Homework - Multivariate Analysis

Javier Esteban Aragoneses - Mauricio Marcos Fajgenbaun

12/12/2020

An analysis of Diabetes in Indian Women

Motivation of Research and Dataset

For this research, we selected a data set of the Diabetes. Diabetes is an illness that occurs when the sugar in blood (also called glucose) is too high. At the same time, insulin (produced by the pancreas), is produced to help the glucose make it to human cells. When the pancreas does not produce enough insulin or none at all, then this glucose remains in the blood causing serious health issues.

With the years, diabetes had become a larger problem, especially for western societies. Eating fast food or high-processed meals have led to enormous ingestions of sugar and, with this, an historical increase of diabetes in most countries of the world.

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The main objective that explains the reason of the existence of this data set, is to diagnostically predict weather or not a patient has diabetes, based on certain measurements of several variables that are included in the dataset. At the same time, this database contains only data of women of at least 21 years old of Pima Indian heritage (north American indigenous women, that live in the State of Arizona and the Mexican states of Sonora and Chihuahua). Diabetes is quite prevalent in this group of Native Americans living in those places in particular.

So, we can say that the aim is to study the cause of diabetes in this particular ethnic group. This disease was very uncommon in Pima Indians until the second half of the 20th Century. Nevertheless, as King et al. informs in 1993, the highest prevalence of type 2 diabetes in the world was found in this particular ethnic group, as more than half of the women older than 35 years-old would suffer from this disease.

So, in this research we will study how health predictors are associated with the presence of diabetes in Pima Indians. According to the World Health Organization criteria, if the 2-hour post-load plasma glucose was at least 200 mg/dl at any survey examination or if found during any moutine medical care.

In this dataset, 768 women were registered in the database. From this total, a 35% (268 in total) had diabetes, while the rest (500: 65%) did not suffer from this disease. In the next table, we show the variables included in the data set with they specific classification.

Variables Description

Pregnancy: quantitative - discrete Glucose: quantitative - continuous Blood Pressure: quantitative - continuous Skin Thickness: quantitative - continuous Insulin quantitative - continuous BMI: quantitative - continuous Diabetes Pedigree Function: quantitative - continuous Age: quantitative - discrete Outcome: qualitative- nominal

The variable “Outcome” is a categorical variable that accounts for weather a specific person is diabetic (takes value = 1) or not diabetic (takes value = 0).

Exploratory Analysis

First, let´s install all the packages we will use. Then, we will read the file and start working on it.

```
library("tidyverse")
```

```
## -- Attaching packages --------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## Warning: package 'stringr' was built under R version 4.0.3

## -- Conflicts ------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library("ggplot2")
library("dplyr")
library("moments")
```

```
## Warning: package 'moments' was built under R version 4.0.3
```

```
library("gridExtra")
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library("rrcov")
```

```
## Loading required package: robustbase

## Scalable Robust Estimators with High Breakdown Point (version 1.5-5)
```

```
library("rpart.plot")
```

```
## Warning: package 'rpart.plot' was built under R version 4.0.3

## Loading required package: rpart
```

```r
library("mice")
```

```
## Warning: package 'mice' was built under R version 4.0.3
```

```
##
## Attaching package: 'mice'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```r
library("MASS")
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```r
library("andrews")
```

```
## Warning: package 'andrews' was built under R version 4.0.3
```

```r
library("ggcorrplot")
```

```
## Warning: package 'ggcorrplot' was built under R version 4.0.3
```

```r
library("FactoMineR")
library("factoextra")
```

```
## Warning: package 'factoextra' was built under R version 4.0.3
```

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

```r
library("paran")
```

```
## Warning: package 'paran' was built under R version 4.0.3
```

```r
library("corrplot")
```

```
## Warning: package 'corrplot' was built under R version 4.0.3
```

```
## corrplot 0.84 loaded
```

```r
library("corpcor")
```

```
## Warning: package 'corpcor' was built under R version 4.0.3
```

```r
library("RSpectra")
```

```
## Warning: package 'RSpectra' was built under R version 4.0.3
```

```r
library("factoextra")
library("cluster")
library("ape")
```

```
## Warning: package 'ape' was built under R version 4.0.3
```

```r
library(mclust)
```

```
## Package 'mclust' version 5.4.6
## Type 'citation("mclust")' for citing this R package in publications.

##
## Attaching package: 'mclust'

## The following object is masked from 'package:purrr':
##
##     map
```

```r
ColClasses=c(rep("numeric",9))
DATOS=read.csv2("diabetes.csv",sep = ",",header = T, colClasses = ColClasses, dec = "." )
summary(DATOS)
```

```
##   Pregnancies        Glucose      BloodPressure    SkinThickness
##  Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00
##  1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00
##  Median : 3.000   Median :117.0   Median : 72.00   Median :23.00
##  Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54
##  3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00
##  Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
##     Insulin           BMI        DiabetesPedigreeFunction      Age
##  Min.   :  0.0   Min.   : 0.00   Min.   :0.0780           Min.   :21.00
##  1st Qu.:  0.0   1st Qu.:27.30   1st Qu.:0.2437           1st Qu.:24.00
##  Median : 30.5   Median :32.00   Median :0.3725           Median :29.00
##  Mean   : 79.8   Mean   :31.99   Mean   :0.4719           Mean   :33.24
##  3rd Qu.:127.2   3rd Qu.:36.60   3rd Qu.:0.6262           3rd Qu.:41.00
##  Max.   :846.0   Max.   :67.10   Max.   :2.4200           Max.   :81.00
##     Outcome
##  Min.   :0.000
##  1st Qu.:0.000
##  Median :0.000
##  Mean   :0.349
##  3rd Qu.:1.000
##  Max.   :1.000
```

When studying the variables, we can tell that there are some missing values in some of the variables. This will be a big burden later, when trying to study correlation between the variables and the real dimension of it. Also, it will blur our understanding of the data. Thus, we will impute the missing values first.

To do so, we change the values "0" from the second to the eight variable to "NA". We only make use of these 7 variables, because the first variable (pregnancy) can take the value "0" and our variable number 9 is "outcome" and its value is either "1" or "0". Of course, we wouldn´t want to change their zero values. Nevertheless, we know that as we are talking about humans (that are actually alive) it is impossible to get an observation of these variables equal to zero. Easy example: it is obvious nobody has a blood preassure equal to zero. This is why we consider values "0" for these variables as missing values.

We will impute the values with the library "MICE".

```
data <- DATOS[2:8]
data[data==0] <-  NA
DATOS[2:8] <- data
summary(DATOS[,1:8])
```

```
##   Pregnancies        Glucose      BloodPressure    SkinThickness
##  Min.   : 0.000   Min.   : 44.0   Min.   : 24.00   Min.   : 7.00
##  1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 64.00   1st Qu.:22.00
##  Median : 3.000   Median :117.0   Median : 72.00   Median :29.00
##  Mean   : 3.845   Mean   :121.7   Mean   : 72.41   Mean   :29.15
##  3rd Qu.: 6.000   3rd Qu.:141.0   3rd Qu.: 80.00   3rd Qu.:36.00
##  Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
##                   NA's   :5       NA's   :35       NA's   :227
##     Insulin           BMI        DiabetesPedigreeFunction      Age
##  Min.   : 14.00   Min.   :18.20   Min.   :0.0780           Min.   :21.00
##  1st Qu.: 76.25   1st Qu.:27.50   1st Qu.:0.2437           1st Qu.:24.00
##  Median :125.00   Median :32.30   Median :0.3725           Median :29.00
##  Mean   :155.55   Mean   :32.46   Mean   :0.4719           Mean   :33.24
##  3rd Qu.:190.00   3rd Qu.:36.60   3rd Qu.:0.6262           3rd Qu.:41.00
##  Max.   :846.00   Max.   :67.10   Max.   :2.4200           Max.   :81.00
##  NA's   :374      NA's   :11
```

```
sum(is.na(DATOS))
```

```
## [1] 652
```

This is a big problem. We can see that we have a lot of missing value in our data (652 values are missing). When checking more closely, we can say that at least 30% of our rows are being affected by this problem. We certainly can not give up 30% of our data, as we would lose too much information. We have to impute this values using the package "Mice".

```
DATOS <- mice(DATOS,m=1,method="pmm")
```

```
##
##  iter imp variable
##   1   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   2   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   3   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   4   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   5   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
```

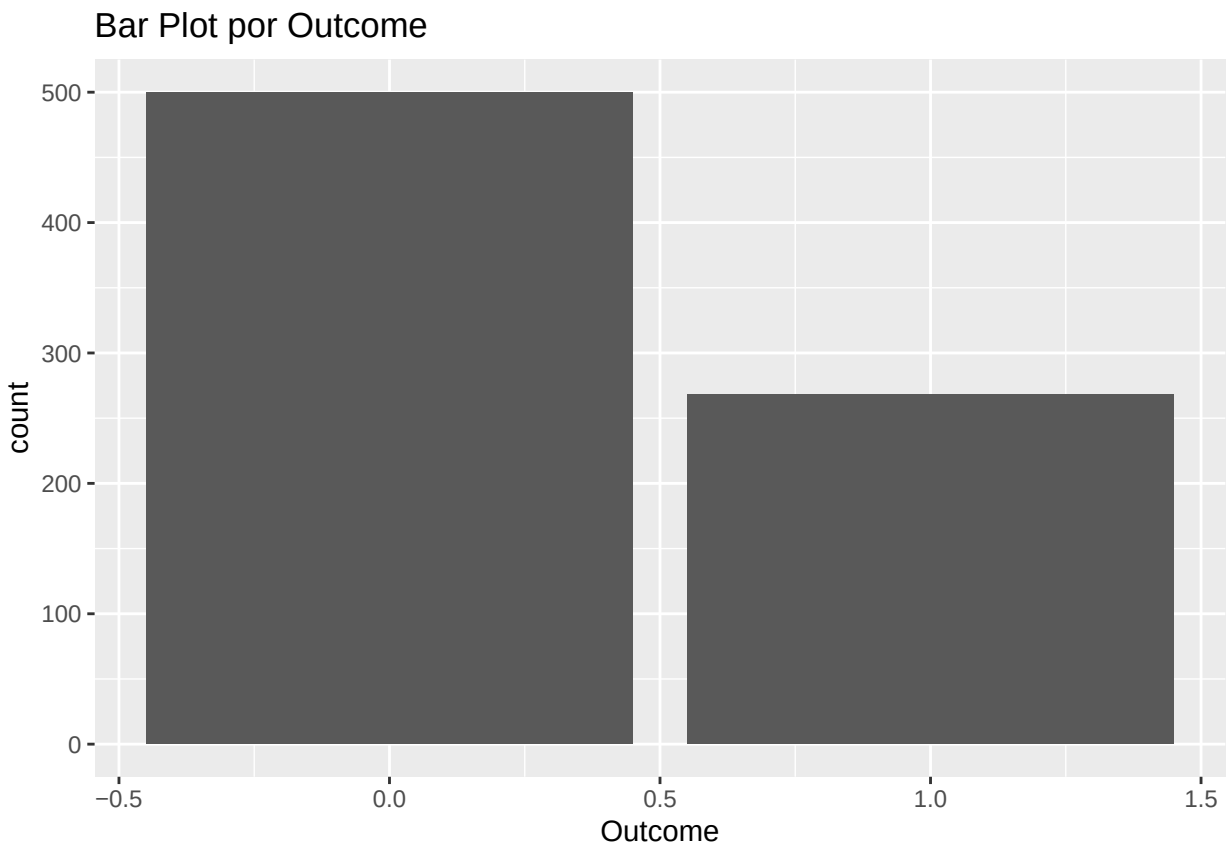
```r
DATOS <- complete(DATOS)
attach(DATOS)
```

Now that we solved the problem of missing values, we will create a new variable called "Dataset.norm" to use later on, when we will need to standarize our variables for principal component analysis.

```r
Dataset.norm <- DATOS
# Unitization with zero minimum ((x-min)/range))
Dataset.norm[1:8] <- as.data.frame(lapply(DATOS[1:8], normalize))
```

Let´s first dig in into our categorical variable.

```r
ggplot(data=DATOS, aes(x = Outcome)) + geom_bar(aes(fill = Outcome)) + ggtitle("Bar Plot por Outcome")
```




```r
frec_cal <- table(DATOS$Outcome)
frec_cal
```

```
##
##   0   1
## 500 268
```

```r
tabl_cont <- prop.table(table(DATOS$Outcome))
tabl_cont
```

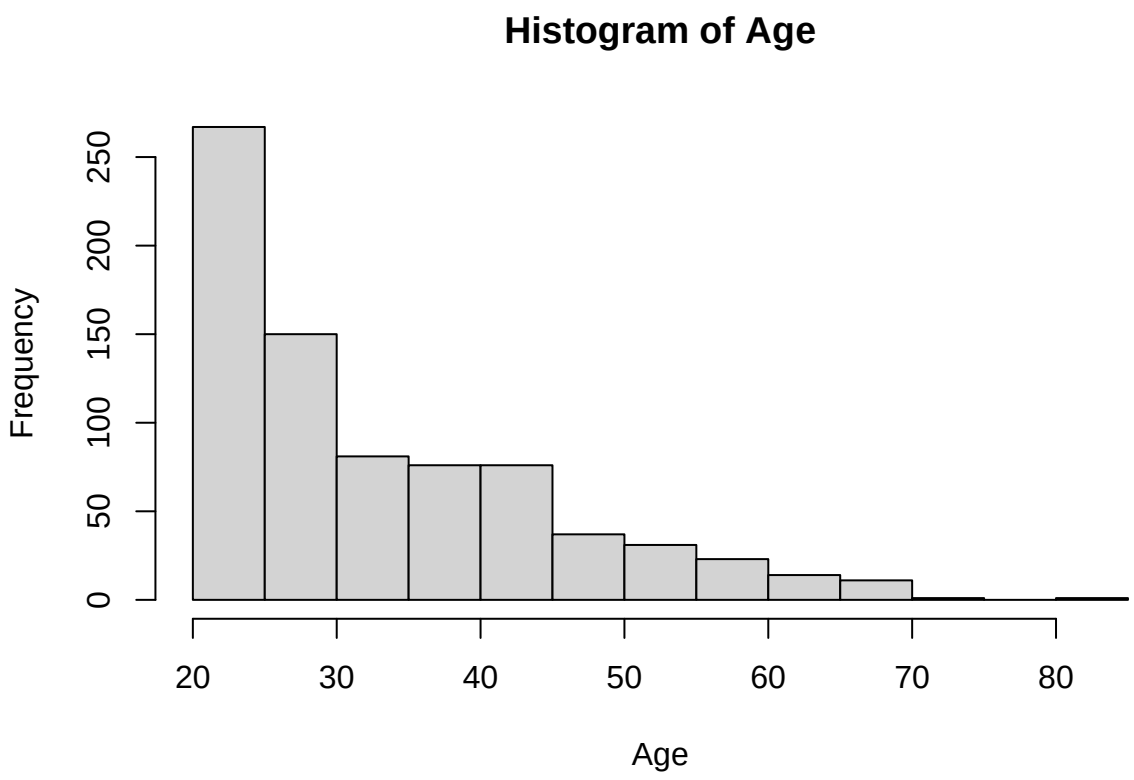
```
##
##         0         1
## 0.6510417 0.3489583
```

As we can see, 65% of our sample does not suffer from diabetes, while 35% does suffer from diabetes. This is a huge proportion, given that this sample is taken from one population in particular (it is certainly a population with a huge incidence of diabetes). One every three persons in this sample has diabetes.

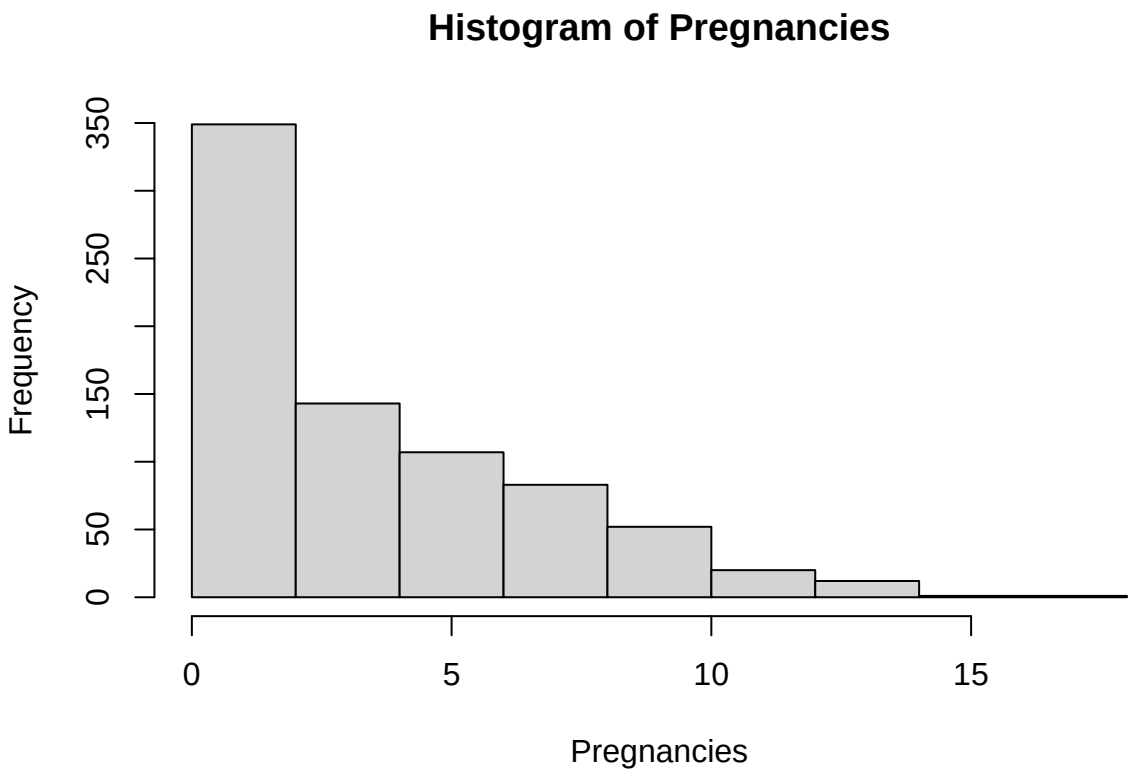
Let´s check the behaviour of our quantitative variables by themselves first. Now, we will check their histograms and boxplots.

Histograms per variable

```
hist(Age)
```

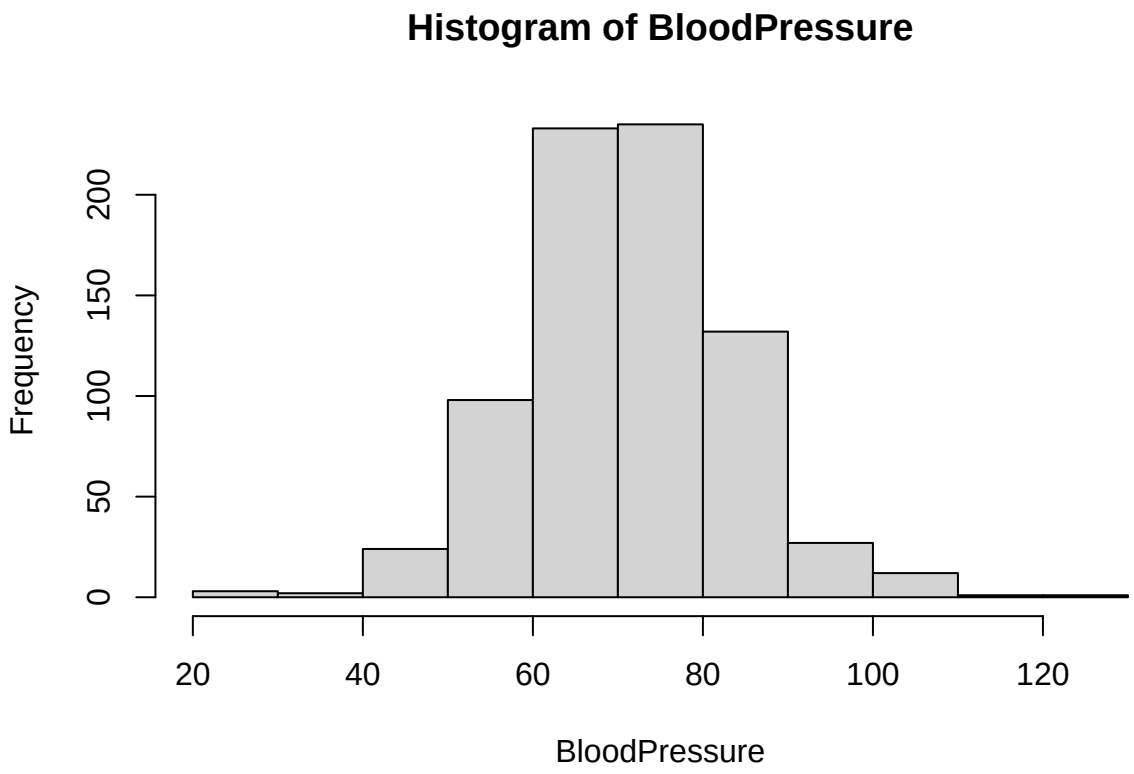



```
hist(Pregnancies)
```

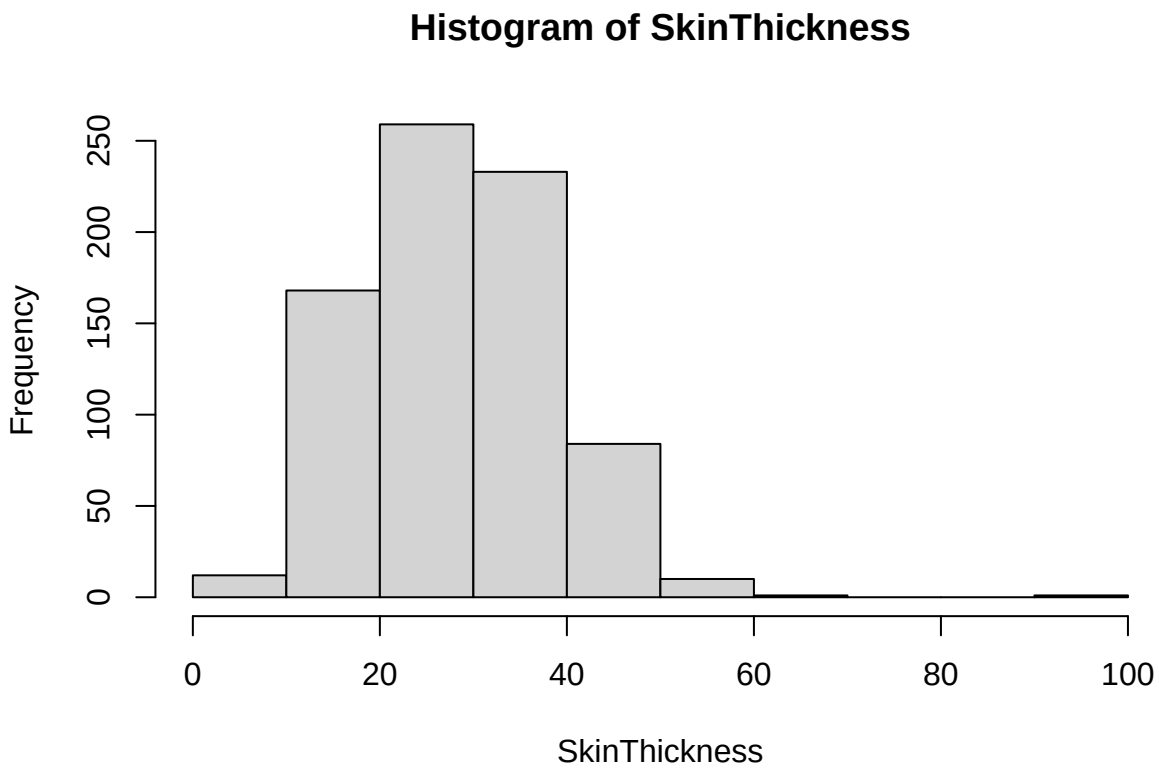



```
hist(Glucose)
```

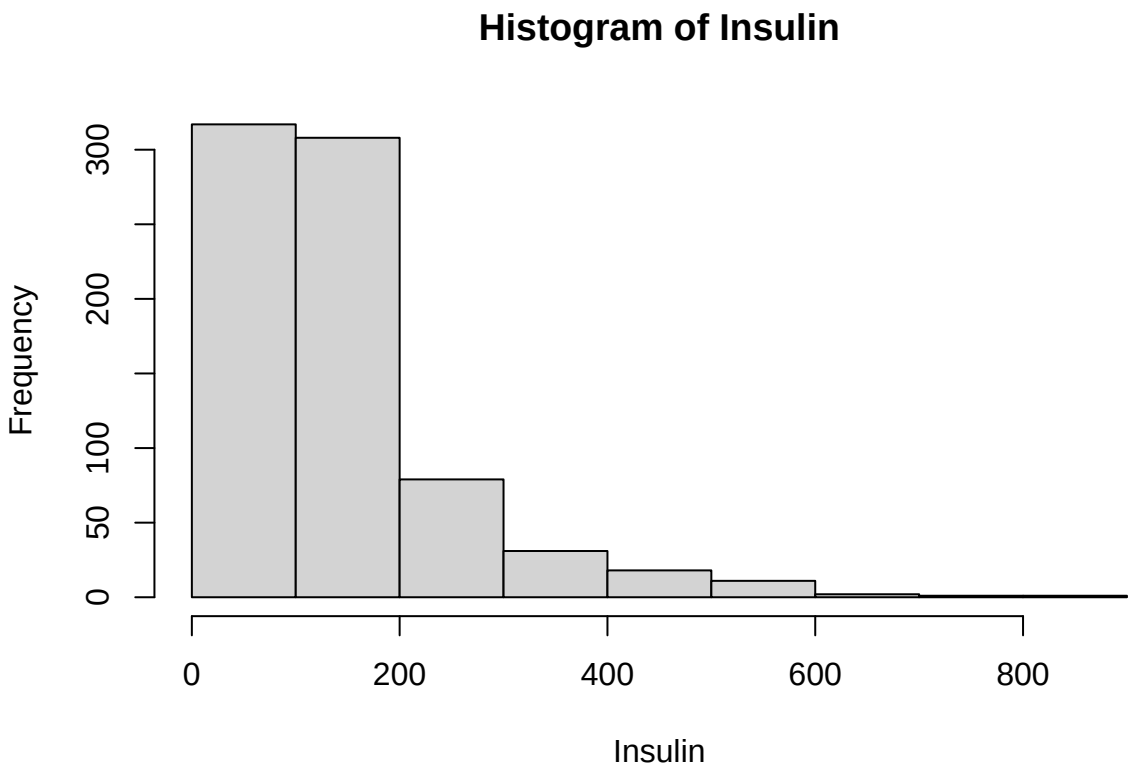

```
hist(BloodPressure)
```



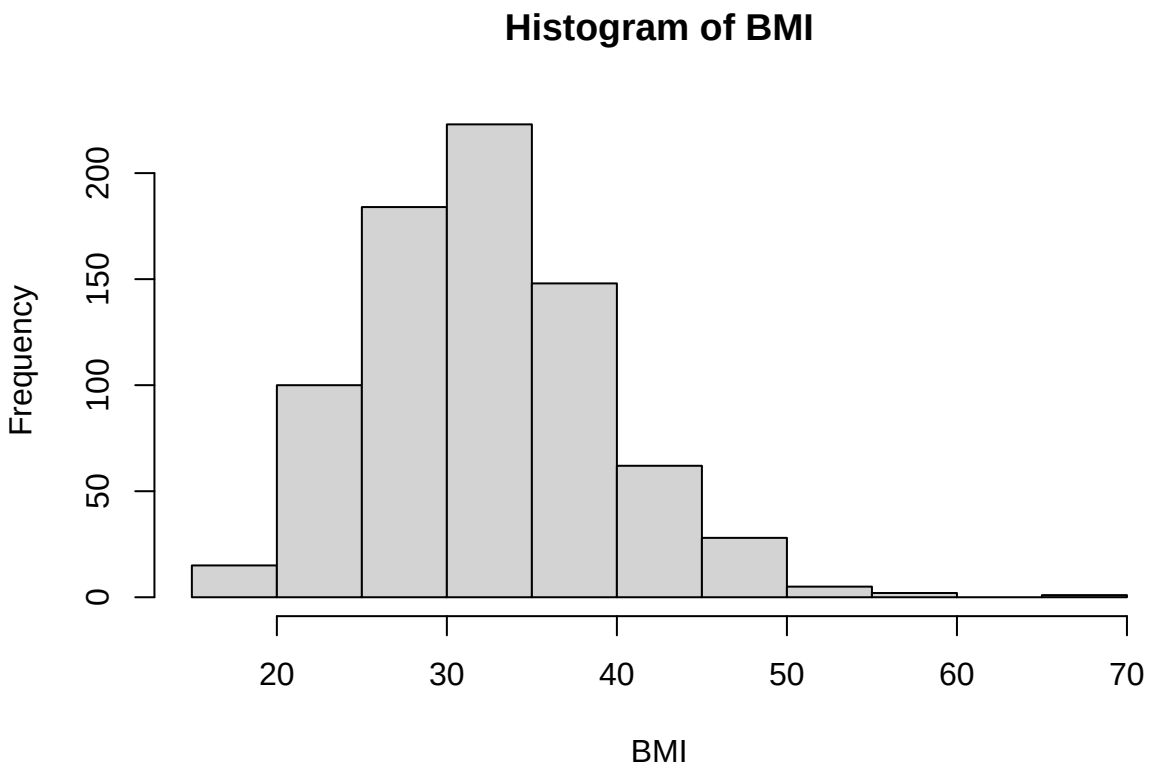

```
hist(SkinThickness)
```



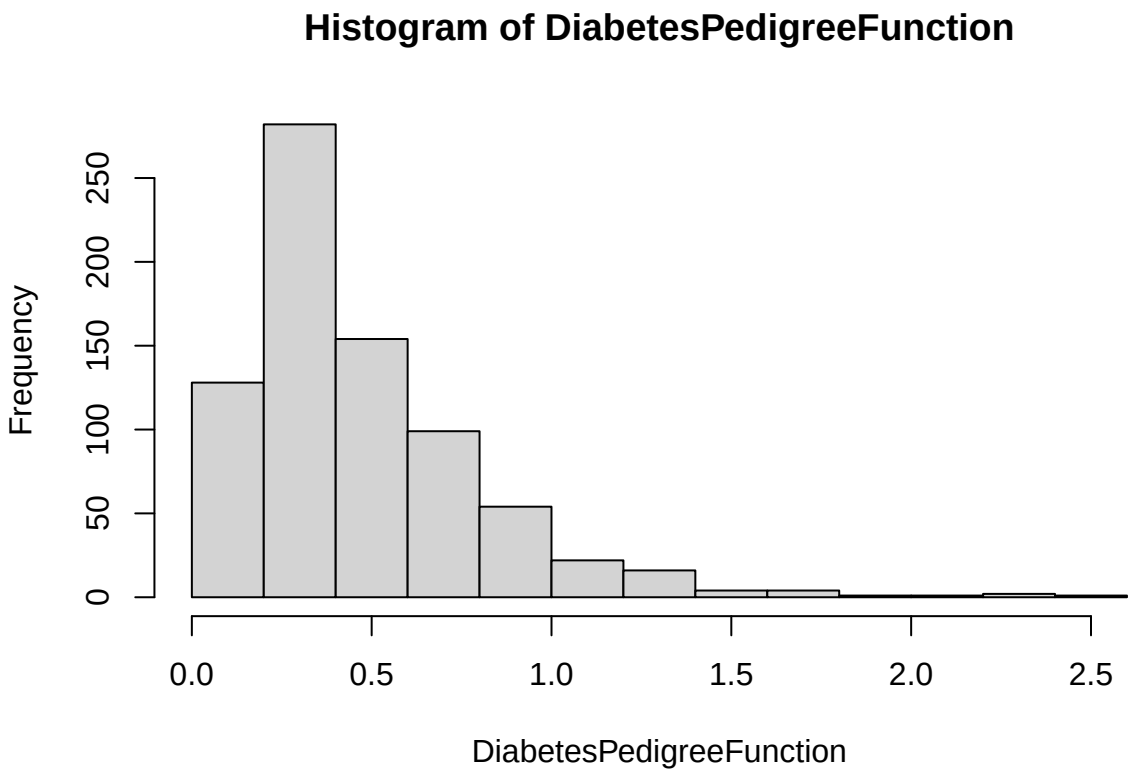

```
hist(Insulin)
```


Histogram of Insulin
Frequency
0
50
100
200
300
0
200
400
600
800
Insulin

```
hist(BMI)
```

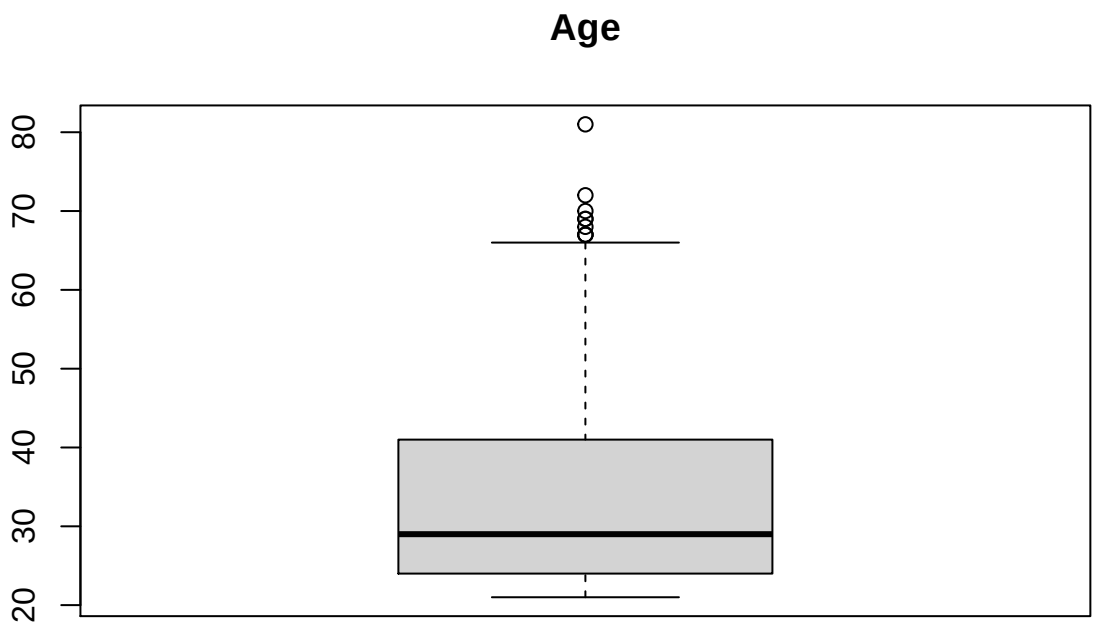



```
hist(DiabetesPedigreeFunction)
```

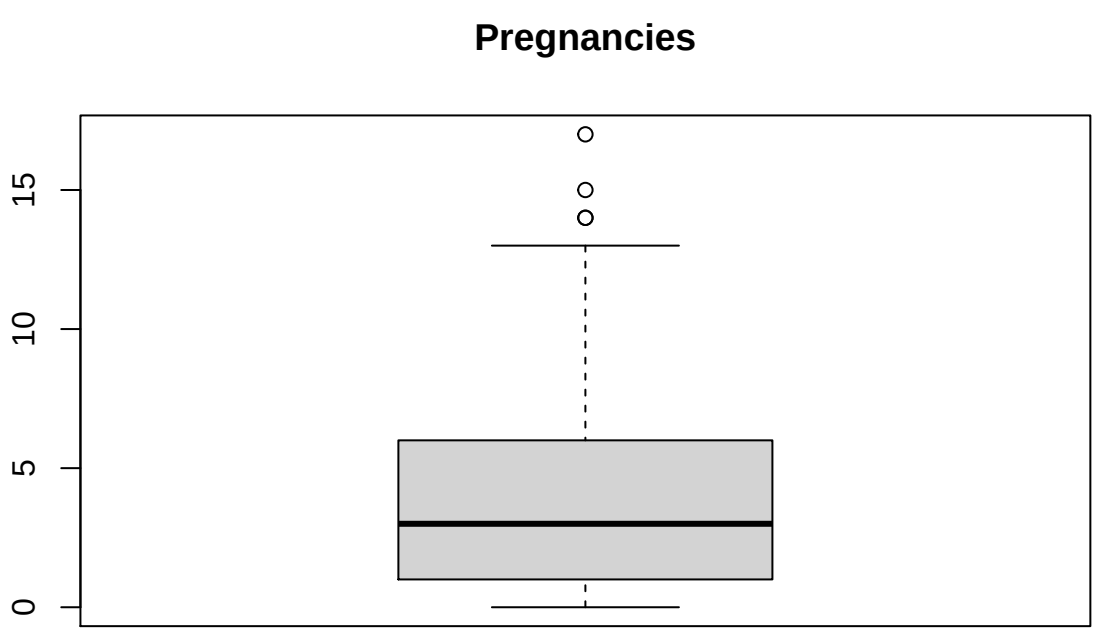


Boxplot per variable

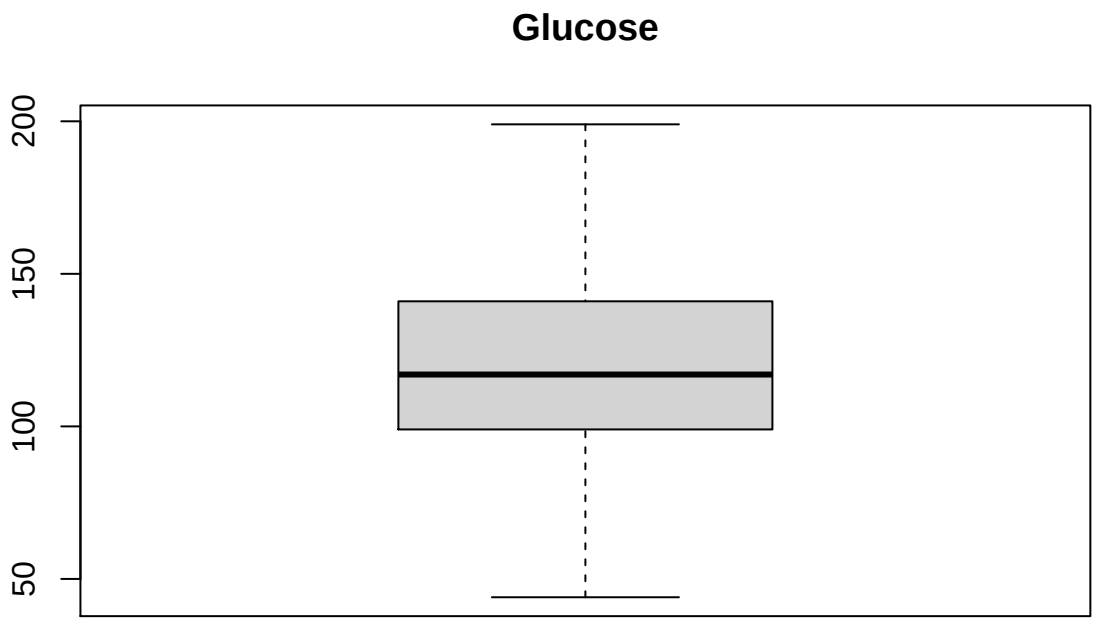
```
par(mfrow=c(1,1))
boxplot(Age,main ="Age")
```



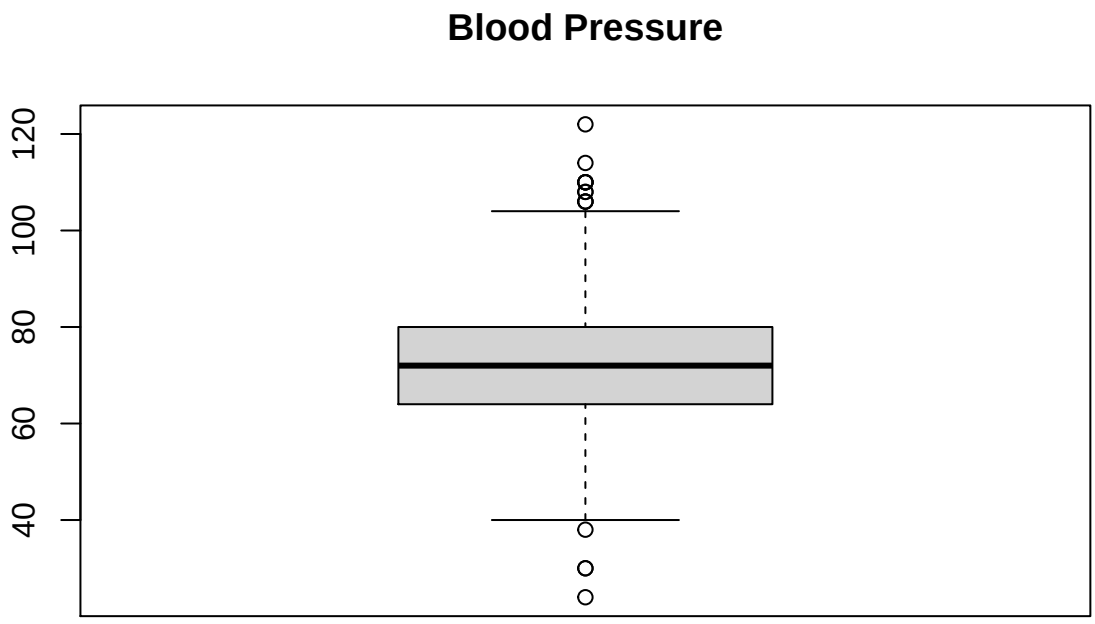

```
boxplot(Pregnancies, main="Pregnancies")
```



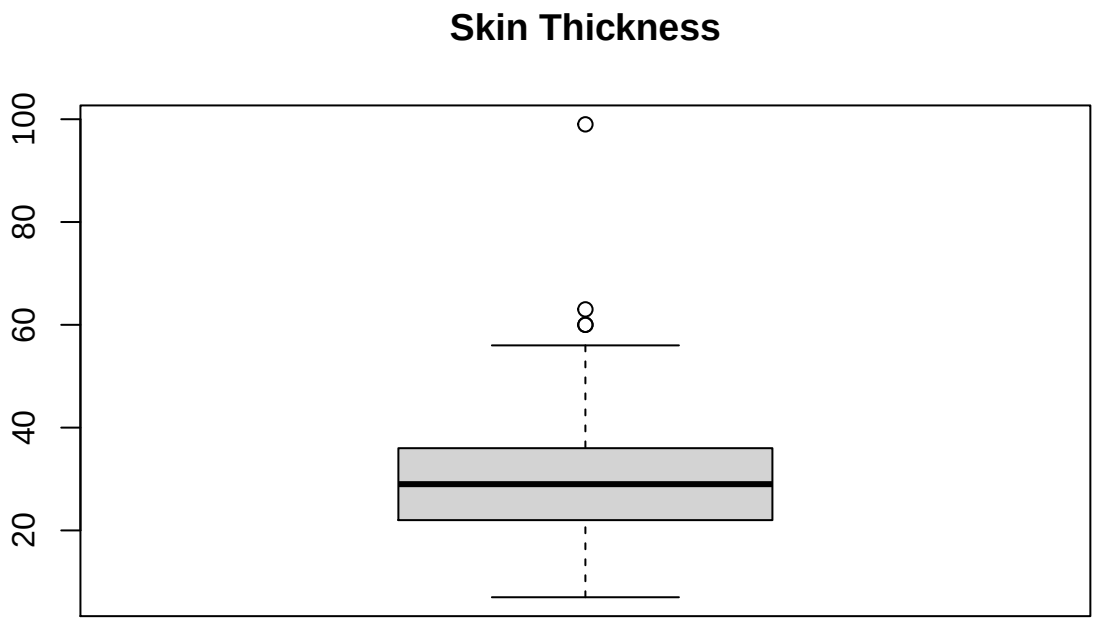

```
boxplot(Glucose,main="Glucose")
```



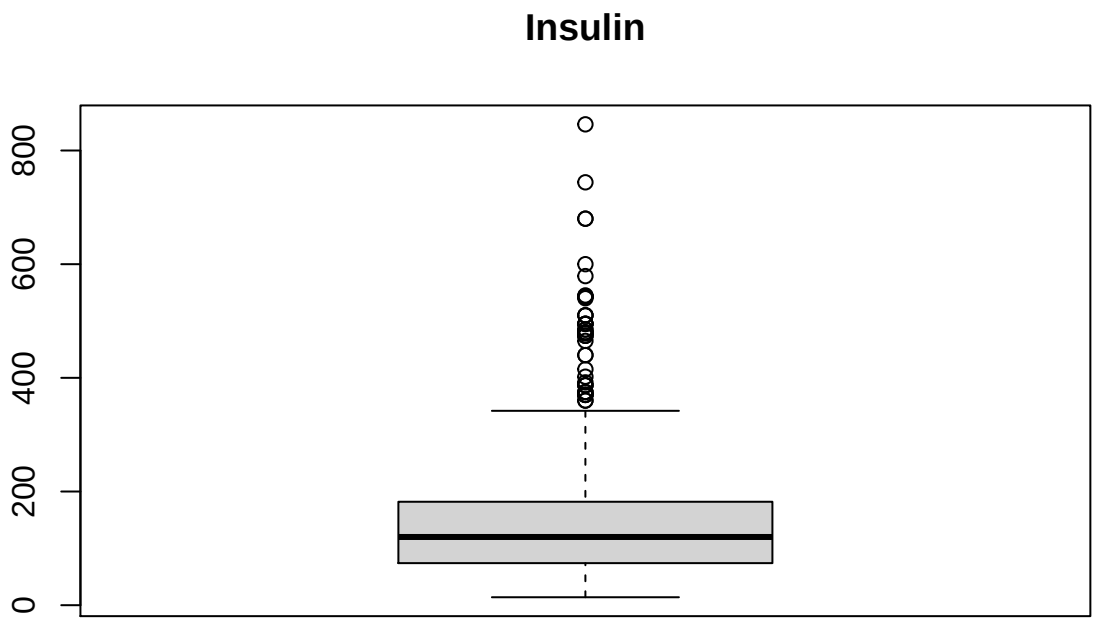

```
boxplot(BloodPressure,main="Blood Pressure")
```



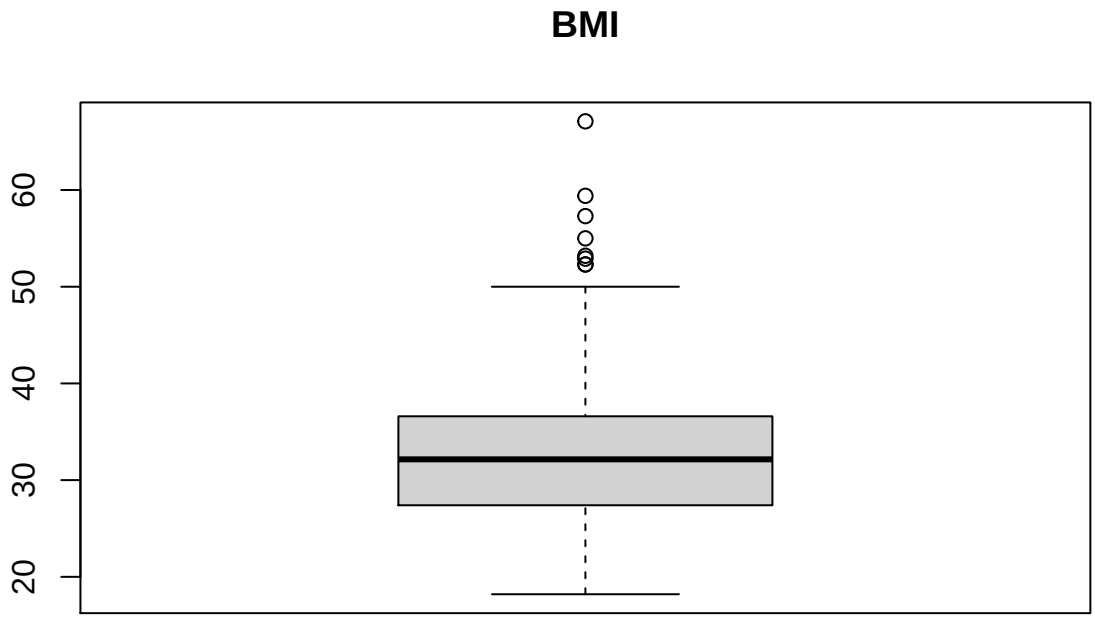

```
boxplot(SkinThickness, main = "Skin Thickness")
```



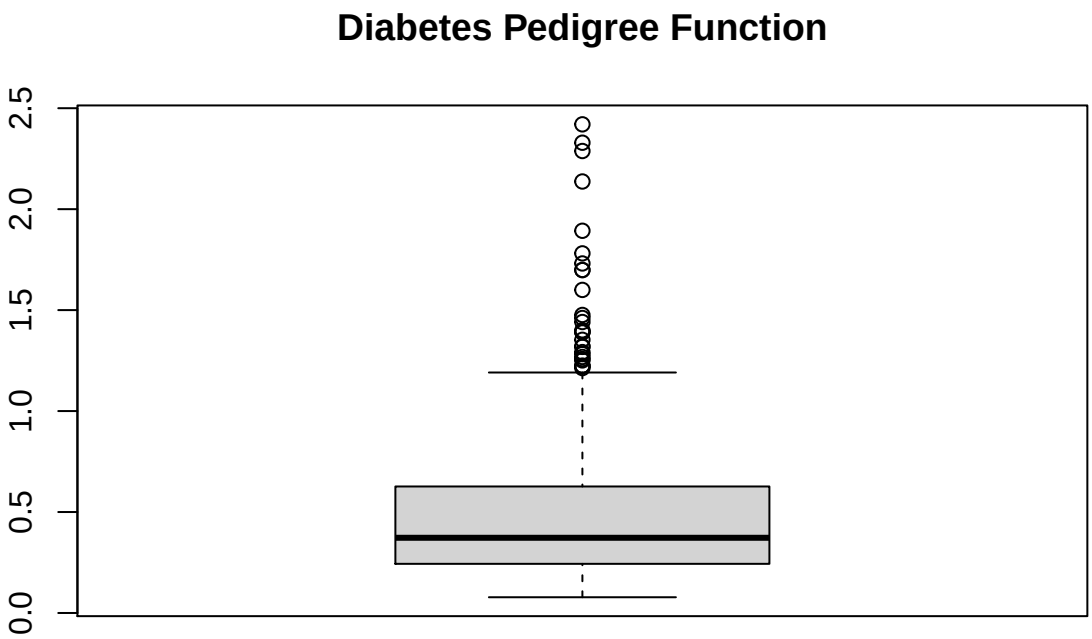

```
boxplot(Insulin, main = "Insulin")
```




```
boxplot(BMI, main = "BMI")
```



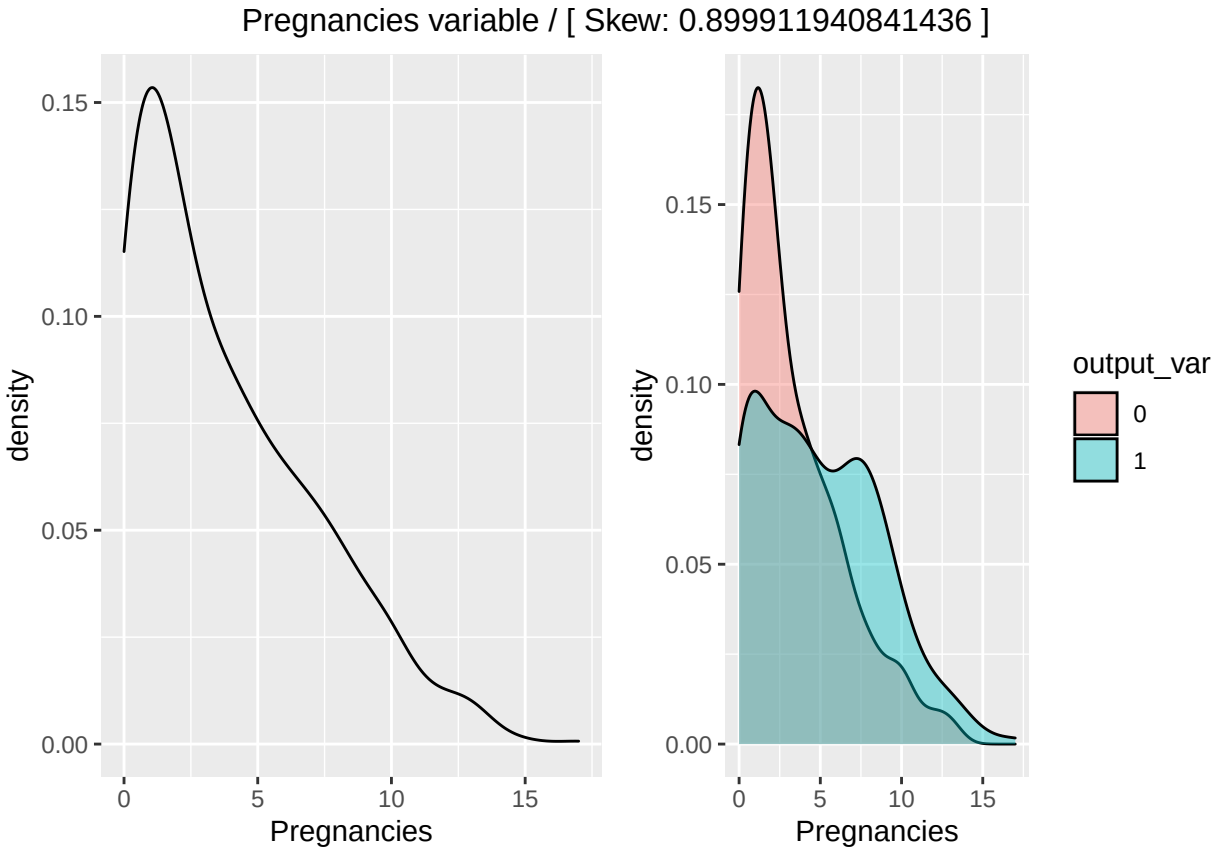

```
boxplot(DiabetesPedigreeFunction, main="Diabetes Pedigree Function")
```



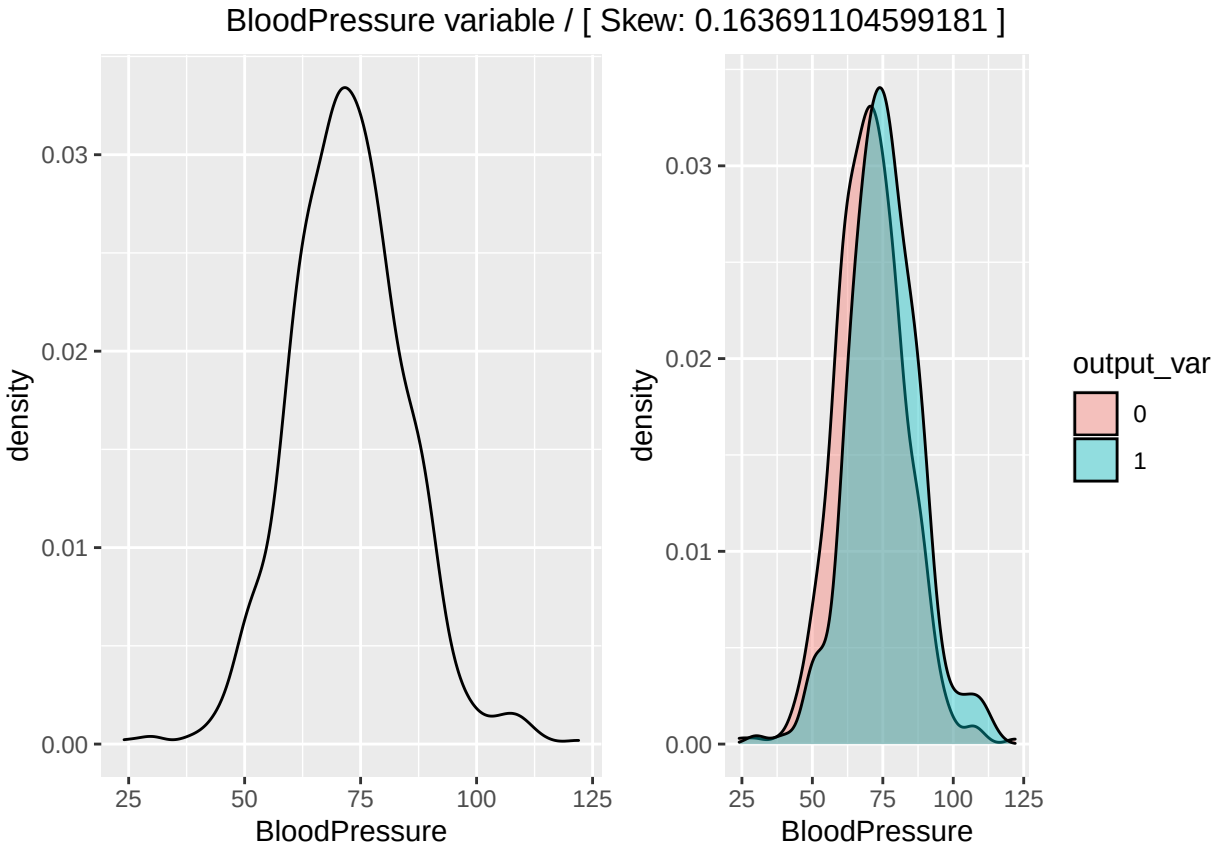
Let´s now plot the kernel for each variable by itself, and then each kernel by subgroups: diabetic and non-diabetic.

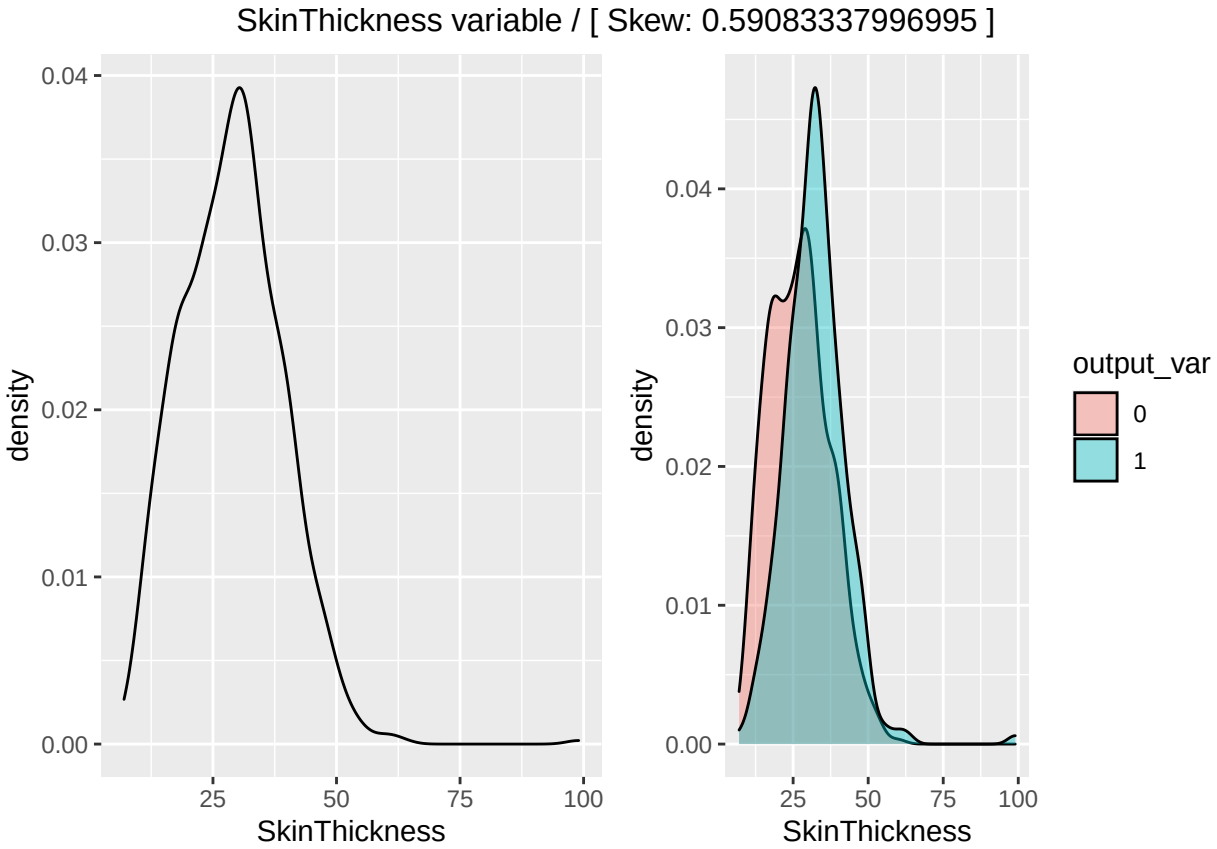
```
dat <- DATOS
dat$Outcome <- as.factor(dat$Outcome)
```

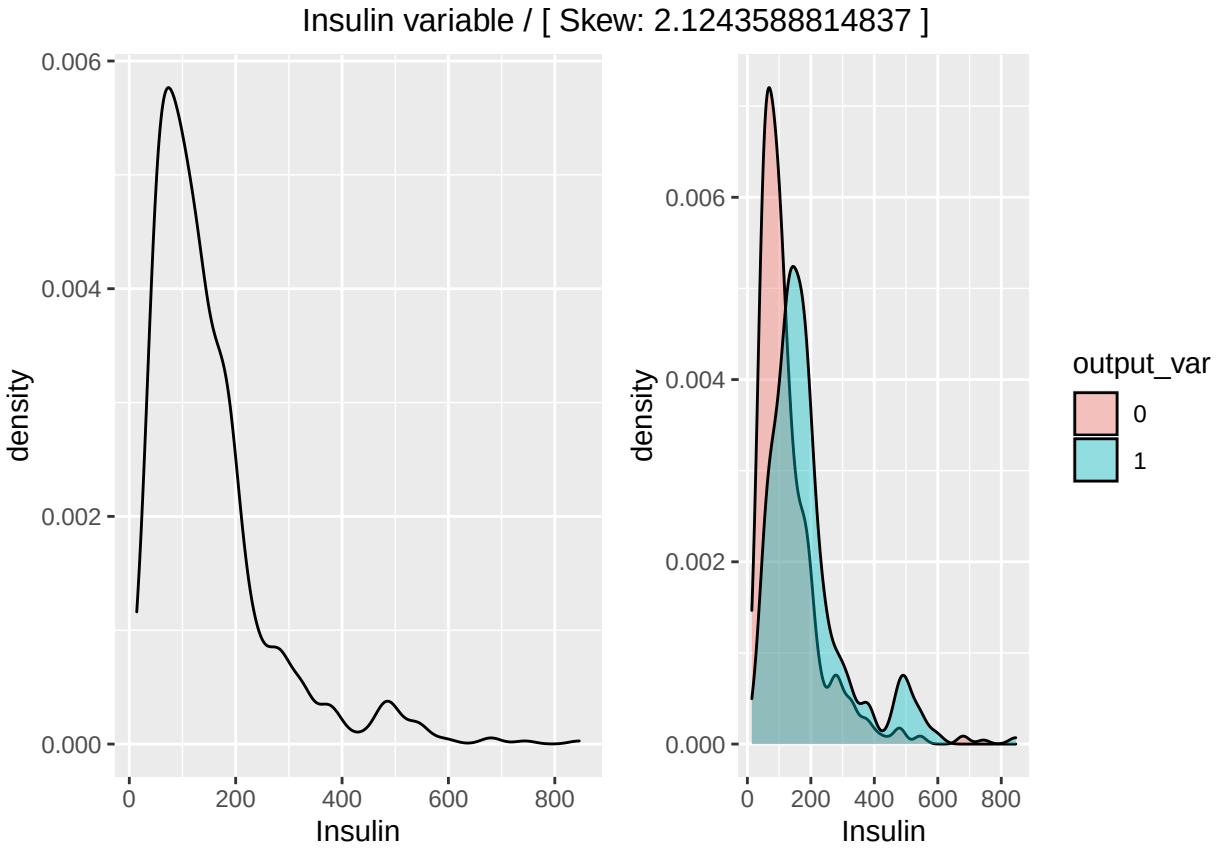
```
univar_graph <- function(univar_name, univar, DATOS, output_var) {
  g_1 <- ggplot(DATOS, aes(x=univar)) + geom_density() + xlab(univar_name)
  g_2 <- ggplot(DATOS, aes(x=univar, fill=output_var)) + geom_density(alpha=0.4) + xlab(univar_name)
  grid.arrange(g_1, g_2, ncol=2, top=paste(univar_name,"variable", "/ [ Skew:",skewness(univar),"]"))
}

for (x in 1:(ncol(DATOS)-1)) {
  univar_graph(names(DATOS)[x], DATOS[,x], DATOS, dat[,'Outcome'])
}
```

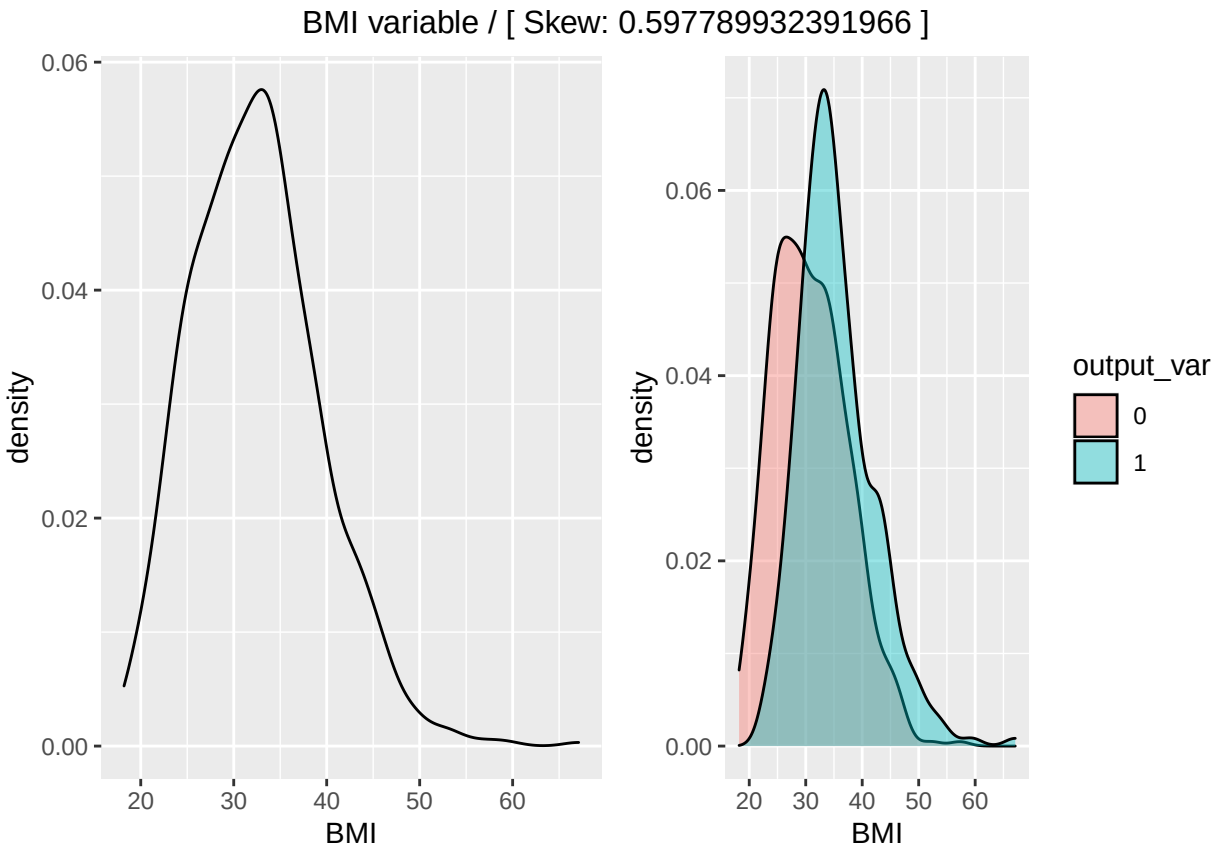

Pregnancies variable / [Skew: 0.899911940841436]
density
0.15
0.10
0.05
0.00
0
5
10
15
Pregnancies
output_var
0
1

