Assignment 1

Álvaro y Pablo

10/6/2021

Preprocesamiento: Carga datos y librerias

Cargamos las librerias a usar:

```
library(tidyverse)
library(GGally)
library(MLTools)
library(caret)
library(ROCR)
```

Cargamos los datos:

```
datos <- read.table("./data/Diabetes.csv", sep = ";", header = TRUE)
```

Análisis Exploratorio

Resumen general del datasets

```
str(datos)
```

```
## 'data.frame':    768 obs. of  9 variables:
##  $ PREGNANT     : int  6 1 8 1 0 5 3 10 2 8 ...
##  $ GLUCOSE      : int  148 85 183 89 137 116 78 115 197 125 ...
##  $ BLOODPRESS   : int  72 66 64 66 40 74 50 0 70 96 ...
##  $ SKINTHICKNESS: int  35 29 0 23 35 0 32 0 45 0 ...
##  $ INSULIN      : int  0 0 0 94 168 0 88 0 543 0 ...
##  $ BODYMASSINDEX: num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
##  $ PEDIGREEFUNC : num  0.627 0.351 0.672 0.167 2.288 ...
##  $ AGE          : int  50 31 32 21 33 30 26 29 53 54 ...
##  $ DIABETES     : int  1 0 1 0 1 0 1 0 1 1 ...
```

```
summary(datos)
```

```
##     PREGNANT         GLUCOSE        BLOODPRESS     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      BODYMASSINDEX    PEDIGREEFUNC         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
##     DIABETES
##  Min.   :0.000
##  1st Qu.:0.000
##  Median :0.000
##  Mean   :0.349
##  3rd Qu.:1.000
##  Max.   :1.000
```

Realizamos algunos cambios en el dataset

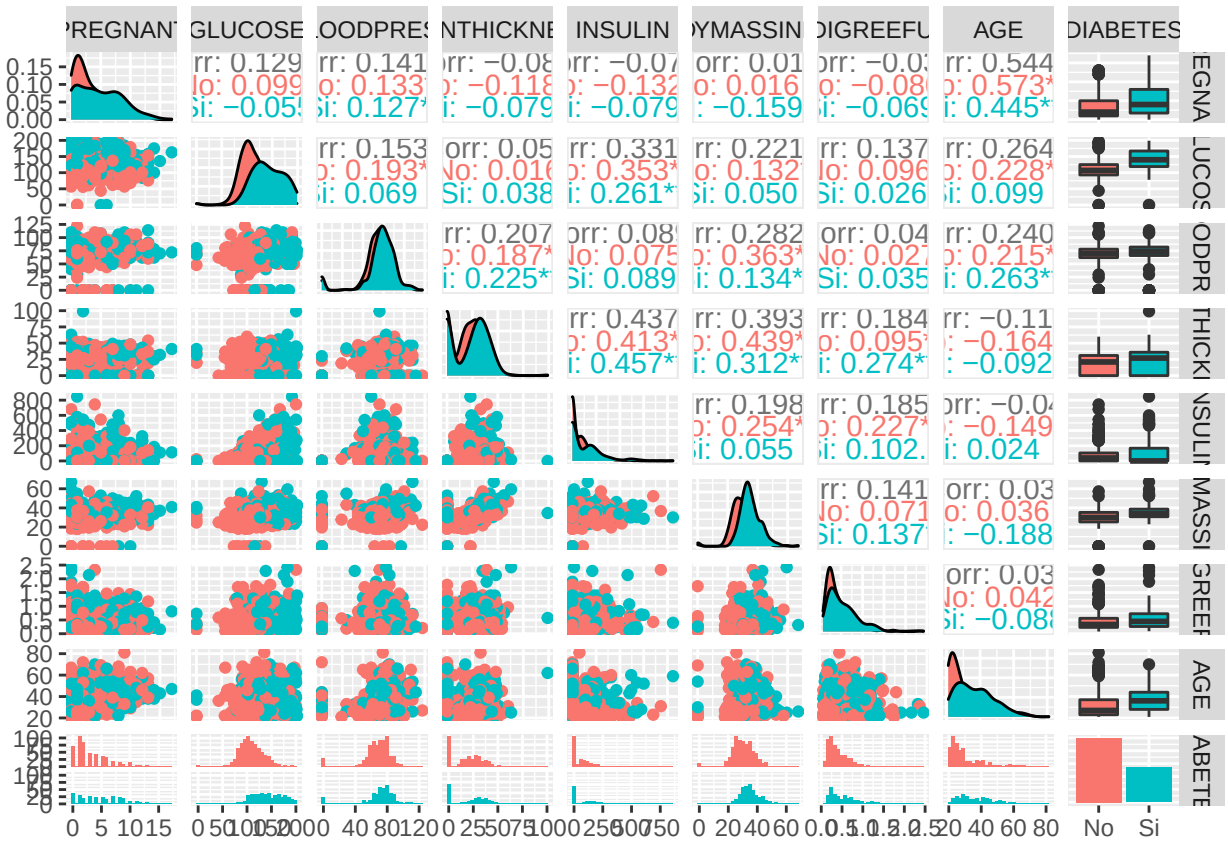
```
datos <- datos %>%
  mutate(DIABETES = ifelse(DIABETES == 1, "Si", "No"))
datos <- datos %>%
  mutate(DIABETES = as.factor(DIABETES))
str(datos)
```

```
## 'data.frame':    768 obs. of  9 variables:
##  $ PREGNANT     : int  6 1 8 1 0 5 3 10 2 8 ...
##  $ GLUCOSE      : int  148 85 183 89 137 116 78 115 197 125 ...
##  $ BLOODPRESS   : int  72 66 64 66 40 74 50 0 70 96 ...
##  $ SKINTHICKNESS: int  35 29 0 23 35 0 32 0 45 0 ...
##  $ INSULIN      : int  0 0 0 94 168 0 88 0 543 0 ...
##  $ BODYMASSINDEX: num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
##  $ PEDIGREEFUNC : num  0.627 0.351 0.672 0.167 2.288 ...
##  $ AGE          : int  50 31 32 21 33 30 26 29 53 54 ...
##  $ DIABETES     : Factor w/ 2 levels "No","Si": 2 1 2 1 2 1 2 1 2 2 ...
```

