Proyecto Final

Juan Alberto Martinez Lopez / Alberto Armijo Ruiz

27 de mayo de 2017

```
#Librerías utilizadas.
library("caret")
```

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

```
## Loading required package: ggplot2
```

```
library("leaps")
library("e1071")
library("ROCR")
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
library("randomForest")
```

```
## randomForest 4.6-12
```

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

```
##
## Attaching package: 'randomForest'
```

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

```
library("neuralnet")
```

```
##
## Attaching package: 'neuralnet'
```

```
## The following object is masked from 'package:ROCR':
##
##     prediction
```

```
library("ada")
```

```
## Loading required package: rpart
```

```
set.seed(1)
```

1. default of credit card clients Data Set (Clasificación)

Introducción.

La base de datos se centra en el caso de los pagos por defecto de los clientes en Taiwán y compara la probabilidad de que un usuario pague o no pague según datos de pagos anteriores. Desde la perspectiva de la gestión de riesgos, la peor predicción erronea es considerar que un cliente pagará cuando es falso. Las caracteristicas son:

Limit_bat: Cantidad de credito bancario dado(en dolares), incluye el credito individual y de su familia. Sex: genero (0 = mujer, 1 = hombre). Education: Educación recivida en 4 variables: Others: Otros estudios. University: Estudios universitarios. High school: Estudios preparatoria. school: Estudios básicos. Marriage: Estado marital en 3 variables: Married: Casado. Single: soltero. Others: otros. Age: Edad. Pay_1-6: Historia del pago anterior. Cada variable es un mes distinto. La escala de medición para el estado de pago es: -1 = pagar debidamente; 1 = retraso de pago de un mes; 2 = retardo de pago por dos meses... Bill_Amt1-6:Estado de la cuenta actual. Pay_Amt1-6: Pago realizado en ese mes. default payment next month: Variable que aprendemos, basamos si la persona pagará o no el siguiente mes.

Nuestro objetivo será intentar conseguir una predicción lo mejor posibles, intentando minimizar los falsos positivos (predecimos que un cliente pagará, pero no este no paga).

Lectura de la base de datos.

Para la lectura de la base de datos, hemos tenido que trasformar el archivo a .csv y además borrar la primera fila por problemas de lectura.

```
credit_card =  read.csv("default_of_credict_card_clients.csv"
                        , sep=",", header = TRUE, row.names =1)
attach(credit_card)
summary(credit_card)
```

```
##    LIMIT_BAL          SEX            EDUCATION        MARRIAGE
##  Min.   :  10000   Min.   :1.000   Min.   :0.000   Min.   :0.000
##  1st Qu.:  50000   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000
##  Median : 140000   Median :2.000   Median :2.000   Median :2.000
##  Mean   : 167484   Mean   :1.604   Mean   :1.853   Mean   :1.552
##  3rd Qu.: 240000   3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:2.000
##  Max.   :1000000   Max.   :2.000   Max.   :6.000   Max.   :3.000
##       AGE            PAY_0             PAY_2             PAY_3
##  Min.   :21.00   Min.   :-2.0000   Min.   :-2.0000   Min.   :-2.0000
##  1st Qu.:28.00   1st Qu.:-1.0000   1st Qu.:-1.0000   1st Qu.:-1.0000
##  Median :34.00   Median : 0.0000   Median : 0.0000   Median : 0.0000
##  Mean   :35.49   Mean   :-0.0167   Mean   :-0.1338   Mean   :-0.1662
##  3rd Qu.:41.00   3rd Qu.: 0.0000   3rd Qu.: 0.0000   3rd Qu.: 0.0000
##  Max.   :79.00   Max.   : 8.0000   Max.   : 8.0000   Max.   : 8.0000
##      PAY_4             PAY_5             PAY_6           BILL_AMT1
##  Min.   :-2.0000   Min.   :-2.0000   Min.   :-2.0000   Min.   :-165580
##  1st Qu.:-1.0000   1st Qu.:-1.0000   1st Qu.:-1.0000   1st Qu.:   3559
##  Median : 0.0000   Median : 0.0000   Median : 0.0000   Median :  22382
##  Mean   :-0.2207   Mean   :-0.2662   Mean   :-0.2911   Mean   :  51223
##  3rd Qu.: 0.0000   3rd Qu.: 0.0000   3rd Qu.: 0.0000   3rd Qu.:  67091
##  Max.   : 8.0000   Max.   : 8.0000   Max.   : 8.0000   Max.   : 964511
##    BILL_AMT2         BILL_AMT3         BILL_AMT4         BILL_AMT5
##  Min.   :-69777   Min.   :-157264   Min.   :-170000   Min.   :-81334
```

```
##  1st Qu.:  2985   1st Qu.:   2666   1st Qu.:   2327   1st Qu.:  1763
##  Median : 21200   Median :  20089   Median :  19052   Median : 18105
##  Mean   : 49179   Mean   :  47013   Mean   :  43263   Mean   : 40311
##  3rd Qu.: 64006   3rd Qu.:  60165   3rd Qu.:  54506   3rd Qu.: 50191
##  Max.   :983931   Max.   :1664089   Max.   : 891586   Max.   :927171
##    BILL_AMT6          PAY_AMT1         PAY_AMT2          PAY_AMT3
##  Min.   :-339603   Min.   :     0   Min.   :      0   Min.   :     0
##  1st Qu.:   1256   1st Qu.:  1000   1st Qu.:    833   1st Qu.:   390
##  Median :  17071   Median :  2100   Median :   2009   Median :  1800
##  Mean   :  38872   Mean   :  5664   Mean   :   5921   Mean   :  5226
##  3rd Qu.:  49198   3rd Qu.:  5006   3rd Qu.:   5000   3rd Qu.:  4505
##  Max.   : 961664   Max.   :873552   Max.   :1684259   Max.   :896040
##     PAY_AMT4          PAY_AMT5          PAY_AMT6
##  Min.   :     0   Min.   :     0.0   Min.   :     0.0
##  1st Qu.:   296   1st Qu.:   252.5   1st Qu.:   117.8
##  Median :  1500   Median :  1500.0   Median :  1500.0
##  Mean   :  4826   Mean   :  4799.4   Mean   :  5215.5
##  3rd Qu.:  4013   3rd Qu.:  4031.5   3rd Qu.:  4000.0
##  Max.   :621000   Max.   :426529.0   Max.   :528666.0
##  default.payment.next.month
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.2212
##  3rd Qu.:0.0000
##  Max.   :1.0000
```

Creación de los conjuntos de training y test.