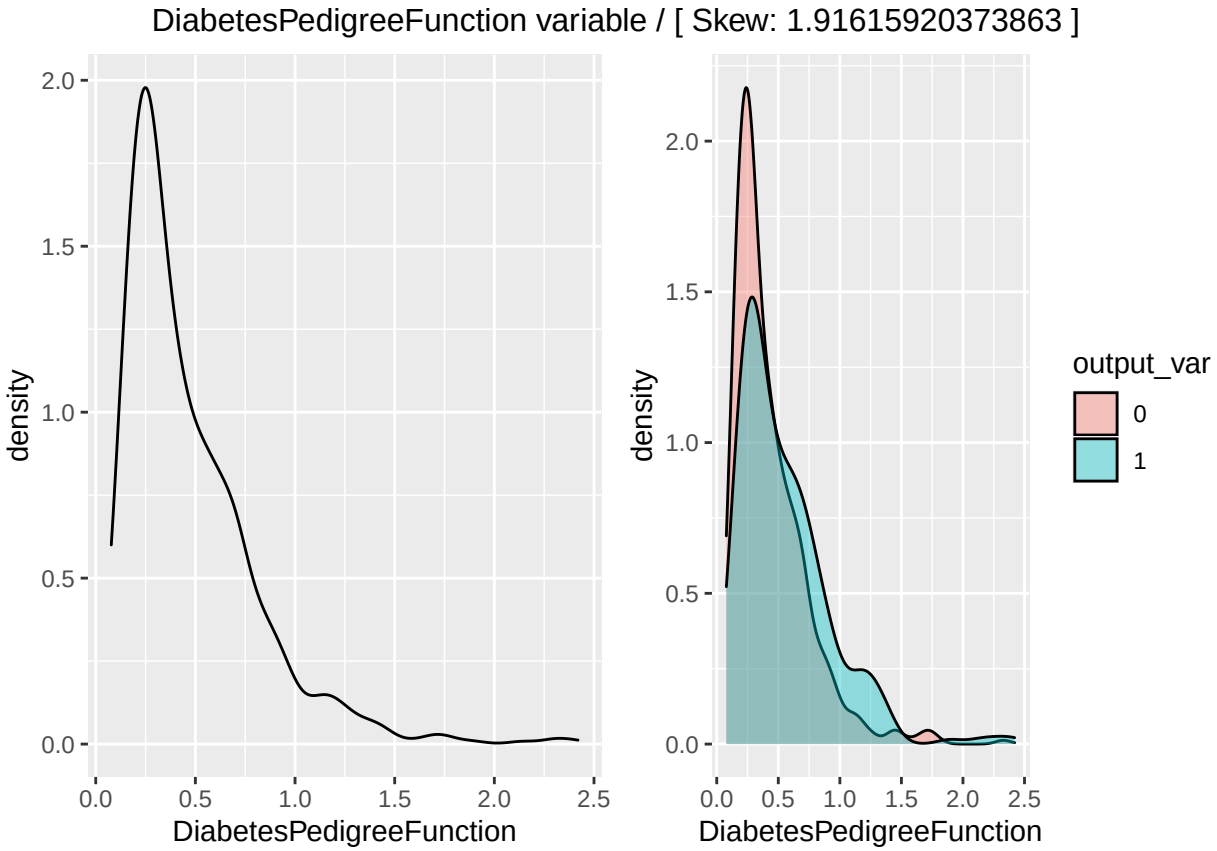

Glucose variable / [Skew: 0.532706545536145]
density
0.010
0.005
0.000
50
100
150
200
Glucose
density
0.015
0.010
0.005
0.000
50
100
150
200
Glucose
output_var
0
1

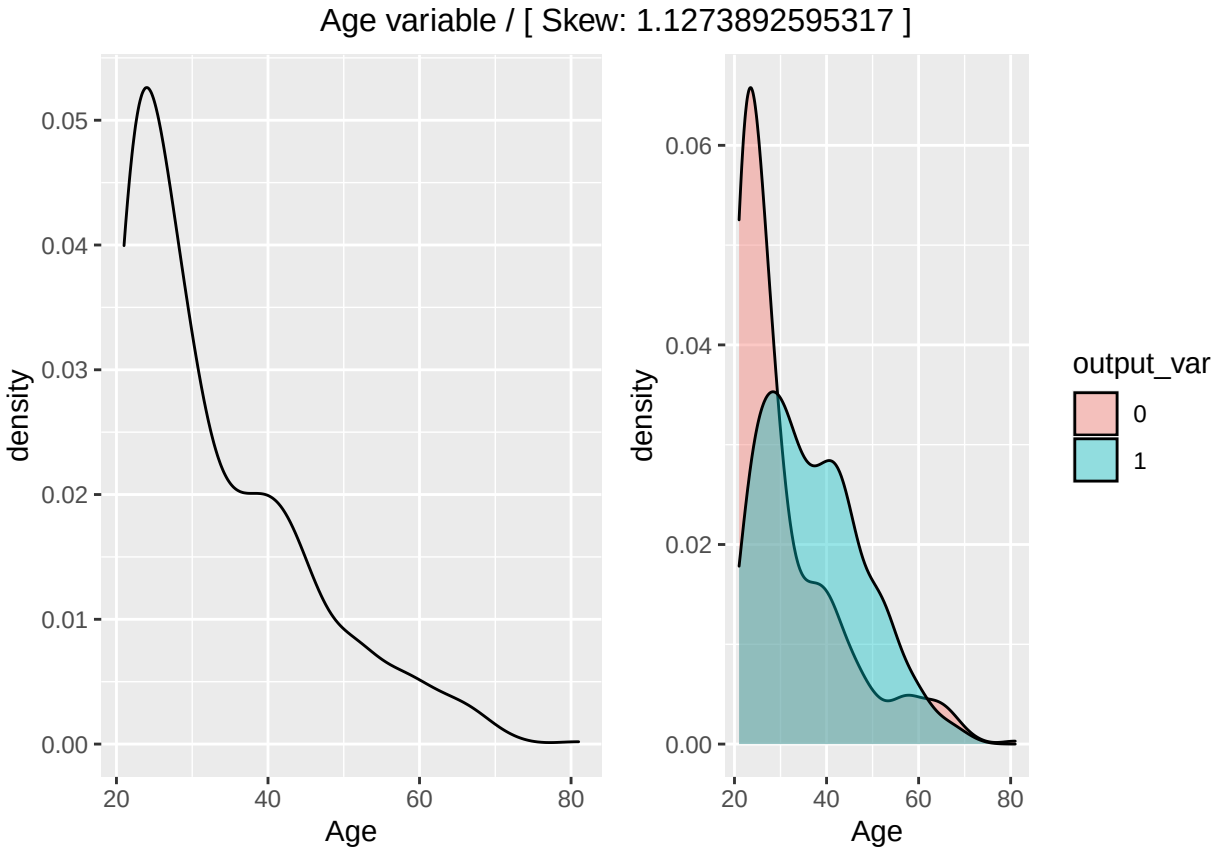

BloodPressure variable / [Skew: 0.163691104599181]
density
0.03
0.02
0.01
0.00
25
50
75
100
125
BloodPressure
output_var
0
1


SkinThickness variable / [Skew: 0.59083337996995]
density
0.04
0.03
0.02
0.01
0.00
25
50
75
100
SkinThickness
output_var
0
1

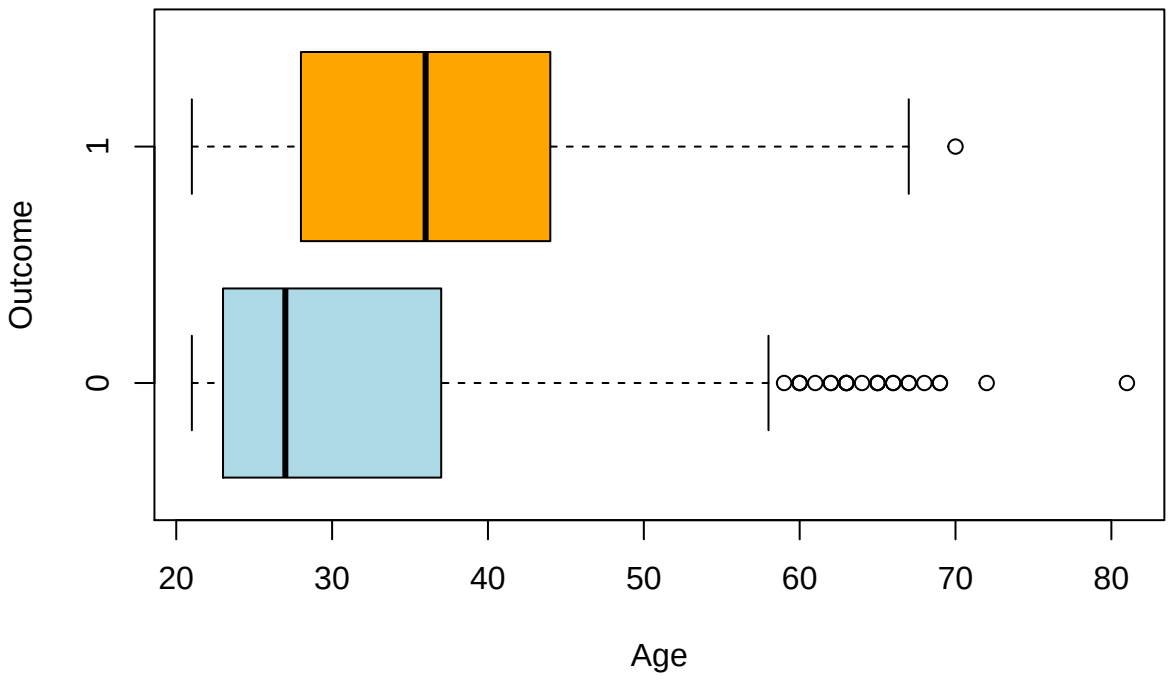

Insulin variable / [Skew: 2.1243588814837]
0.006
0.004
0.002
0.000
density
0
200
400
600
800
Insulin
output_var
0
1


BMI variable / [Skew: 0.597789932391966]
density
0.06
0.04
0.02
0.00
20
30
40
50
60
BMI
output_var
0
1

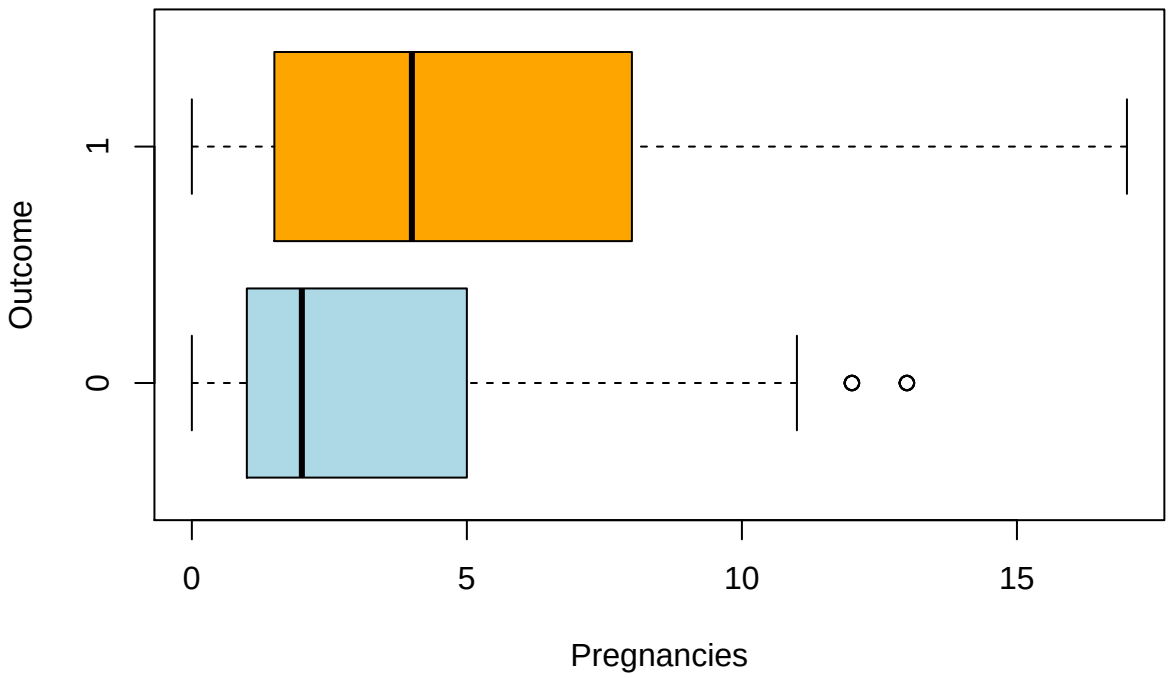

DiabetesPedigreeFunction variable / [Skew: 1.91615920373863]
density
DiabetesPedigreeFunction
output_var
0
1



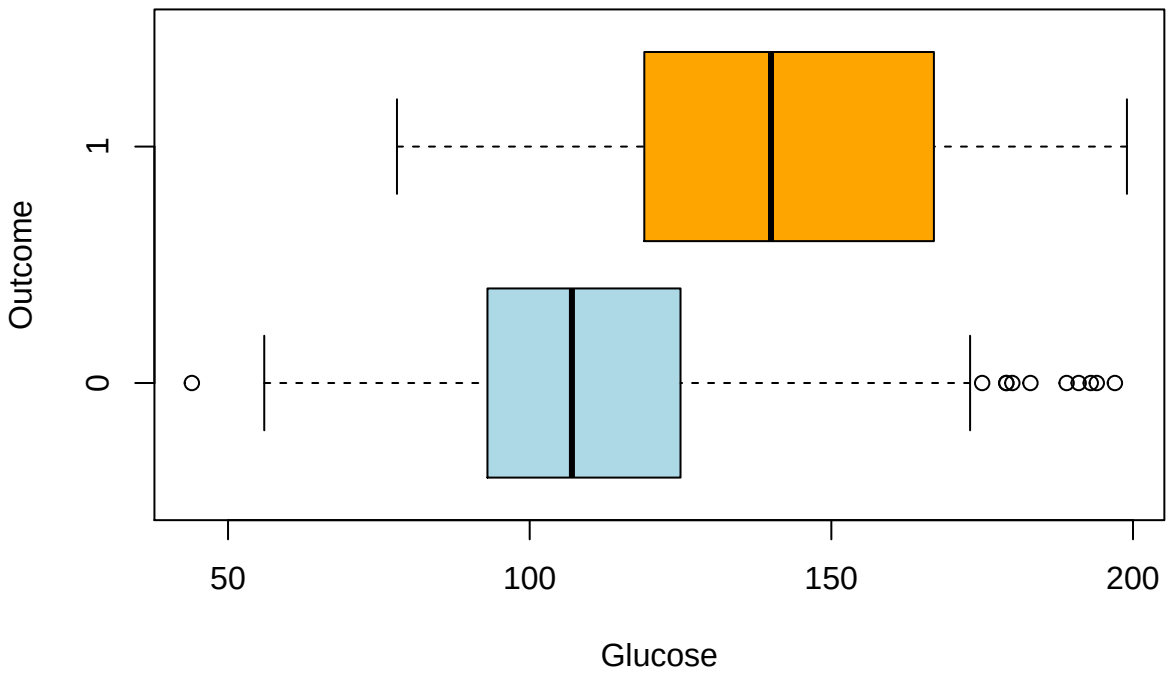

```
par(mfrow=c(1,1))

boxplot(Age~Outcome,ylab="Outcome",xlab="Age",col=c("lightblue","orange"),horizontal=TRUE)
```



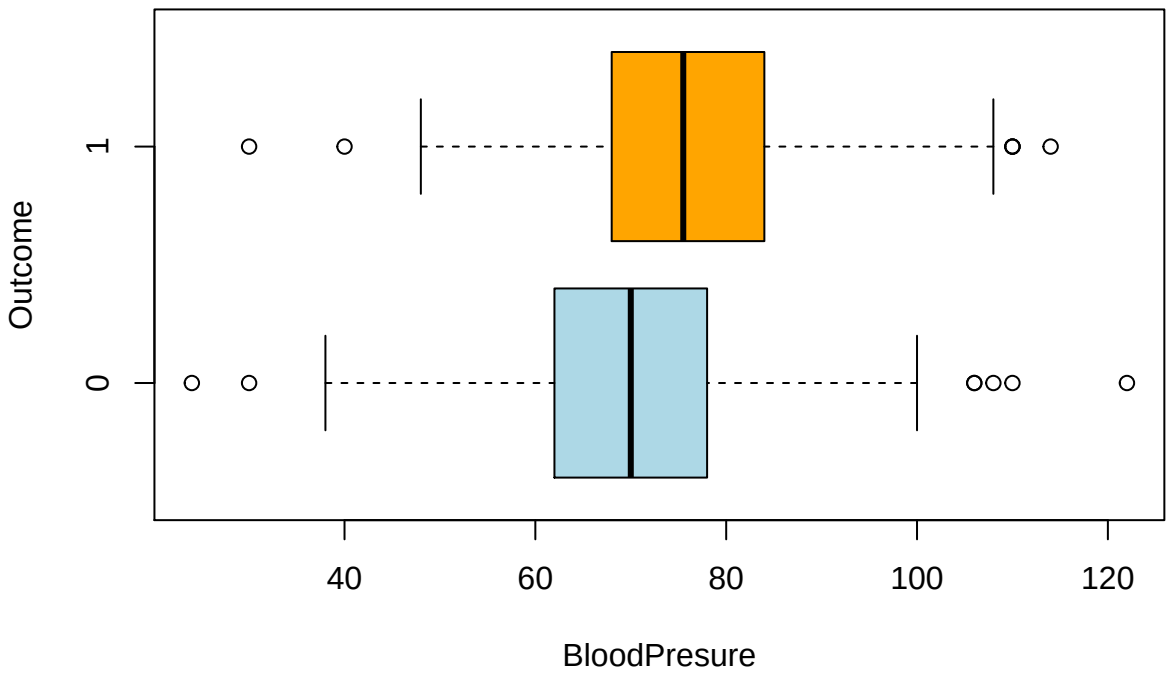

```
boxplot(Pregnancies~Outcome,ylab="Outcome",xlab="Pregnancies",col=c("lightblue","orange"),horizontal=TRU
```



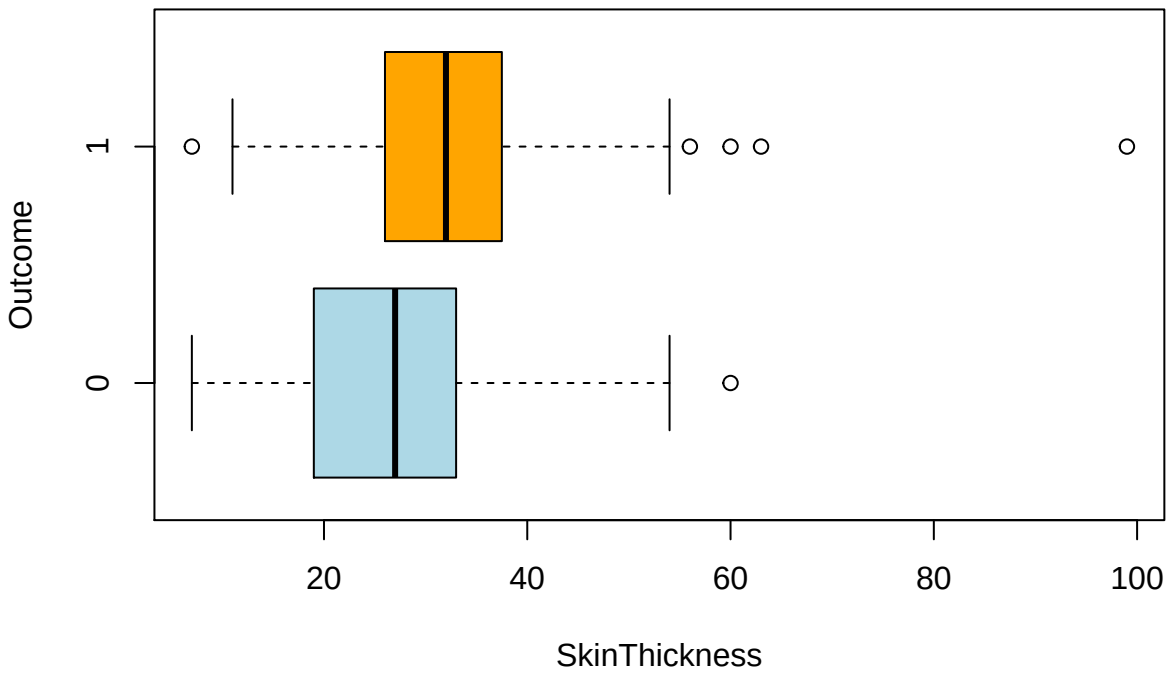

```
boxplot(Glucose~Outcome,ylab="Outcome",xlab="Glucose",col=c("lightblue","orange"),horizontal=TRUE)
```



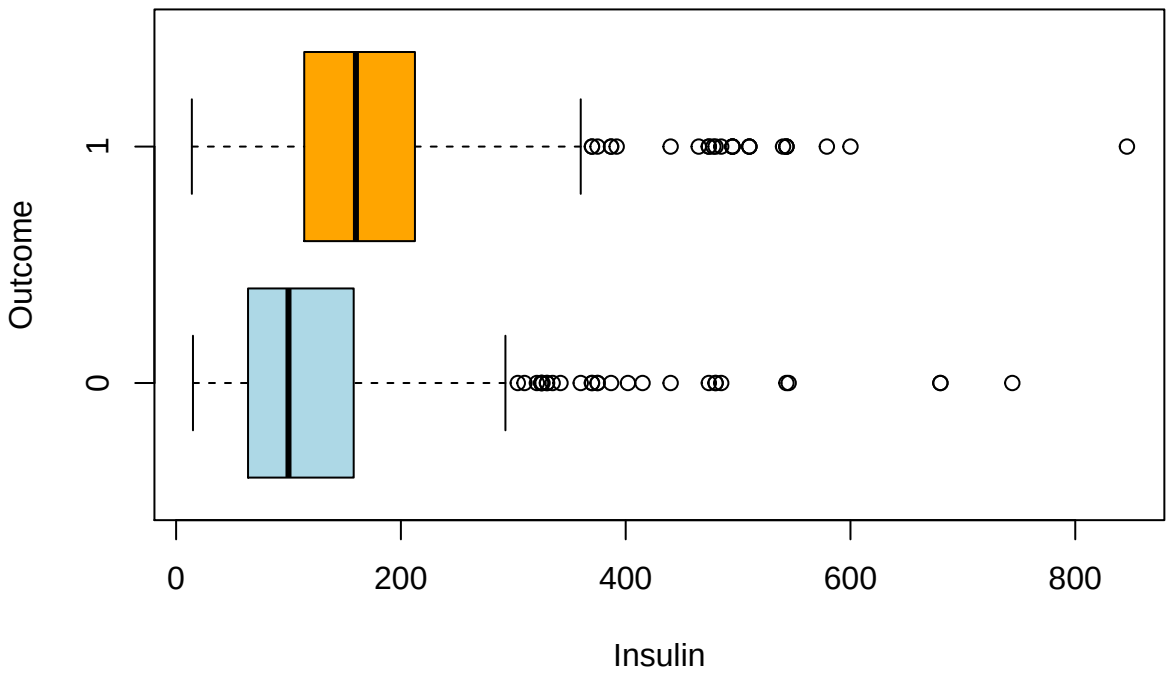

```
boxplot(BloodPressure~Outcome,ylab="Outcome",xlab="BloodPresure",col=c("lightblue","orange"),horizontal=
```



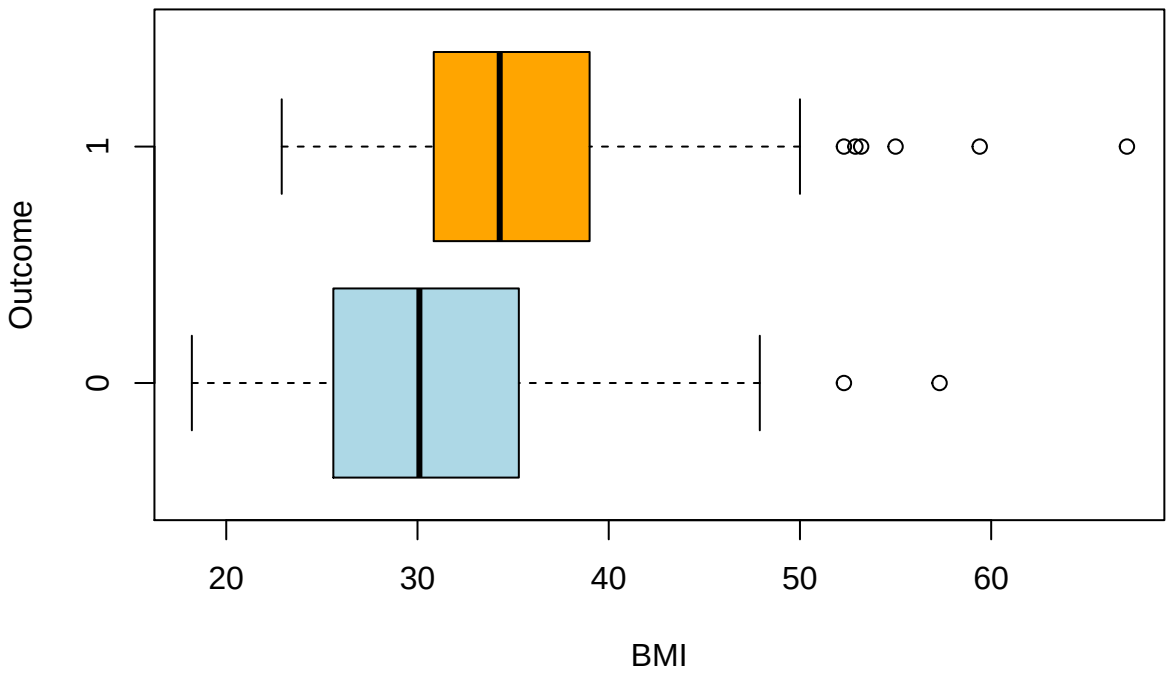

```
boxplot(SkinThickness~Outcome,ylab="Outcome",xlab="SkinThickness",col=c("lightblue","orange"),horizontal
```



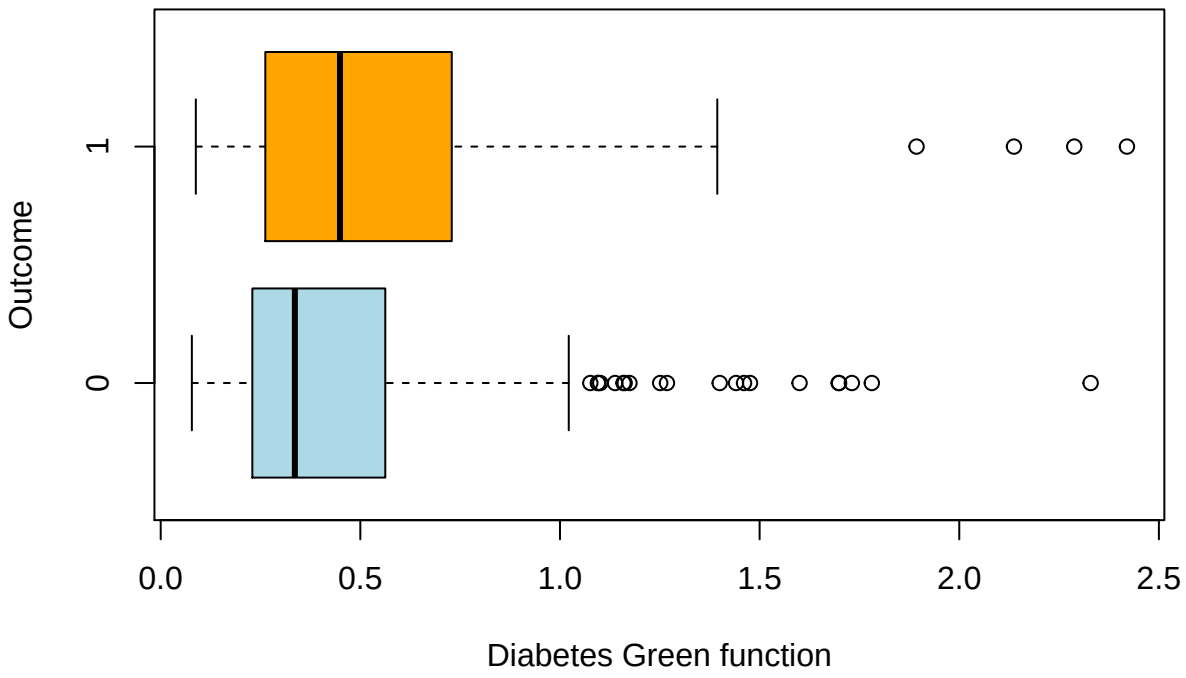

```
boxplot(Insulin~Outcome,ylab="Outcome",xlab="Insulin",col=c("lightblue","orange"),horizontal=TRUE)
```




```
boxplot(BMI~Outcome,ylab="Outcome",xlab="BMI",col=c("lightblue","orange"),horizontal=TRUE)
```




```
boxplot(DiabetesPedigreeFunction~Outcome,ylab="Outcome",xlab="Diabetes Green function",col=c("lightblue
```



Let´s comment on the plots done above.

Age

This variable account for the age of each women in the sample. It varies from a minimum of 21 to a maximum of 81. This means that were included women in that age frame. As we can see in the plot below, the median is on 29, meaning that half of the women are more that 29 years old and the other half is older than 29. As the first quartile is on 24 and the third is on 41, half of the women from this sample is between those ages (24 and 41). This shows us that the population is relatively young, although there are 25% of women in the sample that are from 41 to 81 years old.

At the same time, we can see that when comparing by outcome (if they resulted ill or not) the median of the diabetic ones is higher than the non-diabetic ones. The distribution of this variable looks more left-skewed for the non-diabetic women, in comparison with the ones that are diabetic. This, all together, may mean that inside the non-diabetic woman, most of them are pretty young, and some of them are old. The same happens with the diabetic women: half of them are younger than 35, and the rest are older.

Blood Preassure

When studying blood pressure, we must first say that it is measured in mm/gg. The median is around 70, and the distribution looks a bit symmetrical. When dividing the observations between diabetic and not diabetic, we can say that both distributions look pretty similar.

BMI

The body mass index is a very useful index to take into account the weight, but related to the height of the person. This way, it is measured: weight (in Kg) / [height (in meters)] ^2. As we can see both in the histogram and in the boxplot, the distribution seems symmetrical and the size of the box is quite small (so half of the women of the sample, between the first and third quantile, are concentrated in the values 27.30 and 36.60. Normally having a value between 25 and 30 is considered right, while more than 30 is considered overweight, and more than 35 obesity.

The Diabetes Pedigree Function

The diabetes Pedigree Function scores the likelihood of diabetes based on family history. So the larger value this variable takes, the more likely to be diabetic. As we can see in the plots, our distribution in this case seems right skewed. At the same time, when comparing between groups (diabetic and not diabetic) the distributions look similar, although the third quartile for the diabetic group goes further than the not diabetic one. At the same time, the 4th quartile looks bigger in the case of the not diabetic group

Glucose

Glucose measures the plasma glucose concentration over 2 hours in an oral glucose tolerance test. As it can be seen, the median is 117 and is very close to the mean (120.9) meaning that it is a bit symmetrical. At the same time, there is some skewedness, as the tail on the right looks much heavier. When classifying between diabetic or not, the median of level of plasma glucose in women with diabetes is higher than the ones that have no diabetes. At the same time, the diabetic women present a wider range of glucose values between the 1st. and 3rd. quartile.

Insulin

Insulin is measured in mu U/ml and its distribution looks right-skewed. When looking at the histograms, most of the values are situated in the left side of the distribution, meaning that most women of the sample have low insulin level. A “standard” or “normal” level for this 2 hour test is considered at around 16 to 166 U/ml and the median of our variable in the sample is 30.5 (not very high). When dividing it between diabetic or not, both distributions look similar (with the same long tail) and the biggest interval of frequency at the lowest level of the distribution.

Pregnancies

This variable account for the number of pregnancies that each women had during their life. The minimum in the sample is 0 pregnancies and the maximum is 17 pregnancies. The distribution has a heavier tail on the left, meaning it is right-skewed. This is coherent with the fact that most of the woman from the sample are relatively young, and thus did not have so many kids yet. This explains the fact that the median is at 3, meaning that half of the women have less than 3 kids while the rest account for more than 3. Then, the third quartile is at 6, meaning that only 25% of the women fron the sample have between 6 and 17 kids. When studying this variable by group of diabetic and not diabetic, we can say that the median of the diabetic women is higher than the one of not diabetic.

Skin Thickness

The skin thickness measures the triceps skin fold thickness in mm. As it can be seen in the plots, the distribution looks pretty right skewed, meaning that most of the observation occur in the smallest values of the variable. The median is at 23, meaning 50% of the women have less than 23mm of skin thickness

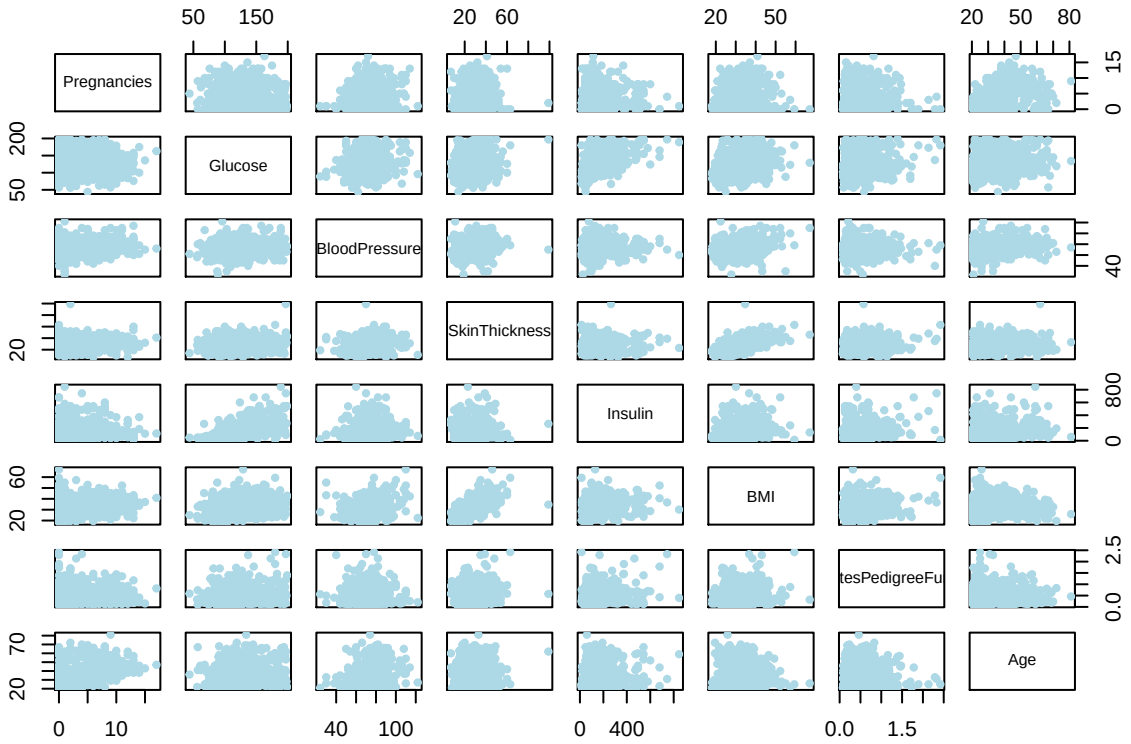
while the other half have more. At the same time, when dividing between groups of diabetic or not, both distribution look similar, although the diabetic looks less dispersed and its median is larger than the ones of non-diabetics.

In general, there are some variables that are highly positively skewed (Insulin, DiabetesPedigreeFunction, Age) while others are highly negative skewed like BloodPressure.

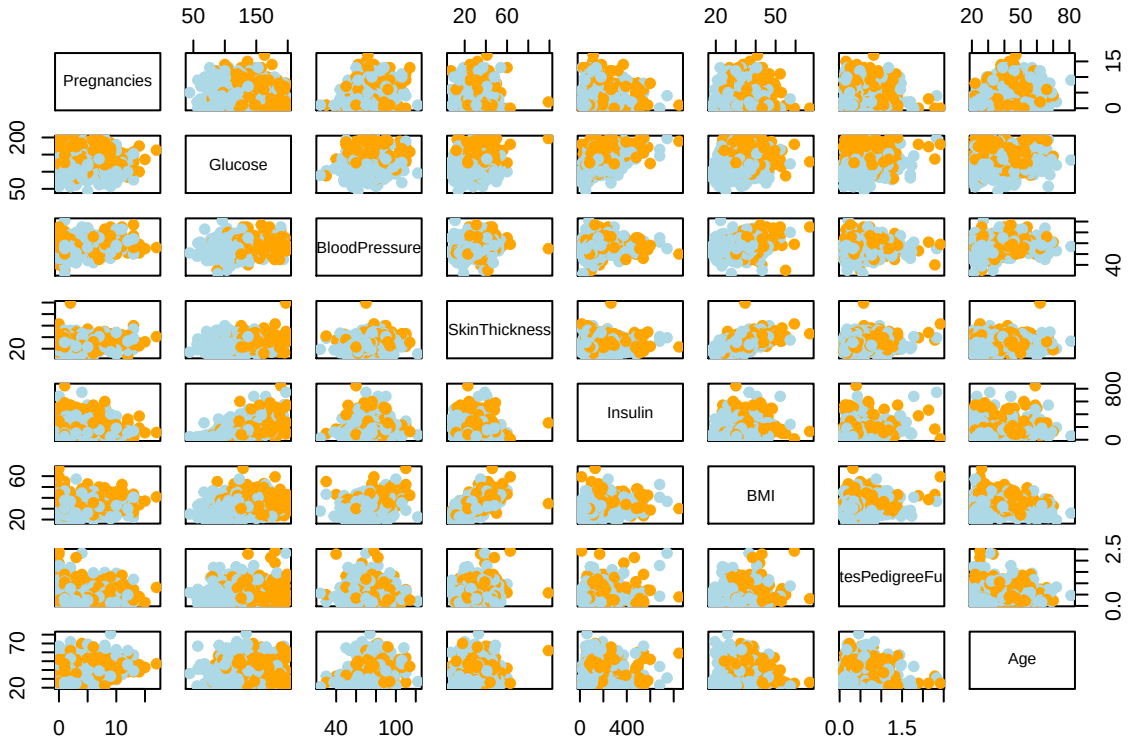
Multivariate Analysis

We now perform a matrix of plots, showing possible correlation between different variables (every variable against every variable). We can not say we see a clear pattern of linear correlation between variables. At least, not yet.

```
X_quan <- DATOS[,1:8]
pairs(X_quan,pch=20,col="lightblue")
```

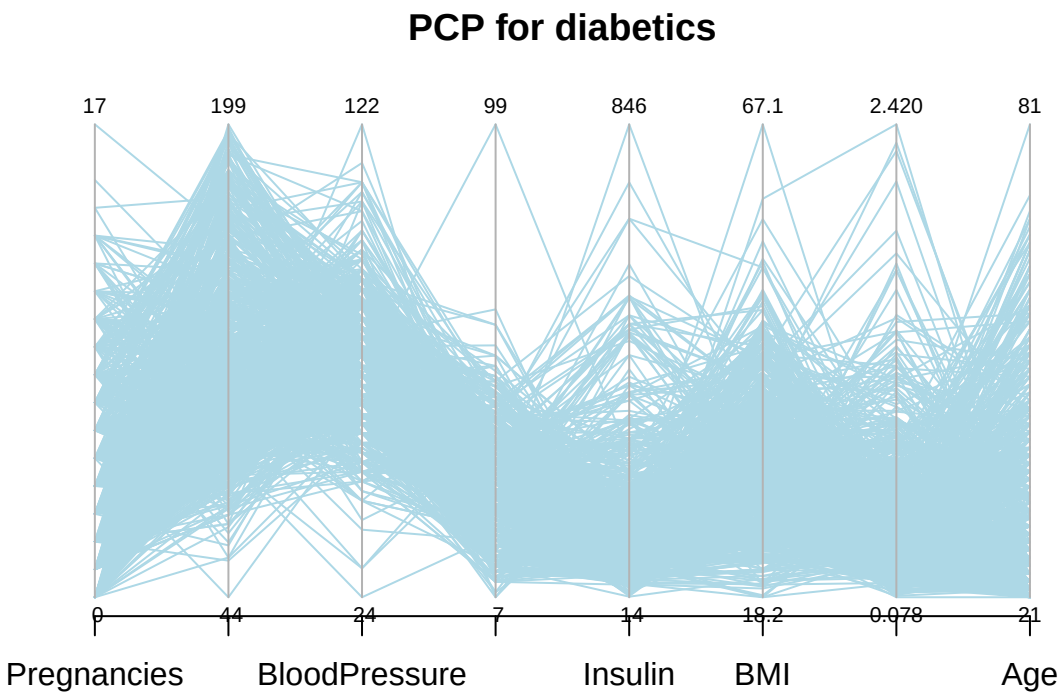



```
colors_Student <- c("lightblue","orange")[1*(Outcome=="1")+1]
pairs(X_quan,pch=19,col=colors_Student)
```

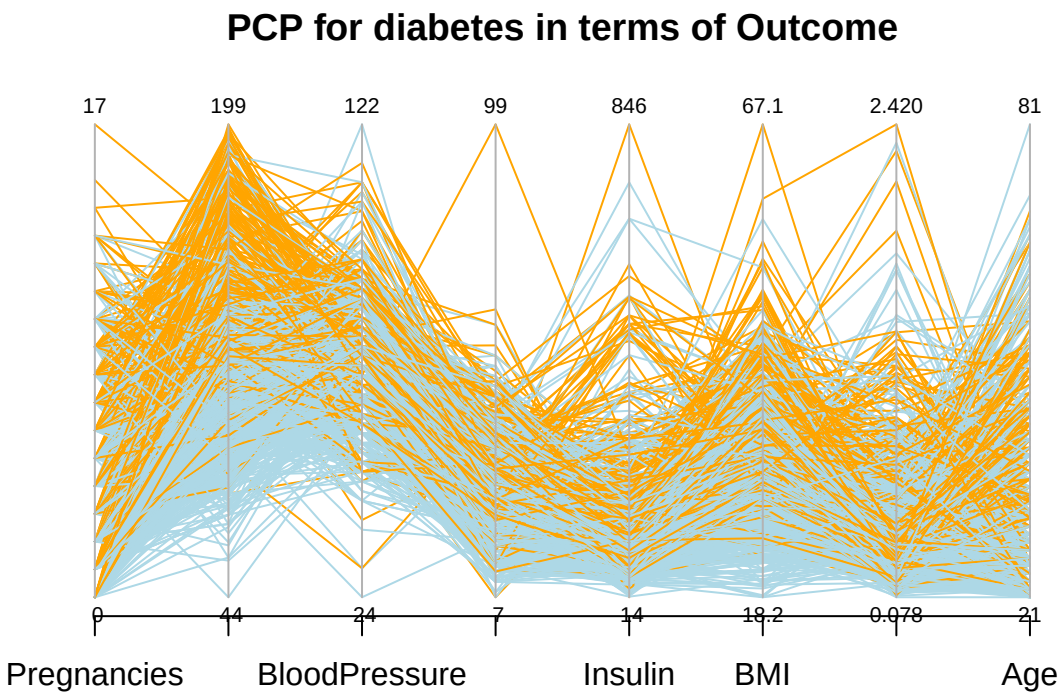


Then, we perform a PCP. The Parallel Coordinates Plot is very useful to visualize and analyze high-dimensional data.

```
parcoord(X_quan,col="lightblue",var.label=TRUE,main="PCP for diabetics")
```

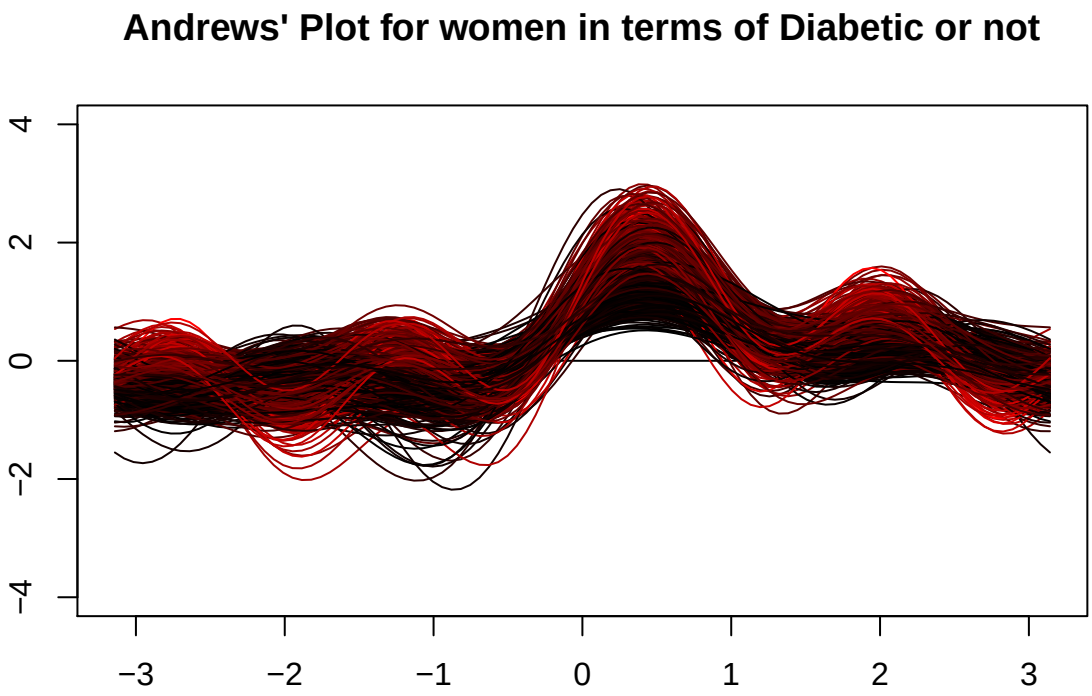



```
parcoord(X_quan,col=colors_Student,var.label=TRUE,main="PCP for diabetes in terms of Outcome")
```



As we might expect, being diabetic is directly related to glucose level. There is also a lesser correlation between the other variables and diabetes, except in cases of pregnancy and diabetes pedigree function in which there is no relationship.

```
par(mfrow=c(1,1))
andrews(as.data.frame(cbind(X_quan,as.factor(DATOS[,9]))),clr=8,ymax=4,main="Andrews' Plot for women in
```



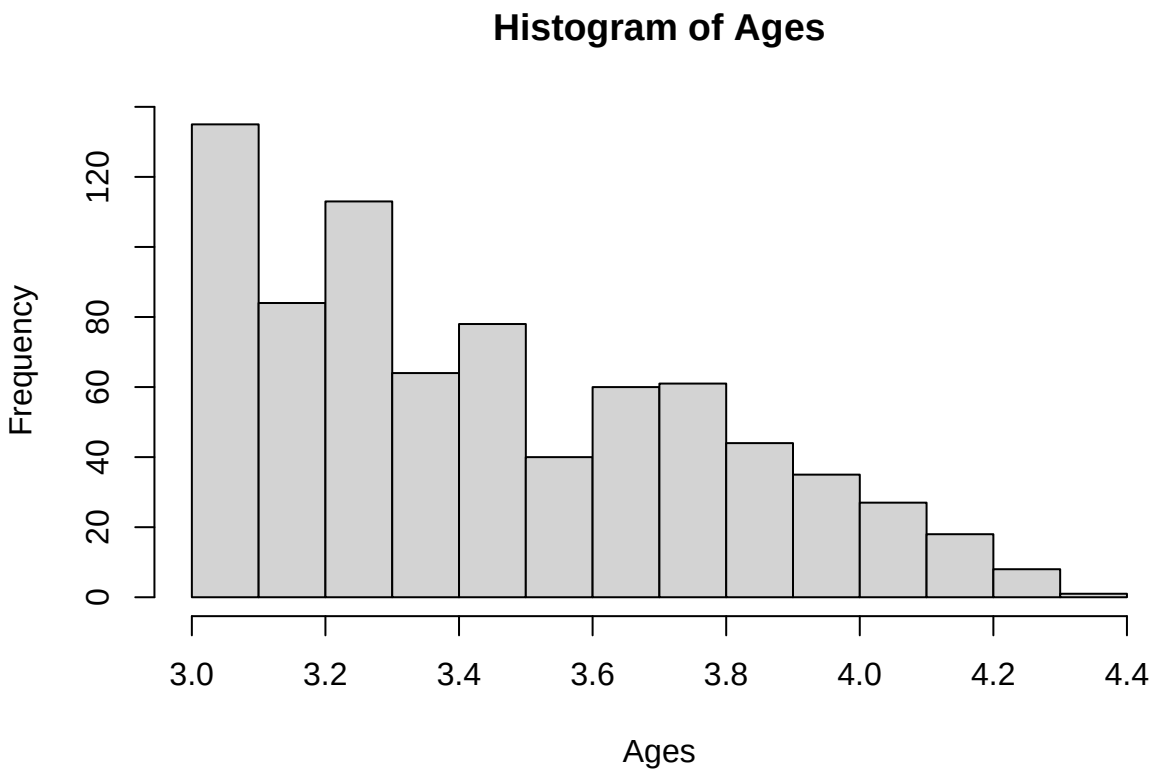
Outliers Detection

In order to study the mean vector, covariance and correlation matrix, we need to take into account outliers, as these measures tend not to be robust. This means that the presence of outliers may change some of our conclusions. This is why we will do this treatment now.

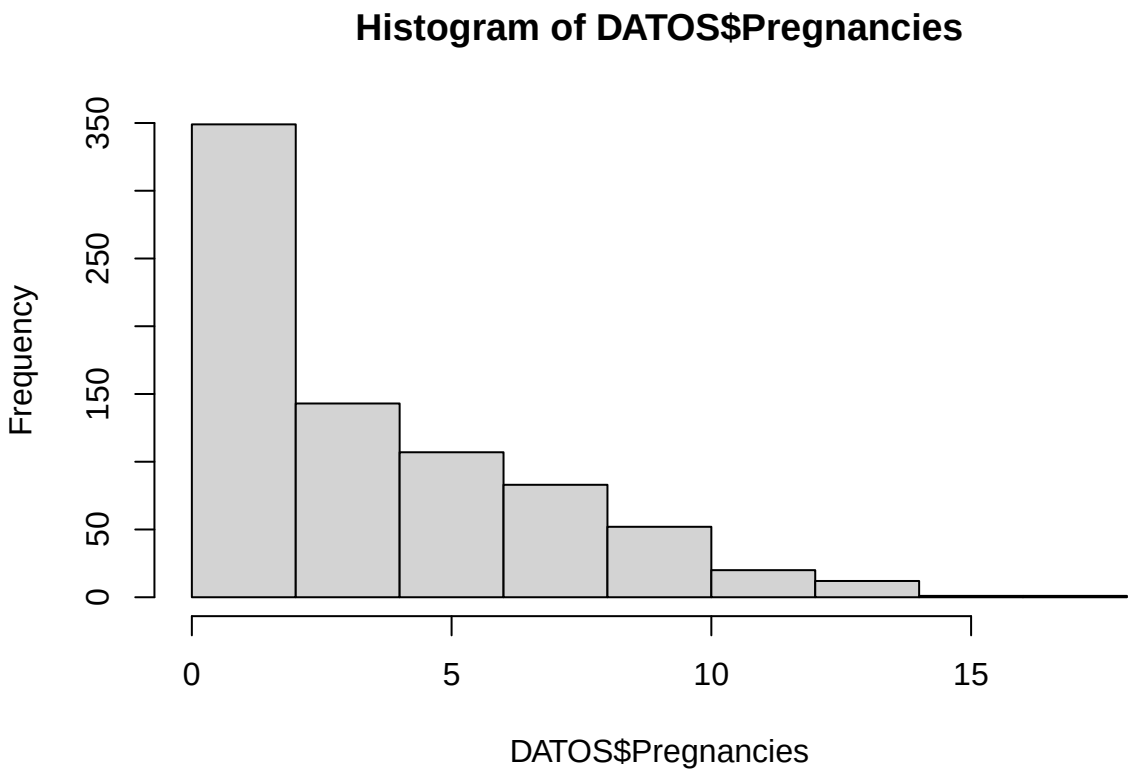
In order to be able to use MCD to treat outliers, we must do some preprocessing of our data. This is because we need our variables to behave as Gaussian variables (symmetric at least). To achieve this, we perform logarithmic transformations when required. Let´s plot histograms for each variable and take a look at each of their behaviour. It is important to check for outlyers because they may affect our interpretatin of the data.

After looking at the histograms, we can tell that variables: “age”, “pregnancies”, “skinthickness”, “insulin” need an appropiate transformation.

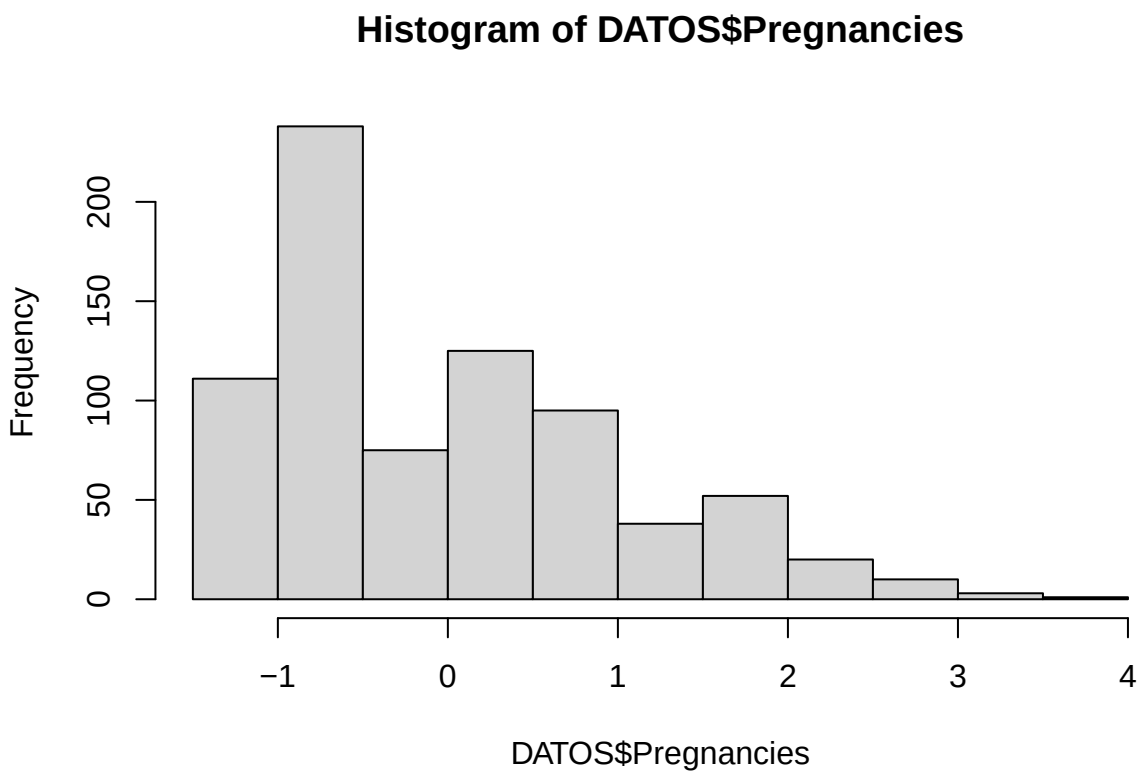
```
# "Ages" transformation
Ages= log(Age)
hist(Ages)
```



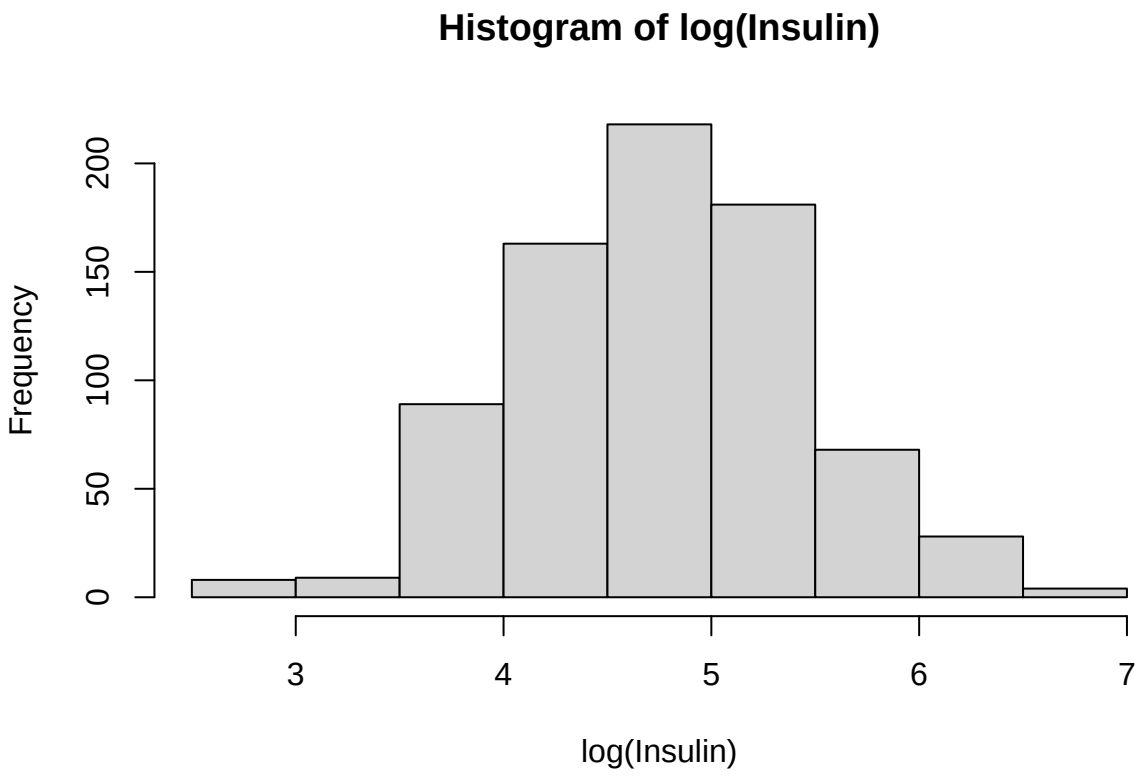

```
DATOS$Age <- Ages

# We standarize the variable Pregnancies
hist(DATOS$Pregnancies)
```




```
DATOS$Pregnancies=scale(Pregnancies, center = TRUE,scale= TRUE)
hist(DATOS$Pregnancies)
```




```r
# We normalize
DATOS$SkinThickness <- scale(log(DATOS$SkinThickness))

# We normalize
hist(log(Insulin))
```




```
DATOS$Insulin <- log(DATOS$Insulin)
attach(DATOS)
```

```
## The following objects are masked from DATOS (pos = 3):
##
##     Age, BloodPressure, BMI, DiabetesPedigreeFunction, Glucose,
##     Insulin, Outcome, Pregnancies, SkinThickness
```

Now that we finished with our transformations, we proceed on applying the MCD.

```
X=DATOS[1:8]
n <- nrow(X)
p <- ncol(X)
color_1 <- "deepskyblue2"
color_2 <- "seagreen2"
color_3 <- "orange2"
MCD_est <- CovMcd(X,alpha=0.75,nsamp="deterministic")
m_MCD <- MCD_est$center
```

Let´s now check on the mean vector, covariance and correlation matrix of the variables before taking the outliers out.

Let´s check first the mean vector, correlation and covariance matrix before taking out outliers.

```r
m <- colMeans(DATOS[,1:8])
m
```

```
##              Pregnancies                  Glucose            BloodPressure
##            -6.913596e-17             1.216458e+02             7.243229e+01
##            SkinThickness                  Insulin                      BMI
##             1.699126e-16             4.765564e+00             3.242279e+01
## DiabetesPedigreeFunction                      Age
##             4.718763e-01             3.448802e+00
```

```r
S <- cov(X)
S
```

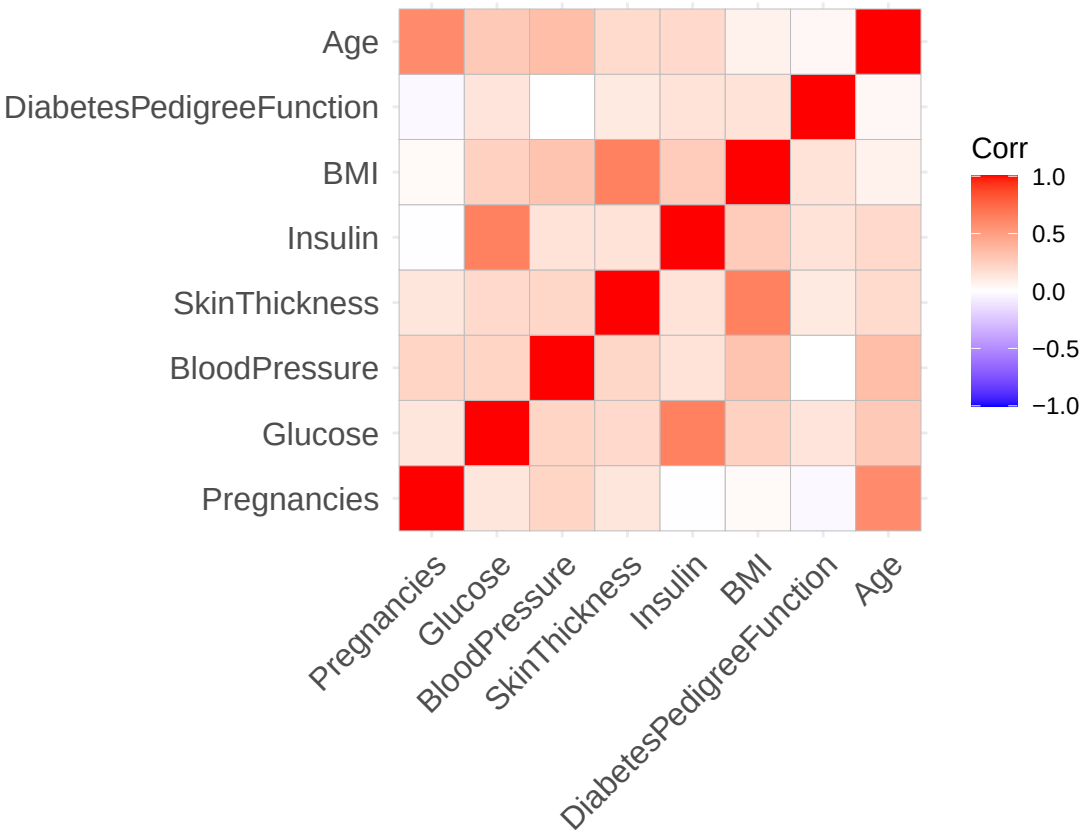
```
##                           Pregnancies    Glucose BloodPressure SkinThickness
## Pregnancies                1.000000000   3.989936    2.69395755    0.13042644
## Glucose                    3.989935512 930.393307   83.68915689    6.10217921
## BloodPressure              2.693957553  83.689157  153.08276293    2.59065454
## SkinThickness              0.130426445   6.102179    2.59065454    1.00000000
## Insulin                   -0.004661073  13.201037    1.23982996    0.10431841
## BMI                        0.203760279  50.291783   26.41243142    4.45037219
## DiabetesPedigreeFunction -0.011107020   1.412882   -0.01567527    0.03731064
## Age                        0.190233567   2.738729    1.35208478    0.06093346
##                              Insulin        BMI DiabetesPedigreeFunction
## Pregnancies               -0.004661073  0.2037603              -0.01110702
## Glucose                   13.201036753 50.2917835               1.41288182
## BloodPressure              1.239829959 26.4124314              -0.01567527
## SkinThickness              0.104318408  4.4503722               0.03731064
## Insulin                    0.464320193  1.2754979               0.03344816
## BMI                        1.275497899 48.2873158               0.33486827
## DiabetesPedigreeFunction   0.033448156  0.3348683               0.10977864
## Age                        0.044297102  0.1505962               0.00438710
##                                 Age
## Pregnancies              0.19023357
## Glucose                  2.73872923
## BloodPressure            1.35208478
## SkinThickness            0.06093346
## Insulin                  0.04429710
## BMI                      0.15059619
## DiabetesPedigreeFunction 0.00438710
## Age                      0.10413664
```

```r
R <- cor(X)
R
```

```
##                           Pregnancies   Glucose BloodPressure SkinThickness
## Pregnancies                1.000000000 0.1308075   0.217734678     0.1304264
## Glucose                    0.130807483 1.0000000   0.221754661     0.2000560
## BloodPressure              0.217734678 0.2217547   1.000000000     0.2093854
## SkinThickness              0.130426445 0.2000560   0.209385382     1.0000000
## Insulin                   -0.006840331 0.6351349   0.147058521     0.1530919
## BMI                        0.029322635 0.2372724   0.307205300     0.6404420
## DiabetesPedigreeFunction -0.033522673 0.1398021  -0.003823781     0.1126092
```

```
## Age                          0.589502114 0.2782364   0.338640575     0.1888226
##                                  Insulin         BMI DiabetesPedigreeFunction
## Pregnancies                -0.006840331 0.02932263              -0.033522673
## Glucose                     0.635134883 0.23727240               0.139802079
## BloodPressure               0.147058521 0.30720530              -0.003823781
## SkinThickness               0.153091883 0.64044199               0.112609166
## Insulin                     1.000000000 0.26937323               0.148150963
## BMI                         0.269373228 1.00000000               0.145444918
## DiabetesPedigreeFunction    0.148150963 0.14544492               1.000000000
## Age                         0.201448826 0.06715768               0.041031442
##                                      Age
## Pregnancies                 0.58950211
## Glucose                     0.27823645
## BloodPressure               0.33864057
## SkinThickness               0.18882263
## Insulin                     0.20144883
## BMI                         0.06715768
## DiabetesPedigreeFunction    0.04103144
## Age                         1.00000000
```

```
ggcorrplot(R)
```



Outliers Treatment

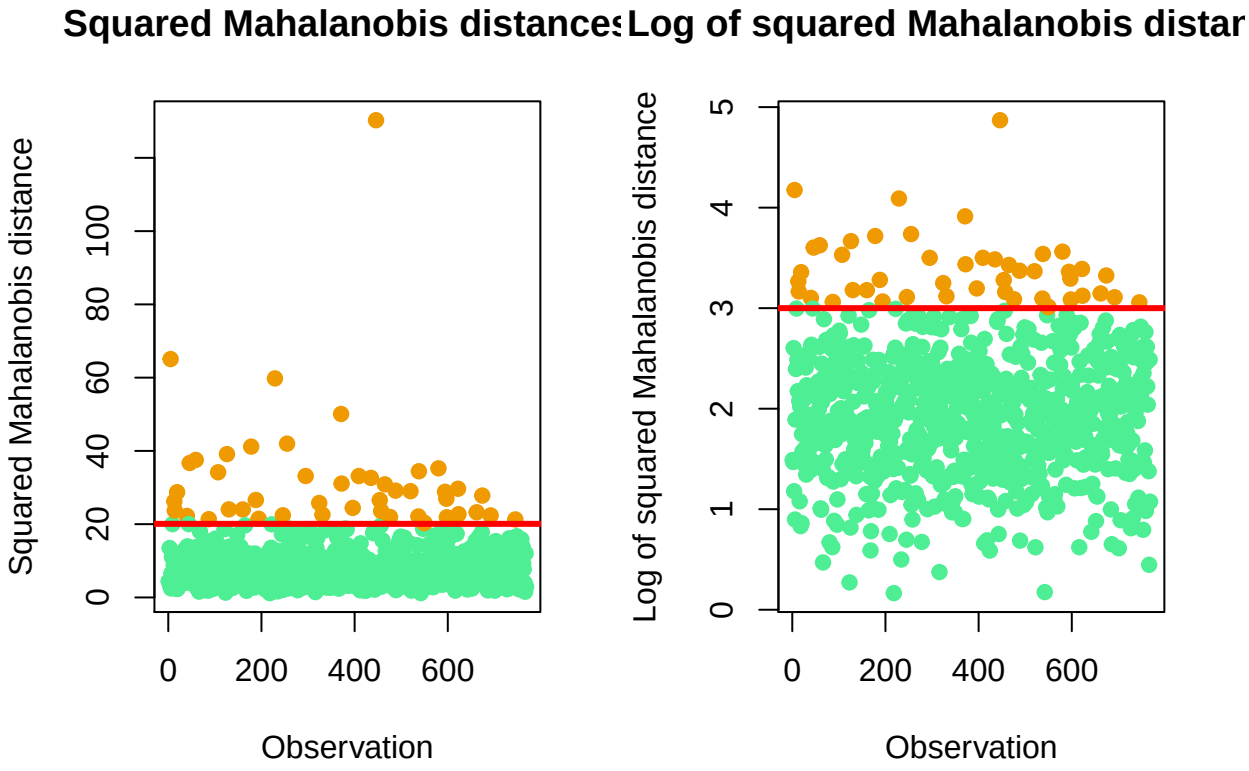
We will find the rows with observations that can be considered outliers.

```
#Let´s check on the outliers

X_sq_Mah_MCD <- MCD_est$mah

col_outliers_Mah_MCD <- rep(color_2,n)
outliers_Mah_MCD <- which(X_sq_Mah_MCD>qchisq(.99,p))
outliers_Mah_MCD
```

```
##  [1]   5  13  14  19  40  46  59  87 107 126 130 160 178 188 194 229 246 255 295
## [20] 324 331 371 372 396 409 435 446 454 457 465 476 488 520 537 538 549 550 580
## [39] 594 597 598 622 623 662 674 692 745
```

We can plot and check for these outliers aboev the Mahalanobis Distance.