Vemos que no hay datos nulos y por lo tanto trabajaremos con todas las filas que hemos cargado

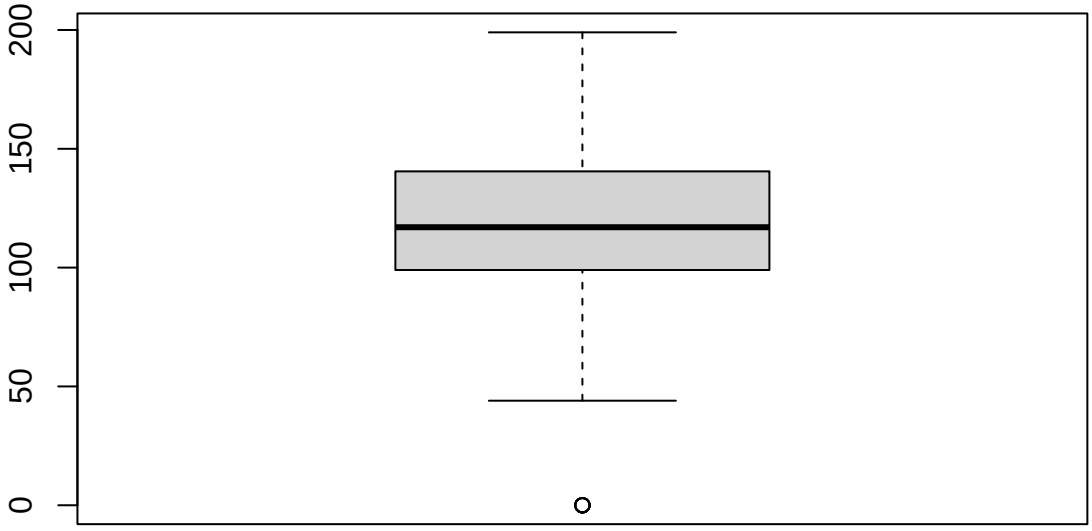
Antes de iniciar con todos los gráficos veamos una representación de todos con todos:

```
ggpairs(datos, aes(color = DIABETES))
```



Da la sensación de existir bastantes valores atipicos, veamoslo:

```
boxplot_glucosa <- boxplot(datos$GLUCOSE)
```




```
boxplot_glucosa$out
```

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

Una persona con nivel de glucosa 0 es imposible que este viva, por ello vamos a eliminar dichas observaciones.

```
datos2 <- datos
datos2 <- datos2[datos2$GLUCOSE > 0, ]
```

```
datos3 <- datos
datos3$GLUCOSE <- ifelse(datos3$GLUCOSE == 0, median(datos3$GLUCOSE), datos3$GLUCOSE)
```

```
datos4 <- datos
datos4$GLUCOSE <- ifelse(datos4$GLUCOSE == 0, mean(datos4$GLUCOSE), datos4$GLUCOSE)
```

