MATH 151 Final Project

Predicting Heart Disease Status Using Logistic Regression and Clustering Algorithms

Professor Antonio Punzo

Phuong Huynh, Bufan Zhou, and Phuc Thinh Nguyen

May 9th, 2024

1 Objectives and motivations

This project aims to forecast whether a person will develop heart disease using certain predictor variables. The CDC identifies numerous factors contributing to heart disease, such as lifestyle choices and health conditions. Major risk factors include high blood pressure, high cholesterol, diabetes, smoking, obesity, poor diet, and lack of exercise. Additionally, age, family history, race/ethnicity, and gender may also play significant roles in determining heart disease risk [1]. Heart disease has many types: congestive heart, blood vessel disease, etc. In this practice, we will group all types of heart disease into binary outcome of having heart disease and not having heart disease.

We're using a dataset called "Heart Failure Prediction" from the Kaggle community. It gives us helpful information about what might cause someone to get heart disease. Our main reason for doing this is because we want to find out who might get heart disease early so we can help them prevent it. By understanding what factors make someone more likely to have heart problems, we can give doctors better ways to help their patients. This could mean suggesting changes in lifestyle, giving personalized treatments, or even finding new ways to stop heart disease before it happens.

Our hope is that by doing this project, we can make it easier to keep people healthy and lower the number of heart problems in the future.

2 Analyze univariate distributions

1. Loading Data and Preliminary Analysis

Through preliminary analysis of the dataset, all 918 rows of the dataset are important as no NA values are detected. We then first illustrate 10 first rows of the dataset.

```
> heart <- read.csv("/Users/Scottie/Downloads/heart.csv")
> head(heart, n = 10)
   Age Sex ChestPainType RestingBP Cholesterol FastingBS RestingECG MaxHR ExerciseAngina Oldpeak ST_Slope HeartDisease
1   40   M           ATA       140         289         0     Normal   172              N     0.0       Up            0
2   49   F           NAP       160         180         0     Normal   156              N     1.0     Flat            1
3   37   M           ATA       130         283         0         ST    98              N     0.0       Up            0
4   48   F           ASY       138         214         0     Normal   108              Y     1.5     Flat            1
5   54   M           NAP       150         195         0     Normal   122              N     0.0       Up            0
6   39   M           NAP       120         339         0     Normal   170              N     0.0       Up            0
7   45   F           ATA       130         237         0     Normal   170              N     0.0       Up            0
8   54   M           ATA       110         208         0     Normal   142              N     0.0       Up            0
9   37   M           ASY       140         207         0     Normal   130              Y     1.5     Flat            1
10  48   F           ATA       120         284         0     Normal   120              N     0.0       Up            0
```

We then look at the structure of the heart dataset. We notice that 918 objects are each described using 12 given variables. The HeartDisease variable is the outcome which refers to the status of a person's heart disease. This response variable has a binary type with value 0 indicating the absent in the disease and 1 otherwise.

Besides, we have categorical variables are:

- ChestPainType:
  - TA: Typical Angina. Angina is chest pain or discomfort caused by poor blood flow to the heart muscle.
  - ATA: Atypical Angina. Atypical chest pain is a type of chest pain that only has some of the features of typical angina chest pain.
  - NAP: Non-Anginal Pain. Non-anginal chest pain, also known as non-cardiac chest pain (NCCP), is chest pain that occurs in patients who do not have heart disease.

  - ASY: Asymptomatic. Asymptomatic chest pain, also known as silent myocardial ischemia (SMI), is a brief and mild condition where the heart receives less oxygen-rich blood flow.

- ST.slope: The slope of the peak exercise ST segment [2]
  - Up: upsloping. The normal ST segment has a slight upward concavity
  - Flat: flat
  - Down: downsloping

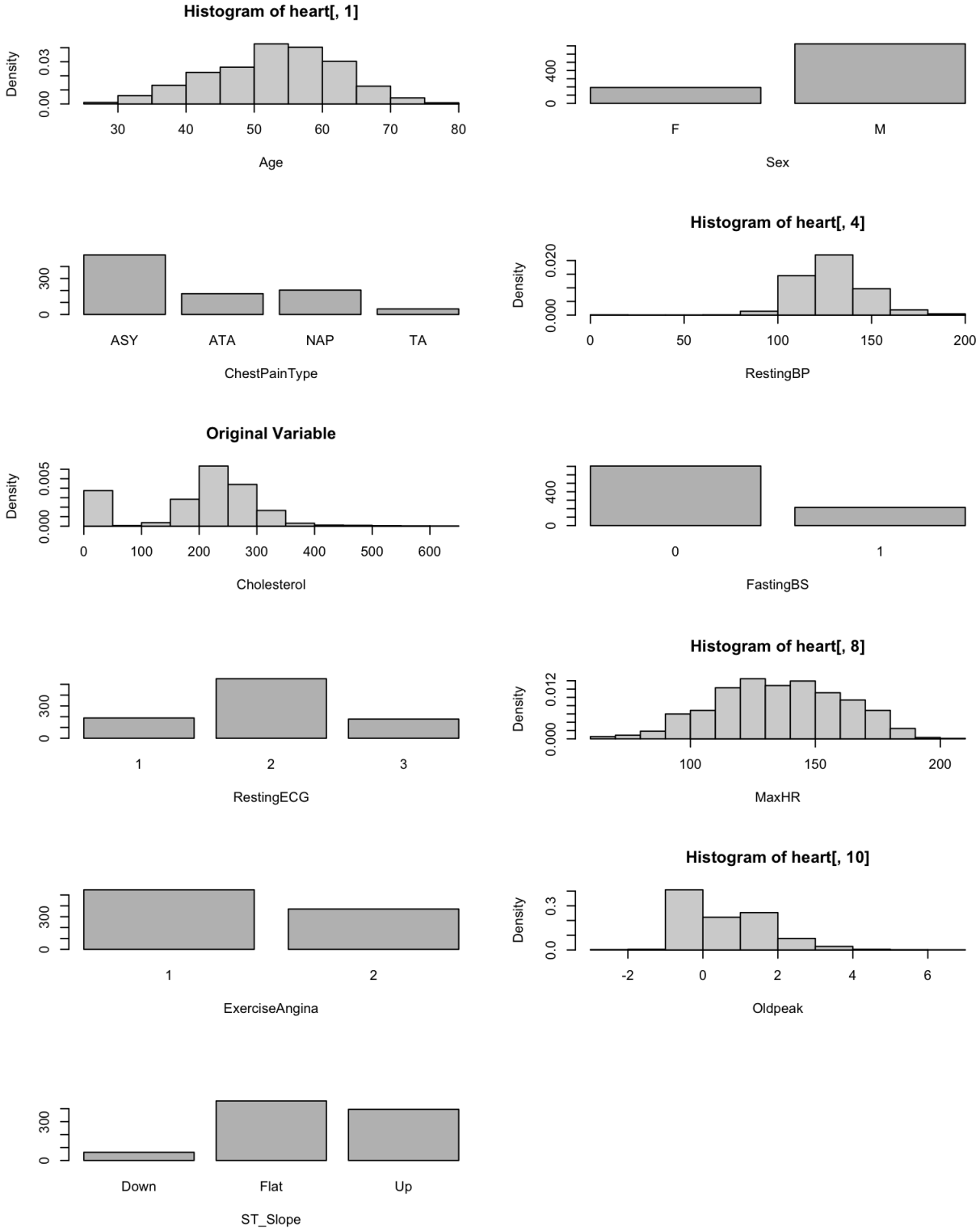
- Resting ECG: A resting electrocardiogram (ECG) is a non-invasive test that measures the electrical activity of the heart while the patient is calm and lying down
  - Normal: the heart is beating at an even rate of 60 to 100 beats per minute and in a normal condition.
  - ST: ST-T wave abnormality refers to an irregularity or deviation from the normal pattern observed on an electrocardiogram (ECG or EKG) recording
  - LVH: LVH stands for Left Ventricular Hypertrophy, a condition characterized by the thickening or enlargement of the muscle wall of the heart's left ventricle.

  Binary variables include:
  - sex: Male or Female
  - FastingBS: A fasting blood sugar test measures your blood sugar after you haven't eaten anything for at least eight hours or overnight. A normal fasting blood sugar level is 99 mg/dL or lower. A level of 100 to 125 mg/dL indicates prediabetes, and 126 mg/dL or higher indicates diabetes [4]. Here, the value is 1 if FastingBS ¿ 120 mg/dl, and 0 otherwise.
  - ExerciseAngina: Exercise-induced angina is a type of angina that occurs when the heart works harder to pump blood to the body during exertion, causing chest pain.

