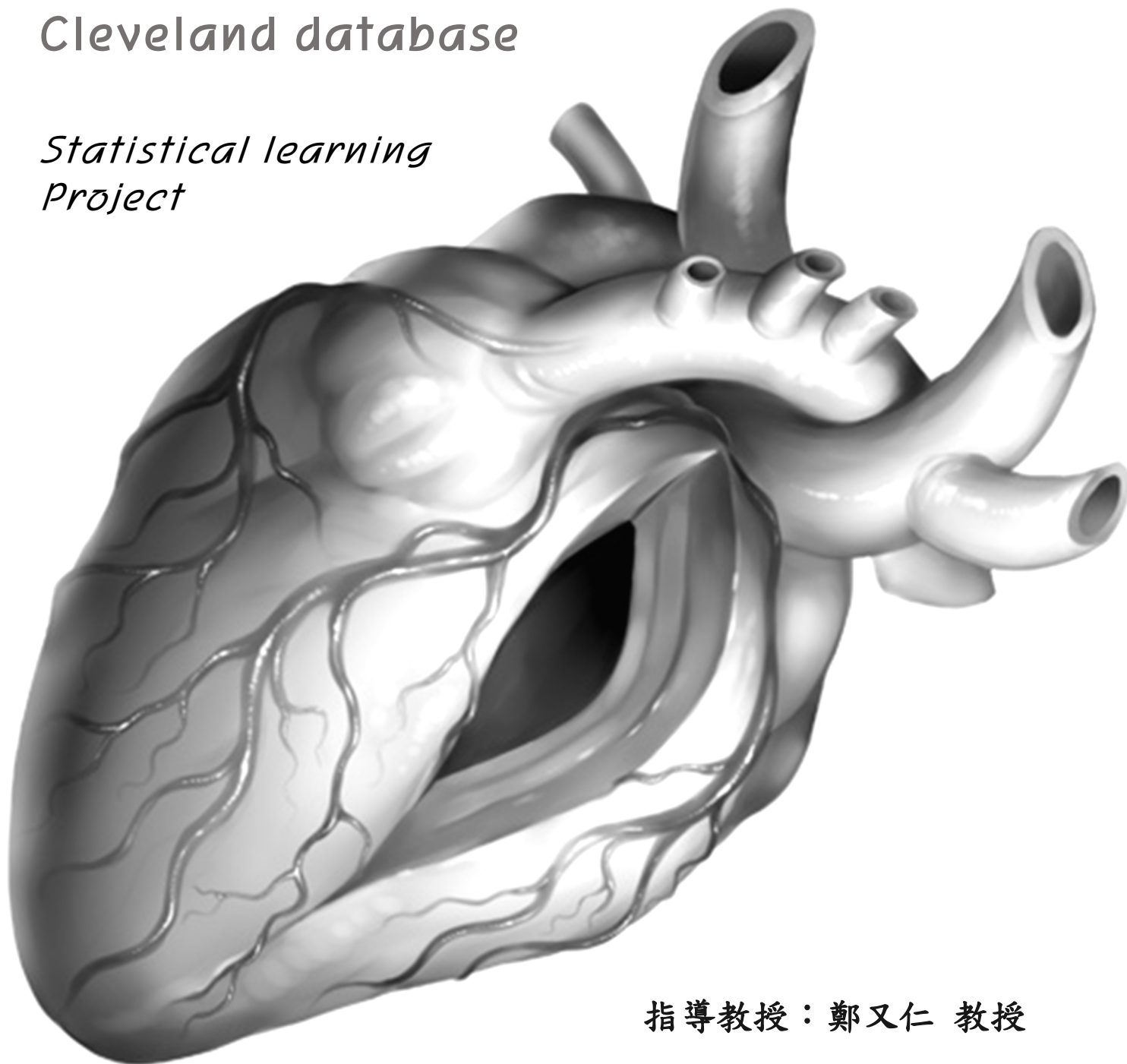


# HEART DISEASE EXPLORATION – Cleveland database

*Statistical learning  
Project*



指導教授：鄭又仁 教授

學生：蘇柏庄、魏志宇、吳岱錡、陳冠維、陳威宇

# 目錄

<b>Introduction and Data Description .....</b>	
(a) 分析目的 .....	2
(b) 資料說明 .....	2
(c) 分析流程 .....	4
<b>Exploratory Data Analysis.....</b>	
(a) Continuous Covariates .....	5
(b) Principal Components Analysis (PCA).....	7
(c) Categorical Covariates .....	9
(d) Odds Ratio .....	11
<b>Methods .....</b>	
(a) KNN (K-Nearest Neighborhood Classifier) .....	12
(b) Naïve Bayes .....	12
(c) Logistic Regression .....	13
(d) Group LASSO .....	14
(e) Random Forest .....	15
(f) XGBOOST .....	17
<b>Data Analysis.....</b>	
(a) KNN (K-Nearest Neighborhood Classifier) .....	20
(b) Naïve Bayes .....	21
(c) Logistic Regression .....	22
(d) Group LASSO .....	24
(e) Random Forest .....	26
(f) XGBOOST .....	28
✓ SHAP Value .....	30
<b>Conclusion.....</b>	33
 附錄 1 工作分配表.....	34
附錄 2 參考文獻.....	34
附錄 3 程式碼.....	35

# 1. Introduction and Data Description

## (a) 分析目的

判斷心臟病的有無，會根據各種不同的標準去檢測，例如病人的膽固醇含量、最大心率、血壓...等。根據這份醫學研究資料，總計有 13 個檢測值，我們想去了解有哪些項目是評估心臟病的重要考量，或是可以藉此額外發現一些趨勢。除此之外，我們也透過一些機器學習的演算法，試著去比較各個演算法的差異、預測的精準度。希望透過這次的期末報告，將一整個學期學習到的知識融會貫通。

## (b) 資料說明

為探討哪些項目是判斷是否有心臟病的重要考量，本次分析使用 Kaggle 上的 Heart Disease UCI 資料集。( <https://www.kaggle.com/ronitf/heart-disease-uci> )

此資料為 1988 年收集共 296 筆的醫療檢測數據，原始資料共有 76 個檢測值，然而過去大多數已發表的公開研究建議採用的為 14 個檢測值（包含 1 個 response 與 13 個 covariates），而其中適合用於機器學習研究用途的資料集稱為 Cleveland database，也正是我們這次採用的資料集，變數名稱和敘述詳見下表。

變數類別	變數名稱	變數解釋
RESPONSE	<i>target</i>	是否罹患心臟病（1 = 否；0 = 是）。 判斷基準為病人經由血管攝影檢查後，其心血管收縮大於或小於 50%。其中收縮大於 50% 的病人，判斷其有心臟病，反之亦然。
COVARIATES	<i>Age</i>	受試者年齡。
	<i>Sex</i>	受試者性別。
	<i>ChestPainType</i>	胸痛的種類 0 = 無症狀 1 = 非典型心絞痛 2 = 非心絞痛的胸痛 3 = 典型心絞痛
	<i>RestingBloodPressure</i>	受試者靜態時的血壓。
	<i>Cholesterol</i>	受試者的膽固醇。
	<i>FastingBloodSugar</i>	受試者空腹時的血糖。

1 = if > 120mg/dl

0 = if < 120mg/dl

正常空腹血糖會 < 100mg/dl，若介於  
100 ~ 125mg/dl 為糖尿病前期。

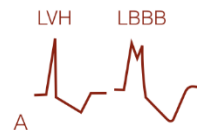
### RestEcg

受試者靜息心電圖結果。

0 = 正常

1 = 顯示該標準下可能或明確有左心室肥厚的情形。當積分大於 4 時「可能」為左心室肥厚，大於 5 分時，則「認為」是左心室肥厚。

2 = 有 ST-T 波異常 (T 波倒置和/或 ST 波段抬高或降低 > 0.05mV)



#### ST 段下降和不對稱 T 波倒置

Secondary to LVH and LBBB  
ST段變化方向與T波變化同向



#### ST 段下降加上 和QRS 同相的 T 波倒置

暗示著缺氧性變化

上圖為  
異常之  
圖示

ST段下降與T波倒置

MaxHeartRateAchieved 受試者的最大心率。

一個年齡 20 歲的人，最大心率大多落在 189 ~ 211 之間。

ExerciseInducedAngina 運動後引發的心絞痛。

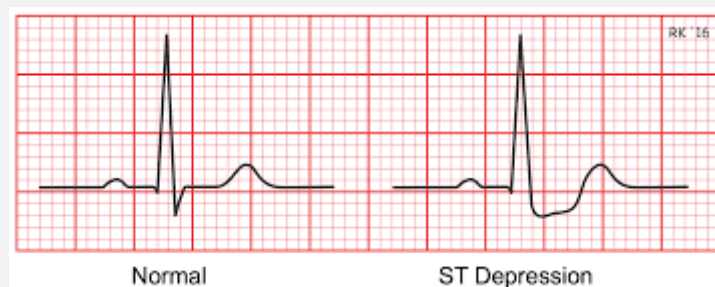
1 = 有

0 = 沒有

### STDepression

代表當運動時所引發之心電圖發生異常低於 base line 的數值。

下圖為正常情況與 ST Depression 之比較



正  
常  
斜  
率  
為  
水

平，若 ST 段上升，暗示了可能有心肌梗塞的情況；若 ST 段下降，則有可能有心肌缺氧的情況發生。

關於 ST 段的解釋於下一變數講解。

*PeakExerciseST*

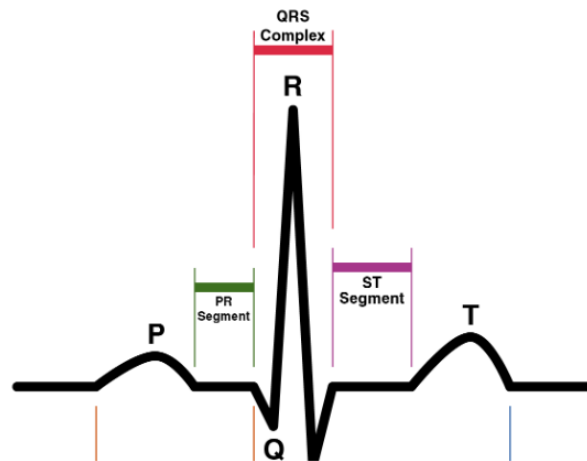
表示運動高峰期的 ST 段斜率

0 = *downsloping*

1 = *flat*

2 = *upsloping*。

下圖紫色區段為 ST 段的圖示



*NumMajorVessels*

受試者經過主動脈造影後，所被標記顏色的主動脈數量（總共 3 條主動脈）。被標記顏色則代表經檢測過後，主動脈有剝離的情形。

*ThalliumDefect*

代表一種心臟壓力測試的結果，對心肌做出放射的檢測，其結果分為 3 種。

0 = 正常

1 = 可逆性心肌缺血

2 = 不可逆心肌缺陷

### (c) 分析流程

在本次的分析當中，首先我們會進行資料探索，了解各個變數的分布以及不同變數之間的關係。對資料有一定的了解後，我們使用了 6 種機器學習的演算法建立模型，比較各個模型所選取出來的重要變數、在測試集上的準確率等。

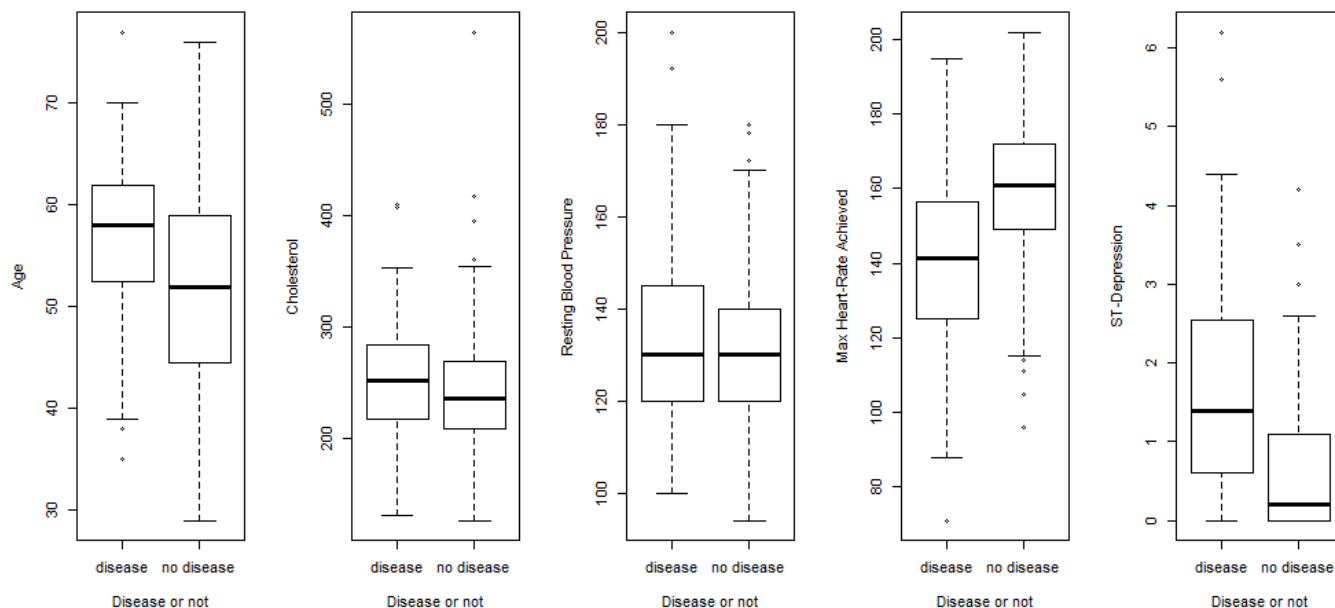
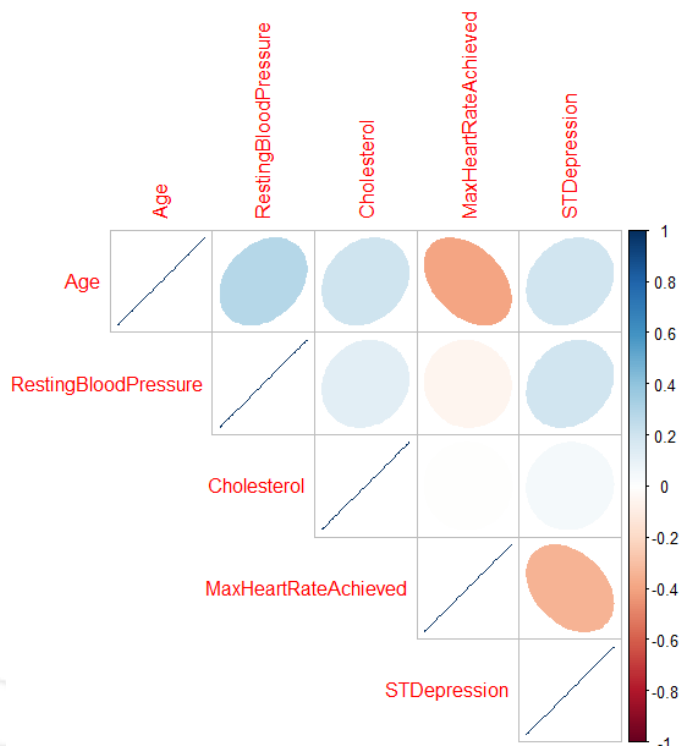
本次報告分為五個章節，第一章為一些分析目的、資料的背景介紹，第二章為對原始資料所進行的一些資料探索，第三章則是介紹使用的分析方法、模型，接著第四章說明各個模型的分析結果和比較，最後，第五章為本報告之總結。

## 2. Exploratory Data Analysis

### (a) Continuous Covariates

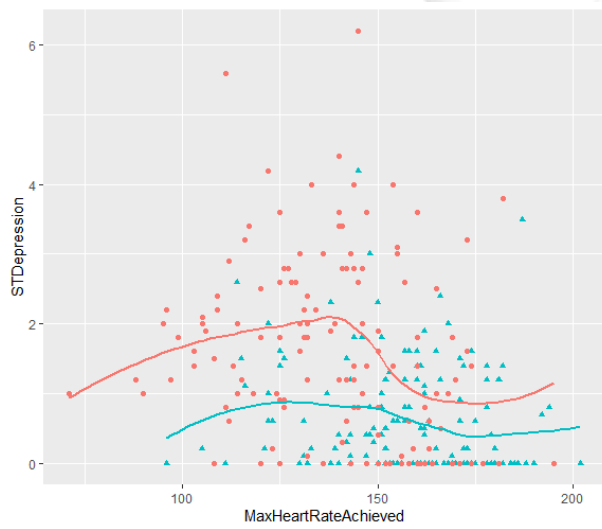
首先，可以看到右圖顯示的是兩兩連續型變數之間的相關性。可以發現的是，右圖中相關性的顏色都偏淡，這也表示著我們資料中的連續型變數，兩兩間並沒有顯著的線性相關。

接著我們將5個連續型變數與是否有得心臟病去做了 box plot。可以發現的是，年齡愈高，愈多人會得到心臟病；最高心率愈高，愈少人會得到心臟病；*ST Depression* 發生的程度愈高，愈多人會得到心臟病。而在膽固醇、靜態時的血壓則看不出有明顯的特徵。



接著我們針對兩兩連續型變數間的散布圖去做觀察，並挑選出可以較明顯看出特徵的圖。首先，可以從下頁中的左圖發現，當年齡、*ST Depression* 愈高時，受試者愈容易罹患心臟病。從中間的圖則可以發現，當受試者年齡愈低且最高心率愈高時，受試者愈不容易罹患心臟病。而由右圖可以發現，當最高心率愈高且 *ST Depression* 愈低時，受試者愈不容易罹患心臟病。

