Trabajo Práctico Nº 4

Arboles de Decision

Presentado en la fecha: 21/10/2023

Hecho por: Huarca Brian

Nicolas Benitez

Facundo Rodriguez

Derlis Walter Hodge

Contents

Resumen

Para la siguiente práctica de árboles de decisión utilizaremos un set de datos que contiene observaciones del programa Aprender 2016 - Primaria.

Sumario

0.1 Sumario

- Preparación del dataset (Valores Null, Outliers, Casteo de Datos).
- Clasificacion de Variables.
- Discretizacion de Valores.
- Analisis del Arbol de Decision.
- Analisis de Importancia de las Variables
- Analisis de Matriz de Confusion.
- Comparacion y Analisis de Modelos RandomForest

Objetivo

El informe tiene por objetivo el analisis de datos para realizar una clasificacion en base a arboles de decision con los metodos RPart Y RForest. Haciendo un analisis de Variables para determinar una predictora, realizar si fuese necesario la discretizacion y la comparacion de distintos modelos con RForest en caso de visualizar altos niveles de Error OOB.

Dataset

1.1 Informacion del dataset

Es el dispositivo nacional de evaluación de los aprendizajes de los estudiantes y de sistematización de información acerca de algunas condiciones en las que ellos se desarrollan.
El documento se basa en una evaluacion a jovenes de una institucion para medir su desenpeño en sus clases de matematicas y practicas del lenguaje. Se nos presentan distintas preguntas realizadas a los alumnos como a que tipo de institucion asisten, en cual sector, si repitio de grado, entre otras.

1.2 Diccionario de datos

- **ap1** = ¿Cuántos años tenés?.
- **ap2** = ¿Sos varón o mujer?.
- **ap3** = ¿Fuiste al jardín de infantes?.
- **ap4** = ¿Alguna vez repetiste de grado?.
- **ap5** = ¿Cómo leés?.
- **ap6** = ¿Cómo escribís?.
- **ap7** = ¿Cómo resolvés problemas o cuentas de matemática?
- **ap8a** = ¿Los maestros y maestras pueden explicar sin que nadie los interrumpa?
- **ap8b** = ¿Los maestros y maestras se enojan con ustedes?
- **ap8c** = ¿Los maestros y maestras explican los temas hasta que los entendés?
- **ap9** = ¿Te llevás bien con tus compañeros?
- **ap10** = Si te dijeran que tenés que cambiarte de escuela, ¿cómo te sentirías?
- **ap11** = Cuando estás en la escuela, ¿estás contento?
- **ap12** = Cuando estás en la escuela, ¿estás aburrido?
- **ap13** = Cuando estás en la escuela, ¿te sentís incómodo?
- **ap14** = Resolver la prueba de Lengua, ¿te fue fácil o difícil?
- **ap15** = Resolver la prueba de Matemática, ¿te fue fácil o difícil?
- **ponder** = Factor de expansión
- **cod**$_p rovincia = Jurisdiccin$
- **lpondera** = Factor de expansión Lengua
- **lpuntaje** = Puntaje Lengua
- **ldesemp** = Nivel de Desempeño en Lengua

- **mpondera** = Factor de expansión Matemática
- **mpuntaje** = Puntaje Matemática
- **mdesemp** = Nivel de Desempeño en Matemática
- **sector** = Sector de Gestión
- **ambito** = Espacio donde se ubica Institucion Rural-Urbano
- **qvulneraa** = Cuartil de alumnos según el porcentaje de hogares en el radio de la escuela en estrato socioeconómico muy bajo Total País
- **qvulneraap** = Cuartil de alumnos según el porcentaje de hogares en el radio de la escuela en estrato socioeconómico muy bajo por jurisdicción
- **iclima** = Índice de Clima escolar
- **autoconl** = Autoconcepto en Lengua alumnos que participaron en evaluación de Lengua
- **autoconm** = Autoconcepto en Matemática alumnos que participaron en evaluación de Matemática

Dataset

2.1 Deteccion de outliers

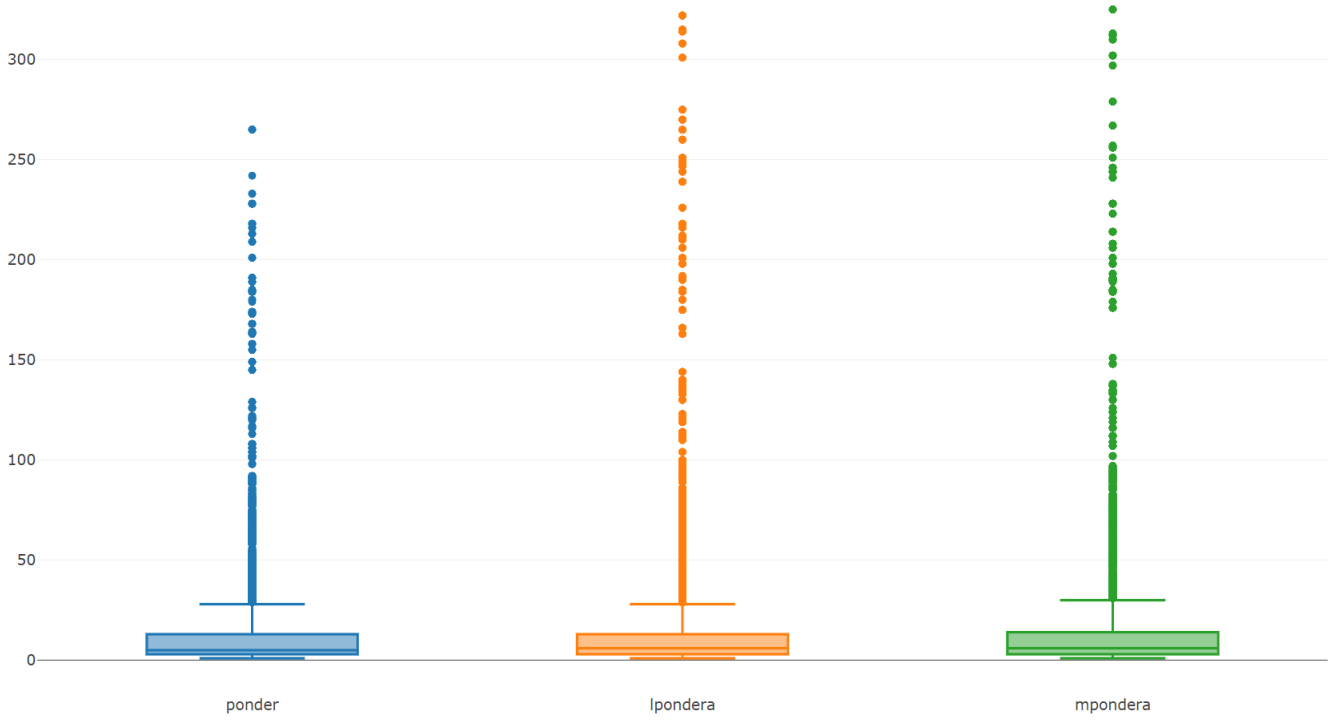


Figure 1: Boxplot

En este BoxPlot podemos ver una cantidad de Outliers Heterogeneos de las variables Ponder, Mpondera y Lpondera. Estas variables representan el factor de expansion general, el factor de expansion en practicas del lenguaje y en matematicas.

