Project

Jorge Mendez-Benegassi

2023-04-04

R Markdown

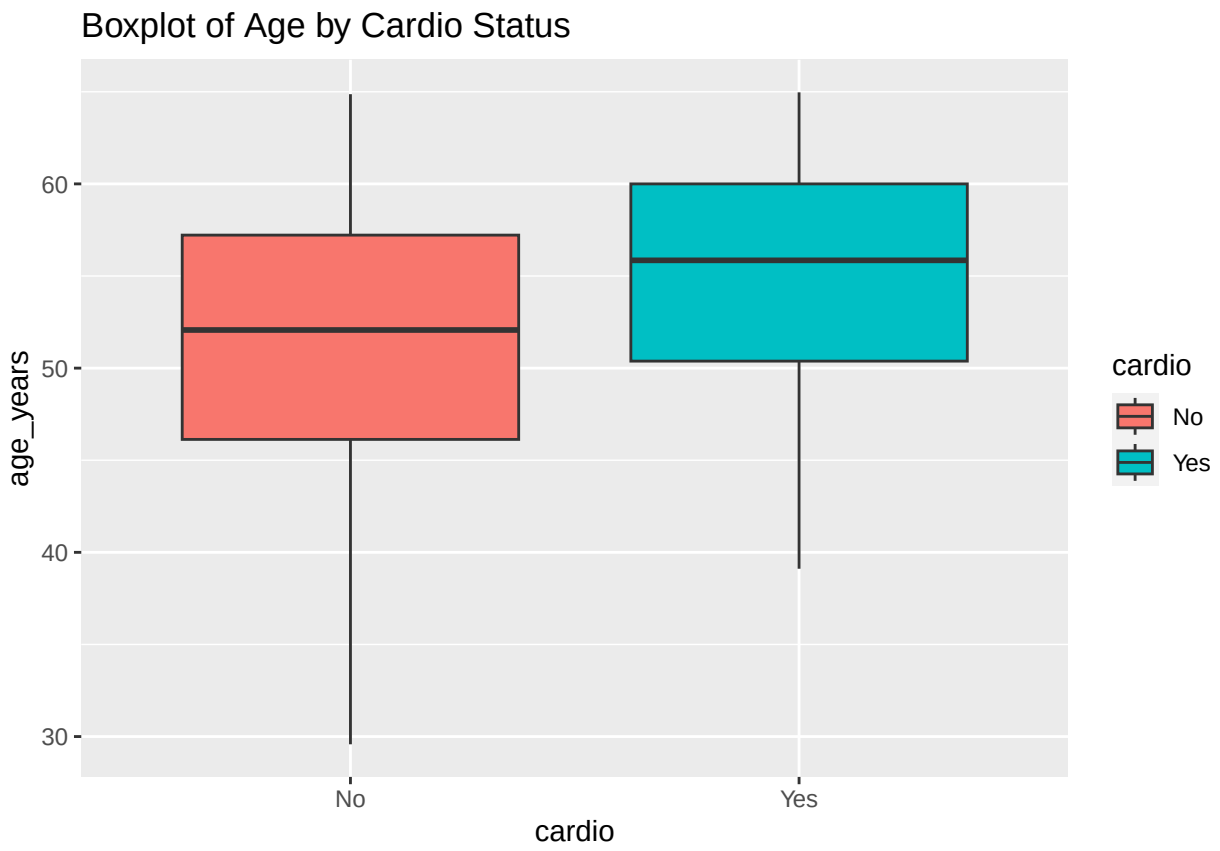
```
setwd("C:/Users/augus/Documents/Chicago/IIT/CSP-571 - Data Preparation/Project")
library(ggplot2)


# Load dataset
df <- read.csv("heart_data.csv")

df$age_years <- round(df$age / 365, 2)

df <- subset(df, select = -c(age, index, id))

missing_values <- sum(is.na(df))
paste("There are", missing_values, "missing values in the dataset.")
```

```
## [1] "There are 0 missing values in the dataset."
```

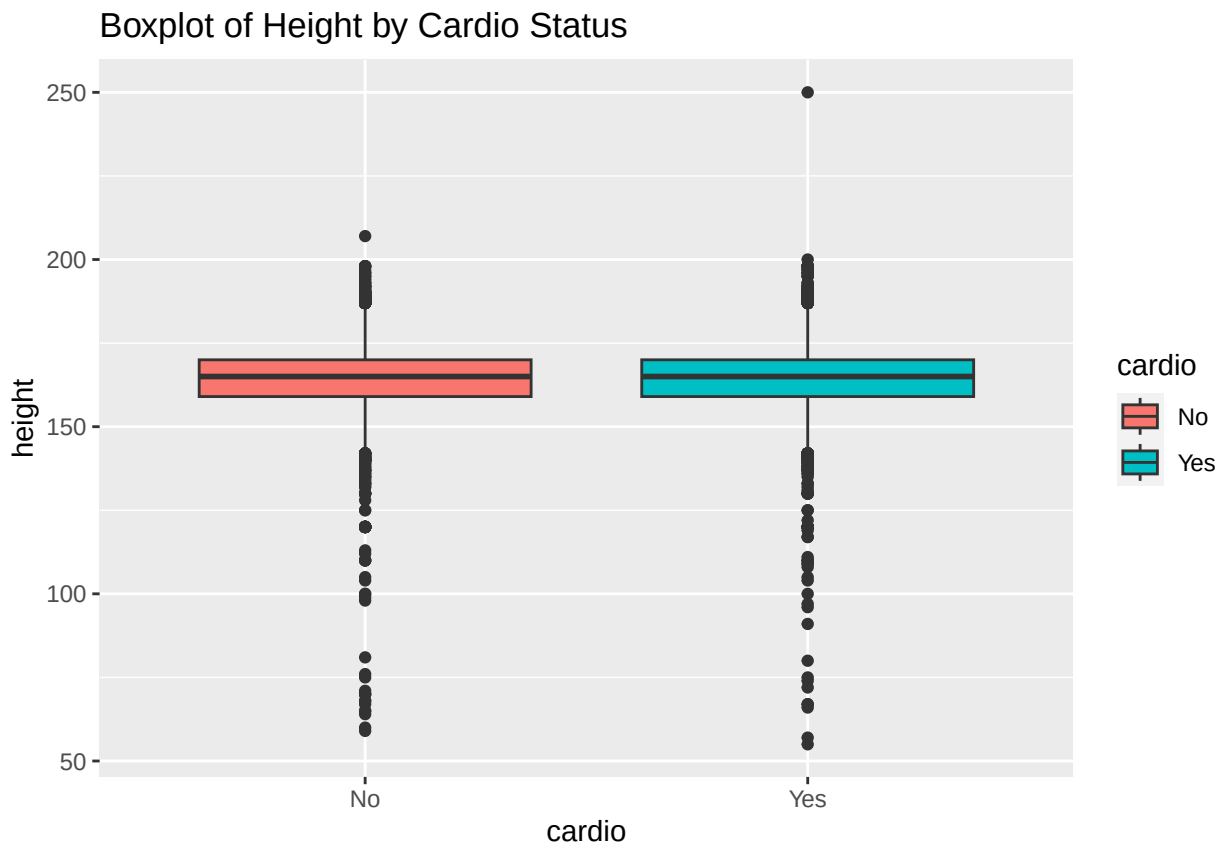
```
duplicate_rows <- duplicated(df)

# Subset the dataset into another with the duplicated rows
duplicate_df <- df[duplicate_rows,]
num_duplicates <- nrow(duplicate_df)
paste("There are", num_duplicates, "duplicates in the dataset.")
```

```
## [1] "There are 77 duplicates in the dataset."
```

