bị bệnh tim rất cao trả dài gần như toàn bộ.

e) MaxHR

_ FastingBS =0 thì lúc MaxHR <140 số người bị bệnh tim rất nhiều nhưng trên 140 thì tỉ lệ giảm dần.

_ FastingBS =1 thì xác suất người bị bệnh tim rất cao trả dài gần như toàn bộ

5.3.4) RestingECG



a) Age

_ Ở mức Normal của RestingECG thì tuổi phải trên 40 mới có tỉ lệ mắc bệnh tim cao và càng tăng khi tuổi càng tăng

_ Ở mức ST thì tuổi phải trên 50 thì tỉ lệ mắc bệnh tim mới cao và cũng tăng dần như mức Normal.

_ Ở mức LVH thì phải hơn gần 50 tuổi thì tỉ lệ mắc bệnh tim sẽ cao hơn không mắc bệnh tim và tăng dần khi về già.

b) RestingBP

_ Khi RestingECG là Normal thì từ RestingBP gần 100 thì mới xuất hiện người mắc bệnh tim tuy cũng tăng tỉ lệ người mắc bệnh nhưng dữ liệu vẫn còn sự chòng chéo khá nhiều.

_ RestingECG là ST thì xác suất người bị bệnh tim lại tăng khá đều từ 100 trở đi.

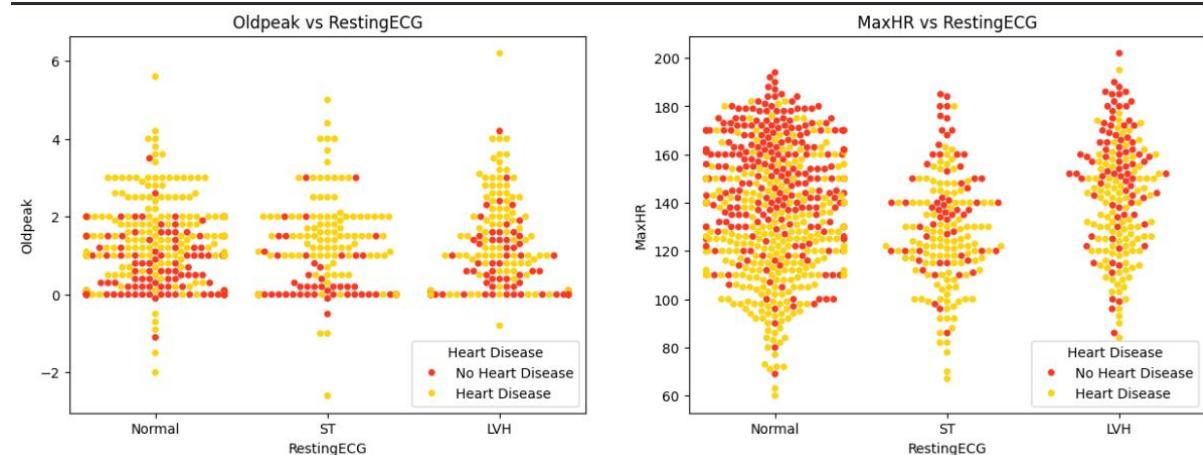
_ RestingECG là LVH xác suất người bị bệnh tim xuất hiện trễ hơn với RestingBP khoảng gần 125 và tăng khi RestingBP tăng.

c) Cholesterol

_ RestingECG là Normal thì chỉ khi Cholesterol trên 200 thì tỉ lệ người mắc bệnh tim mới tăng nhưng cũng còn xấp xỉ so với người không mắc bệnh tim rõ ràng nhất là Cholesterol =300.

_ RestingECG là ST thì kể từ lúc Cholesterol là 200 trở lên là tỉ lệ người mắc bệnh tim đã cao hơn so với tỉ lệ người không mắc bệnh tim.

_ RestingECG là LVH khi Cholesterol gần 100 trở lên thì tỉ lệ người mắc bệnh tim khá cao cho đến 200 thì nó giảm lại và tăng lại khi gần 300.



d) Oldpeak

_ Khi RestingECG là Normal thì Oldpeak là 1 trở đi tỉ lệ người mắc bệnh tim là vô cùng cao và khi trên 2 trở đi thì tỉ lệ là 100% người mắc bệnh tim.

_ RestingECG là ST thì cũng khá giống với Normal nhưng tỉ lệ người mắc bệnh tim thấp hơn một ít cho với Normal.

_ Còn với RestingECG là LVH thì tỉ lệ giữa người mắc bệnh tính từ 1 vẫn còn khá cao nhưng thấp hơn so với Normal và ST.

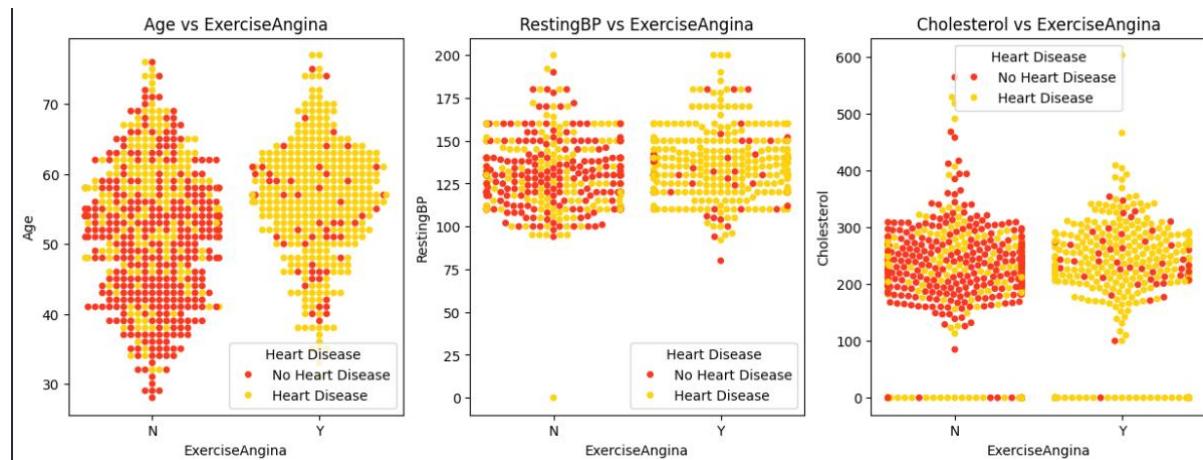
e) MaxHR

_ Khi RestingECG là Normal thì khi giá trị MaxHR từ 60 cho đến gần 140 tỉ lệ người mắc bệnh tim là rất cao. Nhưng từ 140 trở đi thì nó sẽ giảm đi rõ rệt.

_ Khi RestingECG là ST thì từ lúc MaxHR từ 60 trở đi xác suất bệnh tim vẫn rất cao nhưng khi gần với 140 thì giá trị giảm đi nhưng vẫn cao hơn so với người không mắc bệnh tim.

_ RestingECG là LVH thì khi MaxHR lớn hơn 100 thì tỉ lệ người mắc bệnh tim mới cao hơn người không mắc bệnh và càng về sau nó cũng sẽ tăng.

5.3.5) ExerciseAngina



a) Age

_ Với ExerciseAngina là N thì chỉ khi tuổi lớn 50 thì tỉ lệ người bị bệnh tim mới xuất hiện rõ rệt và nó tăng dần theo độ tuổi.

_ Với ExerciseAngina là Y thì tỉ lệ mắc bệnh tim là vô cùng cao và trải dài khắp tất cả độ tuổi.

b) RestingBP

_ Với ExerciseAngina là N thì dữ liệu cho thấy tỉ lệ người mắc bệnh và không mắc bệnh khá là cân bằng và cũng không có dấu hiệu tăng hay giảm theo độ tuổi.

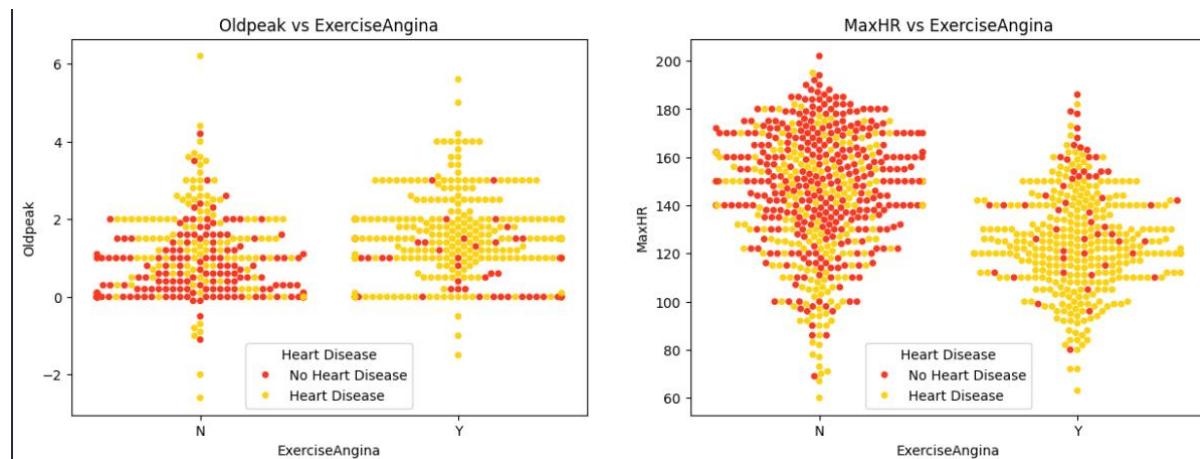
_ Với ExerciseAngina là Y thì tỉ lệ mắc bệnh tim là vô cùng cao và trải dài khắp tất cả giá trị.

c) Cholesterol

_Bỏ qua giá trị 0 của Cholesterol.

_Nếu ExerciseAngina là N thì chỉ khi Cholesterol gần với khoảng 300 thì tỉ lệ người bệnh tim mới xấp xỉ với không bệnh tim.

_Nếu ExerciseAngina là Y thì hầu hết tất cả giá trị trong Cholesterol đều có xác suất người bệnh tim rất cao.



d) Oldpeak

_Khi ExerciseAngina là N khi Oldpeak từ 1 trở lên thì xác suất người bị bệnh tim tăng dần.

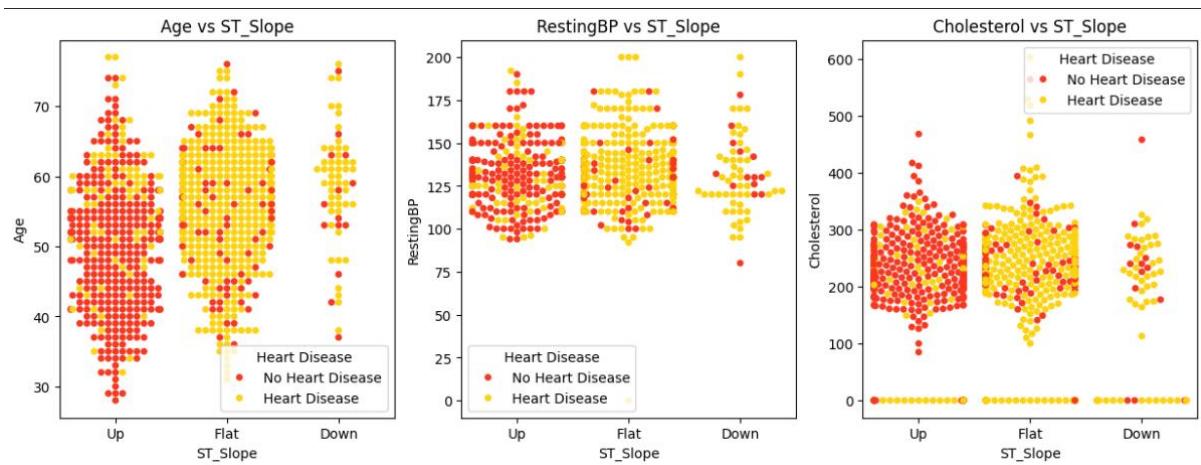
_Khi ExerciseAngina là Y từ chỉ cần Oldpeak từ 0 trở đi thì xác suất bị bệnh tim là vô cùng cao.

e) MaxHR

_Khi ExerciseAngina là N thì khi MaxHR tăng thì xác suất người mắc bệnh tim giảm dần.

_Khi ExerciseAngina là Y thì tỉ lệ mắc bệnh tim rất cao bất kể MaxHR là bao nhiêu.

5.3.6) ST_Slope



a) Age

_Đối với ST_Slope là Up thì xác suất người bị bệnh tim là rất thấp cho dù ở bất cứ độ tuổi nào.

_Đối với ST_Slope là Flat thì ngược lại cho dù bất kỳ độ tuổi nào thì xác suất người bị bệnh tim là vô cùng cao.

_Còn ST_Slope là Down cũng tương tự như Flat.

b) RestingBP

_ST_Slope là Up thì xác suất người bị bệnh tim là khá thấp so với người không mắc bệnh tim.

_ST_Slope là Flat cho dù RestingBP có bất kỳ giá trị nào thì xác suất người bị bệnh tim là vô cùng cao.

_ST_Slope là Down cũng tương tự như Flat.

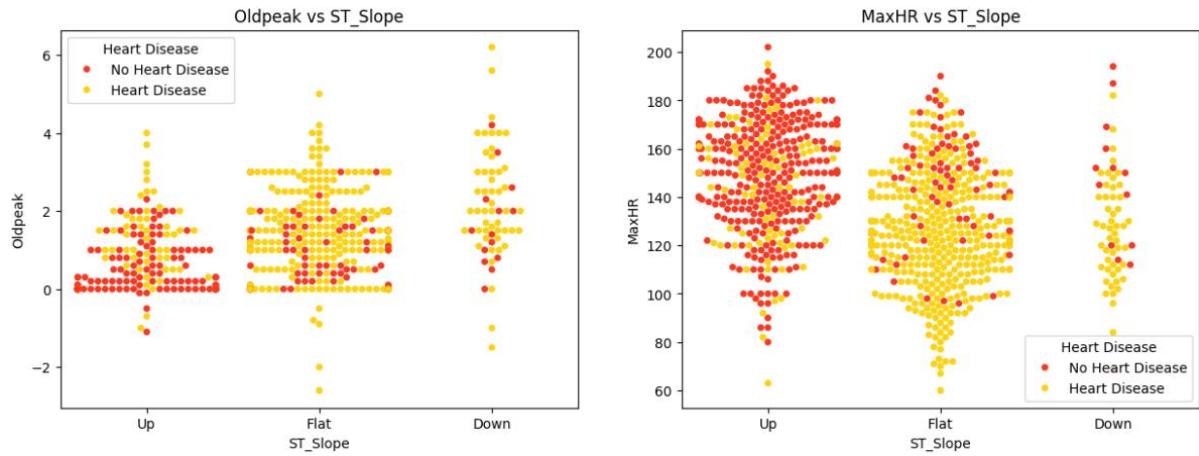
c) Cholesterol

_ Bỏ qua không xét giá trị 0.

_ ST_Slope là Up thì xác suất người bị bệnh tim là vô cùng thấp so với người không mắc bệnh tim.

_ Còn ST_Slope là Flat thì tỉ lệ người bị bệnh tim rất cao, tuy tỉ lệ người không mắc bệnh tim cũng ở khoảng từ 200 đến 300 không bằng tỉ lệ người mắc bệnh nhưng nơi đây lại có số người không mắc bệnh cao hơn so với các nơi còn lại

_ ST_Slope là Down xác suất bị bệnh tim là vô cùng cao.



d) Oldpeak

_ Với ST_Slope là Up thì tỉ lệ người mắc bệnh tim tăng dần khi Oldpeak tăng dần từ 0 nhưng chỉ khi Oldpeak lớn hơn 1 tỉ lệ người mắc bệnh tim mới cao hơn so với người không mắc bệnh tim.

_ Với ST_Slope là Flat thì tỉ lệ người mắc bệnh tim là rất cao.

