CVD7

Francois PLACE

03/11/2020

Introduction

This study is an original work based on the "Cardiovascular Disease dataset", available on Kaggle, with this link : https://www.kaggle.com/sulianova/cardiovascular-disease-dataset. The owner of the dataset is Svetlana Ulianova.

The goal of our project is to predict presence or absence of a cardiovascular disease. The dataset provides several features in the medical field for every patient (factual information, results of examination and information given by the patient), and the target variable (cardio). The value of cardio is zero or one. One represents a patient with a cardiovascular problem.

The description of the features are here :

- Age | Objective Feature | age | int (days)
- Height | Objective Feature | height | int (cm) |
- Weight | Objective Feature | weight | float (kg) |
- Gender | Objective Feature | gender | categorical code | 1=woman 2=man
- Systolic blood pressure | Examination Feature | ap_hi | int |
- Diastolic blood pressure | Examination Feature | ap_lo | int |
- Cholesterol | Examination Feature | cholesterol | 1: normal, 2: above normal, 3: well above normal
- Glucose | Examination Feature | gluc | 1: normal, 2: above normal, 3: well above normal
- Smoking | Subjective Feature | smoke | binary | 1=yes
- Alcohol intake | Subjective Feature | alco | binary | 1=yes
- Physical activity | Subjective Feature | active | binary | 1=yes
- TARGET : Presence or absence of cardiovascular disease | cardio | binary | 1=yes

There are six main parts in the following :

- data cleaning and preparation
- dataset : general information and figures
- dataset : general information - graphs
- analysis section 1 : potential predictors
- analysis section 2 : performance of several models
- result section

In a first step, the original dataset is splitted in two datasets : train (90% of the datas) and validation (10% of the datas). Analysis sections 1 and 2 are performed only with train. The "validation" dataset is used only with the final model, to evaluate its performance.

R code is below.

```
# ------------------------------LIBRARIES
library(readr)
library(dplyr)
library(ggplot2)
library(gridExtra) #used for function grid.arrange
library(ggthemes) #used for theme_hc
library(caret)
library(rpart) #used for classification tree
library(randomForest)

options(digits=4, scipen=100) #force decimal notation
set.seed(10, sample.kind = "Rounding") #to fix the results of randomized processes
```

The original dataset is provided on my GitHub repository (placefr). The code below download the file to the local working directory, and load it in R. dat2 is a copy of dat (the original dataset)

```
# ------------------------------LOAD DATAS (from placefr GitHub repository)
#url is : https://github.com/placefr/CV-Disease
filename= "cardio_train.csv"
path1 = "https://raw.githubusercontent.com/placefr/CV-Disease/master"
path2 <- getwd() #local path (path2) is set to working directory

p_f_remote <- file.path(path1, filename) #remote path and filename
p_f_local <- file.path(path2, filename) #local path and filename

download.file(p_f_remote, filename)
dat <- read_csv2(p_f_local) #read the file, separator is semicolon
dat2 <- dat #dat2 is used in the following
```

Data cleaning and preparation

Firstly we want to know if some values are missing in the dataset (NA or empty cells). We test the whole dataset for this.

Then we discover a problem with blood pressure values (columns ap_hi and ap_lo). Values are present, but after ckecking, some of them are out of range. For this reason we add to dat2 a new column with the result of a test, to ignore bad values :

testbp <- ifelse(dat2$ap_hi < 40 | dat2$ap_hi>300 | dat2$ap_lo < 5 | dat2$ap_lo>150, 2, 3)

The new dataset (dat3) is splitted in two parts, train and validation. We will use validation only in the end of the script, to get the final performance of our model.

train2 is our final training dataset (no BP out of range values).

```
# ------------------------------DATA CLEANING AND PREPARATION
#seeking for 'NA' in dat2 dataset
test1 <- ifelse(dat2 == "NA", 1, 0)
#seeking for blank cells in dat2 dataset
test2 <- ifelse(dat2 == "" | dat2 == " ", 1, 0)
rtest1 <- sum(test1)
rtest2 <- sum(test2) #zero in both cases indicates no "NA" and no blank cells
```

```r
#print results
rtest1
```

```
## [1] 0
```

```r
rtest2
```

```
## [1] 0
```

```r
#managing blood pressure out of range values
#ap_hi and ap_lo : a new column testbp is created and added to dat2
#with '2' if ap_hi or ap_lo are out of range, and '3' if values seems to be normal
testbp <- ifelse(dat2$ap_hi<40 | dat2$ap_hi>300 | dat2$ap_lo<5 | dat2$ap_lo>150, 2, 3)
dat3 <- data.frame(dat2,testbp)

#creating training set and test set
#the size of the test set is 10% of the global dataset
#we will use test set only during the final stage, to evaluate the performance
#of the selected model
test_index <- createDataPartition(dat3$id, times=1, p=0.1, list=FALSE)
train <- dat3[-test_index,]
validation <- dat3[test_index,]

#training set preparation
out_of_range <- train %>% filter(testbp==2) %>% nrow()
normal <- train %>% filter(testbp==3) %>% nrow()
```

```r
#print results
out_of_range
```

```
## [1] 1100
```

```r
normal
```

```
## [1] 61900
```

```r
train2 <- train %>% filter(testbp==3) #train2 is a subset of train,
#with less rows, and only "normal" values for blood pressure (ap_hi and ap_low)
```