由此，我們初步的猜測，年齡(Age)、*ST Depression*、最高心率(MaxHeartRateAchieved)這三個變數為重要的變數。在之後我們會再與模型選取出來的重要變數去做比較，看看是否符合我們的猜想。



target  
 —●— disease  
 —▲— no disease





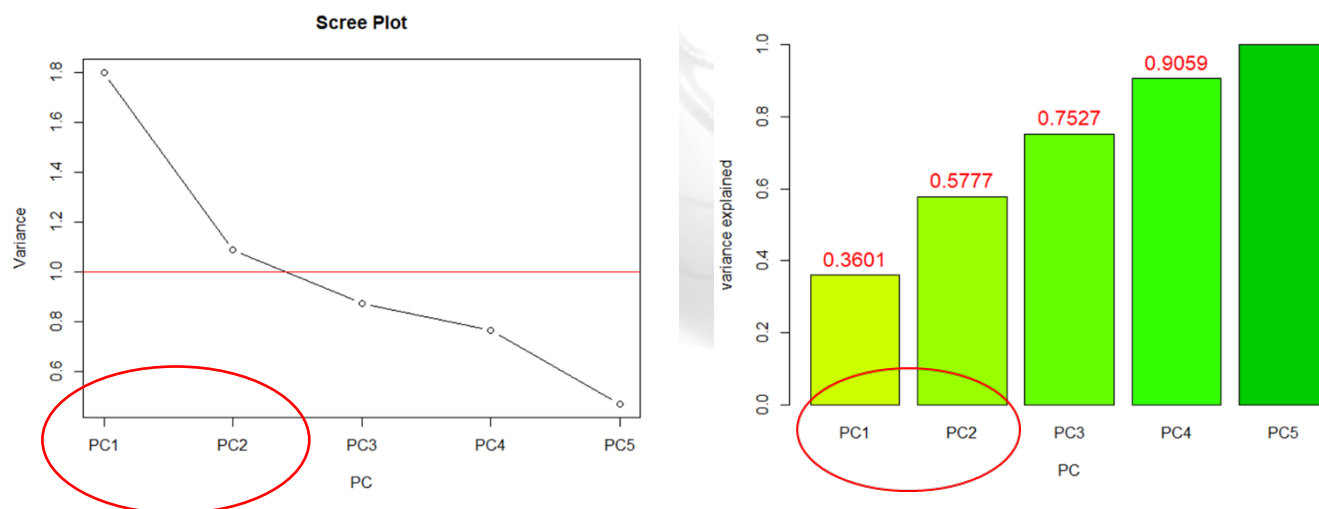
## (b) Principal Components Analysis (PCA)

接著我們也針對連續型變數去做主成分分析，希望可以透過 PCA 發現一些資料的特徵。可以看到下表為 5 個主成分以及各個變數所對應的係數。

	PC1	PC2	PC3	PC4	PC5
Age	-0.5652947	0.1193648	0.1863709	-0.5253511	-0.5962097
Resting Blood Pressure	-0.3880648	0.4115715	-0.7354370	-0.1415096	0.3451407
Cholestero1	-0.2291518	0.7027594	0.5158682	0.3993843	0.1673050
Max Heart Rate Achieved	0.5080352	0.4688008	-0.3285189	0.1528130	-0.6251796
STDepression	-0.4682279	-0.3204934	-0.2243963	0.7218875	-0.3264527

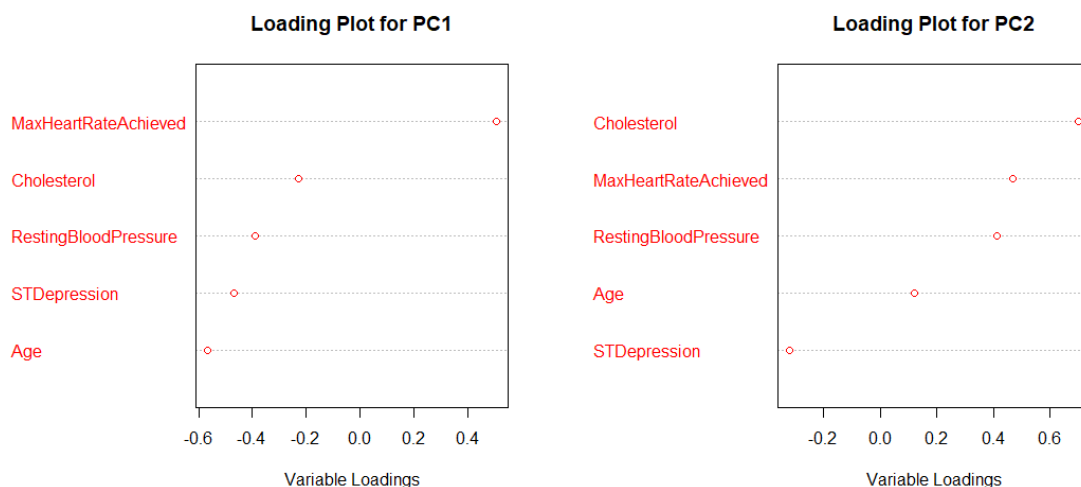
接著我們根據凱莎原則，選取 Variance 大於 1 的主成分，因此這邊我們只選取了第一、第二主成分。也因此接下來我們將針對第一、第二主成分去做更進一步的分析。

另外我們也可以從右下圖中得知，前 2 個主成分共解釋了整個資料的 57.77% 的變異。換句話說，現在只需 2 個維度的資料就代表了先前 5 個維度的資料之 57.77%。



接著我們去觀察 PC1、PC2 的主成分負荷圖。可以從下圖發現，PC1 與年齡、ST Depression、靜態時的血壓、膽固醇呈現負相關，只與最高心率呈現正相關。而 PC2 則是與年齡、靜態時的血壓、最高心率、膽固醇呈現正相關，只與 ST Depression 呈現負相關。





最後，可以看到下頁的圖為在 PC1、PC2 之下各個資料點的散布圖以及雙標圖，並用有病以及沒病去區分開各個資料點。點之間的距離表示了它們所對應的樣本點之間的差異大小，距離愈近表示對應樣本差異小，存在相似性，反之亦然。可以發現的是聚集於 *Age* 向量、*ST Depression* 向量的點更容易被歸類於有病。也就是說，今天樣本點的 *Age*、*ST Depression* 原特徵的值愈高，愈容易罹患心臟病。另外也可以發現，聚集於 *MaxHeartRateAchieved* 向量的點更容易被歸類於沒病。也就是說，今天樣本點的 *Max Heart Rate Achieved* 原特徵的值愈高，愈不容易罹患心臟病。此結果與前面連續型變數的 EDA 結果相同，因此我們猜測 *Age*、*ST Depression*、*Max Heart Rate Achieved* 此三個變數為重要變數。



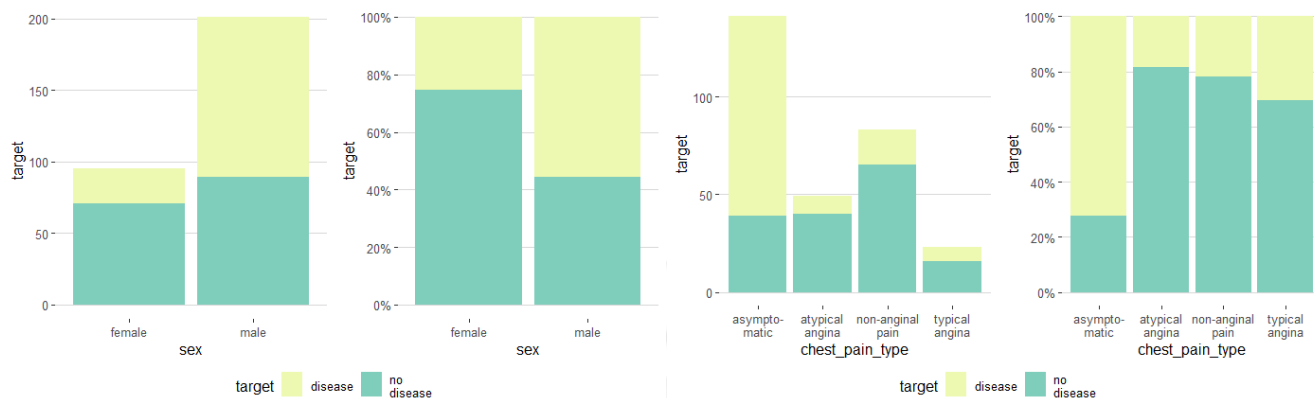
## (c) Categorical Covariates

### ✧ Sex (下左圖)

樣本中生病的人大部分是男性，且男性群體中，有病的比率約為 55.7%，大於女性群體中有病的比率約為 25.26%。

### ✧ Chest Pain Experienced (下右圖)

沒有胸痛的人中得心臟病的人約佔 75%，不禁讓人懷疑胸痛與否是否與心臟病有關；而在其餘類型的心絞痛的人中，沒有心臟病的人都佔了大多數。



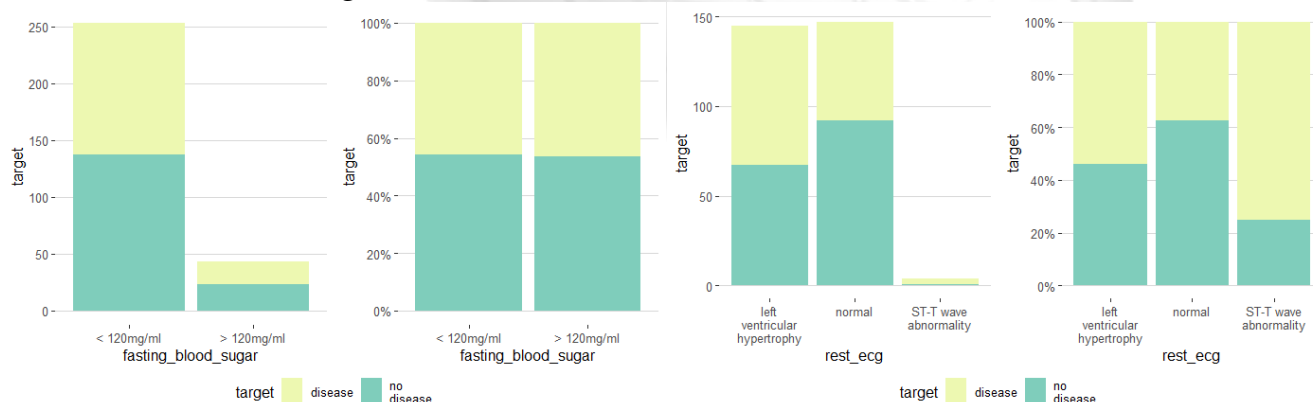
### ✧ Fasting Blood Sugar (下左圖)

空腹血糖一般若高於 100mg/dl，為糖尿病前期，高於 126mg/dl 的話，進入正式糖尿病。我們一般的認知可能會預期高空腹血糖（傾向糖尿病）的人，其心血管狀況就會較差但這裡出現有趣的結果，不管有無心臟疾病的人，血糖多寡的比例都差不多。

高空腹血糖的人有一半左右的人有心臟病另一半沒有，較正常血糖的人有一半左右的人有心臟病另一半沒有，所以“fbs”可能不是一個重要變數。

### ✧ Resting Electrocardiographic Measurement (下右圖)

左心室肥厚的人通常血壓也高，血壓高的人常理而言得心臟病也比較容易，所以這張圖是合理的。所以“restecg”可能是一個重要變數



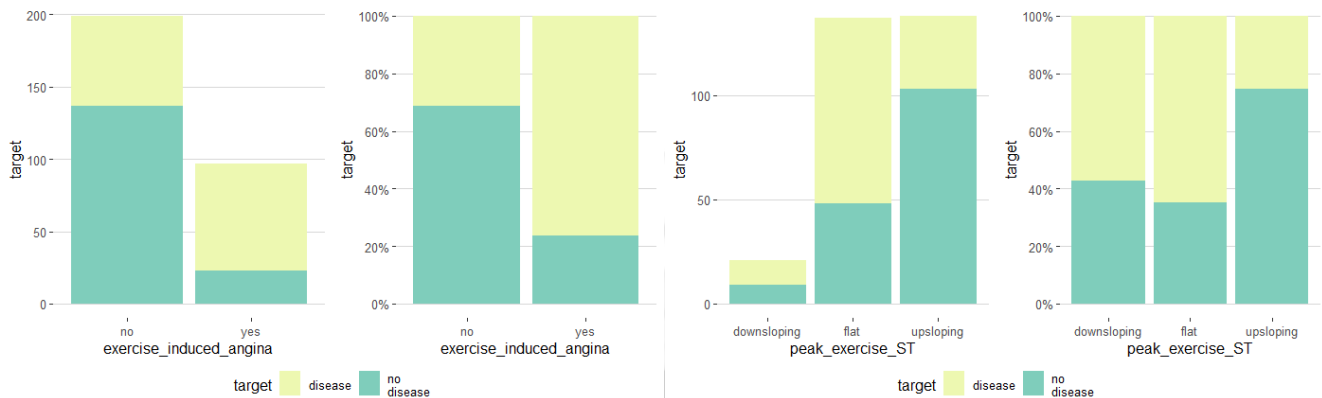
#### ✧ *Exercise Induced Angina* (下左圖)

沒有運動後引發的心絞痛的人大多沒心臟病，反之亦然，符合常理。運動時心跳加速等因素也更有可能是造成心臟疾病，所以“exang”可能是一個重要變數。

#### ✧ *The Slope of the Peak Exercise ST Segment* (下右圖)

(心電圖的 ST 波斜率：一般 ST 波會微微向上的弧線，若 ST 節段位置較正常時為低，或呈現線段較平，則可能代表冠狀動脈缺血。ST 節段下降則可能代表心內膜下梗死、低鉀血症、毛地黃甘中毒)

我們可以看到 ST 波斜率上升的人大多沒有心臟病，ST 波斜率平坦的人很多有心臟病，ST 波斜率下降的人在我們的資料中本身資料點較少，但此族群中仍然有 57.1% 的人患有心臟病；所以“slope”可能是一個重要變數。

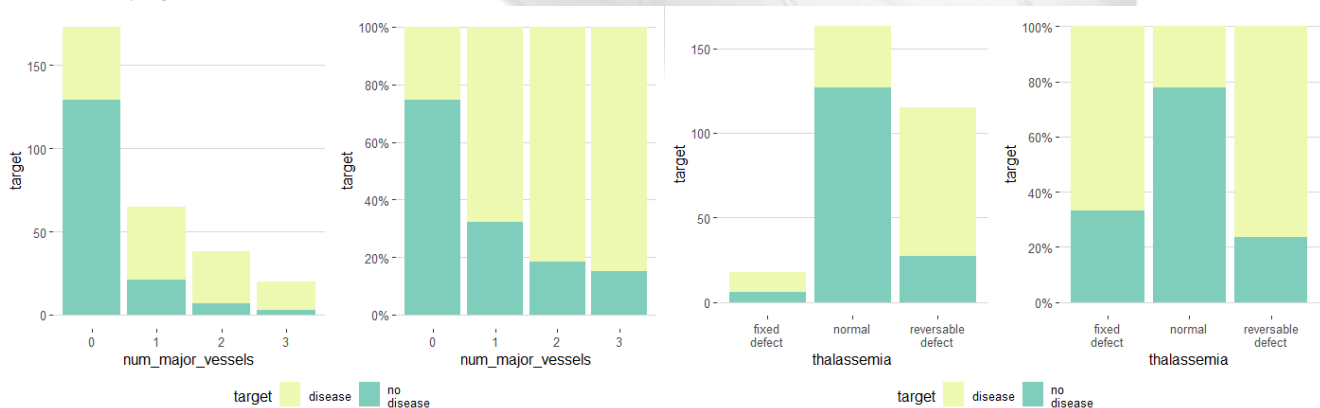


#### ✧ *The number of Major Vessels* (下左圖)

(受試者經過主動脈造影後，所被標記顏色的主動脈數量（總共 3 條主動脈：髂動脈、主動脈、股動脈）。被標記顏色則代表經檢測過後，主動脈有剝離、撕裂的情形；其越多剝離、撕裂的情形，血液可以流入動脈壁層之間的機會越大，產生血管層夾層的症狀，且無法提供心臟或其他器官足夠血液。從樣本內來看，推論其越多主動脈剝離或缺陷情況為一重要變數。

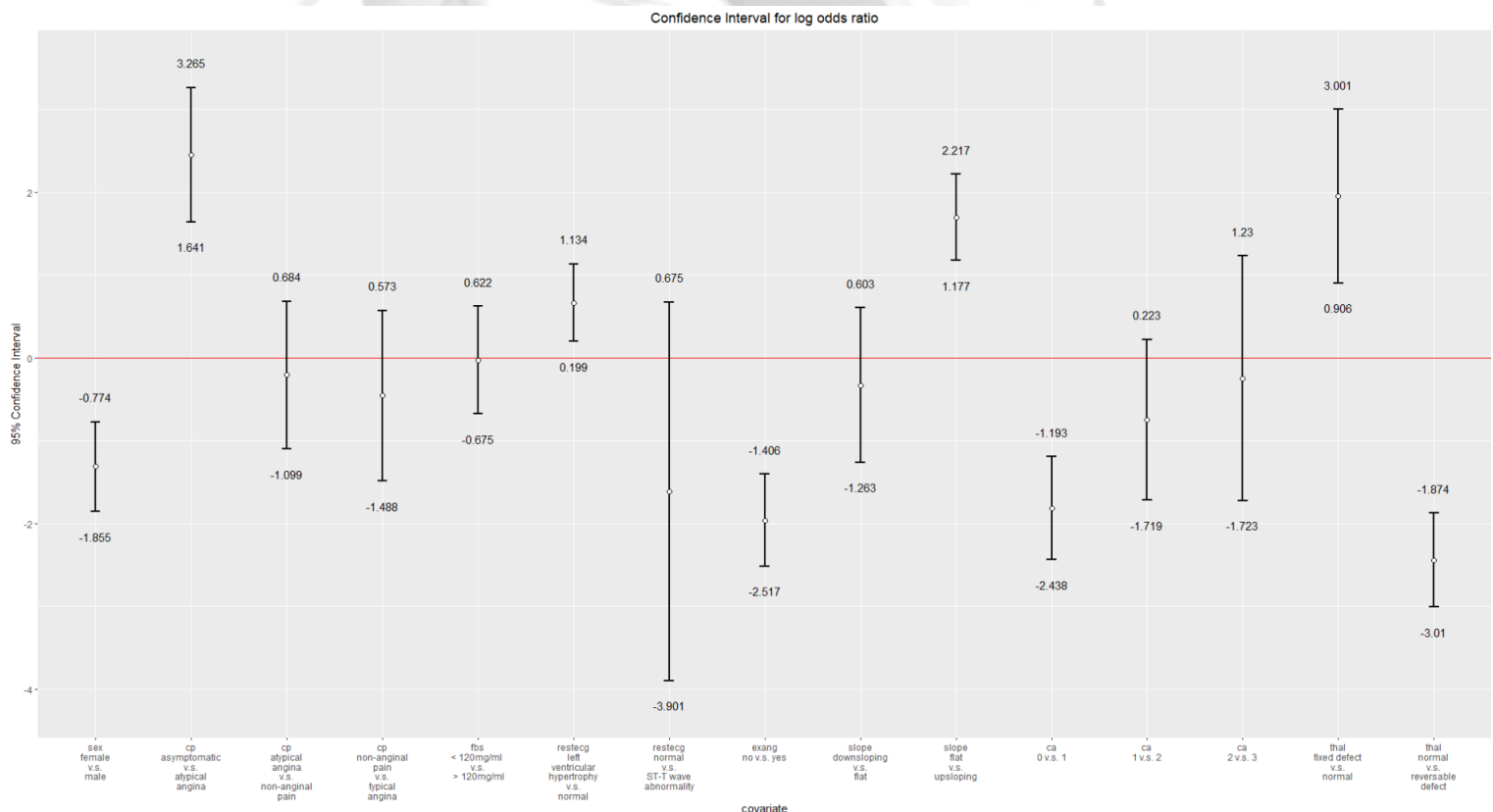
#### ✧ *Thalium Stress Test (Thalium-Defect)* (下右圖)

