DATA ANALYSIS REPORT

UNIVERSITÀ degli STUDI di CATANIA | DIPARTIMENTO di ECONOMIA e IMPRESA

Heart Failure Prediction

12 clinical features to predicting death events due to heart attack

MUHAMMAD HASSAN ALI
ID: 10000219630

INTRODUCTION:

“Data Is Everywhere Learn How To Used It”

As Every day, data is growing enormously through various platforms around the world. In which massive amount of data is generated in different fields i.e; finance, banking sectors, online shopping, education sectors, social media and scientific research etc. This data contains so much information inside it and it has become necessary to understand it in order to gain and extract some meaningful information from it, which may be helpful for us to predict future happens. Moreover in this way we can also take better decision in regarding future by avoiding repeated mistakes.

The statistical learning is known as the set of tools that can help us to understand the data with its insights. The Dataset I have used in my data analysis report is provided by Davide Chicco and Giuseppe Jurman. I have found it on kaggle and downloaded it from https://www.kaggle.com/andrewmvd/heart-failure-clinical-data.

Cardiovascular diseases (CVDs) are the number 1 cause of death globally, taking an estimated 17.9 million lives each year, which accounts for 31% of all deaths world-wide. Heart failure is a common event caused by CVDs and this dataset contains 12 features that can be used to predict mortality by heart failure.

Most cardiovascular diseases can be prevented by addressing behavioral risk factors such as tobacco use, unhealthy diet and obesity, physical inactivity and harmful use of alcohol using population-wide strategies.

People with cardiovascular disease or who are at high cardiovascular risk (due to the presence of one or more risk factors such as hypertension, diabetes, hyperlipidaemia or already established disease) need early detection and management wherein a statistical learning model report can be of great help.

There are some factors that affects Death Event. This dataset contains person's information like age, sex, blood pressure, smoke, diabetes, ejection fraction, creatinine phosphokinase, serum creatinine, serum sodium, time and our goal is to predict the probability of death from heart attack as early warning for patients in hospitals with CVD.

DATA SET:

By using following chunk of code we import our data and assigned it into the variable "heart.attk"

```
getwd()
```

```
## [1] "D:/rprojects/StatisticalAnalysisReport"
```

```
heart.attk<-read.csv("heart_failure_clinical_records_dataset.csv")
```

```
head(heart.attk)
```

```
##   age anaemia creatinine_phosphokinase diabetes ejection_fraction
## 1  75       0                     582        0                20
## 2  55       0                    7861        0                38
## 3  65       0                     146        0                20
## 4  50       1                     111        0                20
## 5  65       1                     160        1                20
## 6  90       1                      47        0                40
##   high_blood_pressure platelets serum_creatinine serum_sodium sex smoking
## 1                   1    265000             1.9          130   1       0
## 2                   0    263358             1.1          136   1       0
## 3                   0    162000             1.3          129   1       1
## 4                   0    210000             1.9          137   1       0
## 5                   0    327000             2.7          116   0       0
## 6                   1    204000             2.1          132   1       1
##   DEATH_EVENT
## 1           1
## 2           1
## 3           1
## 4           1
## 5           1
## 6           1
```

We can examine the structure of our dataset are as follows;

```
str(heart.attk)
```

```
## 'data.frame':    299 obs. of  12 variables:
##  $ age                     : num  75 55 65 50 65 90 75 60 65 80 ...
##  $ anaemia                 : int  0 0 0 1 1 1 1 1 0 1 ...
##  $ creatinine_phosphokinase: int  582 7861 146 111 160 47 246 315 157 123 ...
##  $ diabetes                : int  0 0 0 0 1 0 0 1 0 0 ...
##  $ ejection_fraction       : int  20 38 20 20 20 40 15 60 65 35 ...
##  $ high_blood_pressure     : int  1 0 0 0 0 1 0 0 0 1 ...
##  $ platelets               : num  265000 263358 162000 210000 327000 ...
##  $ serum_creatinine        : num  1.9 1.1 1.3 1.9 2.7 2.1 1.2 1.1 1.5 9.4 ...
##  $ serum_sodium            : int  130 136 129 137 116 132 137 131 138 133 ...
##  $ sex                     : int  1 1 1 1 0 1 1 1 0 1 ...
##  $ smoking                 : int  0 0 1 0 0 1 0 1 0 1 ...
##  $ DEATH_EVENT             : int  1 1 1 1 1 1 1 1 1 1 ...
```

By reviewing the above result we can observed that the dataset has 12 variables and 299 observations. Variables information are as follows;

sex - Gender of patients (Male = 1, Female =0)

age - Age of patients(years)

diabetes - If the patient has diabetes(0 = No, 1 = Yes)

anaemia - Decrease of red blood cells or hemoglobin(0 = No, 1 = Yes)

Smoking - If the patient smokes(0 = No, 1 = Yes)

DEATH_EVENT - 0 = No, 1 = Yes

creatinine_phosphokinase - Level of the CPK enzyme in the blood(mcg/L)

ejection_fraction - Percentage of blood leaving the heart at each contraction

high_blood_pressure - If the patient has hypertension(0 = No, 1 = Yes)

Platelets - Platelets in the blood(kiloplatelets/mL)

serum_creatinine - Level of serum creatinine in the blood(mg/dL)

serum_sodium - Level of serum sodium in the blood(mEq/L)

After plotted the structure of our dataset now we can observed that the dataset is not seems well structured, therefore we need to convert the following variables into factor as they are categorical; “sex”,“anaemia”,“diabetes”,“high_blood_pressure,”smoking" and “DEATH_EVENT”.

```
heart.attk$sex <- factor(heart.attk$sex)
heart.attk$anaemia <- factor(heart.attk$anaemia)
heart.attk$diabetes <- factor(heart.attk$diabetes)
heart.attk$high_blood_pressure <- factor(heart.attk$high_blood_pressure)
heart.attk$smoking <- factor(heart.attk$smoking)
heart.attk$DEATH_EVENT <- factor(heart.attk$DEATH_EVENT)

str(heart.attk)
```

```
## 'data.frame':    299 obs. of  12 variables:
##  $ age                     : num  75 55 65 50 65 90 75 60 65 80 ...
##  $ anaemia                 : Factor w/ 2 levels "0","1": 1 1 1 2 2 2 2 2 1 2 ...
##  $ creatinine_phosphokinase: int  582 7861 146 111 160 47 246 315 157 123 ...
##  $ diabetes                : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 2 1 1 ...
##  $ ejection_fraction       : int  20 38 20 20 20 40 15 60 65 35 ...
##  $ high_blood_pressure     : Factor w/ 2 levels "0","1": 2 1 1 1 1 2 1 1 1 2 ...
##  $ platelets               : num  265000 263358 162000 210000 327000 ...
##  $ serum_creatinine        : num  1.9 1.1 1.3 1.9 2.7 2.1 1.2 1.1 1.5 9.4 ...
##  $ serum_sodium            : int  130 136 129 137 116 132 137 131 138 133 ...
##  $ sex                     : Factor w/ 2 levels "0","1": 2 2 2 2 1 2 2 2 1 2 ...
##  $ smoking                 : Factor w/ 2 levels "0","1": 1 1 2 1 1 2 1 2 1 2 ...
##  $ DEATH_EVENT             : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
```

We can check whether there are NA values present or not in our dataset as follows:

```
any(is.na(heart.attk))
```

```
## [1] FALSE
```

As there are no NA values found in our dataset it means our data does not contain any NA value, therefore we can now perform statistical computations and analysis over it.

1.Univariate Analysis:

AGE:

hence the variable age is the continuous variables in range.

```
length(heart.attk$age)
```

```
## [1] 299
```

By using summary function we can easily find its min,max values.

```
summary(heart.attk$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   40.00   51.00   60.00   60.83   70.00   95.00
```

The length of the sample variable is 299 and the range is 40 to 95.

```
sd(heart.attk$age)
```

```
## [1] 11.89481
```

```
var(heart.attk$age)
```

```
## [1] 141.4865
```

```
# Create the function to find mode.
getmode <- function(v) {
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}

# Create the vector with numbers.
v <- c(heart.attk$age)

# Calculate the mode using the user function.
result <- getmode(v)
print(result)
```

```
## [1] 60
```

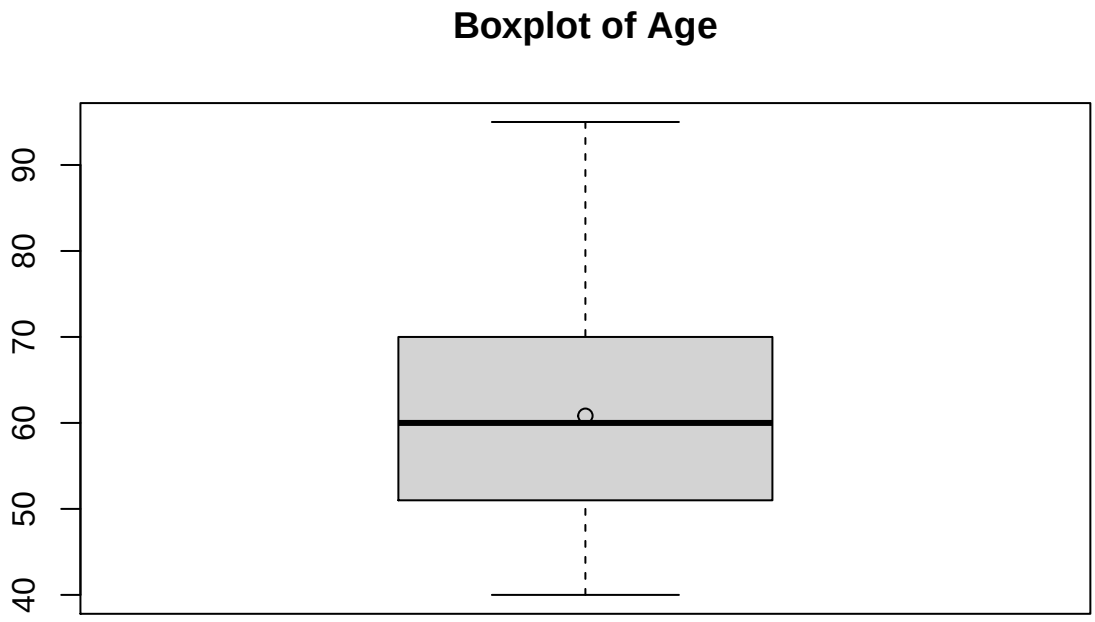
From above statistical computation we can observed that the mean and median are not identical, therefore the distribution is asymmetrical and skewed. And as mean is greater than median, hence the distribution should be skewed to the right and positively skewed.

```
library(e1071)
skewness(heart.attk$age)
```

```
## [1] 0.4188266
```

As skewness is positive, hence our prediction was correct.

```r
boxplot(heart.attk$age, main = "Boxplot of Age")
points(mean(heart.attk$age))
```



Boxplot gives us graphical information about our data and tells us how our data is well distributed in a dataset, so by mean of using this function we can describe the distribution of data in a better way, it contains 1st and 3rd quartile at the upper and lower ends of rectangular box. The median divides the box into two parts. If we plot a box plot for our data the following plot appears. While reviewing the box plot We can observed that there is no outliers in this variable as we didn't see any dot lie outside the whiskers of the boxplot and it was also more clarified that mean is above the median, outliers are the data points located outside the whiskers of the boxplot. Now we can plot the histogram and look for the model that fits better the distribution:

```r
hist(heart.attk$age, freq=FALSE, xlab= "age", main = "Histogram of Age")
lines(density(heart.attk$age), col="red")
```

As our dataset is related to the heart patients so from above plotted figure we can observed that most of the patients lie in between age 55-65.

Data fitting for age:

As per variable domain, the distributions supported on the set of positive infinite real numbers are fitted on the data. Log-likelihood value, BIC and AIC are estimated in order to evaluate the fit of the distribution.

```
round(2*nrow(heart.attk)**(1/3))
```

```
## [1] 13
```

```
library(gamlss.mx)
```

```
## Warning: package 'gamlss.mx' was built under R version 4.0.5
```

```
## Loading required package: gamlss.dist
```

```
## Warning: package 'gamlss.dist' was built under R version 4.0.5
```

```
## Loading required package: MASS
```

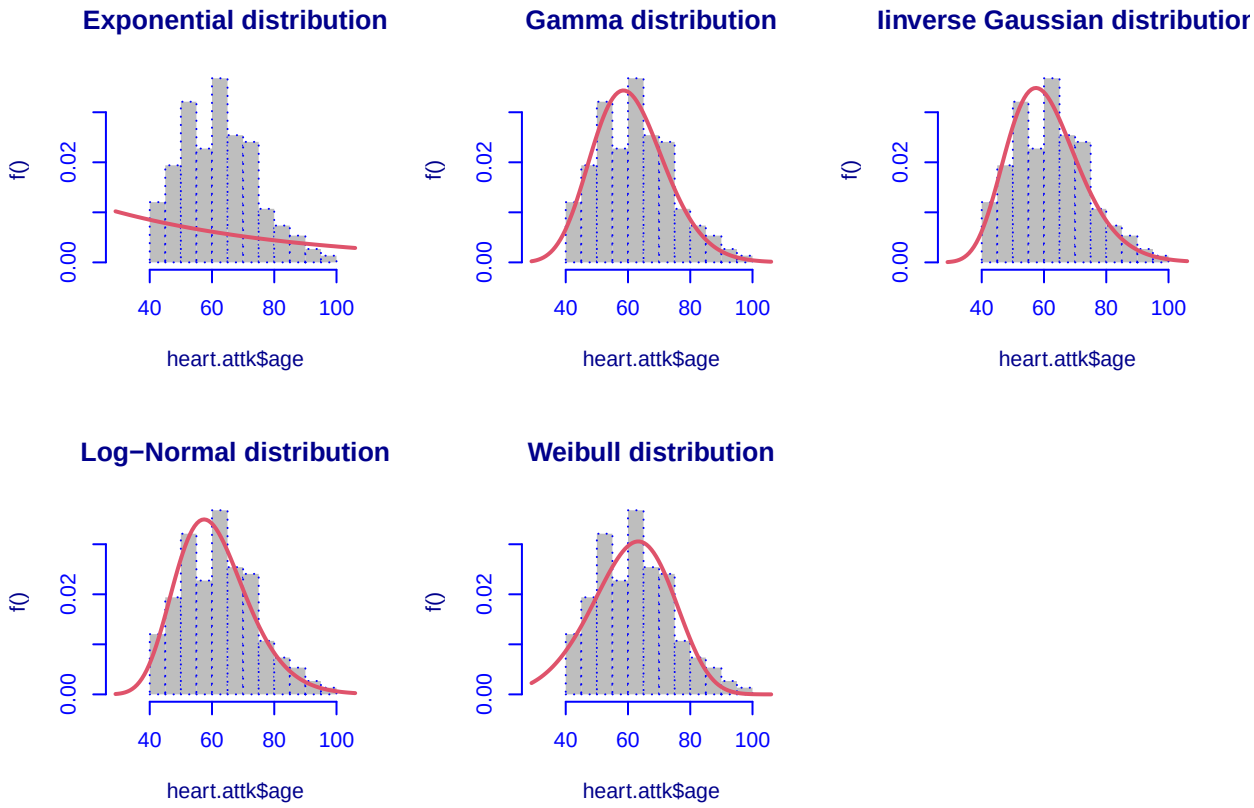
```
## Loading required package: gamlss
```

```
## Warning: package 'gamlss' was built under R version 4.0.5

## Loading required package: splines

## Loading required package: gamlss.data

## Warning: package 'gamlss.data' was built under R version 4.0.4

##
## Attaching package: 'gamlss.data'

## The following object is masked from 'package:datasets':
##
##     sleep

## Loading required package: nlme

## Loading required package: parallel

##  **********   GAMLSS Version 5.3-4  **********

## For more on GAMLSS look at https://www.gamlss.com/

## Type gamlssNews() to see new features/changes/bug fixes.

## Loading required package: nnet
```

```
par(mfrow=c(2,3))
age.EXP  <-histDist(heart.attk$age, family=EXP, nbins = 13, main="Exponential distribution")
age.GA   <-histDist(heart.attk$age, family=GA, nbins = 13, main="Gamma distribution")
age.IG   <-histDist(heart.attk$age, family=IG, nbins = 13, main="Iinverse Gaussian distribution")
```

```
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
```

```
age.LOGNO<-histDist(heart.attk$age, family=LOGNO, nbins = 13, main="Log-Normal distribution")
age.WEI  <-histDist(heart.attk$age, family=WEI, nbins = 13, main="Weibull distribution")
```




```
age.matrix<-matrix(c(age.LOGNO$df.fit, logLik(age.LOGNO), AIC(age.LOGNO),
age.LOGNO$sbc, age.GA$df.fit, logLik(age.GA), AIC(age.GA),
age.GA$sbc, age.WEI$df.fit, logLik(age.WEI),
AIC(age.WEI), age.WEI$sbc,
age.EXP$df.fit, logLik(age.EXP), AIC(age.EXP), age.EXP$sbc,
age.IG$df.fit, logLik(age.IG), AIC(age.IG), age.IG$sbc), ncol=4, byrow=TRUE)
colnames(age.matrix)<-c("df","LogLik", "AIC", "BIC")
rownames(age.matrix)<-c("LOGNO", "GA", "WEI", "EXP", "IG")
age.matrix<-as.table(age.matrix)
age.matrix
```

```
##              df    LogLik       AIC       BIC
## LOGNO     2.000 -1157.861  2319.721  2327.122
## GA        2.000 -1158.386  2320.771  2328.172
## WEI       2.000 -1175.128  2354.256  2361.657
## EXP       1.000 -1527.336  3056.672  3060.372
## IG        2.000 -1157.610  2319.219  2326.620
```

After above statistical analysis we can observed that the model which has large value of log likelihood and small value in AIC and BIC is “Inverse Gaussian distribution”, hence on the basis of likelihood method Our data is more likely to fit into an inverse Gaussian distribution.

Likelihood ratio test:

The goodness-of-fit test was performed between the Exponential model (under the null hypothesis) and the inverse Gaussian model (under the alternative hypothesis).

```
library(lmtest)
```

```
## Warning: package 'lmtest' was built under R version 4.0.5
```

```
## Loading required package: zoo
```

```
## Warning: package 'zoo' was built under R version 4.0.5
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
lrtest(age.IG, age.EXP)
```

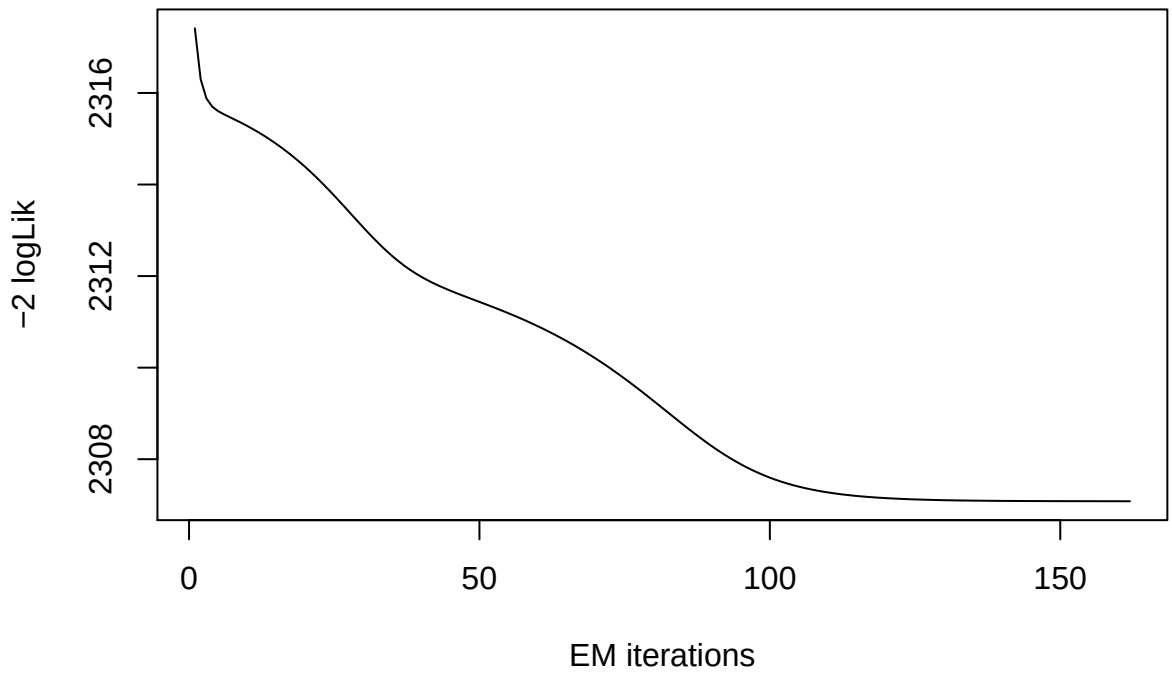
```
## Likelihood ratio test
##
## Model 1: gamlssML(formula = heart.attk$age, family = "IG")
## Model 2: gamlssML(formula = heart.attk$age, family = "EXP")
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   2 -1157.6
## 2   1 -1527.3 -1 739.45  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The null hypothesis would be rejected at nearly every significance level. Thus, we know that we should definitely use the inverse Gaussian model as it increases the accuracy of our model by a substantial amount in terms of AIC and BIC values.

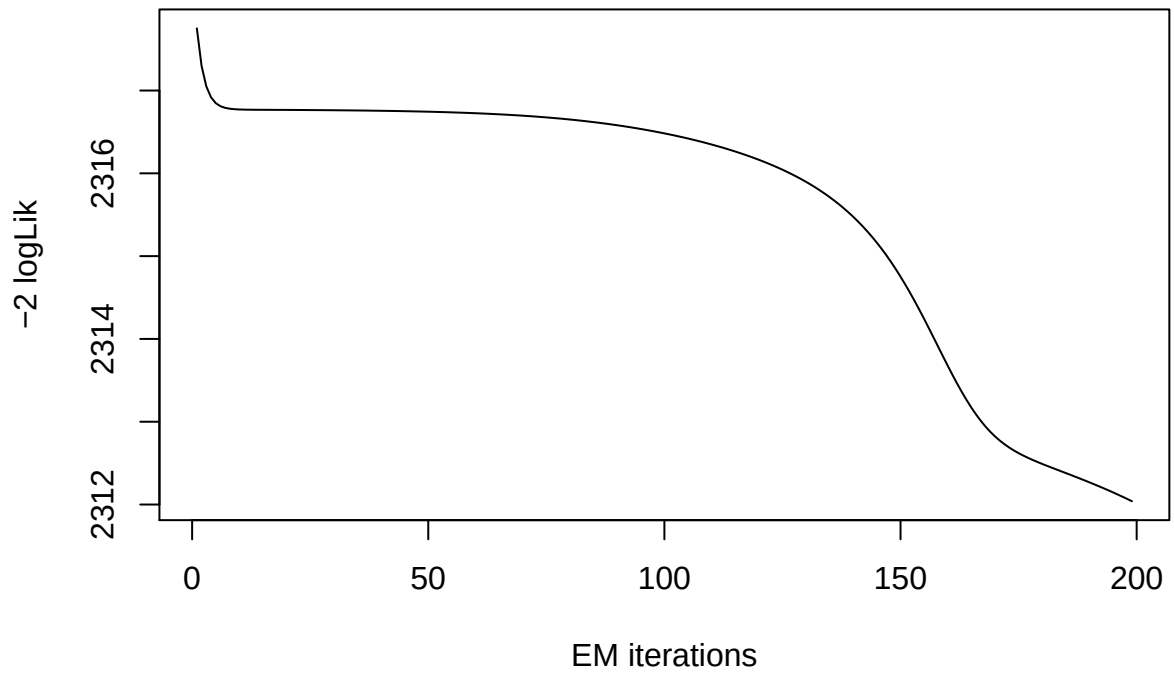
Mixture of distributions:

It is possible to compute a mixture of two gamma distributions In order to find the best mixture,the algorithm is repeated five times.

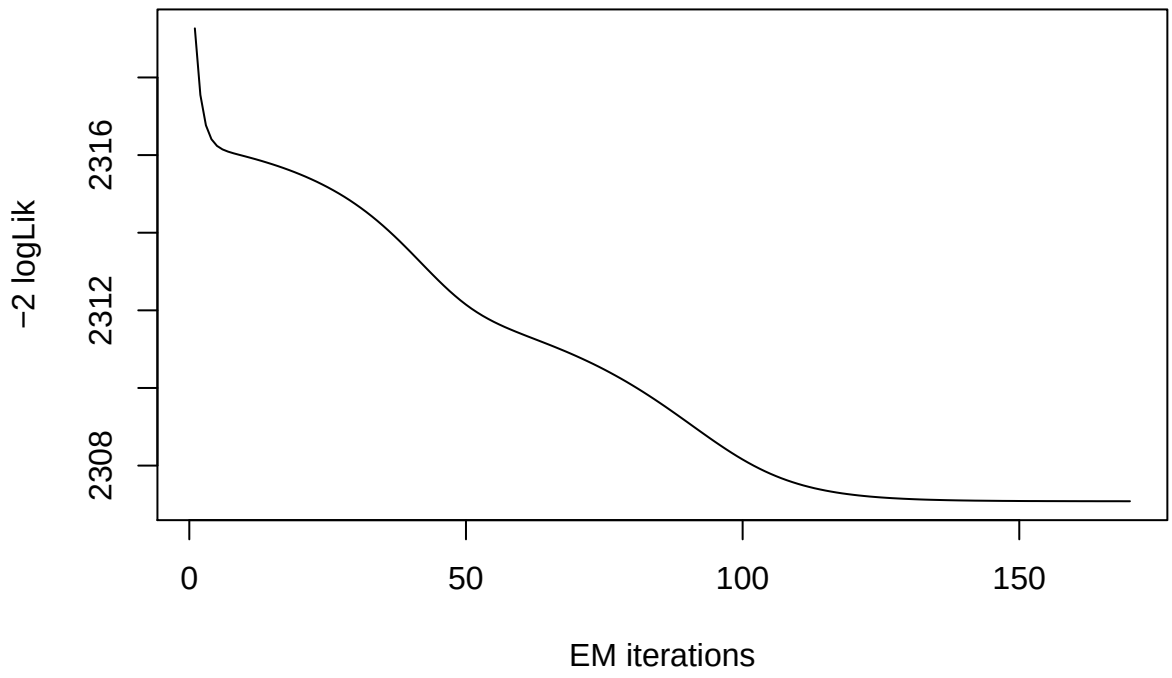
```
mxfit.age <- gamlssMXfits(n = 5, heart.attk$age~1, family = GA, K = 2, data = heart.attk)
```



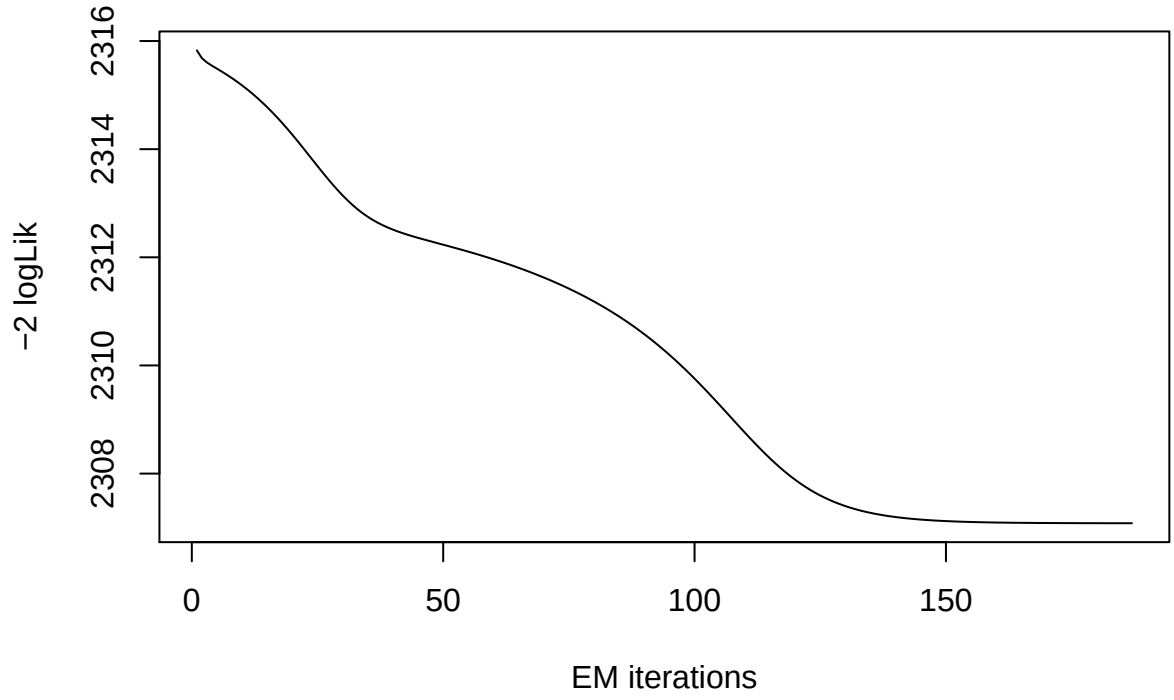

```
## model= 1
```



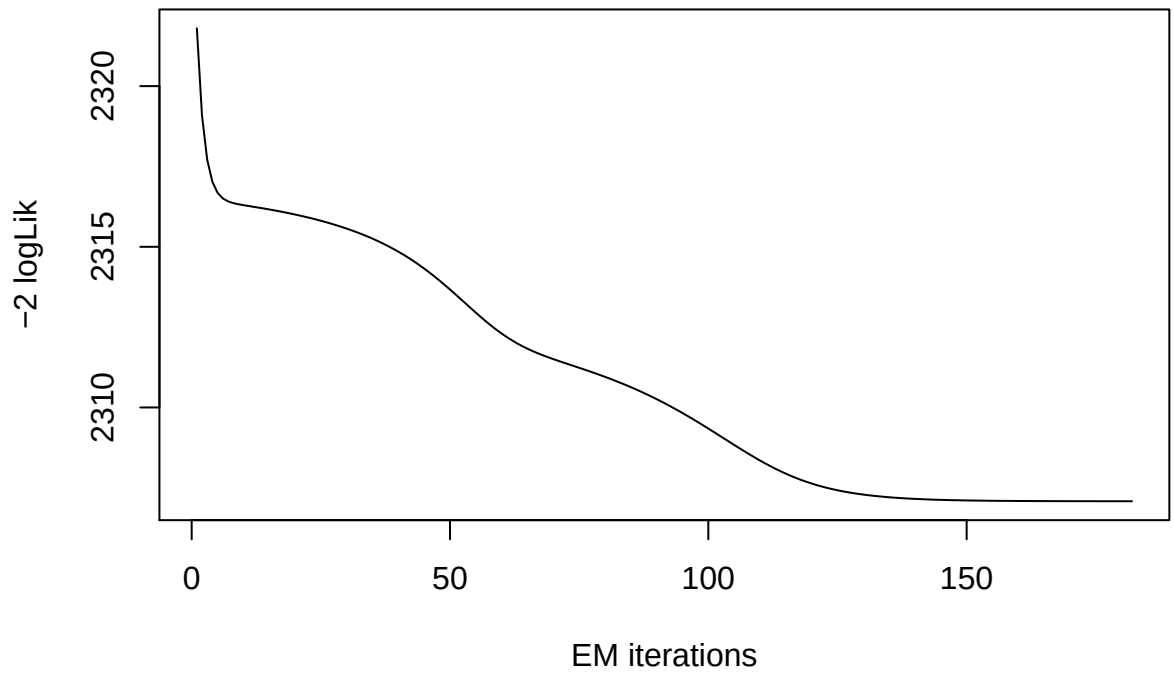

```
## model= 2
```




```
## model= 3
```




```
## model= 4
```




```
## model= 5
```

```
print(mxfit.age)
```

```
##
## Mixing Family:  c("GA", "GA")
##
## Fitting method: EM algorithm
##
## Call:  gamlssMX(formula = heart.attk$age ~ 1, family = GA,
##     K = 2, data = heart.attk)
##
## Mu Coefficients for model: 1
## (Intercept)
##       3.841
## Sigma Coefficients for model: 1
## (Intercept)
##      -2.413
## Mu Coefficients for model: 2
## (Intercept)
##       4.154
## Sigma Coefficients for model: 2
## (Intercept)
##      -1.781
##
```

```
## Estimated probabilities: 0.1678709 0.8321291
##
## Degrees of Freedom for the fit: 5 Residual Deg. of Freedom   294
## Global Deviance:     2307.08
##             AIC:     2317.08
##             SBC:     2335.58
```

Here we can observed that the AIC value has improved with the combination of two Gamma distributions, as it is lower than the one obtained with a inverse guassian distribution: AIC is now equal to 2317.08 , while the previous value was 2319.

```
logLik(mxfit.age)
```

```
## 'log Lik.' -1153.541 (df=5)
```

```
mxfit.age$prob
```

```
## [1] 0.1678709 0.8321291
```

```
fitted(mxfit.age, "mu")[1]
```

```
## [1] 60.83557
```

```
fitted(mxfit.age, "sigma")[2]
```

```
## [1] 60.83557
```

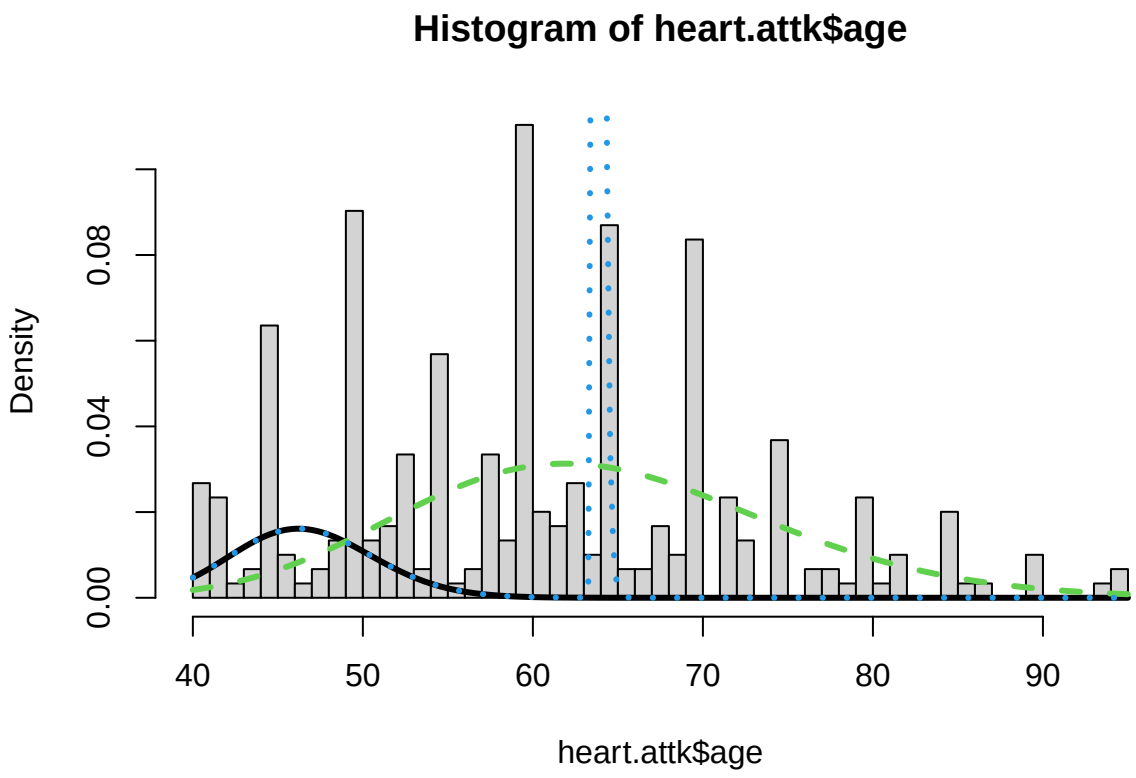
```
hist(heart.attk$age, breaks = 50,freq = FALSE)

mu.hat1.age <- exp(mxfit.age[["models"]][[1]][["mu.coefficients"]])
sigma.hat1.age <- exp(mxfit.age[["models"]][[1]][["sigma.coefficients"]])
mu.hat2.age <- exp(mxfit.age[["models"]][[2]][["mu.coefficients"]])
sigma.hat2.age <- exp(mxfit.age[["models"]][[2]][["sigma.coefficients"]])

hist(heart.attk$age, freq = FALSE, xlab = "age" ,
main="age-Mixture of two Lognormal distributions", plot = FALSE)
```

```
## Warning in hist.default(heart.attk$age, freq = FALSE, xlab = "age", main = "age-
## Mixture of two Lognormal distributions", : arguments 'freq', 'main', 'xlab' are
## not made use of
```

```
## $breaks
##  [1] 40 45 50 55 60 65 70 75 80 85 90 95
##
## $counts
##  [1] 37 37 38 50 48 37 22 12 10  5  3
##
## $density
##  [1] 0.024749164 0.024749164 0.025418060 0.033444816 0.032107023 0.024749164
##  [7] 0.014715719 0.008026756 0.006688963 0.003344482 0.002006689
```

```
##
## $mids
##  [1] 42.5 47.5 52.5 57.5 62.5 67.5 72.5 77.5 82.5 87.5 92.5
##
## $xname
## [1] "heart.attk$age"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

```
lines(seq(min(heart.attk$age),max(heart.attk$age),length=length(heart.attk$age)),
mxfit.age[["prob"]][1]*dGA(seq(min(heart.attk$age),max(heart.attk$age), length=length(heart.attk$age)),m
lines(seq(min(heart.attk$age),max(heart.attk$age),length=length(heart.attk$age)),
mxfit.age[["prob"]][2]*dGA(seq(min(heart.attk$age),max(heart.attk$age),
length=length(heart.attk$age)),mu=mu.hat2.age,sigma = sigma.hat2.age),lty=2,lwd=3,col=3)
lines(seq(min(heart.attk$age),max(heart.attk$age),length=length(heart.attk$age)),
mxfit.age[["prob"]][1]*dGA(seq(min(heart.attk$age),max(heart.attk$age), length=length(heart.attk$age)),m
mxfit.age[["prob"]][2]*dRG(seq(min(heart.attk$age),max(heart.attk$age),
length=length(heart.attk$age)),mu= mu.hat2.age,sigma = sigma.hat2.age),
lty = 3, lwd = 3, col = 4)
```



Since we have selected K=2, we can see in the plot 3 lines, each corresponding to a distribution.The black line is relative to the first distribution, the dotted green one to the second distribution, while the dotted blue

line is relative to the mixture, i.e. the overall model for all data.

Creatinine_phosphokinase (CPK):

CPK is also a continuous variable in range.

```
length(heart.attk$creatinine_phosphokinase)
```

```
## [1] 299
```

```
length(unique(heart.attk$creatinine_phosphokinase))
```

```
## [1] 208
```

```
summary(heart.attk$creatinine_phosphokinase)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    23.0   116.5   250.0   581.8   582.0  7861.0
```

The length of the sample variable is 299 and the range is 23 to 7861.

```
sd(heart.attk$creatinine_phosphokinase)
```

```
## [1] 970.2879
```

```
var(heart.attk$creatinine_phosphokinase)
```

```
## [1] 941458.6
```

```
# Create the function to find mode.
getmode <- function(v) {
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}

# Create the vector with numbers.
v <- c(heart.attk$creatinine_phosphokinase)

# Calculate the mode using the user function.
result <- getmode(v)
print(result)
```

```
## [1] 582
```

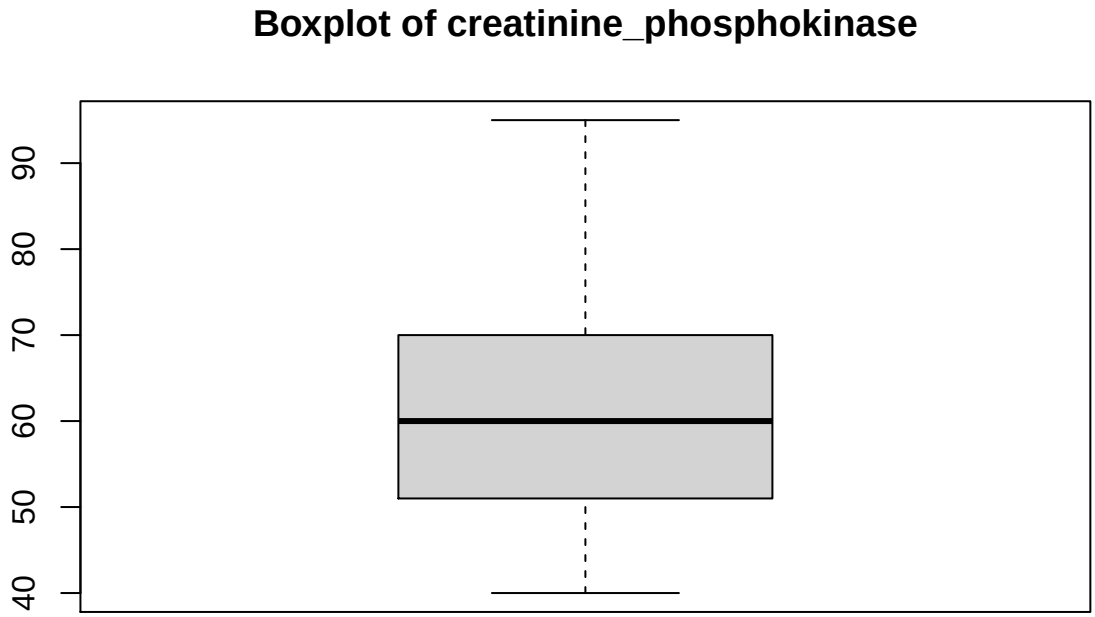
From above statistical computation we can observed that the mean and median are not identical, therefore the distribution is asymmetrical and skewed. And as mean is greater than median, hence the distribution should be skewed to the right and positively skewed.

```
library(e1071)
skewness(heart.attk$creatinine_phosphokinase)
```

```
## [1] 4.41843
```

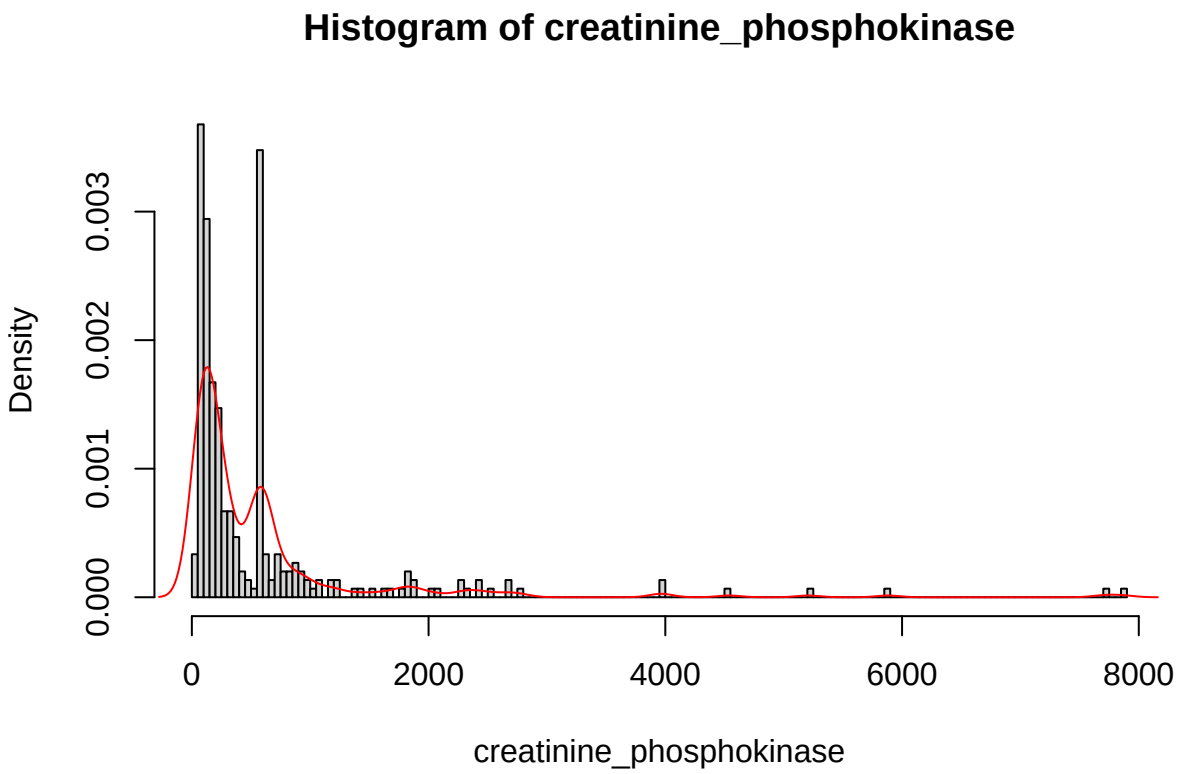
As skewness is positive, hence our prediction was correct.

```
boxplot(heart.attk$age, main = "Boxplot of creatinine_phosphokinase")
points(mean(heart.attk$creatinine_phosphokinase))
```



In the box plot We can observed that there is no outliers in this variable as we didn't see any dot lie outside the whiskers of the boxplot, outliers are the data points located outside the whiskers of the boxplot. Now we can plot the histogram and look for the model that fits better the distribution:

```
hist(heart.attk$creatinine_phosphokinase, prob = T, breaks = 208, freq=FALSE, xlab= "creatinine_phospho
lines(density(heart.attk$creatinine_phosphokinase), col="red")
```

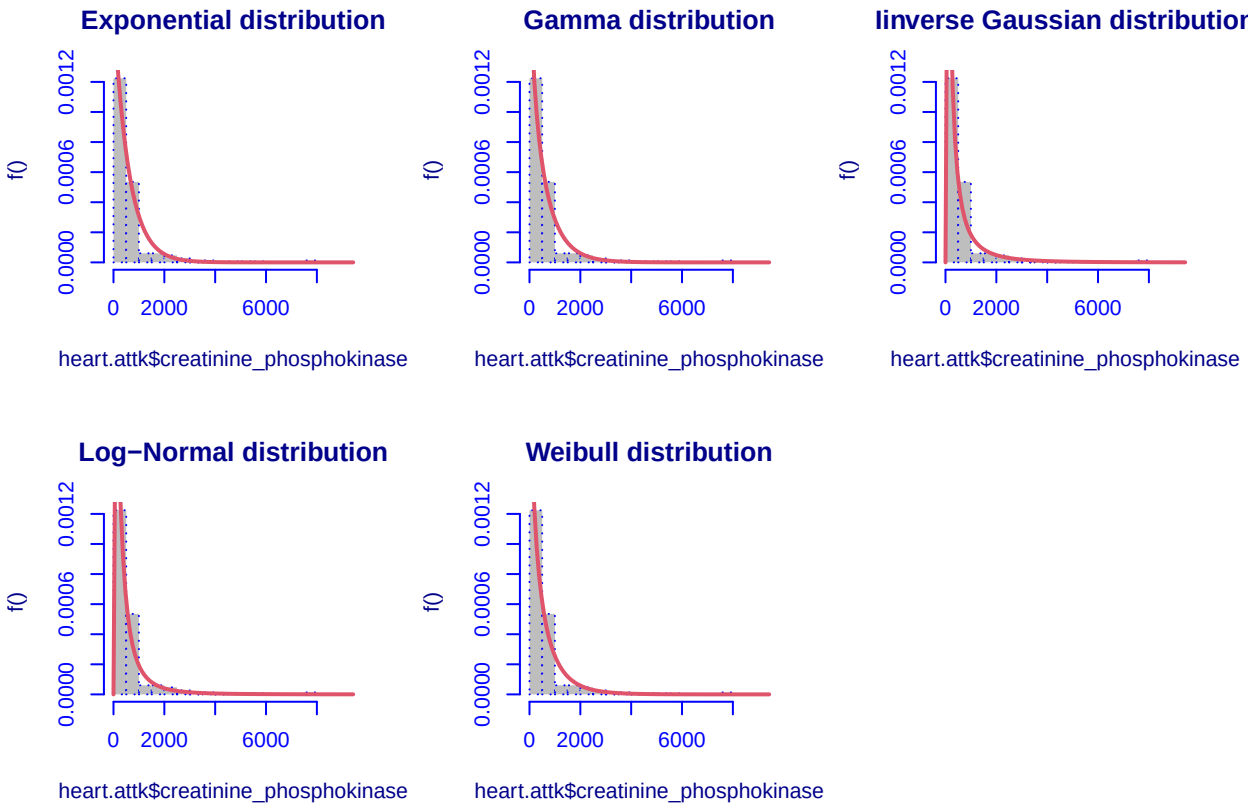


As our dataset is related to the heart patients so above plotted figure showing that majority of patients have Level of the CPK enzyme in the blood is below 2000.

Data fitting for creatinine_phosphokinase:

As per variable domain, the distributions supported on the set of positive infinite real numbers are fitted on the data. Log-likelihood value, BIC and AIC are estimated in order to evaluate the fit of the distribution.

```
library(gamlss.mx)
par(mfrow=c(2,3))
creatinine_phosphokinase.EXP  <-histDist(heart.attk$creatinine_phosphokinase, family=EXP, nbins = 13, ma
creatinine_phosphokinase.GA   <-histDist(heart.attk$creatinine_phosphokinase, family=GA, nbins = 13, mai
creatinine_phosphokinase.IG   <-histDist(heart.attk$creatinine_phosphokinase, family=IG, nbins = 13, mai
creatinine_phosphokinase.LOGNO<-histDist(heart.attk$creatinine_phosphokinase, family=LOGNO, nbins = 13,
creatinine_phosphokinase.WEI  <-histDist(heart.attk$creatinine_phosphokinase, family=WEI, nbins = 13, ma
```




```
creatinine_phosphokinase.matrix<-matrix(c(creatinine_phosphokinase.LOGNO$df.fit, logLik(creatinine_phos
creatinine_phosphokinase.LOGNO$sbc, creatinine_phosphokinase.GA$df.fit, logLik(creatinine_phosphokinase
creatinine_phosphokinase.GA$sbc, creatinine_phosphokinase.WEI$df.fit, logLik(creatinine_phosphokinase.W
AIC(creatinine_phosphokinase.WEI), creatinine_phosphokinase.WEI$sbc,
creatinine_phosphokinase.EXP$df.fit, logLik(creatinine_phosphokinase.EXP), AIC(creatinine_phosphokinase
creatinine_phosphokinase.IG$df.fit, logLik(creatinine_phosphokinase.IG), AIC(creatinine_phosphokinase.I
colnames(creatinine_phosphokinase.matrix)<-c("df","LogLik", "AIC", "BIC")
rownames(creatinine_phosphokinase.matrix)<-c("LOGNO", "GA", "WEI", "EXP", "IG")
creatinine_phosphokinase.matrix<-as.table(creatinine_phosphokinase.matrix)
creatinine_phosphokinase.matrix
```

```
##              df    LogLik       AIC       BIC
## LOGNO     2.000 -2153.786  4311.572  4318.973
## GA        2.000 -2199.111  4402.223  4409.624
## WEI       2.000 -2191.438  4386.876  4394.277
## EXP       1.000 -2202.492  4406.984  4410.685
## IG        2.000 -2144.726  4293.451  4300.852
```

After above statistical analysis we can observed that the model which has large value of log likelihood and small value in AIC and BIC is “Inverse Gaussian distribution”, hence on the basis of likelihood method Our data is more likely to fit into an inverse Gaussian distribution.

Likelihood ratio test:

The goodness-of-fit test was performed between the Exponential model (under the null hypothesis) and the inverse Gaussian model (under the alternative hypothesis).

```
library(lmtest)
lrtest(creatinine_phosphokinase.IG, creatinine_phosphokinase.EXP)
```

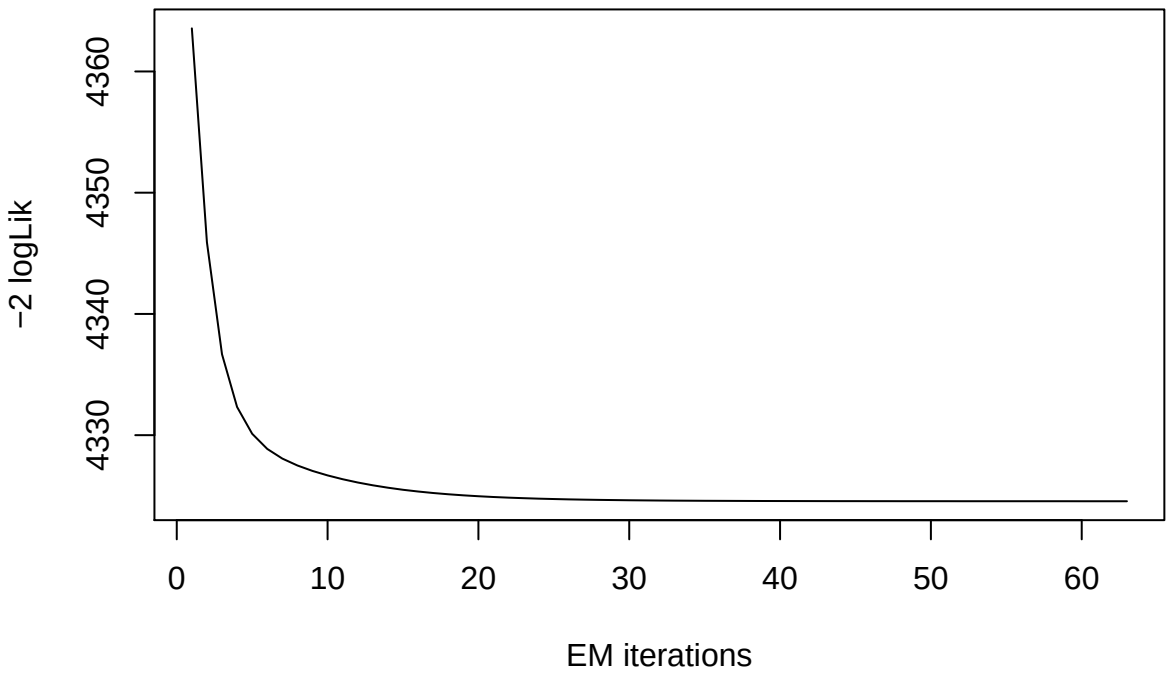
```
## Likelihood ratio test
##
## Model 1: gamlssML(formula = heart.attk$creatinine_phosphokinase, family = "IG")
## Model 2: gamlssML(formula = heart.attk$creatinine_phosphokinase, family = "EXP")
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   2 -2144.7
## 2   1 -2202.5 -1 115.53  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The null hypothesis would be rejected at nearly every significance level. Thus, we know that we should definitely use the IG model as it increases the accuracy of our model by a substantial amount in terms of AIC and BIC values.

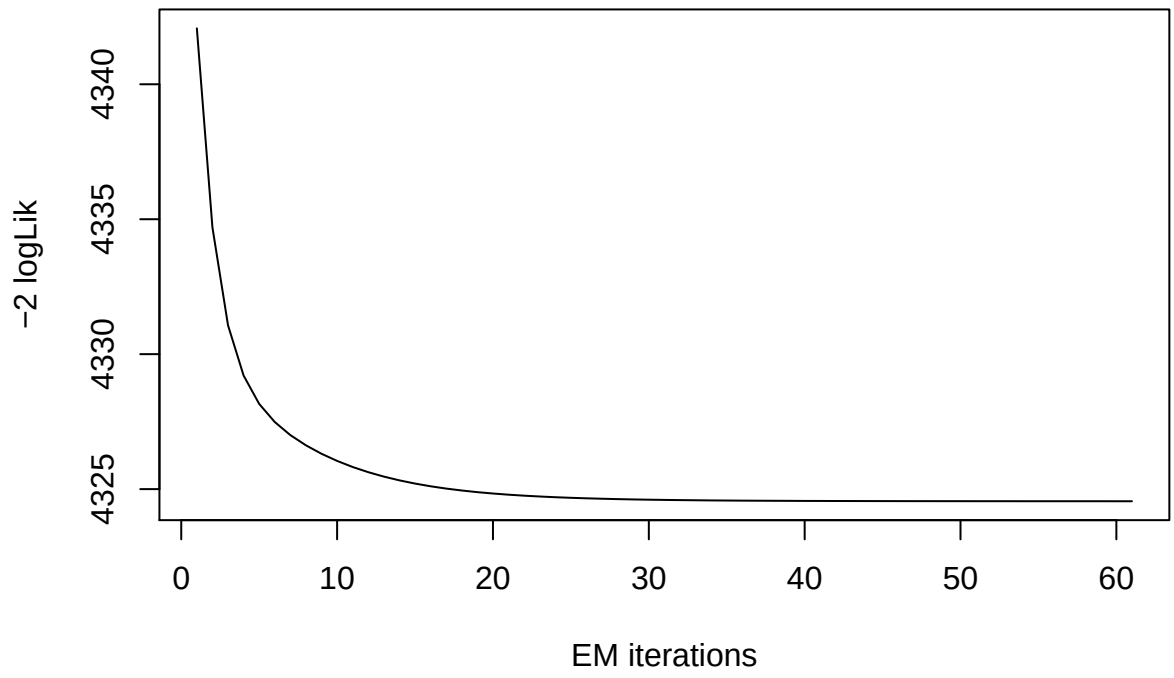
Mixture of distributions:

It is possible to compute a mixture of two gamma distributions In order to find the best mixture,the algorithm is repeated five times.

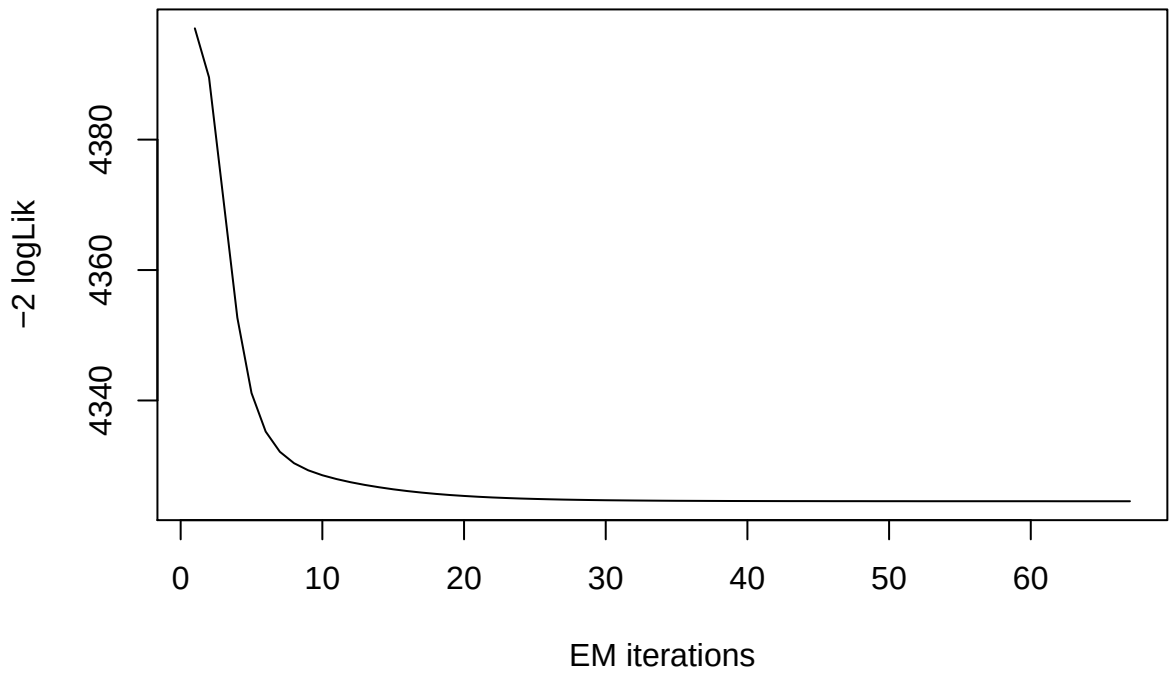
```
mxfit.creatinine_phosphokinase <- gamlssMXfits(n = 5, heart.attk$creatinine_phosphokinase~1, family = G
```



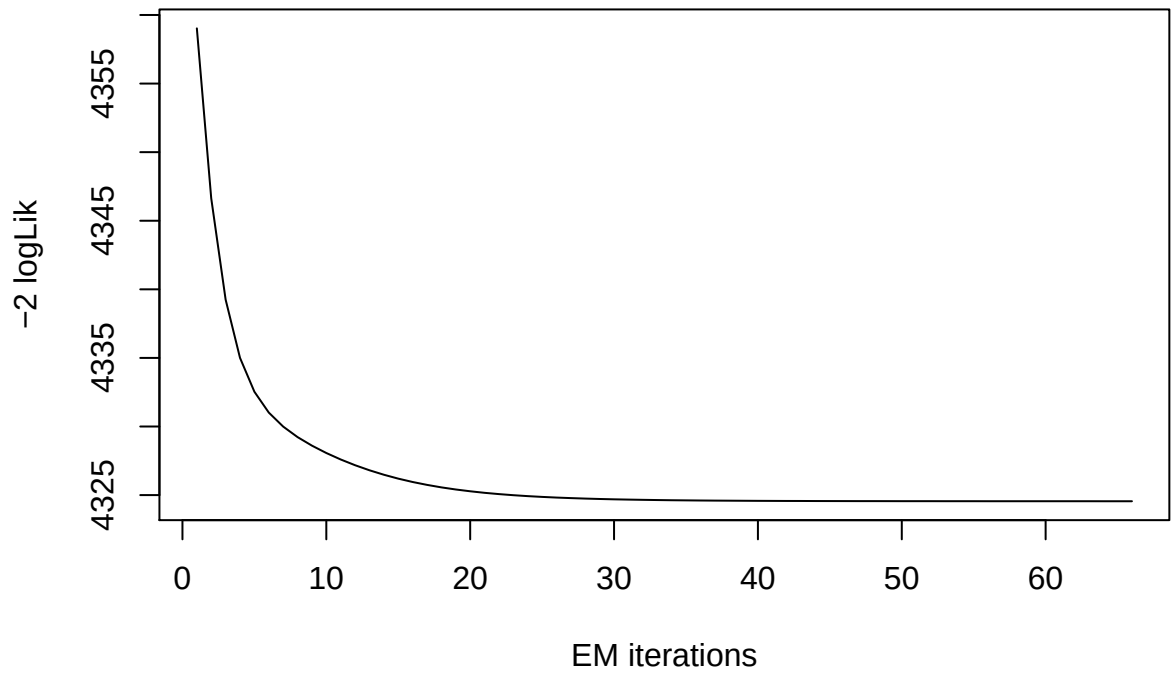

```
## model= 1
```



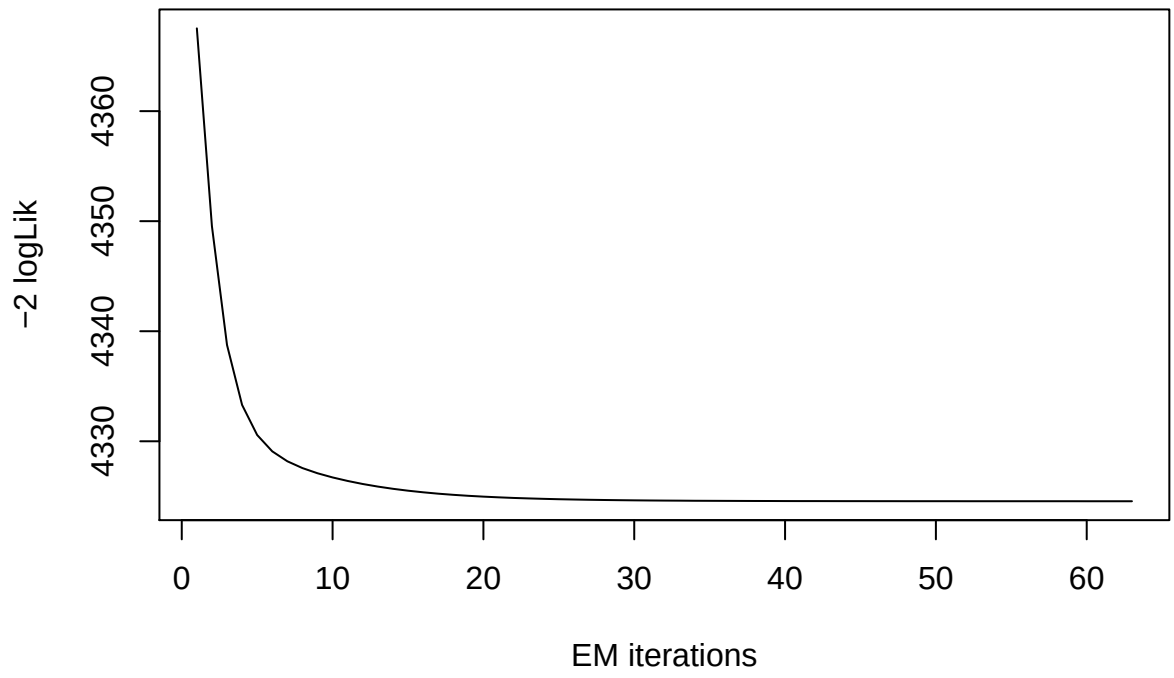

```
## model= 2
```




```
## model= 3
```




```
## model= 4
```




```
## model= 5
```

```
print(mxfit.creatinine_phosphokinase)
```

```
##
## Mixing Family:  c("GA", "GA")
##
## Fitting method: EM algorithm
##
## Call:  gamlssMX(formula = heart.attk$creatinine_phosphokinase ~
##     1, family = GA, K = 2, data = heart.attk)
##
## Mu Coefficients for model: 1
## (Intercept)
##       7.461
## Sigma Coefficients for model: 1
## (Intercept)
##     -0.0782
## Mu Coefficients for model: 2
## (Intercept)
##       5.714
## Sigma Coefficients for model: 2
## (Intercept)
##     -0.2003
##
```

```
## Estimated probabilities: 0.1941478 0.8058522
##
## Degrees of Freedom for the fit: 5 Residual Deg. of Freedom   294
## Global Deviance:     4324.55
##             AIC:     4334.55
##             SBC:     4353.05
```

Here we can observed that the AIC and BIC value has improved with the combination of two Gamma distributions, as it is lower than the one obtained with a single Gamma distribution, but still greater than the IG distribution so hence we can say that IG distribution is best bit.

```
logLik(mxfit.creatinine_phosphokinase)
```

```
## 'log Lik.' -2162.275 (df=5)
```

```
mxfit.creatinine_phosphokinase$prob
```

```
## [1] 0.1941478 0.8058522
```

```
fitted(mxfit.creatinine_phosphokinase, "mu")[1]
```

```
## [1] 581.6915
```

```
fitted(mxfit.creatinine_phosphokinase, "sigma")[2]
```

```
## [1] 581.6915
```

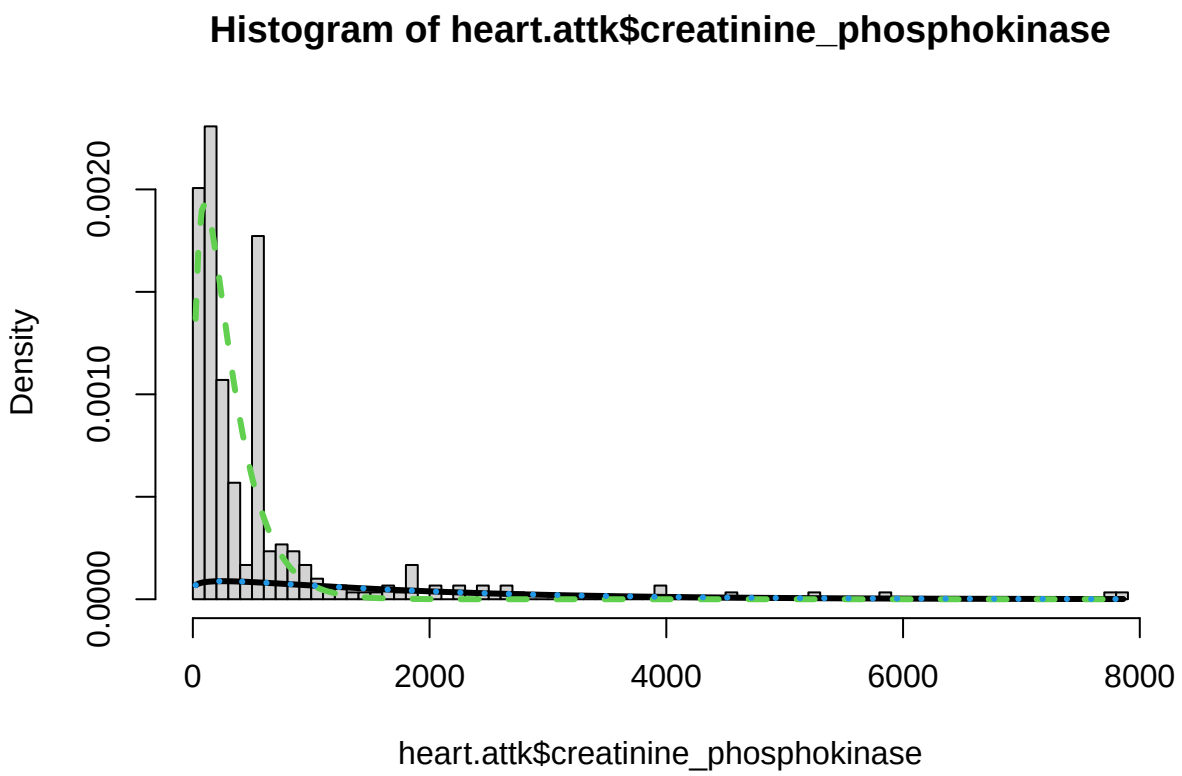
```
hist(heart.attk$creatinine_phosphokinase, breaks = 100,freq = FALSE)
mu.hat1.cpk <- exp(mxfit.creatinine_phosphokinase[["models"]][[1]][["mu.coefficients"]])
sigma.hat1.cpk <- exp(mxfit.creatinine_phosphokinase[["models"]][[1]][["sigma.coefficients"]])
mu.hat2.cpk <- exp(mxfit.creatinine_phosphokinase[["models"]][[2]][["mu.coefficients"]])
sigma.hat2.cpk <- exp(mxfit.creatinine_phosphokinase[["models"]][[2]][["sigma.coefficients"]])
hist(heart.attk$creatinine_phosphokinase, breaks = 100, freq = FALSE, xlab = "creatinine_phosphokinase"
main="CPK-Mixture of two Lognormal distributions", plot = FALSE)
```

```
## Warning in hist.default(heart.attk$creatinine_phosphokinase, breaks = 100, :
## arguments 'freq', 'main', 'xlab' are not made use of
```

```
## $breaks
##  [1]    0  100  200  300  400  500  600  700  800  900 1000 1100 1200 1300 1400
## [16] 1500 1600 1700 1800 1900 2000 2100 2200 2300 2400 2500 2600 2700 2800 2900
## [31] 3000 3100 3200 3300 3400 3500 3600 3700 3800 3900 4000 4100 4200 4300 4400
## [46] 4500 4600 4700 4800 4900 5000 5100 5200 5300 5400 5500 5600 5700 5800 5900
## [61] 6000 6100 6200 6300 6400 6500 6600 6700 6800 6900 7000 7100 7200 7300 7400
## [76] 7500 7600 7700 7800 7900
##
## $counts
##  [1] 60 69 32 17  5 53  7  8  7  5  3  2  2  1  1  1  2  1  5  0  2  0  2  1  2
## [26]  1  2  1  0  0  0  0  0  0  0  0  0  0  0  2  0  0  0  0  0  1  0  0  0  0
## [51]  0  0  1  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

```
## [76]  0  0  1  1
##
## $density
##  [1] 2.006689e-03 2.307692e-03 1.070234e-03 5.685619e-04 1.672241e-04
##  [6] 1.772575e-03 2.341137e-04 2.675585e-04 2.341137e-04 1.672241e-04
## [11] 1.003344e-04 6.688963e-05 6.688963e-05 3.344482e-05 3.344482e-05
## [16] 3.344482e-05 6.688963e-05 3.344482e-05 1.672241e-04 0.000000e+00
## [21] 6.688963e-05 0.000000e+00 6.688963e-05 3.344482e-05 6.688963e-05
## [26] 3.344482e-05 6.688963e-05 3.344482e-05 0.000000e+00 0.000000e+00
## [31] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
## [36] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 6.688963e-05
## [41] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
## [46] 3.344482e-05 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
## [51] 0.000000e+00 0.000000e+00 3.344482e-05 0.000000e+00 0.000000e+00
## [56] 0.000000e+00 0.000000e+00 0.000000e+00 3.344482e-05 0.000000e+00
## [61] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
## [66] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
## [71] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
## [76] 0.000000e+00 0.000000e+00 3.344482e-05 3.344482e-05
##
## $mids
##  [1]   50  150  250  350  450  550  650  750  850  950 1050 1150 1250 1350 1450
## [16] 1550 1650 1750 1850 1950 2050 2150 2250 2350 2450 2550 2650 2750 2850 2950
## [31] 3050 3150 3250 3350 3450 3550 3650 3750 3850 3950 4050 4150 4250 4350 4450
## [46] 4550 4650 4750 4850 4950 5050 5150 5250 5350 5450 5550 5650 5750 5850 5950
## [61] 6050 6150 6250 6350 6450 6550 6650 6750 6850 6950 7050 7150 7250 7350 7450
## [76] 7550 7650 7750 7850
##
## $xname
## [1] "heart.attk$creatinine_phosphokinase"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

```
lines(seq(min(heart.attk$creatinine_phosphokinase),max(heart.attk$creatinine_phosphokinase),length=leng
      mxfit.creatinine_phosphokinase[["prob"]][1]*dGA(seq(min(heart.attk$creatinine_phosphokinase),max(
lines(seq(min(heart.attk$creatinine_phosphokinase),max(heart.attk$creatinine_phosphokinase),length=leng
      mxfit.creatinine_phosphokinase[["prob"]][2]*dGA(seq(min(heart.attk$creatinine_phosphokinase),max(
length=length(heart.attk$creatinine_phosphokinase)),mu=mu.hat2.cpk,sigma = sigma.hat2.cpk),lty=2,lwd=3,
lines(seq(min(heart.attk$creatinine_phosphokinase),max(heart.attk$creatinine_phosphokinase),length=leng
      mxfit.creatinine_phosphokinase[["prob"]][1]*dGA(seq(min(heart.attk$creatinine_phosphokinase),max(
        mxfit.creatinine_phosphokinase[["prob"]][2]*dRG(seq(min(heart.attk$creatinine_phosphokinase),ma
length=length(heart.attk$creatinine_phosphokinase)),mu= mu.hat2.cpk,sigma = sigma.hat2.cpk),
lty = 3, lwd = 3, col = 4)
```



Since we have selected K=2, we can see in the plot 3 lines, each corresponding to a distribution.The black line is relative to the first distribution, the dotted green one to the second distribution, while the dotted blue line is relative to the mixture, i.e. the overall model for all data.

Serum_creatinine:

```
length(heart.attk$serum_creatinine)
```

```
## [1] 299
```

```
length(unique(heart.attk$serum_creatinine))
```

```
## [1] 40
```

```
summary(heart.attk$serum_creatinine)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.500   0.900   1.100   1.394   1.400   9.400
```

The length of the sample variable is 299 and the range is 0.5 to 9.4.

```r
sd(heart.attk$serum_creatinine)
```

```
## [1] 1.03451
```

```r
var(heart.attk$serum_creatinine)
```

```
## [1] 1.070211
```

```r
# Create the function to find mode.
getmode <- function(v) {
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}

# Create the vector with numbers.
v <- c(heart.attk$creatinine_phosphokinase)

# Calculate the mode using the user function.
result <- getmode(v)
print(result)
```

```
## [1] 582
```

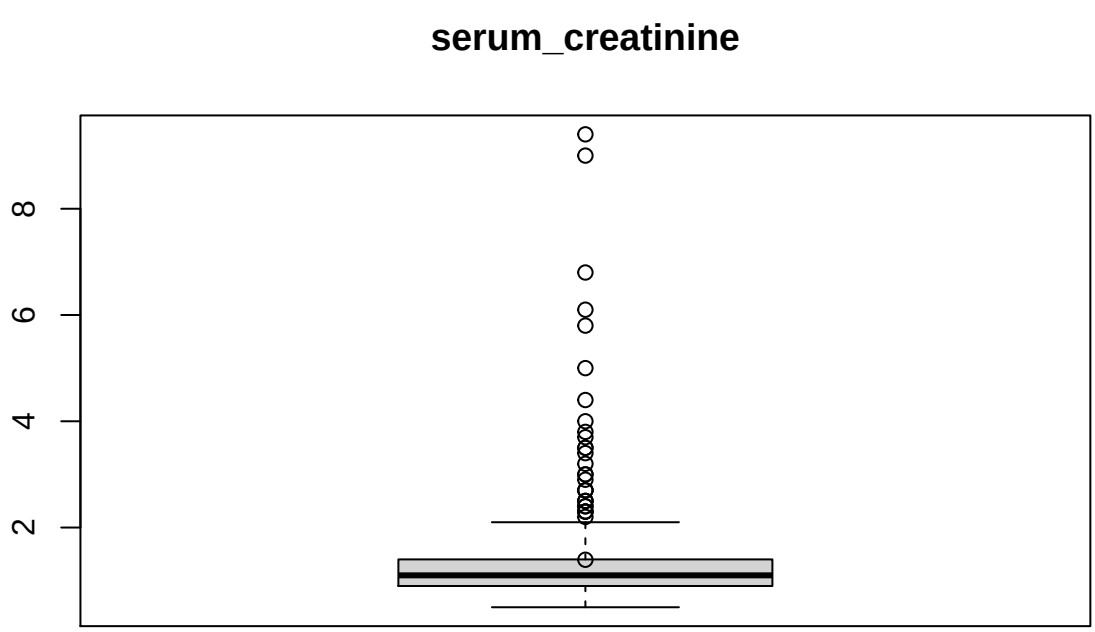
From above statistical computation we can observed that the mean and median are not identical, therefore the distribution is asymmetrical and skewed. And as mean is greater than median, hence the distribution should be skewed to the right and positively skewed.