Dataset : General information and figures

Results are printed in the end of the section. We study : number of women and men in the dataset, average age, height and weight, to get a large overwiew of the datas.

```r
# ------------------------------DATASET : GENERAL INFORMATION & FIGURES
#results are printed in the end of the section
```

```r
#number of rows
row_nb <- nrow(dat3)

#number of women, number of men, and %, in the dataset dat3
wom_nb <- dat3 %>% filter(gender == 1) %>% nrow()
wom_percent <- wom_nb / row_nb * 100
men_nb <- dat3 %>% filter(gender == 2) %>% nrow()
men_percent <- men_nb / row_nb * 100

#women and men : average age & standard deviation
age_in_years <- dat3$age / 365 #age in years
dat4 <- data.frame(dat3,age_in_years)

avg_age <- dat4 %>% group_by(gender) %>% summarize(avg = mean(age_in_years))
#sdev : age, standard deviation
sdev_age <- dat4 %>% group_by(gender) %>% summarize(sdev = sd(age_in_years))

#women and men : average height & standard deviation
avg_height <- dat4 %>% group_by(gender) %>% summarize(avg = mean(height)/100)
#sdev : height, standard deviation
sdev_height <- dat4 %>% group_by(gender) %>% summarize(sdev = sd(height)/100)

#women and men : average weight & standard deviation
avg_weight <- dat4 %>% group_by(gender) %>% summarize(avg = mean(weight)/10)
#sedv : weight, standard deviation
sdev_weight <- dat4 %>% group_by(gender) %>% summarize(sdev = sd(weight)/10)
```

Results part 1

```r
#print the first results
labels1 <- c("number of rows", "number of women", "% of women in the dataset",
             "number of men", "% of men in the dataset")
results1 <- c(row_nb, wom_nb, wom_percent, men_nb, men_percent)
results_1 <- data.frame(labels1, results1)
results_1
```

```
##                     labels1 results1
## 1            number of rows 70000.00
## 2           number of women 45530.00
## 3 % of women in the dataset    65.04
## 4             number of men 24470.00
## 5   % of men in the dataset    34.96
```

Results part 2

```r
#print results (following)
labels2 <- c("average age of women", "sd age of women", "average age of men",
             "sd age of men")
results2 <- c(avg_age$avg[1], sdev_age$sdev[1], avg_age$avg[2], sdev_age$sdev[2])
results_2 <- data.frame(labels2, results2)
results_2
```

```
##                labels2 results2
## 1 average age of women   53.452
## 2      sd age of women    6.663
## 3   average age of men   53.129
## 4        sd age of men    6.931
```

Results part 3

```
#print results (following)
labels3 <- c("average height of women", "sd height of women", "average height of men",
             "sd height of men")
results3 <- c(avg_height$avg[1], sdev_height$sdev[1], avg_height$avg[2],
              sdev_height$sdev[2])
results_3 <- data.frame(labels3, results3)
results_3
```

```
##                   labels3 results3
## 1 average height of women  1.61356
## 2      sd height of women  0.07053
## 3   average height of men  1.69948
## 4        sd height of men  0.07229
```

Results part 4

```
#print results (following)
labels4 <- c("average weight of women", "sd weight of women", "average weight of men",
             "sd weight of men")
results4 <- c(avg_weight$avg[1], sdev_weight$sdev[1], avg_weight$avg[2],
              sdev_weight$sdev[2])
results_4 <- data.frame(labels4, results4)
results_4
```

```
##                   labels4 results4
## 1 average weight of women    72.64
## 2      sd weight of women    15.58
## 3   average weight of men    77.26
## 4        sd weight of men    14.19
```

The results give us interesting information : there are more women than men in the dataset (65% versus 35%), average age of women and men is the same with a similar standard deviation, and the weight of mens is higher, on average.

We keep in mind the prevalence of women in the dataset.