_ Với ST_Slope là Down tỉ lệ người mắc bệnh tim cũng rất cao tương tự như Flat.

e) MaxHR

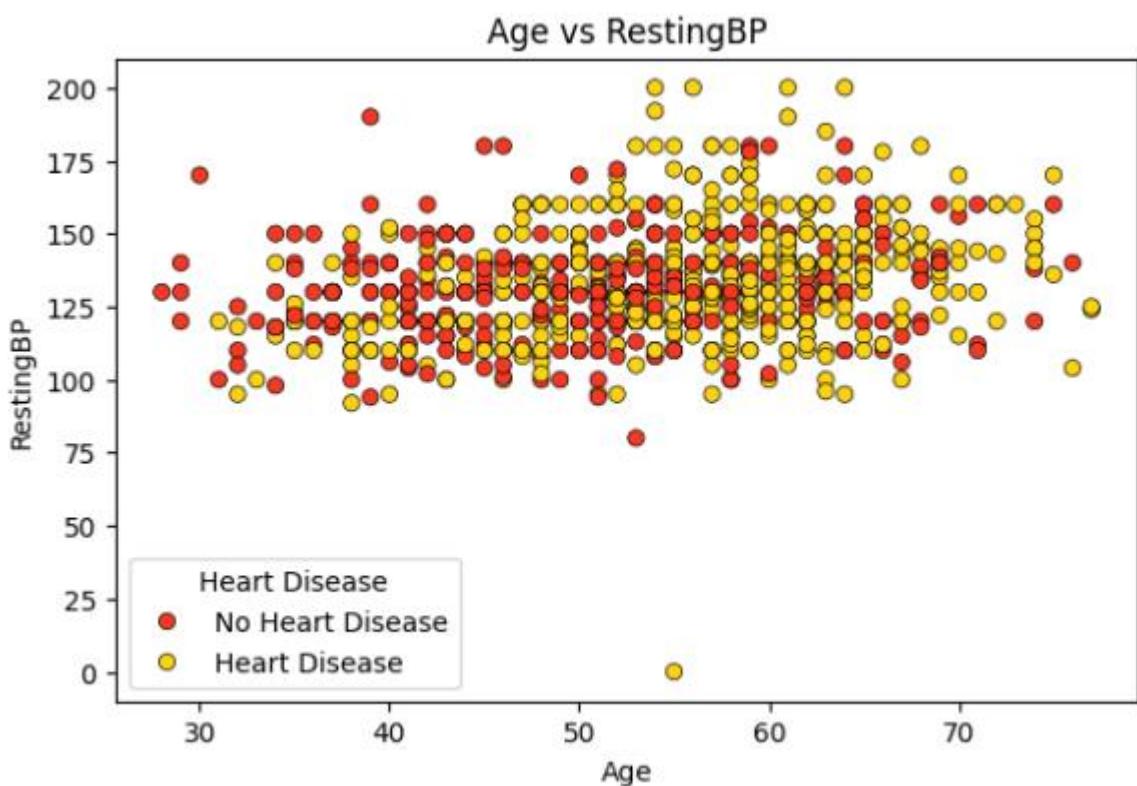
_ Với ST_Slope là Up thì tỉ lệ người không mắc bệnh tim khá cao. Nhưng với MaxHR từ 80 đến gần 120 thì tỉ lệ người mắc bệnh tim tăng nhưng vẫn thấp hơn so với không mắc bệnh tim

_ Với ST_Slope là Flat thì tỉ lệ người mắc bệnh tim là rất cao.

_ Với ST_Slope là Down tỉ lệ người mắc bệnh tim cũng rất cao tương tự như Flat.

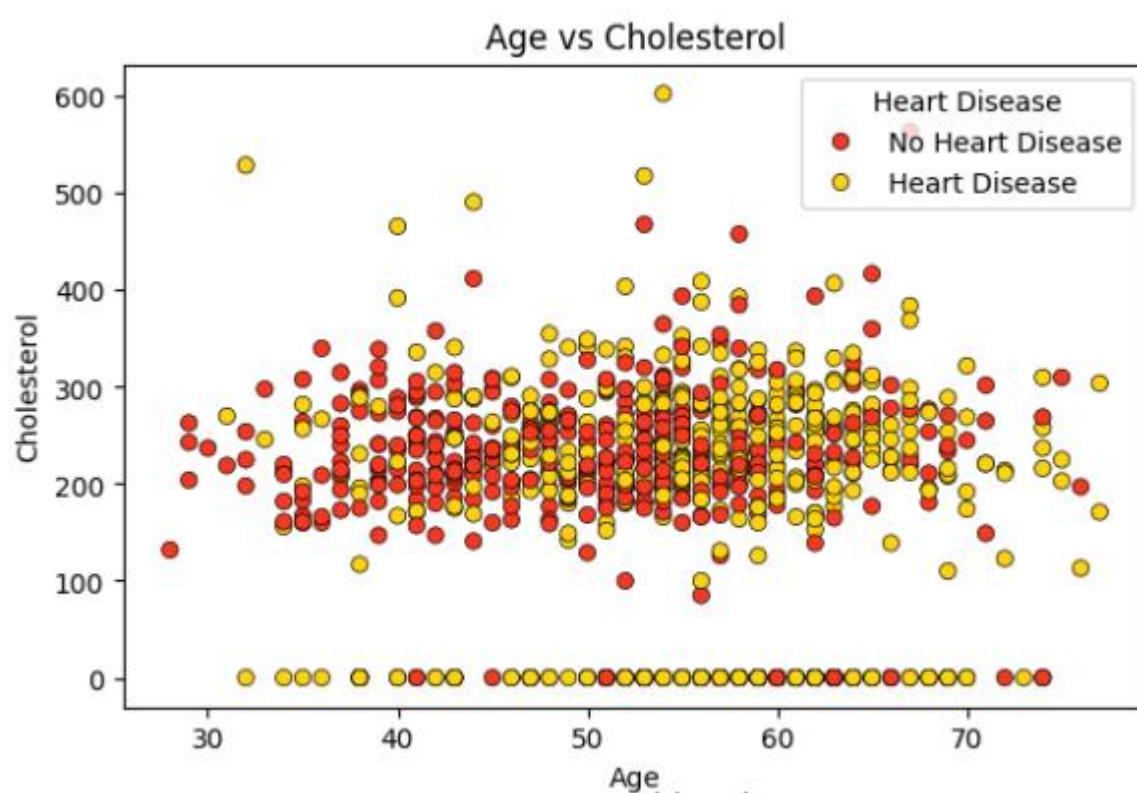
5.4) Tương quan giữa các giá trị dạng số với nhau

5.4.1) Age và RestingBP



_Tỉ lệ người mắc bệnh tim sẽ tăng khi có Age và RestingBP ngày càng cao.

5.4.2) Age và Cholesterol



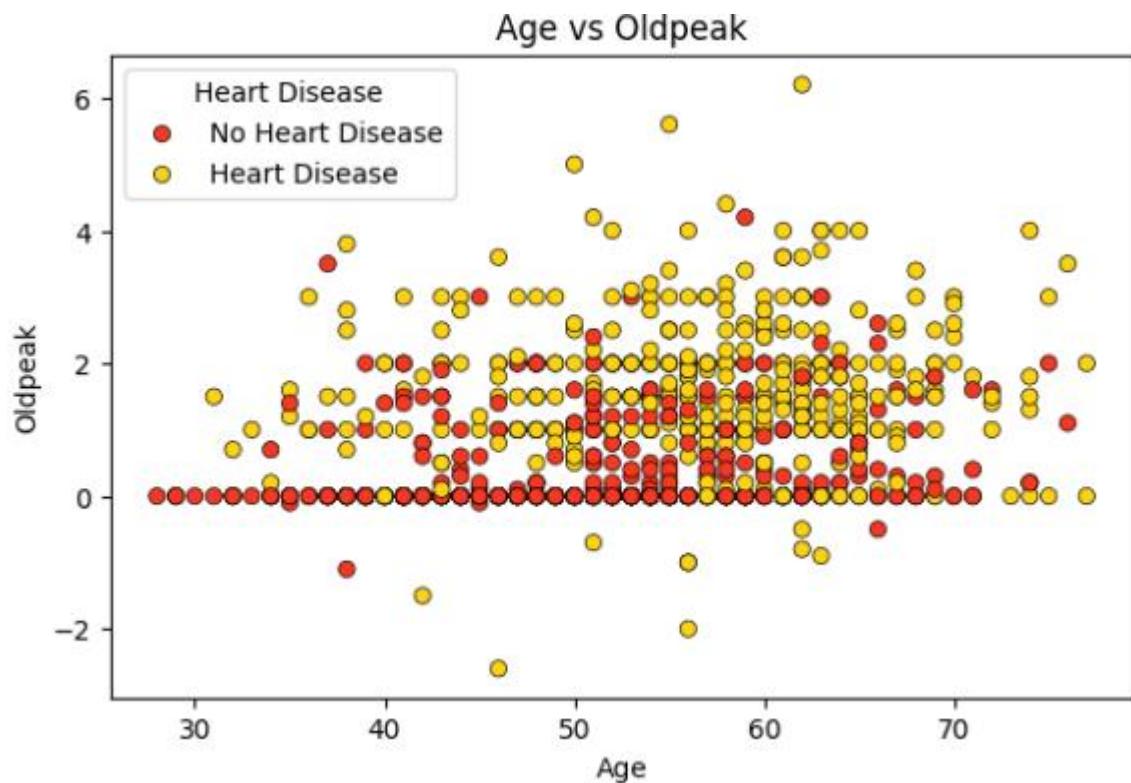
_ Tỉ lệ người bị bệnh tim sẽ tăng khi độ tuổi càng cao và Cholesterol nằm trong khoảng từ 200 trở lên.

5.4.3) Age và MaxHR



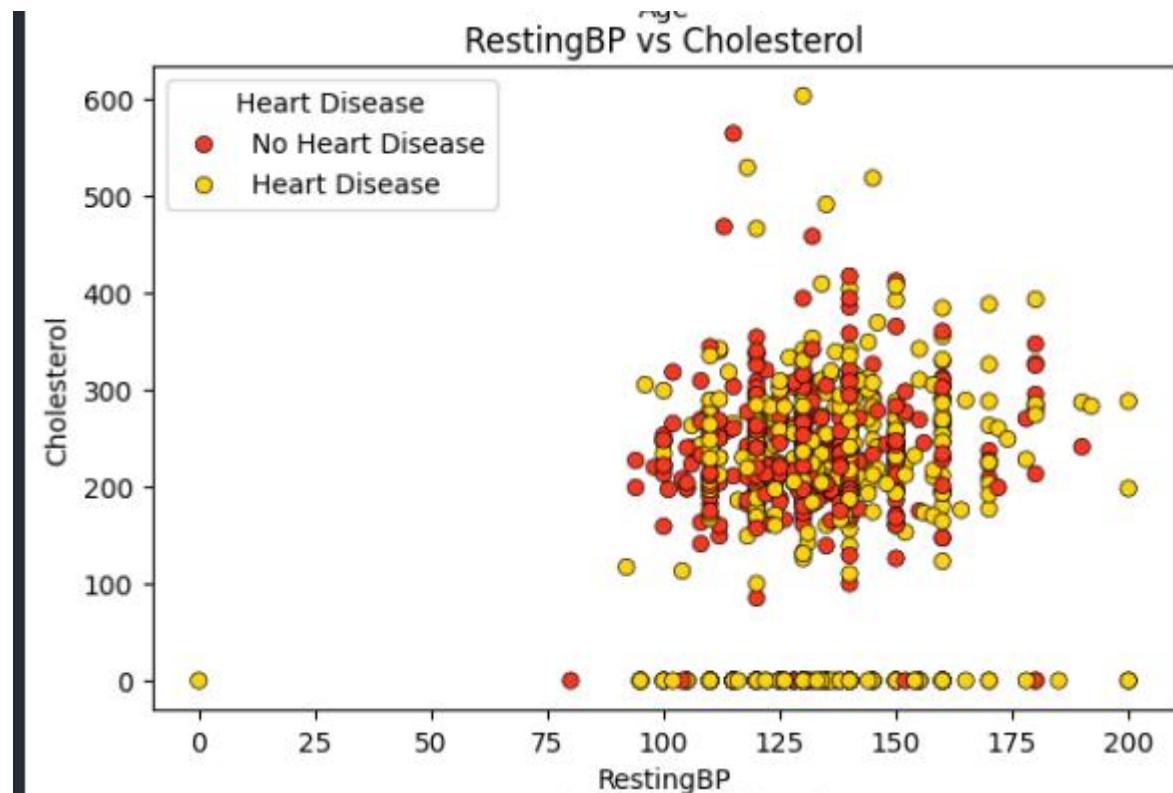
_ Tỉ lệ bị bệnh tim sẽ tăng khi độ tuổi càng tăng và MaxHR nằm trong khoảng từ 100 đến 140.

5.4.4) Age và Oldpeak



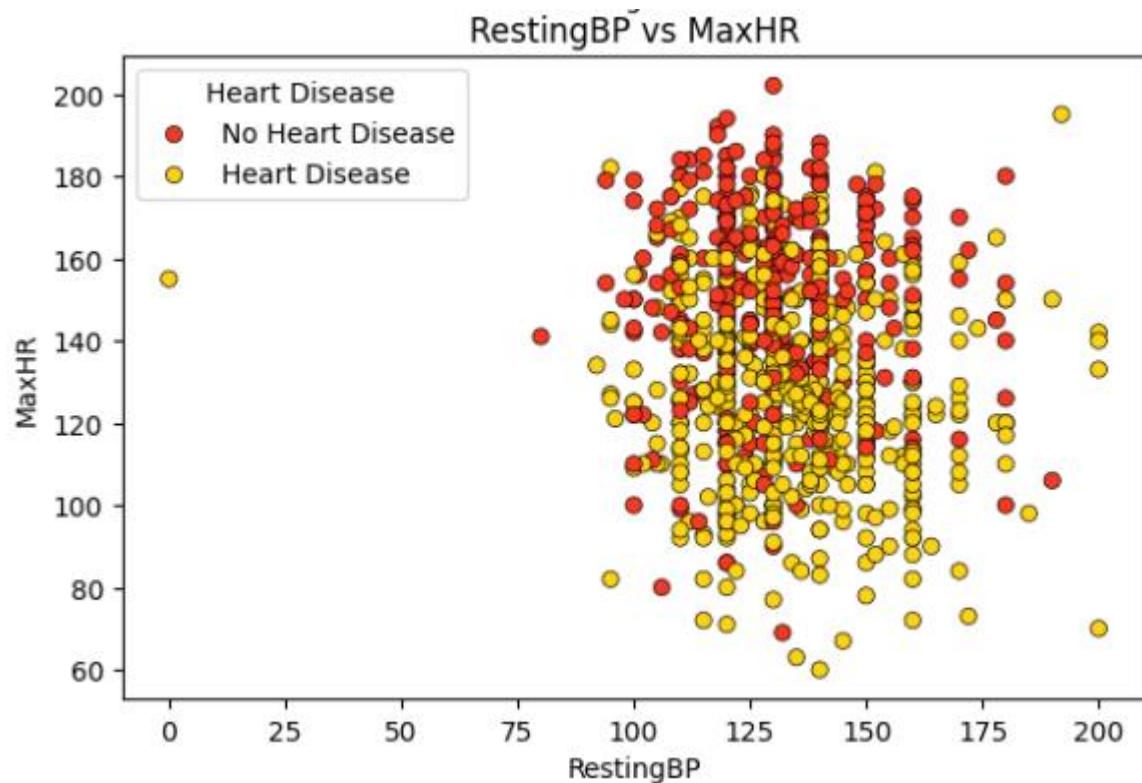
_Tỉ lệ người mắc bệnh tim tăng khi độ tuổi tăng cao và giá trị Oldpeak >0.

5.4.5) RestingBP và Cholesterol



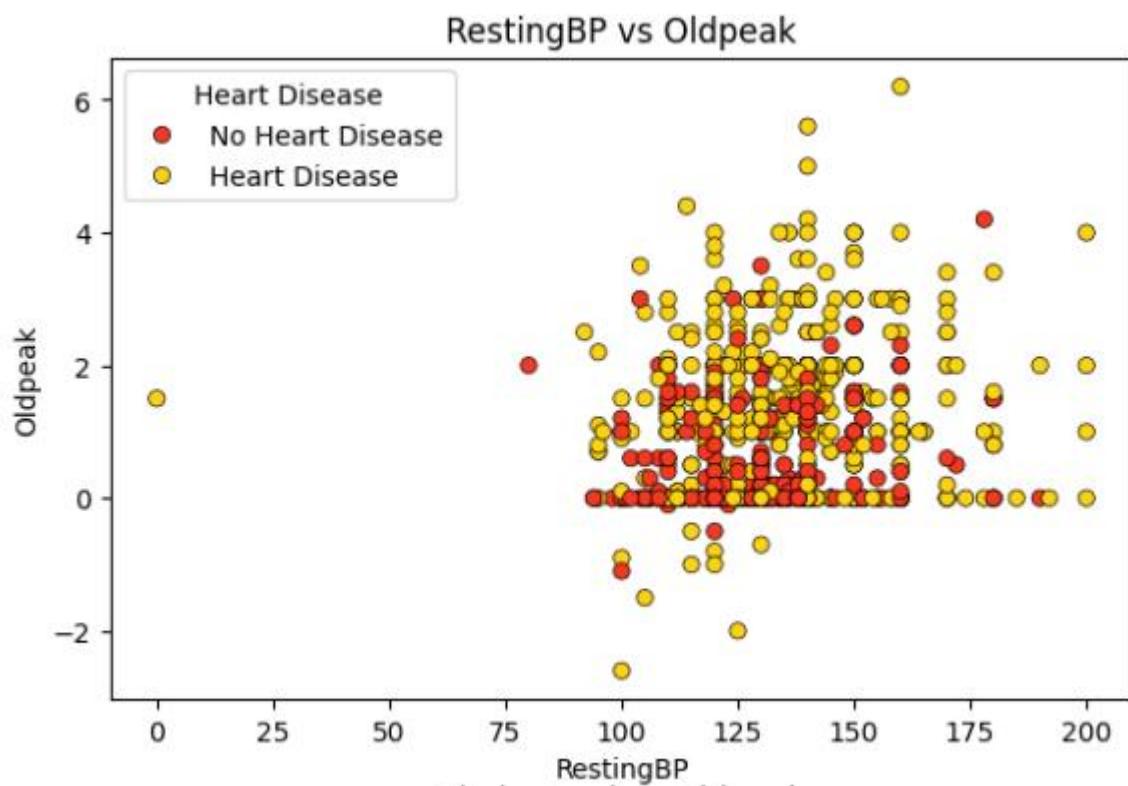
_ Ta thấy tỉ lệ người mắc bệnh tim bắt đầu tăng khi mà RestingBP > 125 và Cholesterol > 200.