從下圖的心臟壓力測試的結果來看，其中可逆性心肌缺血(Reversible Defect)與不可逆心肌缺陷(Fixed Defect)的樣本內患者約有 65-75% 的比例患有心臟病，推論其是否有心肌缺血或缺陷為一重要變數。



## (d) Odds Ratio

- ✓ **Sex**：其勝算比信賴區間沒有跨過 0，顯示男性較女性有較大的可能性為心臟病患者。
- ✓ **Chest Pain Experience**：其中(*asymptomatic v.s. atypical angina*)的勝算比信賴區間沒有跨過 0，顯示沒有過心絞痛或胸痛有較大的可能性為心臟病患者。
- ✓ **Fasting Blood Sugar**：其勝算比的信賴區間跨過 0，顯示高空腹血糖的人與正常血糖的人有可能性為心臟病患者的勝率相當。
- ✓ **Resting Electrocardiographic**：其中左心室肥厚的患者較正常人的勝算高且其勝算比信賴區間沒有跨過 0，顯示左心室肥厚的患者有較大的可能性為心臟病患者。
- ✓ **Exercise Induced Angina**：有(過)運動後引發的心絞痛的人的勝算比沒有過的人高且其勝算比信賴區間沒有跨過 0，顯示運動後引發的心絞痛的人有較大的可能性為心臟病患者。
- ✓ **The Slope of the Peak Exercise ST Segment**：其中 ST 波斜率平坦的患者與 ST 波斜率下降的患者的勝算比信賴區間沒有跨過 0、ST 波斜率平坦的患者與 ST 波斜率上升的患者的勝算比信賴區間有跨過 0，顯示 ST 波斜率平坦、下降的患者有較大的可能性為心臟病患者。
- ✓ **The number of Major Vessels**：其中 1 條主動脈剝離、撕裂的患者與 0 條主動脈剝離、撕裂的患者的勝算比信賴區間沒有跨過 0，顯示 1 條主動脈剝離、撕裂的患者有較大的可能性為心臟病患者；又 1 條、2 條，2 條、3 條主動脈剝離、撕裂的患者的勝算比信賴區間跨過 0，所以顯示有主動脈剝離、撕裂的患者有較大的可能性為心臟病患者。
- ✓ **Thalium Stress Test (Thalium-Defect)**：其中不可逆心肌缺陷(*Fixed Defect*)與正常人的勝算比信賴區間沒有跨過 0、正常人與可逆性心肌缺血(*Reversible Defect*)的勝算比信賴區間沒有跨過 0，顯示有心肌缺陷的患者有較大的可能性為心臟病患者。



### 3. Method

#### (a) KNN (K-Nearest Neighborhood Classifier)

KNN 是一個簡單易懂且利於解釋的分類演算法，其核心概念為在辨識一個未知物件的類別時，尋找特徵空間中距離該物件最接近的 K 個訓練樣本，藉由多數決找出這 K 個鄰居中出現頻率最高的分類，最後再將此物件判給這個分類。

其步驟為：

1. 選定 K 值（可利用交叉驗證提高分類準確率）
2. 運用※歐式距離計算未知樣本與訓練樣本間的距離（需為連續變數）
3. 找出與未知樣本距離最近的 K 個點
4. 藉由這 K 個樣本中類別的眾數去給定此未知樣本分類

※歐式距離（Euclidean Distance）：計算 p 維空間中兩個點(x,y)間的真實距離

$$d(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \dots + (x_p - y_p)^2} = \sqrt{\sum_{i=1}^p (x_i - y_i)^2}$$

此處 p 為樣本集 covariate 的個數

#### (b) Naïve Bayes

單純貝氏(Naive Bayes) 是一個條件機率模型，獨立的類別變數 Y 有若干類別，條件依賴於若干特徵變數  $X=[x_1, x_2, \dots, x_p]$ ，其模型來自於

$$P(Y = y | x_1, x_2, \dots, x_p) = \frac{P(x_1, x_2, \dots, x_p | Y = y)}{P(x_1, x_2, \dots, x_p)}$$

由於分母部分與 y 無關，且各  $x_i$  的值是給定的，故將分母當作常數，不影響分類的判定，故將其忽略。此時模型變成，後者即為聯合機率密度函數。

$$P(Y = y | x_1, x_2, \dots, x_p) = P(x_1, x_2, \dots, x_p | Y = y)$$

在假設所有的隨機變數之間具有條件獨立的情況下，可以直接利用條件機率相乘的方法，計算出聯合機率分布。

$$P(x_1, x_2, \dots, x_p | Y = y) = P(X_1 = x_1 | Y = y)P(X_2 = x_2 | Y = y) \dots P(X_p = x_p | Y = y)$$

針對各種可能的 y 值計算其幾率，找出幾率最大的即為分類的結果。模型寫為

$$\arg \max_y \prod_{j=1}^p P(X_j = x_j | y)$$

若 Y 僅有(0,1)兩種結果，且 X 亦是(0,1)的分類變數可進一步將模型寫成

$$\log \frac{P(y=1|x_1, x_2, \dots, x_p)}{P(y=0|x_1, x_2, \dots, x_p)} = \sum_{j=1}^p w_j x_j - b$$

其值大於 1 便判定  $y=1$ ，反之則是  $y=0$ 。

其中  $w_j$  為  $\log \frac{p_j}{q_j} \frac{1-q_j}{1-p_j}$  ( $p_j = P(X_j = 1 | y = 1)$ ,  $q_j = P(X_j = 1 | y = 0)$ ),

$b$  則是  $\log \frac{P(y=1)}{P(y=0)} + \sum_{j=1}^p \log \frac{1-p_j}{1-q_j}$  利用 training data 估計出  $p_j$  以及  $q_j$  和  $P(y=1)$ ,  $P(y=0)$  即可算出  $w_j$  和  $b$ ，進而完成模型。此外若  $w_j = 0$  則變數  $x_j$  對判別 Y 無幫助。

而倘若  $x_j$  是連續變數，則需假設  $P(X_j | Y)$  符合常態分佈。在  $\log$  轉化及一定的計算下，可以將模型寫成

$$\log \frac{P(y=1|x_1, x_2, \dots, x_p)}{P(y=0|x_1, x_2, \dots, x_p)} = \sum_{j=1}^p s_j x_j + d$$

其值大於 1 便判定  $y=1$ ，反之則是  $y=0$ 。

其中  $s_j$  為  $\frac{u_{j1}-u_{j0}}{\sigma_j^2}$ ,  $d$  則是  $\log \frac{P(y=1)}{P(y=0)} + \sum_{j=1}^p \frac{u_{j1}^2 - u_{j0}^2}{2\sigma_j^2}$ 。利用 training data 估計出  $u_{jy}$  以及  $\sigma_j^2$  和  $P(y=1)$ ,  $P(y=0)$  即可算出  $s_j$  和  $d$ ，進而完成模型。

## (c) Logistic Regression

邏輯迴歸(Logistic Regression)，是用於處理二元變數(binary response) Y 及解釋變數(explanatory variable) X 間的關係的統計方法。其中 X 可為分類型變數，也可以是連續型變數。

首先，定義  $\pi(x)=P(Y=1 | X=x)$  模型為

$$\pi(x) = \frac{\exp(\alpha + \beta x)}{1 + \exp(\alpha + \beta x)}$$

其回傳為介於 0~1 之間的數值。

而邏輯迴歸便建立在  $\frac{\pi(x)}{1-\pi(x)}$  這樣的成功幾率和失敗機率的比值上，我們稱之為 odd。在對其取對數後，便是命名為 logit 的計算法，其同時具備著線性結構。

$$\text{Logit}[\pi(x)] = \log \frac{\pi(x)}{1-\pi(x)} = \alpha + \beta x$$

但有多個解釋變數時，如  $x = (x_1, x_2, \dots, x_p)$ ，其模型可以寫為

$$\text{Logit}[\pi(x)] = \alpha + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p$$

回推可知其

$$\pi(x) = \frac{\exp(\alpha + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p)}{1 + \exp(\alpha + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p)}$$

在估計參數  $\alpha$  與  $\beta_j$  時，我們使用最大蓋似函數估計(MLE)，首先假設有  $n$  個樣本，且  $\pi(x_i)$  為伯努利分配(Bernoulli distribution)，pdf 為

$\pi(x_i)^{y_i}[1 - \pi(x_i)^{1-y_i}]$ ，此時蓋似函數為

$$\begin{aligned}\prod_{i=1}^n \pi(x_i)^{y_i}[1 - \pi(x_i)^{1-y_i}] &= \prod_{i=1}^n \{ \exp(\log(\frac{\pi(x)}{1-\pi(x)})^{y_i} [1 - \pi(x)]) \} \\ &= \prod_{i=1}^n \{ \exp(y_i \log \frac{\pi(x)}{1-\pi(x)}) \} \{ \prod_{i=1}^n [1 - \pi(x)] \} \\ &= \{ \exp[\sum_{i=1}^n y_i \log \frac{\pi(x)}{1-\pi(x)}] \} \{ \prod_{i=1}^n [1 - \pi(x)] \}\end{aligned}$$

由於  $\log(\frac{\pi(x)}{1-\pi(x)}) = \sum_j \beta_j x_{ij}$  因此  $\exp[\sum_{i=1}^n y_i \log \frac{\pi(x)}{1-\pi(x)}]$  便可寫寫作  $\exp[\sum_j (\sum_i y_i x_{ij}) \beta_j]$ ，同時  $[1 - \pi(x)] = 1 + \exp(\sum_j \beta_j x_{ij})^{-1}$  此時蓋似函數便可寫做

$$L(\beta) = \sum_j (\sum_i y_i x_{ij}) \beta_j - \sum_i \log[1 + \exp(\sum_j \beta_j x_{ij})]$$

找出最大蓋似函數便是找到  $L(\beta)$  極值的一階條件

$$\frac{\partial L(\beta)}{\partial \beta_j} = \sum_i y_i x_{ij} - \sum_i x_{ij} \frac{\exp(\sum_k \beta_k x_{ik})}{1 + \exp(\sum_k \beta_k x_{ik})} = 0$$

其解為

$$\sum_i y_i x_{ij} - \sum_i \hat{\pi}_i x_{ij} = 0$$

$\hat{\pi}_i = \frac{\exp(\sum_k \hat{\beta}_k x_{ik})}{1 + \exp(\sum_k \hat{\beta}_k x_{ik})}$  同時是  $\pi(x_i)$  的 MLE，但由於沒有封閉解，故需要用牛頓法等數值方法迭代求出  $\hat{\beta}_k$ ，進而得到目標函數  $\hat{\pi}_i$ 。

## (d) Group LASSO

為了改善線性迴歸中 overfitting 造成的預測誤差進而提高預測準確率，以及在建立模型中同時選取重要的變數，Robert Tibshirani 在 1996 年提出 LASSO(least absolute shrinkage and selection operator)方法，此方法是基於最小平方法，同時對估計參數給定一限制式，目標函數為：

$$\min \sum_{i=1}^n (y_i - x_i' \beta)^2 \text{ s.t. } \sum_{j=1}^p |\beta_j| \leq t$$

其中， $x_i$  為的解釋變數、 $y_i$  為反應變數， $t$  則為限制估計參數的範圍。上式可以拉格朗日形式改寫為：

$$\min \sum_{i=1}^n (y_i - x_i' \beta)^2 + \lambda \sum_{j=1}^p |\beta_j|$$

其中， $\lambda$  為調控參數， $\lambda \sum_{j=1}^p |\beta_j|$  為懲罰項(penalty term)，當  $\lambda \rightarrow 0$  時等同於  $t \rightarrow \infty$ ，則未對參數有任何限制，即為最小平方法的目標函數；當  $\lambda$  遞增  $t$  遞減時，參數會被限制在一範圍內，以此避免 overfitting，甚至有些參數會被壓縮至 0，則對應參數不為 0 的變數則為被挑選出的重要變數。

在上述的方法中，若遇到解釋變數為類別型的變數，則需要轉為 dummy variable，若該類別



中有  $m$  個 class，則會以  $m-1$  個新變數替代原本的類別型變數。當要透過參數壓縮的方式挑選重要變數，則會遇到某個類別型變數中的部分 class 被挑出，而非原本的類別型變數被挑出，因此上述的 lasso 方法並不適用於有類別型解釋變數的資料集。然而，只要類別變數中  $m$  個 class 所形成的  $m-1$  個新變數對應的估計參數同時為 0 會不為 0，就能解決上述問題，因此可以利用 group LASSO 方法處理包含類別型變數的資料集。

Group LASSO 是 2006 年由 Yuan and Lin 提出的方法，其概念是將  $p$  個變數分成不重疊的  $K$  組，也就是  $(1, 2, \dots, p) = \cup_{k=1}^K I_k$  且  $I_k \cap I_{k'} = \emptyset$ ，其中  $I_k$  集合中的個數是  $p_k$ ，則目標函數為：

$$\min \sum_{i=1}^n (y_i - x_i' \beta)^2 + \lambda \sum_{k=1}^K \sqrt{p_k} \|\beta^{(k)}\|_2$$

其中  $\|\beta^{(k)}\|_2 = \sqrt{\sum_{j \in I_k} \beta_j^2}$ ，在此方法中，除了將原本的變數分組，同組的變數會同時被壓縮至 0，也利用  $\sqrt{p_k}$  做為懲罰項的權重，當  $\sqrt{p_k}$  全為 1 時，即為原本的 LASSO。

回到我們同一類別變數的  $m-1$  個新變數是否可以同時被選取的問題，立用 group LASSO 分組的方式，指定同一個類別變數所產生的 dummy variable 新變數為同一組，就可以解決只選到特定 class 的問題。

另外，由於我們所使用的資料反應變數為類別型，因此無法使用 squared error loss function，取而代之的是利用 logistic regression 中的 negative binomial log-likelihood 函數做為 loss function，因此目標函數為：

$$\min - \left[ \sum_{i=1}^n y_i (\beta_0 + x_i' \beta) - \log (1 + e^{(\beta_0 + x_i' \beta)}) \right] + \lambda \sum_{k=1}^K \sqrt{p_k} \|\beta^{(k)}\|_2$$

## (e) Random Forest

Random Forest 是一種 Ensemble Learning 的演算法，而 Ensemble Learning 概念上是結合多個弱學習器來建構一個強穩的模型。每次會用類似 Bootstrap Aggregation 的方法取得一個新的 Decision Tree，再將所有的 Decision Tree 結合起來。

其中 Decision Tree 是一種監督式機器學習模型，利用像樹一樣的圖形去建構預測模型，適用於類別及連續資料類型的預測，相較於其他 Machine Learning 的模型，Decision Tree 的過程直覺、單純且執行效率也很高。Decision Tree 的特點是每個決策階段都很清楚，每個節點代表一個屬性（變數）、每個分支代表對應該屬性的某些可能值（變數範圍）、每個葉節點代表滿足對應路徑的條件下之最終的預測值。

因此將所有的 Decision Tree 結合起來不僅能夠保持 Decision Tree 的差異性，還能減少 fully-grown 造成的 overfitting。此外 Random Forest 最大的精神就是隨機，除了樣本是利用 Bootstrap 採取抽後放回的隨機抽取概念外，變數也是採取隨機抽取的方式，也因此讓 Random Forest 具有高度多樣性。

$$\text{Model : } \hat{f}^{rf}(x) = \underset{k=1,2,\dots,K}{\operatorname{argmax}} \sum_{b=1}^B I(\hat{f}^{tree,b}(x) = k)$$

Random Forest 發展出了一套獨特的 Validation 方法，由於樣本利用 Bootstrap 方法抽取後，會導致有一部份的資料沒有被取用，因此 Random Forest 將這些沒有被抽取到的資料當作類似於測試集的形式，稱之為 Out-of-Bag Data，並可以運用這筆資料計算 Out-of-Bag Error。而透過 Out-of-Bag Error，我們可以運用這個值去選取模型最適合的參數。

以下為參數調控：

初始參數給定 ntree=89（一開始透過 Out-of-Bag Error 選定的），我們主要運用 grid search 的方法去調整參數，概念就是針對每一個模型參數組合，都會配適出一個模型，然後從中挑選表現最佳的模型。這次我們主要調控的參數有：mtry、nodesize、sampsize。