```r
library(e1071)
skewness(heart.attk$serum_creatinine)
```

```
## [1] 4.411387
```

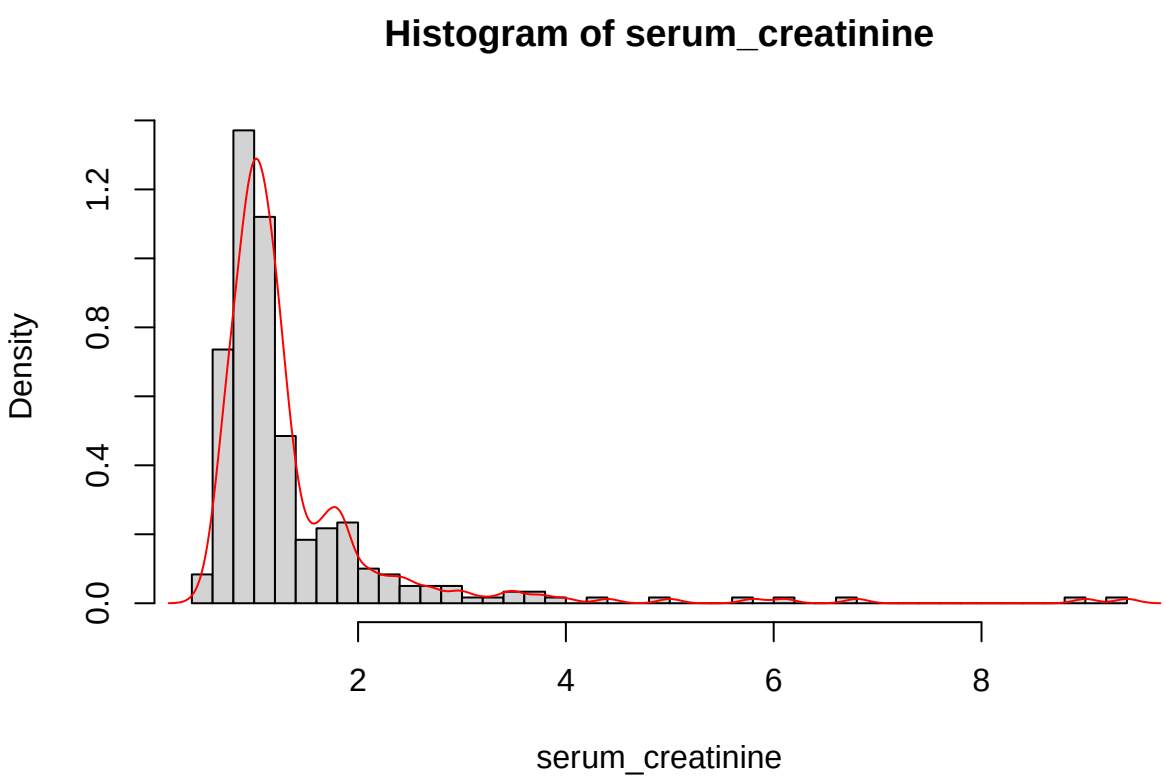
As skewness is positive, hence our prediction was correct.

```r
boxplot(heart.attk$serum_creatinine, main = "serum_creatinine")
points(mean(heart.attk$serum_creatinine))
```



In the box plot We can observed that there is outliers in this sample variable as we see some dots lie outside the upper whiskers of the boxplot, which means that there is some observations that is numerically distant from the rest of the data. Outliers are the data points located outside the whiskers of the boxplot. Now we can plot the histogram and look for the model that fits better the distribution

```
hist(heart.attk$serum_creatinine, prob = T, breaks = 40, freq=FALSE, xlab= "serum_creatinine", main = "
lines(density(heart.attk$serum_creatinine), col="red")
```

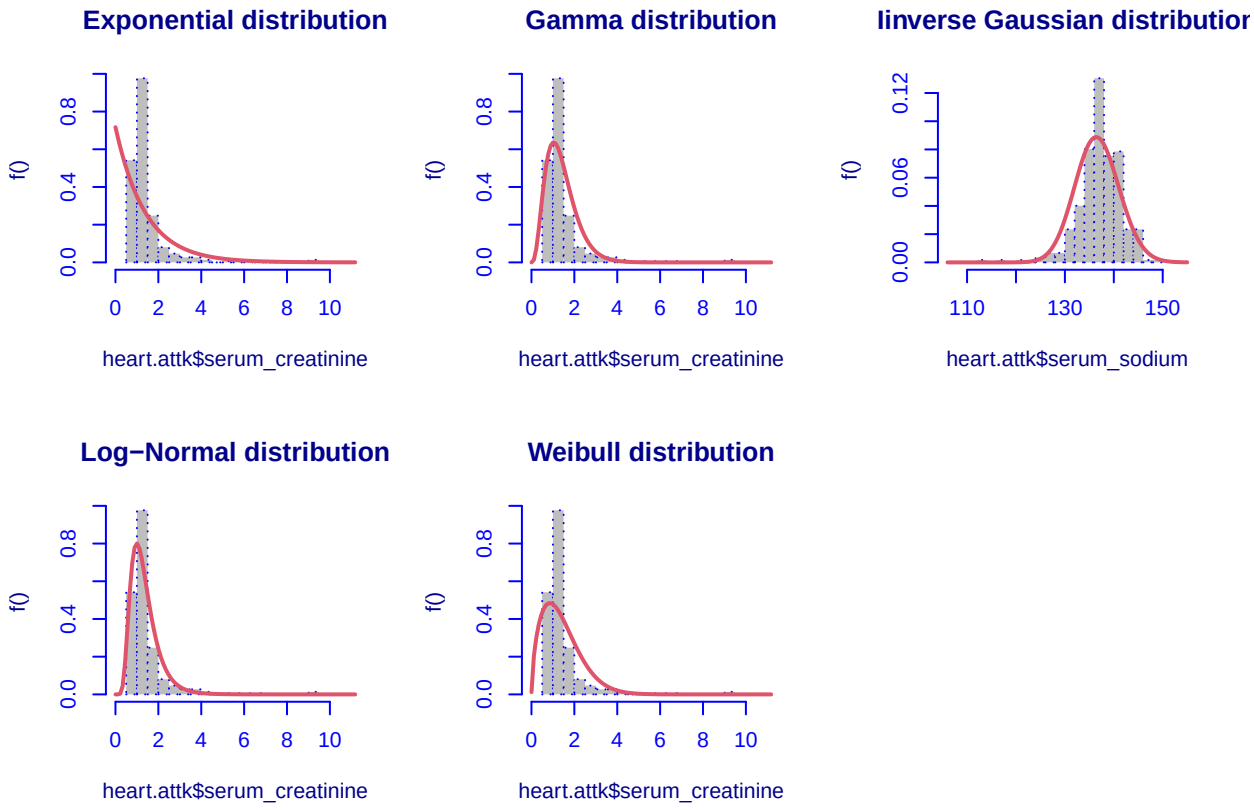


The typical range for serum creatinine is 0.7 to 1.2 milligrams per deciliter (mg/dL) for men and 0.5 to 1.0 (mg/dL) for women. As our dataset is related to the heart patients so from above plotted figure we can say that most of the patients lie within the normal range.

Data fitting for serum_creatinine:

As per variable domain, the distributions supported on the set of positive infinite real numbers are fitted on the data. Log-likelihood value, BIC and AIC are estimated in order to evaluate the fit of the distribution.

```
library(gamlss.mx)
par(mfrow=c(2,3))
serum_creatinine.EXP  <-histDist(heart.attk$serum_creatinine, family=EXP, nbins = 13, main="Exponential
serum_creatinine.GA   <-histDist(heart.attk$serum_creatinine, family=GA, nbins = 13, main="Gamma distri
serum_creatinine.IG   <-histDist(heart.attk$serum_sodium, family=IG, nbins = 13, main="Iinverse Gaussia
serum_creatinine.LOGNO<-histDist(heart.attk$serum_creatinine, family=LOGNO, nbins = 13, main="Log-Norma
serum_creatinine.WEI  <-histDist(heart.attk$serum_creatinine, family=WEI, nbins = 13, main="Weibull dis
```




```
serum_creatinine.matrix<-matrix(c(serum_creatinine.LOGNO$df.fit, logLik(serum_creatinine.LOGNO), AIC(se
serum_creatinine.LOGNO$sbc, serum_creatinine.GA$df.fit, logLik(serum_creatinine.GA), AIC(serum_creatini
serum_creatinine.GA$sbc, serum_creatinine.WEI$df.fit, logLik(serum_creatinine.WEI),
AIC(serum_creatinine.WEI), serum_creatinine.WEI$sbc,
serum_creatinine.EXP$df.fit, logLik(serum_creatinine.EXP), AIC(serum_creatinine.EXP), serum_creatinine.
serum_creatinine.IG$df.fit, logLik(serum_creatinine.IG), AIC(serum_creatinine.IG), serum_creatinine.IG$
colnames(serum_creatinine.matrix)<-c("df","LogLik", "AIC", "BIC")
rownames(serum_creatinine.matrix)<-c("LOGNO", "GA", "WEI", "EXP", "IG")
serum_creatinine.matrix<-as.table(serum_creatinine.matrix)
serum_creatinine.matrix
```

```
##              df    LogLik       AIC       BIC
## LOGNO    2.0000 -246.4434  496.8868  504.2877
## GA       2.0000 -292.7282  589.4564  596.8573
## WEI      2.0000 -339.2119  682.4239  689.8248
## EXP      1.0000 -398.2952  798.5904  802.2908
## IG       2.0000 -873.5110 1751.0220 1758.4229
```

After above statistical analysis we can observed that the model which has large value of log likelihood and small value in AIC and BIC is “Log normal distribution”, hence on the basis of likelihood method Our data is more likely to fit into an log normal distribution.

Likelihood ratio test:

The goodness-of-fit test was performed between the Inverse Guassian model (under the null hypothesis) and the Log Normal model (under the alternative hypothesis).

```
library(lmtest)
lrtest(serum_creatinine.LOGNO, serum_creatinine.IG)
```

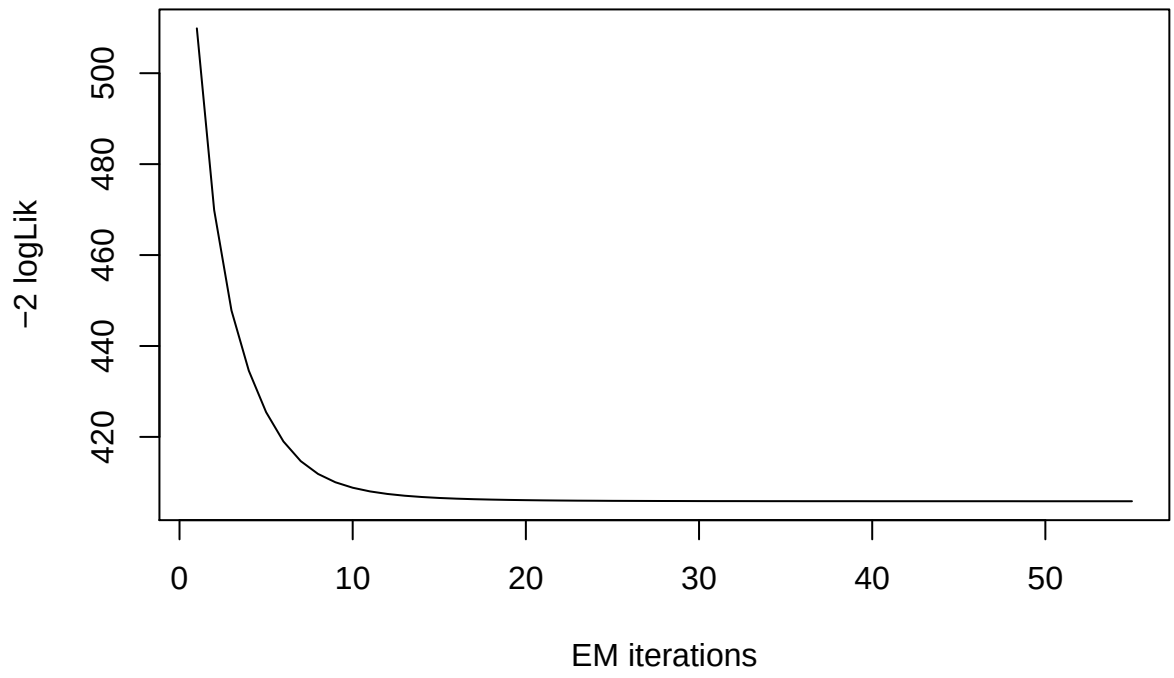
```
## Likelihood ratio test
##
## Model 1: gamlssML(formula = heart.attk$serum_creatinine, family = "LOGNO")
## Model 2: gamlssML(formula = heart.attk$serum_sodium, family = "IG")
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   2 -246.44
## 2   2 -873.51  0 1254.1  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The null hypothesis would be rejected at nearly every significance level. Thus, we know that we should definitely use the log normal model as it increases the accuracy of our model by a substantial amount in terms of both AIC and BIC.

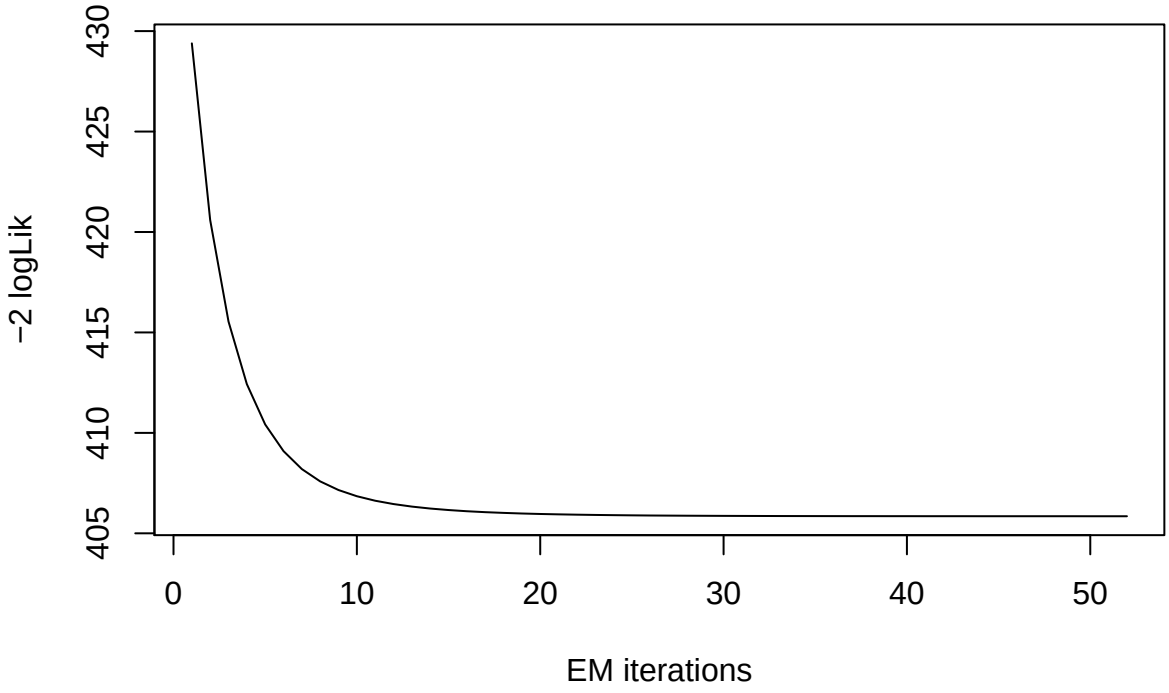
Mixture of distributions:

It is possible to compute a mixture of two gamma distributions In order to find the best mixture,the algorithm is repeated five times.

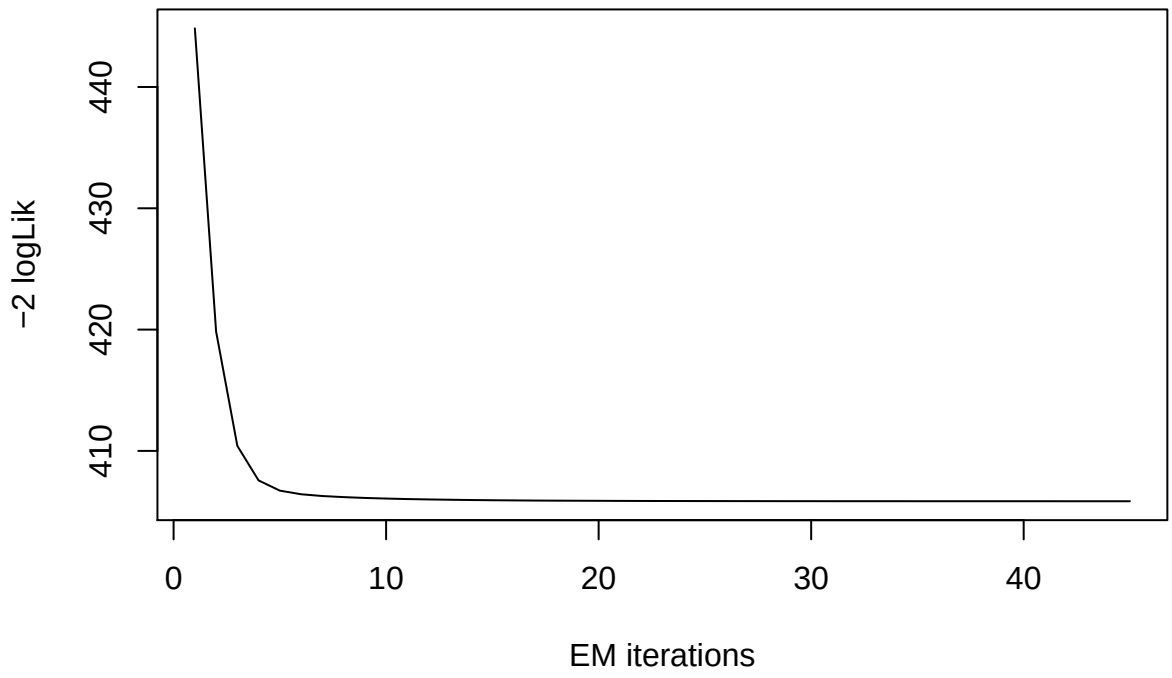
```
mxfit.serum_creatinine <- gamlssMXfits(n = 5, heart.attk$serum_creatinine~1, family = GA, K = 2, data =
```



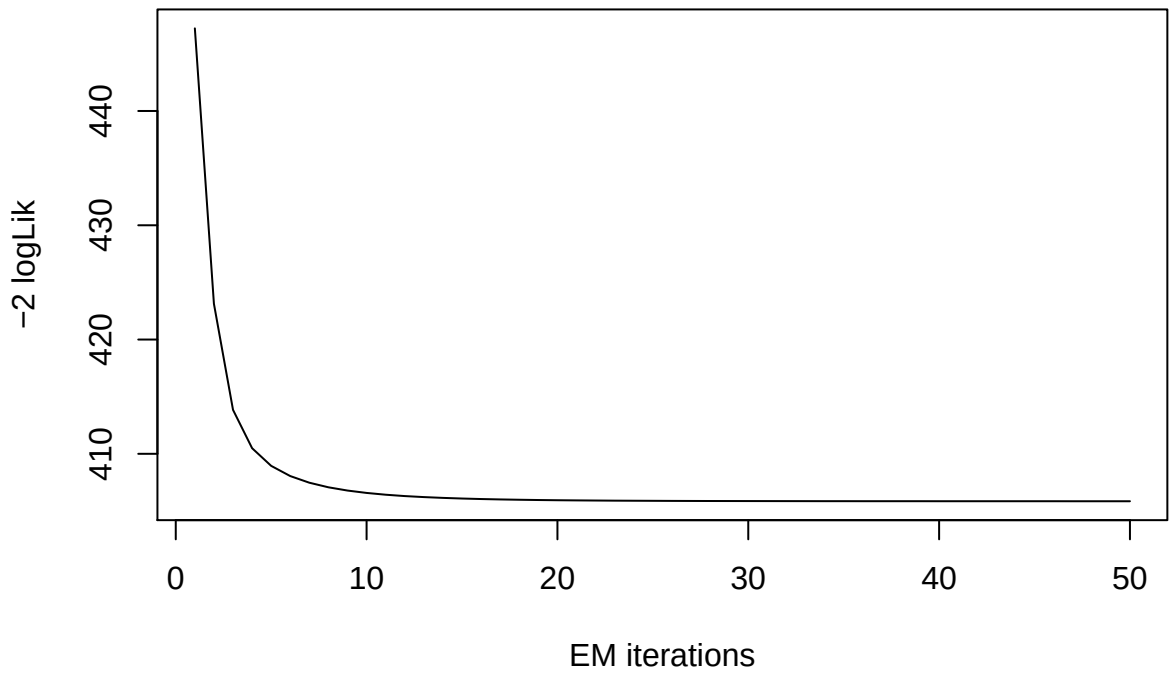

```
## model= 1
```



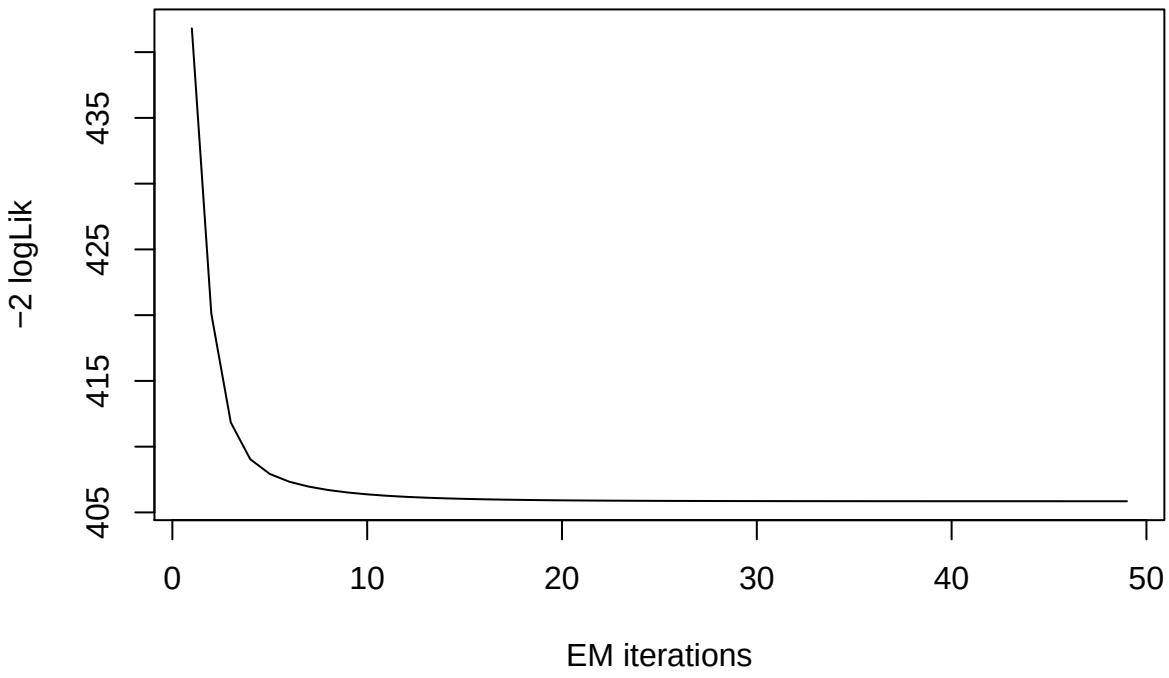

```
## model= 2
```




```
## model= 3
```




```
## model= 4
```




```
## model= 5
```

```
print(mxfit.serum_creatinine)
```

```
##
## Mixing Family:  c("GA", "GA")
##
## Fitting method: EM algorithm
##
## Call:  gamlssMX(formula = heart.attk$serum_creatinine ~ 1,
##     family = GA, K = 2, data = heart.attk)
##
## Mu Coefficients for model: 1
## (Intercept)
##      0.8627
## Sigma Coefficients for model: 1
## (Intercept)
##     -0.5677
## Mu Coefficients for model: 2
## (Intercept)
##      0.0502
## Sigma Coefficients for model: 2
## (Intercept)
##      -1.496
##
```

```
## Estimated probabilities: 0.2596683 0.7403317
##
## Degrees of Freedom for the fit: 5 Residual Deg. of Freedom   294
## Global Deviance:     405.85
##             AIC:     415.85
##             SBC:     434.353
```

Here we can observed that the AIC and BIC value has improved with the combination of two Gamma distributions, as it is lower than the one obtained with a log normal distribution.

```
logLik(mxfit.serum_creatinine)
```

```
## 'log Lik.' -202.9252 (df=5)
```

```
mxfit.serum_creatinine$prob
```

```
## [1] 0.2596683 0.7403317
```

```
fitted(mxfit.serum_creatinine, "mu")[1]
```

```
## [1] 1.393757
```

```
fitted(mxfit.serum_creatinine, "sigma")[2]
```

```
## [1] 1.393757
```

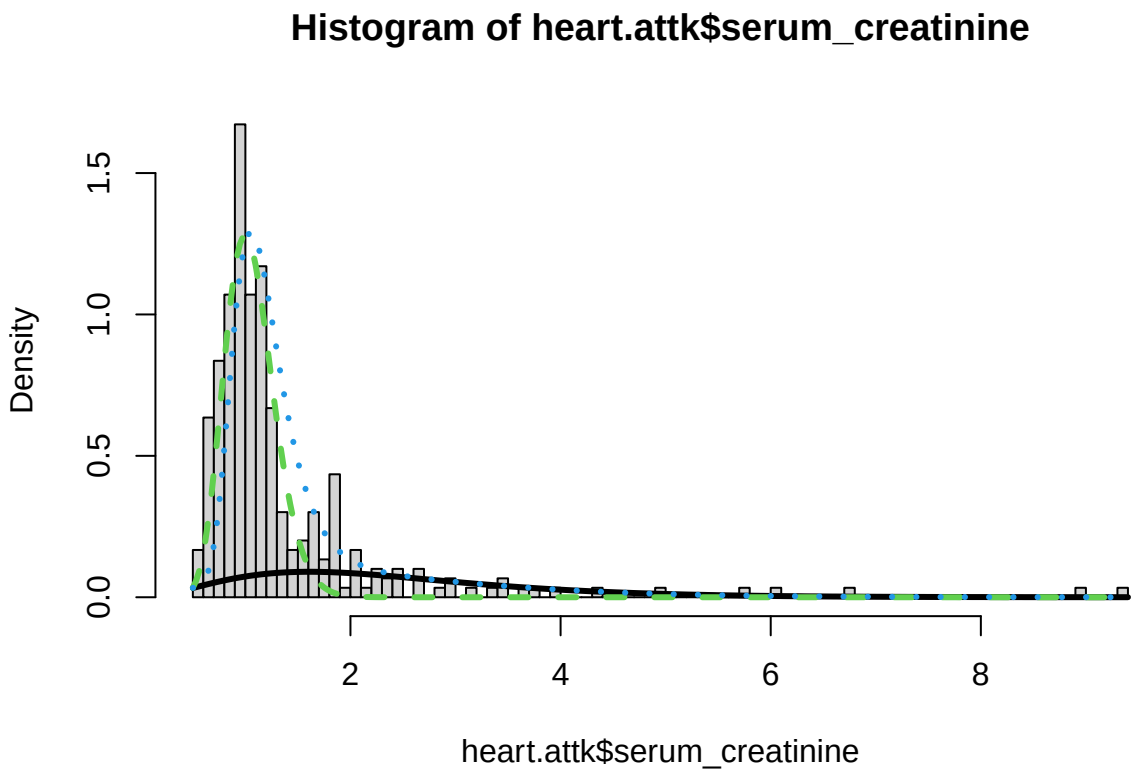
```
hist(heart.attk$serum_creatinine, breaks = 100,freq = FALSE)
mu.hat1.serum_crt <- exp(mxfit.serum_creatinine[["models"]][[1]][["mu.coefficients"]])
sigma.hat1.serum_crt <- exp(mxfit.serum_creatinine[["models"]][[1]][["sigma.coefficients"]])
mu.hat2.serum_crt <- exp(mxfit.serum_creatinine[["models"]][[2]][["mu.coefficients"]])
sigma.hat2.serum_crt <- exp(mxfit.serum_creatinine[["models"]][[2]][["sigma.coefficients"]])
hist(heart.attk$serum_creatinine, breaks = 100, freq = FALSE, xlab = "serum_creatinine" ,
main="serum_creatinine-Mixture of two Lognormal distributions", plot = FALSE)
```

```
## Warning in hist.default(heart.attk$serum_creatinine, breaks = 100, freq =
## FALSE, : arguments 'freq', 'main', 'xlab' are not made use of
```

```
## $breaks
##  [1] 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0 2.1 2.2 2.3
## [20] 2.4 2.5 2.6 2.7 2.8 2.9 3.0 3.1 3.2 3.3 3.4 3.5 3.6 3.7 3.8 3.9 4.0 4.1 4.2
## [39] 4.3 4.4 4.5 4.6 4.7 4.8 4.9 5.0 5.1 5.2 5.3 5.4 5.5 5.6 5.7 5.8 5.9 6.0 6.1
## [58] 6.2 6.3 6.4 6.5 6.6 6.7 6.8 6.9 7.0 7.1 7.2 7.3 7.4 7.5 7.6 7.7 7.8 7.9 8.0
## [77] 8.1 8.2 8.3 8.4 8.5 8.6 8.7 8.8 8.9 9.0 9.1 9.2 9.3 9.4
##
## $counts
##  [1]  5 19 25 32 50 32 35 20  9  5  6  9  4 13  1  5  1  3  2  3  0  3  0  1  2
## [26]  0  1  0  1  2  0  1  1  0  1  0  0  0  1  0  0  0  0  0  1  0  0  0  0  0
## [51]  0  0  1  0  0  1  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0
## [76]  0  0  0  0  0  0  0  0  0  1  0  0  0  1
##
```

```
## $density
##  [1] 0.16722408 0.63545151 0.83612040 1.07023411 1.67224080 1.07023411
##  [7] 1.17056856 0.66889632 0.30100334 0.16722408 0.20066890 0.30100334
## [13] 0.13377926 0.43478261 0.03344482 0.16722408 0.03344482 0.10033445
## [19] 0.06688963 0.10033445 0.00000000 0.10033445 0.00000000 0.03344482
## [25] 0.06688963 0.00000000 0.03344482 0.00000000 0.03344482 0.06688963
## [31] 0.00000000 0.03344482 0.03344482 0.00000000 0.03344482 0.00000000
## [37] 0.00000000 0.00000000 0.03344482 0.00000000 0.00000000 0.00000000
## [43] 0.00000000 0.00000000 0.03344482 0.00000000 0.00000000 0.00000000
## [49] 0.00000000 0.00000000 0.00000000 0.00000000 0.03344482 0.00000000
## [55] 0.00000000 0.03344482 0.00000000 0.00000000 0.00000000 0.00000000
## [61] 0.00000000 0.00000000 0.03344482 0.00000000 0.00000000 0.00000000
## [67] 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## [73] 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## [79] 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## [85] 0.03344482 0.00000000 0.00000000 0.00000000 0.03344482
##
## $mids
##  [1] 0.55 0.65 0.75 0.85 0.95 1.05 1.15 1.25 1.35 1.45 1.55 1.65 1.75 1.85 1.95
## [16] 2.05 2.15 2.25 2.35 2.45 2.55 2.65 2.75 2.85 2.95 3.05 3.15 3.25 3.35 3.45
## [31] 3.55 3.65 3.75 3.85 3.95 4.05 4.15 4.25 4.35 4.45 4.55 4.65 4.75 4.85 4.95
## [46] 5.05 5.15 5.25 5.35 5.45 5.55 5.65 5.75 5.85 5.95 6.05 6.15 6.25 6.35 6.45
## [61] 6.55 6.65 6.75 6.85 6.95 7.05 7.15 7.25 7.35 7.45 7.55 7.65 7.75 7.85 7.95
## [76] 8.05 8.15 8.25 8.35 8.45 8.55 8.65 8.75 8.85 8.95 9.05 9.15 9.25 9.35
##
## $xname
## [1] "heart.attk$serum_creatinine"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

```
lines(seq(min(heart.attk$serum_creatinine),max(heart.attk$serum_creatinine),length=length(heart.attk$se
      mxfit.serum_creatinine[["prob"]][1]*dGA(seq(min(heart.attk$serum_creatinine),max(heart.attk$serum_
lines(seq(min(heart.attk$serum_creatinine),max(heart.attk$serum_creatinine),length=length(heart.attk$se
      mxfit.serum_creatinine[["prob"]][2]*dGA(seq(min(heart.attk$serum_creatinine),max(heart.attk$serum_
length=length(heart.attk$serum_creatinine)),mu=mu.hat2.serum_crt,sigma = sigma.hat2.serum_crt),lty=2,lwd
lines(seq(min(heart.attk$serum_creatinine),max(heart.attk$serum_creatinine),length=length(heart.attk$se
      mxfit.serum_creatinine[["prob"]][1]*dGA(seq(min(heart.attk$serum_creatinine),max(heart.attk$serum_
        mxfit.serum_creatinine[["prob"]][2]*dRG(seq(min(heart.attk$serum_creatinine),max(heart.attk$seru
length=length(heart.attk$serum_creatinine)),mu= mu.hat2.serum_crt,sigma = sigma.hat2.serum_crt),
lty = 3, lwd = 3, col = 4)
```



Since we have selected K=2, we can see in the plot 3 lines, each corresponding to a distribution.The black line is relative to the first distribution, the dotted green one to the second distribution, while the dotted blue line is relative to the mixture, i.e. the overall model for all data.

Serum_sodium:

```
length(heart.attk$serum_sodium)
```

```
## [1] 299
```

```
summary(heart.attk$serum_sodium)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   113.0   134.0   137.0   136.6   140.0   148.0
```

The length of the sample variable is 299 and the range is 113 to 148.

```
sd(heart.attk$serum_sodium)
```

```
## [1] 4.412477
```

```r
var(heart.attk$serum_sodium)
```

```
## [1] 19.46996
```

```r
# Create the function to find mode.
getmode <- function(v) {
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}

# Create the vector with numbers.
v <- c(heart.attk$serum_sodium)

# Calculate the mode using the user function.
result <- getmode(v)
print(result)
```

```
## [1] 136
```

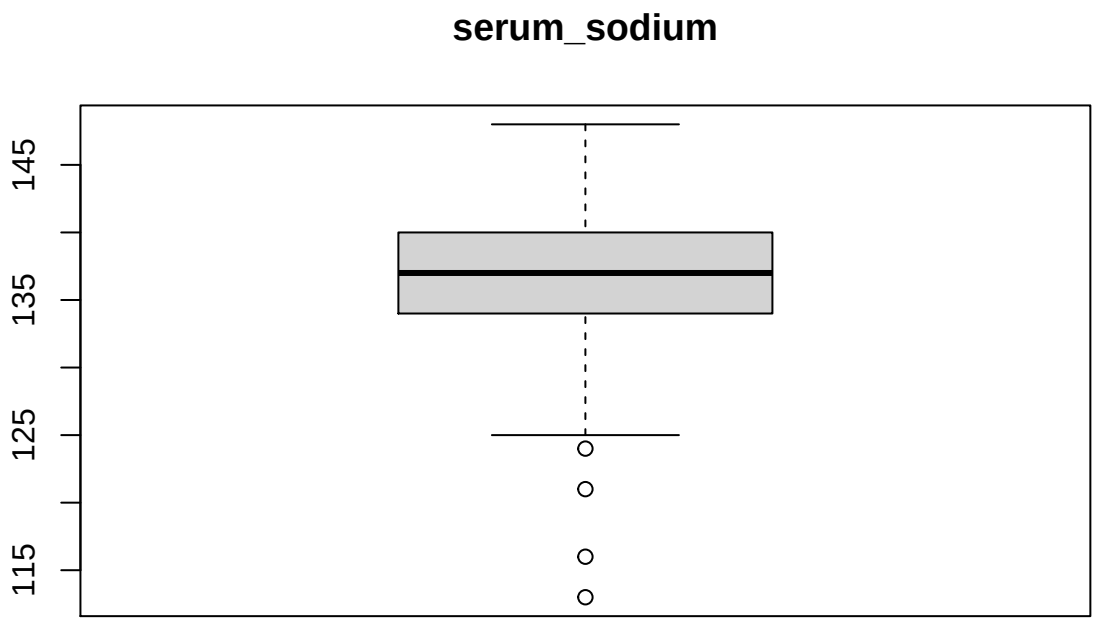
From above statistical computation we can observed that the mean and median are not identical, therefore the distribution is asymmetrical and skewed. And as mean is smaller than median, hence the distribution should be skewed to the left and negatively skewed.

```r
library(e1071)
skewness(heart.attk$serum_sodium)
```

```
## [1] -1.037643
```

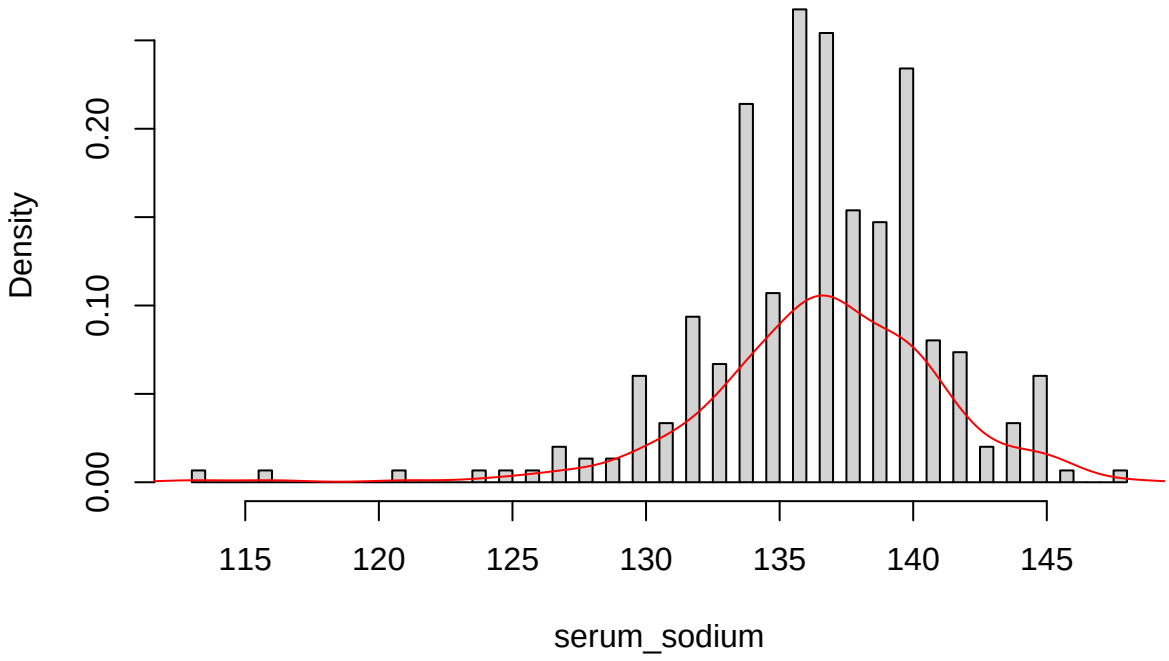
As skewness is negative, hence our prediction was correct.

```r
boxplot(heart.attk$serum_sodium, main = "serum_sodium")
points(mean(heart.attk$serum_creatinine))
```



In the box plot We can observed that there is outliers in this sample variable as we see some dots lie outside the lower whiskers of the boxplot, which means that there is some observations that is numerically distant from the rest of the data. Outliers are the data points located outside the whiskers of the boxplot. Now we can plot the histogram and look for the model that fits better the distribution

```
hist(heart.attk$serum_sodium, prob = T, breaks = 50, freq=FALSE, xlab= "serum_sodium", main = "Histogra
lines(density(heart.attk$serum_sodium), col="red")
```

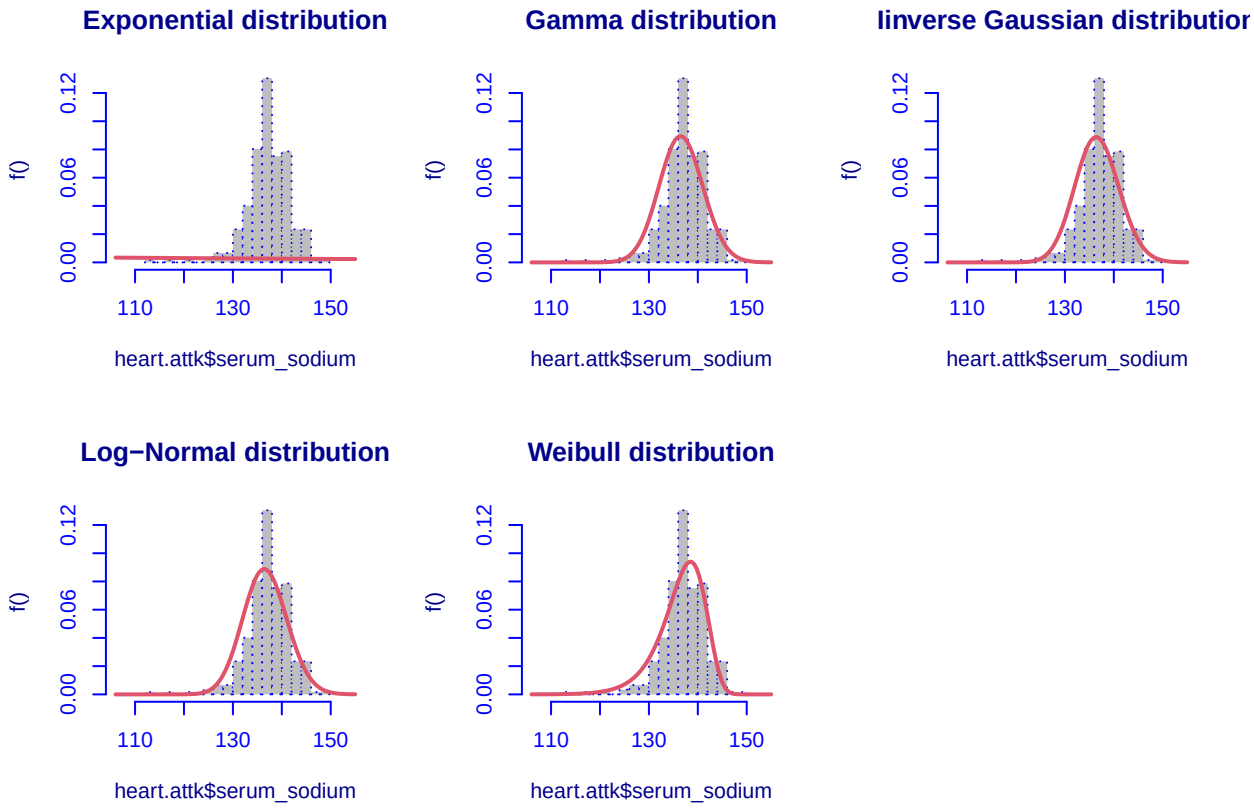


The range for serum sodium is 135-147 mEq/L and as our dataset is related to the heart patients so from above plotted figure we can say that most of the patients lie within a normal range of serum sodium level in their blood and some lie below the 135. so they should intake Intravenous fluids with a high-concentration of sodium or diuretics to raise their blood sodium levels.

Data fitting for serum_sodium:

As per variable domain, the distributions supported on the set of positive infinite real numbers are fitted on the data. Log-likelihood value, BIC and AIC are estimated in order to evaluate the fit of the distribution.

```
library(gamlss.mx)
par(mfrow=c(2,3))
serum_sodium.EXP  <-histDist(heart.attk$serum_sodium, family=EXP, nbins = 13, main="Exponential distribu
serum_sodium.GA   <-histDist(heart.attk$serum_sodium, family=GA, nbins = 13, main="Gamma distribution")
serum_sodium.IG   <-histDist(heart.attk$serum_sodium, family=IG, nbins = 13, main="Iinverse Gaussian di
serum_sodium.LOGNO<-histDist(heart.attk$serum_sodium, family=LOGNO, nbins = 13, main="Log-Normal distril
serum_sodium.WEI  <-histDist(heart.attk$serum_sodium, family=WEI, nbins = 13, main="Weibull distributio
```




```
serum_sodium.matrix<-matrix(c(serum_sodium.LOGNO$df.fit, logLik(serum_sodium.LOGNO), AIC(serum_sodium.L
serum_sodium.LOGNO$sbc, serum_sodium.GA$df.fit, logLik(serum_sodium.GA), AIC(serum_sodium.GA),
serum_sodium.GA$sbc, serum_sodium.WEI$df.fit, logLik(serum_sodium.WEI),
AIC(serum_sodium.WEI), serum_sodium.WEI$sbc,
serum_sodium.EXP$df.fit, logLik(serum_sodium.EXP), AIC(serum_sodium.EXP), serum_sodium.EXP$sbc,
serum_sodium.IG$df.fit, logLik(serum_sodium.IG), AIC(serum_sodium.IG), serum_sodium.IG$sbc), ncol=4, by
colnames(serum_sodium.matrix)<-c("df","LogLik", "AIC", "BIC")
rownames(serum_sodium.matrix)<-c("LOGNO", "GA", "WEI", "EXP", "IG")
serum_sodium.matrix<-as.table(serum_sodium.matrix)
serum_sodium.matrix
```

```
##               df     LogLik        AIC        BIC
## LOGNO     2.0000  -873.4379  1750.8759  1758.2768
## GA        2.0000  -871.3797  1746.7595  1754.1603
## WEI       2.0000  -861.1784  1726.3568  1733.7577
## EXP       1.0000 -1769.2557  3540.5113  3544.2118
## IG        2.0000  -873.5110  1751.0220  1758.4229
```

After above statistical analysis we can observed that the model which has large value of log likelihood and small value in AIC and BIC is “Weibull distribution”, hence on the basis of likelihood method Our data is more likely to fit into an log weibull distribution.

Likelihood ratio test:

The goodness-of-fit test was performed between the Exponential model (under the null hypothesis) and the Weibull model (under the alternative hypothesis).

```
library(lmtest)
lrtest(serum_sodium.WEI, serum_sodium.EXP)
```

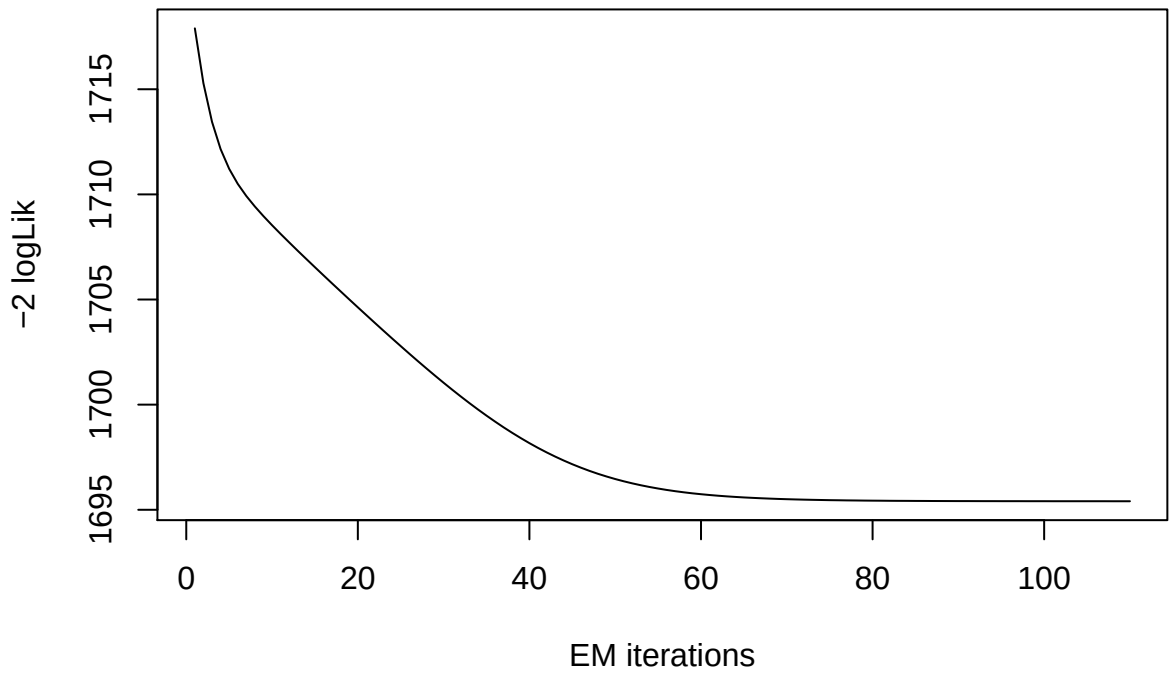
```
## Likelihood ratio test
##
## Model 1: gamlssML(formula = heart.attk$serum_sodium, family = "WEI")
## Model 2: gamlssML(formula = heart.attk$serum_sodium, family = "EXP")
##   #Df   LogLik Df  Chisq Pr(>Chisq)
## 1   2  -861.18
## 2   1 -1769.26 -1 1816.2  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The null hypothesis would be rejected at nearly every significance level. Thus, we know that we should definitely use the weibull model as it increases the accuracy of our model by a substantial amount in terms of both AIC and BIC.

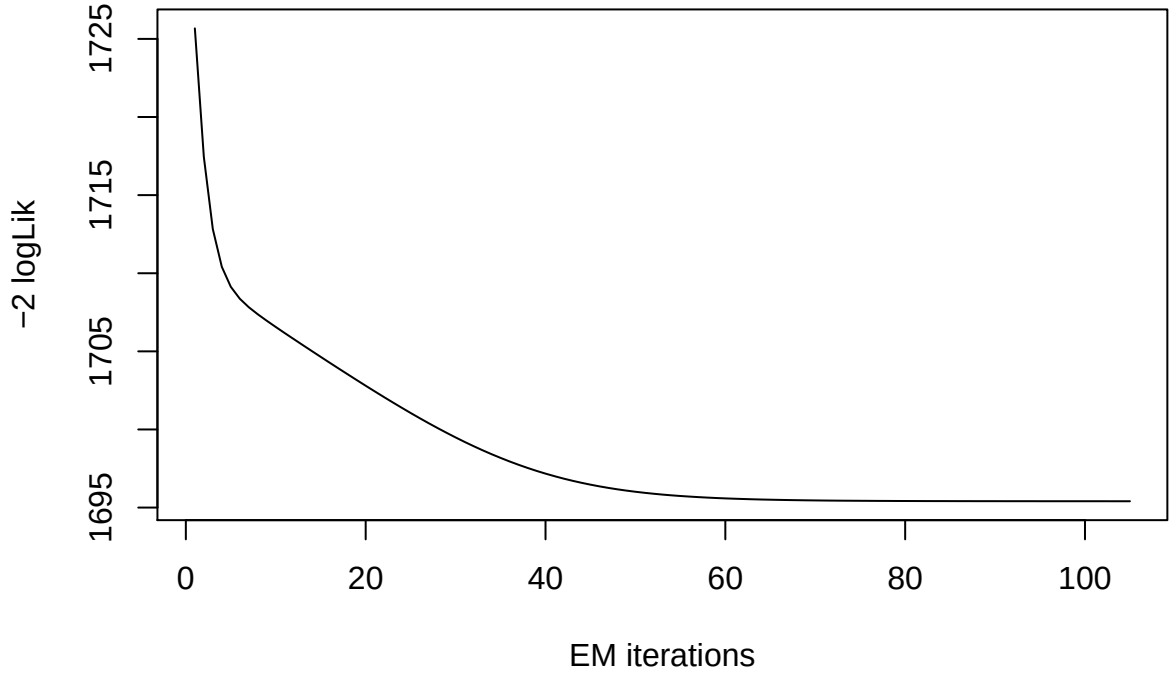
Mixture of distributions:

It is possible to compute a mixture of two gamma distributions In order to find the best mixture,the algorithm is repeated five times.

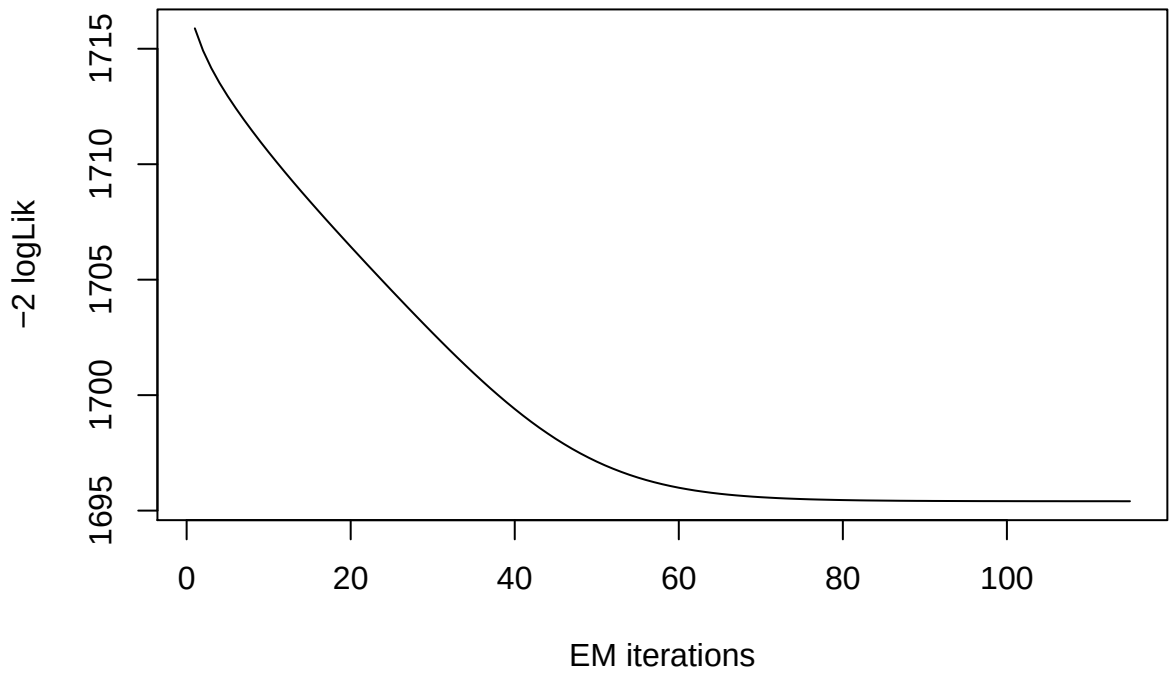
```
mxfit.serum_sodium <- gamlssMXfits(n = 5, heart.attk$serum_sodium~1, family = GA, K = 2, data = heart.a
```


−2 logLik
1695
1700
1705
1710
1715
0
20
40
60
80
100
EM iterations

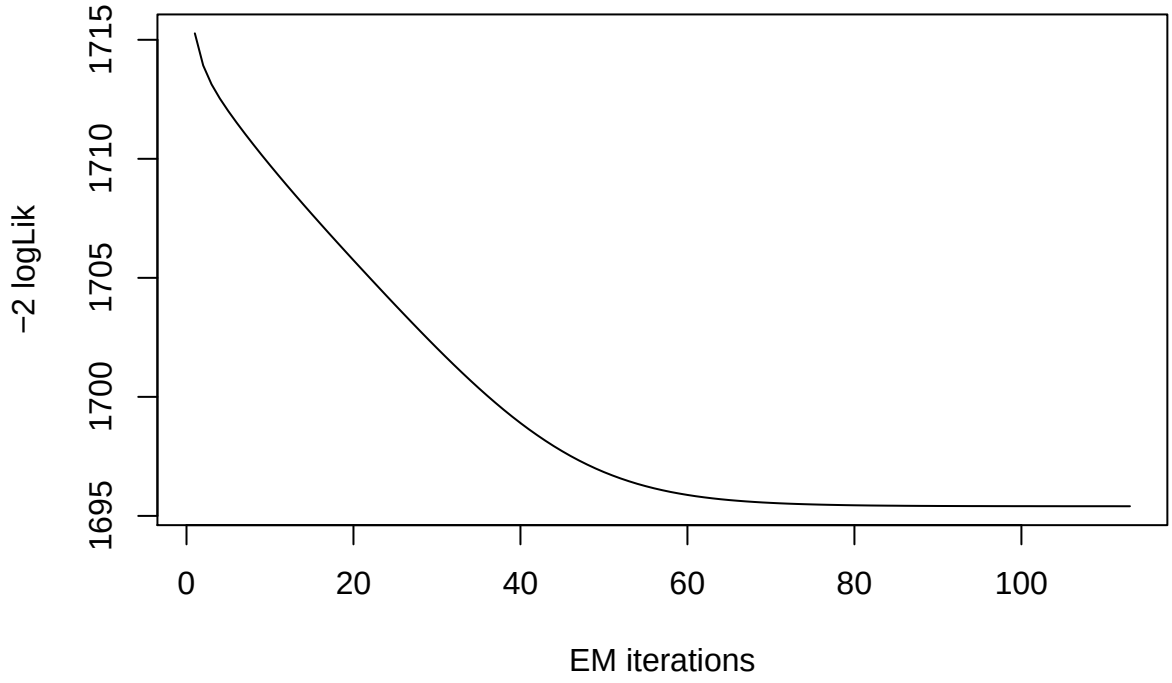
```
## model= 1
```



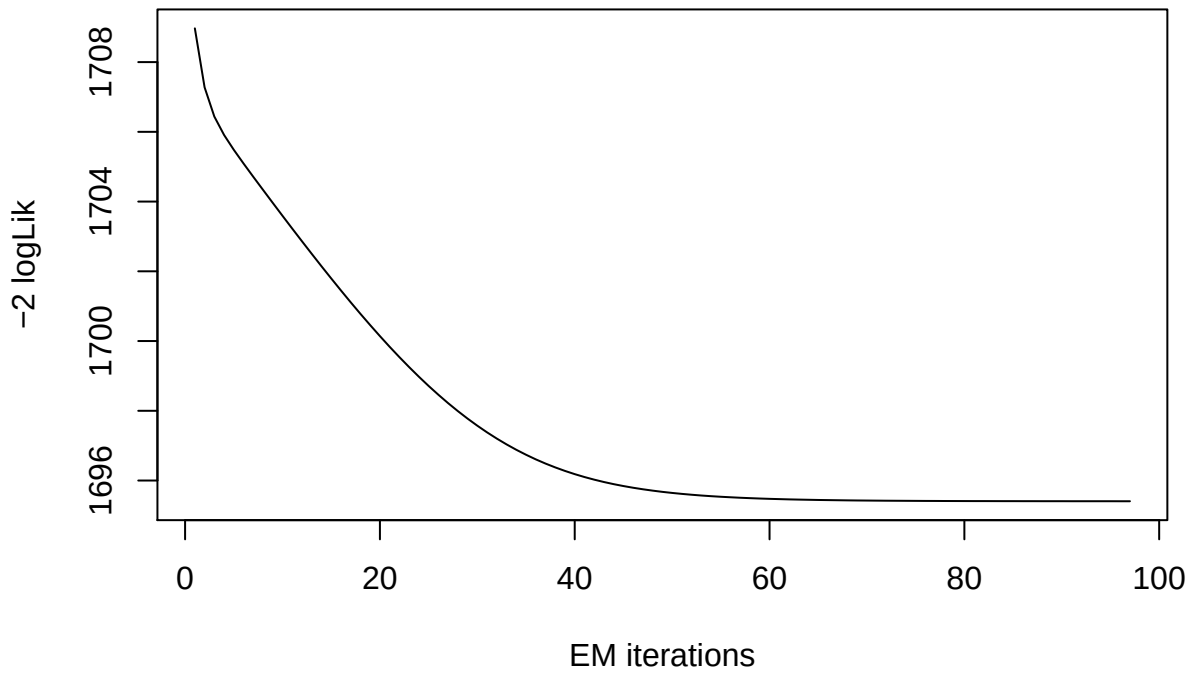

```
## model= 2
```


−2 logLik
1695
1700
1705
1710
1715
0
20
40
60
80
100
EM iterations

```
## model= 3
```


−2 logLik
1695
1700
1705
1710
1715
0
20
40
60
80
100
EM iterations

```
## model= 4
```




```
## model= 5
```

```
print(mxfit.serum_sodium)
```

```
##
## Mixing Family:  c("GA", "GA")
##
## Fitting method: EM algorithm
##
## Call:  gamlssMX(formula = heart.attk$serum_sodium ~ 1, family = GA,
##     K = 2, data = heart.attk)
##
## Mu Coefficients for model: 1
## (Intercept)
##       4.851
## Sigma Coefficients for model: 1
## (Intercept)
##      -2.804
## Mu Coefficients for model: 2
## (Intercept)
##        4.92
## Sigma Coefficients for model: 2
## (Intercept)
##      -3.623
##
```

```
## Estimated probabilities: 0.04777656 0.9522234
##
## Degrees of Freedom for the fit: 5 Residual Deg. of Freedom   294
## Global Deviance:     1695.41
##             AIC:     1705.41
##             SBC:     1723.91
```

Here we can observed that the AIC and BIC value has improved with the combination of two Gamma distributions, as it is lower than the one obtained with a weibull distribution.

```
logLik(mxfit.serum_sodium)
```

```
## 'log Lik.' -847.7029 (df=5)
```

```
mxfit.serum_sodium$prob
```

```
## [1] 0.04777656 0.95222344
```

```
fitted(mxfit.serum_sodium, "mu")[1]
```

```
## [1] 136.626
```

```
fitted(mxfit.serum_sodium, "sigma")[2]
```

```
## [1] 136.626
```

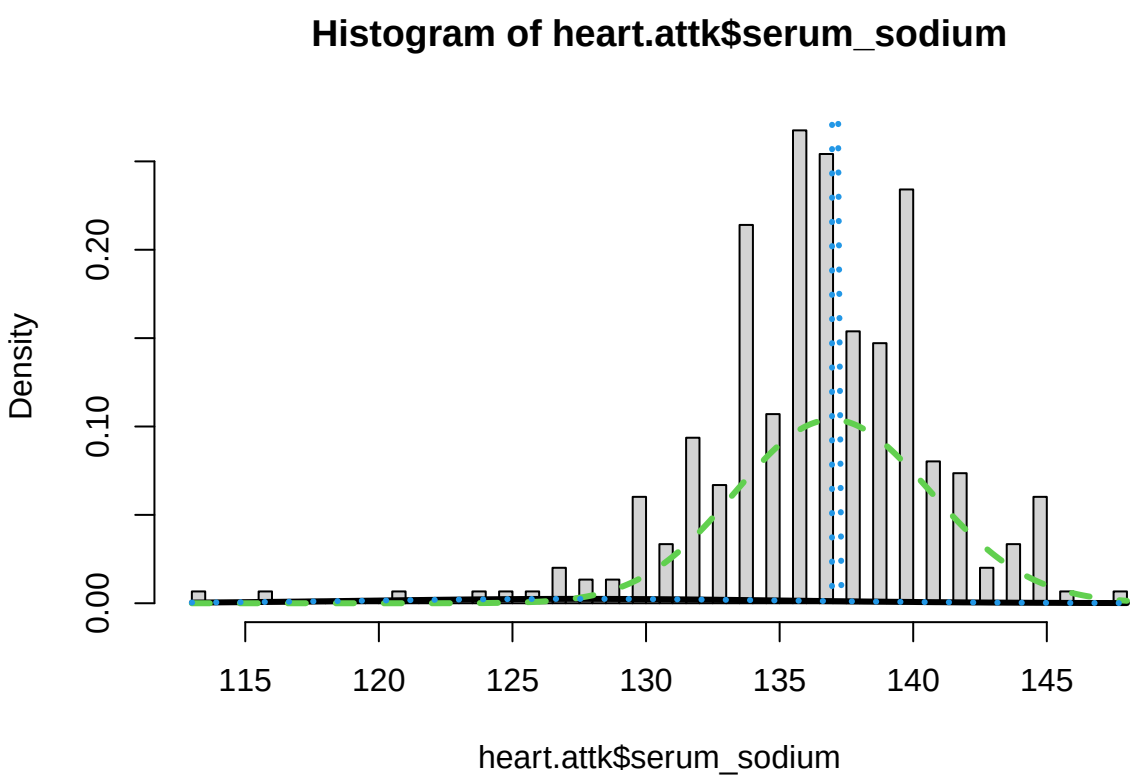
```
hist(heart.attk$serum_sodium, breaks = 100,freq = FALSE)
mu.hat1.serum_sodium <- exp(mxfit.serum_sodium[["models"]][[1]][["mu.coefficients"]])
sigma.hat1.serum_sodium <- exp(mxfit.serum_sodium[["models"]][[1]][["sigma.coefficients"]])
mu.hat2.serum_sodium <- exp(mxfit.serum_sodium[["models"]][[2]][["mu.coefficients"]])
sigma.hat2.serum_sodium <- exp(mxfit.serum_sodium[["models"]][[2]][["sigma.coefficients"]])
hist(heart.attk$serum_sodium, breaks = 100, freq = FALSE, xlab = "serum_sodium" ,main="serum_sodium-Mix
```

```
## Warning in hist.default(heart.attk$serum_sodium, breaks = 100, freq = FALSE, :
## arguments 'freq', 'main', 'xlab' are not made use of
```

```
## $breaks
##  [1] 113.0 113.5 114.0 114.5 115.0 115.5 116.0 116.5 117.0 117.5 118.0 118.5
## [13] 119.0 119.5 120.0 120.5 121.0 121.5 122.0 122.5 123.0 123.5 124.0 124.5
## [25] 125.0 125.5 126.0 126.5 127.0 127.5 128.0 128.5 129.0 129.5 130.0 130.5
## [37] 131.0 131.5 132.0 132.5 133.0 133.5 134.0 134.5 135.0 135.5 136.0 136.5
## [49] 137.0 137.5 138.0 138.5 139.0 139.5 140.0 140.5 141.0 141.5 142.0 142.5
## [61] 143.0 143.5 144.0 144.5 145.0 145.5 146.0 146.5 147.0 147.5 148.0
##
## $counts
##  [1]  1  0  0  0  0  1  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  1  0  1  0
## [26]  1  0  3  0  2  0  2  0  9  0  5  0 14  0 10  0 32  0 16  0 40  0 38  0 23
## [51]  0 22  0 35  0 12  0 11  0  3  0  5  0  9  0  1  0  0  0  1
##
## $density
```

```
##  [1] 0.006688963 0.000000000 0.000000000 0.000000000 0.000000000 0.006688963
##  [7] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
## [13] 0.000000000 0.000000000 0.000000000 0.006688963 0.000000000 0.000000000
## [19] 0.000000000 0.000000000 0.000000000 0.006688963 0.000000000 0.006688963
## [25] 0.000000000 0.006688963 0.000000000 0.020066890 0.000000000 0.013377926
## [31] 0.000000000 0.013377926 0.000000000 0.060200669 0.000000000 0.033444816
## [37] 0.000000000 0.093645485 0.000000000 0.066889632 0.000000000 0.214046823
## [43] 0.000000000 0.107023411 0.000000000 0.267558528 0.000000000 0.254180602
## [49] 0.000000000 0.153846154 0.000000000 0.147157191 0.000000000 0.234113712
## [55] 0.000000000 0.080267559 0.000000000 0.073578595 0.000000000 0.020066890
## [61] 0.000000000 0.033444816 0.000000000 0.060200669 0.000000000 0.006688963
## [67] 0.000000000 0.000000000 0.000000000 0.006688963
##
## $mids
##  [1] 113.25 113.75 114.25 114.75 115.25 115.75 116.25 116.75 117.25 117.75
## [11] 118.25 118.75 119.25 119.75 120.25 120.75 121.25 121.75 122.25 122.75
## [21] 123.25 123.75 124.25 124.75 125.25 125.75 126.25 126.75 127.25 127.75
## [31] 128.25 128.75 129.25 129.75 130.25 130.75 131.25 131.75 132.25 132.75
## [41] 133.25 133.75 134.25 134.75 135.25 135.75 136.25 136.75 137.25 137.75
## [51] 138.25 138.75 139.25 139.75 140.25 140.75 141.25 141.75 142.25 142.75
## [61] 143.25 143.75 144.25 144.75 145.25 145.75 146.25 146.75 147.25 147.75
##
## $xname
## [1] "heart.attk$serum_sodium"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

```
lines(seq(min(heart.attk$serum_sodium),max(heart.attk$serum_sodium),length=length(heart.attk$serum_sodi
      mxfit.serum_sodium[["prob"]][1]*dGA(seq(min(heart.attk$serum_sodium),max(heart.attk$serum_sodium)

lines(seq(min(heart.attk$serum_sodium),max(heart.attk$serum_sodium),length=length(heart.attk$serum_sodi
      mxfit.serum_sodium[["prob"]][2]*dGA(seq(min(heart.attk$serum_sodium),max(heart.attk$serum_sodium)
length=length(heart.attk$serum_sodium)),mu=mu.hat2.serum_sodium,sigma = sigma.hat2.serum_sodium),lty=2,

lines(seq(min(heart.attk$serum_sodium),max(heart.attk$serum_sodium),length=length(heart.attk$serum_sodi
      mxfit.serum_sodium[["prob"]][1]*dGA(seq(min(heart.attk$serum_sodium),max(heart.attk$serum_sodium)
        mxfit.serum_sodium[["prob"]][2]*dRG(seq(min(heart.attk$serum_sodium),max(heart.attk$serum_sodiu
length=length(heart.attk$serum_sodium)),mu= mu.hat2.serum_sodium,sigma = sigma.hat2.serum_sodium),
lty = 3, lwd = 3, col = 4)
```



Since we have selected K=2, we can see in the plot 3 lines, each corresponding to a distribution.The black line is relative to the first distribution, the dotted green one to the second distribution, while the dotted blue line is relative to the mixture, i.e. the overall model for all data.

Ejection_fraction:

```
length(heart.attk$ejection_fraction)
```

```
## [1] 299
```

```
length(unique(heart.attk$ejection_fraction))
```

```
## [1] 17
```

```
summary(heart.attk$ejection_fraction)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   14.00   30.00   38.00   38.08   45.00   80.00
```

The length of the variable is 299 and range is 12 to 80.

```r
sd(heart.attk$ejection_fraction)
```

```
## [1] 11.83484
```

```r
var(heart.attk$ejection_fraction)
```

```
## [1] 140.0635
```

```r
getmode <- function(v) {
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}

# Create the vector with numbers.
v <- c(heart.attk$ejection_fraction)

# Calculate the mode using the user function.
result <- getmode(v)
print(result)
```

```
## [1] 35
```

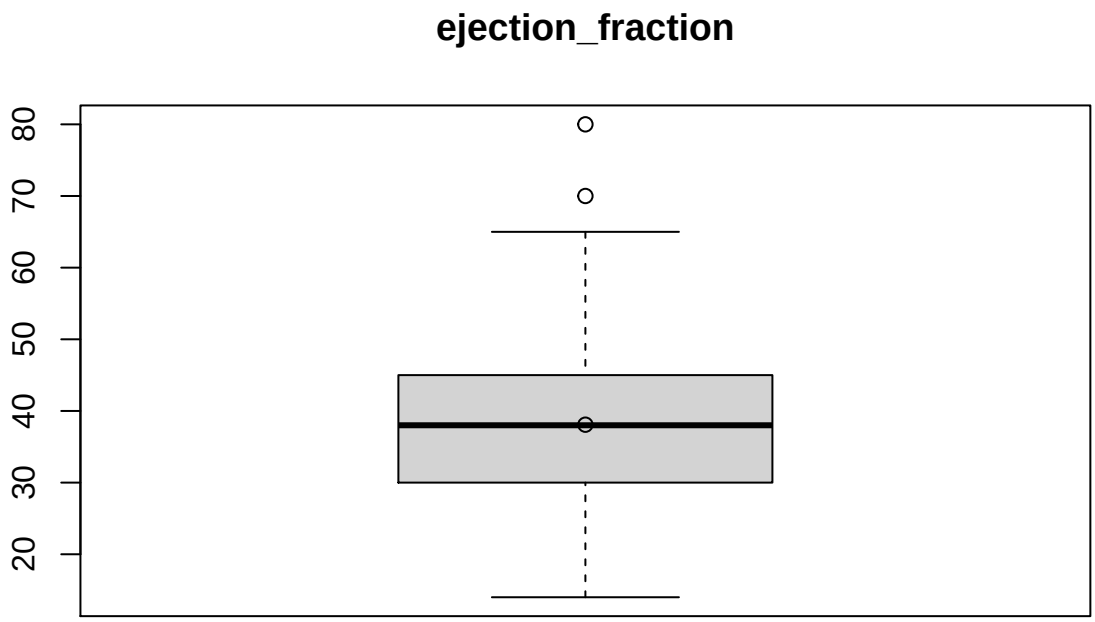
From above statistical computation we can observed that the mean and median are not identical, therefore the distribution is asymmetrical and skewed. And as mean is greater than median, hence the distribution should be skewed to the left and positively skewed.

```r
skewness(heart.attk$ejection_fraction)
```

```
## [1] 0.5498228
```

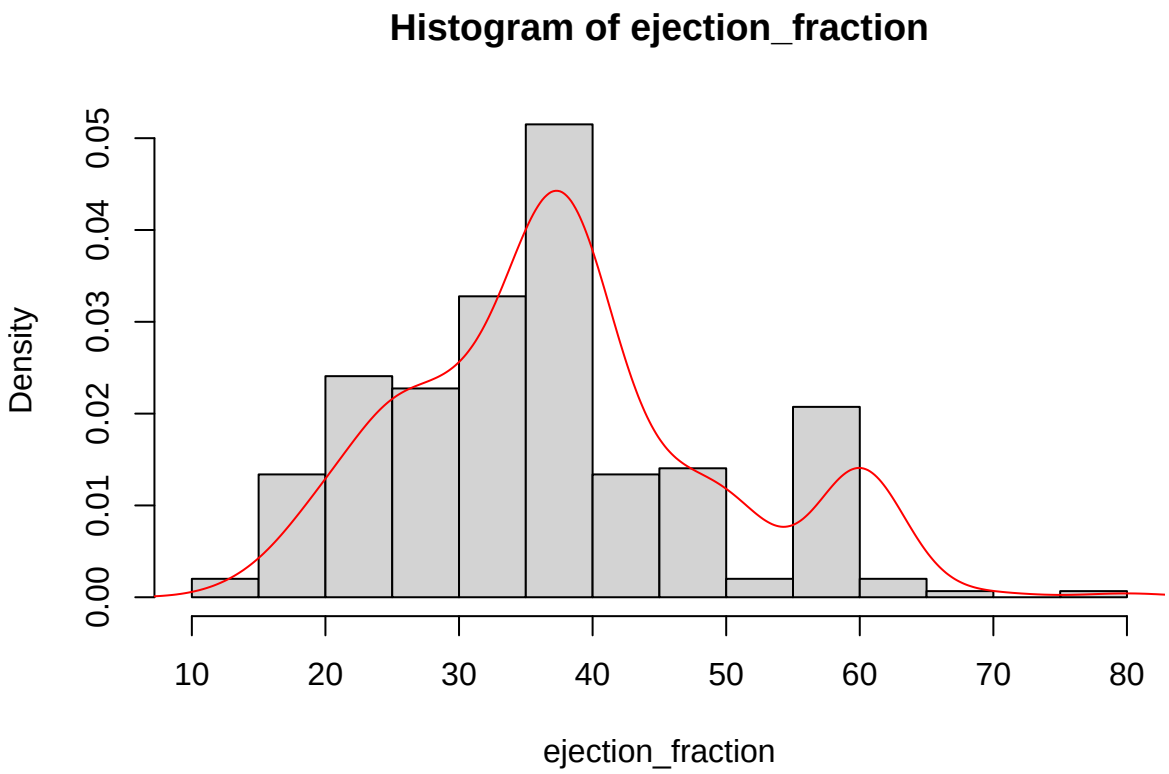
As skewness is positive, hence our prediction was correct.