Para el training, utilizaremos el 70% de los datos, y dejaremos el 30% restante para el test.

```
set.seed(1)
train = sample (nrow(credit_card), round(nrow(credit_card)*0.7))
credit_card.train = credit_card[train,]
credit_card.test = credit_card[-train,]
```

2. Preprocesado de los datos.

Lo primero que queremos hacer es comprobar si hay datos perdidos, y si es así; reemplazaremos el valor perdido.

```
anyNA(credit_card.train)
```

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

Como no tenemos ningún dato pérdido, no tendremos que reemplazar los valores. Si hubieramos tenido valores tenido, podríamos haber utilizado la función *knnImputation* para reemplazar los valores perdidos por los k vecinos más cercanos (por defecto la función utiliza k=3). También podríamos utilizar la media como sustituto de un valor pérdido.

Lo siguiente que vamos a hacer es modificar aquellas columnas que separan los datos en "clases" como por ejemplo la columna *EDUCATION*, que indica que tipo de estudios tiene cada persona. Por cada tipo en los que los separe, crearemos una nueva columna que indique con 0s y 1s la pertenencia a ese tipo. También tenemos que realizar este proceso con la columna *MARRIAGE*

```r
# Modificamos la columna 2, llamada sex, para dividir los datos en 0=mujer, 1=hombre.
credit_card.train$SEX = ifelse(credit_card.train$SEX == 2, 0, 1)
```

```r
# También tenemos que modificar la columna EDUCATION, la dividiremos en cuatro columnas diferentes:
# ed.other, ed.university, ed.high_school, ed.school
ed.other = ifelse(credit_card.train$EDUCATION == 4, 1,0)
ed.university = ifelse(credit_card.train$EDUCATION == 2, 1, 0)
ed.high_school = ifelse(credit_card.train$EDUCATION == 3 | credit_card.train$EDUCATION == 2, 1, 0)
ed.school = ifelse(credit_card.train$EDUCATION == 1 | credit_card.train$EDUCATION == 2 | credit_card.tra

credit_card.train = cbind(credit_card.train,ed.other, ed.high_school, ed.school, ed.university)

# Borramos la columna EDUCATION.
credit_card.train = credit_card.train[,-which(colnames(credit_card.train) == "EDUCATION")]
```

```r
# También tenemos que modificar la columna mariage. Introduciremos tres nueva columnas: marriage.marrie
marriage.married = ifelse(credit_card.train$MARRIAGE == 1, 1,0)
marriage.single = ifelse(credit_card.train$MARRIAGE == 2, 1,0)
marriage.others = ifelse(credit_card.train$MARRIAGE == 3, 1,0)

# Introducimos los datos.
credit_card.train = cbind(credit_card.train, marriage.married, marriage.single, marriage.others)

# Borramos la variable MARRIAGE.
credit_card.train = credit_card.train[, -which(colnames(credit_card.train) == "MARRIAGE")]
```

También vamos a cambiar el nombre de la columna *PAY0* por *PAY1*.

```r
colnames(credit_card.train)[which(colnames(credit_card.train)=="PAY_0")]="PAY_1"
```