模型參數	
ntree	numbers of tree。我們希望有足夠的決策樹來穩定模型的誤差，但過多的樹會導致模型沒有效率。
mtry	每次在決定切割變數時，我們選擇要進行隨機抽樣的變數個數。
nodesize	指定決策樹節點的最小個數，默認情況下，判別模型為 1，回歸模型為 5。
sampsize	訓練每棵樹的樣本隨機採樣比例，較低的值使得算法更加保守，防止 overfitting，但是太小的值也會造成 underfitting。預設值為 63.25%訓練資料集的比例。

此外 Random Forest 的結果可以計算每個特徵的重要程度，在 R 語言中，最後估計出的模型會提供「Mean Decrease Accuracy」及「Mean Decrease Gini」，兩者皆可用來進行特徵選取。Mean Decrease Accuracy 大致上是將 data 中第 i 個變數抽取出後隨機打亂再放入 data 中，將新 data 代入模型計算出新的 Accuracy 後，比較 Accuracy 與原先的差異。Mean Decrease Gini 則是計算該變數讓整個模型之不純度下降的比例。

其中不純度（impurity）可以為(1) Misclassification rate ,(2) Entropy 或 (3) Gini Index

$$(1) \phi(p_1, p_2, \dots, p_K | t) = 1 - \max(p_1, p_2, \dots, p_K | t) \quad ,$$

$$(2) \phi(p_1, p_2, \dots, p_K | t) = - \sum_{j=1}^K p(j|t) \log p(j|t) \quad ,$$

$$(3) \phi(p_1, p_2, \dots, p_K | t) = 1 - \sum_{j=1}^K p(j|t)^2 \quad ,$$

$$\text{where } p(k|t) \text{ is estimated by } \hat{p}(k|t) = \frac{1}{n_t} \sum_{x_i \in \text{node}_t} I(y_i = k)$$

## (f) XGBOOST

所謂的 GBM 算是一種概念，是將梯度下降法（Gradient Descending）跟 Boosting 套件結合在一起的演算法，而後面的 Machine 指不特定的模型，只要能用梯度下降法找尋方向的模型都可以。

如果使用 gbm 的套件，基本上都是 Tree-based 為主，也就是將數百個弱決策樹（CART），跟梯度下降法和 Boosting 結合在一起。

其中使用 `xgb.cv()` 的函式，搭配 cross validation 的技巧，找出最佳的決策樹數量 `nrounds`。過程中，設定 `early_stopping_rounds = 30`（如果當 `nrounds < 30` 時，就已經有 overfitting 情況發生，那表示不用繼續 tune 下去了，可以提早停止），程式會根據 Train 和 Validation 的平均表現，自動判斷模型是否有 overfitting，最後找出較好的 `nrounds`，會是一個最不 overfitting 的模型。

要注意的是，這個最不 overfitting 的模型，是建立在一開始的基本參數設定之下，所以不一定是最好的。（上述 Validation 這個字在 `cv.xgb()` 的 output 會是 Test 這個字）

XGBoost 是 Gradient Boosting Decision Tree（GBDT）的改良版本，使用多個弱分類器來建構一個強分類器，使用前  $m-1$  次迭代結果的負梯度（negative gradient）當作新的反應變數進行下一次迭代，產生新的弱分類器（tree）加到前一次的估計函數上當作新的估計函數。

MODEL :

$$\hat{y}_i = \hat{f}^M(x_i) = \sum_{m=1}^M h_m(x_i) = \hat{f}^{M-1}(x_i) + h_M(x_i)$$

$M$  為迭代次數， $h_m$  為每次迭代所新增的弱分類器。

藉由最小化目標函數

$$Obj = \sum_i L(y_i, \hat{f}^M(x_i)) + \sum_m \Omega(h_m)$$

來找到估計函數，而其中  $\Omega(h_m)$  為 constraint function。

這裡使用 Logistic loss function:

$$L(y_i, f(x)) = y \ln(1 + e^{-f(x)}) + (1 - y) \ln(1 + e^{f(x)})$$

第  $m$  次的迭代結果，目標函數可以寫成:

$$Obj^{(m)} = \sum_i L(y_i, \hat{f}^{m-1}(x_i) + h_m(x_i)) + \Omega(h_m) + constant$$

我們可以把 loss function 當做要展開的函數，使用泰勒展開式的二次逼近

$$Obj^{(m)} = \sum_i \left[ L(y_i, \hat{f}^{m-1}(x_i)) + g_i h_m(x_i) + \frac{1}{2} s_i h_m^2(x_i) \right] + \Omega(h_m) + constant$$

其中  $g_i = \partial_{\hat{f}^{(m-1)}} L(y_i, \hat{f}^{(m-1)}(x_i))$  ,  $s_i = \partial_{\hat{f}^{(m-1)}}^2 L(y_i, \hat{f}^{(m-1)}(x_i))$

因為前  $m-1$  次的結果已確定， $L(y_i, \hat{f}^{(m-1)}(x_i))$  為已知常數併入常數項，目標函數可以改寫成

$$Obj^{(m)} = \sum_i \left[ g_i h_m(x_i) + \frac{1}{2} s_i h_m^2(x_i) \right] + \Omega(h_m) + constant$$

假設樣本  $x$  的輸出落在第  $j$  個葉子上，那麼樣本  $x$  的輸出值為  $w_j$  為每個葉節點相對應權重。

給定一顆樹，起到分類作用的其實是 nodes，輸入一個樣本，葉子的輸出值  $h_m$  就是預測的一顆決策樹，預測的結果是由決策樹的 nodes 所決定的。

對於分類問題，決策樹的葉子就是指類別，對於回歸問題，葉子的值就是數值。

$$h_m(x_i) = \sum_{j=1}^{T_m} w_j I(x_i \in R_{jm})$$

其中  $T_m$  為 nodes 個數。

正則項：決策樹的正則一般考慮的是葉子節點數和葉子權值，常見的是使用葉子節點總數和葉子權值平方和的加權作為正則項：

$$\begin{aligned} \Omega(h_m) &= \gamma T_m + \frac{1}{2} \lambda \sum_{j=1}^{T_m} w_j^2 \\ Obj^{(m)} &\approx \sum_i \left[ g_i h_m(x_i) + \frac{1}{2} s_i h_m^2(x_i) \right] + \Omega(h_m) \\ &\approx \sum_i \left[ g_i \sum_{j=1}^{T_m} w_j I(x_i \in R_{jm}) + \frac{1}{2} s_i \sum_{j=1}^{T_m} w_j^2 I(x_i \in R_{jm}) \right] + \gamma T_m + \frac{1}{2} \lambda \sum_{j=1}^{T_m} w_j^2 \\ &\approx \sum_{j=1}^{T_m} \left[ \left( \sum_{x_i \in R_{jm}} g_i \right) w_j + \frac{1}{2} \left( \sum_{x_i \in R_{jm}} s_i + \lambda \right) w_j^2 \right] + \gamma T_m \end{aligned}$$

Define  $G_j = \sum_{x_i \in R_{jm}} g_i$  ,  $S_j = \sum_{x_i \in R_{jm}} s_i$

$$\approx \sum_{j=1}^{T_m} \left[ G_j w_j + \frac{1}{2} (S_j + \lambda) w_j^2 \right] + \gamma T_m$$

把  $Obj^{(m)}$  對  $w_j$  微分求最大值  $\rightarrow \frac{\partial Obj^{(m)}}{\partial w_j} = 0 \rightarrow w_j^* = -\frac{G_j}{S_j + \lambda}$  帶回  $Obj^{(m)}$

$$Obj^{(m)} = -\frac{1}{2} \sum_{j=1}^{T_m} \frac{G_j^2}{S_j + \lambda} + \gamma T_m$$

從一開始深度為 0 開始切割，找到有最大分數 Gain 的切割點

$$Gain = \frac{1}{2} \left[ \frac{G_L^2}{S_L + \lambda} + \frac{G_R^2}{S_R + \lambda} - \frac{G^2}{S + \lambda} \right] - \gamma$$

不斷地切割下去，直到  $Gain < 0 \forall \text{spilt}$ 。

把此次迭代結果  $h_m(x_i)$  乘上 learning rate  $\nu$  加到前一次迭代結果上

$$\hat{f}^{(m)}(x_i) = \hat{f}^{(m-1)}(x_i) + \nu h_m(x_i)$$

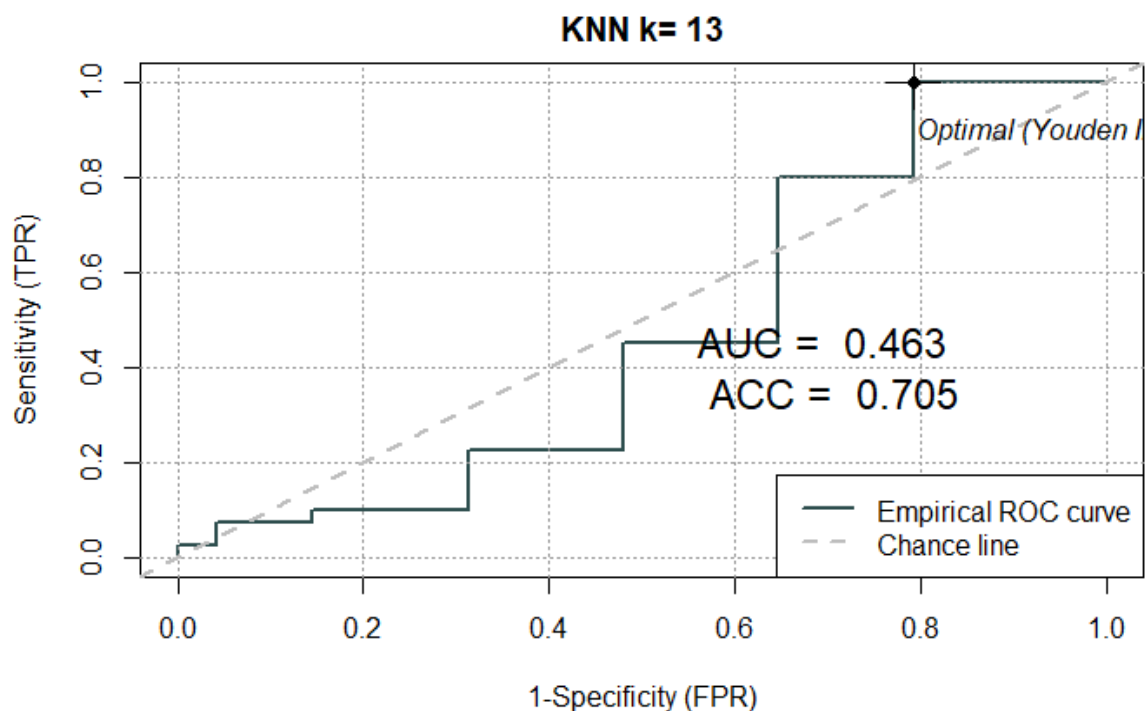
我們不會在每個步驟中進行 full optimization，有助於防止 overfitting



## 4. Data Analysis

### (a) KNN (K-Nearest Neighborhood Classifier)

為選定能夠達成最高模型精確率的最佳  $k$  值，首先利用迴圈 ( $k: 1 \sim 20$ ) 找出其值為 13。代入 Optimal  $k$  進入模型配適後，將測試集資料帶入模型進行預測可得到以下結果：



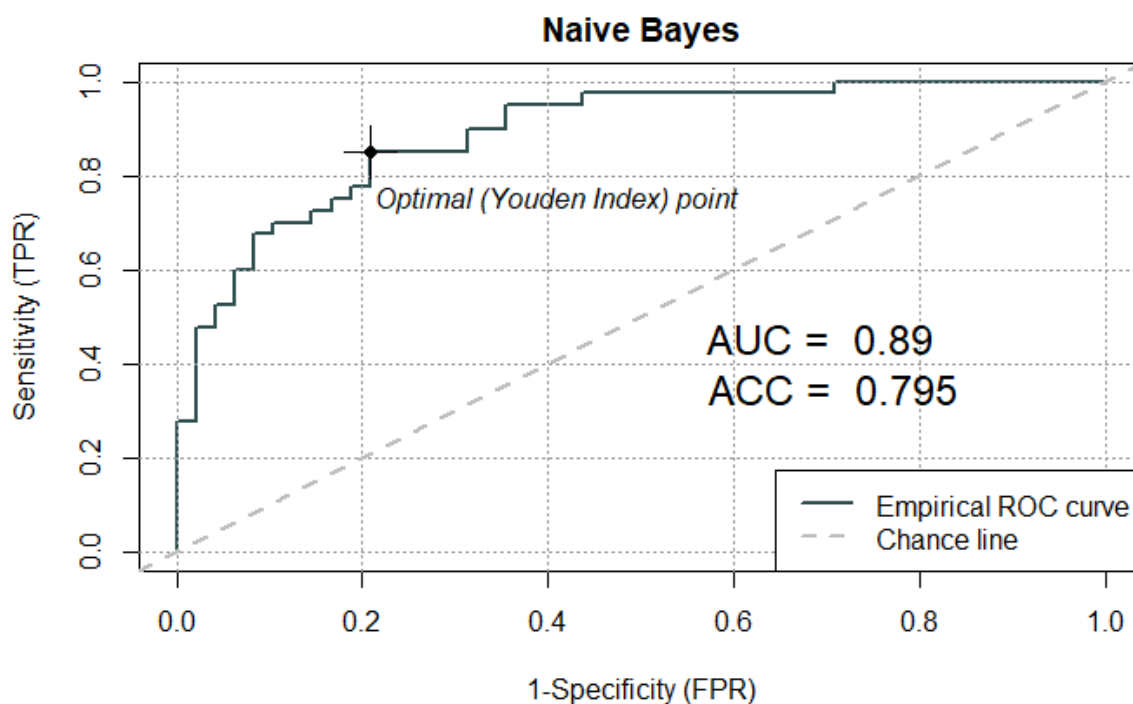
	<i>Sensitivity</i>	<i>Specificity</i>	<i>AUC</i>	<i>ACC</i>
<i>KNN</i>	0.771	0.625	0.463	0.705

可以發現以模型表現而言，Sensitivity 為 77.1%，Accuracy 約為 70.45%

## (b) Naïve Bayes

一開始在未作特徵選擇的情形下建立模型，僅取得 0.7840909 的模型精確度。

而在此前 EDA 篩選出不重要的變數後，類別變數中刪除掉了 *FastingBloodSugar*，連續變數中則是過濾掉 *RestingBloodPressure*、*Cholesterol*，最後將測試集資料帶入模型進行預測可得到以下結果，其精確度為 0.7954545：



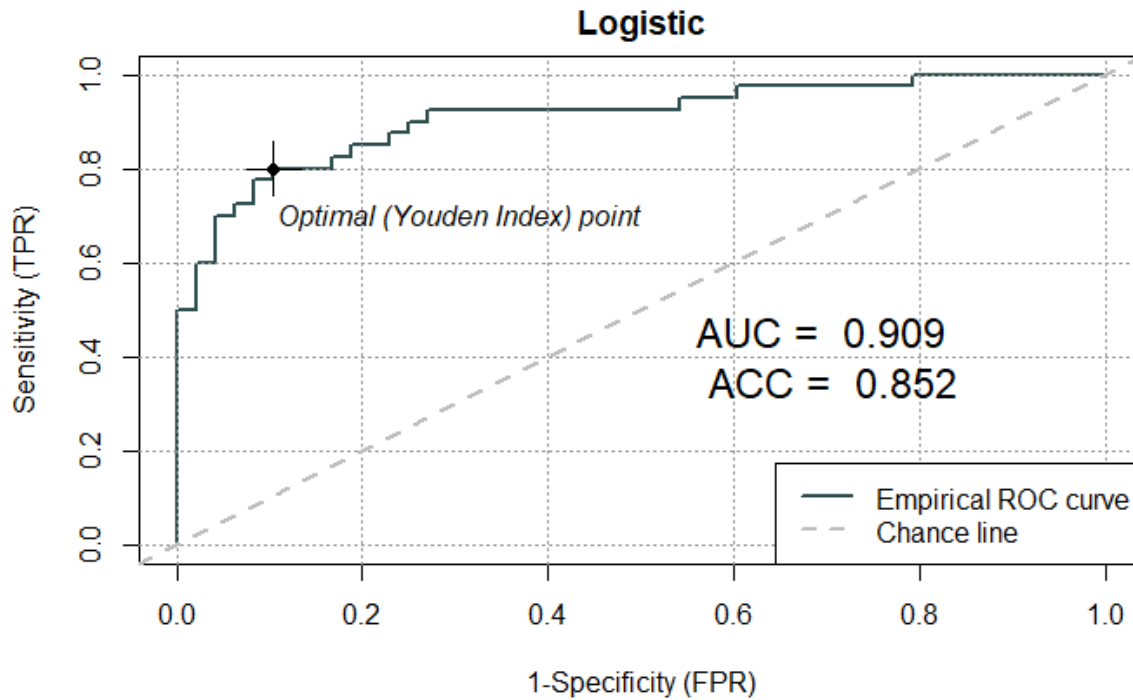
	<i>Sensitivity</i>	<i>Specificity</i>	<i>AUC</i>	<i>ACC</i>
<i>Naïve Bayes</i>	0.833	0.750	0.890	0.795