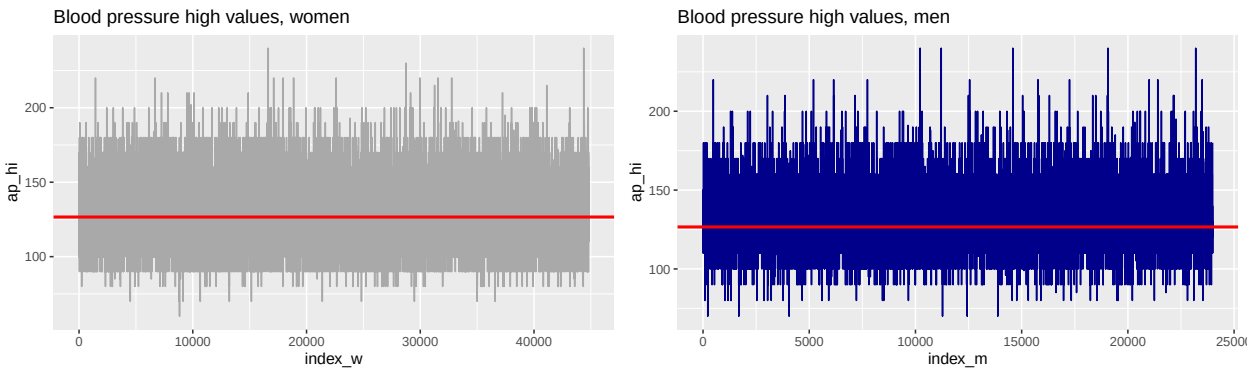
Dataset : General information - Graphs

Blood pressure

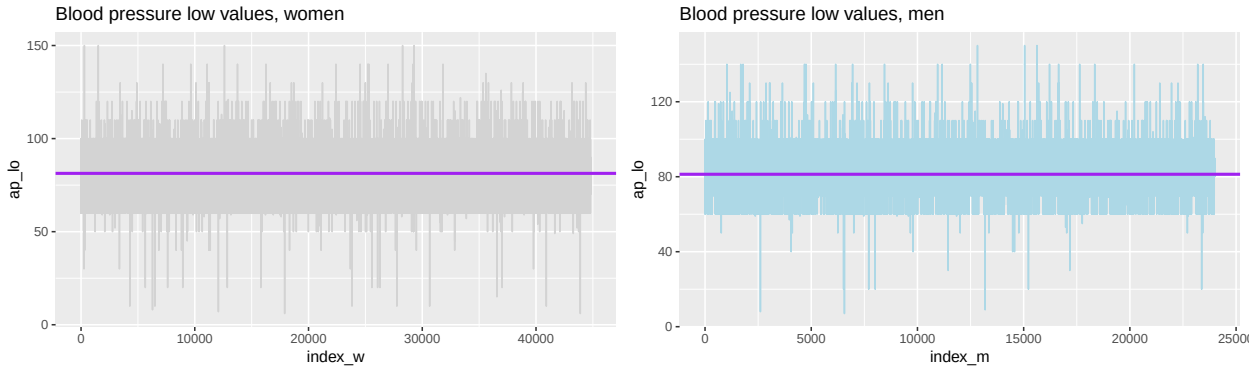
We want to study the distribution of blood pressure values around the average, for women and men, and for high and low values. We plot high values first, then low values.

```r
# ------------------------------DATASET : GENERAL INFORMATION / GRAPHS
dat5 <- dat4 %>% filter(testbp==3) #dat5 is clean ('normal' values for blood pressure)
avg1 <- mean(dat5$ap_hi) #avg1 is the average of ap_hi (men and women)
avg2 <- mean(dat5$ap_lo) #avg2 is the average of ap_lo (men and women)

#blood pressure high values, women
dat_wom <- dat5 %>% filter(gender==1)
nbw <- nrow(dat_wom) #number of women in dat5 dataset

index_w <- seq(1,nbw,1)
g1 <- ggplot(dat_wom, aes(x=index_w, y=ap_hi)) +
  ggtitle("Blood pressure high values, women") +
  geom_line(color="darkgrey") +
  geom_hline(yintercept=avg1, color="red", size=1) #average line

#blood pressure high values, men
dat_men <- dat5 %>% filter(gender==2)
nbm <- nrow(dat_men) #number of men in dat5 dataset

index_m <- seq(1,nbm,1)
g2 <- ggplot(dat_men, aes(x=index_m, y=ap_hi)) +
  ggtitle("Blood pressure high values, men") +
  geom_line(color="darkblue") +
  geom_hline(yintercept=avg1, color="red", size=1) #average line

#blood pressure low values, women
g3 <- ggplot(dat_wom, aes(x=index_w, y=ap_lo)) +
  ggtitle("Blood pressure low values, women") +
  geom_line(color="lightgrey") +
  geom_hline(yintercept=avg2, color="purple", size=1)

#blood pressure low values, men
g4 <- ggplot(dat_men, aes(x=index_m, y=ap_lo)) +
  ggtitle("Blood pressure low values, men") +
  geom_line(color="lightblue") +
  geom_hline(yintercept=avg2, color="purple", size=1)
```

```r
#print the four graphs
grid.arrange(g1, g2, ncol = 2)
```



```
grid.arrange(g3, g4, ncol = 2)
```



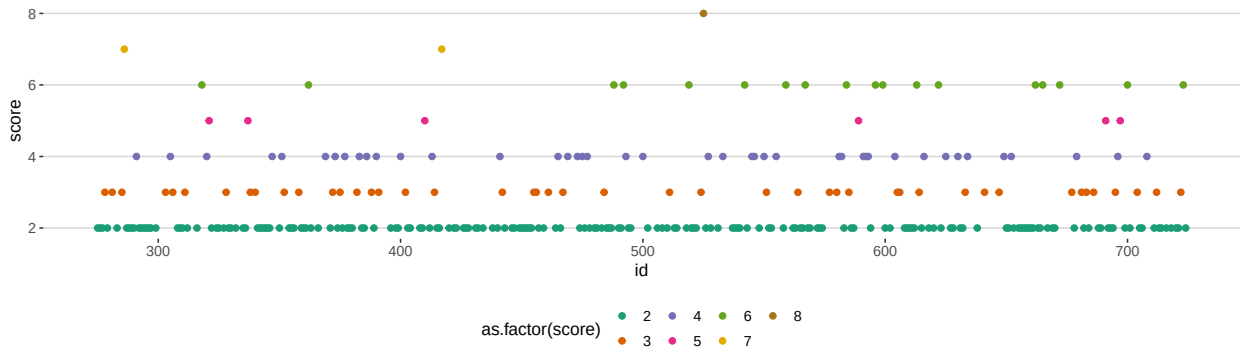
Concerning high values, we note that the "tunnel" of values is larger for women. The distribution around the average is also different : It seems that values are more above average for men.