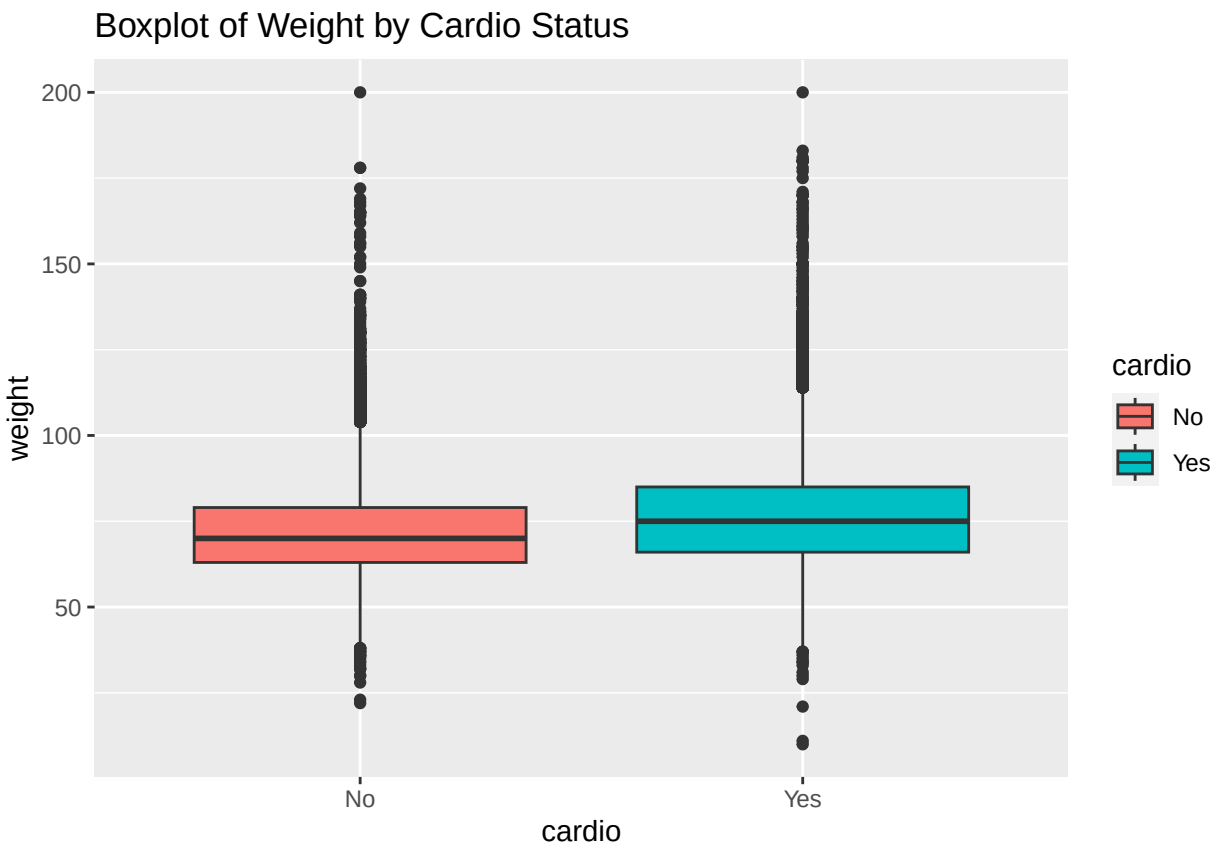
```
# Find outliers / noisy data in numerical (non-binary) features
# We first modify the variable "cardio" to be a categorical variable with two levels
df$cardio <- factor(df$cardio, levels = c(0, 1), labels = c("No", "Yes"))

# We extract the boxplot of the non-binary or non-level features
ggplot(data=df, aes(x=cardio, y=age_years, fill=cardio)) +
  geom_boxplot() +
  ggtitle("Boxplot of Age by Cardio Status")
```



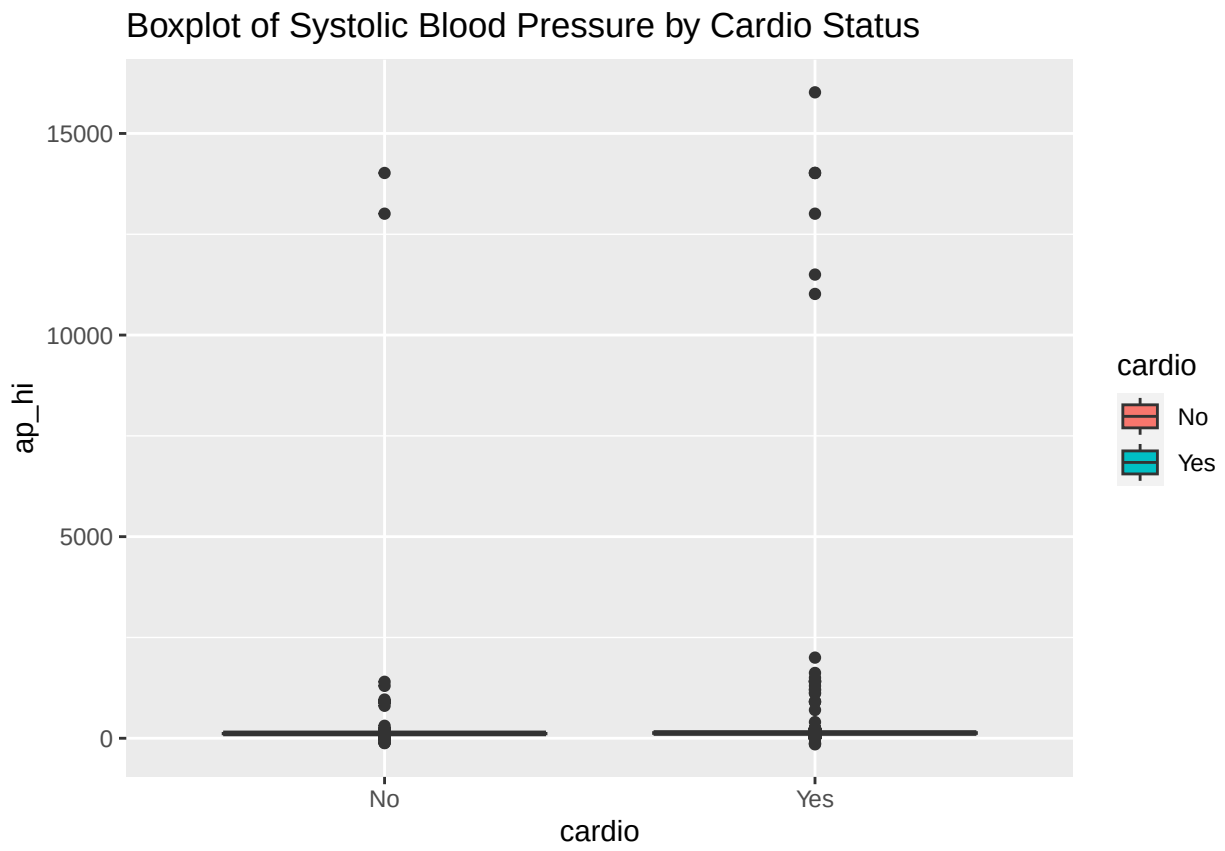

```
ggplot(data=df, aes(x=cardio, y=height, fill=cardio)) +
  geom_boxplot() +
  ggtitle("Boxplot of Height by Cardio Status")
```



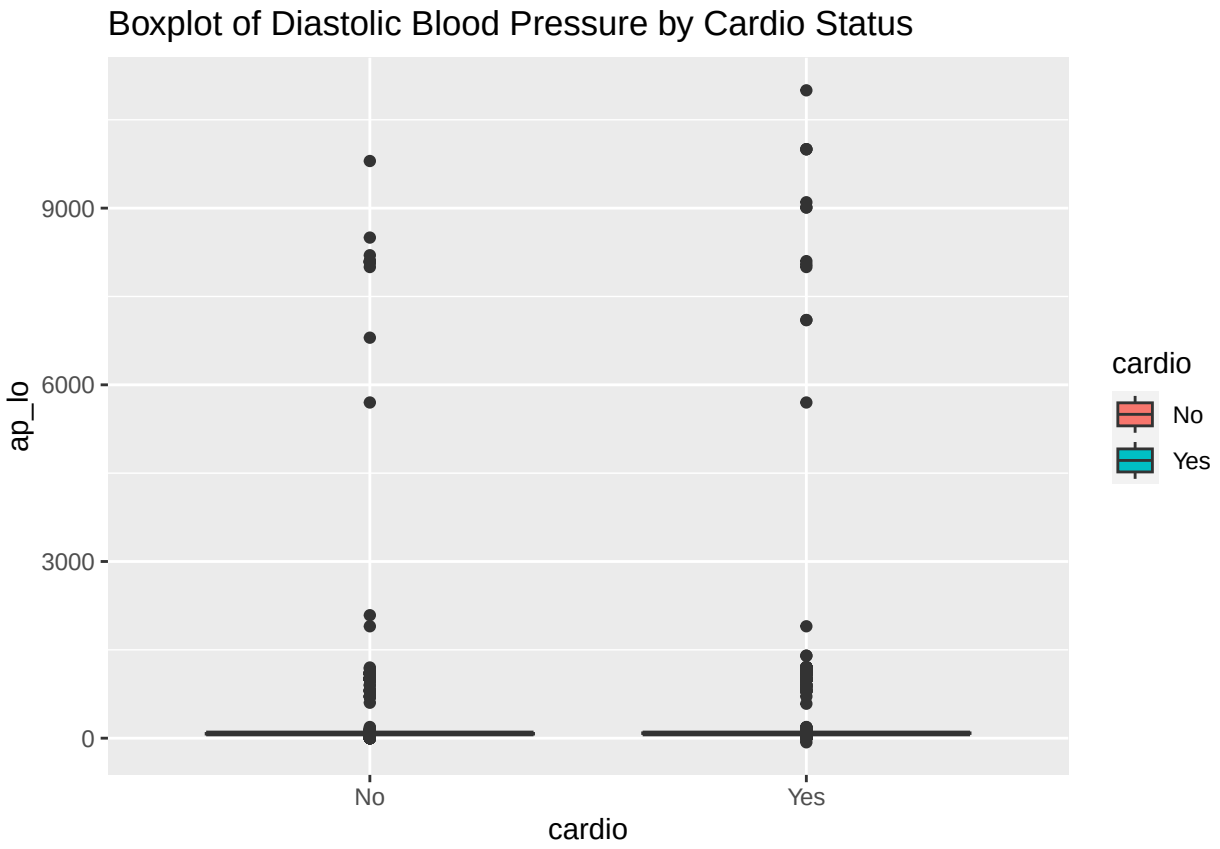

```
ggplot(data=df, aes(x=cardio, y=weight, fill=cardio)) +
  geom_boxplot() +
  ggtitle("Boxplot of Weight by Cardio Status")
```




```
ggplot(data=df, aes(x=cardio, y=ap_hi, fill=cardio)) +
  geom_boxplot() +
  ggtitle("Boxplot of Systolic Blood Pressure by Cardio Status")
```




```
ggplot(data=df, aes(x=cardio, y=ap_lo, fill=cardio)) +
  geom_boxplot() +
  ggtitle("Boxplot of Diastolic Blood Pressure by Cardio Status")
```



