Modelo Machine Learning

Pedro Burgo

24 de noviembre de 2017

Contents

1.Introducción

Como ejercicio del Módulo 3 del Master de Big Data de Telefónica, vamos a realizar un Análisis explotatorio basado apoyado en algún método no supervisado para posteriormente llevar a cabo la construcción de al menos 2 modelos *machine learning* supervisados y comparar dichos modelos. El dataset utilizado, es el que se puede encontran en la siguiente url: https://archive.ics.uci.edu/ml/machine-learning-databases/00320/student.zip y la variable a predecir la nota final G3

2. Carga de los Datos y Análisis Descriptivo

2.1 Carga del dataset original

Descargamos el archivo, lo descomprimimos y leemos el archivo que nos interesa. Vamos a trabajar únicamente con el dataset de la asignatura de portugués.

```
fileURL <- "https://archive.ics.uci.edu/ml/machine-learning-databases/00320/student.zip"
# download.file(fileURL,destfile = paste0(dirPreData,"/student.zip"))
# unzip(paste0(dirPreData,"/student.zip"), exdir=dirPreData)
conexion <- file(paste0(dirPreData,"/student-por.csv"),"r")
stPor <- read.csv2(conexion, sep = ";",  header = TRUE )
close(conexion)
```

2.2 Inspección Inicial de los datos

Realizamos una inspección inicial de los datos

```
class(stPor)#Comprobamos que se trata de un data.frame
```

```
## [1] "data.frame"
```

```
dim(stPor)#Número de variables y de filas correcto
```

```
## [1] 649  33
```

```
any(is.na.data.frame(stPor)) #Vemos si existen NA en la tabla
```

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

```
  names(stPor) <- tolower(names(stPor))
  names(stPor)
```

```
##  [1] "school"     "sex"        "age"        "address"    "famsize"
##  [6] "pstatus"    "medu"       "fedu"       "mjob"       "fjob"
## [11] "reason"     "guardian"   "traveltime" "studytime"  "failures"
## [16] "schoolsup"  "famsup"     "paid"       "activities" "nursery"
## [21] "higher"     "internet"   "romantic"   "famrel"     "freetime"
## [26] "goout"      "dalc"       "walc"       "health"     "absences"
## [31] "g1"         "g2"         "g3"
```

```
  kable(head(stPor[,1:10]))
```

school	sex	age	address	famsize	pstatus	medu	fedu	mjob	fjob
GP	F	18	U	GT3	A	4	4	at_home	teacher
GP	F	17	U	GT3	T	1	1	at_home	other
GP	F	15	U	LE3	T	1	1	at_home	other
GP	F	15	U	GT3	T	4	2	health	services
GP	F	16	U	GT3	T	3	3	other	other
GP	M	16	U	LE3	T	4	3	services	other

2.3 Analizamos la distribución de algunas variables

Nuestra variable objetivo va a ser G3, la nota final, no tanto su valor como si el alumno aprueba o no. Para ello, en un inicio, vamos a ver la relación que existe entre algunas de las otras variables y G3. Primero vamos a crear una serie de variables para trabajar mejor con el dataset