Score of risk

We define a global score for predictors : cholesterol + gluc + smoke + alcohol. The "score" is the sum of these individual features. We print only a subset in the following graph.

```
#the sum represents a score of potential risk
#we print only for a subset of id [200:500,] in the dataset
score <- dat5$cholesterol + dat5$gluc + dat5$smoke + dat5$alco
dat6 <- data.frame(dat5,score)
dat6 <- dat6[200:500,]
g5 <- ggplot(dat6, aes(x=id, y=score, color=as.factor(score))) +
  scale_color_brewer(palette="Dark2") +
  theme_hc() +
  geom_point(size=1.8)
```

```
g5
```



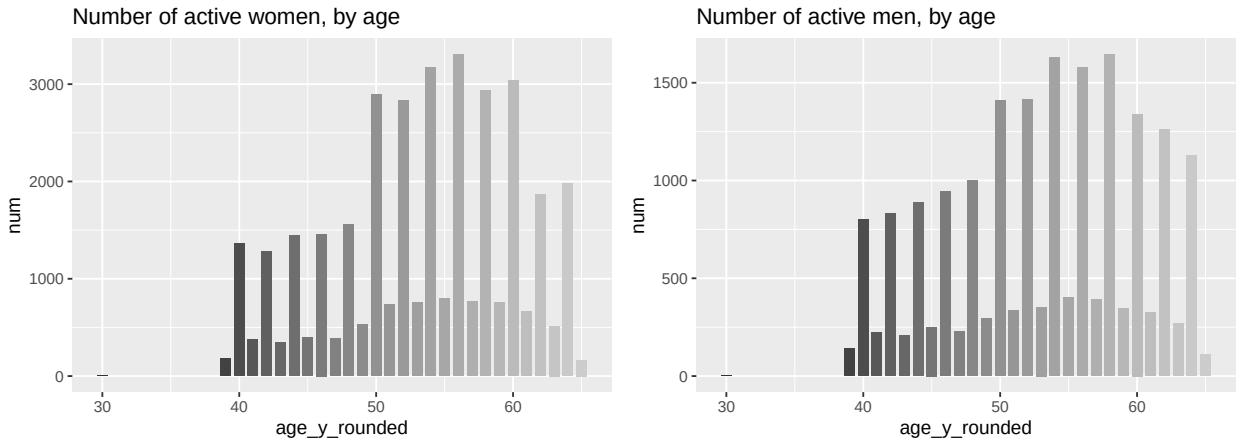
With this graph, we get a view of score distribution. Most of the patients have the lower score two.

Active and Inactive people

Age is rounded, and we plot the proportion of active women and men, by age.

```
#active & inactive people
#we plot the proportion of active women and men, by age
age_y_rounded <- round(age_in_years, digits = 0) #age in years, rounded
dat7 <- data.frame(dat4,age_y_rounded)

active_wom <- dat7 %>% filter(gender == 1 & active == 1) %>%
  group_by(age_y_rounded) %>% summarize(num = n()) %>% mutate()

g6 <- active_wom %>% ggplot(aes(x=age_y_rounded, y=num, fill=factor(age_y_rounded))) +
  ggtitle("Number of active women, by age") +
  geom_col(width=0.8) +
  scale_fill_grey() +
  theme(legend.position='none')

active_men <- dat7 %>% filter(gender == 2 & active == 1) %>%
  group_by(age_y_rounded) %>% summarize(num = n()) %>% mutate()

g7 <- active_men %>% ggplot(aes(x=age_y_rounded, y=num, fill=factor(age_y_rounded))) +
  ggtitle("Number of active men, by age") +
  geom_col(width=0.8) +
  scale_fill_grey() +
  theme(legend.position='none')
```

```
grid.arrange(g6, g7, ncol=2)
```



We notice here that something is not normal concerning the feature "age", in years and rounded. Il can't be seen in the original dataset (age is in days), but a biais concerning age distribution appears. Maybe the problem can be found in the first steps of data collection, when the dataset was built. For our models in the following, we consider age in days, and our hypothesis is that this problem is minor.

Analysis Section 1 : Potential predictors

For age, weight and blood pressure, we plot mean(cardio) against the predictor, to see if we can assume a linear relation, and we use lm() to train a model. Then we compute the individual accuracy (with confusionMatrix).

For sex and score of risk, we calculate the proportion of heart deseases for the different classes, to estimate a possible relationship.

For physical activity, we consider the feature alone, and it is integrated in all evaluations, in the next part of this work.