5.4.6) RestingBP và MaxHR



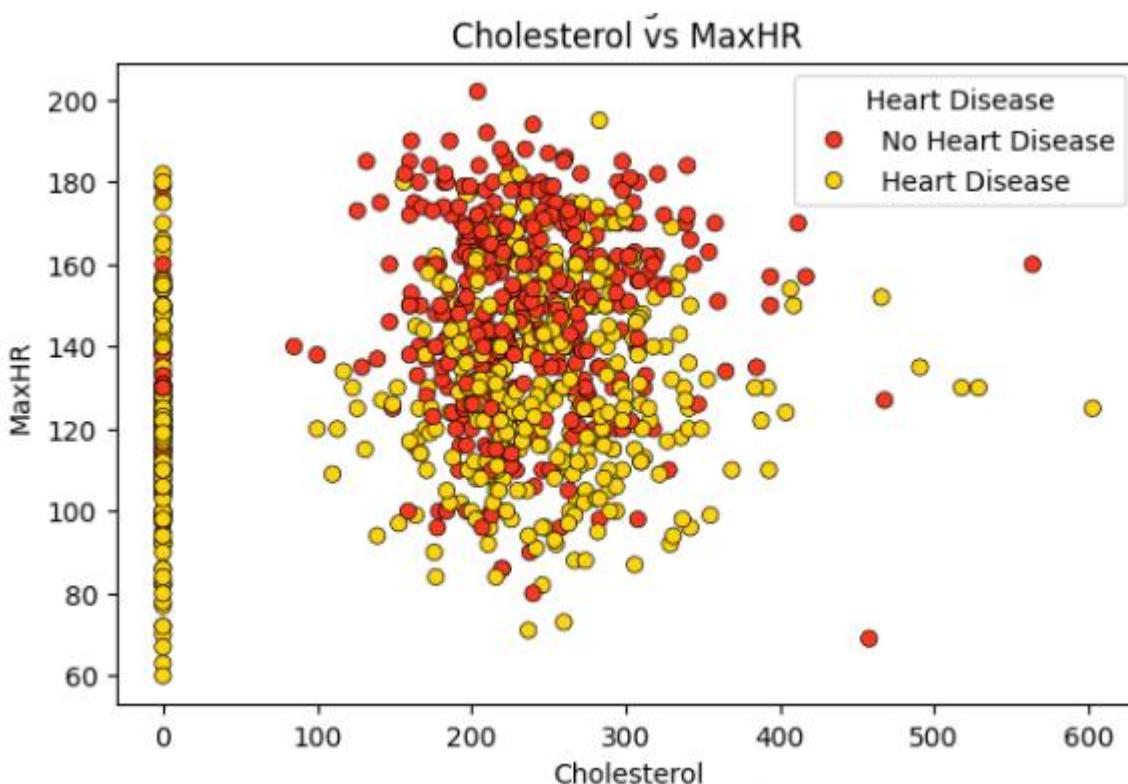
_ Tỉ lệ người mắc bệnh tim sẽ tăng khi mà RestingBP > 125 và $100 < \text{MaxHR} < 140$.

5.4.7) RestingBP và Oldpeak



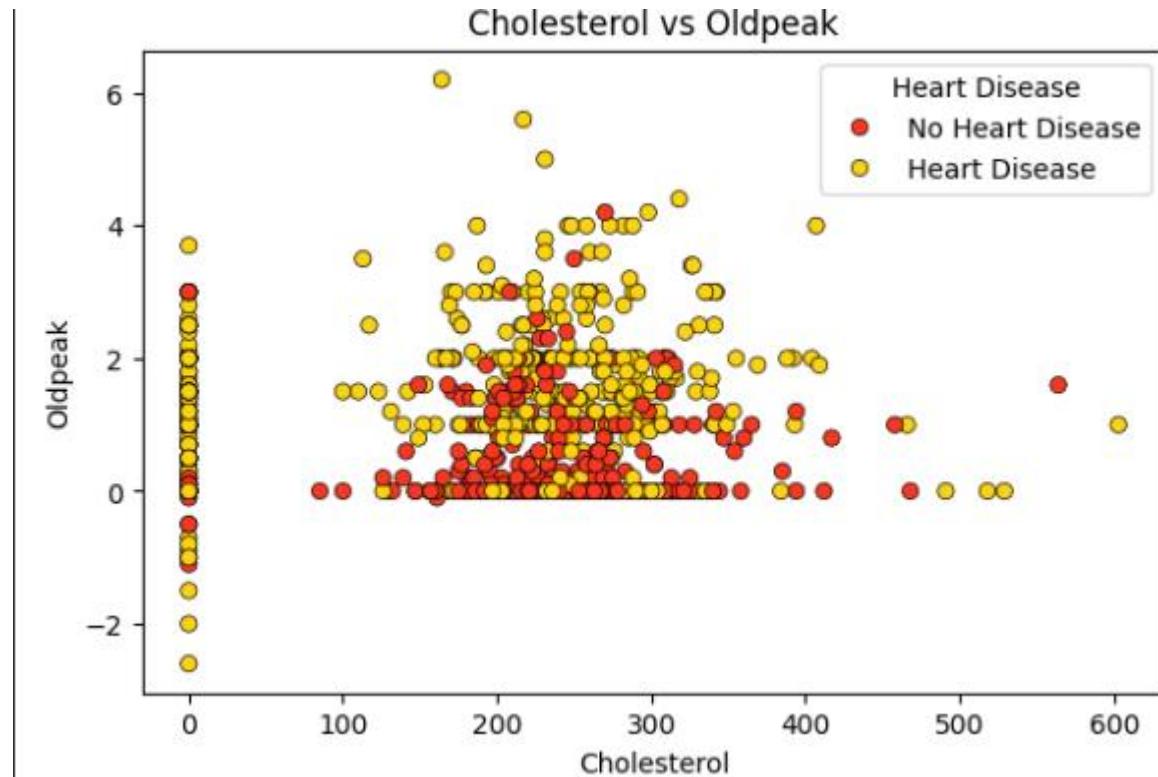
_Tỉ lệ người mắc bệnh tim tăng cao khi RestingBP >100 và Oldpeak ≥ 1 trở đi.

5.4.8) Cholesterol và MaxHR



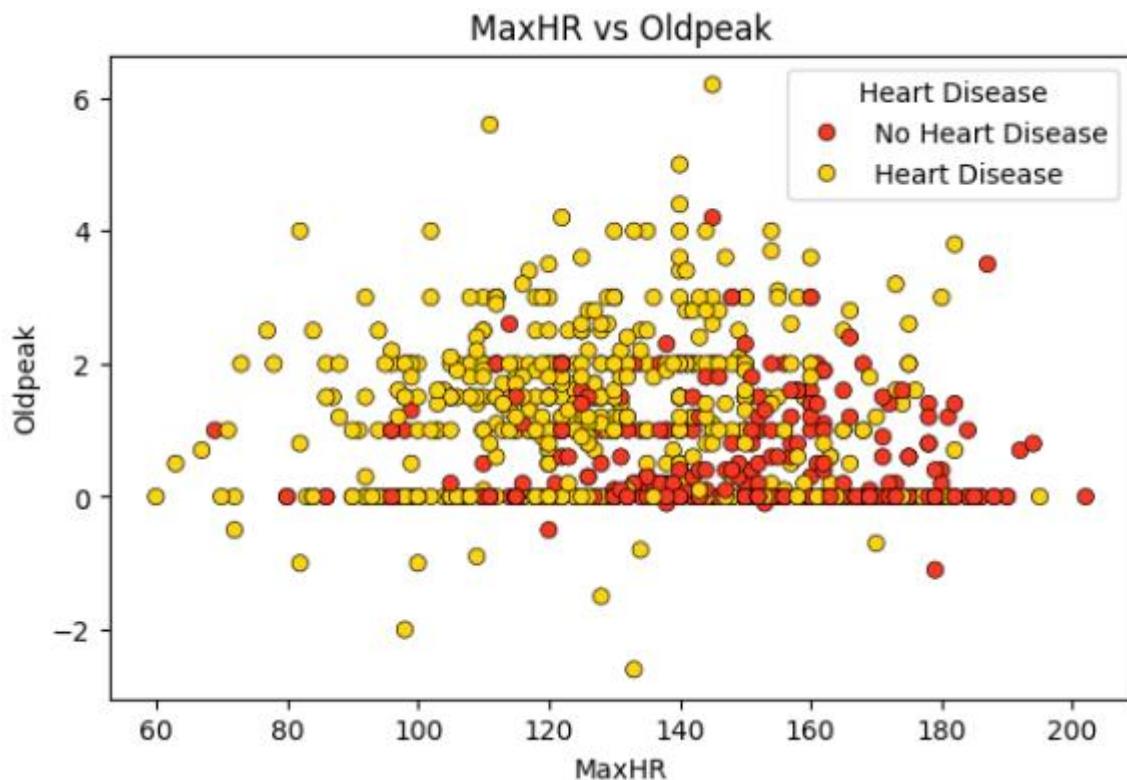
_ Khi ta bỏ qua giá trị 0 của Cholesterol thì ta thấy rằng số người bị bệnh tim tập trung nhiều ở khoảng Cholesterol > 200 và $100 < \text{MaxHR} < 140$.

5.4.9) Cholesterol và Oldpeak



_ Bỏ qua giá trị 0 của Cholesterol ta thấy tỉ lệ người mắc bệnh tim tăng cao khi $\text{Oldpeak} \geq 1$ và $\text{Cholesterol} > 200$.

5.4.10) MaxHR và Oldpeak



_Khi Oldpeak ≥ 1 ta và MaxHR ≥ 100 thì ta thấy tỉ lệ người mắc bệnh tim tăng rất cao.

6) Ma trận tương quan

_Trước khi vẽ vẽ đồ thị về ma trận tương quan ta sẽ chuyển dữ liệu của các giá trị dạng loại sang số rồi ta sẽ vẽ ma trận tương quan giữa các cột trong dữ liệu.

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df1 = df.copy(deep=True)

df1['Sex'] = le.fit_transform(df1['Sex'])
df1['ChestPainType'] = le.fit_transform(df1['ChestPainType'])
df1['RestingECG'] = le.fit_transform(df1['RestingECG'])
df1['ExerciseAngina'] = le.fit_transform(df1['ExerciseAngina'])
df1['ST_Slope'] = le.fit_transform(df1['ST_Slope'])
```

_Sau khi chuyển đổi ta sẽ có

Sex: [M: 1, F: 0]

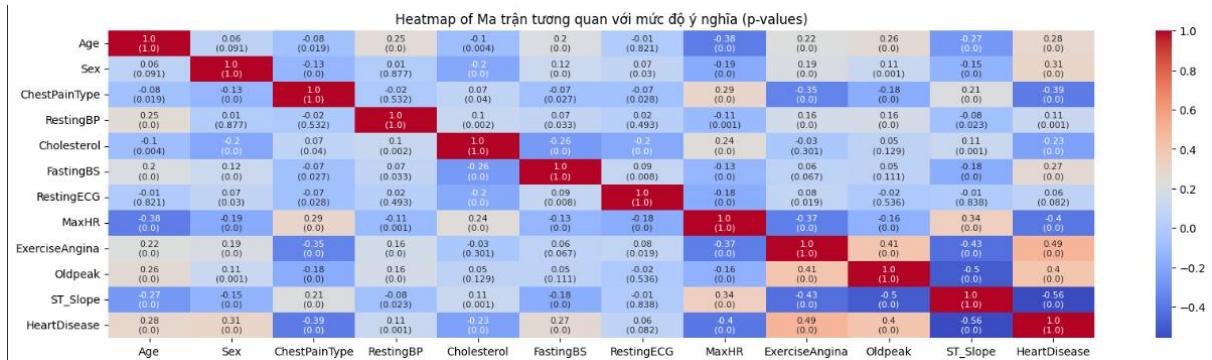
ChestPaintType: [ASY: 0, ATA: 1, NAP: 2, TA: 3]

FastingBS: [0 ,1]

RestingECG: [LVH: 0, Normal: 1, ST: 2]

ExerciseAngina: [No: 0, Yes: 1]

ST_Slope: [Down: 0, Flat: 1, Up: 2]



Để tính độ tương quan giữa các cột với nhau sử dụng công thức tính hệ số tương quan Pearson:

$$\rho_{xy} = \frac{Cov(x,y)}{\sigma_x \sigma_y}$$

Trong đó:

– ρ_{xy} : gọi là hệ số tương quan Pearson

– $Cov(x, y)$: gọi là hiệp phương sai của x, y

– σ_x, σ_y : là độ lệch chuẩn của x và y

Với $Cov(x, y)$ được tính theo công thức:

$$Cov(x, y) = \frac{1}{n-1} \sum_{i=1}^{n-1} (x_i - \bar{x})(y_i - \bar{y})$$

Trong đó:

– x_i, y_i là giá trị thứ i trong lúc tính toán.

– \bar{x}, \bar{y} lần lượt là giá trị trung bình của biến x và y

_ Dựa vào ma trận tương quan ta thấy rằng p-value của RestingECG và HeartDisease lớn hơn > 0.05 nên nó không có ý nghĩa thống kê hay nói cách khác mối tương quan sát được giữa RestingECG và HeartDisease có thể là ngẫu nhiên. Nên khi thống kê ta có thể bỏ qua giá trị RestingECG.

6.1) Đối với giá trị số

_ Các giá trị như Age, RestingBP và Oldpeak có tương quan dương với HeartDisease có nghĩa là khi các giá trị đó tăng thì tỉ lệ bệnh tim cũng sẽ tăng (do HeartDisease có giá trị là 1 trong tập dữ liệu).

_ Còn các giá trị như Cholesterol và MaxHR có tương quan âm với HeartDisease có nghĩa là khi các giá trị đó tăng thì tỉ lệ bệnh tim sẽ giảm dần.

6.2) Đối với giá trị loại

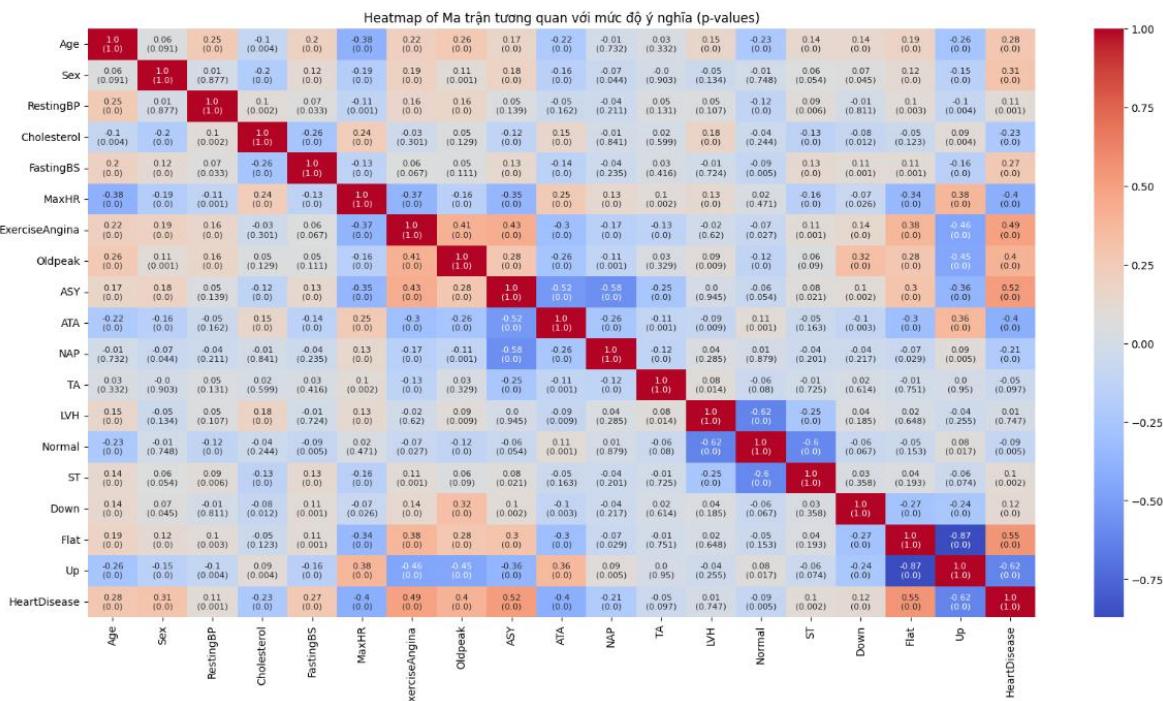
_ Các giá trị như Sex, FastingBS, ExerciseAngina tương quan dương với HeartDisease có nghĩa là khi Sex là **M** (nam) thì tỉ lệ bệnh tim sẽ cao hơn **F** (nữ) hoặc FastingBS là **1** ($\text{FastingBS} > 120 \text{ mg/dl}$) sẽ có tỉ lệ mắc bệnh tim cao hơn Fasting BS là **0** (cho các giá trị đó còn lại), ExerciseAngina cũng như vậy khi giá trị là **Y** (đau thắt ngực khi gắng sức) thì tỉ lệ bệnh tim sẽ cao hơn **N** (không đau thắt ngực khi gắng sức).

_ Còn các giá trị ChestPainType và ST_Slope thì tương quan âm với HeartDisease. Nhưng do có khá nhiều giá trị trong hai loại trên nên không thể dùng ma trận tương quan để đánh giá xem đặc điểm nào sẽ tăng tỉ lệ bệnh tim. Nên ta sẽ chuyển các giá trị trong đó thành các cột riêng biệt.

	Age	Sex	RestingBP	Cholesterol	FastingBS	MaxHR	ExerciseAngina	\		
0	40	1	140	289	0	172		0		
1	49	0	160	180	0	156		0		
2	37	1	130	283	0	98		0		
3	48	0	138	214	0	108		1		
4	54	1	150	195	0	122		0		
	Oldpeak	HeartDisease	ASY	ATA	NAP	TA	LVH	Normal	ST	\
0	0.0		0 False	True	False	False	False	True	False	
1	1.0		1 False	False	True	False	False	True	False	
2	0.0		0 False	True	False	False	False	False	True	
3	1.5		1 True	False	False	False	False	True	False	
4	0.0		0 False	False	True	False	False	True	False	
	Down	Flat	Up							
0	False	False	True							
1	False	True	False							
2	False	False	True							
3	False	True	False							
4	False	False	True							

_ Ở đây ta vẫn giá lại cột RestingECG vì muốn chắc chắn rằng việc chuyển đổi các trị trong RestingECG là các giá trị 0, 1, 2 sẽ không ảnh hưởng đến xác định giá trị p-value.

_ Và sau đó ta sẽ chuyển các giá trị True và False sang dạng nhị phân là 1 và 0. Rồi vẽ biểu đồ tương quan cho Data đó.



_ Lần này ta sẽ chú ý đến các cột được sinh ra từ cột ChestPainType, RestingECG và ST_Slope.

_ ChestPainType có 4 giá trị bao gồm ASY, ATA, NAP, TA: ta thấy rằng chỉ có TA là có p-value > 0.05 (không có ý nghĩa thống kê) nhưng 3 giá trị còn lại thì lại có ý nghĩa thống kê. Có nghĩa rằng khi giá trị ChestPaintType là ASY, ATA, NAP thì nó sẽ có ảnh hưởng đến tỉ lệ người mắc bệnh tim, nhưng đối với TA thì không. Vì có 3 giá trị trên 4 có ý nghĩa thống kê nên ta sẽ giữ lại giá trị.

_ Đổi với RestingECG có 3 giá trị bao gồm LVH, Normal và ST thì ta thấy có LVH là có giá trị p-value > 0.05 rất nhiều nên ta chắc chắn rằng khi RestingECG là LVH thì ta sẽ không có kết luận được gì. Còn đổi với 2 giá trị Normal và ST ta thấy tuy p-value < 0.05 nhưng giá trị tương quan của nó là rất thấp lần lượt là -0.09 và 0.1 nên ta có thể bỏ đi cột này để quá trình tính đơn giản hơn.

_ Đổi với ST_Slope gồm 3 giá trị là Down, Flat và Up thì đều có p-value= 0 và giá trị tương quan cũng khá cao lần lượt là nên ta sẽ giữ lại cột này để tính toán.

7) Kết luận về Dataset

Kiểu dữ liệu là số:

- Age: khi tuổi từ 50 trở đi thì tỉ lệ người mắc bệnh tim tăng cao.
- RestingBP: trong khoảng 95(mm/dl) đến 170 (mm/dl) sẽ có tỉ lệ bệnh tim cao.
- Cholesterol: trong khoảng 160(mm/dl) đến 340(mm/dl) có tỉ lệ bệnh tim cao.
- MaxHR: khi nhịp tim trong khoảng 70 đến 180 thì tỉ lệ bệnh tim cao.
- Oldpeak: khi giá trị trong khoảng giá trị 0 đến 4 thì tỉ lệ bệnh tim cao.

Kiểu dữ liệu là loại.

- Sex: Nam sẽ có tỉ lệ bệnh tim cao hơn so với Nữ.
- ChestPainType: ASY> NAP> ATA <TA (tỉ lệ bệnh tim theo thứ tự giảm dần).
- FastingBS: Tỉ lệ bệnh tim của **1** (FastingBS > 120 mg/dl) > **0** (FastingBS < 120 mg/dl).
- ExerciseAngina: nếu ExerciseAngina là **Yes** thì tỉ lệ bệnh tim sẽ cao hơn rất nhiều so với **No**.
- ST_Slope: Flat > Up > Down (tỉ lệ bệnh tim theo thứ tự giảm dần).

Và ta đã loại bỏ cột RestingECG khỏi tập dữ liệu do p-value > 0.05 và giá trị tương quan của nó so với giá trị mục tiêu thấp.

Chương 2: Tiền xử lí dữ liệu

1) Xử lí các giá trị Null

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 918 entries, 0 to 917
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype  
--- 
 0   Age               918 non-null    int64  
 1   Sex               918 non-null    object  
 2   ChestPainType    918 non-null    object  
 3   RestingBP         918 non-null    int64  
 4   Cholesterol      918 non-null    int64  
 5   FastingBS        918 non-null    int64  
 6   RestingECG        918 non-null    object  
 7   MaxHR             918 non-null    int64  
 8   ExerciseAngina   918 non-null    object  
 9   Oldpeak           918 non-null    float64 
 10  ST_Slope          918 non-null    object  
 11  HeartDisease     918 non-null    int64  
dtypes: float64(1), int64(6), object(5)
```