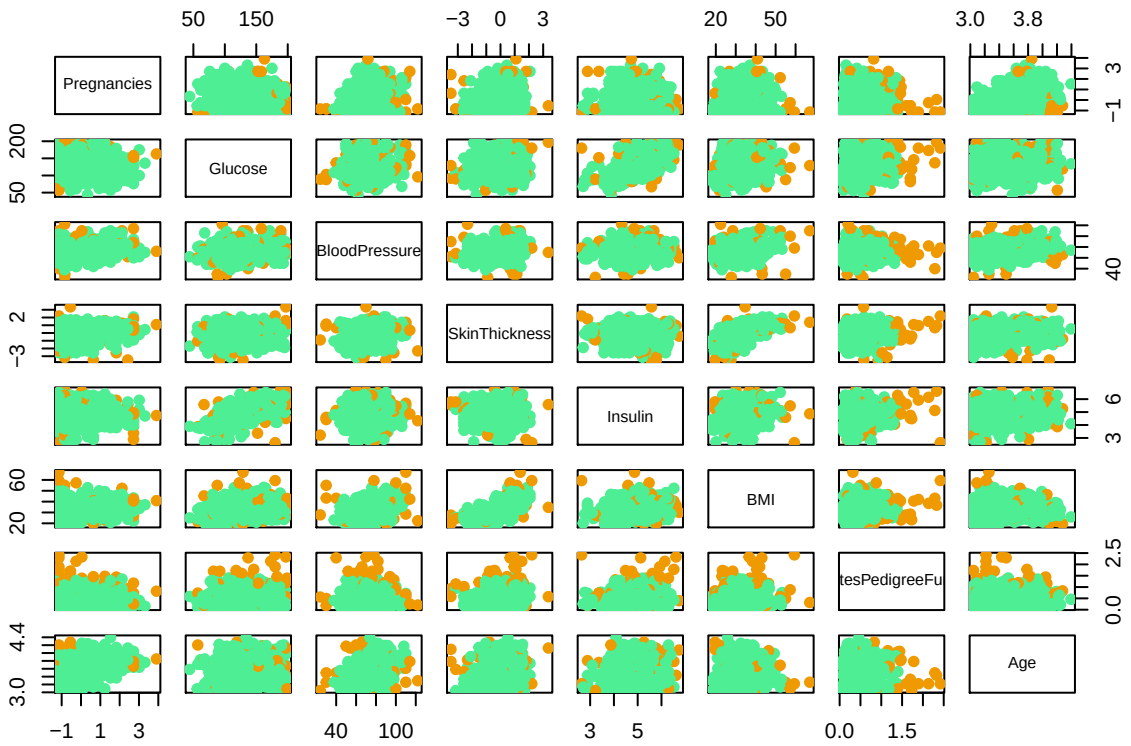
```
col_outliers_Mah_MCD[outliers_Mah_MCD] <- color_3
par(mfrow=c(1,2))
plot(1:n,X_sq_Mah_MCD,pch=19,col=col_outliers_Mah_MCD,main="Squared Mahalanobis distances",xlab="Observa
abline(h=qchisq(.99,p),lwd=3,col="red")
plot(1:n,log(X_sq_Mah_MCD),pch=19,col=col_outliers_Mah_MCD,main="Log of squared Mahalanobis distances",
abline(h=log(qchisq(.99,p)),lwd=3,col="red")
```



Now, we see the observations bellow and above the log of squared Mahalanobis distance. This is a very good way of observing our observations that fall very far away from the centroid of the data. We can also see the outliers in a scatterplot.

We can plot a Scatterplot showing the outliers.

```
pairs(X,pch=19,col=col_outliers_Mah_MCD)
```




```
attach(DATOS)
```

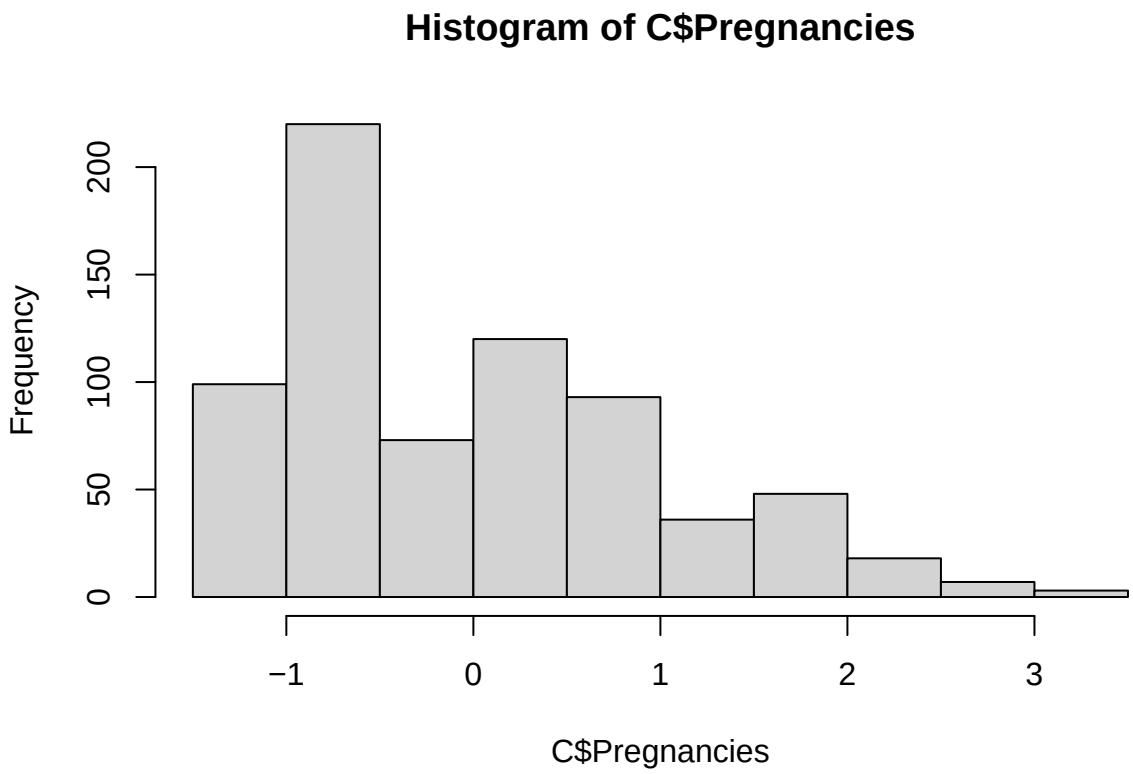
```
## The following objects are masked from DATOS (pos = 3):
##
##     Age, BloodPressure, BMI, DiabetesPedigreeFunction, Glucose,
##     Insulin, Outcome, Pregnancies, SkinThickness

## The following objects are masked from DATOS (pos = 4):
##
##     Age, BloodPressure, BMI, DiabetesPedigreeFunction, Glucose,
##     Insulin, Outcome, Pregnancies, SkinThickness
```

It is easy to see that there are some observations Now, we can substract these rows of observations, in order to be able to understand the mean, and correlations with a better view.

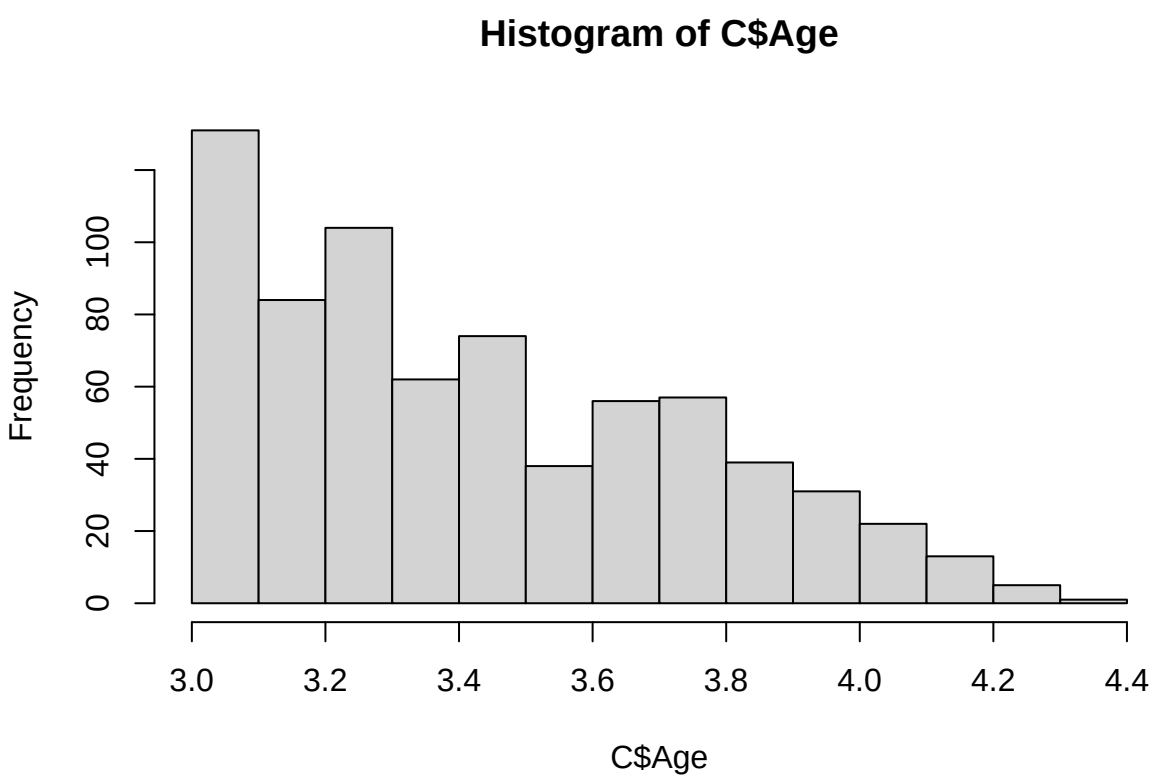
```
C=DATOS[-c(5  , 9  ,13  ,14  ,19  ,40,  44,  46,  59,  68 ,107 ,121 ,126 ,130 ,160 ,178 ,188,194,222, 2

DATOS <- C
```

Even taking out the outliers, these following variables don´t look so gaussian

```r
hist(C$Pregnancies)
```




```r
hist(C$Age)
```



As we have two subgroups inside of our data, people with diabetes and people not suffering from this dessease (and our objective is obviously related with understanding how this illness behaves among this specific population) we will perform this study for both subgroups and for the total, and then compare them.

```
cd <- C[ C$Outcome==1,]
cd <- cd[,1:8]
cnd <- C[ C$Outcome==0,]
cnd <- cnd[,1:8]
attach(DATOS)
```

```
## The following objects are masked from DATOS (pos = 3):
##
##     Age, BloodPressure, BMI, DiabetesPedigreeFunction, Glucose,
##     Insulin, Outcome, Pregnancies, SkinThickness

## The following objects are masked from DATOS (pos = 4):
##
##     Age, BloodPressure, BMI, DiabetesPedigreeFunction, Glucose,
##     Insulin, Outcome, Pregnancies, SkinThickness

## The following objects are masked from DATOS (pos = 5):
##
##     Age, BloodPressure, BMI, DiabetesPedigreeFunction, Glucose,
##     Insulin, Outcome, Pregnancies, SkinThickness
```

First, let´s study this three things for the entire group.

```
# For the sample totallity
# Mean Vector
m_MCD
```

```
##              Pregnancies                  Glucose            BloodPressure
##             -0.005590332            119.416789396             71.709867452
##            SkinThickness                  Insulin                      BMI
##             -0.012253262              4.733580845             32.089690722
## DiabetesPedigreeFunction                      Age
##              0.419329897              3.417426332
```

```
# Covariance Matrix
S_MCD <- MCD_est$cov
S_MCD
```

```
##                           Pregnancies     Glucose BloodPressure SkinThickness
## Pregnancies                1.04753579   4.7957950     2.7311955    0.18315426
## Glucose                    4.79579500 954.4345416    78.9263734    5.56466660
## BloodPressure              2.73119550  78.9263734   143.5315924    3.16402807
## SkinThickness              0.18315426   5.5646666     3.1640281    1.05348723
## Insulin                    0.01285872  14.6805583     1.4085547    0.15735836
## BMI                        0.38459710  54.2832845    26.1453644    4.69808345
## DiabetesPedigreeFunction -0.00551820   0.5271913    -0.1040866    0.03020673
## Age                        0.22769788   2.6526076     1.4282392    0.07517291
##                              Insulin        BMI DiabetesPedigreeFunction
## Pregnancies               0.01285872  0.3845971             -0.005518200
## Glucose                  14.68055827 54.2832845              0.527191295
## BloodPressure             1.40855469 26.1453644             -0.104086643
## SkinThickness             0.15735836  4.6980834              0.030206726
## Insulin                   0.48511105  1.5734775              0.021580831
## BMI                       1.57347747 47.8327042              0.213495744
## DiabetesPedigreeFunction  0.02158083  0.2134957              0.067929907
## Age                       0.04361491  0.2536963              0.001158804
##                                  Age
## Pregnancies               0.227697884
## Glucose                   2.652607552
## BloodPressure             1.428239181
## SkinThickness             0.075172908
## Insulin                   0.043614910
## BMI                       0.253696291
## DiabetesPedigreeFunction 0.001158804
## Age                       0.105919464
```

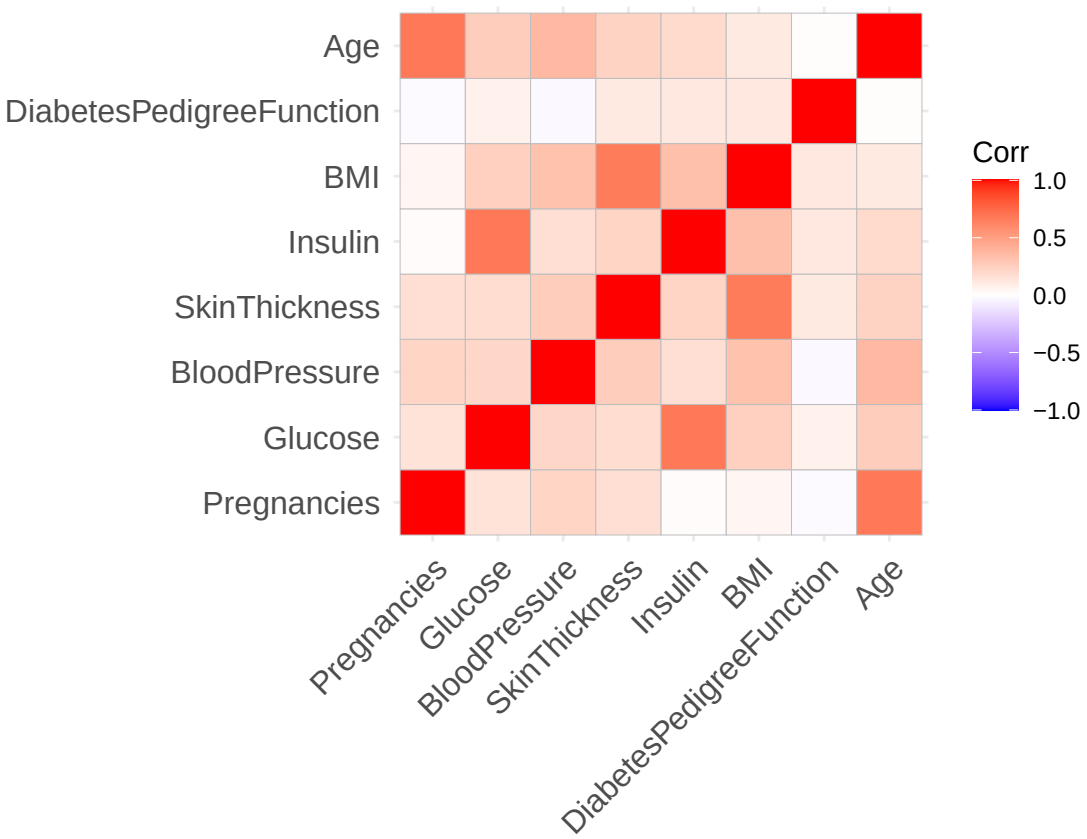
```
# Correlation Matrix
R_MCD <- cov2cor(S_MCD)
R_MCD
```

```
##                           Pregnancies    Glucose BloodPressure SkinThickness
## Pregnancies                1.00000000 0.15167119    0.22273815     0.1743484
## Glucose                    0.15167119 1.00000000    0.21324326     0.1754896
```

```
## BloodPressure             0.22273815 0.21324326    1.00000000     0.2573072
## SkinThickness             0.17434840 0.17548962    0.25730720     1.0000000
## Insulin                   0.01803818 0.68225844    0.16880270     0.2201175
## BMI                       0.05433243 0.25405666    0.31554295     0.6618257
## DiabetesPedigreeFunction -0.02068629 0.06547336   -0.03333425     0.1129168
## Age                       0.68357579 0.26382263    0.36630201     0.2250395
##                             Insulin        BMI DiabetesPedigreeFunction
## Pregnancies              0.01803818 0.05433243              -0.02068629
## Glucose                  0.68225844 0.25405666               0.06547336
## BloodPressure            0.16880270 0.31554295              -0.03333425
## SkinThickness            0.22011752 0.66182569               0.11291679
## Insulin                  1.00000000 0.32664610               0.11888226
## BMI                      0.32664610 1.00000000               0.11843941
## DiabetesPedigreeFunction 0.11888226 0.11843941               1.00000000
## Age                      0.19240950 0.11271034               0.01366129
##                                 Age
## Pregnancies              0.68357579
## Glucose                  0.26382263
## BloodPressure            0.36630201
## SkinThickness            0.22503954
## Insulin                  0.19240950
## BMI                      0.11271034
## DiabetesPedigreeFunction 0.01366129
## Age                      1.00000000
```

Let´s plot this correlation matrix.

```
ggcorrplot(R_MCD)
```



As we can see in this plot, there are some interesting linear relationships between some variables. First, let us say there is no negative linear relationship between any variable. We only have positive correlations, as the negative ones are very close to zero.

They are detailed as follows:

a) there is a positive linear relationship that may be significant between "age" and "pregnancy" meaning that the older people gets they have more pregnancies. This is very trivial: the chance of having had more kids obviosuly increases when the person is older, as pregnancies only increase with age, or stay the same (it is the total amount of pregnancies, a cummulative quantity).
b) Both "insulin" and "BMI" are positively correlated with "glucose". This does not surprise us either. People with higher body mass index have more chances of having high levels of sugar in blood. At the same time, people with high insulin levels after applying the medical test and check their insulin response, tend to have higher levels of glucose.
c) "BMI" and "SkinThickness" also appear to have a significant linear relationship between them. This may tell us thay people with higher body mass index tend to have higher values for skinthikness (and the other way around). This is also reasonable.
d) "Age" looks positively correlated to every other variable measured, with exception of the pedigree diabetes function. The more aged the person, the higher their values of insulin, blood preassure, skinthickness, glucose, BMI.

Let´s check what happens when we include the variable "Output" in our correlation matrix.

```
Y <- DATOS [,9]
Mat_cov <- cov(DATOS)
mat_cor <- cov2cor(Mat_cov)
mat_cor
```

```
##                           Pregnancies    Glucose BloodPressure SkinThickness
## Pregnancies                1.00000000 0.12907384    0.19837832     0.1740750
## Glucose                    0.12907384 1.00000000    0.20830481     0.1632576
## BloodPressure              0.19837832 0.20830481    1.00000000     0.2458406
## SkinThickness              0.17407500 0.16325760    0.24584061     1.0000000
## Insulin                   -0.00427589 0.65731793    0.14218046     0.1712963
## BMI                        0.04666571 0.23355573    0.32430862     0.6491759
## DiabetesPedigreeFunction -0.02957528 0.06053756   -0.02494636     0.1025394
## Age                        0.64844215 0.27169526    0.37032115     0.2233562
## Outcome                    0.22955726 0.49916106    0.17735597     0.2359524
##                              Insulin        BMI DiabetesPedigreeFunction
## Pregnancies               -0.00427589 0.04666571              -0.02957528
## Glucose                    0.65731793 0.23355573               0.06053756
## BloodPressure              0.14218046 0.32430862              -0.02494636
## SkinThickness              0.17129628 0.64917588               0.10253937
## Insulin                    1.00000000 0.29907430               0.09933125
## BMI                        0.29907430 1.00000000               0.13251217
## DiabetesPedigreeFunction   0.09933125 0.13251217               1.00000000
## Age                        0.17330701 0.10466408               0.01242707
## Outcome                    0.33550933 0.30508089               0.19687255
##                                 Age   Outcome
## Pregnancies               0.64844215 0.2295573
## Glucose                   0.27169526 0.4991611
## BloodPressure             0.37032115 0.1773560
## SkinThickness             0.22335619 0.2359524
## Insulin                   0.17330701 0.3355093
## BMI                       0.10466408 0.3050809
## DiabetesPedigreeFunction  0.01242707 0.1968726
## Age                       1.00000000 0.2852331
## Outcome                   0.28523306 1.0000000
```

As we can see, the only value that calls our attention is the correlation between "Outcome" and "Glucose", for being far away from 0. If the correlation between a categorical and quantitative variable is close to 1, it means that subjects with outcome = 1 (diabetics) have larger values than subjects with outcome = 0 (non-diabetic). This basically says that in general the diabetic subset of the sample will have a larger value of glucose in blood. Not a big surprise.

Now, first for the diabetic subgroup (outcome == 1).

```
# Mean Vector
apply(cd,2,mean)
```

```
##              Pregnancies                  Glucose            BloodPressure
##                0.3120991              140.9834711               74.9132231
##            SkinThickness                  Insulin                      BMI
##                0.3071301                5.0485373               34.9479339
## DiabetesPedigreeFunction                      Age
##                0.5068471                3.5570844
```

```
# Covariance Matrix
cov(cd)
```

```
##                           Pregnancies     Glucose BloodPressure SkinThickness
```

```
## Pregnancies               1.125017847  -1.5176226     1.4467140    0.03453153
## Glucose                  -1.517622598 834.0910120    25.3761531   -0.98718473
## BloodPressure             1.446713977  25.3761531   123.1833099    1.86581606
## SkinThickness             0.034531531  -0.9871847     1.8658161    0.58465226
## Insulin                  -0.048758427   9.4951769     0.2139431    0.04978652
## BMI                      -0.633815708  12.7721649    20.3352971    2.48104118
## DiabetesPedigreeFunction -0.001455324  -0.5997038    -0.1479345    0.01211704
## Age                       0.162868586   0.7225274     0.9925202   -0.00866680
##                              Insulin        BMI DiabetesPedigreeFunction
## Pregnancies              -0.048758427 -0.6338157             -0.001455324
## Glucose                   9.495176917 12.7721649             -0.599703782
## BloodPressure             0.213943145 20.3352971             -0.147934484
## SkinThickness             0.049786520  2.4810412              0.012117041
## Insulin                   0.341393111  0.8577588             -0.007681480
## BMI                       0.857758831 34.0191866              0.065205699
## DiabetesPedigreeFunction -0.007681480  0.0652057              0.090390553
## Age                       0.006864765 -0.1627534             -0.006476427
##                                  Age
## Pregnancies               0.162868586
## Glucose                   0.722527382
## BloodPressure             0.992520227
## SkinThickness            -0.008666800
## Insulin                   0.006864765
## BMI                      -0.162753351
## DiabetesPedigreeFunction -0.006476427
## Age                       0.079336288
```

```
l=cov(cd)
# Correlation Matrix
cov2cor(l)
```

```
##                           Pregnancies     Glucose BloodPressure SkinThickness
## Pregnancies               1.000000000 -0.04954244    0.12289297    0.04257820
## Glucose                  -0.049542436  1.00000000    0.07916681   -0.04470364
## BloodPressure             0.122892969  0.07916681    1.00000000    0.21985891
## SkinThickness             0.042578202 -0.04470364    0.21985891    1.00000000
## Insulin                  -0.078676061  0.56268970    0.03299097    0.11143858
## BMI                      -0.102452210  0.07582210    0.31413232    0.55631811
## DiabetesPedigreeFunction -0.004563714 -0.06906667   -0.04433349    0.05270918
## Age                       0.545157108  0.08882018    0.31748813   -0.04024149
##                              Insulin         BMI DiabetesPedigreeFunction
## Pregnancies              -0.07867606 -0.10245221             -0.004563714
## Glucose                   0.56268970  0.07582210             -0.069066674
## BloodPressure             0.03299097  0.31413232             -0.044333495
## SkinThickness             0.11143858  0.55631811              0.052709176
## Insulin                   1.00000000  0.25169574             -0.043727634
## BMI                       0.25169574  1.00000000              0.037184511
## DiabetesPedigreeFunction -0.04372763  0.03718451              1.000000000
## Age                       0.04171213 -0.09906769             -0.076478266
##                                  Age
## Pregnancies               0.54515711
## Glucose                   0.08882018
## BloodPressure             0.31748813
## SkinThickness            -0.04024149
```

```
## Insulin                   0.04171213
## BMI                      -0.09906769
## DiabetesPedigreeFunction -0.07647827
## Age                       1.00000000
```

Same procedure, for the non-diabetic subgroup (outcome == 0).

```
# Mean Vector
apply(cnd,2,mean)
```

```
##              Pregnancies                  Glucose            BloodPressure
##                -0.1601948              109.8589474               70.6484211
##            SkinThickness                  Insulin                      BMI
##                -0.1732587                4.5804548               30.7840000
## DiabetesPedigreeFunction                      Age
##                 0.3968842                3.3687355
```

```
# Covariance Matrix
cov(cnd)
```

```
##                           Pregnancies     Glucose BloodPressure SkinThickness
## Pregnancies                0.78425788   1.4002508     1.9027154    0.15232904
## Glucose                    1.40025084 563.2901888    47.8405996    2.45471631
## BloodPressure              1.90271538  47.8405996   126.7938530    2.42925917
## SkinThickness              0.15232904   2.4547163     2.4292592    1.02668710
## Insulin                   -0.05413379   9.5851293     0.8293490    0.06319990
## BMI                        0.10031430  16.8998481    19.6551224    4.16340267
## DiabetesPedigreeFunction -0.02832130  -0.1395775    -0.1967307    0.01541179
## Age                        0.18507006   1.4335645     1.2125437    0.07536762
##                              Insulin        BMI DiabetesPedigreeFunction
## Pregnancies               -0.05413379  0.1003143             -0.028321298
## Glucose                    9.58512929 16.8998481             -0.139577548
## BloodPressure              0.82934896 19.6551224             -0.196730668
## SkinThickness              0.06319990  4.1634027              0.015411791
## Insulin                    0.41065121  0.8308324              0.012676707
## BMI                        0.83083244 39.8460304              0.153657004
## DiabetesPedigreeFunction   0.01267671  0.1536570              0.055471799
## Age                        0.02069227  0.1365379             -0.002162076
##                                   Age
## Pregnancies                0.185070063
## Glucose                    1.433564541
## BloodPressure              1.212543747
## SkinThickness              0.075367617
## Insulin                    0.020692270
## BMI                        0.136537863
## DiabetesPedigreeFunction -0.002162076
## Age                        0.095145167
```

```
n=cov(cnd)
```

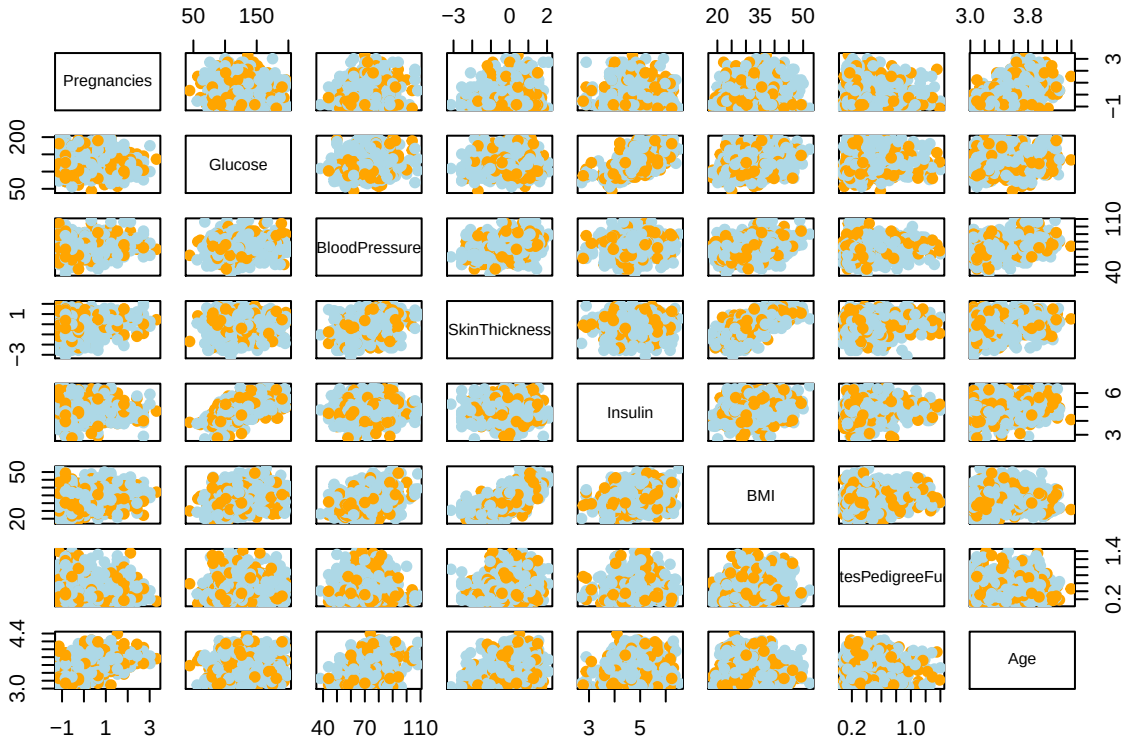
```r
# Correlation Matrix
cov2cor(n)
```

```
##                           Pregnancies     Glucose BloodPressure SkinThickness
## Pregnancies                1.00000000  0.06662087     0.1908074    0.16975955
## Glucose                    0.06662087  1.00000000     0.1790117    0.10207424
## BloodPressure              0.19080743  0.17901170     1.0000000    0.21291473
## SkinThickness              0.16975955  0.10207424     0.2129147    1.00000000
## Insulin                   -0.09538992  0.63022457     0.1149348    0.09733315
## BMI                        0.01794489  0.11280392     0.2765248    0.65093382
## DiabetesPedigreeFunction -0.13578368 -0.02496970    -0.0741800    0.06457998
## Age                        0.67750634  0.19582034     0.3491041    0.24114173
##                              Insulin        BMI DiabetesPedigreeFunction
## Pregnancies               -0.09538992 0.01794489              -0.13578368
## Glucose                    0.63022457 0.11280392              -0.02496970
## BloodPressure              0.11493483 0.27652482              -0.07418000
## SkinThickness              0.09733315 0.65093382               0.06457998
## Insulin                    1.00000000 0.20539240               0.08399120
## BMI                        0.20539240 1.00000000               0.10335315
## DiabetesPedigreeFunction   0.08399120 0.10335315               1.00000000
## Age                        0.10468349 0.07012408              -0.02976059
##                                  Age
## Pregnancies                0.67750634
## Glucose                    0.19582034
## BloodPressure              0.34910410
## SkinThickness              0.24114173
## Insulin                    0.10468349
## BMI                        0.07012408
## DiabetesPedigreeFunction -0.02976059
## Age                        1.00000000
```

Women with diabetes seem to have more pregnancies, have in average higher glucose, blood preassure, skin thickness, BMI. It also seems that there is a positive correlation between age and diabetes (older people tend to get it more) and the variance of the glucose is much larger in non-diabetic people.

Let´s plot a scatterplot without outliers

```r
d=C[,1:8]
pairs(d,pch=19,col=colors_Student)
```



#Principal Component Analysis

As the dimensionalities of the data grows, the feature space grws rapidly. We care about this for several reasons: first, we want to minimize computational cost. Also, when the dimensionality is big, the data starts getting more and more difficult to interpret, and it gets more difficult to detect underlying patterns.

When having so many features, we fail to see the "real" or "intrisic" dimensionality of the data. In other words, the dimensions could probably be reduced to a smaller set of dimensions.

```
dim(DATOS)
```

```
## [1] 717   9
```

```
head(DATOS)
```

```
##   Pregnancies Glucose BloodPressure SkinThickness  Insulin  BMI
## 1   0.6395305     148            72     0.6787277 4.941642 33.6
## 2  -0.8443348      85            66     0.1921970 4.330733 26.6
## 3   1.2330766     183            64    -0.9018226 6.086775 23.3
## 4  -0.8443348      89            66    -0.4075225 4.543295 28.1
## 6   0.3427574     116            74    -1.1895867 4.605170 25.6
## 7  -0.2507887      78            50     0.4468822 4.477337 31.0
##   DiabetesPedigreeFunction      Age Outcome
## 1                    0.627 3.912023       1
## 2                    0.351 3.433987       0
## 3                    0.672 3.465736       1
```

```
## 4                    0.167 3.044522       0
## 6                    0.201 3.401197       0
## 7                    0.248 3.258097       1
```

First I define a variable Y, where my categorical variable will be saved. This is due to the fact that we need quantitative variables to perform Principal Component Analysis. I already defined X_quan as the quantitative variables of my model.

Then, I will define de number or rows and columns as "n" and "p" respectively.

```
X <- Dataset.norm[1:8]

n <- nrow(X)
n
```

```
## [1] 768
```

```
p <- ncol(X)
p
```

```
## [1] 8
```

We have a matrix with dimension 768 x 8. The relationship between principal components and covariance matrix is fundamental to understand. First, the correlation matrix will help us observe if there are variables that are linearly related, and thus may be redundant when trying to explain or describe a sample.

As we can see, sometimes the size of the correlation matrix may blure our interpretations of linear correlations.

It is still hard to trace patterns, by only basing ourselves on the correlation structure of the data. As we are not endocrinologist, who have a lot of knowledge (not even close) of the domain we are studying, we will not go forward in taking a variable out of the study by simple choice. By doing this, we may be omitting something important in our data. This is why we will use PCA. With PCA, we will use a dimensionality reduction technique to remove the redundancy introduced by highly correlated variables.

What does PCA basically do? a) Removes noise from correlations b) changes de original coordinates, and stablishes new ones (principal components) c) Finally, helps us decide what is the real dimensionality of the data (how components are we finally considering)

Principal components are new variables that are built from linear relationships from the original variables. This new variables are uncorrelated to each other, and they try to compress the most information possible in the firsts components.

For this, we will use the previously determined normalized variables. They are also standarized, so that we won´t have any issues with scale differences.By only checking the values, we can say that the variables had also been standarized, so there is no scaling problem. This is fundamental to get a valuable insight of the true dimensionality of the data, as if there variables meassured in very different scales, one will overweight the other, hiding valuable information.

```
head(X)
```

```
##   Pregnancies   Glucose BloodPressure SkinThickness    Insulin       BMI
## 1  0.35294118 0.6709677     0.4897959     0.3043478 0.15144231 0.3149284
## 2  0.05882353 0.2645161     0.4285714     0.2391304 0.07451923 0.1717791
## 3  0.47058824 0.8967742     0.4081633     0.1304348 0.51201923 0.1042945
## 4  0.05882353 0.2903226     0.4285714     0.1739130 0.09615385 0.2024540
```

```
## 5  0.00000000 0.6000000     0.1632653     0.3043478 0.18509615 0.5092025
## 6  0.29411765 0.4645161     0.5102041     0.1086957 0.10336538 0.1513292
##   DiabetesPedigreeFunction       Age
## 1               0.23441503 0.4833333
## 2               0.11656704 0.1666667
## 3               0.25362938 0.1833333
## 4               0.03800171 0.0000000
## 5               0.94363792 0.2000000
## 6               0.05251921 0.1500000
```

We perform the PCA, and proceed to check the first 10 elements with summary.

```
pca_output <- PCA(X,ncp = 8, graph = FALSE)
summary(pca_output, nbelements = 10)
```

```
##
## Call:
## PCA(X = X, ncp = 8, graph = FALSE)
##
##
## Eigenvalues
##                        Dim.1   Dim.2   Dim.3   Dim.4   Dim.5   Dim.6   Dim.7
## Variance               2.462   1.502   1.245   0.926   0.734   0.443   0.393
## % of var.             30.773  18.769  15.561  11.579   9.180   5.544   4.912
## Cumulative % of var.  30.773  49.542  65.104  76.683  85.863  91.406  96.318
##                        Dim.8
## Variance               0.295
## % of var.              3.682
## Cumulative % of var. 100.000
##
## Individuals (the 10 first)
##                           Dist    Dim.1    ctr   cos2    Dim.2    ctr   cos2
## 1                      |  1.950 |  1.389  0.102  0.507 |  0.752  0.049  0.149
## 2                      |  1.928 | -1.622  0.139  0.708 | -0.056  0.000  0.001
## 3                      |  3.984 |  1.020  0.055  0.066 |  0.208  0.004  0.003
## 4                      |  2.238 | -2.046  0.221  0.836 | -0.351  0.011  0.025
## 5                      |  6.425 |  0.836  0.037  0.017 | -3.490  1.056  0.295
## 6                      |  1.842 | -1.214  0.078  0.435 |  1.030  0.092  0.313
## 7                      |  2.582 | -1.812  0.174  0.492 | -0.386  0.013  0.022
## 8                      |  2.818 |  0.558  0.016  0.039 |  0.919  0.073  0.106
## 9                      |  5.019 |  3.113  0.513  0.385 | -0.580  0.029  0.013
## 10                     |  4.316 |  3.440  0.626  0.635 |  0.920  0.073  0.045
##                           Dim.3    ctr   cos2
## 1                      |  0.194  0.004  0.010 |
## 2                      | -0.476  0.024  0.061 |
## 3                      |  3.515  1.292  0.778 |
## 4                      | -0.313  0.010  0.020 |
## 5                      |  0.893  0.083  0.019 |
## 6                      |  0.438  0.020  0.057 |
## 7                      | -0.785  0.064  0.092 |
## 8                      | -1.791  0.336  0.404 |
## 9                      |  2.536  0.673  0.255 |
## 10                     | -2.199  0.506  0.260 |
```

```
##
## Variables
##                              Dim.1    ctr   cos2    Dim.2    ctr   cos2    Dim.3
## Pregnancies                |  0.396  6.379  0.157 |  0.736 36.110  0.542 | -0.073
## Glucose                    |  0.677 18.614  0.458 | -0.116  0.894  0.013 |  0.527
## BloodPressure              |  0.559 12.680  0.312 |  0.245  4.007  0.060 | -0.239
## SkinThickness              |  0.639 16.586  0.408 | -0.276  5.089  0.076 | -0.528
## Insulin                    |  0.575 13.421  0.330 | -0.304  6.161  0.093 |  0.600
## BMI                        |  0.651 17.233  0.424 | -0.418 11.662  0.175 | -0.472
## DiabetesPedigreeFunction |  0.258  2.698  0.066 | -0.339  7.641  0.115 |  0.174
## Age                        |  0.552 12.389  0.305 |  0.653 28.437  0.427 |  0.110
##                              ctr   cos2
## Pregnancies                0.427  0.005 |
## Glucose                   22.329  0.278 |
## BloodPressure              4.596  0.057 |
## SkinThickness             22.387  0.279 |
## Insulin                   28.946  0.360 |
## BMI                       17.907  0.223 |
## DiabetesPedigreeFunction  2.445  0.030 |
## Age                        0.964  0.012 |
```

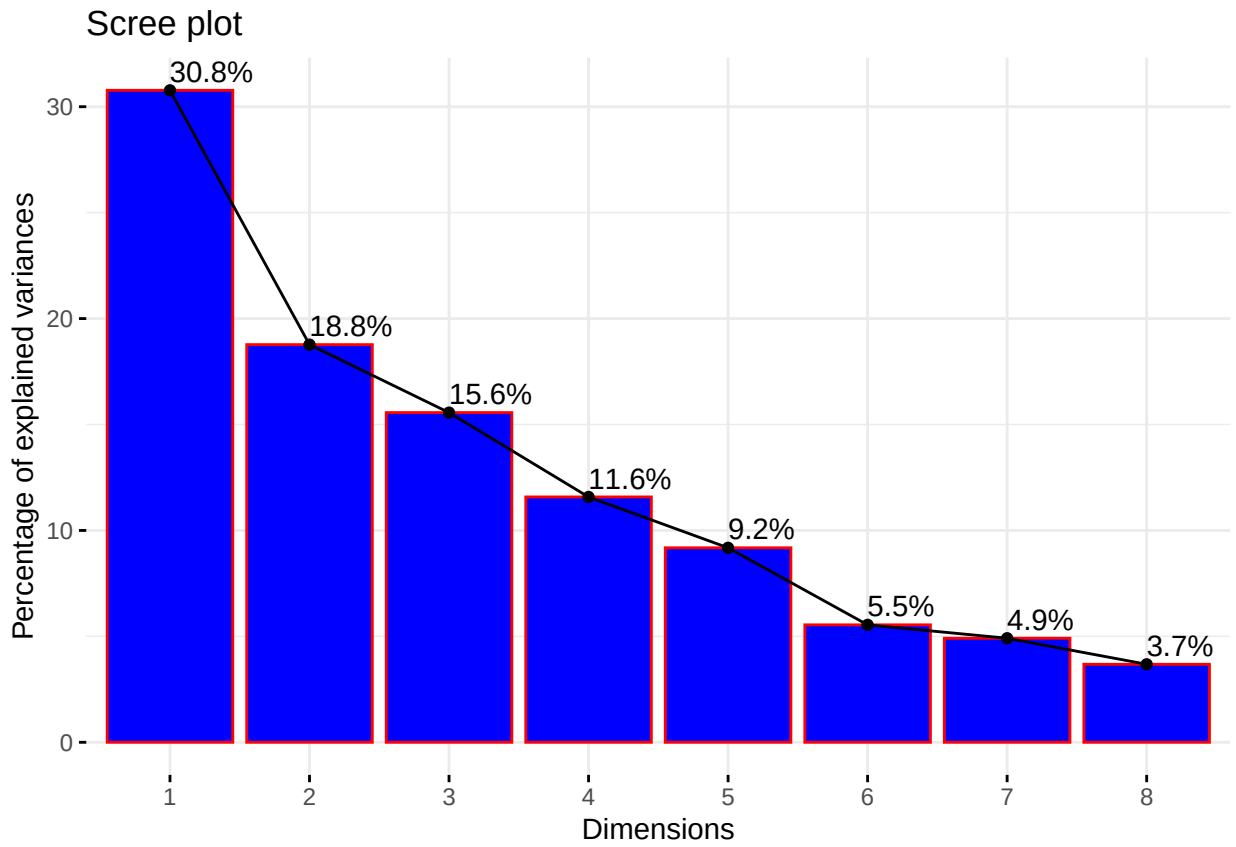
We have as many eigenvectors and eigenvalues as variables in our numerical matrix. But the main idea of this analysis, is to be able to compress the information in less variables. It would not be of any use to take the 8 principal components (as there is no direct interpretation). In other words, if we would stick with eight variables, we would just keep on working with the original ones.

The principal components do have a very interesting geometrical interpretation. This is because the principal components represent the direction of the data that explain a maximum amount of variance. So they act like new axes, that provide the best "angle" to analyze the data.

Let´s try and decide how many principal components we will take.

a) Method Nº1 The screeplot is a barplot of the proportion of explained variance of each dimension.

```
fviz_screeplot(pca_output,ncp=8,addlabels=T,barfill="blue",barcolor="red")
```



According to this rule, we could make use of the first 5 principal components (or only with the first two). Checking the Kaisser-Gutman Rule, we can check which PC has eigenvalue larger than 1.

```
pca_output$eig
```

```
##        eigenvalue percentage of variance cumulative percentage of variance
## comp 1  2.4618132              30.772665                           30.77266
## comp 2  1.5015547              18.769434                           49.54210
## comp 3  1.2449144              15.561430                           65.10353
## comp 4  0.9263599              11.579499                           76.68303
## comp 5  0.7343604               9.179505                           85.86253
## comp 6  0.4434926               5.543658                           91.40619
## comp 7  0.3929461               4.911826                           96.31802
## comp 8  0.2945587               3.681983                          100.00000
```

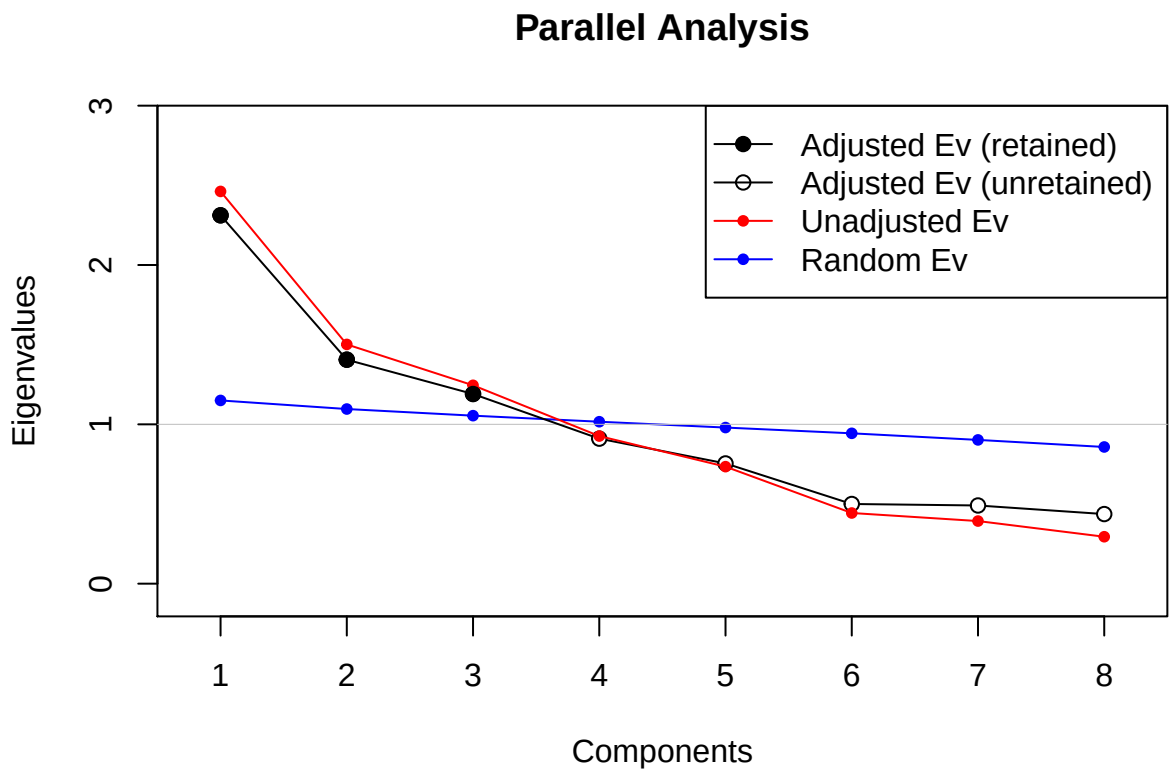
According to this test, we should stick to the first three principal components, and maybe the fourth (as its eigenvalue is very close to 1). This is because components with eigenvalue less than 1, explain less of the total variability that an original variable does, on average. Therefore, by choosing principal components with eigenvalues greater than one, we maintain those that express more of the variability than each of the original variables.

We can also try with Parallel Analysis technique, using the package "paran".

```
pca_output_ret <- paran(X,seed=1,graph = TRUE)
```

```
##
```

```
## Using eigendecomposition of correlation matrix.
## Computing: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
##
##
## Results of Horn's Parallel Analysis for component retention
## 240 iterations, using the mean estimate
##
## ----------------------------------------------------
## Component   Adjusted    Unadjusted    Estimated
##             Eigenvalue  Eigenvalue    Bias
## ----------------------------------------------------
## 1           2.311858    2.461813      0.149955
## 2           1.405604    1.501554      0.095950
## 3           1.190495    1.244914      0.054419
## ----------------------------------------------------
##
## Adjusted eigenvalues > 1 indicate dimensions to retain.
## (3 components retained)
```




```
pca_output_ret$Retained
```

```
## [1] 3
```

In this case, the papallel Analysis tells us that we should stay with the first three principal components, as they have an adjusted eigenvalue larger than 1. But first, we can take a look at the contribution of each variable in the construction of each principal component.

```
pca_output$var$contrib
```

```
##                              Dim.1      Dim.2      Dim.3      Dim.4      Dim.5
## Pregnancies                6.379122 36.1103200  0.4270885  4.7602848  8.5949300
## Glucose                   18.613909  0.8936113 22.3285412  3.8456830  0.5091892
## BloodPressure             12.680385  4.0067036  4.5960312  6.5532694 67.4515797
## SkinThickness             16.585987  5.0893888 22.3872452  0.3595099 11.9715631
## Insulin                   13.420573  6.1607185 28.9458226  3.8322510  0.8509387
## BMI                       17.232983 11.6615801 17.9065132  0.5934146  0.7436612
## DiabetesPedigreeFunction  2.697702  7.6408703  2.4445389 77.5911824  9.2826136
## Age                       12.389339 28.4368074  0.9642192  2.4644049  0.5955246
##                               Dim.6        Dim.7       Dim.8
## Pregnancies               37.9963055 2.861062e+00  2.87088642
## Glucose                    5.5132086 4.502260e+01  3.27326206
## BloodPressure              0.3548922 3.353176e-04  4.35680391
## SkinThickness              6.8728477 6.743809e+00 29.98964946
## Insulin                    0.5807708 3.691103e+01  9.29789290
## BMI                        3.9314632 8.286841e+00 39.64354294
## DiabetesPedigreeFunction   0.2914560 3.785697e-02  0.01378026
## Age                       44.4590559 1.364669e-01 10.55418205
```

Get the variance of the 3 first dimensions

```
pca_output$eig[,2][1:3]
```

```
##   comp 1   comp 2   comp 3
## 30.77266 18.76943 15.56143
```

Get the cummulative variance of the 3 first dimensions

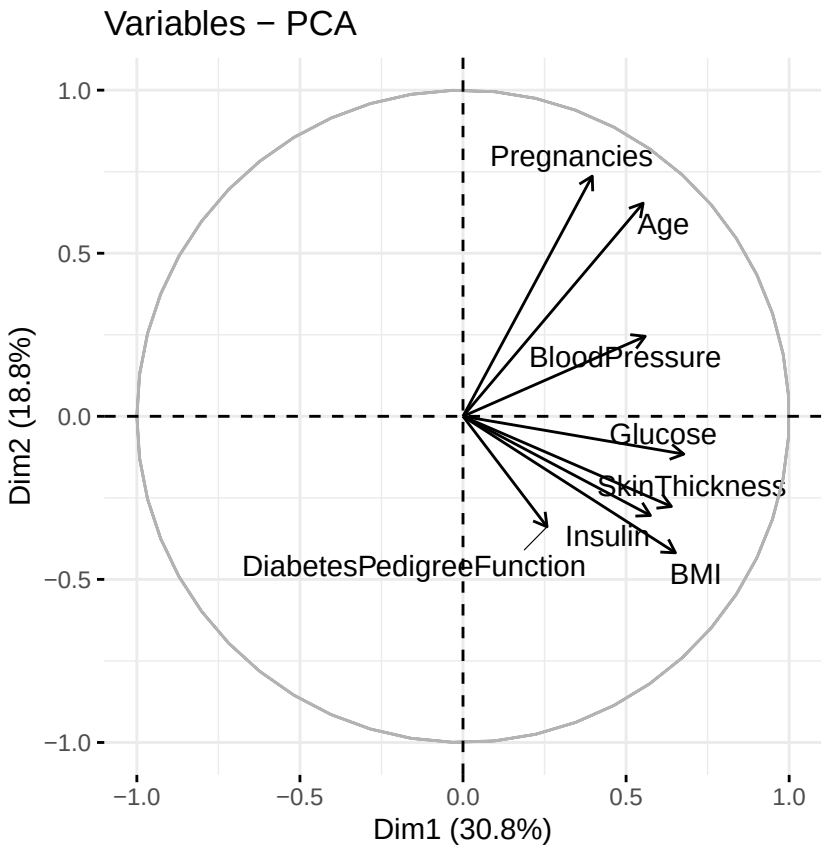
```
pca_output$eig[,3][1:3]
```

```
##   comp 1   comp 2   comp 3
## 30.77266 49.54210 65.10353
```

We can say that sticking with only 3 dimensions (reducing our dimensionality by more than 50%) we could explain more than 65% of the information from our dataset.

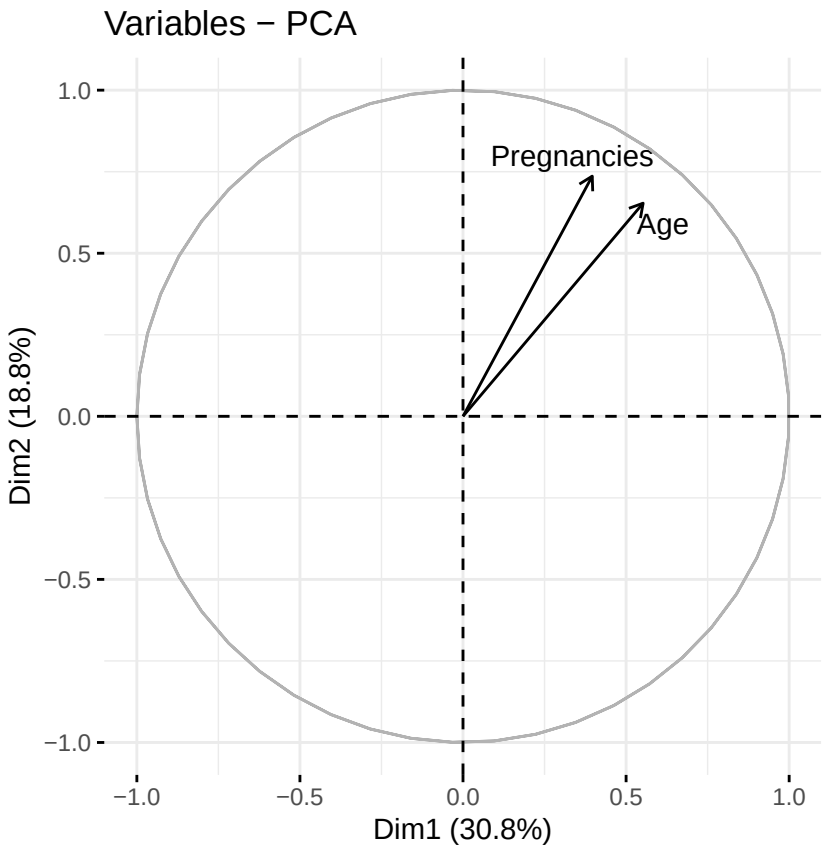
Interpreting the contribution of each variable to the principal component can be a bit unclear by only checking the numbers. Nevertheless, we can use some graphical images to have a better understanding.

```
fviz_pca_var(pca_output,col.bar="contrib",gradient.cols=c("#bb2e00","#002bbb"),repel=TRUE)
```



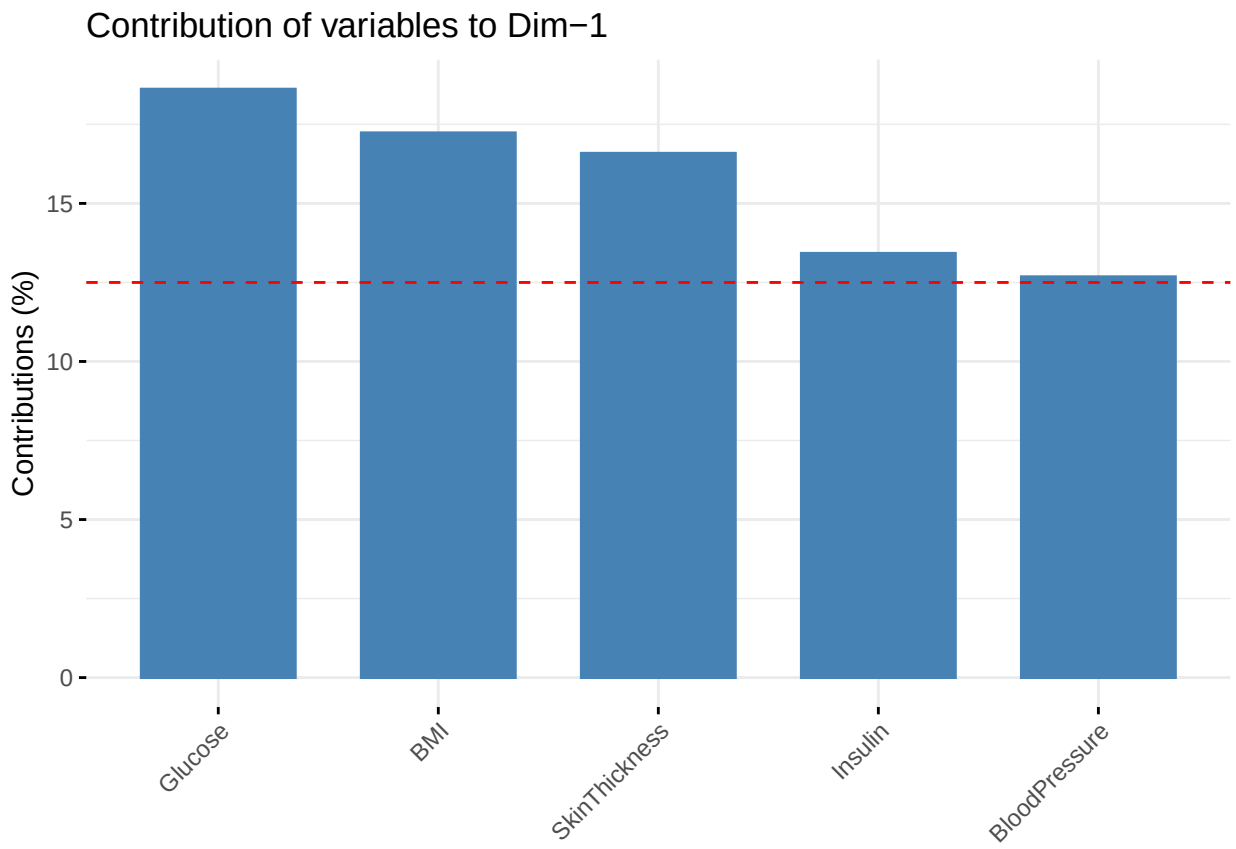
Creating a factor Map for variables with COS2 higher than 0.7. COS2 is also a useful measure that expresses

```
fviz_pca_var(pca_output,select.var=list(cos2=0.6),repel=TRUE)
```

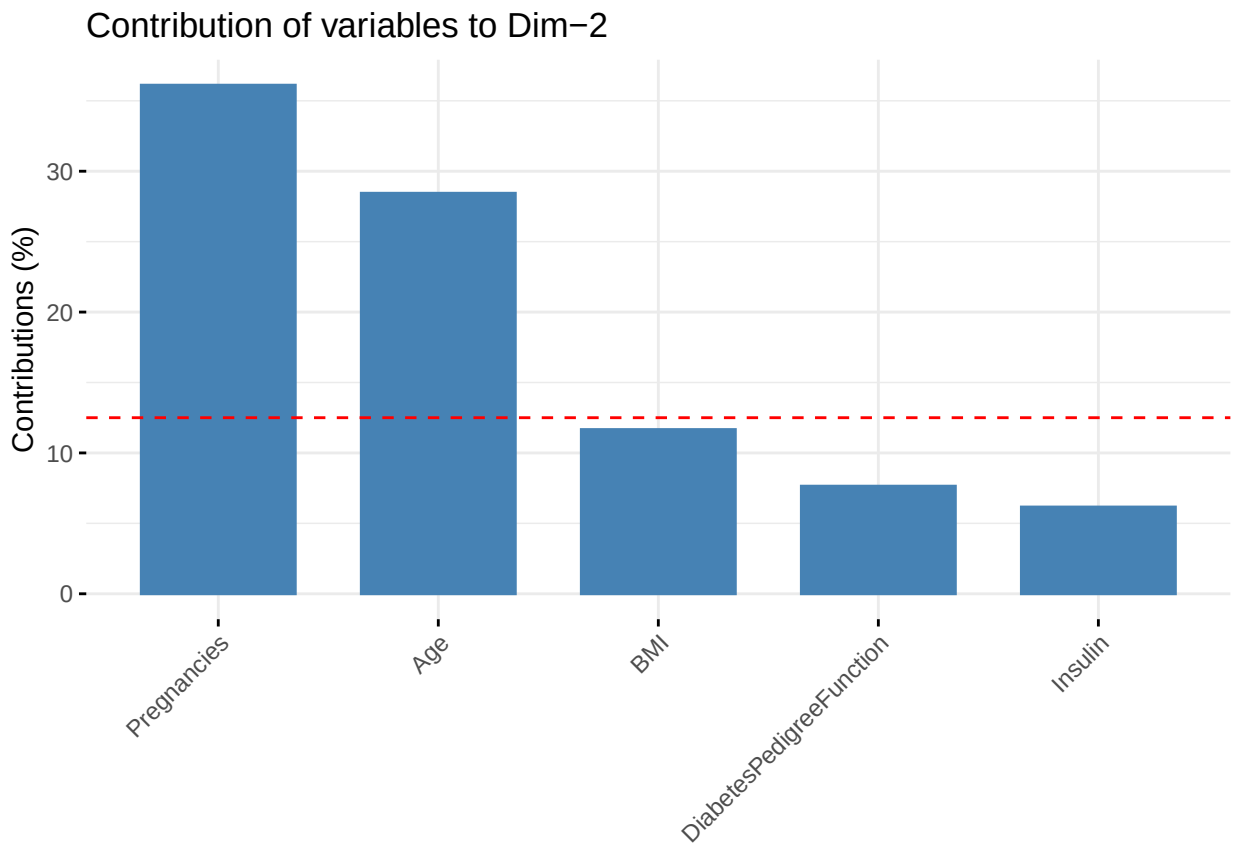


Checking the contribution of the most influential 5 variables on the first three principal components.