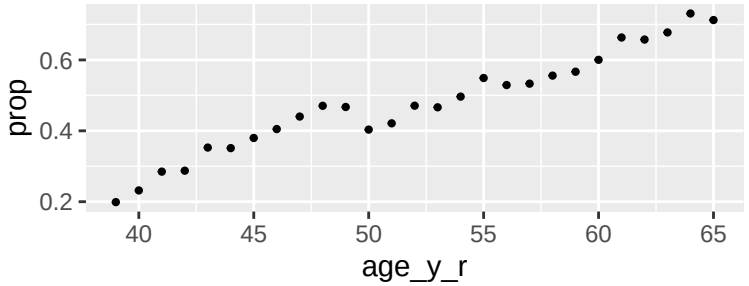
Concerning our predictions, we consider cardio=1 (y_hat=1) if p_hat > 0.5, and cardio=0 if not.

More details are in the code.

```
# ------------------------------ANALYSIS SECTION : POTENTIAL PREDICTORS
#based on train2 (the clean train set created above), we create a new training set
#and test set. The new train set represents 50% of 'train2'
test_index <- createDataPartition(train2$id, times=1, p=0.5, list=FALSE)
trainset <- train2[-test_index,]
testset <- train2[test_index,]

trainset2 <- trainset #trainset2 is used in the following
```

Age vs cardio

```
#AGE / CARDIO
#graph : age vs cardio
#we seek if the trend seems to be linear
age_y_r <- round(trainset2$age/365)
trainset3 <- data.frame(trainset2,age_y_r)
trainset3 %>%
  group_by(age_y_r) %>% filter(n() >= 20) %>%  #we take groups of datas
  summarize(prop = mean(cardio == 1)) %>%      #with more than 20 records
  ggplot(aes(age_y_r, prop)) +
  geom_point(size=0.8)
```




```
#looking for a link between age and cardio with lm()
fit <- lm(cardio ~ age, trainset2)
p_hat <- predict(fit, testset)
y_hat <- ifelse(p_hat > 0.5, 1, 0) %>% factor()
```

```
#accuracy
confusionMatrix(y_hat, factor(testset$cardio))$overall[["Accuracy"]]
```

```
## [1] 0.5986
```

Sex vs cardio

```
#SEX / CARDIO
wom_prop <- trainset2 %>% filter(gender == 1) %>% nrow()
wom_cardio <- trainset2 %>% filter(gender == 1 & cardio ==1) %>% nrow()
wom_p <- wom_cardio / wom_prop

men_prop <- trainset2 %>% filter(gender == 2) %>% nrow()
men_cardio <- trainset2 %>% filter(gender == 2 & cardio ==1) %>% nrow()
men_p <- men_cardio / men_prop

options(digits=4)
#proportion of women with cardio=1 in trainset2
wom_p
```

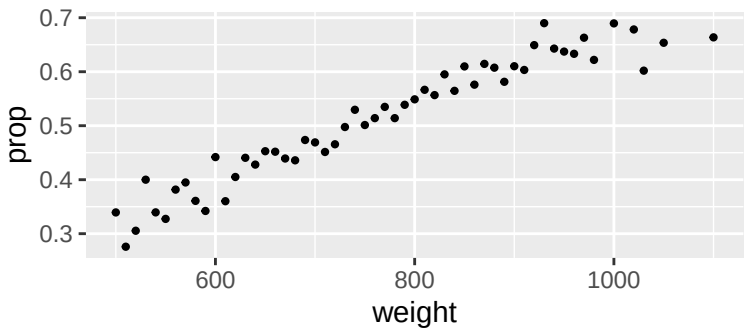
```
## [1] 0.4922
```

```
#proportion of men with cardio=1 in trainset2
men_p
```

```
## [1] 0.5034
```

Weight vs cardio

```
#WEIGHT / CARDIO
#graph : weight vs cardio
#we seek if the trend seems to be linear
trainset2 %>%
  group_by(weight) %>% filter(n() >= 100) %>%
  summarize(prop = mean(cardio == 1)) %>%
  ggplot(aes(weight, prop)) +
  geom_point(size=0.8)
```



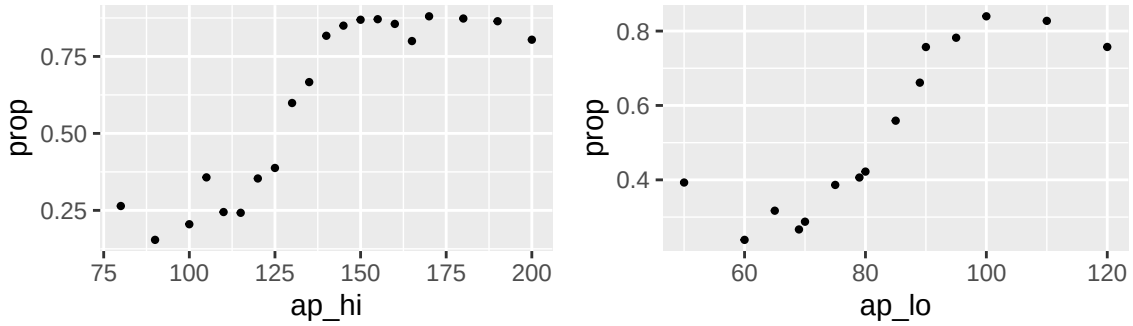
```r
#looking for a link between weight and cardio with lm()
fit <- lm(cardio ~ weight, trainset2)
p_hat <- predict(fit, testset)
y_hat <- ifelse(p_hat > 0.5, 1, 0) %>% factor()
#accuracy
confusionMatrix(y_hat, factor(testset$cardio))$overall[["Accuracy"]]
```

```
## [1] 0.5764
```