Por último, utilizaremos la función preProcess para aplicar la transformación BoxCox a las variables que tengan una varianza elevada.

```r
# Por Último, utilizamos la función preProcess.
trans = preProcess(credit_card.train, c("BoxCox") )
trainTransformado = predict(trans, credit_card.train)
summary(trainTransformado)
```

```
##    LIMIT_BAL          SEX              AGE            PAY_1
##  Min.   : 49.50   Min.   :0.0000   Min.   :1.564   Min.   :-2.00000
##  1st Qu.: 82.29   1st Qu.:0.0000   1st Qu.:1.622   1st Qu.:-1.00000
##  Median :113.27   Median :0.0000   Median :1.657   Median : 0.00000
##  Mean   :111.23   Mean   :0.3974   Mean   :1.656   Mean   :-0.01524
##  3rd Qu.:133.74   3rd Qu.:1.0000   3rd Qu.:1.688   3rd Qu.: 0.00000
##  Max.   :206.99   Max.   :1.0000   Max.   :1.775   Max.   : 8.00000
##      PAY_2             PAY_3             PAY_4            PAY_5
##  Min.   :-2.0000   Min.   :-2.0000   Min.   :-2.000   Min.   :-2.0000
##  1st Qu.:-1.0000   1st Qu.:-1.0000   1st Qu.:-1.000   1st Qu.:-1.0000
##  Median : 0.0000   Median : 0.0000   Median : 0.000   Median : 0.0000
##  Mean   :-0.1321   Mean   :-0.1697   Mean   :-0.221   Mean   :-0.2678
##  3rd Qu.: 0.0000   3rd Qu.: 0.0000   3rd Qu.: 0.000   3rd Qu.: 0.0000
##  Max.   : 8.0000   Max.   : 8.0000   Max.   : 8.000   Max.   : 8.0000
##      PAY_6          BILL_AMT1         BILL_AMT2        BILL_AMT3
##  Min.   :-2.000   Min.   :-154973   Min.   :-69777   Min.   : -46127
##  1st Qu.:-1.000   1st Qu.:   3508   1st Qu.:  2956   1st Qu.:   2552
##  Median : 0.000   Median :  22143   Median : 20895   Median :  20012
```

```
##  Mean   :-0.289   Mean   :  51238   Mean   : 49092   Mean   :  47104
##  3rd Qu.: 0.000   3rd Qu.:  66175   3rd Qu.: 62795   3rd Qu.:  59674
##  Max.   : 8.000   Max.   : 964511   Max.   :983931   Max.   :1664089
##    BILL_AMT4         BILL_AMT5         BILL_AMT6          PAY_AMT1
##  Min.   :-170000   Min.   :-81334   Min.   :-339603   Min.   :     0
##  1st Qu.:   2276   1st Qu.:  1730   1st Qu.:   1261   1st Qu.:  1000
##  Median :  19044   Median : 18093   Median :  17036   Median :  2100
##  Mean   :  43452   Mean   : 40395   Mean   :  38948   Mean   :  5608
##  3rd Qu.:  53536   3rd Qu.: 49719   3rd Qu.:  48740   3rd Qu.:  5006
##  Max.   : 891586   Max.   :927171   Max.   : 961664   Max.   :493358
##     PAY_AMT2           PAY_AMT3          PAY_AMT4          PAY_AMT5
##  Min.   :      0.0   Min.   :     0   Min.   :     0   Min.   :     0.0
##  1st Qu.:    809.8   1st Qu.:   390   1st Qu.:   300   1st Qu.:   269.8
##  Median :   2000.0   Median :  1794   Median :  1500   Median :  1500.0
##  Mean   :   5988.9   Mean   :  5218   Mean   :  4793   Mean   :  4838.0
##  3rd Qu.:   5000.0   3rd Qu.:  4525   3rd Qu.:  4010   3rd Qu.:  4037.8
##  Max.   :1684259.0   Max.   :896040   Max.   :528897   Max.   :426529.0
##     PAY_AMT6      default.payment.next.month    ed.other
##  Min.   :     0   Min.   :0.0000             Min.   :0.000000
##  1st Qu.:   100   1st Qu.:0.0000             1st Qu.:0.000000
##  Median :  1500   Median :0.0000             Median :0.000000
##  Mean   :  5252   Mean   :0.2209             Mean   :0.003857
##  3rd Qu.:  4000   3rd Qu.:0.0000             3rd Qu.:0.000000
##  Max.   :528666   Max.   :1.0000             Max.   :1.000000
##  ed.high_school     ed.school      ed.university    marriage.married
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.000
##  1st Qu.:0.0000   1st Qu.:1.0000   1st Qu.:0.0000   1st Qu.:0.000
##  Median :1.0000   Median :1.0000   Median :0.0000   Median :0.000
##  Mean   :0.6332   Mean   :0.9844   Mean   :0.4708   Mean   :0.458
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.000
##  marriage.single  marriage.others
##  Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.0000   1st Qu.:0.00000
##  Median :1.0000   Median :0.00000
##  Mean   :0.5296   Mean   :0.01076
##  3rd Qu.:1.0000   3rd Qu.:0.00000
##  Max.   :1.0000   Max.   :1.00000
```

Para no tener que repetir este mismo proceso cuando vayamos a calcular el error fuera de la muestra de entrenamiento, vamos a crear una función que se encargue directamente de ello.

```
# Para hacer más sencillo hacer las transformaciones al conjunto de test, se crearán funciones para rea

# Función para reemplazar las columnas.
reemplazarCol = function(x){
  # Columna SEX
  x$SEX = ifelse(x$SEX == 2, 0, 1)

  # Columna EDUCATION.
  ed.other = ifelse(x$EDUCATION == 4, 1,0)
  ed.university = ifelse(x$EDUCATION == 2, 1, 0)
  ed.high_school = ifelse(x$EDUCATION == 3 | x$EDUCATION == 2, 1, 0)
  ed.school = ifelse(x$EDUCATION == 1 | x$EDUCATION == 2 | x$EDUCATION == 3, 1, 0)
  x = cbind(x,ed.other, ed.high_school, ed.school, ed.university)
```

```
  # Borramos la columna EDUCATION.
  x = x[,-which(colnames(x) == "EDUCATION")]

  # Columna MARRIAGE.
  marriage.married = ifelse(x$MARRIAGE == 1, 1,0)
  marriage.single = ifelse(x$MARRIAGE == 2, 1,0)
  marriage.others = ifelse(x$MARRIAGE == 3, 1,0)

  # Introducimos los datos.
  x = cbind(x, marriage.married, marriage.single, marriage.others)

  # Borramos la variable MARRIAGE.
  x = x[, -which(colnames(x) == "MARRIAGE")]

  # Cambiamos el nombre de la variables PAY_0.
  colnames(x)[which(colnames(x)=="PAY_0")]="PAY_1"
  x
}

# Función que realiza la transformación BoxCox.
preprocesar = function(x,pred=trans){
  transTest = predict(pred, x)
  transTest
}

# Función que engloba las funciones anteriores.
prepareTest = function(x){
  tr = reemplazarCol(x)
  tr= preprocesar(x)
  tr
}
```