```r
boxplot(heart.attk$ejection_fraction, main = "ejection_fraction")
points(mean(heart.attk$ejection_fraction))
```



In the above box plot We can observed that there is few outliers in this sample variable as we see some dots lie outside the upper whiskers of the boxplot, which means that there is some observations that is numerically distant from the rest of the data. Outliers are the data points located outside the whiskers of the boxplot. Now we can plot the histogram and look for the model that fits better the distribution

```
hist(heart.attk$ejection_fraction, prob = T, breaks = 17, freq=FALSE, xlab= "ejection_fraction", main =
lines(density(heart.attk$ejection_fraction), col="red")
```

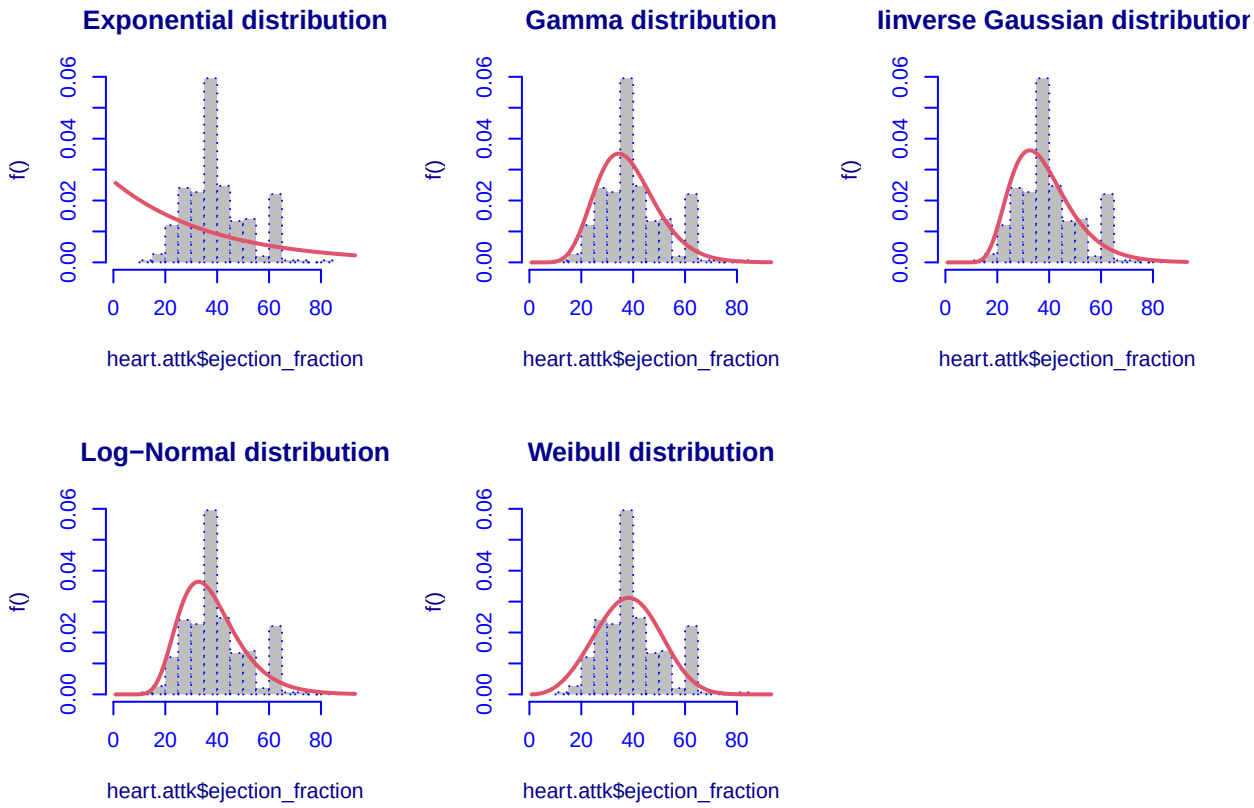


It pumps oxygen-rich blood up into our body's main artery (aorta) to the rest of the body. A normal ejection fraction is about 50% to 75%. So from above plotted figure we can see that most of the patients are lie below average range of ejection fraction.

Data fitting for ejection_fraction:

As per variable domain, the distributions supported on the set of positive infinite real numbers are fitted on the data. Log-likelihood value, BIC and AIC are estimated in order to evaluate the fit of the distribution.

```
library(gamlss.mx)
par(mfrow=c(2,3))
ejection_fraction.EXP  <-histDist(heart.attk$ejection_fraction, family=EXP, nbins = 13, main="Exponentia
ejection_fraction.GA   <-histDist(heart.attk$ejection_fraction, family=GA, nbins = 13, main="Gamma dist
ejection_fraction.IG   <-histDist(heart.attk$ejection_fraction, family=IG, nbins = 13, main="Iinverse G
ejection_fraction.LOGNO<-histDist(heart.attk$ejection_fraction, family=LOGNO, nbins = 13, main="Log-Nor
ejection_fraction.WEI  <-histDist(heart.attk$ejection_fraction, family=WEI, nbins = 13, main="Weibull d
```




```
ejection_fraction.matrix<-matrix(c(ejection_fraction.LOGNO$df.fit, logLik(ejection_fraction.LOGNO), AIC
ejection_fraction.LOGNO$sbc, ejection_fraction.GA$df.fit, logLik(ejection_fraction.GA), AIC(ejection_fra
ejection_fraction.GA$sbc, ejection_fraction.WEI$df.fit, logLik(ejection_fraction.WEI),
AIC(ejection_fraction.WEI), ejection_fraction.WEI$sbc,
ejection_fraction.EXP$df.fit, logLik(ejection_fraction.EXP), AIC(ejection_fraction.EXP), ejection_fracti
ejection_fraction.IG$df.fit, logLik(ejection_fraction.IG), AIC(ejection_fraction.IG), ejection_fraction
colnames(ejection_fraction.matrix)<-c("df","LogLik", "AIC", "BIC")
rownames(ejection_fraction.matrix)<-c("LOGNO", "GA", "WEI", "EXP", "IG")
ejection_fraction.matrix<-as.table(ejection_fraction.matrix)
ejection_fraction.matrix
```

```
##              df    LogLik       AIC       BIC
## LOGNO     2.000 -1154.918  2313.836  2321.237
## GA        2.000 -1152.920  2309.840  2317.241
## WEI       2.000 -1163.278  2330.557  2337.957
## EXP       1.000 -1387.295  2776.591  2780.291
## IG        2.000 -1155.097  2314.194  2321.595
```

After above statistical analysis we can observed that the model which has large value of log likelihood and small value in AIC and BIC is “Gamma distribution”, hence on the basis of likelihood method Our data is more likely to fit into gamma distribution.

Likelihood ratio test:

The goodness-of-fit test was performed between the Exponential model (under the null hypothesis) and the gamma model (under the alternative hypothesis).

```
library(lmtest)
lrtest(ejection_fraction.GA, ejection_fraction.EXP)
```

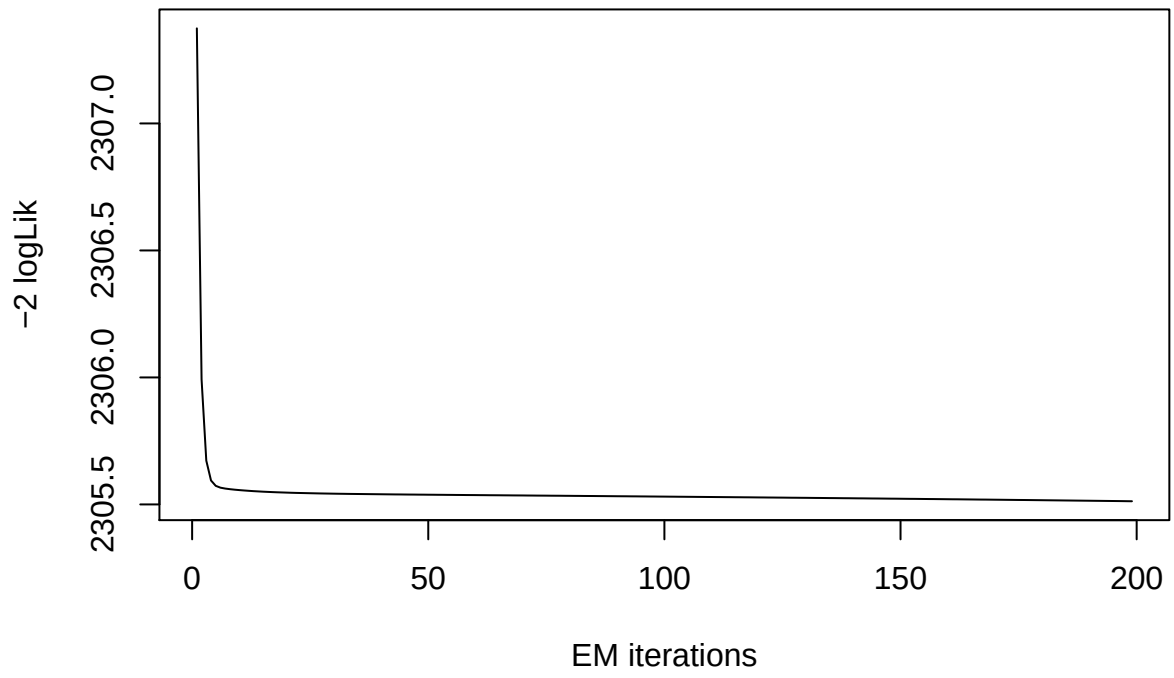
```
## Likelihood ratio test
##
## Model 1: gamlssML(formula = heart.attk$ejection_fraction, family = "GA")
## Model 2: gamlssML(formula = heart.attk$ejection_fraction, family = "EXP")
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   2 -1152.9
## 2   1 -1387.3 -1 468.75  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The null hypothesis would be rejected at nearly every significance level. Thus, we know that we should definitely use the gamma model as it increases the accuracy of our model by a substantial amount in terms of both AIC and BIC.

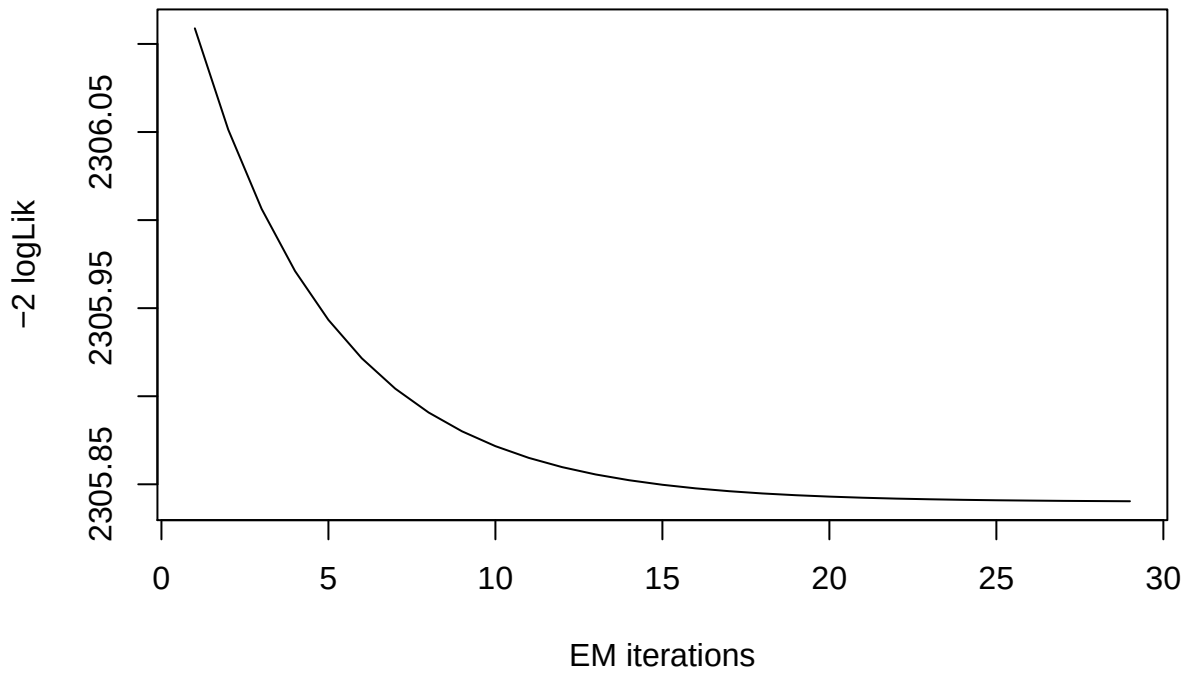
Mixture of distributions:

It is possible to compute a mixture of two gamma distributions In order to find the best mixture,the algorithm is repeated five times.

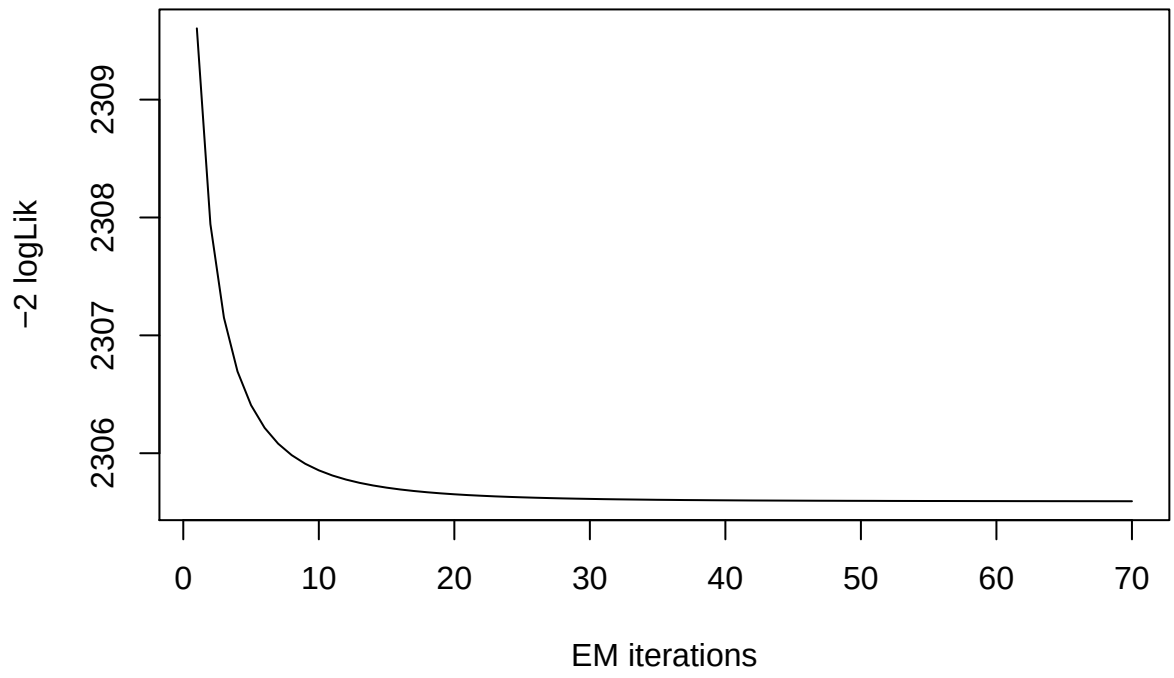
```
mxfit.ejection_fraction <- gamlssMXfits(n = 5, heart.attk$ejection_fraction~1, family = GA, K = 2, data
```


−2 logLik
2305.5
2306.0
2306.5
2307.0
0
50
100
150
200
EM iterations

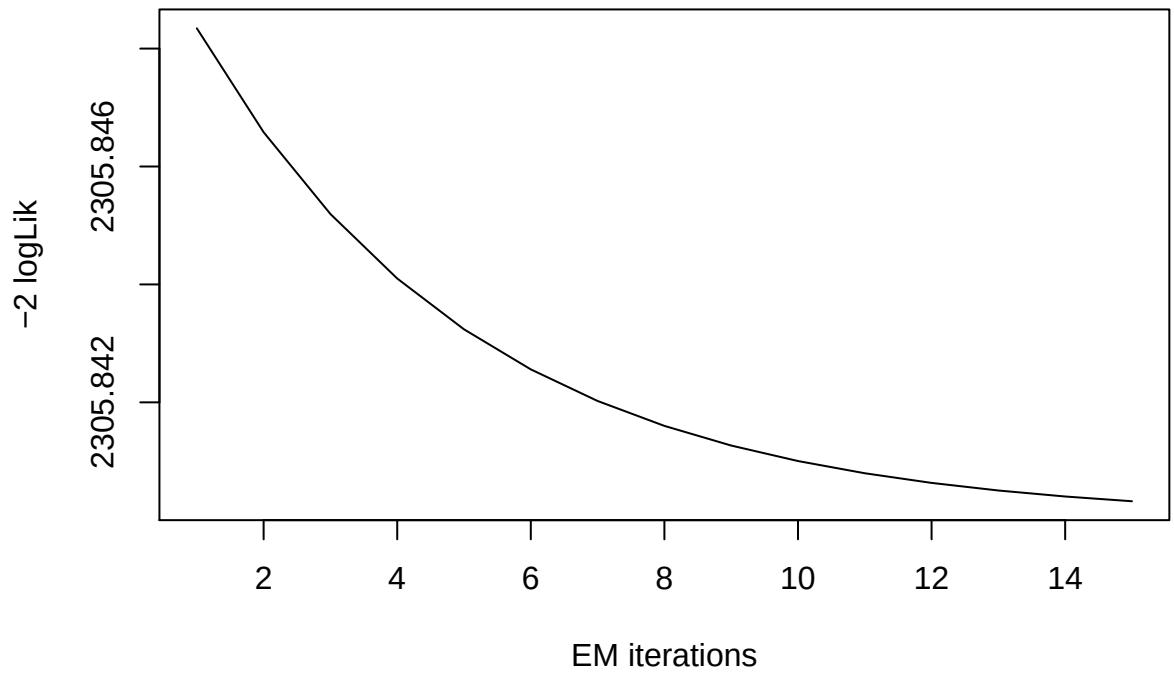
```
## model= 1
```



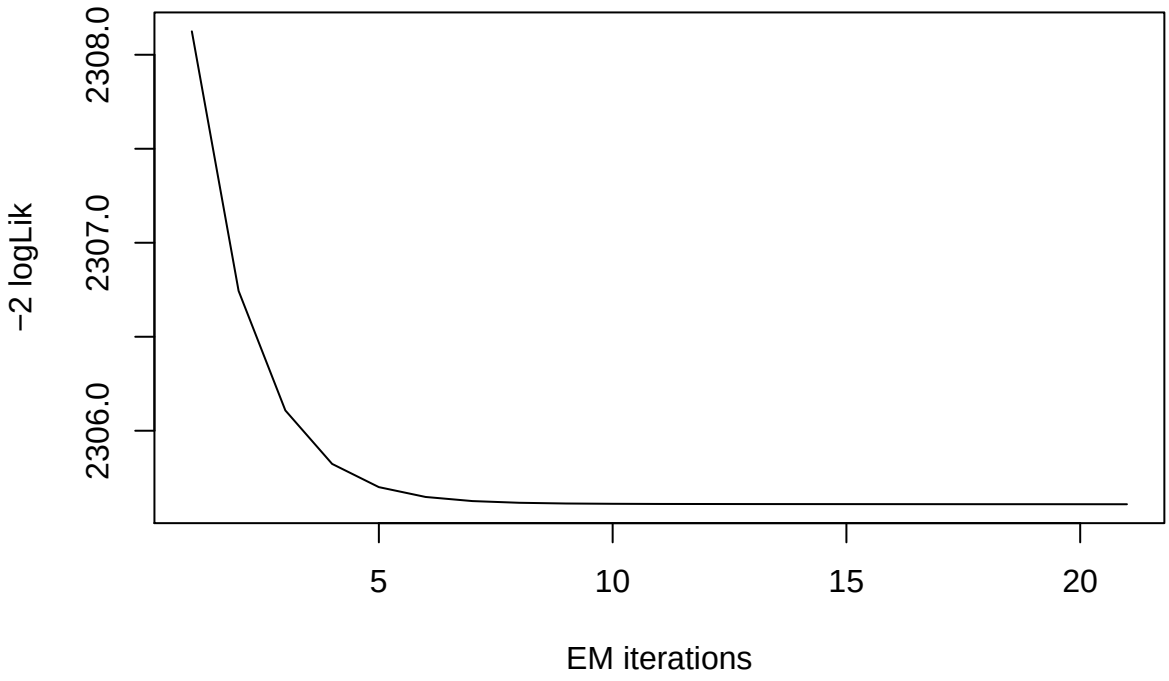

```
## model= 2
```


−2 logLik
2306
2307
2308
2309
0
10
20
30
40
50
60
70
EM iterations

```
## model= 3
```




```
## model= 4
```




```
## model= 5
```

```
print(mxfit.ejection_fraction)
```

```
##
## Mixing Family:  c("GA", "GA")
##
## Fitting method: EM algorithm
##
## Call:  gamlssMX(formula = heart.attk$ejection_fraction ~ 1,
##     family = GA, K = 2, data = heart.attk)
##
## Mu Coefficients for model: 1
## (Intercept)
##       3.835
## Sigma Coefficients for model: 1
## (Intercept)
##      -1.386
## Mu Coefficients for model: 2
## (Intercept)
##       3.526
## Sigma Coefficients for model: 2
## (Intercept)
##      -1.255
##
```

```
## Estimated probabilities: 0.3335583 0.6664417
##
## Degrees of Freedom for the fit: 5 Residual Deg. of Freedom   294
## Global Deviance:     2305.51
##             AIC:     2315.51
##             SBC:     2334.01
```

Here we can observed that the AIC and BIC value has improved with the combination of two Gamma distributions, as it is lower than the one obtained with a single gamma distribution.

```
logLik(mxfit.ejection_fraction)
```

```
## 'log Lik.' -1152.756 (df=5)
```

```
mxfit.ejection_fraction$prob
```

```
## [1] 0.3335583 0.6664417
```

```
fitted(mxfit.ejection_fraction, "mu")[1]
```

```
## [1] 38.08066
```

```
fitted(mxfit.ejection_fraction, "mu")[1]
```

```
## [1] 38.08066
```

```
hist(heart.attk$ejection_fraction, breaks = 100,freq = FALSE)
mu.hat1.ejection_fraction <- exp(mxfit.ejection_fraction[["models"]][[1]][["mu.coefficients"]])
sigma.hat1.ejection_fraction <- exp(mxfit.ejection_fraction[["models"]][[1]][["sigma.coefficients"]])
mu.hat2.ejection_fraction <- exp(mxfit.ejection_fraction[["models"]][[2]][["mu.coefficients"]])
sigma.hat2.ejection_fraction <- exp(mxfit.ejection_fraction[["models"]][[2]][["sigma.coefficients"]])
hist(heart.attk$ejection_fraction, breaks = 100, freq = FALSE, xlab = "ejection_fraction" ,main="ejecti
```

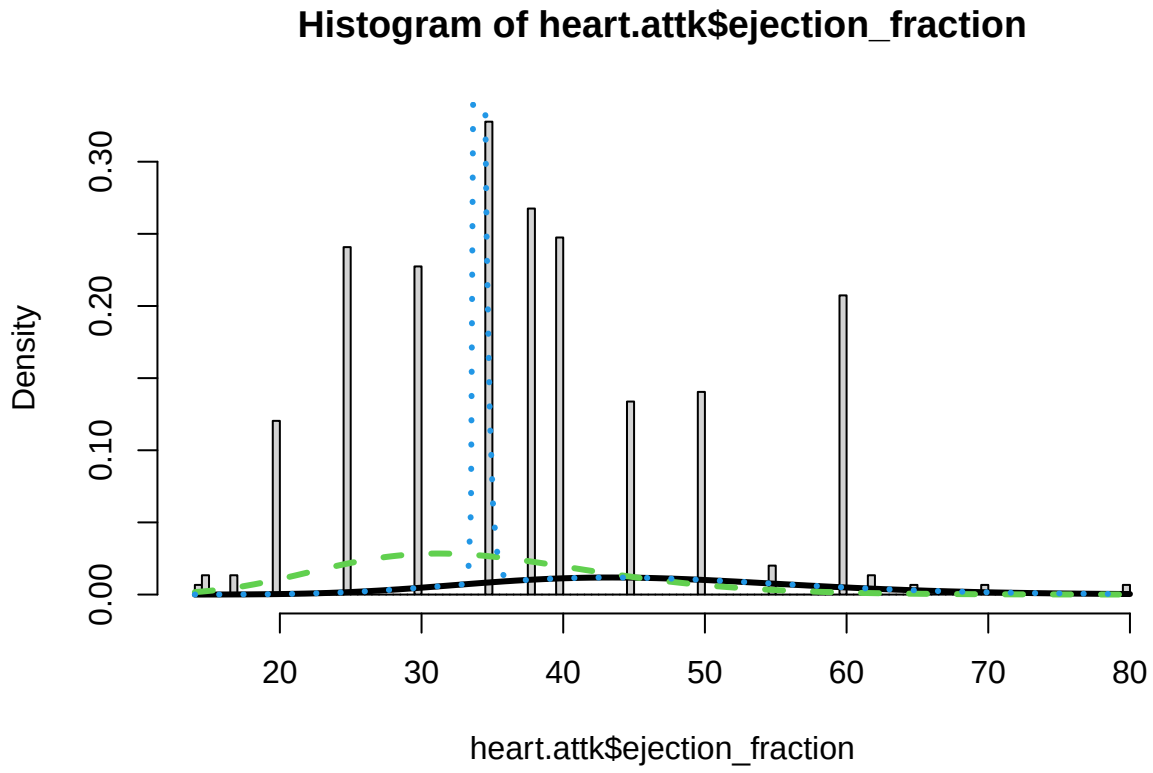
```
## Warning in hist.default(heart.attk$ejection_fraction, breaks = 100, freq =
## FALSE, : arguments 'freq', 'main', 'xlab' are not made use of
```

```
## $breaks
##   [1] 14.0 14.5 15.0 15.5 16.0 16.5 17.0 17.5 18.0 18.5 19.0 19.5 20.0 20.5 21.0
##  [16] 21.5 22.0 22.5 23.0 23.5 24.0 24.5 25.0 25.5 26.0 26.5 27.0 27.5 28.0 28.5
##  [31] 29.0 29.5 30.0 30.5 31.0 31.5 32.0 32.5 33.0 33.5 34.0 34.5 35.0 35.5 36.0
##  [46] 36.5 37.0 37.5 38.0 38.5 39.0 39.5 40.0 40.5 41.0 41.5 42.0 42.5 43.0 43.5
##  [61] 44.0 44.5 45.0 45.5 46.0 46.5 47.0 47.5 48.0 48.5 49.0 49.5 50.0 50.5 51.0
##  [76] 51.5 52.0 52.5 53.0 53.5 54.0 54.5 55.0 55.5 56.0 56.5 57.0 57.5 58.0 58.5
##  [91] 59.0 59.5 60.0 60.5 61.0 61.5 62.0 62.5 63.0 63.5 64.0 64.5 65.0 65.5 66.0
## [106] 66.5 67.0 67.5 68.0 68.5 69.0 69.5 70.0 70.5 71.0 71.5 72.0 72.5 73.0 73.5
## [121] 74.0 74.5 75.0 75.5 76.0 76.5 77.0 77.5 78.0 78.5 79.0 79.5 80.0
##
## $counts
##   [1]  1  2  0  0  0  2  0  0  0  0  0 18  0  0  0  0  0  0  0  0  0 36  0  0  0
##  [26]  0  0  0  0  0  0 34  0  0  0  0  0  0  0  0  0  0 49  0  0  0  0  0 40  0  0
```

```
##  [51]  0 37  0  0  0  0  0  0  0  0  0 20  0  0  0  0  0  0  0  0  0 21  0  0  0
##  [76]  0  0  0  0  0  0  3  0  0  0  0  0  0  0  0  0 31  0  0  0  2  0  0  0  0
## [101]  0  1  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0
## [126]  0  0  0  0  0  0  1
##
## $density
##   [1] 0.006688963 0.013377926 0.000000000 0.000000000 0.000000000 0.013377926
##   [7] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.120401338
##  [13] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
##  [19] 0.000000000 0.000000000 0.000000000 0.240802676 0.000000000 0.000000000
##  [25] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
##  [31] 0.000000000 0.227424749 0.000000000 0.000000000 0.000000000 0.000000000
##  [37] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.327759197
##  [43] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.267558528
##  [49] 0.000000000 0.000000000 0.000000000 0.247491639 0.000000000 0.000000000
##  [55] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
##  [61] 0.000000000 0.133779264 0.000000000 0.000000000 0.000000000 0.000000000
##  [67] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.140468227
##  [73] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
##  [79] 0.000000000 0.000000000 0.000000000 0.020066890 0.000000000 0.000000000
##  [85] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
##  [91] 0.000000000 0.207357860 0.000000000 0.000000000 0.000000000 0.013377926
##  [97] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.006688963
## [103] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
## [109] 0.000000000 0.000000000 0.000000000 0.006688963 0.000000000 0.000000000
## [115] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
## [121] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
## [127] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.006688963
##
## $mids
##   [1] 14.25 14.75 15.25 15.75 16.25 16.75 17.25 17.75 18.25 18.75 19.25 19.75
##  [13] 20.25 20.75 21.25 21.75 22.25 22.75 23.25 23.75 24.25 24.75 25.25 25.75
##  [25] 26.25 26.75 27.25 27.75 28.25 28.75 29.25 29.75 30.25 30.75 31.25 31.75
##  [37] 32.25 32.75 33.25 33.75 34.25 34.75 35.25 35.75 36.25 36.75 37.25 37.75
##  [49] 38.25 38.75 39.25 39.75 40.25 40.75 41.25 41.75 42.25 42.75 43.25 43.75
##  [61] 44.25 44.75 45.25 45.75 46.25 46.75 47.25 47.75 48.25 48.75 49.25 49.75
##  [73] 50.25 50.75 51.25 51.75 52.25 52.75 53.25 53.75 54.25 54.75 55.25 55.75
##  [85] 56.25 56.75 57.25 57.75 58.25 58.75 59.25 59.75 60.25 60.75 61.25 61.75
##  [97] 62.25 62.75 63.25 63.75 64.25 64.75 65.25 65.75 66.25 66.75 67.25 67.75
## [109] 68.25 68.75 69.25 69.75 70.25 70.75 71.25 71.75 72.25 72.75 73.25 73.75
## [121] 74.25 74.75 75.25 75.75 76.25 76.75 77.25 77.75 78.25 78.75 79.25 79.75
##
## $xname
## [1] "heart.attk$ejection_fraction"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

```
lines(seq(min(heart.attk$ejection_fraction),max(heart.attk$ejection_fraction),length=length(heart.attk$
      mxfit.ejection_fraction[["prob"]][1]*dGA(seq(min(heart.attk$ejection_fraction),max(heart.attk$ejec
```

```
lines(seq(min(heart.attk$ejection_fraction),max(heart.attk$ejection_fraction),length=length(heart.attk$
      mxfit.ejection_fraction[["prob"]][2]*dGA(seq(min(heart.attk$ejection_fraction),max(heart.attk$eje
length=length(heart.attk$ejection_fraction)),mu=mu.hat2.ejection_fraction,sigma = sigma.hat2.ejection_f

lines(seq(min(heart.attk$ejection_fraction),max(heart.attk$ejection_fraction),length=length(heart.attk$
      mxfit.ejection_fraction[["prob"]][1]*dGA(seq(min(heart.attk$ejection_fraction),max(heart.attk$eje
        mxfit.ejection_fraction[["prob"]][2]*dRG(seq(min(heart.attk$ejection_fraction),max(heart.attk$e
length=length(heart.attk$ejection_fraction)),mu= mu.hat2.ejection_fraction,sigma = sigma.hat2.ejection_
lty = 3, lwd = 3, col = 4)
```



Since we have selected K=2, we can see in the plot 3 lines, each corresponding to a distribution.The black line is relative to the first distribution, the dotted green one to the second distribution, while the dotted blue line is relative to the mixture, i.e. the overall model for all data.

platelets:

```
length(heart.attk$platelets)
```

```
## [1] 299
```

```
summary(heart.attk$platelets)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   25100  212500  262000  263358  303500  850000
```

The length of the variable is 299 and range is 25100 to 850000.

```
sd(heart.attk$platelets)
```

```
## [1] 97804.24
```

```
var(heart.attk$platelets)
```

```
## [1] 9565668749
```

```
getmode <- function(v) {
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}

# Create the vector with numbers.
v <- c(heart.attk$platelets)

# Calculate the mode using the user function.
result <- getmode(v)
print(result)
```

```
## [1] 263358
```

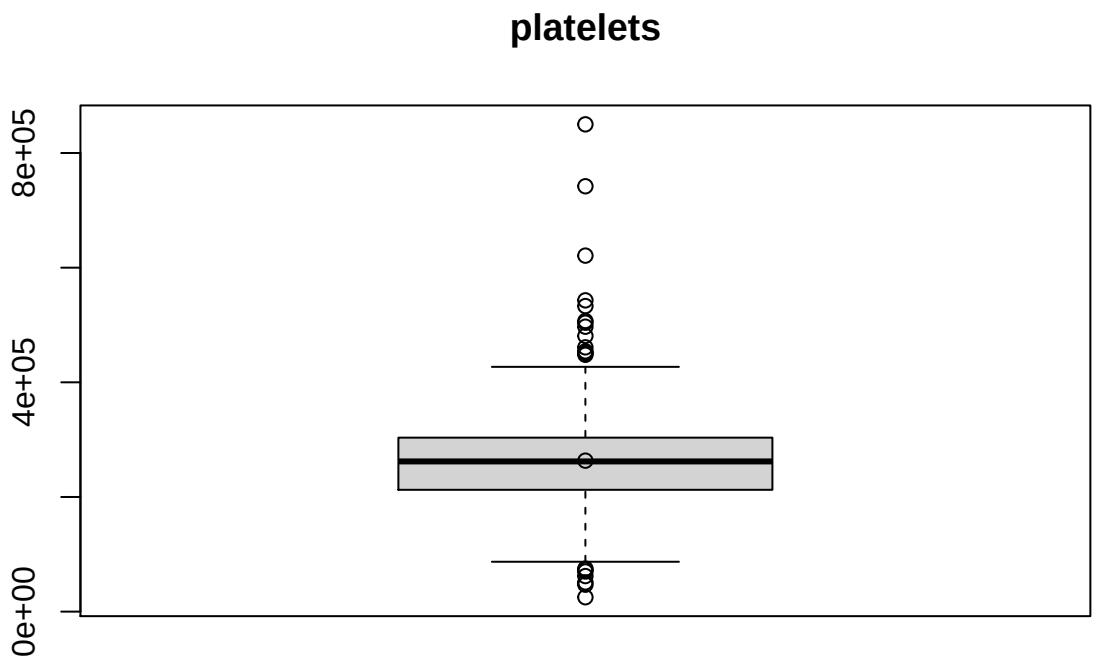
From above statistical computation we can observed that the mean and median are not identical, therefore the distribution is asymmetrical and skewed. And as mean is greater than median, hence the distribution should be skewed to the left and positively skewed.

```
skewness(heart.attk$platelets)
```

```
## [1] 1.447681
```

As skewness is positive, hence our prediction was correct.

```
boxplot(heart.attk$platelets, main = "platelets")
points(mean(heart.attk$platelets))
```

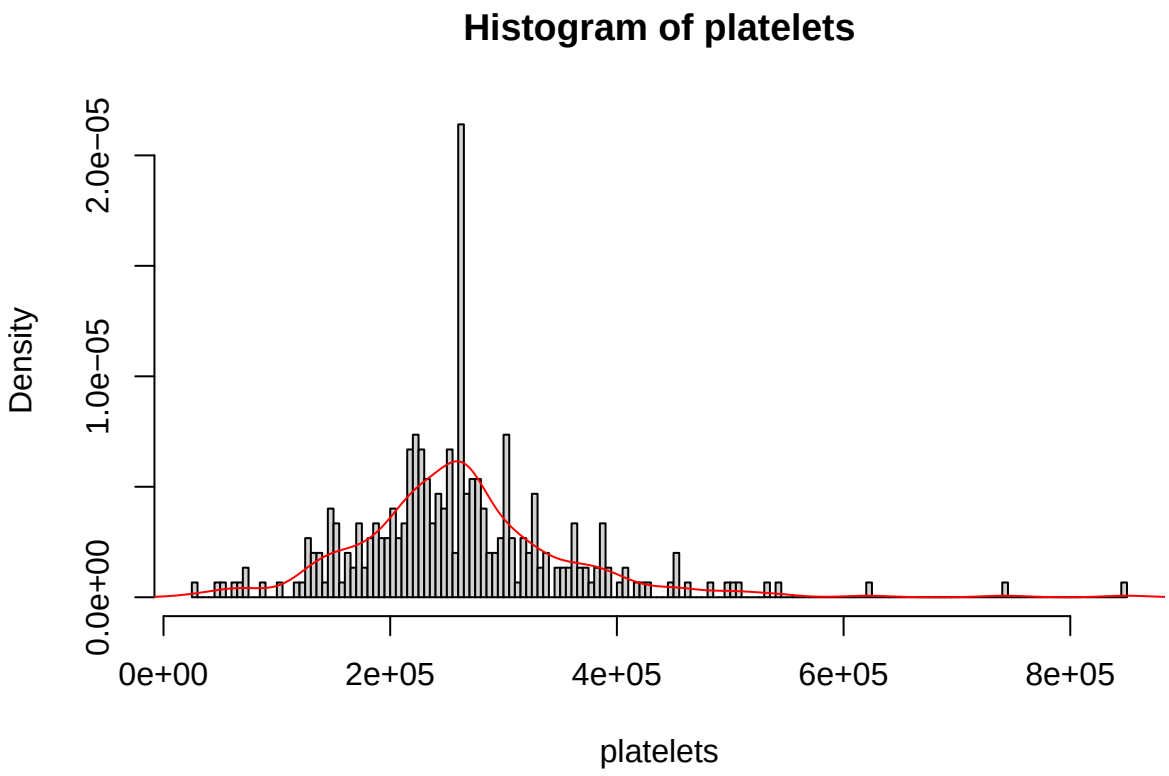


While reviewing the box plot We can observed that there is outliers in this sample variable as we see some dots lie outside the upper and lower whiskers of the boxplot, which means that there is some observations that is numerically distant from the rest of the data. Outliers are the data points located outside the whiskers of the boxplot. Now we can plot the histogram and look for the model that fits better the distribution

```
length(unique(heart.attk$platelets))
```

```
## [1] 176
```

```
hist(heart.attk$platelets, prob = T, breaks = 176, freq=FALSE, xlab= "platelets", main = "Histogram of
lines(density(heart.attk$platelets), col="red")
```

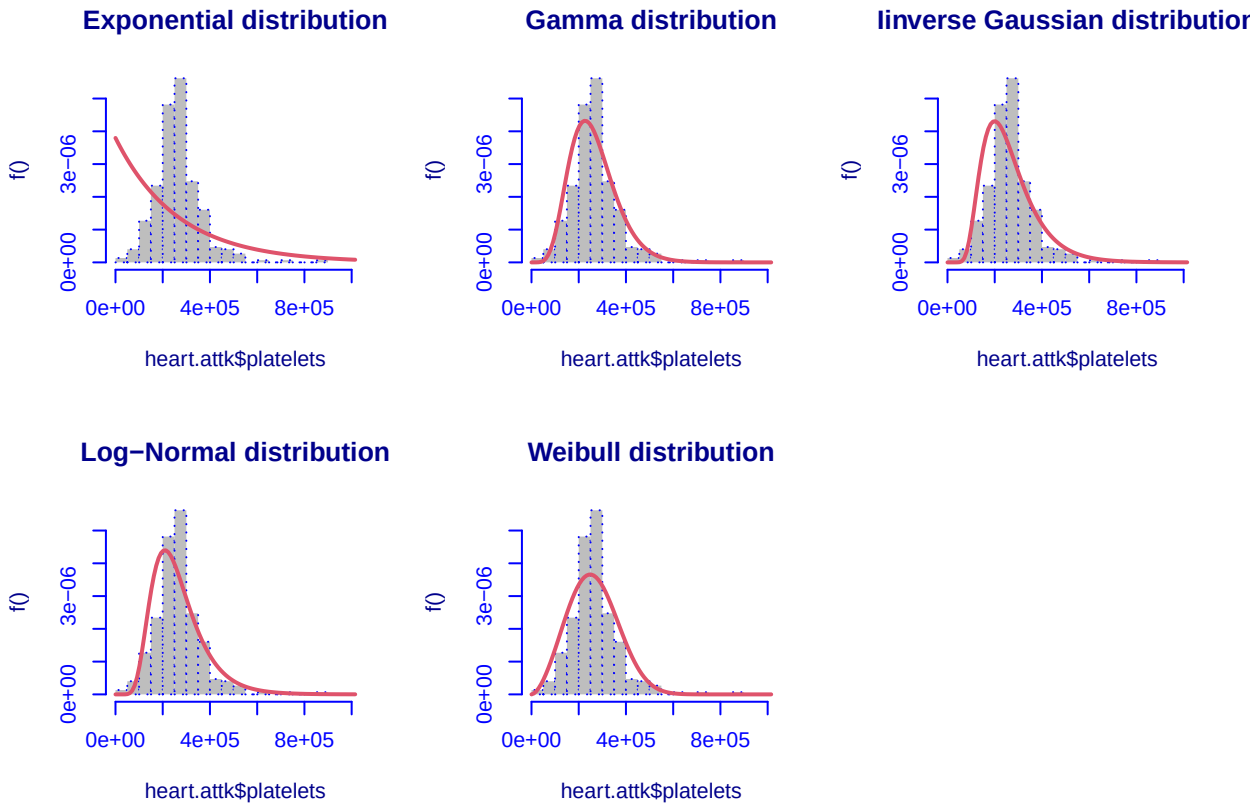


A normal platelet count ranges from 150,000 to 450,000 platelets per microliter of blood. Having more than 450,000 platelets is a condition called thrombocytosis; having less than 150,000 is known as thrombocytopenia. So from above plotted figure we can say that most of the patients lie within the normal range.

Data fitting for platelets:

As per variable domain, the distributions supported on the set of positive infinite real numbers are fitted on the data. Log-likelihood value, BIC and AIC are estimated in order to evaluate the fit of the distribution.

```
library(gamlss.mx)
par(mfrow=c(2,3))
platelets.EXP  <-histDist(heart.attk$platelets, family=EXP, nbins = 13, main="Exponential distribution"
platelets.GA   <-histDist(heart.attk$platelets, family=GA, nbins = 13, main="Gamma distribution")
platelets.IG   <-histDist(heart.attk$platelets, family=IG, nbins = 13, main="Iinverse Gaussian distribu
platelets.LOGNO<-histDist(heart.attk$platelets, family=LOGNO, nbins = 13, main="Log-Normal distribution
platelets.WEI  <-histDist(heart.attk$platelets, family=WEI, nbins = 13, main="Weibull distribution")
```




```
platelets.matrix<-matrix(c(platelets.LOGNO$df.fit, logLik(platelets.LOGNO), AIC(platelets.LOGNO),
platelets.LOGNO$sbc, platelets.GA$df.fit, logLik(platelets.GA), AIC(platelets.GA),
platelets.GA$sbc, platelets.WEI$df.fit, logLik(platelets.WEI),
AIC(platelets.WEI), platelets.WEI$sbc,
platelets.EXP$df.fit, logLik(platelets.EXP), AIC(platelets.EXP), platelets.EXP$sbc,
platelets.IG$df.fit, logLik(platelets.IG), AIC(platelets.IG), platelets.IG$sbc), ncol=4, byrow=TRUE)
colnames(platelets.matrix)<-c("df","LogLik", "AIC", "BIC")
rownames(platelets.matrix)<-c("LOGNO", "GA", "WEI", "EXP", "IG")
platelets.matrix<-as.table(platelets.matrix)
platelets.matrix
```

```
##              df    LogLik       AIC       BIC
## LOGNO     2.000 -3861.283  7726.566  7733.966
## GA        2.000 -3846.612  7697.224  7704.625
## WEI       2.000 -3859.715  7723.429  7730.830
## EXP       1.000 -4030.900  8063.799  8067.500
## IG        2.000 -3874.506  7753.012  7760.413
```

After above statistical analysis we can observed that the model which has large value of log likelihood and small value in AIC and BIC is “Gamma distribution”, hence on the basis of likelihood method Our data is more likely to fit into gamma distribution.

Likelihood ratio test:

The goodness-of-fit test was performed between the Exponential model (under the null hypothesis) and the gamma model (under the alternative hypothesis).

```
library(lmtest)
lrtest(platelets.GA, platelets.EXP)
```

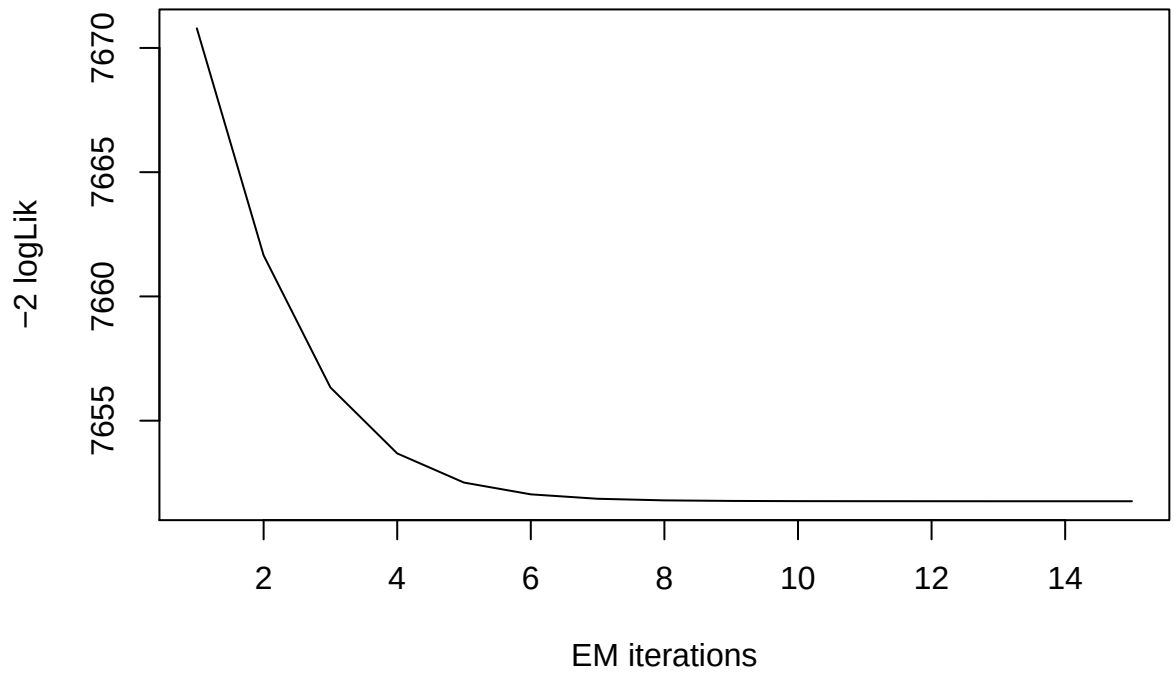
```
## Likelihood ratio test
##
## Model 1: gamlssML(formula = heart.attk$platelets, family = "GA")
## Model 2: gamlssML(formula = heart.attk$platelets, family = "EXP")
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   2 -3846.6
## 2   1 -4030.9 -1 368.58  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The null hypothesis would be rejected at nearly every significance level. Thus, we know that we should definitely use the gamma model as it increases the accuracy of our model by a substantial amount in terms of both AIC and BIC.

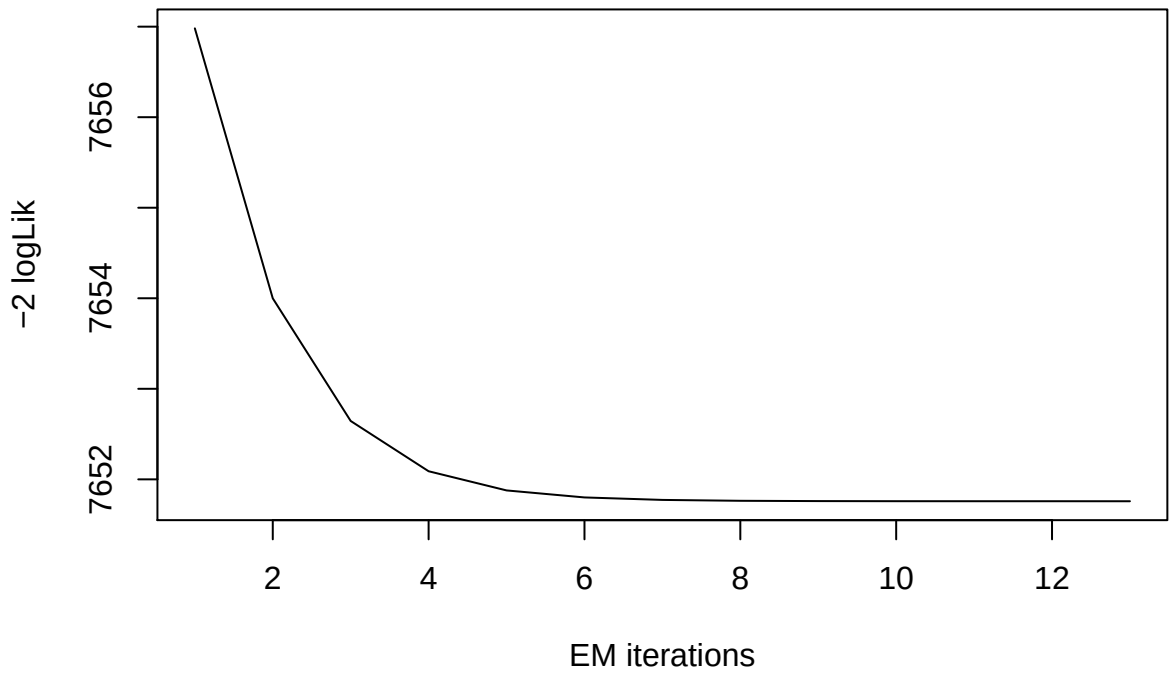
Mixture of distributions:

It is possible to compute a mixture of two gamma distributions In order to find the best mixture,the algorithm is repeated five times.

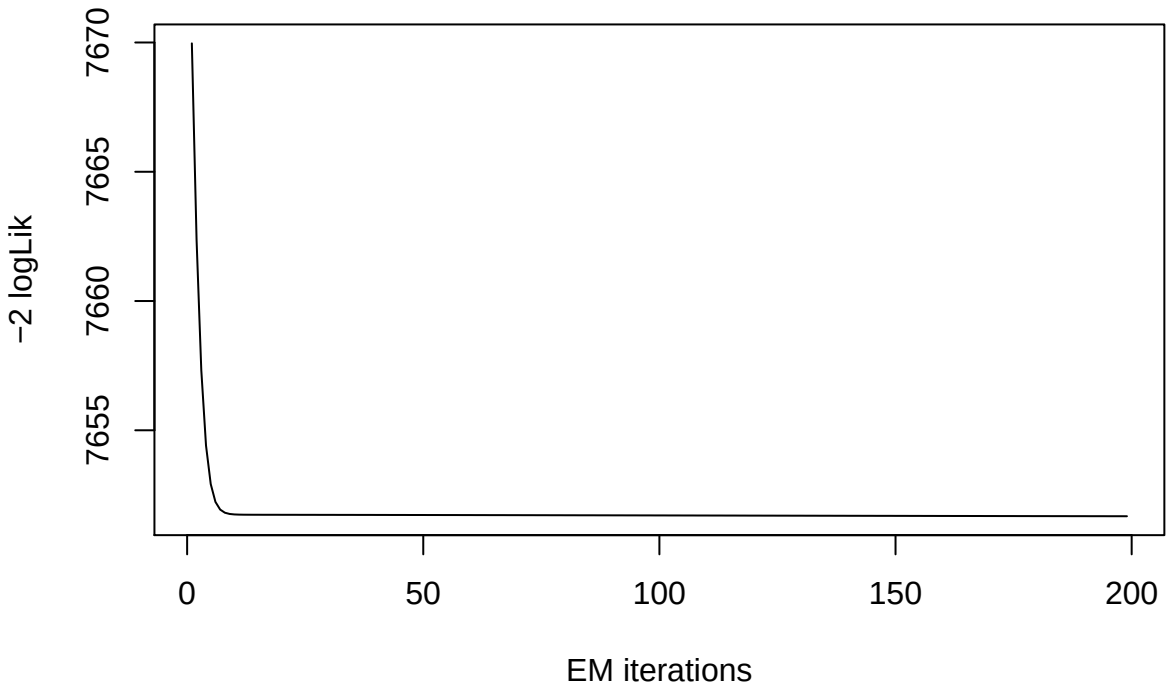
```
mxfit.platelets <- gamlssMXfits(n = 5, heart.attk$platelets~1, family = GA, K = 2, data = heart.attk)
```



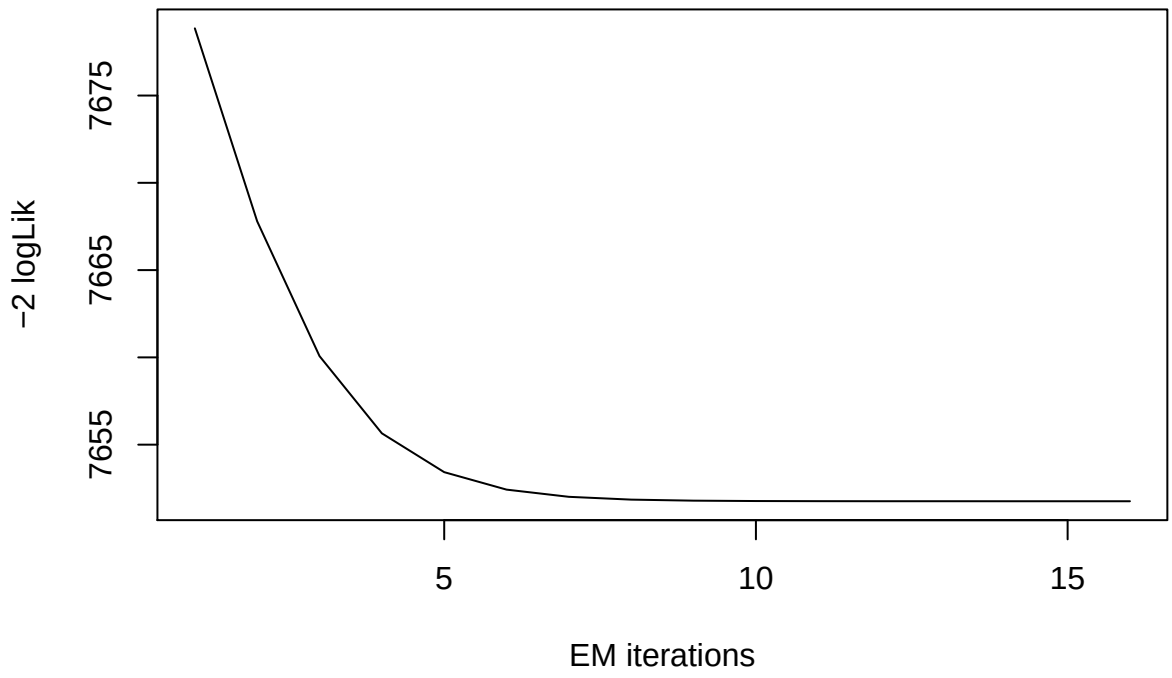

```
## model= 1
```



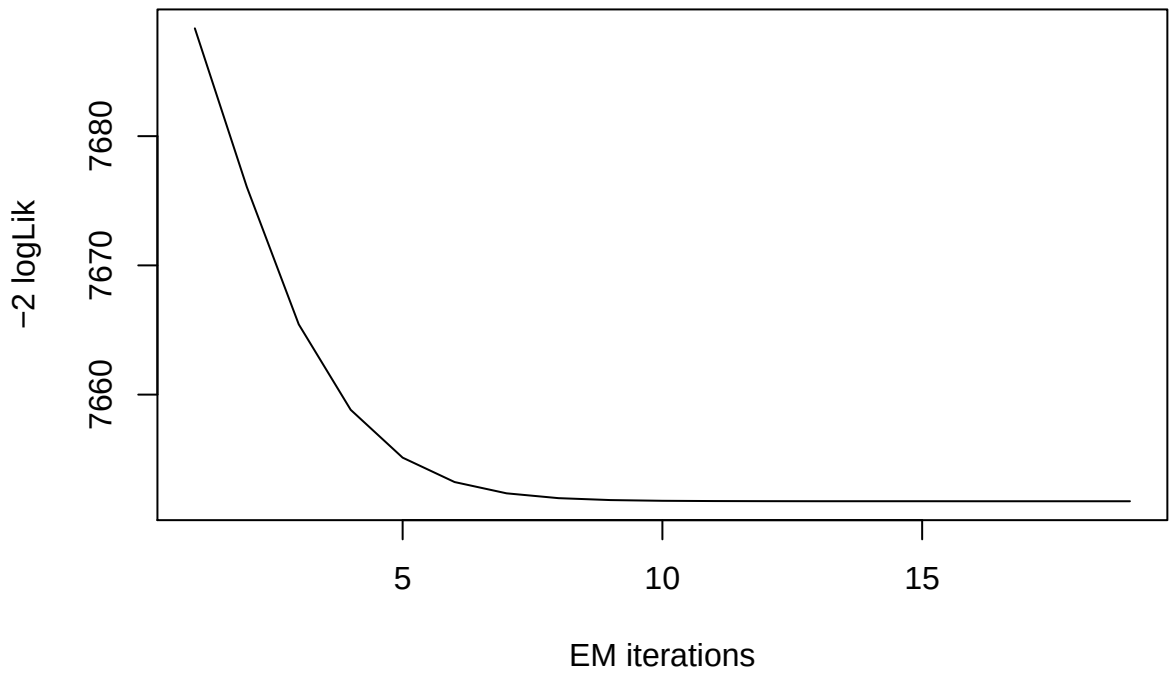

```
## model= 2
```




```
## model= 3
```




```
## model= 4
```




```
## model= 5
```

```
print(mxfit.platelets)
```

```
##
## Mixing Family:  c("GA", "GA")
##
## Fitting method: EM algorithm
##
## Call:  gamlssMX(formula = heart.attk$platelets ~ 1, family = GA,
##     K = 2, data = heart.attk)
##
## Mu Coefficients for model: 1
## (Intercept)
##       12.49
## Sigma Coefficients for model: 1
## (Intercept)
##     -0.7118
## Mu Coefficients for model: 2
## (Intercept)
##       12.47
## Sigma Coefficients for model: 2
## (Intercept)
##      -1.801
##
```

```
## Estimated probabilities: 0.5130459 0.4869541
##
## Degrees of Freedom for the fit: 5 Residual Deg. of Freedom   294
## Global Deviance:     7651.67
##             AIC:     7661.67
##             SBC:     7680.18
```

Here we can observed that the AIC and BIC value has improved with the combination of two Gamma distributions, as it is lower than the one obtained with a single gamma distribution.

```
logLik(mxfit.ejection_fraction)
```

```
## 'log Lik.' -1152.756 (df=5)
```

```
mxfit.platelets$prob
```

```
## [1] 0.5130459 0.4869541
```

```
fitted(mxfit.platelets, "mu")[1]
```

```
## [1] 263359.6
```

```
fitted(mxfit.platelets, "sigma")[2]
```

```
## [1] 263359.6
```

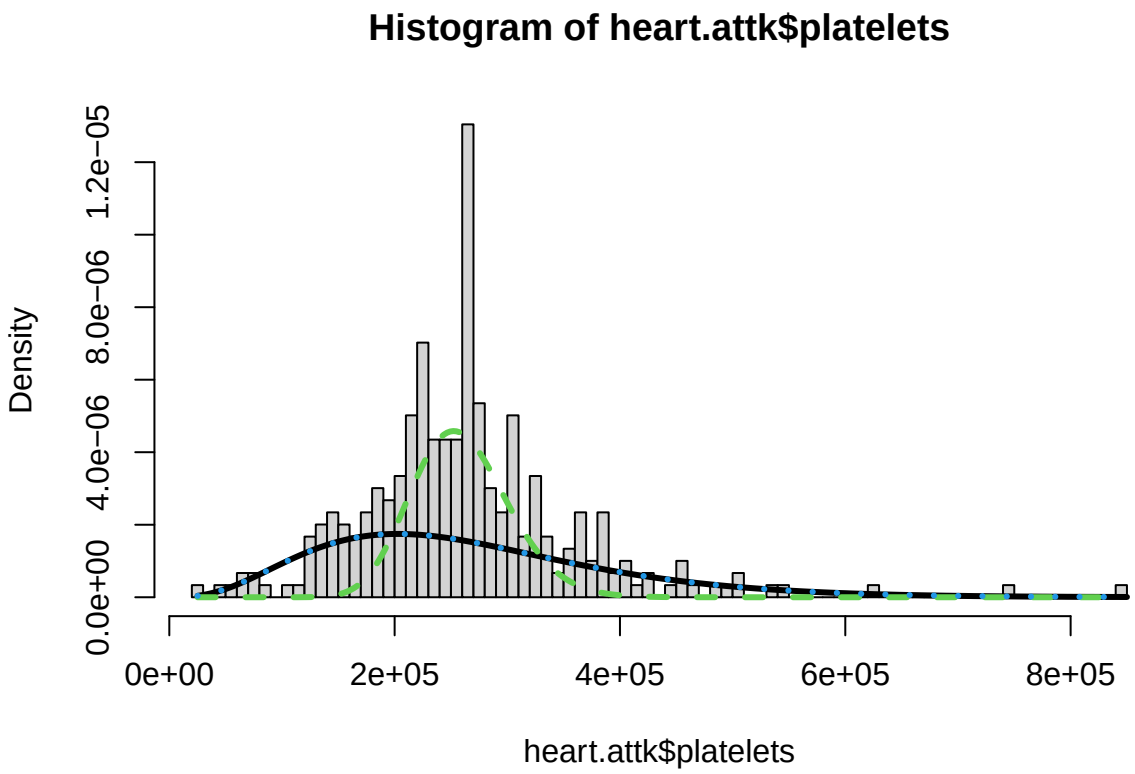
```
hist(heart.attk$platelets, breaks = 100,freq = FALSE)
mu.hat1.platelets <- exp(mxfit.platelets[["models"]][[1]][["mu.coefficients"]])
sigma.hat1.platelets <- exp(mxfit.platelets[["models"]][[1]][["sigma.coefficients"]])
mu.hat2.platelets <- exp(mxfit.platelets[["models"]][[2]][["mu.coefficients"]])
sigma.hat2.platelets <- exp(mxfit.platelets[["models"]][[2]][["sigma.coefficients"]])
hist(heart.attk$platelets, breaks = 100, freq = FALSE, xlab = "platelets" ,main="platelets-Mixture of tw
```

```
## Warning in hist.default(heart.attk$platelets, breaks = 100, freq = FALSE, :
## arguments 'freq', 'main', 'xlab' are not made use of
```

```
## $breaks
##  [1]  20000  30000  40000  50000  60000  70000  80000  90000 100000 110000
## [11] 120000 130000 140000 150000 160000 170000 180000 190000 200000 210000
## [21] 220000 230000 240000 250000 260000 270000 280000 290000 300000 310000
## [31] 320000 330000 340000 350000 360000 370000 380000 390000 400000 410000
## [41] 420000 430000 440000 450000 460000 470000 480000 490000 500000 510000
## [51] 520000 530000 540000 550000 560000 570000 580000 590000 600000 610000
## [61] 620000 630000 640000 650000 660000 670000 680000 690000 700000 710000
## [71] 720000 730000 740000 750000 760000 770000 780000 790000 800000 810000
## [81] 820000 830000 840000 850000
##
## $counts
##  [1]  1  0  1  1  2  2  1  0  1  1  5  6  7  6  5  7  9  8 10 15 21 13 13 13 39
## [26] 16  9  7 15  5 10  5  2  4  7  3  7  2  3  1  2  0  1  3  1  0  1  1  2  0
```

```
## [51]  0  1  1  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  1  0  0
## [76]  0  0  0  0  0  0  0  1
##
## $density
##  [1] 3.344482e-07 0.000000e+00 3.344482e-07 3.344482e-07 6.688963e-07
##  [6] 6.688963e-07 3.344482e-07 0.000000e+00 3.344482e-07 3.344482e-07
## [11] 1.672241e-06 2.006689e-06 2.341137e-06 2.006689e-06 1.672241e-06
## [16] 2.341137e-06 3.010033e-06 2.675585e-06 3.344482e-06 5.016722e-06
## [21] 7.023411e-06 4.347826e-06 4.347826e-06 4.347826e-06 1.304348e-05
## [26] 5.351171e-06 3.010033e-06 2.341137e-06 5.016722e-06 1.672241e-06
## [31] 3.344482e-06 1.672241e-06 6.688963e-07 1.337793e-06 2.341137e-06
## [36] 1.003344e-06 2.341137e-06 6.688963e-07 1.003344e-06 3.344482e-07
## [41] 6.688963e-07 0.000000e+00 3.344482e-07 1.003344e-06 3.344482e-07
## [46] 0.000000e+00 3.344482e-07 3.344482e-07 6.688963e-07 0.000000e+00
## [51] 0.000000e+00 3.344482e-07 3.344482e-07 0.000000e+00 0.000000e+00
## [56] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
## [61] 3.344482e-07 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
## [66] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
## [71] 0.000000e+00 0.000000e+00 3.344482e-07 0.000000e+00 0.000000e+00
## [76] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
## [81] 0.000000e+00 0.000000e+00 3.344482e-07
##
## $mids
##  [1]  25000  35000  45000  55000  65000  75000  85000  95000 105000 115000
## [11] 125000 135000 145000 155000 165000 175000 185000 195000 205000 215000
## [21] 225000 235000 245000 255000 265000 275000 285000 295000 305000 315000
## [31] 325000 335000 345000 355000 365000 375000 385000 395000 405000 415000
## [41] 425000 435000 445000 455000 465000 475000 485000 495000 505000 515000
## [51] 525000 535000 545000 555000 565000 575000 585000 595000 605000 615000
## [61] 625000 635000 645000 655000 665000 675000 685000 695000 705000 715000
## [71] 725000 735000 745000 755000 765000 775000 785000 795000 805000 815000
## [81] 825000 835000 845000
##
## $xname
## [1] "heart.attk$platelets"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

```
lines(seq(min(heart.attk$platelets),max(heart.attk$platelets),length=length(heart.attk$platelets)),
      mxfit.platelets[["prob"]][1]*dGA(seq(min(heart.attk$platelets),max(heart.attk$platelets),

lines(seq(min(heart.attk$platelets),max(heart.attk$platelets),length=length(heart.attk$platelets)),
      mxfit.platelets[["prob"]][2]*dGA(seq(min(heart.attk$platelets),max(heart.attk$platelets),
length=length(heart.attk$platelets)),mu=mu.hat2.platelets,sigma = sigma.hat2.platelets),lty=2,lwd=3,col=

lines(seq(min(heart.attk$platelets),max(heart.attk$platelets),length=length(heart.attk$platelets)),
      mxfit.platelets[["prob"]][1]*dGA(seq(min(heart.attk$platelets),max(heart.attk$platelets),
        mxfit.platelets[["prob"]][2]*dRG(seq(min(heart.attk$platelets),max(heart.attk$platelets),
length=length(heart.attk$platelets)),mu= mu.hat2.platelets,sigma = sigma.hat2.platelets),
lty = 3, lwd = 3, col = 4)
```

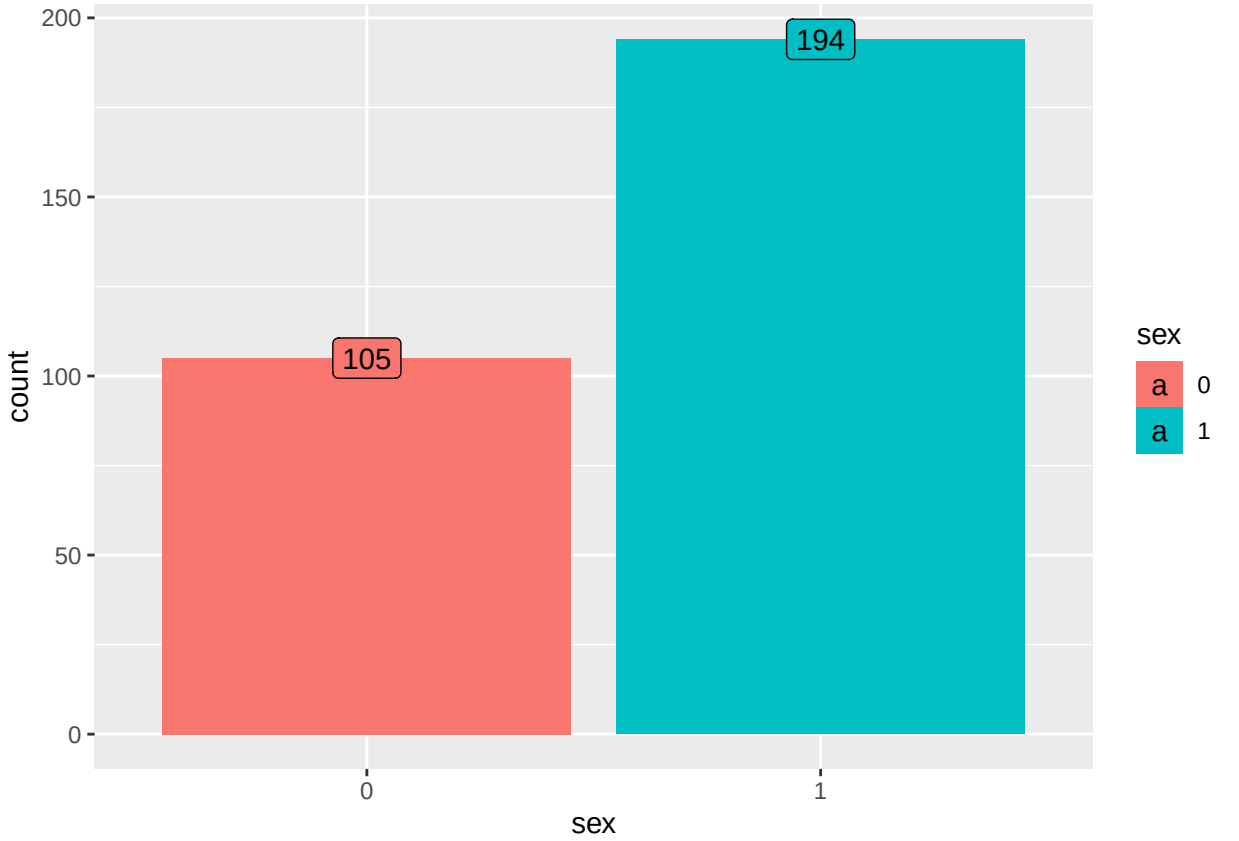


Since we have selected K=2, we can see in the plot 3 lines, each corresponding to a distribution.The black line is relative to the first distribution, the dotted green one to the second distribution, while the dotted blue line is relative to the mixture, i.e. the overall model for all data.

SEX:

Sex is a categorical variable that can take two values 1 for male and 0 for females.