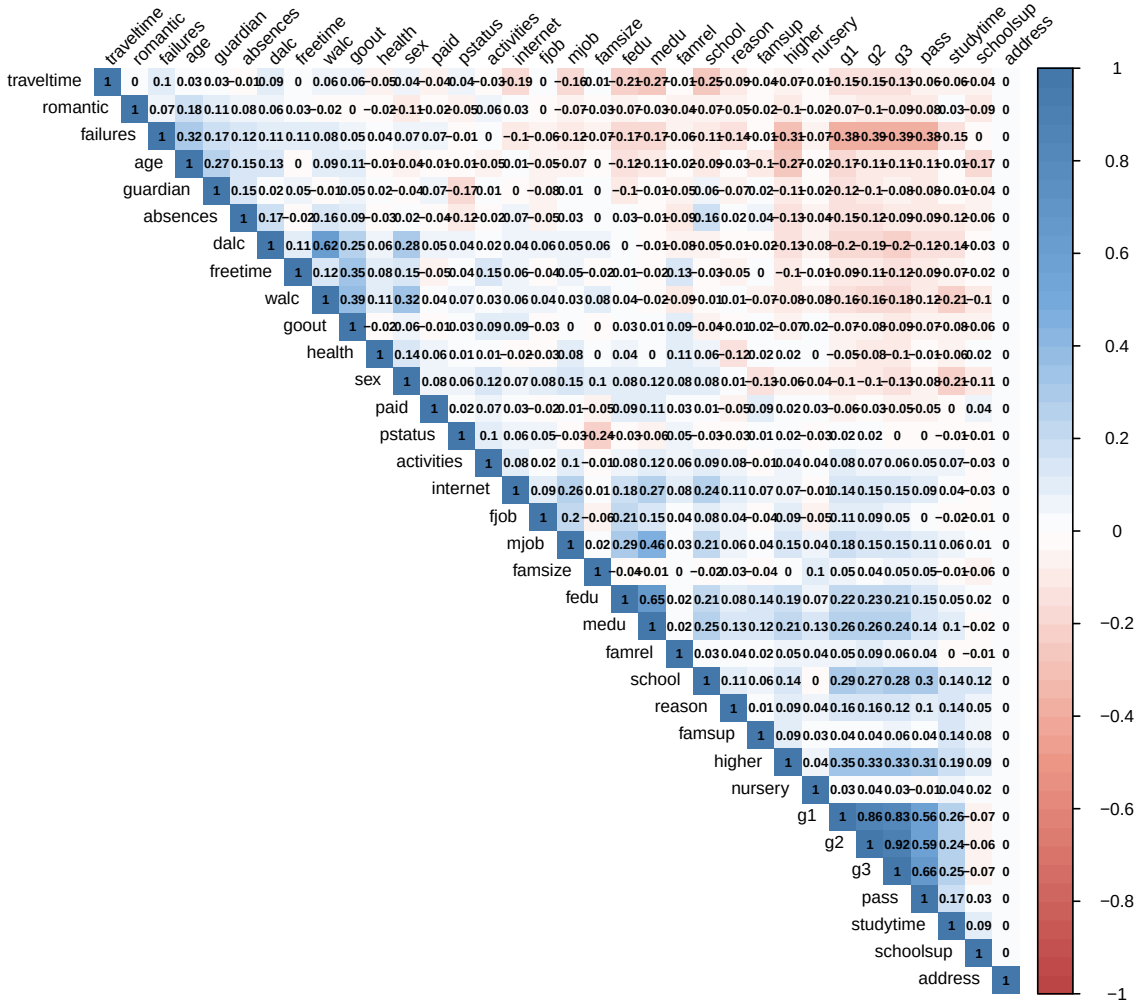
```
  # Creamos una serie de variables para ser usadas posteriormente
  stPor$pass <- ifelse(stPor$g3 > 9, 1, 0)# Consideramos el aprobado a partir del 10 sobre 20
  st <- stPor
  st$sex <- ifelse(st$sex == 'M', 1, 0)
  st$famsize <- ifelse(st$famsize == 'LE3', 1 , 0)
  st$school <- ifelse(st$school == 'GP', 1, 0)
  st$address <- ifelse(st$address == '"U', 1, 0)
  st$pstatus <- ifelse(st$pstatus == 'T',1,0)

  levelsParentsEdu <- c("none", "4th grade", "5th-9th grade", "secondary", "high")
  st$fedu.factor <- factor(sapply(st$fedu,  function(x) {
  x <- car::recode(x,"0='none'; 1='4th grade'; 2='5th-9th grade' ; 3='secondary'; 4='high'"); x}),
        levels = levelsParentsEdu)
  st$medu.factor <- factor(sapply(st$medu,  function(x) {
  x <- car::recode(x,"0='none'; 1='4th grade'; 2='5th-9th grade' ; 3='secondary'; 4='high'"); x}),
        levels = levelsParentsEdu)
  # Sustituimos las columnas con valor yes/no por 1/0
  st <- FromFactorToBinary(st)

  # Convertimos en numéricas las variables que nos faltan
  job.levels <- c('teacher', 'health' , 'services' , 'at_home', 'other')
  st$fjob <- as.integer(st$fjob, levels = job.levels)
  st$mjob <- as.integer(st$mjob, levels = job.levels)

  reason.levels <- c('home', 'reputation', 'course', 'other')
  st$reason <- as.numeric(st$reason, levels = reason.levels)
```

```
guardian.levels <-  c('mother', 'father', 'other')
st$guardian <- as.numeric(st$guardian, levels = guardian.levels)
st <- as.data.frame(st)
```