Outliers

```
cat("\nTo find anomalies, we can visually examine the boxplots and look for any data points that fall fa
```

```
##
## To find anomalies, we can visually examine the boxplots and look for any data points that fall far ou
```

```
cat("To find anomalies, we can visually examine the boxplots and look for any data points that fall far
```

```
## To find anomalies, we can visually examine the boxplots and look for any data points that fall far ou
```

```
cat("There are no anomalies for the feature age.")
```

```
## There are no anomalies for the feature age.
```

```
cat("There are anomalies for the feature height.")
```

```
## There are anomalies for the feature height.
```

```r
cat("There are anomalies for the feature weight.")
```

```
## There are anomalies for the feature weight.
```

```r
cat("There are anomalies for the feature ap_hi.")
```

```
## There are anomalies for the feature ap_hi.
```

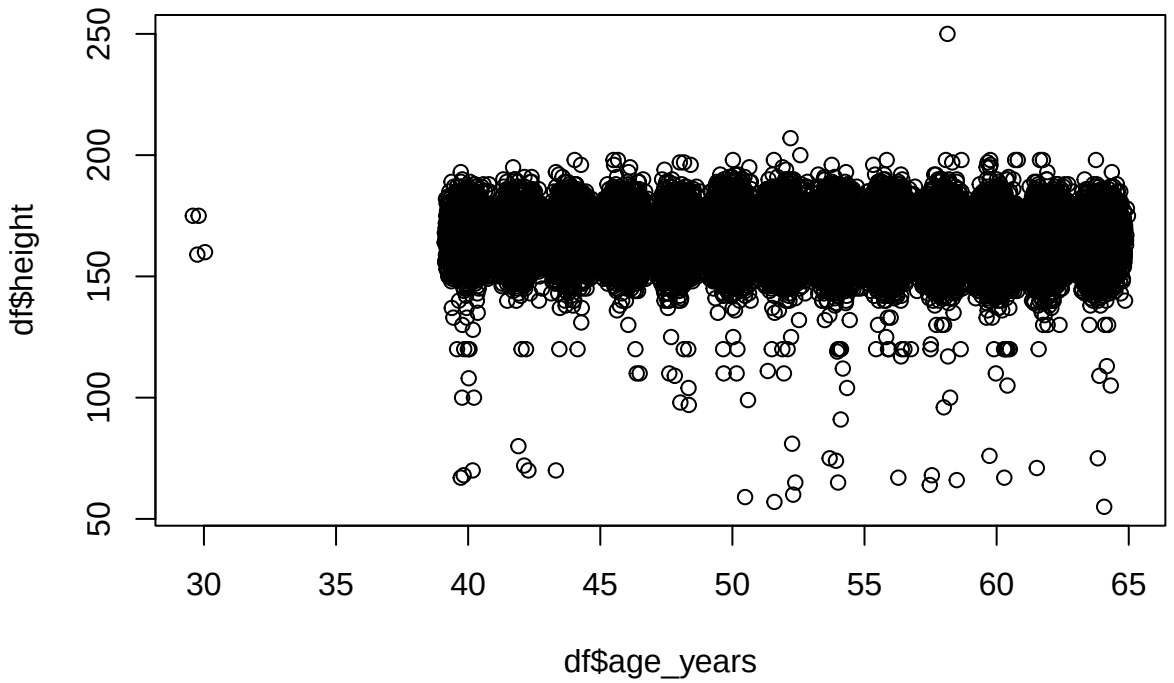
```r
cat("There are anomalies for the feature ap_lo.")
```

```
## There are anomalies for the feature ap_lo.
```

```r
iqr_height <- IQR(df$height)
iqr_weight <- IQR(df$weight)

paste("We may consider outliers of the features height and weight to any data point that falls more thar
```

```
## [1] "We may consider outliers of the features height and weight to any data point that falls more tha
```

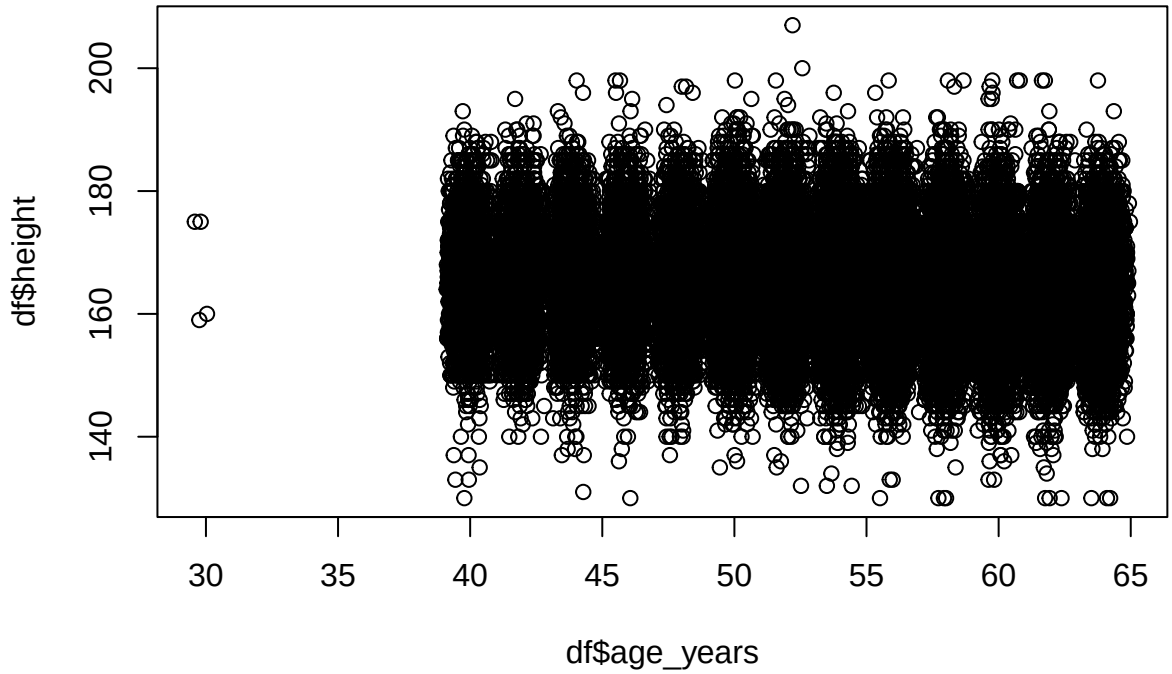
```r
# To find outliers in height feature we are going to plot a scatter between age and height:
# Create scatter plot
plot(df$age_years, df$height)
```