Comprobamos mirando un modelo sencillo cual es la mejor opción:

```
trainIndex <- createDataPartition(datos2$DIABETES,
                                  p = 0.8,
                                  list = FALSE,
                                  times = 1)
fTR2 <- datos2[trainIndex,]
fTS2 <- datos2[-trainIndex,]
fTR_eval2 <- fTR2
fTS_eval2 <- fTS2

ctrl <- trainControl(method = "cv",                         #k-fold cross-validation
                     number = 10,                           #Number of folds
                     summaryFunction = defaultSummary,      #Performance summary for comparing models in
                     classProbs = TRUE)


LogReg.fit2 <- train(form = DIABETES ~ GLUCOSE,
                     data = fTR2,
                     method = "glm",
                     preProcess = c("center","scale"),
                     metric = "Accuracy",
                     trControl = ctrl)
LogReg.fit2
```

```
## Generalized Linear Model
##
## 611 samples
##   1 predictor
##   2 classes: 'No', 'Si'
##
## Pre-processing: centered (1), scaled (1)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 550, 549, 550, 551, 550, 550, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7527886  0.4175729
```

```
trainIndex <- createDataPartition(datos3$DIABETES,
                                  p = 0.8,
                                  list = FALSE,
                                  times = 1)
fTR3 <- datos3[trainIndex,]
fTS3 <- datos3[-trainIndex,]
fTR_eval3 <- fTR3
fTS_eval3 <- fTS3

ctrl <- trainControl(method = "cv",                         #k-fold cross-validation
```

```
                     number = 10,                              #Number of folds
                     summaryFunction = defaultSummary,     #Performance summary for comparing models in
                     classProbs = TRUE)


LogReg.fit3 <- train(form = DIABETES ~ GLUCOSE,
                     data = fTR3,
                     method = "glm",
                     preProcess = c("center","scale"),
                     metric = "Accuracy",
                     trControl = ctrl)
LogReg.fit3
```

```
## Generalized Linear Model
##
## 615 samples
##   1 predictor
##   2 classes: 'No', 'Si'
##
## Pre-processing: centered (1), scaled (1)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 553, 554, 553, 554, 553, 554, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7530407  0.4225187
```

```
trainIndex <- createDataPartition(datos4$DIABETES,
                                  p = 0.8,
                                  list = FALSE,
                                  times = 1)
fTR4 <- datos4[trainIndex,]
fTS4 <- datos4[-trainIndex,]
fTR_eval4 <- fTR4
fTS_eval4 <- fTS4

ctrl <- trainControl(method = "cv",                            #k-fold cross-validation
                     number = 10,                              #Number of folds
                     summaryFunction = defaultSummary,     #Performance summary for comparing models in
                     classProbs = TRUE)


LogReg.fit4 <- train(form = DIABETES ~ GLUCOSE,
                     data = fTR4,
                     method = "glm",
                     preProcess = c("center","scale"),
                     metric = "Accuracy",
                     trControl = ctrl)
LogReg.fit4
```

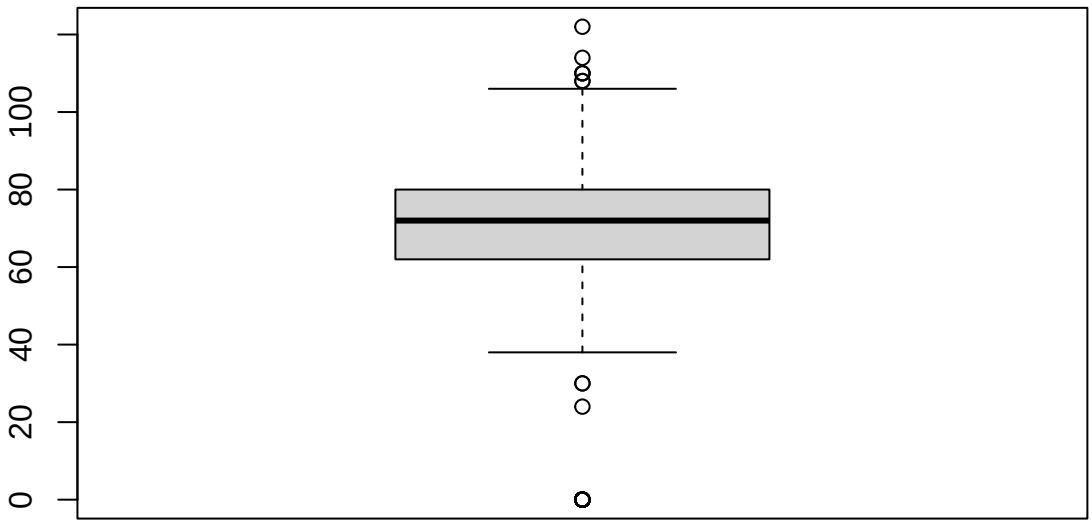
```
## Generalized Linear Model
##
## 615 samples
##   1 predictor
```

```
##   2 classes: 'No', 'Si'
##
## Pre-processing: centered (1), scaled (1)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 553, 553, 554, 554, 553, 553, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7418033  0.3971726
```