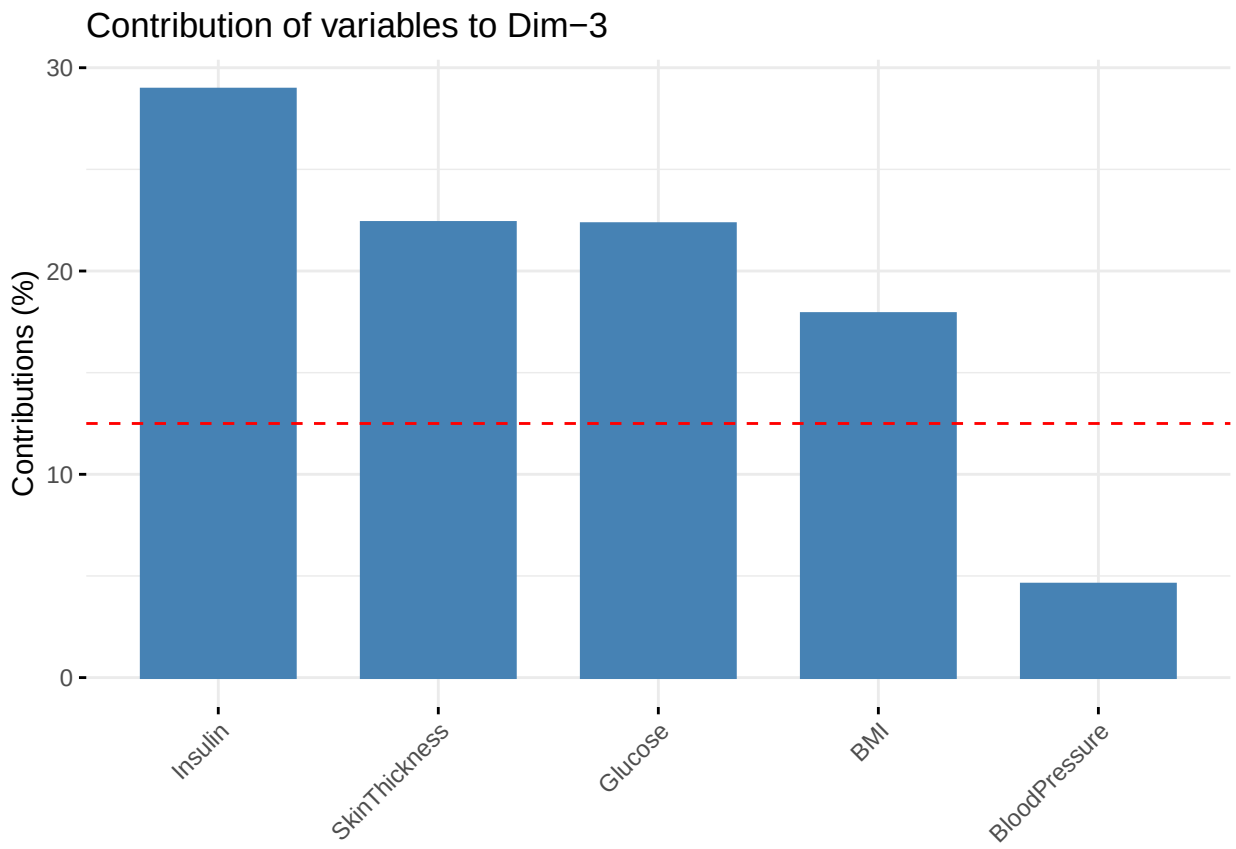
```
fviz_contrib(pca_output, choice="var", axes=1,top=5)
```




```
fviz_contrib(pca_output, choice="var", axes=2,top=5)
```

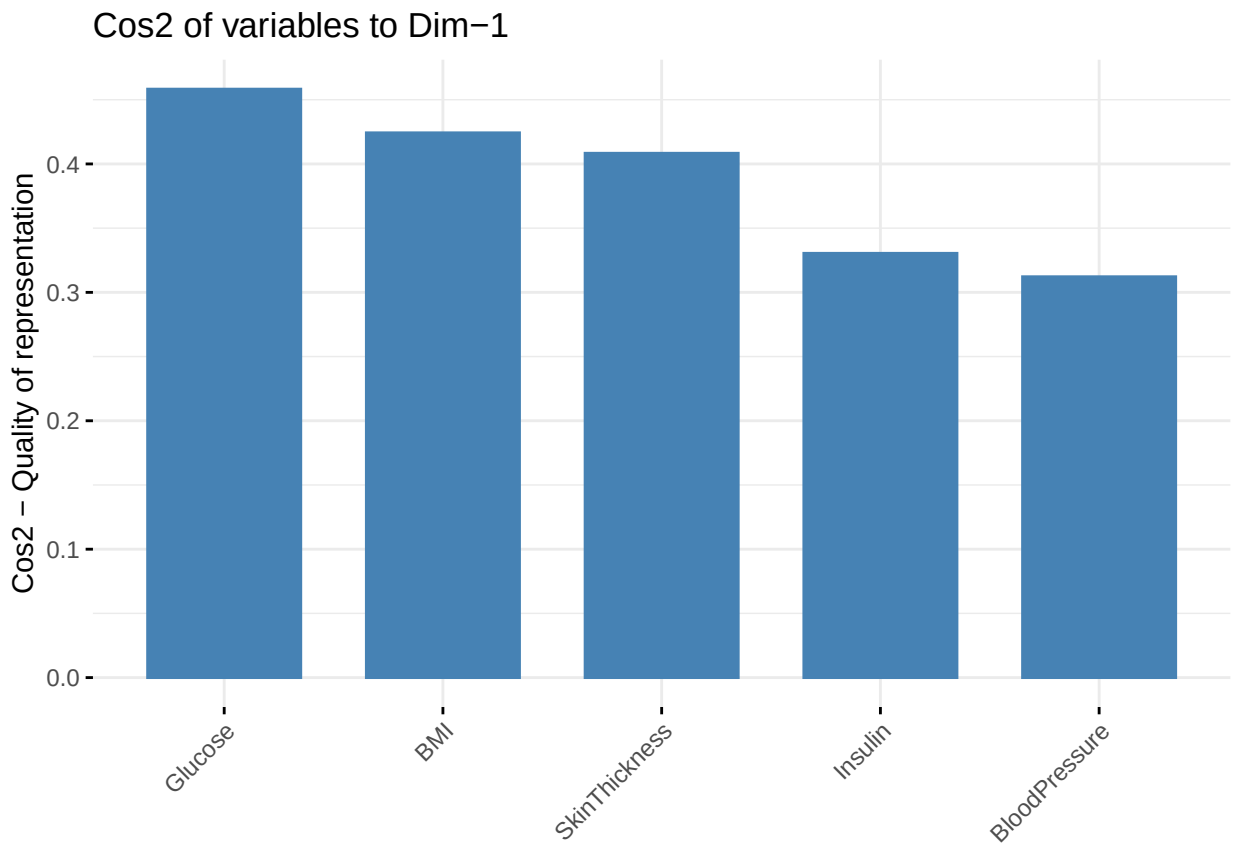



```
fviz_contrib(pca_output, choice="var", axes=3,top=5)
```

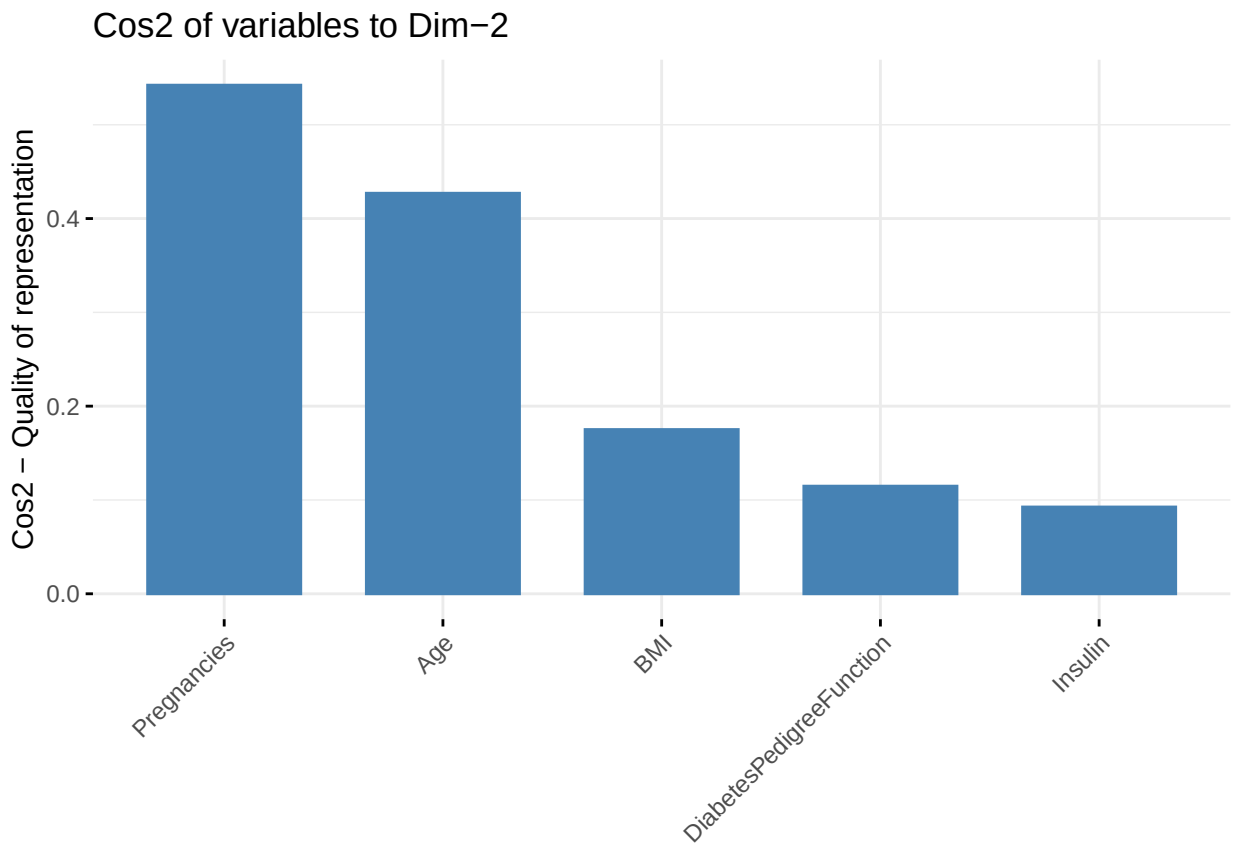


Obtaining a barplot for the variables with highest COS2 in the 1st, 2nd and 3rd principal component. The squared cosine shows the importance of a component for a given observation.

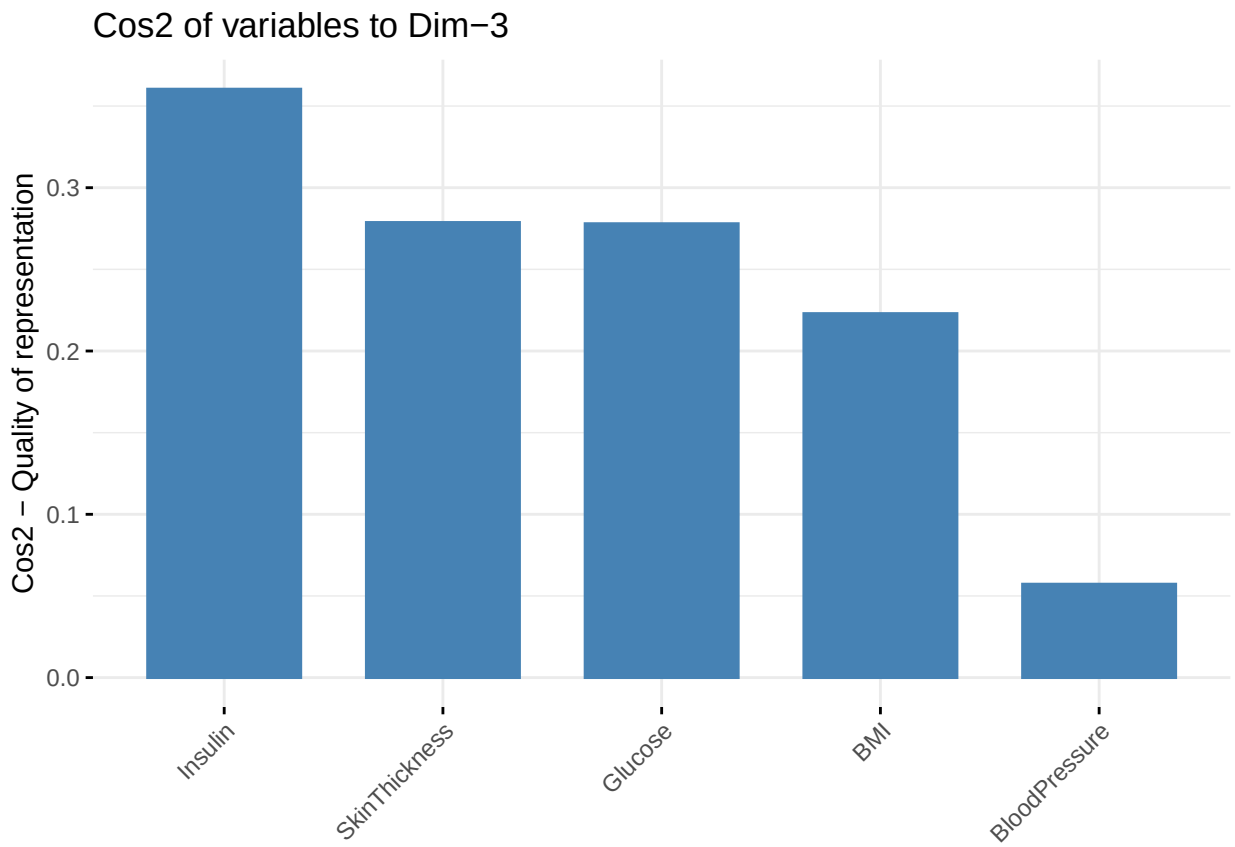
```
fviz_cos2(pca_output, choice = "var", axes = 1, top = 5)
```




```
fviz_cos2(pca_output, choice = "var", axes = 2, top = 5)
```

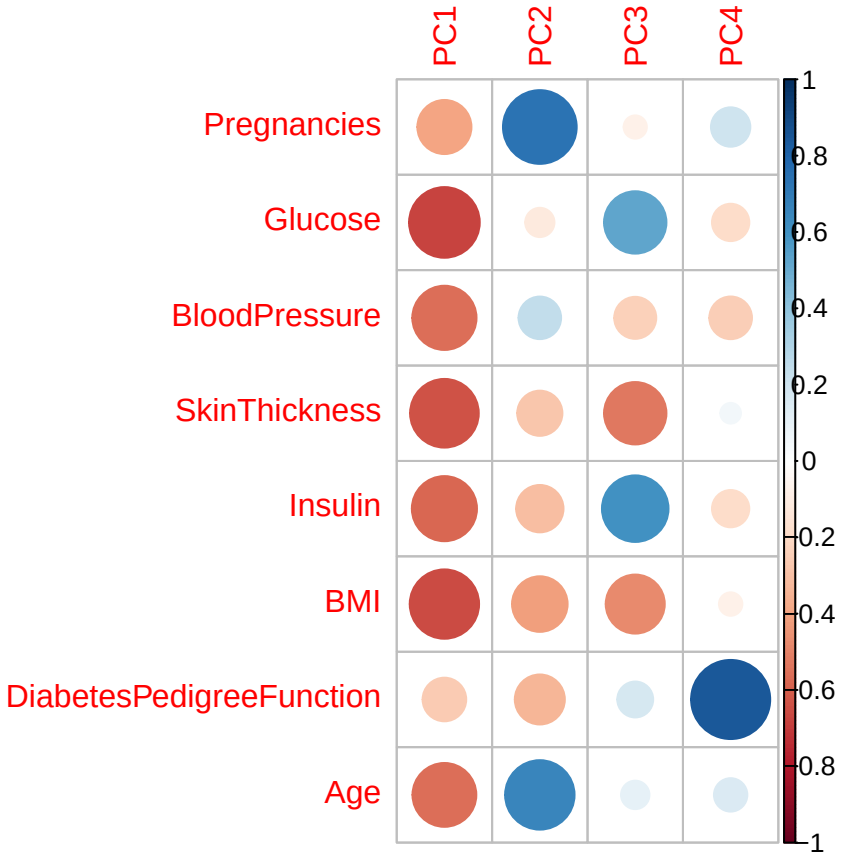



```
fviz_cos2(pca_output, choice = "var", axes = 3, top = 5)
```



We can also plot the correlation matrix of the components with the original variables. This is important, because it can show us the important variables of the original data in terms of variability.

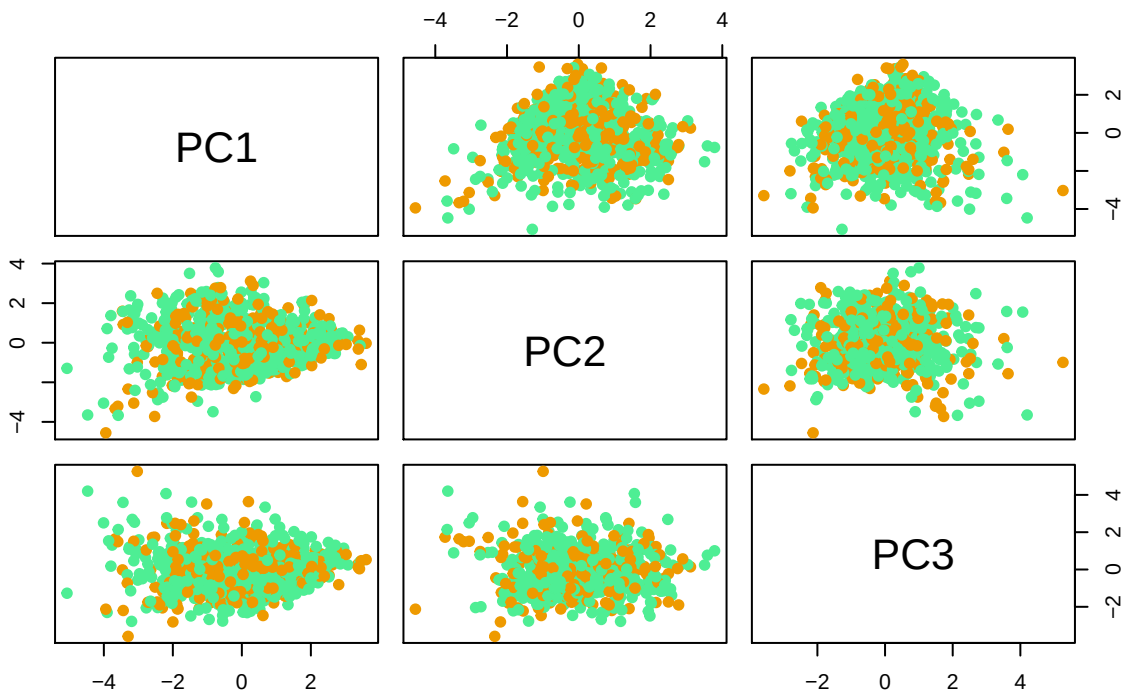
```
X_pcs <- prcomp(X,scale=TRUE)
corrplot(cor(X,X_pcs$x[,1:4]),is.corr=T)
```



By checking the principal component analysis, we can´t say that any of them really help us to understand the classification of observations between diabetic and non-diabetic. Nevertheless, it can give us a hint to what variables we can get. As far as each component is considered: a) Given that the variables that contribuite the most to the first principal components are all related to health, we can say that it makes a ranking in which it ranks how healthy a person is (each observation). As it is a combination of the glucose level, insuline, BMI and Skin Thickness. b) The second principal component seems to rank observations according to some descriptive characteristics related to life cycle or maybe fertility (specially number of pregnancies, and age). It is reasonable that both variables contribute to this component as they are linearly correlated, as seen before.

We can observe the correlation between a component and a variable (thus, the information they share). This is called "loading".

```
# Check if the first two components give me any sign of subgroups.
colors_X <- c(color_2,color_3)[1*(Y=="1")+1]
par(mfrow=c(1,1))
pairs(X_pcs$x[,1:3],col=colors_X,pch=19,main="The first three PCs")
```

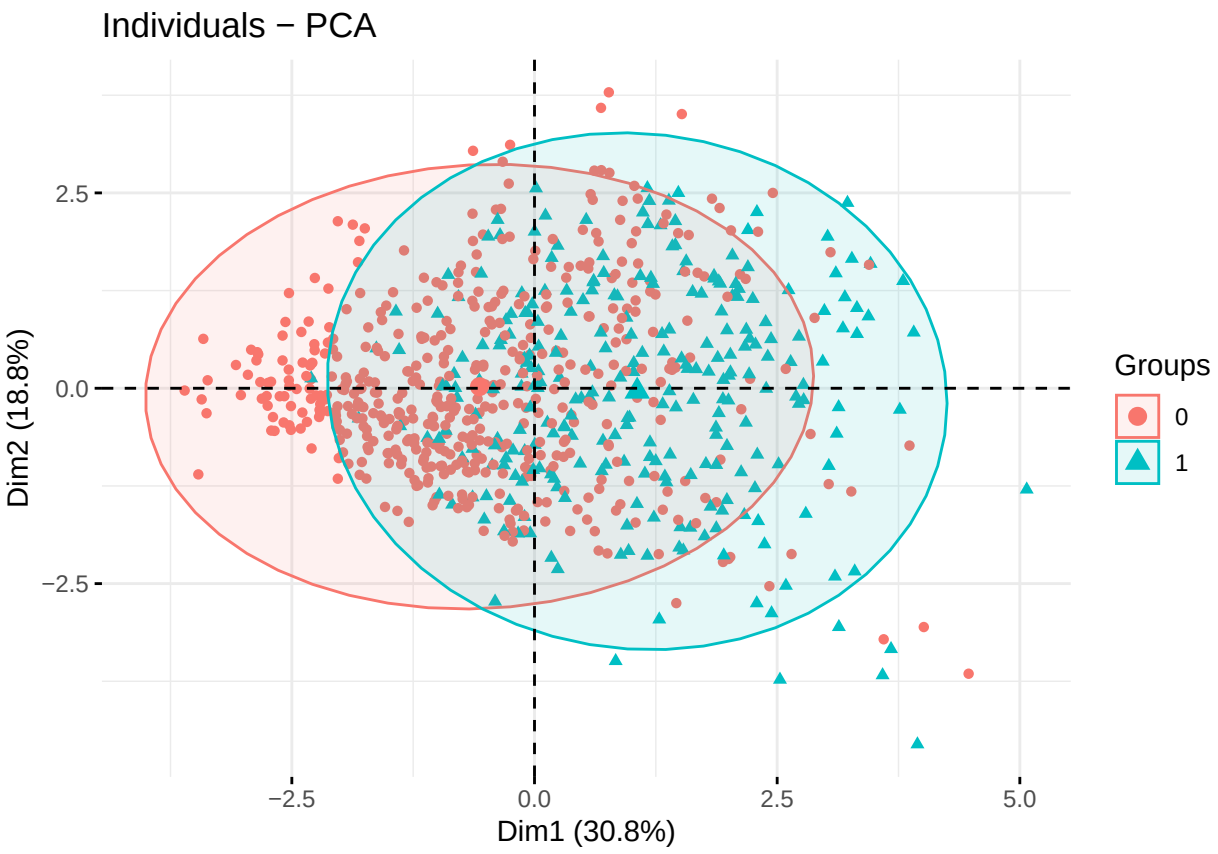


It seems that the principal components don´t help us a lot in dividing the observations between diabetic and non-diabetic.

```
summary(dat$Outcome)
```

```
##   0   1
## 500 268
```

```
fviz_pca_ind(pca_output,label="var",habillage=dat$Outcome,addEllipses=TRUE)
```



We get reassured: the first two principal components don´t seem to give us enough information to determine the subgroups: diabetic and non-diabetic. But we can still see the first principal component may be an indicator of illness (but both groups share a big surface).

Now, let´s perform the subgroup analysis.

PCA for both subgroups

First, I divide the sample in both subsets and normalize (and standarize) all the variables inside of each subset.

```
## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 1 has 717 rows to replace 268 rows

## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 2 has 717 rows to replace 268 rows

## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 3 has 717 rows to replace 268 rows

## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 4 has 717 rows to replace 268 rows

## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 5 has 717 rows to replace 268 rows
```

```
## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 6 has 717 rows to replace 268 rows

## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 7 has 717 rows to replace 268 rows

## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 8 has 717 rows to replace 268 rows

## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 1 has 717 rows to replace 500 rows

## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 2 has 717 rows to replace 500 rows

## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 3 has 717 rows to replace 500 rows

## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 4 has 717 rows to replace 500 rows

## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 5 has 717 rows to replace 500 rows

## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 6 has 717 rows to replace 500 rows

## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 7 has 717 rows to replace 500 rows

## Warning in `[<-.data.frame`(`*tmp*`, 1:8, value = structure(list(Pregnancies =
## c(0.4, : replacement element 8 has 717 rows to replace 500 rows
```

PCA for diabetic subgroup (outcome == 1)

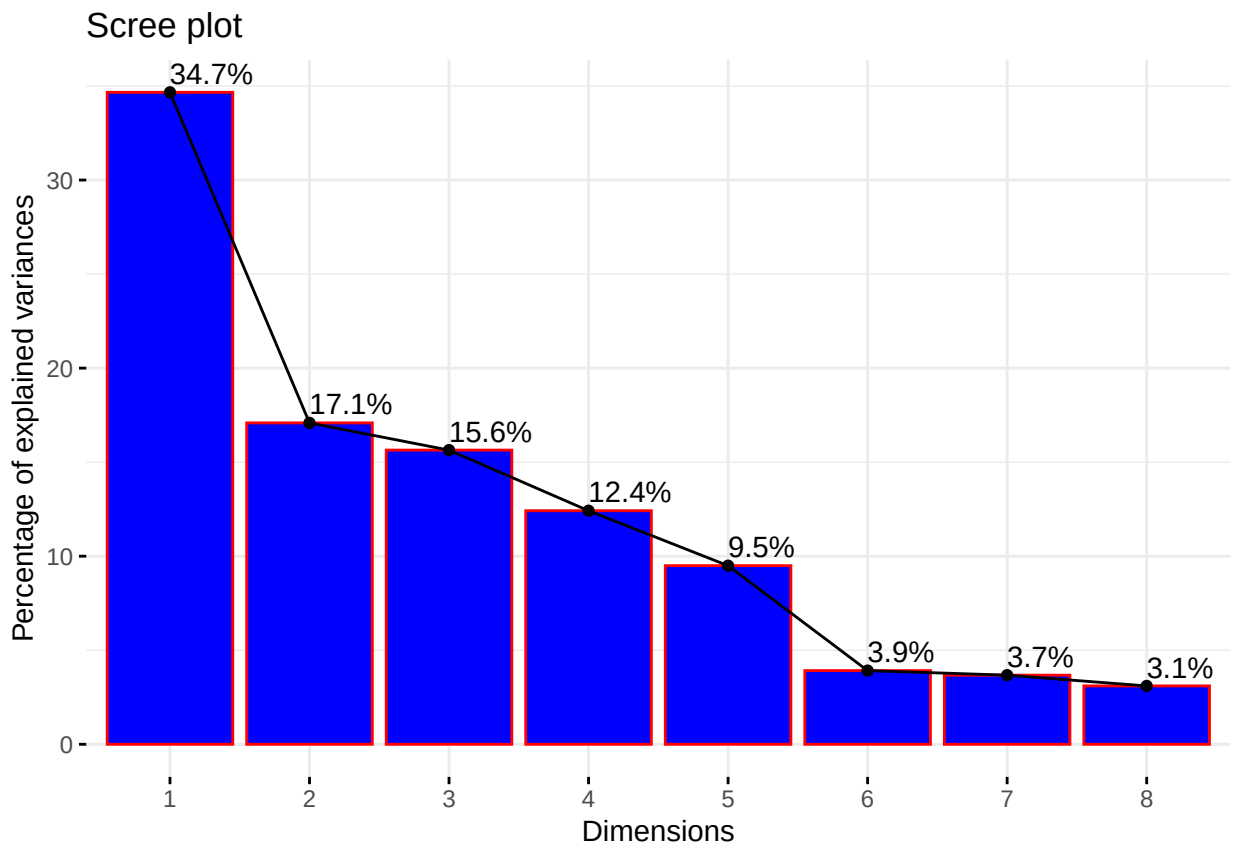
```
pca_output_diab <- PCA(X_diab,ncp = 8, graph = FALSE)
summary(pca_output_diab, nbelements = 10)
```

```
##
## Call:
## PCA(X = X_diab, ncp = 8, graph = FALSE)
##
##
## Eigenvalues
##                        Dim.1   Dim.2   Dim.3   Dim.4   Dim.5   Dim.6   Dim.7
## Variance               2.773   1.367   1.251   0.993   0.760   0.313   0.294
## % of var.             34.663  17.088  15.641  12.418   9.496   3.917   3.675
## Cumulative % of var.  34.663  51.751  67.391  79.809  89.305  93.222  96.897
##                        Dim.8
```

```
## Variance               0.248
## % of var.              3.103
## Cumulative % of var. 100.000
##
## Individuals (the 10 first)
##                            Dist    Dim.1    ctr   cos2    Dim.2    ctr   cos2
## 1                        |  2.228 |  1.770  0.422  0.631 | -0.427  0.050  0.037
## 3                        |  1.941 | -1.477  0.294  0.579 | -0.165  0.007  0.007
## 5                        |  3.727 |  1.024  0.141  0.076 |  0.358  0.035  0.009
## 7                        |  2.366 | -2.024  0.551  0.732 |  0.379  0.039  0.026
## 9                        |  1.881 | -0.949  0.121  0.254 | -0.930  0.236  0.244
## 10                       |  2.656 | -1.661  0.371  0.391 |  0.003  0.000  0.000
## 12                       |  2.846 |  0.630  0.053  0.049 | -1.550  0.656  0.297
## 14                       |  4.473 |  3.751  1.893  0.703 | -0.926  0.234  0.043
## 15                       |  2.738 |  0.166  0.004  0.004 | -0.348  0.033  0.016
## 16                       |  2.770 |  2.230  0.669  0.648 | -0.372  0.038  0.018
##                            Dim.3    ctr   cos2
## 1                        |  0.526  0.083  0.056 |
## 3                        | -0.491  0.072  0.064 |
## 5                        |  3.348  3.343  0.807 |
## 7                        | -0.399  0.047  0.028 |
## 9                        |  0.861  0.221  0.210 |
## 10                       | -0.752  0.169  0.080 |
## 12                       | -1.531  0.699  0.290 |
## 14                       | -1.881  1.056  0.177 |
## 15                       | -0.126  0.005  0.002 |
## 16                       |  0.234  0.016  0.007 |
##
## Variables
##                             Dim.1    ctr   cos2    Dim.2    ctr   cos2    Dim.3
## Pregnancies              |  0.536 10.365  0.287 | -0.688 34.657  0.474 |  0.103
## Glucose                  |  0.695 17.405  0.483 |  0.250  4.563  0.062 |  0.513
## BloodPressure            |  0.549 10.882  0.302 | -0.112  0.923  0.013 | -0.250
## SkinThickness            |  0.608 13.335  0.370 |  0.129  1.215  0.017 | -0.602
## Insulin                  |  0.628 14.235  0.395 |  0.458 15.316  0.209 |  0.462
## BMI                      |  0.684 16.886  0.468 |  0.314  7.216  0.099 | -0.524
## DiabetesPedigreeFunction |  0.158  0.899  0.025 |  0.397 11.509  0.157 |  0.157
## Age                      |  0.666 15.993  0.443 | -0.580 24.601  0.336 |  0.201
##                             ctr   cos2
## Pregnancies               0.855  0.011 |
## Glucose                  21.037  0.263 |
## BloodPressure             4.976  0.062 |
## SkinThickness            28.921  0.362 |
## Insulin                  17.087  0.214 |
## BMI                      21.918  0.274 |
## DiabetesPedigreeFunction  1.966  0.025 |
## Age                       3.240  0.041 |
```

We now check how many components we will use for our analysis.

```
# Method 1
fviz_screeplot(pca_output_diab,ncp=8,addlabels=T,barfill="blue",barcolor="red")
```



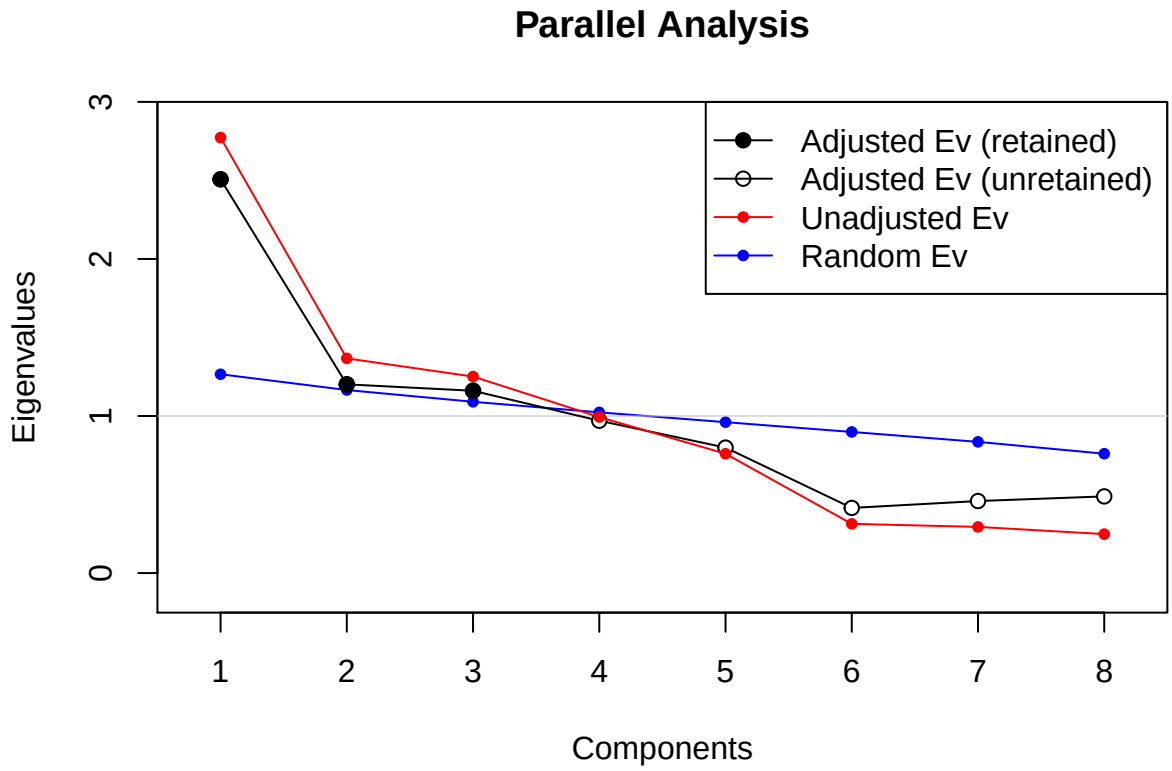

```
# Method 2
pca_output_diab$eig
```

```
##        eigenvalue percentage of variance cumulative percentage of variance
## comp 1  2.7730389              34.662987                           34.66299
## comp 2  1.3670084              17.087606                           51.75059
## comp 3  1.2512565              15.640707                           67.39130
## comp 4  0.9934175              12.417719                           79.80902
## comp 5  0.7596516               9.495645                           89.30466
## comp 6  0.3133825               3.917281                           93.22194
## comp 7  0.2940115               3.675143                           96.89709
## comp 8  0.2482330               3.102913                          100.00000
```

```
# Method 3
pca_output_ret <- paran(X_diab,seed=1,graph = TRUE)
```

```
##
## Using eigendecomposition of correlation matrix.
## Computing: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
##
##
## Results of Horn's Parallel Analysis for component retention
## 240 iterations, using the mean estimate
##
## ----------------------------------------------------
```

```
## Component   Adjusted    Unadjusted    Estimated
##             Eigenvalue  Eigenvalue    Bias
## ------------------------------------------------
## 1           2.507266    2.773038      0.265772
## 2           1.201602    1.367008      0.165405
## 3           1.160666    1.251256      0.090589
## ------------------------------------------------
##
## Adjusted eigenvalues > 1 indicate dimensions to retain.
## (3 components retained)
```




```
pca_output_ret$Retained
```

```
## [1] 3
```

We also keep 3 principal components in this case.

Let´s see how much variability of the data they can explain.

```
pca_output_diab$eig[,2][1:3]
```

```
##   comp 1   comp 2   comp 3
## 34.66299 17.08761 15.64071
```

```
pca_output_diab$eig[,3][1:3]
```

```
##   comp 1   comp 2   comp 3
## 34.66299 51.75059 67.39130
```

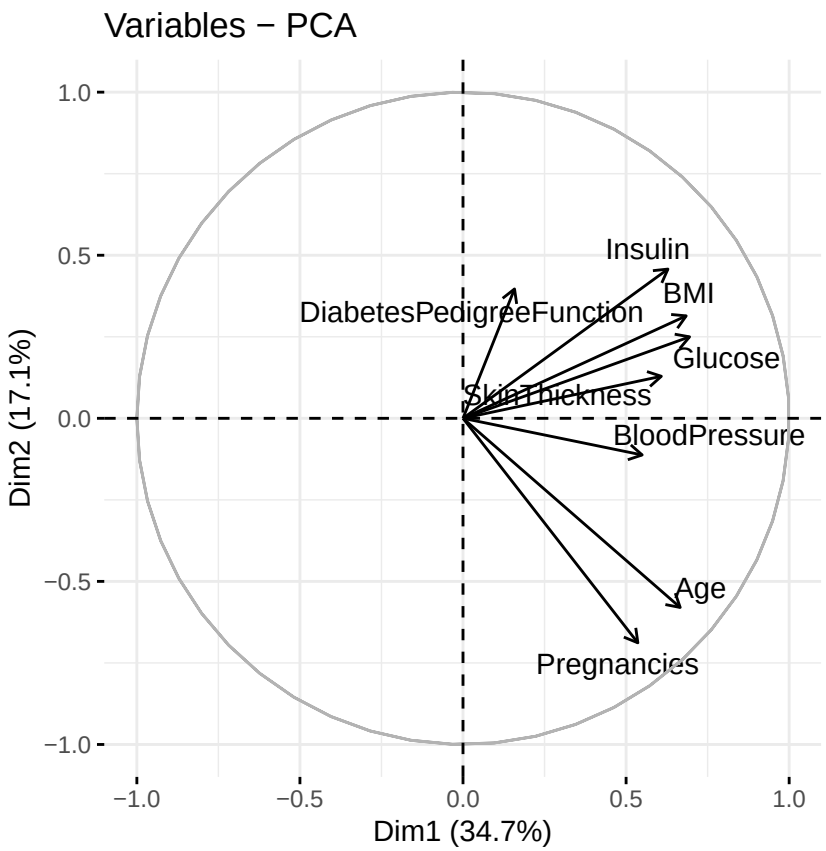
We can explain almost 70% of the information of the data by only taking three variables. This is how powerful this method is: we can only make use of less than half of our variables and explain almost 70% of the variability. Of course, if we would have a larger number of variables in our original data we could see this method much more useful.

Let´s check the contributions of each variable.

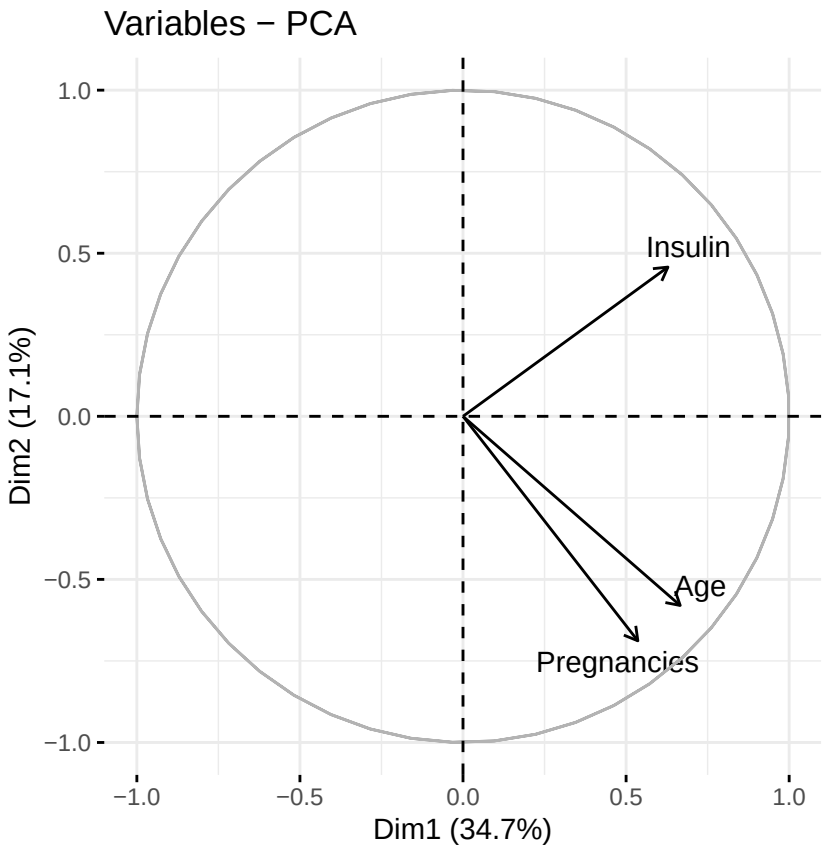
```
pca_output_diab$var$contrib
```

```
##                              Dim.1      Dim.2      Dim.3      Dim.4     Dim.5
## Pregnancies              10.3651861 34.6571504  0.8547013  7.3956207  4.442884
## Glucose                  17.4046522  4.5630960 21.0369590  2.1283167  2.118240
## BloodPressure            10.8815278  0.9232466  4.9757335 17.4729430 55.429834
## SkinThickness            13.3354519  1.2154923 28.9213208  5.7546837 10.295827
## Insulin                  14.2354732 15.3155718 17.0874857  3.6028193  2.063079
## BMI                      16.8856988  7.2157167 21.9182653  0.3079427  1.084022
## DiabetesPedigreeFunction  0.8990625 11.5085959  1.9659938 62.1679404 22.890096
## Age                      15.9929475 24.6011303  3.2395406  1.1697335  1.676016
##                               Dim.6        Dim.7        Dim.8
## Pregnancies              1.88428241 32.105185856  8.294988809
## Glucose                 36.18771353  0.017034627 16.543987990
## BloodPressure            4.54807980  0.317834899  5.450800082
## SkinThickness            8.85160968 22.023173487  9.602440805
## Insulin                 18.74182781  0.003799246 28.949943609
## BMI                     13.11982483 18.516827372 20.951702409
## DiabetesPedigreeFunction  0.01629085  0.550443658  0.001576481
## Age                     16.65037110 26.465700855 10.204559816
```

```
fviz_pca_var(pca_output_diab,col.bar="contrib",gradient.cols=c("#bb2e00","#002bbb"),repel=TRUE)
```

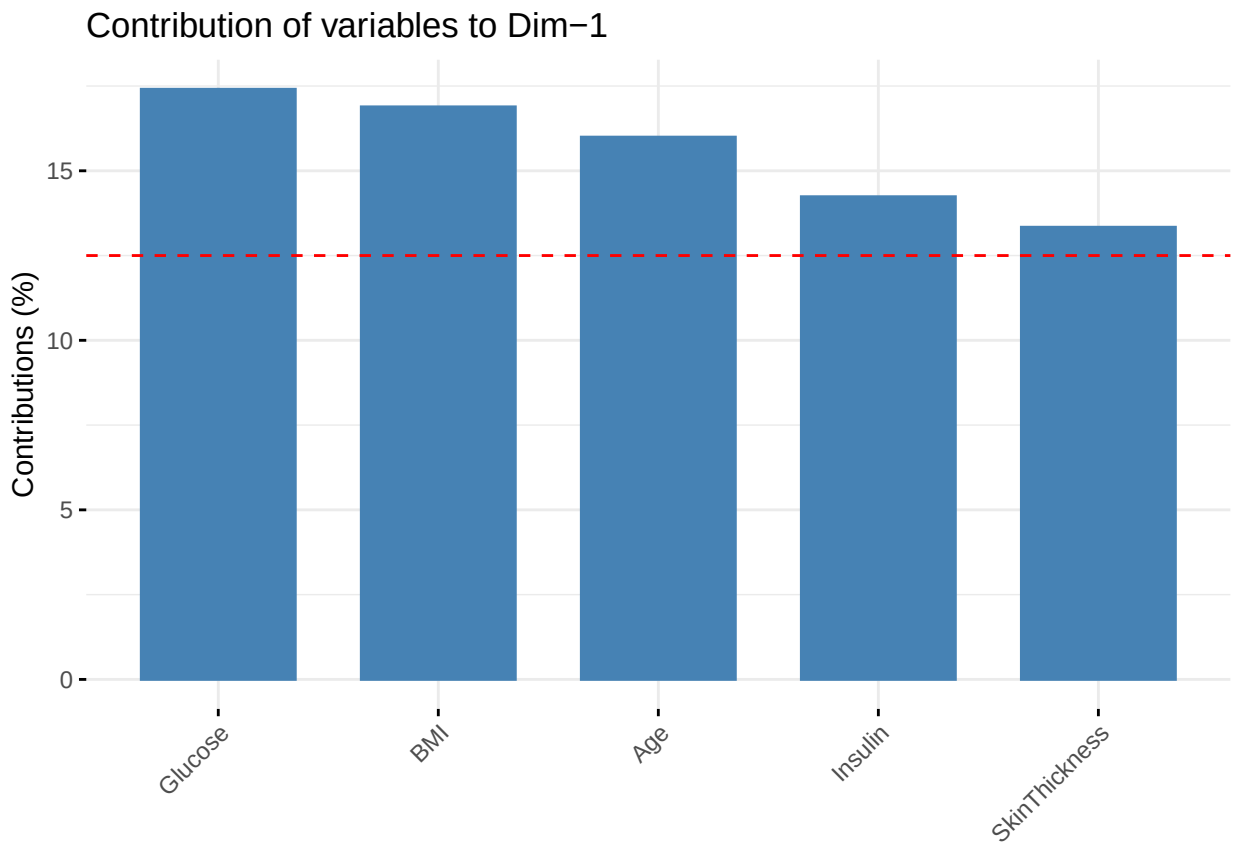



```
fviz_pca_var(pca_output_diab,select.var=list(cos2=0.6),repel=TRUE)
```

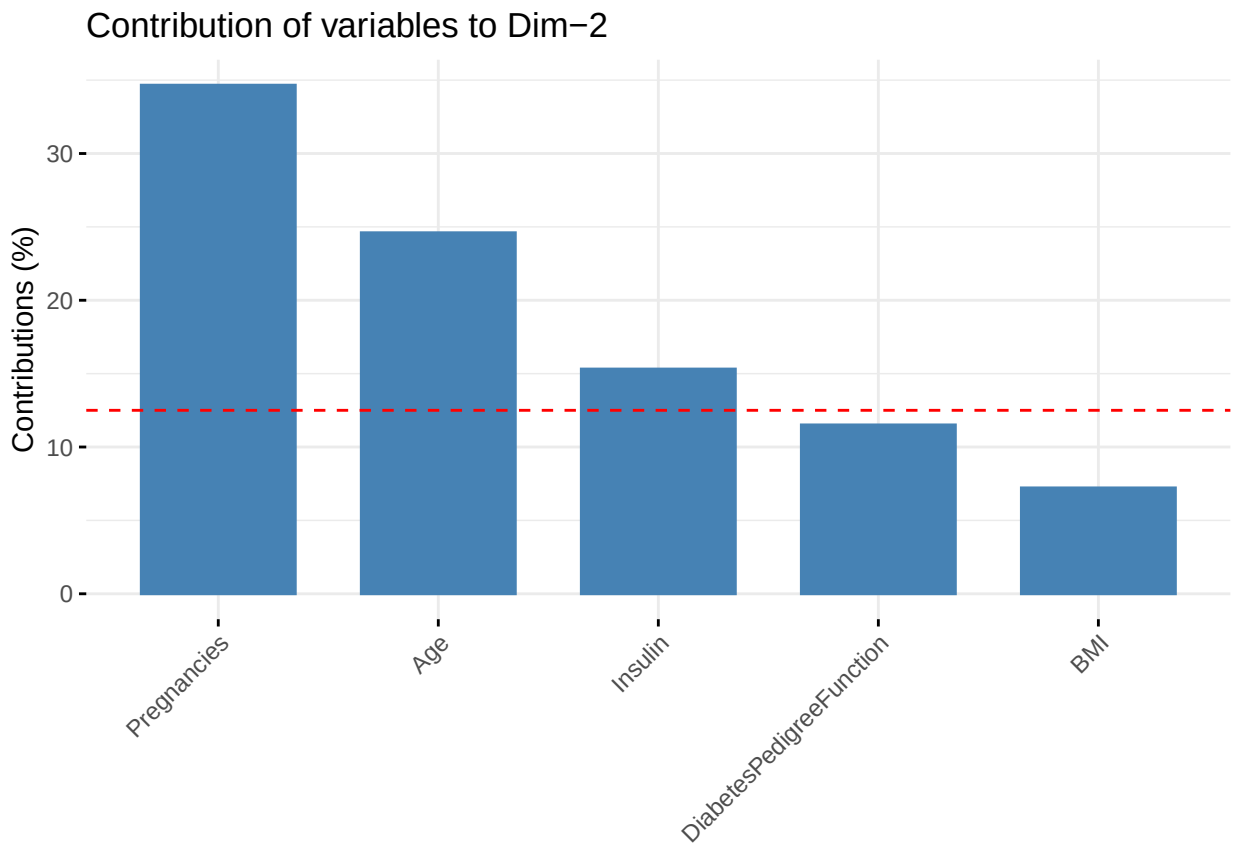


Just as before, all the original variables are positively correlated with the first principal component, but not all of them are positively correlated with the second principal component. Between the variables that have more than 0.6 as COS2, we get “age”, “pregnancies” and “BMI”. We can see that both age and pregnancies are negatively correlated to the second component, while the BMI is positively correlated to both first and second component.

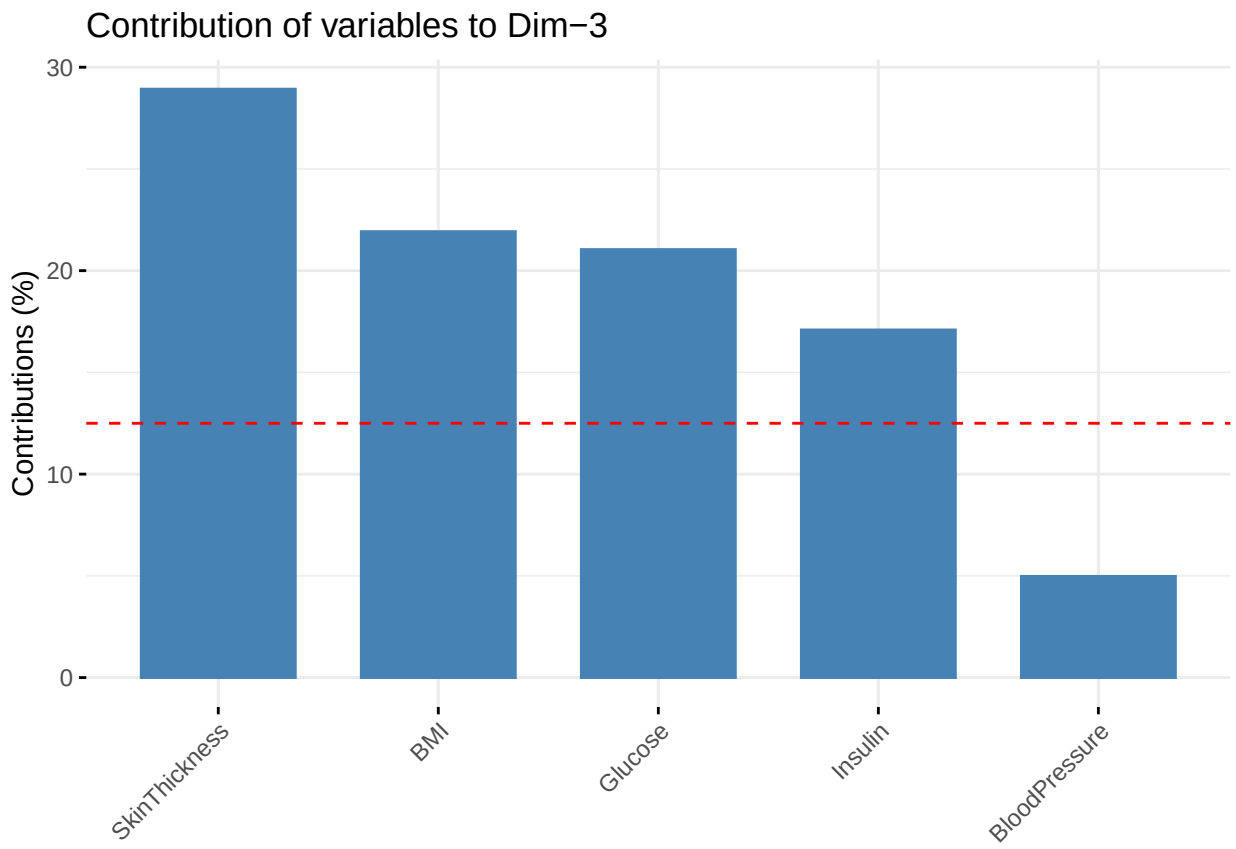
```
fviz_contrib(pca_output_diab, choice="var", axes=1,top=5)
```



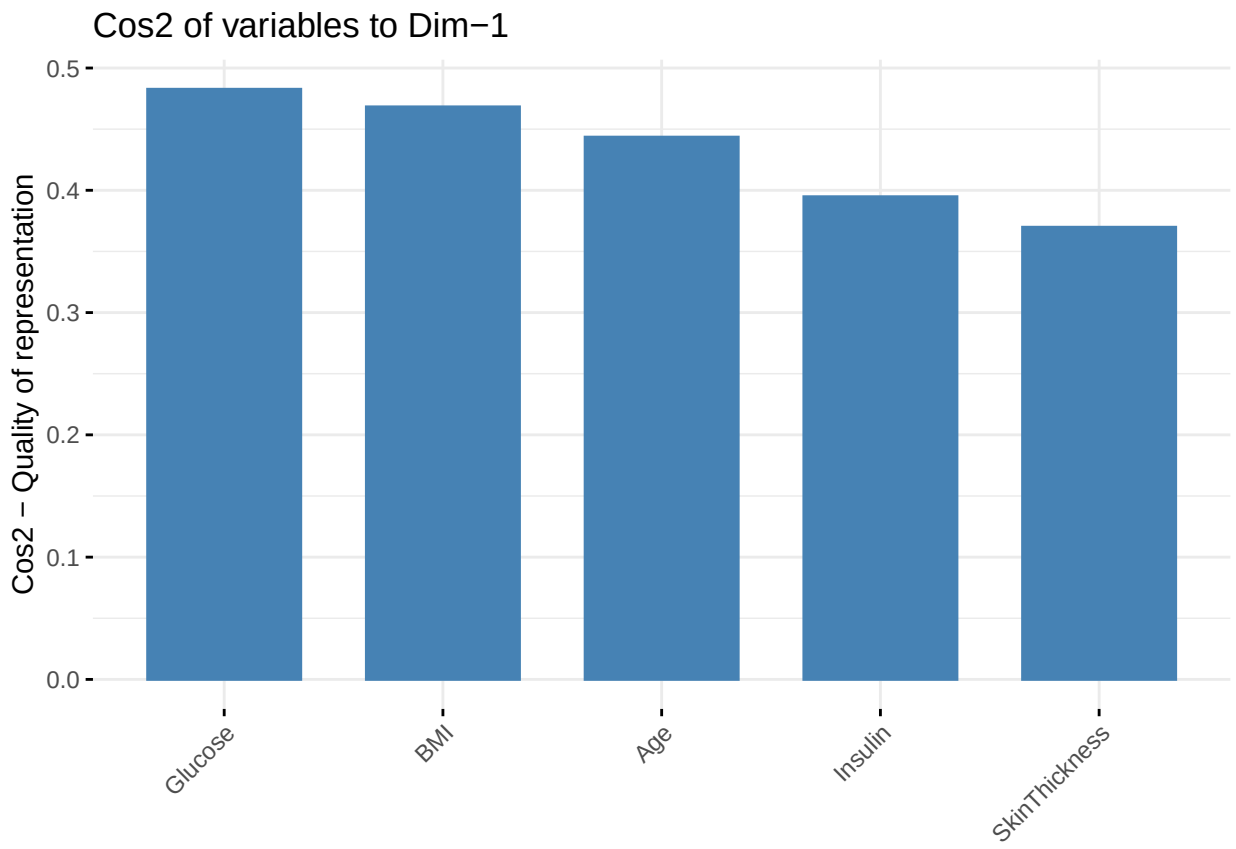

```
fviz_contrib(pca_output_diab, choice="var", axes=2,top=5)
```



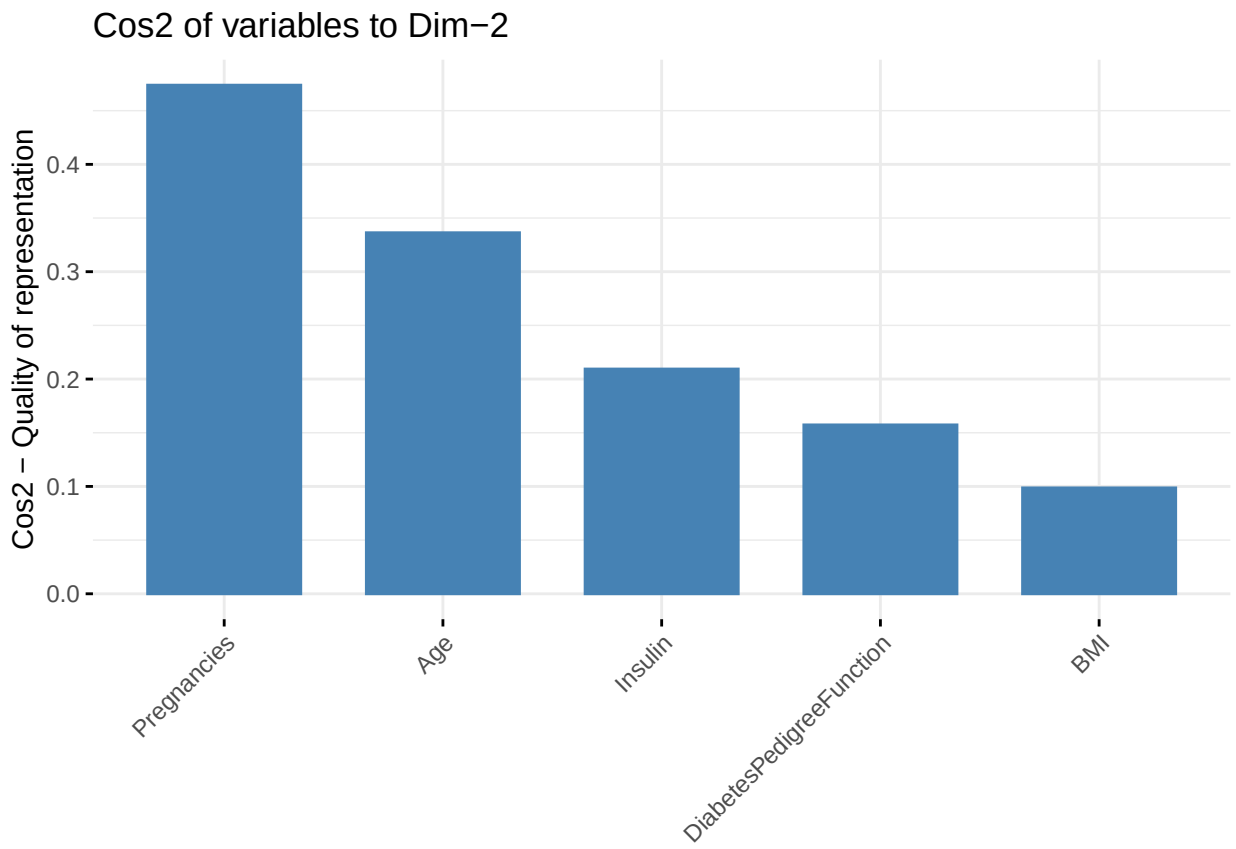

```
fviz_contrib(pca_output_diab, choice="var", axes=3,top=5)
```



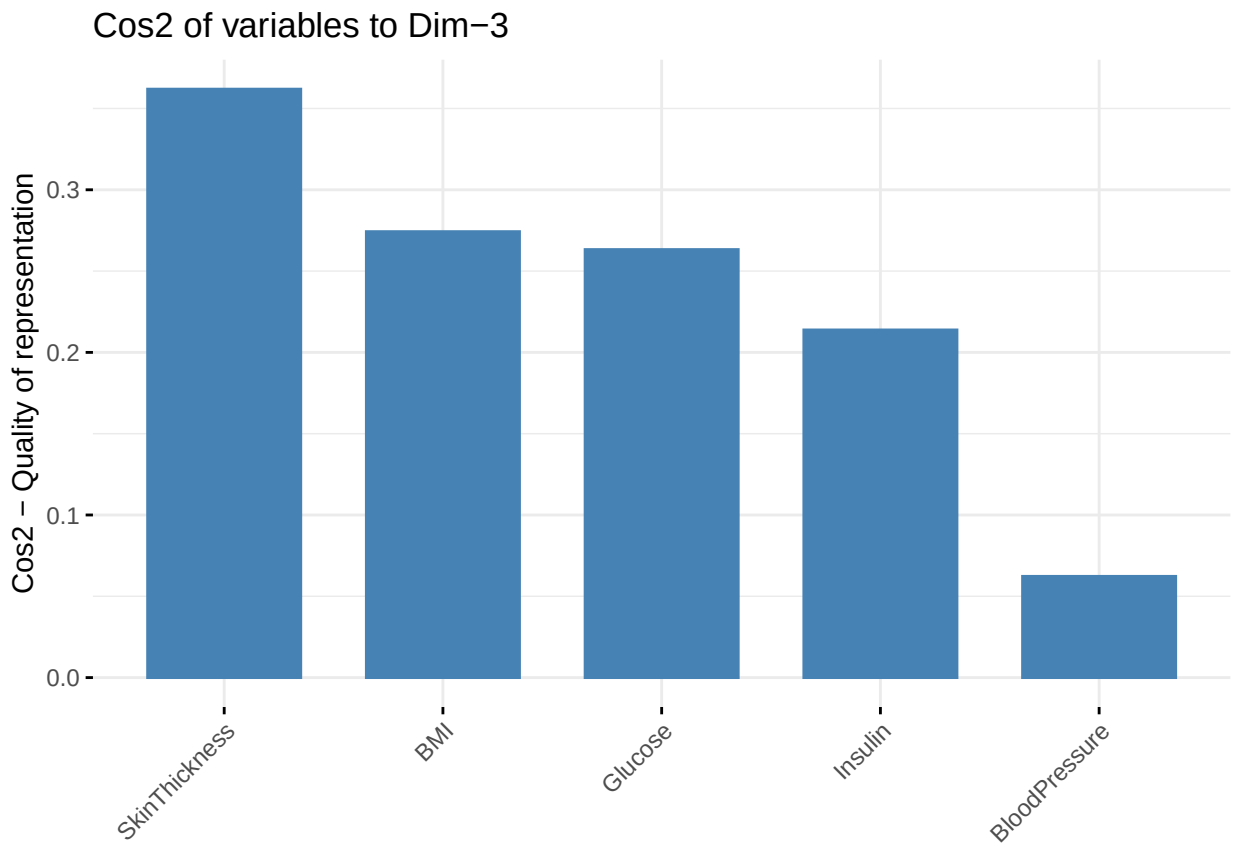

```
fviz_cos2(pca_output_diab, choice = "var", axes = 1, top = 5)
```




```
fviz_cos2(pca_output_diab, choice = "var", axes = 2, top = 5)
```




```
fviz_cos2(pca_output_diab, choice = "var", axes = 3, top = 5)
```



It seems that the first principal component encapsulates a lot of information regarding specifical issues of health. It calls our attention that the variable that contributes more to the third principal component is the Diabetes Pedigree Function. This is an important point, as the Diabetes Pedigree Function talks about the inheritence of the desease. It shows us, somehow, that diabetes may follow a genetic pattern, so those whose ancestors suffered from diabetes may get diabetes too (as now we are only studying people with diabetes).

Now, let´s do the same with subgroup of non-diabetic.