### (c) Logistic Regression

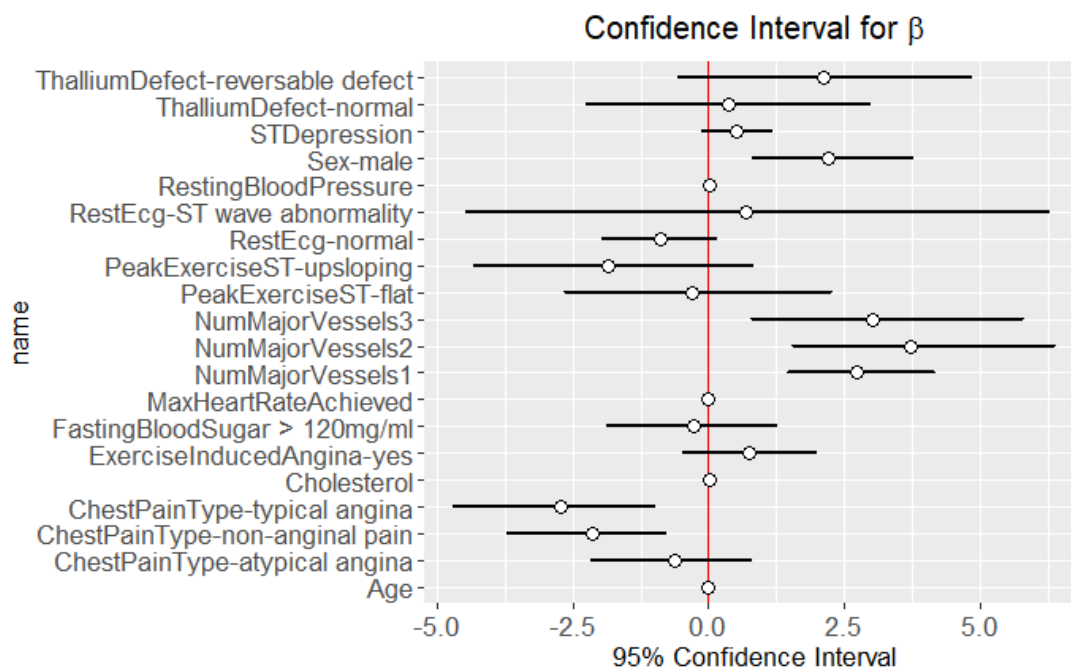
在未選定 Cutpoint (預設為 0.5) 的狀況下去建立模型取得的精確率為 0.8409091。

而後以 0.01 為單位在 0.1 與 0.9 之間尋找 Cutpoint 找出其值為 0.52，並將測試集資料帶入模型進行預測可得到以下結果，其精確度為 0.8522727：



	<i>Sensitivity</i>	<i>Specificity</i>	<i>AUC</i>	<i>ACC</i>
<i>Logistic Regression</i>	0.896	0.800	0.909	0.852

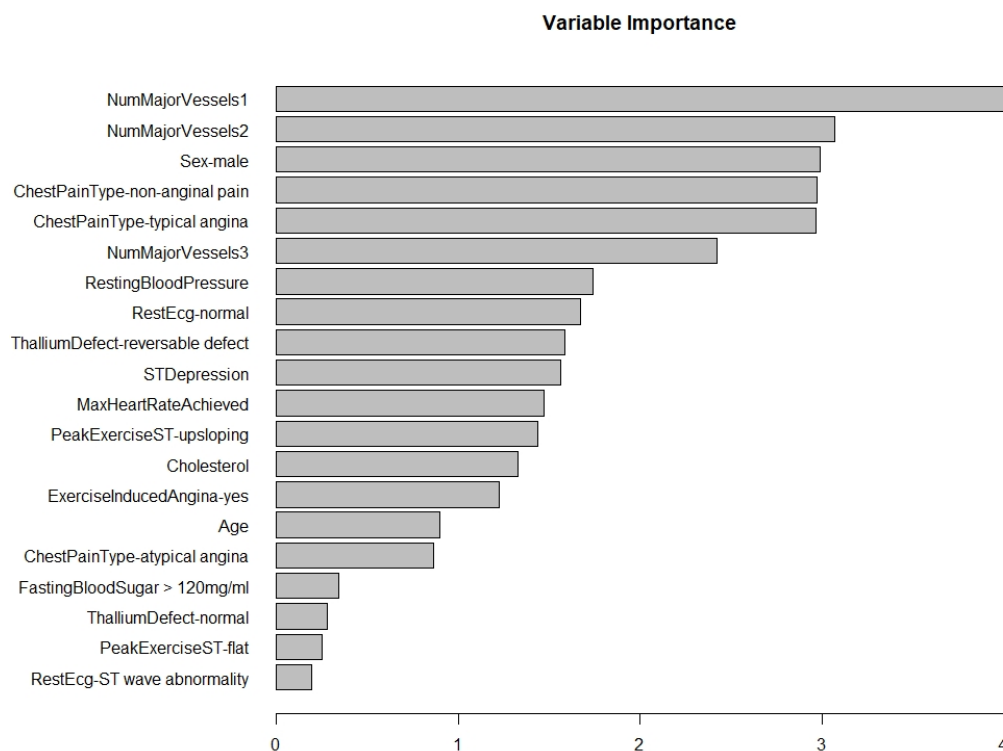
我們給出  $\beta$  的 95%信賴區間如下：



觀察  $\beta$  的 95%信賴區間，發現只有 *Sex-male*、*NumMajorVessels3*、*NumMajorVessels2*、*NumMajorVessels1*、*ChestPainType-typical angina* 與 *ChestPainType-non-anginal pain* 完全沒有包含 0。

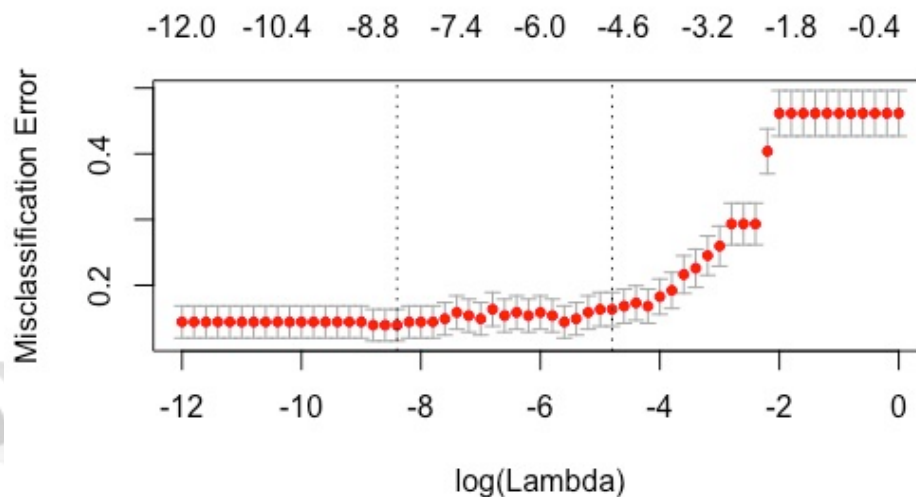
我們也想知道重要變數的優先順序，可以發現前六項與信賴區間觀察出的結果相仿，而其重要度依序為：

*NumMajorVessels*、*Sex*、*ChestPainType*、*RestingBloodPressure* 和 *RestEcg*。



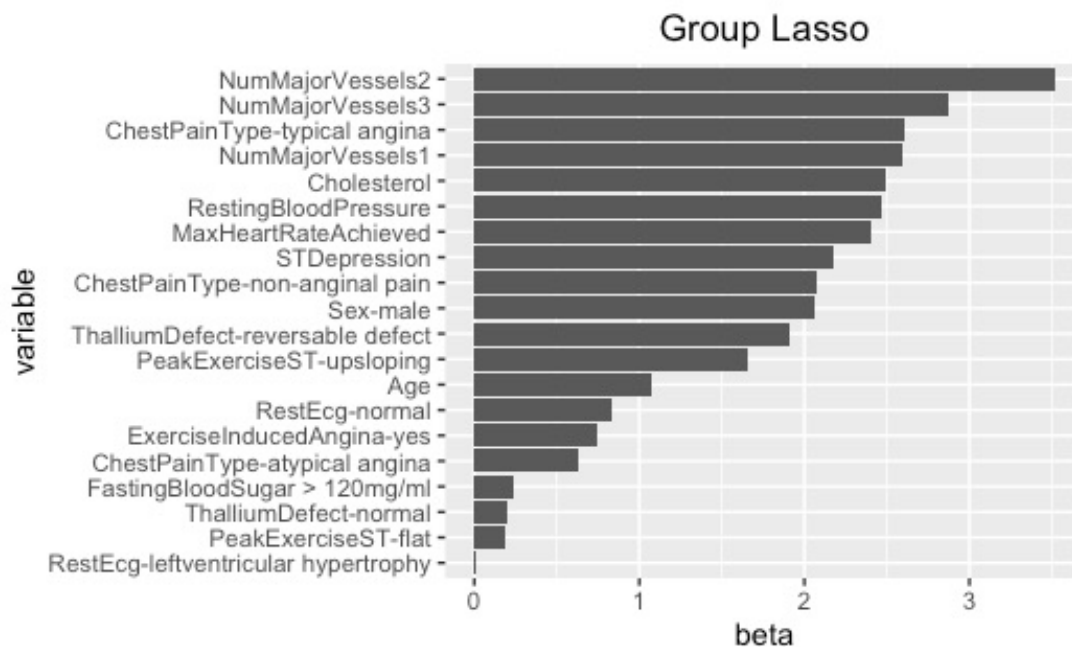
## (d) Group LASSO

跟傳統 Lasso 不同的地方在於，Group Lasso 會針對有多類別變數給予同一個 group，才不會發生同一個類別變數中，有些顯著有些不顯著的問題，因此我們給予的 group 個數，會與原本的變數個數同。首先透過 corss validation 的方式，選出適當的 $\lambda$ 值，最後選取到的 $\lambda = 0.008229747$ 。

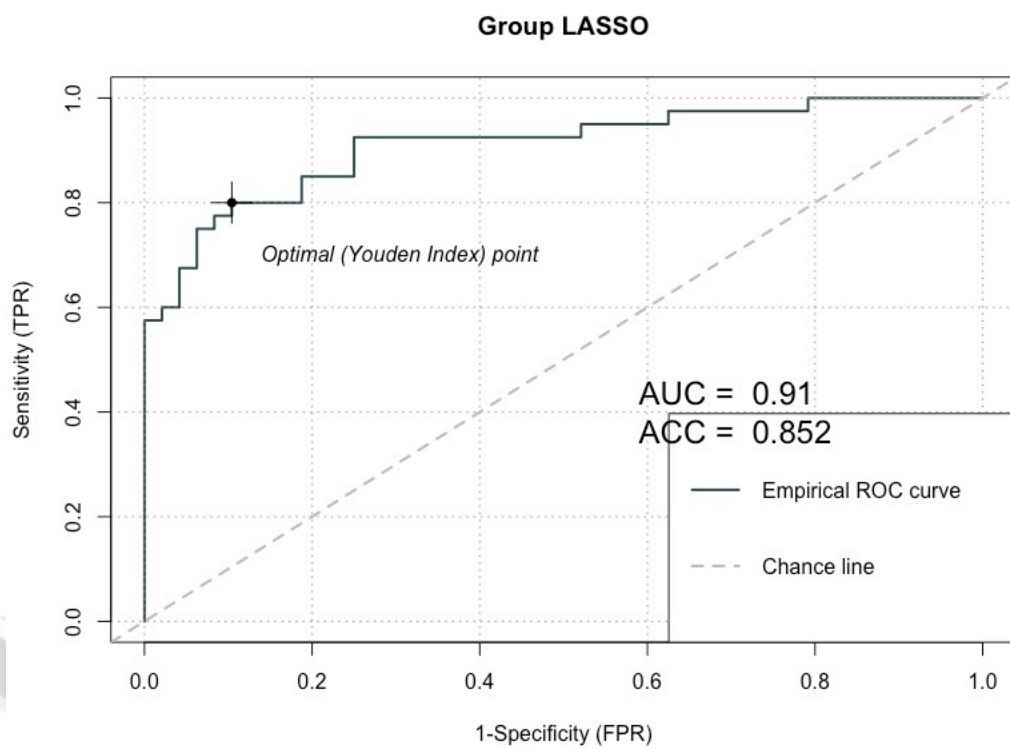


以此建立模型，我們依序列出前五大重要變數如下：

*NumMajorVessels*、*ChestPainType*、*Cholesterol*、*RestingBloodPressure*、  
*MaxHeartRateAchieved*



我們觀察此模型的表現如下：

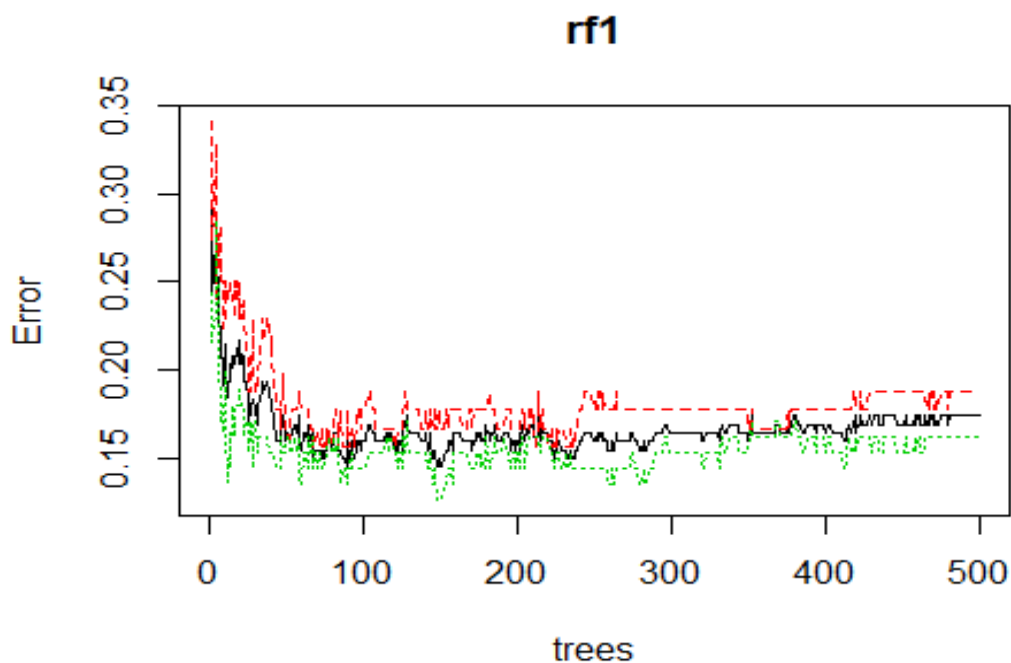


	<i>Sensitivity</i>	<i>Specificity</i>	<i>AUC</i>	<i>ACC</i>
<i>Group LASSO</i>	0.896	0.800	0.910	0.852

以模型的表現來看，其 Specificity 為 80%，Accuracy 也有 85.227%。

## (e) Random Forest

首先，我先在未調整任何參數的情形下，去看看 random forest 的表現。可以由下圖看到 random forest 的 *tree size* 大約在 100 左右誤差就會趨於穩定，因此判斷最佳的 *tree size* 大約落在 100 左右。而確實當 *tree size* = 90 時，模型有最低的 error rate，因此後續在選擇參數時，我皆固定



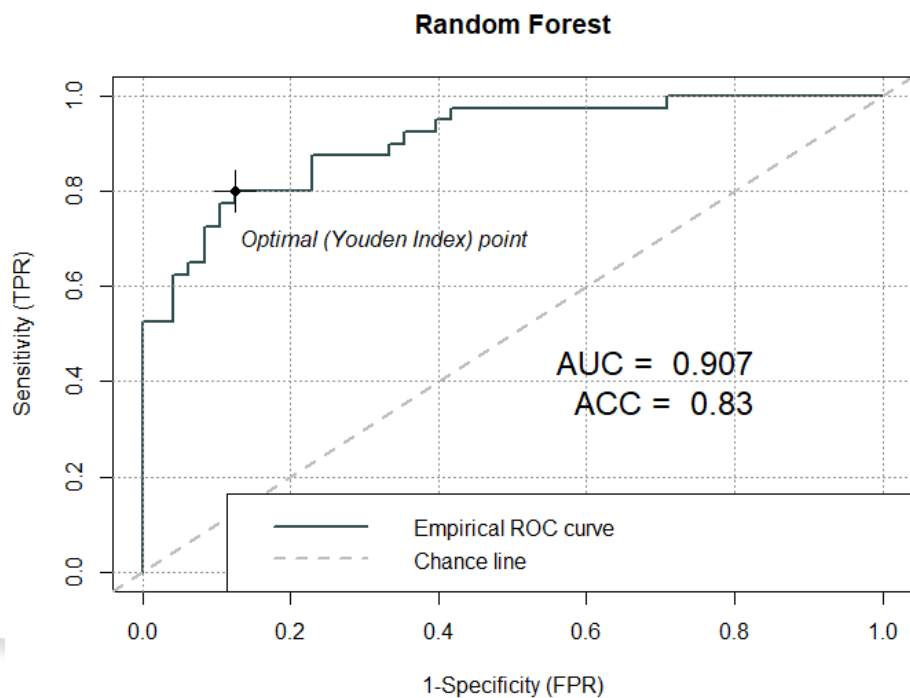
*tree size* = 90 來作選取。

接著我分別對 3 個參數：*mtry*, *node size*, *sample size* 做 grid search，此處我以 Out-of-Bag Error 作為標準進行參數調控。

下表為做完 grid search 候選取出來，根據 **OOB error rate**，選出表現前最好的參數組合。

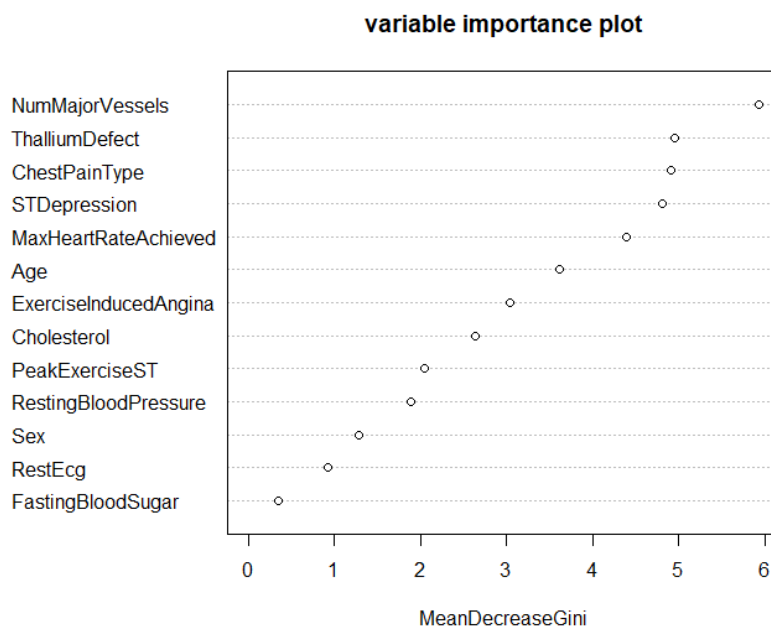
Parameters (模型參數)			
<i>ntree</i>	90	<i>mtry</i>	2
<i>nodesize</i>	7	<i>sampsiz</i>	0.55