```
library(ggplot2)
ggplot(heart.attk, aes(x = sex, fill=sex, )) + geom_bar()+
geom_label(stat = "count", aes(label = ..count..))
```



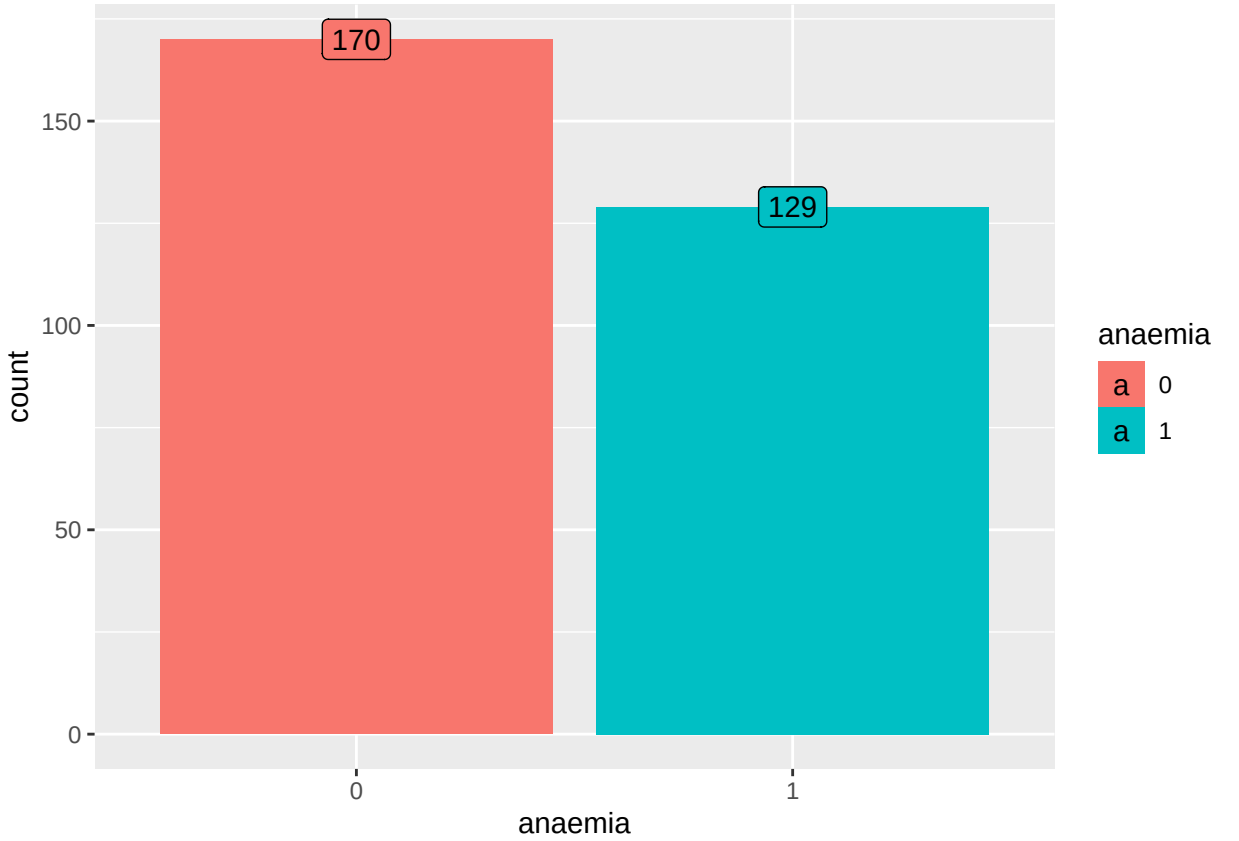
As a result of above statistical computation, we can see that our dataset contains 105 females and 194 males. So, based on the plot and summary, we can conclude that our data is unbalanced and the heart patients are more males than females.

Anaemia:

```
library(ggplot2)
summary(heart.attk$anaemia)
```

```
##   0   1
## 170 129
```

```
ggplot(heart.attk, aes(x = anaemia, fill=anaemia)) + geom_bar()+
geom_label(stat = "count", aes(label = ..count..))
```



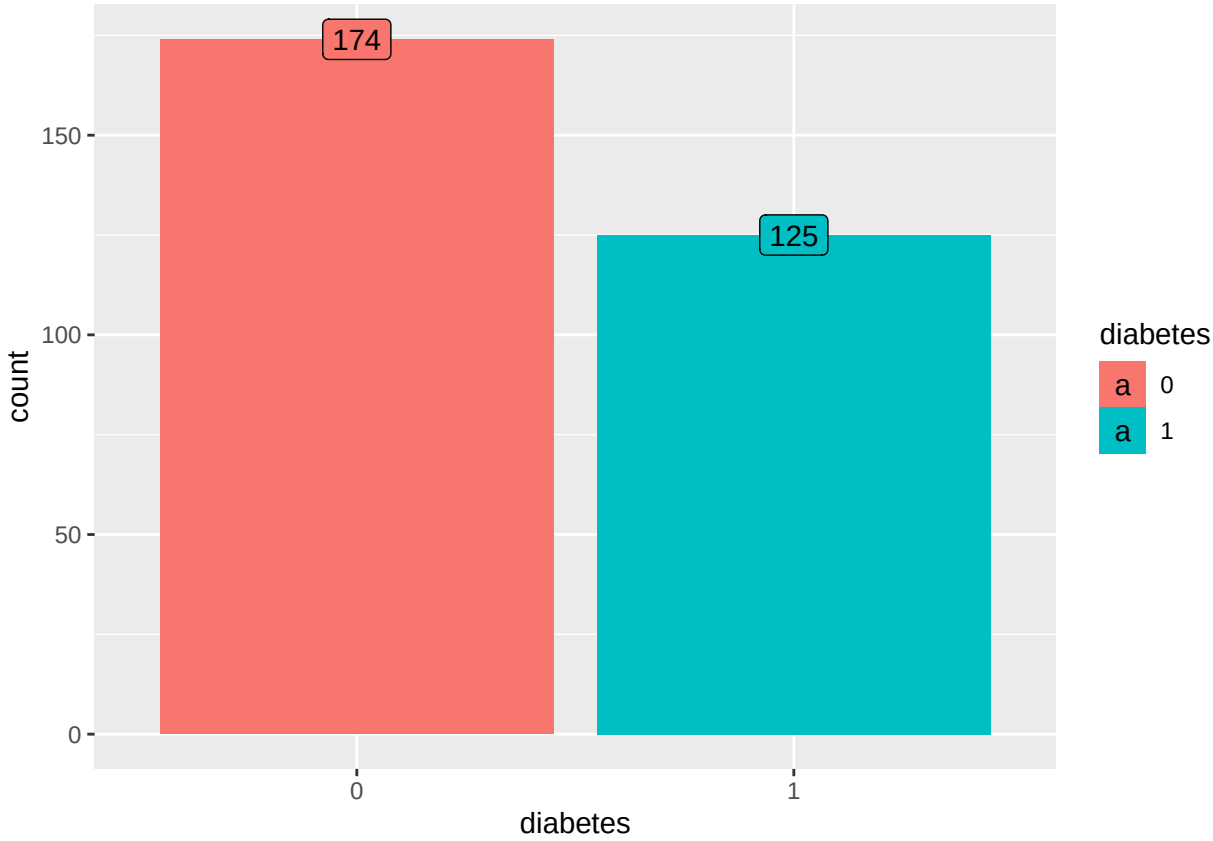
As a result of above plot and summary, we can conclude that our data is unbalanced and among 299 heart patients 129 patients have hemoglobin deficiency.

DIABETES:

```
library(ggplot2)
summary(heart.attk$diabetes)
```

```
##   0   1
## 174 125
```

```
ggplot(heart.attk, aes(x = diabetes, fill=diabetes)) + geom_bar()+
geom_label(stat = "count", aes(label = ..count..))
```



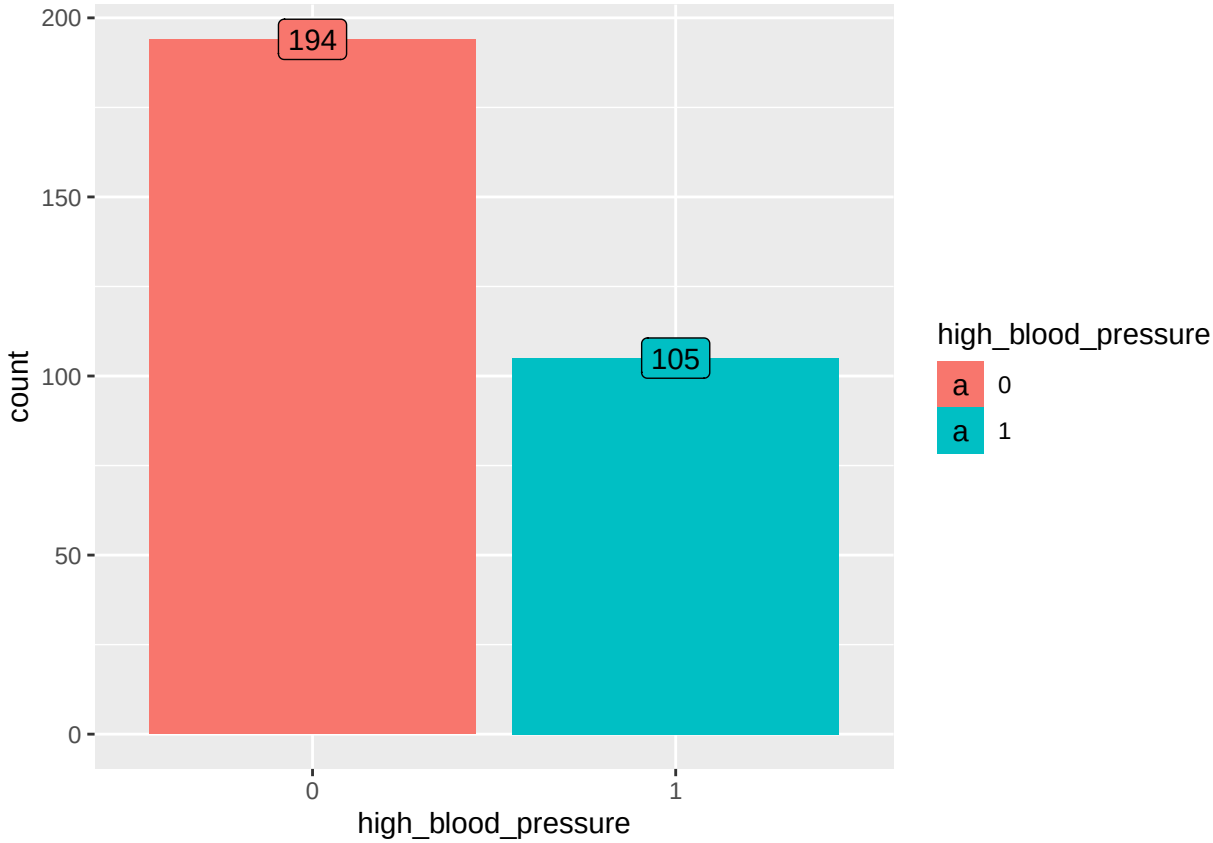
As a result of above plot and summary, we can conclude that our data is unbalanced and among 299 heart patients 125 patients have diabetes.

HIGH_BLOOD_PRESSURE:

```
library(ggplot2)
summary(heart.attk$high_blood_pressure)
```

```
##   0   1
## 194 105
```

```
ggplot(heart.attk, aes(x = high_blood_pressure, fill=high_blood_pressure)) + geom_bar()+ geom_label(sta
```



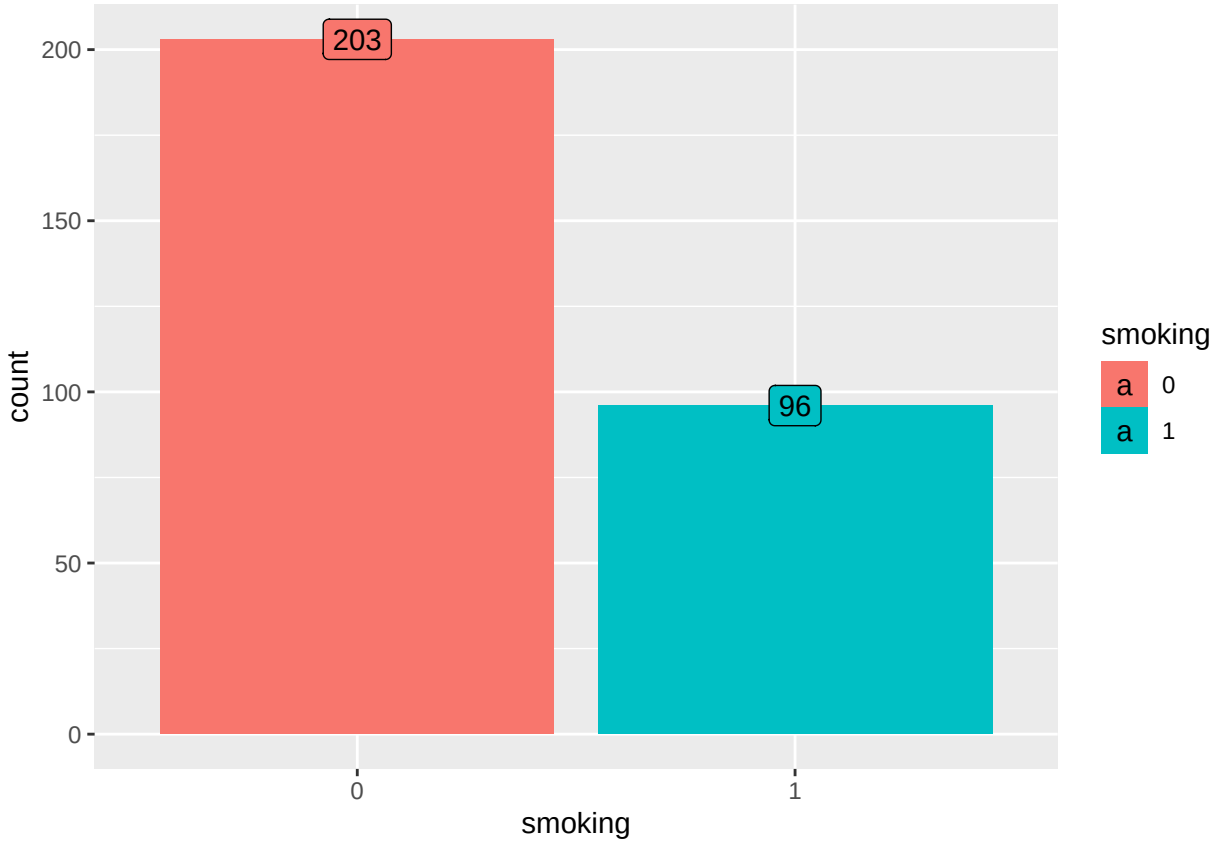
As a result of above plot and summary, we can conclude that our data is unbalanced and among 299 heart patients 105 patients have high blood pressure problem.

SMOKING:

```
library(ggplot2)
summary(heart.attk$smoking)
```

```
##   0   1
## 203  96
```

```
ggplot(heart.attk, aes(x = smoking, fill=smoking)) + geom_bar()+
geom_label(stat = "count", aes(label = ..count..))
```



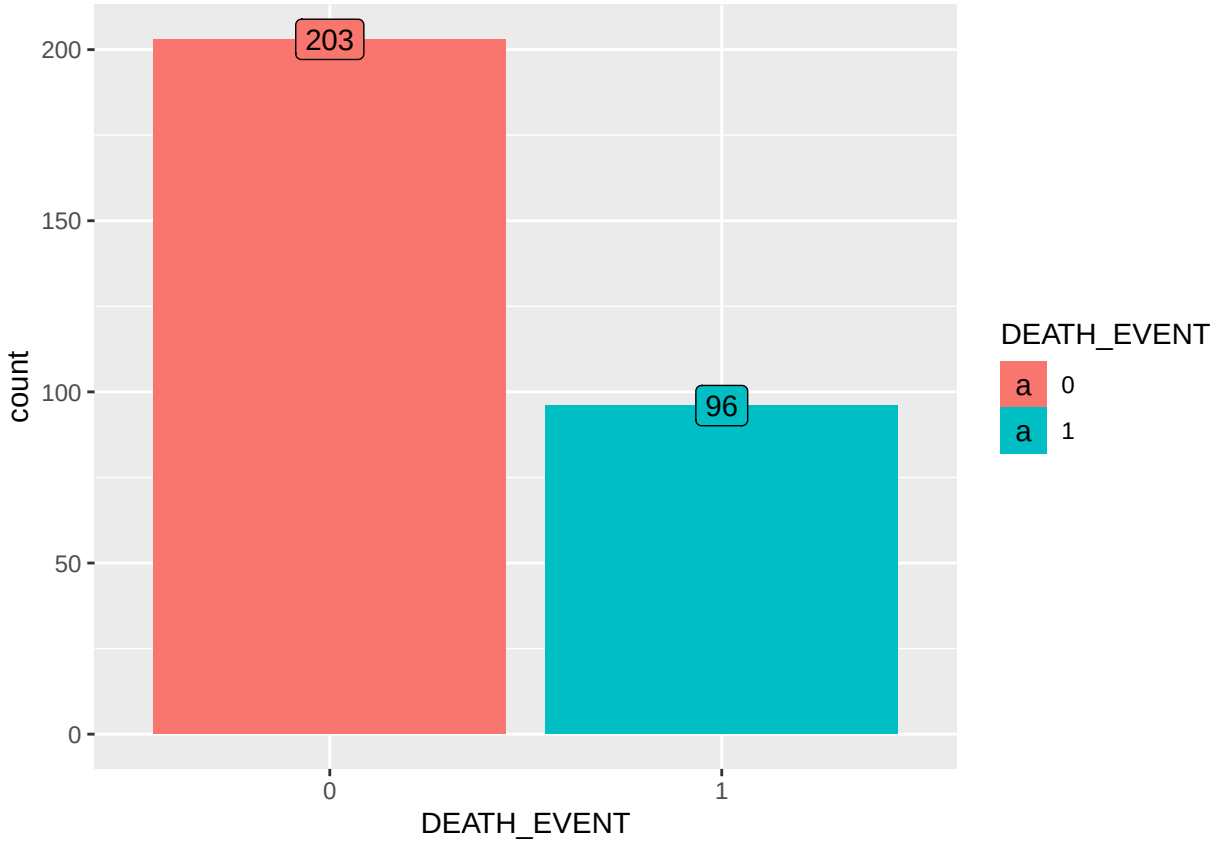
As a result of above plot and summary, we can conclude that our data is unbalanced and most of the patients are non smokers.

DEATH_EVENT:

```
library(ggplot2)
summary(heart.attk$DEATH_EVENT)
```

```
##   0   1
## 203  96
```

```
ggplot(heart.attk, aes(x = DEATH_EVENT, fill=DEATH_EVENT)) + geom_bar()+
geom_label(stat = "count", aes(label = ..count..))
```



As a result of above plot and summary, we can conclude that our data is unbalanced and most of the heart patients were survived from heart attack.

Now Let's Check Skewness Of Continuous Variables:

IF A COLUMN IN THE DATASET IS SKEWED THEN IT AFFECTS THE MODEL'S PERFORMANCE AND WE MIGHT END UP WITH WRONG PREDICTIONS. SKEWNESS IS THE MEASURE OF ASSYMETRY OF PROBABILITY DISTRIBUTION OF RANDOM VARIABLE ABOUT IT'S MEAN.

We take into account the following points:

If skewness is 0, the data are perfectly symmetrical If skewness is less than -1 or greater than 1, the distribution is highly skewed. If skewness is between -1 and -0.5 or between 0.5 and 1, the distribution is moderately skewed. If skewness is between -0.5 and 0.5, the distribution is approximately symmetric.

```
library(e1071)
skewness(heart.attk$age)
```

```
## [1] 0.4188266
```

```
skewness(heart.attk$creatinine_phosphokinase)
```

```
## [1] 4.41843
```

```r
skewness(heart.attk$ejection_fraction)
```

```
## [1] 0.5498228
```

```r
skewness(heart.attk$platelets)
```

```
## [1] 1.447681
```

```r
skewness(heart.attk$serum_sodium)
```

```
## [1] -1.037643
```

```r
skewness(heart.attk$serum_creatinine)
```

```
## [1] 4.411387
```

As we can see that “serum_creatinine”, “platelets” and creatinine_phosphokinase" has values which indicate skewness, therefore we need to treat them.

```r
## Before made any changes to our variables we are making backup.

heart.attk.platelets<-heart.attk$platelets
heart.cpk<-heart.attk$creatinine_phosphokinase
heart.serum.crt<-heart.attk$serum_creatinine
```

```r
# Now performed skewness removing methods:

heart.attk$creatinine_phosphokinase<-log(heart.attk$creatinine_phosphokinase)  ## taking log for greate
heart.attk$serum_creatinine<-log(heart.attk$serum_creatinine) ## taking log for greater skew
heart.attk$platelets<-sqrt(heart.attk$platelets) ##  taking square-root for moderate skew
```

```r
## Now calculating skewness:

skewness(heart.attk$creatinine_phosphokinase)
```

```
## [1] 0.4098623
```

```r
skewness(heart.attk$serum_creatinine)
```

```
## [1] 1.568132
```

```r
skewness(heart.attk$platelets)
```

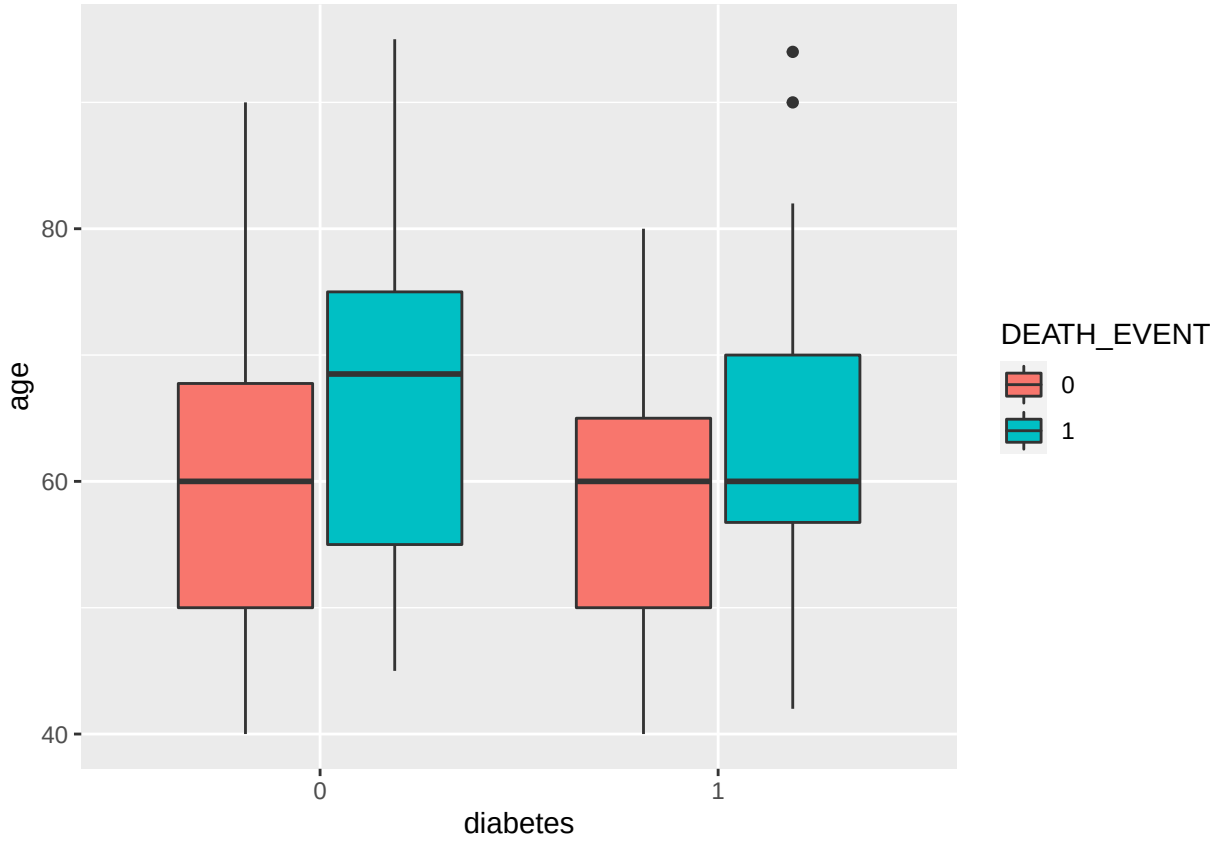
```
## [1] 0.1768912
```

So we can see that skewness is removed.

As we examined from previous statistical computations our target variable “DEATH_EVENT” is imbalanced and we need to treat it by relating it with the continuous variables in order to predict cause of death due to heart attack.

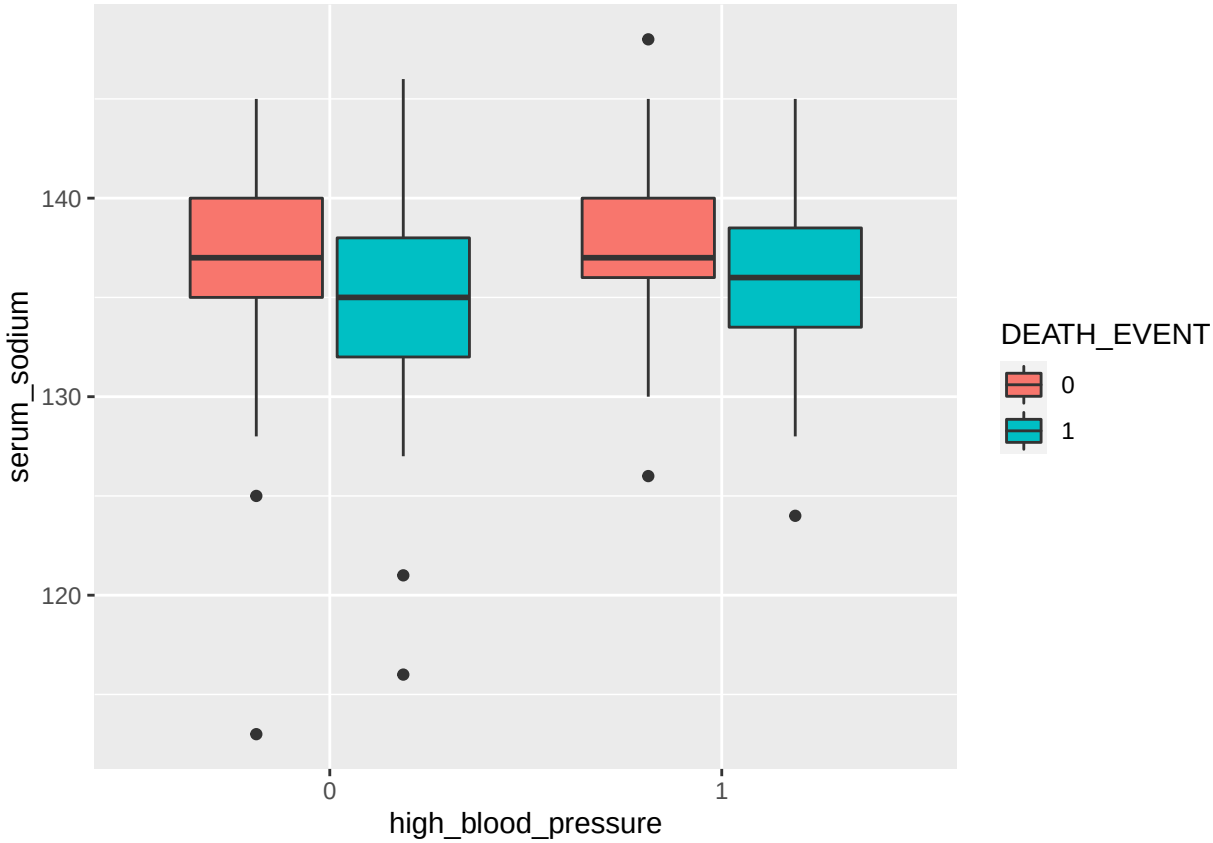
RELATIONSHIP BETWEEN VARIABLES:

```
ggplot(heart.attk, aes(x=diabetes, y=age, fill=DEATH_EVENT)) +
  geom_boxplot()
```



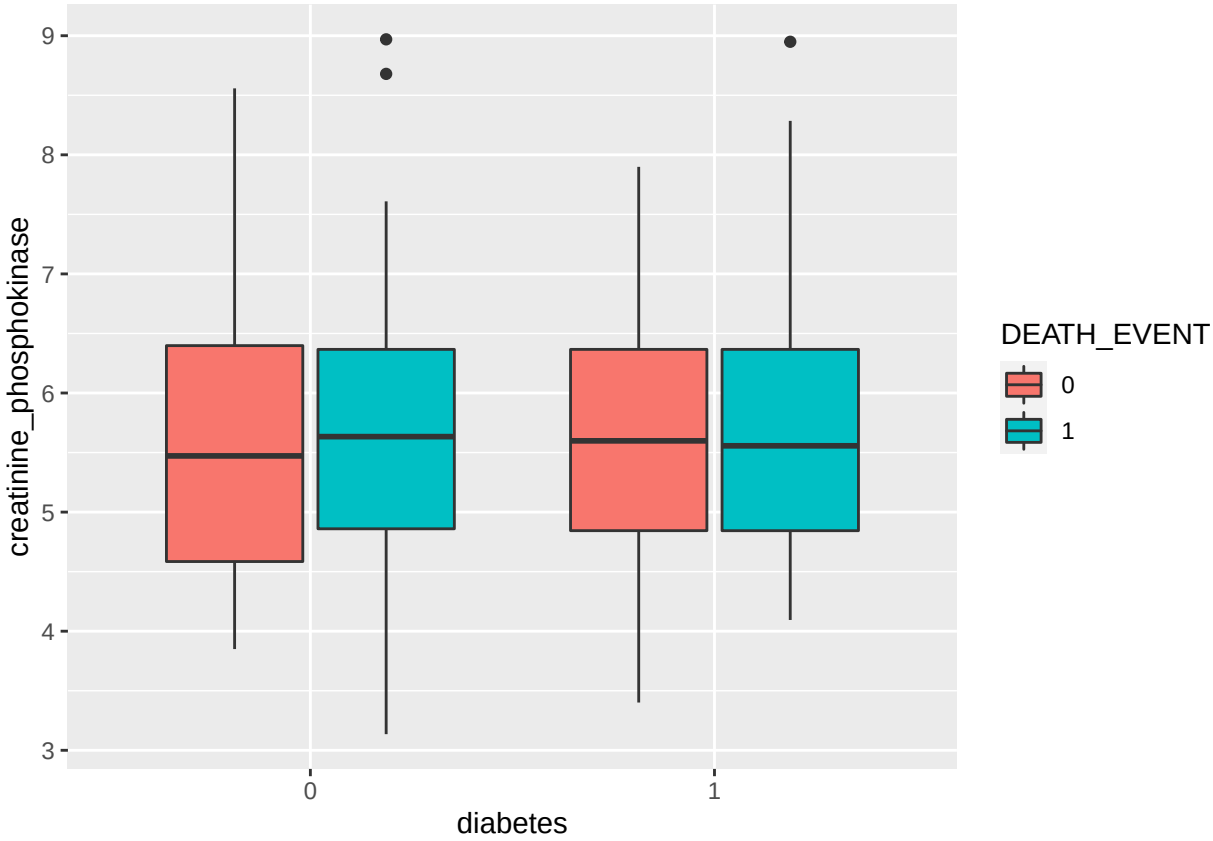
In the provided dataset there are people with diabetes

```
ggplot(heart.attk, aes(x=high_blood_pressure, y=serum_sodium, fill=DEATH_EVENT)) +
  geom_boxplot()
```



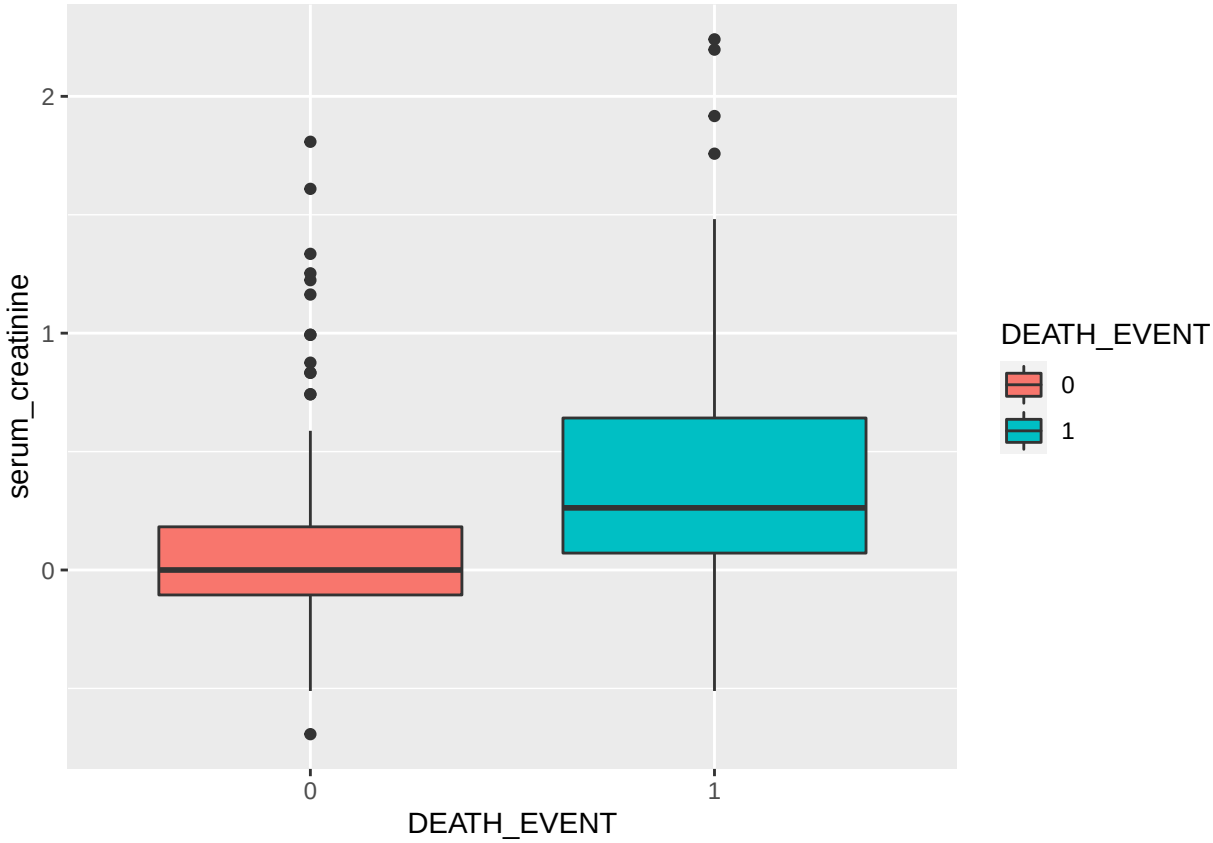
According to the medical research there is a strong link between high blood pressure and blood sodium levels. Normal range of values of serum sodium levels should be lie between 135 - 145(mEq/L). As the values provided in our dataset is within the range, therefore we can see that there are more people without high blood pressure in the plot above and death events are also comparitively less. Thus we can say that control Salt(Sodium) intake in order to prevent your heart.

```
ggplot(heart.attk, aes(x=diabetes, y=creatinine_phosphokinase, fill=DEATH_EVENT)) +
  geom_boxplot()
```



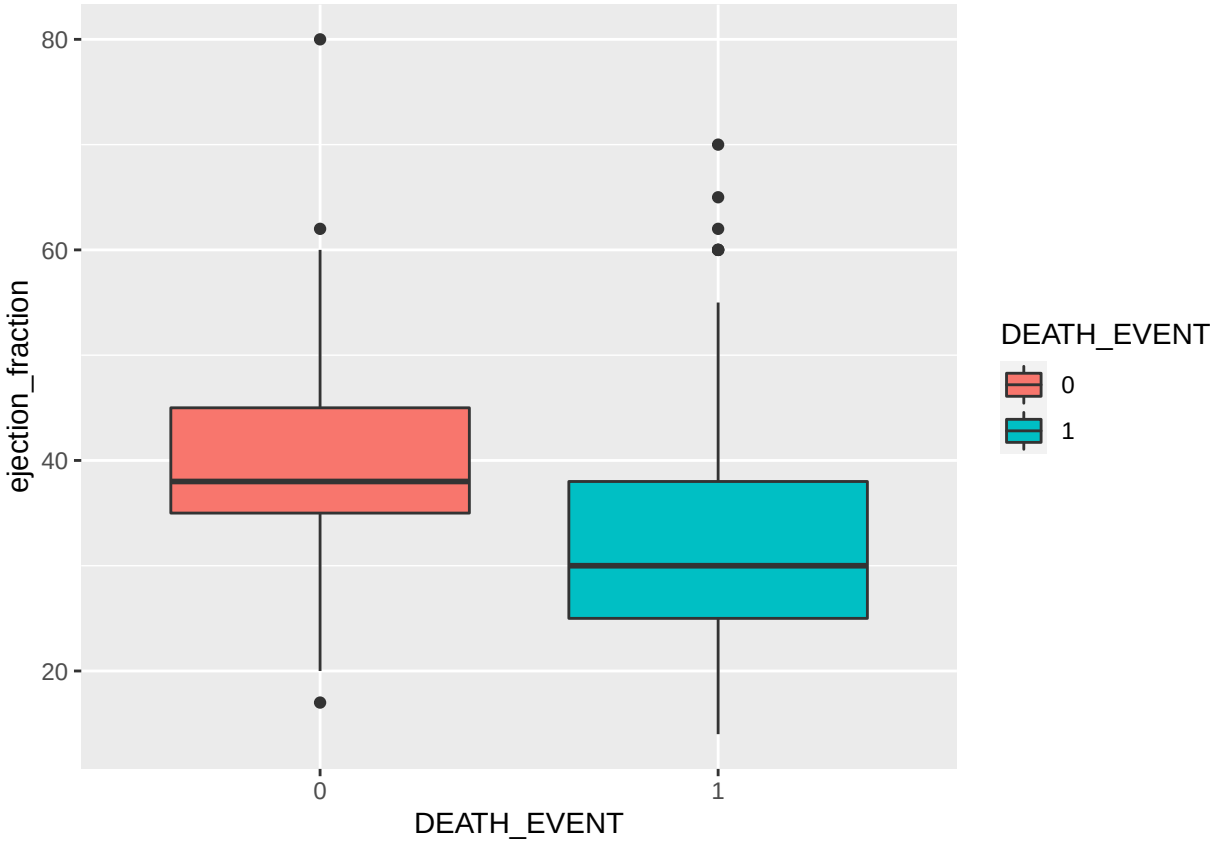
Creatine phosphokinase (CPK) is an enzyme in the body. It is found mainly in the heart, brain, and skeletal muscle.In diabetic patients visiting the clinic, elevated CK levels occur in one-fifth of the cases. Thus we can say that it does not much influence the target.

```
ggplot(heart.attk, aes(x=DEATH_EVENT, y=serum_creatinine, fill=DEATH_EVENT)) +
  geom_boxplot()
```



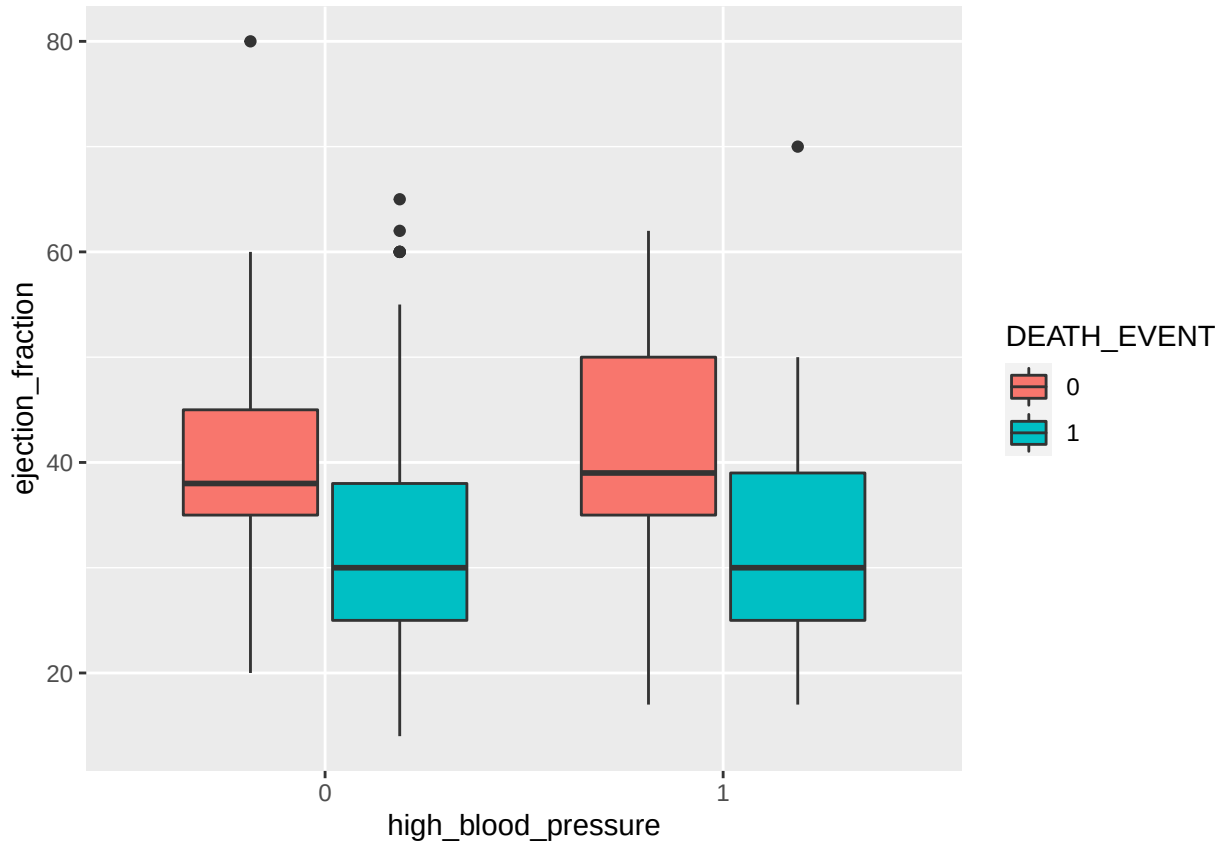
Elevated creatinine marks damage to kidney which in turn affects normal function of other organs. Thus Death event is occur more in the cases of elevated creatinine levels.

```
ggplot(heart.attk, aes(x=DEATH_EVENT, y=ejection_fraction, fill=DEATH_EVENT)) +
  geom_boxplot()
```



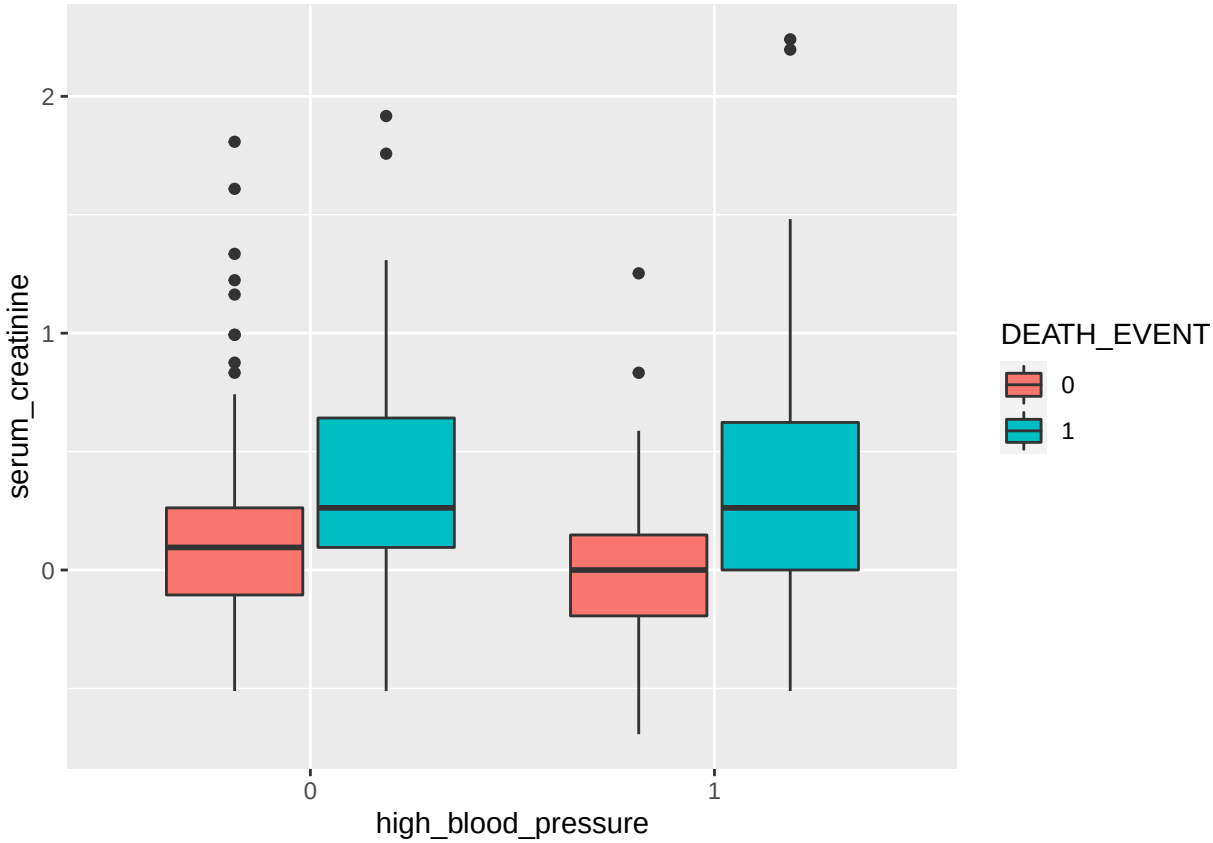
Since normal ejection fraction is should be in between 50% to 75%, so low ejection fraction leads to more death.

```
ggplot(heart.attk, aes(x=high_blood_pressure, y=ejection_fraction, fill=DEATH_EVENT)) +
  geom_boxplot()
```



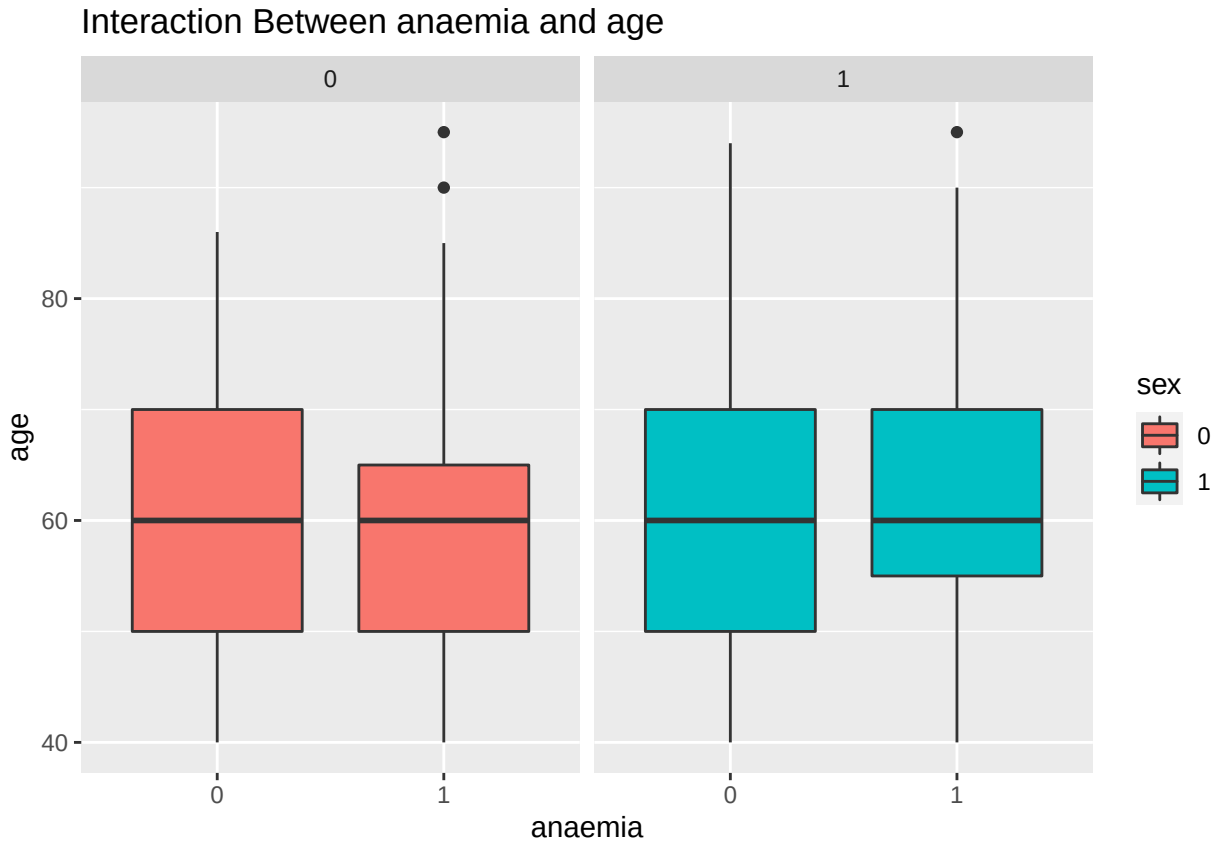
high BP though increases ejection fraction but long term effect of high BP increases chances of death as well.

```
ggplot(heart.attk, aes(x=high_blood_pressure, y=serum_creatinine, fill=DEATH_EVENT)) +
  geom_boxplot()
```



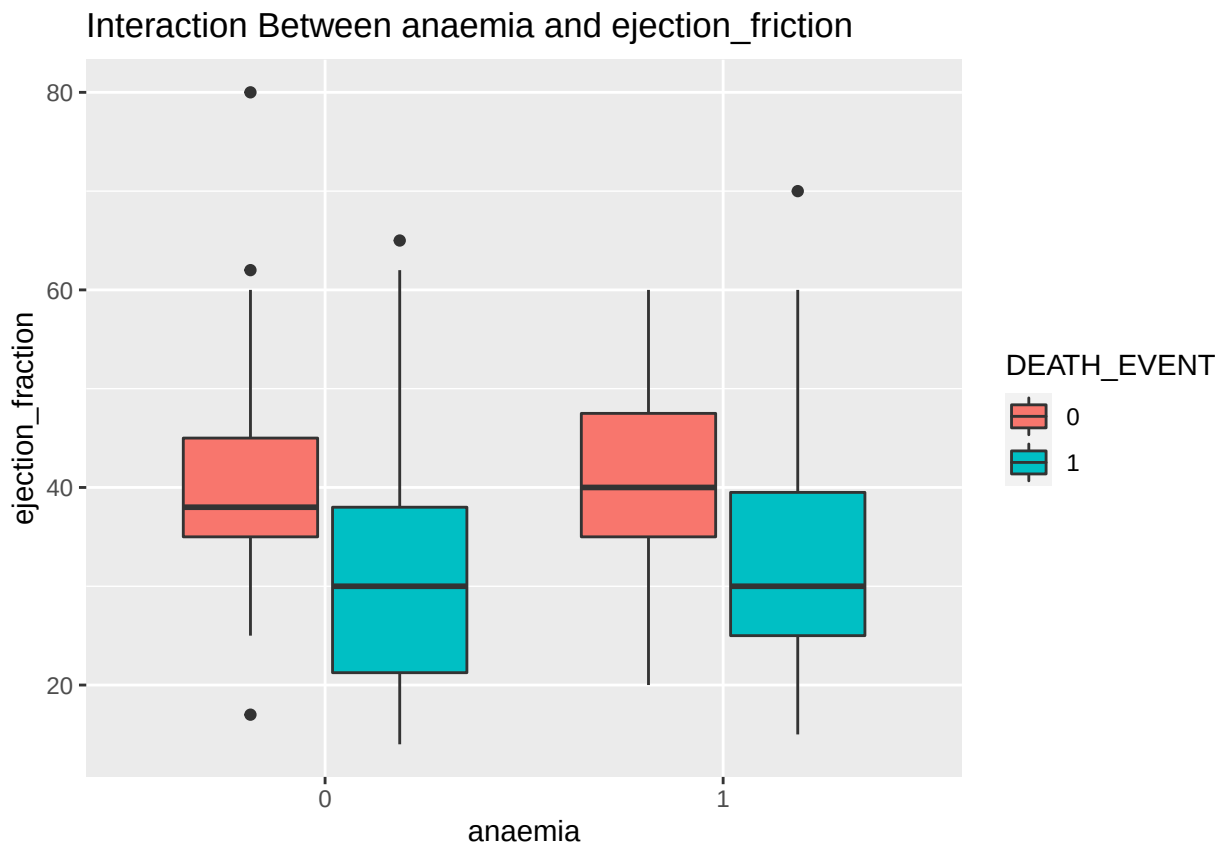
Presence of high Blood pressure affects other organs as well like kidney and increases creatinine.

```
ggplot(heart.attk, aes(x=anaemia, y=age, fill= sex)) +
  geom_boxplot()+
  labs(title = 'Interaction Between anaemia and age') +
  facet_wrap(~sex)
```



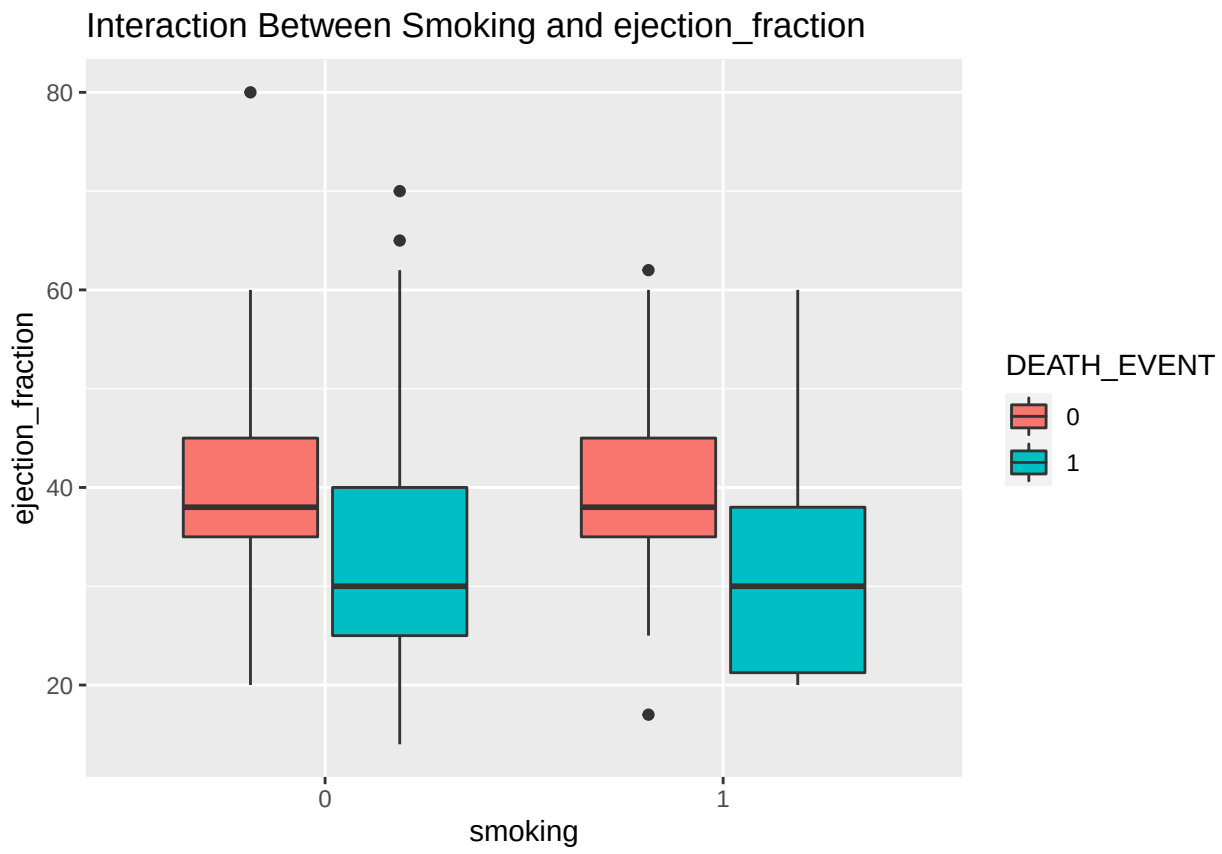
There is no clear relationship between age and anaemia leading to death.

```
ggplot(heart.attk, aes(x=anaemia, y=ejection_fraction, fill=DEATH_EVENT)) +
  geom_boxplot()+
  labs(title = 'Interaction Between anaemia and ejection_friction')
```



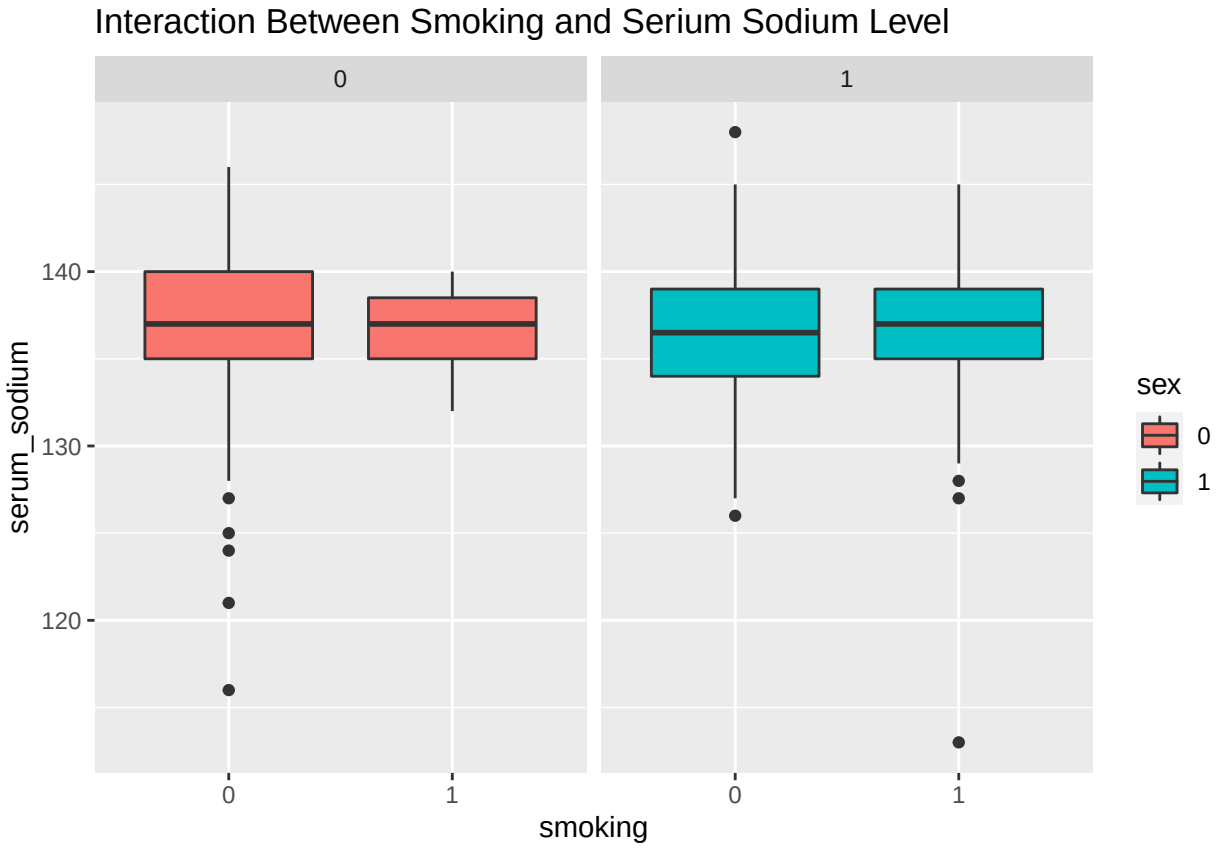
After reviewing above plot we can say that anaemia can effect ejection fraction.

```
ggplot(heart.attk, aes(x=smoking, y=ejection_fraction, fill=DEATH_EVENT)) +
  geom_boxplot()+
    labs(title = 'Interaction Between Smoking and ejection_fraction')
```



On the basis of above plot we can concluded that smoking causes decrease in ejection fraction.

```
ggplot(heart.attk, aes(x=smoking, y=serum_sodium, fill=sex)) +
  geom_boxplot() +
    labs(title = 'Interaction Between Smoking and Serium Sodium Level') +
  facet_wrap(~sex)
```



From the above analysis following insights can be drawn:

!.High Blood Pressure has not Much relation with increase in serum_sodium in the given dataset but research shows that high sodium level can increase BP.

2.Smoking must be stopped as it affects ejection fraction

3.High Levels of serum creatinine which arise due to high BP increase death.

#Preliminary Analysis:

Well before going to implement the PCA it is required to determine, whether or not there is a correlation between the numerical variables. To implement this, we have to first create a subset of the dataset that contains only continuous values.

```
# As in analysis we had changes the values of variables(platelets,creatinine_phosphokinase and serum_cr

heart.attk$platelets<-heart.attk.platelets
heart.attk$creatinine_phosphokinase<-heart.cpk
heart.attk$serum_creatinine<-heart.serum.crt
```

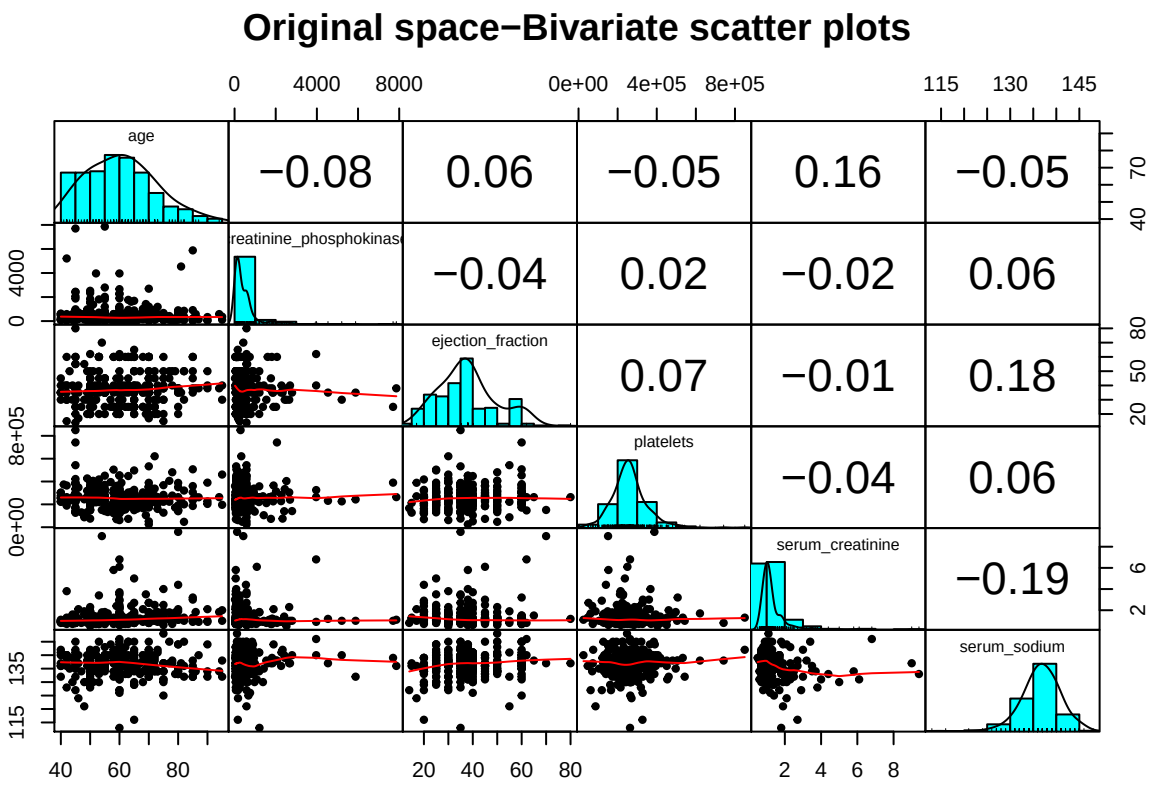
```
heart.attk.sub<-heart.attk[-c(2,4,6,10:12)]
library(psych)
```

```
## Warning: package 'psych' was built under R version 4.0.5
```

```
##
## Attaching package: 'psych'
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha

## The following object is masked from 'package:gamlss':
##
##     cs
```

```
pairs.panels(heart.attk.sub, main= "Original space-Bivariate scatter plots",
ellipses = FALSE, gap = 0)
```



This is a preliminary analysis of the data in the original space, which goal to understand if it would be useful to run a Principal Component Analysis and a Cluster one on this dataset.Thus,the coefficients of correlation between variables are located in the upper triangle of the matrix and are used to determine whether PCA is useful or not, whereas the scatterplots of data are found in the lower triangle and the non-parametric density of the data are located on the main diagonal, both are used to determine whether CA is useful or not. In particular, if we look at the plot, we can see that there is no correlation found among the most of the variables, as their values of coefficients of correlation is less than 0.1 and close to 0, but in some variable there is weak correlation found, so in short we can say that for our data we can apply Principal Component Analysis on the variables age and serum_creatinine that have value of coefficients of correlation (0.23) which means they are weakly positive correlated, variables serum_creatinine and serum_sodium their coefficients of correlation value is (-0.26) which means they are weakly negative correlated. Hence, we can look forward to perform PCA on our data..

as we can observed that there are clusters, because the data are grouped along the diagonal instead of remaining scattered in space.

PRINCIPAL COMPONENT ANALYSIS:

Data Standardization:

In order to evaluate the difference between the variables, the mean and the variance are computed for each variable.

```
apply(heart.attk.sub, 2, mean)
```

```
##                      age creatinine_phosphokinase        ejection_fraction
##                 60.83389                581.83946                 38.08361
##                platelets         serum_creatinine             serum_sodium
##             263358.02926                  1.39388                136.62542
```

```
apply(heart.attk.sub, 2, var)
```

```
##                      age creatinine_phosphokinase        ejection_fraction
##             1.414865e+02             9.414586e+05             1.400635e+02
##                platelets         serum_creatinine             serum_sodium
##             9.565669e+09             1.070211e+00             1.946996e+01
```

The variables are very different from each other. In order to work on homogeneous variables, it is better to standardize the variable in order to have zero mean and unitary variance.

```
scaled_heart.attk<-apply(heart.attk.sub, 2, scale)
head(scaled_heart.attk)
```

```
##             age creatinine_phosphokinase ejection_fraction     platelets
## [1,]  1.1909487              0.000165451      -1.527997920  1.678834e-02
## [2,] -0.4904571              7.502062717      -0.007064906  7.523048e-09
## [3,]  0.3502458             -0.449185725      -1.527997920 -1.036336e+00
## [4,] -0.9108085             -0.485257493      -1.527997920 -5.455595e-01
## [5,]  0.3502458             -0.434757017      -1.527997920  6.507077e-01
## [6,]  2.4520030             -0.551217299       0.161927651 -6.069065e-01
##      serum_creatinine serum_sodium
## [1,]       0.48923681  -1.50151891
## [2,]      -0.28407611  -0.14173853
## [3,]      -0.09074788  -1.72814897
## [4,]       0.48923681   0.08489153
## [5,]       1.26254973  -4.67433977
## [6,]       0.68256504  -1.04825878
```

Computing PCs:

In order to find the Principal Components, the Eigen decomposition is applied to the covariance matrix of the standardized data.

```
heart.attk_cov <- cov(scaled_heart.attk)
heart.attk_eigen<-eigen(heart.attk_cov)
heart.attk_eigen$value
```

```
## [1] 1.3452721 1.1536488 0.9873768 0.9605126 0.8238408 0.7293489
```

As an example the eigen vectors of the first two PCs are shown:

```
phi <- heart.attk_eigen$vectors[,1:3]
phi <- -phi
row.names(phi) <- c("age", "creatinine_phosphokinase", "ejection_fraction", "platelets", "serum_creatini
colnames(phi) <- c("PC1", "PC2", "PC3")
phi
```

```
##                               PC1        PC2         PC3
## age                     -0.3737830  0.5411292 -0.07000120
## creatinine_phosphokinase  0.2000394 -0.3771441 -0.61157347
## ejection_fraction         0.2894085  0.6652120 -0.08330005
## platelets                 0.3023587  0.1226137 -0.63050293
## serum_creatinine         -0.5486448  0.1943877 -0.45312098
## serum_sodium              0.5865845  0.2638445  0.10623682
```

By examining the loading we note that first loading vector phi 1 places most of its weight on serum_sodium(0.540) and much less weight on serum_creatinine(-0.619). The second loading vector phi 2 places most of its weight on ejection_fraction (0.603) and much less weight on creatinine_phosphokinase(-0.517).

PC's Scores:

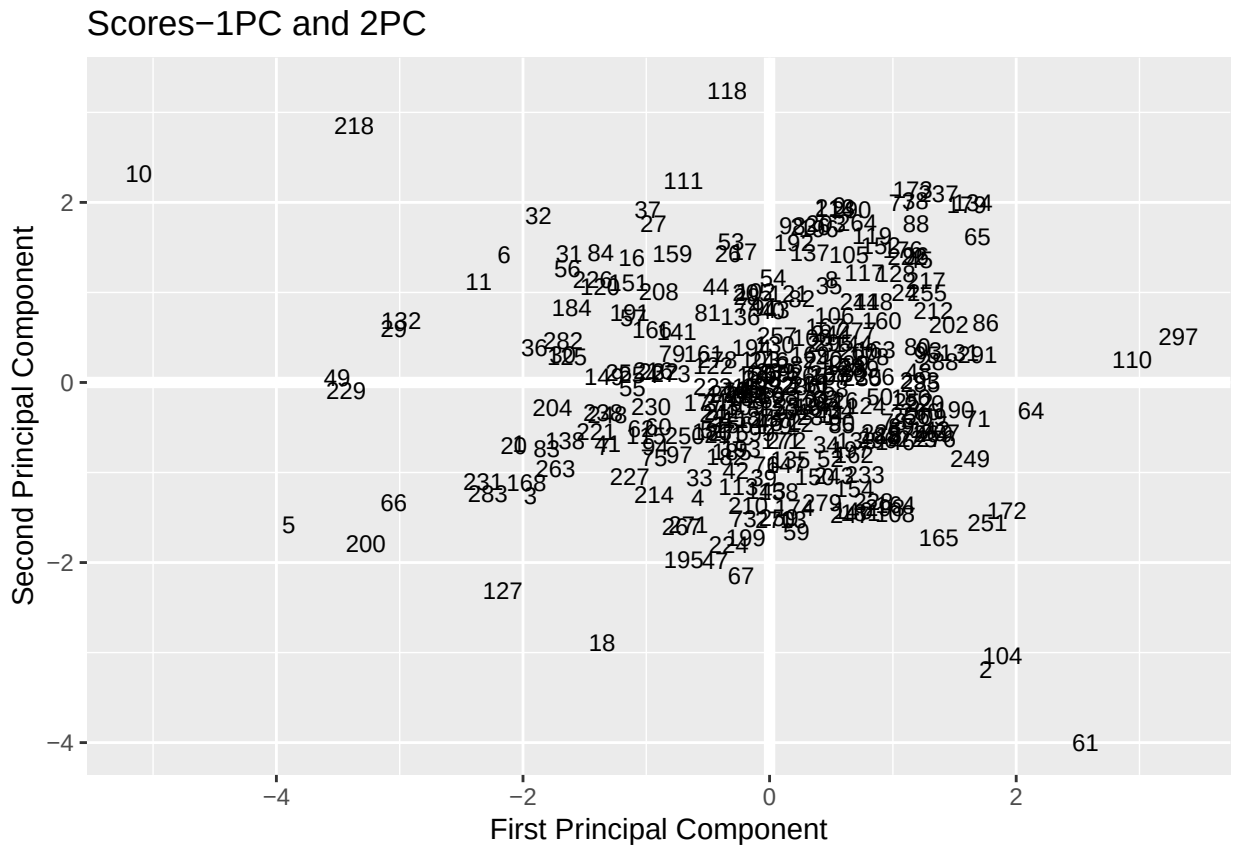
```
PC1 <- scaled_heart.attk %*% phi[,1]
PC2 <- scaled_heart.attk %*% phi[,2]
PC <- data.frame(ID = row.names(heart.attk), PC1, PC2)
head(PC)
```

```
##   ID        PC1        PC2
## 1  1 -2.0314477 -0.6710552
## 2  2  1.7547031 -3.1920769
## 3  3 -1.9402485 -1.2581783
## 4  4 -0.5824173 -1.2756888
## 5  5 -3.8979393 -1.5710370
## 6  6 -2.1528005  1.4241448
```

```
library(ggplot2)
library(modelr)
```

```
## Warning: package 'modelr' was built under R version 4.0.5
```

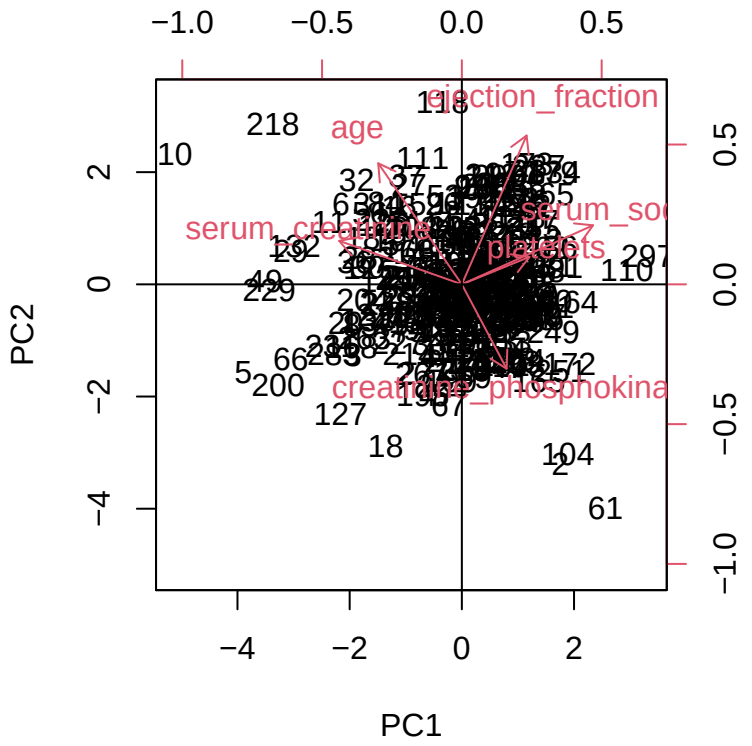
```
ggplot(PC, aes(PC1, PC2)) +
modelr::geom_ref_line(h = 0) +
modelr::geom_ref_line(v = 0) +
geom_text(aes(label = ID), size = 3) +
xlab("First Principal Component") +
ylab("Second Principal Component") +
ggtitle("Scores-1PC and 2PC")
```



Biplot:

It is possible to visualize the scores and the original variable (represented by arrows) in the space spanned by the first two principal components. we set center= True to shift the variable into zero center,

```
#set.seed(125)
heart.attk_pc <- prcomp(heart.attk.sub, scale. = TRUE)
biplot(heart.attk_pc, scale=0)
abline(h=0)
abline(v=0)
```



The angle between the arrows gives information on the correlation between the two variables.For instance age and serum_creatinine are positively correlated as they are closer to each other.

```
cor(heart.attk.sub)
```

```
##                                age creatinine_phosphokinase ejection_fraction
## age                     1.00000000              -0.08158390        0.06009836
## creatinine_phosphokinase -0.08158390               1.00000000       -0.04407955
## ejection_fraction        0.06009836              -0.04407955        1.00000000
## platelets               -0.05235437               0.02446339        0.07217747
## serum_creatinine         0.15918713              -0.01640848       -0.01130247
## serum_sodium            -0.04596584               0.05955016        0.17590228
##                          platelets serum_creatinine serum_sodium
## age                     -0.05235437       0.15918713  -0.04596584
## creatinine_phosphokinase  0.02446339      -0.01640848   0.05955016
## ejection_fraction         0.07217747      -0.01130247   0.17590228
## platelets                 1.00000000      -0.04119808   0.06212462
## serum_creatinine         -0.04119808       1.00000000  -0.18909521
## serum_sodium              0.06212462      -0.18909521   1.00000000
```

In order to select the number of principal components, there are three heuristic methods are required to proposed.

Cumulative proportion of Variance Explained (CPVE):

According to this approach, the first q principal components that explain at least 80% of the total variance are retained.

```
(PVE <- heart.attk_eigen$values/sum(heart.attk_eigen$values))
```

```
## [1] 0.2242120 0.1922748 0.1645628 0.1600854 0.1373068 0.1215582
```

The first PC explains 24.27% of the variability The second PC explains 19.69% of the variability The third PC explains 16.26% of the variability The fourth PC explains 15.31% of the variability The fifth PC explains 13.16% of the variables The fifth PC explains 11.28% of the variables

```
cumsum(PVE)
```

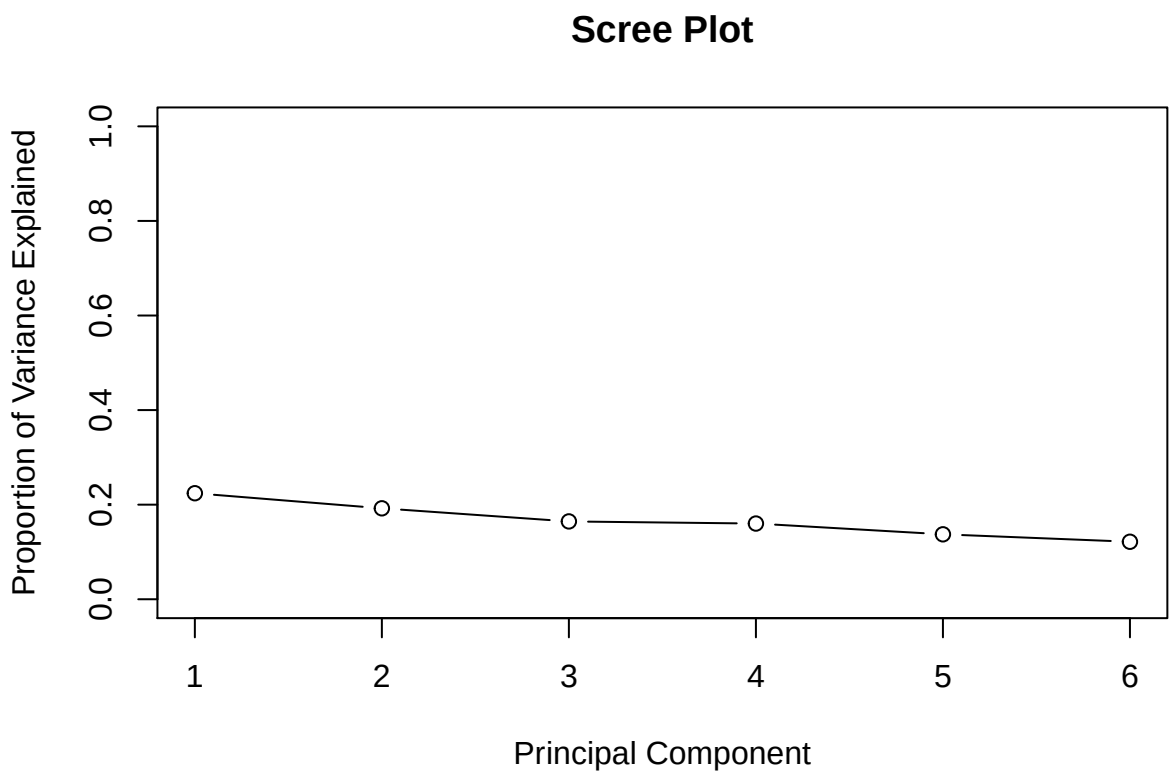
```
## [1] 0.2242120 0.4164868 0.5810496 0.7411351 0.8784418 1.0000000
```

According to the CPVE we have to retain as many PCs as needed to explain at least the 80% of the total variance, hence we have to retain the first 5 PCs.

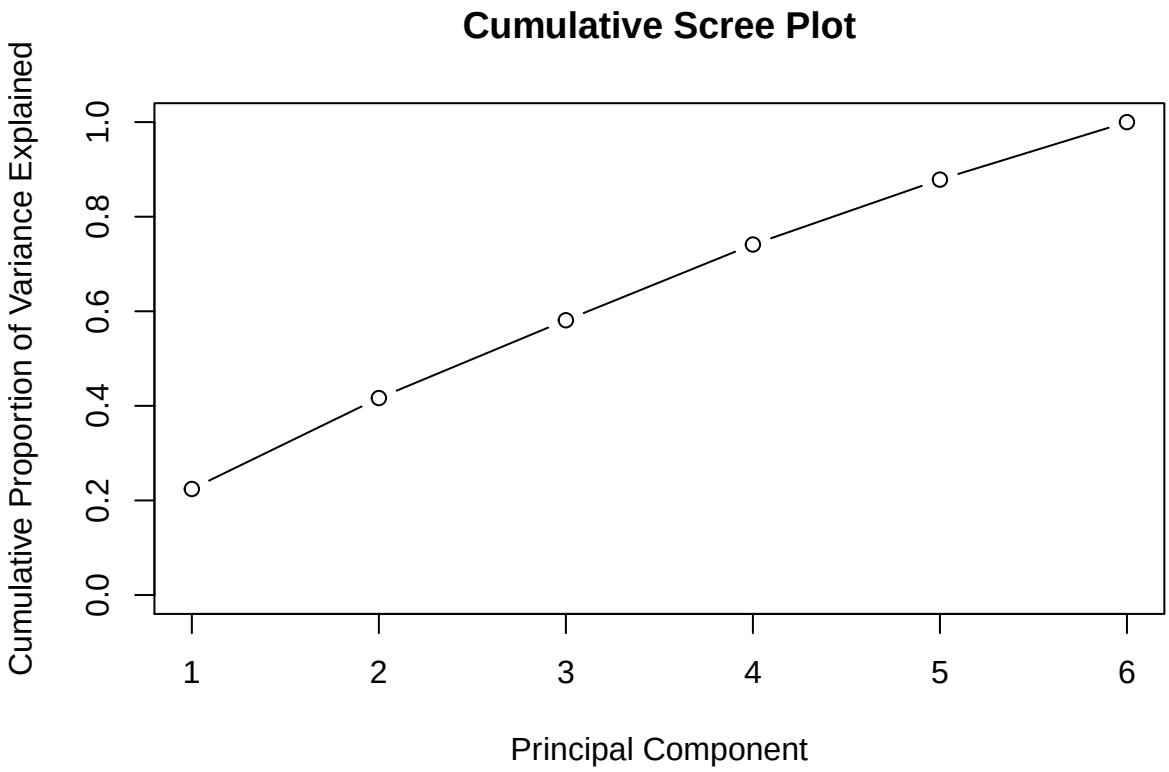
Scree plot:

The scree plot suggests selecting q corresponding to the value of m where the curve becomes flat.

```
plot(PVE, xlab="Principal Component", ylab="Proportion of Variance Explained", main="Scree Plot", ylim=
```



```r
plot(cumsum(PVE), xlab="Principal Component", main="Cumulative Scree Plot", ylab="Cumulative Proportion
```



In the "Proportion of variance Explained" plot, the elbow point is not so clear and it may be at q=2 or q=4. However, according to this method, it seems reasonable to retain the first four principal components.

Kaiser's rule:

According to the Kaiser's rule, for standardized data, the principal components with the variance greater than 1 are taken.