Por último, vamos a utilizar el filtro PCA para comprobar si todos los datos son relevantes. Si encontramos algún datos redundante , lo eliminaremos de nuestro conjunto de datos. Para saber si un dato es redundante, debemos comprobar si todos los atributos PC (PC1, PC2, ..., PCx) son 0 o muy cercanos a 0; si encontramos algún atributo PC que se aleje de 0, no podemos asegurar que el atributos sea redundante, y por lo tanto no lo podremos quitar.

```
pcaTransformation = prcomp(trainTransformado[,-default.payment.next.month], center=F, scale=F)
pcaTransformation$rotation
```

```
##                                  PC1           PC2           PC3
## SEX                    -1.246341e-06 -4.904872e-07  6.438891e-07
## AGE                    -4.969286e-06 -4.554302e-07  2.875395e-06
## PAY_1                  -8.402099e-07 -1.174667e-06 -3.702498e-06
## PAY_2                  -7.831655e-07 -1.423524e-06 -4.412710e-06
## PAY_3                  -6.673071e-07 -4.589883e-07 -3.545257e-06
## PAY_4                  -5.207917e-07  1.417210e-06 -1.570775e-06
## PAY_5                  -4.037429e-07  2.881611e-06 -2.724843e-06
## PAY_6                  -3.766899e-07  3.805578e-06 -4.208659e-06
## BILL_AMT1              -4.504475e-01 -5.518948e-01 -1.939668e-01
## BILL_AMT2              -4.392710e-01 -4.001284e-01 -5.929929e-02
## BILL_AMT3              -4.267653e-01 -3.653856e-02  5.589409e-01
## BILL_AMT4              -3.965494e-01  2.495663e-01  7.254672e-02
```

```
## BILL_AMT5                  -3.674088e-01  4.259793e-01 -2.116082e-01
## BILL_AMT6                  -3.519808e-01  4.823144e-01 -3.119262e-01
## PAY_AMT1                   -3.243380e-02  2.060446e-02  1.527775e-01
## PAY_AMT2                   -3.775548e-02  1.711008e-01  6.730651e-01
## PAY_AMT3                   -3.242933e-02  1.224791e-01 -3.265661e-03
## PAY_AMT4                   -2.677187e-02  9.158369e-02 -5.831541e-02
## PAY_AMT5                   -2.730621e-02  6.424338e-02  3.452144e-02
## PAY_AMT6                   -2.977955e-02 -1.973644e-02  1.347066e-01
## default.payment.next.month -6.351776e-07 -4.214420e-08 -3.104142e-07
## ed.other                   -9.624187e-09 -2.745489e-08  2.474927e-08
## ed.high_school             -1.896296e-06 -6.830605e-07  4.915918e-07
## ed.school                  -2.934032e-06 -1.293661e-07  1.589245e-06
## ed.university              -1.456700e-06 -4.301804e-07  2.616470e-07
## marriage.married           -1.425640e-06 -7.724235e-08  9.020876e-07
## marriage.single            -1.540599e-06 -1.426100e-07  8.016949e-07
## marriage.others            -2.413312e-08 -4.908935e-08  2.310480e-08
##                                      PC4           PC5           PC6
## SEX                        -3.566762e-06 -1.520925e-06  5.286491e-07
## AGE                        -1.461946e-05 -5.976859e-06  3.173099e-06
## PAY_1                       8.990123e-06  5.326504e-06 -2.426989e-06
## PAY_2                       1.002619e-05  6.261145e-06 -3.382277e-06
## PAY_3                       1.159672e-05  5.567521e-06  2.979077e-06
## PAY_4                       1.226031e-05  6.646866e-06  4.967272e-07
## PAY_5                       1.250268e-05  3.280485e-06 -1.587765e-06
## PAY_6                       1.088009e-05  5.014483e-06 -8.464995e-07
## BILL_AMT1                  -2.931502e-01 -3.297459e-03 -4.848133e-01
## BILL_AMT2                   3.499226e-02 -2.453157e-02  5.131682e-01
## BILL_AMT3                   2.082730e-01  2.751252e-01  5.838854e-02
## BILL_AMT4                   3.800910e-01 -4.508734e-01 -1.622145e-01
## BILL_AMT5                   6.119784e-02 -9.923400e-02 -6.073123e-03
## BILL_AMT6                  -2.494273e-01  3.956564e-01  6.400747e-02
## PAY_AMT1                   -1.580537e-01 -1.890328e-01  5.668722e-01
## PAY_AMT2                   -2.936760e-01  8.716394e-02 -2.619771e-01
## PAY_AMT3                   -2.448393e-01 -5.475542e-01 -1.790583e-01
## PAY_AMT4                   -3.731955e-01 -3.350907e-03  7.561300e-02
## PAY_AMT5                   -4.698687e-01  1.592771e-01  6.227597e-02
## PAY_AMT6                   -3.575028e-01 -4.306835e-01  1.900638e-01
## default.payment.next.month -2.441713e-07  8.554837e-08  1.441512e-07
## ed.other                   -5.952895e-08 -6.177029e-08 -4.029573e-08
## ed.high_school             -4.060606e-06 -9.485931e-07  7.318980e-07
## ed.school                  -8.651648e-06 -3.320632e-06  2.003742e-06
## ed.university              -3.086885e-06 -7.760970e-07  5.044920e-07
## marriage.married           -3.977989e-06 -1.701712e-06  6.245093e-07
## marriage.single            -4.704388e-06 -1.882783e-06  1.280249e-06
## marriage.others            -1.013772e-07  7.803180e-09 -1.311795e-08
##                                      PC7           PC8           PC9
## SEX                        -1.174610e-07  1.187276e-06 -7.100879e-07
## AGE                         1.514887e-07  3.516323e-06 -2.187484e-06
## PAY_1                      -7.950970e-07 -3.288392e-06  2.428264e-06
## PAY_2                      -8.862946e-07 -4.894956e-06  1.669249e-06
## PAY_3                       1.712987e-07 -3.554414e-06  1.068221e-06
## PAY_4                      -1.059430e-06 -5.071150e-06  2.949098e-06
## PAY_5                       9.871106e-07 -3.844836e-06  5.277723e-06
## PAY_6                      -3.233630e-06 -2.176669e-06  4.700824e-06
```