2.4 Matriz de correlación

Como punto de partida vamos a estudiar la matriz de correlación usando para ello todas las variables numéricas

Fig 2.7.1 Matriz de Correlación

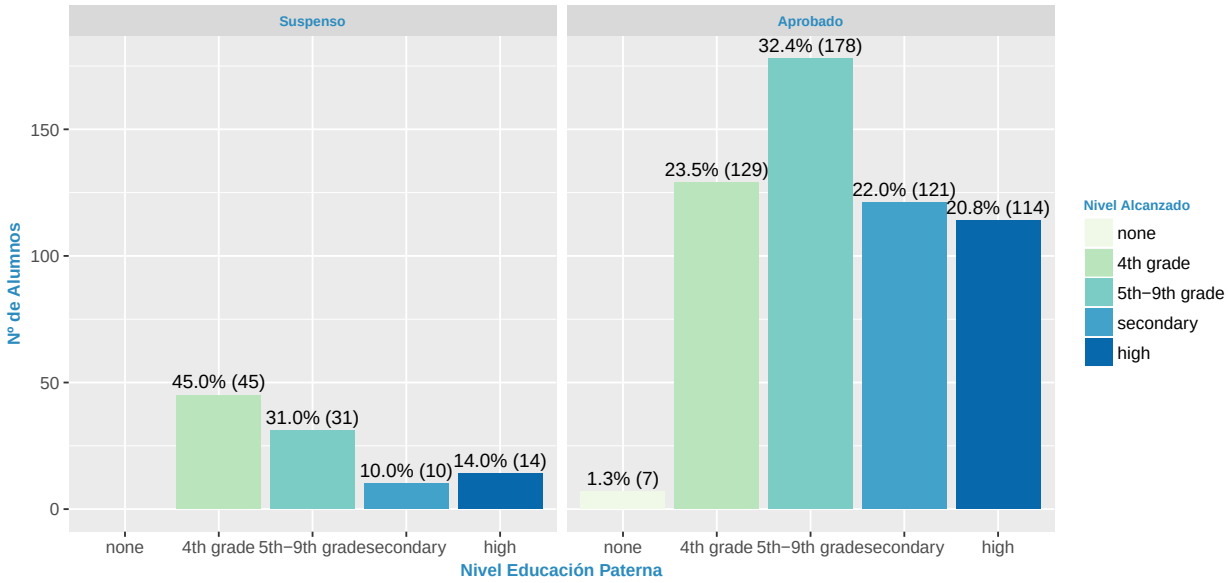


Se puede apreciar un alto nivel de correlación con el nivel educativo de los padres y una correlación negativa significativa con la variable *failures*

2.5 Análisis más detallado de algunas variables

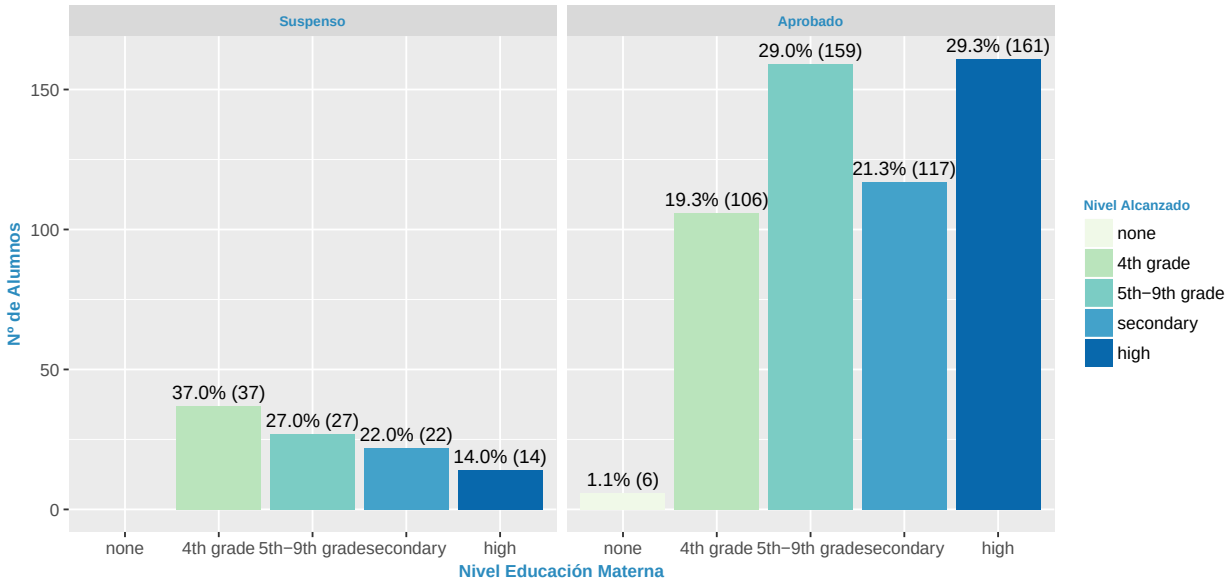
Considerando los resultados de la matriz de correlación, empezamos por observar la distribución del nivel educativo de los padres frente al target con un diagrama de barras. Primero con respecto al padre.