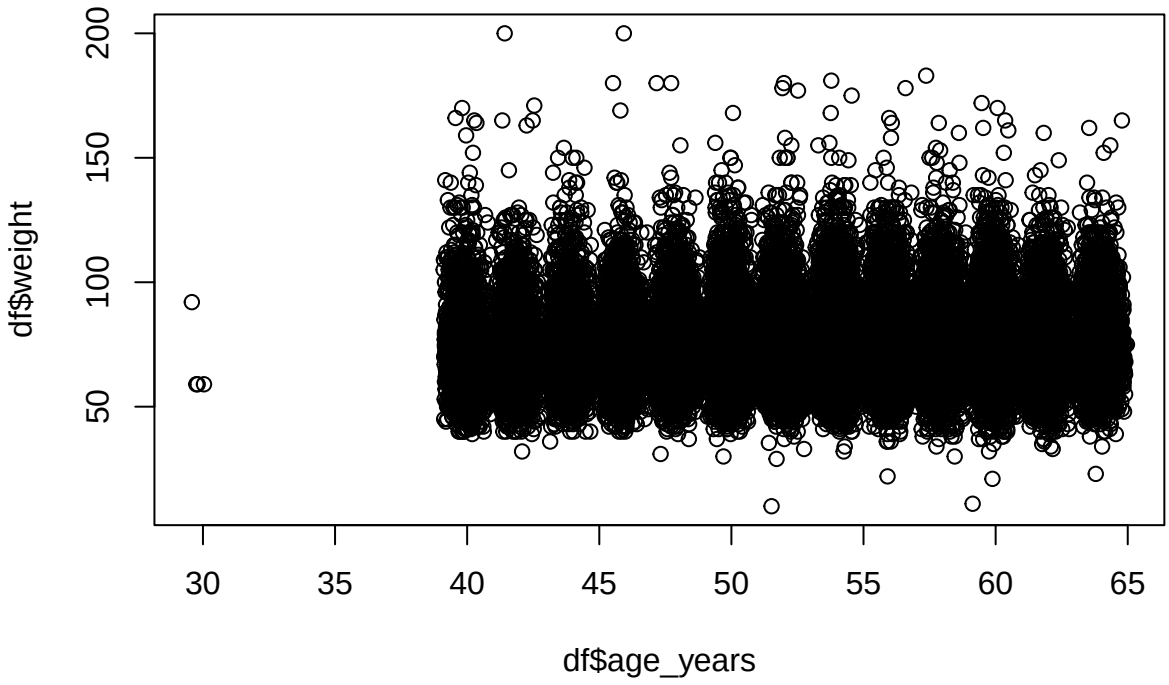
```r
min_age <- min(df$age_years)
max_age <- max(df$age_years)
paste("Since we have ages between", min_age, "and", max_age,", we will consider height outliers to any d
```

```
## [1] "Since we have ages between 29.58 and 64.97 , we will consider height outliers to any data point
```

```r
# Rows with height values outside the range (outliers)
height_outliers <- df$height < 130 | df$height > 210

# Remove those rows
df <- df[!height_outliers, ]

# Scatter plot without those outliers
plot(df$age_years, df$height)
```




```r
# To find outliers in weight feature we are going to plot a scatter between age and weight:
# Create scatter plot
plot(df$age_years, df$weight)
```




```
cat("As we can see in the scatter plot, there are no significant outliers according to the weight. There
```

```
## As we can see in the scatter plot, there are no significant outliers according to the weight. There n
```

```
cat("We consider outliers of the feature ap_hi (systolic) to those values higher than 180 mmHg which wou
```

```
## We consider outliers of the feature ap_hi (systolic) to those values higher than 180 mmHg which would
```

```
# Rows with ap_hi (systolic) values outside the range (outliers)
aphi_outliers <- df$ap_hi > 180
df <- df[!aphi_outliers, ]

cat("We consider outliers of the feature ap_lo (diastolic) to those values higher than 120 mmHg which wo
```

```
## We consider outliers of the feature ap_lo (diastolic) to those values higher than 120 mmHg which woul
```

```
# Rows with ap_lo (diastolic) values outside the range (outliers)
aplo_outliers <- df$ap_lo > 120
df <- df[!aplo_outliers, ]
```

Data reduction and Data transformation

```r
cat("We have reduced our dataset from 14 features to 12, and from 70000 observations to 68561.")
```

```
## We have reduced our dataset from 14 features to 12, and from 70000 observations to 68561.
```

```r
cat("We have alreay transform the age feature so that it gives us the information in years instead of da
```

```
## We have alreay transform the age feature so that it gives us the information in years instead of days
```

```r
# We transform the feature gender to be binary (1->0 and 2->1)
df$gender <- ifelse(df$gender == 1, 0, 1)

# We transform the categorical features cholesterol, gluc from 1 to 0, from 2 to 1, and from 3 to 2
df$cholesterol <- ifelse(df$cholesterol == 1, 0,
                         ifelse(df$cholesterol == 2, 1, 2))
df$gluc <- ifelse(df$gluc == 1, 0,
                         ifelse(df$gluc == 2, 1, 2))

df$cardio <- ifelse(df$cardio == "Yes", 1, 0)


################################################################
################################################################
# creating dummy variables for cholesterol and gluc
cat("It is necessary to transform all categorical values to factors for future model implementation. Th
```

```
## It is necessary to transform all categorical values to factors for future model implementation. This
```

```r
df$cholesterol = factor(df$cholesterol)
df$gluc = factor(df$gluc)

cholesterol_dummies <- model.matrix(~ cholesterol - 1, data = df)
gluc_dummies <- model.matrix(~ gluc - 1, data = df)

# adding the dummy variables to the dataset
df <- cbind(df, cholesterol_dummies)
df <- cbind(df, gluc_dummies)

#df <- subset(df, select = -c(gluc, cholesterol))

cat("For convenience, we are going to change the order of the features so that the target is last.")
```

```
## For convenience, we are going to change the order of the features so that the target is last.
```

```r
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.2.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
df <- select(df, gender, height, weight, ap_hi, ap_lo, cholesterol, cholesterol0, cholesterol1, choleste

cat("At this point we have five integer features corresponding to: age_years, height, weight, ap_hi, and
```

```
## At this point we have five integer features corresponding to: age_years, height, weight, ap_hi, and a
```

Discretization and normalization

Statistics

Repartition statistics of features

```
summary_data <- summary(df)
smokers <- sum(df$smoke == 1)
percentage_smokers <- smokers*100/nrow(df)
cat("\nThe percentage of smokers is", round(percentage_smokers,2),"%.")
```

Statistics of categorical features

```
##
## The percentage of smokers is 8.77 %.
```

```
alcohol <- sum(df$alco == 1)
percentage_alcohol <- alcohol*100/nrow(df)
cat("\n\nThe percentage of individuals who drink alcohol is", round(percentage_alcohol,2), "%.")
```

```
##
##
## The percentage of individuals who drink alcohol is 5.32 %.
```

```
men <- sum(df$gender == 1)
cat("\n\nThe number of men is", men)
```

```
##
##
## The number of men is 23880
```

```r
women <- sum(df$gender == 0)
cat("\n\nThe number of women is", women)
```

```
##
##
## The number of women is 44681
```

```r
gluc_normal <- sum(df$gluc == 0)
cat("\n\nThe number of individuals which glucose level is normal", gluc_normal)
```

```
##
##
## The number of individuals which glucose level is normal 58334
```

```r
cat("\n\nThe percentage of individuals which glucose level is normal", round(gluc_normal*100/nrow(df),2
```

```
##
##
## The percentage of individuals which glucose level is normal 85.08 %.
```

```r
gluc_above <- sum(df$gluc == 1)
cat("\n\nThe number of individuals which glucose level is above average", gluc_above)
```

```
##
##
## The number of individuals which glucose level is above average 5028
```

```r
cat("\n\nThe percentage of individuals which glucose level is above average", round(gluc_above*100/nrow
```

```
##
##
## The percentage of individuals which glucose level is above average 7.33 %.
```

```r
gluc_wellAbove <- sum(df$gluc == 2)
cat("\n\nThe number of individuals which glucose level is well above normal", gluc_wellAbove)
```

```
##
##
## The number of individuals which glucose level is well above normal 5199
```

```r
cat("\n\nThe percentage of individuals which glucose level is  well above normal", round(gluc_wellAbove
```

```
##
##
## The percentage of individuals which glucose level is  well above normal 7.58 %.
```

```r
chol_normal <- sum(df$cholesterol == 0)
cat("\n\nThe number of individuals which cholesterol level is normal", chol_normal)
```

```
##
##
## The number of individuals which cholesterol level is normal 51494
```

```r
cat("\n\nThe percentage of individuals which cholesterol level is normal", round(chol_normal*100/nrow(d
```

```
##
##
## The percentage of individuals which cholesterol level is normal 75.11 %.
```

```r
chol_above <- sum(df$cholesterol == 1)
cat("\n\nThe number of individuals which cholesterol level is above average", chol_above)
```

```
##
##
## The number of individuals which cholesterol level is above average 9253
```

```r
cat("\n\nThe percentage of individuals which cholesterol level is above average", round(chol_above*100/
```

```
##
##
## The percentage of individuals which cholesterol level is above average 13.5 %.
```

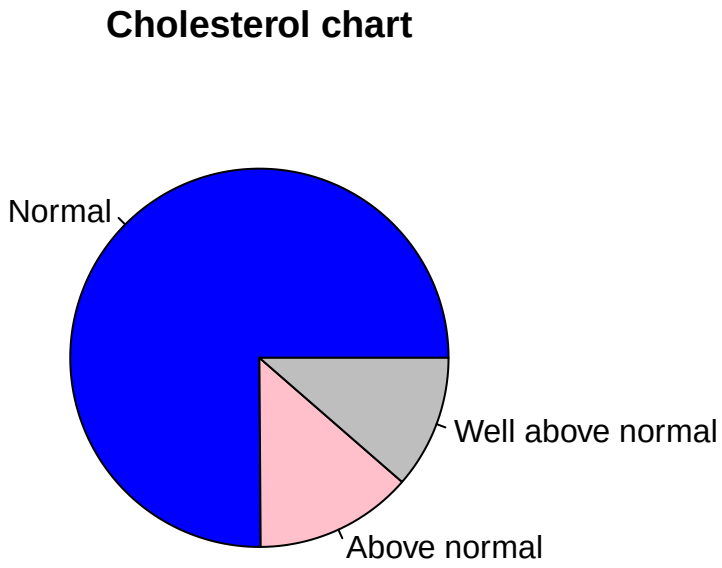
```r
chol_wellAbove <- sum(df$cholesterol == 2)
cat("\n\nThe number of individuals which cholesterol level is well above normal", chol_wellAbove)
```

```
##
##
## The number of individuals which cholesterol level is well above normal 7814
```