Do dữ liệu không có bất kỳ giá trị Null nào nên ta sẽ bỏ qua bước này

2) Data Scaling

_Trong bài báo cáo này làm về thuật toán SVM là một dạng tính khoảng cách để quyết định xem điểm đó sẽ thuộc lớp nào. Và nếu ta không chuẩn hóa dữ liệu thì các dữ liệu đưa vào chỉ đơn giản là các con số thông thường và các giá trị khác nhau ở các cột khác nhau sẽ có thang đo khác. Vì vậy chuẩn hóa dữ liệu sẽ khiến cho dữ liệu được cân bằng giữa các đặc trưng và khiến cho mô hình hiệu quả hơn.

2.1) Normalization là gì

_ Min-Max Scaling là một dạng Normalization nó sẽ chuẩn hóa giá trị dựa trên Max và Min của dữ liệu để chuẩn hóa dữ liệu về khoảng từ 0 đến 1.

$$X_{new} = \frac{X_{old} - X_{min}}{X_{max} - X_{min}}$$

Trong đó:

- + X_{new} : là giá trị của X sau khi chuẩn hóa
- + X_{old} : là giá trị của X ban đầu
- + X_{min} : là giá trị nhỏ nhất trong cột chứa giá trị X
- + X_{max} : là giá trị lớn nhất trong cột chứa giá trị X

2.2) Standardization là gì

_ Standardization là một kỹ thuật chuẩn hóa dữ liệu, trong đó các giá trị sau khi chuẩn hóa sẽ tập trung xung quanh giá trị trung bình với độ lệch chuẩn là 1. Có nghĩa là dữ liệu sau khi chuẩn hóa sẽ có giá trị trung bình (mean) sẽ trở thành 0 và độ lệch chuẩn sẽ bằng 1.

$$X_{new} = \frac{X_{old} - \mu}{\sigma}$$

Trong đó:

- _ X_{new} : là giá dữ X sau khi chuẩn hóa
- _ X_{old} : là giá trị của dữ liệu ban đầu
- _ μ : là giá trị trung bình của cột chứa dữ liệu đó.
- _ σ : là độ lệch chuẩn của dữ liệu

2.3) Chọn phương pháp nào tốt hơn?

_ Trong bài này ta đã chọn Support Vector Machine làm thuật toán để phân tích nên ta sẽ chọn Standardization để chuẩn hóa dữ liệu. Bởi vì SVM (Support Vector Machine) không cần dữ liệu phải nằm trong khoảng cố định (0 ,1) và sẽ xử lý tốt hơn nếu dữ liệu có các outliers. Hơn nữa dữ liệu trong tập dữ liệu này cũng không có thang đo cố định cho từng cột dữ liệu nên việc xài Standardization sẽ hiệu quả hơn.

2.4) Dữ liệu sau khi chuẩn hóa

	Age	Sex	RestingBP	Cholesterol	FastingBS	MaxHR	ExerciseAngina
0	-1.433140	1	0.410909	0.825070	0	1.382928	0
1	-0.478484	0	1.491752	-0.171961	0	0.754157	0
2	-1.751359	1	-0.129513	0.770188	0	-1.525138	0
3	-0.584556	0	0.302825	0.139040	0	-1.132156	1
4	0.051881	1	0.951331	-0.034755	0	-0.581981	0

	Oldpeak	ASY	ATA	NAP	TA	LVH	Normal	ST	Down	Flat	Up	HeartDisease
0	-0.832432	0	1	0	0	0	1	0	0	0	1	0
1	0.105664	0	0	1	0	0	1	0	0	1	0	1
2	-0.832432	0	1	0	0	0	0	1	0	0	1	0
3	0.574711	1	0	0	0	0	1	0	0	1	0	1
4	-0.832432	0	0	1	0	0	1	0	0	0	1	0

3) Xử lý dữ liệu dạng cột

3.1) Đổi với các cột có giá trị là nhị phân (Yes hoặc No, Male hoặc Female, ...)

_Đối với các dữ liệu này ta sẽ sử dụng LabelEncoder để chuyển nó thành kiểu số có giá trị là 1 và 0.

```
df2["Sex"] = le.fit_transform(df2["Sex"])
df2["ExerciseAngina"] = le.fit_transform(df2["ExerciseAngina"])
```

3.2) Đổi với các cột có số giá trị lớn hơn ba

_Còn các kiểu dữ liệu có số giá trị lớn hơn ba ta sẽ chuyển các giá trị đó thành cột riêng với các giá trị True False và sau đó sử dụng LabelEncoder để đưa nó về dạng số tương tự như cột có giá trị nhị phân

```
df2 = pd.get_dummies(df2, columns=['ChestPainType'], prefix='',
prefix_sep=' ')
df2 = pd.get_dummies(df2, columns=['RestingECG'], prefix='',
prefix_sep=' ')
df2 = pd.get_dummies(df2, columns=['ST_Slope'], prefix='',
prefix_sep=' ')
categorical_columns = df2.select_dtypes(exclude=[np.number]).columns
for category in (categorical_columns):
    df2[category] = le.fit_transform(df2[category])
```

Và đây là kết quả

	Age	Sex	RestingBP	Cholesterol	FastingBS	MaxHR	ExerciseAngina	\				
0	-1.433140	1	0.410909	0.825070	0	1.382928	0	0				
1	-0.478484	0	1.491752	-0.171961	0	0.754157	0	0				
2	-1.751359	1	-0.129513	0.770188	0	-1.525138	0	0				
3	-0.584556	0	0.302825	0.139040	0	-1.132156	1	1				
4	0.051881	1	0.951331	-0.034755	0	-0.581981	0	0				
	Oldpeak	ASY	ATA	NAP	TA	LVH	Normal	ST	Down	Flat	Up	HeartDisease
0	-0.832432	0	1	0	0	0	1	0	0	0	1	0
1	0.105664	0	0	1	0	0	1	0	0	1	0	1
2	-0.832432	0	1	0	0	0	0	1	0	0	1	0
3	0.574711	1	0	0	0	0	1	0	0	1	0	1
4	-0.832432	0	0	1	0	0	1	0	0	0	1	0

4) Chia tách dữ liệu

_Thông thường ta cần ít nhất tập dữ liệu là train để huấn luyện model và test để kiểm tra sự chính xác của model. Ta có thể chọn 1 trong 2 thuật toán là K-fold cross-validation và Stratified k-fold cross-validation.

4.1) K-fold cross-validation

_K-fold cross-validation sẽ chia dữ liệu ra thành k tập bằng nhau không quan tâm đến tỉ lệ các lớp. Sau đó thực hiện k vòng lặp và mỗi tập ta sẽ chia ra thành 80% dữ liệu để huấn luyện và 20% còn lại để kiểm tra. Sau đó ta cộng lấy trung bình để xác định tính chính xác.

4.2) Stratified k-fold cross-validation

_Stratified k-fold cross-validation cũng tương tự như K-fold cross-validation nhưng nó quan tâm đến tỉ lệ dữ liệu trong từng tập tương tự như trong dữ liệu gốc (Ví dụ như trong dữ liệu gốc ta có (79% dữ liệu có Sex là Male và 21% là Female thì sau khi chia ra thành K tập thì nó vẫn giữ nguyên tỉ lệ đó trong các tập).

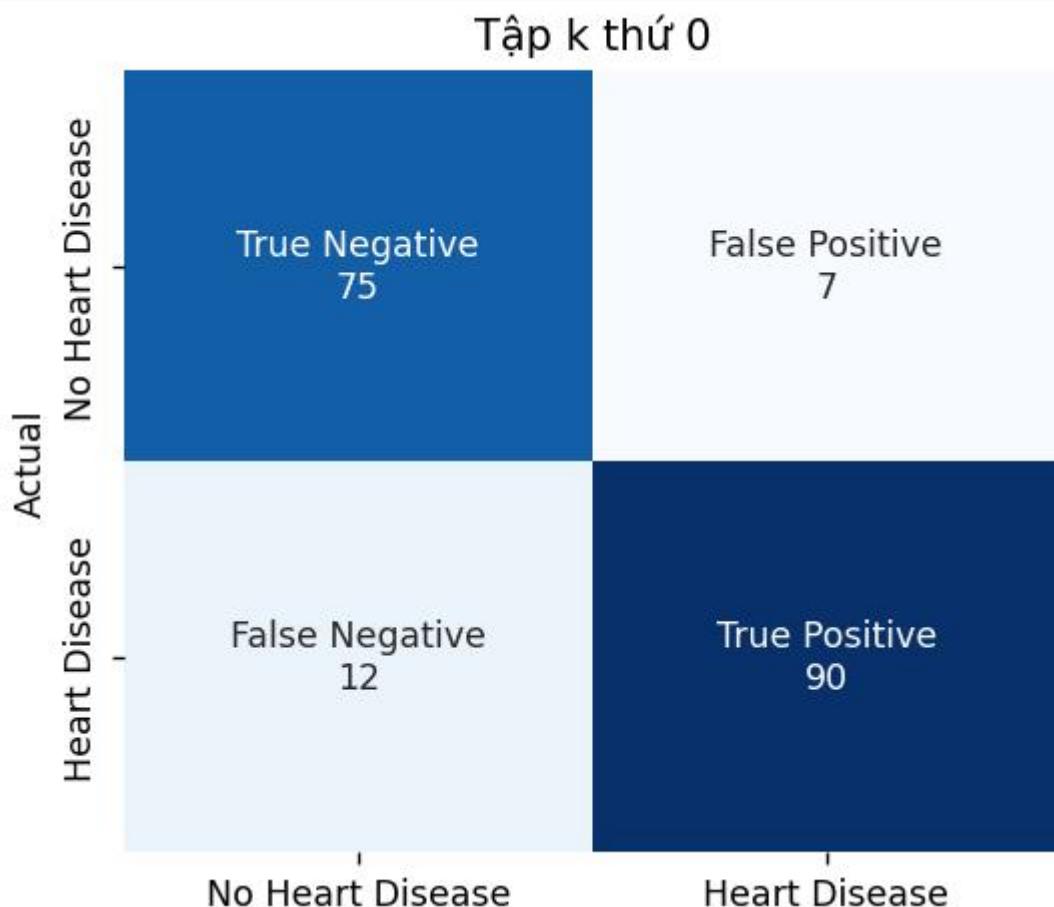
4.3) Cái nào phù hợp hơn

_Do dữ liệu ta có một vài nhược điểm về tỉ lệ như Sex, FastingBS, ... nên ta sẽ chọn Stratified k-fold cross-validation để chia tách dữ liệu và đảm bảo tính chính xác của model.

Chương 3: Thuật toán Support vector machine

1) Giới thiệu về các thuật ngữ, đại lượng dùng để xác định tính chính xác của mô hình

Ví dụ ta có các kết quả dự đoán sau:



No Heart Disease: Precision: 0.86, Recall: 0.91, F1-Score: 0.89, Support: 82

Heart Disease: Precision: 0.93, Recall: 0.88, F1-Score: 0.90, Support: 102

Mức độ chính xác cho tập K thứ 0: 0.90

Ở đây có các loại kết quả như sau:

+**True Negative** 75: True Negative có nghĩa ta đã dự đoán đúng trường hợp âm tính (No Heart Disease) cho người thực sự không mắc bệnh tim. Vậy True Negative 75 là ta đã dự đoán đúng cho 75 người không mắc bệnh tim và thực sự là như vậy

+**False Positive** 7: False Positive là một dạng dự đoán sai trường hợp dương tính (Heart Disease) có nghĩa rằng ta đã chẩn đoán sai cho bệnh nhân rằng họ đã mắc bệnh tim nhưng thực tế thì không. Nên False Positive 7 chính là dự đoán 7 người mắc bệnh tim nhưng 7 người này không hề mắc bệnh tim.

+**False Negative** 12: False Negative thì cũng là một dạng dự đoán sai trường hợp âm tính (No Heart Disease) nhưng là dự đoán sai âm tính tức là ta dự đoán 12 bệnh nhân này không mắc bệnh tim nhưng họ có mắc bệnh tim.