El modelo que mejor nos ha salido es en el que cambiamos los valores de 0 de la glucosa por la mediana.

Veamos ahora la presión sanguinea:

```
boxplot_blood <- boxplot(datos$BLOODPRESS)
```



Volvemos a realizar lo mismo que en el caso anterior:

```
datos2 <- datos2[datos2$BLOODPRESS > 0, ]
```

```
datos3$BLOODPRESS <- ifelse(datos3$BLOODPRESS == 0, median(datos3$BLOODPRESS), datos3$BLOODPRESS)
```

```
datos4$BLOODPRESS <- ifelse(datos4$BLOODPRESS == 0, mean(datos4$BLOODPRESS), datos4$BLOODPRESS)
```

```
trainIndex <- createDataPartition(datos2$DIABETES,
                                  p = 0.8,
                                  list = FALSE,
                                  times = 1)
fTR2 <- datos2[trainIndex,]
fTS2 <- datos2[-trainIndex,]
fTR_eval2 <- fTR2
fTS_eval2 <- fTS2

ctrl <- trainControl(method = "cv",                          #k-fold cross-validation
                     number = 10,                            #Number of folds
                     summaryFunction = defaultSummary,       #Performance summary for comparing models in
                     classProbs = TRUE)


LogReg.fit2 <- train(form = DIABETES ~ BLOODPRESS,
                     data = fTR2,
```

```
                    method = "glm",
                    preProcess = c("center","scale"),
                    metric = "Accuracy",
                    trControl = ctrl)
LogReg.fit2
```

```
## Generalized Linear Model
##
## 583 samples
##   1 predictor
##   2 classes: 'No', 'Si'
##
## Pre-processing: centered (1), scaled (1)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 525, 525, 525, 524, 525, 525, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.6603741  0.05210665
```

```
trainIndex <- createDataPartition(datos3$DIABETES,
                                  p = 0.8,
                                  list = FALSE,
                                  times = 1)
fTR3 <- datos3[trainIndex,]
fTS3 <- datos3[-trainIndex,]
fTR_eval3 <- fTR3
fTS_eval3 <- fTS3

ctrl <- trainControl(method = "cv",                          #k-fold cross-validation
                     number = 10,                            #Number of folds
                     summaryFunction = defaultSummary,       #Performance summary for comparing models in
                     classProbs = TRUE)


LogReg.fit3 <- train(form = DIABETES ~ BLOODPRESS,
                    data = fTR3,
                    method = "glm",
                    preProcess = c("center","scale"),
                    metric = "Accuracy",
                    trControl = ctrl)
LogReg.fit3
```

```
## Generalized Linear Model
##
## 615 samples
##   1 predictor
##   2 classes: 'No', 'Si'
##
## Pre-processing: centered (1), scaled (1)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 554, 554, 554, 553, 553, 553, ...
## Resampling results:
##
```

```
##   Accuracy   Kappa
##   0.6553675  0.03954174
```

```
trainIndex <- createDataPartition(datos4$DIABETES,
                                  p = 0.8,
                                  list = FALSE,
                                  times = 1)
fTR4 <- datos4[trainIndex,]
fTS4 <- datos4[-trainIndex,]
fTR_eval4 <- fTR4
fTS_eval4 <- fTS4

ctrl <- trainControl(method = "cv",                         #k-fold cross-validation
                     number = 10,                           #Number of folds
                     summaryFunction = defaultSummary,      #Performance summary for comparing models in
                     classProbs = TRUE)


LogReg.fit4 <- train(form = DIABETES ~ BLOODPRESS,
                     data = fTR4,
                     method = "glm",
                     preProcess = c("center","scale"),
                     metric = "Accuracy",
                     trControl = ctrl)
LogReg.fit4
```

```
## Generalized Linear Model
##
## 615 samples
##   1 predictor
##   2 classes: 'No', 'Si'
##
## Pre-processing: centered (1), scaled (1)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 554, 553, 554, 553, 554, 554, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.6521682  0.02819505
```