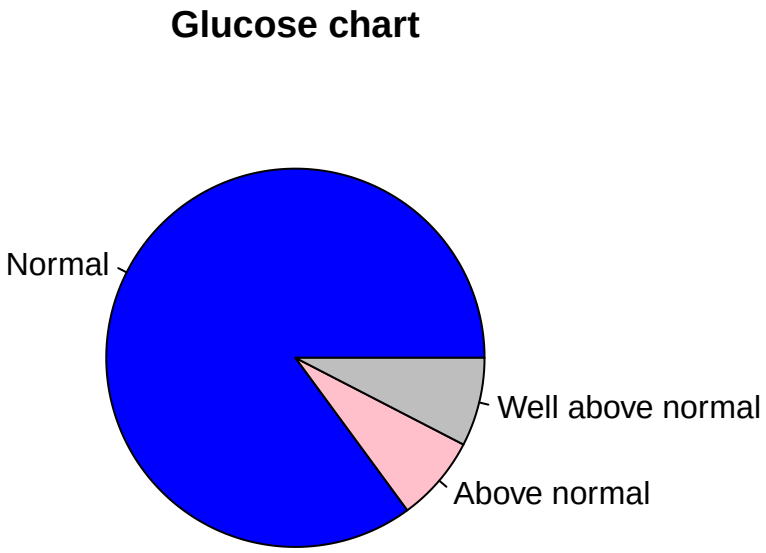
```r
cat("\n\nThe percentage of individuals which cholesterol level is  well above normal", round(chol_wellA
```

```
##
##
## The percentage of individuals which cholesterol level is  well above normal 11.4 %.
```

```r
chol_percentages <- c(chol_normal*100/nrow(df), chol_above*100/nrow(df), chol_wellAbove*100/nrow(df))
labels <- c("Normal", "Above normal", "Well above normal")
chol_colors <- c("blue", "pink", "grey")
pie(chol_percentages, labels = labels, main = "Cholesterol chart", col = chol_colors)
```




```
gluc_percentages <- c(gluc_normal*100/nrow(df), gluc_above*100/nrow(df), gluc_wellAbove*100/nrow(df))
labels <- c("Normal","Above normal","Well above normal")
gluc_colors <- c("blue", "pink", "grey")
pie(gluc_percentages, labels = labels, main = "Glucose chart", col = gluc_colors)
```



Statistics of continuous features

```
min_weight <- min(df$weight)
min_height <- min(df$height)
min_age_years <- min(df$age_years)

max_weight <- max(df$weight)
max_height <- max(df$height)
max_age_years <- max(df$age_years)

mean_weight <- mean(df$weight)
mean_height <- mean(df$height)
mean_age_years <- mean(df$age_years)

library(knitr)
stats_mat <- matrix(c(
  round(min_weight,2), round(mean_weight,2), round(max_weight,2),
  round(min_height,2), round(mean_height,2), round(max_height,2),
  round(min_age_years,2), round(mean_age_years,2), round(max_age_years,2)
), nrow = 3, byrow = TRUE, dimnames = list(
  c("Weight", "Height", "Age"),
  c("Minimum", "Mean", "Max")
))
stats_df <- as.data.frame(stats_mat)
kable(stats_df, format = "markdown")
```

	Minimum	Mean	Max
Weight	11.00	74.06	200.00
Height	130.00	164.44	207.00
Age	29.58	53.32	64.97

Advanced Statistics

```
library(plotly)
```

```
##
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##     last_plot

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

## The following object is masked from 'package:graphics':
##
##     layout
```

```
cardio_cases <- sum(df$cardio == 1)

men_cardio  <- sum(df$gender == 1 & df$cardio == 1)
cat("\n\nThe pertentage of men with cardio problems", round(men_cardio*100/cardio_cases,2), "%")
```

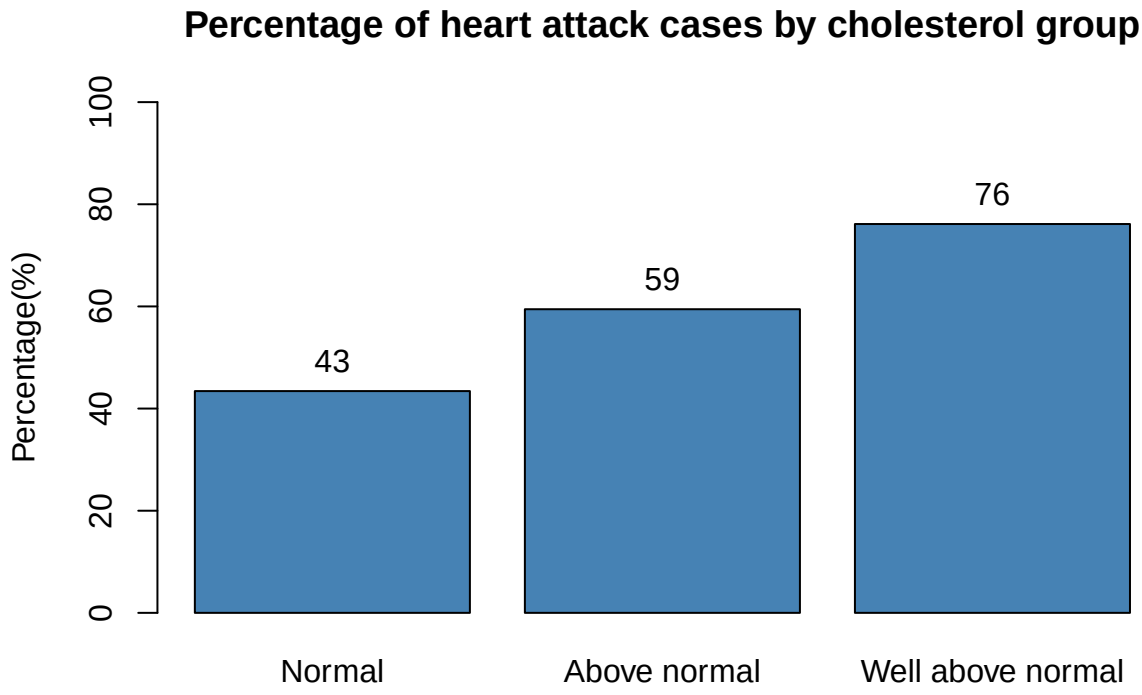
```
##
##
## The pertentage of men with cardio problems 35.15 %
```

```
women_cardio  <- sum(df$gender == 0 & df$cardio == 1)
cat("\n\nThe pertentage of women is with cardio problems", round(women_cardio*100/cardio_cases,2), "%")
```

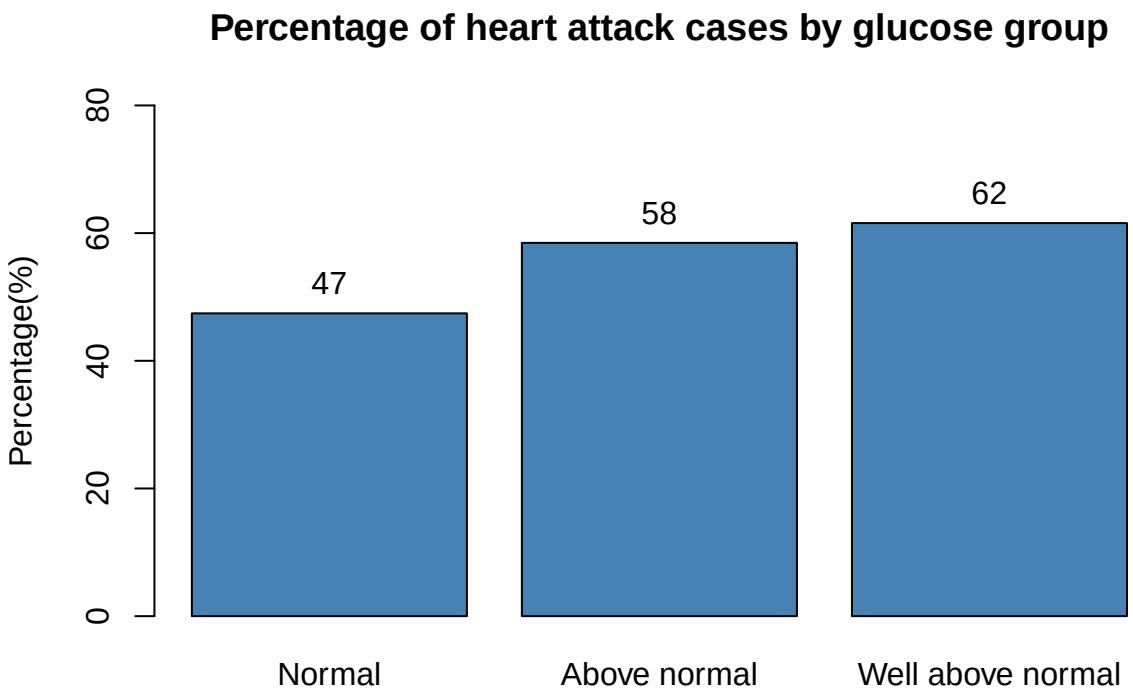
```
##
##
## The pertentage of women is with cardio problems 64.85 %
```

```
gluc_normal_cardio <- sum(df$gluc == 0 & df$cardio == 1) * 100 / gluc_normal

gluc_above_cardio  <- sum(df$gluc == 1 & df$cardio == 1) * 100 / gluc_above

gluc_wellAbove_cardio  <- sum(df$gluc == 2 & df$cardio == 1) * 100 / gluc_wellAbove

chol_normal_cardio <- sum(df$cholesterol == 0 & df$cardio == 1) * 100 /chol_normal
```

```r
chol_above_cardio <- sum(df$cholesterol == 1 & df$cardio == 1) * 100 / chol_above

chol_wellAbove_cardio <- sum(df$cholesterol == 2 & df$cardio == 1) * 100 / chol_wellAbove

labels <- c("Normal", "Above normal", "Well above normal")
cholvalues <- c(chol_normal_cardio,chol_above_cardio , chol_wellAbove_cardio)

graph0 <- barplot(cholvalues, names.arg = labels, ylab = "Percentage(%)", col = "steelblue",
        main = "Percentage of heart attack cases by cholesterol group", ylim = c(0, 100))
text(graph0, cholvalues , labels = round(cholvalues), pos = 3)
```




```r
glucvalues <- c(gluc_normal_cardio,gluc_above_cardio , gluc_wellAbove_cardio)


graph <- barplot(glucvalues, names.arg = labels, ylab = "Percentage(%)", col = "steelblue",
        main = "Percentage of heart attack cases by glucose group", ylim = c(0, 80))
text(graph, glucvalues , labels = round(glucvalues), pos = 3)
```