+**True Positive** 90: True Positive là dự đoán đúng trường hợp dương tính (Heart Disease) có nghĩa rằng ta dự đoán là người đó mắc bệnh và thực tế người đó cũng mắc bệnh.

+**Support** đại diện cho số lượng dữ liệu, ví dụ như ở No Heart Disease to có Support là 82 tức là có 82 hàng có giá trị mục tiêu là No Heart Disease trong tập K thứ 0 và Heart Disease có 102 hàng (Support :102).

Precision: chính là độ chính xác đo lường khả năng mô hình đưa ra dự đoán là Positive (Heart Disease) và dự đoán đó là đúng trên tập dữ liệu thứ K, giá trị Precision cao chứng tỏ tập K này ít bị True Negative. Và được xác định bằng công thức:

$$Precision = \frac{True\ Positive}{True\ Positive + True\ Negative}$$

Recall: là giá trị đo lường khả năng mô hình phát hiện tất cả giá trị Positive thật sự. Giá trị này cao chứng tỏ tập K ít bị False Negative, hay còn gọi là bỏ sót các trị thực sự là Positive. Giá trị được xác định bằng:

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

F1-Score: là harmonic mean (điểm hài hòa) của Precision và Recall, khi giá trị này càng cao thì mức độ phân loại càng tốt và ngược lại. Giá trị được xác định bằng công thức:

$$F1 = 2 \frac{Precision \cdot Recall}{Precision + Recall}$$

_ Do F1-Score phụ thuộc vào cả 2 giá trị Precision và cả Recall nên nếu một trong hai thấp thì F1-Score cũng giảm đi đáng kể. Ví dụ:

Precision	Recall	F1-Score

1	1	1
0.1	0.1	0.1
0.5	0.5	0.5
1	0.1	0.182
0.3	0.8	0.36

Kết quả trên cho ta thấy Precision =0.3 và Recall =0.8 sẽ có F1-Score bé hơn so với Precision và Recall đều là 0.5.

+**Accuaracy** (mức độ chính xác) chính là tỉ lệ dự đoán chính xác trên tập dữ liệu thứ K, được tính bằng:

$$Accuaracy = \frac{True\ Negative + True\ Positive}{True\ Negative + False\ Positive + False\ Negative + True\ Positive}$$

2) Giới thiệu thuật toán SVM

2.1) Support Vector Machine là gì

_ Support Vector Machine là một mô hình học máy có khả năng thực hiện phân loại dữ liệu tuyến tính, phi tuyến tính, hồi quy hay thậm chí là phát hiện ngoại lệ. SVM đặc biệt phù hợp cho việc phân loại các tập dữ liệu phức tạp có quy mô vừa và nhỏ.

2.2) Giải thích sơ lược về thuật toán

_ Mục tiêu chính của thuật toán SVM (Support Vector Machine) dựa trên việc tìm siêu phẳng tối ưu nhất trong không gian N chiều để có thể phân tách các hiệu quả các điểm dữ liệu thành các lớp khác nhau trong không gian đặc điểm. Và phải đảm bảo rằng khoảng cách giữa các điểm gần nhất của các lớp khác nhau so với siêu phẳng phải là lớn nhất và khoảng cách đó gọi là **margin** (hay còn gọi là **lề**) và các điểm gần nhất so với siêu phẳng đó gọi là **Support Vector**.

3) Thuật toán SVM

3.1) SVM phân loại dữ liệu tuyến tính

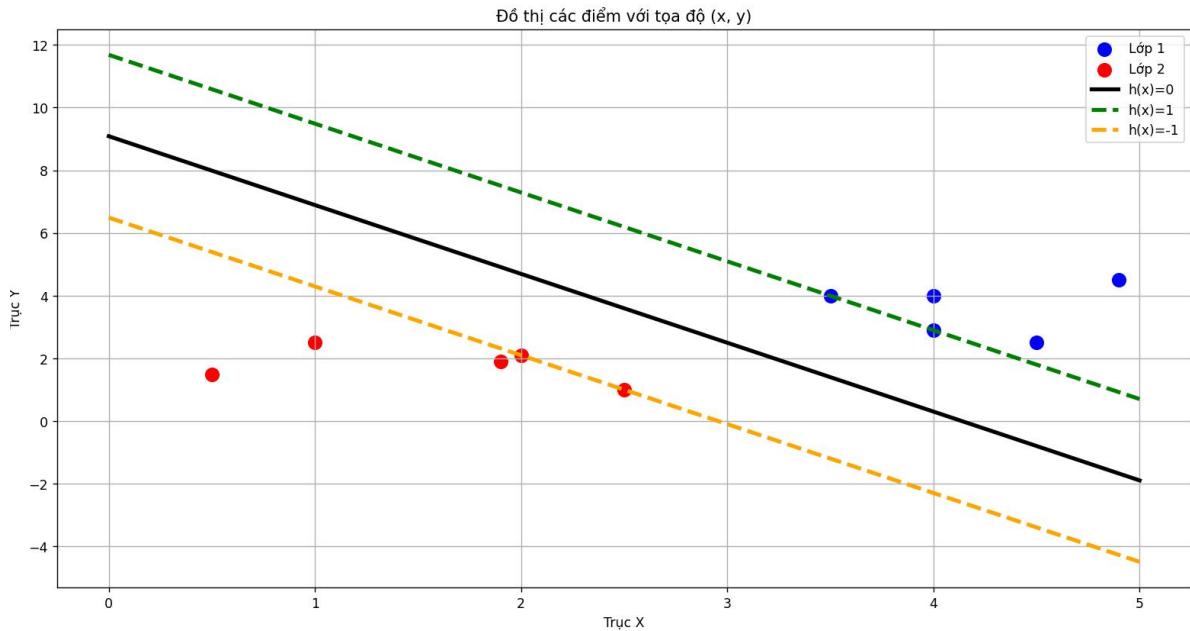
3.1.1) Dữ liệu tuyến tính là gì

Dữ liệu tuyến tính là dữ liệu mà mối quan hệ giữa giá trị đầu vào và giá trị đầu ra có thể được biểu diễn bằng một đường thẳng hoặc một siêu phẳng trong không gian nhiều chiều.

3.1.2) Hard margin

a) Hard margin là gì?

_Hard margin là một thuật toán của SVM dùng để phân loại dữ liệu tuyến tính với margin sẽ tách biệt hoàn toàn các điểm dữ liệu mà không có bất kỳ sai sót nào.



_Đây là ví dụ về một Hard margin khi mà nó sẽ chia tách hoàn hảo dữ liệu và không có dữ nào sai sót, có nghĩa rằng trong quá trình huấn luyện khi tính margin thì sẽ không có bất kỳ điểm dữ liệu màu đỏ nào mà nằm bên phía margin của Lớp 2 và ngược lại.

b) Cơ sở lý thuyết

_Mục tiêu của thuật toán chính là tìm ra được một siêu phẳng sao cho có margin lớn nhất (tức là khoảng cách từ siêu phẳng đến đến điểm dữ liệu gần nhất của mỗi lớp).

Ta cho $D = \{(x_i, y_i)\}_{i=1}^n$ là tập dữ liệu và x_i là một điểm dữ liệu trong dữ liệu và y_i là giá trị đại diện cho các lớp cần phân loại và vì ta chỉ có 2 lớp nên $y_i \in \{1, -1\}$.

Một siêu phẳng trong mặt phẳng d chiều được xác định bởi tất cả các điểm $x \in \mathbb{R}^d$ mà nó thỏa mãn phương trình $h(x) = 0$ là phương trình siêu phẳng với $h(x)$ được viết là:

$$h(x) = w^T x + b = w_1 x_1 + w_2 x_2 + w_3 x_3 + \dots + w_d x_d + b$$

Với w là một vector trọng số d chiều và b là một giá trị vô hướng được gọi là độ lệch (bias) ta có thể điều chỉnh giá trị này khiến cho mô hình trở nên phức tạp hơn. Nếu b quá cao thì mô hình sẽ bị underfitting (mô hình sẽ quá đơn giản do học không đủ tốt từ dữ liệu huấn luyện)

hoặc khi b quá nhỏ thì mô hình sẽ bị overfitting (mô hình quá phức tạp và chỉ phù hợp với mô hình huấn luyện).

Với các điểm dữ liệu nằm trên siêu phẳng thì ta có phương trình:

$$h(x) = w^T x + b = 0$$

Trong đó:

- w là một vector trọng số sẽ chỉ định hướng vuông góc với siêu phẳng giúp cố định hướng của siêu phẳng.
- b là độ lệch của siêu phẳng trong không gian d chiều tức là độ lệch của siêu phẳng so với trục.
- x là tập các điểm dữ liệu

Siêu phẳng sẽ chia dữ liệu thành 2 nửa. Vậy để có thể áp dụng được hard margin vào tập dữ liệu ta phải đảm bảo rằng mỗi một nửa của dữ liệu chỉ chứa dữ liệu thuộc về 1 class. Tức là $\forall x \in \mathbb{R}^d$ phải thỏa mãn nếu $y_i = -1$ thì ta sẽ có $h(x) < 0$ và $y_i = 1$ thì $h(x) > 0$. Nếu điều kiện trên đã thỏa mãn ta có thể bắt đầu tìm siêu phẳng $h(x)$ để chia đôi dữ liệu.

Ta có khoảng cách từ một điểm x bất kỳ đến siêu phẳng $h(x) = 0$ được thể hiện qua công thức

$$\delta = \frac{y_i h(x_i)}{\|w\|} = \frac{y_i (w^T x + b)}{\|w\|}$$

Trong đó:

y_i là giá trị đại diện cho 1 lớp

$h(x_i)$ là kết quả của phương trình siêu phẳng với x_i

$\|w\|$ là độ dài của vector trọng số w được tính theo công thức:

$$\|w\| = \sqrt{w_1^2 + w_2^2 + w_3^2 + \dots + w_d^2}$$

Vậy khoảng cách từ margin đến một điểm ngắn nhất sẽ là:

$$\delta^* = \min_{xi} \left\{ \frac{y^* (w^T x^* + b)}{\|w\|} \right\}$$

Với y^* là lớp của giá trị x^* và x^* chính là support vector.

Để đơn giản hơn trong việc kiểm tra xem 1 điểm có phải là support vector hay không ta có thể chuẩn hóa phương trình bằng cách nhân s vào 2 vế của phương trình $h(x)$:

$$s h(x) = s (w^T x + b)$$

Và để có được các siêu phẳng duy nhất ta sẽ chọn $s = \frac{1}{y^*(w^T x^* + b)}$. Sau đó thay vào phương trình tính khoảng cách:

$$\delta^* = \frac{y^* (w^T x^* + b)}{\|w\|} = \frac{1}{\|w\|}$$

Vậy các giá trị là support vector sẽ có $y_i(w^T x_i + b) = 1$ trong khi các điểm không phải là support vector sẽ có giá trị $y_i(w^T x_i + b) > 1$.

Và ta muốn giá trị δ^* này là lớn nhất nên ta sẽ có:

- **Hàm mục tiêu:** $\min_{w,b} = \left\{ \frac{\|w\|^2}{2} \right\}$
- **Ràng buộc:** $y_i(w^T x_i + b) \geq 1, \forall x_i \in D$

Vậy để giải quyết bài toán tối ưu hóa đi kèm điều kiện ta sẽ sử dụng Lagrange với hệ số α_i cho mỗi ràng buộc. Ta sẽ được hàm mục tiêu mới gọi là Lagrangian là:

$$\min L = \frac{1}{2} \|w\|^2 - \sum_{i=1}^n \alpha_i (y_i (w^T x_i + b) - 1)$$

$$\frac{\partial}{\partial w} L = w - \sum_{i=1}^n \alpha_i y_i x_i = 0 \Leftrightarrow w = \sum_{i=1}^n \alpha_i x_i y_i$$

$$\frac{\partial}{\partial b} L = \sum_{i=1}^n \alpha_i y_i = 0$$

Chúng ta có thể thấy rằng w được biểu diễn bằng tổng các tích của các điểm dữ liệu x và $\alpha_i y_i$, với $\alpha_i y_i$ là hệ số Lagrangian.

Và điều kiện là tổng có dấu của các $\alpha_i y_i$ phải bằng 0. ($\forall x_i \in D$ thì $\alpha_i \geq 0$)

Bằng cách kết hợp $w = \sum_{i=1}^n \alpha_i x_i y_i$ và $\sum_{i=1}^n \alpha_i y_i = 0$ vào Lagrangian ta sẽ có được một hàm Lagrangian kép mới, được xác định hoàn toàn bằng các vector nhân tử Lagrange.

$$\text{Hàm mục tiêu: } \max_a L_{dual} = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j x_i^T x_j$$

Ràng buộc: $\alpha_i \geq 0, \forall x_i \in D$ và $\sum_{i=1}^n \alpha_i y_i = 0$

Mỗi giá trị α_i đều phải thỏa mãn điều kiện KKT (Karush-Kuhn-Tucker) để tìm nghiệm α_i tối ưu và thỏa mãn các điều kiện ở trên:

$$\alpha_i [y_i(w^T x_i + b) - 1] = 0$$

Thì ta có 2 trường hợp:

$$y_i (w^T x_i + b) - 1 = 0 \text{ hay } y_i (w^T x_i + b) = 1$$

$$\alpha_i = 0$$

Nếu $\alpha_i > 0$ thì $y_i (w^T x_i + b) = 1$ tức x_i là một support vector.

Ngược lại nếu $y_i (w^T x_i + b) > 1$ thì $\alpha_i = 0$, nên nếu $\alpha_i = 0$ thì điểm đó không phải là support vector.

Sau khi tính được tất cả giá trị α_i thì ta sẽ tính được vector trọng số w và b bằng cách thế vào phương trình:

$$w = \sum_{i=1}^{m \in S} \alpha_i x_i y_i$$

$$b = \frac{1}{m} \sum_{i=1}^{m \in S} y_i - w^T x_i$$

Với m là số phần tử support vector trong tập dữ liệu.

c) Áp dụng thuật toán

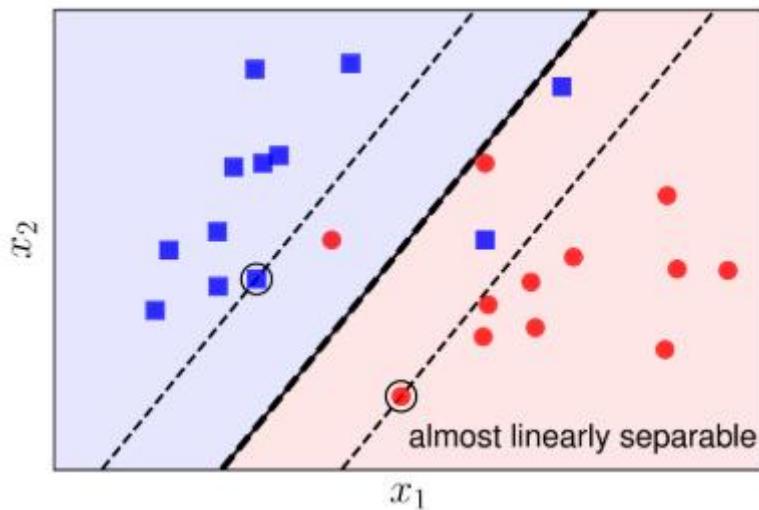
_Do hard margin không cho phép sai do tất cả điểm dữ liệu đều phải thỏa mãn $y_i(w^T x_i + b) \geq 1 \forall x_i \in D$ nhưng trong tập dữ liệu này chúng ta lại có rất nhiều giá trị cá biệt (outliers) nên việc mà mô hình huấn luyện tìm ra được siêu phẳng chia cắt hoàn toàn dữ liệu thành 2 nửa là một điều không thể.

d) Đánh giá thuật toán

_Do phải thỏa mãn ràng buộc của thuật toán nên mô hình huấn luyện không thể tìm ra được siêu phẳng chia cắt hoàn hảo dữ liệu thành 2 nửa. Vậy hard margin khó có thể áp dụng cho các tập dữ liệu thực tế khi mà các trường hợp ngoại lệ có thể xảy ra.