```
## BILL_AMT1                  -5.784232e-02  2.612453e-02  1.639371e-01
## BILL_AMT2                   7.860051e-02  1.918421e-01  3.484413e-02
## BILL_AMT3                  -6.908146e-02 -1.534870e-01 -3.474197e-01
## BILL_AMT4                   2.872122e-01 -1.086755e-01 -1.511017e-01
## BILL_AMT5                  -4.807444e-01  1.034837e-01  1.026033e-01
## BILL_AMT6                   2.294940e-01 -1.042140e-01  2.420389e-01
## PAY_AMT1                    9.399958e-02  3.042951e-01  2.706315e-01
## PAY_AMT2                   -6.229616e-02  2.569055e-01  3.745537e-01
## PAY_AMT3                    2.838555e-01  3.527173e-01 -1.530299e-01
## PAY_AMT4                   -4.703760e-01  2.596694e-01 -6.002146e-01
## PAY_AMT5                    4.883207e-01 -1.949813e-01 -3.736668e-01
## PAY_AMT6                   -2.577384e-01 -7.191139e-01  1.497786e-01
## default.payment.next.month -1.028962e-07 -2.499193e-07 -1.679699e-07
## ed.other                    4.080268e-08  7.368398e-08 -9.062423e-08
## ed.high_school              3.044095e-08  1.277443e-06 -1.851432e-07
## ed.school                   8.033230e-08  1.986790e-06 -1.117627e-06
## ed.university               2.021123e-07  5.957239e-07 -7.079063e-08
## marriage.married           -4.819974e-07  1.138148e-06 -7.005156e-07
## marriage.single             5.108585e-07  9.349753e-07 -5.659778e-07
## marriage.others             6.052524e-08  2.104871e-08 -6.005771e-08
##                                    PC10          PC11          PC12
## SEX                         2.036624e-06 -7.436572e-07  1.129676e-06
## AGE                         8.792866e-06 -3.923402e-06  2.457274e-06
## PAY_1                      -4.146034e-06  2.369751e-06 -2.616384e-06
## PAY_2                      -6.314026e-06  4.068402e-06 -1.224670e-06
## PAY_3                      -5.350430e-06  1.189033e-06 -1.563536e-06
## PAY_4                      -2.355507e-06  2.953267e-06  2.803969e-08
## PAY_5                      -4.564415e-06  1.561645e-06 -2.916779e-06
## PAY_6                      -6.017116e-06  4.932324e-07  8.077485e-07
## BILL_AMT1                   2.478302e-01 -2.037808e-01  6.647957e-03
## BILL_AMT2                  -3.210795e-01  4.804633e-01  1.101541e-02
## BILL_AMT3                  -1.827823e-01 -4.374404e-01 -8.857876e-02
## BILL_AMT4                   3.769429e-01  2.292044e-01  2.989633e-01
## BILL_AMT5                   6.006808e-02  7.354659e-02 -6.031386e-01
## BILL_AMT6                  -1.972868e-01 -1.425941e-01  3.767805e-01
## PAY_AMT1                    4.808991e-01 -4.325673e-01  8.535603e-03
## PAY_AMT2                   -4.298254e-03  3.726952e-01  6.866880e-02
## PAY_AMT3                   -5.275991e-01 -2.733176e-01 -1.031887e-01
## PAY_AMT4                    1.413774e-01  9.280919e-02  4.061418e-01
## PAY_AMT5                    2.639755e-01  2.206015e-01 -4.576697e-01
## PAY_AMT6                   -1.347172e-01  4.938737e-04  8.378889e-02
## default.payment.next.month  1.118034e-07  1.525853e-07 -1.476304e-07
## ed.other                   -5.982512e-08 -1.087642e-07 -1.822524e-08
## ed.high_school              2.824303e-06 -1.328202e-06  2.085331e-06
## ed.school                   5.245684e-06 -2.347095e-06  1.718855e-06
## ed.university               2.245202e-06 -4.064207e-07  1.662469e-06
## marriage.married            2.420860e-06 -1.849774e-06  2.169335e-07
## marriage.single             2.849312e-06 -3.909275e-07  1.299498e-06
## marriage.others             2.200491e-08 -1.101321e-07 -1.341060e-08
##                                    PC13          PC14          PC15
## SEX                        -4.994388e-02  1.758165e-01  2.101293e-02
## AGE                        -2.596834e-01  6.524050e-01  1.111274e-01
## PAY_1                       2.845449e-01  2.077701e-01 -6.139335e-01
## PAY_2                       3.829019e-01  1.854558e-01 -4.240037e-01
```

```
## PAY_3                        4.076562e-01  1.668681e-01 -1.036243e-01
## PAY_4                        4.146728e-01  1.292552e-01  2.389101e-01
## PAY_5                        4.050352e-01  8.311893e-02  3.917349e-01
## PAY_6                        3.947186e-01  5.291373e-02  4.403649e-01
## BILL_AMT1                    1.198614e-05 -6.690806e-06  1.697121e-06
## BILL_AMT2                   -4.397442e-06 -1.336072e-06  2.072544e-06
## BILL_AMT3                   -4.387547e-06 -5.561526e-07  2.887197e-07
## BILL_AMT4                    3.533276e-07 -2.319828e-06 -2.162308e-06
## BILL_AMT5                   -5.729437e-06 -6.515125e-07 -1.071000e-06
## BILL_AMT6                   -4.331925e-06 -3.200078e-06 -2.163376e-06
## PAY_AMT1                     2.204872e-05 -8.197985e-06 -5.764811e-06
## PAY_AMT2                     1.028501e-05 -1.219756e-06 -1.911368e-06
## PAY_AMT3                     1.398044e-05  4.959350e-08 -1.034443e-06
## PAY_AMT4                     2.118311e-05 -5.844697e-06 -1.279737e-06
## PAY_AMT5                     1.554901e-05 -3.765884e-06  1.462420e-06
## PAY_AMT6                     8.854075e-06 -3.437856e-06 -1.349495e-06
## default.payment.next.month   3.565418e-03  1.389473e-01 -5.492867e-02
## ed.other                    -1.083454e-03  1.140839e-04  2.688836e-04
## ed.high_school              -7.551419e-02  3.187417e-01  4.941901e-02
## ed.school                   -1.531398e-01  3.949888e-01  6.825188e-02
## ed.university               -5.780995e-02  2.414498e-01  4.875016e-02
## marriage.married            -7.853307e-02  1.762228e-01  1.671912e-02
## marriage.single             -7.627864e-02  2.129252e-01  5.047921e-02
## marriage.others             -1.034405e-03  5.002343e-03  3.949675e-04
##                                      PC16          PC17          PC18
## SEX                          1.627752e-02  8.750777e-02 -6.665852e-02
## AGE                         -2.995491e-03  5.744671e-02 -6.626594e-02
## PAY_1                       -6.173908e-01  6.414465e-02 -2.587711e-01
## PAY_2                        2.871176e-01 -2.202249e-02  5.427556e-01
## PAY_3                        5.637225e-01 -3.201344e-02 -1.454100e-01
## PAY_4                        1.583899e-01 -1.742211e-02 -5.598575e-01
## PAY_5                       -1.962281e-01  1.550347e-02 -5.120993e-02
## PAY_6                       -3.779363e-01  1.789307e-02  4.471568e-01
## BILL_AMT1                    9.071210e-07  1.563716e-07 -1.181259e-06
## BILL_AMT2                   -2.002981e-06 -4.971292e-07  1.673945e-07
## BILL_AMT3                   -1.182029e-06  7.465039e-07  1.339919e-06
## BILL_AMT4                   -1.326285e-07 -4.726019e-07  1.113125e-06
## BILL_AMT5                    2.382202e-06  2.480886e-07 -2.326239e-07
## BILL_AMT6                    9.022128e-07 -1.679566e-07 -1.558364e-06
## PAY_AMT1                    -1.194680e-06 -2.048297e-07  2.436347e-06
## PAY_AMT2                     1.247216e-06 -4.297826e-07 -2.029782e-07
## PAY_AMT3                     5.342328e-07 -1.082609e-07 -1.891098e-06
## PAY_AMT4                    -1.664132e-06 -1.993218e-07 -8.711472e-07
## PAY_AMT5                    -2.171168e-06 -7.178214e-07  1.838785e-06
## PAY_AMT6                    -6.307774e-07 -4.082202e-07 -3.308277e-08
## default.payment.next.month -7.173902e-02 -8.771720e-04 -5.920080e-02
## ed.other                    -7.705337e-04  1.797029e-03 -2.304360e-03
## ed.high_school               4.699754e-03 -2.166173e-01  1.747435e-01
## ed.school                    9.975398e-04  4.730832e-02 -2.564323e-02
## ed.university                8.158430e-03 -1.705680e-01  2.176285e-01
## marriage.married            -5.806579e-02 -6.455242e-01 -8.811367e-02
## marriage.single              5.742924e-02  6.982767e-01  5.131589e-02
## marriage.others             -2.065799e-04 -1.827002e-03 -8.426785e-04
##                                      PC19          PC20          PC21
```

```
## SEX                          -3.011371e-01 -5.439066e-02 -9.269997e-01
## AGE                          -2.605972e-01  2.322482e-03  2.268556e-01
## PAY_1                         1.030241e-01  1.444884e-01 -1.842676e-02
## PAY_2                        -1.848766e-01 -4.601682e-01  5.295734e-02
## PAY_3                        -9.123618e-03  6.248619e-01 -3.065796e-03
## PAY_4                         1.689886e-01 -3.203558e-01 -1.227148e-04
## PAY_5                         4.500454e-03 -3.466360e-01  4.049519e-02
## PAY_6                        -1.704839e-01  3.882150e-01 -1.424139e-02
## BILL_AMT1                     5.299544e-07  1.303697e-06  2.534744e-07
## BILL_AMT2                    -3.737755e-07 -1.141532e-06  2.102662e-07
## BILL_AMT3                     1.286125e-07 -7.714272e-07 -2.177252e-07
## BILL_AMT4                    -5.381460e-07  1.608945e-06  2.150151e-08
## BILL_AMT5                     1.224179e-06 -2.527579e-07 -6.550249e-07
## BILL_AMT6                    -9.827655e-08 -8.248516e-07 -6.730994e-08
## PAY_AMT1                      7.972164e-07 -2.254360e-06 -2.572910e-07
## PAY_AMT2                      6.795517e-08  1.892255e-06  1.468281e-07
## PAY_AMT3                      1.843346e-06 -9.428161e-07 -4.989937e-08
## PAY_AMT4                      1.167263e-06 -1.423284e-06  5.328227e-07
## PAY_AMT5                      4.860265e-07  1.235560e-06 -5.761471e-07
## PAY_AMT6                      8.858846e-07  4.280460e-07 -3.692929e-07
## default.payment.next.month -8.387689e-02  1.283156e-02  6.368024e-02
## ed.other                     -5.287202e-03  8.967046e-04 -5.917765e-05
## ed.high_school                5.012724e-01  7.962890e-03 -1.206010e-01
## ed.school                    -1.267566e-01  3.491108e-03  1.384555e-01
## ed.university                 6.150463e-01  1.401235e-02 -1.904139e-01
## marriage.married             -2.690614e-01 -2.538711e-03  7.052401e-02
## marriage.single               1.174594e-01  5.750873e-04  7.215873e-02
## marriage.others               3.094603e-03  3.576971e-03 -2.084945e-03
##                                      PC22          PC23          PC24
## SEX                          -2.272875e-02 -7.700855e-03 -1.221135e-02
## AGE                           1.563736e-02  1.275858e-01  6.968118e-02
## PAY_1                        -2.515380e-02  1.190260e-01  8.279088e-03
## PAY_2                         