模型配適後，將測試集資料帶入模型進行預測可得到以下結果：



	<i>Sensitivity</i>	<i>Specificity</i>	<i>AUC</i>	<i>ACC</i>
<i>Random Forest</i>	0.896	0.750	0.907	0.83

接著利用 **Mean Decrease Gini** 去選擇重要變數，可得前五個重要變數為 *Num Major Vessels*、*Thallium Defect*、*Chest Pain Type*、*ST Depression*、*Max Heart Rate Achieved*。



## (f) XGBOOST

初始參數給定 `nround` 為 200，除了調控 `eta` 以及 `max_depth` 對於樹的模型結構影響比較大的參數外，也調控 `max_depth`、`subsample`、`colsample_bytree`、`lambda` 等參數；此處使用 5-fold CV 進行 Grid Search，並以 AUC 當作選模標準進行調控。

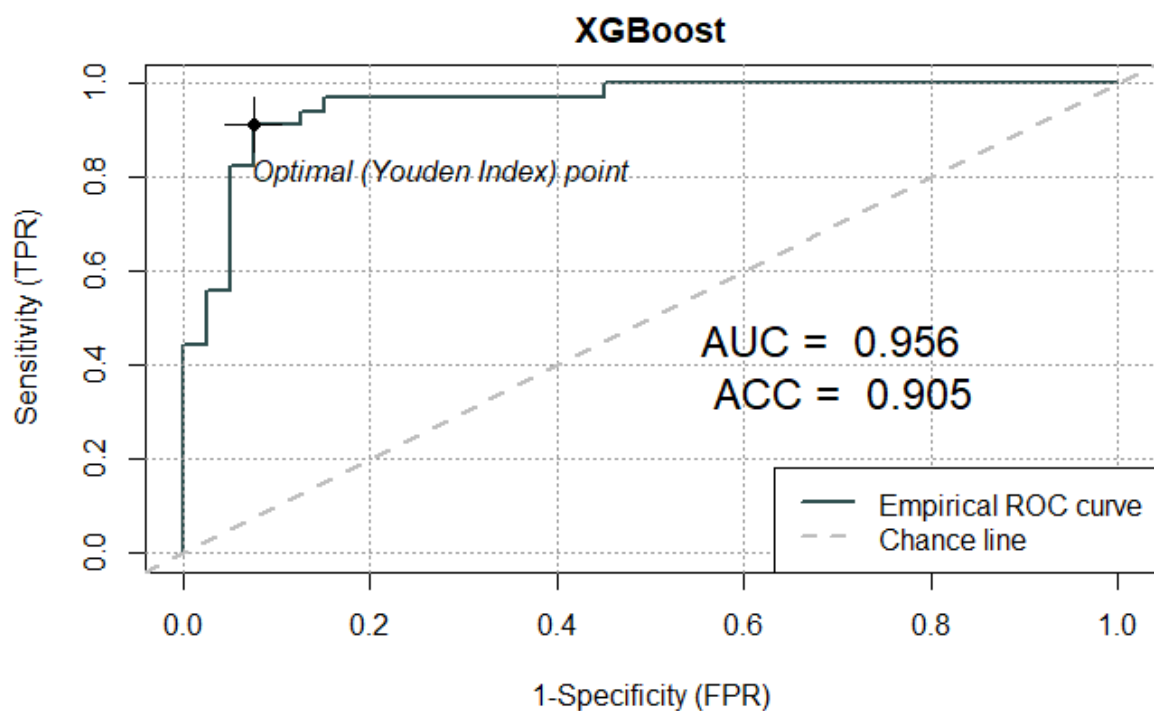
Booster Parameters (模型參數)	
<code>eta</code>	shrinkage 參數，用於更新 nodes 權重時，乘以該係數，避免步長過大。
<code>min_child_weight</code>	控制葉子節點中二階導的和的最小值，該參數值越小，越容易 overfitting。
<code>max_depth</code>	每顆樹的最大深度，樹高越深，越容易 overfitting。
<code>subsample</code>	樣本隨機採樣，較低的值使得算法更加保守，防止 overfitting，但是太小的值也會造成 underfitting。
<code>colsample_bytree</code>	列採樣，對每棵樹的生成用的特徵進行列採樣。
<code>lambda</code>	控制模型複雜度的權重值的 L2 正則化項參數，參數越大，模型越不容易 overfitting。
<code>alpha</code>	控制模型複雜度的權重值的 L1 正則化項參數，參數越大，模型越不容易 overfitting。
Learning Task Parameters (學習任務參數)	
<code>Objective</code>	定義最小化損失函數類型， <b>binary:logistic</b> : logistic regression for binary classification
<code>eval_metric</code>	The metric to be used for validation data. <b>auc</b> : Area under the curve

在調控參數中，針對每一個參數組合適配出一個模型，然後從中挑選出 AUC 最佳的模型。以下為調控解果所選擇的最佳參數，並以此進行模型配適。

Booster Parameters (模型參數)					
<code>eta</code>	0.1	<code>subsample</code>	1.0	<code>lambda</code>	0.7
<code>max_depth</code>	12	<code>colsample_bytree</code>	0.6	<code>alpha</code>	1.0

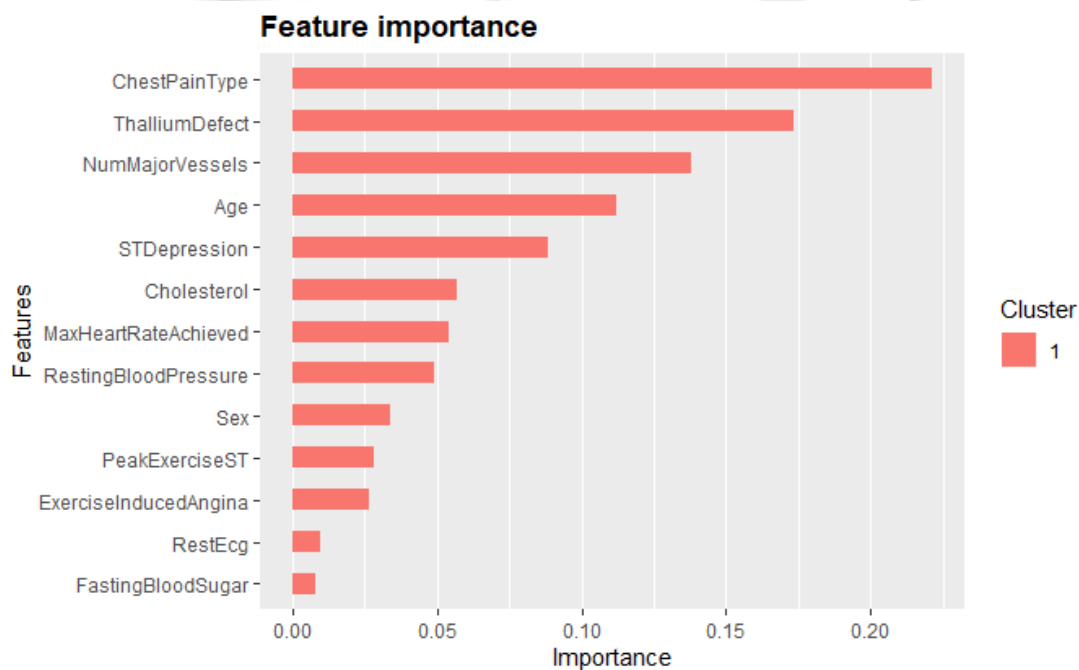
模型配適後，將測試集資料帶入模型進行預測可得到以下結果：





	<i>Sensitivity</i>	<i>Specificity</i>	<i>AUC</i>	<i>ACC</i>
XGBOOST	0.875	0.941	0.956	0.905

接著利用 **Gain** 值去選擇重要變數，可得 *Chest Pain Type*、*Thallium Defect*、*Num Major Vessels*、*Age*、*ST-Depression* 為前五個重要變數。



## ✓ 從 SHAP (SHapley Additive exPlanations) 理解模型參數

SHAP 是由 Shapley value 啟發的可加性解釋模型。對於每個預測樣本，模型都產生一個預測值，SHAP value 就是該樣本中每個特徵所分配到的數值，其由 2017 年 Lundberg & Lee 論文提出的，SHAP Value 是一種 Additive Feature Attribution Method。

**Definition 1 Additive feature attribution methods** have an explanation model that is a linear function of binary variables:

$$g(z') = \phi_0 + \sum_{i=1}^M \phi_i z'_i, \quad (1)$$

where  $z' \in \{0, 1\}^M$ ,  $M$  is the number of simplified input features, and  $\phi_i \in \mathbb{R}$ .

其中，*Shapley value* 起源於合作博弈論。比如說甲乙丙丁四個工人一起打工，甲和乙完成了價值 100 元的工件，甲、乙、丙完成了價值 120 元的工件，乙、丙、丁完成了價值 150 元的工件，甲、丁完成了價值 90 元的工件，該如何公平、合理地分配這四個人的工錢？*Shapley* 提出了一個合理的計算方法，我們稱每個參與者分配到的數額為 *Shapley value*。

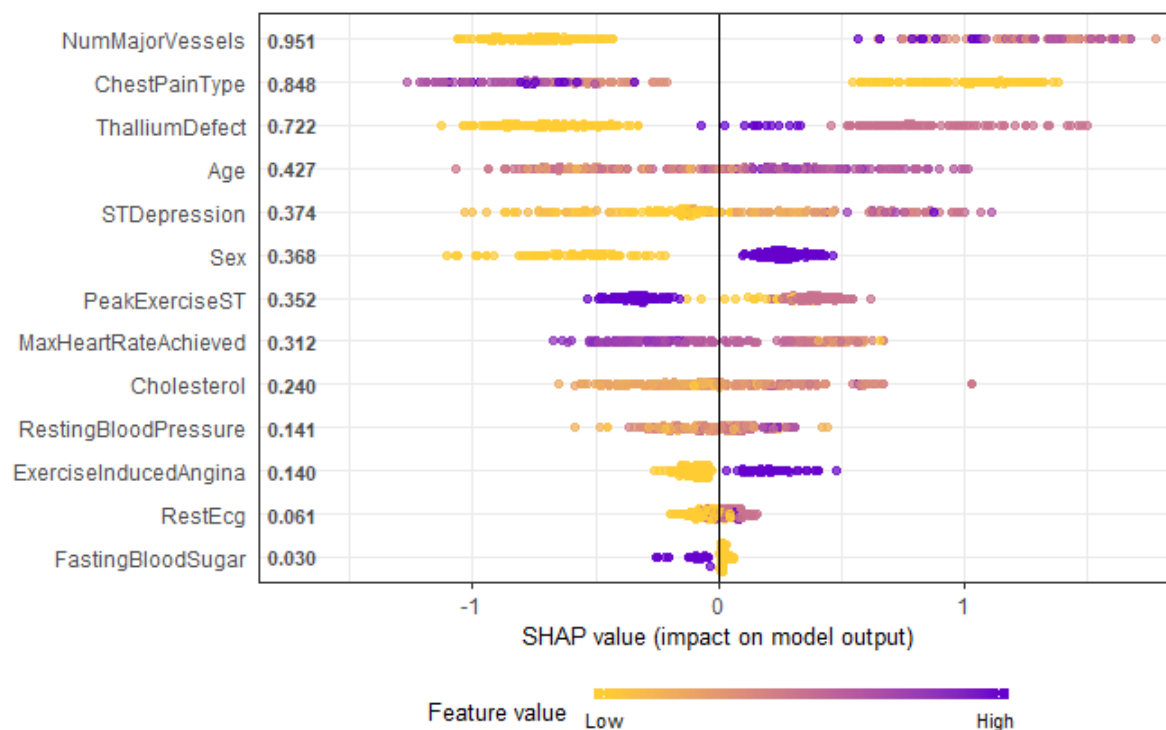
有  $n$  個元素  $x_i$ ，任意多個人形成的子集  $S \subseteq N$ ，有  $v(S)$  表示  $S$  子集中所包括的元素共同合作所產生的價值。最終分配的價值（Shapley Value） $\phi_i(N, v)$  其實是求累加貢獻（marginal contribution）的均值。例如  $A$  單獨工作產生價值  $v(\{A\})$ ，後加入  $B$  之後共同產生價值  $v(\{A, B\})$ ，那麼  $B$  的累加貢獻為  $v(\{A, B\}) - v(\{A\})$ 。

對於所有能夠形成全集  $N$  的序列，求其中關於元素  $x_i$  的累加貢獻，然後取均值即可得到  $x_i$  的 Shapley Value 值。但枚舉所有序列可能性的方式效率不高，注意到累加貢獻的計算實際為集合相減，對於同樣的集合計算次數過多是效率低下的原因。公式中  $S$  表示序列中位元於  $x_i$  前面的元素集合，進而  $N \setminus S \setminus \{x_i\}$  表示的是位於  $x_i$  後面的元素集合，而滿足只有  $S$  集合中的元素位元於  $x_i$  之前的序列總共有  $|S|!(|N| - |S| - 1)!$  個，其內序列中產生的  $x_i$  累加貢獻都是  $v(S \cup \{x_i\}) - v(S)$ ；最後對所有序列求和之後再取均值。

假設特徵全集為  $F$ ，並利用下式可以估計每一維特徵的 *Shapley Value*：

$$\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} [f_{S \cup \{i\}}(x_{S \cup \{i\}}) - f_S(x_S)].$$

與上一節中 feature importance 相比，SHAP value 最大的優勢是 SHAP 能對於反映出每一個樣本中的特徵的影響力，而且還表現出影響的正負性。

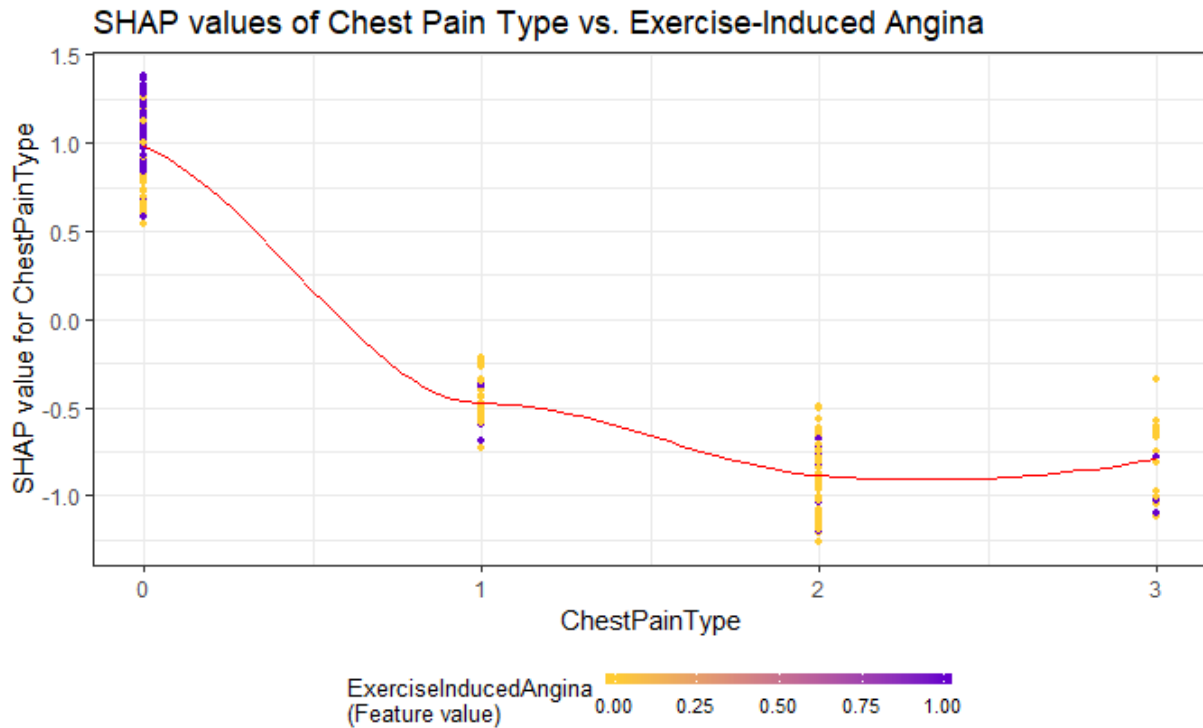


上頁圖中的樣本點，黃色代表低值域、紫色代表高值域；其中前五重要變數為：

- ✓ **Num Major Vessels**：主動脈造影判斷剝離、夾層、撕裂，越多主動脈剝離等異狀的患者有較大的可能性有心臟病。
- ✓ **Chest Pain Type**：無胸痛症狀的人中，有較大的可能性有心臟病；反之，其他類型的胸痛患者有較大的可能性有心臟病。
- ✓ **Thallium Defect**：可逆性心肌缺血與不可逆心肌缺陷的樣本內患者，有較大的可能性有心臟病。
- ✓ **Age**：年齡愈長的患者，有較大的可能性有心臟病。
- ✓ **ST-Depression**：ST 節段下降幅度越高的患者，有較大的可能性有心臟病。

## ✓ Chest Pain Type 疑點

從上頁 SHAP value 圖中，對 *Chest Pain Type* 變數解釋為：無胸痛症狀的人中，有較大的可能性有心臟病；反之，其他類型的胸痛患者有較大的可能性有心臟病。因此需進一步來探討為何無胸痛症狀的患者會有較高的機會有心臟病，並找到下圖關係。



從下圖可以發現到，無胸痛患者中(*ChestPainType* = 0)，有(過)運動後引發的心絞痛的人(*Exercise-Induced Angina* = 1，紫色點)較其他組別高許多；其代表無胸痛患者並非完全無心絞痛症狀，而是在運動時，心臟負荷因此增加、血液供應量不足，發生心絞痛症狀。