3.1.3) Soft Margin

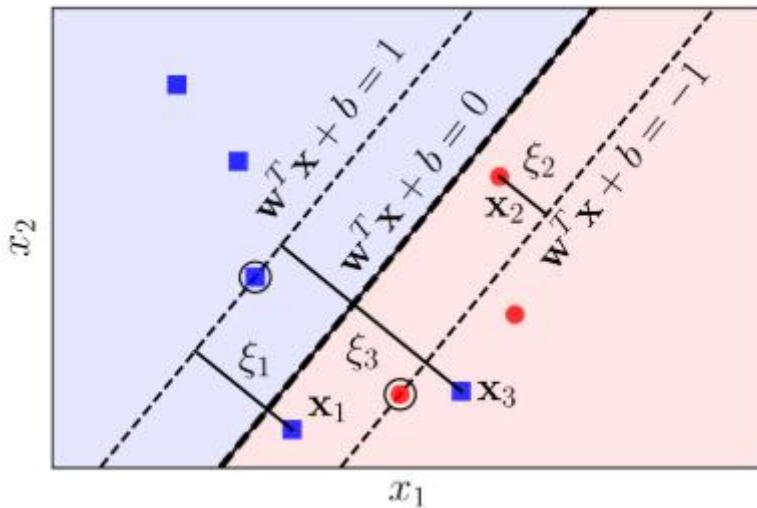
a) Soft Margin là gì



_Đối với các tập dữ liệu mà không thể tìm ra siêu phẳng phân tách hoàn hảo dữ liệu thành 2 nửa như trên thì ta cần một giải pháp khác. Soft margin cũng là 1 phương pháp SVM với mục tiêu chính là dùng 1 siêu phẳng có margin lớn nhất để phân loại dữ liệu thành 2 nửa như hard margin. Nhưng ở đây nó sẽ hy sinh một vài điểm dữ liệu bằng cách cho phép chúng rơi vào 1 lớp khác không phải của nó. Và ta cần phải hạn chế sự hy sinh này, nếu không ta có thể tạo ra một margin cực lớn bằng cách hy sinh một lượng lớn điểm dữ liệu. Vậy hàm mục tiêu của chúng ta bây giờ sẽ là sự kết hợp giữa tối đa margin và tối thiểu sự hy sinh. Vậy đối với các loại dữ liệu thực tế mà hard margin không thể xử lý được ta có thể sử dụng margin và điều chỉnh để cân bằng giữa sự hy sinh và margin để chọn ra giải pháp tốt nhất thay vì chỉ chú trọng vào margin hoàn hảo nhất mà không có giải pháp nào được đưa ra.

b) Cơ sở lý thuyết

_Cũng giống như hard margin mục tiêu của chúng ta là tối ưu khoảng cách giữa support vector nhưng lần này ta có thêm một biến ξ_i thể hiện mức độ vi phạm của điểm dữ liệu (hay mức độ hy sinh).



Trong hình ta có một siêu phẳng phân tách dữ liệu nhưng chưa hoàn toàn do có các điểm màu xanh nằm bên phía bên lề của màu đỏ. Với các điểm như x_1, x_2, x_3 là các điểm nằm không đúng bên lề của nó, ta sẽ gọi đó là các điểm vi phạm với mức độ vi phạm ξ_i được tính bằng công thức:

$$\xi_i = |w^T x_i + b - y_i|$$

Với $\xi_i = 0$ là các điểm được phân loại đúng và nằm ngoài margin.

Với $0 < \xi_i < 1$ là các điểm được nằm trong margin nhưng vẫn được phân loại đúng ví dụ như x_2 .

Với $\xi_i \geq 1$ là các điểm dữ liệu nằm sai lề so với lề nó đã được phân loại ví dụ như x_1 và x_3 .

Với hard margin ta có:

- **Hàm mục tiêu:** $\min_{w,b} = \left\{ \frac{\|w\|^2}{2} \right\}$
- **Ràng buộc:** $y_i(w^T x_i + b) \geq 1, \forall x_i \in D$

Nhưng đối với soft margin ta có thêm một biến để tối thiểu sự hy sinh, nên ta sẽ có hàm mục tiêu và ràng buộc mới là:

- **Hàm mục tiêu:** $\min_{w,b,\xi_i} = \left\{ \frac{\|w\|^2}{2} + C \sum_{i=1}^n \xi_i \right\}$
- **Ràng buộc:** $y_i(w^T x_i + b) \geq 1 - \xi_i, \forall x_i \in D$
 $\xi_i \geq 0 \forall x_i \in D$

Trong đó:

C và k là hằng số để điều chỉnh mức độ ảnh hưởng của ξ_i lên hàm mục tiêu. Tức là C và k là hằng số để ta xác định xem nên tập trung vào mức độ ảnh hưởng ξ_i bao nhiêu và tối ưu hóa margin là bao nhiêu.

Khi $C \rightarrow 0$ có nghĩa là ta sẽ loại bỏ việc tối thiểu sự hy sinh và chỉ tập trung vào tối đa margin.

Khi $C \rightarrow \infty$ thì thuật toán sẽ tập trung để làm sao tìm ra margin sao cho $\xi = 0$.

Hàm Hinge loss sẽ tối ưu tổng của các giá trị ξ_i , với Hinge loss ta có Lagrangian:

$$\min L = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i - \sum_{i=1}^n \alpha_i (y_i (w^T x_i + b) - 1 + \xi_i) - \sum_{i=1}^n \mu_i \xi_i \quad (1)$$

$$\frac{\partial}{\partial w} L = w - \sum_{i=1}^n \alpha_i y_i x_i = 0 \Leftrightarrow w = \sum_{i=1}^n \alpha_i x_i y_i \quad (2)$$

$$\frac{\partial}{\partial b} L = \sum_{i=1}^n \alpha_i y_i = 0 \quad (3)$$

$$\frac{\partial}{\partial \xi_i} L = C - \alpha_i - \mu_i = 0 \Leftrightarrow \alpha_i = C - \mu_i \quad (4)$$

Trong đó $\alpha_i \geq 0$, nên $C - \mu_i \geq 0$ là trong đó α_i, μ_i là các vector nhân tử Lagrange. Thay các biểu thức (2), (3) và (4) vào (1).

Ta có:

$$\textbf{Hàm mục tiêu: } \max_{\alpha} L_{dual} = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j x_i^T x_j$$

$$\textbf{Ràng buộc: } 0 \leq \alpha_i \leq C, \forall x_i \in D \text{ và } \sum_{i=1}^n \alpha_i y_i = 0$$

Sau khi tìm được α_i thay vào (2) và (3) ta sẽ được:

$$w = \sum_{i=1}^{m \in S} \alpha_i x_i y_i$$

$$b = \frac{1}{m} \sum_{i=1}^{m \in S} y_i - w^T x_i$$

Với m là các điểm support vector trong tập dữ liệu.

Ngoài ra ta còn có có Quadratic loss khá giống với Hinge Loss chỉ khác nhau ở chỗ hàm mục tiêu của nó là sẽ tối ưu tổng của ξ_i^2 với hàm mục tiêu như sau:

- **Hàm mục tiêu:** $\min_{w,b,\xi_i} = \left\{ \frac{\|w\|^2}{2} + C \sum_{i=1}^n \xi_i^2 \right\}$
- **Ràng buộc:** $y_i(w^T x_i + b) \geq 1 - \xi_i, \forall x_i \in D$

$$\xi_i \geq 0 \quad \forall x_i \in D$$

Và nó được khiết trai Lagrange khá giống với Hinge loss nên ta sẽ bỏ qua bước này.

Ta thấy hàm mục tiêu và ràng buộc của hard margin và soft margin khá giống nhau, chỉ khác ở chỗ ta có chặn trên cho mỗi α_i . Vậy khi C rất lớn, ta có thể coi là 2 thuật toán là như nhau.

c) Áp dụng thuật toán

```
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report, roc_auc_score,
confusion_matrix
from sklearn.preprocessing import StandardScaler, LabelEncoder
import pandas as pd
from sklearn import model_selection
from sklearn.svm import SVC

acc_svm = []
confusion_matrices = []
le = LabelEncoder()
df = pd.read_csv("heart.csv")
df = df.drop(columns=["RestingECG"])
df["Sex"] = le.fit_transform(df["Sex"])
df["ExerciseAngina"] = le.fit_transform(df["ExerciseAngina"])
df = pd.get_dummies(df, columns=['ChestPainType'], prefix='',
prefix_sep='')
```

```

df = pd.get_dummies(df, columns=['ST_Slope'], prefix='', prefix_sep='')

target = "HeartDisease"
feature = df.columns.to_list()
feature.remove(target)
y = df[target].values

kf = model_selection.StratifiedKFold(n_splits=5)
avg_accuracy = 0

for fold, (train, value) in enumerate(kf.split(X=df, y=y)):
    X_train = df.loc[train, feature]
    heart_diseaseTrain = df.loc[train, target]
    X_valid = df.loc[value, feature]
    heart_diseaseTest = df.loc[value, target]

    ss = StandardScaler()
    X_train = ss.fit_transform(X_train)
    X_valid = ss.transform(X_valid)

    model_classification = SVC(kernel="linear", C=0.001)
    model_classification.fit(X_train, heart_diseaseTrain)
    result_predict = model_classification.predict(X_valid)

    cm = confusion_matrix(heart_diseaseTest, result_predict)
    confusion_matrices.append(cm)

    acc = roc_auc_score(heart_diseaseTest, result_predict)
    acc_svm.append(acc)
    avg_accuracy += acc

report = classification_report(
    heart_diseaseTest,
    result_predict,
    target_names=['No Heart Disease', 'Heart Disease'],
)

```

```

        output_dict=True
    )

    print(f"\nTập k thứ: {fold} :")
    for label in ['No Heart Disease', 'Heart Disease']:
        print(f"{label}: Precision: {report[label]['precision']:.2f},
Recall: {report[label]['recall']:.2f}, F1-Score: {report[label]['f1-
score']:.2f}, Support: {int(report[label]['support'])}")

    print(f"Mức độ chính xác cho tập K thứ {fold}: {acc:.2f}")

avg_accuracy = avg_accuracy / 5
print("Tỉ lệ chính xác trung bình: " + str(avg_accuracy))

fig, axes = plt.subplots(2, 3, figsize=(18, 10))
axes = axes.flatten()

for i, cm in enumerate(confusion_matrices):
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', ax=axes[i],
                xticklabels=['No Heart Disease', 'Heart Disease'],
                yticklabels=['No Heart Disease', 'Heart Disease'])
    axes[i].set_title(f"Tập k thứ {i}")
    axes[i].set_xlabel("Predicted")
    axes[i].set_ylabel("Actual")

for j in range(len(confusion_matrices), len(axes)):
    fig.delaxes(axes[j])

plt.tight_layout()
plt.show()

```

_Tiền xử lí dữ liệu

- Bỏ cột RestingECG
- Chuyển cột Sex và ExerciseAngina về dạng số [1, 0]

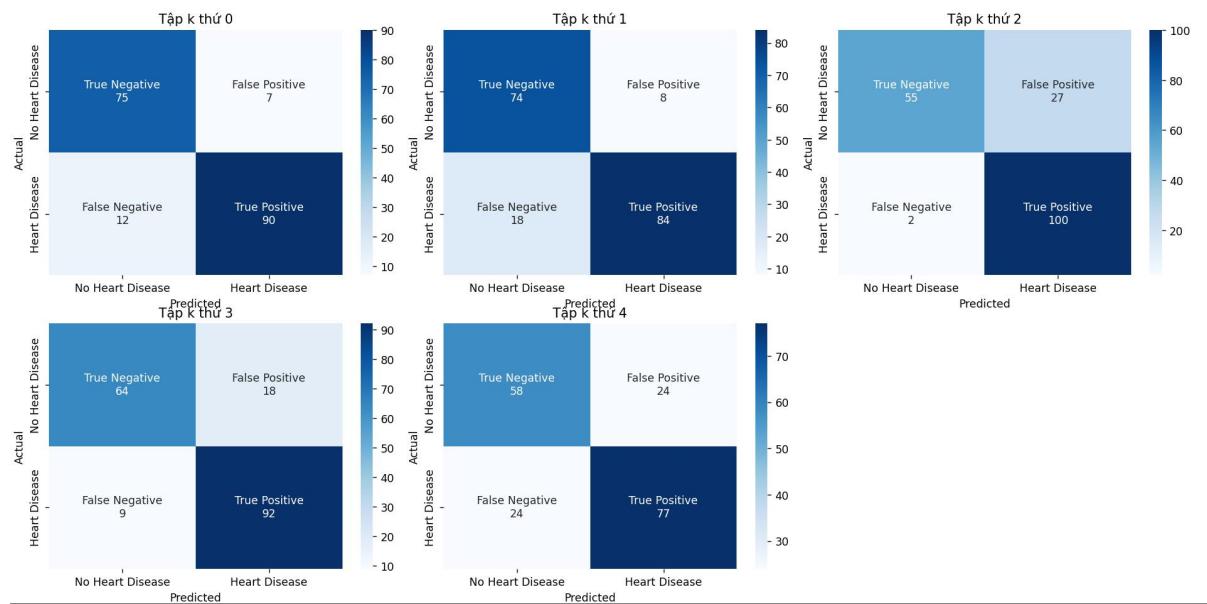
- Lấy các giá trị trong cột ChestPainType, ST_Slope tạo thành các cột với tên là cột mới là giá trị của nó trong cột ChestPainType và ST_Slope.
- Sử dụng **StratifiedKFold** để tách dữ liệu thành 5 tập nhưng vẫn giữ nguyên tỉ lệ dữ liệu giữa các tập.

_ Duyệt qua từng tập k:

- Lấy giá trị mục tiêu gán vào heart_diseaseTrain và heart_diseaseTest và loại bỏ giá trị mục tiêu khỏi dữ liệu huấn luyện và thử nghiệm.
- Áp dụng StandardScaler lên dữ liệu huấn luyện và thử nghiệm để chuẩn hóa dữ liệu (Standardization).
- Áp dụng mô hình SVM tuyến tính với C được mặc định là 0 tức là mô hình huấn luyện sẽ tập trung vào việc tối đa margin chứ không quan tâm đến mức độ hy sinh.
- Tính toán độ chính xác của dữ liệu: recal, precision, F1 Score.
- Vẽ confusion matrix cho mỗi tập dữ liệu

_ Tính độ chính xác trung bình của dữ liệu để đánh giá mô hình:

d) Kết quả và đánh giá mô hình



Tập k thứ: 0 :

No Heart Disease: Precision: 0.86, Recall: 0.91, F1-Score: 0.89, Support: 82

Heart Disease: Precision: 0.93, Recall: 0.88, F1-Score: 0.90, Support: 102

Mức độ chính xác cho tập K thứ 0: 0.90

Tập k thứ: 1 :

No Heart Disease: Precision: 0.80, Recall: 0.90, F1-Score: 0.85, Support: 82

Heart Disease: Precision: 0.91, Recall: 0.82, F1-Score: 0.87, Support: 102

Mức độ chính xác cho tập K thứ 1: 0.86

Tập k thứ: 2 :

No Heart Disease: Precision: 0.96, Recall: 0.67, F1-Score: 0.79, Support: 82

Heart Disease: Precision: 0.79, Recall: 0.98, F1-Score: 0.87, Support: 102

Mức độ chính xác cho tập K thứ 2: 0.83

Tập k thứ: 3 :

No Heart Disease: Precision: 0.88, Recall: 0.78, F1-Score: 0.83, Support: 82

Heart Disease: Precision: 0.84, Recall: 0.91, F1-Score: 0.87, Support: 101

Mức độ chính xác cho tập K thứ 3: 0.85

Tập k thứ: 4 :

No Heart Disease: Precision: 0.71, Recall: 0.71, F1-Score: 0.71, Support: 82

Heart Disease: Precision: 0.76, Recall: 0.76, F1-Score: 0.76, Support: 101

Mức độ chính xác cho tập K thứ 4: 0.73

Tỉ lệ chính xác trung bình: 0.8335151592634155

Ta nhận thấy ở các tập dữ liệu từ 0 đến 3 thì mức độ dự đoán đúng của dữ liệu dựa trên giá trị F1-Score khá cao trong khoảng từ 0.87 đến 0.9 và tỉ lệ chính xác thì trong khoảng từ 0.83 đến 0.9. Trong khi đó tập dữ liệu thứ 4 lại cho ra kết quả khá thấp với F1-Score là 0.76 và độ chính xác là 0.73 khiến cho tỉ lệ chính xác chung của mô hình giảm chỉ còn 83.35%. Và với giá trị C hiện tại là 0.001 tức là ta tập trung rất nhiều vào việc tối ưu Margin hơn là giảm thiểu sự hy sinh các điểm dữ liệu. Nếu ta tăng dần C lên ta sẽ có:

Khi C=0.01 mức độ chính trung bình khoảng: 82.76%.

Khi C=0.1 mức độ chính xác trung bình: 82.3%.

Khi C= 1 mức độ chính là trung bình: 82.89%.

Khi C=10 mức độ chính xác trung bình: 82.89%.

Khi C= 100 mức độ chính trung bình: 82.89%

Khi C= 1000 mức độ chính xác trung bình: 82.89%

Khi C= 1000 mức độ chính xác trung bình: 82.89%

Khi $C = 10000$ mức độ chính xác trung bình: 82.89%

Nhận xét:

- Thuật toán Soft margin có thể áp dụng vào các tập dữ liệu thực và đưa ra kết quả chính xác khá cao.
- Khi C tăng ta thấy được mức độ chính xác có thể tăng hoặc giảm nhưng không quá lớn. Và việc tăng C cũng có giới như khi $C = 1$ đến 10000 giá trị gần như không đổi, nhưng thuật toán tốn rất nhiều thời gian để giảm thiểu sự sinh các điểm dữ liệu nhưng việc đó thường như là không thể nên dẫn đến tỉ lệ chính xác thường như không đổi mặc dù thay đổi hướng đi của thuật toán.