Hacemos lo mismo con el bodymassindex:

```
datos2 <- datos2[datos2$BODYMASSINDEX > 0, ]
```

```
datos3$BODYMASSINDEX <- ifelse(datos3$BODYMASSINDEX == 0, median(datos3$BODYMASSINDEX), datos3$BODYMASS
```

```
datos4$BODYMASSINDEX <- ifelse(datos4$BODYMASSINDEX == 0, mean(datos4$BODYMASSINDEX), datos4$BODYMASSIN
```

Y entrenamos modelos:

```
trainIndex <- createDataPartition(datos2$DIABETES,
                                  p = 0.8,
                                  list = FALSE,
                                  times = 1)
fTR2 <- datos2[trainIndex,]
fTS2 <- datos2[-trainIndex,]
fTR_eval2 <- fTR2
```

```r
fTS_eval2 <- fTS2

ctrl <- trainControl(method = "cv",                         #k-fold cross-validation
                     number = 10,                           #Number of folds
                     summaryFunction = defaultSummary,      #Performance summary for comparing models in
                     classProbs = TRUE)


LogReg.fit2 <- train(form = DIABETES ~ BODYMASSINDEX,
                     data = fTR2,
                     method = "glm",
                     preProcess = c("center","scale"),
                     metric = "Accuracy",
                     trControl = ctrl)
LogReg.fit2
```

```
## Generalized Linear Model
##
## 580 samples
##   1 predictor
##   2 classes: 'No', 'Si'
##
## Pre-processing: centered (1), scaled (1)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 522, 522, 522, 522, 522, 522, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.6586207  0.1178875
```

```r
trainIndex <- createDataPartition(datos3$DIABETES,
                                  p = 0.8,
                                  list = FALSE,
                                  times = 1)
fTR3 <- datos3[trainIndex,]
fTS3 <- datos3[-trainIndex,]
fTR_eval3 <- fTR3
fTS_eval3 <- fTS3

ctrl <- trainControl(method = "cv",                         #k-fold cross-validation
                     number = 10,                           #Number of folds
                     summaryFunction = defaultSummary,      #Performance summary for comparing models in
                     classProbs = TRUE)


LogReg.fit3 <- train(form = DIABETES ~ BODYMASSINDEX,
                     data = fTR3,
                     method = "glm",
                     preProcess = c("center","scale"),
                     metric = "Accuracy",
                     trControl = ctrl)
LogReg.fit3
```

```
## Generalized Linear Model
```

```
##
## 615 samples
##   1 predictor
##   2 classes: 'No', 'Si'
##
## Pre-processing: centered (1), scaled (1)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 553, 554, 553, 553, 554, 554, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.6635378  0.1360155
```

```
trainIndex <- createDataPartition(datos4$DIABETES,
                                  p = 0.8,
                                  list = FALSE,
                                  times = 1)
fTR4 <- datos4[trainIndex,]
fTS4 <- datos4[-trainIndex,]
fTR_eval4 <- fTR4
fTS_eval4 <- fTS4

ctrl <- trainControl(method = "cv",                         #k-fold cross-validation
                     number = 10,                           #Number of folds
                     summaryFunction = defaultSummary,      #Performance summary for comparing models in
                     classProbs = TRUE)


LogReg.fit4 <- train(form = DIABETES ~ BODYMASSINDEX,
                     data = fTR4,
                     method = "glm",
                     preProcess = c("center","scale"),
                     metric = "Accuracy",
                     trControl = ctrl)
LogReg.fit4
```

```
## Generalized Linear Model
##
## 615 samples
##   1 predictor
##   2 classes: 'No', 'Si'
##
## Pre-processing: centered (1), scaled (1)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 554, 554, 553, 554, 554, 553, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.6618191  0.1524195
```

Hacemos lo mismo con el insulina, lo unico que en este caso no contemplamos eliminar los datos que son 0 ya que representan gran cantidad de nuestro dataset.

```
datos3$INSULIN <- ifelse(datos3$INSULIN == 0, median(datos3$INSULIN), datos3$INSULIN)
```

```r
datos4$INSULIN <- ifelse(datos4$INSULIN == 0, mean(datos4$INSULIN), datos4$INSULIN)
```