RPart

3.1

Separamos los datos en entrenamiento y testing (70/30%)

- entrenamiento: 34153 observaciones
- testing: 14636 observaciones

Con la función Rpart creamos un árbol.
El Rpart decide con qué variables se queda para la creación del árbol a partir de cuáles son las mas significativas.
Todo en concordancia con nuestro análisis previo. Así sucesivamente se van formando las ramas del árbol hasta quedar con las hojas/categorías finales.

3.2 Arbol de Decision

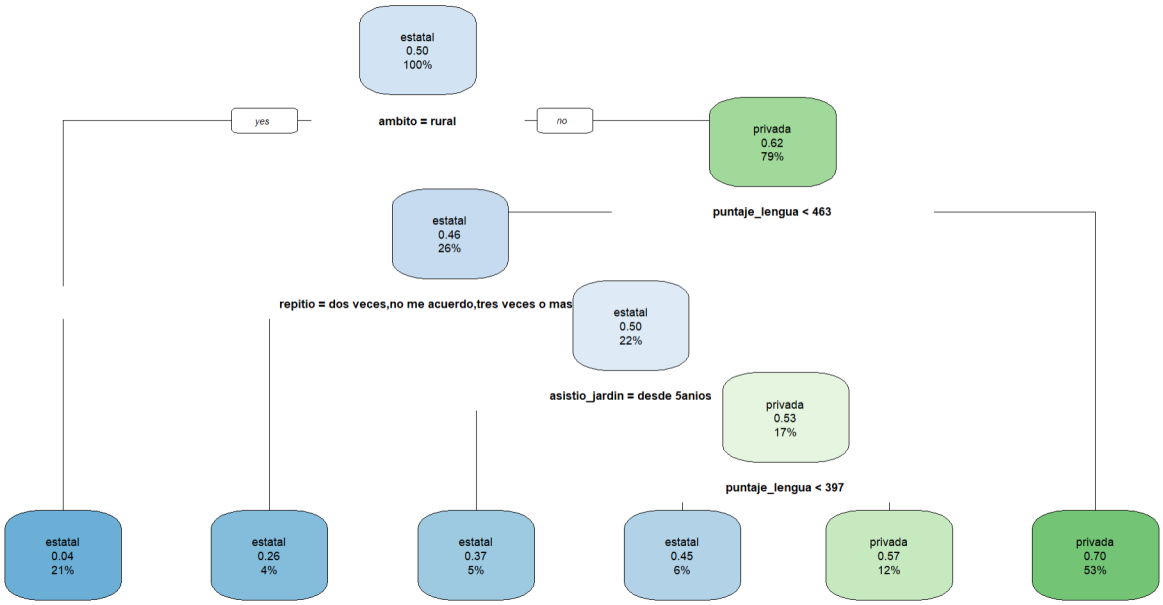


Figure 2

Aca podemos apreciar nuestro arbol de decision el cual nuestra variable "Predictora" a utilizar es la variable 'Sector', la cual indica si el alumno estudia en una institucion publica o privada, por lo cual nos da de resultado en nuestra raiz que un 50 del 100 de nuestra muestra total estudia en una institucion publica. Bajando un poco en detalles en el arbol notamos que la variable ambito es igual a Rural por ende si vamos a la Izquierda, osea por el lado del SI, vemos que la gran mayoria tiene chances de estudiar en un ambito rural y un 0.04 NO.

3.3 Analisis de Importancia de las Variables

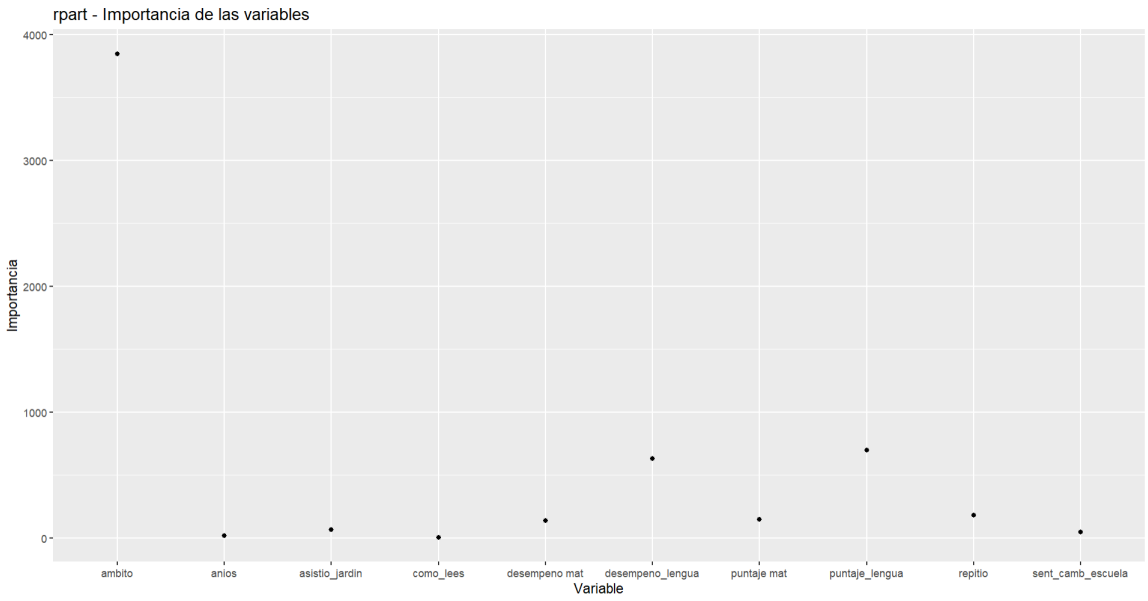


Figure 3

Aca podemos observar que entre las variables con mayor importancia estan las variables que preguntan si los alumnos alguna vez repitieron, cuando comenzaron el jardin, su puntaje y desempeño en matematicas y practicas del lenguaje. Pero la que mas resalta entre ellas es el ambito escolar en el cual ellos estudian.