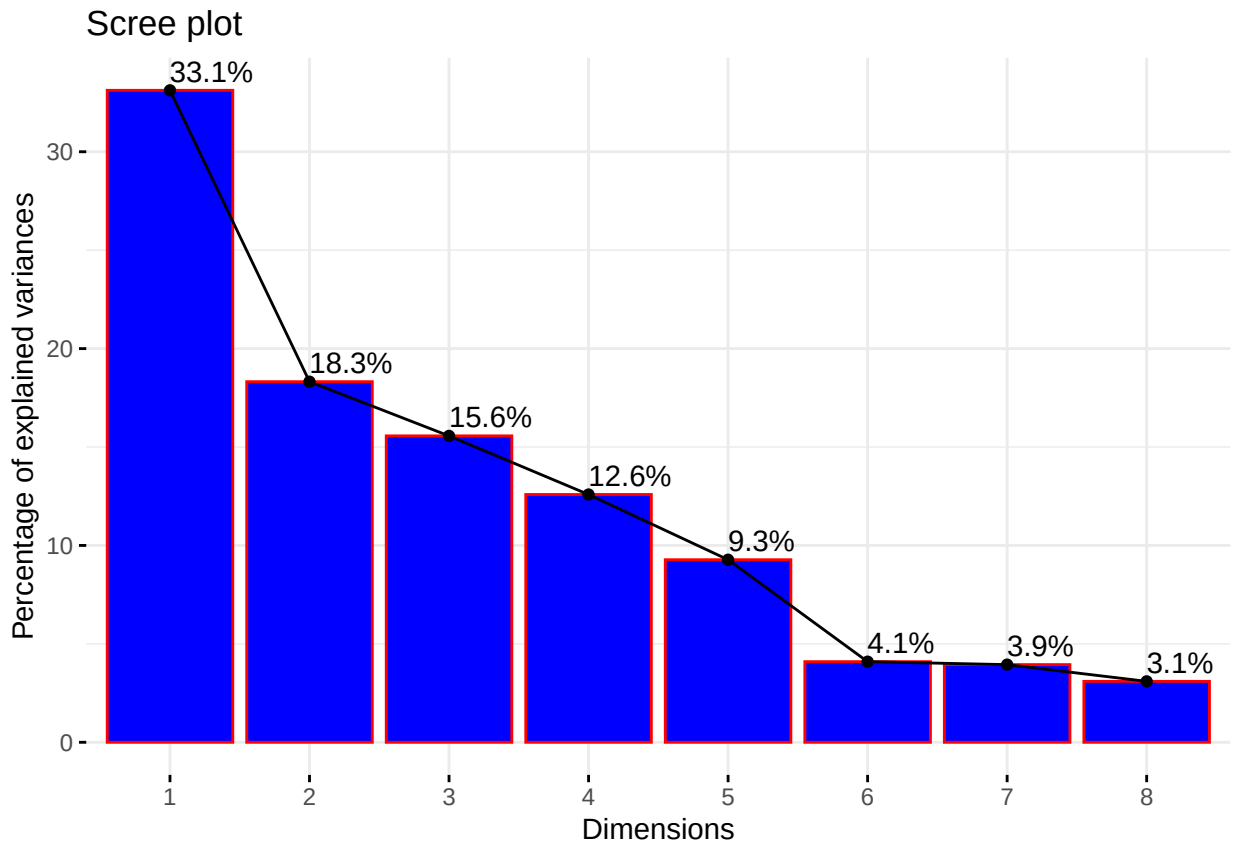
```
pca_output_no_diab <- PCA(X_nodiab,ncp = 8, graph = FALSE)
summary(pca_output_no_diab, nbelements = 10)
```

```
##
## Call:
## PCA(X = X_nodiab, ncp = 8, graph = FALSE)
##
##
## Eigenvalues
##                        Dim.1   Dim.2   Dim.3   Dim.4   Dim.5   Dim.6   Dim.7
## Variance               2.650   1.465   1.245   1.007   0.742   0.328   0.316
## % of var.             33.119  18.318  15.564  12.587   9.275   4.095   3.946
## Cumulative % of var.  33.119  51.437  67.002  79.589  88.863  92.958  96.904
##                        Dim.8
## Variance               0.248
## % of var.              3.096
## Cumulative % of var. 100.000
##
## Individuals (the 10 first)
```

```
##                             Dist    Dim.1    ctr   cos2    Dim.2    ctr   cos2
## 2                         |  2.235 |  1.760  0.234  0.620 | -0.459  0.029  0.042
## 4                         |  1.906 | -1.404  0.149  0.543 | -0.195  0.005  0.010
## 6                         |  3.758 |  0.976  0.072  0.067 |  0.101  0.001  0.001
## 8                         |  2.306 | -1.922  0.279  0.694 |  0.319  0.014  0.019
## 11                        |  1.854 | -0.902  0.061  0.236 | -1.107  0.167  0.357
## 13                        |  2.686 | -1.594  0.192  0.352 |  0.154  0.003  0.003
## 19                        |  2.927 |  0.698  0.037  0.057 | -1.391  0.264  0.226
## 21                        |  4.508 |  3.844  1.116  0.727 | -0.836  0.095  0.034
## 22                        |  2.720 |  0.243  0.004  0.008 | -0.558  0.043  0.042
## 28                        |  2.828 |  2.219  0.372  0.616 | -0.358  0.017  0.016
##                             Dim.3    ctr   cos2
## 2                         | -0.340  0.019  0.023 |
## 4                         |  0.551  0.049  0.084 |
## 6                         | -3.250  1.697  0.748 |
## 8                         |  0.239  0.009  0.011 |
## 11                        | -0.851  0.116  0.211 |
## 13                        |  0.787  0.100  0.086 |
## 19                        |  1.545  0.383  0.279 |
## 21                        |  1.716  0.473  0.145 |
## 22                        | -0.019  0.000  0.000 |
## 28                        | -0.155  0.004  0.003 |
##
## Variables
##                             Dim.1    ctr   cos2    Dim.2    ctr   cos2    Dim.3
## Pregnancies               |  0.502  9.502  0.252 | -0.693 32.750  0.480 | -0.002
## Glucose                   |  0.659 16.370  0.434 |  0.223  3.391  0.050 | -0.591
## BloodPressure             |  0.556 11.660  0.309 | -0.199  2.705  0.040 |  0.199
## SkinThickness             |  0.645 15.720  0.417 |  0.214  3.132  0.046 |  0.558
## Insulin                   |  0.612 14.123  0.374 |  0.429 12.560  0.184 | -0.537
## BMI                       |  0.663 16.596  0.440 |  0.399 10.846  0.159 |  0.490
## DiabetesPedigreeFunction  |  0.095  0.338  0.009 |  0.378  9.735  0.143 |  0.031
## Age                       |  0.645 15.691  0.416 | -0.604 24.881  0.365 | -0.123
##                             ctr   cos2
## Pregnancies               0.000  0.000 |
## Glucose                  28.093  0.350 |
## BloodPressure             3.180  0.040 |
## SkinThickness            25.013  0.311 |
## Insulin                  23.183  0.289 |
## BMI                      19.247  0.240 |
## DiabetesPedigreeFunction  0.077  0.001 |
## Age                       1.207  0.015 |
```

Let´s check how many principal components we are taking for our analysis.

```
# Method 1
fviz_screeplot(pca_output_no_diab,ncp=8,addlabels=T,barfill="blue",barcolor="red")
```



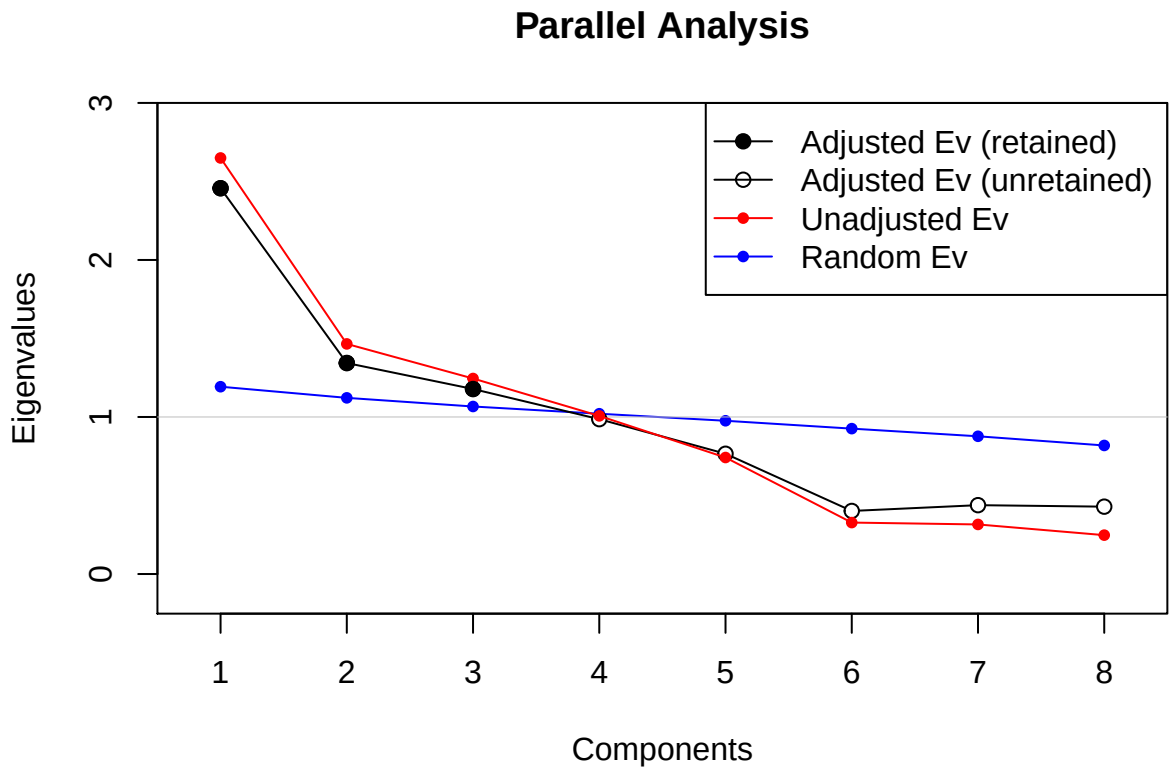

```
# Method 2
pca_output_no_diab$eig
```

```
##        eigenvalue percentage of variance cumulative percentage of variance
## comp 1  2.6495566              33.119457                           33.11946
## comp 2  1.4654213              18.317766                           51.43722
## comp 3  1.2451566              15.564457                           67.00168
## comp 4  1.0069601              12.587001                           79.58868
## comp 5  0.7419623               9.274528                           88.86321
## comp 6  0.3276196               4.095246                           92.95846
## comp 7  0.3156578               3.945722                           96.90418
## comp 8  0.2476658               3.095822                          100.00000
```

```
# Method 3
pca_output_ret <- paran(X_nodiab,seed=1,graph = TRUE)
```

```
##
## Using eigendecomposition of correlation matrix.
## Computing: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
##
##
## Results of Horn's Parallel Analysis for component retention
## 240 iterations, using the mean estimate
##
## ----------------------------------------------------
```

```
## Component   Adjusted    Unadjusted    Estimated
##             Eigenvalue  Eigenvalue    Bias
## ------------------------------------------------
## 1           2.456760    2.649556      0.192795
## 2           1.343635    1.465421      0.121785
## 3           1.178277    1.245156      0.066879
## ------------------------------------------------
##
## Adjusted eigenvalues > 1 indicate dimensions to retain.
## (3 components retained)
```




```
pca_output_ret$Retained
```

```
## [1] 3
```

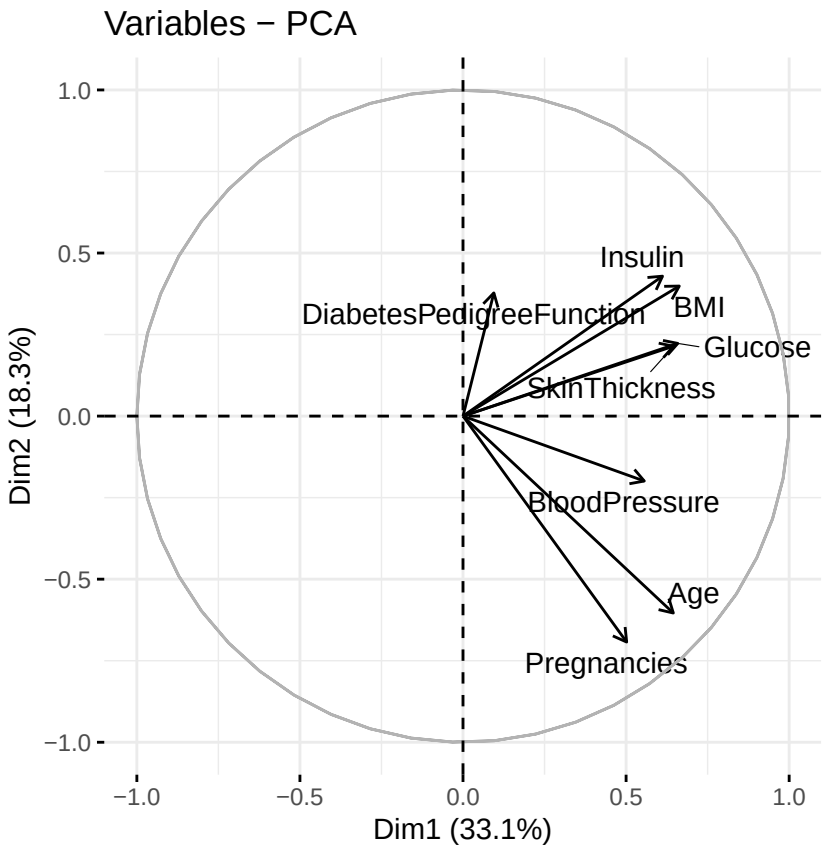
And again, we will take just 3. Let´s check contributions.

```
pca_output_no_diab$var$contrib
```

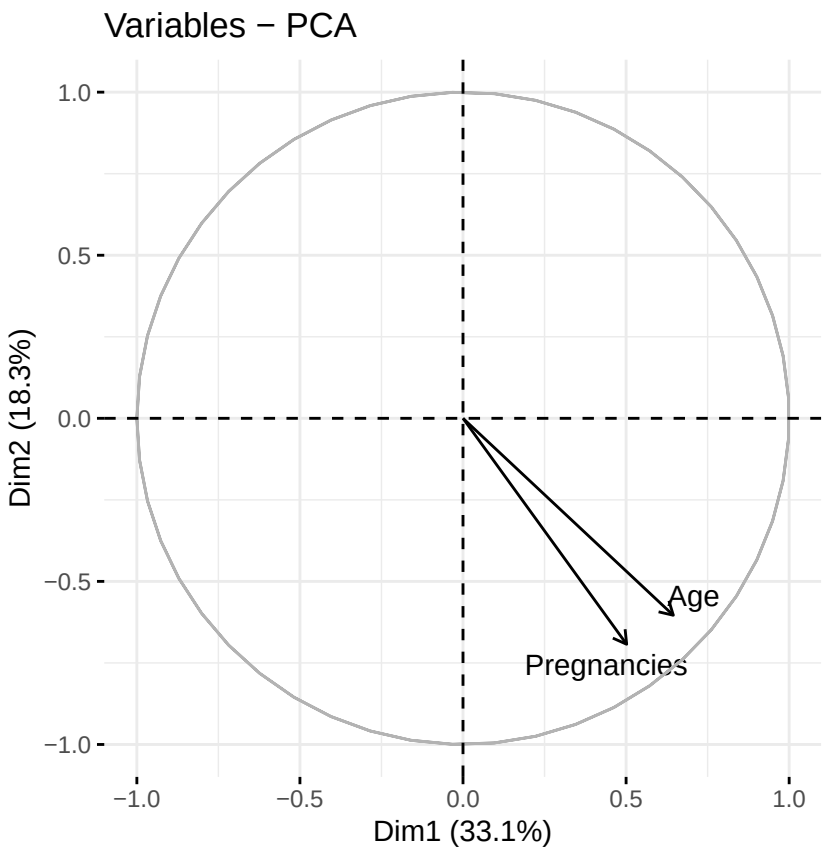
```
##                              Dim.1     Dim.2        Dim.3      Dim.4
## Pregnancies              9.5015094 32.749569 3.834708e-04  8.1137898
## Glucose                 16.3703953  3.390721 2.809304e+01  0.5038383
## BloodPressure           11.6596369  2.705434 3.179829e+00 15.7808716
## SkinThickness           15.7204213  3.132371 2.501254e+01  0.4695294
```

```
## Insulin                  14.1232573 12.559546 2.318343e+01  0.7901287
## BMI                      16.5958183 10.846427 1.924715e+01  1.0636013
## DiabetesPedigreeFunction  0.3375416  9.735049 7.687518e-02 69.5625846
## Age                      15.6914199 24.880883 1.206749e+00  3.7156562
##                               Dim.5      Dim.6        Dim.7      Dim.8
## Pregnancies               7.4693499 33.4836479  0.007467069  8.6742829
## Glucose                   0.8906687  1.1292416 41.899053719  7.7230432
## BloodPressure            57.8361889  0.7493883  1.932180809  6.1564699
## SkinThickness            12.6274158 18.4983113  9.313450089 15.2259651
## Insulin                   0.2893697  0.2950769 31.248007900 17.5111798
## BMI                       0.7281549 14.7384779  9.104194512 27.6761709
## DiabetesPedigreeFunction 19.4654889  0.3349141  0.354742273  0.1328046
## Age                       0.6933631 30.7709421  6.140903628 16.9000836
```

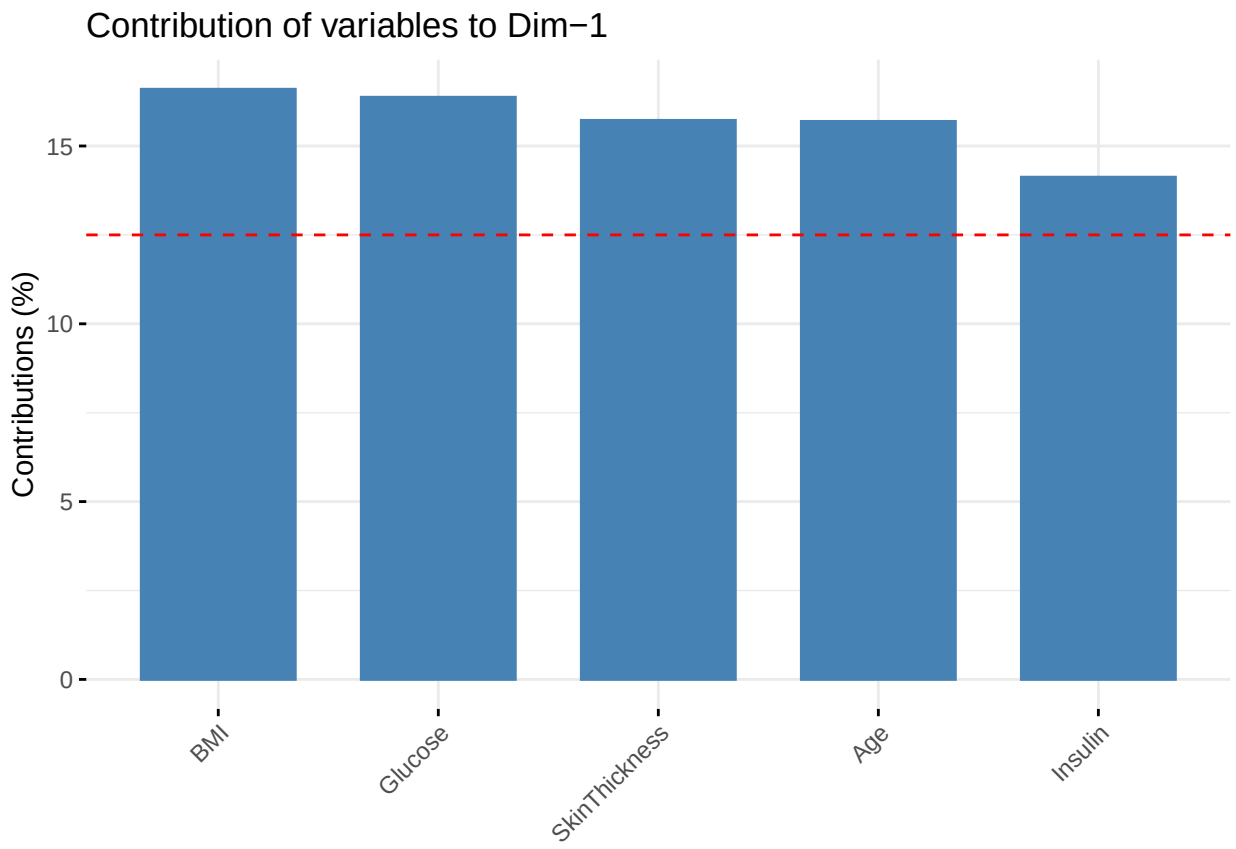
```
fviz_pca_var(pca_output_no_diab,col.bar="contrib",gradient.cols=c("#bb2e00","#002bbb"),repel=TRUE)
```



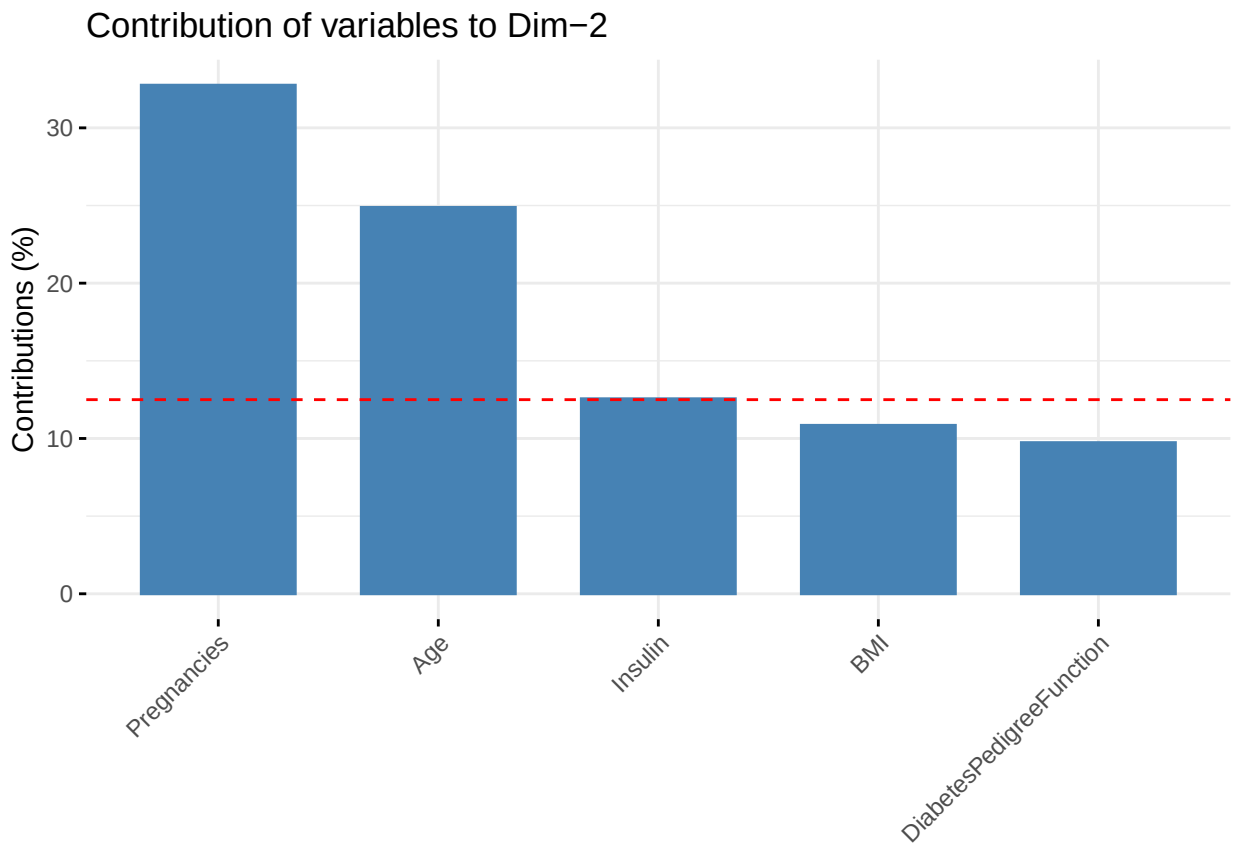

```
fviz_pca_var(pca_output_no_diab,select.var=list(cos2=0.6),repel=TRUE)
```



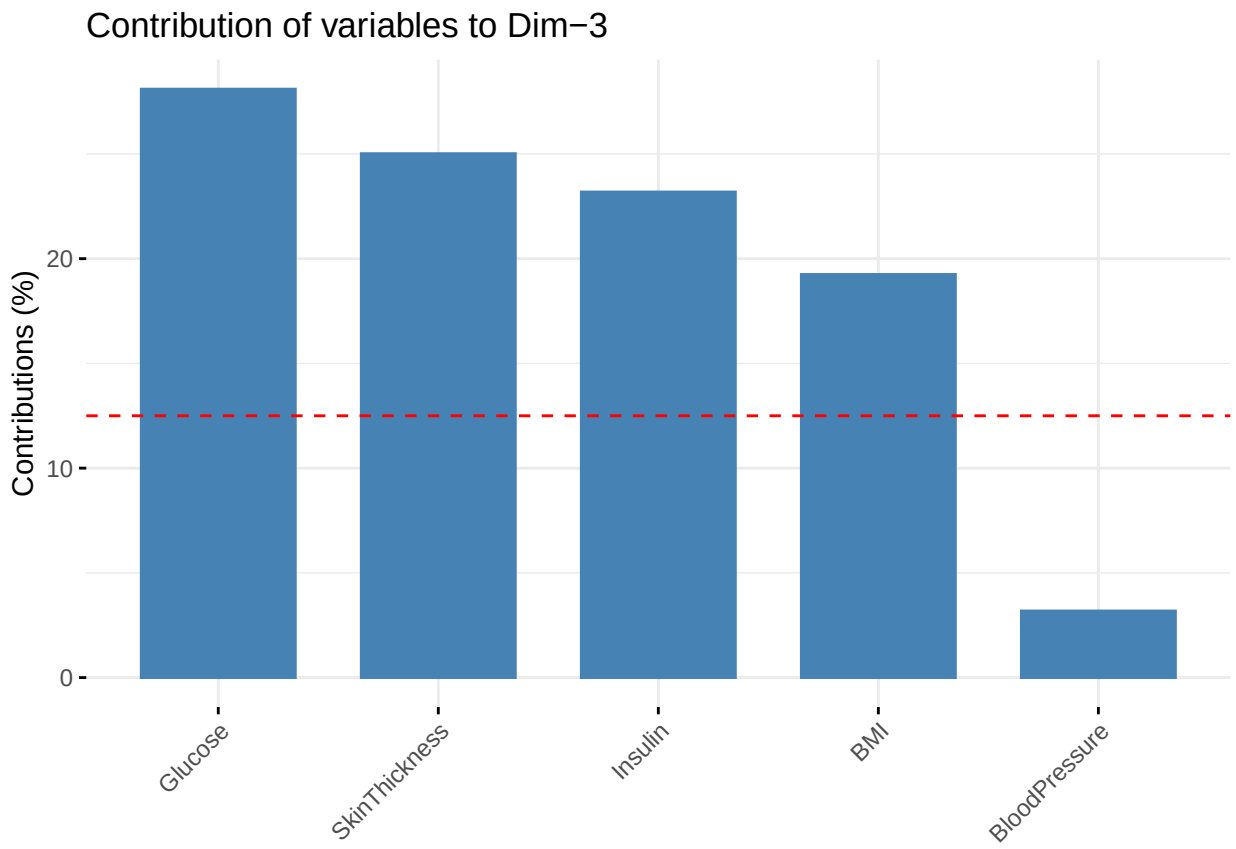

```
fviz_contrib(pca_output_no_diab, choice="var", axes=1,top=5)
```



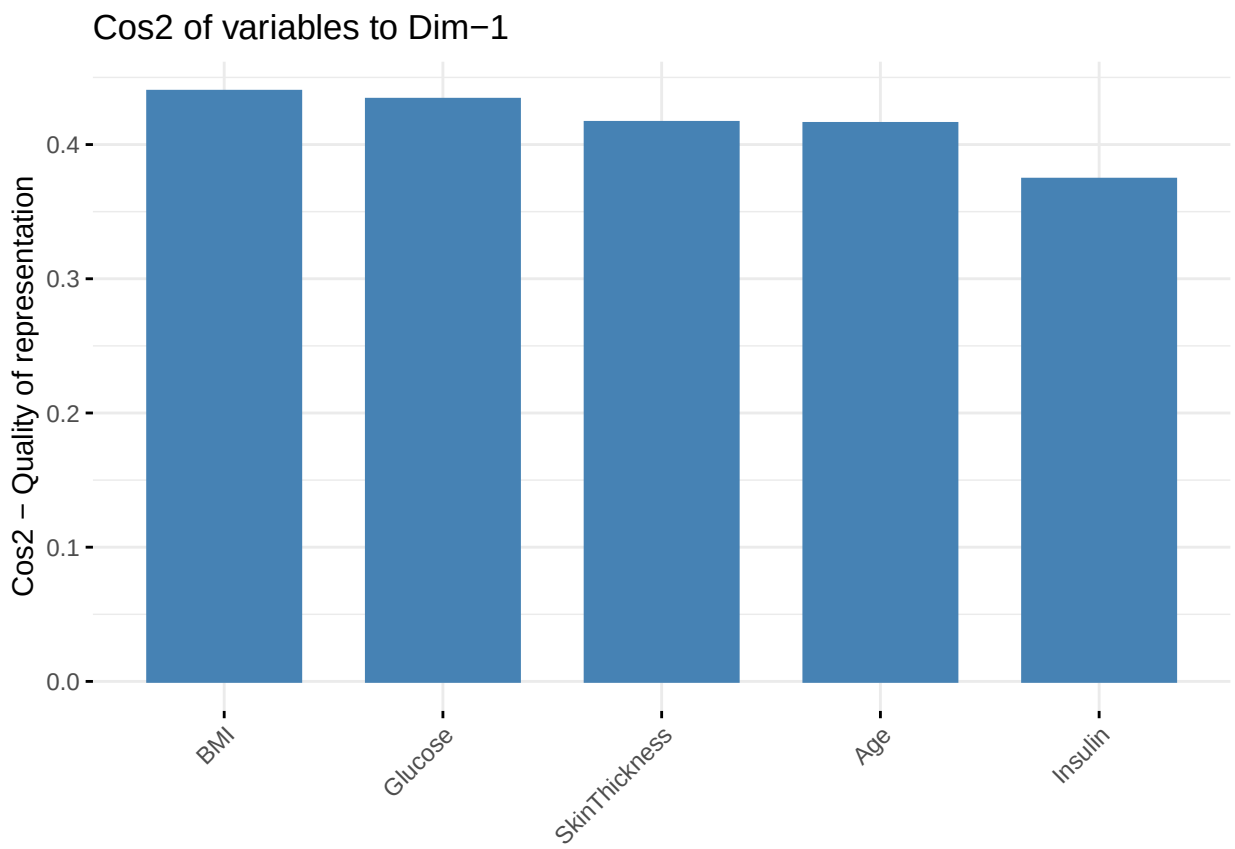

```
fviz_contrib(pca_output_no_diab, choice="var", axes=2,top=5)
```



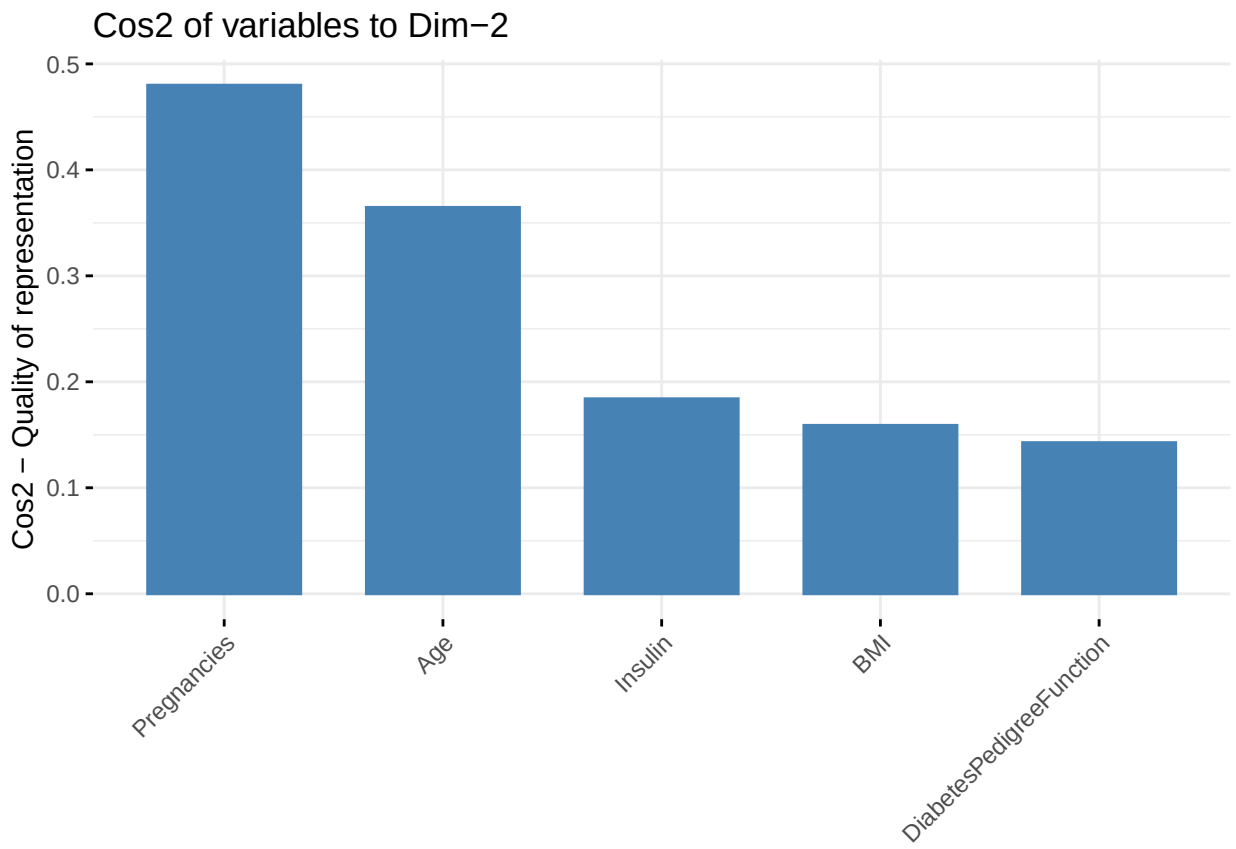

```
fviz_contrib(pca_output_no_diab, choice="var", axes=3,top=5)
```



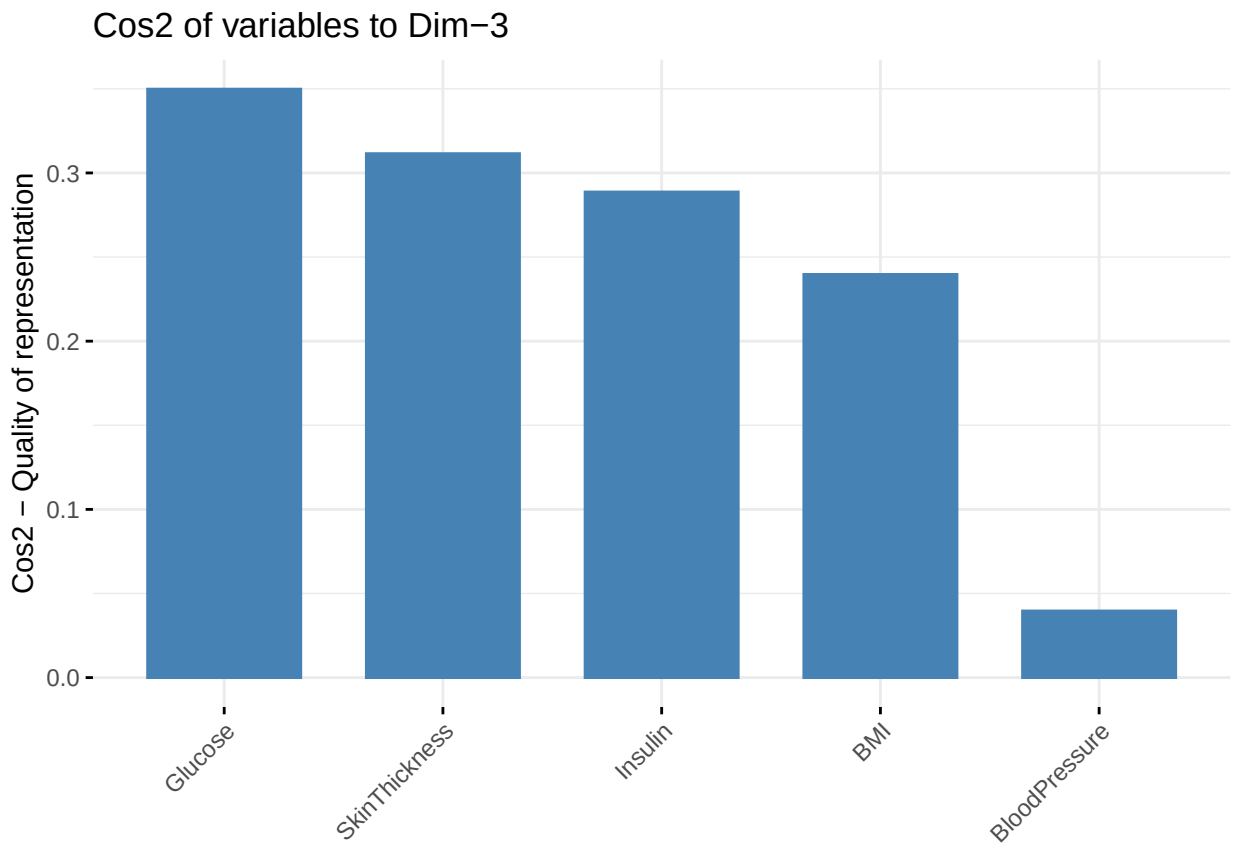

```
fviz_cos2(pca_output_no_diab, choice = "var", axes = 1, top = 5)
```




```
fviz_cos2(pca_output_no_diab, choice = "var", axes = 2, top = 5)
```



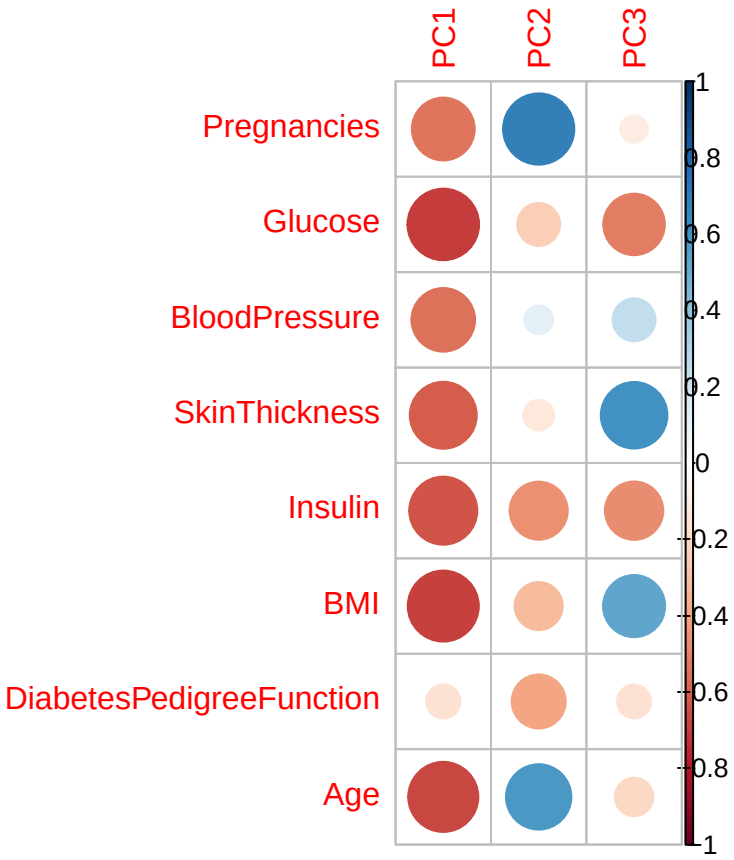

```
fviz_cos2(pca_output_no_diab, choice = "var", axes = 3, top = 5)
```



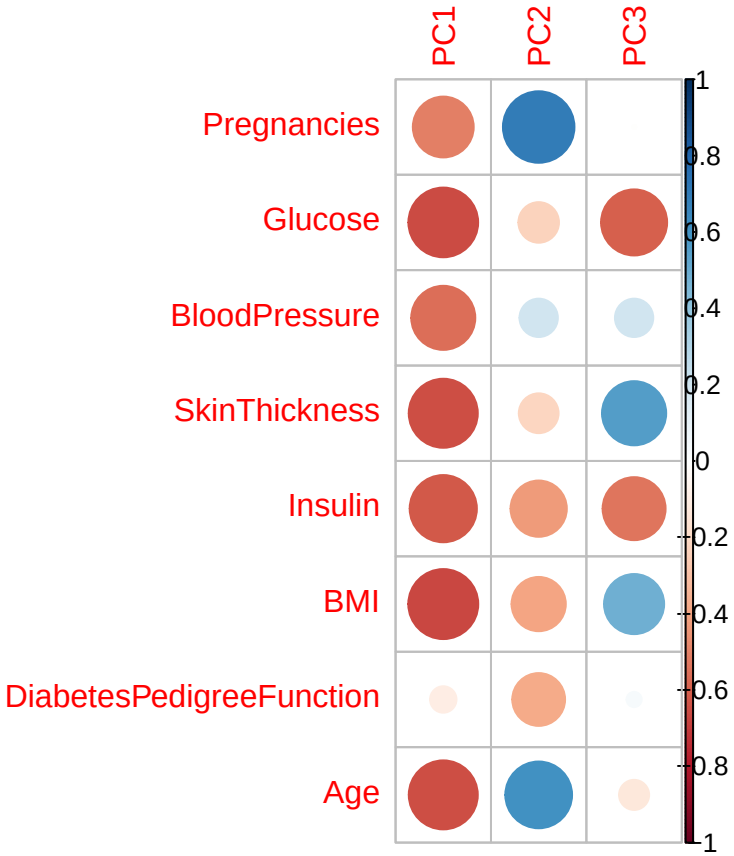
Now we can see that the new variables behave similar to the ones in the whole data. The first component takes us to a ranking of health, the second one talks about cycle of life or fertility. Now the pedigree function don´t seem to have a big influence on any of the principal components.

Let´s compare the correlation matrix of both subgroups (their principal components and original variables).

```
X_pc_diab <- prcomp(X_diab,scale=TRUE)
corrplot(cor(X_diab,X_pc_diab$x[,1:3]),is.corr=T)
```




```
X_pc_no_diab <- prcomp(X_nodiab, scale=TRUE)
corrplot(cor(X_nodiab,X_pc_no_diab$x[,1:3]), is.corr=T)
```



We can see that for bouth groups, both the first and second principal component look similar in regards to their relationship with the original variables. The one that changes a bit is the third one. This analysis may be telling us that our variables do not help us a lot in distinguishing diabetic from non-diabetic. But we would have to perform clustering techniques to check this more in depht.

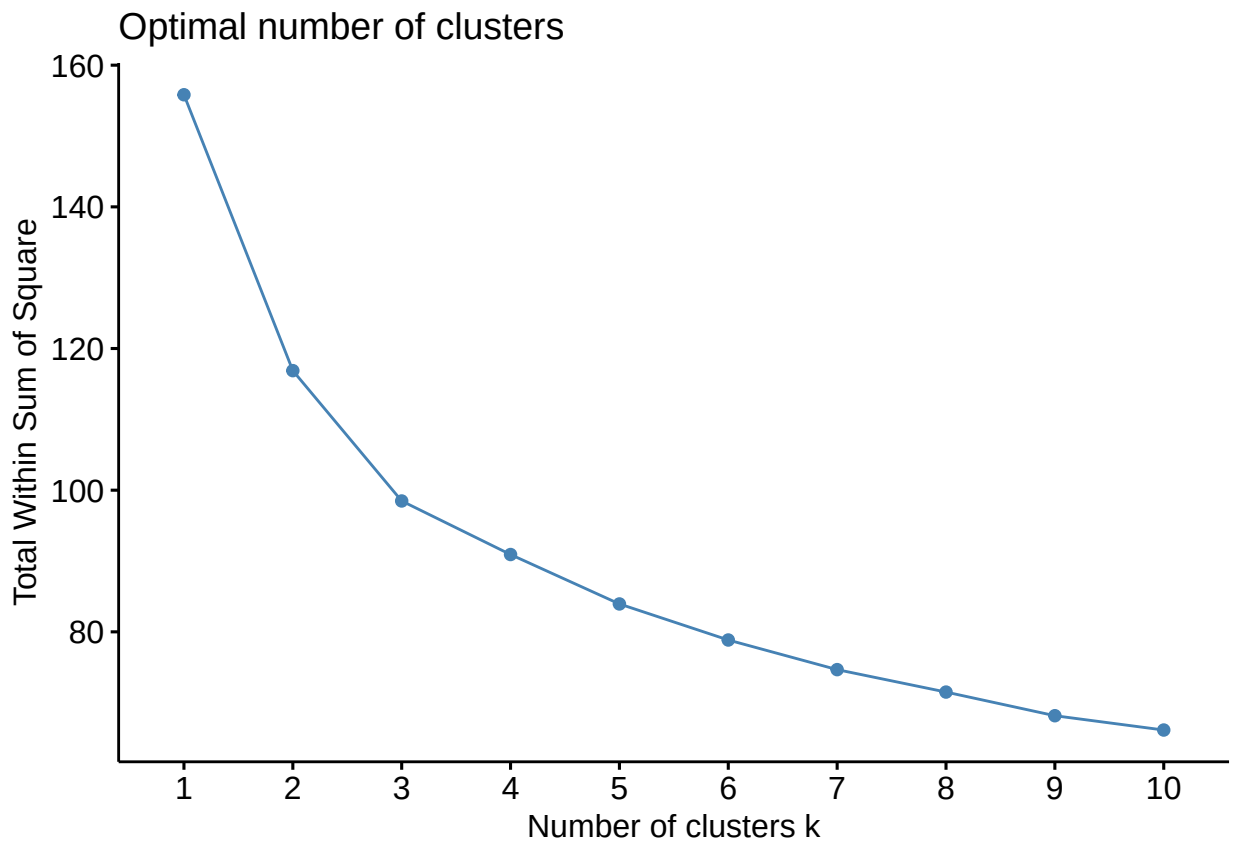
Second part

In this part we will continue with the clustering classification. This is a kind of classification is in the groups of unsupervised classification because we dont need a previous knowledge about the groups that exists.In supervised classification we need to train the data knowing that each observation belongs to a group. Here. We create the groups and we put each observation in each group.

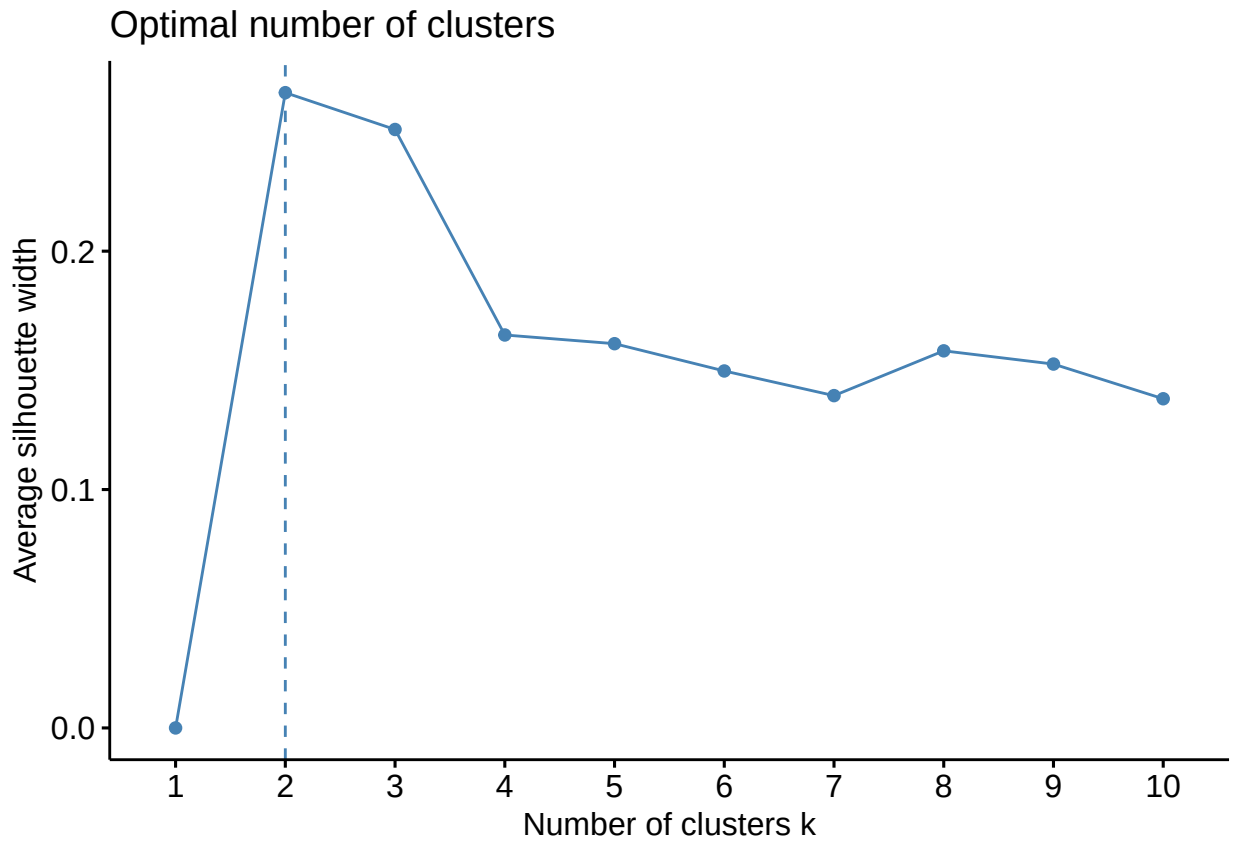
Firsts we start with partitional clustering. We start from an initial cluster definition and then we proceed by exchanging elements between cluster until an appropriate cluster structure is found. In order to do that , we have to select the number of clustering. For it, we will check the optimal number of clusters with three criteria; WSS, average silhouette and gap statistic.

```
covar <- cov(X)
S <- covar[1:p,1:p]
eig_S <- eigs_sym(S,2)
X_centred <- scale(X,scale=FALSE)
eigen_vectors_S <- eig_S$vectors[,1:2]
Z <- X_centred %*% eigen_vectors_S
```

```
fviz_nbclust(X,kmeans,method="wss",k.max=10)
```



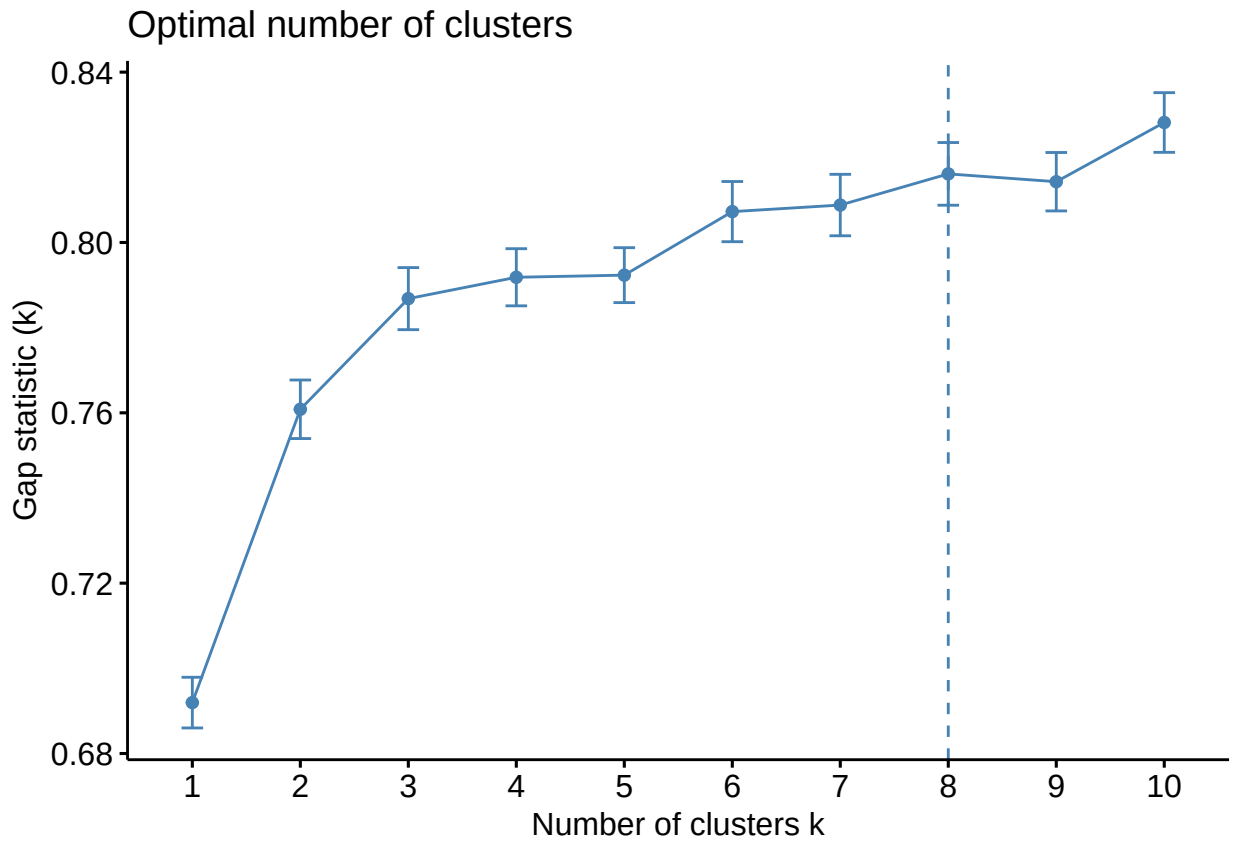

```
fviz_nbclust(X,kmeans,method="silhouette",k.max=10)
```




```
gap_stat <- clusGap(X,FUN=kmeans,K.max=10,B=100)
```

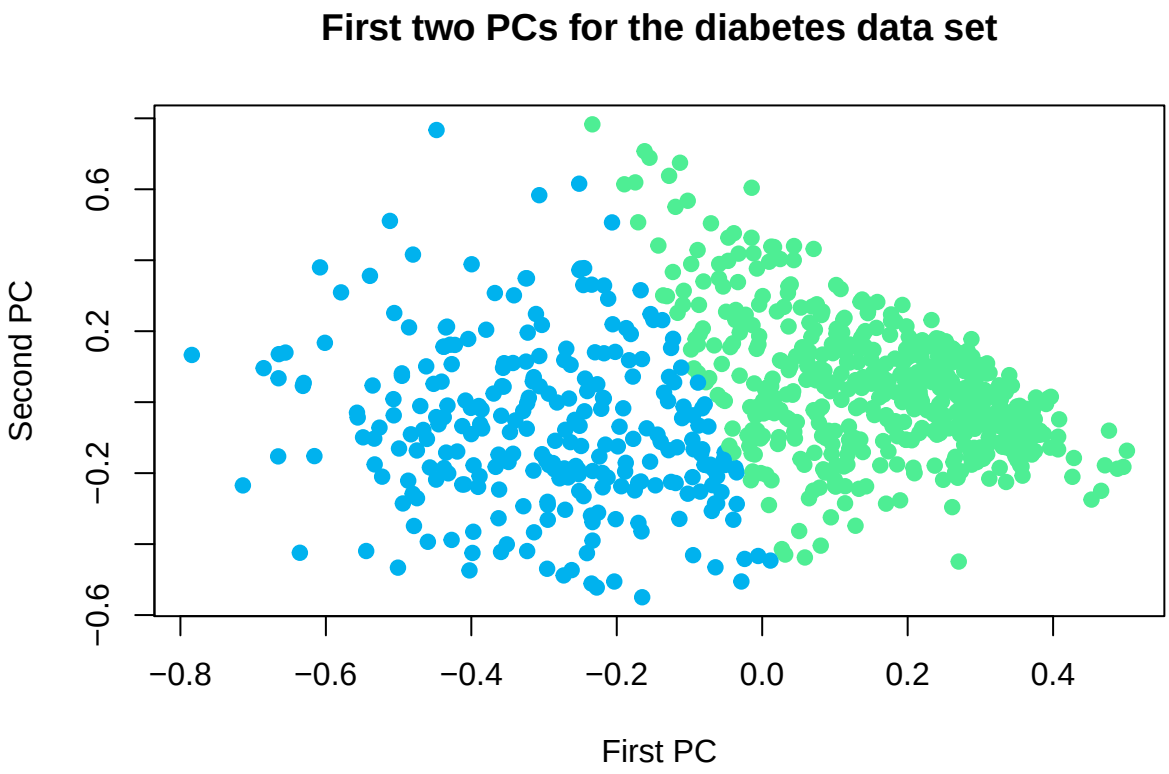
```
## Warning: did not converge in 10 iterations
```

```
fviz_gap_stat(gap_stat,linecolor="steelblue",maxSE=list(method="firstmax",SE.factor = 1))
```



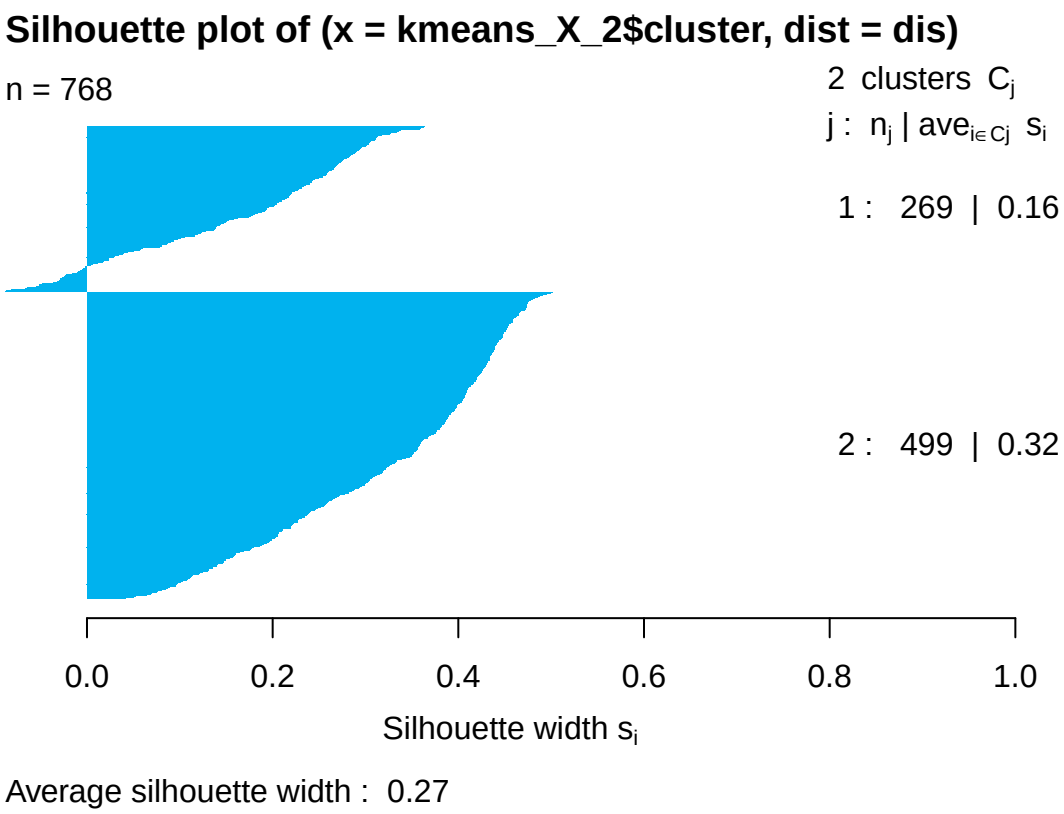
Whit wss we cant see a clear optimal number o cluste. However, average silhouette the best number of clustering is 2 and with gap is 6.For our problem is better to use k=2 because we are classifying our pacients in diabetics or not.We will check both only to see how the results are, but only with partitional algorithm because we doesnt matter to classify our data in 6 groups, only in 2. Now, I will check the two options. We will use means and medoids(PAM and clara). With two cluster , we can see the groups with the PCA

```
color_1 <- "deepskyblue2"
color_2 <- "seagreen2"
color_3 <- "orange2"
color_4 <- "darkorchid4"
color_5 <- "firebrick2"
color_6 <- "darkgreen"

kmeans_X_2 <- kmeans(X,centers=2,iter.max=1000,nstart=100)
colors_kmeans_X_2 <- c(color_1,color_2)[kmeans_X_2$cluster]
plot(Z,pch=19,col=colors_kmeans_X_2,main="First two PCs for the diabetes data set",xlab="First PC",ylab=
```



The two groups are well differentiated and are very similar to the classification of a person is diabetic or not. Now we check the silhouette.

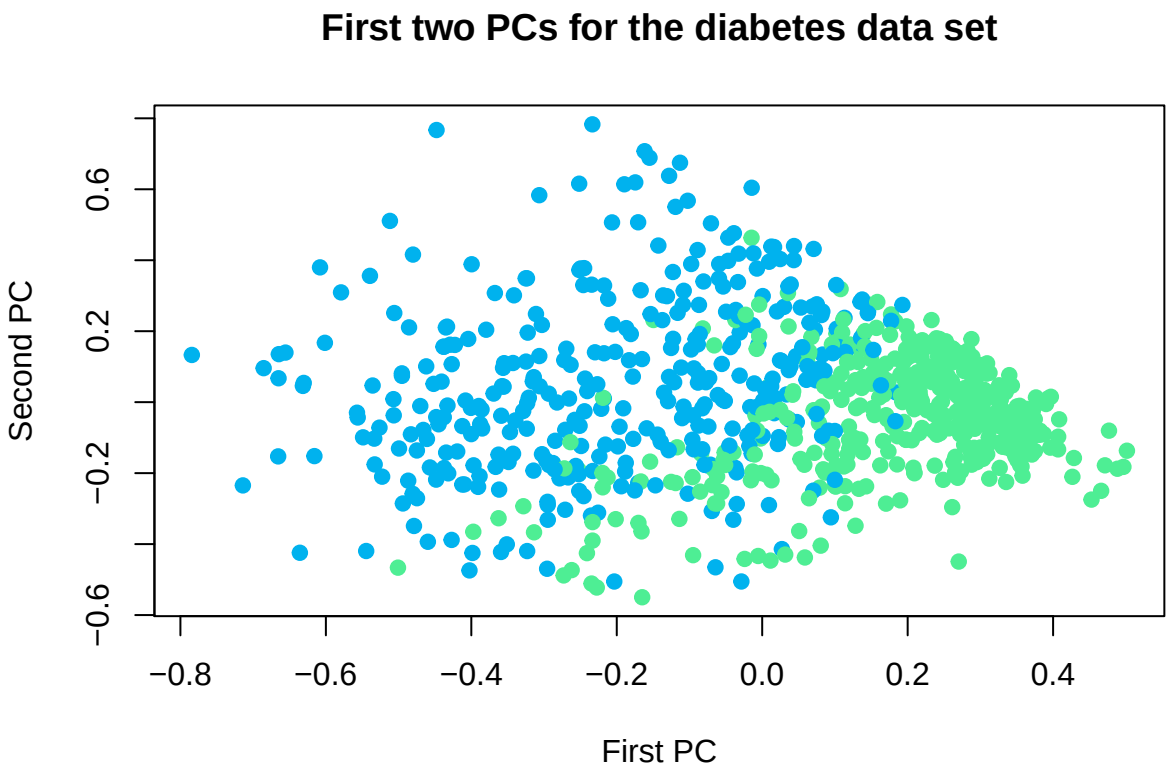
```
dis = dist(X)

sil_kmeans_X_2 <-silhouette(kmeans_X_2$cluster,dis)
plot(sil_kmeans_X_2, border=NA,col=color_1)
```



The results are good

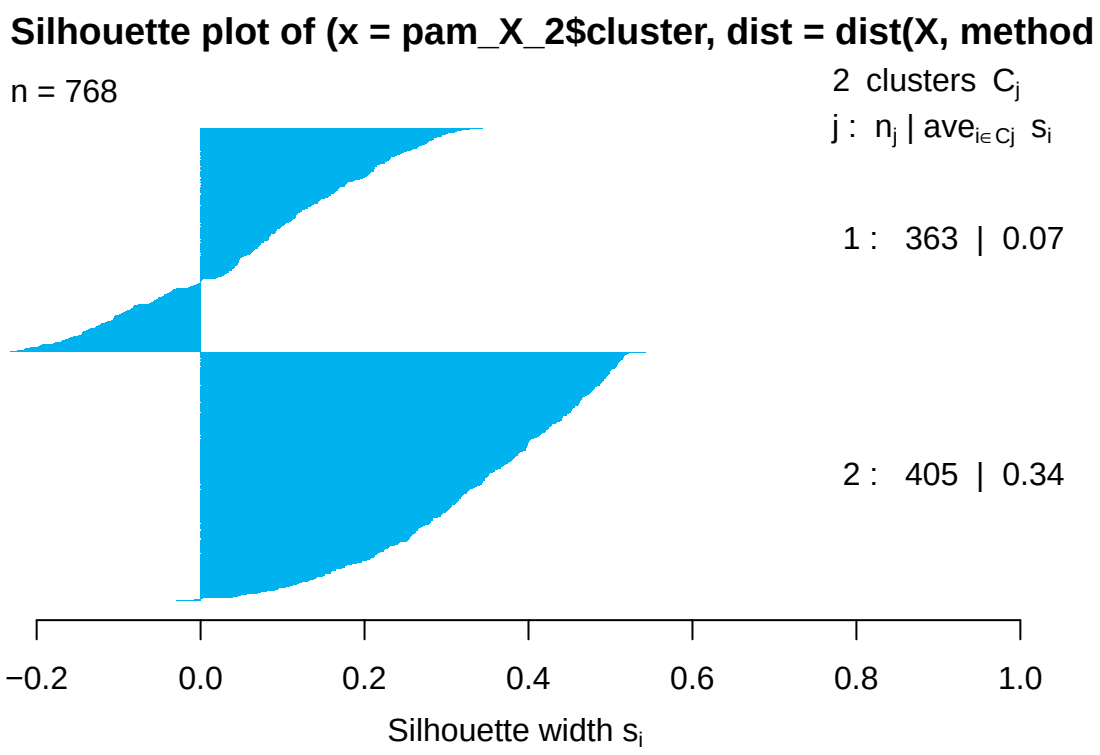
Then ,With k.medoids, we will use PAM algorithm

```
pam_X_2 <- pam(X,k=2,metric="manhattan",stand=FALSE)
colors_pam_X_2 <- c(color_1,color_2)[pam_X_2$cluster]
plot(Z,pch=19,col=colors_pam_X_2,main="First two PCs for the diabetes data set",xlab="First PC",ylab="S
```



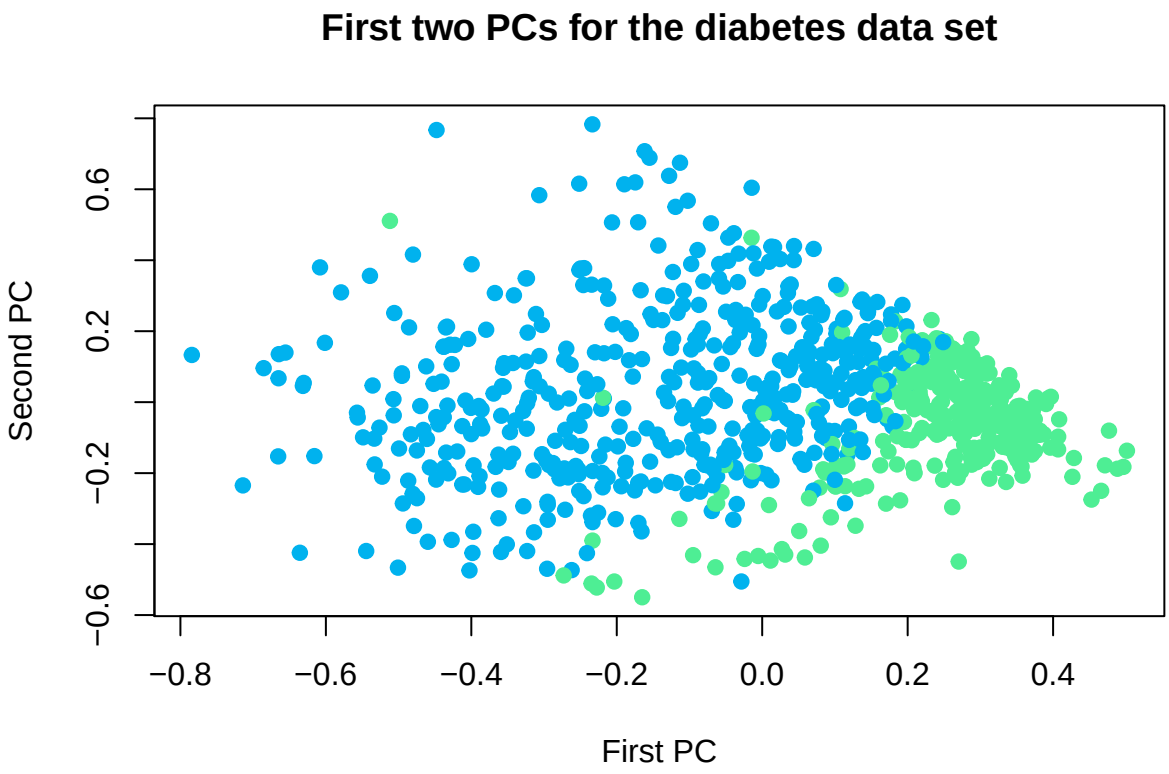
Its seems that the groups are more mixed than with mean, but it could be wrong because the Pca representation is only a projection.

```
sil_pam_X_2 <- silhouette(pam_X_2$cluster,dist(X,method="manhattan"))
plot(sil_pam_X_2,col=color_1, border=NA)
```



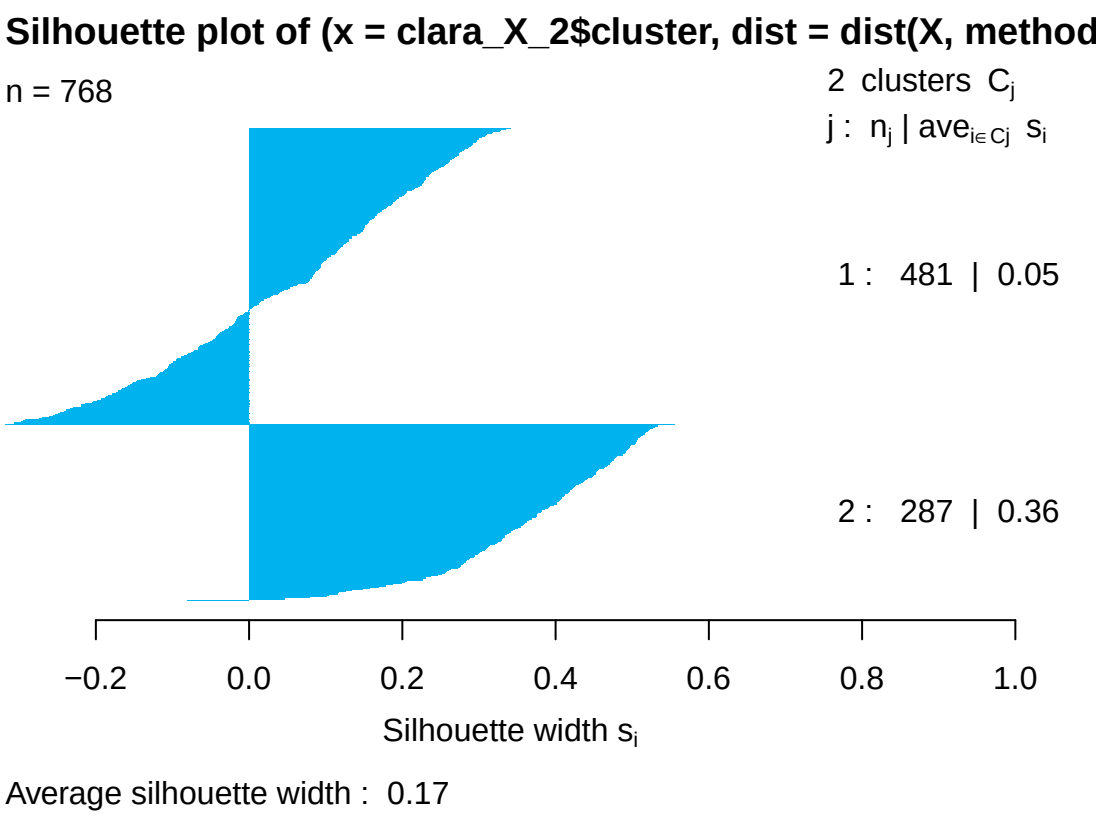
We can see that the results are slightly worse. Now we use CLARA. Probably we dont have enough data nto feed the algothim

```
clara_X_2 <- clara(X,k=2,metric="manhattan",stand=FALSE)
colors_clara_X_2 <- c(color_1,color_2)[clara_X_2$cluster]
plot(Z,pch=19,col=colors_clara_X_2,main="First two PCs for the diabetes data set",xlab="First PC",ylab=
```



The groups are very different. There are less green points than in other applications.