Standardization

```
df <- subset(df, select = -c(gluc, cholesterol))

# selecting only the continuous variables
continuous_vars <- c("height", "weight", "ap_hi", "ap_lo", "age_years")

# scaling the continuous variables
df[, continuous_vars] <- scale(df[, continuous_vars])
```

Model training

```


# apply PCA to the scaled data
pca <- prcomp(df, center = TRUE, scale. = TRUE)
pca
```

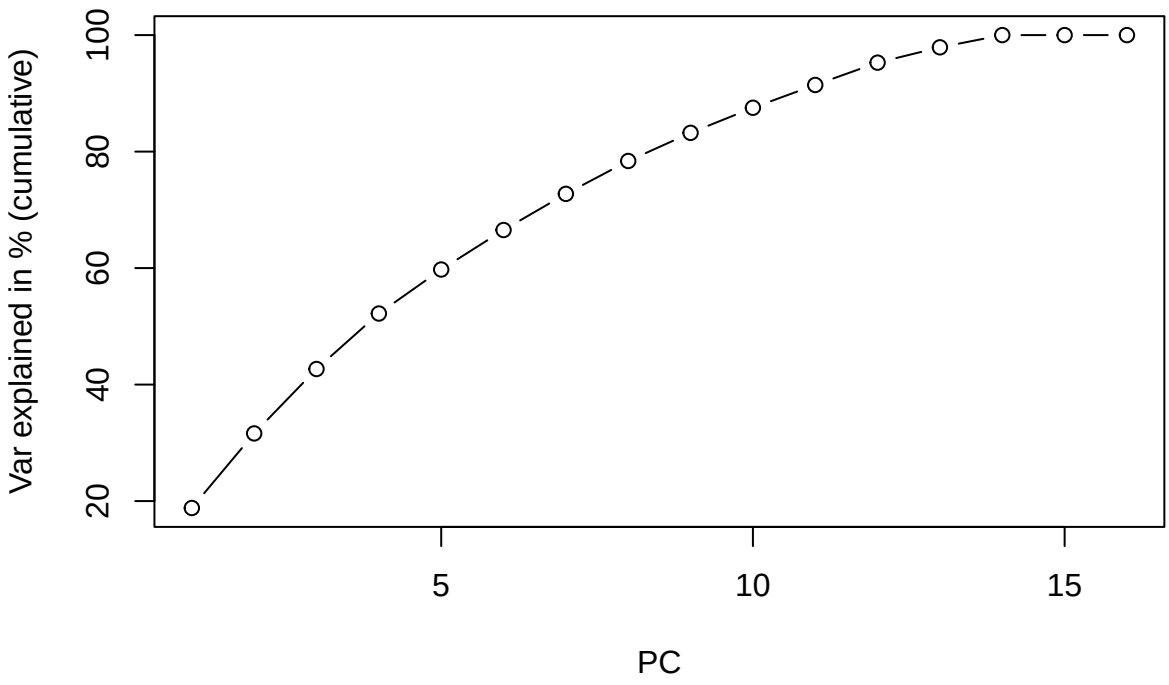
Feature selection

```
## Standard deviations (1, .., p=16):
##  [1] 1.734880e+00 1.431409e+00 1.330004e+00 1.234687e+00 1.099284e+00
```