3.4 Matriz de Confusion

Aqui podemos visualizar lo que es nuestra Matriz de Confusion sobre el modelo RPart realizado, en la cual se puede apreciar nuestra Taza de Exactitud (Accuracy) es del 0.72, un P-Value = 2.2e-16 que se encuentra por debajo del 0.05 por lo cual se rechaza la hipotesis nula, una sensibilidad del 0.82 y una especificidad del 0.66.

```
> tabla_mc_aprender
Confusion Matrix and Statistics

          Predicho
Actual     estatal privada
  estatal    4224    3161
  privada     888    6363

               Accuracy : 0.7234
                 95% CI : (0.716, 0.7306)
    No Information Rate : 0.6507
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4482

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8263
            Specificity : 0.6681
         Pos Pred Value : 0.5720
         Neg Pred Value : 0.8775
             Prevalence : 0.3493
         Detection Rate : 0.2886
   Detection Prevalence : 0.5046
      Balanced Accuracy : 0.7472

       'Positive' Class : estatal

>
```

Figure 4

RForest

4.1 Análisis con Random Forest

Para realizar un modelo mas profundo y buscar mejorar la exactitud de la categorización procedimos a utilizar Random Forest, obteniendo los siguientes resultados.

```
           edad            sexo                                                                    grado
15 anios      :7202   Femenino :17229   1er anio/10mo grado nivel Polimodal o 3er anio nivel Secundario:7
14 anios      :7020   Masculino:14636   2do anio/11vo grado nivel Polimodal o 4to anio nivel Secundario:7
16 anios      :6738                     3er anio/12vo grado nivel Polimodal o 5to anio nivel Secundario:5
13 anios      :5854                     8vo grado nivel Primario/Polimodal  o 1er anio nivel Secundario:4
17 anios      :4773                     9no grado nivel Primario/Polimodal o 2do anio nivel Secundario :6
18 anios o mas: 201
(Other)       :  77
     dias_con_hambre     ult_12meses_atac_fisi    ult_12meses_part_pelea         lesion_grave
Algunas veces: 2743   Ninguna       :27076     Ninguna       :25422      Ninguna       :21559
Casi siempre :  341   1 vez         : 2648     1 vez         : 3472      1 vez         : 6138
Nunca        :22093   2 o 3 veces   : 1192     2 o 3 veces   : 1758      2 o 3 veces   : 2893
Rara vez     : 6568   4 o 5 veces   :  334     4 o 5 veces   :  462      4 o 5 veces   :  670
Siempre      :  120   12 o mas veces:  329     12 o mas veces:  366      6 o 7 veces   :  224
                      6 o 7 veces   :  145     6 o 7 veces   :  196      12 o mas veces:  208
                      (Other)       :  141     (Other)       :  189      (Other)       :  173
```

Figure 5

```
consideraste_suicidio plan_de_suicidio   veces_intentaste_suici amigos_cercanos                    edad_f
No:25433              No:26838         0 veces      :27833      0       : 1604   Nunca probÃ© cigarrillos
Si: 6432              Si: 5027         1 vez        : 2470      1       : 2565   12 o 13 aÃ±os
                                       2 o 3 veces  : 1022      2       : 5281   14 o 15 aÃ±os
                                       4 o 5 veces  :  270      3 o mÃ¡s:22415   10 u 11 aÃ±os
                                       6 o mas veces:  270                       16 o 17 aÃ±os
                                                                                 8 o 9 aÃ±os
                                                                                 (Other)
```