  Numerical variables are: RestingBP, Cholesterol, MaxHR, and OldPeak: ST depression caused by activity in comparison to rest.

```
> str(heart)
'data.frame':   918 obs. of  12 variables:
 $ Age           : int  40 49 37 48 54 39 45 54 37 48 ...
 $ Sex           : chr  "M" "F" "M" "F" ...
 $ ChestPainType : chr  "ATA" "NAP" "ATA" "ASY" ...
 $ RestingBP     : int  140 160 130 138 150 120 130 110 140 120 ...
 $ Cholesterol   : int  289 180 283 214 195 339 237 208 207 284 ...
 $ FastingBS     : int  0 0 0 0 0 0 0 0 0 0 ...
 $ RestingECG    : chr  "Normal" "Normal" "ST" "Normal" ...
 $ MaxHR         : int  172 156 98 108 122 170 170 142 130 120 ...
 $ ExerciseAngina: chr  "N" "N" "N" "Y" ...
 $ Oldpeak       : num  0 1 0 1.5 0 0 0 0 1.5 0 ...
 $ ST_Slope      : chr  "Up" "Flat" "Up" "Flat" ...
 $ HeartDisease  : int  0 1 0 1 0 0 0 0 1 0 ...
>
```

2. Initial observations



Through the variables plot, age and MaxHR seems to have normal distribution and do not have any unusual observations. The odds appear here are the value zero existed in cholesterol and resting blood pressure. These values should not be zero as they indicate the heart's rate per minute. For further analysis, we decided to replace one zero value by the median for the resting blood pressure variable. For the cholesterol variable, there are many value zeros. We decided to replace them with the reasonable statistic values (mean or median), but first, we need to find its closest type of distribution.

The candidates distributions for this univariate analysis are Gamma, Normal, and Cauchy. The methods to find the distribution for this variable include maximum likelihood,

Kolmogorov-Smirnov test, and Akaike Information Criterion.

The steps will be:

- using the cholesterol column excluding 0 values to find the parameters for each distribution using maximum likelihood estimation.
- using AIC (trying to maximize) to compare the models

```
> remove_0_cholesterol <- heart[heart,"Cholesterol"] != 0, "Cholesterol"]
> cholesterol_dist_normal <- fitdistrplus::fitdist(remove_0_cholesterol, "norm","mle")
> cholesterol_dist_cauchy <- fitdistrplus::fitdist(remove_0_cholesterol, "cauchy","mle")
> cholesterol_dist_gamma <- fitdistrplus::fitdist(remove_0_cholesterol, "gamma","mle")
>
> #trying to maximize
> AIC_norm <- 2*cholesterol_dist_normal$loglik - 2*2
> AIC_norm
[1] -10092.33
> AIC.cauchy <- 2*cholesterol_dist_cauchy$loglik - 2*2
> AIC.cauchy
[1] -10259.55
> AIC.gamma <- 2*cholesterol_dist_gamma$loglik - 2*2 #largest so far
> AIC.gamma
[1] -10014.36
```

- using one-sample Kolmogorov-Smirnov test to test for the goodness-of-fit, and

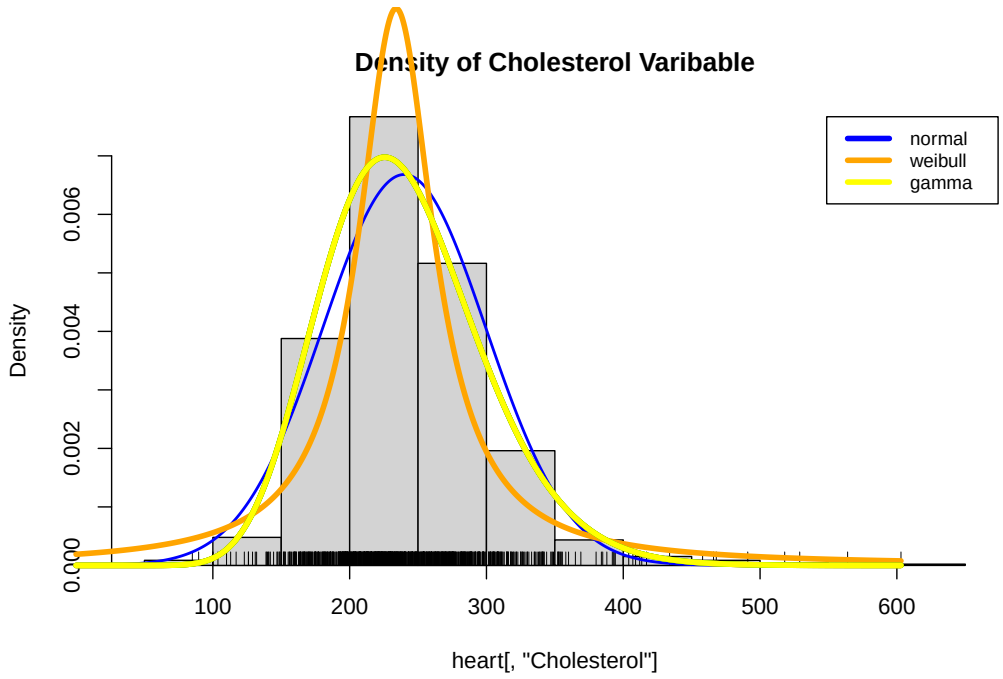
```
ks.normal <- ks.test(remove_0_cholesterol, "plnorm",
                     meanlog = cholesterol_dist_normal$estimate[1],
                     sdlog = cholesterol_dist_normal$estimate[2])
ks.cauchy <- ks.test(remove_0_cholesterol, "pcauchy",
                      shape = cholesterol_dist_weibull$estimate[1],
                     rate = cholesterol_dist_weibull$estimate[2])
ks.gamma <- ks.test(remove_0_cholesterol, "pgamma",
                    shape = cholesterol_dist_gamma$estimate[1],
                    rate = cholesterol_dist_gamma$estimate[2])
```

The result for KS test is:

```
> ks.normal

        Asymptotic one-sample Kolmogorov-Smirnov test

data:  remove_0_cholesterol
D = 0.99997, p-value < 2.2e-16
alternative hypothesis: two-sided

> ks.cauchy

        Asymptotic one-sample Kolmogorov-Smirnov test

data:  Loss
D = 0.222, p-value < 2.2e-16
alternative hypothesis: two-sided

> ks.gamma

        Asymptotic one-sample Kolmogorov-Smirnov test

data:  remove_0_cholesterol
D = 0.032191, p-value = 0.2974
alternative hypothesis: two-sided
```

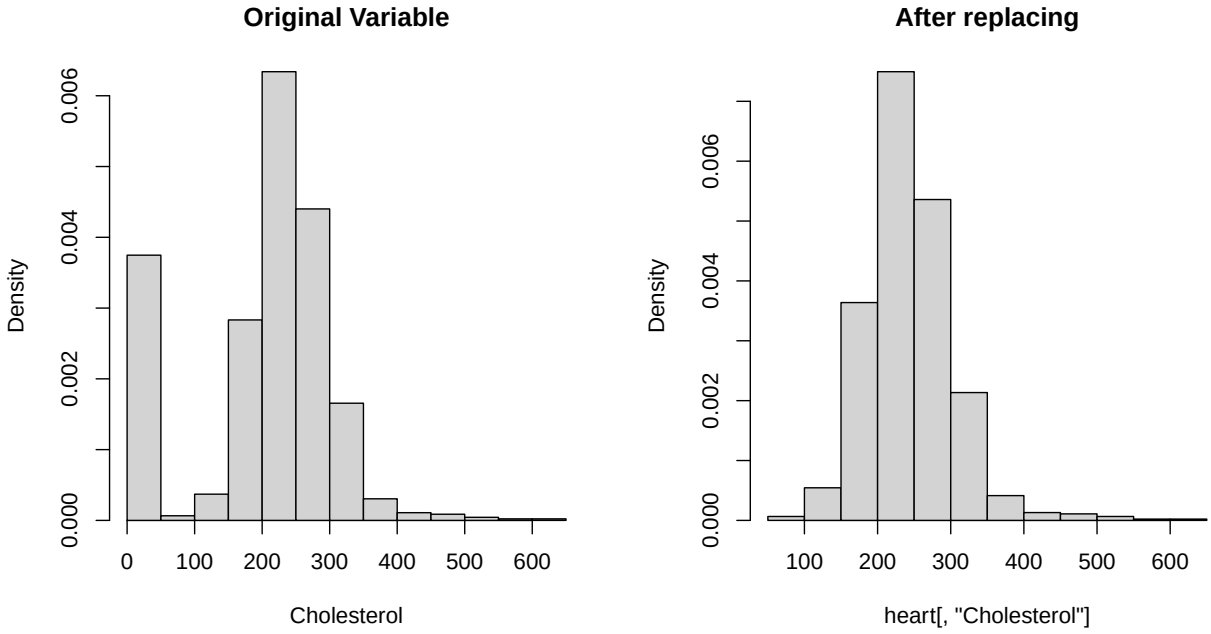
- using goodness-of-fit Pearson's Chi-square test,

```
> # goodness of fit test
> dist <- list (cholesterol_dist_normal,
+               cholesterol_dist_cauchy,
+               cholesterol_dist_gamma)
> res <- fitdistrplus :: gofstat(f = dist ,
+                                fitnames = c("norm","Cauchy", "Gamma"))
> qchisq(p = .9999999999, df = length(remove_0_cholesterol)-1, lower.tail = FALSE)
[1] 525.1118
> res$chisq
     norm    Cauchy     Gamma
 49.57820 166.64966  21.54711
```



Based on AIC, one-sample Kolmogorov-Smirnov test, and goodnesss-fit-fit test, Gamma distribution has the highest AIC score, the largest p-value for the KS-test, and the lowest Chi-square test statistics leading to fail to reject, the best candidate distribution for Cholesterol variable is Gamma distribution. After finding the best fitted model and the random generated values to replace zeros, we now can proceed to the process of using ML algorithms to predict a person's heart disease status.