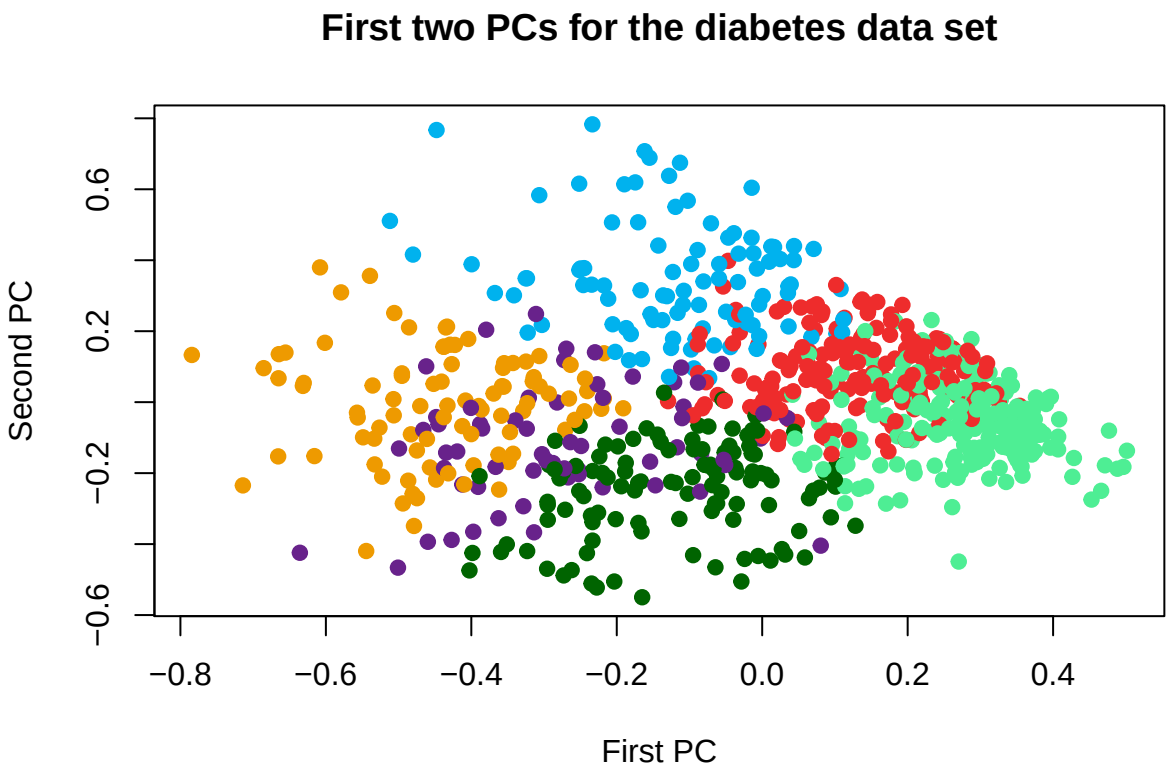
```
sil_clara_X_2 <- silhouette(clara_X_2$cluster,dist(X,method="manhattan"))
plot(sil_clara_X_2,col=color_1, border=NA)
```



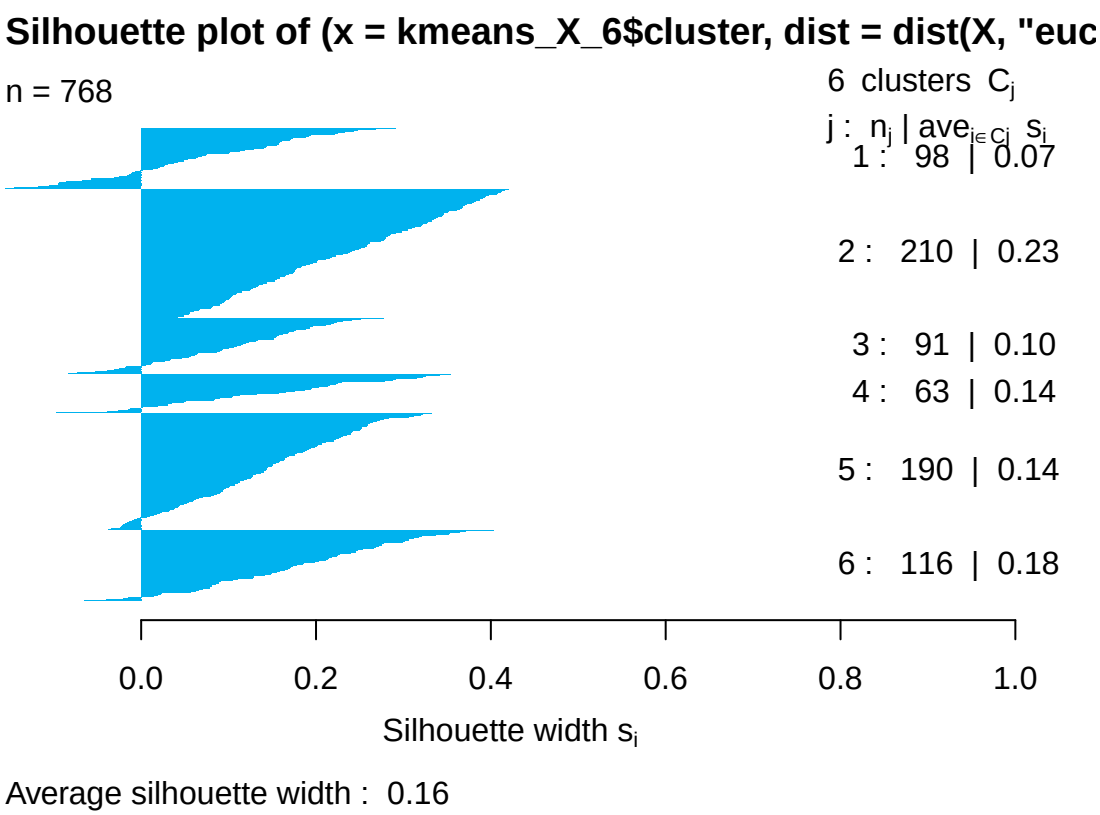
The average silhouette is lower .

We continue with 3 folds.

```
kmeans_X_6 <- kmeans(X,centers=6,iter.max=1000,nstart=100)
colors_kmeans_X_6 <- c(color_1,color_2,color_3,color_4,color_5,color_6)[kmeans_X_6$cluster]
plot(Z,pch=19,col=colors_kmeans_X_6,main="First two PCs for the diabetes data set",xlab="First PC",ylab=
```

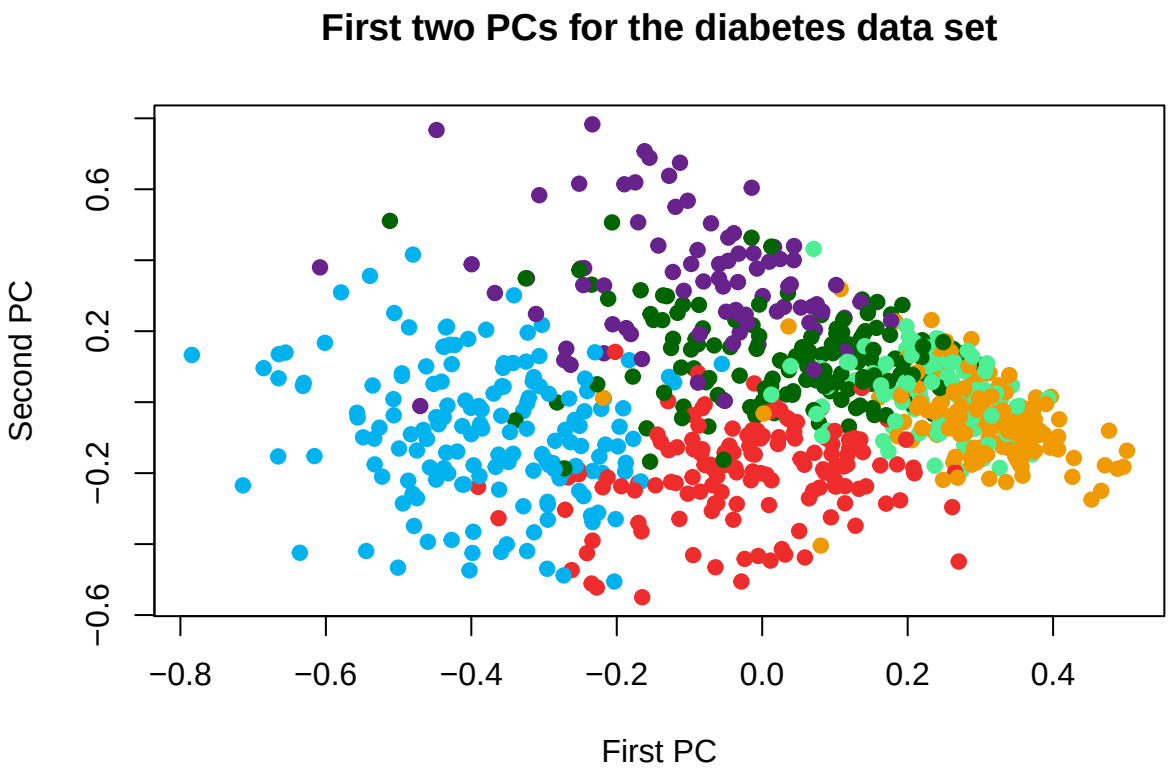



```
sil_kmeans_X_6 <- silhouette(kmeans_X_6$cluster,dist(X,"euclidean"))
plot(sil_kmeans_X_6,col=color_1, border=NA)
```



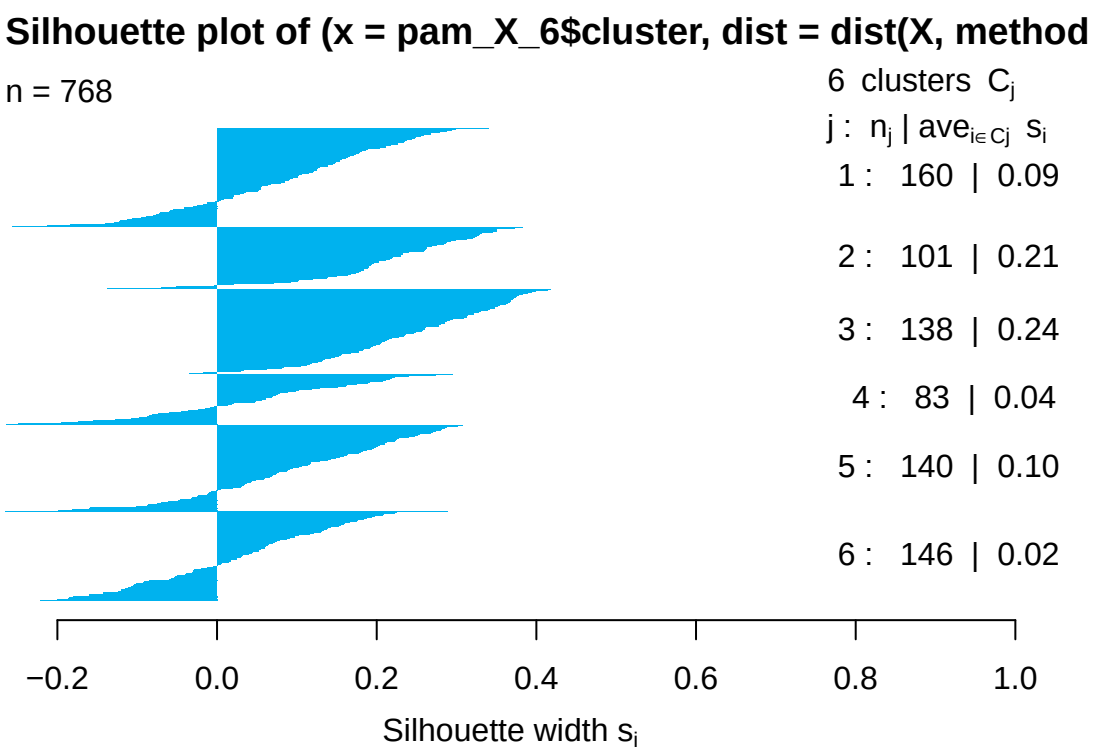
The average silhouette is lower. Its indicates that with 6 cluster the performance is worse. Now we try with PAM.

```
pam_X_6 <- pam(X,k=6,metric="manhattan",stand=FALSE)
colors_pam_X_6 <- c(color_1,color_2,color_3,color_4,color_5,color_6)[pam_X_6$cluster]
plot(Z,pch=19,col=colors_pam_X_6,main="First two PCs for the diabetes data set",xlab="First PC",ylab="S
```



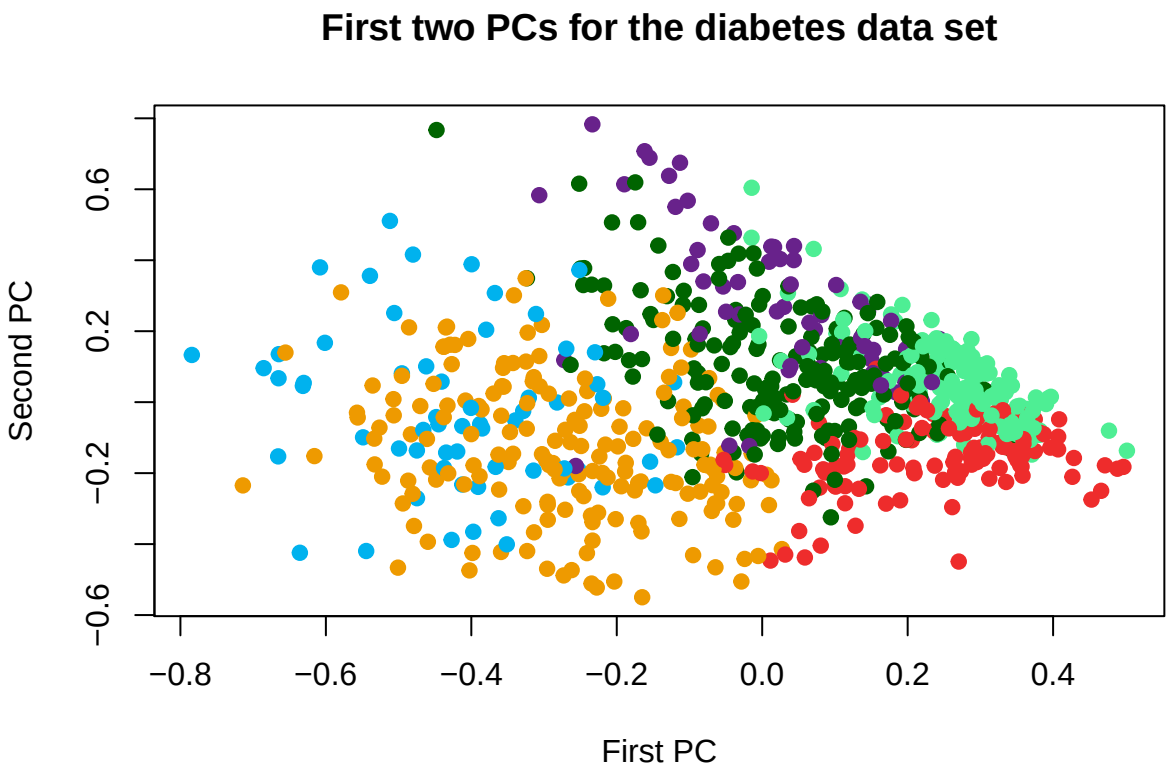
Its seems that is slightly better with PAM. Now we check the silhouttes

```
sil_pam_X_6 <- silhouette(pam_X_6$cluster,dist(X,method="manhattan"))
plot(sil_pam_X_6,col=color_1, border=NA)
```

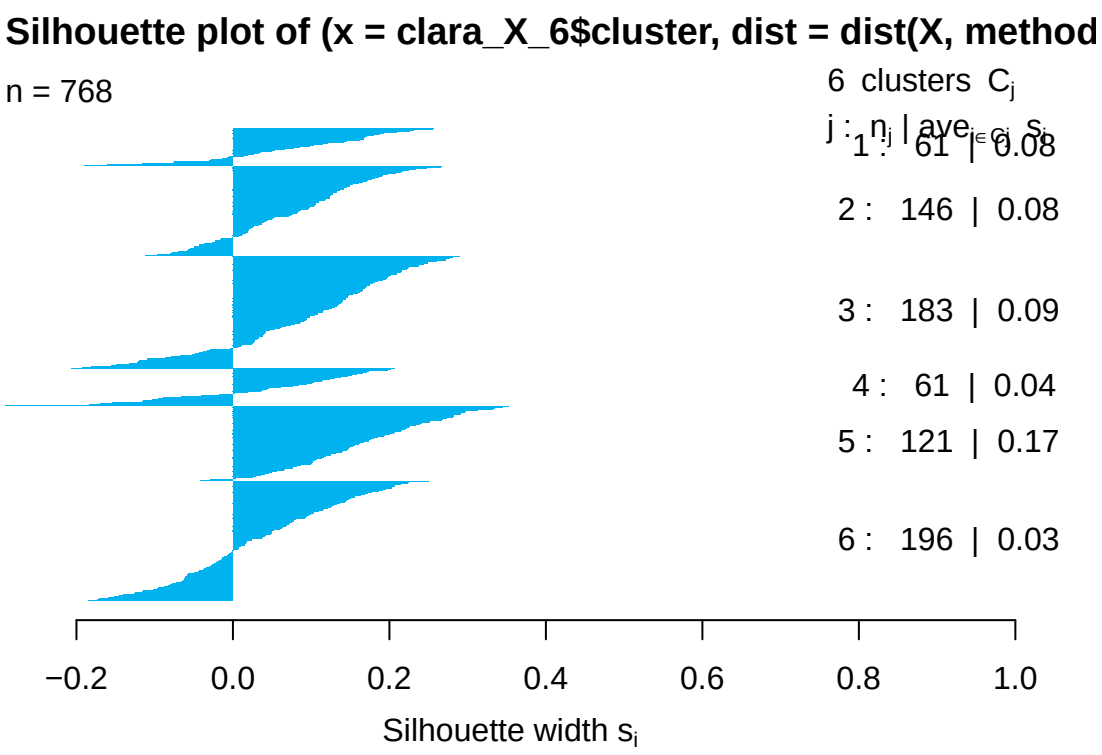


Average silhouette width : 0.12

```
clara_X_6 <- clara(X,k=6,metric="manhattan",stand=FALSE)
colors_clara_X_6 <- c(color_1,color_2,color_3,color_4,color_5,color_6)[clara_X_6$cluster]
plot(Z,pch=19,col=colors_clara_X_6,main="First two PCs for the diabetes data set",xlab="First PC",ylab=
```




```
sil_clara_X_6 <- silhouette(clara_X_6$cluster,dist(X,method="manhattan"))
plot(sil_clara_X_6,col=color_1, border=NA)
```



For our application, the best partitional clustering is k-means with two clusters.

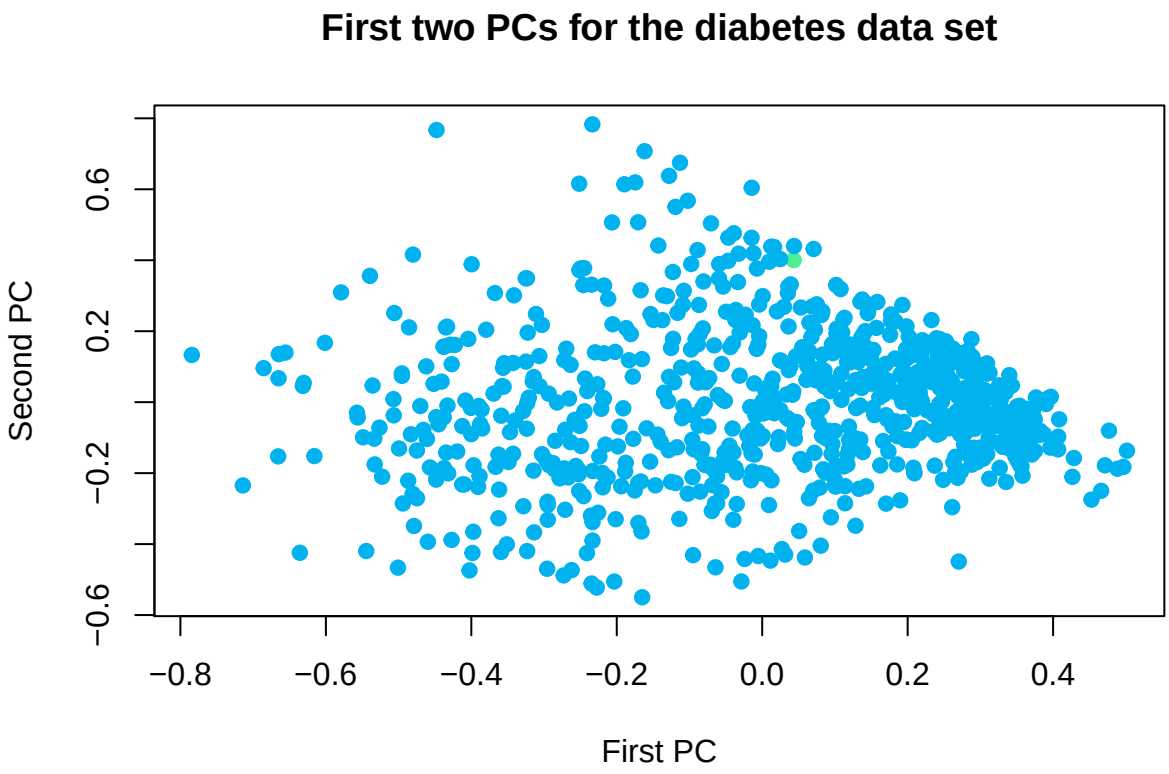
Now we try hierarchical clustering methods

Agglomerative clustering method is a kind of hierarchical clustering methods in which for each observation we make a cluster and we try to reduce the number of gruop joined gruops. We have four linkage methods, so we try four difference agglomerative clusters. We start with single linkage, where we compute the distance between a new cluster and all the others with the minimum between the new cluster and each of merge clusters We dont make the dendogam because there are many observations to do that

```
man_dist_X <- daisy(X,metric="manhattan",stand=FALSE)
single_X  <- hclust(man_dist_X,method="single")
cl_single_X <- cutree(single_X,6)
table(cl_single_X)
```

```
## cl_single_X
##   1   2   3   4   5   6
## 763   1   1   1   1   1
```

```
colors_single_X <- c(color_1,color_2)[cl_single_X]
plot(Z,pch=19,col=colors_single_X,main="First two PCs for the diabetes data set",xlab="First PC",ylab="
```

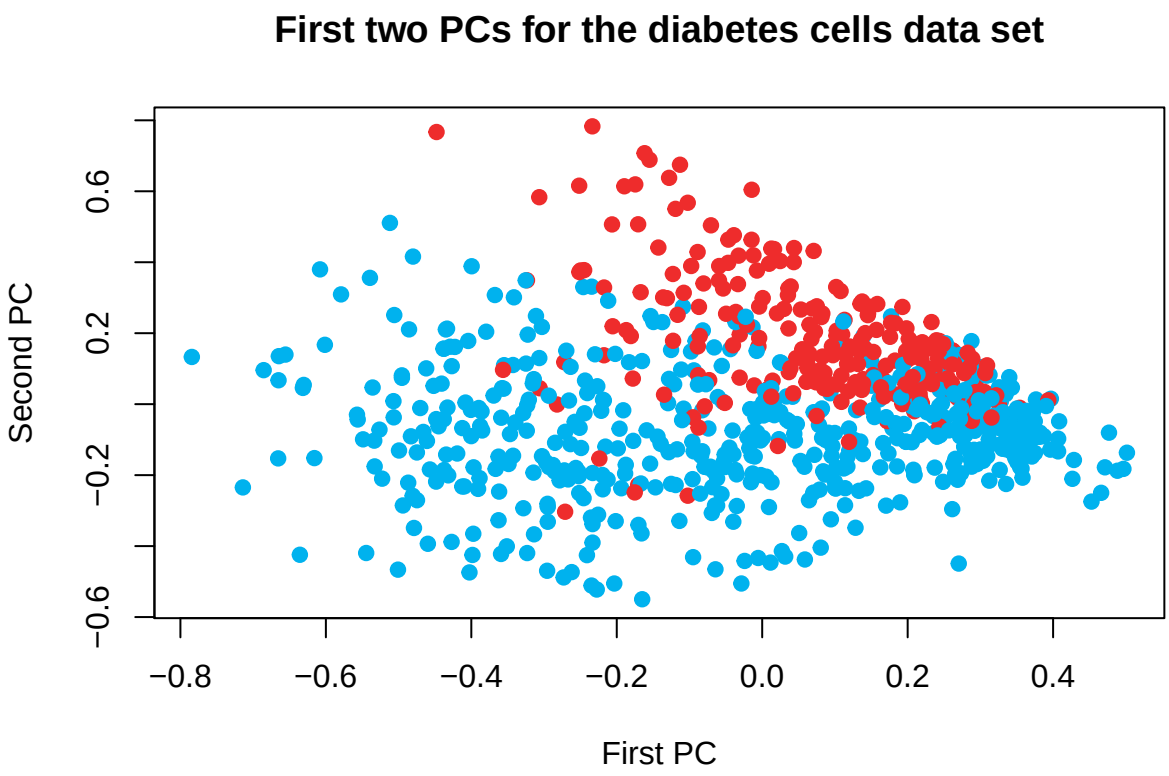


The results are awful. Lets we try with Complete linkage

```
complete_X <- hclust(man_dist_X,method="complete")
cl_complete_X <- cutree(complete_X,2)
table(cl_complete_X)
```

```
## cl_complete_X
##   1   2
## 519 249
```

```
colors_complete_X <- c(color_1,color_5)[cl_complete_X]
plot(Z,pch=19,col=colors_complete_X,main="First two PCs for the diabetes cells data set",xlab="First PC
```

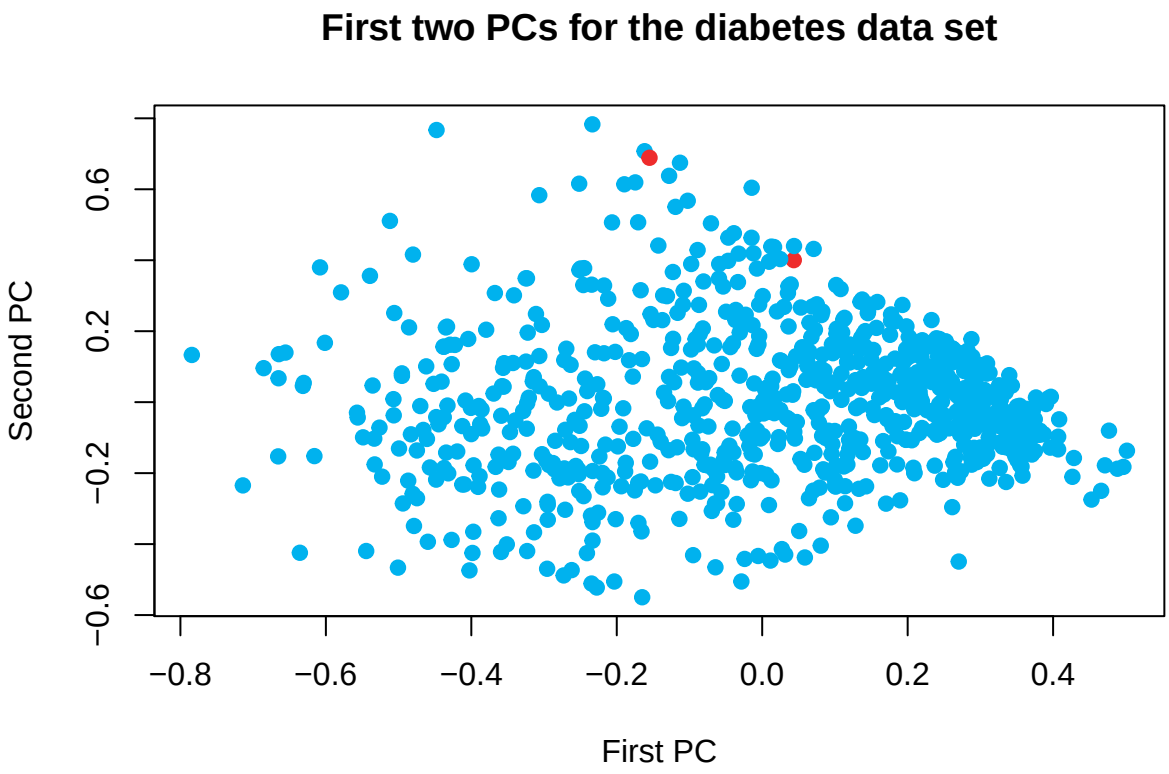


The results are also awfull.Then we try Average linkage:

```
average_X <- hclust(man_dist_X,method="average")
cl_average_X <- cutree(average_X,2)
table(cl_average_X)
```

```
## cl_average_X
##   1   2
## 766   2
```

```
colors_average_X <- c(color_1,color_5)[cl_average_X]
plot(Z,pch=19,col=colors_average_X,main="First two PCs for the diabetes data set",xlab="First PC",ylab=
```



The results are still very bad. Finally we try ward linkage

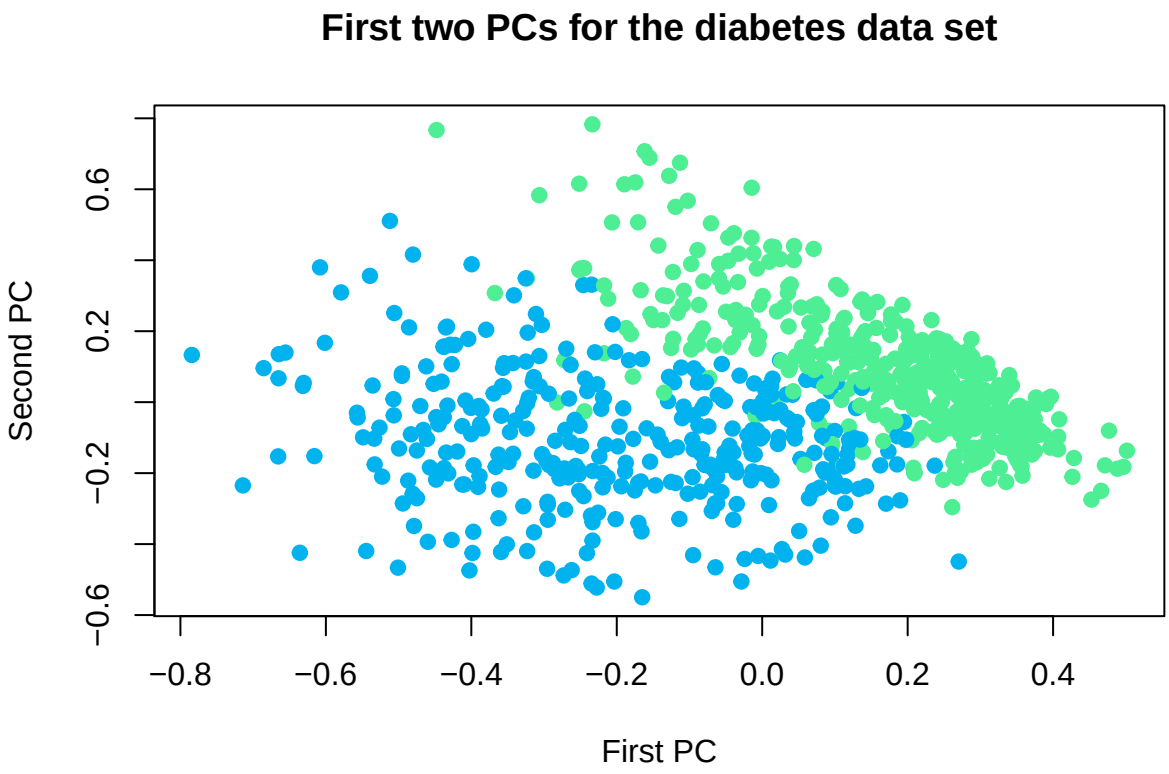
```
ward_X <- hclust(man_dist_X,method="ward")
```

```
## The "ward" method has been renamed to "ward.D"; note new "ward.D2"
```

```
cl_ward_X <- cutree(ward_X,2)
table(cl_ward_X)
```

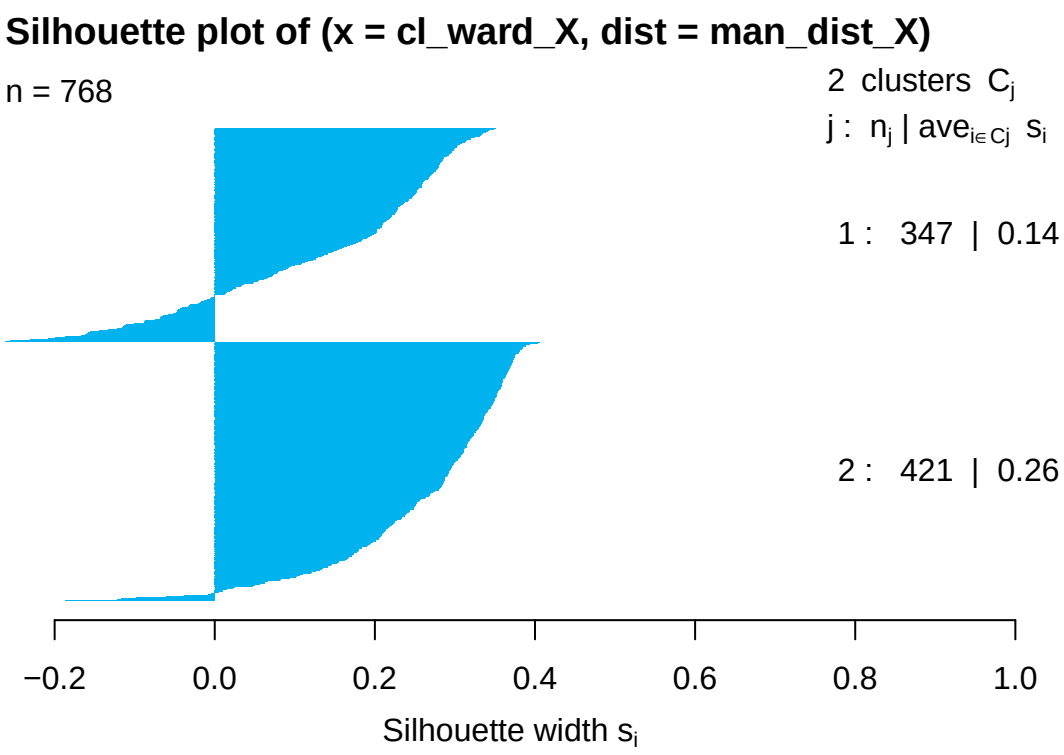
```
## cl_ward_X
##   1   2
## 347 421
```

```
colors_ward_X <- c(color_1,color_2)[cl_ward_X]
plot(Z,pch=19,col=colors_ward_X,main="First two PCs for the diabetes data set",xlab="First PC",ylab="Se
```



This is the best agglomerative hierarchical clustering algorithm by far. The shilouttes are

```
sil_ward_X <- silhouette(cl_ward_X,man_dist_X)
plot(sil_ward_X,col=color_1, border=NA)
```



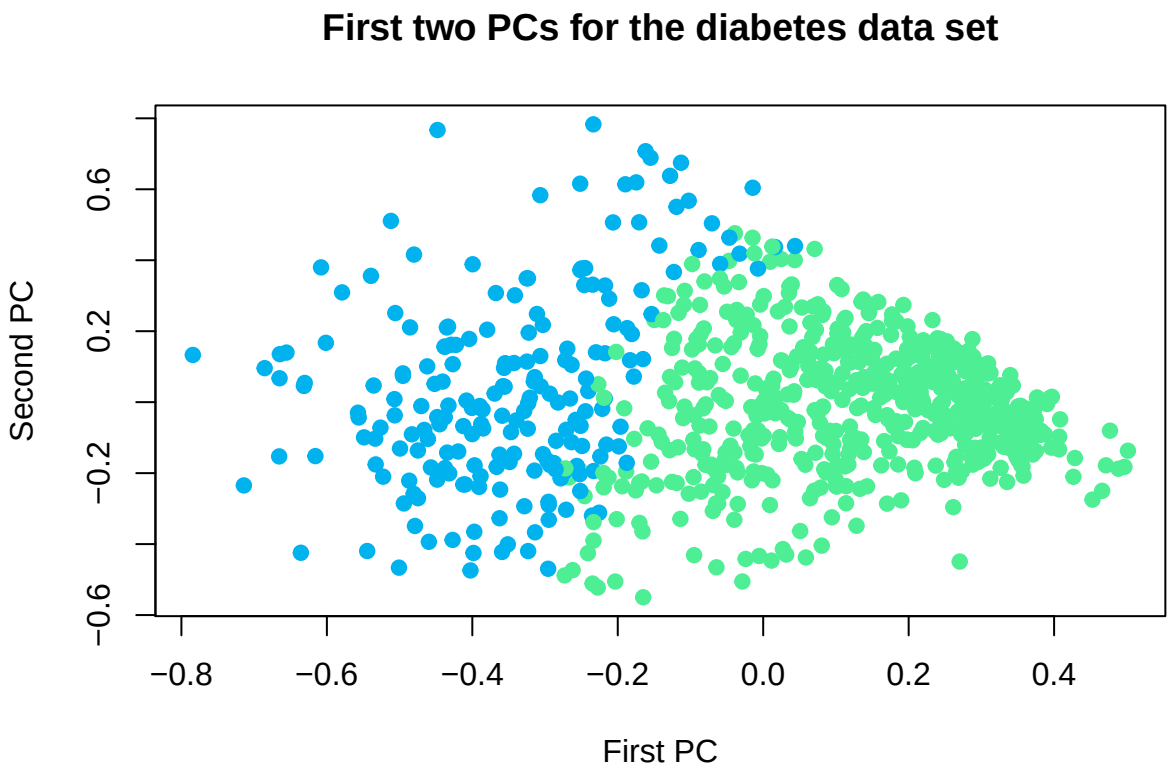
The shiloutte average is bigger than CLARA, but it also smaller than PAM and kmeans.

Now we will work with divisive algorithms in qhich we initally have all observations in one cluster and then and each step divide into two parts the cluster. For that, we will use the most popular algorithm, DIANA.

```
diana_X <- diana(X,metric="manhattan")
cl_diana_X <- cutree(diana_X,2)
table(cl_diana_X)
```

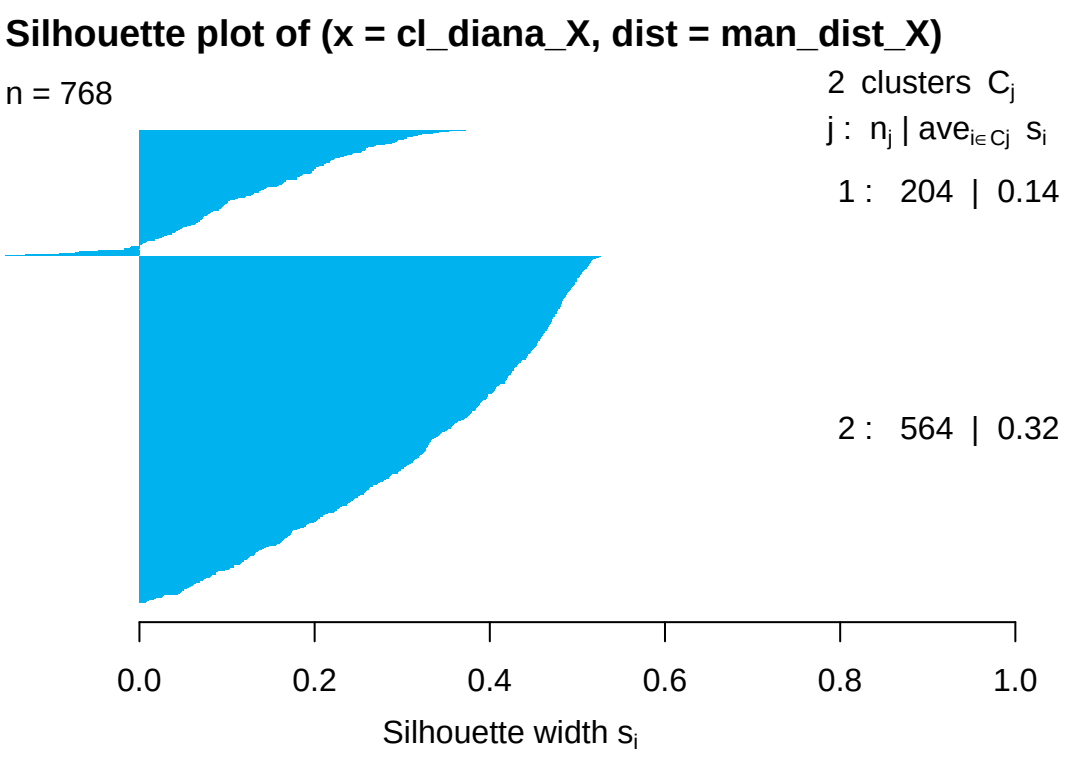
```
## cl_diana_X
##   1   2
## 204 564
```

```
colors_diana_X <- c(color_1,color_2)[cl_diana_X]
plot(Z,pch=19,col=colors_diana_X,main="First two PCs for the diabetes data set",xlab="First PC",ylab="S
```



Its seems a goog perfomace so we wiil check the silhouette

```
sil_diana_X <- silhouette(cl_diana_X,man_dist_X)
plot(sil_diana_X,col=color_1,border=NA)
```

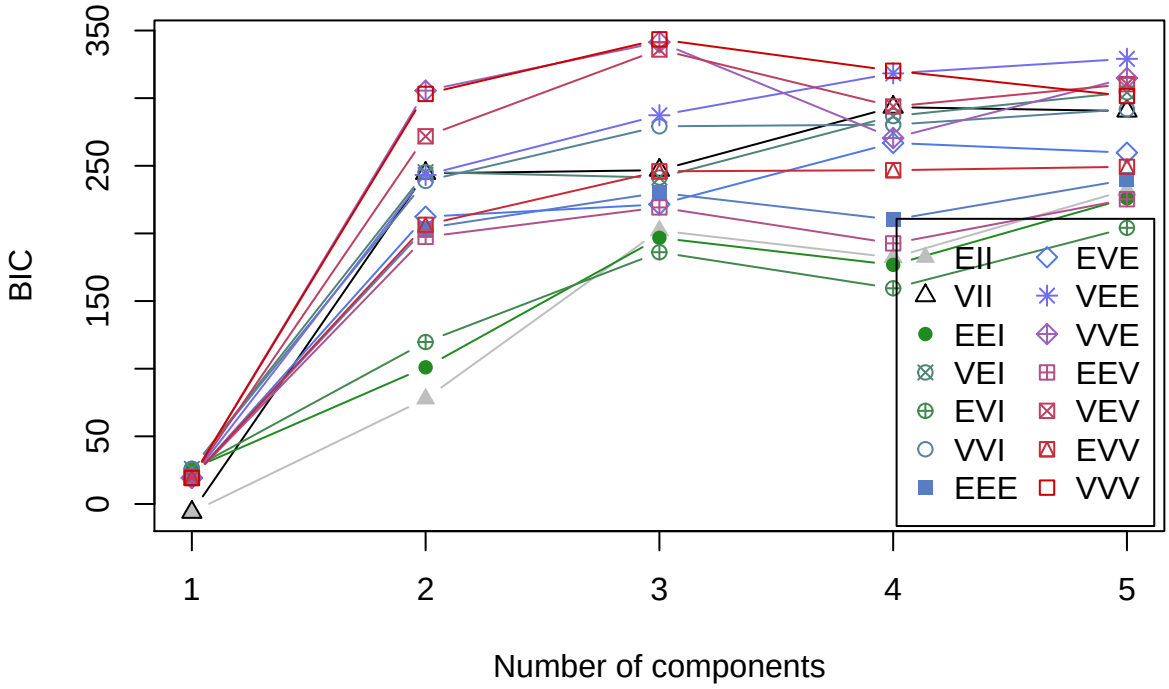


We have the best average silhouette with kmeans. Finally, we will check the performance of model-bases clustering methods.First of all, we will check the best assumption of the Mclust

```
BIC_X <- mclustBIC(Z,G=1:5)
BIC_X
```

```
## Bayesian Information Criterion (BIC):
##          EII        VII      EEI      VEI      EVI      VVI       EEE       EVE
## 1  -6.110824  -6.110824  25.9162  25.9162  25.9162  25.9162  19.27241  19.27241
## 2  77.753729 244.393988 101.0145 245.3506 119.7063 238.8056 203.89673 212.36004
## 3 202.129893 246.732798 196.7428 241.2908 186.0949 279.1930 230.10930 221.48414
## 4 182.159207 293.343369 176.7631 286.7201 159.4428 280.3669 210.17994 266.88400
## 5 232.092917 290.478976 225.9213 304.2685 204.1606 291.8112 239.61446 259.69390
##          VEE       VVE       EEV       VEV       EVV       VVV
## 1  19.27241  19.27241  19.27241  19.27241  19.27241  19.27241
## 2 243.17611 305.54881 197.43700 271.78815 206.34024 303.11803
## 3 287.35757 341.37586 219.21999 335.96086 245.90882 343.46546
## 4 318.30715 270.48955 192.64277 293.82432 246.74748 320.24994
## 5 329.05469 314.86968 225.25805 310.32365 249.25769 301.58288
##
## Top 3 models based on the BIC criterion:
##    VVV,3    VVE,3    VEV,3
## 343.4655 341.3759 335.9609
```

```
plot(BIC_X)
```

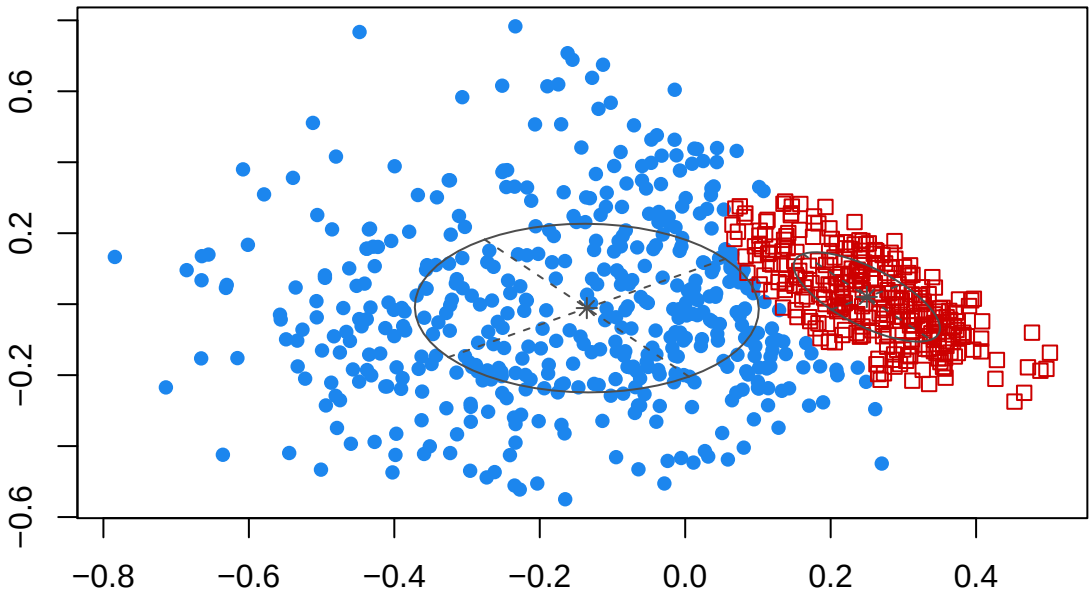


The best performace are with three clusters and VVV.Nonetheless, we want to classify in two groups and the best methods with that is VVE(ellipsoidal, equal orientation)

```
BIC_X <- mclustBIC(Z,G=1:2)
Mclust_X <- Mclust(Z,x=BIC_X)
summary(Mclust_X)
```

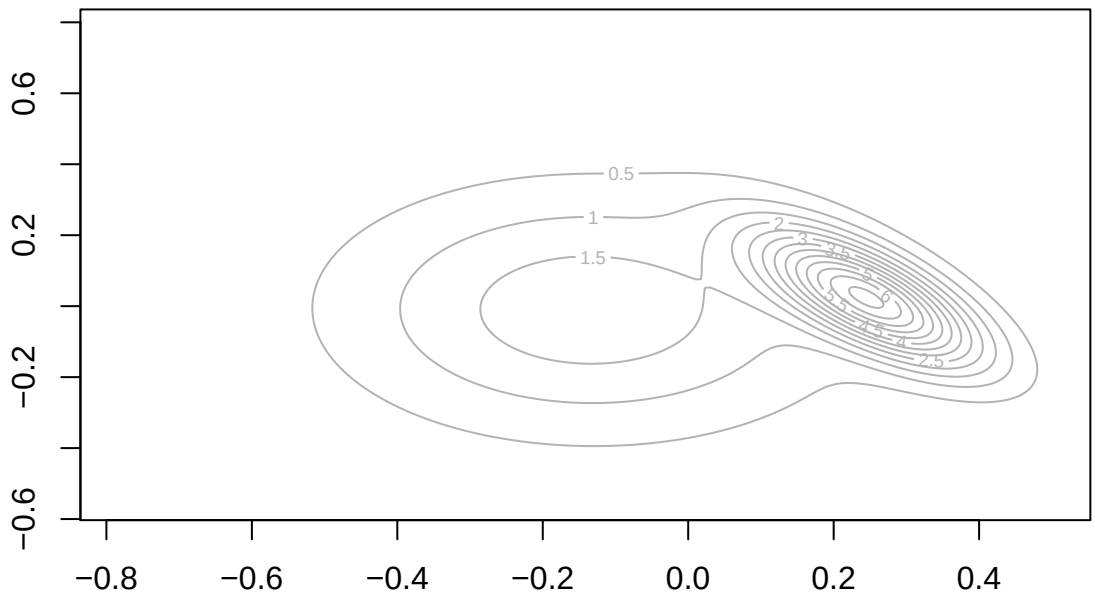
```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VVE (ellipsoidal, equal orientation) model with 2 components:
##
##  log-likelihood   n df      BIC      ICL
##        185.9934 768 10 305.5488 151.7457
##
## Clustering table:
##   1   2
## 473 295
```

```
plot(Mclust_X,what="classification")
```



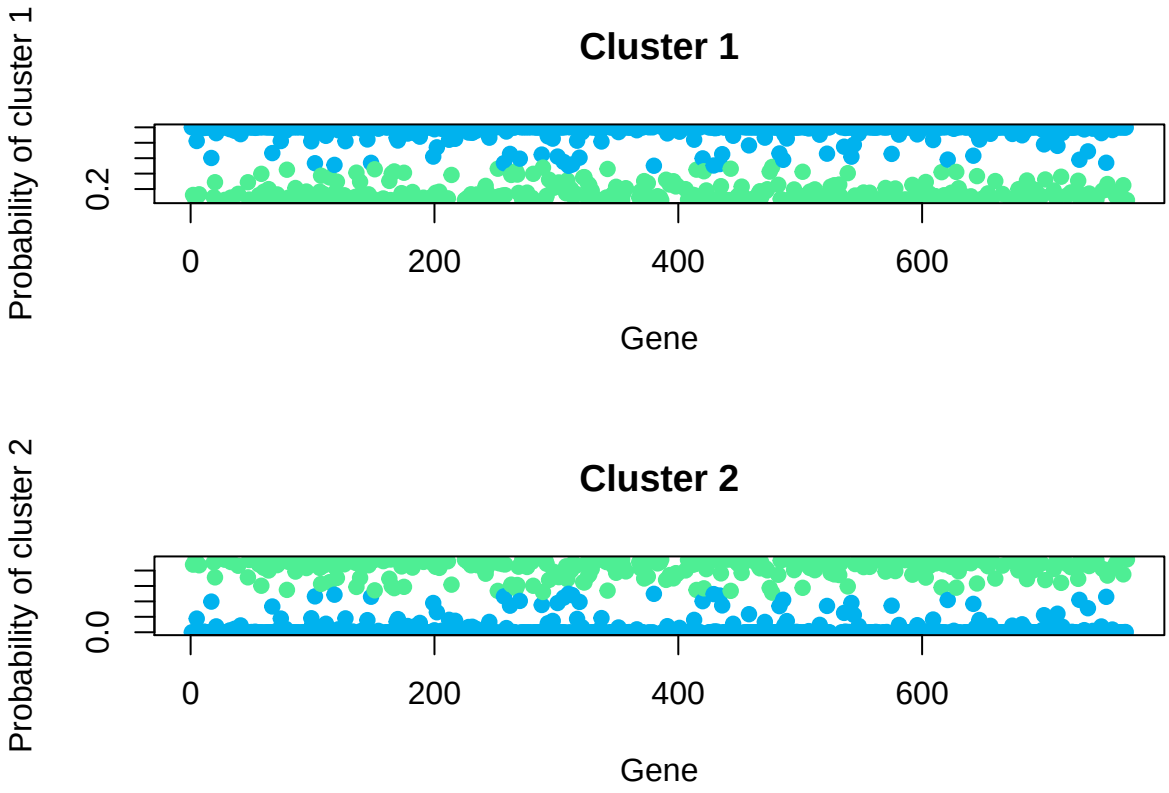
Its seem worse than hieretical and partitional clustering The estimed densities are:

```
plot(Mclust_X,what="density")
```



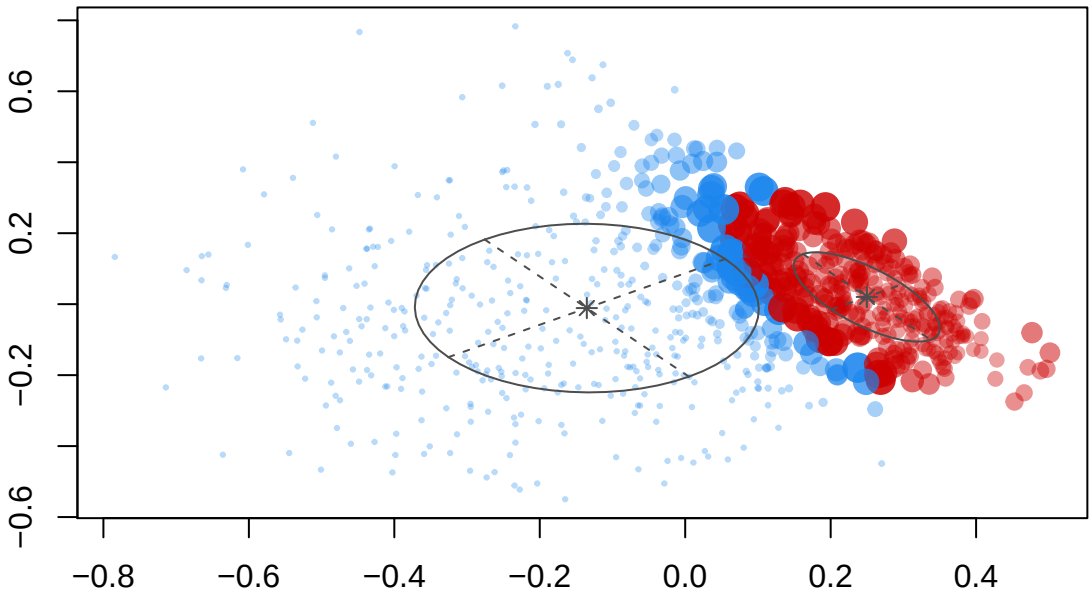
Now we play the estimated probabilities

```
colors_Mclust_X <- c(color_1,color_2)[Mclust_X$classification]
par(mfrow=c(2,1))
plot(1:n,Mclust_X$z[,1],pch=19,col=colors_Mclust_X,main="Cluster 1",xlab="Gene",ylab="Probability of clu
plot(1:n,Mclust_X$z[,2],pch=19,col=colors_Mclust_X,main="Cluster 2",xlab="Gene",ylab="Probability of clu
```



Finally we plot tge points with uncertain

```
par(mfrow=c(1,1))
plot(Mclust_X,what="uncertainty")
```



THere are many ponits with uncertain so is better not to use this tools for our problem.

Factor Analysis

The aim of the factor analysis is to explain the outcome of our variables from the matrix using fewer variables, that we will call "factors". Ideally, all the information of the women from our dataset can be explained by a small number of factors. We will interpret this factors as latent (unobserver) shared characteristics of the observed data.

Now, let´s focus on finding the latent variables in our data (that is, the variables that we can not meassure), their relationship, their relationship with the variables we indeed can observe, and the underlying structure of the variables we can meassure. In order to do this, we will use three methods: first we will apply the principal component factor analysis, then we will refine it a bit and apply the principal factor analysis, and finally we will apply maximum likelihood estimation to check if we get a similar result.

Let´s clarify that, as with PCA and Cluster Analysism, we haven´t found two groups that locate very far away one from the other (although with a specific techique we found the clusters), we will perform the factor analysis in the whole dataset, and not in each group separately.

```
library("psych")
```

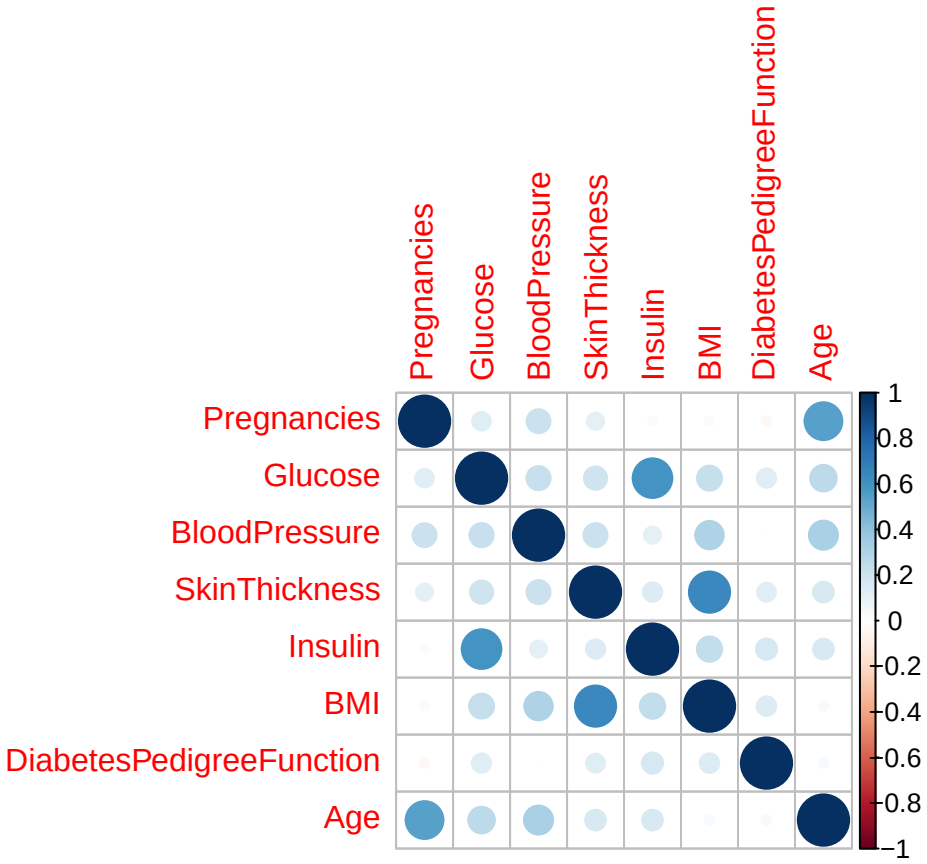
Principal Component Factor Analysis

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

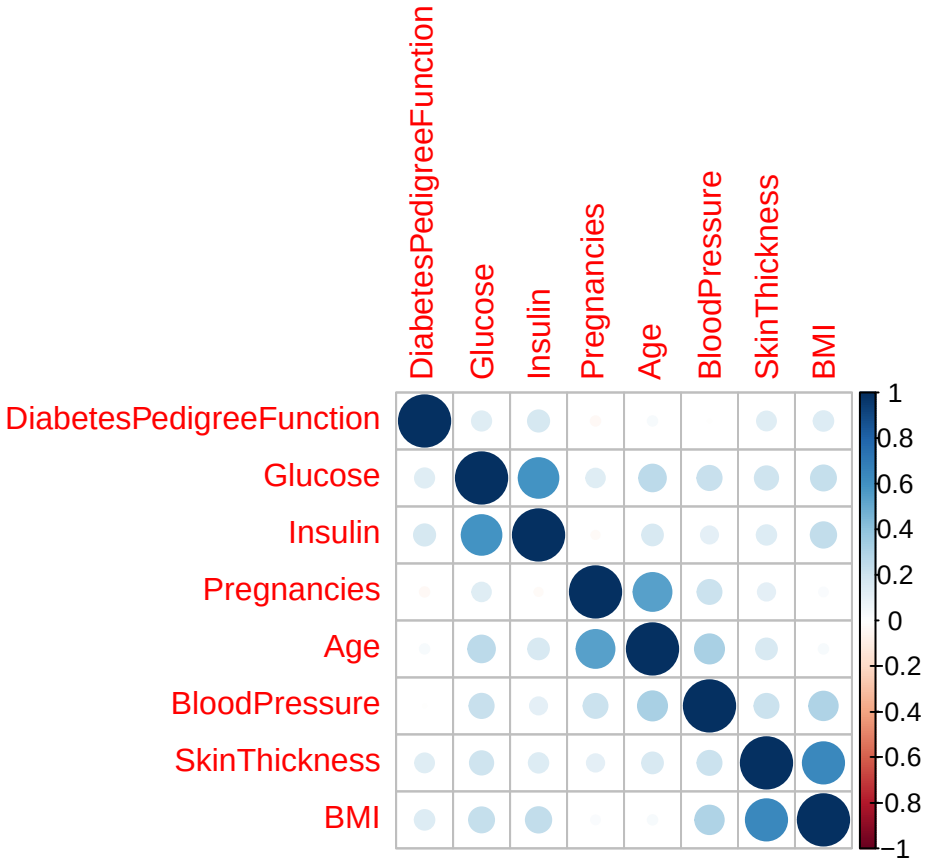
```
##
## Attaching package: 'psych'

## The following object is masked from 'package:mclust':
##
##     sim

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

Let´s first go back to the correlation plot of our observable variables.

```
Y <- scale(X)
corrplot(cor(Y))
```




```
corrplot(cor(Y),order="hclust")
```



As we have seen, there are some variables that hold correlations. This is interesting, because now that we will try to find the factors that describe our dataset, we have to see how these variables will be grouped. so the fact that there are some groups of variables that are correlated may suggest us that there may be a factor structure, let´s check if this happens or not.

We will work with our principal component analysis for this part. We said that through three different methods, we would finally stick with 3 principal components.

```
r <- 3
```

Let´s first estimate the matrix M (the loading matrix) and use the varimax rotation, as seen in class. The interpretation of the loadings would be very simple if the variables could be split into disjoint sets, eah being associated with one factor. A very well known analytical algorithm to rotate the loadings is given by the varimax rotation method. The idea of this method is to find the angle that maximizes the sum of variances of the squred loadings within each column of the matrix.

```
Y_pcs <- prcomp(Y)
M_pcfa <- Y_pcs$rotation[,1:r] %*% diag(Y_pcs$sdev[1:r])
M_pcfa
```

```
##                              [,1]       [,2]        [,3]
## Pregnancies             -0.3962853  0.7363533 -0.07291698
## Glucose                 -0.6769340 -0.1158364  0.52722977
## BloodPressure           -0.5587194  0.2452812 -0.23920003
## SkinThickness           -0.6389961 -0.2764416 -0.52792238
## Insulin                 -0.5747951 -0.3041489  0.60029219
```

```
## BMI                       -0.6513400 -0.4184555 -0.47214486
## DiabetesPedigreeFunction -0.2577060 -0.3387209  0.17444889
## Age                       -0.5522702  0.6534480  0.10956142
```

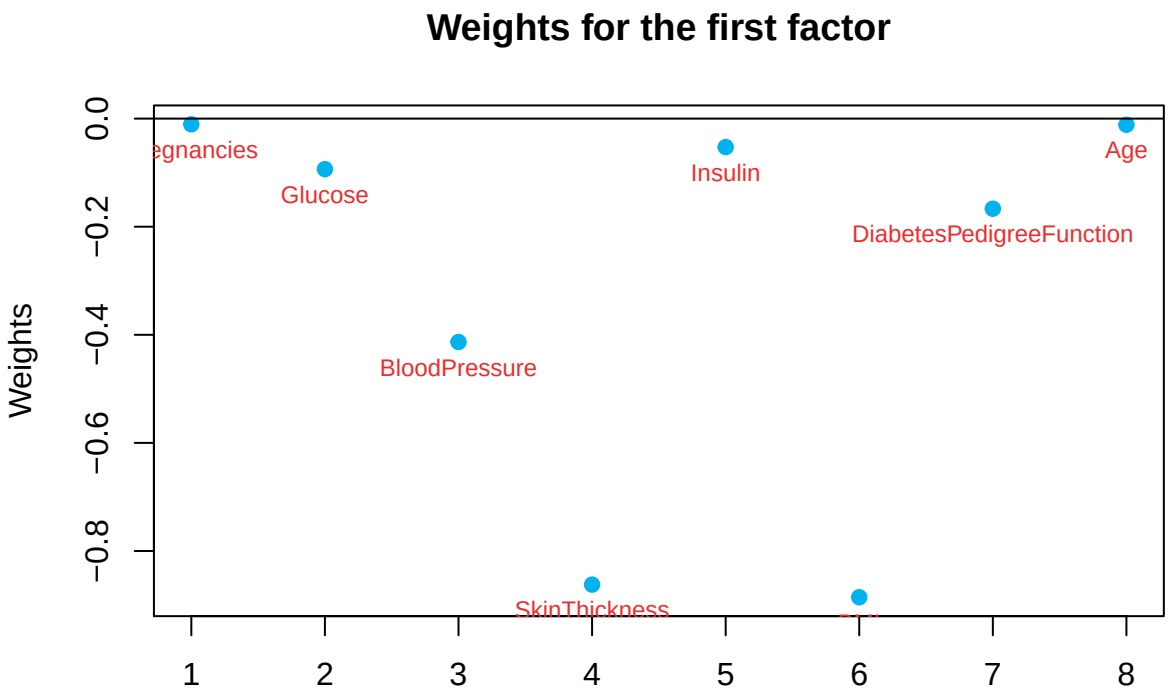
```
M_pcfa <- varimax(M_pcfa)
M_pcfa <- loadings(M_pcfa)[1:p,1:r]
M_pcfa
```

```
##                                [,1]        [,2]        [,3]
## Pregnancies              -0.01063286  0.83395493 -0.09476969
## Glucose                  -0.09352834  0.26515768  0.81888510
## BloodPressure            -0.41327737  0.50481920  0.06250021
## SkinThickness            -0.86216054  0.10549494  0.09480518
## Insulin                  -0.05266222  0.05164657  0.88193313
## BMI                      -0.88553447 -0.00781807  0.19503130
## DiabetesPedigreeFunction -0.16670132 -0.14828822  0.40224111
## Age                      -0.01151716  0.84772690  0.15882939
```

As we can see here, we have three factors, each with different weights from each variable. Let´s plot each factor, with the weights of the variable to see more clearly what charateristic of the population is each factor expressing.

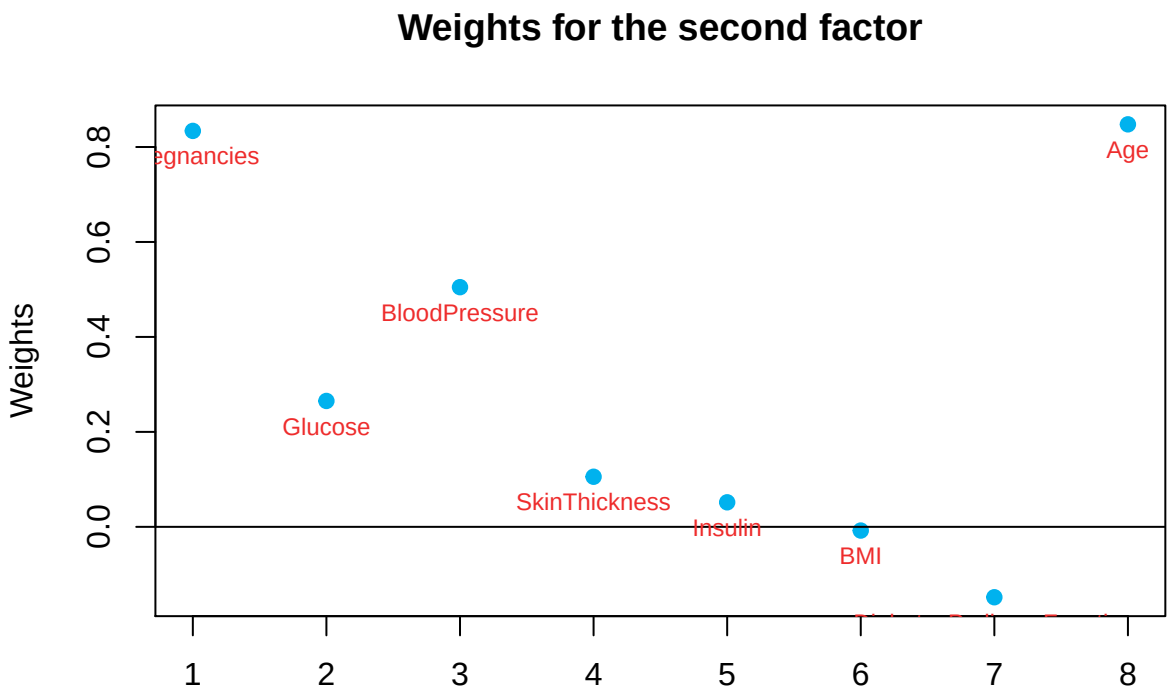
In contrast with the PCA, where the most important component is the one that maximizes the projected variance, here the most important factor in the analysis is the one that (after rotation) gives the maximal interpretation.