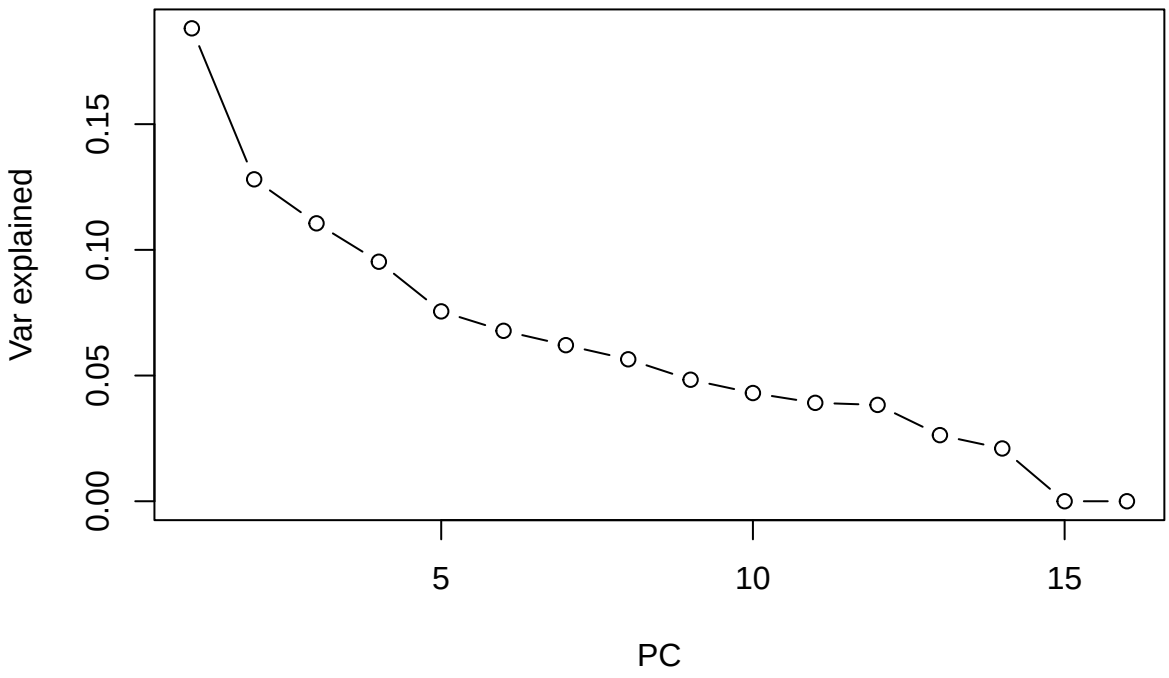
```
##  [6] 1.041502e+00 9.966812e-01 9.504518e-01 8.794459e-01 8.298359e-01
## [11] 7.913758e-01 7.830708e-01 6.486223e-01 5.794597e-01 2.611147e-14
## [16] 1.421422e-14
##
## Rotation (n x k) = (16 x 16):
##                       PC1          PC2         PC3         PC4         PC5
## gender       -0.019299906  0.473987426 -0.27232249  0.09631200 -0.02667212
## height       -0.008313125  0.452123153 -0.25767221  0.10327731 -0.16364497
## weight       -0.208072014  0.312117457  0.00937784  0.01294855 -0.20425707
## ap_hi        -0.301505336  0.261302543  0.40042680 -0.05575893 -0.06883631
## ap_lo        -0.277072503  0.270911771  0.38063866 -0.05158977 -0.08701425
## cholesterol0  0.436452568  0.138177402  0.12704409  0.16430721 -0.43129494
## cholesterol1 -0.223636512 -0.049132497 -0.12971191 -0.60205015  0.31347237
## cholesterol2 -0.353415461 -0.135187580 -0.03339775  0.42376810  0.24980390
## gluc0         0.404191924  0.208838657  0.31891947 -0.04924837  0.38955724
## gluc1        -0.232567429 -0.087285541 -0.25073273 -0.41432889 -0.48438142
## gluc2        -0.314917429 -0.195088370 -0.18227486  0.47428296 -0.04725087
## smoke        -0.023796115  0.347060006 -0.29551528  0.02488778  0.25748102
## alco         -0.041175448  0.252268260 -0.23415598 -0.01053848  0.30318274
## active        0.008364329  0.003475734 -0.02718571  0.01814434  0.15298576
## age_years    -0.179744868 -0.009106179  0.25159509  0.06959742  0.03719892
## cardio       -0.268163424  0.136277782  0.34066293 -0.01171695  0.01335319
##                       PC6           PC7         PC8         PC9           PC10
## gender        0.139010902 -0.0052840696 -0.23644580  0.38877450 -1.573604e-02
## height        0.388377253 -0.1011192616 -0.11300065  0.03464625 -9.052982e-05
## weight        0.297780121 -0.1223240546  0.11660103 -0.70754630  7.605484e-02
## ap_hi        -0.146045102 -0.0515070780  0.20190275  0.18879341  1.605974e-01
## ap_lo        -0.140348517 -0.0699273131  0.28314429  0.21624585  2.708140e-01
## cholesterol0 -0.258775525  0.0208537950 -0.01975713 -0.02174136  2.794332e-02
## cholesterol1  0.240668656 -0.0196630190 -0.02361846  0.08604059  1.864648e-01
## cholesterol2  0.093339504 -0.0072333158  0.05227837 -0.06292972 -2.385135e-01
## gluc0         0.179599823  0.0024779791  0.00129546 -0.06910258 -4.047637e-02
## gluc1        -0.244031543  0.0001519272 -0.09958471 -0.03050711 -1.993166e-01
## gluc2        -0.001385995 -0.0034843210  0.09632201  0.12303566  2.507463e-01
## smoke        -0.363322220  0.1551752017 -0.01566676  0.15960845 -1.060765e-01
## alco         -0.507200204  0.1943416274  0.14524787 -0.42706930  8.994933e-02
## active       -0.266297804 -0.9290320028 -0.18078624 -0.03397925 -6.082266e-02
## age_years    -0.118325657  0.1724598210 -0.83240942 -0.15105912  3.551837e-01
## cardio       -0.023680960  0.1079721615 -0.16426685 -0.01054029 -7.409187e-01
##                      PC11         PC12         PC13          PC14          PC15
## gender        0.03497942  0.207683182 -0.647738503  1.560963e-02 -1.683094e-14
## height        0.14407334  0.192572901  0.676397517 -3.589059e-02 -2.278390e-15
## weight       -0.14287711 -0.315036773 -0.271497664  1.735123e-02  7.252809e-16
## ap_hi        -0.04070135  0.040684047  0.024274962 -7.365758e-01  9.332224e-16
## ap_lo        -0.08965168  0.106210102  0.077965418  6.666795e-01 -1.347902e-16
## cholesterol0  0.13144013 -0.112869821 -0.014468275 -1.721350e-03  2.383557e-01
## cholesterol1  0.23682493 -0.155909924 -0.006649889  7.850097e-03  1.883504e-01
## cholesterol2 -0.43348844  0.321218374  0.026836891 -6.098393e-03  1.751732e-01
## gluc0        -0.12163427  0.067980437 -0.003936558 -2.285053e-03 -6.481548e-01
## gluc1        -0.24943400  0.198840115  0.027605162  3.525905e-05 -4.742868e-01
## gluc2         0.40931633 -0.287290328 -0.021886417  3.040363e-03 -4.816351e-01
## smoke        -0.37300025 -0.598357950  0.185225210  1.460231e-02  1.990153e-17
## alco          0.34988812  0.398342053 -0.005354042 -9.053549e-03  1.129383e-16
## active        0.05630838 -0.025081708  0.002847230  9.196033e-03  1.018479e-16
```

```
## age_years    -0.07583028  0.003437931  0.077158239  2.725148e-02 -8.277571e-17
## cardio        0.41426176 -0.152730743  0.003673905  9.961404e-02  1.723512e-16
##                      PC16
## gender       -6.077011e-15
## height       -1.399502e-15
## weight       -1.165793e-15
## ap_hi         2.843779e-16
## ap_lo        -8.466046e-16
## cholesterol0  6.365363e-01
## cholesterol1  5.029956e-01
## cholesterol2  4.678056e-01
## gluc0         2.427063e-01
## gluc1         1.776002e-01
## gluc2         1.803518e-01
## smoke        -1.395129e-16
## alco          8.321385e-17
## active       -1.273763e-16
## age_years     2.203695e-16
## cardio        1.590878e-16
```

```
plot(100*cumsum(pca$sdev^2)/sum(pca$sdev^2), type="b", xlab="PC", ylab="Var explained in % (cumulative)")
```




```
plot(pca$sdev^2/sum(pca$sdev^2), type="b", xlab="PC", ylab="Var explained")
```



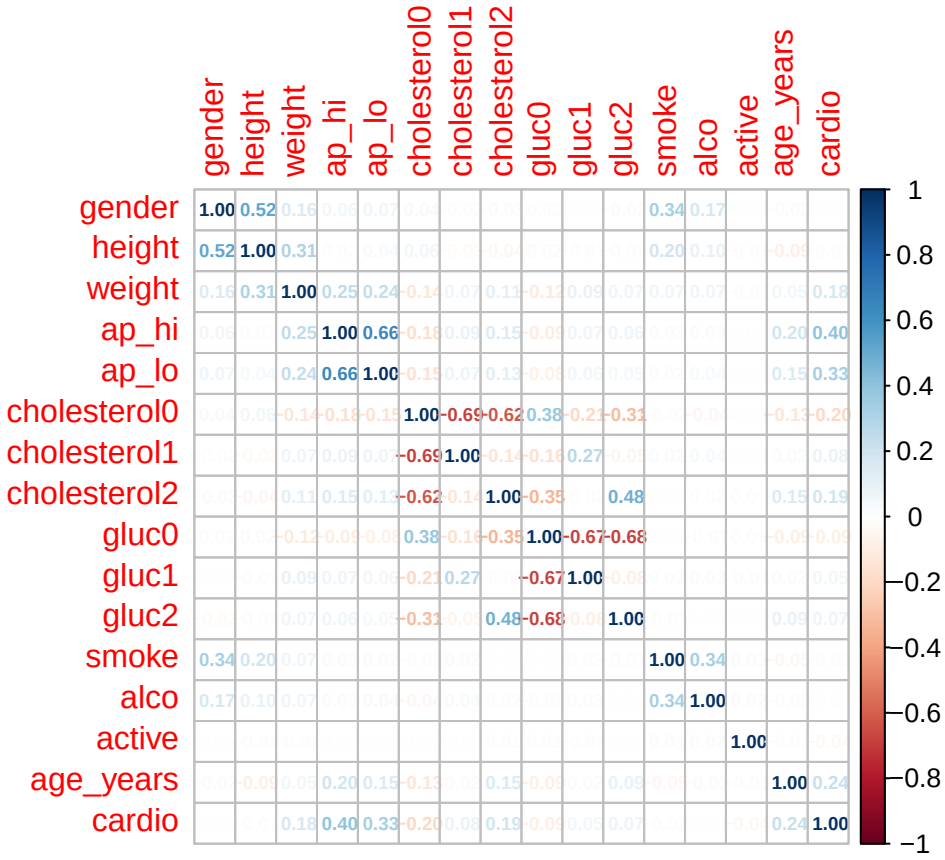

```
cat("\nAs we can see in the cumulative plot, to explain around 89% of the variance we need the first 9 P
```

```
##
## As we can see in the cumulative plot, to explain around 89% of the variance we need the first 9 PC. W
```

```
# We will plot a correlation matrix
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
# Calculate the correlation matrix for df
corr_matrix <- cor(df)

# Generate the correlation matrix plot using corrplot
corrplot(corr_matrix, method = "number", number.cex = 0.55)
```




```
# No sé si está bien... no sé si para el correlation plot hay que meter dummy variables o si tiene que
# Tampoco veo que haya mucha correlación entre las variables....
# No sé si hay que hacer la correlación solo del training set, pero lo he hecho de todo...
#As we can see no features are considered correlated, since there are not any value above 0.70 or below

#Next, we will consider only the statistically significant attributes
#Linear regression:
lm_heart <- lm(cardio ~ ., df)
summary(lm_heart)
```

```
##
## Call:
## lm(formula = cardio ~ ., data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.42560 -0.38139 -0.06069  0.40109  2.65720
##
## Coefficients: (2 not defined because of singularities)
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.6534870  0.0075450  86.612  < 2e-16 ***
## gender       -0.0009582  0.0043451  -0.221    0.825
## height       -0.0085488  0.0020783  -4.113 3.90e-05 ***
## weight        0.0342596  0.0018710  18.311  < 2e-16 ***
## ap_hi         0.1362601  0.0022887  59.535  < 2e-16 ***
## ap_lo         0.0477322  0.0022569  21.150  < 2e-16 ***
```

```
## cholesterol0 -0.2089635  0.0062606 -33.378  < 2e-16 ***
## cholesterol1 -0.1238404  0.0074709 -16.576  < 2e-16 ***
## cholesterol2         NA         NA      NA       NA
## gluc0         0.0583522  0.0073123   7.980 1.49e-15 ***
## gluc1         0.0607421  0.0095079   6.389 1.68e-10 ***
## gluc2                NA         NA      NA       NA
## smoke        -0.0259750  0.0066651  -3.897 9.74e-05 ***
## alco         -0.0402084  0.0080269  -5.009 5.48e-07 ***
## active       -0.0448806  0.0042515 -10.556  < 2e-16 ***
## age_years     0.0726252  0.0017476  41.556  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.442 on 68547 degrees of freedom
## Multiple R-squared:  0.2187, Adjusted R-squared:  0.2185
## F-statistic:  1476 on 13 and 68547 DF,  p-value: < 2.2e-16
```

```
#Attributes with p-value<0.5 are selected
paste("By looking at the summary, we can see that the only feature which is not statistically significa
```

```
## [1] "By looking at the summary, we can see that the only feature which is not statistically signific
```

```
df <- subset(df, select = -c(gender))

#Split data into Train/Test set
library(caret)
```

```
## Loading required package: lattice
```

```
set.seed(333) # set random seed for reproducibility
trainIndex <- createDataPartition(df$cardio, p = 0.8, list = FALSE)
train <- df[trainIndex,]
test <- df[-trainIndex,]

#It is convenient to check if the data set is balanced. In case it is extremely unbalanced, it can lead
table(train$cardio)
```

```
##
##     0     1
## 27813 27036
```