- generate random values based on the best candidate model and its parameters.



3 Logistic Regression

In statistics, it is necessary to train the model using a training dataset, and then perform model validation via a test dataset. Therefore, we will split the "heart" dataset into two parts, in which we will use 80% of the dataset for training purposes, and the remaining rows for testing purposes.

First of all, based on our univariate distribution analysis above, we replaced the zero values of RestingBP with the median of the values of RestingBP in the dataset, and display the first 50 values of RestingBP.

```
> heart[heart,"RestingBP"] == 0, "RestingBP"] <- mean(heart[,"RestingBP"])
> head(heart[,"RestingBP"], n = 50)
 [1] 140 160 130 138 150 120 130 110 140 120 130 136 120 140 115 120 110 120 100 120 100 120 124 150 130 130 124 120 113
[30] 125 145 130 125 130 150 125 140 110 120 150 150 130 150 140 120 130 120 140 112 110
>
```

After that, we replaced the zero values of RestingBP with the values that fit the Gamma distribution. The histogram above in Page 6 shows that there are no longer zero values in the Cholesterol column.

```
> remove_0_cholesterol <- heart[heart[,"Cholesterol"] != 0, "Cholesterol"]
> cholesterol_dist_gamma <- fitdistrplus::fitdist(remove_0_cholesterol, "gamma","mle")
> heart[heart[,"Cholesterol"] == 0, "Cholesterol"] <- rgamma(length(heart[heart[,"Cholesterol"] == 0, "Cholesterol"] ),
+                                                           shape =cholesterol_dist_gamma$estimate[1],
+                                                           cholesterol_dist_gamma$estimate[2])
>
```

After that, since 80% of 918, the number of rows in the dataset, is 734.4, we would fix the size of the training dataset as 750, and we would create a random training dataset using a seed to fix the training dataset. We then display the first 10 rows of the training dataset along with a line of code to confirm the length of the training data for certainty purposes.

```
> set.seed(123)
> smp_size <- sample(seq(1,length(heart[,1]),1), 750)
> train_test <- heart[smp_size, ]
> head(train_test, n = 10)
    Age Sex ChestPainType RestingBP Cholesterol FastingBS RestingECG MaxHR ExerciseAngina Oldpeak ST_Slope HeartDisease
415  54   M           ASY       130    186.9324         1     Normal   110              Y     3.0     Flat            1
463  59   M           ASY       122    233.0000         0     Normal   117              Y     1.3     Down            1
179  37   M           NAP       130    194.0000         0     Normal   150              N     0.0       Up            0
526  45   M           NAP       130    236.0000         0     Normal   144              N     0.1       Up            0
195  41   F           ATA       125    184.0000         0     Normal   180              N     0.0       Up            0
818  60   M           ASY       125    258.0000         0        LVH   141              Y     2.8     Flat            1
118  59   F           ASY       130    338.0000         1         ST   130              Y     1.5     Flat            1
299  51   M           ASY       110    233.1501         1     Normal    92              N     0.0     Flat            1
229  41   M           ATA       120    295.0000         0     Normal   170              N     0.0       Up            0
244  43   F           ATA       120    266.0000         0     Normal   118              N     0.0       Up            0
> length(train_test[,1])
[1] 750
>
```

Subsequently, we define the test dataset as the dataset that contains the data that is not in the training dataset. We would also display the first 10 rows of the test dataset along with a line of code to confirm the length of the test data for certainty purposes.

```
> test <- heart[!seq(1,length(heart[,1]),1) %in% smp_size, ]
> head(test, n = 10)
   Age Sex ChestPainType RestingBP Cholesterol FastingBS RestingECG MaxHR ExerciseAngina Oldpeak ST_Slope HeartDisease
1   40   M           ATA       140         289         0     Normal   172              N       0       Up             0
3   37   M           ATA       130         283         0         ST    98              N       0       Up             0
7   45   F           ATA       130         237         0     Normal   170              N       0       Up             0
12  58   M           ATA       136         164         0         ST    99              Y       2     Flat             1
22  44   M           ATA       120         184         0     Normal   142              N       1     Flat             0
27  53   M           ASY       124         260         0         ST   112              Y       3     Flat             0
28  52   M           ATA       120         284         0     Normal   118              N       0       Up             0
32  56   M           NAP       130         167         0     Normal   114              N       0       Up             0
35  43   F           ATA       150         186         0     Normal   154              N       0       Up             0
43  35   M           ATA       150         264         0     Normal   168              N       0       Up             0
> length(test[,1]) #Expecting 168 since 918 - 750 = 168
[1] 168
> ## Expecting 168 since 918 - 750 = 168
```

After building the training and test dataset, we then turn our attention to model implementation. Since we are trying to predict whether or not a person has a heart attack, we are working on binary classification where 0 classifies a person who does not have a heart attack, and 1 otherwise. Therefore, since the most common model to model the probability of binary outcome is logistic regression, we firstly proceed with logistic regression.

First of all, for notation simplicity, we attach the dataset to R so that we can directly use the variables in the dataset

$$\textbf{attach}(\text{heart})$$

We then use forward selection to step-by-step add the variables that has the most importance based on the p-values of the logistic regression model. As a definition, all variables with a p-value at least 0.01 is considered "important". We first apply logistic regression on the null model to predict the outcome of the binary variable HeartDisease training dataset, and we use forward selection for our first attempt to determine which variables are important.

```
> training_model <- glm(HeartDisease ~ 1, family = binomial, data = train_test)
> add1(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType + RestingBP
+      + Cholesterol + FastingBS + RestingECG + MaxHR + ExerciseAngina
+      + Oldpeak + ST_Slope, test = "F")
Single term additions

Model:
HeartDisease ~ 1
              Df Deviance     AIC  F value    Pr(>F)
<none>            1030.74 1032.74
Age            1   966.58  970.58  49.6484 4.180e-12 ***
Sex            1   957.52  961.52  57.1960 1.158e-13 ***
ChestPainType  3   775.40  783.40  81.8858 < 2.2e-16 ***
RestingBP      1  1018.54 1022.54   8.9593  0.002852 **
Cholesterol    1  1018.16 1022.16   9.2369  0.002454 **
FastingBS      1   985.88  989.88  34.0376 8.047e-09 ***
RestingECG     2  1020.35 1026.35   3.8006  0.022790 *
MaxHR          1   903.78  907.78 105.0792 < 2.2e-16 ***
ExerciseAngina 1   842.27  846.27 167.3762 < 2.2e-16 ***
Oldpeak        1   889.49  893.49 118.7762 < 2.2e-16 ***
ST_Slope       2   725.83  731.83 156.8990 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Warning message:
In add1.glm(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType +  :
  F test assumes quasibinomial family
>
```