1.133896e-01  5.308730e-03  8.391557e-03
## PAY_3                        -2.535791e-01  1.969639e-02  6.448905e-03
## PAY_4                         5.328498e-01 -1.617403e-02  7.003669e-03
## PAY_5                        -7.145802e-01  4.150947e-02  4.710116e-03
## PAY_6                         3.523879e-01 -1.360022e-02 -3.681348e-03
## BILL_AMT1                    -2.313214e-07 -6.527956e-07  3.440264e-08
## BILL_AMT2                     1.999122e-07  1.954971e-07 -5.419416e-08
## BILL_AMT3                    -3.088154e-07 -1.540538e-08  1.553855e-07
## BILL_AMT4                     5.348892e-07 -9.749554e-08 -2.426431e-07
## BILL_AMT5                     6.851993e-07 -1.891170e-08 -2.367308e-07
## BILL_AMT6                    -1.038164e-06 -3.287585e-09  7.421536e-08
## PAY_AMT1                      6.217775e-07 -1.035258e-06 -3.598166e-08
## PAY_AMT2                     -7.740161e-07 -2.707333e-07 -7.879775e-08
## PAY_AMT3                      2.481757e-06 -2.825527e-07  7.292261e-09
## PAY_AMT4                     -3.089837e-06 -2.474531e-07 -1.977012e-08
## PAY_AMT5                      1.119133e-06 -2.857288e-07 -4.870618e-07
## PAY_AMT6                     -2.829919e-07 -1.951784e-07 -3.777309e-07
## default.payment.next.month -4.800197e-02 -9.774235e-01 -7.388131e-03
## ed.other                      1.287133e-04  2.006302e-03  7.356346e-03
## ed.high_school               -1.058611e-02 -1.286482e-02 -7.377545e-01
## ed.school                     9.579186e-03  6.751613e-02  2.515492e-02
## ed.university                -1.166051e-02 -5.175566e-02  6.642126e-01
```

```
## marriage.married           -1.885696e-03  6.257076e-02  8.768240e-02
## marriage.single             7.377785e-03  1.502572e-02  8.398431e-03
## marriage.others             3.679473e-03 -2.343316e-03 -2.774480e-02
##                                     PC25          PC26          PC27
## SEX                        -1.098827e-03 -9.080783e-03 -1.157969e-03
## AGE                         6.552254e-02  4.633396e-01  1.661856e-01
## PAY_1                      -2.668582e-04 -1.975495e-03 -3.450206e-06
## PAY_2                       1.762340e-03  3.197400e-03 -4.811322e-04
## PAY_3                      -2.006863e-03  5.812044e-04  4.312657e-04
## PAY_4                      -1.713267e-03 -1.433924e-03 -7.021459e-04
## PAY_5                       4.996782e-03  1.125720e-03 -3.671973e-04
## PAY_6                      -3.357714e-03  1.973568e-03  1.212946e-03
## BILL_AMT1                  -1.616656e-08 -1.106234e-07 -5.425752e-09
## BILL_AMT2                   5.079917e-08 -2.879529e-08 -3.599314e-08
## BILL_AMT3                  -3.631167e-08 -8.000041e-08  4.779923e-08
## BILL_AMT4                   5.283877e-08 -1.715224e-08 -3.930950e-08
## BILL_AMT5                   5.696305e-08 -2.114595e-08 -5.038830e-08
## BILL_AMT6                  -6.921757e-08  2.245749e-07  5.992019e-08
## PAY_AMT1                   -2.370285e-08  4.074902e-08 -3.536513e-09
## PAY_AMT2                    3.617539e-08  5.421583e-09 -4.055122e-08
## PAY_AMT3                   -1.590220e-08 -2.214886e-07  7.734258e-08
## PAY_AMT4                    4.652286e-08 -9.243737e-09  8.456886e-09
## PAY_AMT5                   -6.327530e-08 -1.044964e-07 -3.507302e-08
## PAY_AMT6                    7.958019e-09  3.563298e-08 -3.530966e-08
## default.payment.next.month -4.693718e-03  4.323984e-03  4.404874e-04
## ed.other                   -7.698843e-02  3.299730e-01 -9.406210e-01
## ed.high_school             -4.431952e-02  1.706590e-02  1.336661e-04
## ed.school                   2.933921e-01 -7.744452e-01 -2.940008e-01
## ed.university               2.240248e-02  1.100169e-02  3.112533e-03
## marriage.married           -3.919074e-01 -2.735200e-02  1.360402e-02
## marriage.single            -3.935797e-01 -1.743775e-03  2.221317e-02
## marriage.others             7.698497e-01  2.745588e-01  2.204228e-02
##                                     PC28
## SEX                        -2.526586e-03
## AGE                         3.154843e-01
## PAY_1                       4.450828e-04
## PAY_2                      -7.801586e-05
## PAY_3                       2.016428e-04
## PAY_4                       4.108864e-04
## PAY_5                       3.252575e-04
## PAY_6                       6.809812e-06
## BILL_AMT1                   4.225614e-09
## BILL_AMT2                  -1.503755e-08
## BILL_AMT3                   1.305471e-08
## BILL_AMT4                  -4.364877e-09
## BILL_AMT5                  -6.587599e-09
## BILL_AMT6                  -2.826747e-09
## PAY_AMT1                   -1.678314e-08
## PAY_AMT2                   -4.651063e-09
## PAY_AMT3                   -1.739533e-09
## PAY_AMT4                    4.393709e-09
## PAY_AMT5                   -8.656330e-09
## PAY_AMT6                   -8.000439e-09
## default.payment.next.month  5.074205e-04
```

```
## ed.other                    1.804076e-02
## ed.high_school             -9.043577e-03
## ed.school                   1.280622e-02
## ed.university               9.999352e-03
## marriage.married           -5.395885e-01
## marriage.single            -5.272744e-01
## marriage.others            -5.749932e-01
```