Figure 6

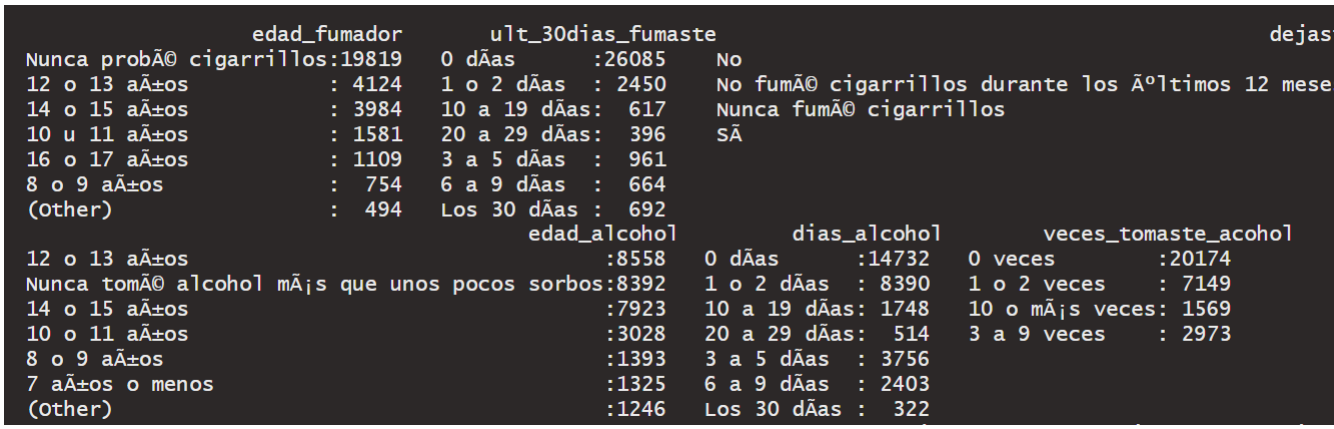


Figure 7

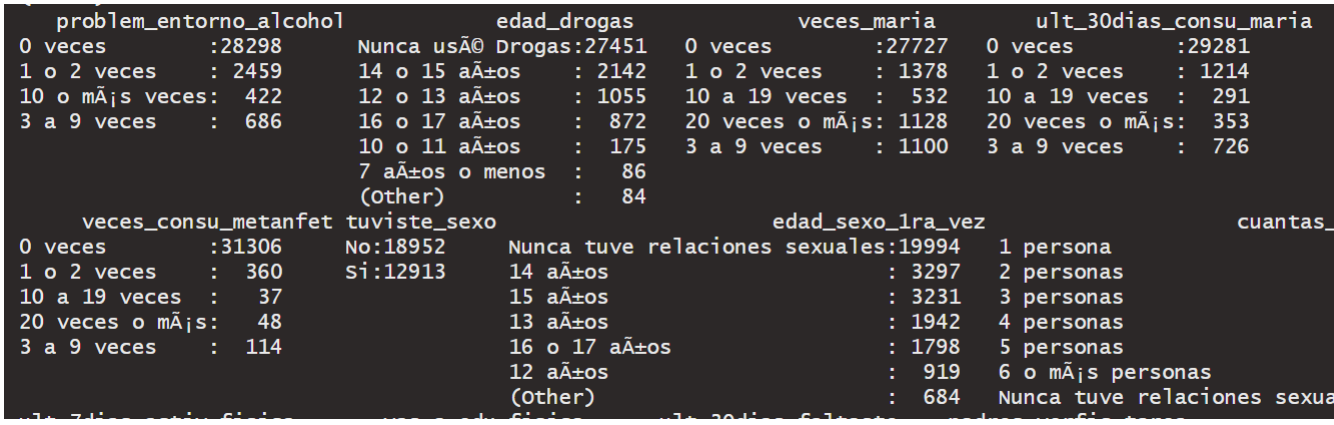


Figure 8

4.2 Resumen Matriz de Confusion

```
> emse_red_rf_pred_mc
Confusion Matrix and Statistics

          Predicho
Actual      No   Si
    No    7767  284
    Si     629  879

                  Accuracy : 0.9045
                    95% CI : (0.8984, 0.9103)
    No Information Rate : 0.8783
    P-Value [Acc > NIR] : 3.559e-16

                   Kappa : 0.6037

 Mcnemar's Test P-Value : < 2.2e-16

              Sensitivity : 0.9251
              Specificity : 0.7558
           Pos Pred Value : 0.9647
           Neg Pred Value : 0.5829
               Prevalence : 0.8783
           Detection Rate : 0.8125
     Detection Prevalence : 0.8422
        Balanced Accuracy : 0.8404

         'Positive' Class : No
```

Figure 9

En esta matriz nuestros casos positivos se representan con los casos Negativos como Verdaderos Positivos y los valores Positivos como Falsos Negativos. Observamos que Nuestra Exactitud del modelo es del 0.90 y nuestro P-Value es igual a 3.559e-16, Nuestra Sensibilidad es del 0.92 y la especificidad es del 0.75.

Tecnicas de balanceo de clases

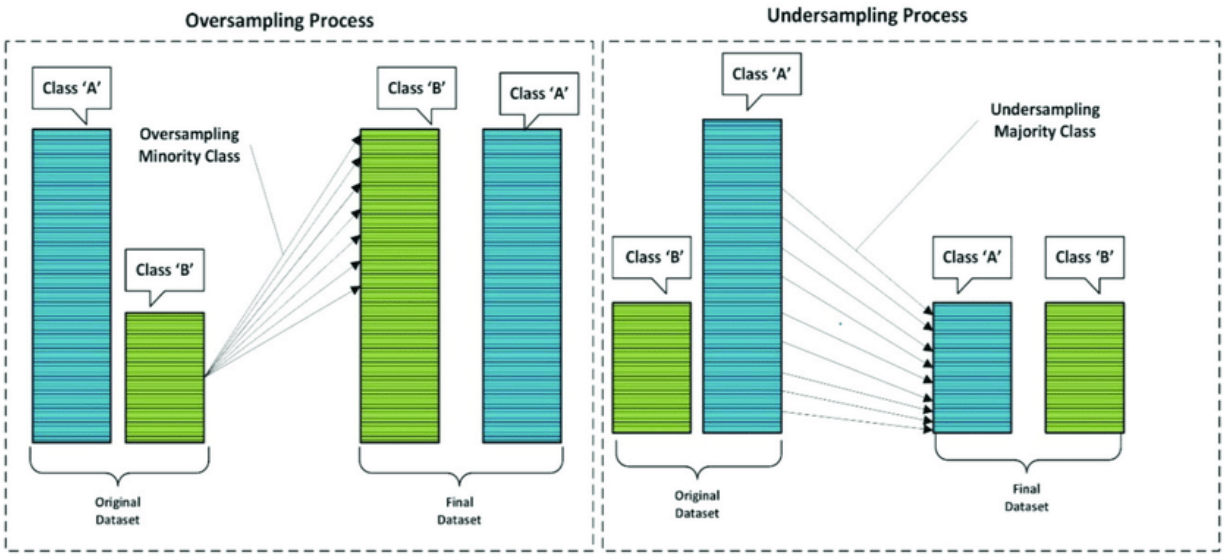
5.1 Undersampling

Es una técnica que consiste en mantener todos los datos de la clase de menor frecuencia y reducir la cantidad de los de la clase de mayor frecuencia, haciendo que las observaciones del conjunto tengan datos con la variable objetivo balanceada.

5.2 Oversampling

Es una técnica que consiste en aumentar el número de registros de la clase con menor frecuencia hasta que la base de datos tenga un número equilibrado entre las clases de la variable objetivo. Para aumentar la cantidad de registros, podemos duplicar aleatoriamente los registros de la clase con menos frecuencia. Sin embargo, esto hará que mucha información sea idéntica, lo que puede afectar el modelo.

Una ventaja de esta técnica es que no se pierde ninguna información de los registros que tenían la clase con mayor frecuencia. Esto hace que el conjunto de datos tenga muchos registros para alimentar los algoritmos de aprendizaje automático. A su vez, el tiempo de almacenamiento y procesamiento crece significativamente y existe la posibilidad de sobreajustar los datos que se han duplicado. Este sobreajuste ocurre cuando el modelo se vuelve muy bueno para predecir los resultados de los datos de entrenamiento, pero no generaliza bien los datos nuevos.


Oversampling Process
Class 'A'
Oversampling
Minority Class
Class 'B'
Original
Dataset
Class 'B'
Class 'A'
Final
Dataset
Undersampling Process
Class 'A'
Undersampling
Majority Class
Class 'B'
Original
Dataset
Class 'A'
Class 'B'
Final
Dataset

Conclusión

Para empezar realizamos varios análisis, un summary inicial para observar la distribución de las variables
Contando con un amplio número de observaciones y de variables se pudo obtener una predicción de clasificación aceptable.
A la vez afianzamos nuestro dominio sobre la herramienta R.