From the results above, RestingECG is considered unimportant since its p-value is 0.0227, which is larger than our suggested cutoff 0.01. Therefore, we would choose one of the remaining variables (which will be **Age**) to update our current logistic regression model, and continue the process until the displayed variables are all no longer important.

```
> training_model <- glm(HeartDisease ~ 1 + Age, family = binomial,
+                       data = train_test)
> add1(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType + RestingBP
+      + Cholesterol + FastingBS + RestingECG + MaxHR + ExerciseAngina
+      + Oldpeak + ST_Slope, test = "F")
Single term additions

Model:
HeartDisease ~ 1 + Age
               Df Deviance    AIC  F value    Pr(>F)
<none>              966.58 970.58
Sex             1   895.81 901.81  59.0149 4.919e-14 ***
ChestPainType   3   743.75 753.75  74.4019 < 2.2e-16 ***
RestingBP       1   964.23 970.23   1.8176   0.17801
Cholesterol     1   957.47 963.47   7.1104   0.00783 **
FastingBS       1   937.24 943.24  23.3879 1.608e-06 ***
RestingECG      2   963.00 971.00   1.3886   0.25007
MaxHR           1   883.87 889.87  69.9066 3.033e-16 ***
ExerciseAngina  1   809.55 815.55 144.9004 < 2.2e-16 ***
Oldpeak         1   861.24 867.24  91.3630 < 2.2e-16 ***
ST_Slope        2   702.71 710.71 140.0631 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Warning message:
In add1.glm(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType +  :
  F test assumes quasibinomial family
> ## Choose Sex as the next variable in the model and continue the process
```

```
> training_model <- glm(HeartDisease ~ 1 + Age + Sex, family = binomial,
+                       data = train_test)
> add1(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType + RestingBP
+      + Cholesterol + FastingBS + RestingECG + MaxHR + ExerciseAngina
+      + Oldpeak + ST_Slope, test = "F")
Single term additions

Model:
HeartDisease ~ 1 + Age + Sex
               Df Deviance    AIC  F value    Pr(>F)
<none>              895.81 901.81
ChestPainType   3   698.01 710.01  70.2785 < 2.2e-16 ***
RestingBP       1   893.85 901.85   1.6370  0.201138
Cholesterol     1   879.66 887.66  13.6939  0.000231 ***
FastingBS       1   873.14 881.14  19.3708 1.233e-05 ***
RestingECG      2   893.03 903.03   1.1589  0.314400
MaxHR           1   830.61 838.61  58.5569 6.112e-14 ***
ExerciseAngina  1   761.68 769.68 131.3726 < 2.2e-16 ***
Oldpeak         1   800.02 808.02  89.3254 < 2.2e-16 ***
ST_Slope        2   643.92 653.92 145.7131 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Warning message:
In add1.glm(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType +  :
  F test assumes quasibinomial family
> ## Choose ChestPainType as the next variable in the model and continue the process
```

```
> training_model <- glm(HeartDisease ~ 1 + Age + Sex + ChestPainType,
+                       family = binomial, data = train_test)
> add1(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType + RestingBP
+      + Cholesterol + FastingBS + RestingECG + MaxHR + ExerciseAngina
+      + Oldpeak + ST_Slope, test = "F")
Single term additions

Model:
HeartDisease ~ 1 + Age + Sex + ChestPainType
               Df Deviance    AIC  F value    Pr(>F)
<none>              698.01 710.01
RestingBP       1   695.49 709.49   2.6862 0.1016441
Cholesterol     1   686.52 700.52  12.4291 0.0004486 ***
FastingBS       1   680.98 694.98  18.5822 1.847e-05 ***
RestingECG      2   695.21 711.21   1.4903 0.2259696
MaxHR           1   678.76 692.76  21.0654 5.210e-06 ***
ExerciseAngina  1   644.41 658.41  61.8014 1.336e-14 ***
Oldpeak         1   652.22 666.22  52.1571 1.271e-12 ***
ST_Slope        2   543.55 559.55 105.4233 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Warning message:
In add1.glm(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType +  :
  F test assumes quasibinomial family
> ## Choose FastingBS as the next variable in the model and continue the process
```

```
> training_model <- glm(HeartDisease ~ 1 + Age + Sex + ChestPainType
+                       + FastingBS, family = binomial, data = train_test)
> add1(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType + RestingBP
+      + Cholesterol + FastingBS + RestingECG + MaxHR + ExerciseAngina
+      + Oldpeak + ST_Slope, test = "F")
Single term additions

Model:
HeartDisease ~ 1 + Age + Sex + ChestPainType + FastingBS
               Df Deviance    AIC  F value    Pr(>F)
<none>              680.98 694.98
RestingBP       1   678.81 694.81   2.3701 0.1241024
Cholesterol     1   670.14 686.14  11.9937 0.0005644 ***
RestingECG      2   679.19 697.19   0.9743 0.3779271
MaxHR           1   661.88 677.88  21.4036 4.390e-06 ***
ExerciseAngina  1   625.72 641.72  65.5213 2.351e-15 ***
Oldpeak         1   633.98 649.98  55.0020 3.296e-13 ***
ST_Slope        2   527.24 545.24 108.0332 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Warning message:
In add1.glm(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType +  :
  F test assumes quasibinomial family
> ## Choose MaxHR as the next variable in the model and continue the process
```

```
> training_model <- glm(HeartDisease ~ 1 + Age + Sex + ChestPainType
+                       + FastingBS + MaxHR, family = binomial, data = train_test)
> add1(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType + RestingBP
+      + Cholesterol + FastingBS + RestingECG + MaxHR + ExerciseAngina
+      + Oldpeak + ST_Slope, test = "F")
Single term additions

Model:
HeartDisease ~ 1 + Age + Sex + ChestPainType + FastingBS + MaxHR
               Df Deviance    AIC F value    Pr(>F)
<none>              661.88 677.88
RestingBP       1   659.96 677.96  2.1639 0.1417086
Cholesterol     1   649.40 667.40 14.2471 0.0001731 ***
RestingECG      2   660.91 680.91  0.5435 0.5809389
ExerciseAngina  1   619.03 637.03 51.2985 1.917e-12 ***
Oldpeak         1   613.90 631.90 57.9144 8.334e-14 ***
ST_Slope        2   524.70 544.70 96.7330 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Warning message:
In add1.glm(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType +  :
  F test assumes quasibinomial family
> ## Choose ExerciseAngina as our next variable in the model and continue the process
```

```
> training_model <- glm(HeartDisease ~ 1 + Age + Sex + ChestPainType
+                       + FastingBS + MaxHR + ExerciseAngina, family = binomial, data = train_test)
> add1(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType + RestingBP
+      + Cholesterol + FastingBS + RestingECG + MaxHR + ExerciseAngina
+      + Oldpeak + ST_Slope, test = "F")
Single term additions

Model:
HeartDisease ~ 1 + Age + Sex + ChestPainType + FastingBS + MaxHR +
    ExerciseAngina
            Df Deviance    AIC F value    Pr(>F)
<none>           619.03 637.03
RestingBP    1   618.49 638.49  0.6467  0.421567
Cholesterol  1   610.15 630.15 10.7659  0.001082 **
RestingECG   2   618.51 640.51  0.3082  0.734892
Oldpeak      1   592.11 612.11 33.6469 9.791e-09 ***
ST_Slope     2   512.38 534.38 76.9081 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Warning message:
In add1.glm(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType +  :
  F test assumes quasibinomial family
> ## Choose Oldpeak as the next variable in the model and continue the process
```

```
> training_model <- glm(HeartDisease ~ 1 + Age + Sex + ChestPainType
+                       + FastingBS + MaxHR + ExerciseAngina + Oldpeak, family = binomial,
+                       data = train_test)
> add1(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType + RestingBP
+      + Cholesterol + FastingBS + RestingECG + MaxHR + ExerciseAngina
+      + Oldpeak + ST_Slope, test = "F")
Single term additions

Model:
HeartDisease ~ 1 + Age + Sex + ChestPainType + FastingBS + MaxHR +
    ExerciseAngina + Oldpeak
            Df Deviance    AIC F value    Pr(>F)
<none>           592.11 612.11
RestingBP    1   592.05 614.05  0.0740  0.785743
Cholesterol  1   584.65 606.65  9.4306  0.002212 **
RestingECG   2   592.02 616.02  0.0547  0.946741
ST_Slope     2   502.06 526.06 66.1803 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Warning message:
In add1.glm(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType +  :
  F test assumes quasibinomial family
> ## Choose ST_Slope as the next variable in the model and continue the process
```

```
> training_model <- glm(HeartDisease ~ 1 + Age + Sex + ChestPainType
+                       + FastingBS + MaxHR + ExerciseAngina + Oldpeak + ST_Slope, family = binomial,
+                       data = train_test)
> add1(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType + RestingBP
+      + Cholesterol + FastingBS + RestingECG + MaxHR + ExerciseAngina
+      + Oldpeak + ST_Slope, test = "F")
Single term additions

Model:
HeartDisease ~ 1 + Age + Sex + ChestPainType + FastingBS + MaxHR +
    ExerciseAngina + Oldpeak + ST_Slope
            Df Deviance    AIC F value  Pr(>F)
<none>           502.06 526.06
RestingBP    1   502.06 528.06  0.0090 0.92432
Cholesterol  1   499.80 525.80  3.3416 0.06795 .
RestingECG   2   501.91 529.91  0.1129 0.89325
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Warning message:
In add1.glm(training_model, HeartDisease ~ 1 + Age + Sex + ChestPainType +  :
  F test assumes quasibinomial family
> ## All variables have p-value > 0.01, so they are unimportant. We stop the process.
```

Through the forward selection process, we can see that a person's heart disease is mainly based on one's **age**, **sex**, **chest pain type**, whether or not one's **normal fasting blood sugar (FastingBS** is more than 120mg/dl, one's **maximum heart rate (MaxHR)**, whether or not one has an **exercise-induced angina**, the value of the **oldpeak**, and the changing rate of the peak exercise ST segment, while one's *resting blood pressure* (RestingBP), *resting electrocardio diagram* (RestingECG), and *cholesterol* results remain unimportant. Hence, we proceed to remove these three variables from our testing dataset, and look at the first 10 rows of the test dataset to confirm that two variables are removed.

```
> test <- subset(test, select = -c(RestingBP, RestingECG, Cholesterol))
> head(test, n = 10)
   Age Sex ChestPainType FastingBS MaxHR ExerciseAngina Oldpeak ST_Slope HeartDisease
1   40   M           ATA         0   172              N       0       Up             0
3   37   M           ATA         0    98              N       0       Up             0
7   45   F           ATA         0   170              N       0       Up             0
12  58   M           ATA         0    99              Y       2     Flat             1
22  44   M           ATA         0   142              N       1     Flat             0
27  53   M           ASY         0   112              Y       3     Flat             0
28  52   M           ATA         0   118              N       0       Up             0
32  56   M           NAP         0   114              N       0       Up             0
35  43   F           ATA         0   154              N       0       Up             0
43  35   M           ATA         0   168              N       0       Up             0
>
```

To dive deeper down into the analysis, we need to understand whether the model can predict, with high probability, a person's heart attack based on certain sex, age, and the other important variables. Therefore, since it is important to test the validity of the model through the test dataset, we predict the probability that an individual will have a heart attack using the training model on the test dataset.

$$\text{predicted\_values} <- \textbf{predict}(\text{training\_}\textbf{model}, \text{ test }, \text{ "response"})$$