Y entrenamos modelos:

```r
trainIndex <- createDataPartition(datos3$DIABETES,
                                  p = 0.8,
                                  list = FALSE,
                                  times = 1)
fTR3 <- datos3[trainIndex,]
fTS3 <- datos3[-trainIndex,]
fTR_eval3 <- fTR3
fTS_eval3 <- fTS3

ctrl <- trainControl(method = "cv",                         #k-fold cross-validation
                     number = 10,                           #Number of folds
                     summaryFunction = defaultSummary,      #Performance summary for comparing models in
                     classProbs = TRUE)


LogReg.fit3 <- train(form = DIABETES ~ INSULIN,
                     data = fTR3,
                     method = "glm",
                     preProcess = c("center","scale"),
                     metric = "Accuracy",
                     trControl = ctrl)
LogReg.fit3
```

```
## Generalized Linear Model
##
## 615 samples
##   1 predictor
##   2 classes: 'No', 'Si'
##
## Pre-processing: centered (1), scaled (1)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 554, 554, 554, 554, 553, 553, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.6600212  0.07092442
```

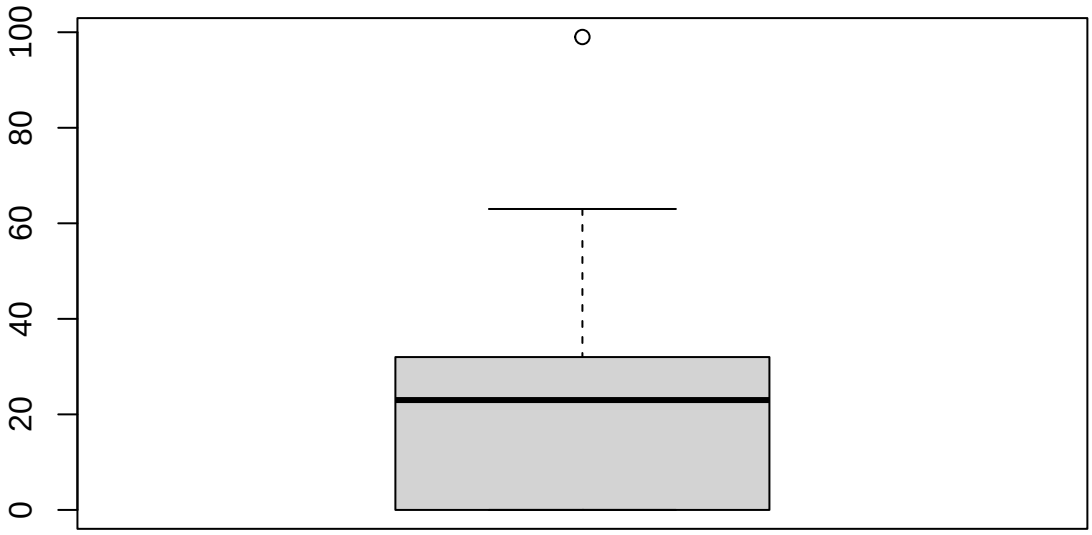
```r
trainIndex <- createDataPartition(datos4$DIABETES,
                                  p = 0.8,
                                  list = FALSE,
                                  times = 1)
fTR4 <- datos4[trainIndex,]
fTS4 <- datos4[-trainIndex,]
fTR_eval4 <- fTR4
fTS_eval4 <- fTS4

ctrl <- trainControl(method = "cv",                         #k-fold cross-validation
                     number = 10,                           #Number of folds
                     summaryFunction = defaultSummary,      #Performance summary for comparing models in
                     classProbs = TRUE)
```

```
LogReg.fit4 <- train(form = DIABETES ~ INSULIN,
                     data = fTR4,
                     method = "glm",
                     preProcess = c("center","scale"),
                     metric = "Accuracy",
                     trControl = ctrl)
LogReg.fit4
```

```
## Generalized Linear Model
##
## 615 samples
##   1 predictor
##   2 classes: 'No', 'Si'
##
## Pre-processing: centered (1), scaled (1)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 553, 554, 553, 554, 554, 553, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.655156  0.06712536
```

Por último nos queda ver la variable Sthickness que era la otra que tenía datos atípicos a simple vista:

```
boxplot_sthick <- boxplot(datos$SKINTHICKNESS)
```



Aunque no se vean tambien existen valores atípicos, que son erroneos, como es el caso del 0.