Anexo

```
1
2
3
4  ################################################################
5  # Arboles de desicion del dataset Aprender 2016.
6  #
7  # Creado: 19/10/2023
8  # Version: 1.0
9  # Autor: Grupo 4
10
11 ################################################################
12
13 ##### Importamos las biliotecas #####
14 library(plyr)
15 library(rpart)
16 library(rpart.plot)
17 library(caret)
18 library(ggplot2)
19 library(gridExtra)
20 library(tidyverse)
21 library(rsample)
22 library(e1071)
23 library(GGally)
24 library(data.table)
25 library(DT)
26 library(readr)
27
28 library(dplyr)
29 library(tidyr)
30 library(corrplot)
31 #library(rms)
32 library(MASS)
33 library(e1071)
34 library(ROCR)
35 library(gplots)
36 library(pROC)
37
38 library(randomForest)
39 library(ggpubr)
40 library(plotly)
41
42 ##### Lectura y verificacion de la informacion leida #####
```

```r
# Seteamos el entorno de desarrollo.
setwd(paste0(getwd(),"/" ))

aprender <- read_delim("./UNO_EDD/TP4_Arboles_Decision/aprender2016-primaria-3.csv",
                       delim = ";", escape_double = FALSE, locale = locale(decimal_mark = ",",
                                                                           grouping_mark = "."), trim_ws = TRUE)
View(aprender)
dim(aprender)
nrow(aprender)

head(aprender)


# Verificamos si hay valores nulos.
sapply(aprender, function(x) sum(is.na(x)))

# complete.cases() devuelve un vector con los valores completos y que sean no nulos.
aprender_clean <- aprender[complete.cases(aprender), ]

# Verificamos si hay valores nulos.
sapply(aprender_clean, function(x) sum(is.na(x)))

summary(aprender_clean)
aprender_clean$mpondera
##### Normalizacion y analisis #####
# Grafico dinamico de boxplot para identificacion de outliers
grafico.boxplot <- plot_ly(y=aprender_clean$ponder, type="box", name="ponder") %>%
  add_trace(y=aprender_clean$lpondera, type="box", name="lpondera") %>%
  add_trace(y=aprender_clean$mpondera, type="box", name="mpondera")

grafico.boxplot

names(aprender_clean)

aprender_clean = subset(aprender_clean, select = c(Ap1, Ap2, Ap3, Ap4, Ap5, Ap6, Ap7,
                                                   Ap8a, Ap8b, Ap8c, Ap9, Ap10, Ap11,
                                                   Ap12, Ap13, Ap14, Ap15, lpuntaje,
                                                   ldesemp, mpuntaje, mdesemp, sector,
                                                   ambito, iclima,autoconl, autoconm
))


# Normalizamos campos.
aprender_clean$Ap1 <- as.factor(mapvalues(aprender_clean$Ap1,
                                          from=c(1,2,3,4,5, -1,-9),
                                          to=c("7 a os o menos", "8 a os","9 a os","10 a os","11 a os o mas","En blanco","No respondio")))

aprender_clean$Ap2 <- as.factor(mapvalues(aprender_clean$Ap2,
                                          from=c(1,2, -1,-9),
                                          to=c("varon", "mujer","En blanco","No respondio")))