```r
heart.attk_eigen$values
```

```
## [1] 1.3452721 1.1536488 0.9873768 0.9605126 0.8238408 0.7293489
```

The Kaiser's rule suggests keeping the first two principal components.

PCA result anaysis:

Based on different strategies I obtained different results.The cumulative PVE rule suggests retaining the first five principal components, the Kaiser's rule suggests retaining the first two principal components while the Scree plot doesn't provide a clear image,. Therefore, I decided to choose CPVE results as it will obtain more PCs.

Cluster Analysis:

Clustering is the method of dividing the data points into a number of groups such that data points in the same groups are more similar to other data points in the same group than those in other groups. In simple words, the clustering analysis's goal is to segregate groups with similar traits and assign them into clusters.

Hopkins statistic:

```
heart.attk_scale <- scale(heart.attk.sub)
```

```
library(clustertend)
```

```
## Warning: package 'clustertend' was built under R version 4.0.5
```

```
hopkins(heart.attk_scale, n = nrow(heart.attk_scale)-1)
```

```
## $H
## [1] 0.1718656
```

The Hopkins statistics value is near zero. Under the assumption that the configuration without the cluster is the uniform distribution, hence the result indicates clustered data.

Computing Euclidean distance:

```
dist.eucl <- dist(heart.attk_scale, method = "euclidean")
eucl <- round(as.matrix(dist.eucl)[1:6,1:6],2)
rownames(eucl) <- c("age", "CPK", "serum_crt", "serum_sodium", "platelets","eject_fra")
colnames(eucl) <- c("age", "CPK", "serum_crt", "serum_sodium", "platelets", "eject_frac")
eucl
```

```
##               age  CPK serum_crt serum_sodium platelets eject_frac
## age          0.00 7.99      1.55         2.74      3.46       2.32
## CPK          7.99 0.00      8.36         8.20      9.45       8.70
## serum_crt    1.55 8.36      0.00         2.34      3.65       2.92
## serum_sodium 2.74 8.20      2.34         0.00      5.13       3.94
## platelets    3.46 9.45      3.65         5.13      0.00       4.73
## eject_fra    2.32 8.70      2.92         3.94      4.73       0.00
```

According to this distance matrix, relative to a subset of data, the most distant observations are (2.09)serum_creatinine and age , while the most similar ones are (6.30)platelets and CPK.

Computing Manhattan distance:

```
dist.man <- dist(heart.attk_scale, method = "manhattan")
manhattan<- round(as.matrix(dist.man)[1:6,1:6],2)
rownames(manhattan) <- c("age", "CPK", "serum_crt", "serum_sodium", "platelets","eject_fra")
colnames(manhattan) <- c("age", "CPK", "serum_crt", "serum_sodium", "platelets","eject_fra")
manhattan
```

```
##                age   CPK serum_crt serum_sodium platelets eject_fra
## age           0.00 12.85      3.15         4.74      5.86      4.77
## CPK          12.85  0.00     13.13        11.47     17.03     13.64
## serum_crt     3.15 13.13      0.00         4.18      6.00      5.78
## serum_sodium  4.74 11.47      4.18         0.00      8.04      6.51
## platelets     5.86 17.03      6.00         8.04      0.00      9.37
## eject_fra     4.77 13.64      5.78         6.51      9.37      0.00
```

According to this distance matrix, relative to a subset of data, the most distant observations are (4.32) serum_creatinine and age , while the most similar ones are (12.94)platelets and CPK.

In the above symmetric matrix, each value represents the distance between units. The values on the diagonal represent the distance between units and themselves (which is zero).

Visualizing distance matrices:

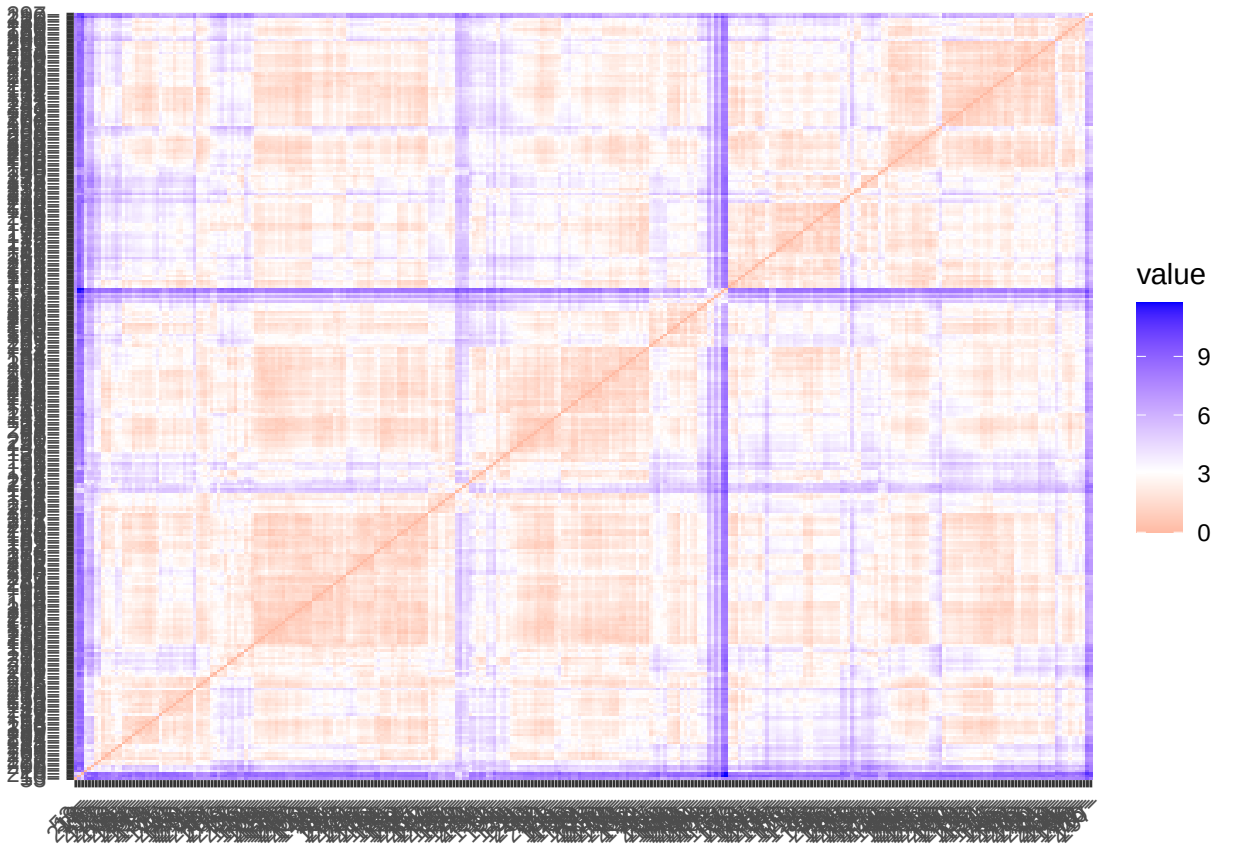
Euclidean:

```
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 4.0.5
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

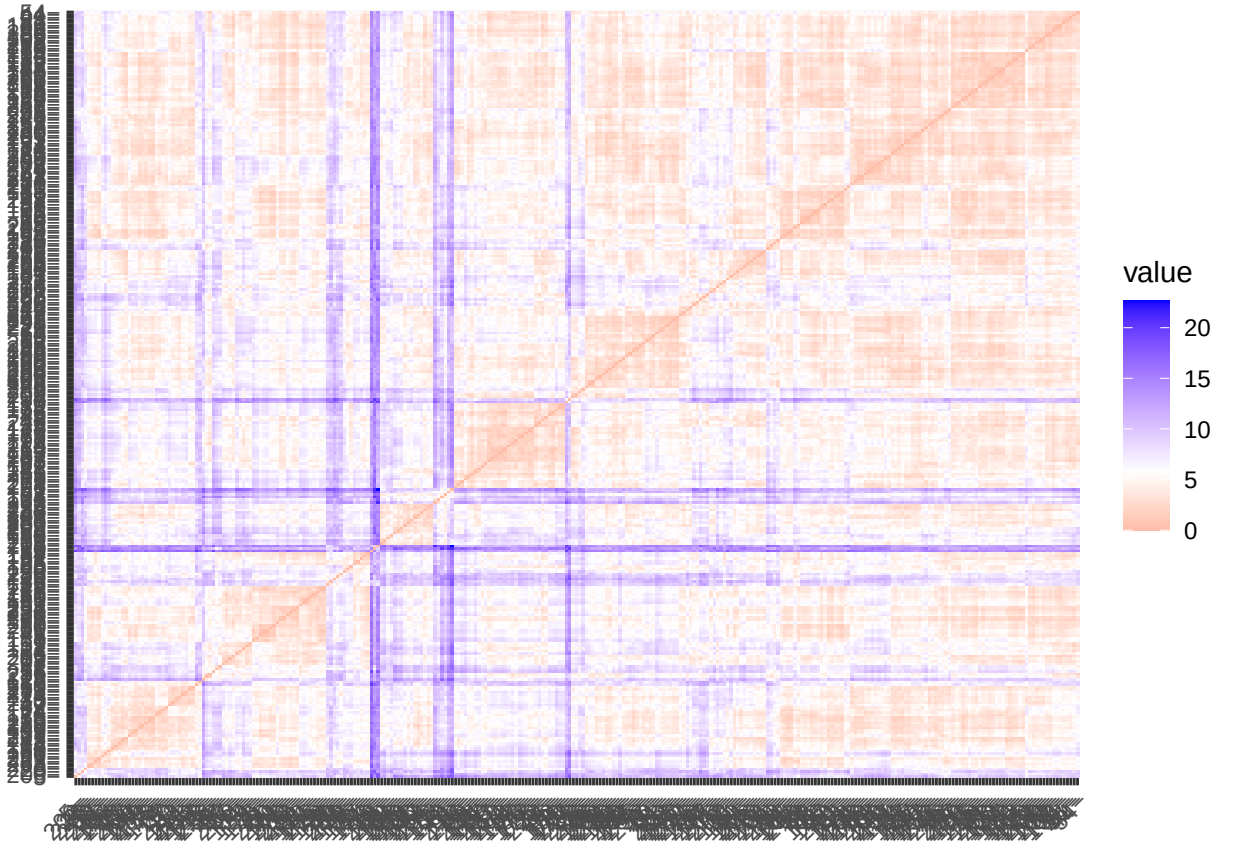
```
fviz_dist(dist.eucl)
```



The color level is proportional to the value of the dissimilarity between observations. Red indicates high similarity (i.e.: low dissimilarity) while blue indicates low similarity. According to the Visual method, that uses the Euclidean distance as distance between units, the data should not contain a noticeable clustering structure. While not sure of the presence of a clustering structure, the analysis continues.

Manhattan:

```
fviz_dist(dist.man)
```



Optimal number of clusters:

Using both Euclidean and Manhattan distance, according to different clustering methods, the optimal number of clusters will be computed

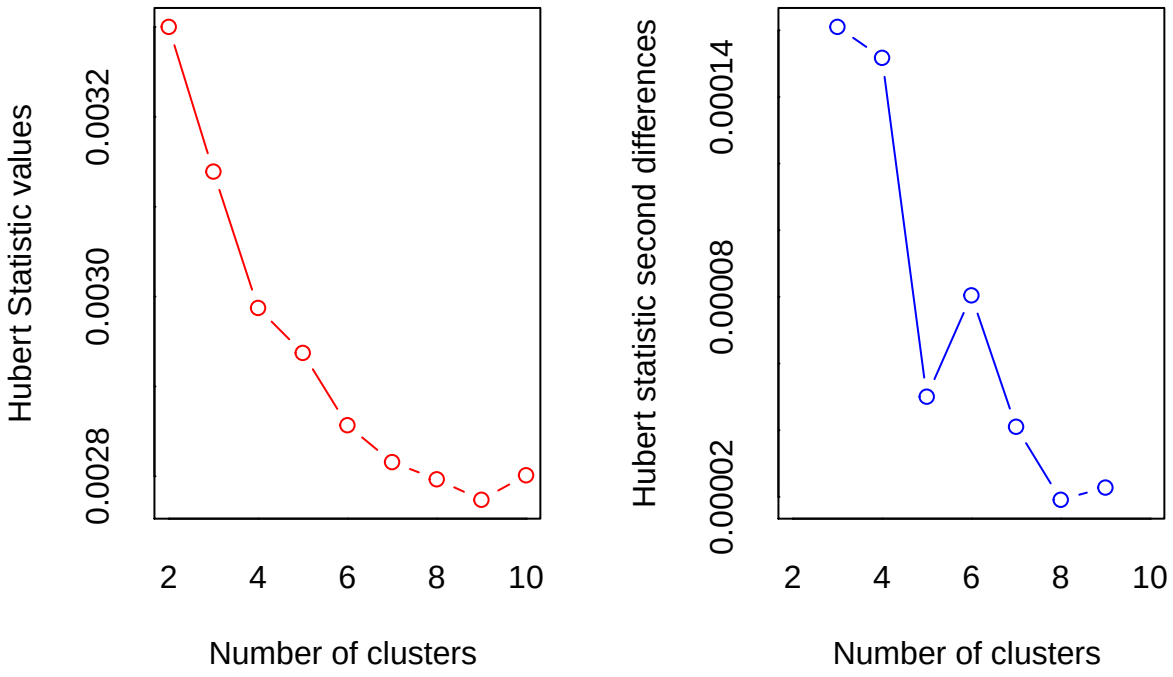
Hierarchical method:

Average linkage method and Euclidean distance

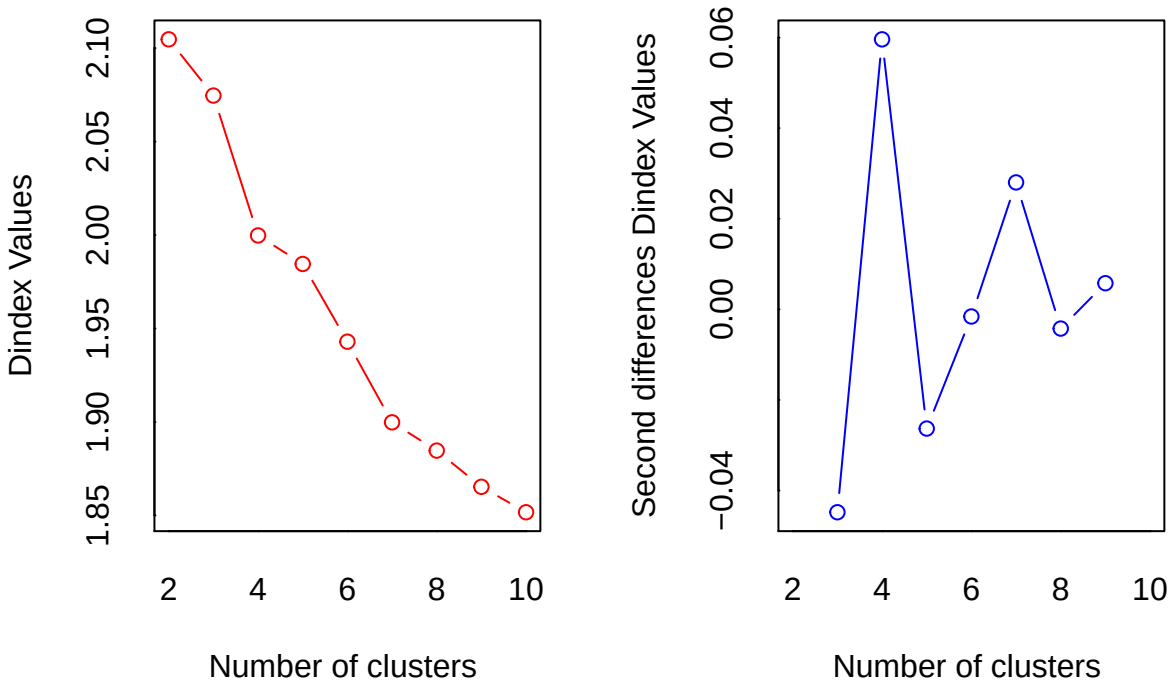
```
library(NbClust)
nb <- NbClust(heart.attk_scale, distance = "euclidean", min.nc = 2, max.nc = 10,
method = "average")
```

```
## Warning in pf(beale, pp, df2): NaNs produced
```

```
## [1] "Frey index : No clustering structure in this data set"
```




```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```

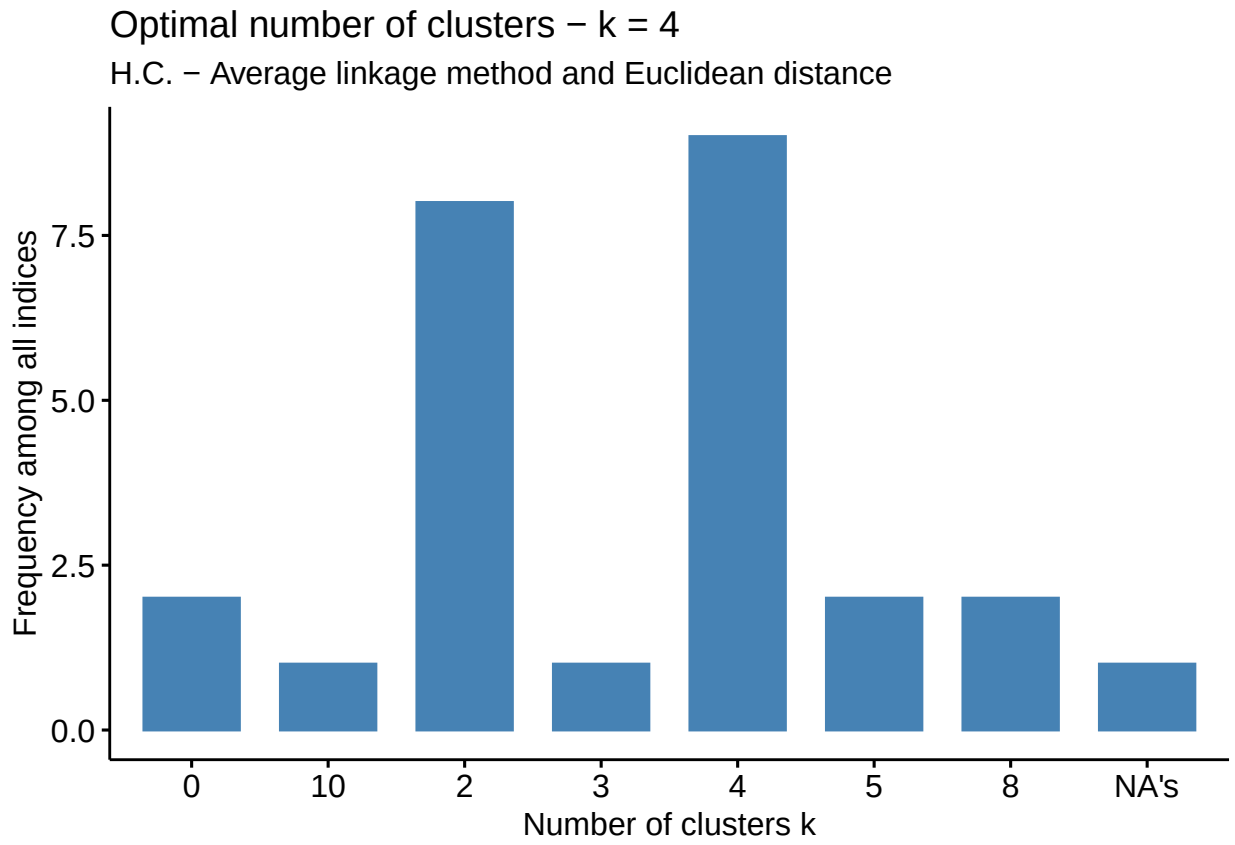



```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 8 proposed 2 as the best number of clusters
## * 1 proposed 3 as the best number of clusters
## * 9 proposed 4 as the best number of clusters
## * 2 proposed 5 as the best number of clusters
## * 2 proposed 8 as the best number of clusters
## * 1 proposed 10 as the best number of clusters
##
##                     ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  4
##
##
## *******************************************************************
```

```
library(factoextra)
fviz_nbclust(nb)+labs(subtitle = "H.C. - Average linkage method and Euclidean distance", cex.sub= 0.5)
```

```
## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
```

```
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") .viz_NbClust(x, print.summary, : the
## condition has length > 1 and only the first element will be used

## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") {: the condition has length > 1 and
## only the first element will be used

## Among all indices:
## ===================
## * 2 proposed  0 as the best number of clusters
## * 8 proposed  2 as the best number of clusters
## * 1 proposed  3 as the best number of clusters
## * 9 proposed  4 as the best number of clusters
## * 2 proposed  5 as the best number of clusters
## * 2 proposed  8 as the best number of clusters
## * 1 proposed  10 as the best number of clusters
## * 1 proposed  NA's as the best number of clusters
##
## Conclusion
## =========================
## * According to the majority rule, the best number of clusters is  4 .
```




```
hc <- hclust(dist.eucl, method = "average")
group <- cutree(hc, k=4)
table(group)
```

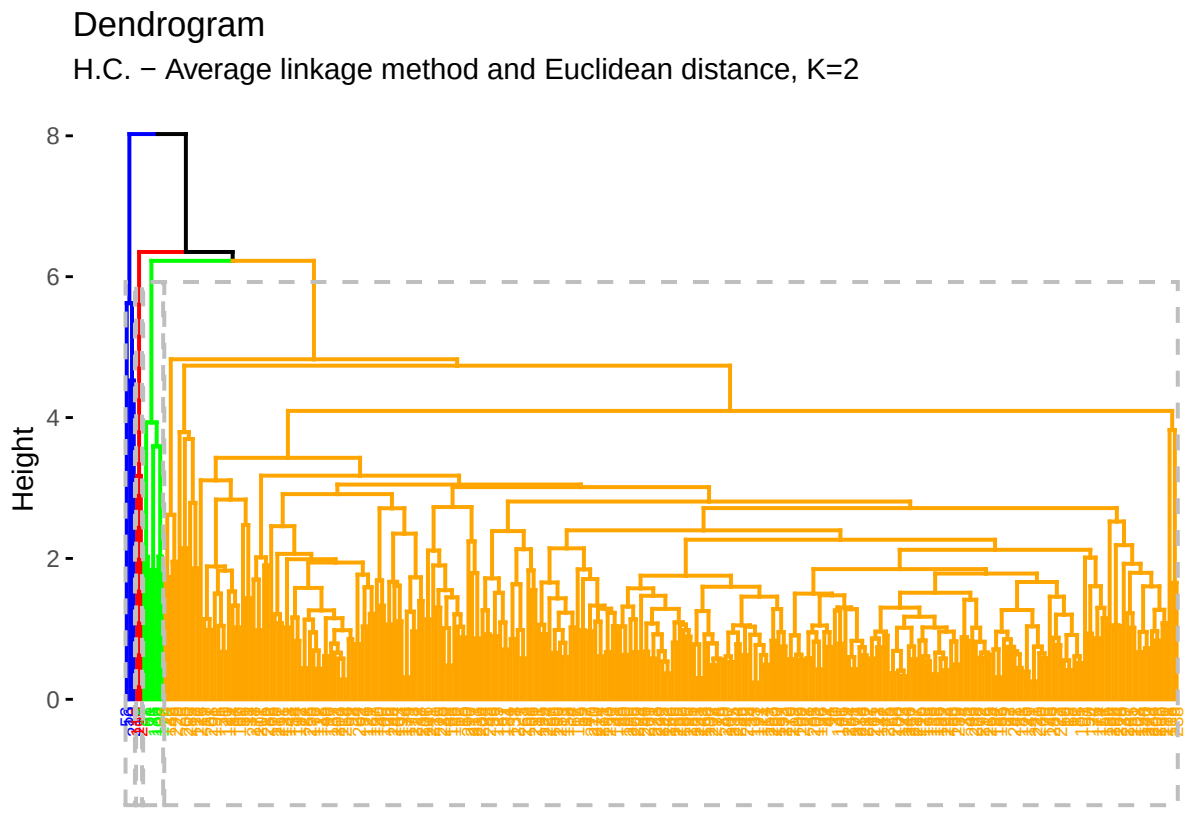
```
## group
##   1   2   3   4
## 288   6   3   2
```

```
group
```

```
##   [1] 1 2 1 1 1 1 1 1 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 2 1
##  [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 4 1
## [112] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1
## [149] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1
## [186] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 1
## [223] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [260] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [297] 4 1 1
```

In fact, cluster 1 contains more units than cluster 2, which contain only 2 units while cluster 1 contains 297 units respectively.

```
fviz_dend(hc, k = 4, cex = 0.5, k_colors = c(
"blue","red","green","orange"), color_labels_by_k = TRUE, rect = TRUE)+labs(title = "Dendrogram", subti
```

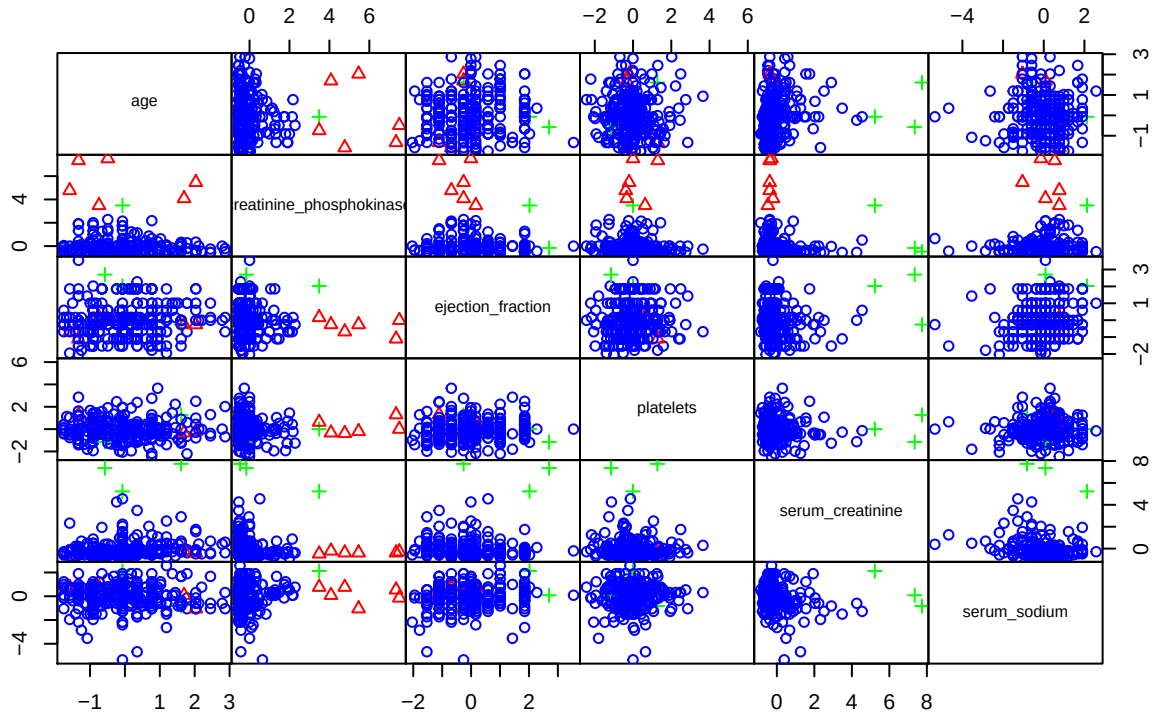



```
cor(dist.eucl, cophenetic(hc))
```

```
## [1] 0.8504769
```

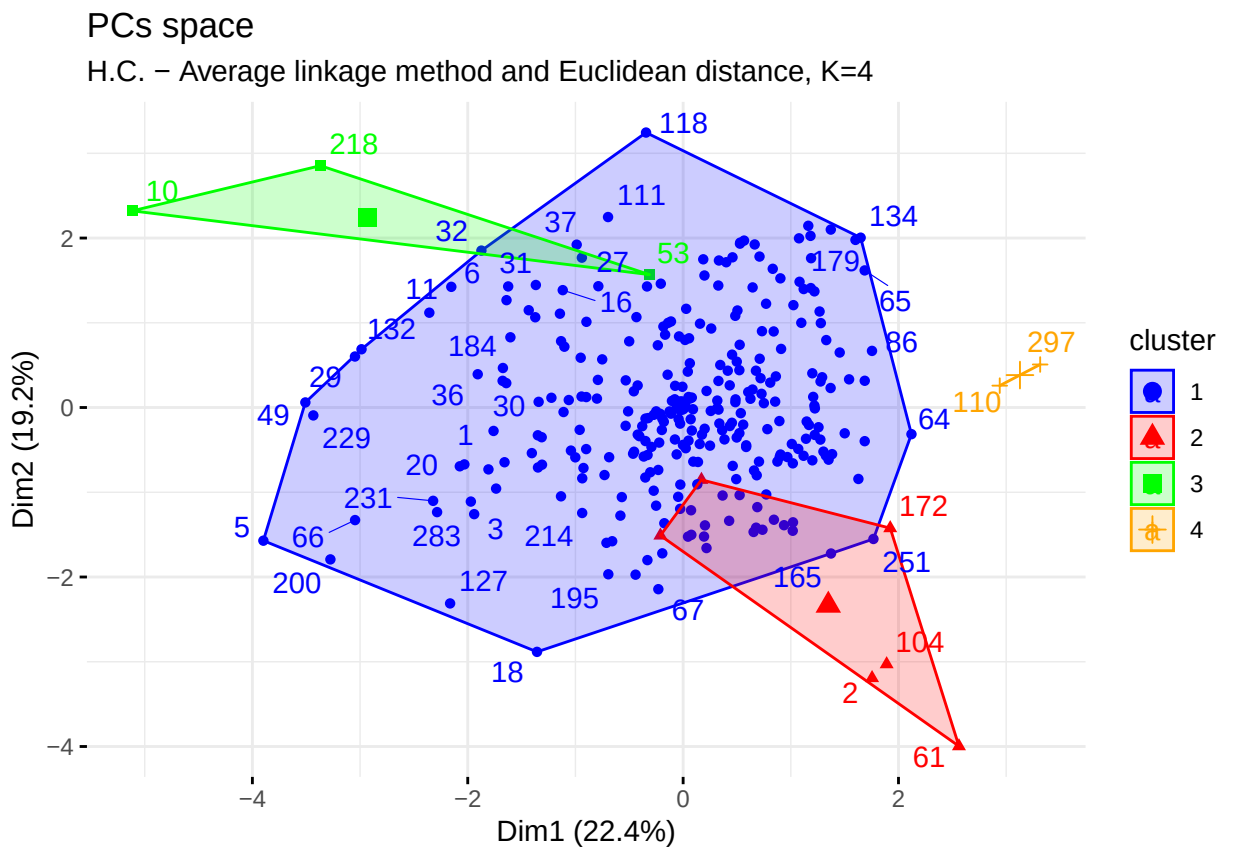
According to the result of "NbClust", the best number of clusters, applying hierarchical clustering method and using average linkage method and Euclidean distance is 4.

```
pairs(heart.attk_scale, gap=0, pch=group, cex.main= 0.7, main="Original space\nH.C.-Average linkage met
```




```
fviz_cluster(list(data = heart.attk_scale, cluster = group), palette = c("blue","red","green","orange")
             show.clust.aver = FALSE, ggtheme = theme_minimal())+
  labs(subtitle = "H.C. - Average linkage method and Euclidean distance, K=4", cex.sub= 0.5)
```

```
## Warning: ggrepel: 254 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



Cluster Validation Statistics:

Cluster validation is used to evaluate the goodness of clustering results.It can be categorized into 3 classes: internal, external and relative cluster validation.

Internal validation measures:

The internal cluster validation allows us to estimate the optimal number of clusters and to select the appropriate clustering algorithm, by means of two indices: 1. Silhouette Width 2. Dunn index.

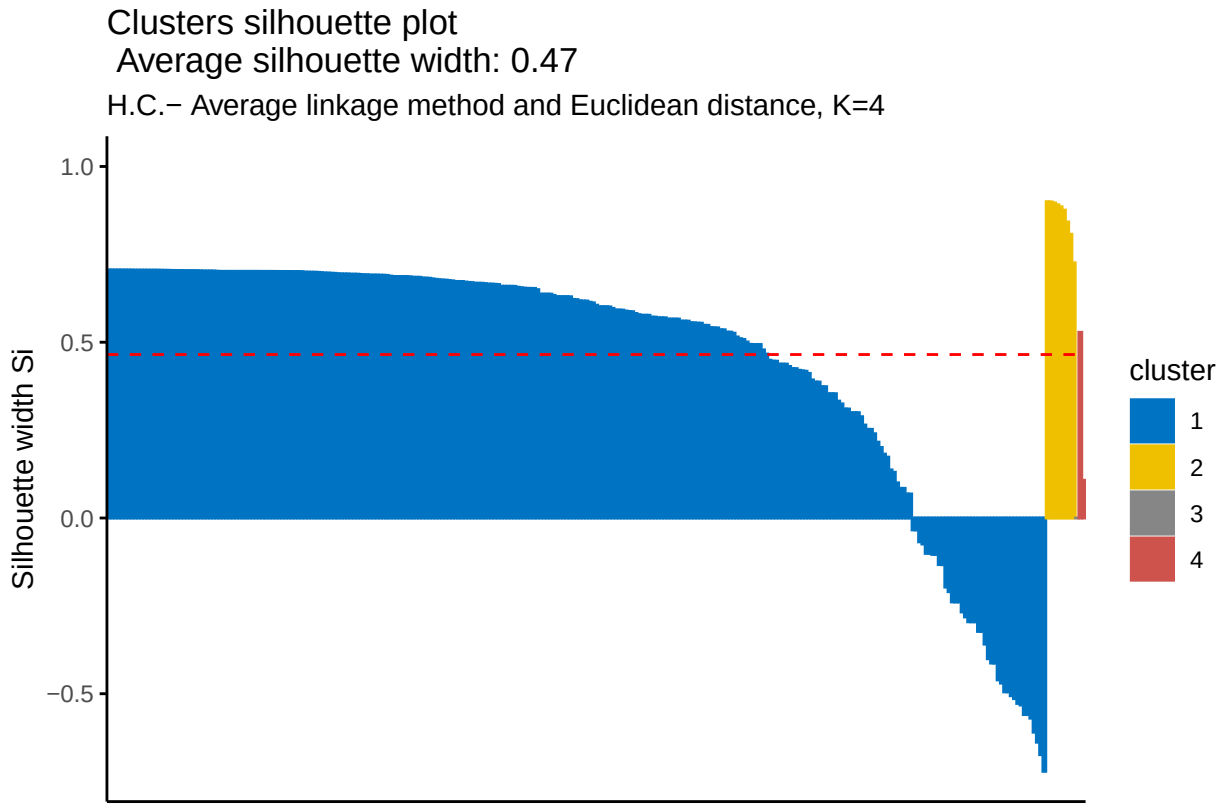
Silhouette width:

```
hclust<- eclust(heart.attk.sub,k=4 ,"hclust",hc_method  = "average",nboot = 50, hc_metric = "euclidean"
silinfo <- hclust$silinfo
silinfo$avg.width
```

```
## [1] 0.4650077
```

```
fviz_silhouette(hclust, palette = "jco", ggtheme = theme_classic())+
labs(subtitle = "H.C.- Average linkage method and Euclidean distance, K=4", cex.sub= 0.5)
```

```
##   cluster size ave.sil.width
## 1       1  287          0.46
## 2       2    9          0.86
## 3       3    1          0.00
## 4       4    2          0.32
```



The average silhouette width is 0.4. All clusters participates to the overall silhouette width have lower weights because there are few units in it. After looking at the plot,we can observed that in cluster 1 most of the units have a silhouette width lower the average one and some have a negative silhouette width; while in cluster 2 60% of the units have a silhouette width lower than the average one and 40% above the average one. And in clusters 3 and 4 most of the units have silhouette width lower the average one.

```
silinfo$clus.avg.widths
```

```
## [1] 0.4553245 0.8581591 0.0000000 0.3178655
```

As regard to the units with a negative silhouette value,this means that they are not in the right cluster,so we have to find their neighbor clusters:

```
sil <- hclust$silinfo$widths[, 1:3]
neg_sil_index_aver.eu<- which(sil[, "sil_width"] < 0)
sil[neg_sil_index_aver.eu, , drop = FALSE]
```

```
##     cluster neighbor   sil_width
## 51        1        2 -0.03400742
```

```
## 18        1        2 -0.03400783
## 116       1        2 -0.06789985
## 163       1        3 -0.07422813
## 122       1        2 -0.10088434
## 3         1        2 -0.10088601
## 48        1        3 -0.10372193
## 280       1        3 -0.10372642
## 8         1        3 -0.13253312
## 194       1        2 -0.13311127
## 56        1        3 -0.19677094
## 295       1        2 -0.20975925
## 29        1        2 -0.23895572
## 42        1        2 -0.23895658
## 187       1        2 -0.23898935
## 218       1        2 -0.26756042
## 275       1        2 -0.28158005
## 26        1        2 -0.29544933
## 74        1        2 -0.29545077
## 103       1        2 -0.29550239
## 265       1        2 -0.32280209
## 212       1        2 -0.32280211
## 213       1        3 -0.35905178
## 228       1        2 -0.40056387
## 49        1        2 -0.41288294
## 298       1        2 -0.41289744
## 13        1        2 -0.46065214
## 70        1        3 -0.46990999
## 268       1        2 -0.49475754
## 292       1        2 -0.49477157
## 178       1        2 -0.50586281
## 225       1        3 -0.51401970
## 224       1        2 -0.52770697
## 118       1        3 -0.53220896
## 7         1        2 -0.55932348
## 124       1        2 -0.55934946
## 195       1        2 -0.56964236
## 72        1        2 -0.60948826
## 66        1        2 -0.63794643
## 241       1        3 -0.67337828
## 288       1        3 -0.72081918
```

According to the silhouette method units(163,48,280,8,56,213,70,225,118,241,288) should be in cluster 3 and rest of the units from above table should be in cluster 2.

2.Dunn index:

```
library(fpc)
```

```
## Warning: package 'fpc' was built under R version 4.0.5
```

```
stats <- cluster.stats(dist(heart.attk_scale), hclust$cluster)
stats$dunn
```

```
## [1] 0.08806854
```

According to the Dunn index, the units are not clustered well enough.

External validation measures:

1.confusion matrix 2.correct Rand index 3.Meila's IV index

1.Confusion matrix:

According to the Confusion matrix the number of clusters is more than nominal values. the nominal variable can take 2 possible values and the clusters we found are also 2.

```
table(heart.attk$smoking, hclust$cluster)
```

```
##
##       1   2   3   4
##   0 195   7   0   1
##   1  92   2   1   1
```

Most of the smokers (n = 95) has been classified in cluster 1 while cluster 2 have 0 number of values. The same happened for non-smokers (n=197) classified in cluster 1 while cluster 2 has 6 values.

2.Correct Rand index:

```
smoking <- as.numeric(heart.attk$smoking)
stats<- cluster.stats(d = dist(heart.attk_scale), smoking, hclust$cluster)
stats$corrected.rand
```

```
## [1] 0.001385486
```

According to the Correct Rand Index, there is no agreement between the numerical value and the cluster solution. From -1 to +1, the agreement is very close to 0.

3.Meila's IV index:

```
stats$vi
```

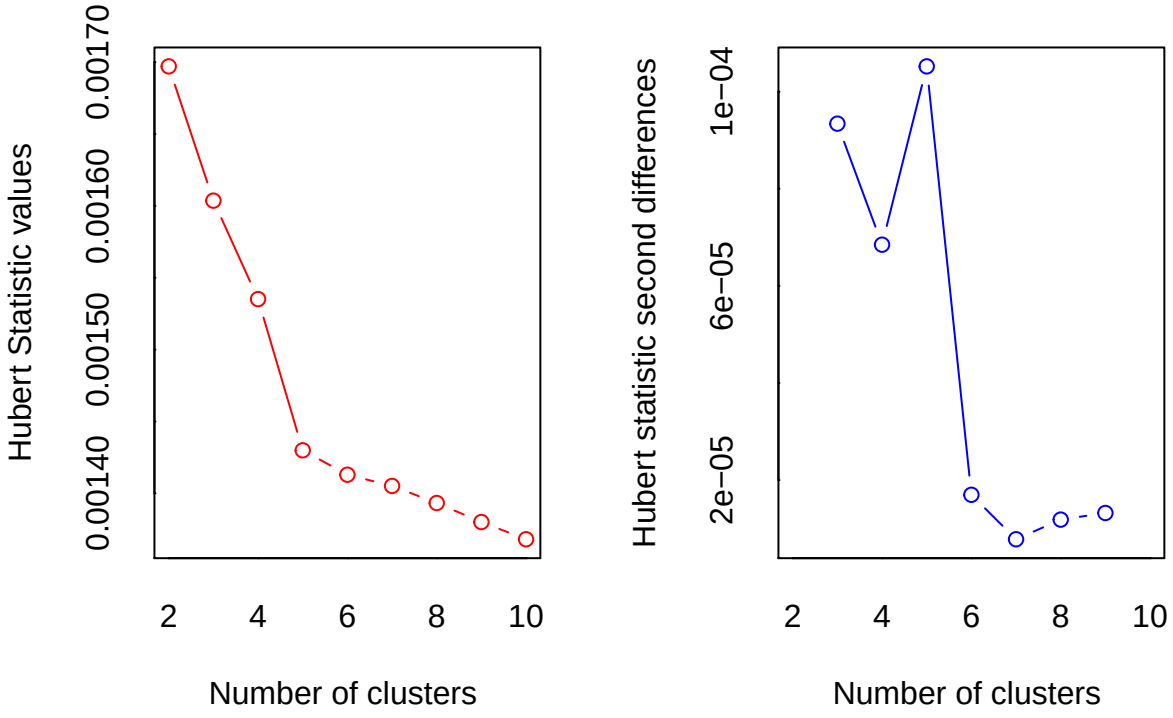
```
## [1] 0.8150391
```

Average linkage method and Manhattan distance:

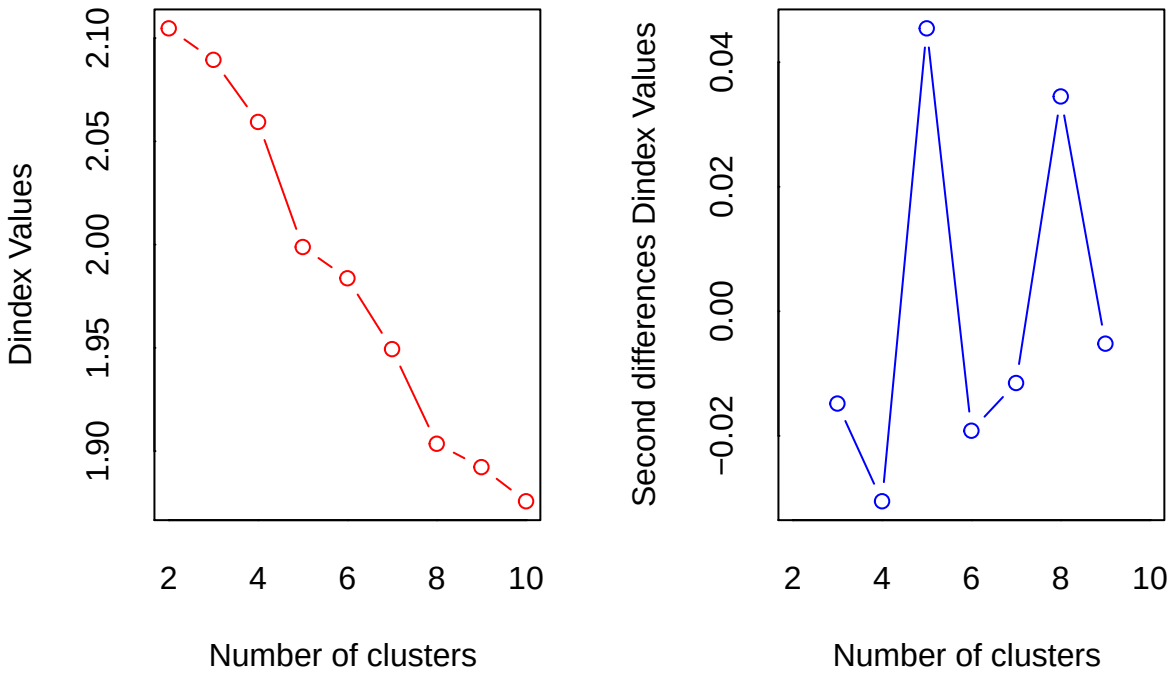
```
nb <- NbClust(heart.attk_scale, distance = "manhattan", min.nc = 2, max.nc = 10,
method = "average")
```

```
## Warning in pf(beale, pp, df2): NaNs produced
```

```
## [1] "Frey index : No clustering structure in this data set"
```




```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```

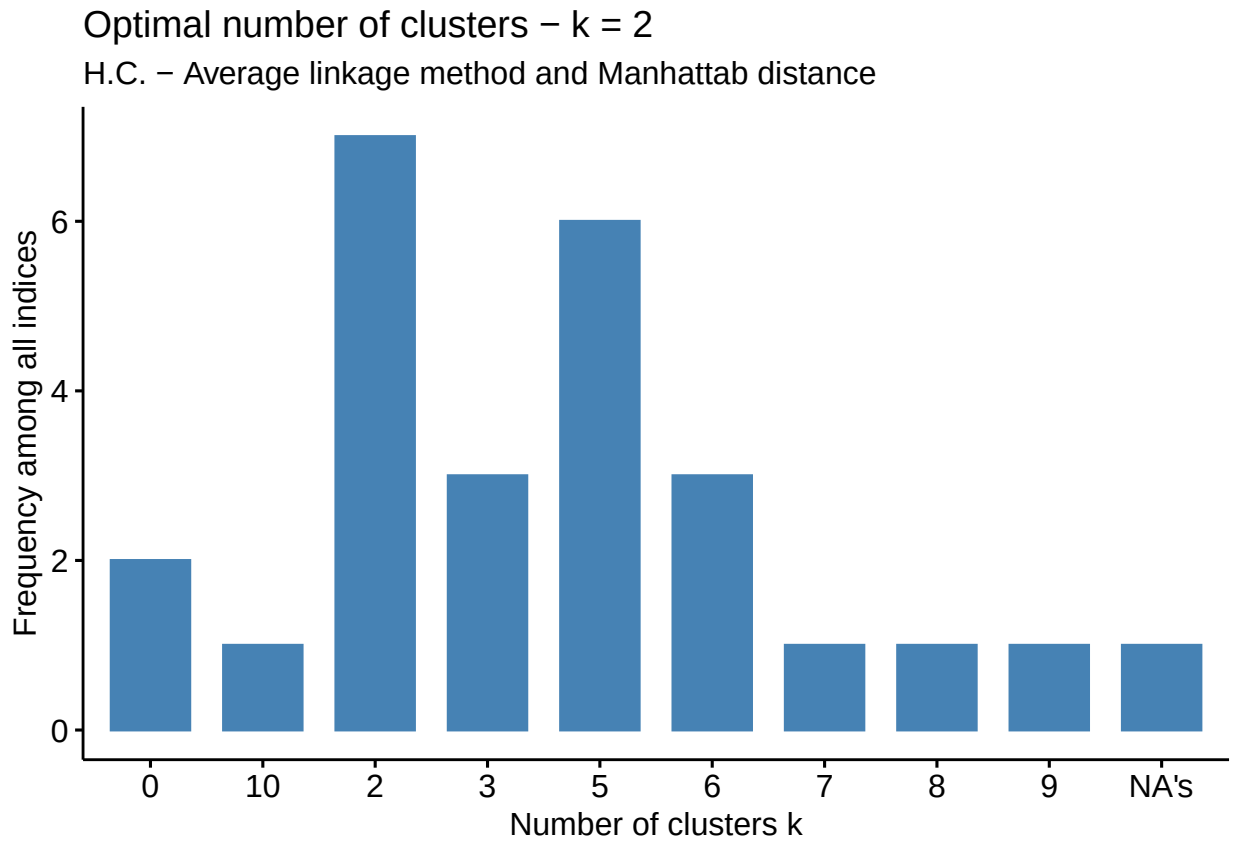



```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 7 proposed 2 as the best number of clusters
## * 3 proposed 3 as the best number of clusters
## * 6 proposed 5 as the best number of clusters
## * 3 proposed 6 as the best number of clusters
## * 1 proposed 7 as the best number of clusters
## * 1 proposed 8 as the best number of clusters
## * 1 proposed 9 as the best number of clusters
## * 1 proposed 10 as the best number of clusters
##
##                     ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  2
##
##
## *******************************************************************
```

```
fviz_nbclust(nb)+labs(subtitle = "H.C. - Average linkage method and Manhattab distance", cex.sub= 0.5)
```

```
## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
```

```
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") .viz_NbClust(x, print.summary, : the
## condition has length > 1 and only the first element will be used

## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") {: the condition has length > 1 and
## only the first element will be used

## Among all indices:
## ===================
## * 2 proposed  0 as the best number of clusters
## * 7 proposed  2 as the best number of clusters
## * 3 proposed  3 as the best number of clusters
## * 6 proposed  5 as the best number of clusters
## * 3 proposed  6 as the best number of clusters
## * 1 proposed  7 as the best number of clusters
## * 1 proposed  8 as the best number of clusters
## * 1 proposed  9 as the best number of clusters
## * 1 proposed  10 as the best number of clusters
## * 1 proposed  NA's as the best number of clusters
##
## Conclusion
## ==========================
## * According to the majority rule, the best number of clusters is  2 .
```




```
dist.man <- dist(heart.attk_scale, method = "manhattan")
hc <- hclust(dist.man, method = "average")
group <- cutree(hc, k=4)
table(group)
```

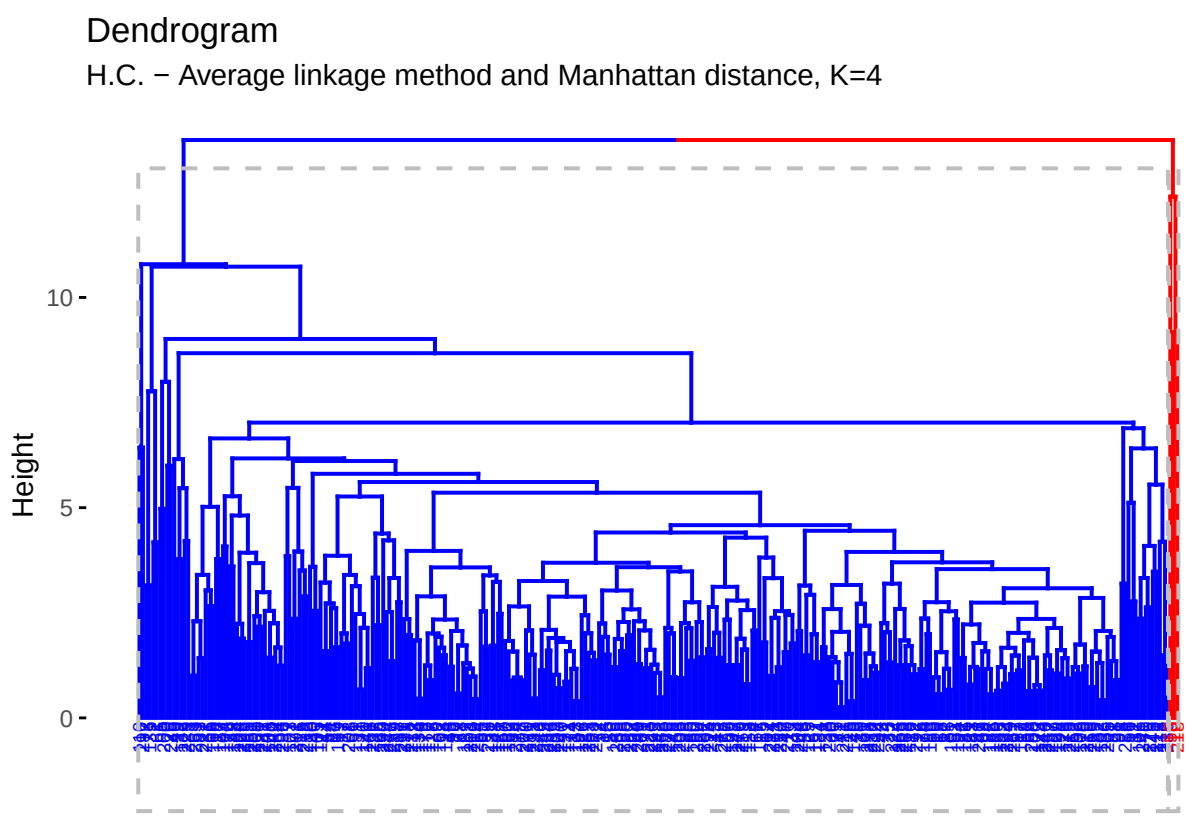
```
## group
##   1   2   3   4
## 294   2   1   2
```

```
group
```

```
##   [1] 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 4 1
## [112] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [149] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [186] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1
## [223] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [260] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [297] 4 1 1
```

```
fviz_dend(hc, k = 2, cex = 0.5, k_colors = c("blue", "red","green","orange"), color_labels_by_k = TRUE,
subtitle = "H.C. - Average linkage method and Manhattan distance, K=4", cex.subtitle= 0.5)
```

```
## Warning in get_col(col, k): Length of color vector was longer than the number of
## clusters - first k elements are used
```

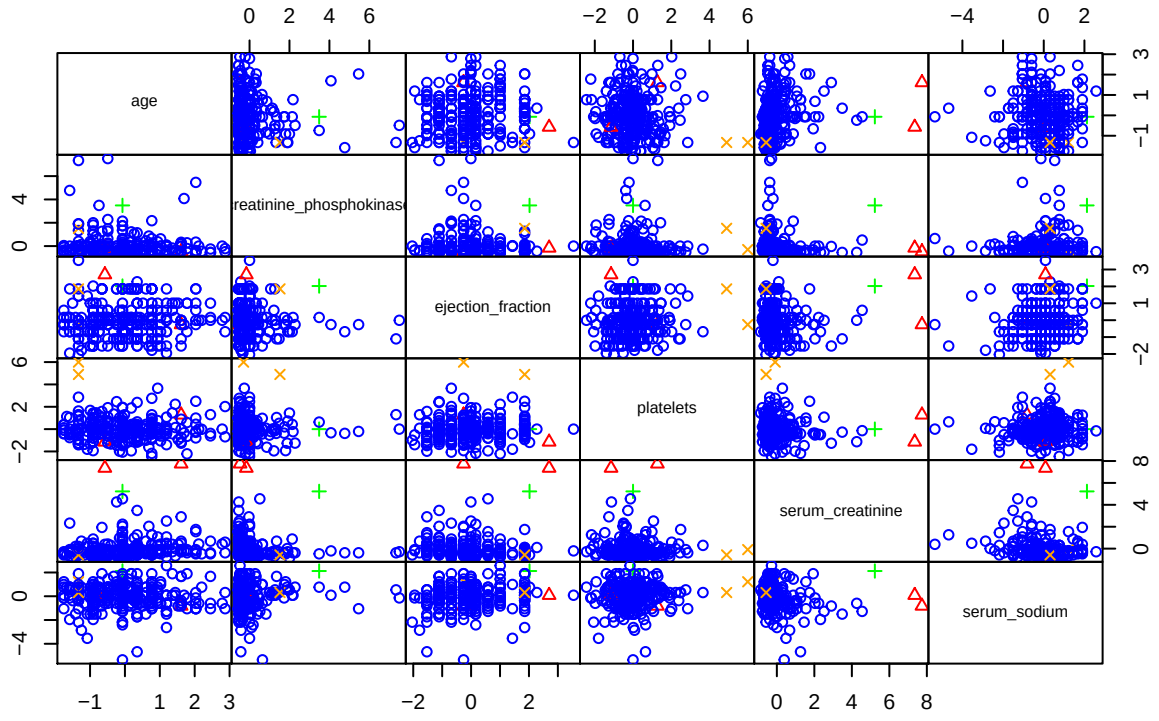



```
cor(dist.man, cophenetic(hc))
```

```
## [1] 0.7936353
```

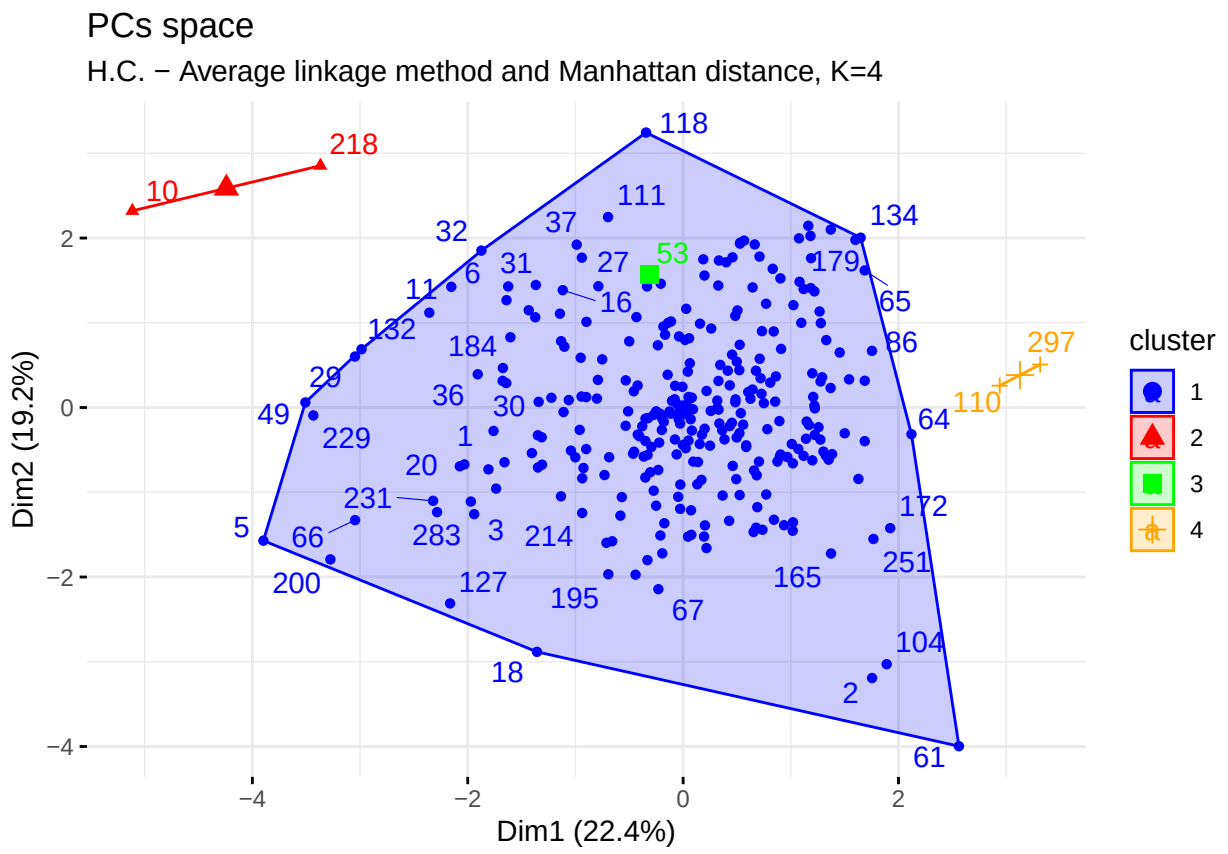
According to the result of "NbClust", the best number of clusters, applying hierarchical clustering method and using average linkage method and Manhattan distance is 4.

```
pairs(heart.attk_scale, gap=0, pch=group, cex.main= 0.7, main="Original space\nH.C.-Average linkage met
```




```
fviz_cluster(list(data = heart.attk_scale, cluster = group), palette = c("blue", "red","green","orange"),
subtitle = "H.C. - Average linkage method and Manhattan distance, K=4", cex.sub= 0.5)
```

```
## Warning: ggrepel: 254 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



To examine the goodness of clustering algorithm results, we have to analyst internal and external validation measures.

Internal validation measures:

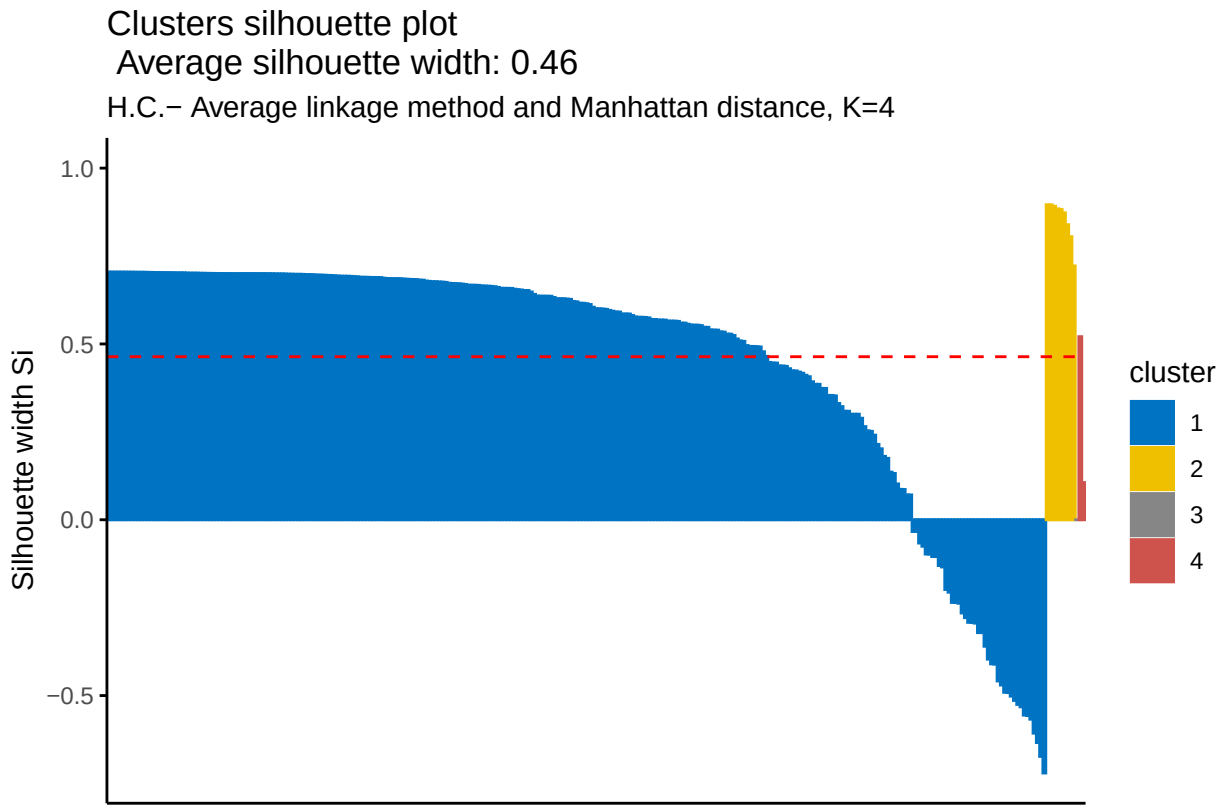
1.silhouette width

```
hclust<- eclust(heart.attk.sub,k=4 ,"hclust",hc_method = "average",nboot = 50, hc_metric = "manhattan")
silinfo <- hclust$silinfo
silinfo$avg.width
```

```
## [1] 0.4636361
```

```
fviz_silhouette(hclust, palette = "jco", ggtheme = theme_classic())+
labs(subtitle = "H.C.- Average linkage method and Manhattan distance, K=4", cex.sub= 0.5)
```

```
##   cluster size ave.sil.width
## 1       1  287          0.45
## 2       2    9          0.85
## 3       3    1          0.00
## 4       4    2          0.31
```




```
silinfo$clus.avg.widths
```

```
## [1] 0.4540470 0.8543543 0.0000000 0.3132482
```

```
sil <- hclust$silinfo$widths[, 1:3]
neg_sil_index_aver.eu<- which(sil[, "sil_width"] < 0)
sil[neg_sil_index_aver.eu, , drop = FALSE]
```

```
##     cluster neighbor   sil_width
## 51        1        2 -0.03345492
## 18        1        2 -0.03346152
## 116       1        2 -0.06627908
## 163       1        3 -0.07538733
## 122       1        2 -0.09759387
## 3         1        2 -0.09778148
## 280       1        3 -0.10470671
## 48        1        3 -0.10477115
## 194       1        2 -0.13027349
## 8         1        3 -0.13440890
## 56        1        3 -0.19846744
## 295       1        2 -0.20610372
## 29        1        2 -0.23520798
## 42        1        2 -0.23534540
## 187       1        2 -0.23754798
## 218       1        2 -0.26529467
```

```
## 275       1        2 -0.27847367
## 26        1        2 -0.29182706
## 74        1        2 -0.29217383
## 103       1        2 -0.29414449
## 212       1        2 -0.32101848
## 265       1        2 -0.32108408
## 213       1        3 -0.35987391
## 228       1        2 -0.39651354
## 298       1        2 -0.40994998
## 49        1        2 -0.41066885
## 13        1        2 -0.45855466
## 70        1        3 -0.47006412
## 268       1        2 -0.49079981
## 292       1        2 -0.49158901
## 178       1        2 -0.50163214
## 225       1        3 -0.51396854
## 224       1        2 -0.52534332
## 118       1        3 -0.53212061
## 7         1        2 -0.55568423
## 124       1        2 -0.55694244
## 195       1        2 -0.56716698
## 72        1        2 -0.60702015
## 66        1        2 -0.63363695
## 241       1        3 -0.67306161
## 288       1        3 -0.72025318
```

The value of average silhouette width indicates that in average the units are well enough clustered. In particular, in cluster 1 (blue cluster) the units are on average the same silhouette value with respect to the silhouette width; in cluster 2 (the yellow one) also the units are on average the same silhouette value with respect to silhouette width and same in cluster 3. According to this index, 41 units that belong to cluster 1 are not well clustered: from which 11 units (163,48,280,8,56,213,70,225,118,241,288) should belong to cluster 3 and 30 units should be belong to cluster 2.

Dunn index:

```
stats <- cluster.stats(dist(heart.attk_scale), hclust$cluster)
stats$dunn
```

```
## [1] 0.08806854
```

According to the Dunn index, the units are not clustered well enough.

External validation measures:

confusion matrix:

As we know that according to the Confusion matrix, the number of clusters is equal to nominal values.

```r
table(heart.attk$smoking, hclust$cluster)
```

```
##
##       1   2   3   4
##   0 195   7   0   1
##   1  92   2   1   1
```

For smokers smoking data classified in clusters 2,3 and 4 while cluster 1 has 92 values, safe to say data are not well balanced in each cluster. And for non-smokers smoking data classified in clusters 1,2 and 4 which is seems to be not balanced in each cluster.

Correct Rand Index:

```r
smoking <- as.numeric(heart.attk$smoking)
stats<- cluster.stats(d = dist(heart.attk_scale), smoking, hclust$cluster)
stats$corrected.rand
```

```
## [1] 0.001385486
```

According to the Correct Rand Index, there is no agreement between the numerical values and the cluster solution. From -1 to +1, the agreement is very close to 0.

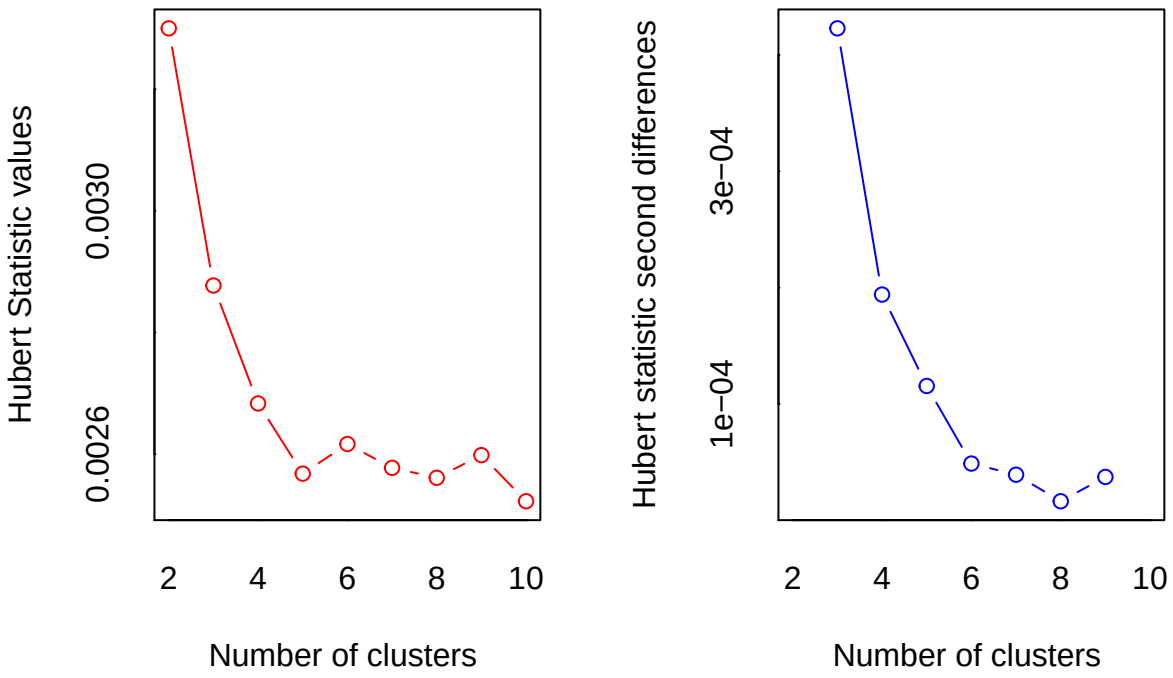
Meila's VI Index:

```r
stats$vi
```

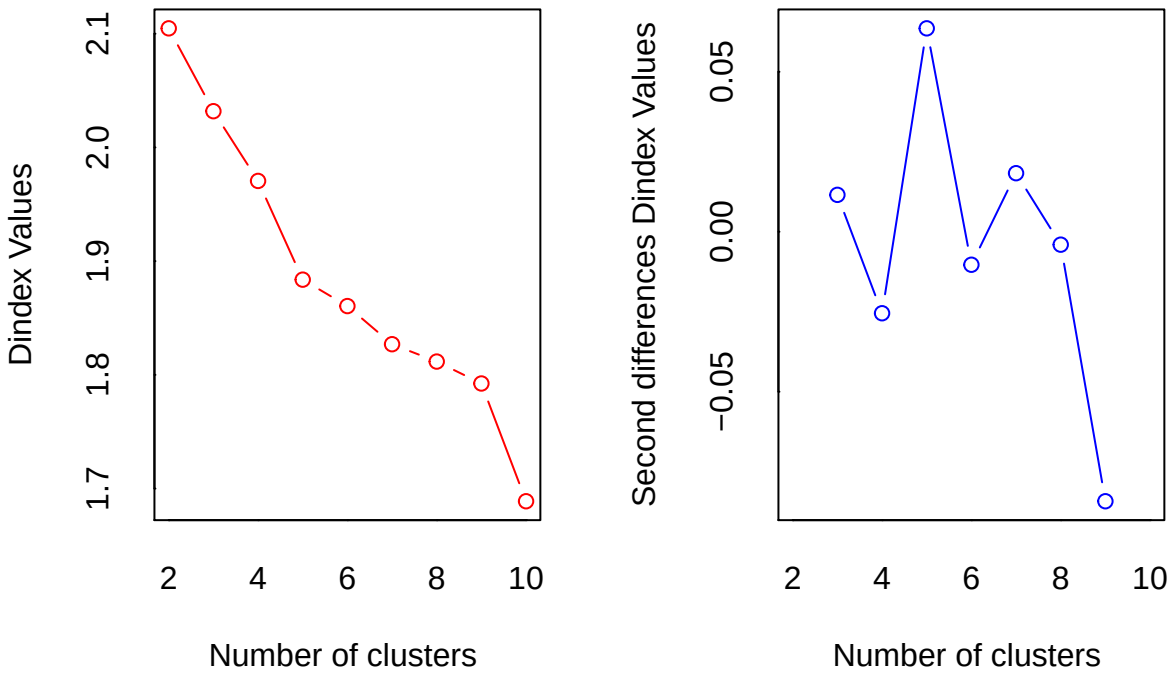
```
## [1] 0.8150391
```

Complete linkage method and Euclidean distance:

```r
nb <- NbClust(heart.attk_scale, distance = "euclidean", min.nc = 2, max.nc = 10,
method = "complete")
```




```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```

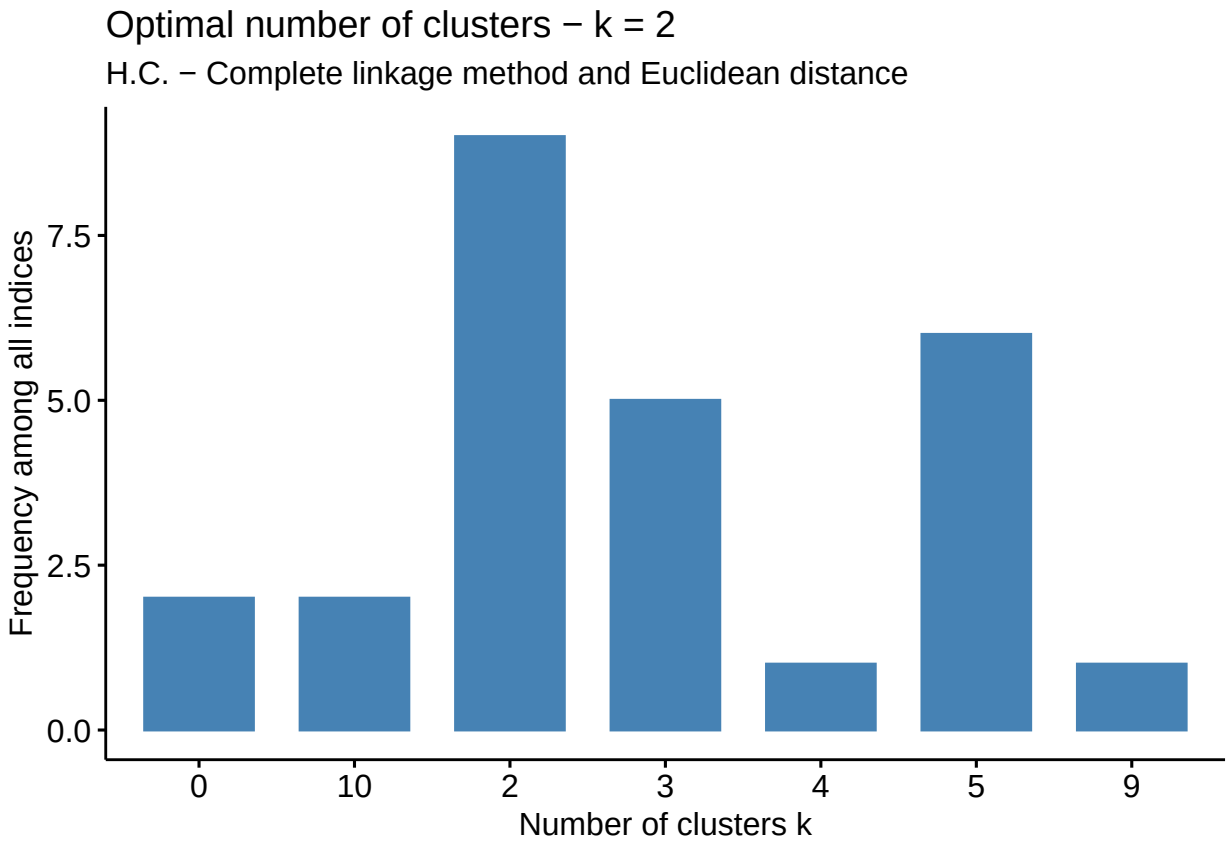



```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 9 proposed 2 as the best number of clusters
## * 5 proposed 3 as the best number of clusters
## * 1 proposed 4 as the best number of clusters
## * 6 proposed 5 as the best number of clusters
## * 1 proposed 9 as the best number of clusters
## * 2 proposed 10 as the best number of clusters
##
##                   ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  2
##
##
## *******************************************************************
```

```
fviz_nbclust(nb)+labs(
subtitle = "H.C. - Complete linkage method and Euclidean distance",
cex.sub= 0.5)
```

```
## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
```

```
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") .viz_NbClust(x, print.summary, : the
## condition has length > 1 and only the first element will be used

## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") {: the condition has length > 1 and
## only the first element will be used

## Among all indices:
## ==================
## * 2 proposed  0 as the best number of clusters
## * 9 proposed  2 as the best number of clusters
## * 5 proposed  3 as the best number of clusters
## * 1 proposed  4 as the best number of clusters
## * 6 proposed  5 as the best number of clusters
## * 1 proposed  9 as the best number of clusters
## * 2 proposed  10 as the best number of clusters
##
## Conclusion
## =========================
## * According to the majority rule, the best number of clusters is  2 .
```



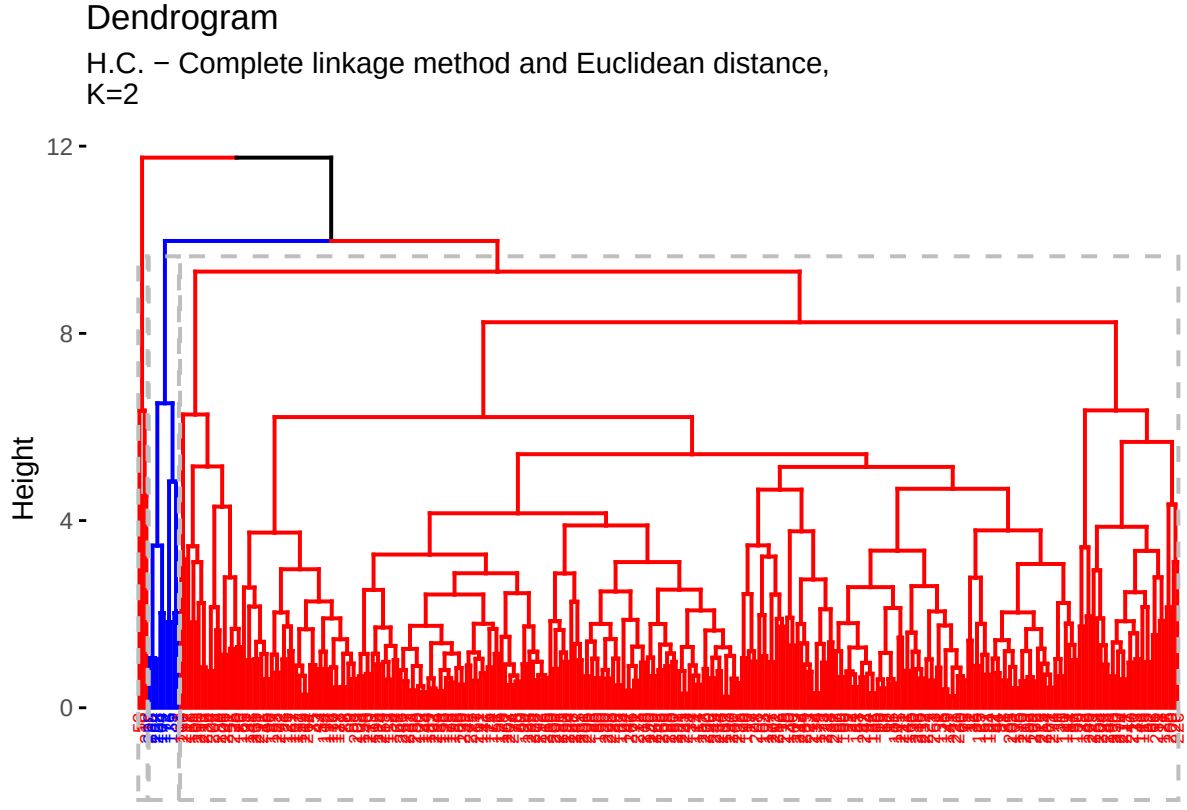
```
hc <- hclust(dist.eucl, method = "complete")
group <- cutree(hc, k=2)
```

```
table(group)
```

```
## group
##   1   2
## 296   3
```

```
fviz_dend(hc, k = 3, cex = 0.5, k_colors = c("red", "blue"),
color_labels_by_k = TRUE, rect = TRUE)+labs(title = "Dendrogram",
subtitle = "H.C. - Complete linkage method and Euclidean distance,
K=2",
cex.subtitle= 0.5)
```

```
## Warning in get_col(col, k): Length of color vector was shorter than the number
## of clusters - color vector was recycled
```

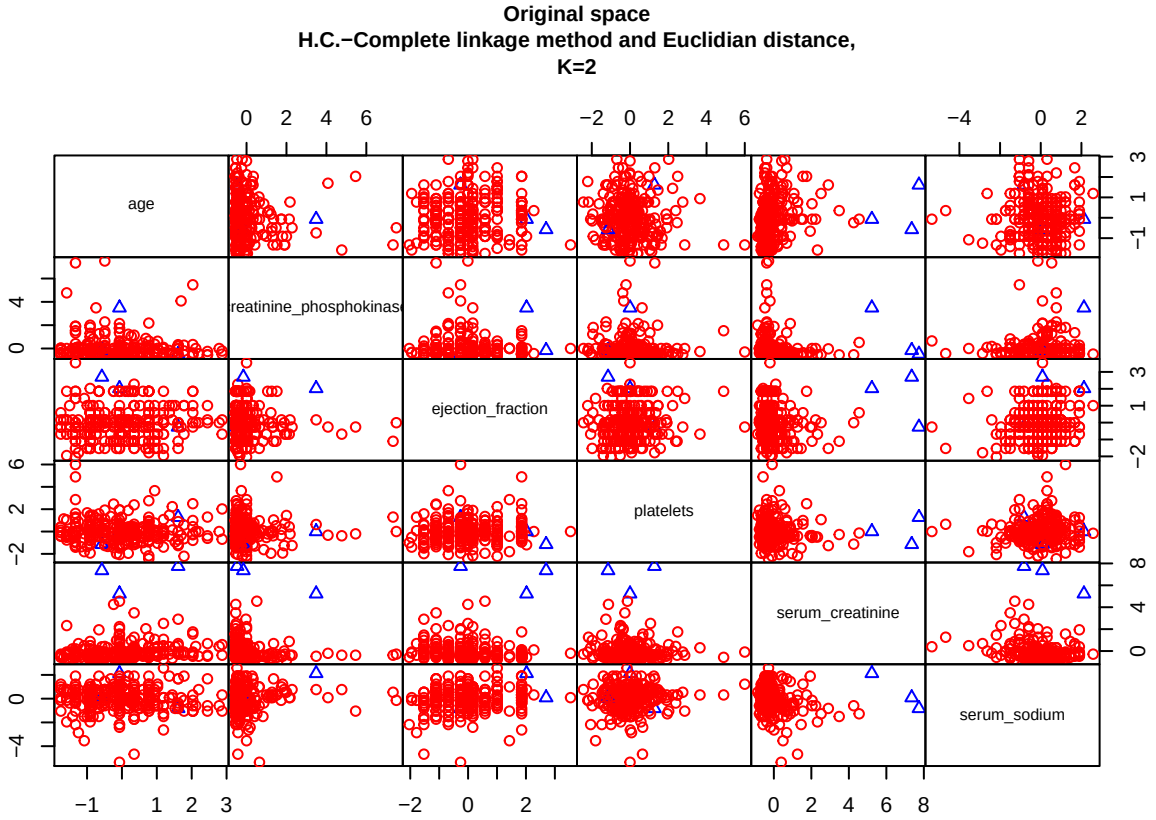



```
cor(dist.eucl, cophenetic(hc))
```

```
## [1] 0.6667467
```

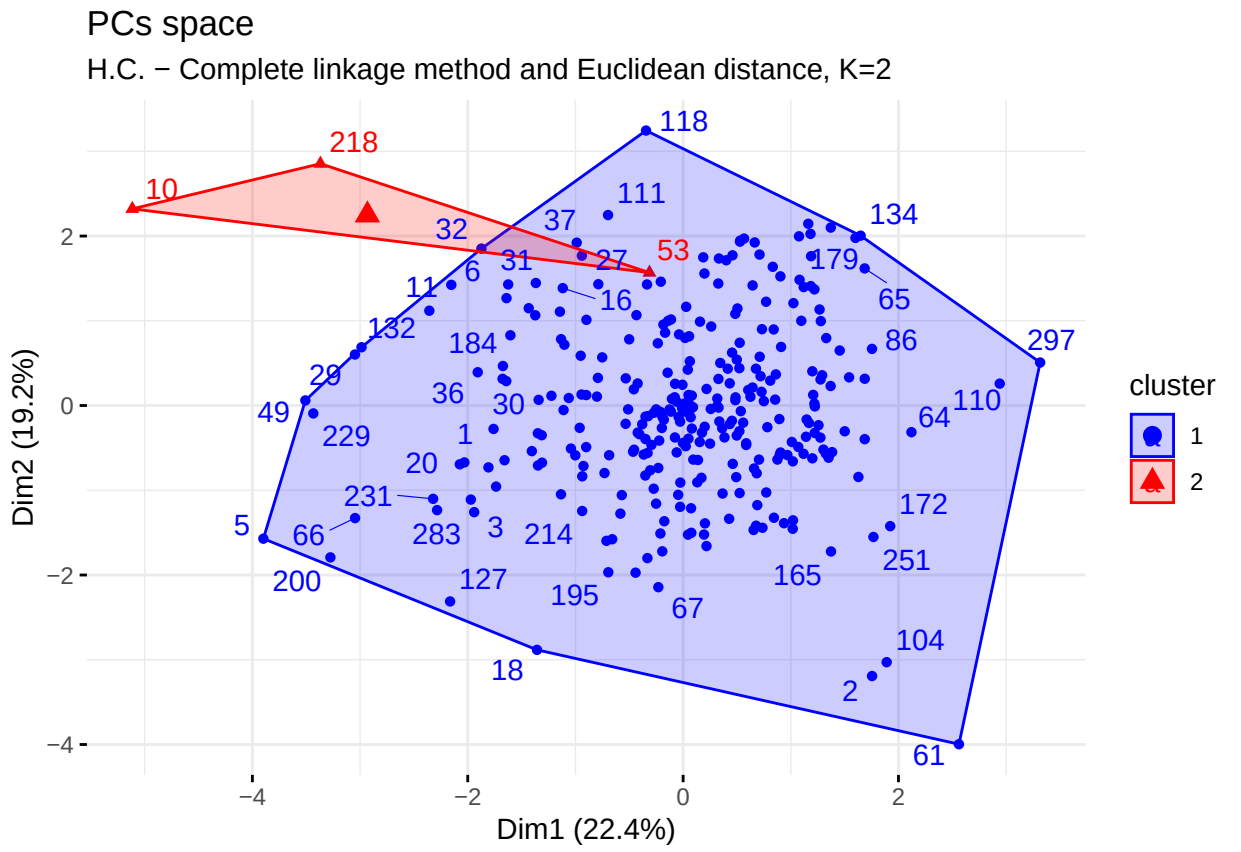
According to the result of "NbClust", the best number of clusters, applying hierarchical clustering method and using complete linkage method and Euclidean distance is 2.