Fig 2.5.1 Diagrama barras educación paterna



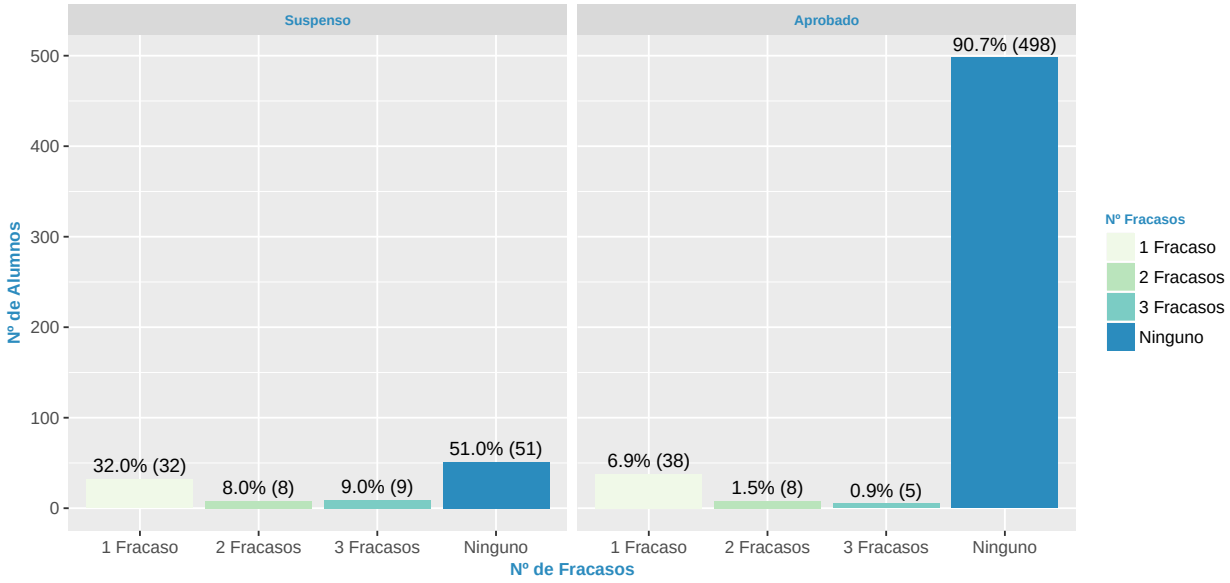
Lo mismo para el nivel educativo de la madre

Fig 2.5.2 Diagrama barras educación materna



Se aprecia en ambos casos, tanto con respecto al padre como a la madre que la proporción de niveles bajos educativos es mayor en el subgrupo de suspensos que en el aprobados. Haremos lo mismo para la variable *failures* (fracasos en anteriores asignaturas) que como hemos visto en la matriz de correlación tiene una correlación negativa.

Fig 2.5.3 Fracaso en pasadas asignaturas



Vemos que la distribución varía en ambas gráficas, la proporción de alumnos con con fracasos en clases anteriores con respecto a su grupo es considerablemente mayor en aquellos que no han aprobado la asignatura.