We then determine 1 as the classification of an individual with heart disease if the person's respective probability is more than 75%. From the statistics below, each individual will be categorized as a binary variable as desired.

```
> predicted_values[predicted_values > 0.75] = 1
> predicted_values[predicted_values <= 0.75] = 0
> predicted_values
  1   3   7  12  22  27  28  32  35  43  47  66  70  86  97 101 107 109 126 133 140 144 145 146 147 149 150 154 157 176 182
  0   0   0   1   0   1   0   0   0   0   0   0   0   1   0   1   0   0   0   1   1   0   0   0   0   0   1   0   1   1   0
183 192 198 208 215 216 245 248 249 253 254 269 285 298 300 312 314 320 321 324 325 341 351 354 356 361 363 368 375 380 383
  1   0   0   0   1   0   1   1   1   0   0   1   0   1   1   0   0   1   1   1   1   1   1   1   0   1   0   0   1   0   1
405 408 411 416 419 423 427 432 438 440 443 449 454 460 462 469 470 482 485 487 489 491 492 495 499 505 506 507 521 525 529
  1   1   1   1   1   1   0   0   1   0   1   1   1   0   1   1   1   0   1   0   0   1   1   1   1   1   1   1   1   0   1
530 533 540 542 546 550 552 556 559 561 569 572 574 577 580 583 592 600 620 622 625 626 634 636 664 670 675 679 682 684 692
  1   1   0   0   0   0   0   0   1   0   1   1   1   1   1   1   0   1   0   1   1   1   1   1   0   0   0   0   0   0   1
700 701 713 714 724 730 734 735 736 739 740 741 742 757 762 763 765 770 776 777 781 787 795 798 799 805 807 809 810 814 820
  1   0   0   0   0   0   0   1   0   0   0   0   1   0   0   1   0   0   1   1   0   0   0   0   0   1   1   0   0   0   1
835 841 855 867 875 876 892 902 903 908 912 915 916
  0   0   0   0   0   0   0   1   0   1   1   1   1
```

After that, by determining such classification, we can use the confusion matrix to figure out the precision of our model, and determine the percentage of precision to determine the correctness of the model. Through the below analysis, we can see that there are 68 well-defined 0 values and 72 well-defined 1 values, which correspond to an approximation of 83.33% of precision. This shows that the model can correctly predict 83.33% of the value of the test dataset overall. Moreover, the model can also correctly predict 89.47% of the individuals without a heart attack, and 78.26% of the individuals with a heart attack, indicating a good model for heart disease prediction.

```
> table_classification <- table(predicted_values, test$HeartDisease)
> table_classification

predicted_values  0  1
               0 68 20
               1  8 72
> percentage_of_well_defined_0_value <- (table_classification[1])/
+     (table_classification[1] + table_classification[2])
> percentage_of_well_defined_0_value * 100
[1] 89.47368
> percentage_of_well_defined_1_value <- (table_classification[4])/
+     (table_classification[3] + table_classification[4])
> percentage_of_well_defined_1_value * 100
[1] 78.26087
> percentage_of_precision_overall <- (table_classification[1] + table_classification[4])/
+     (table_classification[1] + table_classification[2] +
+          table_classification[3] + table_classification[4])
> percentage_of_precision_overall * 100
[1] 83.33333
```

To further validate the logistic regression, one of the most important criterion is the R-squared value. To determine the R-squared value for the test dataset, we first create the null logistic model where the null model only contains the intercept. For the saturated model for the dataset, since we dropped the three columns from the test dataset so that the number of variables of the test dataset is equal to the number of variables in the training model, the model is considered as the saturated model, so we do not need to consider the saturated model.

$$\textbf{null\_model} <- \textbf{glm}(\text{HeartDisease} \sim 1, \textbf{family} = \textbf{binomial}, \textbf{data} = \text{test})$$

To use the training model for our test dataset, we first define a logistic regression model using the same number of variables as the training model, and replace the coefficients of the newly created model with the training model. We then compare the coefficients of the test model and the coefficients of the training model for certainty purposes.

```
> test_model <- glm(HeartDisease ~ 1 + Age + Sex + ChestPainType
+                   + FastingBS + MaxHR + ExerciseAngina + Oldpeak + ST_Slope,
+                   family = binomial, data = test)
> test_model$coefficients <- training_model$coefficients
> training_model$coefficients == test_model$coefficients
     (Intercept)             Age            SexM ChestPainTypeATA ChestPainTypeNAP  ChestPainTypeTA        FastingBS
            TRUE            TRUE            TRUE            TRUE            TRUE            TRUE            TRUE
           MaxHR  ExerciseAnginaY         Oldpeak     ST_SlopeFlat       ST_SlopeUp
            TRUE            TRUE            TRUE            TRUE            TRUE
>
```

After that, we perform ANOVA analysis to determine whether we should reject the null model. Through the analysis below, since the p-value is 2.2e-16, which is significantly lower than the most popular significance level (0.05), we therefore reject the null model and accept the test model.

```
> anova(null_model, test_model, test = "Chisq")
Analysis of Deviance Table

Model 1: HeartDisease ~ 1
Model 2: HeartDisease ~ 1 + Age + Sex + ChestPainType + FastingBS + MaxHR +
    ExerciseAngina + Oldpeak + ST_Slope
  Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
1       167    231.371
2       156     96.053 11   135.32 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
>
```

The pseudo R^2 is calculated below, and since the pseudo R^2 is approximately 0.584, the logistic regression is able to explain 58.4% of the variability of the dataset. Based on McFadden's R-squared criteria, since a good model represents the R-squared of somewhere around 40%, our logistic regression model manages to be close to McFadden's own words, ergo proving that logistic regression is able to predict heart disease well.

```
> pseudo_r_squared <- (null_model$deviance - test_model$deviance)/(null_model$deviance)
> pseudo_r_squared * 100
[1] 58.48534
>
```

4 Clustering

Using cluster analysis, we can identify homogeneous subgroups within the dataset. Intuitively, since we are trying to predict the heart disease status of an individual based on some characteristics, we will apply cluster analysis to see if cluster analysis is able to predict heart disease status based on the given characteristics.

First of all, we remove the HeartDisease column and put the modified dataset into a new variable; however, we will use the removed column later to compare the results to our cluster results.

```
> new_heart_df <- heart_df[, -12]
> new_heart_df
   Age Sex ChestPainType RestingBP Cholesterol FastingBS RestingECG MaxHR ExerciseAngina Oldpeak ST_Slope
1   40   M           ATA       140         289         0     Normal   172              N     0.0       Up
2   49   F           NAP       160         180         0     Normal   156              N     1.0     Flat
3   37   M           ATA       130         283         0         ST    98              N     0.0       Up
4   48   F           ASY       138         214         0     Normal   108              Y     1.5     Flat
5   54   M           NAP       150         195         0     Normal   122              N     0.0       Up
6   39   M           NAP       120         339         0     Normal   170              N     0.0       Up
7   45   F           ATA       130         237         0     Normal   170              N     0.0       Up
8   54   M           ATA       110         208         0     Normal   142              N     0.0       Up
9   37   M           ASY       140         207         0     Normal   130              Y     1.5     Flat
10  48   F           ATA       120         284         0     Normal   120              N     0.0       Up
11  37   F           NAP       130         211         0     Normal   142              N     0.0       Up
12  58   M           ATA       136         164         0         ST    99              Y     2.0     Flat
13  39   M           ATA       120         204         0     Normal   145              N     0.0       Up
14  49   M           ASY       140         234         0     Normal   140              Y     1.0     Flat
15  42   F           NAP       115         211         0         ST   137              N     0.0       Up
16  54   F           ATA       120         273         0     Normal   150              N     1.5     Flat
17  38   M           ASY       110         196         0     Normal   166              N     0.0     Flat
18  43   F           ATA       120         201         0     Normal   165              N     0.0       Up
19  60   M           ASY       100         248         0     Normal   125              N     1.0     Flat
20  36   M           ATA       120         267         0     Normal   160              N     3.0     Flat
21  43   F            TA       100         223         0     Normal   142              N     0.0       Up
22  44   M           ATA       120         184         0     Normal   142              N     1.0     Flat
23  49   F           ATA       124         201         0     Normal   164              N     0.0       Up
24  44   M           ATA       150         288         0     Normal   150              Y     3.0     Flat
25  40   M           NAP       130         215         0     Normal   138              N     0.0       Up
```