```

```r
aprender_clean$Ap3 <- as.factor(mapvalues(aprender_clean$Ap3,
                                          from=c(1,2,3,4, -1,-9),
                                          to=c("antes 4anios", "a 4anios","desde 5anios",
    "no_fui_jardin","blanco","No_respondio")))

aprender_clean$Ap4 <- as.factor(mapvalues(aprender_clean$Ap4,
                                          from=c(1,2,3,4,5, -1,-9),
                                          to=c("nunca", "una vez","dos veces","tres veces
     o mas","no me acuerdo","En blanco","No respondio")))

aprender_clean$Ap5 <- as.factor(mapvalues(aprender_clean$Ap5,
                                          from=c(1,2,3,4, -1,-9),
                                          to=c("muy bien", "bien","mas o menos bien","mal
    ","En blanco","No respondio")))

aprender_clean$Ap6 <- as.factor(mapvalues(aprender_clean$Ap6,
                                          from=c(1,2,3,4, -1,-9),
                                          to=c("muy bien", "bien","mas o menos bien","mal
    ","En blanco","No respondio")))

aprender_clean$Ap7 <- as.factor(mapvalues(aprender_clean$Ap7,
                                          from=c(1,2,3,4, -1,-9),
                                          to=c("muy bien", "bien","mas o menos bien","mal
    ","En blanco","No respondio")))


aprender_clean$Ap8a <- as.factor(mapvalues(aprender_clean$Ap8a,
                                          from=c(1,2,3,-1,-9),
                                          to=c("muchas veces", "algunas veces","casi
    nunca","En blanco","No respondio")))

aprender_clean$Ap8b <- as.factor(mapvalues(aprender_clean$Ap8b,
                                          from=c(1,2,3, -1,-9),
                                          to=c("muchas veces", "algunas veces","casi
    nunca","En blanco","No respondio")))

aprender_clean$Ap8c <- as.factor(mapvalues(aprender_clean$Ap8c,
                                           from=c(1,2,3, -1,-9),
                                           to=c("muchas veces", "algunas veces","casi
    nunca","En blanco","No respondio")))


aprender_clean$Ap9 <- as.factor(mapvalues(aprender_clean$Ap9,
                                          from=c(1,2,3,4, -1,-9),
                                          to=c("con todos", "con la mayoria","con algunos
    ","con ninguno","En blanco","No respondio")))

aprender_clean$Ap10 <- as.factor(mapvalues(aprender_clean$Ap10,
                                          from=c(1,2,3,4, -1,-9),
                                          to=c("alegria", "lo mismo","poca tristeza","
    mucha tristeza","En blanco","No respondio")))


aprender_clean$Ap11 <- as.factor(mapvalues(aprender_clean$Ap11,
                                          from=c(1,2,3, -1,-9),
```

```
                                          to=c("siempre o casi siempre","a veces", "nunca
    o casi nunca","En blanco","No respondio")))

aprender_clean$Ap12 <- as.factor(mapvalues(aprender_clean$Ap12,
                                          from=c(1,2,3, -1,-9),
                                          to=c("siempre o casi siempre","a veces", "nunca
    o casi nunca","En blanco","No respondio")))

aprender_clean$Ap13 <- as.factor(mapvalues(aprender_clean$Ap13,
                                          from=c(1,2,3, -1,-9),
                                          to=c("siempre o casi siempre","a veces", "nunca
    o casi nunca","En blanco","No respondio")))

aprender_clean$Ap14 <- as.factor(mapvalues(aprender_clean$Ap14,
                                           from=c(1,2,3,4, -1,-9),
                                           to=c("muy facil","un poco facil", "un poco
    dificil","dificil","En blanco","No respondio")))

aprender_clean$Ap15 <- as.factor(mapvalues(aprender_clean$Ap15,
                                           from=c(1,2,3,4, -1,-9),
                                           to=c("muy facil","un poco facil", "un poco
    dificil","dificil","En blanco","No respondio")))

aprender_clean$ldesemp <- as.factor(mapvalues(aprender_clean$ldesemp,
                                           from=c(1,2,3,4),
                                           to=c("por debajo del nivel basico","basico", "
    satisfactorio","avanzado")))

aprender_clean$mdesemp <- as.factor(mapvalues(aprender_clean$mdesemp,
                                             from=c(1,2,3,4),
                                             to=c("por debajo del nivel basico","basico"
    , "satisfactorio","avanzado")))

aprender_clean$sector <- as.factor(mapvalues(aprender_clean$sector,
                                             from=c(1,2,3),
                                             to=c("estatal","privada", "sin datos")))

aprender_clean$ambito <- as.factor(mapvalues(aprender_clean$ambito,
                                             from=c(1,2,3),
                                             to=c("urbano","rural", "sin datos")))

aprender_clean$iclima <- as.factor(mapvalues(aprender_clean$iclima,
                                               from=c(1,2,3,-1),
                                               to=c("bajo","medio", "alto","en blanco"))
    )

aprender_clean$autoconl <- as.factor(mapvalues(aprender_clean$autoconl,
                                            from=c(1,2,3,-1),
                                            to=c("bajo","medio", "alto","en blanco")))

aprender_clean$autoconm <- as.factor(mapvalues(aprender_clean$autoconm,
                                              from=c(1,2,3,-1),
```

```
                                                    to=c("bajo","medio", "alto","en blanco")))
aprender_clean %>% distinct(Ambito)

names(aprender_clean)

aprender %>% distinct(ambito)
names(aprender_clean) <- c("anios",
                           "genero",
                           "asistio_jardin",
                           "repitio",
                           "como_lees",
                           "como_escribis",
                           "como_resolv_mat",
                           "interrum_maestros",
                           "maestros_se_enojan",
                           "maestros_explican",
                           "relacion_companeros",
                           "sent_camb_escuela",
                           "contento_escuela",
                           "abuurido",
                           "incomodo",
                           "examen_lengua",
                           "examen_mat",
                           "puntaje_lengua",
                           "desempeno_lengua",
                           "puntaje mat",
                           "desempeno mat",
                           "sector",
                           "ambito",
                           "clima escolar",
                           "autoconcepto lengua",
                           "autoconcepto matematica"
)

attach(aprender_clean)
aprender_clean %>% distinct (sector)
#aprender_clean_cordoba <- subset(aprender_clean, aprender_clean$Provincia == "Cordoba")
#head(aprender_clean_cordoba)

##### Modelado - Arbol de desicion #####
set.seed(1250) # Numero inicial del cual comenzara a generar una secuencia aleatoria.
split_train_test <- createDataPartition(aprender_clean$sector, p=0.7, list=FALSE) #p =
    porcentaje de los datos a usar, list = Si el resultado devolvera una lista o una
    matriz.
dtrain<- aprender_clean[split_train_test,]
dtest<-  aprender_clean[-split_train_test,]

sqldf('
SELECT asistio_jardin, count(*)
FROM dtrain
WHERE 'puntaje lengua' < 463
GROUP BY asistio_jardin
ORDER BY 2 DESC
')