Por tanto nos encontramos en el mismo caso:

```
datos2 <- datos2[datos2$SKINTHICKNESS > 0 & datos2$SKINTHICKNESS < 90, ]
```

```
datos3$SKINTHICKNESS <- ifelse(datos3$SKINTHICKNESS == 0 , median(datos3$SKINTHICKNESS), datos3$SKINTHI
datos3$SKINTHICKNESS <- ifelse(datos3$SKINTHICKNESS > 90 , median(datos3$SKINTHICKNESS), datos3$SKINTHI
```

```
datos4$SKINTHICKNESS <- ifelse(datos4$SKINTHICKNESS == 0, mean(datos4$SKINTHICKNESS), datos4$SKINTHICKN
datos4$SKINTHICKNESS <- ifelse(datos4$SKINTHICKNESS > 90, mean(datos4$SKINTHICKNESS), datos4$SKINTHICKN
```

Y entrenamos modelos:

```
trainIndex <- createDataPartition(datos2$DIABETES,
                                  p = 0.8,
                                  list = FALSE,
                                  times = 1)
fTR2 <- datos2[trainIndex,]
fTS2 <- datos2[-trainIndex,]
fTR_eval2 <- fTR2
fTS_eval2 <- fTS2

ctrl <- trainControl(method = "cv",                         #k-fold cross-validation
                     number = 10,                           #Number of folds
                     summaryFunction = defaultSummary,      #Performance summary for comparing models in
                     classProbs = TRUE)


LogReg.fit2 <- train(form = DIABETES ~ SKINTHICKNESS,
                     data = fTR2,
                     method = "glm",
                     preProcess = c("center","scale"),
                     metric = "Accuracy",
                     trControl = ctrl)
LogReg.fit2
```

```
## Generalized Linear Model
##
## 425 samples
##   1 predictor
##   2 classes: 'No', 'Si'
##
## Pre-processing: centered (1), scaled (1)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 383, 383, 382, 383, 383, 382, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.6728128  0.08781002
```

```
trainIndex <- createDataPartition(datos3$DIABETES,
                                  p = 0.8,
                                  list = FALSE,
                                  times = 1)
fTR3 <- datos3[trainIndex,]
fTS3 <- datos3[-trainIndex,]
fTR_eval3 <- fTR3
fTS_eval3 <- fTS3

ctrl <- trainControl(method = "cv",                         #k-fold cross-validation
                     number = 10,                           #Number of folds
                     summaryFunction = defaultSummary,      #Performance summary for comparing models in
                     classProbs = TRUE)


LogReg.fit3 <- train(form = DIABETES ~ SKINTHICKNESS,
                     data = fTR3,
```

```
                    method = "glm",
                    preProcess = c("center","scale"),
                    metric = "Accuracy",
                    trControl = ctrl)
LogReg.fit3
```

```
## Generalized Linear Model
##
## 615 samples
##   1 predictor
##   2 classes: 'No', 'Si'
##
## Pre-processing: centered (1), scaled (1)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 554, 553, 554, 553, 554, 554, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.6505288  0.05336638
```

```
trainIndex <- createDataPartition(datos4$DIABETES,
                                  p = 0.8,
                                  list = FALSE,
                                  times = 1)
fTR4 <- datos4[trainIndex,]
fTS4 <- datos4[-trainIndex,]
fTR_eval4 <- fTR4
fTS_eval4 <- fTS4

ctrl <- trainControl(method = "cv",                          #k-fold cross-validation
                     number = 10,                            #Number of folds
                     summaryFunction = defaultSummary,       #Performance summary for comparing models in
                     classProbs = TRUE)


LogReg.fit4 <- train(form = DIABETES ~ SKINTHICKNESS,
                    data = fTR4,
                    method = "glm",
                    preProcess = c("center","scale"),
                    metric = "Accuracy",
                    trControl = ctrl)
LogReg.fit4
```

```
## Generalized Linear Model
##
## 615 samples
##   1 predictor
##   2 classes: 'No', 'Si'
##
## Pre-processing: centered (1), scaled (1)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 554, 553, 554, 554, 553, 554, ...
## Resampling results:
##
```

```
##    Accuracy    Kappa
##    0.6438921   0.05777162
```