Blood Pressure vs cardio

```r
#BLOOD PRESSURE / CARDIO
#graph : blood pressure vs cardio
#we seek if the trend seems to be linear
#graph : ap_hi
g10 <- trainset2 %>%
  group_by(ap_hi) %>% filter(n() >= 20) %>%
  summarize(prop = mean(cardio == 1)) %>%
  ggplot(aes(ap_hi, prop)) +
  geom_point(size=0.8)
#graph : ap_lo
g11 <- trainset2 %>%
  group_by(ap_lo) %>% filter(n() >= 20) %>%
  summarize(prop = mean(cardio == 1)) %>%
  ggplot(aes(ap_lo, prop)) +
  geom_point(size=0.8)
```

```r
grid.arrange(g10, g11, ncol = 2)
```




```r
#looking for a link between ap_hi and cardio with lm()
fit <- lm(cardio ~ ap_hi, trainset2)
p_hat <- predict(fit, testset)
y_hat <- ifelse(p_hat > 0.5, 1, 0) %>% factor()
#accuracy
confusionMatrix(y_hat, factor(testset$cardio))$overall[["Accuracy"]]
```

```
## [1] 0.7097
```

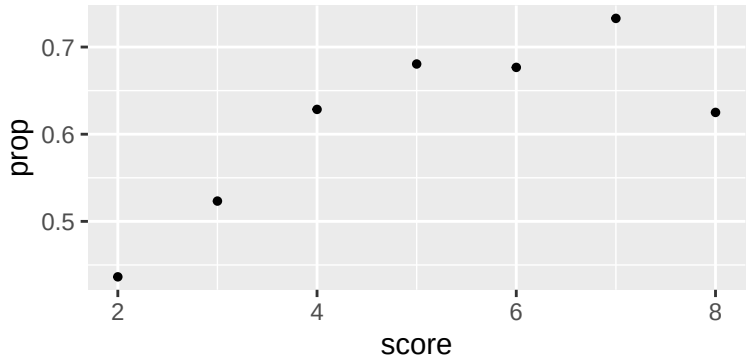
```r
#looking for a link between ap_lo and cardio with lm()
fit <- lm(cardio ~ ap_lo, trainset2)
p_hat <- predict(fit, testset)
y_hat <- ifelse(p_hat > 0.5, 1, 0) %>% factor()
#accuracy
confusionMatrix(y_hat, factor(testset$cardio))$overall[["Accuracy"]]
```

```
## [1] 0.6569
```

Score of risk

```r
#SCORE OF RISK (as defined above)
score <- trainset2$cholesterol + trainset2$gluc + trainset2$smoke + trainset2$alco
trainset4 <- data.frame(trainset2, score)
```

```r
#graph : score vs cardio
g12 <- trainset4 %>%
  group_by(score) %>%
  summarize(prop = mean(cardio)) %>% #gives the proportion of people
  #with a cardiovascular disease for each score of risk
  ggplot(aes(score, prop)) +
  geom_point(size=1)

g12
```



Analysis section 2 : Performance of several models

Based on trainset and testset defined in the beginning of the analysis section, we try several models, and report accuracy in the end of the current section.

Model 1 : Linear Regression

```r
# ----------MODEL 1 : LINEAR REGRESSION
#Predictors are :
#age, gender, weight, ap_hi and ap_lo, score of risk and physical activity
```

```r
#our hypothesis is also that height is not linked to heart diseases in general

fit <- lm(cardio ~ age + gender + weight + ap_hi + ap_lo + active + score, trainset4)

#
score <- testset$cholesterol + testset$gluc + testset$smoke + testset$alco
testset_cpl <- data.frame(testset, score) #add score to clean version of testset

p_hat_model1 <- predict(fit, testset_cpl)
y_hat <- ifelse(p_hat_model1 > 0.5, 1, 0) %>% factor()
accur_lr <-confusionMatrix(y_hat, factor(testset$cardio))$overall[["Accuracy"]]
```

Model 2 : Regression Tree

There are three parts for model 2 : 1st version with standard parameters, optimization of complexity parameter and minsplit, and final version.

We ignore warnings, because we get a continuous outcome and we transform it to a binary outcome.

```r
# ----------MODEL 2 : REGRESSION TREE
#a lightened version of trainset4 & testset_cpl are used
trainset_ctree <- trainset4 %>% select(id, age, gender, weight, ap_hi, ap_lo,
                                       active, cardio, score)
testset_ctree <- testset_cpl %>% select(id, age, gender, weight, ap_hi, ap_lo,
                                        active, cardio, score)

#2.1/ works with standard parameters
fit <- rpart(cardio ~ .,
             data = trainset_ctree)
p_hat <- predict(fit, testset_ctree)
y_hat1 <- ifelse(p_hat > 0.5, 1, 0) %>% factor()
accur_rt1 <- confusionMatrix(y_hat1, factor(testset$cardio))$overall[["Accuracy"]]


#2.2/ we try several values for complexity parameter
calc1 <- function(x){
  fit <- rpart(cardio ~ .,
               data = trainset_ctree,
               control=rpart.control(cp=x, minsplit=20))
  p_hat <- predict(fit, testset_ctree)
  y_hat2 <- ifelse(p_hat > 0.5, 1, 0) %>% factor()
  confusionMatrix(y_hat2, factor(testset$cardio))$overall[["Accuracy"]]
}
x <- seq(0.0001, 0.0009, 0.0001) #range for cp
out1 <- sapply(x, calc1)

max1 <- which.max(out1)
#out1
max1
```