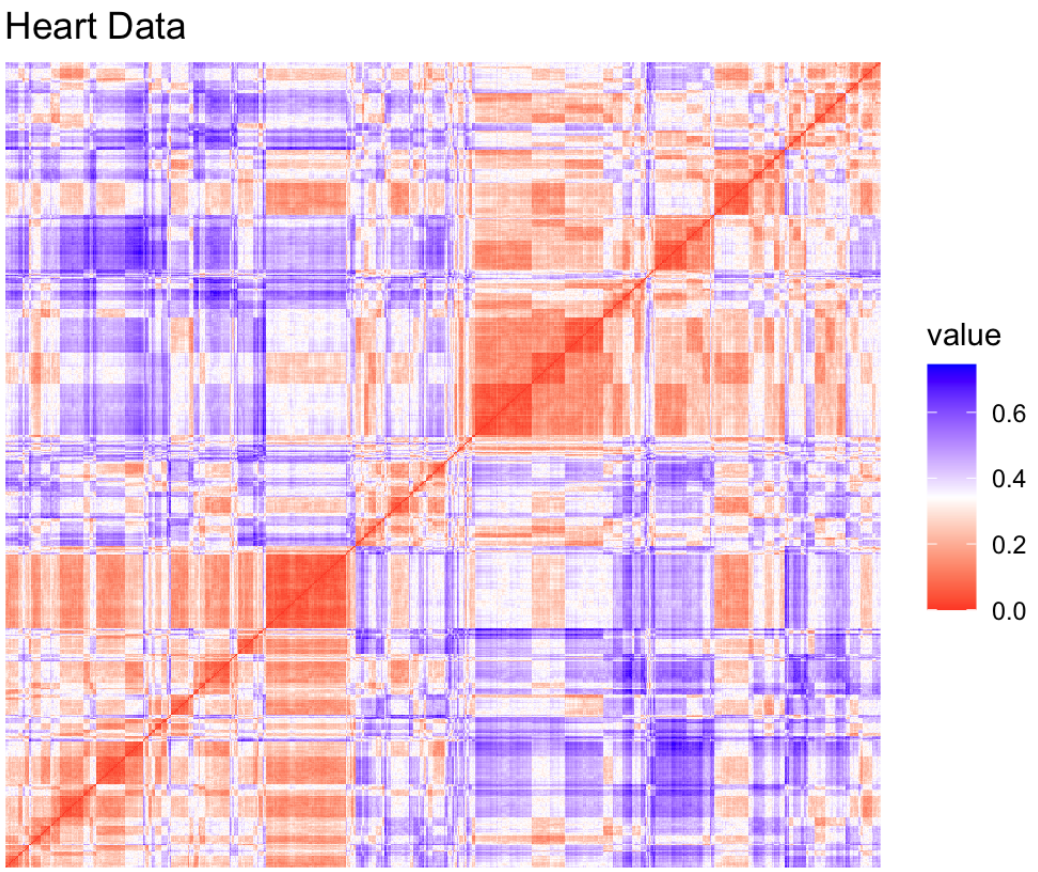
In order to apply clustering techniques, we need to know whether or not the dataset is able to split up into meaningful clusters. In this case, since the dataset contains categorical variables, we will use the VAT algorithm to assess cluster tendency.

To use the VAT algorithm, we need the dissimilarity matrix. We first convert the categorical column vectors to factors, and then use the **daisy()** function from package **cluster** package to calculate the dissimilarity matrix.

```
> for (i in c(2,3,7,9,11)) {
+     new_heart_df[,i] = as.factor(new_heart_df[,i])
+ }
> library(cluster)
> dissimilarity <- daisy(new_heart_df, metric = "gower")
Warning message:
In daisy(new_heart_df, metric = "gower") :
  binary variable(s) 6 treated as interval scaled
```

We then display the ordered dissimilarity matrix (ODI) by using **fviz_dist()** function from package **factoextra**.

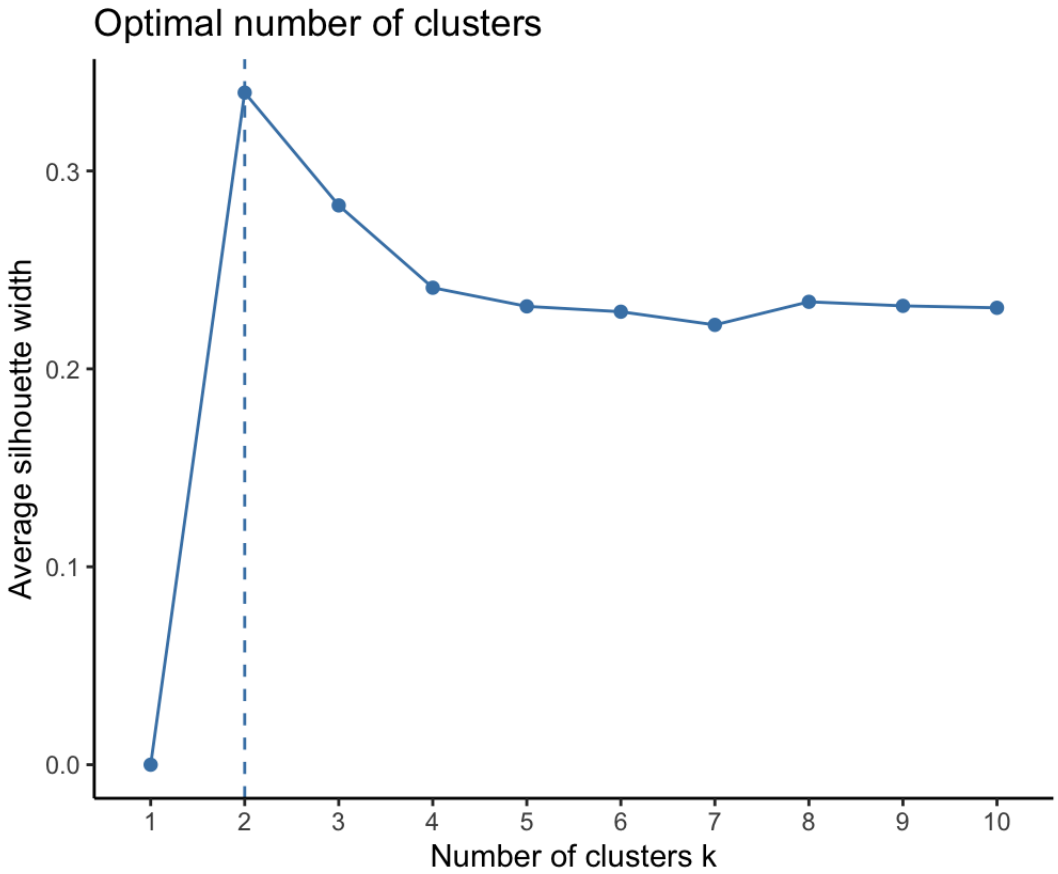
```
dissimilarity <- as.dist(dissimilarity)

library(factoextra)
fviz_dist(dissimilarity, show_labels = FALSE) + labs(title = "Heart Data")
```



From the ODI above, based on visual inspection, we can see that blue squares at the top left and bottom right of the ODI indicates more or less that there is evidence of clusters.

After knowing that clustering is appropriate, we would need to determine the optimal needed number of clusters. We can proceed with using the silhouette method as follows.

```
fviz_nbclust(as.matrix(dissimilarity), pam, method = "silhouette") + theme_classic()
```



From the results of silhouette method, we can see that the optimal number of clusters for the dataset is 2, so we proceed with applying clustering techniques with 2 clusters.

4.1 K-medoids

For a dataset with categorical variables, we will apply k-medoids (or PAM algorithm) to determine the outcome of the clusters, and print out the information of the medoids that the algorithm gives.

```
> pam.res <- pam(new_heart_df, 2, metric = "euclidean", nstart = 25)
> pam.res
Medoids:
      ID Age Sex ChestPainType RestingBP Cholesterol FastingBS RestingECG MaxHR ExerciseAngina Oldpeak ST_Slope
[1,] 641  48   1             3       130         275         0          2   139             1     0.2        3
[2,] 561  54   2             3       133         203         0          3   137             1     0.2        3
Clustering vector:
  [1] 1 2 1 2 2 1 2 2 2 1 2 2 2 2 2 1 2 2 1 1 2 2 2 1 2 2 1 1 2 1 2 2 2 2 1 1 1 2 2 1 2 2 1 1 1 2 2 1 2 2 1 1 1 2 2
 [74] 2 1 2 1 2 2 2 1 2 2 2 2 2 1 2 1 2 2 1 2 1 2 1 2 2 2 1 1 2 1 1 2 2 2 1 1 1 2 1 1 2 2 1 1 1 2 1 2 2 2 2 1 2 1 2
[147] 2 1 2 1 2 2 1 1 1 1 2 2 1 2 1 2 1 2 2 1 2 2 2 2 1 2 1 2 1 1 2 1 2 2 2 2 1 1 1 2 1 2 1 1 1 2 1 2 2 2 1 1 1 1 2
[220] 1 2 1 2 1 1 1 2 2 1 2 1 1 1 1 1 2 1 1 1 2 1 2 1 2 1 1 1 2 2 1 2 1 1 1 1 1 1 1 1 2 2 1 1 2 2 2 2 2 1 1 2 1 2 2
[293] 2 2 1 1 1 1 1 2 1 2 2 2 1 2 1 2 2 2 2 1 2 2 2 1 1 2 1 2 1 1 1 1 1 2 2 2 1 2 2 2 1 1 2 2 1 1 2 1 2 1 2 1 2 1 1
[366] 1 1 1 1 1 1 2 2 2 1 1 1 1 2 1 2 1 2 1 2 2 2 2 2 1 1 2 2 1 2 2 1 2 2 1 2 1 2 1 1 1 2 1 1 2 2 2 2 2 2 1 1 2 2 1
[439] 2 1 1 2 2 2 2 2 2 2 2 2 1 2 1 1 2 2 2 1 1 1 1 1 2 2 2 1 2 1 2 2 2 2 2 2 1 1 2 2 2 1 2 2 1 2 2 1 1 1 2 2 2 2 2
[512] 1 2 1 1 2 2 2 2 2 1 2 1 1 2 2 2 1 1 2 2 2 1 2 1 2 2 2 1 2 2 2 2 2 1 2 1 1 2 2 1 2 1 2 2 2 2 2 1 2 2 1 2 2 1 2
[585] 1 1 1 1 2 2 2 2 1 1 1 2 1 2 2 2 2 1 1 1 1 1 1 1 2 1 1 2 2 1 1 1 1 1 1 2 1 1 1 1 2 2 2 2 1 2 2 2 2 1 1 2 1 1 1
[658] 2 2 2 2 2 2 2 1 2 1 2 1 2 2 2 2 2 2 1 1 2 1 2 1 1 1 1 1 1 1 1 1 2 2 1 2 2 2 2 2 2 2 1 2 2 1 2 1 1 2 1 1 2 2 1
[731] 1 1 1 2 1 2 1 1 1 1 2 1 2 2 2 1 2 2 1 2 1 2 2 1 1 2 2 2 1 2 2 1 1 2 2 2 1 1 2 1 1 1 2 2 1 2 2 1 2 1 2 1 1 2 2
[804] 1 1 1 2 1 1 1 1 1 2 2 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 1 2 1 2 1 1 2 1 2 2 2 1 1 1 2 1 2 1 2 1 1 2 2 2 1 2 2 1
[877] 2 1 1 2 2 1 1 2 1 2 2 2 1 2 2 2 2 1 1 2 2 2 1 2 2 2 2 1 2 1 2 2 2 2 2 1 1 2 2 2 2
Objective function:
   build     swap
51.78916 43.42697

Available components:
 [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"  "clusinfo"   "silinfo"    "diss"       "call"       "data"
```