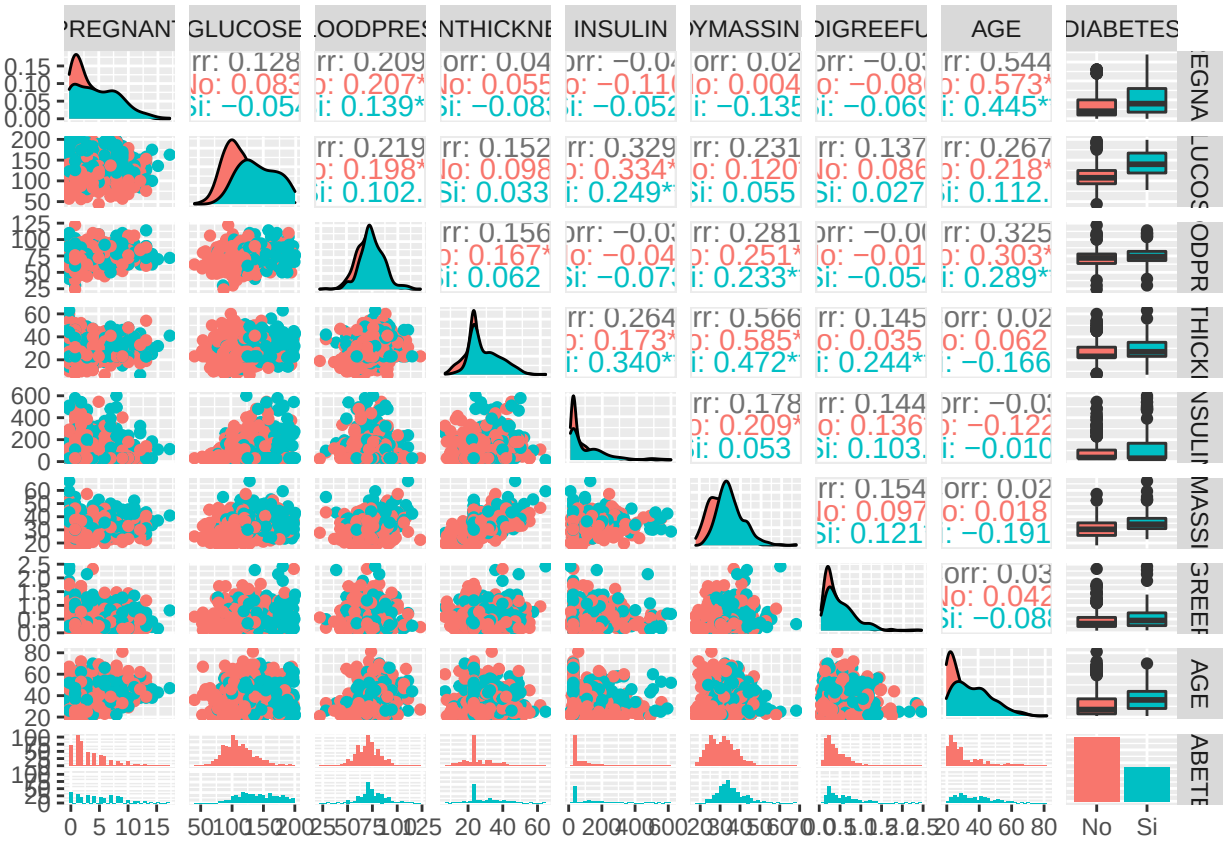
Hacemos todos los cambios que hemos ido deciciendo sobre el dataset original:

```
datos$GLUCOSE <- ifelse(datos$GLUCOSE == 0, median(datos$GLUCOSE), datos$GLUCOSE)
datos$BODYMASSINDEX <- ifelse(datos$BODYMASSINDEX == 0, median(datos$BODYMASSINDEX), datos$BODYMASSINDE
datos$BLOODPRESS <- ifelse(datos$BLOODPRESS == 0, median(datos$BLOODPRESS), datos$BLOODPRESS)
datos$INSULIN <- ifelse(datos$INSULIN == 0, median(datos$INSULIN), datos$INSULIN)
datos$INSULIN <- ifelse(datos$INSULIN > 600, median(datos$INSULIN), datos$INSULIN)
datos$SKINTHICKNESS <- ifelse(datos$SKINTHICKNESS == 0, median(datos$SKINTHICKNESS), datos$SKINTHICKNES
datos$SKINTHICKNESS <- ifelse(datos$SKINTHICKNESS > 90, median(datos$SKINTHICKNESS), datos$SKINTHICKNES
```

Y volvemos a hacer un EDA:

```
ggpairs(datos, aes(color = DIABETES))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Todo bien, todo correcto.

Identification and fitting process of classification models

Lo primero que hacemos antes de realizar ningún modelo será informarnos acerca del tema, con la finalidad de conocer cuales son las variables que más afectan a la diabetes.

Tras leer numerosos articulos podemos llegar a la conclusion que las variables que mas afectan a nuestra variable respuesta son tener obesidad, edad, presión arterial alta, antecedentes familiares y altos niveles de glucosa.

Para comenzar con el modelo lo que haremos será dividir nuestra muestra en 2 grupos, el de entrenamiento y el de test. La proporción con la que trabajaremos será de un 80-20.