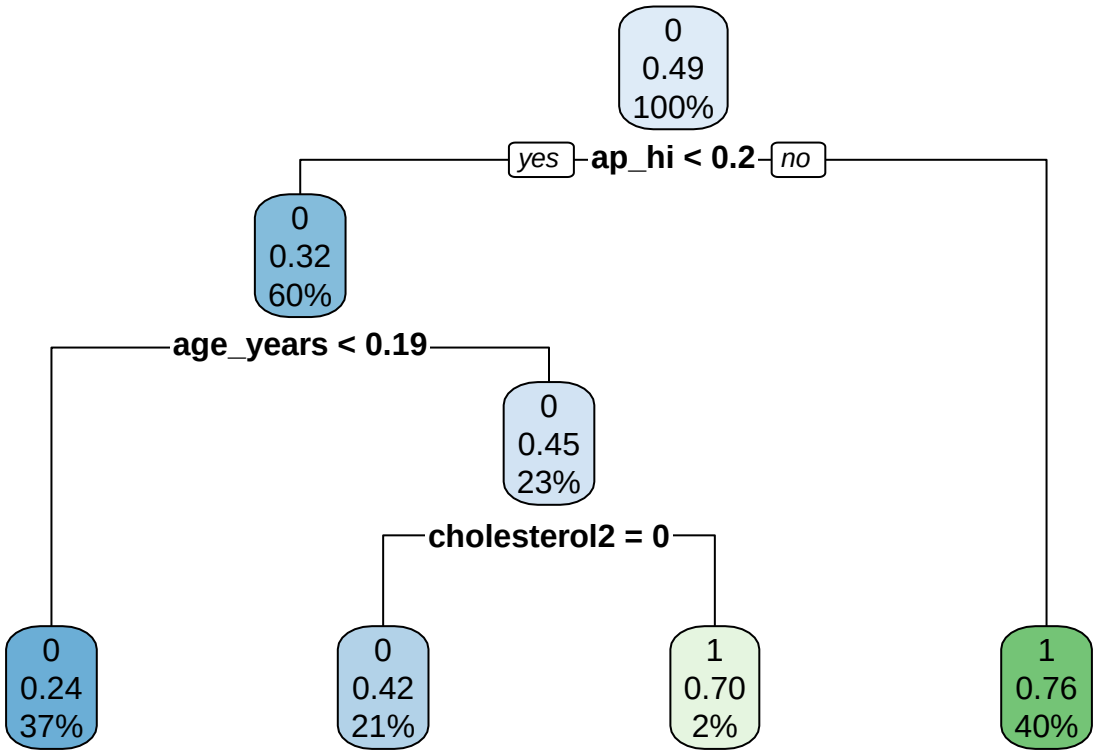
```
#In this case, we can see it is pretty balanced.
```

```
library(rpart)
library(rpart.plot)
library(randomForest)
```

Classification

```
## randomForest 4.7-1.1

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

##
## Attaching package: 'randomForest'

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

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
#Since this is a classification problem, the first approach we are taking is a decision tree.
# cp_values <- c(0.01, 0.02, 0.03, 0.04, 0.05)
#
# ctrl <- rpart.control(cp = cp_values, xval = 10, trace = 1)

# model <- rpart(cardio ~ ., data=train, control = ctrl)
#
# # Crea una lista con los valores de cp y la tasa de error correspondiente en cada iteración del model
# cp_values <- data.frame(cp = model$cptable[, "CP"], xerror = model$cptable[, "xerror"])
#
# # Grafica los valores de cp en un plot
# plot(cp_values$cp, cp_values$xerror, type = "b", xlab = "CP", ylab = "Cross-validation error")
#
# # Encuentra el valor óptimo de cp y entrena un modelo de árbol de decisión utilizando ese valor
# optimal_cp <- model$cptable[which.min(model$cptable[, "xerror"]),"CP"]
tree <- rpart(cardio ~ ., data = train, method = "class")

rpart.plot(tree)
```




```
# Evaluation
heart_pred <- predict(tree, test, type="class")

#heart_pred <- ifelse(heart_pred > 0.5 , 1, 0)
ConfMatrix <- confusionMatrix(as.factor(heart_pred), as.factor(test$cardio))
ConfMatrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 5519 2355
##          1 1420 4418
##
##                Accuracy : 0.7247
##                  95% CI : (0.7171, 0.7322)
##     No Information Rate : 0.5061
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4484
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.7954
##             Specificity : 0.6523
##          Pos Pred Value : 0.7009
```

```
##          Neg Pred Value : 0.7568
##              Prevalence : 0.5061
##          Detection Rate : 0.4025
##    Detection Prevalence : 0.5742
##       Balanced Accuracy : 0.7238
##
##        'Positive' Class : 0
##
```

```
#Random Forest
train$cardio <- as.factor(train$cardio)
rf <- randomForest(cardio ~ ., data=train)
pred_rf <- predict(rf, test, type="class")

ConfMatrix_Rf <- confusionMatrix(pred_rf, as.factor(test$cardio))
ConfMatrix_Rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 5501 2178
##          1 1438 4595
##
##                Accuracy : 0.7363
##                  95% CI : (0.7288, 0.7437)
##     No Information Rate : 0.5061
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4718
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.7928
##             Specificity : 0.6784
##          Pos Pred Value : 0.7164
##          Neg Pred Value : 0.7616
##              Prevalence : 0.5061
##          Detection Rate : 0.4012
##    Detection Prevalence : 0.5600
##       Balanced Accuracy : 0.7356
##
##        'Positive' Class : 0
##
```

```
train_glm <- subset(train, select = -c(gluc2, cholesterol2))
library(dplyr)
#Now, we are going to perform a Logistic regression to compare the results
glm_model <- glm(cardio ~ ., data=train, family = binomial)
glm_pred <- predict(glm_model, newdata=test, type="response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```r
predicted_classes <- ifelse(glm_pred > 0.5, 1, 0)

accuracy <- mean(predicted_classes == test$cardio)
precision <- sum(predicted_classes == 1 & test$cardio == 1) / sum(predicted_classes == 1)
recall <- sum(predicted_classes == 1 & test$cardio == 1) / sum(test$cardio == 1)
f1_score <- 2 * precision * recall / (precision + recall)

# Print the evaluation metrics
cat("\nAccuracy:", accuracy, "\n")
```

```
##
## Accuracy: 0.728486
```

```r
cat("\nPrecision:", precision, "\n")
```

```
##
## Precision: 0.7524834
```

```r
cat("\nRecall:", recall, "\n")
```

```
##
## Recall: 0.6710468
```

```r
cat("\nF1 Score:", f1_score, "\n")
```

```
##
## F1 Score: 0.7094357
```

```r
#NO SE MUY BIEN QUE HACER CON LAS DUMMY VARIABLES. AL HACER "SUMMARY(LM_HEART)" CHOLESTEROL2 Y GLUC2 NO
#The results from the decision trees, random forest and logistic regression are really similar.
```

```r
library(PRROC)
```

```
## Warning: package 'PRROC' was built under R version 4.2.3
```

```r
PRROC_obj <- roc.curve(scores.class0 = pred_rf, weights.class0= test$cardio,curve=TRUE)

plot(PRROC_obj$curve, main="ROC Curve for Three Models", type="l", col="black", xlab="False Positive Ra

PRROC_obj2 <- roc.curve(scores.class0 = predicted_classes, weights.class0= test$cardio,curve=TRUE)

PRROC_obj3 <- roc.curve(scores.class0 = heart_pred, weights.class0= test$cardio,curve=TRUE)

lines(PRROC_obj$curve, col="blue")

# Add the second ROC curve to the plot
lines(PRROC_obj2$curve, col="green")

# Add the third ROC curve to the plot
```

```r
lines(PRROC_obj3$curve, col="red")

abline(a=0, b=1, lty=2, col="gray")


# Add a legend to the plot
legend("bottomright", legend=c("Random Forest Model", "Predicted Classes", "Heart Prediction"), col=c("
```