3. Análisis exploratorio apoyado en algún método NO Supervisado

Aquí nos vamos a centrar en buscar asociaciones entre algunas de las variables de los estudiantes. En concreto trabajaremos con las variables: *medu.factor*,*fedu.factor*,*famsup* y *pass*. Tratamos de ver si existe asociación entre los niveles educativos de los padres, el soporte familiar a la educación del alumno, y si éste aprueba o no. Nos valdremos para ello del algoritmo *apriori*, que se usa para la búsqueda de asociaciones entre cualquier conjunto de items. Se usarán, como decía, los siguientes attibutos: * medu.factor & fedu.factor(sin estudios, educación elemental * educación primaria, educación secundaria, educación superior) * famsup (existencia o ausencia de soporte educacional familiar) * pass(aprobado, suspenso).

Para ello, vamos a definir las reglas de asociación, que están formadas por uno o más antecedentes (lhs) y una consecuencia (rhs). Las reglas de asociación están definidas por su *soporte* (ratio de los casos en los que se dan los antecentes conjuntamente en el *dataset* completo) y su *confianza* (ratio de transacciones que conteniendo los antecedentes también contienen el consecuente). Estudiamos las reglas de asociación estableciendo un soporte del 5% y una confianza del 70%. En este caso en concreto vamos a buscar exclusivamente los antecedentes asociados a la variable *pass*.

```
library(arules)
library(arulesViz)
aso.student <- st %>% select(medu.factor, fedu.factor, famsup, pass)
aso.student <- data.frame(sapply(aso.student, as.factor))

reglas <- apriori(aso.student ,parameter = list( supp = 0.05 , conf = 0.7 , target = "rules"),
                        appearance = list(rhs = c("pass=1", "pass=0"), default = "lhs"))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.7    0.1    1 none FALSE            TRUE       5    0.05      1
##  maxlen target   ext
```

```
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 32
##
## set item appearances ...[2 item(s)] done [0.00s].
## set transactions ...[14 item(s), 649 transaction(s)] done [0.00s].
## sorting and recoding items ... [12 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [37 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
quality(reglas) <- round(quality(reglas), digits = 3)
summary(reglas)
```