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

```r
#we keep 0.0003 as the best value for cp


#We try several values for minsplit
calc2 <- function(x){
  fit <- rpart(cardio ~ .,
               data = trainset_ctree,
               control=rpart.control(cp=0.0003, minsplit=x))
  p_hat <- predict(fit, testset_ctree)
  y_hat2 <- ifelse(p_hat > 0.5, 1, 0) %>% factor()
  confusionMatrix(y_hat2, factor(testset$cardio), positive="1")$overall[["Accuracy"]]
}
x <- seq(15,35,1) #range for minsplit
out2 <- sapply(x, calc2)

max2 <- which.max(out2)
#out2
max2
```

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

```r
#we keep 15 as the best value for minsplit (default -20- gives the same result)


#2.3/ final version (parameters : cp=0.0003, minsplit=15)
fit <- rpart(cardio ~ .,
             data = trainset_ctree,
             control=rpart.control(cp=0.0003, minsplit=15))
p_hat_model23 <- predict(fit, testset_ctree)
y_hat3 <- ifelse(p_hat_model23 > 0.5, 1, 0) %>% factor()
accur_rt2 <- confusionMatrix(y_hat3, factor(testset$cardio))$overall[["Accuracy"]]
```

Model 3 : Random Forest

There are three parts for model 3 : 1st version with standard parameters, optimization of mtry and nodesize, and final version.

We ignore warnings, because we get a continuous outcome and we transform it to a binary outcome.

```r
# ----------MODEL 3 : RANDOM FOREST
#only a subset is used to optimize calculation time
trainset_ctree_r <- trainset_ctree[1:5000,]
testset_ctree_r <- testset_ctree[1:5000,]


#3.1/ works with standard parameters
fit <- randomForest(cardio ~ ., data=trainset_ctree_r)
p_hat <- predict(fit, testset_ctree_r)
y_hat1 <- ifelse(p_hat > 0.5, 1, 0) %>% factor()
accur_rf1 <- confusionMatrix(y_hat1, factor(testset_ctree_r$cardio))$overall[["Accuracy"]]
#2 minutes to get the result
```

```r
#3.2/ we try several values for mtry (number of predictors randomly selected)
fit <- train(cardio ~ .,
             method = "rf",
             data = trainset_ctree_r,
             ntree=100,
             tuneGrid = data.frame(mtry = c(1,2,3,4)),
             nodesize = 20)

p_hat <- predict(fit, testset_ctree_r)
y_hat2 <- ifelse(p_hat > 0.5, 1, 0) %>% factor()
confusionMatrix(y_hat2, factor(testset_ctree_r$cardio))$overall[["Accuracy"]]
```

```
## [1] 0.7128
```

```r
fit$bestTune
```

```
##   mtry
## 2    2
```

```r
#we keep mtry=2


#we try to optimize the parameter nodesize
calc3 <- function(x){
  train(cardio ~ .,
        method = "rf",
        data = trainset_ctree_r,
        ntree=50,
        tuneGrid = data.frame(mtry=2),
        nodesize = x)
  y_hat2 <- ifelse(p_hat > 0.5, 1, 0) %>% factor()
  confusionMatrix(y_hat2, factor(testset_ctree_r$cardio),
                  positive="1")$overall[["Accuracy"]]
  }

x <- seq(1,101,20)
out3 <- sapply(x,calc3)

max3 <- which.max(out3)
#out3
max3
```

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

```r
#we keep nodesize=20 (no optimization found)


#3.3/ final version (ntree=100, mtry=2, nodesize=20)
fit <- train(cardio ~ .,
             method = "rf",
             data = trainset_ctree,
```

```
             ntree=100,
             tuneGrid = data.frame(mtry=2),
             nodesize = 20)

p_hat_model33 <- predict(fit, testset_ctree)
y_hat2 <- ifelse(p_hat_model33 > 0.5, 1, 0) %>% factor()
accur_rf2 <- confusionMatrix(y_hat2, factor(testset_ctree$cardio))$overall[["Accuracy"]]
```

Models 1,2 & 3 : Accuracy

```
#Linear regression
accur_lr
```

```
## [1] 0.7204
```

```
#regression tree std parameters
accur_rt1
```

```
## [1] 0.7098
```

```
#regression tree tuned parameters
accur_rt2
```

```
## [1] 0.7213
```

```
#random forest std parameters
accur_rf1
```

```
## [1] 0.7094
```

```
#random forest tuned parameters
accur_rf2
```

```
## [1] 0.7227
```

Model 4 : Ensemble

We use p_hat from model 1 (linear regression) and p_hat from model 3.3 (random forest final version).

```
# ----------MODEL 4 : ENSEMBLE
p_hat_avg <- (p_hat_model1 + p_hat_model33) /2
y_hat_avg <- ifelse(p_hat_avg > 0.5, 1, 0) %>% factor()
accur_ens <- confusionMatrix(y_hat_avg,
                             factor(testset_ctree$cardio))$overall[["Accuracy"]]

#ensemble : accuracy
accur_ens
```

```
## [1] 0.7244
```

Result section

Finally we select a combination of two models, Linear Regression and Random Forest, in an Ensemble. The sources are the datasets train and validation, from the original dataset dat3.