```

```
dtrain$‘puntaje lengua‘

sqldf(’
SELECT DISTINCT repitio
FROM dtrain

’)
#11.853 < 463 --> 33%
#22.299 > 463 --> 66%
#34153

tr_fit <- rpart(sector ~., data = dtrain, method="class") # Indicamos que deseamos un arbol de clasificacion, tambien podemos armar un arbol de regresion.
tr_fit # Nuestro arbol obtenido.
rpart.plot(tr_fit, tweak = 1.6) # Graficamos el arbol.

prp(tr_fit,
    type = 2, #Espefica que el cada nodo quede etiquetado, y que el split quede debajo de cada nodo
    extra = 104, #Muestra la probabilidad de cada clase en el nodo
    nn = TRUE, #Etiqueta el nro de nodo
    fallen.leaves = TRUE, #Muestra los nodos hojas abajo de todo el grafico
    faclen = 10, #Se utiliza para abreviar el nombre de las clases en 4caracteres
    varlen = 10, #Se utiliza para abreviar el nombre de las variables en 8caracteres
    shadow.col = "gray")

#Importancia de las variables.
qplot(x = names(tr_fit$variable.importance), y=tr_fit$variable.importance,
      xlab="Variable", ylab="Importancia", main="rpart - Importancia de las variables")

#
    ##############################################################################################

######################### 05 - TEST Y VALIDACION DEL MODELO
    ################################
#
    ##############################################################################################

#EVALUACION DE CALIDAD DEL MODELO: MATRIZ DE CONFUSION:

#5.1 - CREAMOS EL OBJETO DE PREDICCION SOBRE EL CUAL SE HAR  LA MATRIZ DE CONFUSION
aprender_rpart_pred <- predict(tr_fit, dtest, type="class")
aprender_rpart_pred
#5.2 - MATRIZ DE CONFUSION
tabla_mc_aprender <-confusionMatrix(table(dtest$sector, aprender_rpart_pred,
      dnn = c("Actual", "Predicho")))

tabla_mc_aprender

#5.3 - DIAGRAMA ROC: MIDE EL RENDIMIENTO: NOS DICE QUE TAN BIEN PUEDE CLASIFICAR EL MODELO ENTRES 2 CLASES
#5.3 - SE GENERA EL INDICE DE PROBABILIDAD EN LUGAR DE LAS CANTIDADES DE CLASIFICACIONES
#aprender_rpart_pred2 <- predict(tr_fit, dtest, type = "prob")
#aprender_rpart_pred2
#5.3.1 - CON QUE PROB CLASIFICA?
```

```
#head(aprender_rpart_pred2)

#aprender_rpart_pred2_roc <- prediction(aprender_rpart_pred2[,2], dtest[, "sector"])
#aprender_rpart_pred2_roc_perf <- performance(aprender_rpart_pred2_roc, "tpr", "fpr")
#plot(aprender_rpart_pred2_roc_perf)

aprender_rpart_pred2_roc
aprender_rpart_pred2_roc_perf

#
    ##############################################################################################

################################## PAQUETES NECESARIOS
    ###################################
#
    ##############################################################################################

install.packages("ROCR")

library(TTR)
library(tsibble)
library(readr)
library(sqldf)
library(tidyverse)
library(stringi)
library(stringr)
library(ROCR)
library(plyr)
library(caret)
library(gridExtra)
library(dplyr)
library(tidyr)
library(corrplot)
library(ggplot2)

library(rpart)
library(pROC)
library(MASS)
library(e1071)
library(ggpubr)
library(rsample)
library(e1071)
library(GGally)
library(data.table)
library(DT)
library(ROCR)
library(gplots)
library(randomForest)
library(rpart.plot)
#
    ##############################################################################################

################################ 01 - IMPORTAR LOS DATOS
    ###################################
```

```
#
    ##############################################################################################
emse <- read.csv("./UNO_EDD/TP4_Arboles_Decision/emse_datosabiertos/EMSE_DatosAbiertos.
    csv")
str(emse)
View(emse)
names(emse)

emse_red <- emse[,(4:144)]
names(emse_red)
rm(emse_red)
emse_red = subset(emse_red, select = c(texto_q1,
                                       texto_q2,
                                       texto_q3,
                                       texto_q6,
                                       texto_q15,
                                       texto_q16,
                                       texto_q17,
                                       texto_q18,
                                       texto_q19,
                                       texto_q22,
                                       texto_q23,
                                       texto_q24,
                                       texto_q25,
                                       texto_q26,
                                       texto_q27,
                                       texto_q28,
                                       texto_q29,
                                       texto_q31,
                                       texto_q34,
                                       texto_q35,
                                       texto_q38,
                                       texto_q39,
                                       texto_q40,
                                       texto_q41,
                                       texto_q42,
                                       texto_q43,
                                       texto_q44,
                                       texto_q45,
                                       texto_q46,
                                       texto_q49,
                                       texto_q51,
                                       texto_q53,
                                       texto_q55,
                                       texto_q56,
                                       texto_q57,
                                       texto_q59,
                                       texto_q60,
                                       texto_q61,
                                       texto_q62,
                                       texto_q65,
                                       texto_q66,
                                       texto_q68,
                                       texto_q73,
```

```
                                        texto_q74,
                                        texto_q76,
                                        texto_q79
))
#
    ##############################################################################################

################################## 02 - ETL DEL MODELO
    ####################################
#
    ##############################################################################################

#borrar:
#2.1.1 - RENOMBRAR COLUMNAS
names(emse_red) <- c("edad",
                     "sexo",
                     "grado",
                     "dias_con_hambre",
                     "ult_12meses_atac_fisi",
                     "ult_12meses_part_pelea",
                     "lesion_grave",
                     "lesion_mas_seria",
                     "causa_lesion_seria",
                     "sentirse_solo",
                     "preocupacion",
                     "consideraste_suicidio", #pred
                     "plan_de_suicidio", #pred
                     "veces_intentaste_suici",
                     "amigos_cercanos",
                     "edad_fumador",
                     "ult_30dias_fumaste",
                     "dejaste_fumar",
                     "edad_alcohol",
                     "dias_alcohol",
                     "veces_tomaste_acohol",
                     "problem_entorno_alcohol",
                     "edad_drogas",
                     "veces_maria",
                     "ult_30dias_consu_maria",
                     "veces_consu_metanfet",
                     "tuviste_sexo",
                     "edad_sexo_1ra_vez",
                     "cuantas_perso_tuv_sex",
                     "ult_7dias_activ_fisica",
                     "vas_a_edu_fisica",
                     "ult_30dias_faltaste",
                     "padres_verfic_tarea",
                     "ult_30dias_padres_atend_salud",
                     "ult_30dias_padres_sabian_activi",
                     "educacion_padre",
                     "educacion_madre",
                     "ult_7dias_frutas",
                     "ult_7dias_verd",
                     "comida_grasa",
                     "intimidacion_en_esc",
```

```
                    "intimidacion_en_int",