```
pairs(heart.attk_scale, gap=0, pch=group, cex.main= 0.7,
main="Original space\nH.C.-Complete linkage method and Euclidian distance,
K=2",
col=c("red", "blue")[group])
```




```
fviz_cluster(list(data = heart.attk_scale, cluster = group),
palette = c("blue", "red"), ellipse.type = "convex",
main="PCs space", repel = TRUE, show.clust.aver = FALSE,
ggtheme = theme_minimal())+
labs(subtitle = "H.C. - Complete linkage method and Euclidean distance, K=2",
cex.sub= 0.5)
```

```
## Warning: ggrepel: 254 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



To examine the goodness of clustering algorithm results, we have to analyst internal and external validation measures.

Internal validation measures:

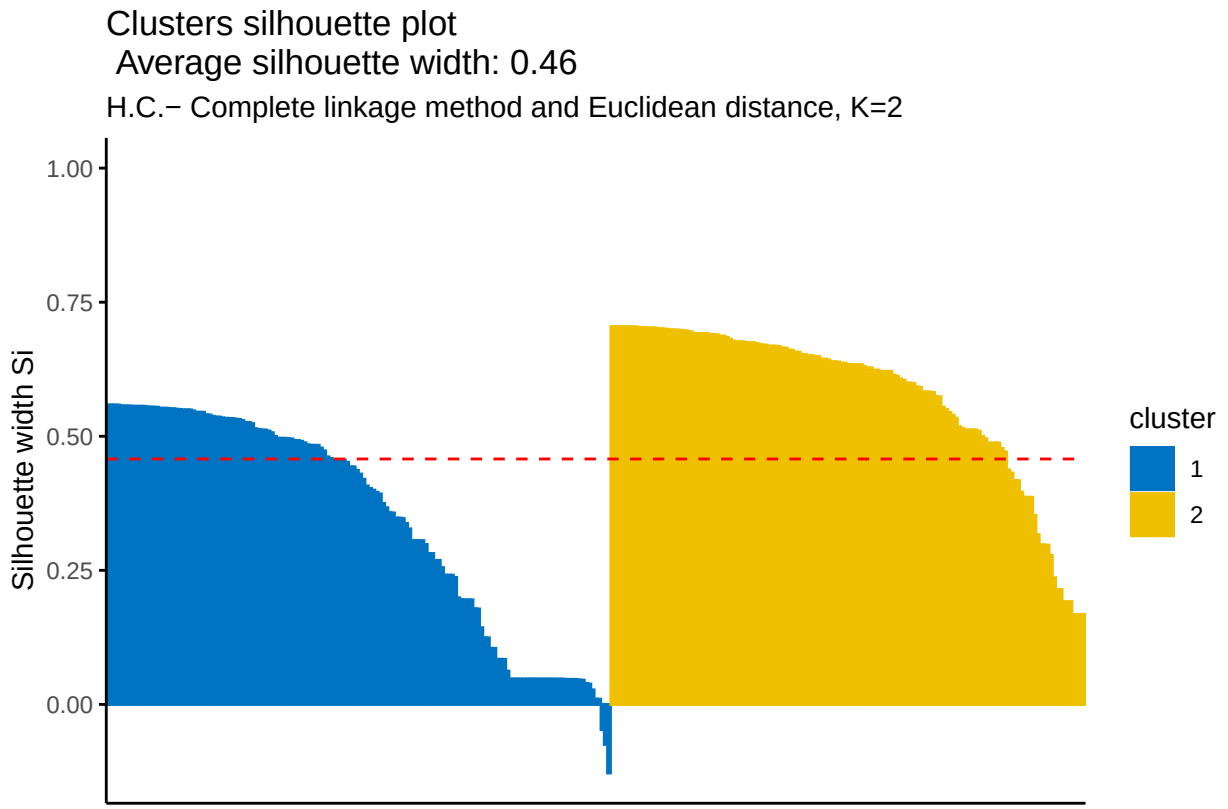
1.Silhouette width:

```
hclust<- eclust(heart.attk.sub,k=2 ,"hclust",hc_method = "complete",nboot = 50, hc_metric = "euclidean"
silinfo <- hclust$silinfo
silinfo$avg.width
```

```
## [1] 0.4576278
```

```
fviz_silhouette(hclust, palette = "jco", ggtheme = theme_classic())+
labs(subtitle = "H.C.- Complete linkage method and Euclidean distance, K=2",
cex.sub= 0.5)
```

```
##   cluster size ave.sil.width
## 1       1  154          0.34
## 2       2  145          0.58
```




```
silinfo$clus.avg.widths
```

```
## [1] 0.3445371 0.5777379
```

```
sil <- hclust$silinfo$widths[, 1:3]
neg_sil_index_aver.eu<- which(sil[, "sil_width"] < 0)
sil[neg_sil_index_aver.eu, , drop = FALSE]
```

```
##     cluster neighbor   sil_width
## 193       1        2 -0.04707262
## 198       1        2 -0.07466558
## 239       1        2 -0.12791572
```

The value of average silhouette width indicates that in average the units are well enough clustered. In particular, in cluster 1 (blue cluster) the units are on average the same silhouette value with respect to the silhouette width; in cluster 2 (the yellow one) also the units are on average the same silhouette value with respect to silhouette width. According to this index, 3 units (193,198,239)that belong to cluster 1 are not well clustered: they should belong to cluster 2.

Dunn index:

```
stats <- cluster.stats(dist(heart.attk_scale), hclust$cluster)
stats$dunn
```

```
## [1] 0.0333656
```

According to the Dunn index, the units are not clustered well enough.

#External validation measures:

#1.Confusion matrix:

```
table(heart.attk$smoking, hclust$cluster)
```

```
##
##       1   2
##   0 104  99
##   1  50  46
```

Most of the smokers (n = 50) has been classified in cluster 1 while cluster 2 have 46 number of values. The same happened for non-smokers (n=104) classified in cluster 1 while cluster 2 has 99 values.

Correct Rand Index:

```
smoking <- as.numeric(heart.attk$smoking)
stats<- cluster.stats(d = dist(heart.attk_scale), smoking, hclust$cluster)
stats$corrected.rand
```

```
## [1] -0.003037386
```

According to the Correct Rand Index, there is no agreement between the numerical values and the cluster solution. From -1 to +1, the agreement is very close to 0.

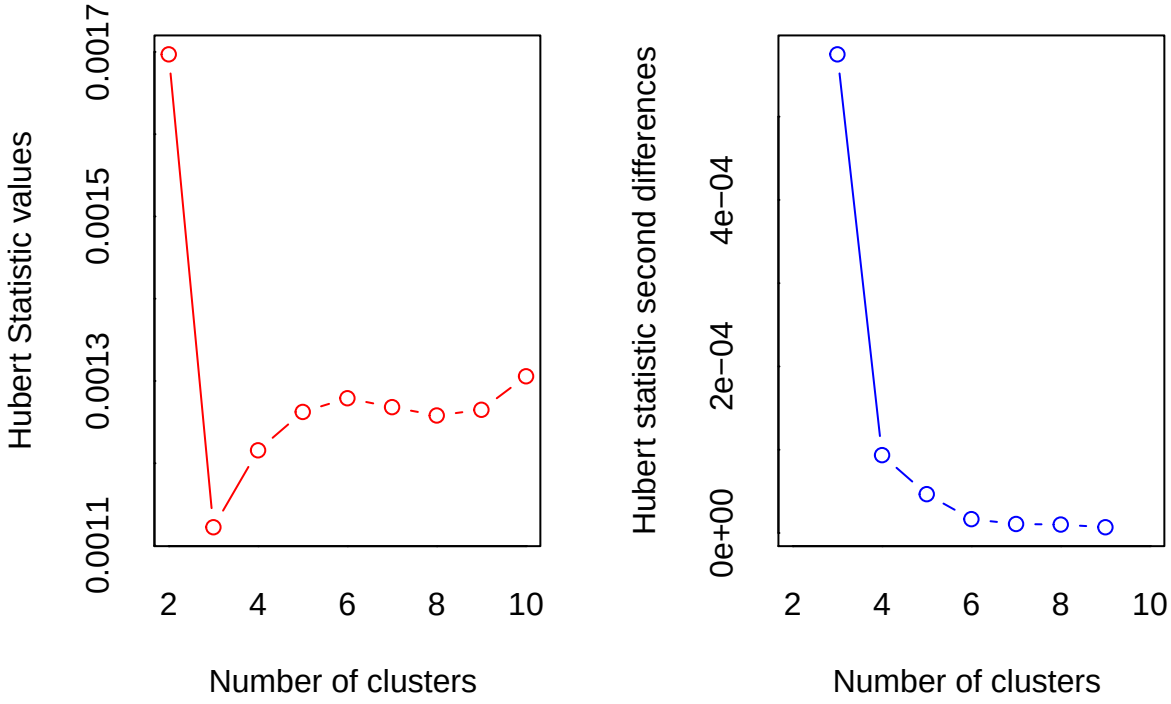
Meila's VI Index:

```
stats$vi
```

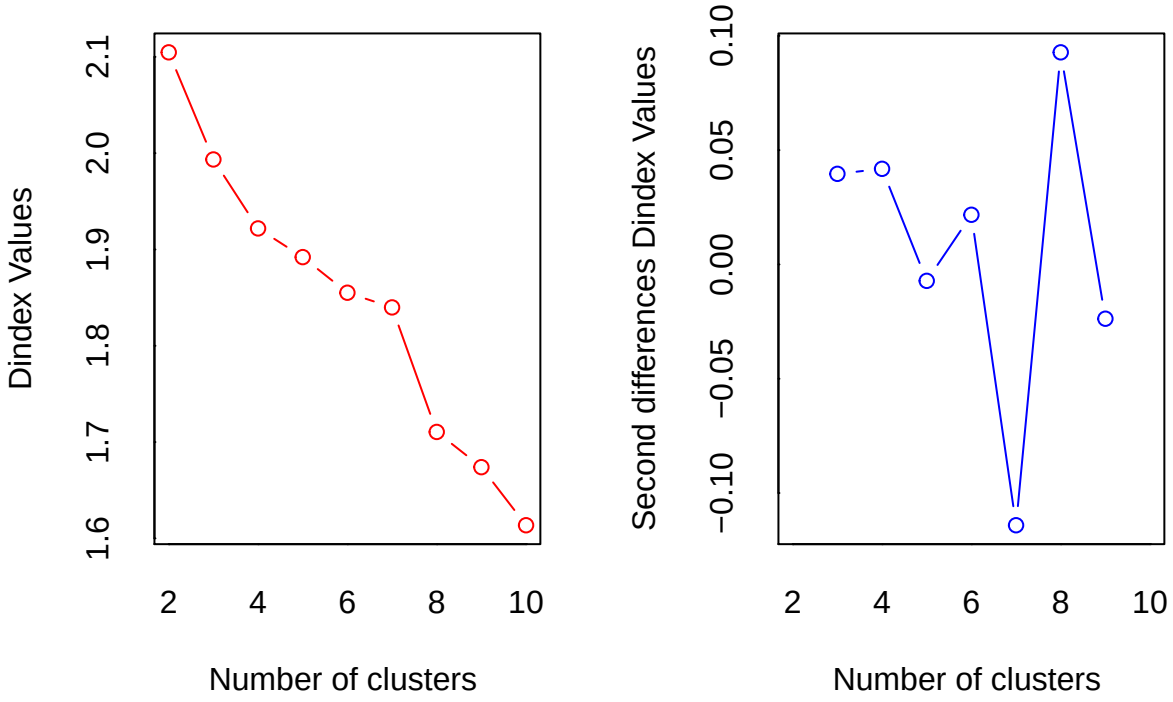
```
## [1] 1.320304
```

Complete linkage method and Manhattan distance:

```
nb <- NbClust(heart.attk_scale, distance = "manhattan", min.nc = 2, max.nc = 10,
method = "complete")
```




```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```

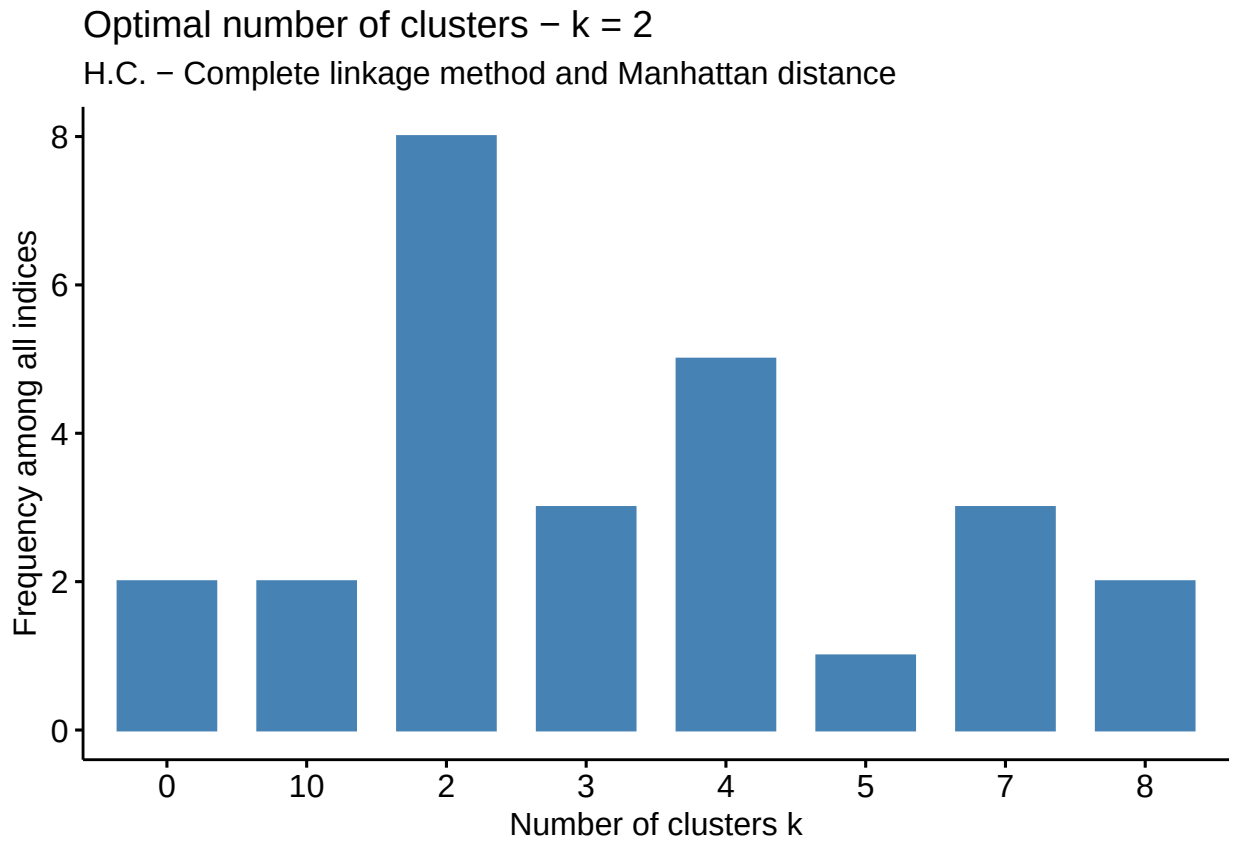



```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 8 proposed 2 as the best number of clusters
## * 3 proposed 3 as the best number of clusters
## * 5 proposed 4 as the best number of clusters
## * 1 proposed 5 as the best number of clusters
## * 3 proposed 7 as the best number of clusters
## * 2 proposed 8 as the best number of clusters
## * 2 proposed 10 as the best number of clusters
##
##                     ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  2
##
##
## *******************************************************************
```

```
fviz_nbclust(nb)+labs(subtitle = "H.C. - Complete linkage method and Manhattan distance", cex.sub= 0.5)
```

```
## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
```

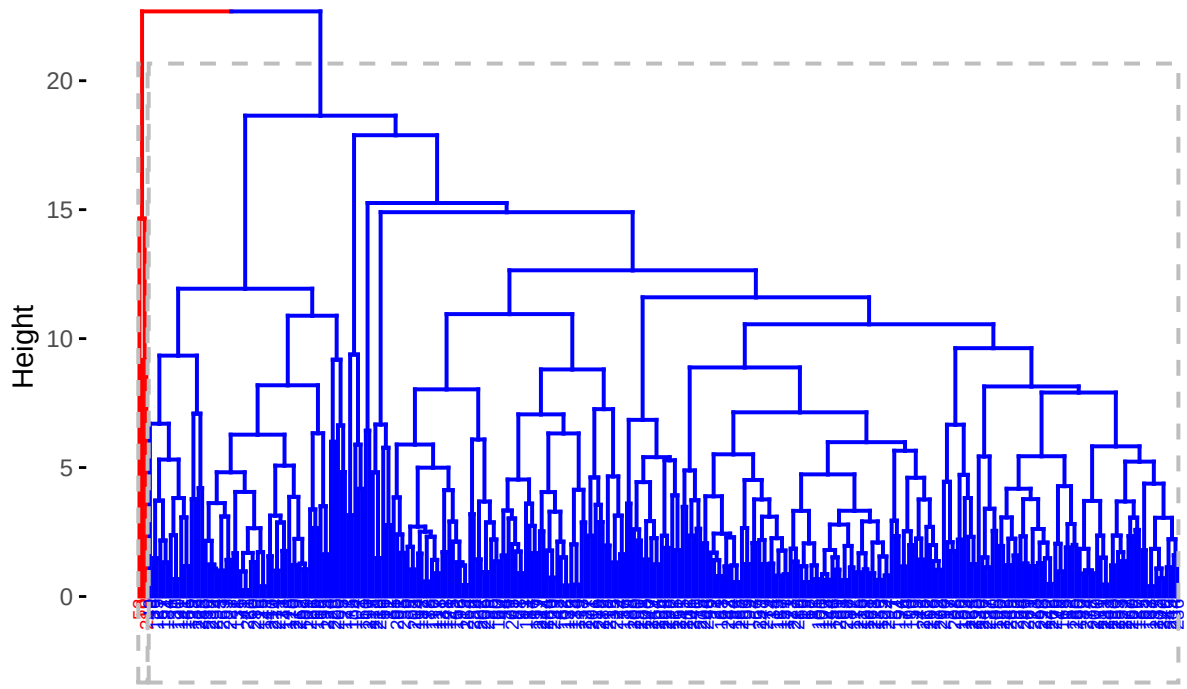
```
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") .viz_NbClust(x, print.summary, : the
## condition has length > 1 and only the first element will be used

## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") {: the condition has length > 1 and
## only the first element will be used

## Among all indices:
## ===================
## * 2 proposed  0 as the best number of clusters
## * 8 proposed  2 as the best number of clusters
## * 3 proposed  3 as the best number of clusters
## * 5 proposed  4 as the best number of clusters
## * 1 proposed  5 as the best number of clusters
## * 3 proposed  7 as the best number of clusters
## * 2 proposed  8 as the best number of clusters
## * 2 proposed  10 as the best number of clusters
##
## Conclusion
## =========================
## * According to the majority rule, the best number of clusters is  2 .
```




```
dist.man <- dist(heart.attk_scale, method = "manhattan")
hc <- hclust(dist.man, method = "complete")
group <- cutree(hc, k=2)
table(group)
```

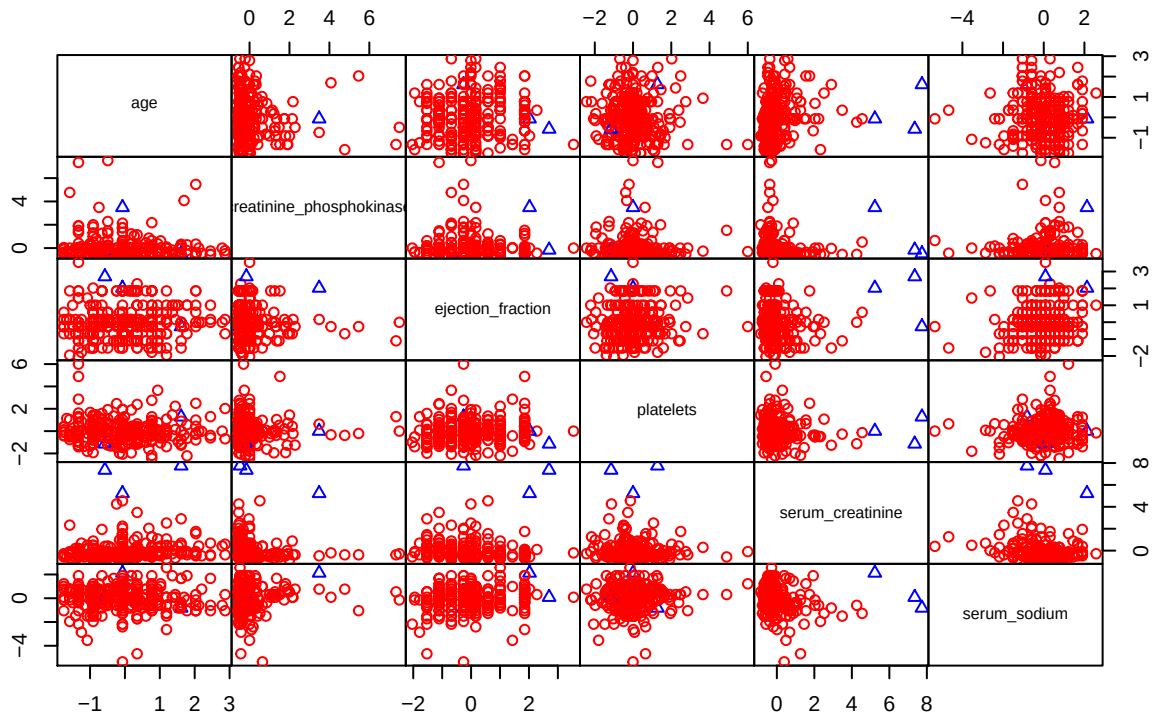
```
## group
##   1   2
## 296   3
```

```
fviz_dend(hc, k = 2, cex = 0.5, k_colors = c("red", "blue"),
r_labels_by_k = TRUE, rect = TRUE)+labs(title = "Dendrogram",
subtitle = "H.C. - Complete linkage method and Manhattan distance, K=2",cex.subtitle= 0.5)
```



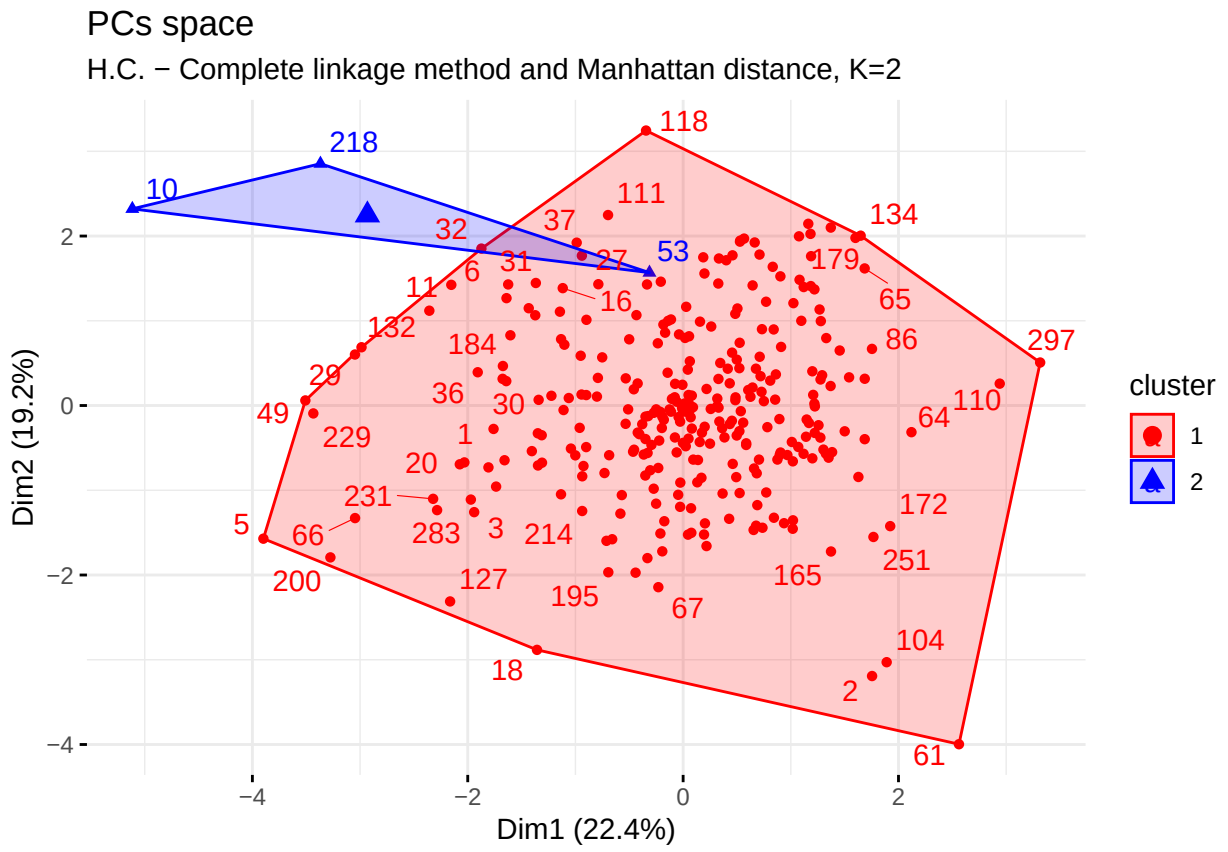
According to the result of "NbClust", the best number of clusters, applying hierarchical clustering method and using complete linkage method and Manhattan distance is 2.

```
pairs(heart.attk_scale, gap=0,  pch=group, cex.main= 0.7, main="Original space\nH.C.-Complete linkage m
"blue")[group])
```




```
fviz_cluster(list(data = heart.attk_scale, cluster = group),  palette = c("red", "blue"), ellipse.type =
             show.clust.aver = FALSE, ggtheme = theme_minimal())+labs(
               subtitle = "H.C. - Complete linkage method and Manhattan distance, K=2", cex.sub= 0.5)
```

```
## Warning: ggrepel: 254 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



To examine the goodness of clustering algorithm results, we have to analyst internal and external validation measures.

Internal validation measures::

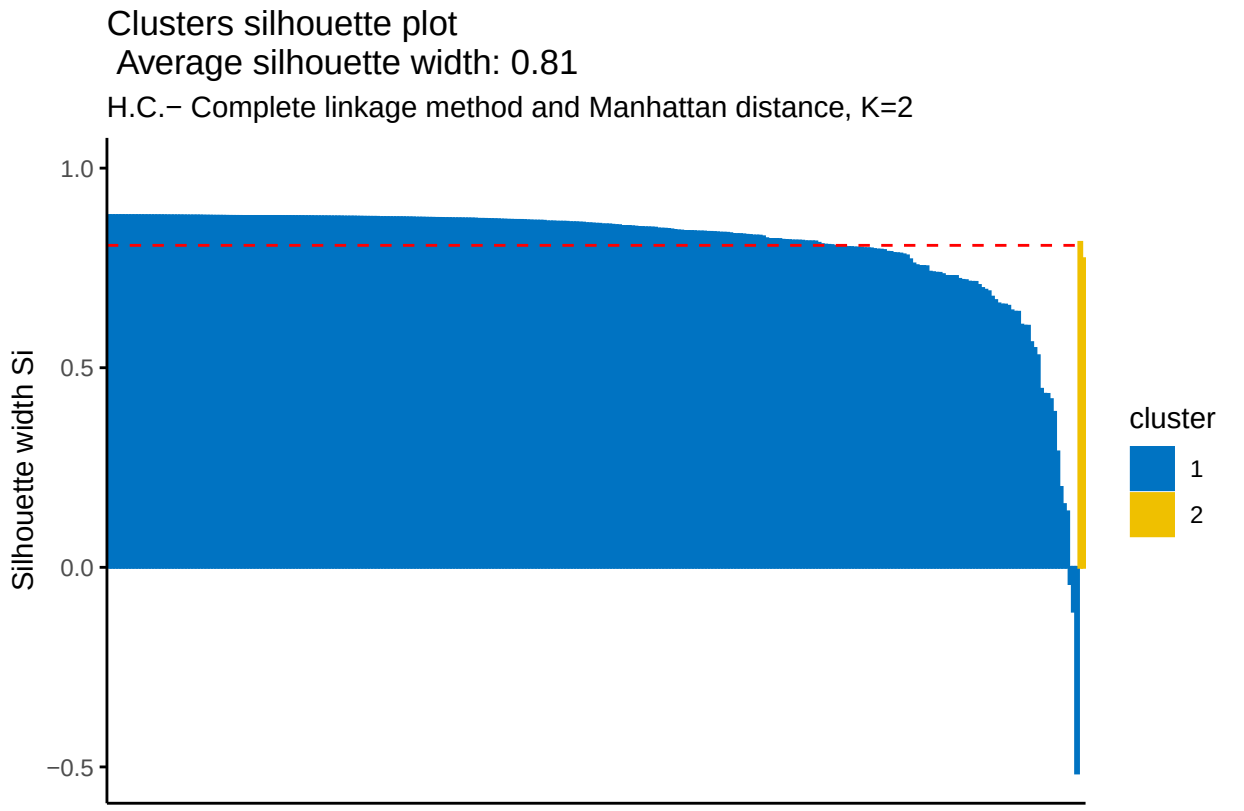
1.Silhouette width:

```
hclust<- eclust(heart.attk.sub,k=2 ,"hclust",hc_method = "complete",nboot = 50, hc_metric = "manhattan")
silinfo <- hclust$silinfo
silinfo$avg.width
```

```
## [1] 0.8065395
```

```
fviz_silhouette(hclust, palette = "jco", ggtheme = theme_classic())+
labs(subtitle = "H.C.- Complete linkage method and Manhattan distance, K=2", cex.sub= 0.5)
```

```
##   cluster size ave.sil.width
## 1       1  297          0.81
## 2       2    2          0.79
```




```
silinfo$clus.avg.widths
```

```
## [1] 0.8066265 0.7936243
```

```
sil <- hclust$silinfo$widths[, 1:3]
neg_sil_index_aver.eu<- which(sil[, "sil_width"] < 0)
sil[neg_sil_index_aver.eu, , drop = FALSE]
```

```
##     cluster neighbor   sil_width
## 241       1        2 -0.04071011
## 288       1        2 -0.10993660
## 106       1        2 -0.51525928
```

The value of complete silhouette width indicates that on average the units are well enough clustered. As in particular, in each cluster the units are on average the same silhouette value with respect to the silhouette width. According to this index, 3 units (241,288,106)that belong to cluster 1 are not well clustered: they should belong to cluster 2.

Dunn index:

```
stats <- cluster.stats(dist(heart.attk_scale), hclust$cluster)
stats$dunn
```

```
## [1] 0.2483991
```

According to the Dunn index, the units are not clustered well enough.

External validation measures:

1.Confusion matrix:

According to the Confusion matrix, the number of clusters is equal to nominal values. The clusters found are 2 and the nominal variable can take 2 possible values.

```
table(heart.attk$smoking, hclust$cluster)
```

```
##
##       1   2
##   0 202   1
##   1  95   1
```

Most of the smokers (n = 95) has been classified in cluster 1 while cluster 2 have 1 number of value. The same happened for non-smokers (n=202) classified in cluster 1 while cluster 2 has 1 value.

2.Correct Rand Index:

```
smoking <- as.numeric(heart.attk$smoking)
stats<- cluster.stats(d = dist(heart.attk_scale), smoking, hclust$cluster)
stats$corrected.rand
```

```
## [1] 0.003799734
```

According to the Correct Rand Index, there is no agreement between the numerical value and the cluster solution. From -1 to +1, the agreement is very close to 0.

3.Meila's VI Index:

```
stats$vi
```

```
## [1] 0.6669104
```

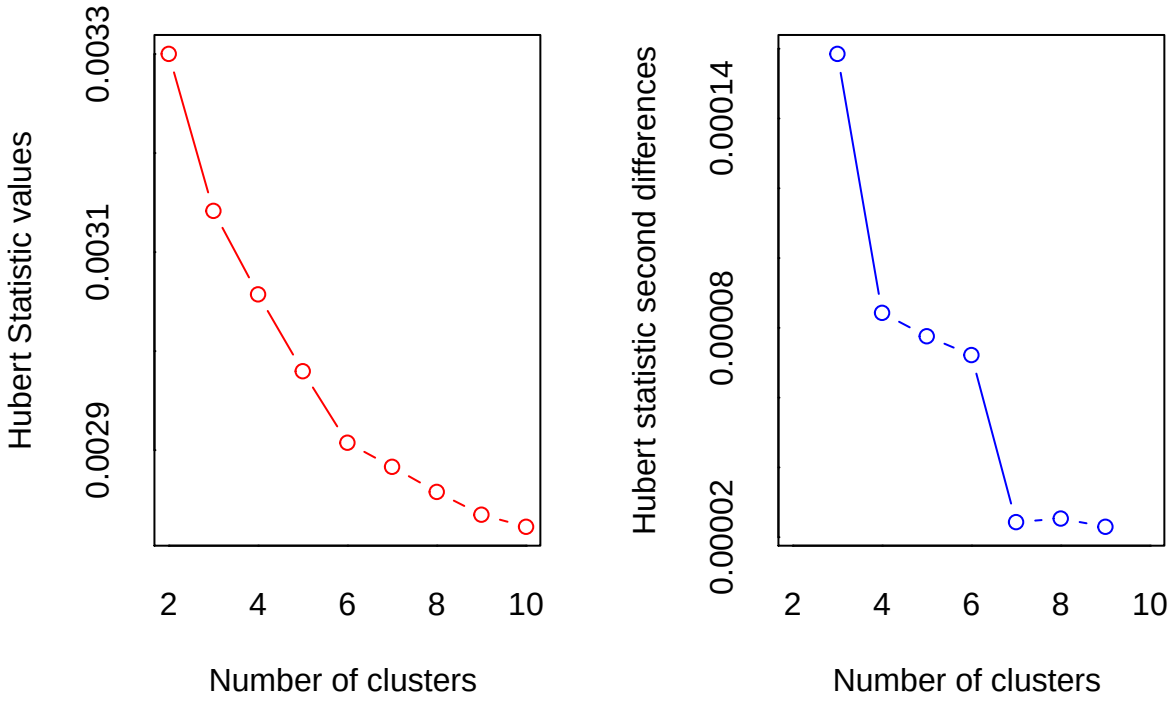
Centroid linkage method and Euclidean distance:

```
library(NbClust)
nb <- NbClust(heart.attk_scale, distance = "euclidean", min.nc = 2, max.nc = 10,
method = "centroid")
```

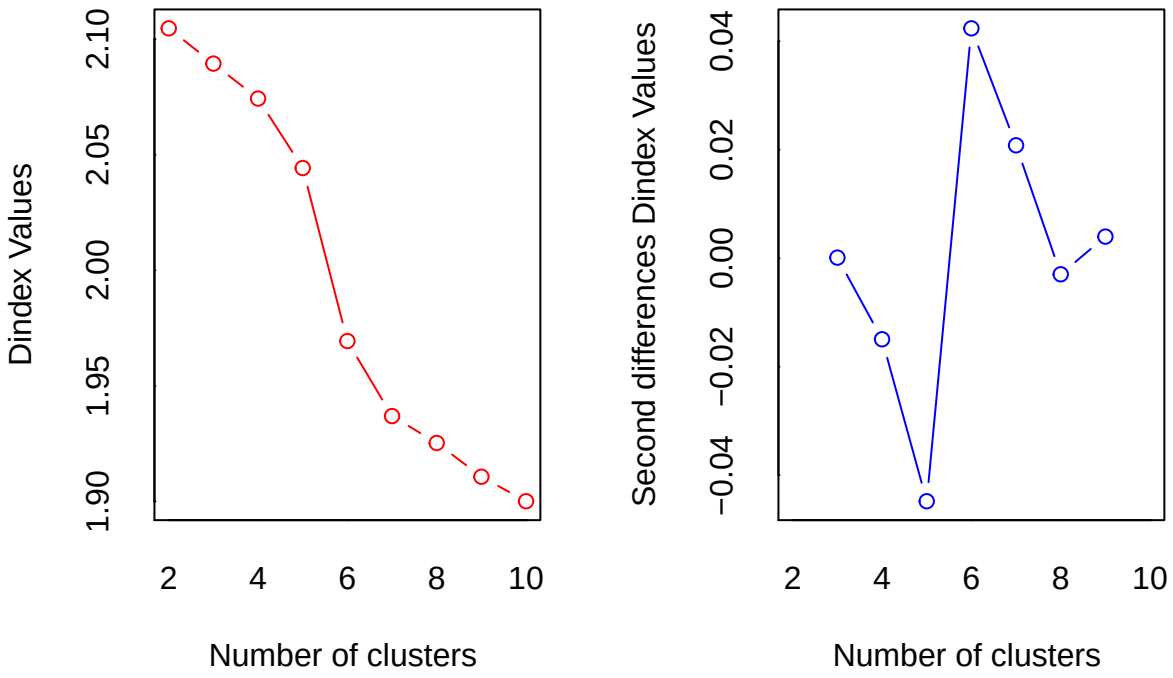
```
## Warning in pf(beale, pp, df2): NaNs produced
```

```
## Warning in pf(beale, pp, df2): NaNs produced
```

```
## [1] "Frey index : No clustering structure in this data set"
```



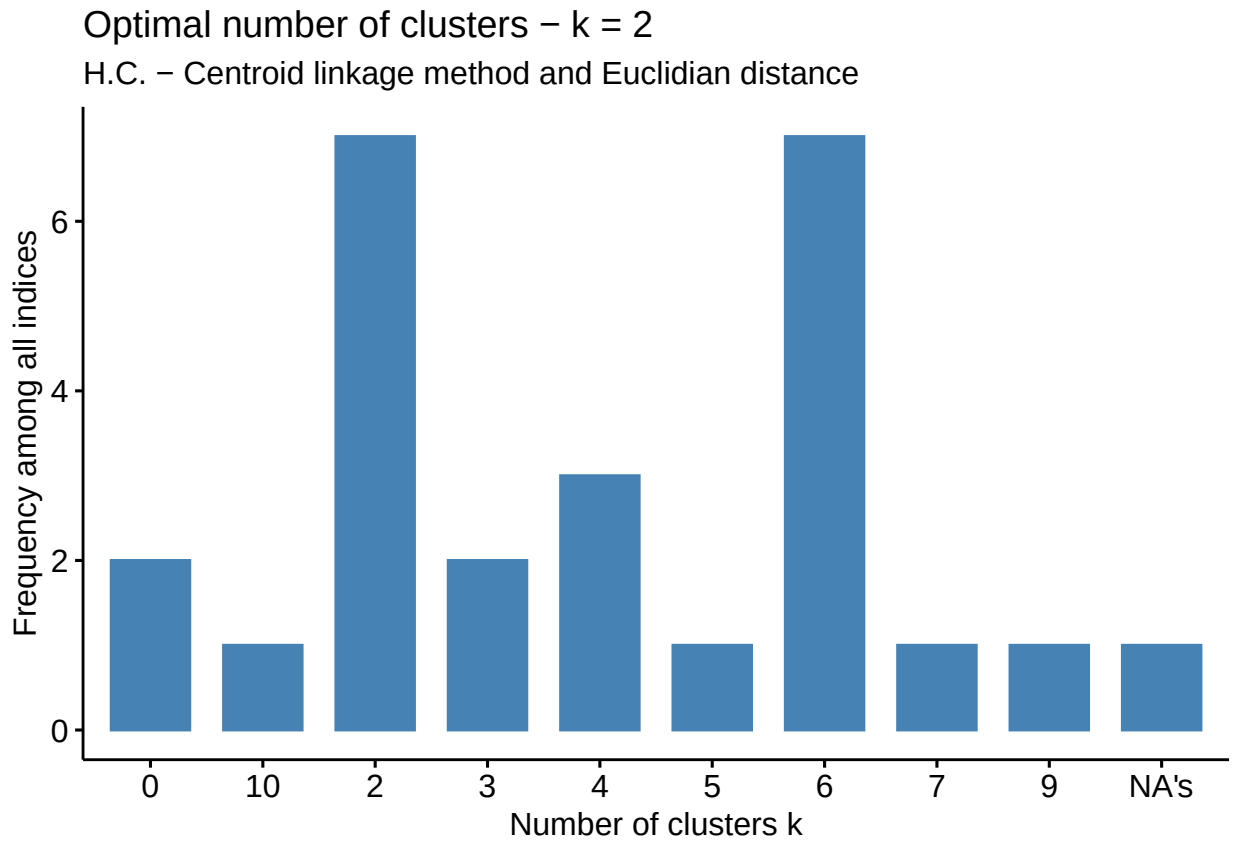

```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```




```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 7 proposed 2 as the best number of clusters
## * 2 proposed 3 as the best number of clusters
## * 3 proposed 4 as the best number of clusters
## * 1 proposed 5 as the best number of clusters
## * 7 proposed 6 as the best number of clusters
## * 1 proposed 7 as the best number of clusters
## * 1 proposed 9 as the best number of clusters
## * 1 proposed 10 as the best number of clusters
##
##                     ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  2
##
##
## *******************************************************************
```

```
#library(factoextra)
fviz_nbclust(nb)+labs(subtitle = "H.C. - Centroid linkage method and Euclidian distance", cex.sub= 0.5)
```

```
## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") .viz_NbClust(x, print.summary, : the
## condition has length > 1 and only the first element will be used

## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") {: the condition has length > 1 and
## only the first element will be used

## Among all indices:
## ===================
## * 2 proposed  0 as the best number of clusters
## * 7 proposed  2 as the best number of clusters
## * 2 proposed  3 as the best number of clusters
## * 3 proposed  4 as the best number of clusters
## * 1 proposed  5 as the best number of clusters
## * 7 proposed  6 as the best number of clusters
## * 1 proposed  7 as the best number of clusters
## * 1 proposed  9 as the best number of clusters
## * 1 proposed  10 as the best number of clusters
## * 1 proposed  NA's as the best number of clusters
##
## Conclusion
## =========================
## * According to the majority rule, the best number of clusters is  2 .
```



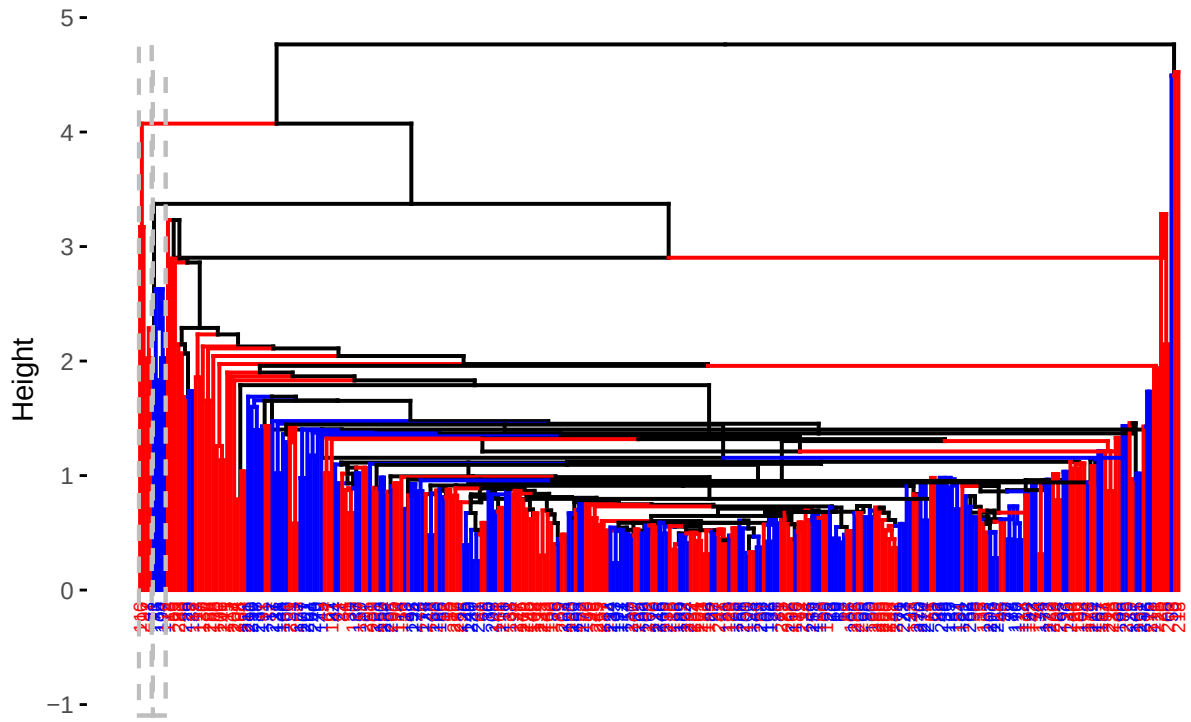

```
dist.eucl <- dist(heart.attk_scale, method = "euclidian")
hc <- hclust(dist.eucl, method = "centroid")
group <- cutree(hc, k=2)
table(group)
```

```
## group
##   1   2
## 296   3
```

```
fviz_dend(hc, k = 2, cex = 0.5, k_colors = c("red", "blue"), color_labels_by_k = TRUE, rect = TRUE)+lab
```

```
## Warning in get_col(col, k): Length of color vector was shorter than the number
## of clusters - color vector was recycled
```

```
## Warning in data.frame(xmin = unlist(xleft), ymin = unlist(ybottom), xmax =
## unlist(xright), : row names were found from a short variable and have been
## discarded
```

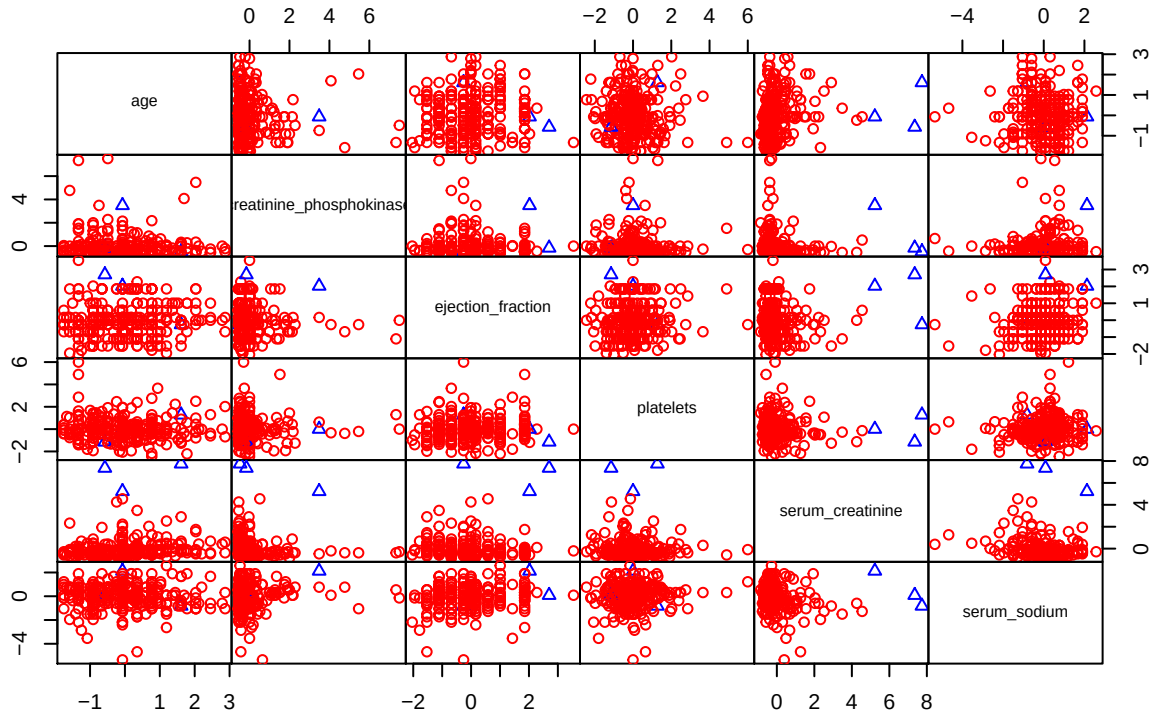



```
cor(dist.eucl, cophenetic(hc))
```

```
## [1] 0.8338457
```

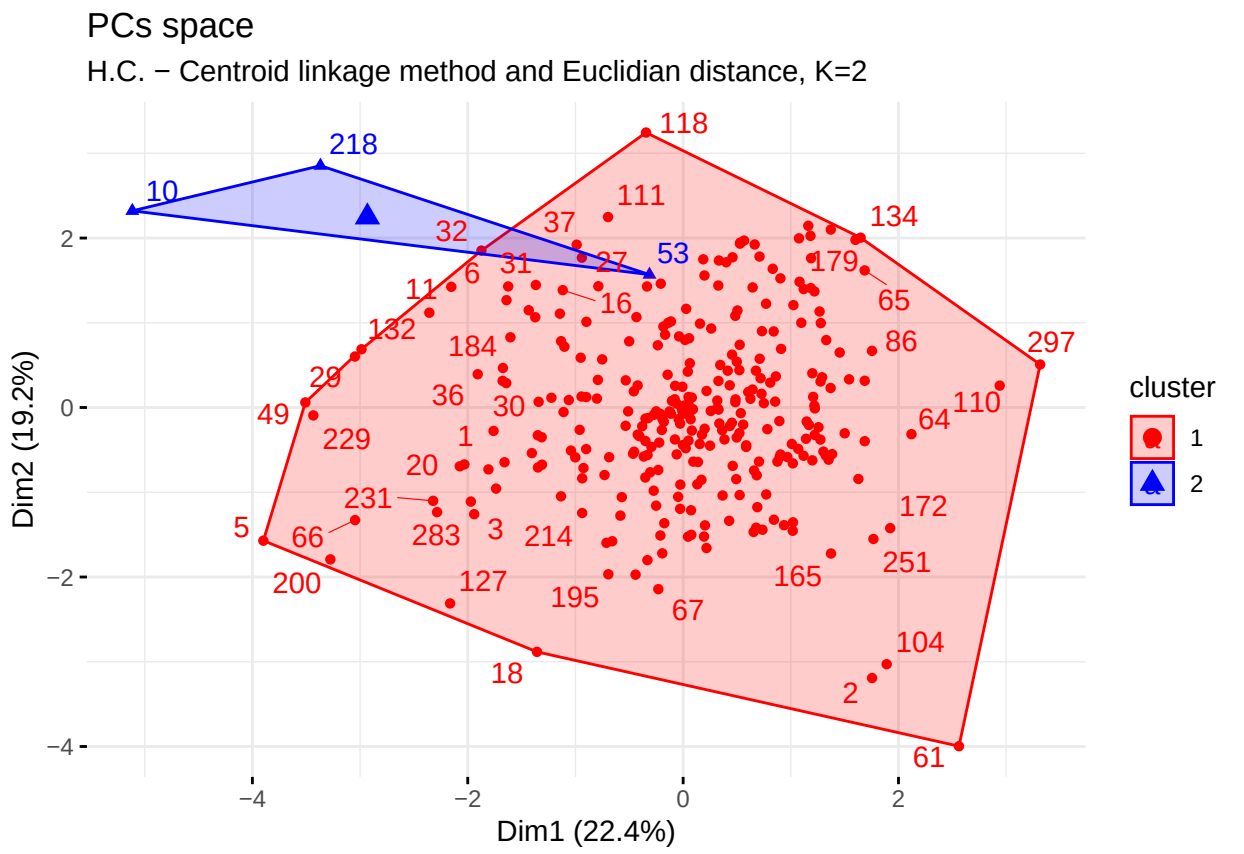
According to the result of "NbClust", the best number of clusters, applying hierarchical clustering method and using the Centroid linkage method and Euclidean distance is 2.

```
pairs(heart.attk_scale, gap=0, pch=group, cex.main= 0.7, main="Original space\nH.C.-Centroid linkage me
```




```
fviz_cluster(list(data = heart.attk_scale, cluster = group), palette = c("red", "blue"), ellipse.type =
```

```
## Warning: ggrepel: 254 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



To examine the goodness of clustering algorithm results, we have to analyst internal and external validation measures.

Internal validation measures:

1.Silhouette width:

```
hclust<- eclust(heart.attk.sub,k=2 ,"hclust",hc_method = "centroid", nboot = 50, hc_metric = "euclidean
```

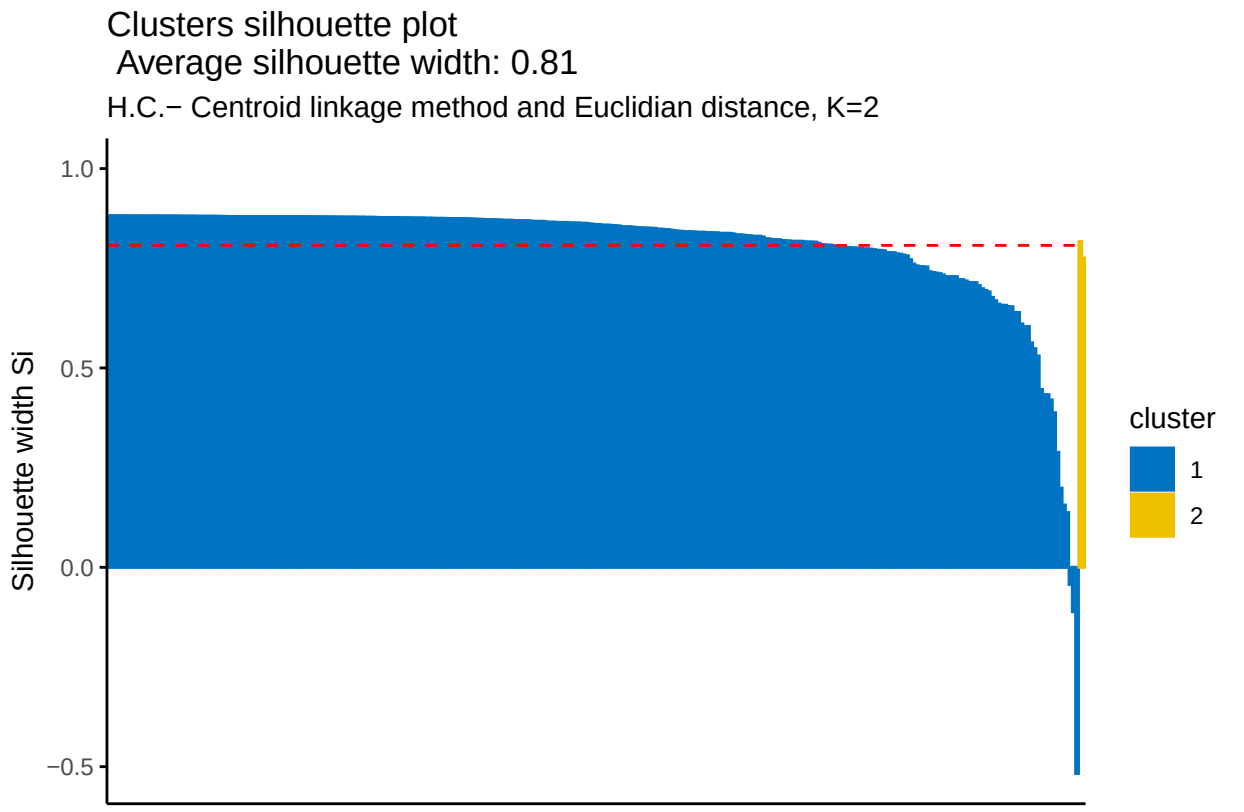
```
## Warning in get_col(col, k): Length of color vector was shorter than the number
## of clusters - color vector was recycled
```

```
silinfo <- hclust$silinfo
silinfo$avg.width
```

```
## [1] 0.8074437
```

```
fviz_silhouette(hclust, palette = "jco", ggtheme = theme_classic())+labs(
subtitle = "H.C.- Centroid linkage method and Euclidian distance, K=2",
cex.sub= 0.5)
```

```
##   cluster size ave.sil.width
## 1       1  297          0.81
## 2       2    2          0.80
```




```
silinfo$clus.avg.widths
```

```
## [1] 0.8075174 0.7965037
```

```
sil <- hclust$silinfo$widths[, 1:3]
neg_sil_index_aver.eu<- which(sil[, "sil_width"] < 0)
sil[neg_sil_index_aver.eu, , drop = FALSE]
```

```
##     cluster neighbor   sil_width
## 241       1        2 -0.04298099
## 288       1        2 -0.11138118
## 106       1        2 -0.51715859
```

The value of average silhouette width indicates that in average the units are well enough clustered. In particular, in cluster 1 (blue cluster) the units are on average the same silhouette value with respect to the silhouette width; in cluster 2 (the yellow one) also the units are on average the same silhouette value with respect to silhouette width. According to this index, 3 units (241,288,106)that belong to cluster 1 are not well clustered: they should belong to cluster 2.

2.Dunn index:

```
library(fpc)
stats <- cluster.stats(dist(heart.attk_scale), hclust$cluster)
stats$dunn
```

```
## [1] 0.2483991
```

According to the Dunn index, the units are not clustered well enough.

External validation measures:

1.Confusion matrix:

According to the Confusion matrix, the number of clusters is equal to the nominal values.

```
table(heart.attk$smoking, hclust$cluster)
```

```
##
##       1   2
##   0 202   1
##   1  95   1
```

Most of the smokers (n = 95) has been classified in cluster 1 while cluster 2 have 1 number of value. The same happened for non-smokers (n=202) classified in cluster 1 while cluster 2 has 1 value.

2.Correct Rand Index:

```
smoking <- as.numeric(heart.attk$smoking)
stats<- cluster.stats(d = dist(heart.attk_scale), smoking, hclust$cluster)
stats$corrected.rand
```

```
## [1] 0.003799734
```

According to the Correct Rand Index, there is no agreement between the numerical values and the cluster solution. From -1 to +1, the agreement is very close to 0.

3.Meila's VI Index:

```
stats$vi
```

```
## [1] 0.6669104
```

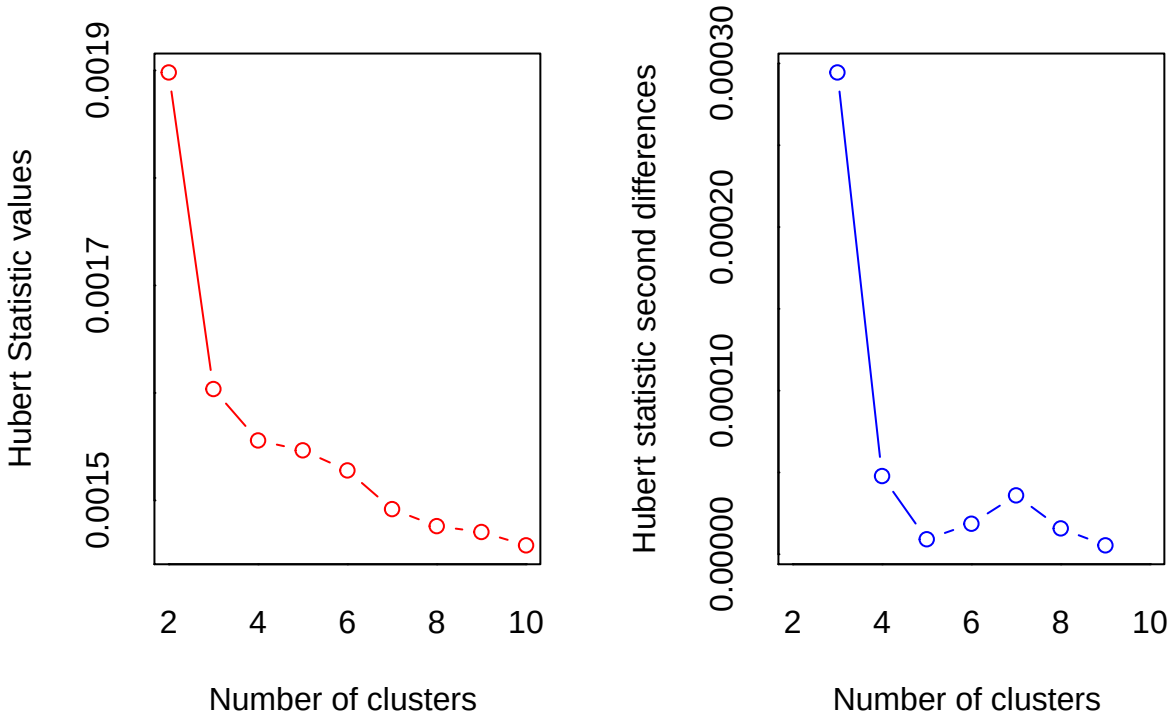
Centroid linkage method and Manhattan distance:

```
nb <- NbClust(heart.attk_scale, distance = "manhattan", min.nc = 2, max.nc = 10,
method = "centroid")
```

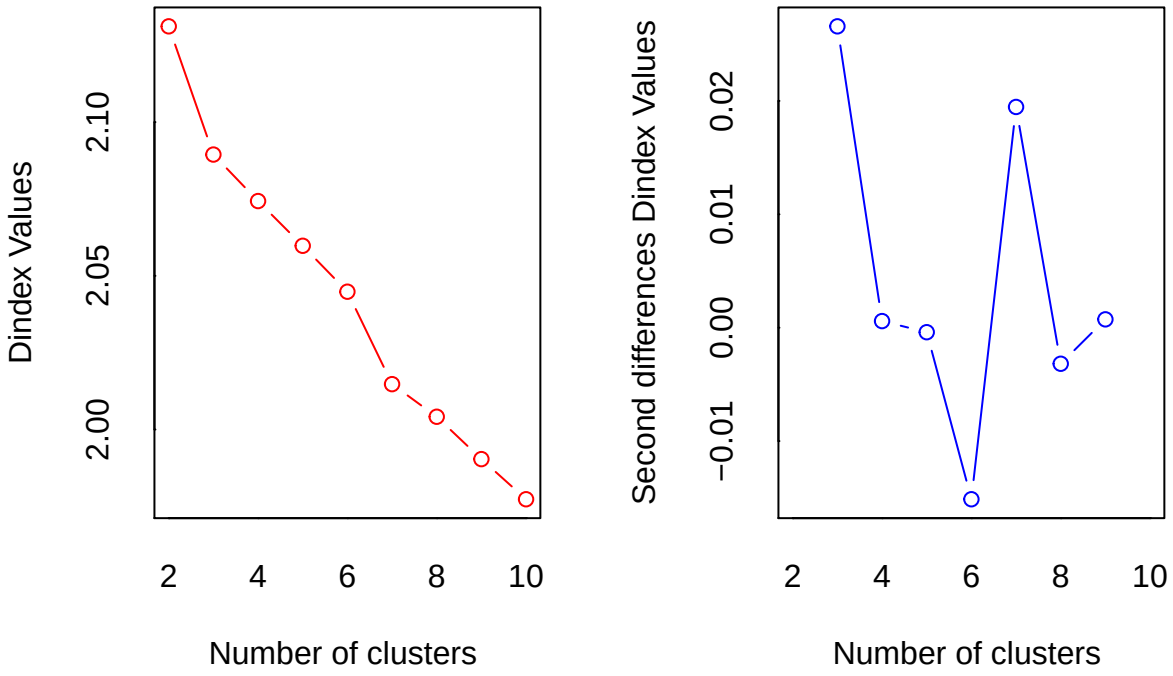
```
## Warning in pf(beale, pp, df2): NaNs produced
```

```
## Warning in pf(beale, pp, df2): NaNs produced
```

```
## [1] "Frey index : No clustering structure in this data set"
```




```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```

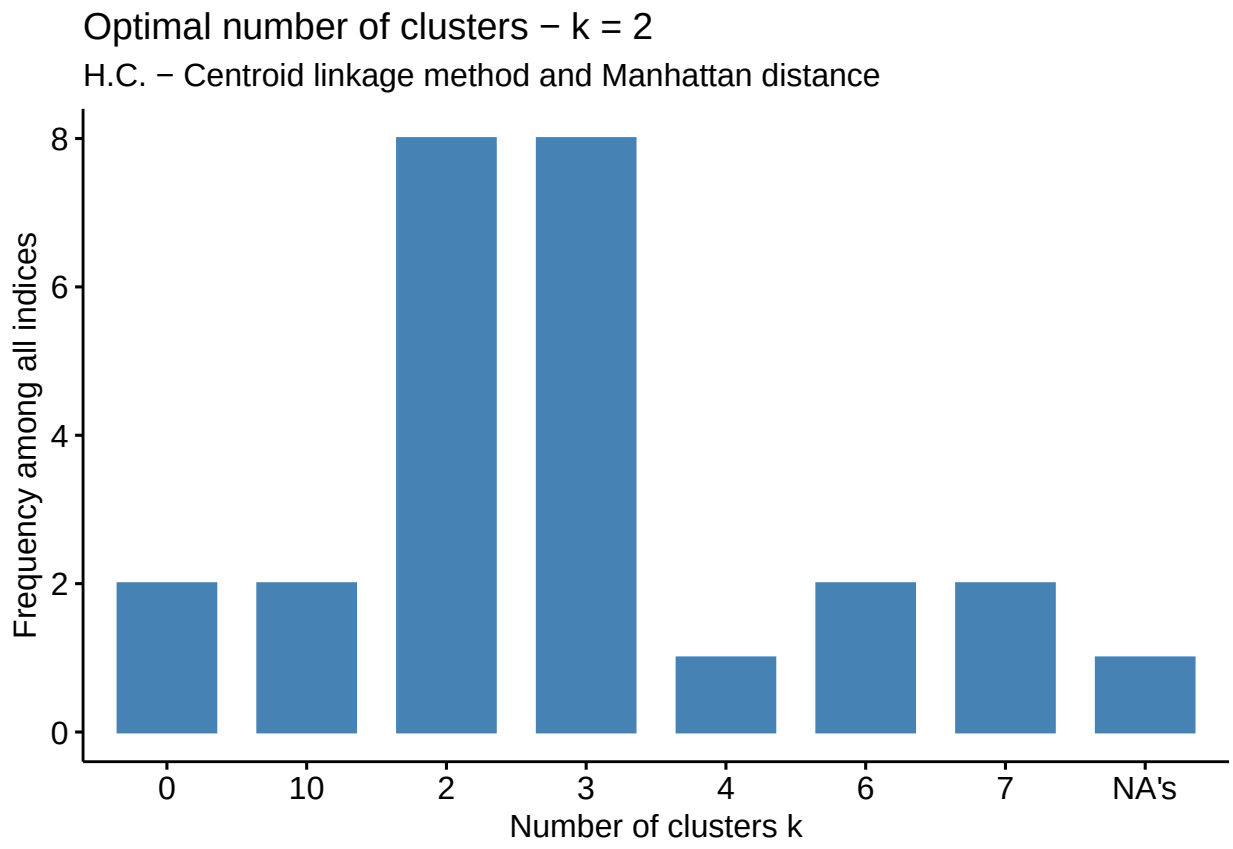



```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 8 proposed 2 as the best number of clusters
## * 8 proposed 3 as the best number of clusters
## * 1 proposed 4 as the best number of clusters
## * 2 proposed 6 as the best number of clusters
## * 2 proposed 7 as the best number of clusters
## * 2 proposed 10 as the best number of clusters
##
##                     ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  2
##
##
## *******************************************************************
```

```
fviz_nbclust(nb)+labs(subtitle = "H.C. - Centroid linkage method and Manhattan distance", cex.sub= 0.5)
```

```
## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
## (class(best_nc) == : the condition has length > 1 and only the first element
```

```
## will be used

## Warning in if (class(best_nc) == "matrix") .viz_NbClust(x, print.summary, : the
## condition has length > 1 and only the first element will be used

## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") {: the condition has length > 1 and
## only the first element will be used

## Among all indices:
## ===================
## * 2 proposed  0 as the best number of clusters
## * 8 proposed  2 as the best number of clusters
## * 8 proposed  3 as the best number of clusters
## * 1 proposed  4 as the best number of clusters
## * 2 proposed  6 as the best number of clusters
## * 2 proposed  7 as the best number of clusters
## * 2 proposed  10 as the best number of clusters
## * 1 proposed  NA's as the best number of clusters
##
## Conclusion
## =========================
## * According to the majority rule, the best number of clusters is  2 .
```



```
dist.man <- dist(heart.attk_scale, method = "manhattan")
hc <- hclust(dist.man, method = "centroid")
group <- cutree(hc, k=2)
table(group)
```

```
## group
##   1   2
## 298   1
```

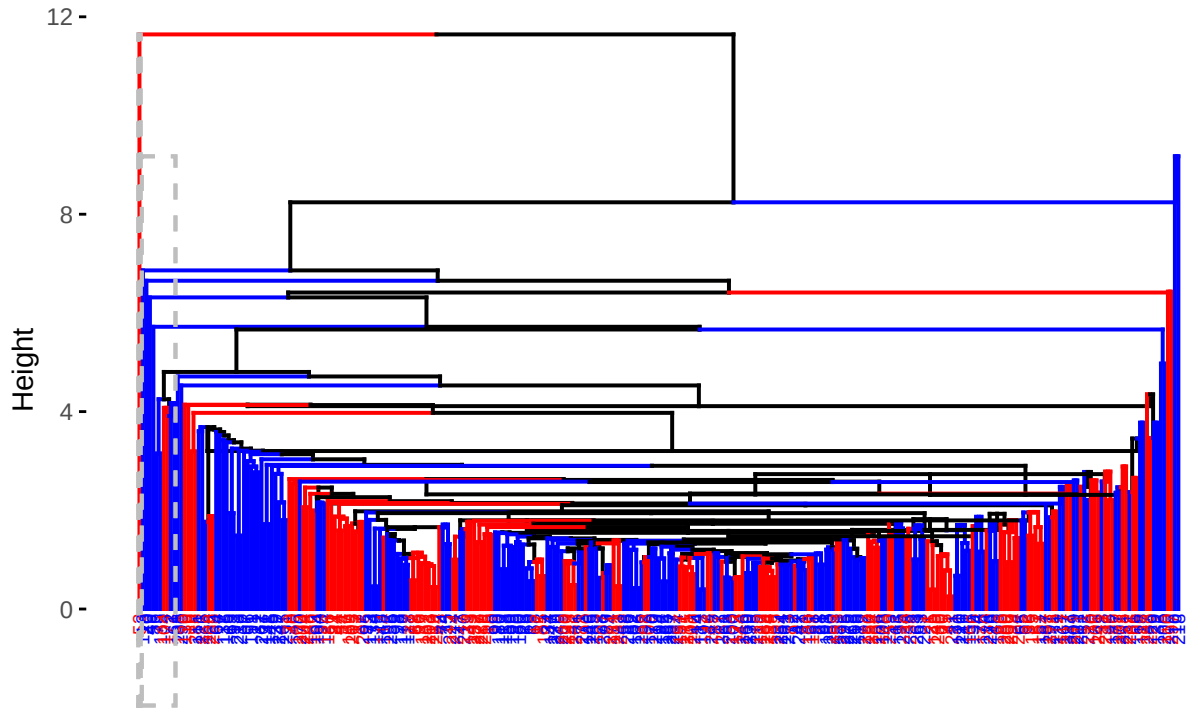
```
group
```

```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [112] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [149] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [186] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [223] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [260] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [297] 1 1 1
```

```
fviz_dend(hc, k = 2, cex = 0.5, k_colors = c("red", "blue"),
color_labels_by_k = TRUE, rect = TRUE)+labs(title = "Dendrogram",
subtitle = "H.C. - Centroid linkage method and Manhattan distance, K=2", cex.subtitle= 0.5)
```

```
## Warning in get_col(col, k): Length of color vector was shorter than the number
## of clusters - color vector was recycled
```

```
## Warning in data.frame(xmin = unlist(xleft), ymin = unlist(ybottom), xmax =
## unlist(xright), : row names were found from a short variable and have been
## discarded
```

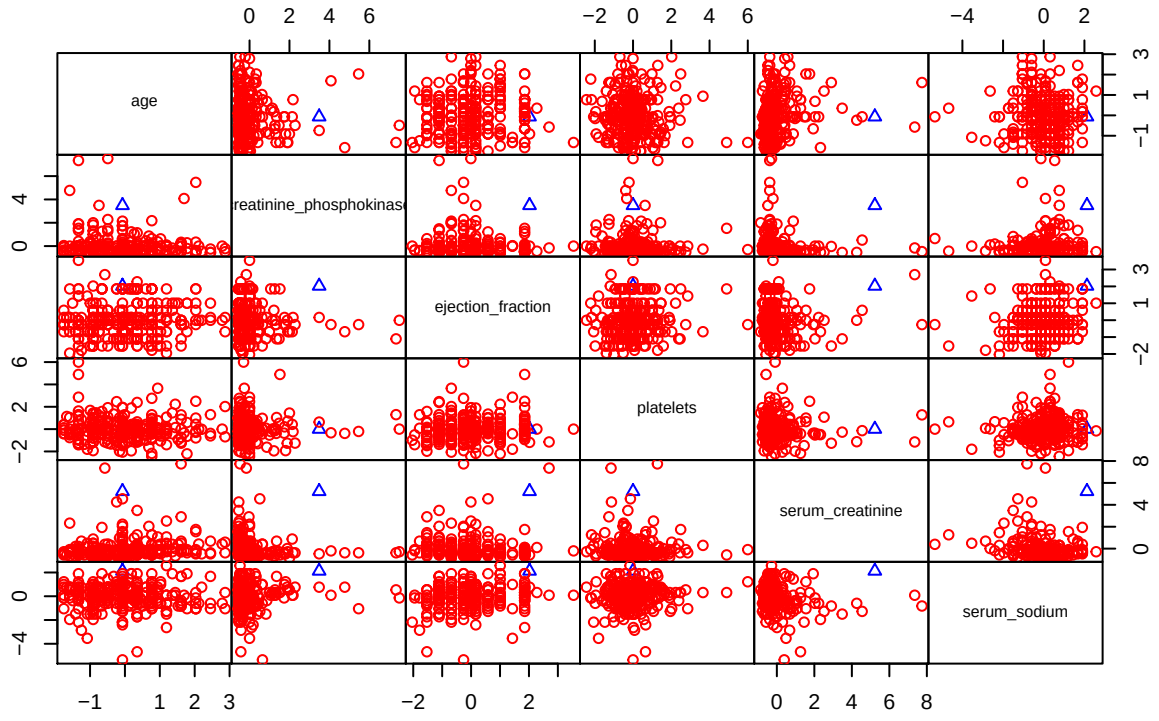



```
cor(dist.eucl, cophenetic(hc))
```

```
## [1] 0.7794849
```

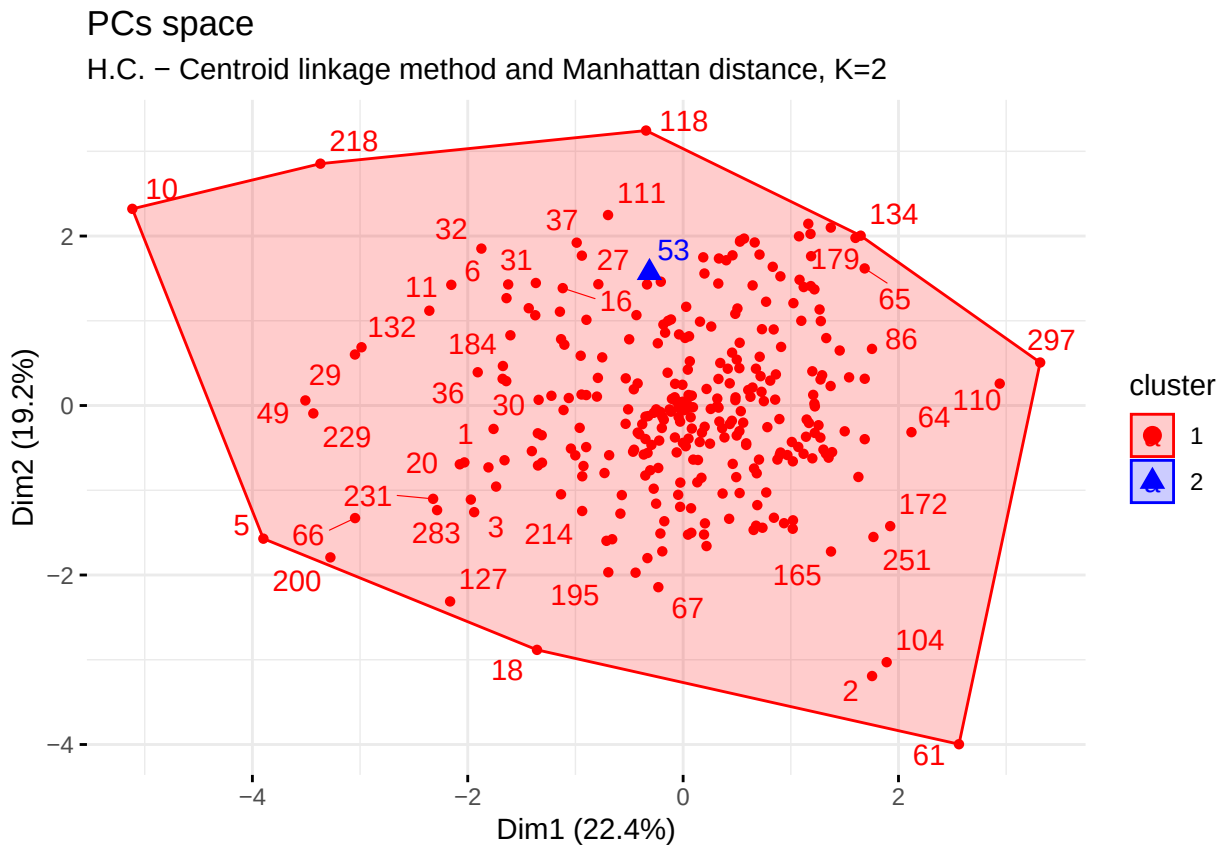
According to the result of "NbClust", the best number of clusters, applying hierarchical clustering method and using complete linkage method and Manhattan distance is 2.