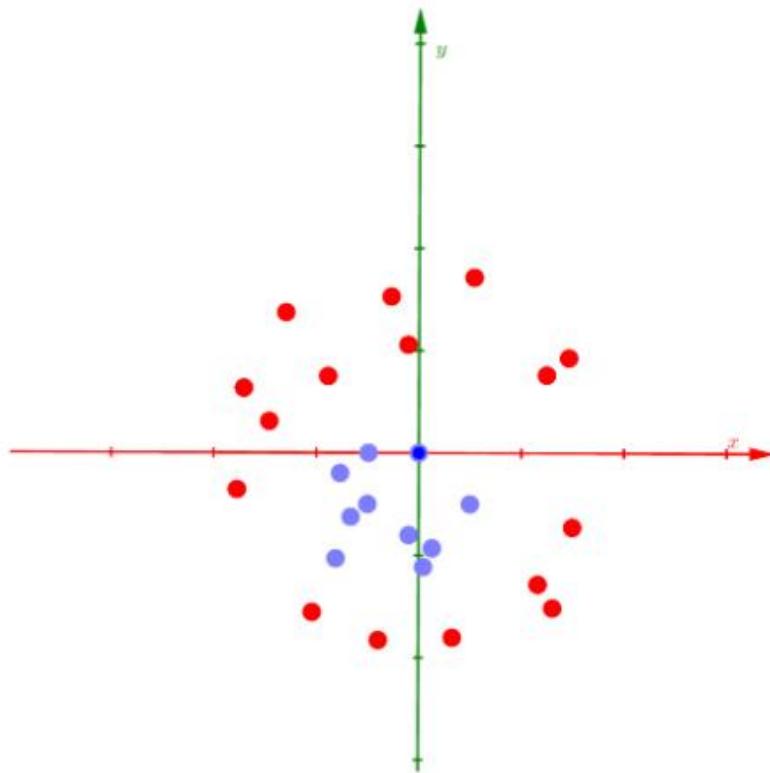
Vậy hằng số C cần phải được điều chỉnh hợp lý để phù hợp với tập dữ liệu từ đó mới có được mức độ chính xác cao.

- Nếu C cao (10, 100, 1000...) có thể sẽ dẫn đến overfitting do mức độ phức tạp cũng tăng cao dẫn đến mô hình làm việc tốt với dữ liệu huấn luyện nhưng đưa ra kết quả không tốt cho dữ liệu kiểm tra.
- Khi C thấp (0.1, 0.01, 0.001...) ta sẽ chấp nhận bỏ qua nhiều giá trị hơn nên mức độ phức tạp giảm từ đó giảm đi mức độ overfitting của mô hình nhưng nếu quá nhỏ mô hình sẽ quá đơn giản và underfitting. Underfitting có nghĩa là khi mô hình quá đơn giản nó sẽ không học đủ sâu để hiểu rõ dữ liệu từ đó đưa ra kết quả không tốt trên cả dữ liệu huấn luyện và dữ liệu kiểm tra.

Do tỉ lệ chính xác vẫn chưa đủ cao nên Soft margin vẫn chưa đủ phù hợp với đa số các tập dữ liệu thực tế khi mối quan hệ giữa các giá trị đầu vào và đầu ra rất phức tạp và có thể không phải là mối quan hệ tuyến tính, và còn xuất hiện nhiều giá trị ngoại lệ.

2.2) SVM phân loại dữ liệu phi tuyến tính

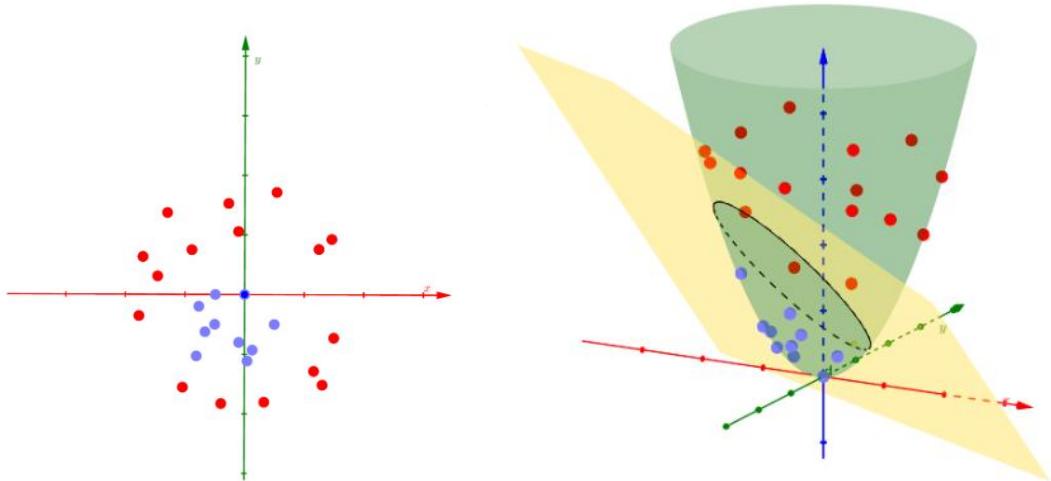
_Dữ liệu phi tuyến tính là dữ liệu mà mối quan hệ giữa giá trị đầu vào và giá trị đầu ra không thể được biểu diễn bằng một đường thẳng hay một siêu phẳng trong không gian nhiều chiều. Ví dụ như hình bên dưới:



2.3 Kernel

2.3.1) Kernel là gì

_Kernel là một hàm số biến đổi dữ liệu x từ không gian ban đầu thành không gian dữ liệu mới bằng hàm số $\Phi()$. Các hàm $\Phi()$ thường sẽ tạo ra dữ liệu với số chiều cao hơn số chiều của dữ liệu ban đầu, thậm chí là vô hạn chiều với mối quan hệ giữa giá trị đầu vào và đầu ra gần như là mối quan hệ tuyến tính và khi đó ta có thể dùng Soft margin để đưa ra được kết quả tốt hơn.



_ Ban đầu dữ liệu đầu vào và đầu ra không thể được chia cắt bằng bát cứ đường thẳng nào nhưng khi sử dụng kernel ta sẽ tăng số chiều từ 2 lên 3 và khi đó ta có thể dùng 1 siêu phẳng để thể hiện mối quan hệ giữa giá trị đầu vào và đầu ra.

2.3.2) Cơ sở lý thuyết

Ở đây ta có hàm mục tiêu của Soft margin là:

$$\text{Hàm mục tiêu: } \max_{\alpha} L_{dual} = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j x_i^T x_j \quad (1)$$

$$\text{Ràng buộc: } 0 \leq \alpha_i \leq C, \forall x_i \in D \text{ và } \sum_{i=1}^n \alpha_i y_i = 0 \quad (2)$$

Sau khi giải được α_i ta sẽ tìm được

$$w^T x + b = \sum_{i=1}^{m \in S} \alpha_i y_i x_i^T x + \frac{1}{m} \sum_{i=1}^{m \in S} \left(y_i - \sum_{i=1}^{m \in S} \alpha_i y_i x_i^T x_i \right) \quad (3)$$

Khi ta áp dụng Kernel vào thì ta sẽ dùng hàm $\Phi()$ để chuyển x thành giá trị của hàm $\Phi(x)$ thì khi đó ta sẽ được hàm mục tiêu và ràng buộc mới là:

$$\text{Hàm mục tiêu: } \max_{\alpha} L_{dual} = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \Phi(x_i)^T \Phi(x_j) \quad (4)$$

$$\text{Ràng buộc: } 0 \leq \alpha_i \leq C, \forall x_i \in D \text{ và } \sum_{i=1}^n \alpha_i y_i = 0 \quad (5)$$

Sau khi giải được α_i ta sẽ tìm được

$$w^T \Phi(x) + b = \sum_{i=1}^{m \in S} \alpha_i y_i \Phi(x_i)^T \Phi(x_j) + \frac{1}{m} \sum_{i=1}^{m \in S} \left(y_i - \sum_{i=1}^{m \in S} \alpha_i y_i \Phi(x_i)^T \Phi(x_j) \right) \quad (6)$$

_Việc tính toán trực tiếp $\Phi(x)$ cho mỗi điểm dữ liệu là vô cùng tốn thời gian và bộ nhớ, vì số chiều của $\Phi(x)$ có thể rất lớn. Hơn nữa để xác định một điểm x mới thuộc lớp nào thì ta phải biến đổi nó sang $\Phi(x)$ rồi tích vô hướng nó với tất cả các $\Phi(x)_m$ trong tập support vector.

_Trong biểu thức (4) và (6) chúng ta không cần trực tiếp tính $\Phi(x)$ cho mọi điểm dữ liệu mà chúng ta chỉ cần tính được tích vô hướng của $\Phi(a)^T$ và $\Phi(b)$ dựa trên 2 điểm dữ liệu a, b bất kỳ và kỹ thuật này gọi là kernel trick và các phương pháp dựa trên kỹ thuật này được gọi là kernel method. Lúc này bằng cách định nghĩa:

$$k(a, b) = \Phi(a)^T \Phi(b)$$

Ta sẽ được biểu thức (4) và (6) mới

$$\textbf{Hàm mục tiêu: } \max_{\alpha} L_{dual} = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

$$\textbf{Ràng buộc: } 0 \leq \alpha_i \leq C, \forall x_i \in D \text{ và } \sum_{i=1}^n \alpha_i y_i = 0 \quad (5)$$

Sau khi giải được α_i ta sẽ tìm được

$$w^T \Phi(x) + b = \sum_{i=1}^{m \in S} \alpha_i y_i k(x_i, x_j) + \frac{1}{m} \sum_{i=1}^{m \in S} \left(y_i - \sum_{i=1}^{m \in S} \alpha_i y_i k(x_i, x_j) \right)$$

2.3.3) Tính chất của các hàm Kernel

Không phải hàm kernel nào cũng có thể sử dụng được. Mà hàm đó cần phải thỏa các tính chất:

- Đổi xứng: $k(a, b) = k(b, a)$
- Dương bán xác định: $\sum_{i=1}^n \sum_{j=1}^n k(x_m, x_n) c_m c_n \geq 0, \forall x \in \mathbb{R}, i = 1, 2, \dots, N$

Việc đảm bảo Kernel hợp lệ để:

- Dữ liệu được ánh xạ đúng từ không gian gốc sang không gian mới nhiều chiều hơn.
- Có nghiệm cho bài toán tối ưu.
- Đảm bảo các phép tính như nghịch đảo ma trận, phân rã ma trận được thực hiện mà không bị lỗi.

_ Đảm bảo rằng $k(a, b)$ luôn đối xứng do Kernel thường giả định $k(a, b) = k(b, a)$. Nếu giả định sai sẽ gây ra lỗi trong thuật toán kernel.

2.3.4) Các hàm kernel thông dụng

Linear kernel

$$k(x, y) = x^T y$$

_ Kernel này đơn giản là tích vô hướng của x^T và y . Phù hợp cho các bài toán mà dữ liệu có thể được phân tách tuyến tính trong không gian gốc hoặc khi không cần ánh xạ dữ liệu sang một không gian đặc trưng cao hơn

Polynomial kernel

$$k(x, y) = (r + \gamma x^T y)^d$$

_ Với d là một số dương chỉ bậc của đa thức. d có thể không là số tự nhiên vì mục đích chính của ta không phải là bậc của đa thức mà là cách tính kernel. Polynomial kernel có thể dùng để mô tả hầu hết các đa thức có bậc không vượt quá d nếu d là một số tự nhiên.

Radial Basic Function (RBF)

$$k(x, y) = \exp\{-\gamma \|x - y\|^2\}$$

_ Với $\sigma > 0$ là tham số phân tán đóng vai trò giống như độ lệch chuẩn trong hàm mật độ chuẩn. Đây cũng là hàm Kernel được dùng nhiều nhất trong thực tế.

2.4) Áp dụng các kernel vào SVM

2.4.1) Polynomial Kernel

a) Cơ sở lý thuyết

Ta có công thức tổng quát với $k(x_i, x_j)$ là:

$$\text{Hàm mục tiêu: } \max_{\alpha} L_{dual} = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j k(x_i, x_j) \quad (1)$$

$$\text{Ràng buộc: } 0 \leq \alpha_i \leq C, \forall x_i \in D \text{ và } \sum_{i=1}^n \alpha_i y_i = 0 \quad (2)$$

Sau khi giải được α_i ta sẽ tìm được

$$w^T \Phi(x) + b = \sum_{i=1}^{m \in S} \alpha_i y_i k(x_i, x_j) + \frac{1}{m} \sum_{i=1}^{m \in S} (y_i - \sum_{i=1}^{m \in S} \alpha_i y_i k(x_i, x_j)) \quad (3)$$

Với Polynomial có $k(x_i, x_j) = (r + \gamma x_i^T x_j)^d$ thay vào (1), (2), (3) ta được:

$$\text{Hàm mục tiêu: } \max_{\alpha} L_{dual} = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j (r + \gamma x_i^T x_j)^d$$

Ràng buộc: $0 \leq \alpha_i \leq C, \forall x_i \in D$ và $\sum_{i=1}^n \alpha_i y_i = 0$

$$w^T \Phi(x) + b = \sum_{i=1}^{m \in S} \alpha_i y_i (r + \gamma x_i^T x_j)^d + \frac{1}{m} \sum_{i=1}^{m \in S} (y_i - \sum_{i=1}^{m \in S} \alpha_i y_i (r + \gamma x_i^T x_j)^d)$$

b) Áp dụng thuật toán

```
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report, roc_auc_score,
confusion_matrix
from sklearn.preprocessing import StandardScaler, LabelEncoder
import pandas as pd
from sklearn import model_selection
from sklearn.svm import SVC

acc_svm = []
confusion_matrices = []
le = LabelEncoder()
df = pd.read_csv("heart.csv")
df = df.drop(columns=["RestingECG"])
df["Sex"] = le.fit_transform(df["Sex"])
df["ExerciseAngina"] = le.fit_transform(df["ExerciseAngina"])
df = pd.get_dummies(df, columns=['ChestPainType'], prefix='', prefix_sep=' ')
df = pd.get_dummies(df, columns=['ST_Slope'], prefix='', prefix_sep=' ')
target = "HeartDisease"
feature = df.columns.to_list()
feature.remove(target)
y = df[target].values

kf = model_selection.StratifiedKFold(n_splits=5)
avg_accuracy = 0
```

```

for fold, (train, value) in enumerate(kf.split(X=df, y=y)):
    X_train = df.loc[train, feature]
    heart_diseaseTrain = df.loc[train, target]
    X_valid = df.loc[value, feature]
    heart_diseaseTest = df.loc[value, target]

    ss = StandardScaler()
    X_train = ss.fit_transform(X_train)
    X_valid = ss.transform(X_valid)

    model_classification =
SVC(kernel="poly", degree=3, gamma='scale', coef0=3, C=0.001,
probability=True)
    model_classification.fit(X_train, heart_diseaseTrain)
    prob_predict = model_classification.predict_proba(X_valid)[:, 1]
    result_predict = model_classification.predict(X_valid)

    cm = confusion_matrix(heart_diseaseTest, result_predict)
    confusion_matrices.append(cm)

    acc = roc_auc_score(heart_diseaseTest, prob_predict)
    acc_svm.append(acc)
    avg_accuracy += acc

    report = classification_report(
        heart_diseaseTest,
        result_predict,
        target_names=['No Heart Disease', 'Heart Disease'],
        output_dict=True
    )

    print(f"\nTập k thứ: {fold} :")
    for label in ['No Heart Disease', 'Heart Disease']:

```

```

        print(f"\{label\}: Precision: {report[label]['precision']:.2f},"
Recall: {report[label]['recall']:.2f}, F1-Score: {report[label]['f1-
score']:.2f}, Support: {int(report[label]['support'])}"))
    print(f"Mức độ chính xác cho tập K thứ {fold}: {acc:.2f}")

avg_accuracy = avg_accuracy / 5
print("Tỉ lệ chính xác trung bình: " + str(avg_accuracy))

# Vẽ confusion matrices
fig, axes = plt.subplots(2, 3, figsize=(18, 10))
axes = axes.flatten()

labels_text = [['True Negative', 'False Positive'],
               ['False Negative', 'True Positive']]

for i, cm in enumerate(confusion_matrices):
    total = cm.sum()

    annotations = [[f"\{labels_text[row][col]}\n{cm[row, col]}"
                    for col in range(2)] for row in range(2)]

    sns.heatmap(cm, annot=annotations, fmt='', cmap='Blues', ax=axes[i],
                xticklabels=['No Heart Disease', 'Heart Disease'],
                yticklabels=['No Heart Disease', 'Heart Disease'])
    axes[i].set_title(f"Tập k thứ {i}")
    axes[i].set_xlabel("Predicted")
    axes[i].set_ylabel("Actual")

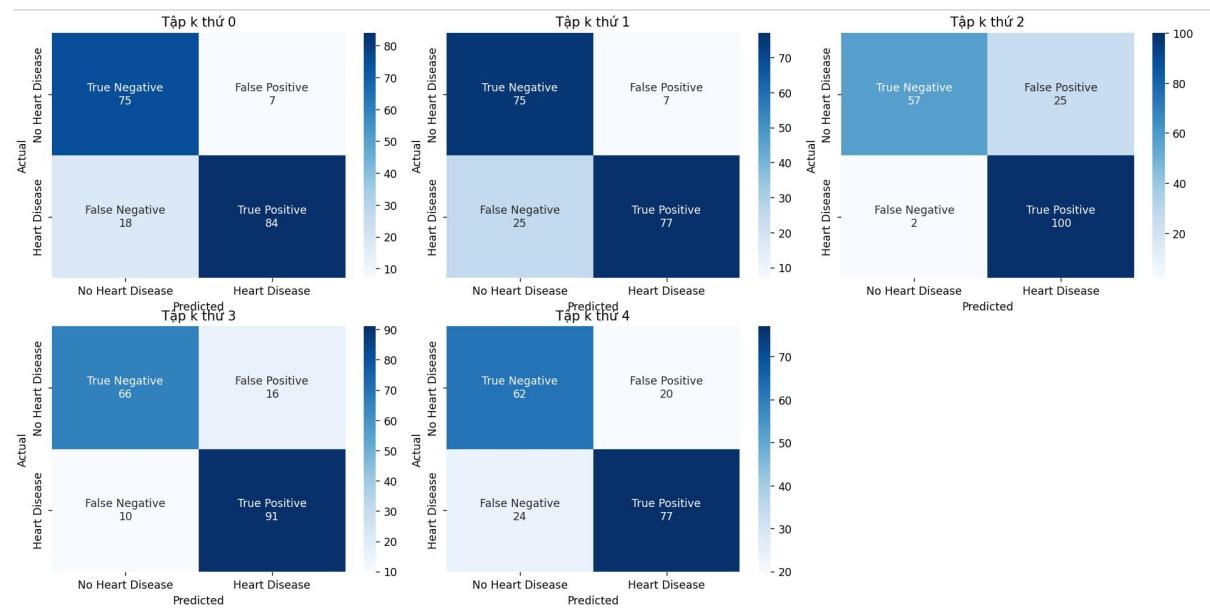
for j in range(len(confusion_matrices), len(axes)):
    fig.delaxes(axes[j])

plt.tight_layout()
plt.show()