We filter the two datasets to get datasets with blood pressure normal values, and we add the "score" as defined above.

All predictors are used, except height. Our hypothesis is that height is not a relevant factor in heart deseases.

We ignore warnings, because we get a continuous outcome and we transform it to a binary outcome.

```
# ------------------------------RESULT SECTION
# sources : datasets train and validation (from original dataset)
# train and validation already hold a column 'testbp'
# to manage blood pressure out of range values

set.seed(10, sample.kind = "Rounding")

ztrain <- train %>% filter(testbp==3)
#train dataset with blood pressure normal values
zvalid <- validation %>% filter(testbp==3)
#validation dataset with blood pressure normal values

#as above, score is the sum of cholesterol + gluc + smoke + alco
score <- ztrain$cholesterol + ztrain$gluc + ztrain$smoke + ztrain$alco
ztrain <- data.frame(ztrain, score) #score is added to ztrain
#we keep only features used in our model
ztrain <- ztrain %>% select(id, age, gender, weight, ap_hi, ap_lo, active, cardio, score)

score <- zvalid$cholesterol + zvalid$gluc + zvalid$smoke + zvalid$alco
zvalid <- data.frame(zvalid, score) #score is added to zvalid
#we keep only features used in our model
zvalid <- zvalid %>% select(id, age, gender, weight, ap_hi, ap_lo, active, cardio, score)

#Step 1 : Linear Regression
#Predictors are :
#age, gender, weight, ap_hi and ap_lo, physical activity & score of risk (defined above)
#our hypothesis is also that height is not linked to heart diseases in general
fit1 <- lm(cardio ~ age + gender + weight + ap_hi + ap_lo + active + score, ztrain)
p_hat1 <- predict(fit1, zvalid)
y_hat1 <- ifelse(p_hat1 > 0.5, 1, 0) %>% factor()
prec1 <- confusionMatrix(y_hat1, factor(zvalid$cardio))$overall[["Accuracy"]]
prec1
```

```
## [1] 0.7185
```

```
#Step 2 : Random Forest
fit2 <- train(cardio ~ .,
              method = "rf",
              data = ztrain,
              ntree=100,
              tuneGrid = data.frame(mtry=2),
              nodesize = 20)
p_hat2 <- predict(fit2, zvalid)
```

```r
y_hat2 <- ifelse(p_hat2 > 0.5, 1, 0) %>% factor()
prec2 <- confusionMatrix(y_hat2, factor(zvalid$cardio))$overall[["Accuracy"]]
prec2
```

```
## [1] 0.7262
```

```r
#Step 3 : Ensemble
p_hat_avg <- (p_hat1 + p_hat2) /2
y_hat_avg <- ifelse(p_hat_avg > 0.5, 1, 0) %>% factor()

#Step 4 : Evaluation : Accuracy
#Overall accuracy
precision1 <- confusionMatrix(y_hat_avg, factor(zvalid$cardio))
precision1
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2730 1093
##          1  778 2288
##
##                Accuracy : 0.728
##                  95% CI : (0.718, 0.739)
##     No Information Rate : 0.509
##     P-Value [Acc > NIR] : < 0.0000000000000002
##
##                   Kappa : 0.456
##
##  Mcnemar's Test P-Value : 0.000000000000389
##
##             Sensitivity : 0.778
##             Specificity : 0.677
##          Pos Pred Value : 0.714
##          Neg Pred Value : 0.746
##              Prevalence : 0.509
##          Detection Rate : 0.396
##    Detection Prevalence : 0.555
##       Balanced Accuracy : 0.727
##
##        'Positive' Class : 0
##
```

```r
precision1$overall[["Accuracy"]]
```

```
## [1] 0.7284
```

```r
#RMSE
#calculation with cardio and predicted values before rounding (p_hat_avg)
rmserr <- RMSE(zvalid$cardio, p_hat_avg)
rmserr
```

```
## [1] 0.4338
```

The performance of our final model (accuracy) is ~ 73%. We are able to make a correct prediction for a large majority of patients.

We have to take in account that our set of predictors is perhaps not complete, to diagnose a health disease. Other unknown factors may be useful to improve the prediction.

Conclusion

After cleaning and a first exploration of datas, the process was to take features one by one, and try to estimate if they are relevant predictors for our model.

In a second time we try several models with a subset of the initial dataset, and we compute accuracy to evaluate the precision.

Then we select a combination (linear model + random forest), which gives the best result.

In terms of improvements, I think an additional work is maybe possible with the weight of predictions : when we 'convert' p_hat to y_hat, some values are clearly near 0 or near 1. Some values are near 5, and we could consider that they are less relevant.