```
pairs(heart.attk_scale, gap=0, pch=group, cex.main= 0.7, main="Original space\nH.C.-Centroid linkage me
```




```
fviz_cluster(list(data = heart.attk_scale, cluster = group), palette = c("red", "blue"), ellipse.type =
labs(subtitle = "H.C. - Centroid linkage method and Manhattan distance, K=2", cex.sub= 0.5)
```

```
## Warning: ggrepel: 254 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



To examine the goodness of clustering algorithm results, we have tp analyst internal and external validation measures.

Internal validation measures:

1.Silhouette width:

```
hclust<- eclust(heart.attk.sub,k=2 ,"hclust",hc_method = "centroid",nboot = 50, hc_metric = "manhattan")
```

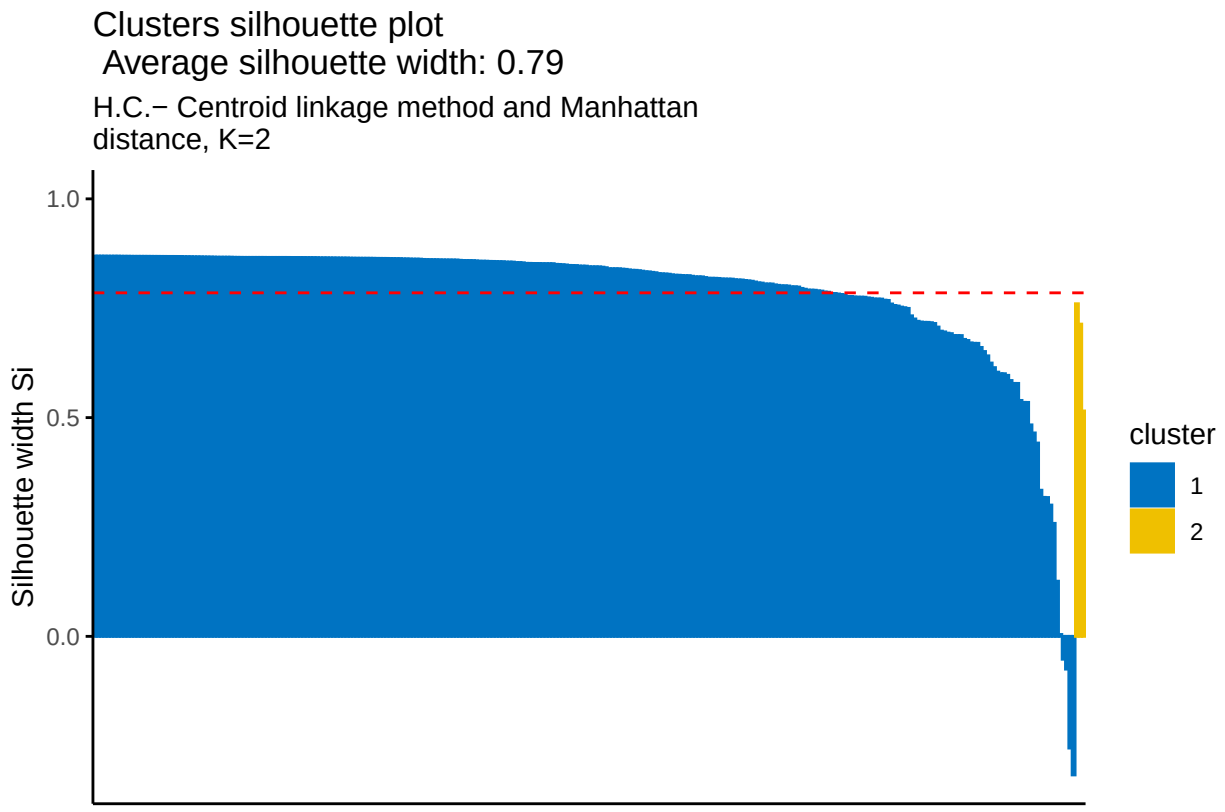
```
## Warning in get_col(col, k): Length of color vector was shorter than the number
## of clusters - color vector was recycled
```

```
silinfo <- hclust$silinfo
silinfo$avg.width
```

```
## [1] 0.7851753
```

```
fviz_silhouette(hclust, palette = "jco", ggtheme = theme_classic())+labs(
subtitle = "H.C.- Centroid linkage method and Manhattan
distance, K=2", cex.sub= 0.5)
```

```
##   cluster size ave.sil.width
## 1       1  296          0.79
## 2       2    3          0.66
```




```
silinfo$clus.avg.widths
```

```
## [1] 0.7864125 0.6631076
```

```
sil <- hclust$silinfo$widths[, 1:3]
neg_sil_index_aver.eu<- which(sil[, "sil_width"] < 0)
sil[neg_sil_index_aver.eu, , drop = FALSE]
```

```
##     cluster neighbor   sil_width
## 225       1        2 -0.05228841
## 118       1        2 -0.07490273
## 241       1        2 -0.25534598
## 288       1        2 -0.31679614
```

The value of complete silhouette width indicates that on average the units are well enough clustered. As in particular, in each cluster the units are on average the same silhouette value with respect to the silhouette width. According to this index, 4 units (225,118,241,288)that belong to cluster 1 are not well clustered: they should belong to cluster 2.

2.Dunn index:

```
stats <- cluster.stats(dist(heart.attk_scale), hclust$cluster)
stats$dunn
```

```
## [1] 0.09632583
```

External validation measures:

Confusion matrix:

According to the Confusion matrix, the number of clusters is equal to nominal values. The clusters found are 2 and the nominal variable can take 2 possible values.

```
table(heart.attk$smoking, hclust$cluster)
```

```
##
##       1   2
##   0 202   1
##   1  94   2
```

Most of the smokers (n = 94) has been classified in cluster 1 while cluster 2 have 2 number of values. The same happened for non-smokers (n=202) classified in cluster 1 while cluster 2 has 1 value.

2.Correct Rand Index:

```
smoking <- as.numeric(heart.attk$smoking)
stats<- cluster.stats(d = dist(heart.attk_scale), smoking, hclust$cluster)
stats$corrected.rand
```

```
## [1] 0.0111802
```

According to the Correct Rand Index, there is no agreement between the numerical value and the cluster solution. From -1 to +1, the agreement is very close to 0.

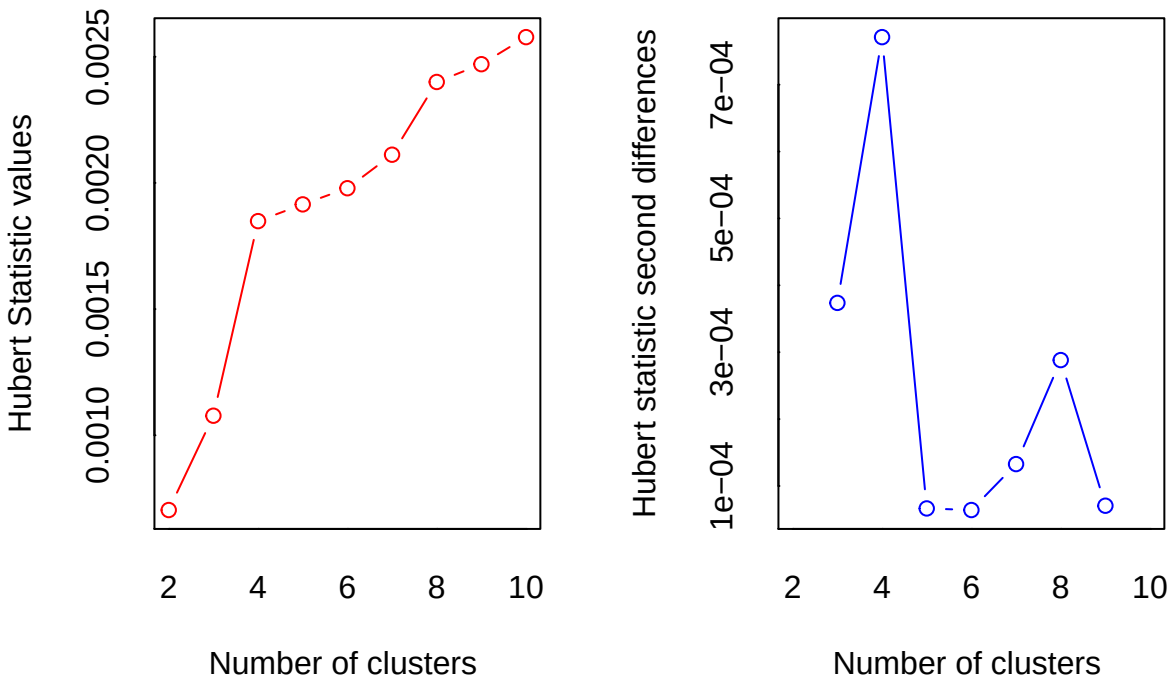
Meila's VI Index:

```
stats$vi
```

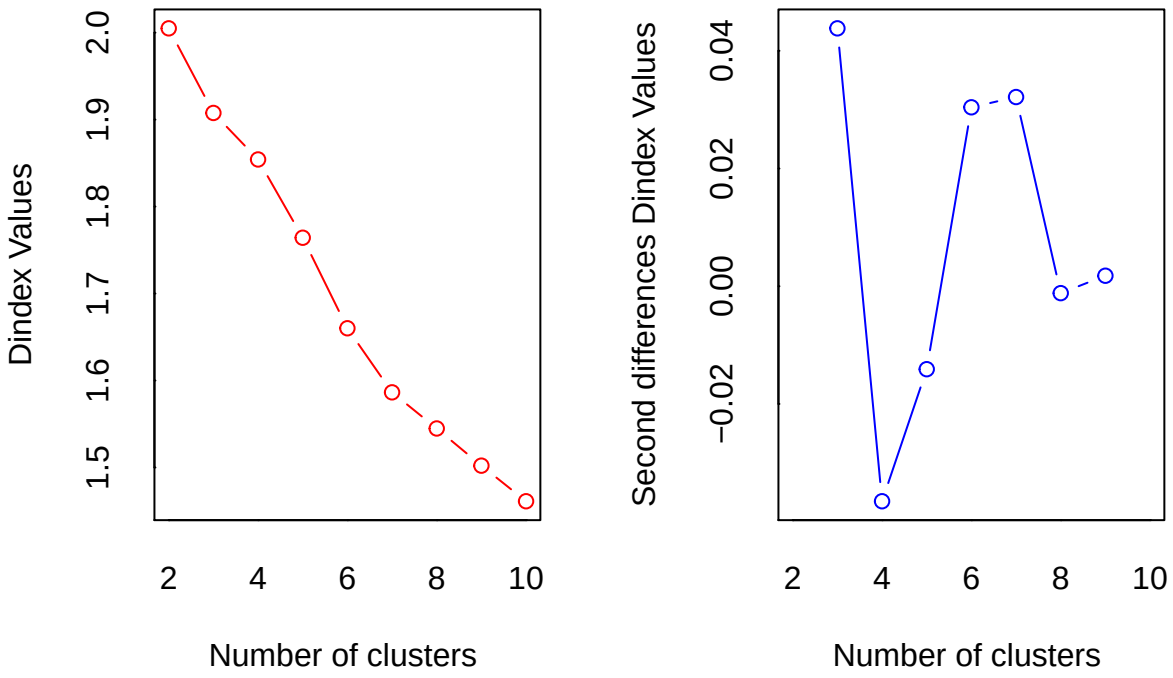
```
## [1] 0.6787569
```

Ward's (minimum deviance) method:

```
library(NbClust)
nb <- NbClust(heart.attk_scale, distance = "euclidean", min.nc = 2, max.nc = 10,
method = "ward.D2")
```




```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```

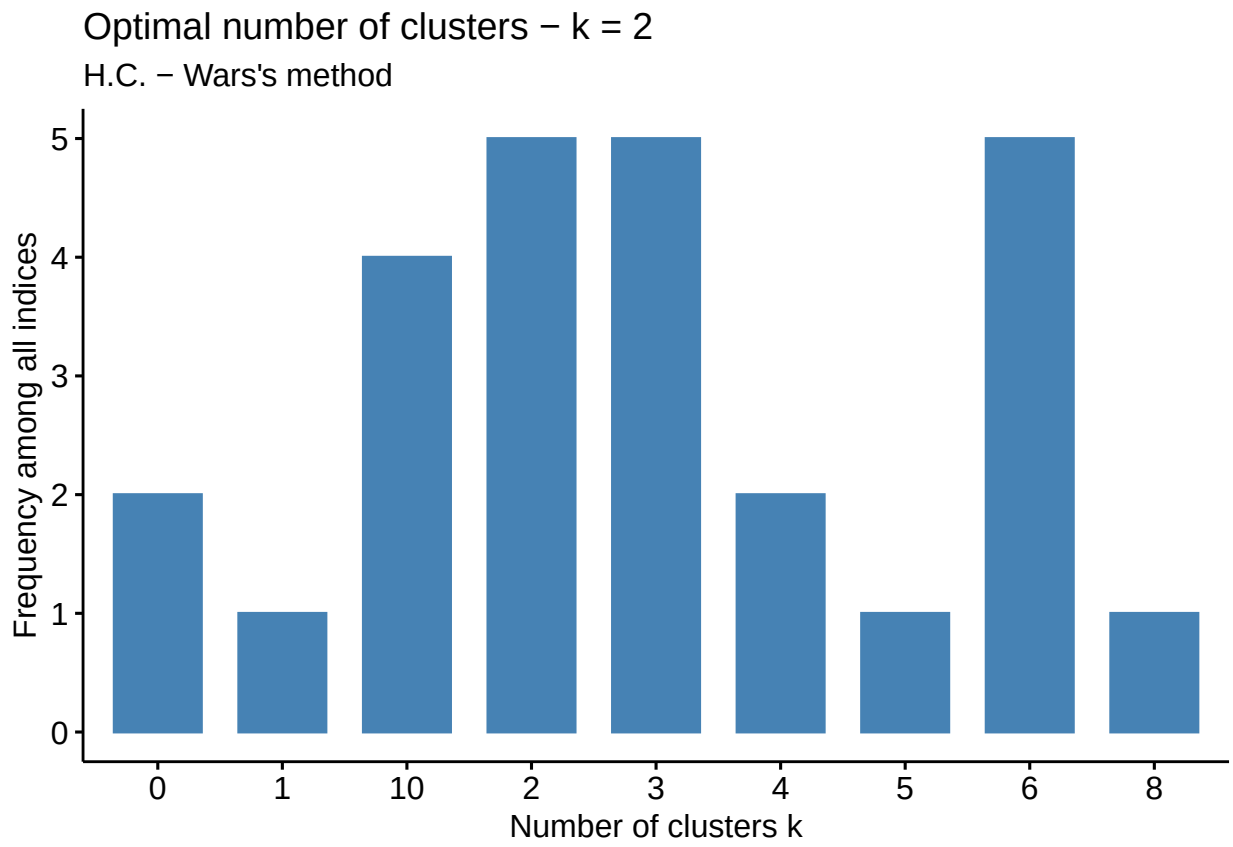



```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 5 proposed 2 as the best number of clusters
## * 5 proposed 3 as the best number of clusters
## * 2 proposed 4 as the best number of clusters
## * 1 proposed 5 as the best number of clusters
## * 5 proposed 6 as the best number of clusters
## * 1 proposed 8 as the best number of clusters
## * 4 proposed 10 as the best number of clusters
##
##                     ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  2
##
##
## *******************************************************************
```

```
fviz_nbclust(nb)+labs(subtitle = "H.C. - Wars's method", cex.sub= 0.5)
```

```
## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
```

```
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") .viz_NbClust(x, print.summary, : the
## condition has length > 1 and only the first element will be used

## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") {: the condition has length > 1 and
## only the first element will be used

## Among all indices:
## ===================
## * 2 proposed  0 as the best number of clusters
## * 1 proposed  1 as the best number of clusters
## * 5 proposed  2 as the best number of clusters
## * 5 proposed  3 as the best number of clusters
## * 2 proposed  4 as the best number of clusters
## * 1 proposed  5 as the best number of clusters
## * 5 proposed  6 as the best number of clusters
## * 1 proposed  8 as the best number of clusters
## * 4 proposed  10 as the best number of clusters
##
## Conclusion
## =========================
## * According to the majority rule, the best number of clusters is  2 .
```



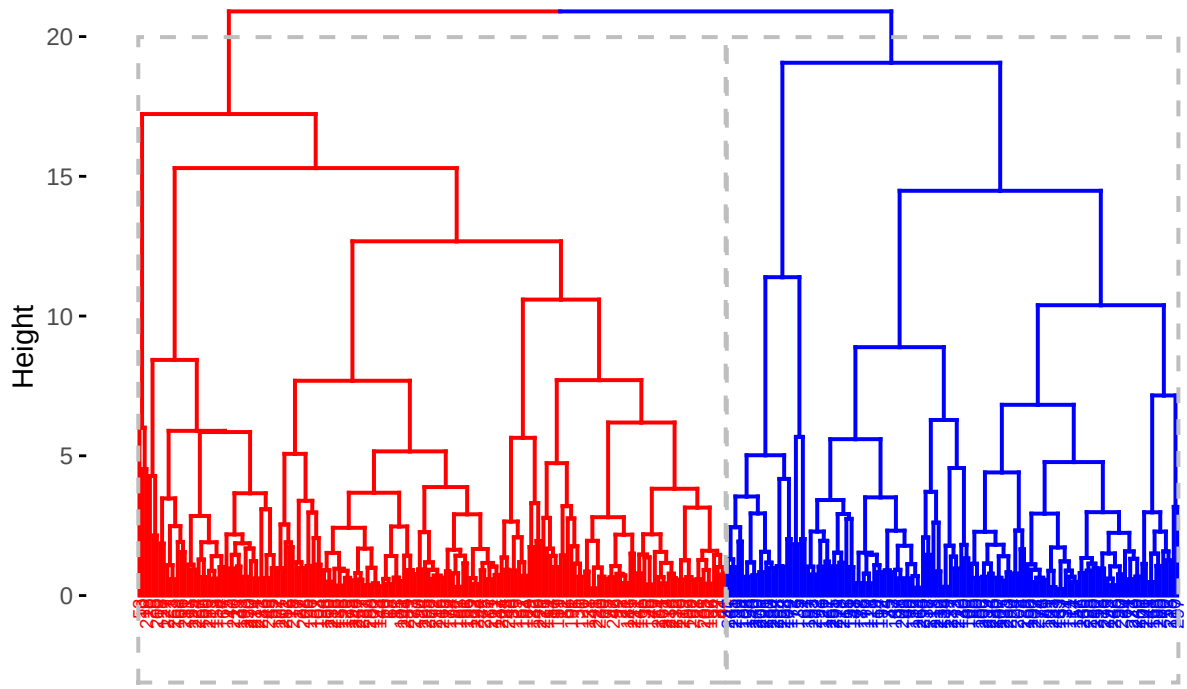

```
hc <- hclust(dist.eucl, method = "ward.D2")
grop <- cutree(hc, k=2)
table(grop)
```

```
## grop
##   1   2
## 169 130
```

```
group
```

```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [112] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [149] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [186] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [223] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [260] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [297] 1 1 1
```

```
fviz_dend(hc, k = 2, cex = 0.5, k_colors = c("red", "blue"),  color_labels_by_k = TRUE, rect = TRUE)+lab
subtitle = "H.C. - Ward's method, K=2", cex.subtitle= 0.5)
```

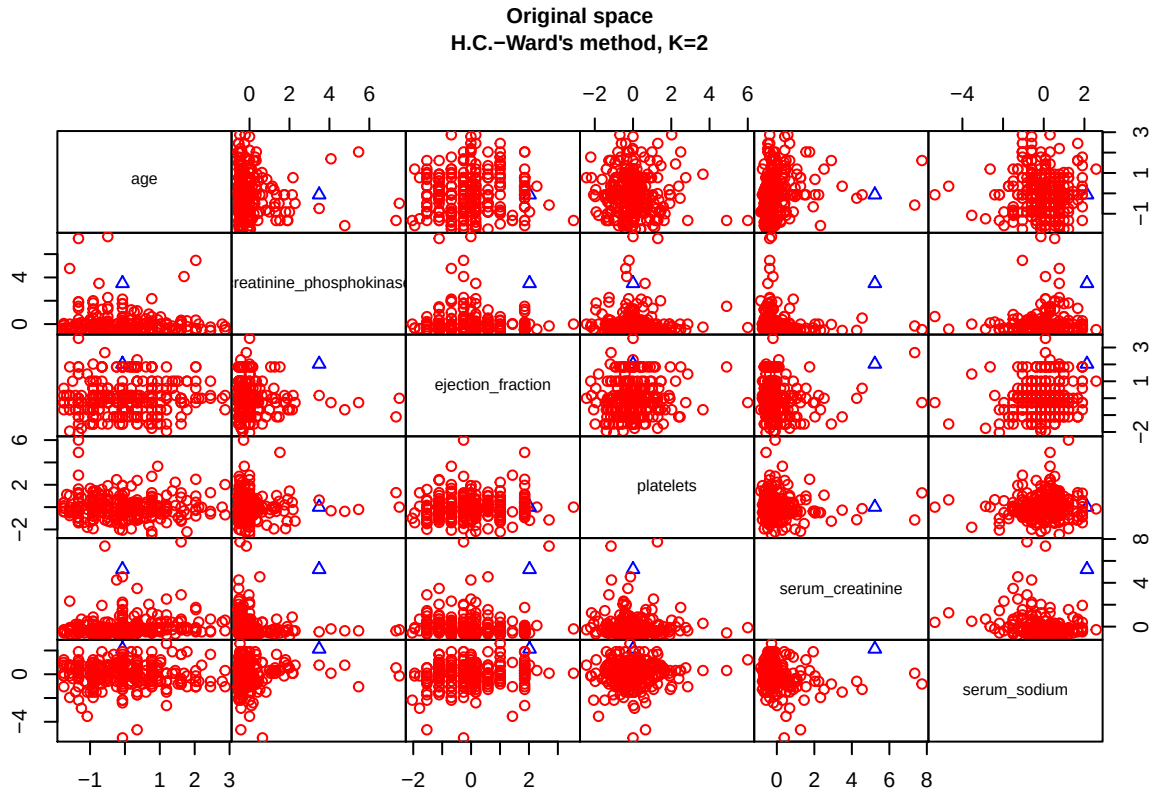



```
cor(dist.eucl, cophenetic(hc))
```

```
## [1] 0.3488687
```

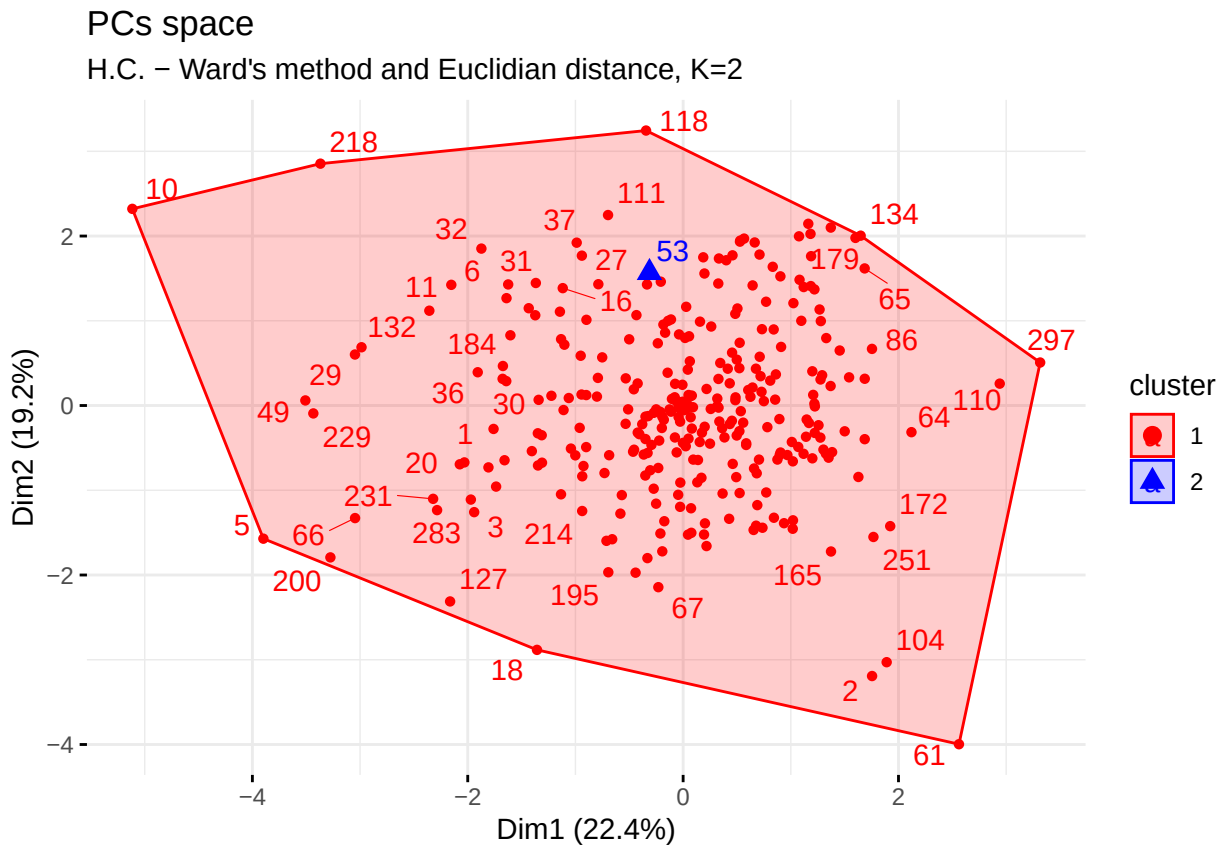
According to the function "NbClust", the best number of clusters, applying hierarchical clustering method and using Ward's method and Euclidean distance, is 2: cluster 1 with 169 units, cluster 2 with 130 units.

```
pairs(heart.attk_scale, gap=0, pch=group, cex.main= 0.7, main="Original space\nH.C.-Ward's method, K=2"
```




```
fviz_cluster(list(data = heart.attk_scale, cluster = group), palette = c("red", "blue"), ellipse.type =
  labs(subtitle = "H.C. - Ward's method and Euclidian distance, K=2", cex.sub= 0.5)
```

```
## Warning: ggrepel: 254 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



To examine the goodness of clustering algorithm results, we have to analyst internal and external validation measures.

Internal validation measures:

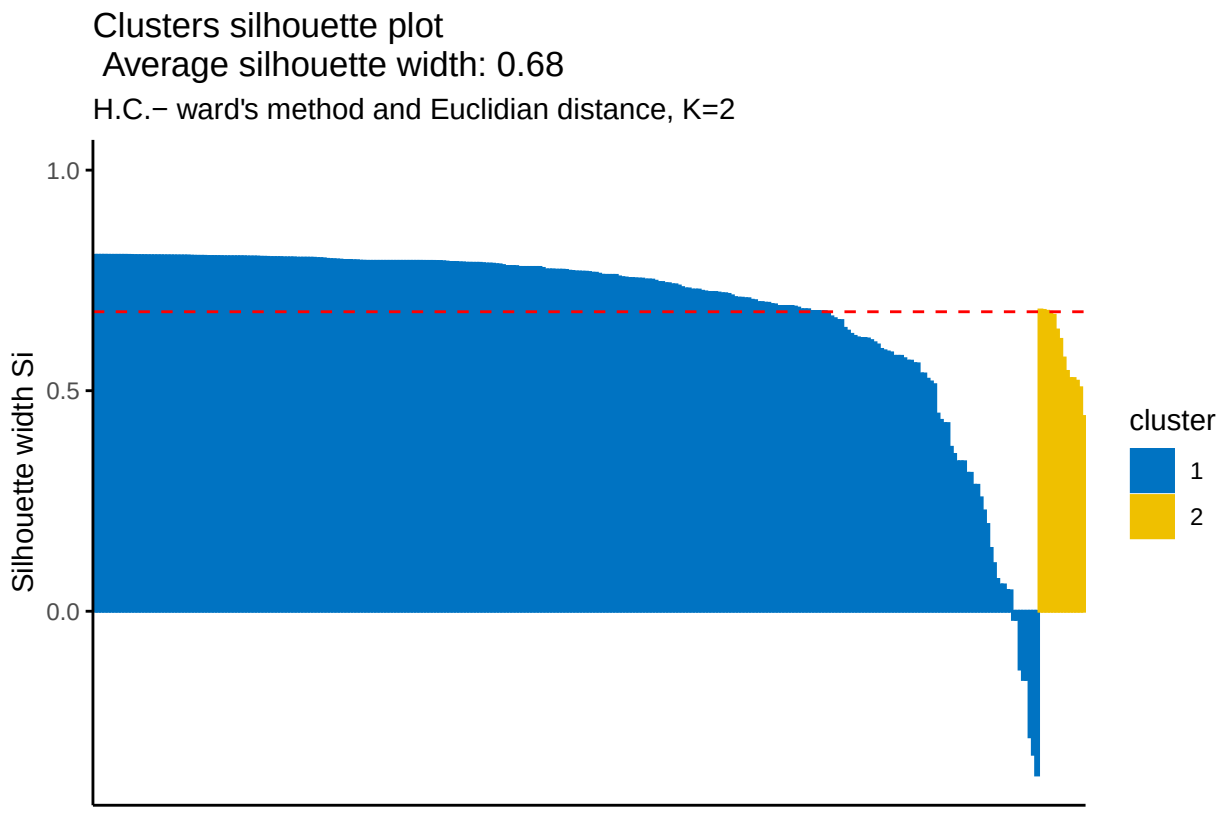
Silhouette width:

```
hclust<- eclust(heart.attk.sub,k=2 ,"hclust",hc_method = "ward.D2",nboot = 50)
silinfo <- hclust$silinfo
silinfo$avg.width
```

```
## [1] 0.6789307
```

```
fviz_silhouette(hclust, palette = "jco", ggtheme = theme_classic())+labs(
subtitle = "H.C.- ward's method and Euclidian distance, K=2", cex.sub= 0.5)
```

```
##   cluster size ave.sil.width
## 1       1  285          0.68
## 2       2   14          0.59
```




```
silinfo$clus.avg.widths
```

```
## [1] 0.6831903 0.5922174
```

```
sil <- hclust$silinfo$widths[, 1:3]
neg_sil_index_aver.eu<- which(sil[, "sil_width"] < 0)
sil[neg_sil_index_aver.eu, , drop = FALSE]
```

```
##     cluster neighbor   sil_width
## 299       1        2 -0.01850804
## 50        1        2 -0.01851028
## 251       1        2 -0.13115724
## 236       1        2 -0.15465786
## 86        1        2 -0.15465848
## 52        1        2 -0.28470476
## 276       1        2 -0.32422527
## 15        1        2 -0.37128168
```

The value of complete silhouette width indicates that on average the units are well enough clustered. As in particular, in each cluster the units are on average the same silhouette value with respect to the silhouette width. According to this index, 8 units that belong to cluster 1 are not well clustered: they should belong to cluster 2.

2.Dunn index:

```
library(fpc)
stats <- cluster.stats(dist(heart.attk_scale), hclust$cluster)
stats$dunn
```

```
## [1] 0.07341173
```

According to the Dunn index, the units are not clustered well enough.

External validation measures:

Confusion matrix:

According to the Confusion matrix, the number of clusters is equal to nominal values. The clusters found are 2 and the nominal variable can take 2 possible values.

```
table(heart.attk$smoking, hclust$cluster)
```

```
##
##       1   2
##   0 195   8
##   1  90   6
```

Most of the smokers (n = 90) has been classified in cluster 1 while cluster 2 have 6 number of values. The same happened for non-smokers (n=195) classified in cluster 1 while cluster 2 has 8 values.

2.Correct Rand Index:

```
smoking <- as.numeric(heart.attk$smoking)
stats<- cluster.stats(d = dist(heart.attk_scale), smoking, hclust$cluster)
stats$corrected.rand
```

```
## [1] 0.01447412
```

According to the Correct Rand Index, there is no agreement between the numerical value and the cluster solution. From -1 to +1, the agreement is very close to 0.

Meila's VI Index:

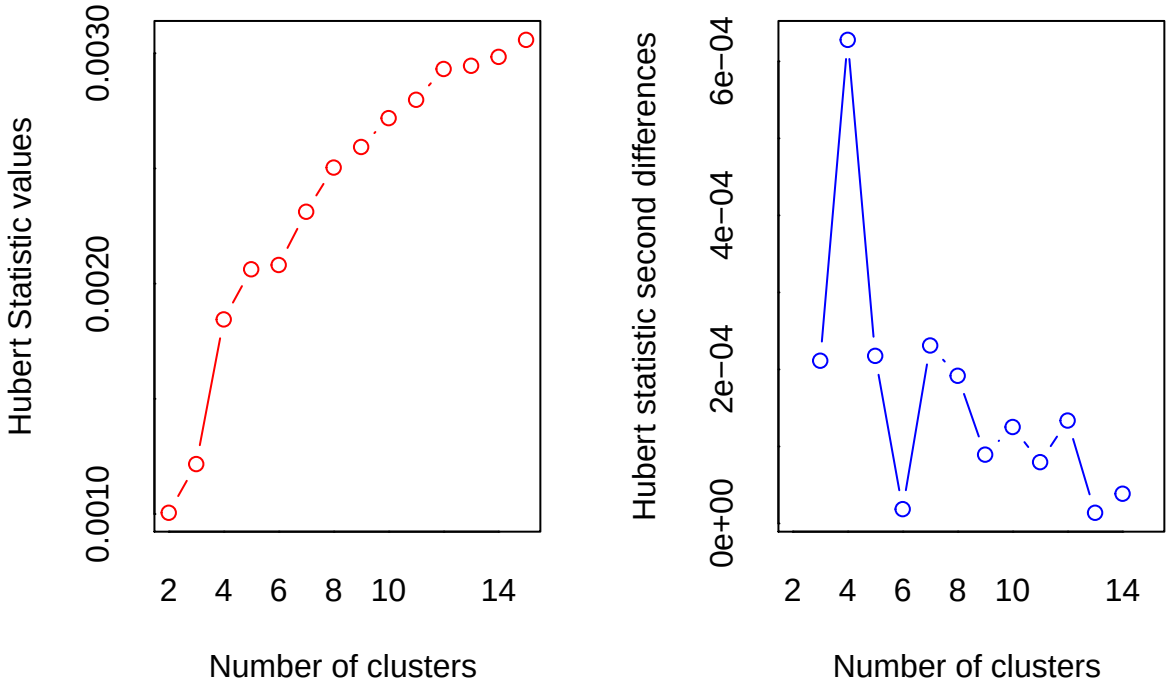
```
stats$vi
```

```
## [1] 0.8142365
```

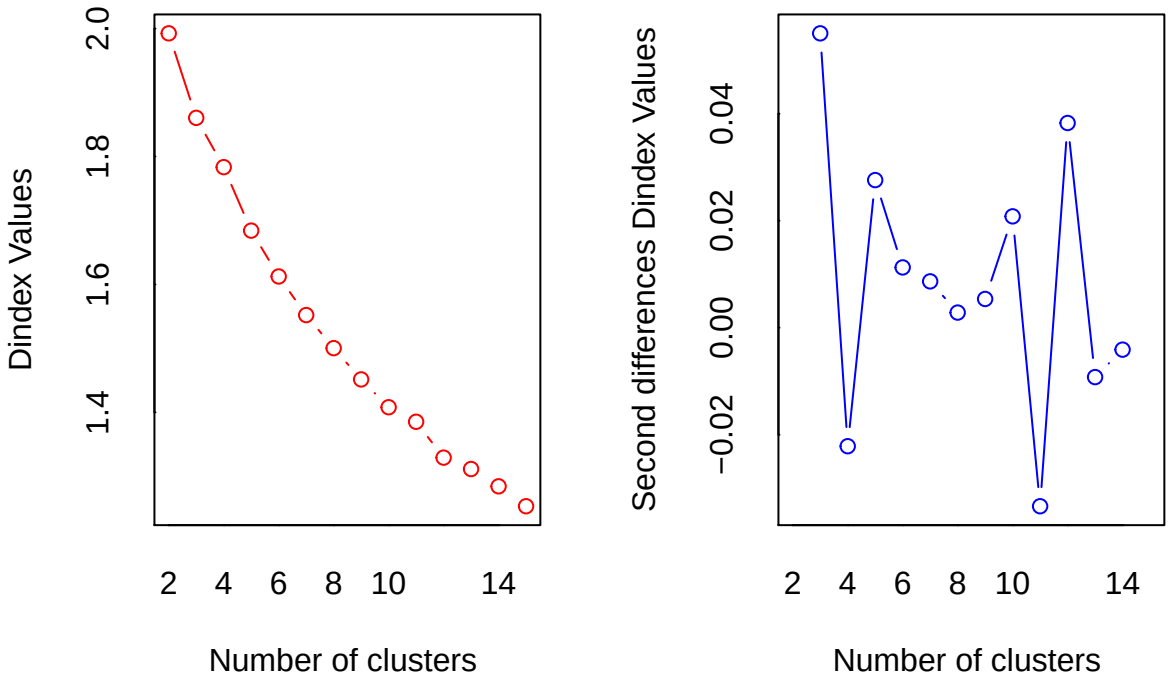
Partitional method:

K-means:

```
library(NbClust)
library(ggplot2)
nb <- NbClust(heart.attk_scale, min.nc=2, max.nc=15, method="kmeans")
```




```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```

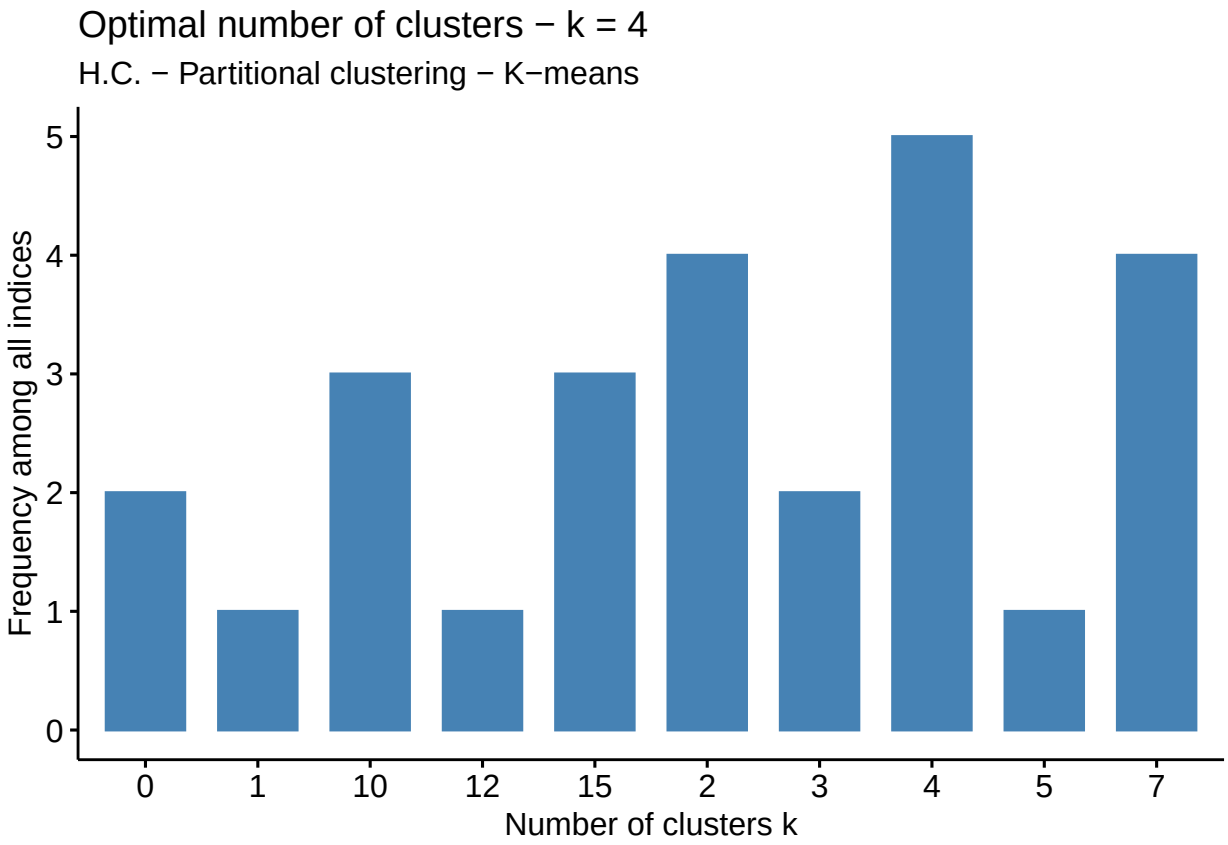



```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 4 proposed 2 as the best number of clusters
## * 2 proposed 3 as the best number of clusters
## * 5 proposed 4 as the best number of clusters
## * 1 proposed 5 as the best number of clusters
## * 4 proposed 7 as the best number of clusters
## * 3 proposed 10 as the best number of clusters
## * 1 proposed 12 as the best number of clusters
## * 3 proposed 15 as the best number of clusters
##
##                     ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  4
##
##
## *******************************************************************
```

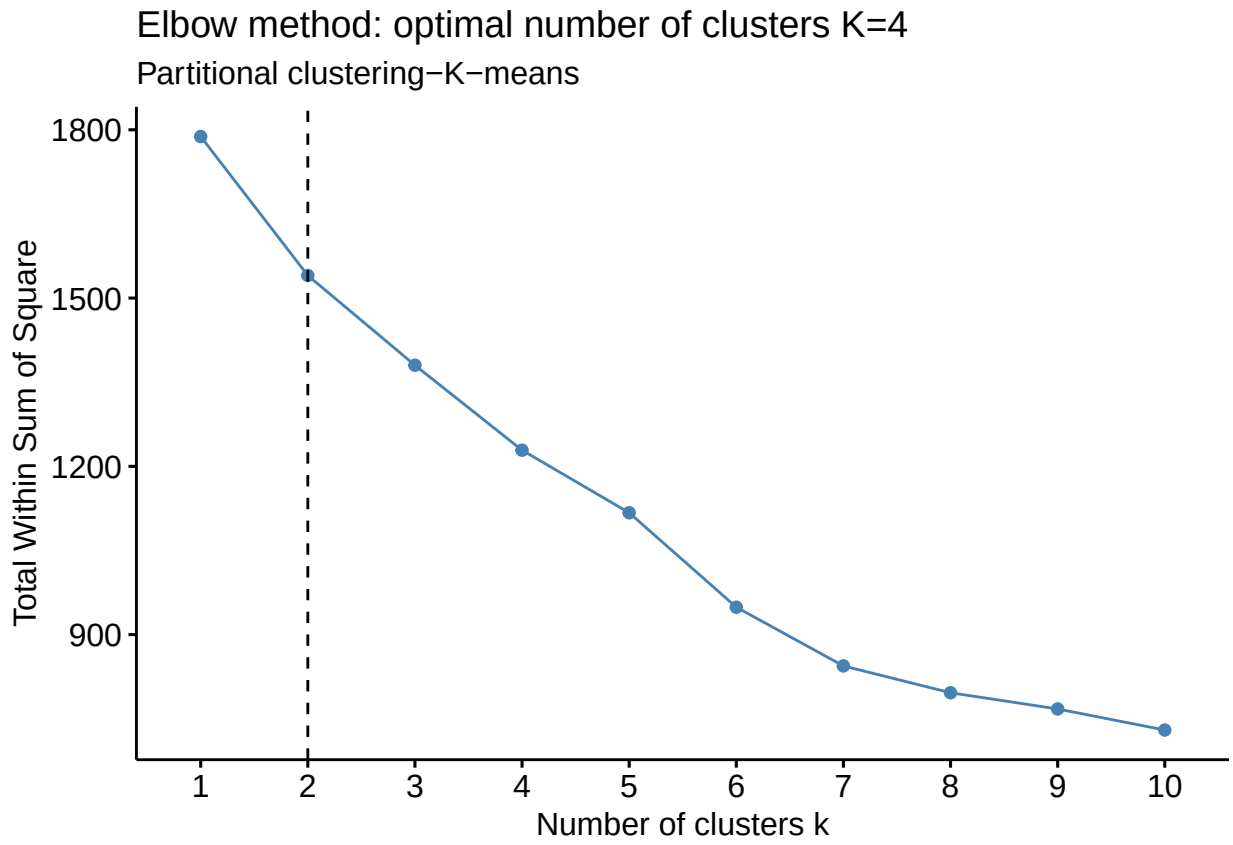
```
fviz_nbclust(nb)+labs(subtitle = "H.C. - Partitional clustering - K-means")
```

```
## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
```

```
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") .viz_NbClust(x, print.summary, : the
## condition has length > 1 and only the first element will be used

## Warning in if (class(best_nc) == "numeric") print(best_nc) else if
## (class(best_nc) == : the condition has length > 1 and only the first element
## will be used

## Warning in if (class(best_nc) == "matrix") {: the condition has length > 1 and
## only the first element will be used

## Among all indices:
## ===================
## * 2 proposed  0 as the best number of clusters
## * 1 proposed  1 as the best number of clusters
## * 4 proposed  2 as the best number of clusters
## * 2 proposed  3 as the best number of clusters
## * 5 proposed  4 as the best number of clusters
## * 1 proposed  5 as the best number of clusters
## * 4 proposed  7 as the best number of clusters
## * 3 proposed  10 as the best number of clusters
## * 1 proposed  12 as the best number of clusters
## * 3 proposed  15 as the best number of clusters
##
## Conclusion
## ==========================
## * According to the majority rule, the best number of clusters is  4 .
```




```
fviz_nbclust(heart.attk_scale, kmeans, method = "wss") +
geom_vline(xintercept = 2, linetype = 2)+
labs(title= "Elbow method: optimal number of clusters K=4", subtitle = "Partitional clustering-K-means")
```




```
(km.res<- kmeans(heart.attk_scale, 4, nstart = 25))
```

```
## K-means clustering with 4 clusters of sizes 7, 161, 55, 76
##
## Cluster means:
##           age creatinine_phosphokinase ejection_fraction  platelets
## 1 -0.07010562               5.16063685        -0.0191358  0.1434496
## 2 -0.42605503              -0.03284006        -0.4311208 -0.1059488
## 3  0.75225463              -0.23667148        -0.3296871 -0.3738796
## 4  0.36462625              -0.23447734         1.1536471  0.4818024
##   serum_creatinine serum_sodium
## 1        0.4450475    0.4410245
## 2       -0.2648033    0.1876493
## 3        1.0846878   -1.1512724
## 4       -0.2649977    0.3950169
##
## Clustering vector:
##   [1] 3 1 3 2 3 3 2 4 4 3 3 2 2 2 2 3 4 2 2 3 2 2 2 4 3 4 4 4 3 3 3 3 2 2 4 3 4
##  [38] 4 2 2 3 2 4 4 4 2 2 4 3 2 2 2 1 4 3 3 3 2 2 2 1 3 2 2 4 3 2 2 2 2 2 2 1 4
##  [75] 2 2 4 2 3 4 4 4 3 3 2 4 2 4 2 2 4 4 4 2 2 4 2 4 2 2 2 4 2 1 4 4 2 2 2 4 4
## [112] 2 2 4 3 2 4 4 4 3 4 2 2 2 3 2 3 4 2 2 4 3 2 4 1 4 4 3 2 2 3 2 2 2 2 2 2 4
## [149] 3 2 3 4 2 2 2 2 2 2 4 4 2 2 4 2 2 3 4 3 2 2 2 1 4 2 2 4 2 2 4 2 2 2 2 3 2
## [186] 2 2 2 2 2 3 4 2 2 2 4 2 2 2 3 2 4 4 3 4 4 2 3 2 2 2 4 4 2 2 3 4 3 2 2 3 4
## [223] 2 2 2 3 2 2 3 3 3 3 2 2 2 4 4 3 2 2 4 2 2 4 2 2 2 3 2 2 2 2 2 3 4 2 2 2 2
## [260] 4 2 2 3 4 2 2 2 2 2 2 2 2 3 2 2 2 4 2 2 4 2 3 3 2 2 2 2 4 2 4 4 2 2 2 2 2
```

```
## [297] 4 2 4
##
## Within cluster sum of squares by cluster:
## [1]  69.89188 437.56376 354.35835 322.99384
##  (between_SS / total_SS =  33.7 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
aggregate(heart.attk.sub, by=list(cluster=km.res$cluster), mean)
```

```
##   cluster      age creatinine_phosphokinase ejection_fraction platelets
## 1       1 60.00000                5589.1429          37.85714  277388.0
## 2       2 55.76605                 549.9752          32.98137  252995.8
## 3       3 69.78182                 352.2000          34.18182  226791.0
## 4       4 65.17105                 354.3289          51.73684  310480.3
##   serum_creatinine serum_sodium
## 1         1.854286     138.5714
## 2         1.119938     137.4534
## 3         2.516000     131.5455
## 4         1.119737     138.3684
```

As the results shows first cluster contains lower units of variables except 2 variables.

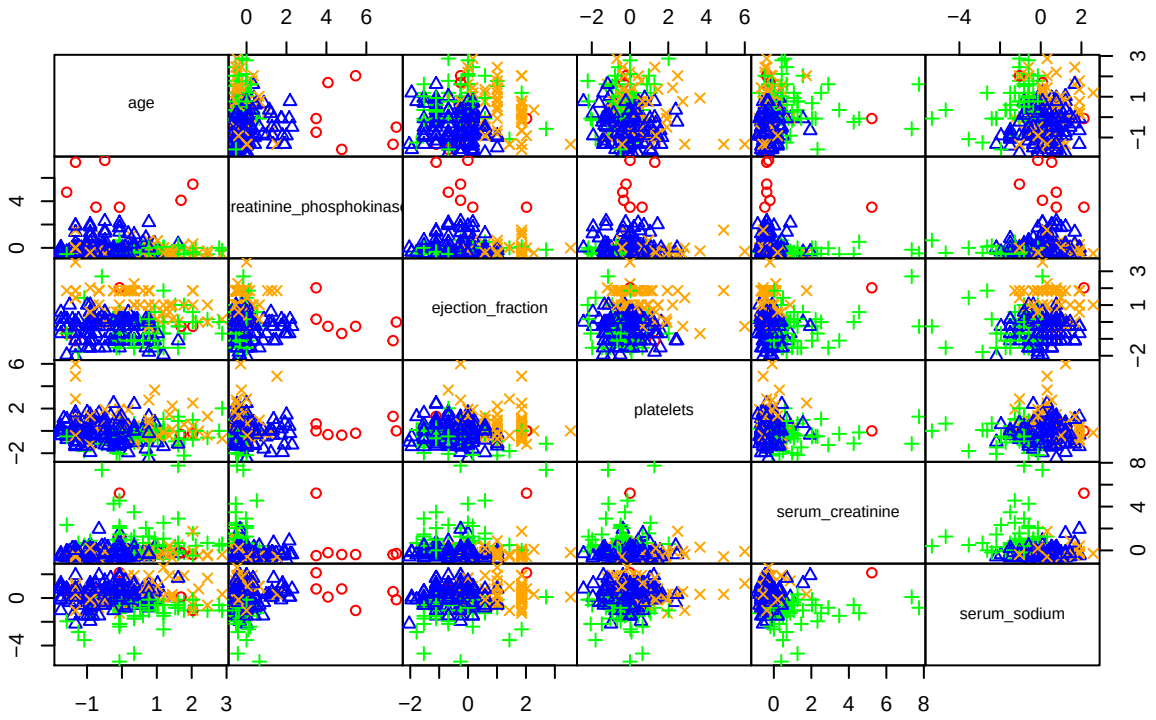
```
dd <- cbind(heart.attk.sub, cluster = km.res$cluster)
head(dd)
```

```
##   age creatinine_phosphokinase ejection_fraction platelets serum_creatinine
## 1  75                      582                20    265000              1.9
## 2  55                     7861                38    263358              1.1
## 3  65                      146                20    162000              1.3
## 4  50                      111                20    210000              1.9
## 5  65                      160                20    327000              2.7
## 6  90                       47                40    204000              2.1
##   serum_sodium cluster
## 1          130       3
## 2          136       1
## 3          129       3
## 4          137       2
## 5          116       3
## 6          132       3
```

```
cl <- km.res$cluster
table(cl)
```

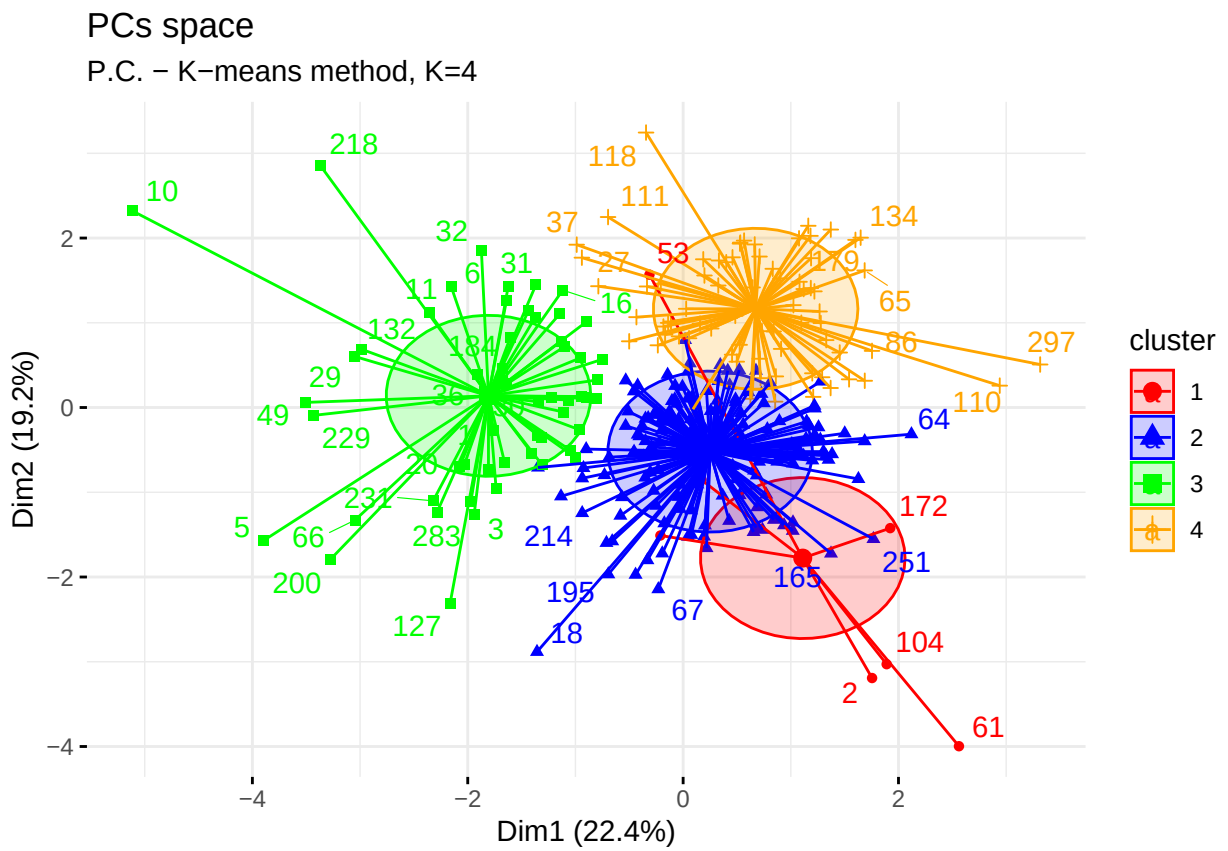
```
## cl
##   1   2   3   4
##   7 161  55  76
```

```
pairs(heart.attk_scale, gap=0, pch=cl, main="Original space\nP.C.-K-means method, K=4", cex.main= 1, col
```




```
fviz_cluster(km.res, data = heart.attk_scale, palette = c("red", "blue","green","orange"),
ellipse.type = "euclid", star.plot = TRUE, repel = TRUE,
main= "PCs space", ggtheme = theme_minimal())+
labs(subtitle = "P.C. - K-means method, K=4")
```

```
## Warning: ggrepel: 254 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



According to the function "NbClust", the best number of clusters, applying partitional clustering method and using K-means method is 4: cluster 1 with 161 units, cluster 2 with 76 units, cluster 3 with 7 units and cluster 4 with 55 units. The arability between different clusters: 33.7% of the total variability is explained by the separation between clusters. In the PCs space, there is not separation between clusters.

To examine the goodness of clustering algorithm results, we have to analyst internal and external validation measures.

Internal validation measures:

1.Silhouette width:

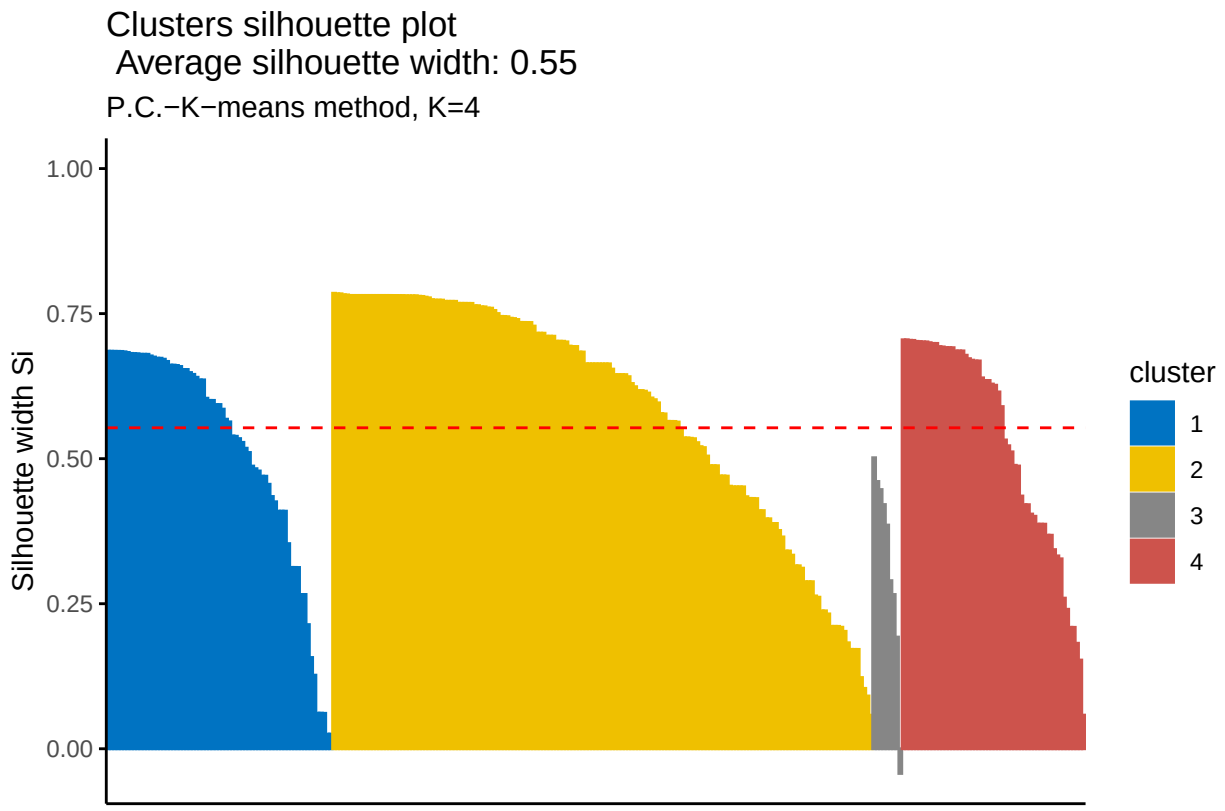
```
hclust<- eclust(heart.attk.sub,k=4 ,"kmeans", nstart=25, graph = FALSE)
silinfo <- hclust$silinfo
silinfo$avg.width
```

```
## [1] 0.5532114
```

```
fviz_silhouette(hclust, palette = "jco", ggtheme = theme_classic())+
labs(subtitle = "P.C.-K-means method, K=4")
```

```
##   cluster size ave.sil.width
## 1       1   69          0.51
```

```
## 2       2  165          0.59
## 3       3    9          0.32
## 4       4   56          0.53
```




```
silinfo$clus.avg.widths
```

```
## [1] 0.5135899 0.5893795 0.3245948 0.5322062
```

```
sil <- hclust$silinfo$widths[, 1:3]
neg_sil_index_aver.eu<- which(sil[, "sil_width"] < 0)
sil[neg_sil_index_aver.eu, , drop = FALSE]
```

```
##     cluster neighbor   sil_width
## 213       3        4 -0.04254567
```

The value of complete silhouette width indicates that on average the units are well enough clustered. As in particular, in each cluster the units are on average the same silhouette value with respect to the silhouette width. According to this index, 1 units that belong to cluster 3 are not well clustered: it should belong to cluster 4.

2.Dunn index:

```
stats <- cluster.stats(dist(heart.attk_scale), hclust$cluster)
stats$dunn
```

```
## [1] 0.02370275
```

According to the Dunn index, the units are not clustered well enough.

External validation measures:

Confusion matrix:

According to the Confusion matrix, the number of clusters is equal to nominal values. The clusters found are 2 and the nominal variable can take 2 possible values.

```
table(heart.attk$smoking, hclust$cluster)
```

```
##
##       1   2   3   4
##   0  44 116   6  37
##   1  25  49   3  19
```

Most of the smokers (n = 116) has been classified in cluster 2 while cluster 1 have 44 number of values, cluster 3 have 6 values and cluter 4 have 37 values. The same happened for non-smokers (n=49) classified in cluster 2 while cluster 1 has 25 value, cluster 3 have 3 values and cluster 4 have 19 values.

2.Correct Rand Index:

```
smoking <- as.numeric(heart.attk$smoking)
stats<- cluster.stats(d = dist(heart.attk_scale), smoking, hclust$cluster)
stats$corrected.rand
```

```
## [1] 0.01128107
```

According to the Correct Rand Index, there is no agreement between the numerical values and the cluster solution. From -1 to +1, the agreement is very close to 0.