The given medoids do not give any information about its heart disease status, so we use the medoids ID (641 and 561) to print out the medoids' original data.

```
> heart_df[641,]
    Age Sex ChestPainType RestingBP Cholesterol FastingBS RestingECG MaxHR ExerciseAngina Oldpeak ST_Slope HeartDisease
641  48   F           NAP       130         275         0     Normal   139              N     0.2       Up            0
> heart_df[561,]
    Age Sex ChestPainType RestingBP Cholesterol FastingBS RestingECG MaxHR ExerciseAngina Oldpeak ST_Slope HeartDisease
561  54   M           NAP       133         203         0         ST   137              N     0.2       Up            0
```

From the information above, both of the medoids belong to two individuals who do not have heart disease. This means that based on heart disease alone, k-medoids is not capable of accurately predicting heart disease status of an individual.

However, since a group of datums would share some similar information, we will dive deeper down into looking at the datums that belong to one cluster by listing some of them out. We proceed by binding the cluster information to the original dataset, create a dataset that contains datums in cluster 1 and cluster 2.

```
> dd <- cbind(heart_df, cluster = pam.res$cluster)
> cluster_1 <- dd[dd$cluster == 1,]
> cluster_2 <- dd[dd$cluster == 2,]
```

We then first list out the datums in cluster 1,

```
> cluster_1
    Age Sex ChestPainType RestingBP Cholesterol FastingBS RestingECG MaxHR ExerciseAngina Oldpeak ST_Slope HeartDisease cluster
1    40   M           ATA       140         289         0     Normal   172              N     0.0       Up            0       1
3    37   M           ATA       130         283         0         ST    98              N     0.0       Up            0       1
6    39   M           NAP       120         339         0     Normal   170              N     0.0       Up            0       1
10   48   F           ATA       120         284         0     Normal   120              N     0.0       Up            0       1
16   54   F           ATA       120         273         0     Normal   150              N     1.5     Flat            0       1
19   60   M           ASY       100         248         0     Normal   125              N     1.0     Flat            1       1
20   36   M           ATA       120         267         0     Normal   160              N     3.0     Flat            1       1
24   44   M           ATA       150         288         0     Normal   150              Y     3.0     Flat            1       1
27   53   M           ASY       124         260         0         ST   112              Y     3.0     Flat            0       1
28   52   M           ATA       120         284         0     Normal   118              N     0.0       Up            0       1
29   53   F           ATA       113         468         0     Normal   127              N     0.0       Up            0       1
31   53   M           NAP       145         518         0     Normal   130              N     0.0     Flat            1       1
36   32   M           ATA       125         254         0     Normal   155              N     0.0       Up            0       1
37   65   M           ASY       140         306         1     Normal    87              Y     1.5     Flat            1       1
38   41   F           ATA       110         250         0         ST   142              N     0.0       Up            0       1
42   54   F           NAP       130         294         0         ST   100              Y     0.0     Flat            1       1
43   35   M           ATA       150         264         0     Normal   168              N     0.0       Up            0       1
44   52   M           NAP       140         259         0         ST   170              N     0.0       Up            0       1
46   59   M           NAP       130         318         0     Normal   120              Y     1.0     Flat            0       1
49   36   M           NAP       112         340         0     Normal   184              N     1.0     Flat            0       1
50   41   M           ASY       110         289         0     Normal   170              N     0.0     Flat            1       1
54   41   F           ATA       130         245         0     Normal   150              N     0.0       Up            0       1
57   31   M           ASY       120         270         0     Normal   153              Y     1.5     Flat            1       1
59   54   M           ASY       150         365         0         ST   134              N     1.0       Up            0       1
60   52   M           ASY       112         342         0         ST    96              Y     1.0     Flat            1       1
61   49   M           ATA       100         253         0     Normal   174              N     0.0       Up            0       1
62   43   F           NAP       150         254         0     Normal   175              N     0.0       Up            0       1
```

, and cluster 2:

```
> cluster_2
    Age Sex ChestPainType RestingBP Cholesterol FastingBS RestingECG MaxHR ExerciseAngina Oldpeak ST_Slope HeartDisease cluster
2    49   F          NAP       160         180         0     Normal   156              N     1.0     Flat            1       2
4    48   F          ASY       138         214         0     Normal   108              Y     1.5     Flat            1       2
5    54   M          NAP       150         195         0     Normal   122              N     0.0       Up            0       2
7    45   F          ATA       130         237         0     Normal   170              N     0.0       Up            0       2
8    54   M          ATA       110         208         0     Normal   142              N     0.0       Up            0       2
9    37   M          ASY       140         207         0     Normal   130              Y     1.5     Flat            1       2
11   37   F          NAP       130         211         0     Normal   142              N     0.0       Up            0       2
12   58   M          ATA       136         164         0         ST    99              Y     2.0     Flat            1       2
13   39   M          ATA       120         204         0     Normal   145              N     0.0       Up            0       2
14   49   M          ASY       140         234         0     Normal   140              Y     1.0     Flat            1       2
15   42   F          NAP       115         211         0         ST   137              N     0.0       Up            0       2
17   38   M          ASY       110         196         0     Normal   166              N     0.0     Flat            1       2
18   43   F          ATA       120         201         0     Normal   165              N     0.0       Up            0       2
21   43   F           TA       100         223         0     Normal   142              N     0.0       Up            0       2
22   44   M          ATA       120         184         0     Normal   142              N     1.0     Flat            0       2
23   49   F          ATA       124         201         0     Normal   164              N     0.0       Up            0       2
25   40   M          NAP       130         215         0     Normal   138              N     0.0       Up            0       2
26   36   M          NAP       130         209         0     Normal   178              N     0.0       Up            0       2
30   51   M          ATA       125         188         0     Normal   145              N     0.0       Up            0       2
32   56   M          NAP       130         167         0     Normal   114              N     0.0       Up            0       2
33   54   M          ASY       125         224         0     Normal   122              N     2.0     Flat            1       2
34   41   M          ASY       130         172         0         ST   130              N     2.0     Flat            1       2
35   43   F          ATA       150         186         0     Normal   154              N     0.0       Up            0       2
39   48   F          ATA       120         177         1         ST   148              N     0.0       Up            0       2
40   48   F          ASY       150         227         0     Normal   130              Y     1.0     Flat            0       2
41   54   F          ATA       150         230         0     Normal   130              N     0.0       Up            0       2
45   43   M          ASY       120         175         0     Normal   120              Y     1.0     Flat            1       2
```

We can see, based on the given information, that in cluster 1, 54 first rows of cluster 1 have cholesterol values ranging between 250 and 350, while cluster 2 has cholesterol values ranging between 150 and 240. Therefore, we can infer that individuals in cluster 1 have high cholesterol values, while individuals in cluster 2 do not. We then calculate the mean of the cholesterol values in two clusters:

```
> mean(cluster_1$Cholesterol)
[1] 289.0046
> mean(cluster_2$Cholesterol)
[1] 201.9071
```

As we can see, the mean of cholesterol values in cluster 1 is 289.0046 while in cluster 2, the mean is only 201.9071. This infers that high cholesterol value might indicate higher probability of getting a heart disease, and low cholesterol value might indicate that a person will less likely have a heart disease. We then calculate the number of people who have a heart disease in cluster 1, and the number of people who do not have a heart disease in cluster 2 to see how good our inferences are:

```
> sum(cluster_1$HeartDisease == 1)/nrow(cluster_1) * 100
[1] 59.375
> sum(cluster_2$HeartDisease == 0)/nrow(cluster_2) * 100
[1] 48.51064
```

From the information above, 59.375% of individuals in cluster 1 have a heart disease, while 48.51064% of individuals in cluster 2 do not. This means that:

- k-medoids is able to help us predict that people with heart disease have a high cholesterol value, with a prediction rate of 59.375%.