```
plot(1:p,M_pcfa[,1],pch=19,col=color_1,xlab="",ylab="Weights",main="Weights for the first factor")
abline(h=0)
text(1:p,M_pcfa[,1],labels=colnames(X),pos=1,col=color_5,cex=0.75)
```



Factor 1
As we can see in the plot, there are some variables close to zero, but there are others that have a "big" negative weight on the factor. These two variables are "skin thickness" and "BMI". This factor may be an (inverse) index of the patients "external signs", characteristics that can be seen by a doctor by just looking at the patient.

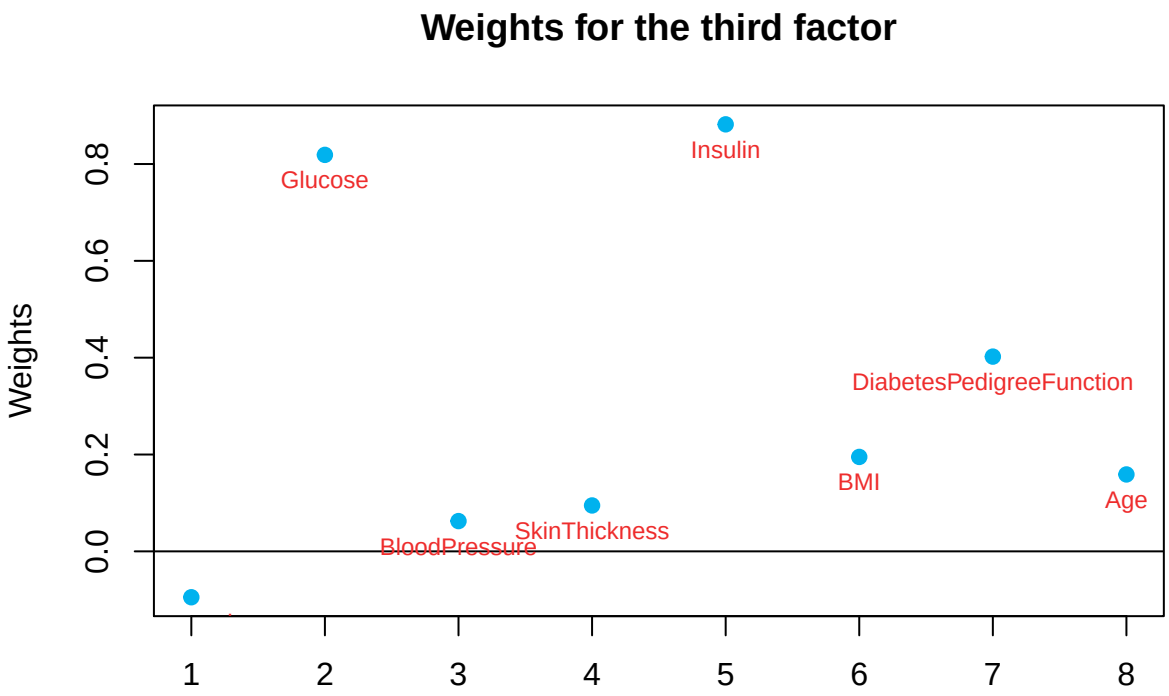
```
plot(1:p,M_pcfa[,2],pch=19,col=color_1,xlab="",ylab="Weights",main="Weights for the second factor")
abline(h=0)
text(1:p,M_pcfa[,2],labels=colnames(X),pos=1,col=color_5,cex=0.75)
```



Second Factor

Our second factor is very interesting, and seems to be explained spetially by variables like "pregnancies", "age" and have obviosuly more to do with the life cycle pr the chronology of the patient. It is obvious that one big characteristic of the data is the moment of the life of the women, and this factor is showing this hidden characteristic.

```
plot(1:p,M_pcfa[,3],pch=19,col=color_1,xlab="",ylab="Weights",main="Weights for the third factor")
abline(h=0)
text(1:p,M_pcfa[,3],labels=colnames(X),pos=1,col=color_5,cex=0.75)
```



Factor 3
Our third factor seems to be showing variables with positive and high weights such as "glucose", "insulin" and basically is showing us an index of of metabolism characteristics, related to the sugar in blood and the hormones that the person produces.

This first approach to factor analysis brought some very interesting results. What we can say is that there are three fundamentl factors in our data: one describing more external characteristics of the person (the inverse of a healthy looking person), second factor is showing characteristics related to life cycle and the moment in which the woman is in her life, and third factor is showing an index of metabolism (sugar and hormonal values).

Let´s now check the covariance matrix of the error.

```
Sigma_nu_pcfa <- diag(diag(cov(Y) - M_pcfa %*% t(M_pcfa)))
Sigma_nu_pcfa
```

```
##           [,1]     [,2]      [,3]     [,4]      [,5]      [,6]      [,7]
## [1,] 0.2954248 0.000000 0.0000000 0.000000 0.0000000 0.0000000 0.0000000
## [2,] 0.0000000 0.250371 0.0000000 0.000000 0.0000000 0.0000000 0.0000000
## [3,] 0.0000000 0.000000 0.5704531 0.000000 0.0000000 0.0000000 0.0000000
## [4,] 0.0000000 0.000000 0.0000000 0.236562 0.0000000 0.0000000 0.0000000
## [5,] 0.0000000 0.000000 0.0000000 0.000000 0.2167533 0.0000000 0.0000000
## [6,] 0.0000000 0.000000 0.0000000 0.000000 0.0000000 0.1777304 0.0000000
## [7,] 0.0000000 0.000000 0.0000000 0.000000 0.0000000 0.0000000 0.7884234
## [8,] 0.0000000 0.000000 0.0000000 0.000000 0.0000000 0.0000000 0.0000000
##           [,8]
## [1,] 0.0000000
## [2,] 0.0000000
```

```
## [3,] 0.0000000
## [4,] 0.0000000
## [5,] 0.0000000
## [6,] 0.0000000
## [7,] 0.0000000
## [8,] 0.2559997
```

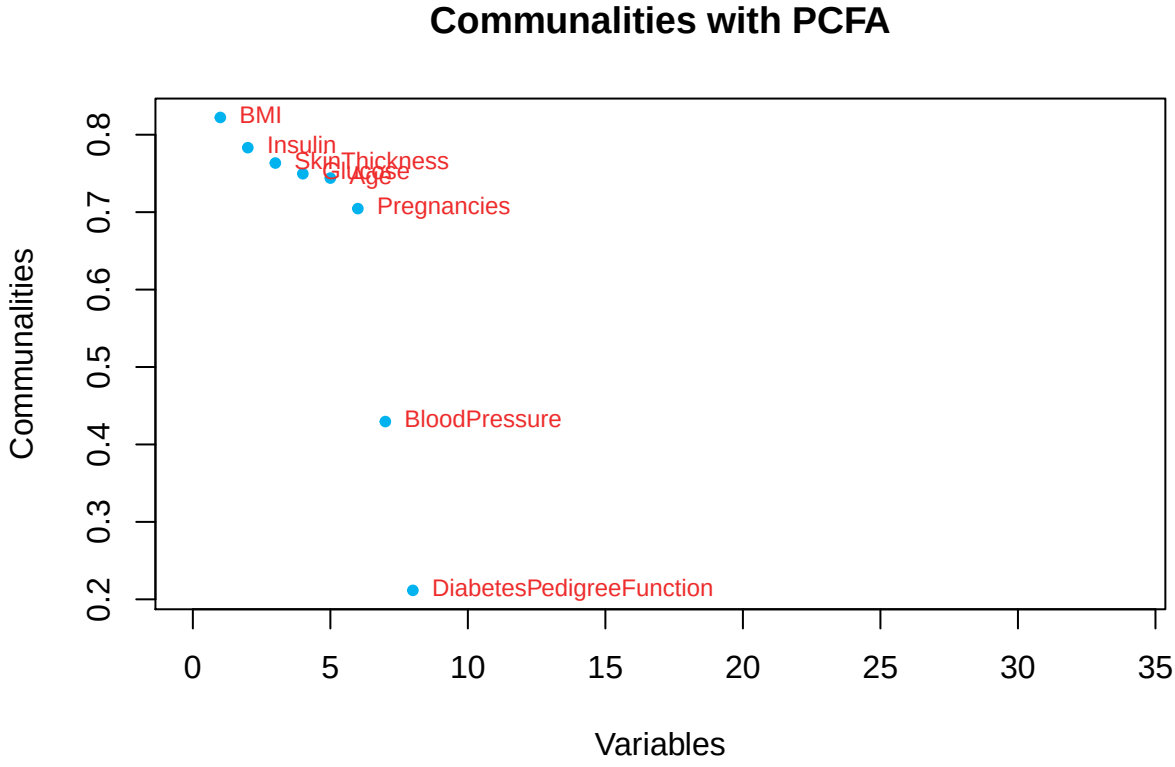
The matrix of erros will be later used to estimate the scores of the factors.

Let´s check now the communalities and uniqueness. First, we calculate the communalities and plot them with the variables to see what variables better explain the factors.

```
comm_pcfa <- diag(M_pcfa %*% t(M_pcfa))
comm_pcfa
```

```
##              Pregnancies                  Glucose            BloodPressure
##                0.7045752                0.7496290                0.4295469
##            SkinThickness                  Insulin                      BMI
##                0.7634380                0.7832467                0.8222696
## DiabetesPedigreeFunction                      Age
##                0.2115766                0.7440003
```

```
plot(1:p,sort(comm_pcfa,decreasing=TRUE),pch=20,col=color_1,xlim=c(0,34),xlab="Variables",ylab="Communal
     main="Communalities with PCFA")
text(1:p,sort(comm_pcfa,decreasing=TRUE),labels=names(sort(comm_pcfa,decreasing=TRUE)),pos=4,col=color_
```



The variables that are better explained by the factors are BMI, SkinThickness, Glucose, Insuine, Age and

Pregnancies. These characteristics are the ones that have a bigger weight on the communality part, and not in the uniqueness part. Of course, when we plot the uniqueness and the variables, we get the opposite plot, as follows:

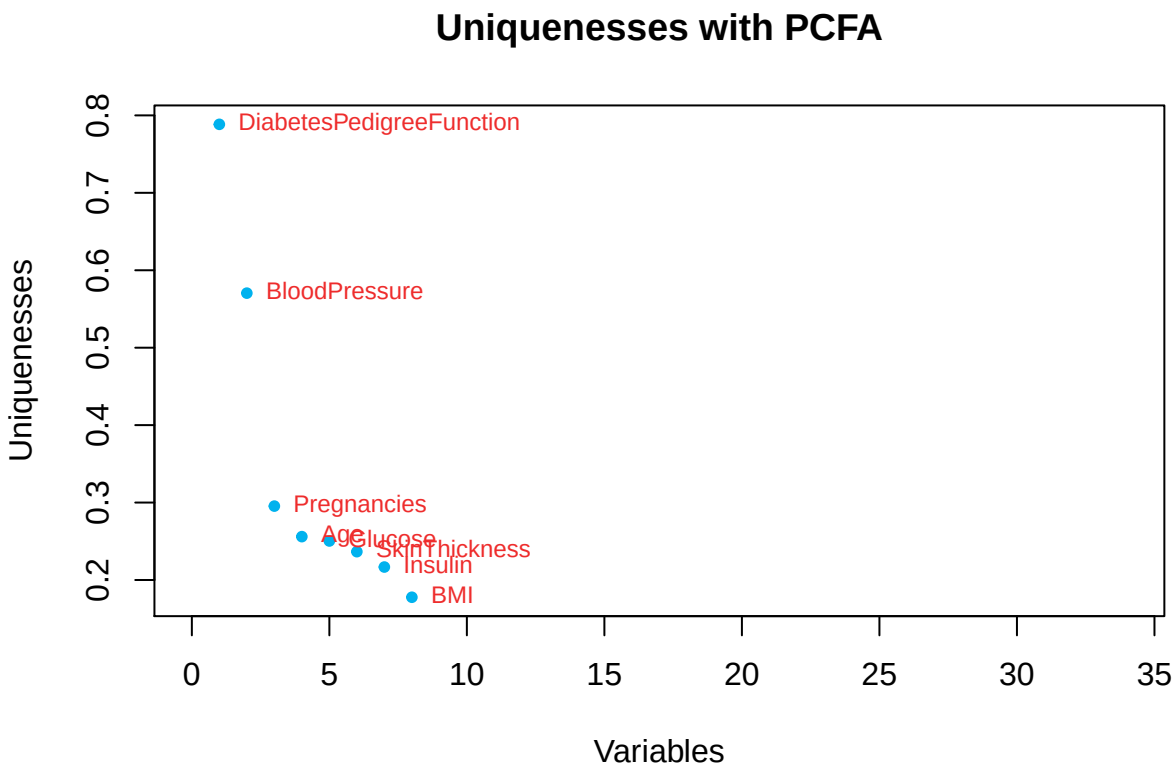
```
uniq_pcfa <- 1 - comm_pcfa
uniq_pcfa
```

```
##              Pregnancies                  Glucose            BloodPressure
##                0.2954248                0.2503710                0.5704531
##            SkinThickness                  Insulin                      BMI
##                0.2365620                0.2167533                0.1777304
## DiabetesPedigreeFunction                      Age
##                0.7884234                0.2559997
```

```
names(uniq_pcfa) <- names(comm_pcfa)
uniq_pcfa
```

```
##              Pregnancies                  Glucose            BloodPressure
##                0.2954248                0.2503710                0.5704531
##            SkinThickness                  Insulin                      BMI
##                0.2365620                0.2167533                0.1777304
## DiabetesPedigreeFunction                      Age
##                0.7884234                0.2559997
```

```
plot(1:p,sort(uniq_pcfa,decreasing=TRUE),pch=20,col=color_1,xlim=c(0,34),xlab="Variables",ylab="Uniquen
     main="Uniquenesses with PCFA")
text(1:p,sort(uniq_pcfa,decreasing=TRUE),labels=names(sort(uniq_pcfa,decreasing=TRUE)),pos=4,col=color_
```



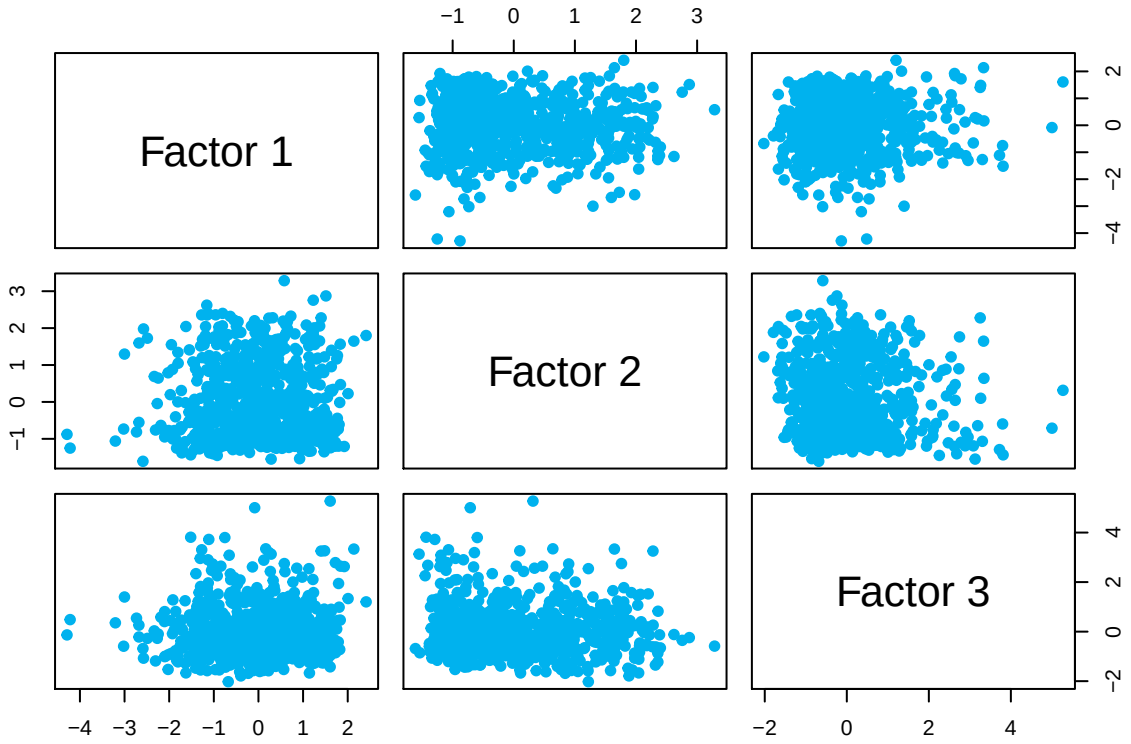
As we can see, the diabetes pedigree function is the variables the less explaines the factors, followed by blood pressure.

Let´s estimate the factors scores now by utilizing the estimation of matrix M, the estimation of the correlation matrix of the errors and the scaled dataset (we have to define it as a matrix).

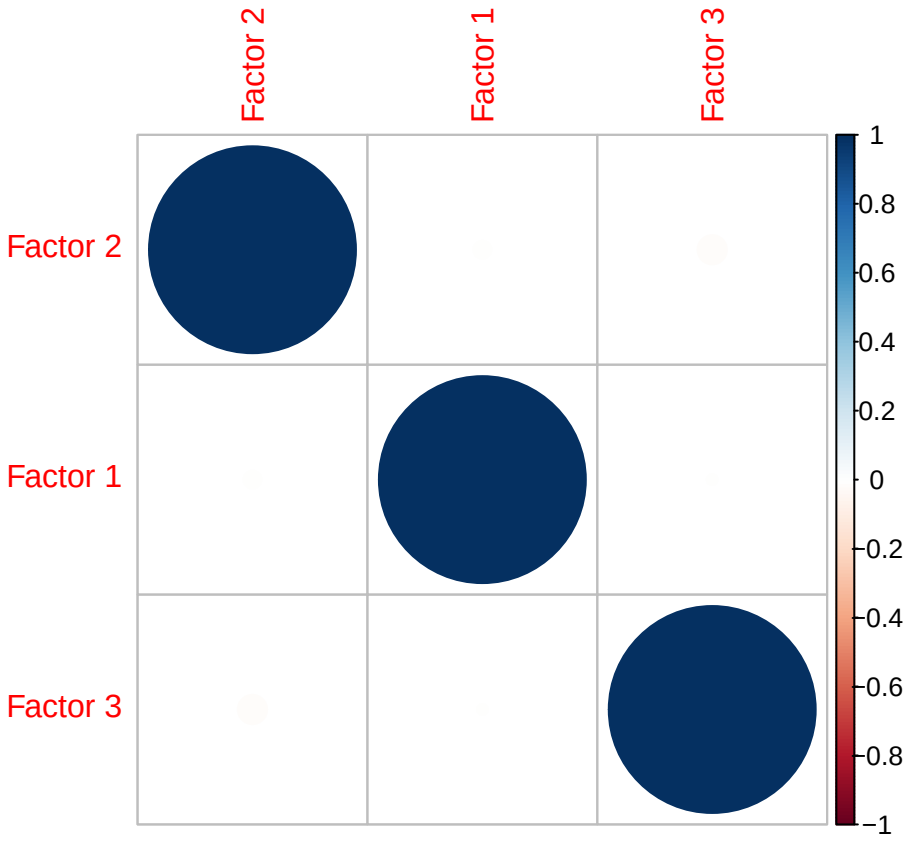
```
Y <- as.matrix(Y)
F_pcfa <- Y %*% solve(Sigma_nu_pcfa) %*% M_pcfa %*% solve(t(M_pcfa) %*% solve(Sigma_nu_pcfa) %*% M_pcfa

colnames(F_pcfa) <- c("Factor 1","Factor 2","Factor 3")
```

Now, we can plot correlation matrix of the factor scores and a simple scatterplot of the 3 factors, to check if there are any relationships between them.

```
pairs(F_pcfa,pch=19,col=color_1)
```

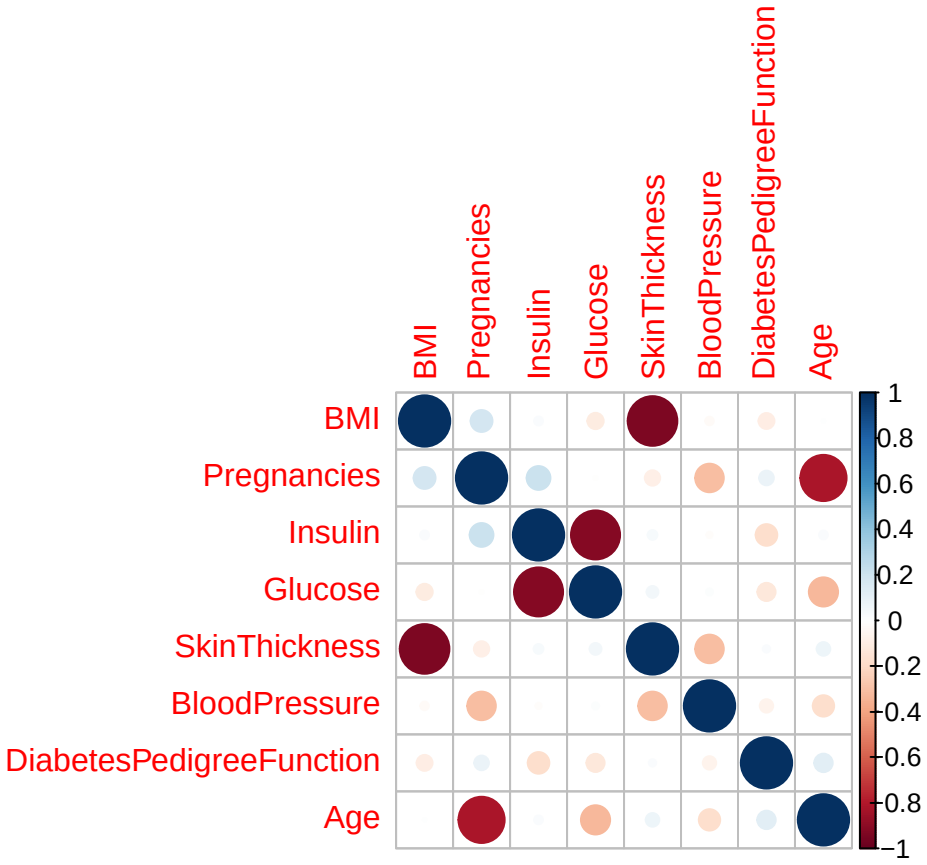



```
corrplot(cor(F_pcfa),order="hclust")
```



As we can see, the factors are for the most part uncorrelated. To check how well the model is working, let´s estimate the residuals and see the correlation matrix of them.

```
Nu_pcfa <- Y- F_pcfa %*% t(M_pcfa)
corrplot(cor(Nu_pcfa),order="hclust")
```



As we can see, there are some correlations between the residuals. This is not a good sign, as it may be telling us that there is a lot in our model that is not being explained.

Let´s study what happens when we do Principal Factor Analysis.

Principal Factor Analysis In order to do this, we will first have to estimate the correlation matrix of X.

```
R_X <- cor(X)
```

After doing this, we have to get sigma, its eigenvalues and eigenvectors, as follows:

```
MM <- R_X - Sigma_nu_pcfa
MM_eig <- eigen(MM)
MM_values <- MM_eig$values
MM_vectors <- MM_eig$vectors
```

We can now estimate the matrix M and use the varimax criterion. Again, we will keep r=3, as this is the result we got from different testing methods in the previous part of the research.

```
M_pfa <- MM_eig$vectors[,1:r] %*% diag(MM_eig$values[1:r])^(1/2)
M_pfa <- varimax(M_pfa)
M_pfa <- loadings(M_pfa)[1:p,1:r]
M_pfa
```

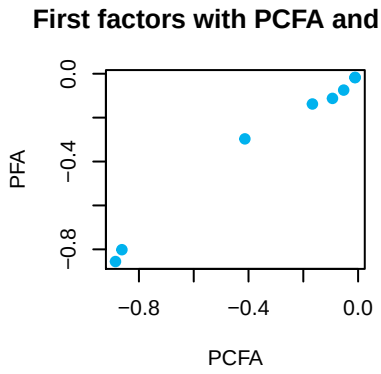
```
##                   [,1]         [,2]         [,3]
```

```
## [1,] -0.01687175  0.77308316  0.07665040
## [2,] -0.11240779  0.23324847 -0.76991693
## [3,] -0.29678878  0.37991598 -0.11887391
## [4,] -0.80158992  0.14001947 -0.09442597
## [5,] -0.07459842  0.02760014 -0.84418062
## [6,] -0.85550145  0.01079962 -0.19788740
## [7,] -0.13790347 -0.03720595 -0.20269795
## [8,] -0.01768420  0.80352352 -0.15295720
```

Now, we can compare the results we get through this method and the one we got we the previous method.

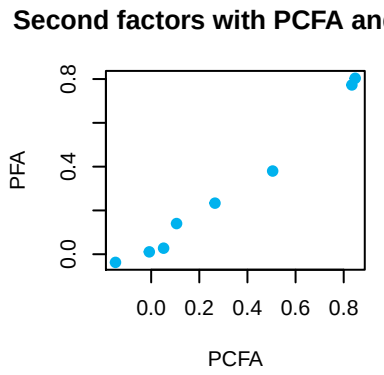
For the first factor, we can see there is not a lot of linearity.

```
par(mfrow=c(2,3))
plot(M_pcfa[,1],M_pfa[,1],pch=19,col=color_1,main="First factors with PCFA and PFA",xlab="PCFA",ylab="PF
```



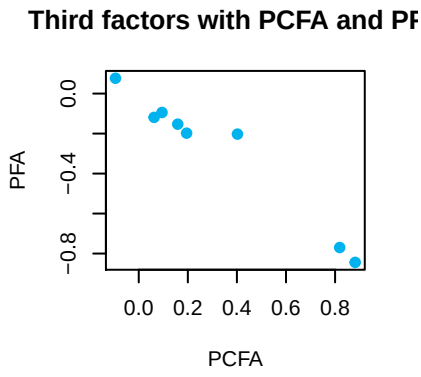
For the second factor, there is also not so much linearity.

```
par(mfrow=c(2,3))
plot(M_pcfa[,2],M_pfa[,2],pch=19,col=color_1,main="Second factors with PCFA and PFA",xlab="PCFA",ylab="P
```



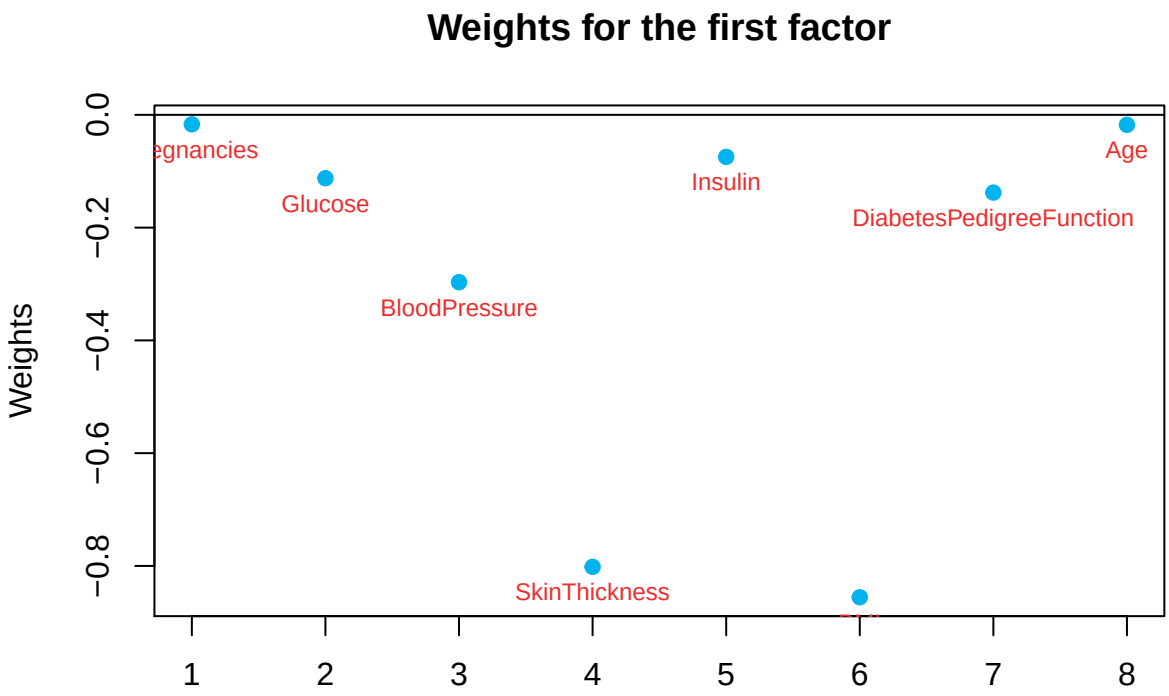
For the third factor, there is also not so much linearity.

```
par(mfrow=c(2,3))
plot(M_pcfa[,3],M_pfa[,3],pch=19,col=color_1,main="Third factors with PCFA and PFA",xlab="PCFA",ylab="PF
```

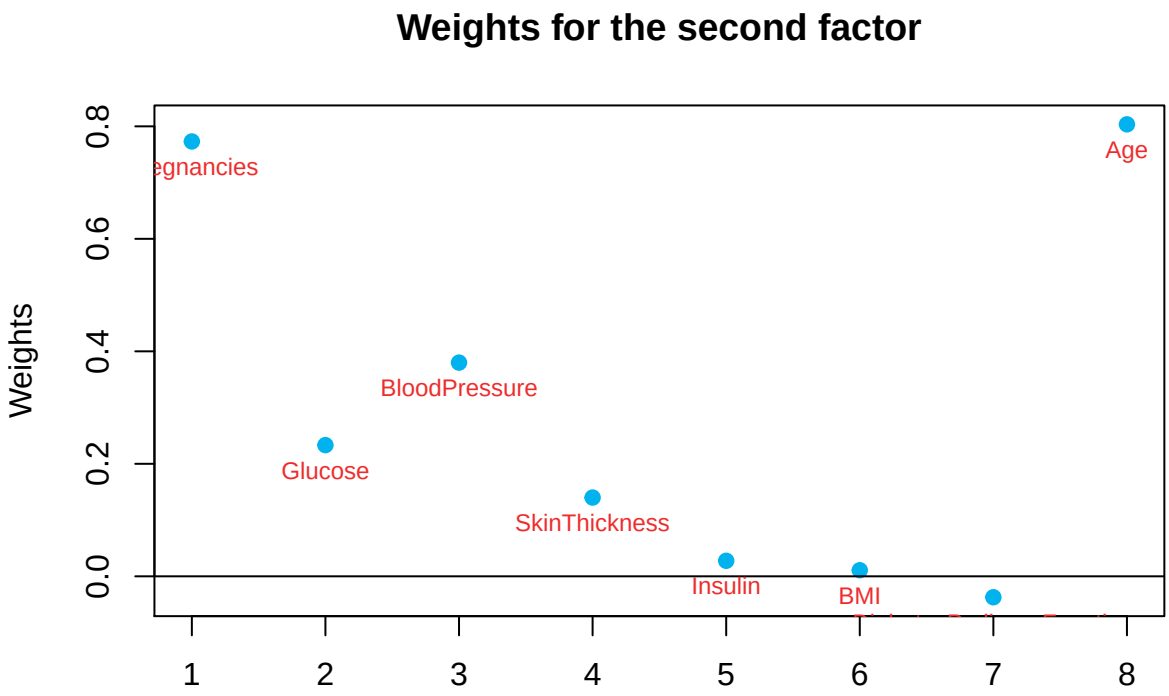


Our three factors don´t seem to be equal through both methods, but look quite alike. Let´s see what happens when we check the weights.

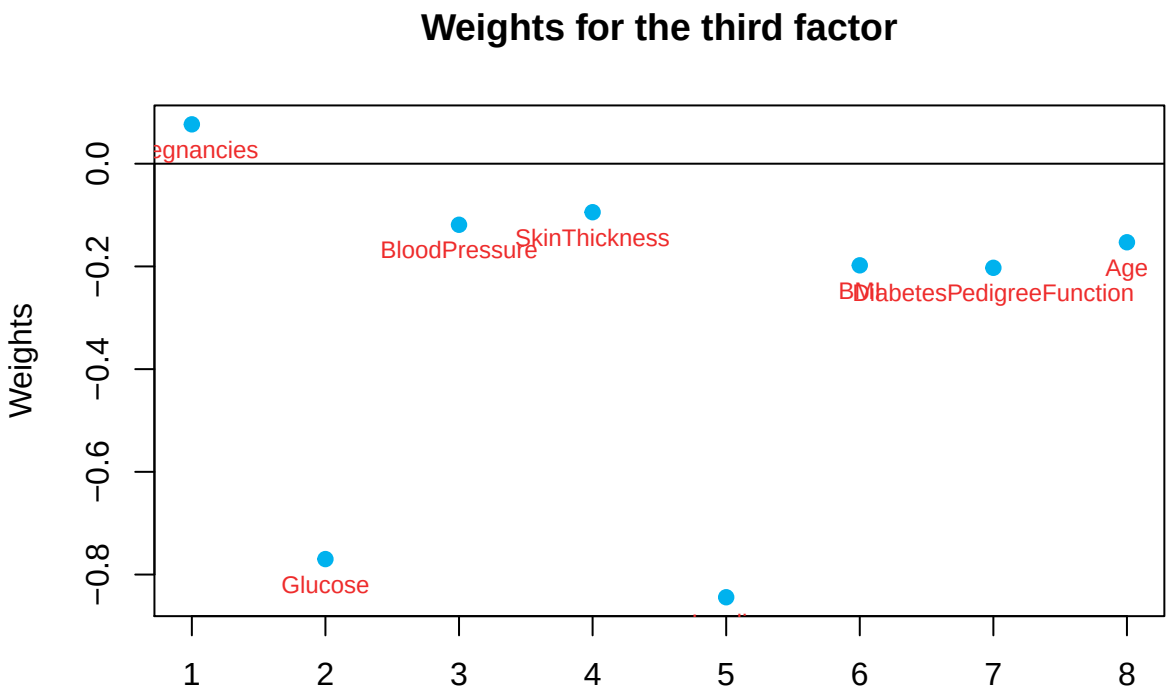
```
plot(1:p,M_pfa[,1],pch=19,col=color_1,xlab="",ylab="Weights",main="Weights for the first factor")
abline(h=0)
text(1:p,M_pfa[,1],labels=colnames(X),pos=1,col=color_5,cex=0.75)
```




```
plot(1:p,M_pfa[,2],pch=19,col=color_1,xlab="",ylab="Weights",main="Weights for the second factor")
abline(h=0)
text(1:p,M_pfa[,2],labels=colnames(X),pos=1,col=color_5,cex=0.75)
```



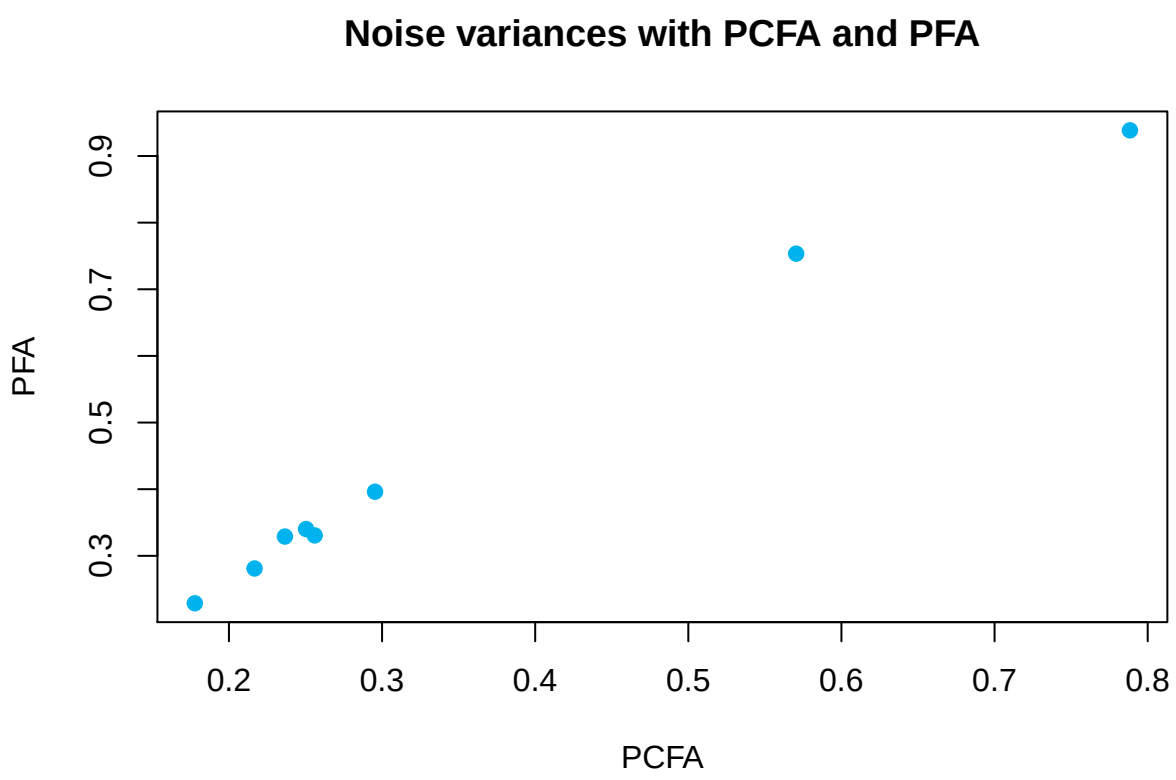

```
plot(1:p,M_pfa[,3],pch=19,col=color_1,xlab="",ylab="Weights",main="Weights for the third factor")
abline(h=0)
text(1:p,M_pfa[,3],labels=colnames(X),pos=1,col=color_5,cex=0.75)
```



The three factors look very similar with one exception, the third one is taking into account the inverse of what the third factor in the other technique was taking into account. So here, the factors is describing the metabolism of the woman but in terms of the inverse.

Let´s estimate the covariance matrix of the errors and comparen them with the one obtained in the previous model.

```
Sigma_nu_pfa <- diag(diag(R_X - M_pfa %*% t(M_pfa)))
par(mfrow=c(1,1))
plot(diag(Sigma_nu_pcfa),diag(Sigma_nu_pfa),pch=19,col=color_1,main="Noise variances with PCFA and PFA"
     xlab="PCFA",ylab="PFA")
```



We can see that there are no big differences between our two techniques of finding the factors underlying our dataset (the main characteristics of the indian women taken into consideration). Let´s now check the communalities and uniqueness and compare them with the ones we got in the previous technique:

```
comm_pfa <- diag(M_pfa %*% t(M_pfa))
names(comm_pfa) <- colnames(Y)
```

First, communalities obtained in the Principal Factor Analysis:

```
sort(comm_pfa,decreasing=TRUE)
```

```
##                   BMI               Insulin         SkinThickness
##           0.77115879            0.71896762            0.67106811
##                   Age               Glucose           Pregnancies
##           0.66935868            0.65981245            0.60381752
##         BloodPressure DiabetesPedigreeFunction
##           0.24655074            0.06148811
```

Now, communalities obtained in the Principal Component Factor Analysis:

```
sort(comm_pcfa,decreasing=TRUE)
```

```
##                   BMI               Insulin         SkinThickness
##             0.8222696             0.7832467             0.7634380
##               Glucose                   Age           Pregnancies
```

```
##               0.7496290               0.7440003               0.7045752
##           BloodPressure DiabetesPedigreeFunction
##               0.4295469               0.2115766
```

As we can see the communalities in the Principal Component Factor Analysis are slightly bigger than in our Principal Factor Analysis. Let´s compare the uniqueness we get from both techniques.

First, the uniquensses in our Principal Factor Analysis:

```
uniq_pfa <- diag(Sigma_nu_pfa)
names(uniq_pfa) <- names(comm_pfa)
sort(uniq_pfa,decreasing=TRUE)
```

```
## DiabetesPedigreeFunction            BloodPressure              Pregnancies
##                0.9385119                0.7534493                0.3961825
##                  Glucose                      Age            SkinThickness
##                0.3401876                0.3306413                0.3289319
##                  Insulin                      BMI
##                0.2810324                0.2288412
```

Noe, the uniquenesses in our Principal Component Factor Analysis:

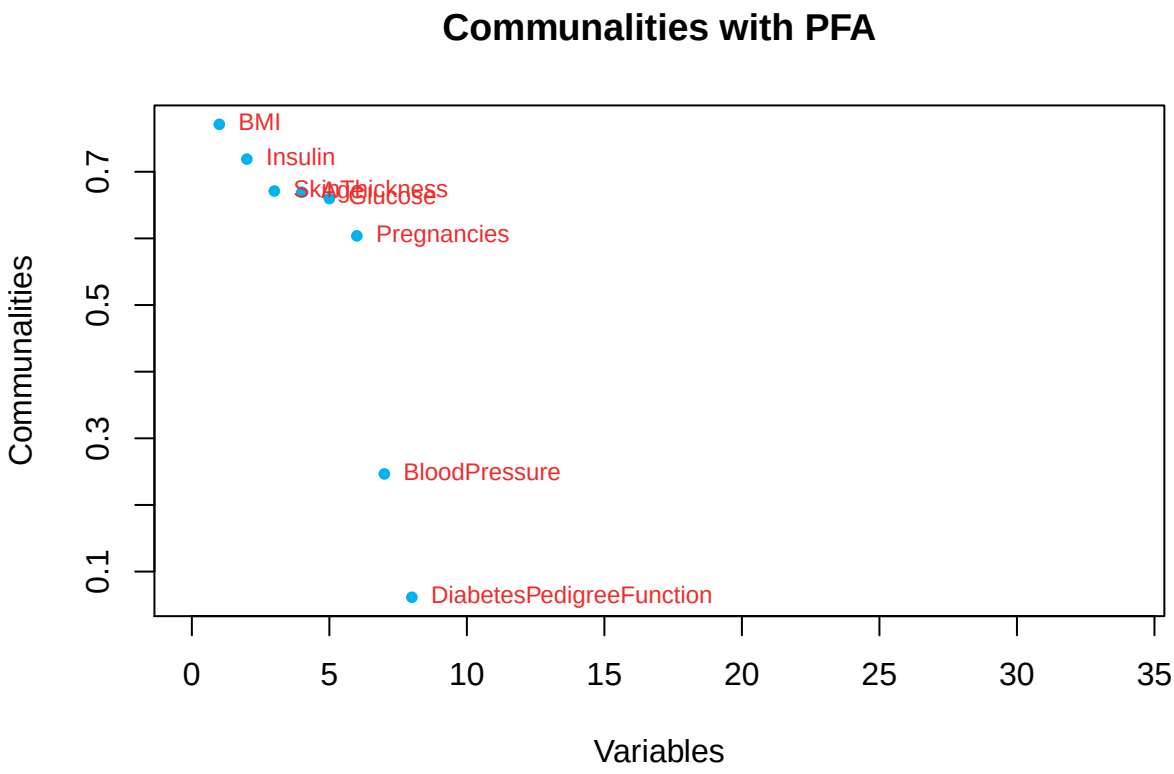
```
sort(uniq_pcfa,decreasing=TRUE)
```

```
## DiabetesPedigreeFunction            BloodPressure              Pregnancies
##                0.7884234                0.5704531                0.2954248
##                      Age                  Glucose            SkinThickness
##                0.2559997                0.2503710                0.2365620
##                  Insulin                      BMI
##                0.2167533                0.1777304
```

As we can see, the uniqueness tends to increase in the Principal Factor Analysis in comparison with the Principal Component Factor Analysis.

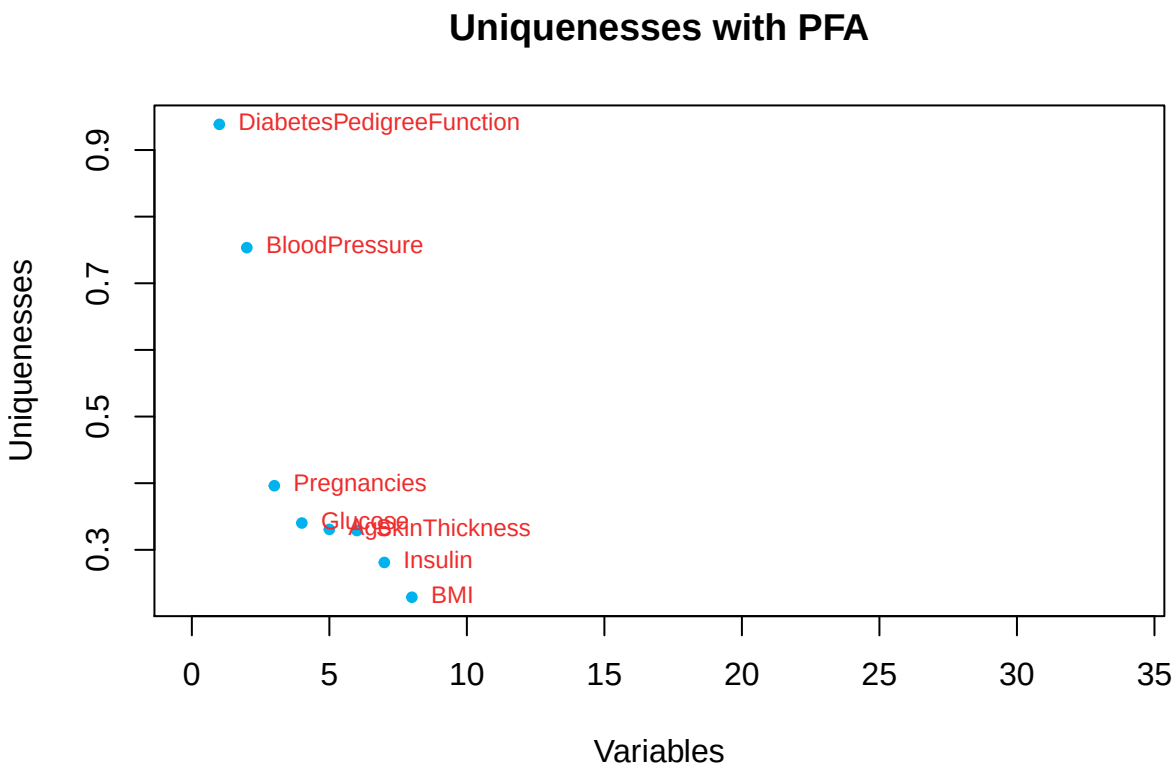
Let´s plot the variables that better explain the factors.

```
plot(1:p,sort(comm_pfa,decreasing=TRUE),pch=20,col=color_1,xlim=c(0,34),xlab="Variables",ylab="Communali
     main="Communalities with PFA")
text(1:p,sort(comm_pfa,decreasing=TRUE),labels=names(sort(comm_pfa,decreasing=TRUE)),pos=4,col=color_5,
```



As we can see, they are: BMI, Insulin, Skin Thickness, Age and Pregnancies. Let´s plot the uniquenesses and get the opposite result:

```
plot(1:p,sort(uniq_pfa,decreasing=TRUE),pch=20,col=color_1,xlim=c(0,34),xlab="Variables",ylab="Uniquenes
     main="Uniquenesses with PFA")
text(1:p,sort(uniq_pfa,decreasing=TRUE),labels=names(sort(uniq_pfa,decreasing=TRUE)),pos=4,col=color_5,
```



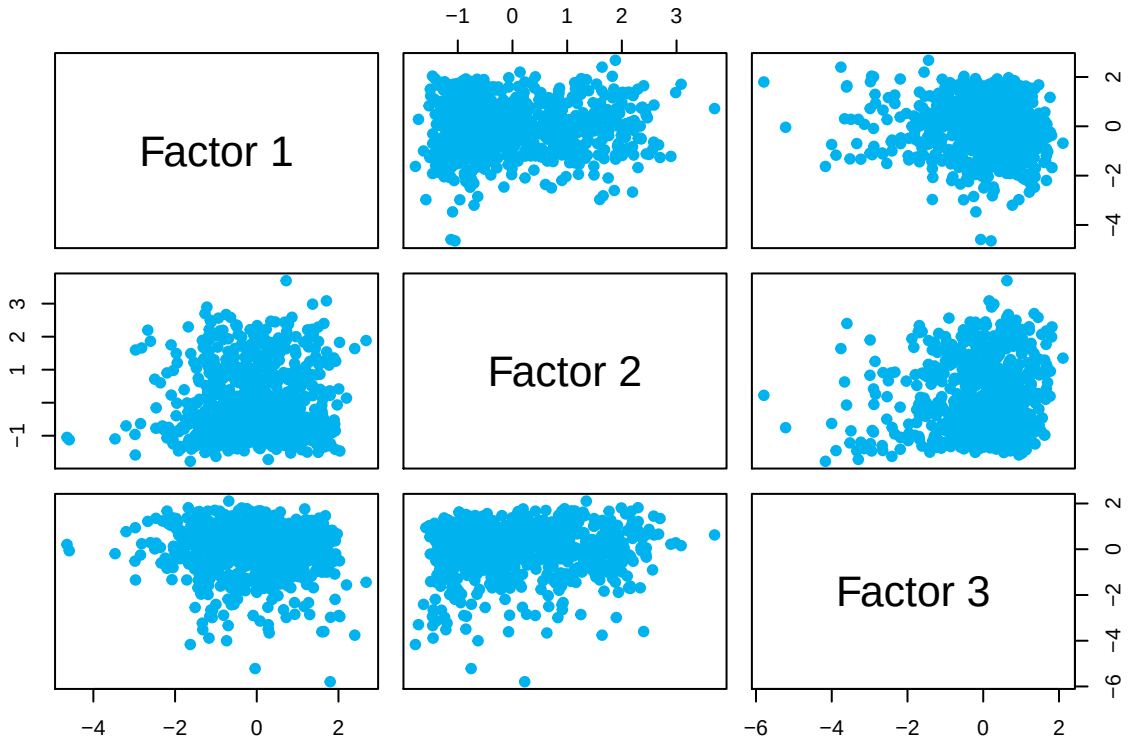
As we can see, the variables that less explain the factors are the Diabetes Pedigree Function and the Blood Preassure.

Now, we can estimate the scores of the factors we got with this method.

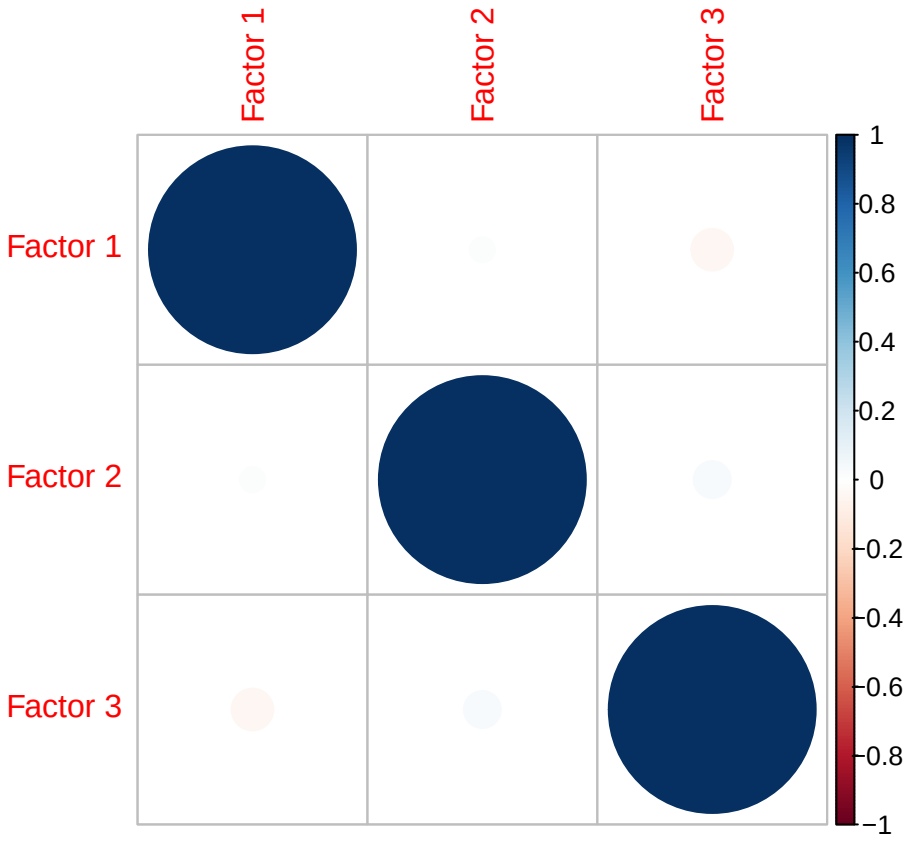
```
F_pfa <- Y %*% solve(Sigma_nu_pfa) %*% M_pfa %*% solve(t(M_pfa) %*% solve(Sigma_nu_pfa) %*% M_pfa)
colnames(F_pfa) <- c("Factor 1","Factor 2","Factor 3")
```

We can plot the scatterplot and a correlation matrix and check if the factors in Principal Factor Analysis are or not correlated.

```
pairs(F_pfa,pch=19,col=color_1)
```




```
corrplot(cor(F_pfa),order="hclust")
```

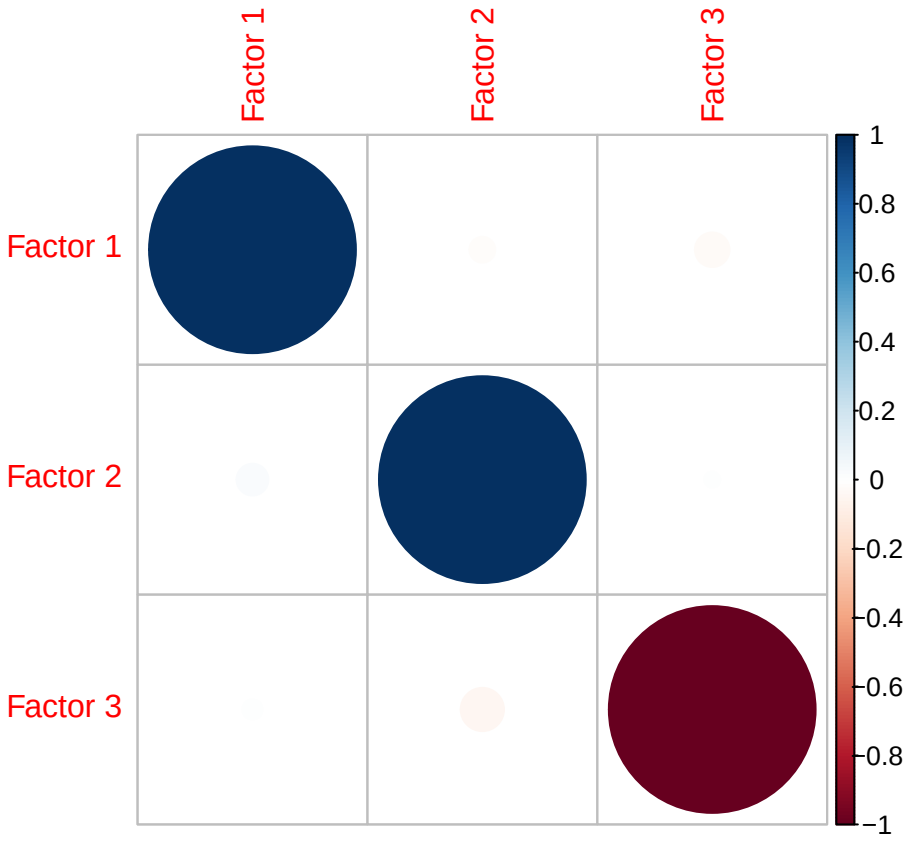


There is no correlation between the factors. Let´s chech now a correlation matrix between the estimates of the Principal Component Factor Analysis and the Principal Factor Analysis:

```
cor(F_pcfa,F_pfa)
```

```
##              Factor 1    Factor 2     Factor 3
## Factor 1 0.998961948 -0.01541042 -0.027832355
## Factor 2 0.023517512  0.99847591  0.005774429
## Factor 3 0.009075637 -0.04366243 -0.998317705
```

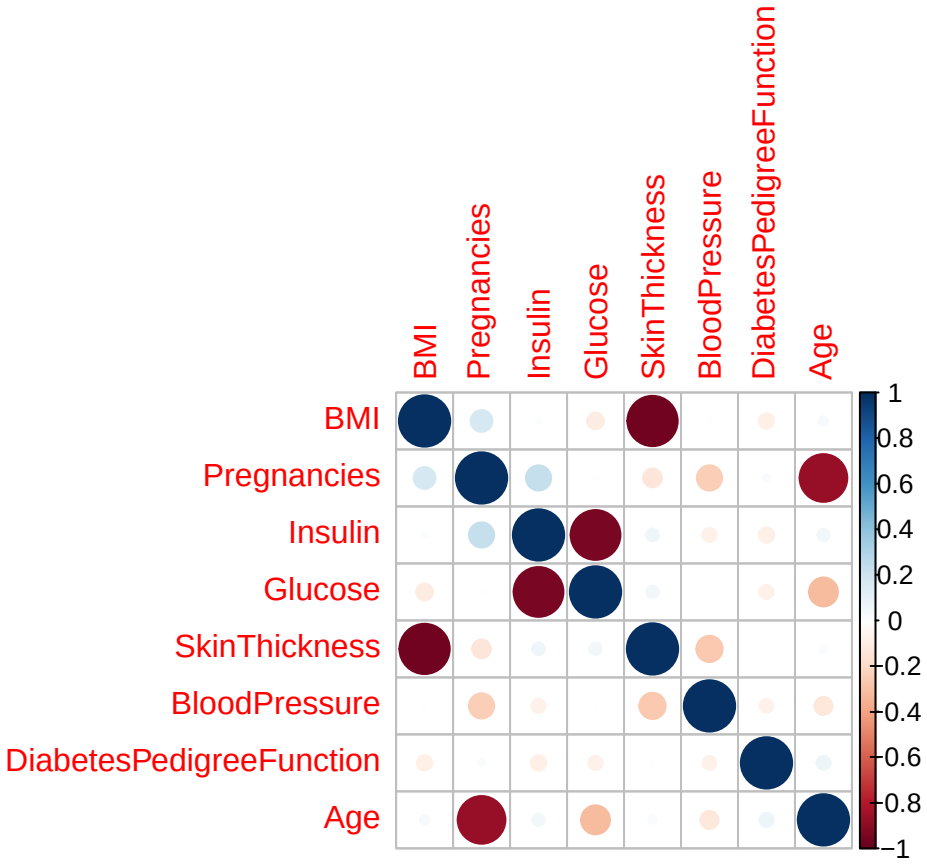
```
corrplot(cor(F_pcfa,F_pfa))
```



Both ways of estimating the factors seem very similar. As we have seen, the factors are perfectly correlated, and the third one has a perfect inverse correlation, as we expressed before when we commented that it was expressing the opposite index in comparison with the technique used before.

Finally, let´s plot the correlation matrix between the residuals, and check how much variability we can not explain with our model.

```
Nu_pfa <- Y - F_pfa %*% t(M_pfa)
corrplot(cor(Nu_pfa),order="hclust")
```



As we can see, there still serious correlations between the residuals, showing us that there is still some vairability that we can not explain with the model.

Maximum Likelihood Estimation As now we are assuming our data comes from a gaussian distribution (and indeed, it had been transformed into normality in the first part of the project) we can perform likelihood ratio test to say how many factors are we going to use. By doing this test, we will be collecting evidence to reject (or not) our null hypotesis of a reduced model, against an alternative hypotesis of a more complete model. For example, first we will try if 1 factor is enough (H0) ir not (H1), and so on.

```
Y_mle_1 <- factanal(Y,factors=1,rotation="varimax",scores="Bartlett")
Y_mle_1$STATISTIC
```

```
## objective
##  751.4978
```

```
Y_mle_1$PVAL
```

```
##     objective
## 2.748741e-146
```

We reject the null hypotesis, as our p-value is very small. Let´s go to the next one, with two factors:

```
Y_mle_2 <- factanal(Y,factors=2,rotation="varimax",scores="Bartlett")
Y_mle_2$STATISTIC
```

```
## objective
##   335.843
```

```
Y_mle_2$PVAL
```

```
##    objective
## 7.342829e-64
```

Our p-value is very small, so we reject again the null hypotesis.

```
Y_mle_3 <- factanal(Y,factors=3,rotation="varimax",scores="Bartlett")
Y_mle_3$STATISTIC
```

```
## objective
##   18.6258
```

```
Y_mle_3$PVAL
```

```
##   objective
## 0.009443999
```

Now, we can reject the null hypotesis saying that our model with three factors is not enough (as p-value is larger than 0.05). This way, we can say that maybe three factors are not the right number, confirming that the way in which we were working maybe was not alright.