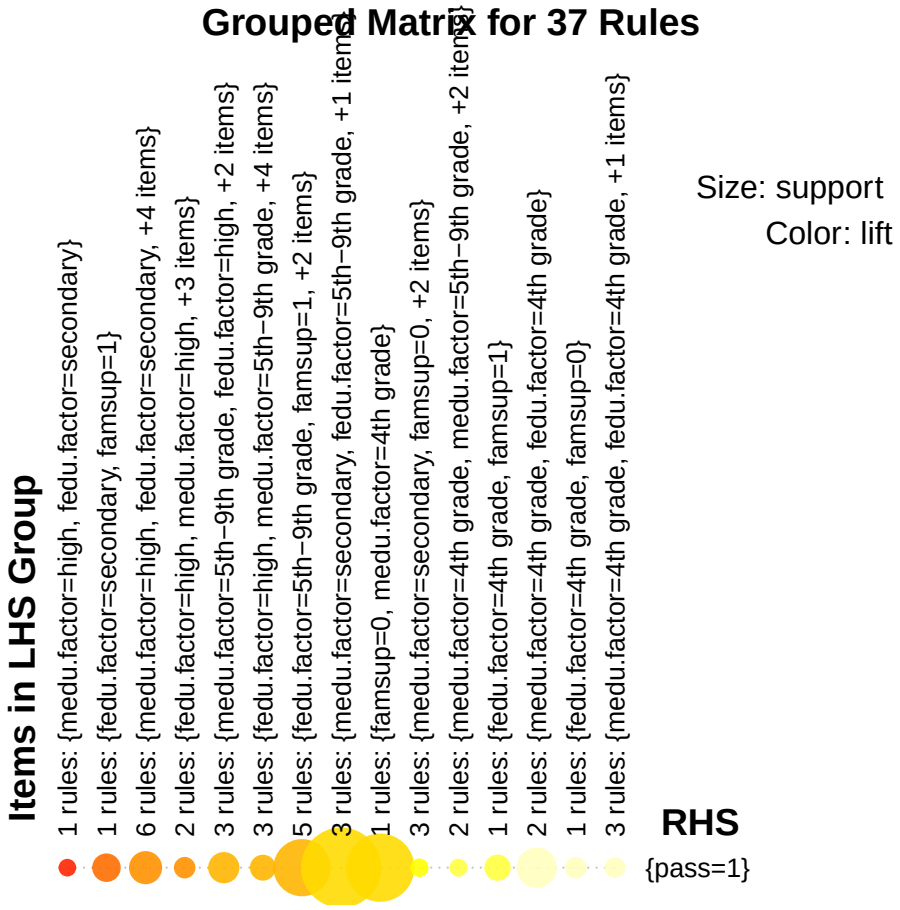
```
## set of 37 rules
##
## rule length distribution (lhs + rhs):sizes
##  1  2  3  4
##  1 10 22  4
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   2.784   3.000   4.000
##
## summary of quality measures:
##     support        confidence          lift           count
##  Min.   :0.052   Min.   :0.7090   Min.   :0.838   Min.   : 34.0
##  1st Qu.:0.077   1st Qu.:0.8000   1st Qu.:0.946   1st Qu.: 50.0
##  Median :0.119   Median :0.8570   Median :1.013   Median : 77.0
##  Mean   :0.160   Mean   :0.8461   Mean   :1.000   Mean   :103.8
##  3rd Qu.:0.176   3rd Qu.:0.9010   3rd Qu.:1.066   3rd Qu.:114.0
##  Max.   :0.846   Max.   :1.0000   Max.   :1.182   Max.   :549.0
##
## mining info:
##         data ntransactions support confidence
##  aso.student           649    0.05        0.7
```

```
inspect(sort(reglas, by = "lift")[1:10])
```

```
##      lhs                          rhs      support confidence  lift count
## [1]  {medu.factor=high,
##       fedu.factor=secondary} => {pass=1}   0.068      1.000 1.182    44
## [2]  {fedu.factor=secondary,
##       famsup=1}              => {pass=1}   0.122      0.952 1.125    79
## [3]  {medu.factor=high,
##       famsup=0}              => {pass=1}   0.076      0.925 1.093    49
## [4]  {fedu.factor=secondary} => {pass=1}   0.186      0.924 1.092   121
## [5]  {medu.factor=secondary,
##       fedu.factor=secondary,
##       famsup=1}              => {pass=1}   0.054      0.921 1.089    35
## [6]  {medu.factor=high}      => {pass=1}   0.248      0.920 1.088   161
```

```
## [7]  {medu.factor=high,
##       famsup=1}              => {pass=1}   0.173      0.918 1.085   112
## [8]  {medu.factor=high,
##       fedu.factor=high}      => {pass=1}   0.134      0.916 1.083    87
## [9]  {medu.factor=secondary,
##       fedu.factor=secondary} => {pass=1}   0.076      0.907 1.073    49
## [10] {medu.factor=high,
##       fedu.factor=high,
##       famsup=1}              => {pass=1}   0.099      0.901 1.066    64
```

```
plot(reglas, method = "grouped", control = list(k = 15, col = heat.colors(10)))
```



Se observa un clara asociación entre los niveles educativos altos de los padres y la consecución del aprobado. Esta relación ya asomaba en las distribuciones mostradas en el apartado anterior.

4. Construcción de 2 o más modelos de *Machine Learning* supervisados

4.1 Selección de variables

Vamos a usar todas las siguientes variables numéricas del dataset: *age,medu,fedu,traveltime,studytime,failures,famrel,freetime,goo*
pass

```
  st <- st %>%  select(age,medu,fedu,traveltime,studytime,failures,famrel,freetime,goout,dalc,
                       walc,health,absences,g1,g2,g3, pass)
  st$pass <- ifelse(st$pass == '1', 'A', 'S')
  st$pass <- factor(st$pass,levels = c('A', 'S') )
```

Creamos las particiones de los dataset de entrenamiento y test con la libreria *caret*, y comprobamos que las proporciones se mantienen en ambas particiones y similares a las del dataset original

```
## Creamos las particiones
library(mlbench)
library(caret)
library(ROCR)
set.seed(1973)
index.st <- createDataPartition(st$pass, p = 0.8, list = FALSE)
train.st <- st[index.st,]
test.st <- st[ -index.st, ]
# Train
prop.table(table(train.st$pass))
```

```
##
##         A         S
## 0.8461538 0.1538462
```

```
#Test
prop.table(table(test.st$pass))
```

```
##
##         A         S
## 0.8449612 0.1550388
```

Buscamos aquellas variables con una varianza cercana a cero para ser eliminadas.

```
## character(0)
```

Ahora buscamos las variables fuertemente correladas también para eliminarlas.

```
## [1] "g2" "g3"
```