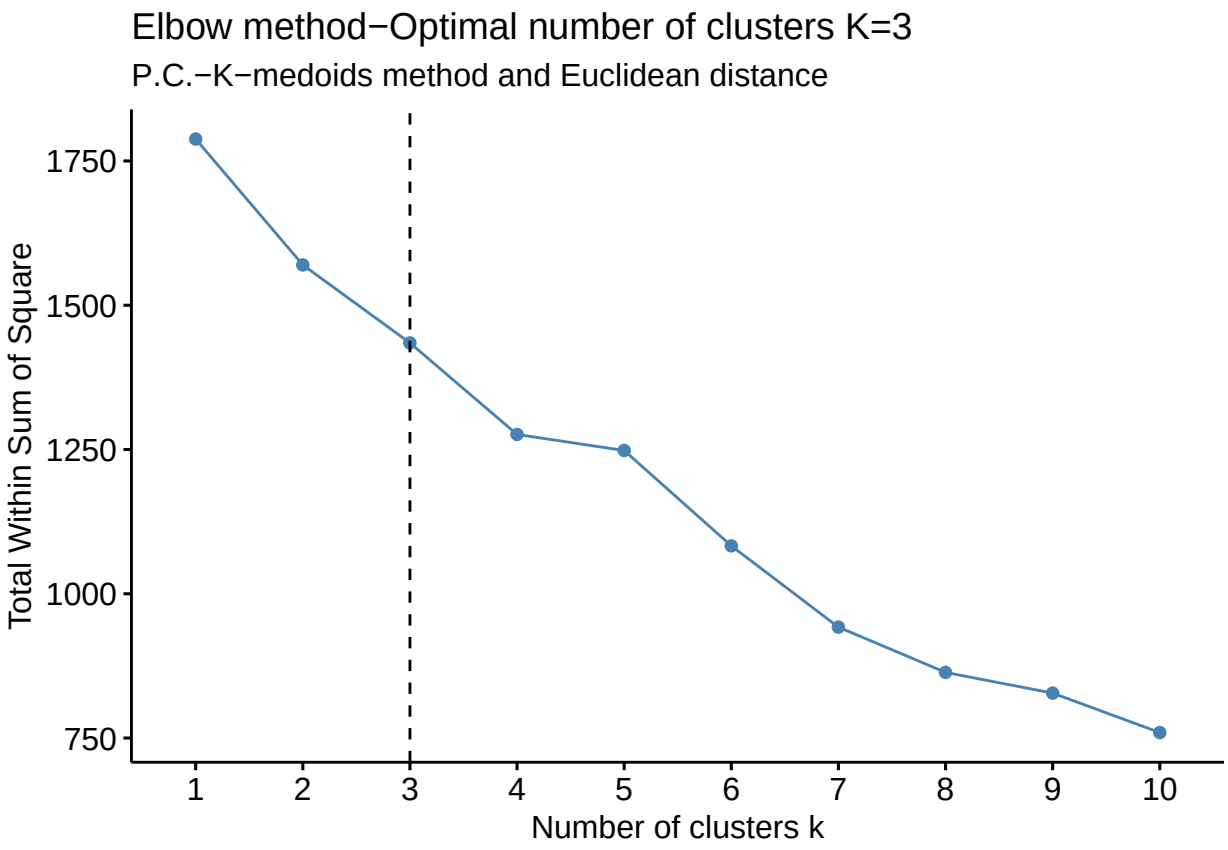
Meila's VI Index:

```
stats$vi
```

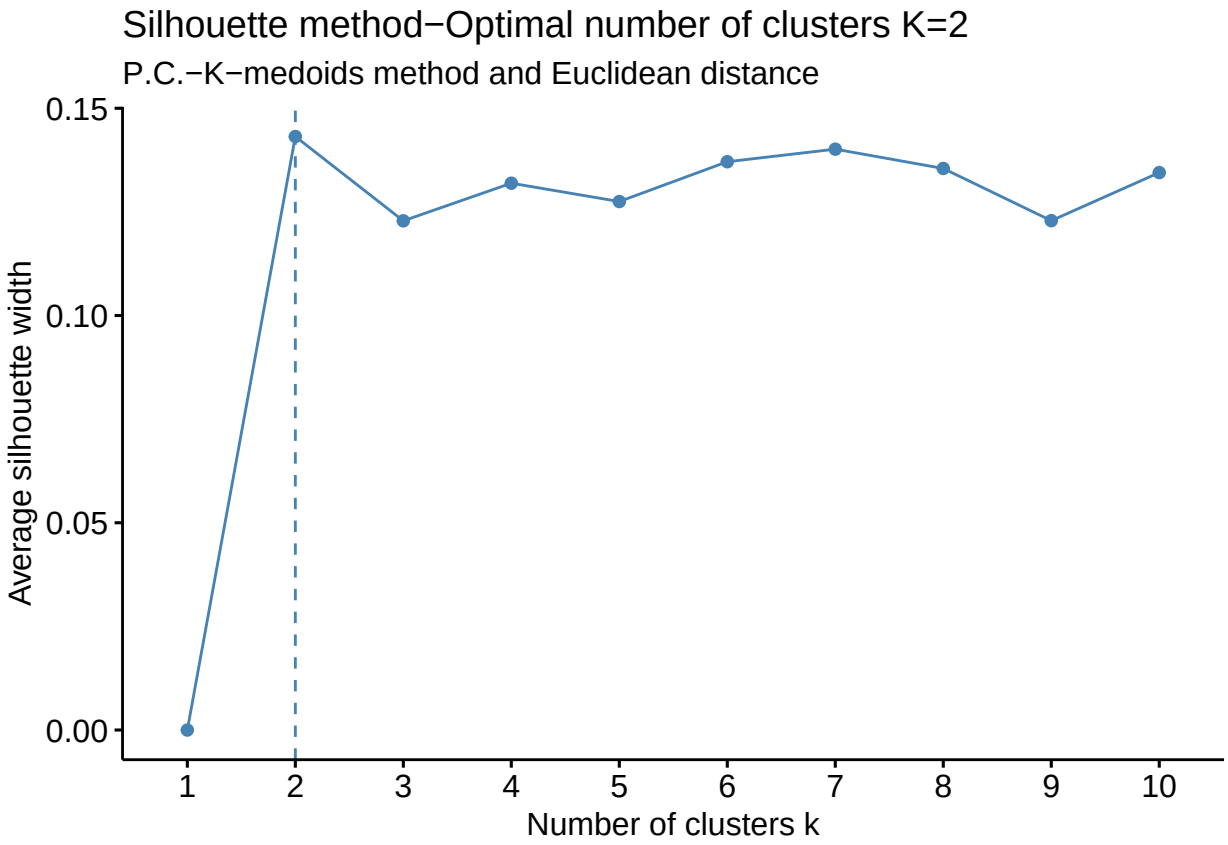
```
## [1] 1.709746
```

Partitioning around medoids (PAM) method and Euclidean distance:

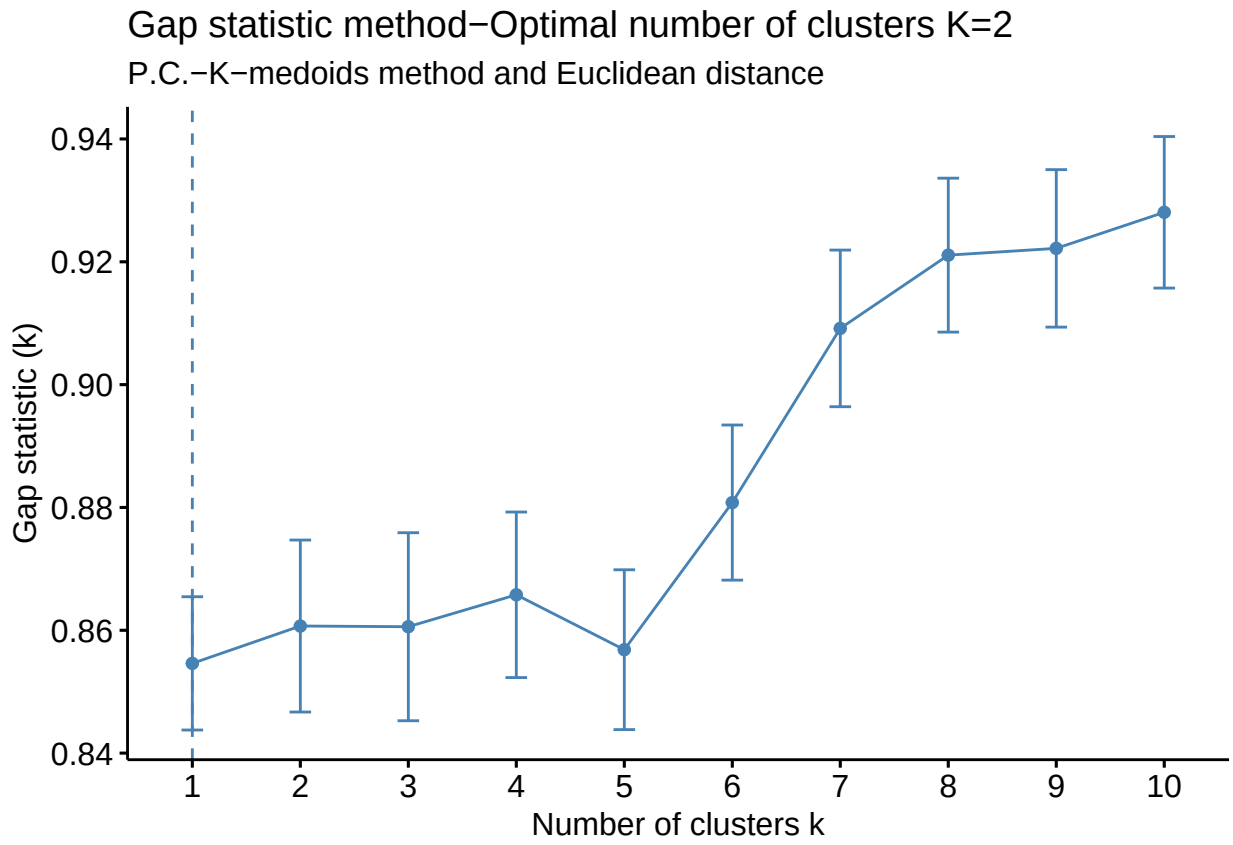
```
fviz_nbclust(heart.attk_scale, cluster::pam, method = "wss") +
geom_vline(xintercept = 3, linetype = 2)+
labs(title = "Elbow method-Optimal number of clusters K=3",
subtitle="P.C.-K-medoids method and Euclidean distance", cex.sub= 0.5)
```




```
fviz_nbclust(heart.attk_scale, cluster::pam, method = "silhouette")+
labs(title = "Silhouette method-Optimal number of clusters K=2",
subtitle="P.C.-K-medoids method and Euclidean distance", cex.sub= 0.5)
```




```
fviz_nbclust(heart.attk_scale, cluster::pam, method = "gap_stat", nboot = 500)+
labs(title = "Gap statistic method-Optimal number of clusters K=2",
subtitle="P.C.-K-medoids method and Euclidean distance", cex.sub= 0.5)
```



In order to find the optimal number of clusters, three indices were used. The elbow method does not seem to give a very clear result; according to Total within sum of square, the suggested number of clusters is considered to be 2. Silhouette method suggests 2 clusters. Gap statistics 2 cluster, i.e. no presence of clusters in the data. It was decided to proceed by identifying 2 clusters.

```
library(cluster)
(pam.res <- pam(heart.attk_scale, 2, metric = "euclidean"))
```

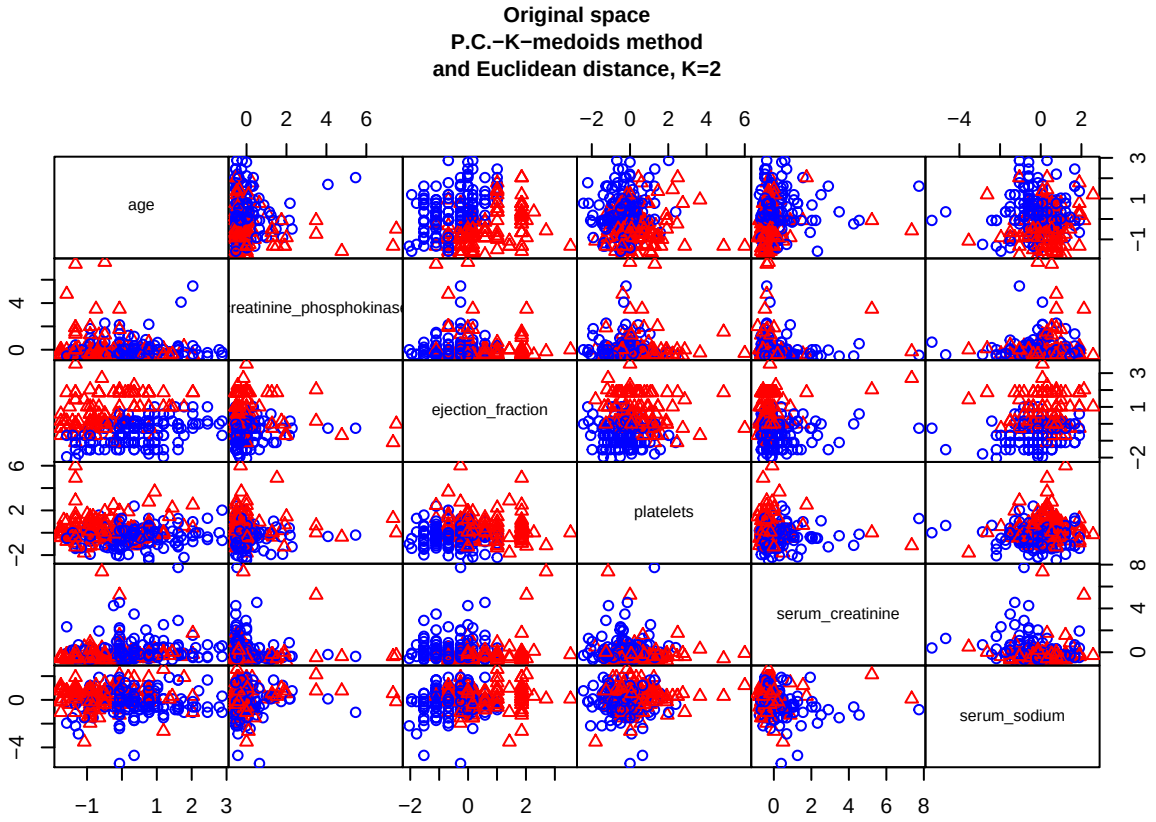
```
## Medoids:
##       ID        age creatinine_phosphokinase ejection_fraction  platelets
## [1,] 242  0.3502458              0.000165451        -0.6830351 -0.1468038
## [2,] 235 -0.6585976              0.000165451         0.5844090  0.4257686
##      serum_creatinine serum_sodium
## [1,]      -0.09074788  -0.14173853
## [2,]      -0.28407611   0.08489153
## Clustering vector:
##   [1] 1 2 1 1 1 1 1 2 2 1 1 1 1 2 2 1 1 1 1 2 1 1 1 2 1 1 1 1 1 1 1 2 2 2 1 1
##  [38] 2 1 2 1 1 1 1 2 2 1 2 1 2 1 1 2 2 1 1 1 2 1 1 2 2 2 2 2 1 1 1 1 1 2 1 1 2
##  [75] 1 1 2 2 1 2 1 2 1 1 1 2 1 2 2 1 1 2 2 1 2 2 1 2 1 1 1 1 1 2 2 2 2 2 1 2 2
## [112] 1 1 2 1 1 2 2 2 1 2 1 1 1 1 2 1 2 1 1 2 1 2 2 1 1 2 1 1 1 1 2 2 2 1 2 1 2
## [149] 1 1 1 2 2 2 1 1 2 1 1 2 1 2 2 1 2 1 2 1 1 1 2 2 2 1 1 2 1 2 2 2 2 1 1 1 1
## [186] 1 2 1 1 2 1 2 2 1 1 1 2 1 1 1 1 2 2 1 1 2 2 1 2 2 1 2 2 1 1 1 2 2 1 2 1 2
## [223] 2 1 2 2 1 1 1 1 1 1 2 2 2 2 2 1 1 2 2 1 2 1 2 1 1 1 2 2 2 2 2 1 2 2 1 2 1
## [260] 2 2 1 1 2 1 2 1 1 2 2 1 2 1 2 1 2 1 1 1 2 1 1 1 1 2 2 1 2 1 1 2 1 2 1 1 2
## [297] 2 2 2
## Objective function:
```

```
##    build     swap
## 2.059728 1.994025
##
## Available components:
##  [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"
##  [6] "clusinfo"   "silinfo"    "diss"       "call"       "data"
```

```
pam.res$clusinfo
```

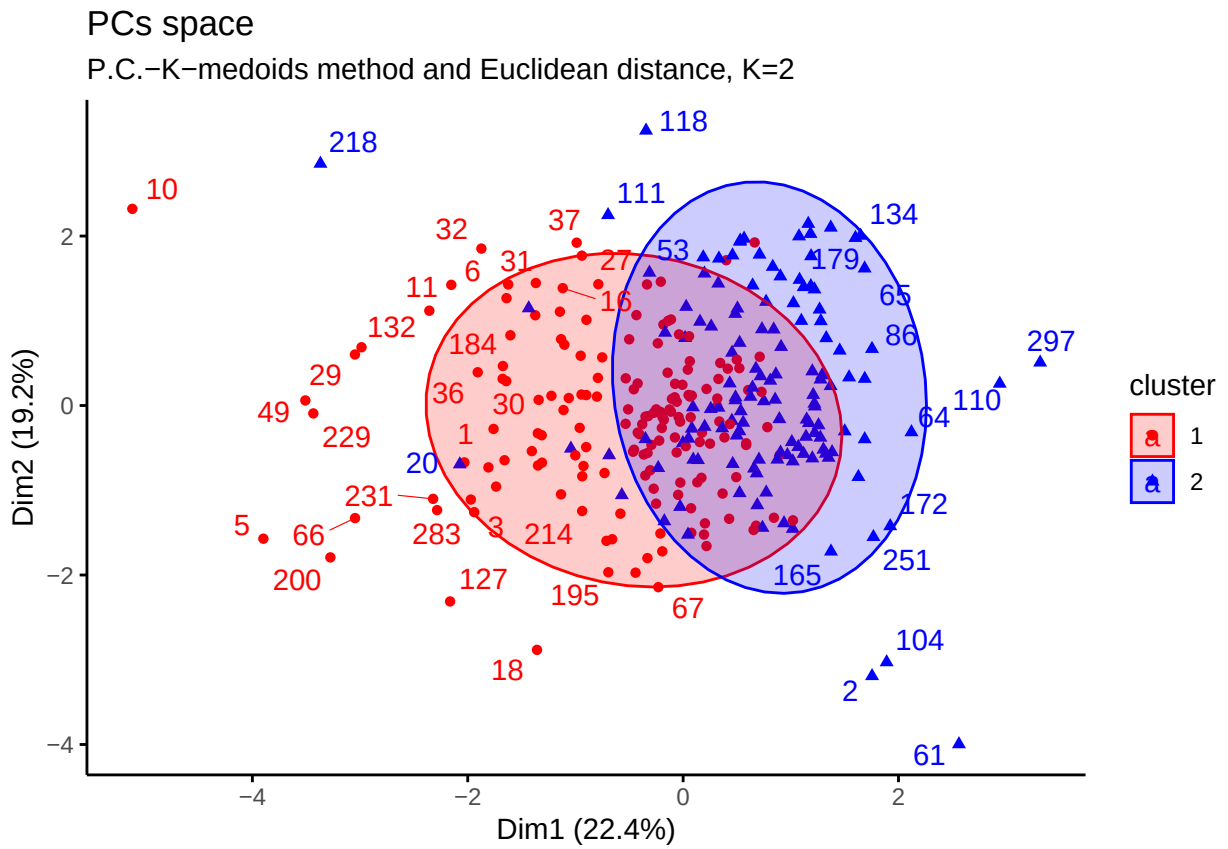
```
##      size max_diss  av_diss diameter separation
## [1,]  165 8.110504 1.932540 10.17769  0.3290423
## [2,]  134 8.080206 2.069733 11.71431  0.3290423
```

```
cc <- pam.res$cluster
pairs(heart.attk_scale, gap=0, pch=cc, main="Original space\nP.C.-K-medoids method
and Euclidean distance, K=2", cex.main= 0.7,
col=c("blue","red", "green", "orange")[cc])
```




```
fviz_cluster(pam.res, palette = c("red","blue"), ellipse.type = "t",
repel = TRUE, main= "PCs space", ggtheme = theme_classic())+labs(
subtitle = "P.C.-K-medoids method and Euclidean distance, K=2", cex.sub= 0.5)
```

```
## Warning: ggrepel: 254 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



Applying partitioning clustering method and using PAM algorithm and Euclidean distance, TWO clusters are composed in this way: cluster 1 with 165 units, cluster 2 with 134 units.

To examine the goodness of clustering algorithm results, we have to analyst internal and external validation measures.

Internal validation measures:

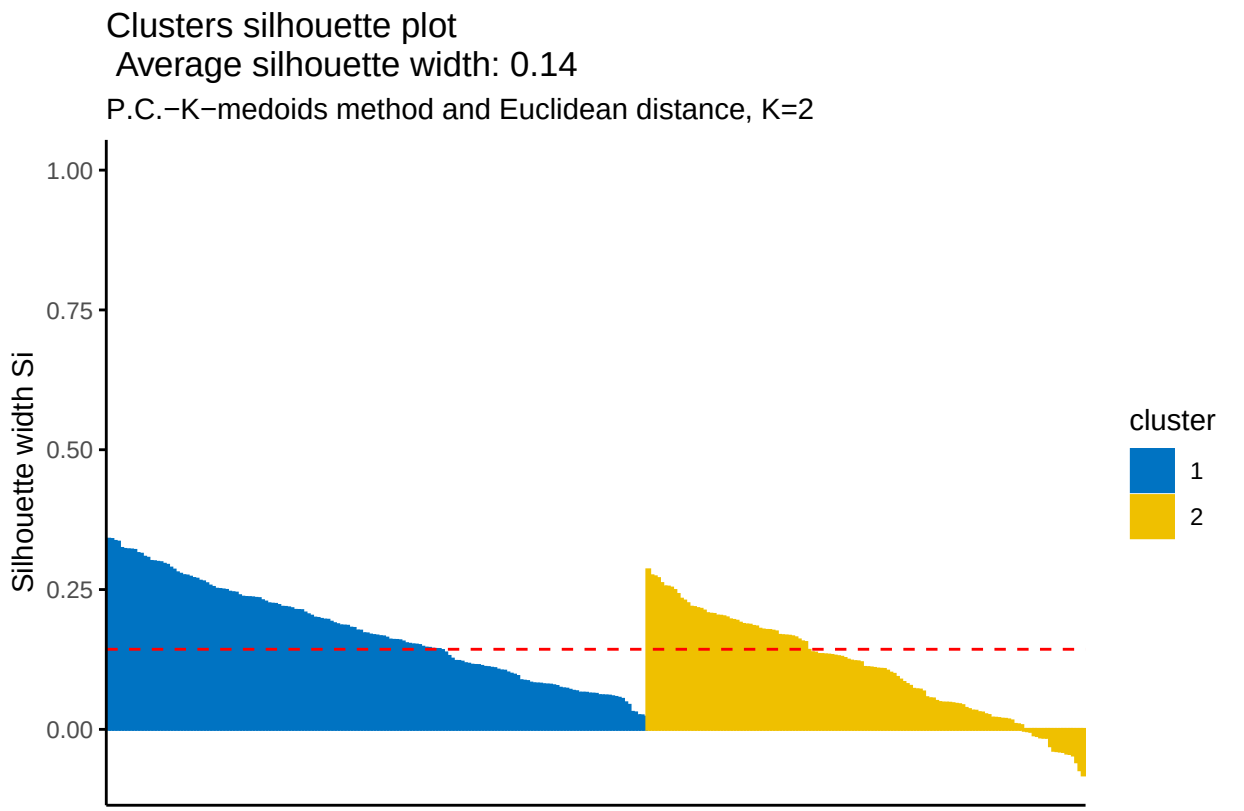
Silhouette width:

```
hclust<- eclust(heart.attk.sub,k=2 ,"pam", graph = FALSE, hc_metric = "euclidean")
silinfo <- hclust$silinfo
silinfo$avg.width
```

```
## [1] 0.4630251
```

```
fviz_silhouette(pam.res, palette = "jco",
ggtheme = theme_classic())+labs(
subtitle = "P.C.-K-medoids method and Euclidean distance, K=2", cex.sub= 0.5)
```

```
##   cluster size ave.sil.width
## 1       1  165          0.17
## 2       2  134          0.11
```




```
silinfo$clus.avg.widths
```

```
## [1] 0.3572776 0.5914886
```

```
sil <- hclust$silinfo$widths[, 1:3]
neg_sil_index_aver.eu<- which(sil[, "sil_width"] < 0)
sil[neg_sil_index_aver.eu, , drop = FALSE]
```

```
##     cluster neighbor    sil_width
## 239       1        2 -0.008213382
## 233       1        2 -0.062316672
## 55        1        2 -0.062318290
## 243       1        2 -0.062324834
## 98        1        2 -0.062337457
## 60        1        2 -0.088962534
## 63        1        2 -0.088987015
## 97        1        2 -0.088988538
## 12        1        2 -0.115348613
## 258       1        2 -0.115352555
## 113       1        2 -0.141336720
```

The value of complete silhouette width indicates that on average the units are well enough clustered. As in particular, in each cluster the units are on average the same silhouette value with respect to the silhouette width. According to this index, 11 units that belong to cluster 1 are not well clustered: they should belong to cluster 2.

2.Dunn index:

According to the Dunn index, the units are not clustered well enough.

External validation measures:

Confusion matrix:

According to the Confusion matrix, the number of clusters is equal to nominal values. The clusters found are 2 and the nominal variable can take 2 possible values.

```
table(heart.attk$smoking, hclust$cluster)
```

```
##
##       1   2
##   0 109  94
##   1  55  41
```

Most of the smokers (n = 55) has been classified in cluster 1 while cluster 2 have 41 values.

Correct Rand Index:

```
smoking <- as.numeric(heart.attk$smoking)
stats<- cluster.stats(d = dist(heart.attk_scale), smoking, hclust$cluster)
stats$corrected.rand
```

```
## [1] -0.00410881
```

According to the Correct Rand Index, there is no agreement between the numerical values and the cluster solution. From -1 to +1, the agreement is very close to 0.

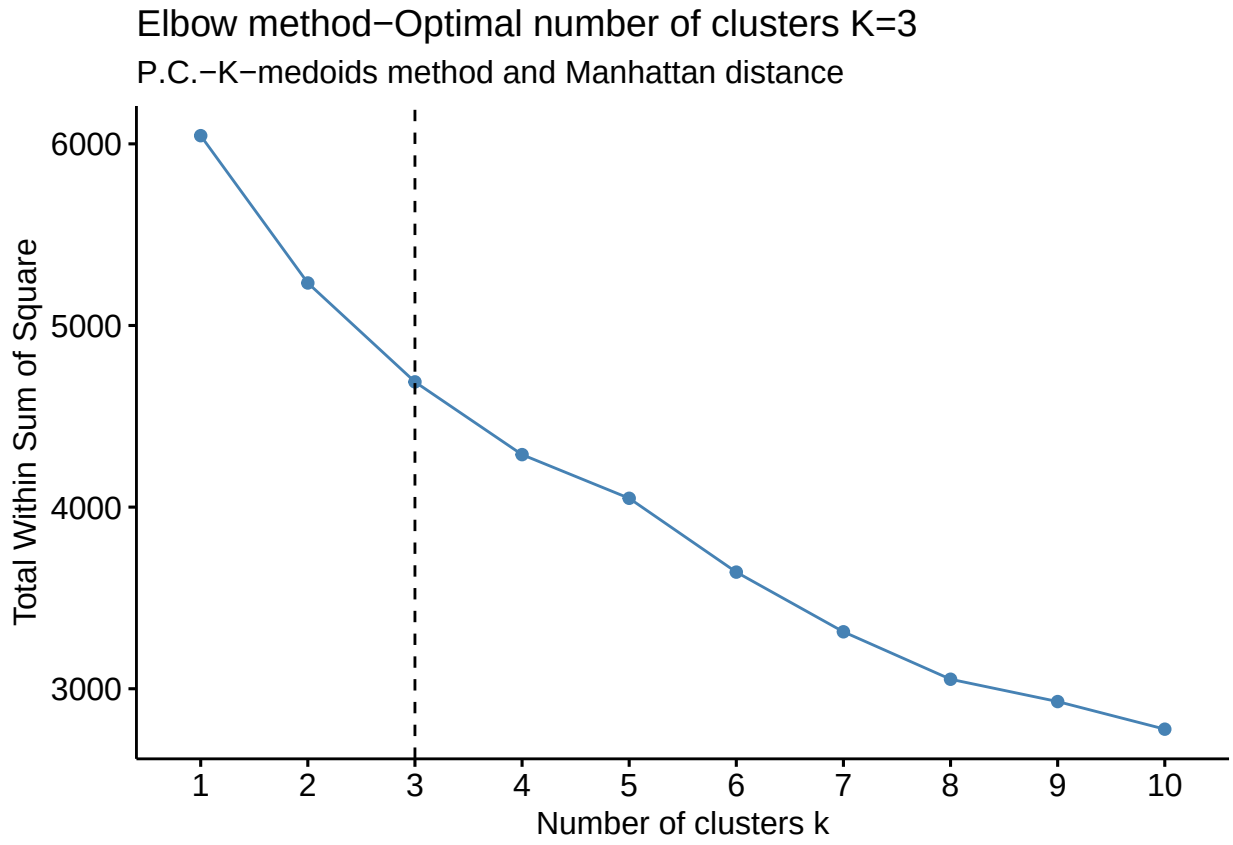
Meila's VI Index:

```
stats$vi
```

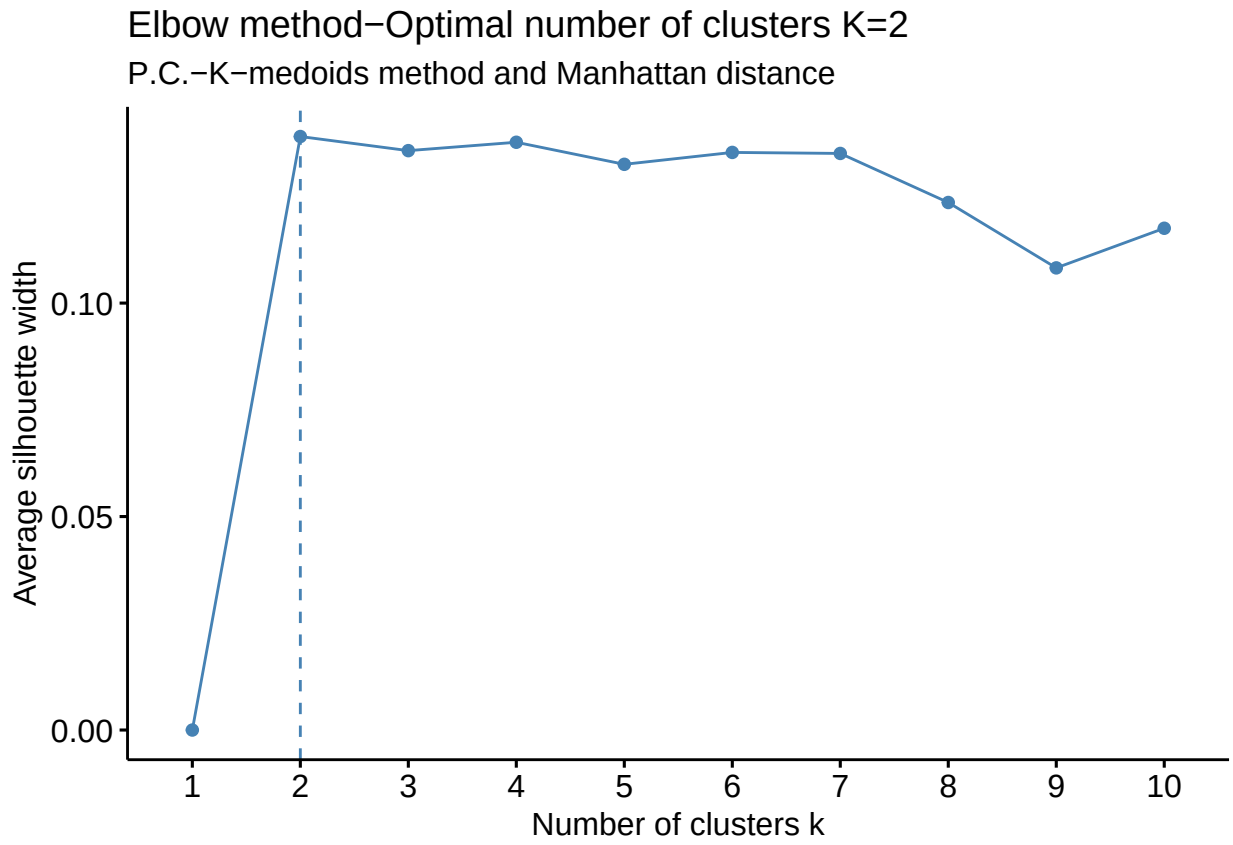
```
## [1] 1.314969
```

Partitioning around medoids (PAM) method and Manhattan distance:

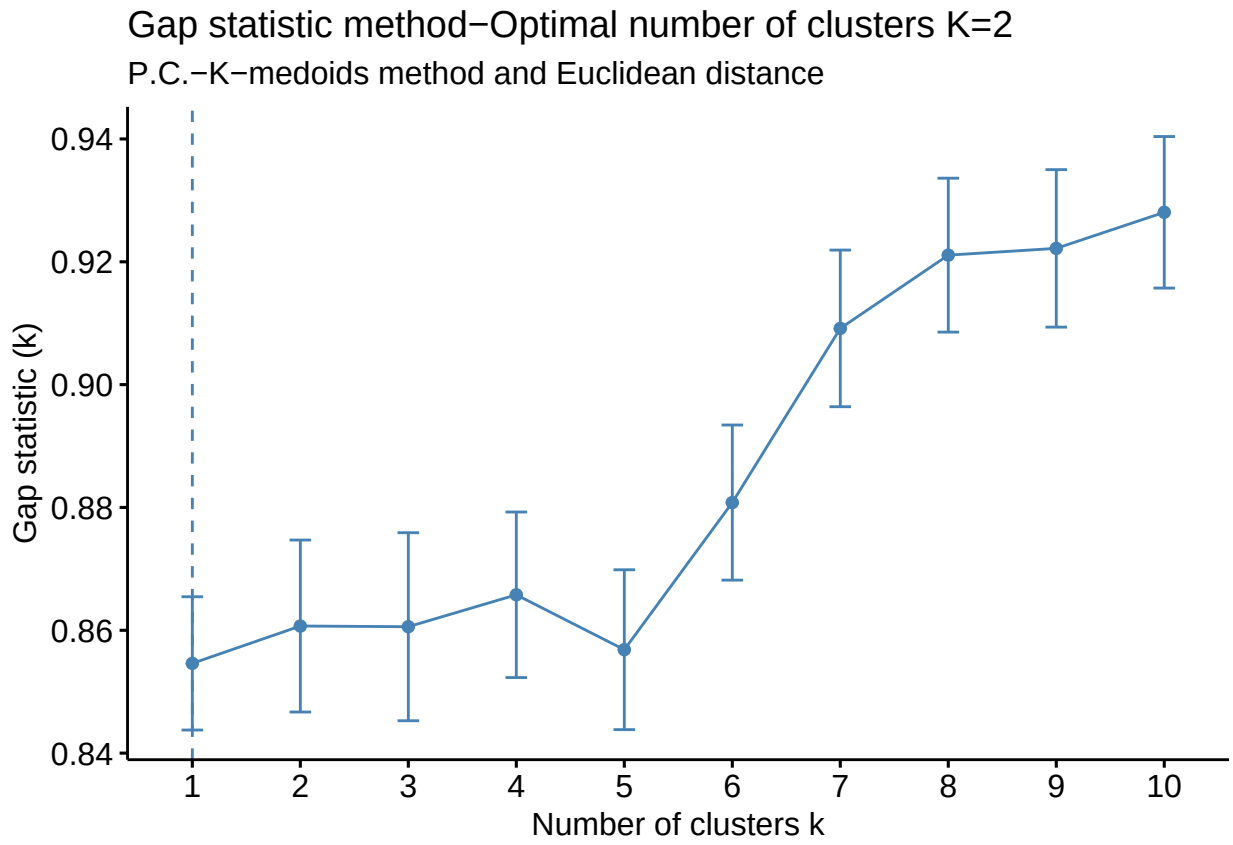
```
fviz_nbclust(heart.attk_scale, cluster::pam, method = "wss",
diss = dist(heart.attk_scale, method = "manhattan"))+
geom_vline(xintercept = 3, linetype = 2)+ labs(title = "Elbow method-Optimal number of clusters K=3",
subtitle="P.C.-K-medoids method and Manhattan distance", cex.sub= 0.5)
```




```
fviz_nbclust(heart.attk_scale, cluster::pam, method = "silhouette", diss = dist(heart.attk_scale, metho
```




```
fviz_nbclust(heart.attk_scale, cluster::pam, method = "gap_stat", nboot = 500, diss=dist(heart.attk_scal
```



In order to find the optimal number of clusters, three indices were used. Th elbow method does not seem suggest k=3. Silhouette method suggests 2 clusters. Gap statistics 2 cluster.It was decided to proceed by identifying 2 clusters.

```
library(cluster)
set.seed(123)
(pam.res <- pam(heart.attk_scale, 2, metric="manhattan"))
```

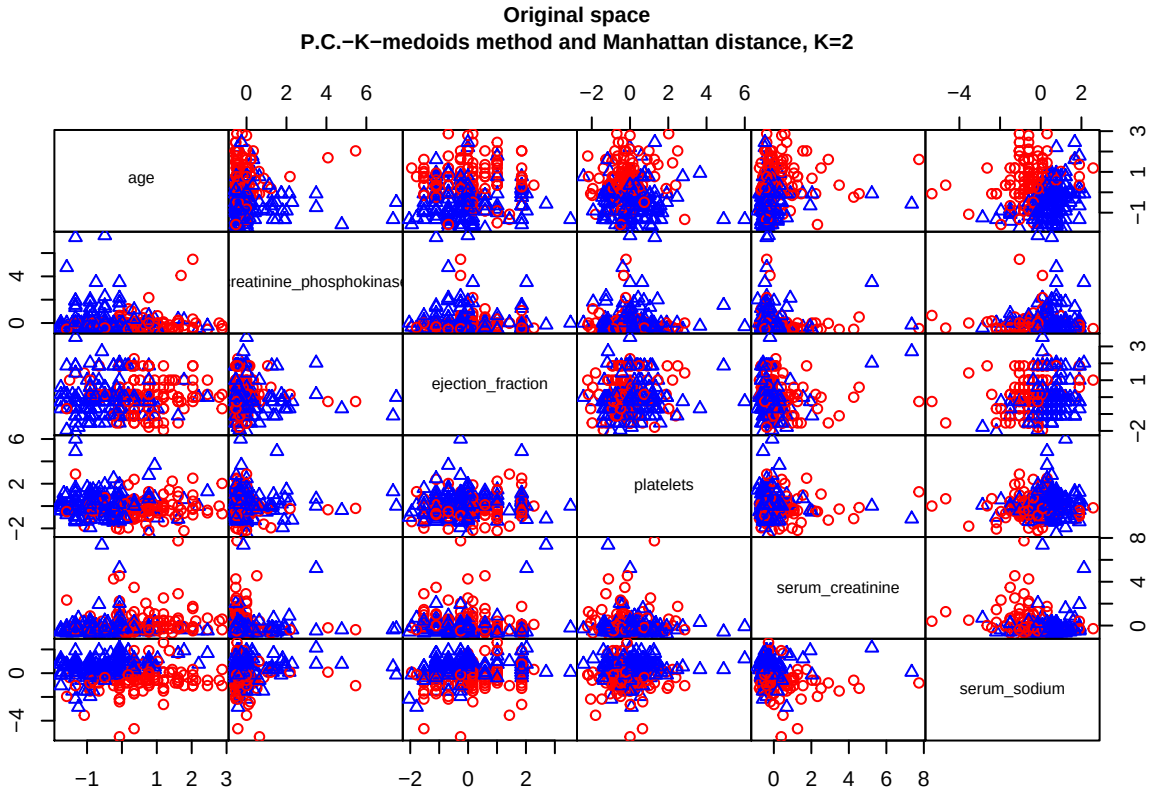
```
## Medoids:
##       ID        age creatinine_phosphokinase ejection_fraction   platelets
## [1,] 161  0.4343161               -0.5254517         0.1619277 -0.21837530
## [2,]  63 -0.4904571               -0.4873187        -0.2605537 -0.09568123
##      serum_creatinine serum_sodium
## [1,]       -0.1874120   -0.5949987
## [2,]       -0.2840761    0.5381517
## Clustering vector:
##   [1] 1 2 1 2 1 1 1 1 1 1 1 2 2 2 2 1 2 2 2 1 1 1 2 2 1 1 1 1 1 1 1 1 2 2 1 1 1
##  [38] 2 2 2 1 2 1 1 2 2 2 2 1 2 2 2 2 1 1 1 1 2 2 1 2 1 2 2 2 1 2 2 2 1 2 2 1 1
##  [75] 1 2 2 2 1 2 1 1 1 1 2 2 2 2 2 2 1 1 2 1 2 2 1 1 2 2 2 1 2 2 2 2 2 2 1 2 1
## [112] 2 2 1 1 2 1 1 1 1 1 1 2 2 1 1 2 2 2 2 2 1 2 2 1 1 1 1 1 2 1 2 2 2 2 2 2 1
## [149] 1 2 1 2 1 2 1 2 2 2 1 2 1 2 1 2 2 1 1 1 1 1 2 2 1 2 2 2 1 2 2 2 2 2 1 1 2
## [186] 2 1 2 1 2 1 1 1 2 2 1 2 2 2 1 1 2 1 2 1 2 2 1 2 2 1 2 1 2 1 1 2 2 1 2 1 1
## [223] 2 2 2 1 1 2 1 1 1 1 2 2 2 1 1 1 1 2 2 1 2 1 1 2 2 1 2 1 2 2 1 1 2 2 2 2 2
## [260] 2 2 1 1 1 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 1 1 1 2 2 1 2 1 2 2 2 2 2 1 2 2
## [297] 2 2 2
## Objective function:
```

```
##    build     swap
## 3.667061 3.641016
##
## Available components:
##  [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"
##  [6] "clusinfo"   "silinfo"    "diss"       "call"       "data"
```

```
pam.res$clusinfo
```

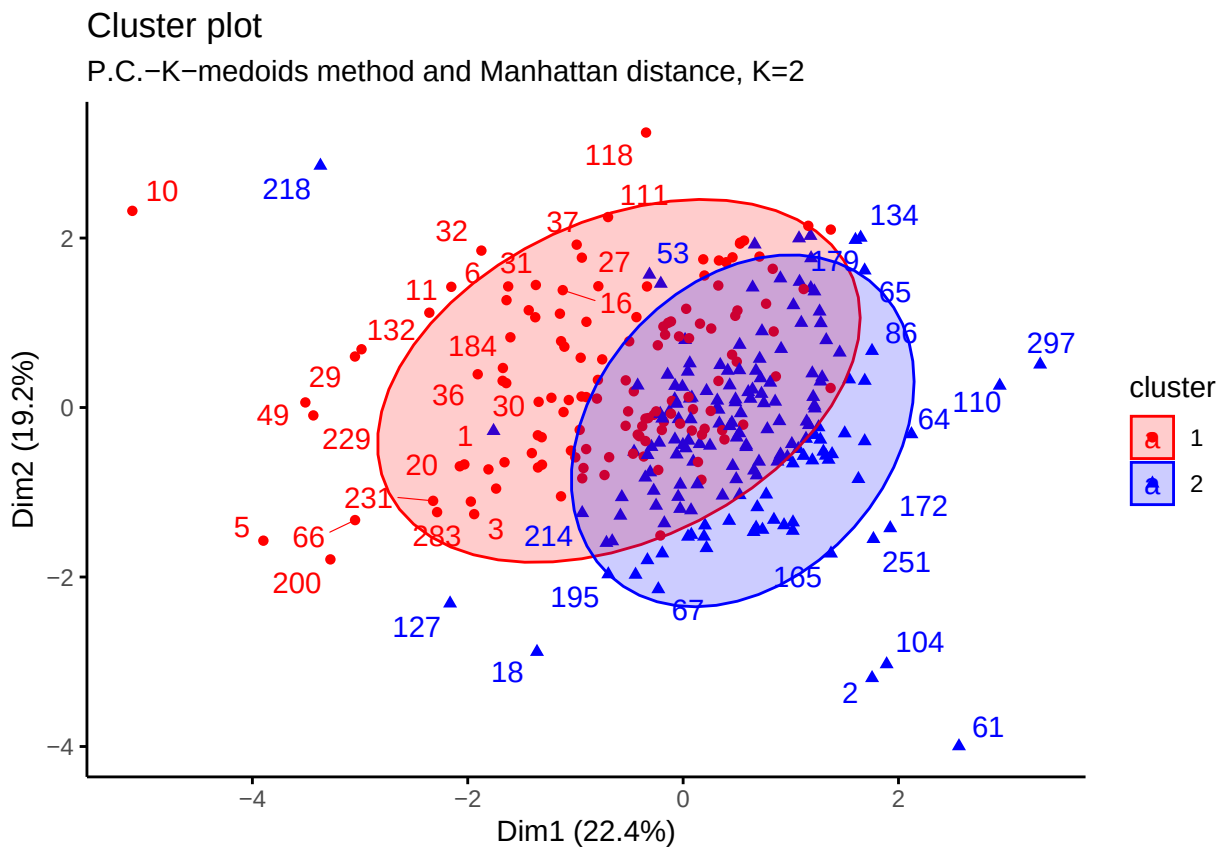
```
##      size max_diss  av_diss diameter separation
## [1,]  132 11.29789 3.739802 17.90018  0.8951301
## [2,]  167 13.86674 3.562933 22.68679  0.8951301
```

```
cm <- pam.res$cluster
pairs(heart.attk_scale, gap=0, main="Original space\nP.C.-K-medoids method and Manhattan distance, K=2"
```




```
fviz_cluster(pam.res, palette = c("red", "blue"), ellipse.type = "t",
repel = TRUE, ggtheme = theme_classic())+labs(
subtitle = "P.C.-K-medoids method and Manhattan distance, K=2", cex.sub= 0.5)
```

```
## Warning: ggrepel: 254 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



Applying partitioning clustering method and using PAM algorithm and Manhattan distance, two clusters are composed in this way: cluster 1 with 132 units, cluster 2 with 167 units. In order to examine the goodness of clustering algorithm results, we have to analyst internal and external validation measures.

Internal validation measures:

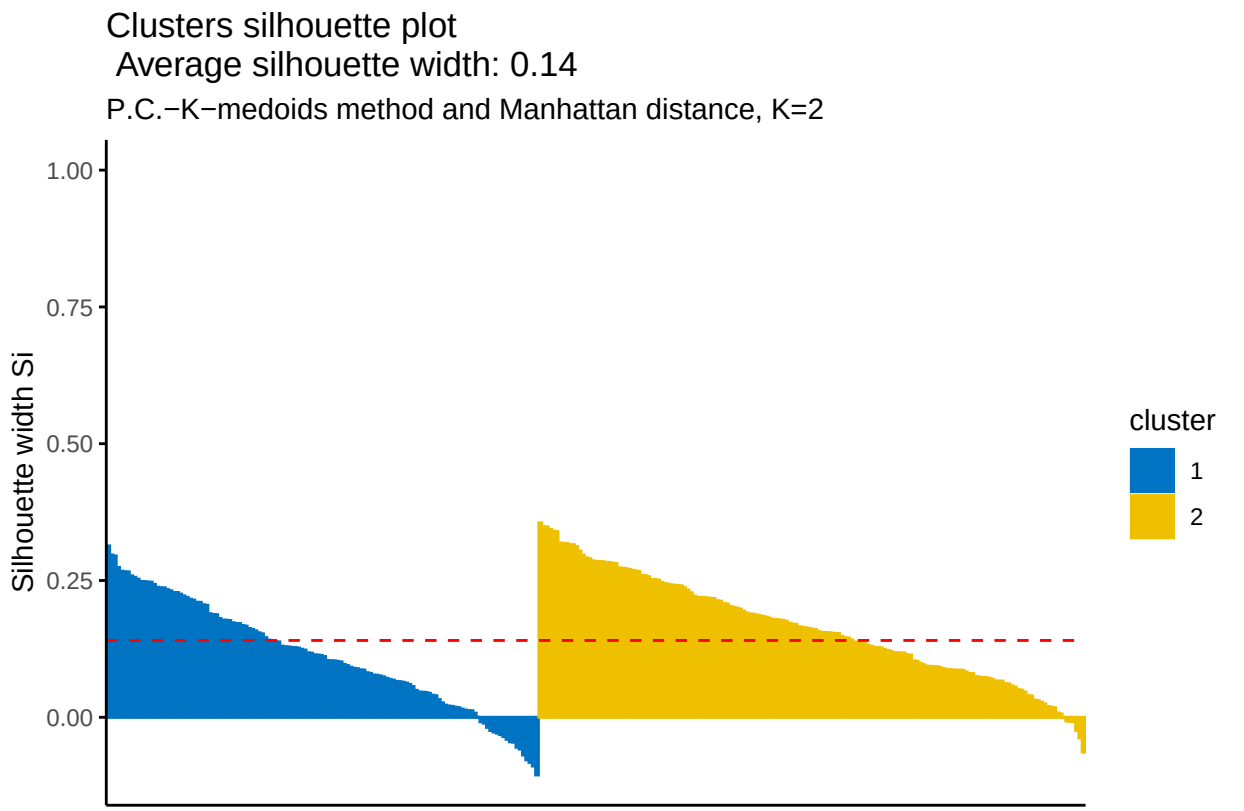
Silhouette width:

```
hclust<- eclust(heart.attk.sub,k=2 ,"pam", graph = FALSE, hc_metric = "manhattan")
silinfo <- hclust$silinfo
silinfo$avg.width
```

```
## [1] 0.4630251
```

```
fviz_silhouette(pam.res, palette = "jco",
ggtheme = theme_classic())+labs(
subtitle = "P.C.-K-medoids method and Manhattan distance, K=2", cex.sub= 0.5)
```

```
##   cluster size ave.sil.width
## 1       1  132          0.11
## 2       2  167          0.16
```




```
silinfo$clus.avg.widths
```

```
## [1] 0.3572776 0.5914886
```

```
sil <- hclust$silinfo$widths[, 1:3]
neg_sil_index_aver.eu<- which(sil[, "sil_width"] < 0)
sil[neg_sil_index_aver.eu, , drop = FALSE]
```

```
##     cluster neighbor    sil_width
## 239       1        2 -0.008213382
## 233       1        2 -0.062316672
## 55        1        2 -0.062318290
## 243       1        2 -0.062324834
## 98        1        2 -0.062337457
## 60        1        2 -0.088962534
## 63        1        2 -0.088987015
## 97        1        2 -0.088988538
## 12        1        2 -0.115348613
## 258       1        2 -0.115352555
## 113       1        2 -0.141336720
```

The value of complete silhouette width indicates that on average the units are well enough clustered. As in particular, in each cluster the units are on average the same silhouette value with respect to the silhouette width. According to this index, 11 units that belong to cluster 1 are not well clustered: they should belong to cluster 2.

2.Dunn index:

```
stats <- cluster.stats(dist(heart.attk_scale), hclust$cluster)
stats$dunn
```

```
## [1] 0.0333656
```

According to the Dunn index, the units are not clustered well enough.

External validation measures:

Confusion matrix:

According to the Confusion matrix, the number of clusters is equal to nominal values. The clusters found are 2 and the nominal variable can take 2 possible values.

```
table(heart.attk$smoking, hclust$cluster)
```

```
##
##       1   2
##   0 109  94
##   1  55  41
```

Most of the smokers (n = 109) has been classified in cluster 2 while cluster 1 have 94 number of values. And for non-smokers (n=55) has been classified in cluster 1 while cluster 2 has 41 values.

2.Correct Rand Index:

```
smoking <- as.numeric(heart.attk$smoking)
stats<- cluster.stats(d = dist(heart.attk_scale), smoking, hclust$cluster)
stats$corrected.rand
```

```
## [1] -0.00410881
```

According to the Correct Rand Index, there is no agreement between the numerical values and the cluster solution. From -1 to +1, the agreement is very close to 0.

Meila's VI Index:

```
stats$vi
```

```
## [1] 1.314969
```

Soft clustering approach:

Model based-clustering:

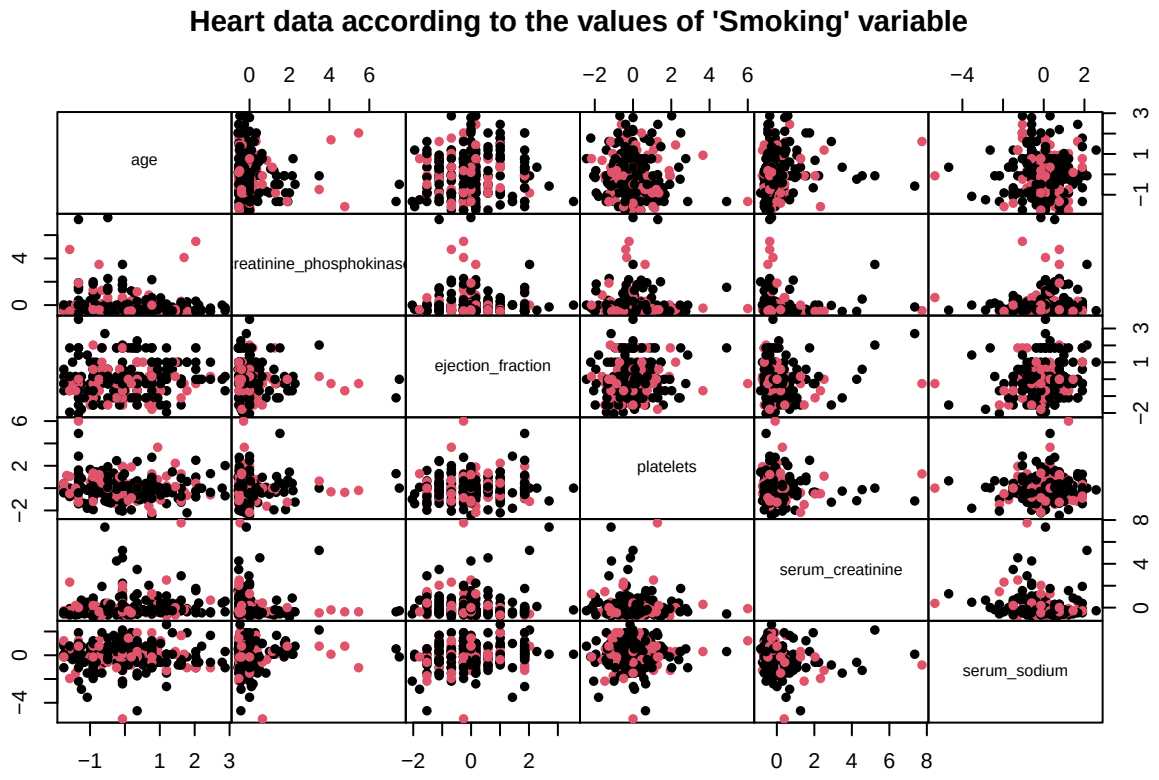
```
summary(heart.attk$smoking)
```

```
##   0   1
## 203  96
```

```
head(heart.attk)
```

```
##   age anaemia creatinine_phosphokinase diabetes ejection_fraction
## 1  75       0                     582        0                20
## 2  55       0                    7861        0                38
## 3  65       0                     146        0                20
## 4  50       1                     111        0                20
## 5  65       1                     160        1                20
## 6  90       1                      47        0                40
##   high_blood_pressure platelets serum_creatinine serum_sodium sex smoking
## 1                   1    265000             1.9          130   1       0
## 2                   0    263358             1.1          136   1       0
## 3                   0    162000             1.3          129   1       1
## 4                   0    210000             1.9          137   1       0
## 5                   0    327000             2.7          116   0       0
## 6                   1    204000             2.1          132   1       1
##   DEATH_EVENT
## 1           1
## 2           1
## 3           1
## 4           1
## 5           1
## 6           1
```

```
X <- data.matrix(heart.attk.sub)
sX <- scale(X)
pairs(sX, gap=0, pch = 16, col = as.numeric(heart.attk$smoking), cex.main = 0.9 ,main="Heart data accor
```



To examine whether there is a relation between the categorical variable Smoking and the underlying clustering, the variable is deleted and the data are standardized. The data are visualized by pairwise scatterplots, in which the colors represent the two possible values of smoking: "0", "1". It seems difficult to distinguish separate groups. Different Parsimonious Gaussian mixtures are fitted on the standardized data by using the function Mclust() in R.

```
library(mclust)
```

```
## Warning: package 'mclust' was built under R version 4.0.5
```

```
## Package 'mclust' version 5.4.7
## Type 'citation("mclust")' for citing this R package in publications.
```

```
##
## Attaching package: 'mclust'
```

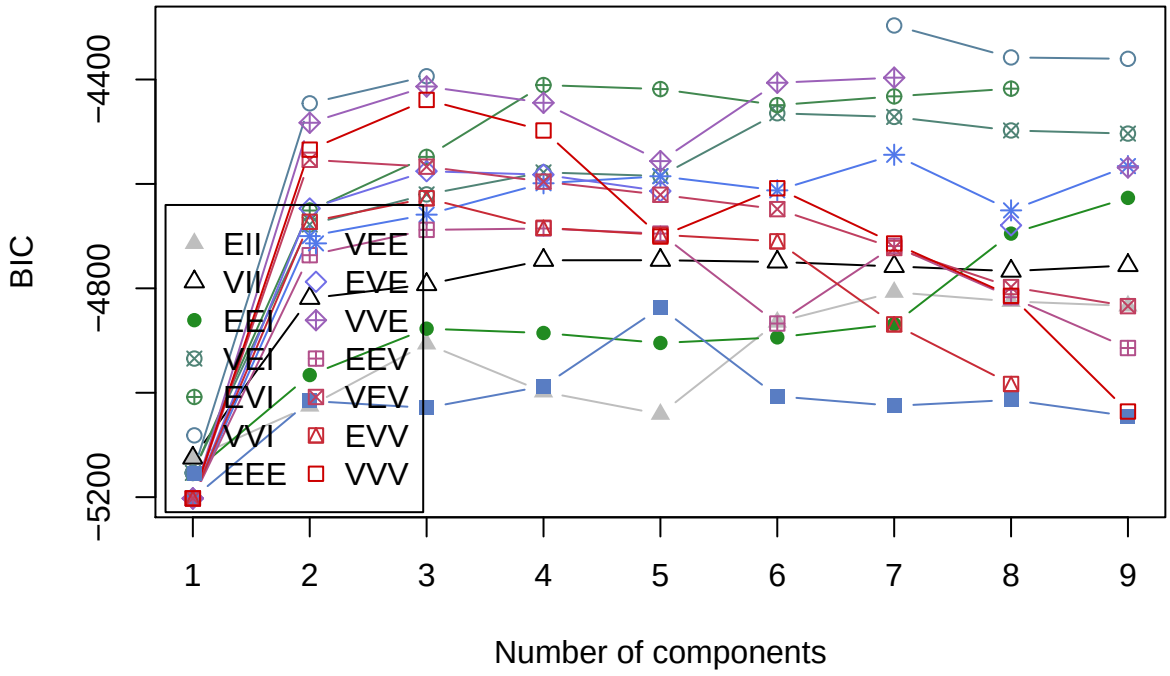
```
## The following object is masked from 'package:psych':
##
##     sim
```

```
## The following object is masked from 'package:gamlss.data':
##
##     acidity
```

```
mod <- Mclust(heart.attk_scale)
summary(mod$BIC)
```

```
## Best BIC values:
##              VVI,7       VVI,8       VVI,9
## BIC      -4296.508 -4357.70740 -4360.37800
## BIC diff     0.000   -61.19954   -63.87013
```

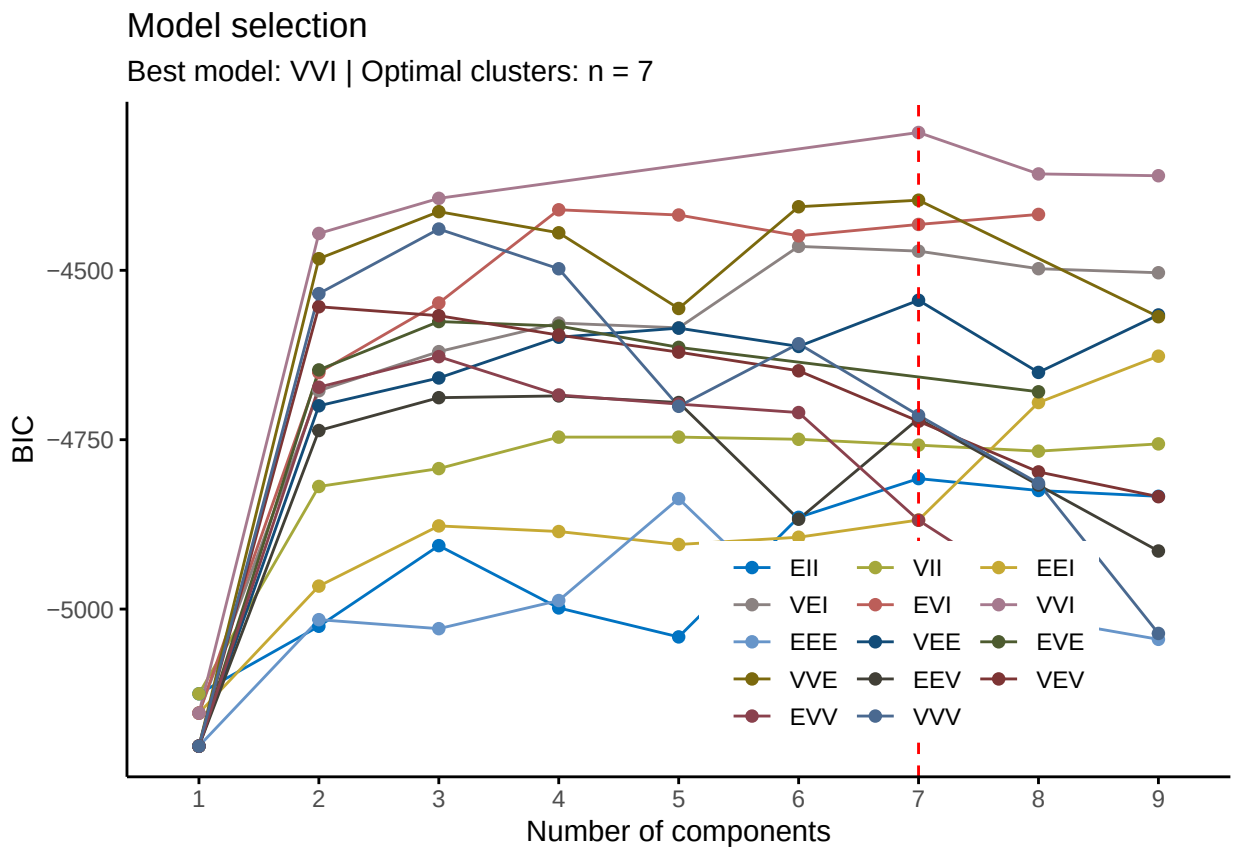
```
plot(mod, what = "BIC", ylim = range(mod$BIC, na.rm = TRUE),
legendArgs = list(x = "bottomleft"))
```




```
summary(mod)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VVI (diagonal, varying volume and shape) model with 7 components:
##
##  log-likelihood   n df       BIC     ICL
##       -1891.734 299 90 -4296.508 -4403.1
##
## Clustering table:
##   1   2   3   4   5   6   7
##  53   9  43 107  27  28  32
```

```
fviz_mclust(mod, "BIC", palette = "jco")
```




```
head(round(mod$z, 6), 15)
```

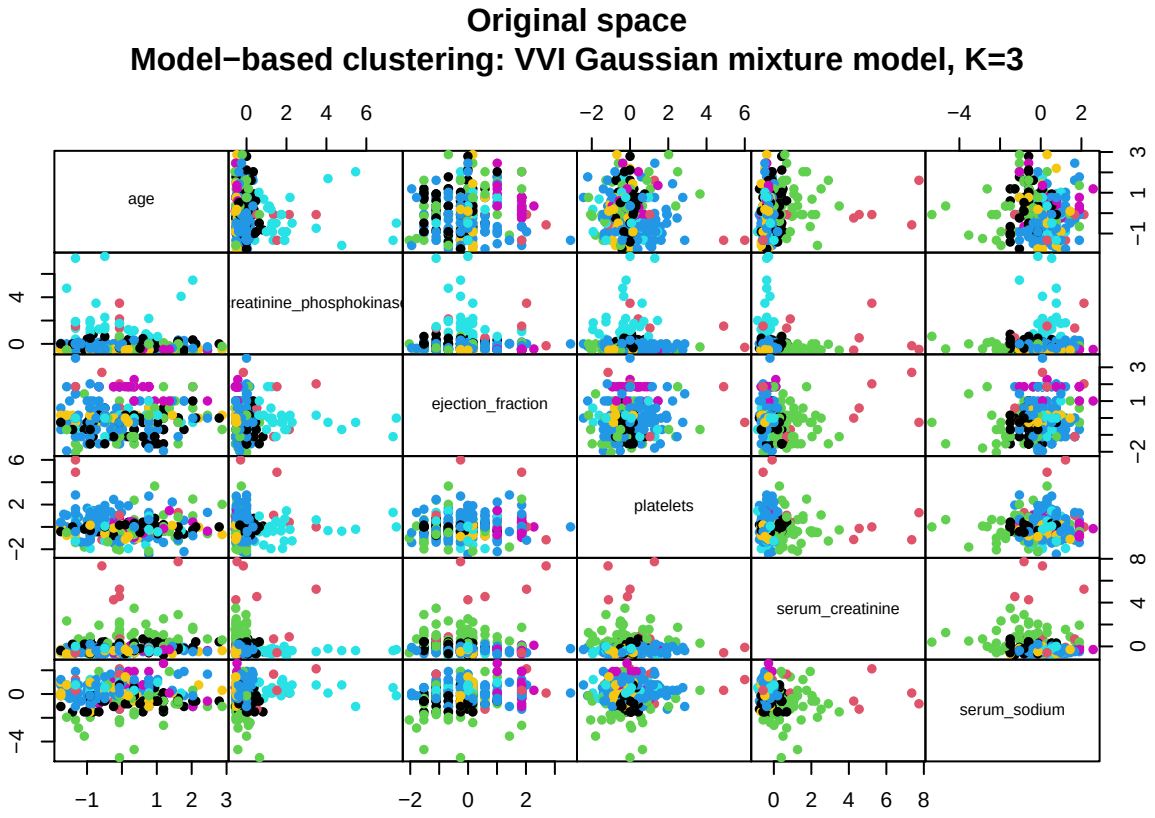
```
##           [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]
##  [1,] 0.767016 0.000122 0.232849 0.000014 0.000000 0.000000 0.000000
##  [2,] 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000
##  [3,] 0.045288 0.000391 0.862831 0.091472 0.000018 0.000000 0.000000
##  [4,] 0.579609 0.001824 0.416512 0.002055 0.000000 0.000000 0.000000
##  [5,] 0.000000 0.000001 0.999999 0.000000 0.000000 0.000000 0.000000
##  [6,] 0.210629 0.000118 0.789253 0.000000 0.000000 0.000000 0.000000
##  [7,] 0.001439 0.000258 0.165968 0.826050 0.006284 0.000000 0.000000
##  [8,] 0.000000 0.007431 0.065124 0.927362 0.000056 0.000027 0.000000
##  [9,] 0.001175 0.000377 0.004029 0.032950 0.000015 0.961454 0.000000
## [10,] 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000
## [11,] 0.000000 0.007126 0.992874 0.000000 0.000000 0.000000 0.000000
## [12,] 0.052498 0.000102 0.015440 0.902693 0.029268 0.000000 0.000000
## [13,] 0.005934 0.002039 0.005240 0.285175 0.701612 0.000000 0.000000
## [14,] 0.086868 0.000058 0.004991 0.483764 0.021643 0.000030 0.402646
## [15,] 0.000002 0.000381 0.008460 0.975432 0.015719 0.000000 0.000006
```

```
head(mod$classification, 15)
```

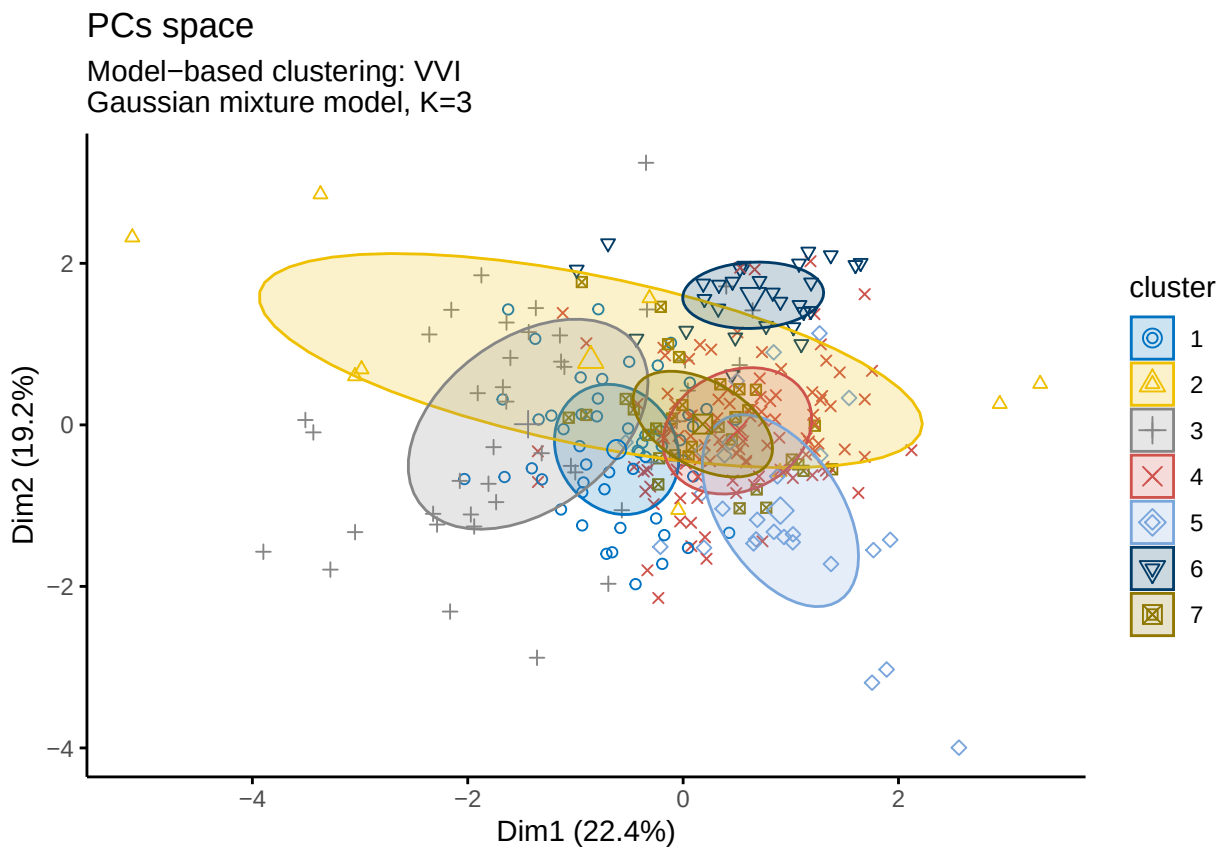
```
##  [1] 1 5 3 1 3 3 4 4 6 2 3 4 5 4 4
```

According to the penalized selection criterion called “BIC” (Bayesian Information Criterion), the three best Gaussian mixture models are: VVI with 7 clusters, VVI with 8 clusters and VVI with 9 clusters.

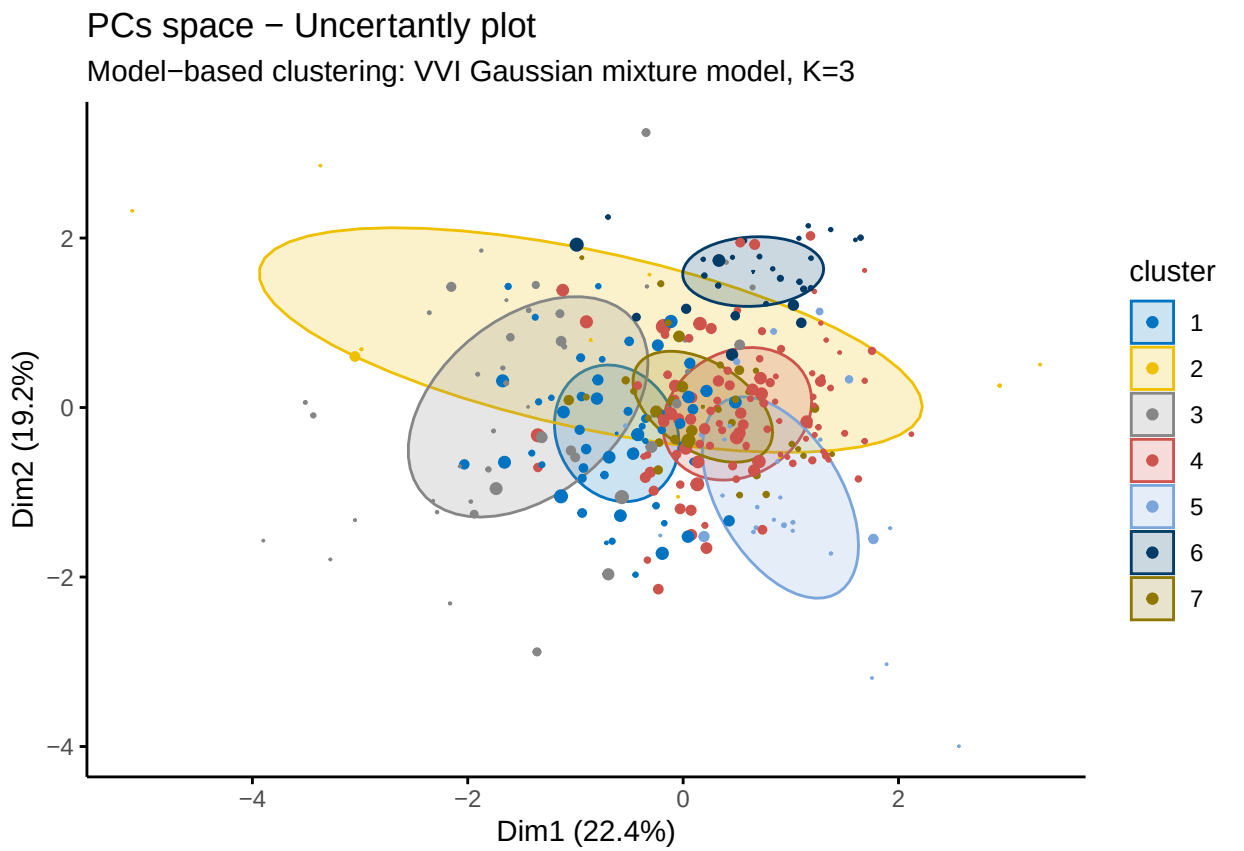
```
pairs(sX, gap=0, pch = 16, col = mod$classification, cex.main = 1,main="Original space\nModel-based clus
```




```
fviz_mclust(mod, "classification", geom = "point", pointsize = 1.5, palette = "jco", main = "PCs space"
Gaussian mixture model, K=3")
```




```
fviz_mclust(mod, "uncertainty", palette = "jco",
main = "PCs space - Uncertantly plot")+
labs(subtitle= "Model-based clustering: VVI Gaussian mixture model, K=3")
```



Both in the Original space and in the pcs space there is much less separation between cluster. Moreover, from the Uncertainty plot, it is noted that some units (big points) are problematic for the soft approach because they belong to different clusters with the same (or similar) probability.

External validation measures:

Confusion matrix:

```
table(heart.attk$smoking, mod$classification)
```

```
##
##       1  2  3  4  5  6  7
##   0 38  7 32 65 18 21 22
##   1 15  2 11 42  9  7 10
```

According to the Confusion matrix, A large number of "female" sex (n = 45) has been classified in cluster 2.Most of the smokers (n = 55) has been classified in cluster 4 while cluster 2 have 7 , cluster 1 have 38, cluster 3 have 32, cluster 5 have 18, cluster 6 have 21 and cluster 7 have 22 number of values. For non-smokers (n=42) classified in cluster 1 while in other cluster values are smalls.

2.Correct Rand Index:

```
adjustedRandIndex(heart.attk$smoking, mod$classification)
```

```
## [1] -0.01379716
```

According the Correct Rand Index, there is a better agreement between the nominal variable and the cluster solution.

The best clustering algorithm:

In order to choose the best clustering algorithm among those proposed, the clValid package of R. is used.The clValid function enables to compare clustering algorithms using two cluster validation measures: internal measures (Connectivity, Silhouette coefficient, Dunn index). The methods considered are: hierarchical method (Euclidean- Manhattan), k-means method, pam algorithm (Euclidean-Manhattan) and model-based clustering.

```
library(clValid)
```

```
## Warning: package 'clValid' was built under R version 4.0.5
```

```
clmethods <- c ("hierarchical", "kmeans", "pam")
V_eucl<-clValid(heart.attk_scale, nClust=2:6, clMethods= clmethods, metric="euclidean", validation="int
```

```
## Warning in clValid(heart.attk_scale, nClust = 2:6, clMethods = clmethods, :
## rownames for data not specified, using 1:nrow(data)
```

```
summary(V_eucl)
```

```
##
## Clustering Methods:
##  hierarchical kmeans pam
##
## Cluster sizes:
##  2 3 4 5 6
##
## Validation Measures:
##                                  2        3        4        5        6
##
## hierarchical Connectivity    7.2036  11.5615  19.5242  20.2742  26.4306
##              Dunn            0.3998   0.3016   0.2311   0.2311   0.2311
##              Silhouette      0.6176   0.5162   0.5023   0.4990   0.3938
## kmeans       Connectivity   80.5179 116.5063 124.5841 140.9147 189.6635
##              Dunn            0.0498   0.0495   0.0649   0.0541   0.0782
##              Silhouette      0.1967   0.1523   0.1646   0.1847   0.1646
## pam          Connectivity  121.3349 184.2381 173.6313 188.2964 190.3841
##              Dunn            0.0281   0.0300   0.0366   0.0386   0.0470
##              Silhouette      0.1432   0.1228   0.1319   0.1275   0.1371
```

```
##
## Optimal Scores:
##
##              Score  Method       Clusters
## Connectivity 7.2036 hierarchical 2
## Dunn         0.3998 hierarchical 2
## Silhouette   0.6176 hierarchical 2
```

According to the result, the best clustering algorithm using the Euclidean distance is the hierarchical clustering method with 2 clusters.

```
clmethods <- c ("hierarchical", "kmeans", "pam")
V_eucl<-clValid(heart.attk_scale, nClust=2:6, clMethods= clmethods, metric="manhattan", validation="int
```

```
## Warning in clValid(heart.attk_scale, nClust = 2:6, clMethods = clmethods, :
## rownames for data not specified, using 1:nrow(data)
```

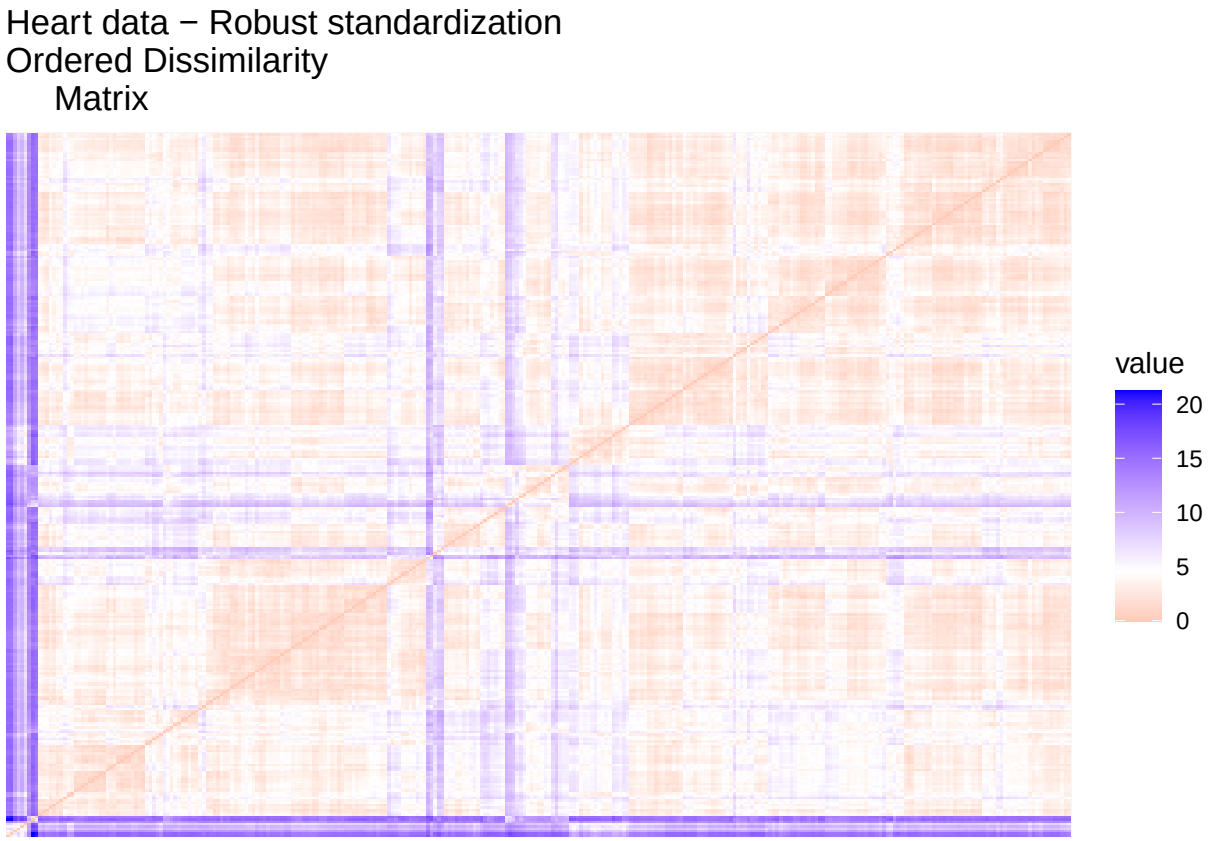
```
summary(V_eucl)
```

```
##
## Clustering Methods:
##  hierarchical kmeans pam
##
## Cluster sizes:
##  2 3 4 5 6
##
## Validation Measures:
##                                  2        3        4        5        6
##
## hierarchical Connectivity    7.8663   8.3425  13.8075  20.7567  21.2012
##              Dunn            0.3537   0.3537   0.3030   0.2090   0.2090
##              Silhouette      0.5836   0.5545   0.4566   0.4359   0.4276
## kmeans       Connectivity   87.8810  94.2548 145.0984 160.6020 188.8909
##              Dunn            0.0454   0.0527   0.0425   0.0288   0.0708
##              Silhouette      0.2049   0.2128   0.1483   0.1714   0.1653
## pam          Connectivity  149.3651 174.0063 245.9373 257.4230 275.9595
##              Dunn            0.0395   0.0441   0.0511   0.0413   0.0380
##              Silhouette      0.1405   0.1479   0.1128   0.1276   0.1149
##
## Optimal Scores:
##
##              Score  Method       Clusters
## Connectivity 7.8663 hierarchical 2
## Dunn         0.3537 hierarchical 2
## Silhouette   0.5836 hierarchical 2
```

In this measure according to the APN and ADM the best method is hierarchical with 2 clusters and according to AD and FOM the best method is pam with 6 number of clusters.

```
scale_rob <- scale(heart.attk.sub, center = apply(heart.attk.sub, 2, median), scale = apply(heart.attk.s
rownames(scale_rob) <- rownames(heart.attk)
fviz_dist(dist(scale_rob), show_labels = FALSE)+
```

```
labs(title = "Heart data - Robust standardization\nOrdered Dissimilarity
     Matrix", gradient = list(low = "red", mid = "white", high =
"blue"))
```




```
hopkins(scale_rob, n = nrow(scale_rob)-1)
```

```
## $H
## [1] 0.1737538
```

The data set is uniformly distributed, according to the Hopkins statistic (H), because the values are close to 0. A clusters structure is also present in the dissimilarity matrix image of the data.

Clustering result:

According to the proposed indices, the hierarchical method with two clusters, computed using the Euclidean distance, appears to be the most appropriate of the clustering algorithms used.