```

_ Code cũng khá tương tự như soft margin nhưng mô hình huấn luyện đã chọn là Polynomial với $d=3$ (bậc của đa thức), gamma chính là γ trong công thức (có 2 kiểu mặc định là scale và auto hoặc có thể tự định nghĩa), $C=0.001$ (hàng số điều chỉnh mức độ của mức độ hy sinh), coef_0 chính là hệ số r trong công thức.

c) Kết quả và đánh giá mô hình



Tập k thứ: 0 :

No Heart Disease: Precision: 0.81, Recall: 0.91, F1-Score: 0.86, Support: 82
Heart Disease: Precision: 0.92, Recall: 0.82, F1-Score: 0.87, Support: 102
Mức độ chính xác cho tập K thứ 0: 0.95

Tập k thứ: 1 :

No Heart Disease: Precision: 0.75, Recall: 0.91, F1-Score: 0.82, Support: 82
Heart Disease: Precision: 0.92, Recall: 0.75, F1-Score: 0.83, Support: 102
Mức độ chính xác cho tập K thứ 1: 0.95

Tập k thứ: 2 :

No Heart Disease: Precision: 0.97, Recall: 0.70, F1-Score: 0.81, Support: 82
Heart Disease: Precision: 0.80, Recall: 0.98, F1-Score: 0.88, Support: 102
Mức độ chính xác cho tập K thứ 2: 0.92

Tập k thứ: 3 :

No Heart Disease: Precision: 0.87, Recall: 0.80, F1-Score: 0.84, Support: 82
Heart Disease: Precision: 0.85, Recall: 0.90, F1-Score: 0.88, Support: 101
Mức độ chính xác cho tập K thứ 3: 0.90

Tập k thứ: 4 :

No Heart Disease: Precision: 0.72, Recall: 0.76, F1-Score: 0.74, Support: 82
Heart Disease: Precision: 0.79, Recall: 0.76, F1-Score: 0.78, Support: 101
Mức độ chính xác cho tập K thứ 4: 0.84

Tỉ lệ chính xác trung bình: 0.9121385381005819

_Mức độ chính xác trung bình của thuật toán đã lên thành 91,21% cao hơn so với Soft margin khi không có kernel. Vậy việc sử dụng Polynomial kernel để hỗ trợ cho SVM giúp cho việc phân loại dữ liệu đa số chính xác cao hơn. Việc sử dụng kernel để chuyển dữ liệu từ phi tuyến về tuyến tính hoặc gần tuyến tính sẽ cho ra mức độ chính xác cao hơn, do dữ liệu đầu vào và đầu ra đơn giản là mối quan hệ tuyến tính.

_ Các giá trị d thường thấy là 2 và 3 sẽ cho ra tỉ lệ chính xác khá cao, và khi tăng d cao mô hình sẽ phức tạp hơn và có thể bị overfitting.

_Với γ có các giá trị mặc định là:

$$1. \text{ "scale": } \gamma = \frac{1}{n \cdot Var(x)}$$

Trong đó:

- n số điểm trong tập dữ liệu
- $Var(x)$ là phương sai trung bình của tập dữ liệu

$$2. \text{ "auto": } \gamma = \frac{1}{n}$$

Trong đó:

- $\frac{1}{n}$ (với n là số điểm trong tập dữ liệu)

3. Hoặc ta có thể điều chỉnh γ bằng một giá trị bất kỳ ta muốn.

_ Khi γ lớn thì mức độ ảnh hưởng của $\gamma x^T y$ sẽ lớn nên nó sẽ khuếch đại sự khác biệt giữa các điểm dữ liệu, khiến việc tạo margin trở nên phức tạp. Nhưng nếu quá lớn mô hình sẽ bị overfitting. Vậy khi nào dùng “auto”, “scale” hay là giá trị bất kỳ thì phù hợp:

- “auto”: khá dễ sử dụng do sự đơn giản thích hợp với các tập dữ liệu đơn giản và nhỏ.
- “scale”: phức tạp hơn so với “auto” nhưng phù hợp với các tập dữ liệu phức tạp do có khả năng tự điều chỉnh tốt hơn so với “auto”.
- Giá trị bất kỳ: chỉ sử dụng khi ta đã hiểu rõ về mô hình và biết rằng việc nên để giá trị bao nhiêu là phù hợp.

_ C được chọn cũng dựa trên mục tiêu tương tự như Soft margin ở trên.

_ Với $coef0$ hay là r quyết định mức độ ảnh hưởng của thành phần phi tuyến. Khi r nhỏ hoặc bằng 0 thì mô hình của ta sẽ ít phức tạp hơn phù hợp với các tập dữ liệu tuyến tính hoặc gần tuyến tính. Khi giá trị r lớn ta sẽ quan tâm tới phần phi tuyến nhiều hơn thường dùng cho dữ liệu phức tạp, phi tuyến tính khá nhiều. Khi tăng r càng lớn mô hình sẽ dễ bị overfitting do quá phức tạp.

2.4.2) Gaussian RBF Kernel

a) Cơ sở lý thuyết

Ta có công thức tổng quát với $k(x_i, x_j)$ là:

$$\text{Hàm mục tiêu: } \max_{\alpha} L_{dual} = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j k(x_i, x_j) \quad (1)$$

$$\text{Ràng buộc: } 0 \leq \alpha_i \leq C, \forall x_i \in D \text{ và } \sum_{i=1}^n \alpha_i y_i = 0 \quad (2)$$

Sau khi giải được α_i ta sẽ tìm được

$$w^T \Phi(x) + b = \sum_{i=1}^m \alpha_i y_i k(x_i, x) + \frac{1}{m} \sum_{i=1}^m (\gamma_i - \sum_{i=1}^m \alpha_i y_i k(x_i, x)) \quad (3)$$

Cũng tương tự như Polynomial ta có $k(x_i, x_j) = \exp\{-\gamma \|x - y\|^2\}$ thay vào (1), (2), (3) ta có:

Hàm mục tiêu: $\max_{\alpha} L_{dual} = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \exp \{-\gamma \|x - y\|^2\}$

Ràng buộc: $0 \leq \alpha_i \leq C, \forall x_i \in D$ và $\sum_{i=1}^n \alpha_i y_i = 0$

$$w^T \Phi(x) + b = \sum_{i=1}^m \alpha_i y_i k(x_i, x_j) + \frac{1}{m} \sum_{i=1}^m (y_i - \sum_{i=1}^m \alpha_i y_i \exp \{-\gamma \|x - y\|^2\})$$

b) Áp dụng thuật toán

```
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report, roc_auc_score,
confusion_matrix
from sklearn.preprocessing import StandardScaler, LabelEncoder
import pandas as pd
from sklearn import model_selection
from sklearn.svm import SVC

acc_svm = []
confusion_matrices = []
le = LabelEncoder()
df = pd.read_csv("heart.csv")
df = df.drop(columns=["RestingECG"])
df["Sex"] = le.fit_transform(df["Sex"])
df["ExerciseAngina"] = le.fit_transform(df["ExerciseAngina"])
df = pd.get_dummies(df, columns=['ChestPainType'], prefix='', prefix_sep=' ')
df = pd.get_dummies(df, columns=['ST_Slope'], prefix='', prefix_sep=' ')
target = "HeartDisease"
feature = df.columns.to_list()
feature.remove(target)
y = df[target].values

kf = model_selection.StratifiedKFold(n_splits=5)
avg_accuracy = 0

for fold, (train, value) in enumerate(kf.split(X=df, y=y)):
```

```

X_train = df.loc[train, feature]
heart_diseaseTrain = df.loc[train, target]
X_valid = df.loc[value, feature]
heart_diseaseTest = df.loc[value, target]

ss = StandardScaler()
X_train = ss.fit_transform(X_train)
X_valid = ss.transform(X_valid)

model_classification = SVC(kernel="rbf", gamma="scale", C=0.01,
probability=True)
model_classification.fit(X_train, heart_diseaseTrain)
prob_predict = model_classification.predict_proba(X_valid)[:, 1]
result_predict = model_classification.predict(X_valid)

cm = confusion_matrix(heart_diseaseTest, result_predict)
confusion_matrices.append(cm)

acc = roc_auc_score(heart_diseaseTest, prob_predict)
acc_svm.append(acc)
avg_accuracy += acc

report = classification_report(
    heart_diseaseTest,
    result_predict,
    target_names=['No Heart Disease', 'Heart Disease'],
    output_dict=True
)

print(f"\nTập k thứ: {fold} :")
for label in ['No Heart Disease', 'Heart Disease']:
    print(f"{label}: Precision: {report[label]['precision']:.2f},"
Recall: {report[label]['recall']:.2f}, F1-Score: {report[label]['f1-score']:.2f}, Support: {int(report[label]['support'])}"))

```

```

print(f"Mức độ chính xác cho tập K thứ {fold}: {acc:.2f}")

avg_accuracy = avg_accuracy / 5
print("Tỉ lệ chính xác trung bình: " + str(avg_accuracy))

# Vẽ confusion matrices
fig, axes = plt.subplots(2, 3, figsize=(18, 10))
axes = axes.flatten()

labels_text = [['True Negative', 'False Positive'],
               ['False Negative', 'True Positive']]

for i, cm in enumerate(confusion_matrices):
    total = cm.sum()

    annotations = [[f'{labels_text[row][col]}\n{cm[row, col]}'
                    for col in range(2)] for row in range(2)]

    sns.heatmap(cm, annot=annotations, fmt='', cmap='Blues', ax=axes[i],
                xticklabels=['No Heart Disease', 'Heart Disease'],
                yticklabels=['No Heart Disease', 'Heart Disease'])

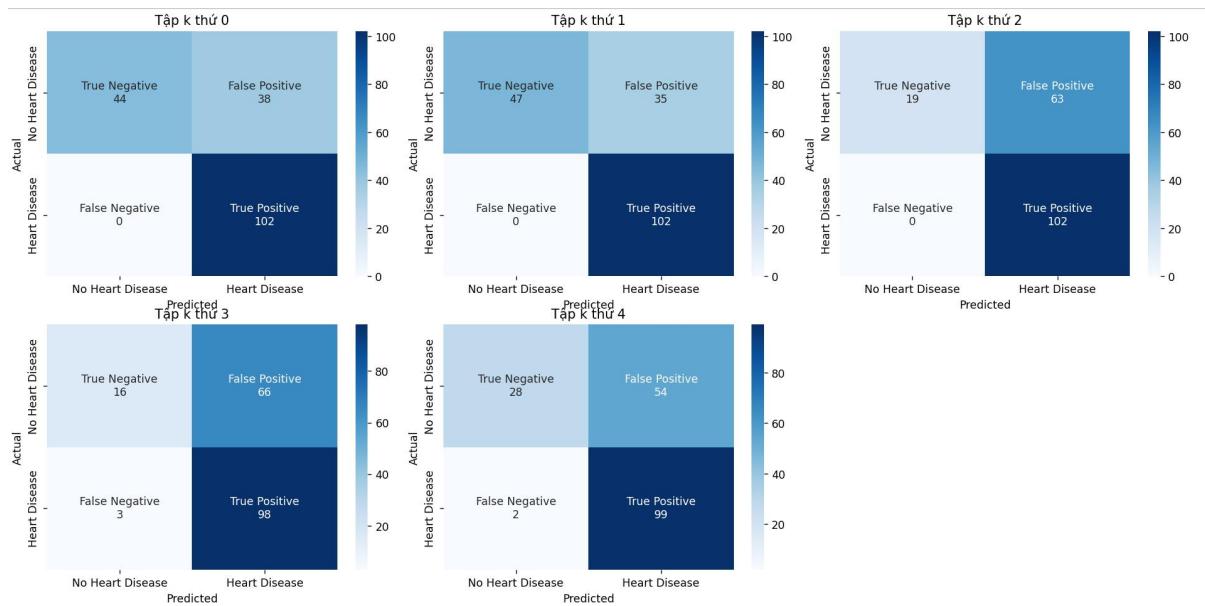
    axes[i].set_title(f"Tập k thứ {i}")
    axes[i].set_xlabel("Predicted")
    axes[i].set_ylabel("Actual")

for j in range(len(confusion_matrices), len(axes)):
    fig.delaxes(axes[j])

plt.tight_layout()
plt.show()

```

c) Kết quả và đánh giá mô hình



Tập k thứ: 0 :

No Heart Disease: Precision: 1.00, Recall: 0.54, F1-Score: 0.70, Support: 82
 Heart Disease: Precision: 0.73, Recall: 1.00, F1-Score: 0.84, Support: 102
 Mức độ chính xác cho tập K thứ 0: 0.96

Tập k thứ: 1 :

No Heart Disease: Precision: 1.00, Recall: 0.57, F1-Score: 0.73, Support: 82
 Heart Disease: Precision: 0.74, Recall: 1.00, F1-Score: 0.85, Support: 102
 Mức độ chính xác cho tập K thứ 1: 0.95

Tập k thứ: 2 :

No Heart Disease: Precision: 1.00, Recall: 0.23, F1-Score: 0.38, Support: 82
 Heart Disease: Precision: 0.62, Recall: 1.00, F1-Score: 0.76, Support: 102
 Mức độ chính xác cho tập K thứ 2: 0.92

Tập k thứ: 3 :

No Heart Disease: Precision: 0.84, Recall: 0.20, F1-Score: 0.32, Support: 82
 Heart Disease: Precision: 0.60, Recall: 0.97, F1-Score: 0.74, Support: 101
 Mức độ chính xác cho tập K thứ 3: 0.89

Tập k thứ: 4 :

No Heart Disease: Precision: 0.93, Recall: 0.34, F1-Score: 0.50, Support: 82
 Heart Disease: Precision: 0.65, Recall: 0.98, F1-Score: 0.78, Support: 101
 Mức độ chính xác cho tập K thứ 4: 0.86
 Tỉ lệ chính xác trung bình: 0.9149551827492649

Mức độ chính xác trung bình 91.5% khá cao cao hơn so với Polynomial 1 ít và cao hơn khá nhiều so với soft margin thông thường. Nên việc sử dụng RBF kernel lên dữ liệu để ánh xạ

dữ liệu sang một không gian khác sẽ tốt hơn soft margin thông thường và RBF cũng đưa ra kết quả tốt hơn Polynomial.

Chương 4: Kết luận

Qua việc sử dụng Support vector machine để huấn luyện mô hình và kiểm tra ta thấy:

_ Hard Margin không phù hợp với tập dữ liệu do có nhiều dữ liệu nhiễu, điều mà khiến cho việc tìm ra margin chia cắt hoàn hảo dữ liệu thành 2 loại là hoàn toàn không thể.

_ Soft Margin tuy có thể áp dụng vào tập dữ liệu nhưng độ chính xác chưa cao, và hằng số C được thiết lập bởi người dùng cần phải được lựa chọn hợp lý để cân bằng giữa việc tối thiểu việc hy sinh các điểm dữ liệu và việc tối ưu margin.

_ Sử dụng kernel để hỗ trợ thuật toán Soft Margin là cần thiết do mối quan hệ giữa dữ liệu đầu vào và ra không đơn giản chỉ là tuyến tính hay gần tuyến tính. Nên kernel cần thiết để ánh xạ dữ liệu đó sang không gian khác sao cho dữ liệu gần tuyến tính để có thể áp dụng.

_ Khi kết hợp Soft Margin và Polynomial kết quả đưa ra tăng lên khá cao từ 83.35% thành 91.21%.

_ Còn khi dùng RBF kernel thì kết quả tốt hơn là 91.5% tuy không cao hơn Polynomial bao nhiêu nhưng khi dữ liệu lớn thì sự tăng này cũng rất quan trọng.

Nguồn tài liệu tham khảo

1. Giảng viên Từ Tuyết Hùng, bài giảng học phần Học máy, TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM
2. Sách tài liệu Aurelien Geron - Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow-O'reilly (2019)
3. <https://machinelearningcoban.com/>