Obtenemos las variables *g2* y *g3* El número de variables predictoras se reduce únicamente en 2 Centramos y escalamos las variables para reducir la desviación con la función preProcess()

```
xTrans.st <- preProcess(cor.train.st[, -dim(cor.train.st)[2]])
train.data <- predict( xTrans.st, cor.train.st[,-dim(cor.train.st)[2]])
train.data$pass <- cor.train.st$pass

test.data <- predict( xTrans.st, cor.test.st[,-dim(cor.test.st)[2]])
test.data$pass <- cor.test.st$pass
```

4.2 Construcción de modelos

Vamos a construir varios modelos para compararlos posteriormente

4.2.1 Classification and Regression Trees (CART)

```
library(pROC)
# Utilizamos validación cruzada como técnica de remuestreo
control <- trainControl(method = "repeatedcv", repeats = 10, classProbs = TRUE,
```

```
                        summaryFunction = twoClassSummary)

set.seed(1973)
#Creamos el modelo
cart.model <- train(pass ~ ., data = train.data, method = "rpart", metric = "ROC",
                    trControl = control, tuneLength = 5)
pred.cart.model <- as.vector(predict(cart.model, newdata = test.data, type = "prob")[,"A"])

#Calculamos el AUC
roc.cart.model <- pROC::roc(test.data$pass, pred.cart.model)
auc.cart.model <- pROC::auc(roc.cart.model)
```

4.2.2 Regresión Lineal Penalizada (ELASTIC NET)

```
set.seed(1973)
#Creamos el modelo
elastic.model <- train(pass ~ ., data = train.data, method = "glmnet", metric = "ROC",
                       trControl = control, family = "binomial", tuneLength = 5)
pred.elastic.model <- as.vector(predict(elastic.model, newdata = test.data, type = "prob")[,"A"])
#Calculamos el AUC
roc.elastic.model <- pROC::roc(test.data$pass, pred.elastic.model)
auc.elastic.model <- pROC::auc(roc.elastic.model)
```

4.2.3 Regresión Lineal Binaria clásica

```
set.seed(1973)
#Creamos el modelo
glm.model <- train(pass ~ ., data = train.data, method = "glm", metric = "ROC",
                       trControl = control, family = "binomial", tuneLength = 5)
pred.glm.model <- as.vector(predict(glm.model, newdata = test.data, type = "prob")[,"A"])
#Calculamos el AUC
roc.glm.model <- pROC::roc(test.data$pass, pred.glm.model)
auc.glm.model <- pROC::auc(roc.glm.model)
```

4.2.4 Gradient Boosted Machines (GBM)

```
set.seed(1973)
#Creamos el modelo
gbm.model <- train(pass ~ ., data = train.data, method = "gbm", metric = "ROC",
                   trControl = control, verbose = FALSE, tuneLength = 5)
pred.gbm.model <- as.vector(predict(gbm.model, newdata = test.data, type = "prob")[,"A"])
#Calculamos el AUC
roc.gbm.model <- pROC::roc(test.data$pass, pred.gbm.model)
auc.gbm.model <- pROC::auc(roc.gbm.model)
```

4.2.5 Random Forest

```
set.seed(1973)
#Creamos el modelo
randomf.model <- train(pass ~ ., data = train.data, method = "rf", metric = "ROC",
                       trControl = control, verbose = FALSE, tuneLength = 5)
pred.randomf.model <- as.vector(predict(randomf.model, newdata = test.data, type = "prob")[,"A"])
```

```
#Calculamos el AUC
roc.randomf.model <- pROC::roc(test.data$pass, pred.randomf.model)
auc.randomf.model <- pROC::auc(roc.randomf.model)
```