## 5. Conclusion

### (a) Covariates Importance

將各個模型選出來比較重要的前五個變數拿出來看，如下表：

Logistic Regression	Group LASSO	Random Forest	XGBOOST
<i>NumMajorVessels</i>	<i>NumMajorVessels</i>	<i>NumMajorVessels</i>	<i>ChestPainType</i>
<i>Sex</i>	<i>ChestPainType</i>	<i>Thallium Defect</i>	<i>Thallium Defect</i>
<i>ChestPainType</i>	<i>Cholesterol</i>	<i>ChestPainType</i>	<i>NumMajorVessels</i>
<i>RestingBloodPressure</i>	<i>RestingBloodPressure</i>	<i>ST Depression</i>	<i>Age</i>
<i>RestEcg</i>	<i>MaxHeartRate</i>	<i>MaxHeartRate</i>	<i>ST Depression</i>

根據上表發現 Random Forest 跟 XGboost 選到的前五個重要變數有四個是一樣的，而 *NumMajorVessels*、*ChestPainType* 是四種模型都有選到的重要變數，推測此二變數為評估心臟病的重要考量。而僅次於此二變數，*Thallium Defect*、*ST Depression*、*RestingBloodPressure*、*MaxHeartRate* 皆有二種模型選到為重要變數。

### (b) Conclusion

根據以上的分析，找到心臟病的重要考量有主動脈有剝離、撕裂的情形、胸痛(心絞痛)類型、心肌缺血情況、ST 節段異常、靜態血壓、最大心率。其中較為有疑慮的為胸痛(心絞痛)類型：其無胸痛症狀患者，在四組之中，比有心絞痛或胸痛的患者有較大的可能性有心臟病；而認知上，有心絞痛的患者患有心臟病的可能性會較無症狀的高。但若從 XGBOOST - SHAP Value 的模型解釋中，可以發現到，樣本內的無胸痛症狀患者幾乎在運動時，會有心絞痛的症狀，而可以進一步解釋此類患者，也會有心絞痛。因此，若患者有主動脈有剝離、撕裂的情形、運動時心絞痛、心肌缺血情況、ST 節段異常等狀況，患者會有較大的可能性患有心臟病，並須進行後續醫療處理。

## 附錄 1：工作分配表

組員	工作分配
蘇柏庄	XGBOOST、EDA、Code 統整、彙整結論
陳冠維	<i>Random Forest</i> 、EDA、書面報告彙整
魏志宇	<i>Group LASSO</i> 、EDA、書面報告彙整
吳岱錡	<i>KNN</i> 、 <i>Naïve Bayes</i> 、 <i>Logistic</i> 、EDA、簡報彙整
陳威宇	<i>KNN</i> 、 <i>Naïve Bayes</i> 、 <i>Logistic</i> 、EDA、簡報彙整

## 附錄 2：參考文獻

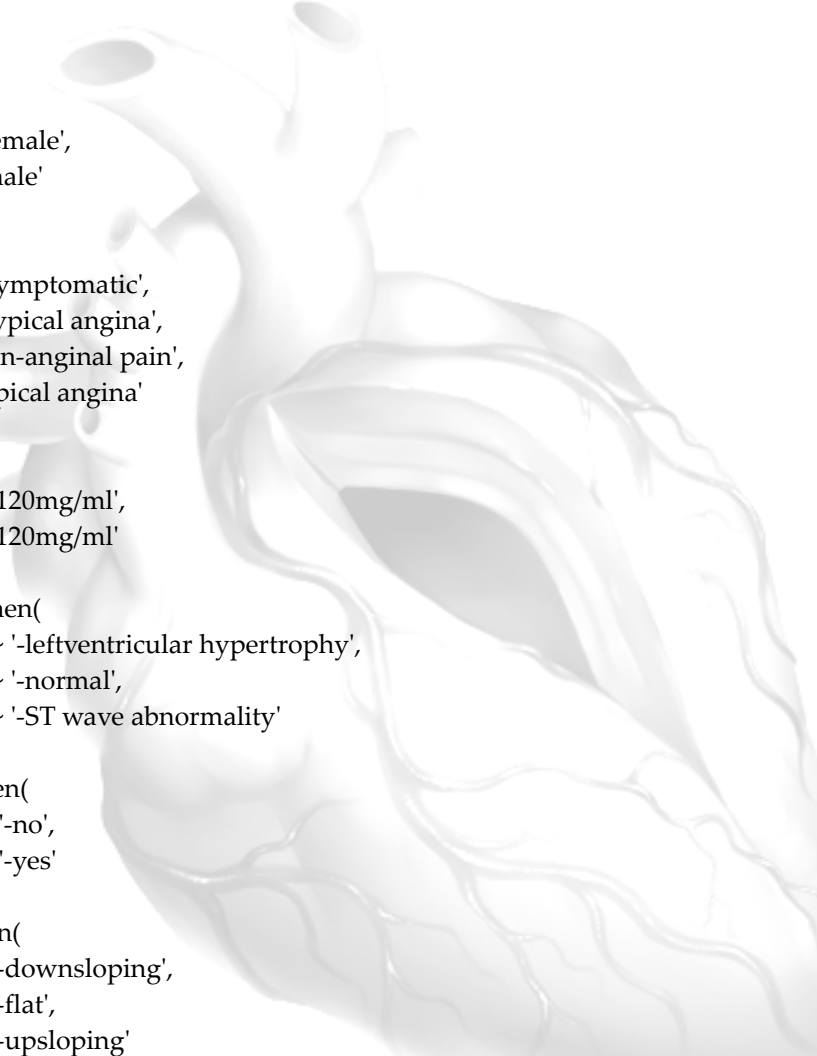
*A Unified Approach to Interpreting Model Predictions* -- Scott M. Lundberg & Su-In Lee

<http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf>

### 附錄 3：程式碼 (R)

```
## Data Preperation
### Import Data
data_ <- read.csv('./heart.csv')
colnames(data_) <- c('age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg',
                    'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target')

### Data Cleasning
data <- data_ %>%
  filter(
    thal != 0 & ca != 4
  ) %>%
  mutate(
    sex = case_when(
      sex == 0 ~ '-female',
      sex == 1 ~ '-male'
    ),
    cp = case_when(
      cp == 0 ~ '-asymptomatic',
      cp == 1 ~ '-atypical angina',
      cp == 2 ~ '-non-anginal pain',
      cp == 3 ~ '-typical angina'
    ),
    fbs = case_when(
      fbs == 0 ~ '< 120mg/ml',
      fbs == 1 ~ '> 120mg/ml'
    ),
    restecg = case_when(
      restecg == 0 ~ '-leftventricular hypertrophy',
      restecg == 1 ~ '-normal',
      restecg == 2 ~ '-ST wave abnormality'
    ),
    exang = case_when(
      exang == 0 ~ '-no',
      exang == 1 ~ '-yes'
    ),
    slope = case_when(
      slope == 0 ~ '-downsloping',
      slope == 1 ~ '-flat',
      slope == 2 ~ '-upsloping'
    ),
    thal = case_when(
      thal == 1 ~ '-fixed defect',
      thal == 2 ~ '-normal',
      thal == 3 ~ '-reversable defect'
    ),
    target = case_when(
      target == 0 ~ '> 50% diameter narrowing',
      target == 1 ~ '< 50% diameter narrowing'
    ),
    ca = as.factor(ca)
  )
```





```

) %>%
mutate_if(is.character, as.factor) %>%
dplyr::select(everything())

colnames(data) <- c('Age', 'Sex', 'ChestPainType',
                    'RestingBloodPressure', 'Cholesterol',
                    'FastingBloodSugar', 'RestEcg',
                    'MaxHeartRateAchieved', 'ExerciseInducedAngina',
                    'STDepression', 'PeakExerciseST',
                    'NumMajorVessels', 'ThalliumDefect',
                    'target')

data_xgb <- data_ %>%
  filter(
    thal != 0 & ca != 4
  ) %>%
  mutate(

    restecg = case_when(
      restecg == 0 ~ 1, #'-leftventricular hypertrophy',
      restecg == 1 ~ 0, #'-normal',
      restecg == 2 ~ 2 #'-ST wave abnormality'
    ),

    thal = case_when(
      thal == 1 ~ 2, #'-fixed defect',
      thal == 2 ~ 0, #'-normal',
      thal == 3 ~ 1 #'-reversible defect'
    ),

    target = case_when(
      target == 0 ~ '> 50% diameter narrowing',
      target == 1 ~ '< 50% diameter narrowing'
    )
  ) %>%
  mutate_if(is.character, as.factor) %>%
  dplyr::select(everything())

colnames(data_xgb) <- c('Age', 'Sex', 'ChestPainType',
                        'RestingBloodPressure', 'Cholesterol',
                        'FastingBloodSugar', 'RestEcg',
                        'MaxHeartRateAchieved', 'ExerciseInducedAngina',
                        'STDepression', 'PeakExerciseST',
                        'NumMajorVessels', 'ThalliumDefect',
                        'target')

### Train Test Split
set.seed(100)

train <- createDataPartition(data$target, p=.75, list=FALSE)
train_data <- data[train, ]
test_data <- data[-train, ]

```

```

train_xgb <- data_xgb[train, ]
test_xgb <- data_xgb[-train, ]

tr_target_ <- train_data$target
te_target_ <- test_data$target

tr_target <- as.numeric(tr_target_) - 1
te_target <- as.numeric(te_target_) - 1

tr_data <- sparse.model.matrix(target ~ .-1, data=train_xgb)
te_data <- sparse.model.matrix(target ~ .-1, data=test_xgb)

dtrain <- xgb.DMatrix(data=tr_data, label=tr_target)
dtest <- xgb.DMatrix(data=te_data, label=te_target)

### Performance Function
performance <- function(pred.prob, pred.class, method, test, positive){
  con <- confusionMatrix(pred.class, test, positive=positive)
  Sensitivity <- con$byClass[1]
  Specificity <- con$byClass[2]
  ROCit_obj <- rocit(score=pred.prob, class=test)
  AUC <- ROCit_obj$AUC
  ACC <- sum(pred.class==test)/length(test)

  plot(ROCit_obj); title(method)
  text(0.7, 0.4, paste("AUC = ", round(AUC, 3), "\n", "ACC = ", round(ACC, 3)), cex = 1.5)
  return(c(Sensitivity, Specificity, AUC = AUC, ACC = ACC))
}

## EDA
### Numerical
corr <- cor(data[sapply(data, is.numeric)], method = 'pearson')
corr

pairs(data[sapply(data, is.numeric)], lower.panel=NULL)
library(corrplot)
corrplot(corr, method = "ellipse", type="upper")

par(mfrow=c(1,5))
boxplot(formula = Age ~ target,
  data = data,
  xlab = "Disease or not",
  ylab = "Age")
boxplot(formula = Cholesterol ~ target,
  data = data,
  xlab = "Disease or not",
  ylab = "Cholesterol")
boxplot(formula = RestingBloodPressure ~ target,
  data = data,
  xlab = "Disease or not",
  ylab = "Resting Blood Pressure")

```

```

boxplot(formula = MaxHeartRateAchieved ~ target,
        data = data,
        xlab = "Disease or not",
        ylab = "Max Heart-Rate Achieved")
boxplot(formula = STDepression ~ target,
        data = data,
        xlab = "Disease or not",
        ylab = "ST-Depression")

ggplot(data, aes(x=Age, y=Cholesterol, color=target, shape=target)) +
  geom_point() +
  geom_smooth(se = FALSE)
ggplot(data, aes(x=Age, y=STDepression, color=target, shape=target)) +
  geom_point() +
  geom_smooth(se = FALSE)

ggplot(data, aes(x=Age, y=MaxHeartRateAchieved, color=target, shape=target)) +
  geom_point() +
  geom_smooth(se = FALSE)
ggplot(data, aes(x=Age, y=RestingBloodPressure, color=target, shape=target)) +
  geom_point() +
  geom_smooth(se = FALSE)

ggplot(data, aes(x=Cholesterol, y=STDepression, color=target, shape=target)) +
  geom_point() +
  geom_smooth(se = FALSE)
ggplot(data, aes(x=Cholesterol, y=MaxHeartRateAchieved, color=target, shape=target)) +
  geom_point() +
  geom_smooth(se = FALSE)

ggplot(data, aes(x=MaxHeartRateAchieved, y=STDepression, color=target, shape=target)) +
  geom_point() +
  geom_smooth(se = FALSE)
ggplot(data, aes(x=RestingBloodPressure, y=Cholesterol, color=target, shape=target)) +
  geom_point() +
  geom_smooth(se = FALSE)

data %>%
  ggplot(aes(x=Age, fill=target))+
  geom_histogram()+
  xlab("Age") +
  ylab("Number")+
  guides(fill = guide_legend(title = "target"))

data %>%
  ggplot(aes(x=Cholesterol, fill=target))+
  geom_histogram()+
  xlab("Cholesterol") +
  ylab("Number")+
  guides(fill = guide_legend(title = "target"))

```

```

data %>%
  ggplot(aes(x=RestingBloodPressure,fill=target))+
  geom_histogram()+
  xlab("RestingBloodPressure") +
  ylab("Number")+
  guides(fill = guide_legend(title = "target"))

data %>%
  ggplot(aes(x=MaxHeartRateAchieved,fill=target))+
  geom_histogram()+
  xlab("MaxHeartRateAchieved") +
  ylab("Number")+
  guides(fill = guide_legend(title = "target"))

data %>%
  ggplot(aes(x=STDepression,fill=target))+
  geom_histogram()+
  xlab("STDepression") +
  ylab("Number")+
  guides(fill = guide_legend(title = "target"))

data_disease <- data[data$target=='> 50% diameter narrowing',]
data_disease %>%
  ggplot(aes(x=Age,fill=Sex))+
  geom_histogram()+
  xlab("Age") +
  ylab("numbers of people with disease")+
  guides(fill = guide_legend(title = "Gender"))

data_disease %>%
  ggplot(aes(x=Cholesterol,fill=Sex))+
  geom_histogram()+
  xlab("Cholesterol") +
  ylab("numbers of people with disease")+
  guides(fill = guide_legend(title = "Gender"))

data_disease %>%
  ggplot(aes(x=RestingBloodPressure,fill=Sex))+
  geom_histogram()+
  xlab("RestingBloodPressure") +
  ylab("numbers of people with disease")+
  guides(fill = guide_legend(title = "Gender"))

data_disease %>%
  ggplot(aes(x=STDepression,fill=Sex))+
  geom_histogram()+
  xlab("STDepression") +
  ylab("numbers of people with disease")+
  guides(fill = guide_legend(title = "Gender"))

```

### PCA

```

pca <- prcomp(data[sapply(data, is.numeric)], scale = TRUE)
pca$rotation

plot(pca, type = "line", main = "Scree Plot")
abline(h = 1, col = "red")

vars <- (pca$sdev)^2
props <- vars / sum(vars)
cumulative_props <- cumsum(props)
cumulative_props
fviz_screplot(pca)

col <- c('#99CC00', '#CCFF00', '#99FF00', '#66FF00', '#33FF00')
x <- c('PC1', 'PC2', 'PC3', 'PC4', 'PC5')
bar <- barplot(cumulative_props, names.arg = x, xlab = 'PC',
               y_lab = 'variance explained', col = col, ylim = c(0, 1))
text(x = bar, y = round(cumulative_props, digit = 4),
     label = round(cumulative_props, digit = 4), pos = 3, cex = 1.3, col = "red")

top2_pca.data <- pca$x[, 1:2]
head(top2_pca.data)
cat('this data set dim:', dim(top2_pca.data))

pc1 <- pca$rotation[, 1]
pc2 <- pca$rotation[, 2]
par(mfrow=c(1,2))
pc1[order(pc1, decreasing = FALSE)]
dotchart(pc1[order(pc1, decreasing=FALSE)],
         main="Loading Plot for PC1",
         xlab="Variable Loadings",
         col="red")

pc2[order(pc2, decreasing = FALSE)]
dotchart(pc2[order(pc2, decreasing=FALSE)],
         main="Loading Plot for PC2",
         xlab="Variable Loadings",
         col="red")

fviz_pca(pca, habillage = data$target, addEllipses=TRUE, ellipse.level=0.95) +
  labs(title = "PCA", x = "PC1", y = "PC2")

### Categorical
bar_plot <- function(y, x){
  g1 <- ggplot(data,
               aes(x=factor(eval(parse(text=x))),
                   fill=factor(eval(parse(text=y)))) +
    geom_bar(stat='count') +
    scale_fill_brewer(palette="YlGnBu") +
    theme_hc() +
    labs(y=y,
         fill=y,

```

```

x=x)

g2 <- ggplot(data,
             aes(x=factor(eval(parse(text=x))),
                 fill=factor(eval(parse(text=y)))) +
  geom_bar(stat='count', position='fill') +
  scale_y_continuous(breaks=seq(0, 1, .2),
                    label=percent) +
  scale_fill_brewer(palette="YlGnBu") +
  theme_hc() +
  labs(y=y,
       fill=y,
       x=x)

figure <- ggarrange(g1, g2, ncol=2,
                   common.legend = TRUE, legend = "bottom")
figure
}

bar_plot('target', 'Sex')
bar_plot('target', 'ChestPainType')
bar_plot('target', 'FastingBloodSugar')
bar_plot('target', 'RestEcg')
bar_plot('target', 'ExerciseInducedAngina')
bar_plot('target', 'PeakExerciseST')
bar_plot('target', 'NumMajorVessels')
bar_plot('target', 'ThalliumDefect')

### Odds Ratio
log.odd <- function(covariate, response){
  tab <- matrix(xtabs(~ covariate + response), ncol=2)
  odr <- summary(oddsratio(tab, log=TRUE))
  est <- odr[, 1]
  sde <- odr[, 2]
  dci <- est - 1.96 * sde
  uci <- est + 1.96 * sde

  return(list(tab=tab, est=est, dci=dci, uci=uci))
}

od_sex <- log.odd(data$Sex, data$target)
od_cp <- log.odd(data$ChestPainType, data$target) #3
od_fbs <- log.odd(data$FastingBloodSugar, data$target)
od_rect <- log.odd(data$RestEcg, data$target) # 2
od_exang <- log.odd(data$ExerciseInducedAngina, data$target)
od_slope <- log.odd(data$PeakExerciseST, data$target) # 2
od_ca <- log.odd(data$NumMajorVessels, data$target) # 3
od_thal <- log.odd(data$ThalliumDefect, data$target) # 2

sex_df <- data.frame(covariate=c('sex\nfemale\nv.s.\nmale'),
                    est=c(od_sex$est), dci=c(od_sex$dci), uci=c(od_sex$uci))