- k-medoids is also able to help us predict that people with low heart disease have a low cholesterol value, with a prediction rate of 48.51064%. While the given prediction rate is more or less not really persuading (as it is lower than 50%), its closure to 50% persuades us to conduct more analysis to prove our point.

To make sure that the individuals who get a heart disease actually have high cholesterol values, and the individuals who do not have low cholesterol values, we perform a summary of their characteristics:

```
> high_cholesterol <- dd[dd$cluster == 1 & dd$HeartDisease == 1,]
> summary(high_cholesterol$Cholesterol)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  239.0   260.0   279.1   290.7   305.8   603.0
```

```
> low_cholesterol <- dd[dd$cluster == 2 & dd$HeartDisease == 0,]
> summary(low_cholesterol$Cholesterol)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   85.0   185.8   207.0   201.8   221.2   241.0
```

From the summary above, a high median and high mean for high cholesterol individuals (279.1 and 290.7 respectively), and a low median and low mean for low cholesterol individuals (207.0 and 201.8 respectively) makes good quantitative inferences for our claim.

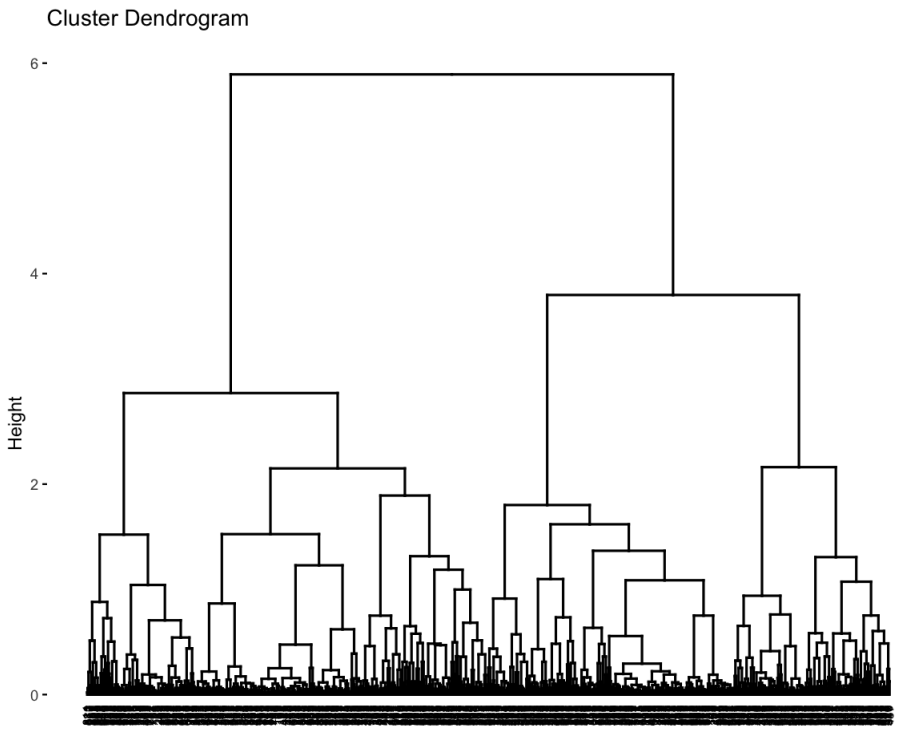
4.2 Hierarchical clustering

While we have made an inference about the relationship between heart disease status and cholesterol value, k-medoids do not correctly assign each individual to his or her correct heart disease status in the sense that the medoids belong to two individuals who both do not have heart disease. Therefore, we would perform hierarchical clustering to see if the method is able to further prove our inferred relationship as well as correctly predict heart disease status of an individual.

We use hclust() to cluster the data hierarchically using Ward's method 2 and cut the tree into 2 groups.

```
> res.hc <- hclust(d= c, method = "ward.D2")
> grp <- cutree(res.hc, k=2)
> y <- grp-1
> sum(y == heart[,"HeartDisease"])/918
[1] 0.7908497
```

This clustering algorithm correctly groups the data into a group of heart disease patients and a group of healthy patients up to 79.084%, a lot more than k-medoids algorithm. Here we have the dendrogram showing the hierarchical clustering result.



To ensure the accuracy of both models, we compare their predictions for sick and healthy patients by assessing how closely the statistics of the variables match the true values.

```
> ## k-medoids
> mean(heart[z==1,"Cholesterol"])    # no-disease
[1] 207.555
> mean(heart[z==0,"Cholesterol"])    # disease
[1] 299.4505
>
> ## hierarchical
> mean(heart[y==0,"Cholesterol"])     # no-disease
[1] 241.0448
> mean(heart[y==1,"Cholesterol"])     # disease
[1] 248.3265
>
> ## true data
> mean(heart[heart["HeartDisease"]== 0,"Cholesterol"])   # no-disease
[1] 239.4741
> mean(heart[heart["HeartDisease"]== 1,"Cholesterol"])   # disease
[1] 248.9061
```

The K-medoids method indicates that individuals with high cholesterol are grouped into the disease group and vice versa (Part 4.1). However, the true data shows that the mean cholesterol levels are similar between the two groups, and hierarchical clustering yields similar results. Specifically, the mean cholesterol level is around 240 for the no-disease group and 248 for those with heart disease.

When applying the same strategy to the ST-slope variable, the hierarchical method explains much more of the variability in this variable compared to the k-medoids method.

```
> ########### no-disease
> ## true data
> sum(no_disease["ST_Slope"] == "Up")/length(no_disease[,"HeartDisease"])
[1] 0.7731707
> ## k-medoids
> sum(heart[z==1,"ST_Slope"] == "Up")/length(heart[z==1,"ST_Slope"])
[1] 0.4533821
> ## hierarchical
> sum(heart[y==0,"ST_Slope"] == "Up")/length(heart[y==0,"ST_Slope"])
[1] 0.6724891
>
> ########### disease
> ## true data
> sum(disease["ST_Slope"] == "Flat")/length(disease[,"HeartDisease"])
[1] 0.75
> ## k-medoids
> sum(heart[z==0,"ST_Slope"] == "Flat")/length(heart[z==0,"ST_Slope"])
[1] 0.5363881
> ## hierarchical
> sum(heart[y==1,"ST_Slope"] == "Flat")/length(heart[y==1,"ST_Slope"])
[1] 0.7130435
```

Here we have y and z are the groups after clustering and cutting tree. In the non-disease group, approximately 77% of the patients have the ST slope type "Up". The K-medoids method predicts that there is only about 45% non disease patients have this type of ST slope, whereas the hierarchical method predicts about 67%, which is a much better performance. So, hierarchical clustering correctly predicts 67%/77% = 87% of the proportion of people with no heart disease and have ST-slope type "Up". Similarly, in the no-disease group, the majority of patients have the ST slope type "Flat", for about 75%, and the hierarchical clustering method again provides a more accurate prediction compared to the K-medoids method, about 71%/77% = 92%.

5 Conclusion

Through careful analysis, most of the predictors in our dataset could be the factors causing heart disease. First of all, by using forward selection to select the best appropriate variables with respect to proportion of variability explained and simplicity purposes, using Age, Sex, ChestPainType, FastingBP, MaxHR, ExerciseAngina, Oldpeak, and ST-slope, our logistic regression model effectively predicts the testing data up to 83% accuracy while being able to explain 54% of the dataset's variability, also an indication of a good model based on McFadden's criteria and the principle of parsimony.

Secondly, with respect to clustering analysis, we found that the hierarchical clustering method successfully grouped most of the data into two groups: healthy individuals and those with at least one heart disease issue. Although its performance is somewhat less effective than logistic regression, we confirmed the minimal necessity of including the cholesterol variable in predicting heart disease status. This is surprising, given that cholesterol has consistently been considered a significant factor affecting cardiac problems. Moreover, both logistic regression and the hierarchical clustering method highlight the importance of the ST-slope variable, which is a crucial factor directly affecting a person's heart disease status. In fact, "the S–T segment/heart rate (ST/HR) slope has been proposed as a more accurate electrocardiographic criterion for the diagnosis of coronary artery disease" [3]

The selected input variables align with the leading risk factors for heart disease and stroke that the Nation's Risk Factors and CDC's Response have inferred. High blood pressure, high low-density lipoprotein (LDL) cholesterol, diabetes, smoking and secondhand smoke exposure, obesity, unhealthy diet, and physical inactivity are the important modifiable risk factors that contribute directly to a person's heart disease. By using Logistic Regression Model and Hierarchical Clustering, we hope to use it to predict future observations, be able to improve our data further to produce the best accuracy, and to help others alter their daily suitable routines to minimize the risk of heart disease and failure.

References

[1] Center for Disease control and Prevention. Heart Disease and Stroke

[2] Wikipedia ST segment

[3] ST-segment

[4] Center for Disease control and Prevention. Fasting blood sugar test.