```
Y_mle_4 <- factanal(Y,factors=4,rotation="varimax",scores="Bartlett")
Y_mle_4$STATISTIC
```

```
## objective
##  4.410521
```

```
Y_mle_4$PVAL
```

```
## objective
## 0.1102218
```

As we see, we keep on rejecting the null hypotesis. This may be due to the fact that our data does not really behave as propper gaussian distribution. We will still work with three factors, as we only have 8 variables, 3 seems like a reasonable number.

Let´s check the loading matrix (the one that multiplies the factor matrix in our model, and is a matrix of constants).

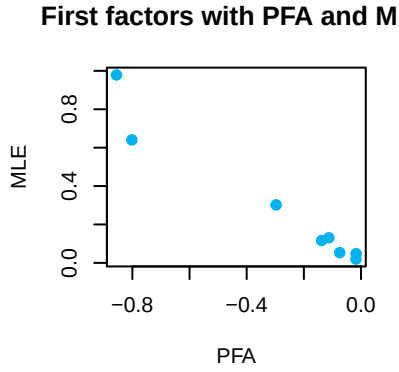
```
M_mle <- loadings(Y_mle_3)[1:p,1:r]
M_mle
```

```
##                              Factor1     Factor2      Factor3
## Pregnancies              0.04781513 -0.02652968  0.634376601
## Glucose                  0.13055873  0.59254545  0.209337224
## BloodPressure            0.30180151  0.09602071  0.346100621
```

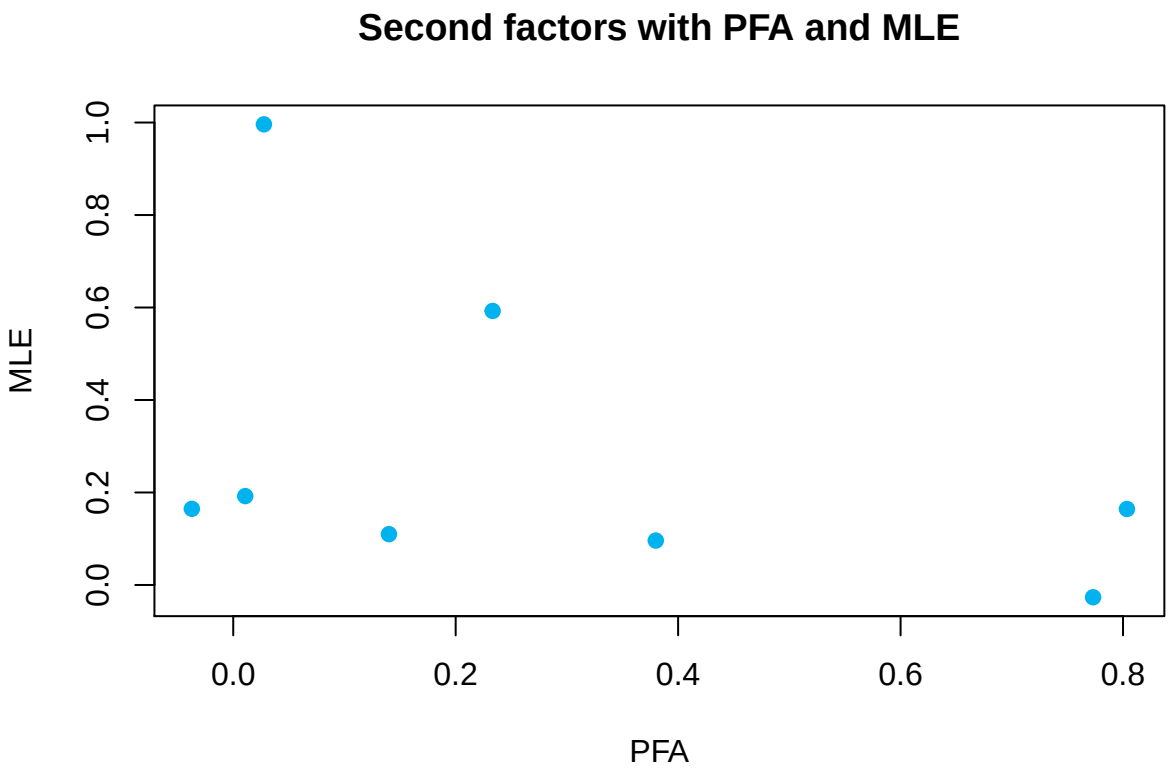
```
## SkinThickness            0.64026738  0.10986973  0.149485303
## Insulin                  0.05316940  0.99607763 -0.001559536
## BMI                      0.97860670  0.19220609 -0.019646928
## DiabetesPedigreeFunction 0.11633335  0.16468833 -0.005762662
## Age                      0.01981743  0.16428788  0.860859605
```

We can compare with the estimations we got from the PFA.

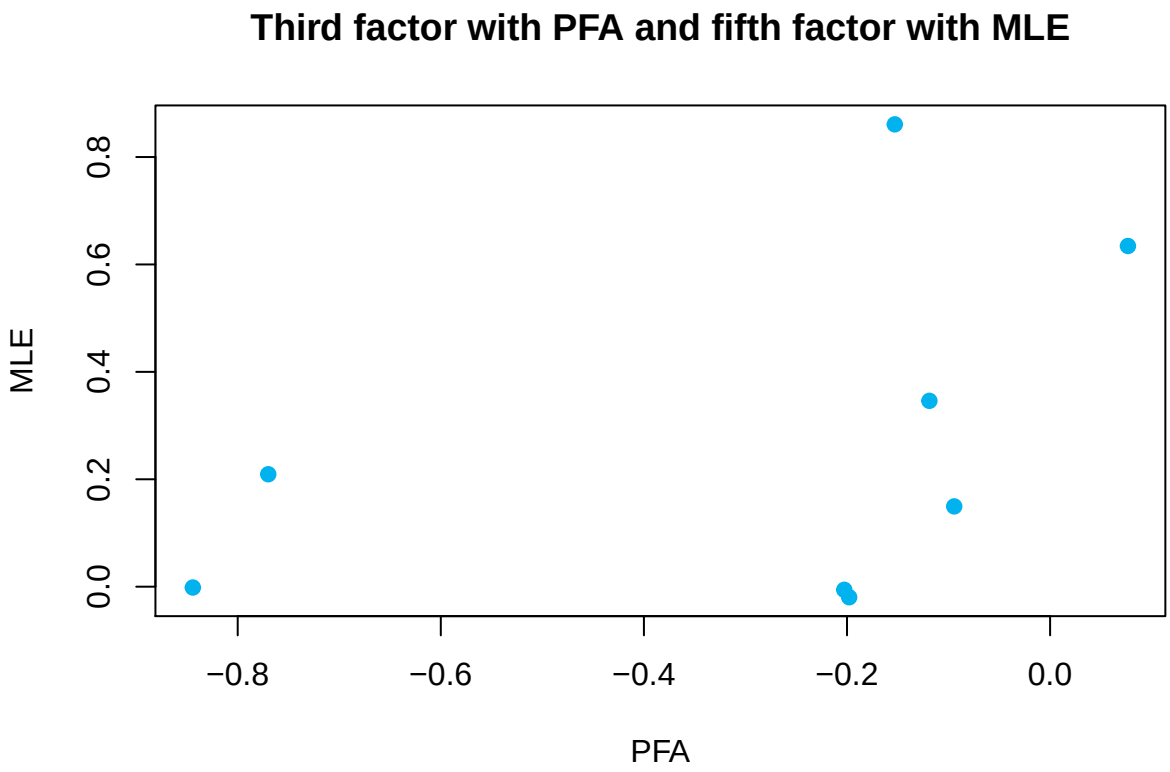
```
par(mfrow=c(2,3))
plot(M_pfa[,1],M_mle[,1],pch=19,col="deepskyblue2",main="First factors with PFA and MLE",xlab="PFA",ylab
```




```
plot(M_pfa[,2],M_mle[,2],pch=19,col="deepskyblue2",main="Second factors with PFA and MLE",xlab="PFA",yla
```

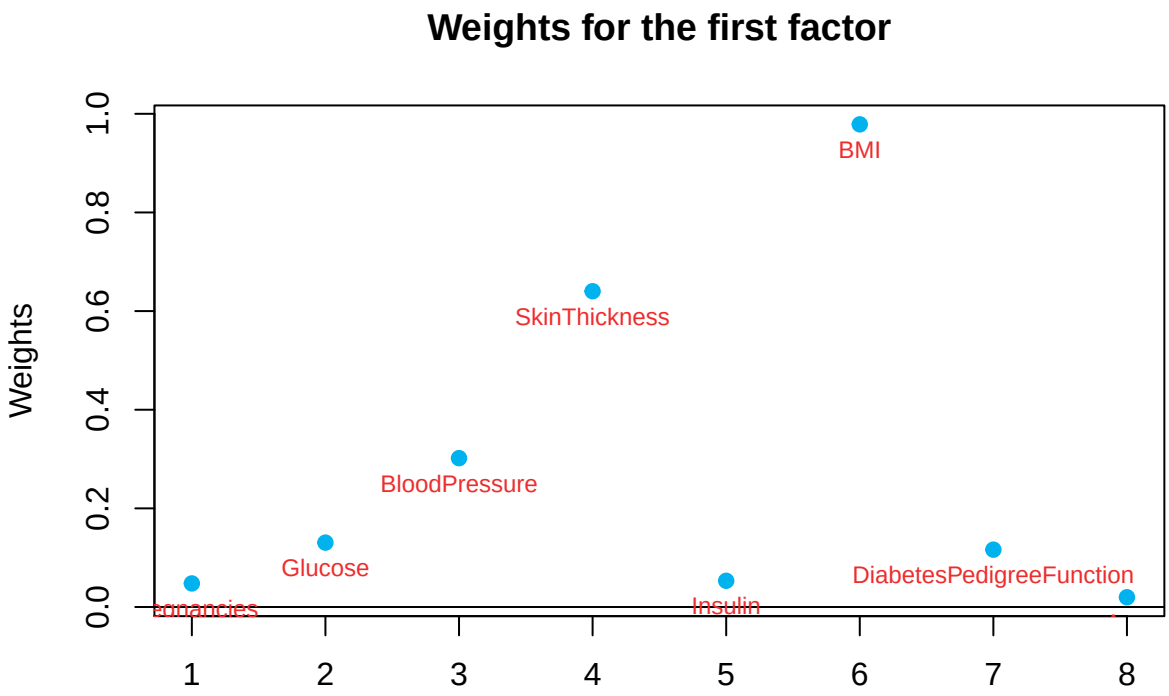



```
plot(M_pfa[,3],M_mle[,3],pch=19,col="deepskyblue2",main="Third factor with PFA and fifth factor with MLE
```

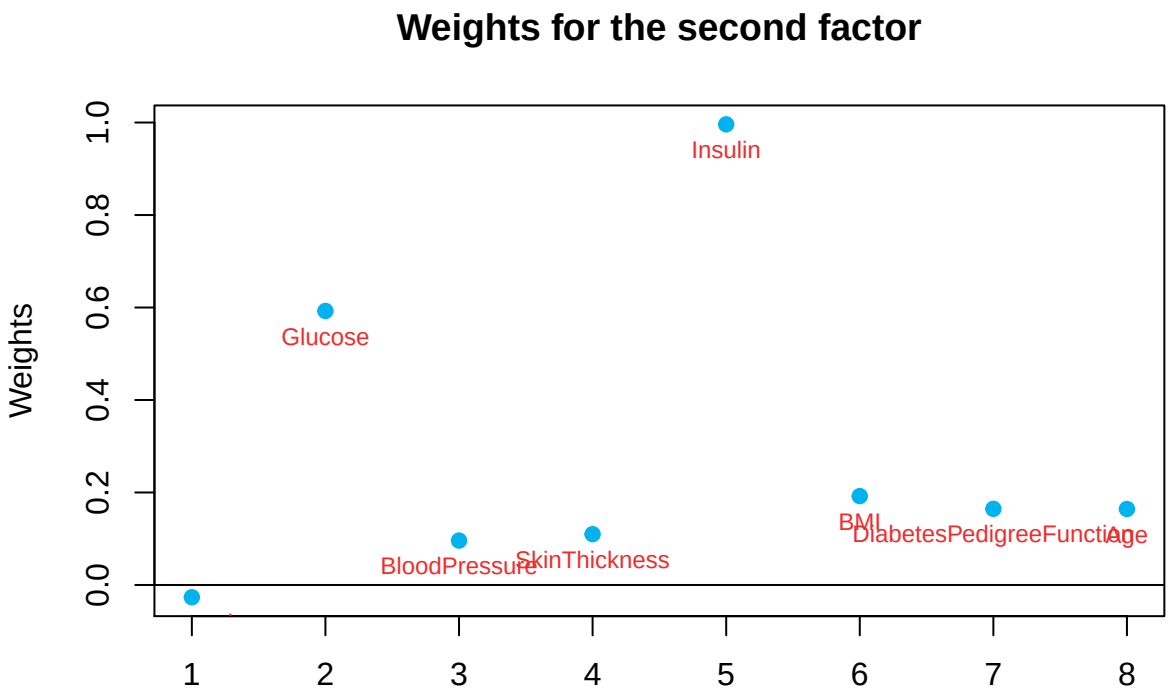


The estimates of the loading matrix don´t seem to be very similar, at least not at first view. So we will plot the weights to see what variables are explaining each factor.

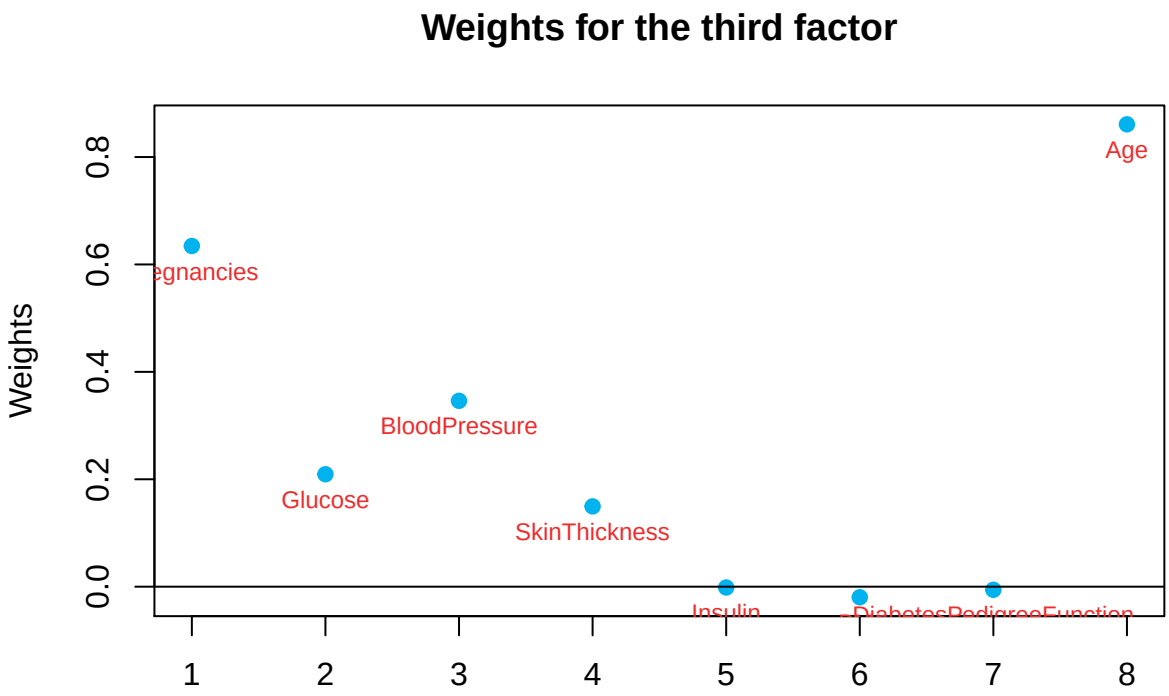
```
plot(1:p,M_mle[,1],pch=19,col=color_1,xlab="",ylab="Weights",main="Weights for the first factor")
abline(h=0)
text(1:p,M_mle[,1],labels=colnames(X),pos=1,col=color_5,cex=0.75)
```




```
plot(1:p,M_mle[,2],pch=19,col=color_1,xlab="",ylab="Weights",main="Weights for the second factor")
abline(h=0)
text(1:p,M_mle[,2],labels=colnames(X),pos=1,col=color_5,cex=0.75)
```



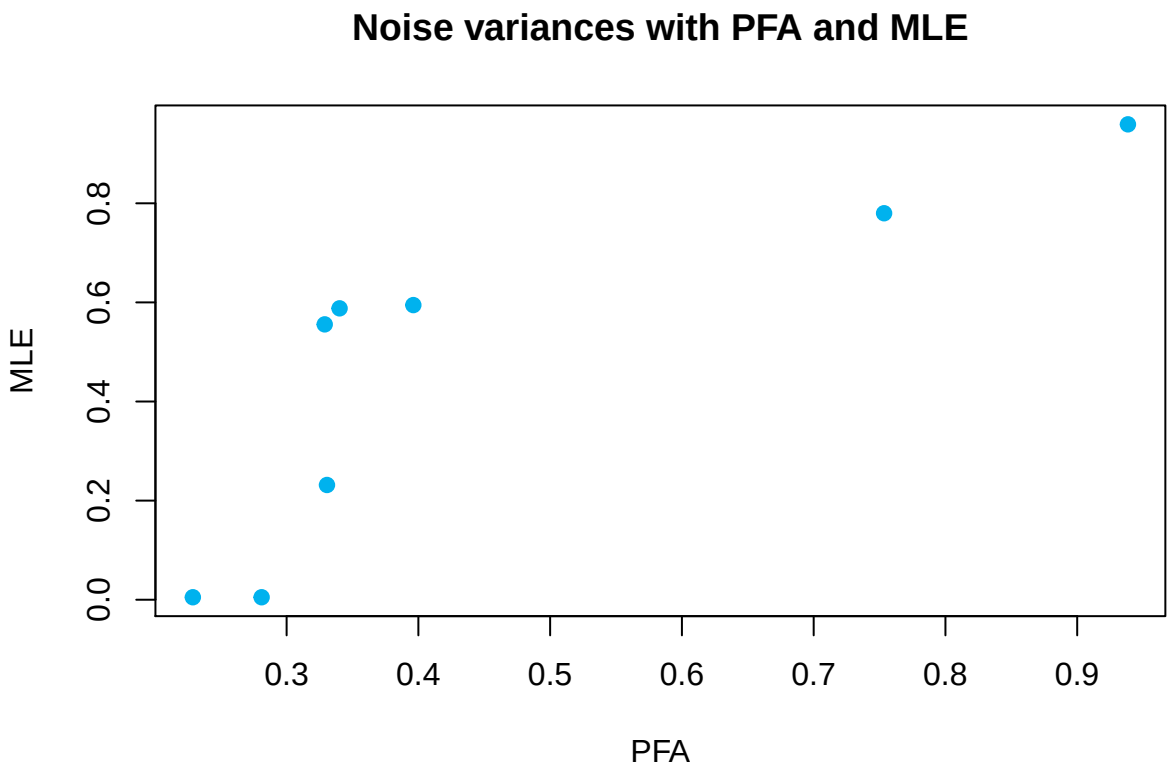

```
plot(1:p,M_mle[,3],pch=19,col=color_1,xlab="",ylab="Weights",main="Weights for the third factor")
abline(h=0)
text(1:p,M_mle[,3],labels=colnames(X),pos=1,col=color_5,cex=0.75)
```



Finally, we see that the weights assure us that the variables that explain each factor are the same in this case as in the other two before (PFA and PCFA). Only in the case of the first and the third factors, they are inverted.

Let´s check the covariance matrix of the errors now and compare them with the ones that we got in our Principal Factor Analysis:

```
Sigma_nu_mle <- diag(diag(cov(Y) - M_mle %*% t(M_mle)))
par(mfrow=c(1,1))
plot(diag(Sigma_nu_pfa),diag(Sigma_nu_mle),pch=19,col=color_1,main="Noise variances with PFA and MLE",
     xlab="PFA",ylab="MLE")
```



As we can see, there are some differences between the noise in both models. Let´s check on the comunnalities and uniquenesses.

First, let´s compare the communalities between our Maximum Likelihood estimation and the one obtained by our Principal Factor Analysis:

```
comm_mle <- diag(M_mle %*% t(M_mle))
names(comm_mle) <- colnames(Y)
sort(comm_mle,decreasing=TRUE)
```

```
##                   BMI               Insulin                   Age
##             0.9950003             0.9950001             0.7684625
##         SkinThickness               Glucose           Pregnancies
##             0.4443595             0.4119778             0.4054238
##         BloodPressure DiabetesPedigreeFunction
##             0.2200898             0.0406889
```

```
sort(comm_pfa,decreasing=TRUE)
```

```
##                   BMI               Insulin         SkinThickness
##            0.77115879            0.71896762            0.67106811
##                   Age               Glucose           Pregnancies
##            0.66935868            0.65981245            0.60381752
##         BloodPressure DiabetesPedigreeFunction
##            0.24655074            0.06148811
```

As we can see, the communalities in the MLE are highers than the one obtained with PFA in many cases: BMI, Age, Glucose while not in others such as: pregnancies, insulin, blooad preassure, diabetes pedigree function and skinthickness.

Let´s check what happens with the uniquensses, although we should expect to be the opposite as what we just saw, as communalities + uniquensses sum 1.

Uniquensses in the MLE:

```
uniq_mle <- diag(Sigma_nu_mle)
names(uniq_mle) <- names(comm_mle)
sort(uniq_mle,decreasing=TRUE)
```

```
## DiabetesPedigreeFunction          BloodPressure            Pregnancies
##              0.959311098            0.779910233            0.594576218
##                  Glucose          SkinThickness                    Age
##              0.588022231            0.555640470            0.231537502
##                  Insulin                    BMI
##              0.004999942            0.004999739
```

Uniquenesses in the PFA:

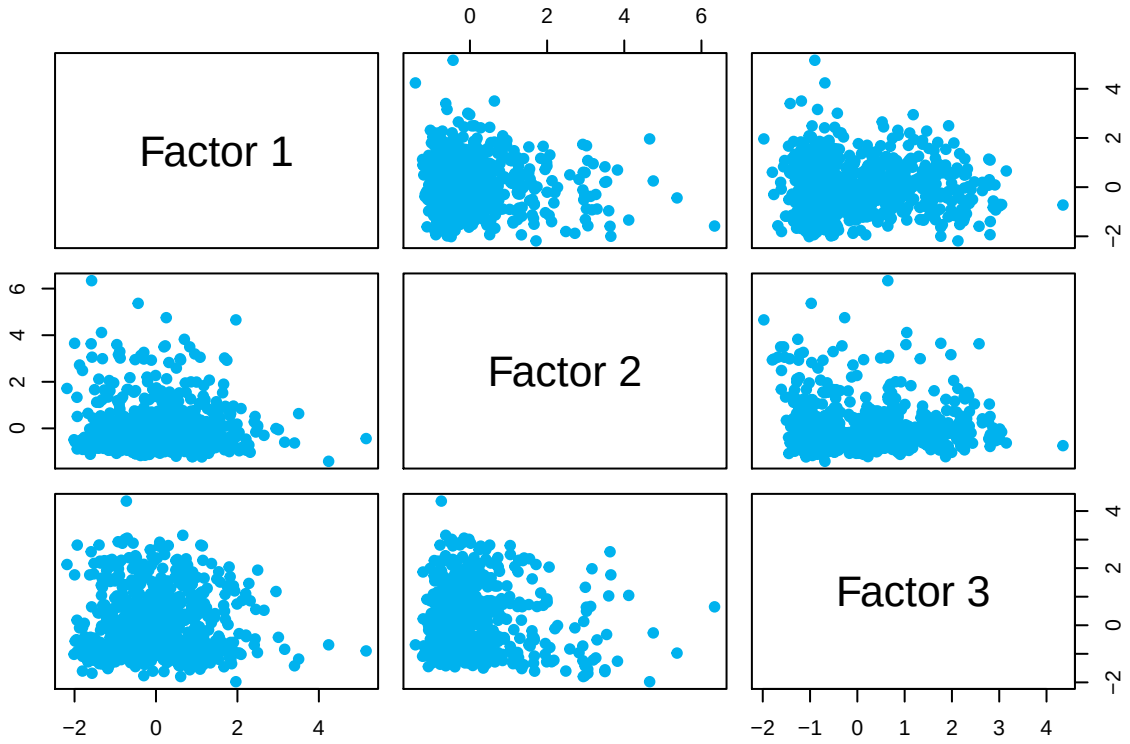
```
sort(uniq_pfa,decreasing=TRUE)
```

```
## DiabetesPedigreeFunction          BloodPressure            Pregnancies
##                0.9385119              0.7534493              0.3961825
##                  Glucose                    Age          SkinThickness
##                0.3401876              0.3306413              0.3289319
##                  Insulin                    BMI
##                0.2810324              0.2288412
```

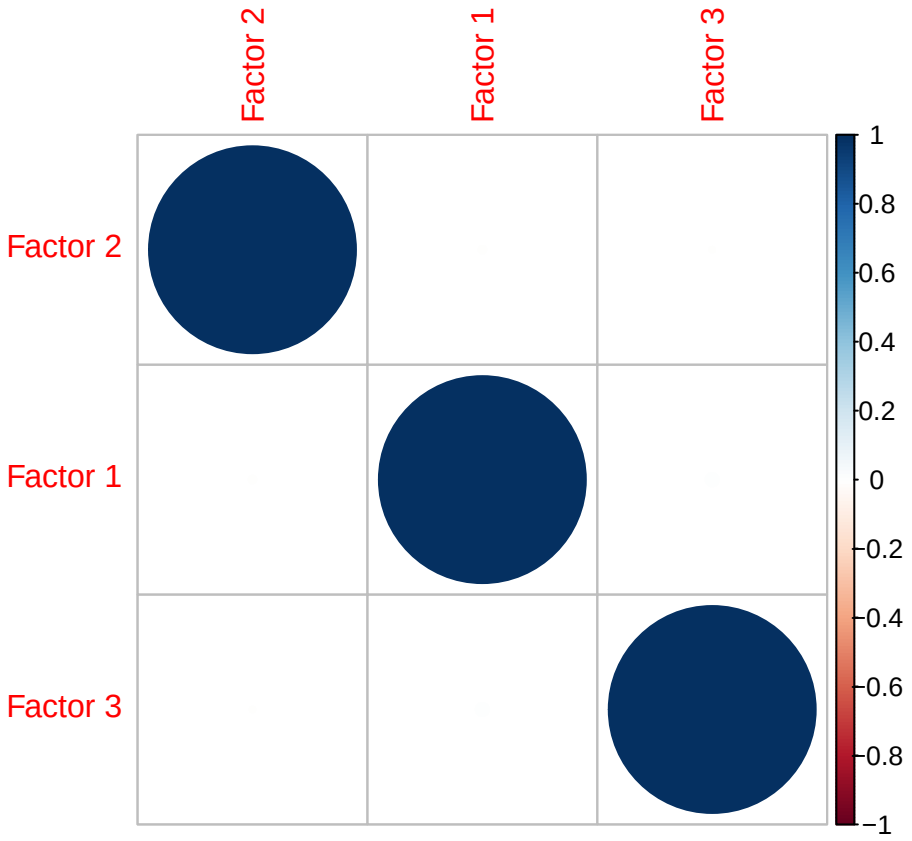
The variables that are best explained by the factors in both cases are the BMI, Age, Glucose, insulin. While the worst explained by the factors are: diabetespedigreefunction, blood preassure.

Let´s estimate the factors scores and check the correlation:

```
F_mle <- Y %*% solve(Sigma_nu_mle) %*% M_mle %*% solve(t(M_mle) %*% solve(Sigma_nu_mle) %*% M_mle)
colnames(F_mle) <- c("Factor 1","Factor 2","Factor 3")
pairs(F_mle,pch=19,col="deepskyblue2")
```




```
corrplot(cor(F_mle),order="hclust")
```

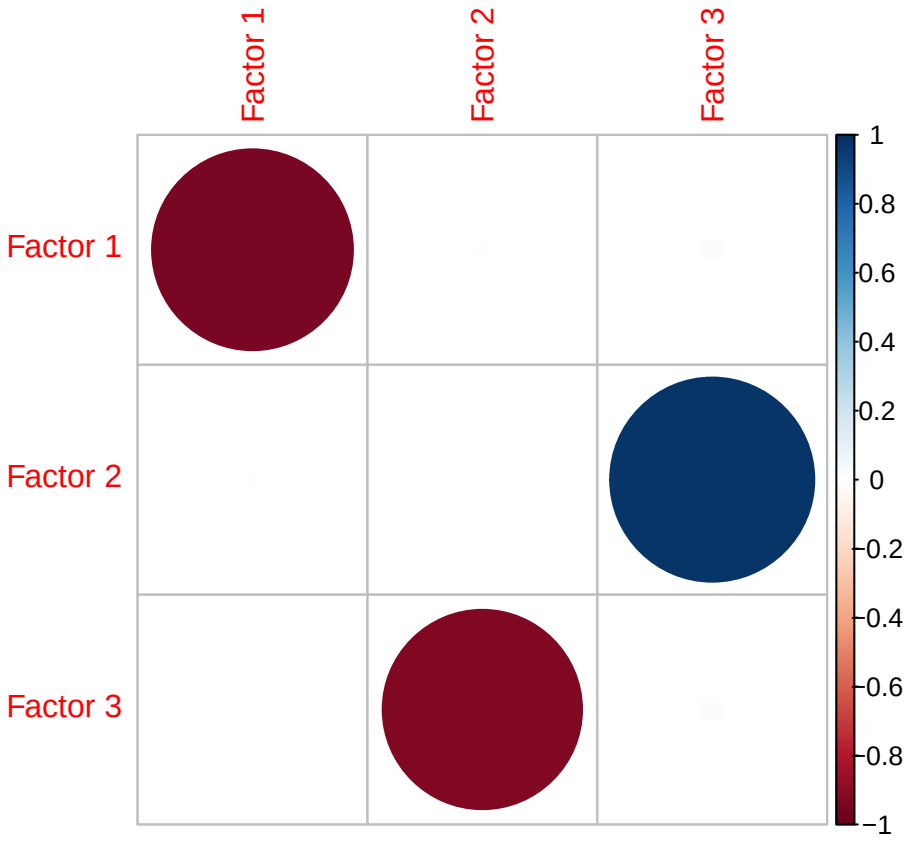


There are some small correlations between some factors, but very small. Let´s now check the correlation matrix between the PFA and MLE estimations.

```
cor(F_pfa,F_mle)
```

```
##                Factor 1     Factor 2   Factor 3
## Factor 1 -0.9421884441  0.006465188 0.01062394
## Factor 2  0.0053954773  0.001607796 0.97326049
## Factor 3 -0.0003655289 -0.927982922 0.01261019
```

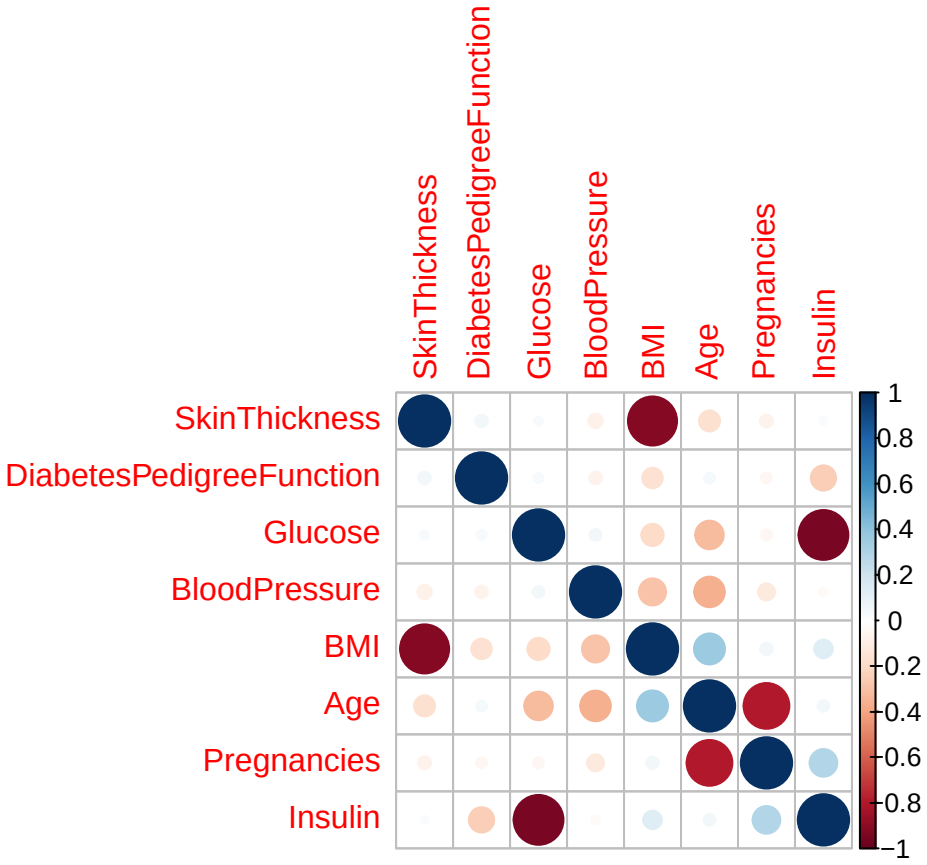
```
corrplot(cor(F_pfa,F_mle))
```



Again, we see some very small (almost null) correlation between different factors.

Let´s estimate the residuals and check the correlation matrix:

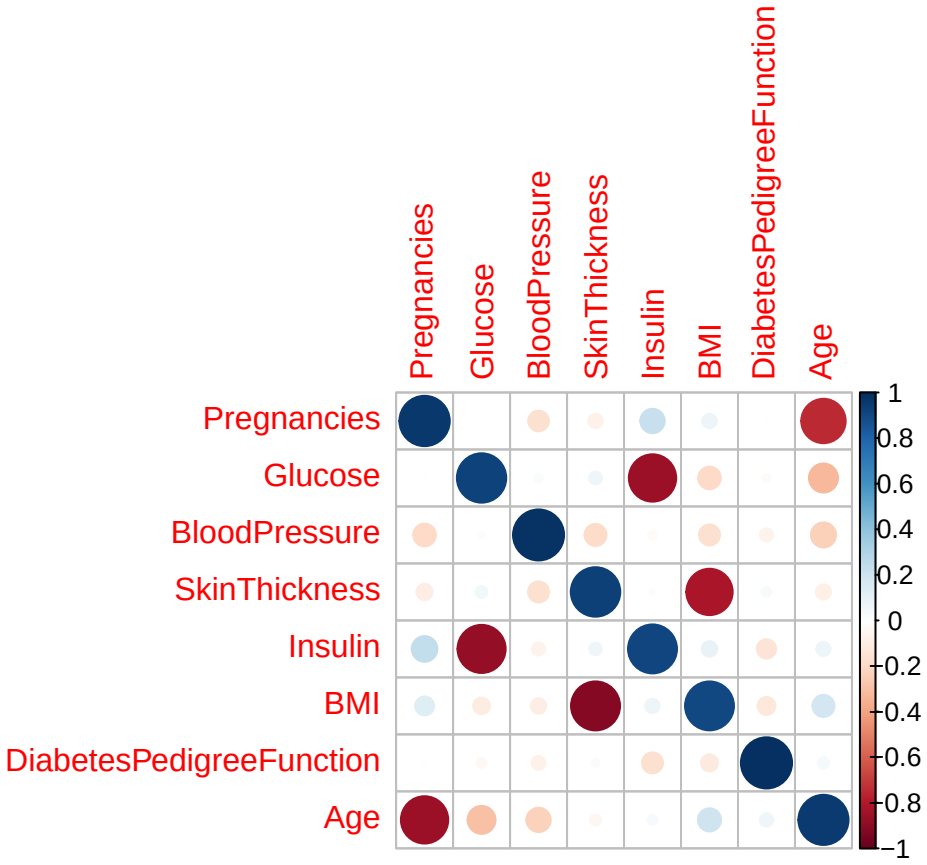
```
Nu_mle <- Y - F_mle %*% t(M_mle)
corrplot(cor(Nu_mle),order="hclust")
```



Again, we see that there is some correlation between the residuals of some variables. Variability that our model can not account for.

Let´s the the correlation matrix between this residuals and the ones obtained in Princial Factor Analysis.

```
corrplot(cor(Nu_pfa,Nu_mle))
```



As always before, there is correlation between the residuals of some variables that our model can not explain.

Multidimensional Scaling

First, I will create a matrix of the similarities between the different political parties in Uruguay. Let us first introduce what each party name refers to:

- PN (“Partido Nacional): one of the two historical most classical parties of Uruguay (existing since independance). It is characterized as conservador and neoliberal, specially related to church. -PC (”Partido Colorado"): the other historical most classical party of Uruguay (also existing since independance). These two parties were the only ones existing in Uruguay until the seventies. This is also conservator party.
- FA (“Frente Amplio”): this is a left-wing party, including many small left parties (from communist, socialist and parties related to worker unions).
- PERI (“Partido Ecologista Radical Instransigente”): this is a new party, that is related a bit to the left wing but is specially worried about environmental resources and the use of agrochemicals (being Uruguay an economy based in agriculture).
- PVA (“Partido Verde Animalista”): this is a party related to veganism and is very opposed to all other parties, believing in big conspirataional theories and very against projects related to selling resources to other countries.
- PDLG (“Partido de la Gente”): this is very small partythat follows an entrepreneur as leader who had a lot of success, but mainly believes in the hard work and decreasing taxes.
- CA (“Cabildo Abierto”): is a party of ex military people, spetially worried about security and law enforcement.

Based on my own opinion, here is a simmilarity matrix between the different parties.

```
pol_part <- c(1,0.7,0.1,0.2,0.05,0.6,0.85,0.7,1,0.2,0.7,0.05,0.7,0.8,0.1,0.2,1,0.7,0.5,0.1,0.05,0.2,0.2
pol_part <- matrix(pol_part,nrow=7,dimnames = list(c("PN","PC","FA","PERI","PVA","PDLG","CA"), c("PN","
pol_part
```

```
##        PN   PC   FA PERI  PVA PDLG   CA
## PN   1.00 0.70 0.10 0.20 0.05  0.6 0.85
## PC   0.70 1.00 0.20 0.20 0.05  0.7 0.80
## FA   0.10 0.20 1.00 0.70 0.50  0.1 0.05
## PERI 0.20 0.70 0.70 1.00 0.80  0.1 0.05
## PVA  0.05 0.05 0.50 0.80 1.00  0.2 0.10
## PDLG 0.60 0.70 0.10 0.10 0.20  1.0 0.70
## CA   0.85 0.80 0.05 0.05 0.10  0.7 1.00
```

Now, I will transform this matrix into a dissimilarity one.

```
library(smacof)
```

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

## Loading required package: plotrix

## Warning: package 'plotrix' was built under R version 4.0.3

##
## Attaching package: 'plotrix'

## The following object is masked from 'package:psych':
##
##     rescale

## Loading required package: colorspace

## Loading required package: e1071

##
## Attaching package: 'e1071'

## The following objects are masked from 'package:moments':
##
##     kurtosis, moment, skewness

##
## Attaching package: 'smacof'

## The following object is masked from 'package:psych':
##
##     Procrustes

## The following object is masked from 'package:base':
##
##     transform
```

```
diss_pol_part <- sim2diss(pol_part,method="reverse",to.dist=TRUE)
diss_pol_part
```

```
##        PN   PC   FA PERI  PVA PDLG
## PC   0.35
## FA   0.95 0.85
## PERI 0.85 0.35 0.35
## PVA  1.00 1.00 0.55 0.25
## PDLG 0.45 0.35 0.95 0.95 0.85
## CA   0.20 0.25 1.00 1.00 0.95 0.35
```

```
max(diss_pol_part)
```

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

```
min(diss_pol_part)
```

```
## [1] 0.2
```

We can see that, for example "Cabildo Abierto" and "Frente Amplio" the most left and right parties have a dissimiliarity equal to 1. "CA" also has a 100% dissimilarity with the ecologist party (PERI) as they really have nothing in common, they are completly one against the other. Meanwhile, the rest have values between 0 and 1.

Let´s now find the first k=6 principal coordinates.

```
mds_parties <- cmdscale(diss_pol_part,k=6,eig=TRUE)
```

```
## Warning in cmdscale(diss_pol_part, k = 6, eig = TRUE): only 5 of the first 6
## eigenvalues are > 0
```

```
mds_parties
```

```
## $points
##            [,1]        [,2]        [,3]        [,4]         [,5]
## PN   -0.3952759  0.05091707 -0.10548711  0.16296922 -0.079195989
## PC   -0.2529878  0.31628395  0.10216454 -0.07016692  0.024670742
## FA    0.4979302  0.03165600 -0.28922439 -0.08755142  0.009076949
## PERI  0.4565528  0.25604483  0.13642981  0.03405676 -0.007809024
## PVA   0.5100942 -0.32032096  0.13427219  0.07917889  0.003755835
## PDLG -0.3537611 -0.21955354  0.06883832 -0.18827527 -0.046291609
## CA   -0.4625524 -0.11502735 -0.04699337  0.06978874  0.095793096
##
## $eig
## [1]  1.275919e+00  3.332297e-01  1.488051e-01  8.689482e-02  1.835736e-02
## [6] -6.106227e-16 -2.832056e-01
##
## $x
## NULL
##
```

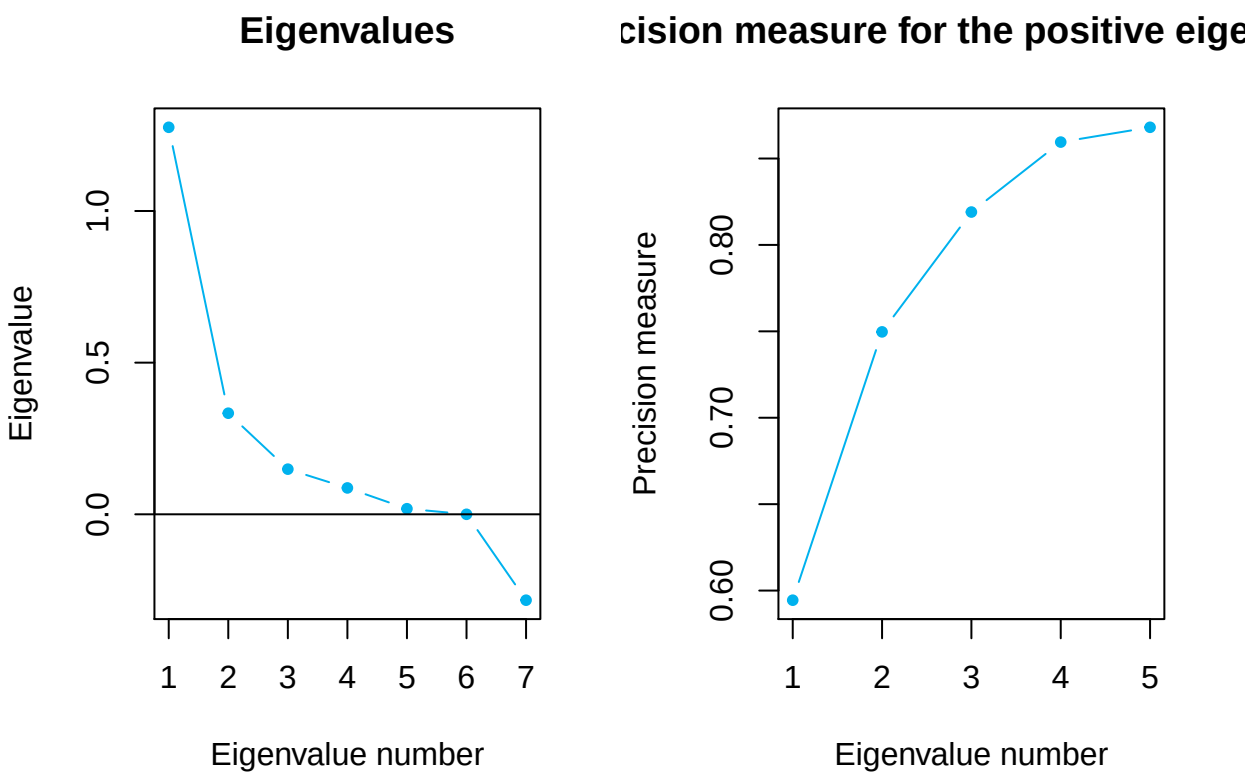
```
## $ac
## [1] 0
##
## $GOF
## [1] 0.8680562 1.0000000
```

As we can see, 5 out of the 6 eigenvalues are negative, so we will just work with these 5. We can now find the precision measures for the 5 possitive eigenvalues.

```
mds.m <- cumsum(mds_parties$eig[1:5]/sum(abs(mds_parties$eig)))
mds.m
```

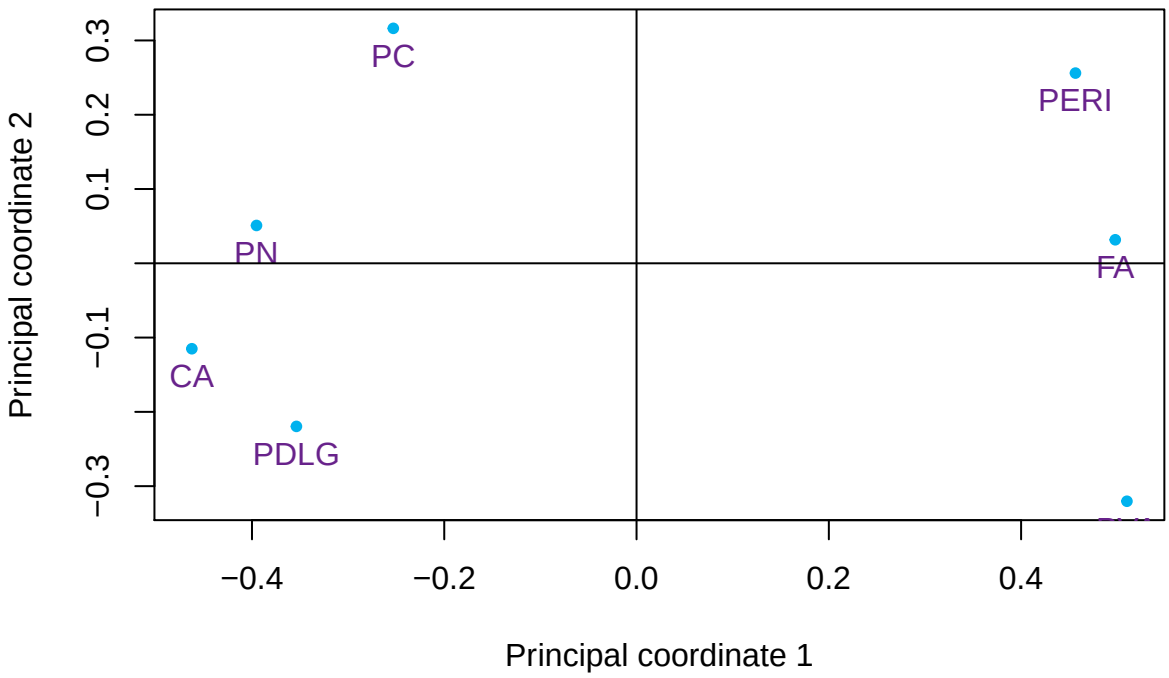
```
## [1] 0.5944428 0.7496925 0.8190199 0.8595036 0.8680562
```

```
par(mfrow=c(1,2))
plot(1:7,mds_parties$eig,type="b",col=color_1,pch=20,xlab="Eigenvalue number",ylab="Eigenvalue",main="E
abline(h=0)
plot(1:5,mds.m,type="b",col=color_1,pch=20,xlab="Eigenvalue number",ylab="Precision measure",main="Preci
```



As we can see, with two eigenvalues we explain more than 75% of the variability of our dataset. We can plot a perceptual map:

```
par(mfrow=c(1,1))
plot(mds_parties$points[,1],mds_parties$points[,2],xlab="Principal coordinate 1",ylab="Principal coordi
text(mds_parties$points[,1],mds_parties$points[,2],labels=rownames(mds_parties$points),col=color_4,pos=
abline(v=0,h=0)
```



As we can see, our first principal coordinate divides clearly between parties that are left wing and the ones that are right wing. This is why PERI, FA, and PVA are on the right side and PC, PN, CA, PDLG on the other side (it´s funny, left and right are exchanged in the plot, but just as a coincidence).

The second coordinate seems to be creating a separation between the parties that are more focused on the countriside (or have more support there), while others do not. "PC" and "PERI" have in common that they are both concerned about the countryside and are always talking about ways of making the countryside more productive. Meanwhile "PDLG" is very related to the cities and always proposses projects to bring people to the cities from the country, while "PVA" is against animal exploitation (the first industry of the country) so it is very rejected outsude the cities. So we could say that this second coordinate meassures the acceptance in the countryside.

Correspondance Analysis

The correspondence analysis provides tools for analyzing the associations between rows and columns of contingency tables.So, the main idea of this analysis is to develop simple indices that can show the relatoinships between the row and the columns categories. These indices, will tell us simultaneously which column categories actually have more weight in a row category and vice-versa. We will finally use two factors (indices), to show the result in a two dimensional plot, showing the relationships between the rows and the columns of the table.

First, we will add the data provided by the professor on health.

```
health <- matrix(c(243,789,167,18,6,220,809,164,35,6,147,658,181,41,8,90,469,236,50,16,53,414,306,106,3

row.names(health)<-c("16-24","25-34","35-44","45-54","55-64","65-74","75+")
```

```
colnames(health)<-c("VG","G","R","B","VB")

health <- as.table(health)

health
```

```
##        VG   G   R   B  VB
## 16-24 243 789 167  18   6
## 25-34 220 809 164  35   6
## 35-44 147 658 181  41   8
## 45-54  90 469 236  50  16
## 55-64  53 414 306 106  30
## 65-74  44 267 284  98  20
## 75+    20 136 157  66  17
```

```
sum(health)
```

```
## [1] 6371
```

As we can see, we have a contingency table of 6371 individuals that are classified between ages (intervals) and their state of health being:

-VG : “very good” -G: “good” -R: “Fair” -B: “Bad” -VB: “Very Bad”

Let´s plot a baloon plot to see ghrpahically the magnitude of each frequency.

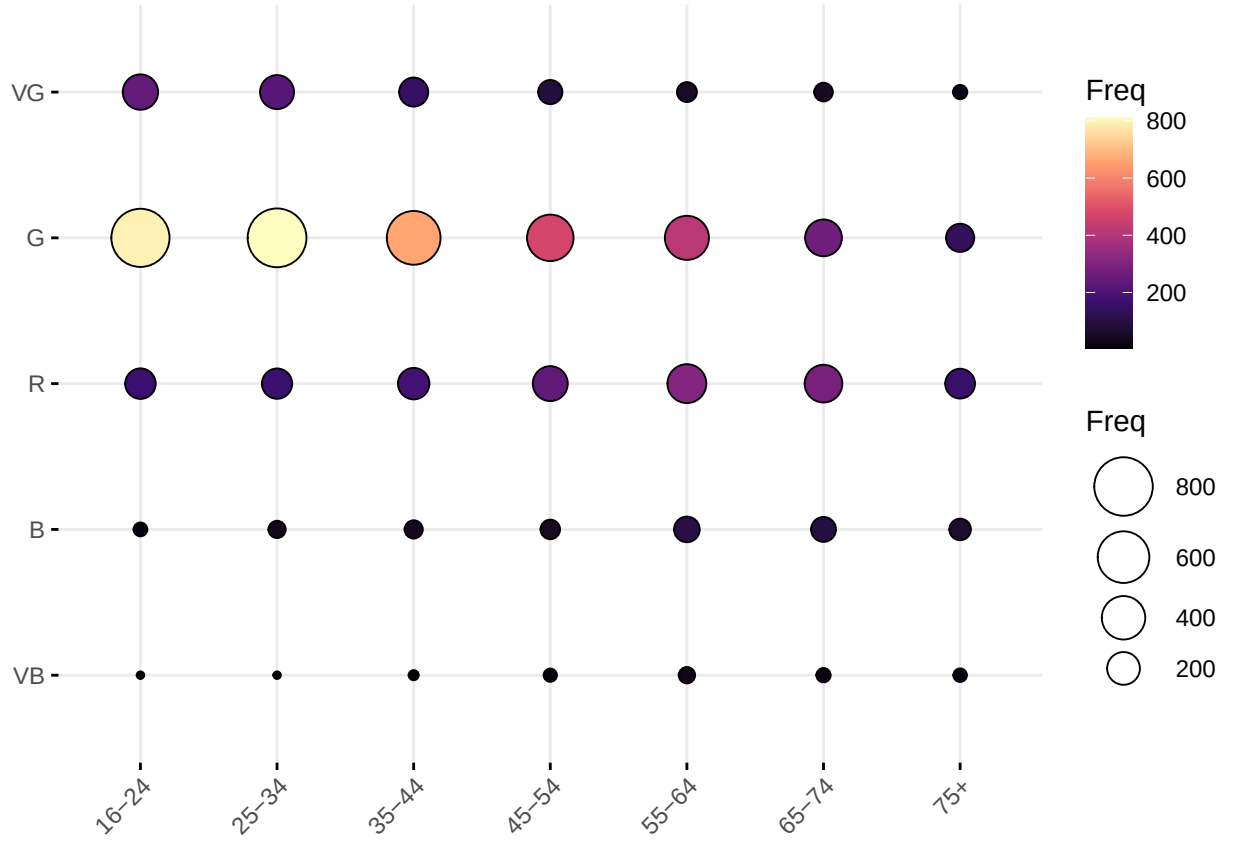
```
library(ggpubr)
```

```
## Warning: package 'ggpubr' was built under R version 4.0.3
```

```
##
## Attaching package: 'ggpubr'
```

```
## The following object is masked from 'package:ape':
##
##     rotate
```

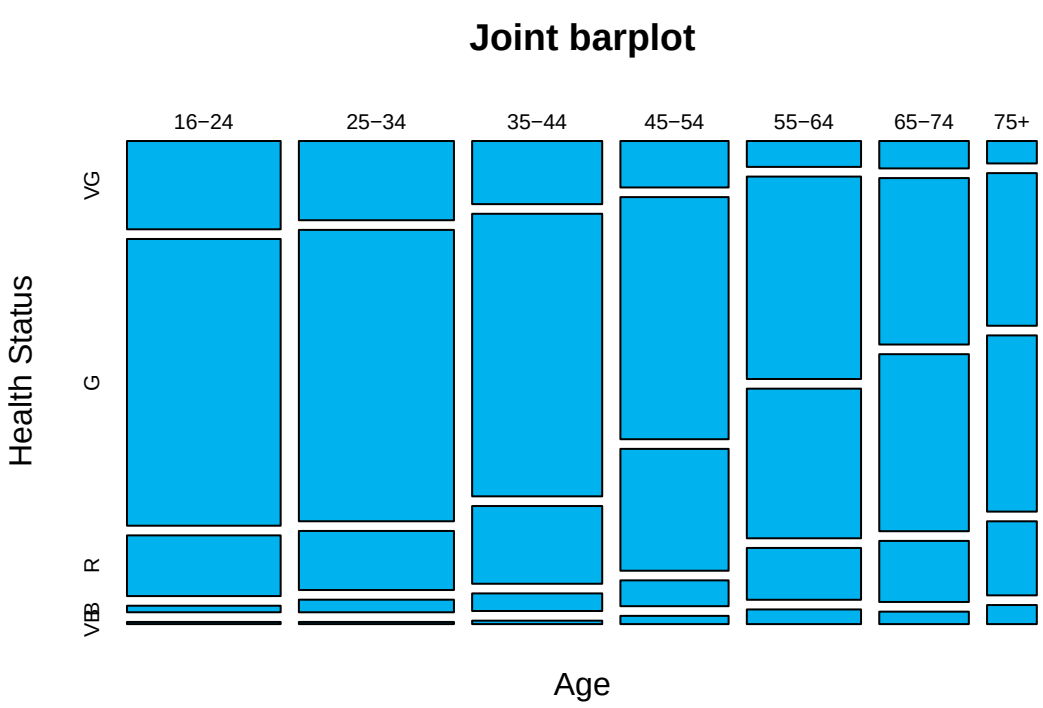
```
health_df <- as.data.frame(health)
ggballoonplot(health_df,fill="value")+scale_fill_viridis_c(option="A")
```



As we can see in the ballonplot, the classes are not similarly distributed in our contingency table. We can see that most of our dataset are young people (aged between 16-24 and 25-34) whose health is good. What is more, most of our dataset is composed by people felling good while a minority has a very bad health (specially when young). We can say that there are more young people than old people in our dataset.

We can do a joint barplot with the two variables to check on the distribution of our data.

```
plot(health,xlab="Age",ylab="Health Status",col=color_1,main="Joint barplot")
```



This plot can help us check if there is any kind of relationship between the two variables. We can see from this joint barplot that for people that have a good health the width of their bars change with the age but not so much. Specially with the people that have a very bad health, we see that the classes look very homogeneous. In the people that have a bad health the classes look homogeneous until the people start getting older, so in the larger ages the bars are bigger. All in all, we can say that as we check people who have better health, the classes become more homogeneous, in some way. Although the relationship is not extremely clear we can say that for sure the two variables are somehow related. This is something trivial: we have every reason to believe that "age" and "health" are dependant variables, as it is to expect that health deteriorates while age increases.

The intention now is to find quantitative variables that represent the classes obtained by the qualitative variables here described. To do so, first we have to determine if both qualitative variables are independent or not. This is because the whole intention is to understand the relationship between two qualitative variables, but if we prove they are independendant, there we do not have to follow with the exercise.

Even though common sense would say that these vairables have a dependance relationship, let´s study it in depth.

This way, we will test the independency of the two qualitative variables by contrasting the null hypotesis ("there is no independance between the variables") and the alternative hypotesis ("there exists some dependency, unknown, between the variables"). We will, then, measure the divergence between the correspondence table and the one we should expect under the null hypotesis. In other words, we will compare the relative frequencies (total frequencies divided by the total number of observations) and the ones we would obtain if we would consider there ir independence between the variables (we would multiply the relative frequencies) and then we will use a Chi-square statistic (this is how the distribution behaves under H0) to see if we reject or not the null hypotesis.

First, let´s produce our relative contingency table, and check that the sum of coumns and rows is 1.

```
P <- prop.table(health)
addmargins(P)
```

```
##                  VG            G            R            B           VB
## 16-24 0.0381415790 0.1238424109 0.0262125255 0.0028253022 0.0009417674
## 25-34 0.0345314707 0.1269816355 0.0257416418 0.0054936431 0.0009417674
## 35-44 0.0230733009 0.1032804897 0.0284099827 0.0064354105 0.0012556898
## 45-54 0.0141265108 0.0736148171 0.0370428504 0.0078480615 0.0025113797
## 55-64 0.0083189452 0.0649819495 0.0480301366 0.0166378904 0.0047088369
## 65-74 0.0069062941 0.0419086486 0.0445769895 0.0153822006 0.0031392246
## 75+   0.0031392246 0.0213467274 0.0246429132 0.0103594412 0.0026683409
## Sum   0.1282373254 0.5559566787 0.2346570397 0.0649819495 0.0161670067
##                 Sum
## 16-24 0.1919635850
## 25-34 0.1936901585
## 35-44 0.1624548736
## 45-54 0.1351436195
## 55-64 0.1426777586
## 65-74 0.1119133574
## 75+   0.0621566473
## Sum   1.0000000000
```

```
health_ct <- chisq.test(health)
health_ct
```

```
##
##  Pearson's Chi-squared test
##
## data:  health
## X-squared = 894.86, df = 24, p-value < 2.2e-16
```

As our p-value is lower than 0.01 we can say that at a confidence level of 99% we can have enough evidence to reject the null hypotesis of independence and believe there may be some kind of dependency between our observable variables. There is clearly a significance dependance between the two variables.

Let´s do correspondance analysis now.

Our objective is to first define the distances between the classes (separately) and then get a configuration of points such that their distances is similar to the one between classes.

In order to first define the distnces between classes, we have to get rid of the effect that will come up for the fact that each row and column have very different sums. As we saw before, the classes are unbalances: most of the people in our dataset are young, for instance. In order to do so, we will do a kind of normalization of the relative frequency matrix, getting what is called a matrix of row profiles, dividing each element by the total sum of the row. Then we will use the Chi squared distance, using a distance that resembles a lot to the Mahalanobich distance. This way we perform kind of a double standarization. We then do the same with a matrix of columns profiles.

```
trf_r <- rowSums(P)
trf_r
```

```
##      16-24      25-34      35-44      45-54      55-64      65-74        75+
## 0.19196358 0.19369016 0.16245487 0.13514362 0.14267776 0.11191336 0.06215665
```

```
trf_c <- colSums(P)
trf_c
```

```
##         VG          G          R          B         VB
## 0.12823733 0.55595668 0.23465704 0.06498195 0.01616701
```

We can calculate the Row and columns profiles

```
D_r <- diag(trf_r)
D_r
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]       [,7]
## [1,] 0.1919636 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.00000000
## [2,] 0.0000000 0.1936902 0.0000000 0.0000000 0.0000000 0.0000000 0.00000000
## [3,] 0.0000000 0.0000000 0.1624549 0.0000000 0.0000000 0.0000000 0.00000000
## [4,] 0.0000000 0.0000000 0.0000000 0.1351436 0.0000000 0.0000000 0.00000000
## [5,] 0.0000000 0.0000000 0.0000000 0.0000000 0.1426778 0.0000000 0.00000000
## [6,] 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.1119134 0.00000000
## [7,] 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.06215665
```

```
D_c <- diag(trf_c)
D_c
```

```
##           [,1]      [,2]     [,3]       [,4]       [,5]
## [1,] 0.1282373 0.0000000 0.000000 0.00000000 0.00000000
## [2,] 0.0000000 0.5559567 0.000000 0.00000000 0.00000000
## [3,] 0.0000000 0.0000000 0.234657 0.00000000 0.00000000
## [4,] 0.0000000 0.0000000 0.000000 0.06498195 0.00000000
## [5,] 0.0000000 0.0000000 0.000000 0.00000000 0.01616701
```

We can now compute the matrices of row and columns profiles

```
P_r <- solve(D_r) %*% P
P_r
```

```
##              VG         G         R          B           VB
## [1,] 0.19869174 0.6451349 0.1365495 0.01471791 0.004905969
## [2,] 0.17828201 0.6555916 0.1329011 0.02836305 0.004862237
## [3,] 0.14202899 0.6357488 0.1748792 0.03961353 0.007729469
## [4,] 0.10452962 0.5447154 0.2740999 0.05807201 0.018583043
## [5,] 0.05830583 0.4554455 0.3366337 0.11661166 0.033003300
## [6,] 0.06171108 0.3744741 0.3983170 0.13744741 0.028050491
## [7,] 0.05050505 0.3434343 0.3964646 0.16666667 0.042929293
```

```
apply(P_r,1,sum)
```

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

```r
P_c <- solve(D_c) %*% t(P)
P_c
```

```
##           16-24      25-34      35-44     45-54      55-64      65-74
## [1,] 0.29742962 0.26927785 0.17992656 0.1101591 0.06487148 0.05385557
## [2,] 0.22275551 0.22840203 0.18577075 0.1324111 0.11688312 0.07538114
## [3,] 0.11170569 0.10969900 0.12107023 0.1578595 0.20468227 0.18996656
## [4,] 0.04347826 0.08454106 0.09903382 0.1207729 0.25603865 0.23671498
## [5,] 0.05825243 0.05825243 0.07766990 0.1553398 0.29126214 0.19417476
##             75+
## [1,] 0.02447980
## [2,] 0.03839639
## [3,] 0.10501672
## [4,] 0.15942029
## [5,] 0.16504854
```

```r
apply(P_c,1,sum)
```

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

With these matrixes, we will create a matrix M, that will be the result of substracting the expected and observed relative frequencies (under the independance hypotesis), standarized by the relative frequencies corresponding to the rows and the one to the columns (first and second variables).

Then we do a singular value decomposition of this matrix.

```r
M <- diag(1/sqrt(trf_r)) %*% (P - trf_r %*% t(trf_c)) %*% diag(1/sqrt(trf_c))
M
```

```
##             [,1]         [,2]        [,3]         [,4]         [,5]
## [1,]  0.08620063  0.052401995 -0.08873504 -0.086391392 -0.038803691
## [2,]  0.06150416  0.058809131 -0.09244781 -0.063221200 -0.039129176
## [3,]  0.01552300  0.043132623 -0.04973815 -0.040110998 -0.026746522
## [4,] -0.02433773 -0.005542315  0.02993293 -0.009964963  0.006985322
## [5,] -0.07376388 -0.050918059  0.07951736  0.076503529  0.050016099
## [6,] -0.06214794 -0.081424582  0.11302292  0.095098928  0.031265829
## [7,] -0.05411753 -0.071060440  0.08327716  0.099449615  0.052474969
```

```r
M_svd <- svd(M)
M_svd
```

```
## $d
## [1] 3.695985e-01 4.571116e-02 3.594393e-02 2.176889e-02 4.907848e-17
##
## $u
##             [,1]       [,2]       [,3]       [,4]        [,5]
## [1,] -0.43988571  0.4055135  0.3580394  0.4051477 -0.17502756
## [2,] -0.39281241  0.1878872 -0.3264349 -0.2359385  0.15007504
## [3,] -0.21696458 -0.3593238 -0.2881977 -0.4407184 -0.60289704
## [4,]  0.07053349 -0.5698555  0.4570550  0.2872186 -0.32748658
## [5,]  0.40421675 -0.2747270 -0.3757860  0.3289609  0.32688500
```

```
## [6,]  0.48934481  0.2513851  0.4740176 -0.5715127  0.02415671
## [7,]  0.44418561  0.4557838 -0.3276935  0.2619526 -0.60718464
##
## $v
##            [,1]        [,2]       [,3]        [,4]      [,5]
## [1,] -0.4097126  0.76083252  0.3111209  0.16803540 0.3581024
## [2,] -0.4001414 -0.41367632 -0.3220458 -0.09533413 0.7456250
## [3,]  0.5754638 -0.17534580  0.6350704 -0.01112854 0.4844141
## [4,]  0.5207555  0.46808034 -0.5438450 -0.38595996 0.2549156
## [5,]  0.2639651  0.01266605 -0.3168729  0.90199034 0.1271495
```

```
# Define matrix Lambda, Gamma and Theta

Lambda_M <- diag(M_svd$d)
Gamma_M <- M_svd$u
Theta_M <- M_svd$v
```

With this decomposition, we can find both matrixes X_r and X_c (for first and second variable). As both matrixes have the same amount of columns, I can plot them in the same space. We will plot just two of them (two columns of the two matrixes). When we do this, it is important to check how much vairability is explained between the two dimensions together.

```
X_r <- diag(1/sqrt(trf_r)) %*% Gamma_M[,1:2] %*% Lambda_M[1:2,1:2]
X_r
```

```
##             [,1]        [,2]
## [1,] -0.37107411  0.04230757
## [2,] -0.32988430  0.01951487
## [3,] -0.19895401 -0.04075134
## [4,]  0.07091332 -0.07085805
## [5,]  0.39551813 -0.03324647
## [6,]  0.54063511  0.03434953
## [7,]  0.65849263  0.08356749
```

```
X_c <- diag(1/sqrt(trf_c)) %*% Theta_M[,1:2] %*% Lambda_M[1:2,1:2]
X_c
```

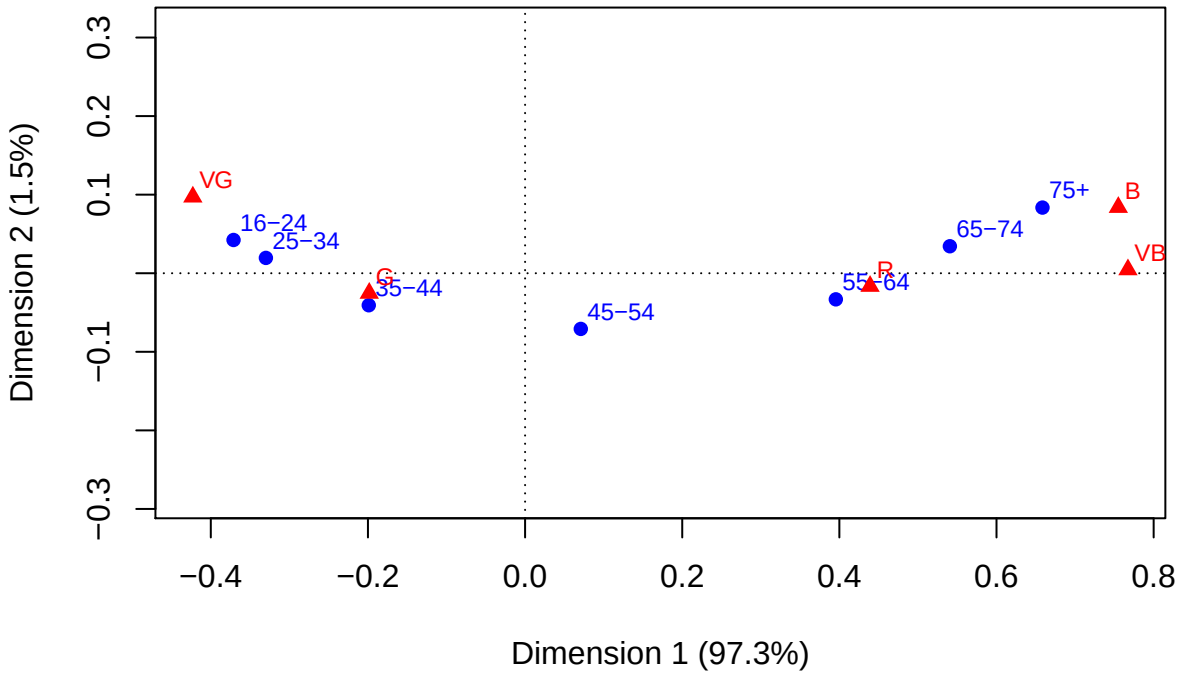
```
##            [,1]         [,2]
## [1,] -0.4228656  0.097118969
## [2,] -0.1983459 -0.025360769
## [3,]  0.4390676 -0.016546296
## [4,]  0.7550362  0.083935611
## [5,]  0.7672942  0.004553535
```

We can plot just two of them (two columns of the two matrixes). When we do this, it is important to check how much vairability is explained between the two dimensions together. We can also use the package "ca" (easier and straightforward) to do the correspondance analysis, and this way we can avoid all the preovious coding, and just plot the results.

```
library(ca)

ca_health <- ca(health)
ca_health
```

```
##
##  Principal inertias (eigenvalues):
##            1        2       3        4
## Value      0.136603 0.00209 0.001292 0.000474
## Percentage 97.25%   1.49%   0.92%    0.34%
##
##
##  Rows:
##              16-24     25-34     35-44     45-54     55-64    65-74      75+
## Mass     0.191964  0.193690  0.162455  0.135144  0.142678 0.111913 0.062157
## ChiDist  0.375172  0.331740  0.206084  0.111067  0.398971 0.545384 0.665846
## Inertia  0.027020  0.021316  0.006900  0.001667  0.022711 0.033288 0.027557
## Dim. 1  -1.003992 -0.892548 -0.538298  0.191866  1.070129 1.462763 1.781643
## Dim. 2   0.925541  0.426917 -0.891497 -1.550126 -0.727316 0.751447 1.828164
##
##
##  Columns:
##                VG         G         R        B       VB
## Mass     0.128237  0.555957  0.234657 0.064982 0.016167
## ChiDist  0.435117  0.200582  0.441899 0.764259 0.787803
## Inertia  0.024279  0.022368  0.045823 0.037955 0.010034
## Dim. 1  -1.144121 -0.536652  1.187958 2.042855 2.076021
## Dim. 2   2.124623 -0.554805 -0.361975 1.836217 0.099615
```

```
plot(ca_health)
```



What can we conclude from this plot?

First of all, with dimension one we can explain 97.3% of the variability of our data. With the two together we can explain 98.8% of the total variability of the data. This is an outstanding result. This means that the summary is not leaving out information about the data.

Second of all, from the structure we can see of the plot, it is obvious there is a clear relationship between some rows and some columns. Let´s see what are the characteristics of it:

1- In general the proximity of two rows (two columns) indicates a similar profile in these two rows (columns). By profile I reffer to the conditional frequency distribution of the row (or column). In other words, rows 16-24 and 25-34 are almost proportional, they have very similar profile.

2- in general, the proximity of a particular row to a particular column indicates that one has an important weight in the other. On the contrary, a row that is very distant from a particlar column may indicate that there almost no observations in this column for this row. From this we can say:

i) The health category "very good" is very related to the two first intervals of age, that is the youngest people. This is coherent with our first analysis.
ii) Good health is specially associated with the category of people between 35-44 years old.
iii) Label "fair" has a big weight (is very related) to the interval 55-64 years old.
iv) People feeling very bad are specially old people.

3 - In general, the origin is the average of the factors. This is why a label projected close to the origin indicates an avarage profile. In particular, people between 45-54 years old are not so distinct in the different helth categories, relatively speaking.

4- In general, labels that are in the opposite side of the origin tend to be negatively associated one to the other. As categories "bad" and "very bad" are on opposite sides of the origine to categories describing young people, we can say that these lables are probably negatively associated (young people and bad health).