```

```

cp_df <- data.frame(covariate=c('cp\nasymptomatic\nv.s.\natypical\nangina',
                                'cp\natypical\nangina\nv.s.\nnon-anginal\npain',
                                'cp\nnon-anginal\npain\nv.s.\natypical\nangina'),
                    est=c(od_cp$est), dci=c(od_cp$dci), uci=c(od_cp$uci))
fbs_df <- data.frame(covariate=c('fbs\n< 120mg/ml \nv.s.\n> 120mg/ml'),
                    est=c(od_fbs$est), dci=c(od_fbs$dci), uci=c(od_fbs$uci))
rect_df <- data.frame(covariate=c('restecg\nleft\nventricular\nhypertrophy\nv.s.\nnormal',
                                'restecg\nnormal\nv.s.\nST-T wave\nabnormality'),
                    est=c(od_rect$est), dci=c(od_rect$dci), uci=c(od_rect$uci))
exang_df <- data.frame(covariate=c('exang\nno v.s. yes'),
                    est=c(od_exang$est), dci=c(od_exang$dci), uci=c(od_exang$uci))
slope_df <- data.frame(covariate=c('slope\ndownsloping\nv.s.\nflat',
                                'slope\nflat\nv.s.\nupsloping'),
                    est=c(od_slope$est), dci=c(od_slope$dci), uci=c(od_slope$uci))
ca_df <- data.frame(covariate=c('ca\n0 v.s. 1', 'ca\n1 v.s. 2', 'ca\n2 v.s. 3'),
                    est=c(od_ca$est), dci=c(od_ca$dci), uci=c(od_ca$uci))
thal_df <- data.frame(covariate=c('thal\nfixed defect\nv.s.\nnormal',
                                'thal\nnormal\nv.s.\nreversible\ndefect'),
                    est=c(od_thal$est), dci=c(od_thal$dci), uci=c(od_thal$uci))

od <- rbind(sex_df, cp_df, fbs_df, rect_df, exang_df, slope_df, ca_df, thal_df)

od_plot <- function(df){
  ggplot(df, aes(x=covariate, y=est, group=1))+
    geom_hline(yintercept=0, color='red')+
    geom_errorbar(width=0.1, aes(ymin=dci, ymax=uci), lwd=1)+
    geom_point(shape=21, size=2, fill='white')+
    #ylim(c(-2.5,6.5))+
    #labs(title = "Confidence Interval for log odds ratio", y = "95% Confidence Interval") +
    theme(plot.title=element_text(hjust=0.5)) +
    geom_text(aes(x=covariate, y=uci, label=round(uci, 3)), nudge_y=0.3, cex=4) +
    geom_text(aes(x=covariate, y=dci, label=round(dci, 3)), nudge_y=-0.3, cex=4) +
    geom_label_repel(aes(x=covariate, y=est, label=round(est, 3)), nudge_x=0.3, cex=3.5)
}
od_plot(od)

## Naive Bayes
### Model Training
fit_nb <- naiveBayes(target~., data = train_data)
pred_nb <- predict(fit_nb, newdata = test_data, type = 'class')
cat('accuracy:', mean(pred_nb == test_data$target))

nb_model <- naiveBayes(target~.-FastingBloodSugar -RestingBloodPressure,
                      data=train_data, prob=TRUE)
nb_prob <- predict(nb_model, newdata=test_data, type='raw')[,2]
nb_pred <- ifelse(nb_prob > 0.5, '> 50% diameter narrowing',
                 '< 50% diameter narrowing')
cat('accuracy:', mean(nb_pred == test_data$target))

### Performance Summary
confusionMatrix(factor(nb_pred), factor(test_data$target))

```



```

nb_perform <- performance(nb_prob, as.factor(nb_pred),
                          'Naive Bayes', as.factor(te_target_),
                          '> 50% diameter narrowing')

## KNN
### Optimal K
train_X <- train_data[, -14]
test_X <- test_data[, -14]
k_value <- data.frame(k=seq(1, 20),
                      ACC=0)
for(i in 1:20){
  knn_pred <- knn(cl=tr_target, train=tr_data, test=te_data,
                  k=i, prob=TRUE)
  k_value$ACC[i] <- mean(knn_pred==te_target)
}

k_best <- k_value$k[which.max(k_value$ACC)]

knn_pred <- knn(cl=tr_target, train=tr_data, test=te_data,
                k=k_best, prob=TRUE)
knn_prob <- attr(knn_pred, 'prob')
knn_pred <- ifelse(knn_pred == 1, '> 50% diameter narrowing',
                  '< 50% diameter narrowing')

### Performance Summary
confusionMatrix(factor(knn_pred), factor(te_target_))
performance(knn_prob, as.factor(knn_pred),
            paste('KNN k=', k_best), as.factor(te_target_),
            '> 50% diameter narrowing')

## Logistic
### Model Training
log_model <- glm(target ~., data=train_data, family=binomial)
log_prob <- predict(log_model, test_data, type="response")
log_pred <- ifelse(log_prob > 0.5, '> 50% diameter narrowing',
                  '< 50% diameter narrowing')
mean(log_pred == test_data$target)

### Cutpoint
cutpoints <- data.frame(cut=seq(0.1, 0.9, by = 0.01),
                       ACC=0)
for(i in 1:nrow(cutpoints)){
  pred_log <- ifelse(log_prob > cutpoints$cut[i], '> 50% diameter narrowing',
                    '< 50% diameter narrowing')
  cutpoints$ACC[i] <- mean(pred_log==test_data$target)
}

cut_best <- cutpoints$cut[which.max(cutpoints$ACC)]
log_pred <- ifelse(log_prob > cut_best, '> 50% diameter narrowing',
                  '< 50% diameter narrowing')

```



```

### Performance Summary
coef <- coef(log_model)
CI <- data.frame(confint(log_model))%>%
  mutate(name=rownames(.),beta_hat=coef)
colnames(CI) <- c("L","U","name","beta_hat")

ggplot(CI[-1,], aes(x = name, y = beta_hat, group = 1)) +
  geom_hline(yintercept = 0, col='red') +
  geom_errorbar(width = 0.1, aes(ymin = L, ymax = U), lwd = 1) +
  geom_point(shape=21, size=3, fill="white")+
  theme(plot.title = element_text(hjust=0.5,size=15),
        axis.title = element_text(size=13),
        axis.text = element_text(size=13))+
  labs(title = TeX("Confidence Interval for  $\beta$ "),y = "95% Confidence Interval") +
  coord_flip()

confusionMatrix(factor(log_pred), factor(te_target_))
performance(log_prob, as.factor(log_pred),
            'Logistic', as.factor(te_target_),
            '> 50% diameter narrowing')

## Group Lasso
### Data Preparation
train_X <- model.matrix(~., train_data[, -14])[, -1]
train_X_std <- apply(train_X, 2, function(x){(x-min(x))/(range(x)[2]-range(x)[1])})
train_Y <- ifelse(tr_target==0,-1,1)
test_X <- model.matrix(~., test_data[, -14])[, -1]
test_X_std <- NULL
test_Y <- te_target_

for (i in 1:dim(test_X)[2]){
  test_X_std <- cbind(test_X_std, (test_X[, i] - min(train_X[, i]))/(range(train_X[, i])[2]- range(train_X[, i])[1]))
}

### Grouping & Training
group <- c(1:2, rep(3, 3), 4, 5, 6, rep(7, 2), 8, 9, 10, rep(11,2), rep(12, 3), rep(13, 2))
path <- gglasso(x=train_X_std, y=train_Y,
               group=group,
               lambda=exp(seq(-12, 0, by=0.2)),
               loss='logit')
apply(path$beta, 2, function(x){rownames(path$beta)[which(x == 0)]})
path.d <- data.frame(lambda=path$lambda,t(path$beta))

### CV
glcv <- cv.gglasso(x = train_X_std, y = train_Y, group = group,
                  lambda = exp(seq(-12, 0, by = 0.2)),
                  lambda.factor = 0.5,
                  loss = "logit",
                  pred.loss = "misclass")
plot(glcvc)

```

```

### Best Lambda
beta <- coef(glc_v, s = "lambda.min")
beta.name <- rownames(beta)[-1][order(abs(beta)[-1]),decreasing = T)]
est <- data.frame(name = rownames(beta)[-1],beta = abs(beta)[-1]))

### Prediction
gl_prob <- predict(glc_v, newx=test_X_std, s=glcv$lambda.min, type = "link")
gl_prob <- 1/(1 + exp(-gl_prob))
gl_pred <- ifelse(gl_prob > 0.5, '> 50% diameter narrowing',
                 '< 50% diameter narrowing')

### Performance Summary
ggplot(data=est)+
  geom_bar(aes(reorder(name, beta), beta), stat = "identity", position = "identity") +
  coord_flip() +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(title = "Group Lasso",y = "beta",x = "variable")
performance(c(gl_prob), as.factor(gl_pred),
            'XGBoost', as.factor(te_target_),
            '> 50% diameter narrowing')

## Random Forest
### Model Training
rf <- randomForest(target ~., data=train_data)
rf_pred <- predict(rf, newdata=test_data, type = 'class')
cat('accuracy:', mean(rf_pred == test_data$target))

### Grid Search CV
ntree <- which.min(rf$err.rate[, 1])
cat("best tree size:", ntree)

hyper_grid <- expand.grid(mtry = seq(2, 12, by = 2),
                          node_size = seq(3, 9, by = 2),
                          sample_size = c(0.55, 0.632, 0.7, 0.8),
                          OOB_error = 0)

for (i in 1:nrow(hyper_grid)) {
  # train model
  model <- ranger(formula = target ~ ., data = train_data,
                  num.trees = ntree,
                  mtry = hyper_grid$mtry[i],
                  min.node.size = hyper_grid$node_size[i],
                  sample.fraction = hyper_grid$sample_size[i])
  hyper_grid$OOB_error[i] <- model$prediction.error
}
min_OOB_error <- hyper_grid %>%
  dplyr::arrange(OOB_error) %>%
  head(10)

ACC_rf <- data.frame(mtry=rep(0, 10),
                     node_size=rep(0, 10),
                     sample_size=rep(0, 10),

```

```

      OOB_error=rep(0, 10),
      ACC=rep(0, 10))
for (i in 1:10){
  rf_param <- min_OOB_error[i,]

  rf_ <- randomForest(formula=target ~., data=train_data,
                      ntree=ntree,
                      mtry=rf_param$mtry,
                      nodesize=rf_param$node_size,
                      sampsize=ceiling(rf_param$sample_size * nrow(train_data)))

  rf_pred <- predict(rf_, newdata=test_data, type='class')
  acc <- mean(rf_pred==test_data$target)
  ACC_rf[i, ] <- cbind(min_OOB_error[i,], ACC=acc)
}

best_rf_param <- ACC_rf %>%
  dplyr::arrange(desc(ACC)) %>%
  head(1)

### Model After Tuning
rf_best <- randomForest(formula=target ~., data=train_data,
                      ntree=ntree,
                      mtry=best_rf_param$mtry,
                      nodesize=best_rf_param$node_size,
                      sampsize=ceiling(best_rf_param$sample_size * nrow(train_data)))

rf_prob <- predict(rf_best, test_data, type='prob')[,2]
rf_pred <- ifelse(rf_prob > 0.5, '> 50% diameter narrowing',
                 '< 50% diameter narrowing')

### Performance Summary
confusionMatrix(factor(rf_pred), factor(te_target_))
performance(rf_prob, as.factor(rf_pred),
            'Random Forest', as.factor(te_target_),
            '> 50% diameter narrowing')
imp <- randomForest::importance(rf_best)
randomForest::varImpPlot(rf_best,main="variable importance plot")

## XGBOOST
### Parameters (After Grid Search)
params <- list(booster='gbtree',
              objective='binary:logistic',
              eval_metric='auc', # aucpr, ndcg, cox-nloglik
              eta=0.1,          #learning rate
              max_depth=12,     #tree depth
              min_child_weight=1.,
              #max_delta_step=0,
              subsample=1,      #percentage data use in every iteration
              colsample_bytree=0.6, #percentage covariate use in every iteration
              lambda=0.7,

```

```

        alpha=1
    )

### Model Training
xgb_model <- xgb.train(params=params,
                      data=dtrain,
                      nrounds=200,
                      print_every_n=10,
                      watchlist=list(val=dtest, train=dtrain),
                      early_stopping_rounds=10,
                      maximize=T)

xgb_prob <- predict(xgb_model, dtest)
xgb_pred <- ifelse(xgb_prob > 0.5, '> 50% diameter narrowing',
                  '< 50% diameter narrowing')

### Performance Summary
confusionMatrix(factor(xgb_pred), factor(te_target_))
imp <- xgb.importance(feature_names=colnames(tr_data),
                     model=xgb_model)
xgb.ggplot.importance(importance_matrix=imp,
                     measure='Gain')
performance(xgb_prob, as.factor(xgb_pred),
            'XGBoost', as.factor(te_target_),
            '> 50% diameter narrowing')

xgb.plot.tree(feature_names=colnames(tr_data),
              model=xgb_model,
              trees=110) # best iteration

### Grid Search
searchGridSubCol <- expand.grid(eta=seq(0.02, 0.1, by=0.02),
                               max_depth=seq(10, 15, by=1),
                               subsample=seq(0.6, 1.0, by=0.1),
                               colsample_bytree=seq(0.6, 1.0, by=0.1),
                               lambda=seq(0.6, 1.0, by=0.1),
                               alpha=seq(0.6, 1.0, by=0.1))

ntrees <- 200

system.time(AUCHyperparameters <- apply(searchGridSubCol, 1, function(parameterList){
  #Extract Parameters to test
  currentEta <- parameterList[["eta"]]
  currentDepth <- parameterList[["max_depth"]]
  currentSubsampleRate <- parameterList[["subsample"]]
  currentColsampleRate <- parameterList[["colsample_bytree"]]
  currentLambda <- parameterList[["lambda"]]
  currentAlpha <- parameterList[["alpha"]]

  xgboostModelCV <- xgb.cv(data=dtrain, nrounds=ntrees, nfold=5, showsd=TRUE,
                           metrics = "auc", booster = "gbtree",
                           "eval_metric"="auc", "objective"="binary:logistic",

```

```

      "eta"=currentEta, "max_depth"=currentDepth,
      "subsample"=currentSubsampleRate, "colsample_bytree"=currentColsampleRate,
      "lambda"=currentLambda, "alpha"=currentAlpha,
      print_every_n=10,
      early_stopping_rounds=10)

xvalidationScores <- as.data.frame(xgboostModelCV$evaluation_log)
#auc <- tail(xvalidationScores$test_auc_mean, 2)
auc <- xvalidationScores$test_auc_mean[xgboostModelCV$best_ntreelimit]
#xgbcv$evaluation_log$test_auc_mean[xgbcv$best_ntreelimit]
output <- return(c(auc, currentEta, currentDepth, currentSubsampleRate,
                  currentColsampleRate, currentLambda, currentAlpha
                  ))))

output <- as.data.frame(t(AUCHyperparameters))
varnames <- c('TestAUC', 'eta', 'Depth', 'SubSampleRate',
              'ColSampRate', 'Lambda', 'Alpha')
names(output) <- varnames
output[which.max(output$TestAUC), ][, -1]

### SHAP Value Plot
shap.score <- function(model, tr_dmat, shap_approx=FALSE){

  shap_contrib <- predict(model, tr_dmat,
                        predcontrib=TRUE,
                        approxcontrib=shap_approx)
  shap_contrib <- as.data.table(shap_contrib)
  shap_contrib[, BIAS:=NULL]
  mean_shap_score <- colMeans(abs(shap_contrib))[order(colMeans(abs(shap_contrib)), decreasing=T)]

  return(list(shap_score=shap_contrib,
              mean_shap_score=mean_shap_score))
}

std.val <- function(x){
  return ((x - min(x, na.rm = T))/(max(x, na.rm = T) - min(x, na.rm = T)))
}

shap.long <- function(shap, tr_mat, top_n){
  if (missing(top_n)) top_n <- dim(tr_mat)[2]

  shap_score_sub <- as.data.table(shap$shap_score)
  shap_score_sub <- shap_score_sub[, names(shap$mean_shap_score)[1:top_n], with = F]
  shap_score_long <- melt.data.table(shap_score_sub, measure.vars = colnames(shap_score_sub))

  fv_sub <- as.data.table(as.matrix(tr_mat))[, names(shap$mean_shap_score)[1:top_n], with = F]
  fv_sub_long <- melt.data.table(fv_sub, measure.vars = colnames(fv_sub))
  fv_sub_long[, stdfvalue := std.val(value), by = "variable"]

  names(fv_sub_long) <- c("variable", "rfvalue", "stdfvalue" )
  shap_long <- cbind(shap_score_long, fv_sub_long[,c('rfvalue','stdfvalue')])
  shap_long[, mean_value := mean(abs(value)), by = variable]
  setkey(shap_long, variable)

```

```

    return(shap_long)
}

shap_values <- shap.score(model=xgb_model,
                        tr_dmat=tr_data)
shap_long <- shap.long(shap=shap_values,
                      tr_mat=as.matrix(tr_data),
                      top_n=13)
shap.plot.summary(shap_long, x_bound = 1.7)

g1 <- shap.plot.dependence(data_long = shap_long,
                          x = 'ChestPainType', y = 'ChestPainType',
                          color_feature = 'ExerciseInducedAngina') +
  ggtitle("SHAP values of Chest Pain Type vs. Exercise-Induced Angina")

g1

```