                    "que_bebida_tomas",
                    "con_quien_tomas",
                    "si_te_ofrecen_tomas",
                    "quedaste_embarazada"
)

#2.1.2 CORRECCION DE CARACTERES ESPECIALES
#CAMBIO DE VALORES POR CONFLICTO EN APLICACION DE FUNCIONES CON LOS MISMOS
names(emse_red)
str(emse_red)
library(stringi)
attach(emse_red)
emse_red$veces_intentaste_suici
emse_red %>% distinct(veces_intentaste_suici)

emse_red$veces_intentaste_suici <- stri_replace_all_regex(emse_red$veces_intentaste_suici
    ,
                                                          pattern=c('     ','    ','    ','=>
    ',    '    ','    ','    ','    ','    ','  ','=<'),
                                                          replacement=c('E','e','a','mayor a
    ','ni','',   'o','u', 'er','i','menor a '),
                                                          vectorize=FALSE)

emse_red$causa_lesion_seria <- stri_replace_all_regex(emse_red$causa_lesion_seria,
                                                          pattern=c('     ','    ','    ','=>
    ',    '    ','    ','    ','    ','    ','  ','=<'),
                                                          replacement=c('E','e','a','mayor a
    ','ni','',   'o','u', 'er','i','menor a '),
                                                          vectorize=FALSE)

emse_red$causa_lesion_seria <- stri_replace_all_regex(emse_red$causa_lesion_seria,
                                                      pattern=c('     ','    ','    ','=>',
        '    ','    ','    ','    ','    ','  ','=<'),
                                                          replacement=c('E','e','a','mayor a ',
    'ni','',   'o','u', 'er','i','menor a '),
                                                          vectorize=FALSE)

emse_red$lesion_mas_seria <- stri_replace_all_regex(emse_red$lesion_mas_seria,
                                                    pattern=c('     ','    ','    ','=>',
    '    ','    ','    ','    ','    ','  ','=<'),
                                                          replacement=c('E','e','a','mayor a ','ni'
    ,'',   'o','u', 'er','i','menor a '),
                                                          vectorize=FALSE)

emse_red$lesion_grave <- stri_replace_all_regex(emse_red$lesion_grave,
                                                pattern=c('     ','    ','    ','=>',    '    ','
        ','    ','    ','    ','  ','=<'),
                                                replacement=c('E','e','a','mayor a ','ni','',   '
    o','u', 'er','i','menor a '),
                                                vectorize=FALSE)
##
emse_red$edad <- stri_replace_all_regex(emse_red$edad,
                                        pattern=c('     ','    ','    ','=>',    '    ','
        ','    ','    ','    ','  ','=<'),
```

```
                                        replacement=c('E','e','a','mayor a ','ni','',   'o','u', 'er','i','menor a '),
                                        vectorize=FALSE)

emse_red$grado <- stri_replace_all_regex(emse_red$grado,
                                         pattern=c('     ','    ','    ','=>',    '    ','    ','    ','    ','    ','  ','=<'),
                                         replacement=c('E','e','a','mayor a ','ni','',   'o','u', 'er','i','menor a '),
                                         vectorize=FALSE)

emse_red$ult_12meses_part_pelea <- stri_replace_all_regex(emse_red$ult_12meses_part_pelea,
                                                          pattern=c('     ','    ','    ','=>',    '    ','    ','    ','    ','    ','  ','=<'),
                                                          replacement=c('E','e','a','mayor a ','ni','',   'o','u', 'er','i','menor a '),
                                                          vectorize=FALSE)

emse_red$ult_12meses_atac_fisi <- stri_replace_all_regex(emse_red$ult_12meses_atac_fisi,
                                         pattern=c('     ','    ','    ','=>',    '    ','    ','    ','    ','    ','  ','=<'),
                                         replacement=c('E','e','a','mayor a ','ni','',   'o','u', 'er','i','menor a '),
                                         vectorize=FALSE)

#2.1.3 - TRATA DE VALORES NAs
View(emse_red)
str(emse_red)

emse_red[emse_red == ""] <- NA
emse_red[emse_red == "Dato perdido"] <- NA

#CONTAR NAs
View(summarise_all(emse_red, funs(sum(is.na(.)))))

#ELIMINAR TODOS LOS NAs
emse_red <- na.omit(emse_red)

View(summarise_all(emse_red, funs(sum(is.na(.)))))

emse_red <- data.frame(lapply(emse_red, as.factor))
str(emse_red)
#
    ##############################################################################################

############################## 03 - DIVIDIMOS EN TRAINEE Y TEST
    ##############################
#
    ##############################################################################################

#ACLARACION: EN RF, NO NECESARIAMENTE DIVIDIR LOS DATOS PARA MODELO, YA QUE EL PROPIO
    MODELO
#YA TIENE INCORPORADO EN EL ALGORITMO LA PARTICION.
set.seed(2018)
```

```r
#plan_de_suicidio
#consideraste_suicidio

split_train_test <- createDataPartition(emse_red$plan_de_suicidio, p=0.7, list=FALSE) #p = porcentaje de los datos a usar, list = Si el resultado devolvera una lista o una matriz.
dtrain <- emse_red[split_train_test,]
dtest <- emse_red[-split_train_test,]

sqldf('
SELECT plan_de_suicidio, count(*)
FROM dtrain
GROUP BY plan_de_suicidio
ORDER BY 2 DESC
')
#     plan_de_suicidio cantidad
#1               No    18787
#2               Si     3519
#
    ##############################################################################################

############################### 04 - CONSTRUCCION DEL MODELO ###############################
#
    ##############################################################################################

names(emse_red)
emse_red_rf <- randomForest(x = dtrain[,-13], #VARIABLES INDEPENDIENTES
                            y = dtrain$plan_de_suicidio, #VARIABLE DEPENDIENTE
                            ntree = 500, #CANTIDAD DE ARBOLES. POR DEFECTO SON 500
                            keep.forest = TRUE) #TRUE = ME DEJA LOS ARBOLES INTERMEDIOS. OJO QUE ES COSTOSO, Y EN LA REALIDAD NO ES RECOMENDABLE

#KEEP.FOREST: POR DEFECTO, EL MODELO NO SE QUEDA CON LOS ARBOLES QUE FUE UTILIZANDO PARA CONSTRUIR EL MODELO.
#POR ENDE DICHOS ARBOLES NO SE USAN PARA PREDECIR SOBRE LOS CUALES SE VA CLASIFICAR. SIN EMBARGO, SE PUEDE FORZAR.

#
    ##############################################################################################

############################### MATRIZ DE CONFUSION ################################
#
    ##############################################################################################

#NOTA: PREDICT CATALOGA/CLASIFICA; PREDICTION HACE UNA PREDICION PORCENTUAL
#4.1 - SE PREDICE LA BONDAD DEL MODELO: SE PREDICE EN BASE AL MODELO ACTUAL, LA CLASIFICACION DE LOS DATOS QUE NO INGRESARON AL MODELO
emse_red_rf_pred <- predict(emse_red_rf, dtest, type = "class")

#4.2 - MATRIZ DE CONFUSION:
emse_red_rf_pred_mc <-confusionMatrix(table(dtest$plan_de_suicidio, emse_red_rf_pred,
                                            dnn = c("Actual", "Predicho")))

emse_red_rf_pred_mc
```

```

#CONCLUSION 1:  Balanced Accuracy : 0.8404  NOS INDICA QUE EL MODELO PRESENTA UN BALANCEO DE CLASES
#EN DONDE LA CLASE A PRESENTA MAYOR CANTIDAD DE CASOS QUE LA OTRA CLASE.

#4.3 - PREDICR DE PROBABILIDADES: SE PREDICE CON QUE PROBABILIDAD EL MODELO PUEDE CLASIFICAR SOBRE LOS
#DATOS DE TEST
#banknote_probs <- predict(banknote_rf, banknote[-training.ids,], type = "prob")
#head(banknote_probs)

#4.4 - ROC: SOBRE LOS EXITOS,
#banknote_probs_pred <- prediction(banknote_probs[,2], banknote[-training.ids,"class"])
#banknote_probs_pred_perf <- performance(banknote_probs_pred, "tpr", "fpr") #TPR: TRUE POSITIVE RATE; FALSE POSITVE RATE
#plot(banknote_probs_pred_perf)

#CONCLUSION: RF ES COSTOSO CONPUTANCIONALMENTE, PERO CLASIFICA MUY BIEN.
```