5. Comparación de modelos

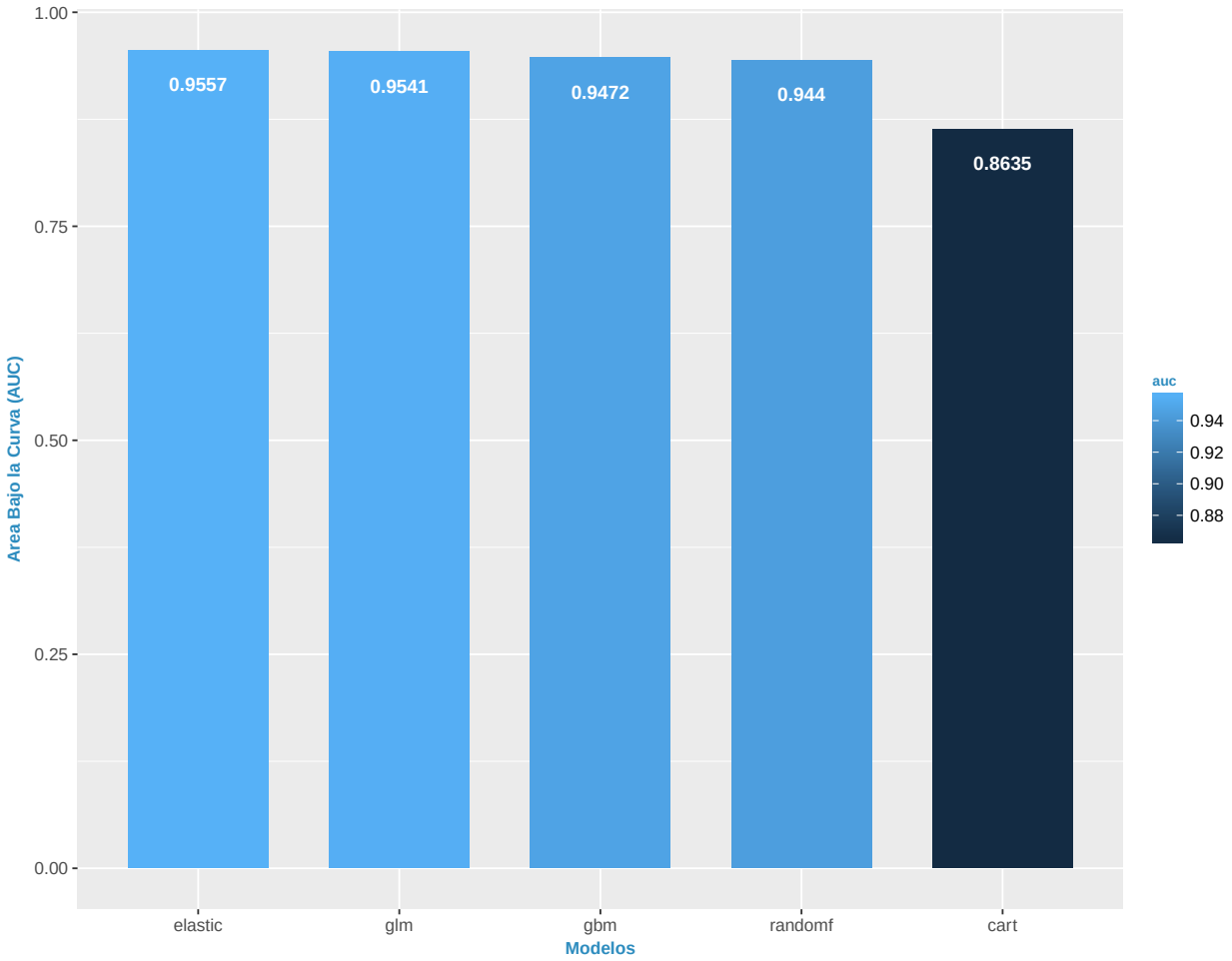
Creamos un *dataframe* con los valores de las *auc* obtenidas.

```
  eva.auc <- data.frame(model = c("cart", "elastic","glm", "gbm","randomf" ),
                        auc = c(auc.cart.model, auc.elastic.model, auc.glm.model, auc.gbm.model,
                                auc.randomf.model))

eva.auc <- eva.auc[order(eva.auc$auc, decreasing = TRUE),]#Ordenamos de mayor a menor
eva.auc$auc <- round(eva.auc$auc, digits = 4)#Redondeamos a 4 dígitos
eva.auc$model <- factor(eva.auc$model, levels = eva.auc$model)
eva.auc
```

```
##     model    auc
## 2 elastic 0.9557
## 3     glm 0.9541
## 4     gbm 0.9472
## 5 randomf 0.9440
## 1    cart 0.8635
```

En el siguiente gráfico observamos la AUC que obtenemos de cada uno de los modelos.

Fig 5.1 Comparación de modelos



Se aprecia que el modelo *elastic net* es el que presenta una mayor AUC, aunque no con una gran ventaja frente a los modelos que le siguen.