Según los resultados del filtro PCA, no hay ningún dato que sea redundate. Por lo tanto, no eliminaremos ninguno de los atributos del conjunto de datos.

Estudio de los parámetros e hiperparámetros.

Vamos a realizar un estudio sobre los parámetros para saber cuáles de ellos son los más importantes. Con los más importantes crearemos los modelos lineales que vamos a ajustar.

Con la función regsubsets, podemos encontrar que atributos son mejores a la hora de escogerlos antes.

```
muestra_regsubsets = regsubsets(default.payment.next.month ~ ., data = trainTransformado, nvmax = 28, m
summary((muestra_regsubsets))
```

```
## Subset selection object
## Call: regsubsets.formula(default.payment.next.month ~ ., data = trainTransformado,
##     nvmax = 28, method = "exhaustive")
## 28 Variables  (and intercept)
##                  Forced in Forced out
## LIMIT_BAL            FALSE      FALSE
## SEX                  FALSE      FALSE
## AGE                  FALSE      FALSE
## PAY_1                FALSE      FALSE
## PAY_2                FALSE      FALSE
## PAY_3                FALSE      FALSE
## PAY_4                FALSE      FALSE
## PAY_5                FALSE      FALSE
## PAY_6                FALSE      FALSE
## BILL_AMT1            FALSE      FALSE
## BILL_AMT2            FALSE      FALSE
## BILL_AMT3            FALSE      FALSE
## BILL_AMT4            FALSE      FALSE
## BILL_AMT5            FALSE      FALSE
## BILL_AMT6            FALSE      FALSE
## PAY_AMT1             FALSE      FALSE
## PAY_AMT2             FALSE      FALSE
## PAY_AMT3             FALSE      FALSE
## PAY_AMT4             FALSE      FALSE
## PAY_AMT5             FALSE      FALSE
## PAY_AMT6             FALSE      FALSE
## ed.other             FALSE      FALSE
## ed.high_school       FALSE      FALSE
## ed.school            FALSE      FALSE
## ed.university        FALSE      FALSE
## marriage.married     FALSE      FALSE
## marriage.single      FALSE      FALSE
## marriage.others      FALSE      FALSE
## 1 subsets of each size up to 28
```

```
## Selection Algorithm: exhaustive
##           LIMIT_BAL SEX AGE PAY_1 PAY_2 PAY_3 PAY_4 PAY_5 PAY_6 BILL_AMT1
## 1  ( 1 )  " "       " " " " "*"   " "   " "   " "   " "   " "   " "
## 2  ( 1 )  " "       " " " " "*"   " "   " "   " "   " "   " "   "*"
## 3  ( 1 )  " "       " " " " "*"   "*"   " "   " "   " "   " "   "*"
## 4  ( 1 )  "*"       " " " " "*"   "*"   " "   " "   " "   " "   "*"
## 5  ( 1 )  "*"       " " "*" "*"   "*"   " "   " "   " "   " "   "*"
## 6  ( 1 )  "*"       " " "*" "*"   "*"   " "   " "   " "   " "   "*"
## 7  ( 1 )  "*"       " " "*" "*"   " "   "*"   " "   " "   " "   "*"
## 8  ( 1 )  "*"       " " "*" "*"   "*"   " "   " "   "*"   " "   "*"
## 9  ( 1 )  "*"       " " "*" "*"   "*"   " "   " "   "*"   " "   "*"
## 10  ( 1 ) "*"       " " "*" "*"   "*"   "*"   " "   "*"   " "   "*"
## 11  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   " "   "*"   " "   "*"
## 12  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   " "   "*"   " "   "*"
## 13  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   " "   "*"   " "   "*"
## 14  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   " "   "*"   " "   "*"
## 15  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   " "   "*"   " "   "*"
## 16  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   " "   "*"   " "   "*"
## 17  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   " "   "*"   " "   "*"
## 18  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   " "   "*"   " "   "*"
## 19  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   "*"   "*"   " "   "*"
## 20  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   "*"   "*"   " "   "*"
## 21  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   "*"   "*"   " "   "*"
## 22  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   "*"   "*"   " "   "*"
## 23  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   "*"   "*"   "*"   "*"
## 24  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   "*"   "*"   "*"   "*"
## 25  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   "*"   "*"   "*"   "*"
## 26  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   "*"   "*"   "*"   "*"
## 27  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   "*"   "*"   "*"   "*"
## 28  ( 1 ) "*"       "*" "*" "*"   "*"   "*"   "*"   "*"   "*"   "*"
##           BILL_AMT2 BILL_AMT3 BILL_AMT4 BILL_AMT5 BILL_AMT6 PAY_AMT1
## 1  ( 1 )  " "       " "       " "       " "       " "       " "
## 2  ( 1 )  " "       " "       " "       " "       " "       " "
## 3  ( 1 )  " "       " "       " "       " "       " "       " "
## 4  ( 1 )  " "       " "       " "       " "       " "       " "
## 5  ( 1 )  " "       " "       " "       " "       " "       " "
## 6  ( 1 )  " "       " "       " "       " "       " "       " "
## 7  ( 1 )  " "       " "       " "       " "       " "       "*"
## 8  ( 1 )  " "       " "       " "       " "       " "       "*"
## 9  ( 1 )  " "       " "       " "       " "       " "       "*"
## 10  ( 1 ) " "       " "       " "       " "       " "       "*"
## 11  ( 1 ) " "       " "       " "       " "       " "       "*"
## 12  ( 1 ) " "       " "       " "       " "       " "       "*"
## 13  ( 1 ) " "       " "       " "       " "       " "       "*"
## 14  ( 1 ) " "       " "       " "       " "       " "       "*"
## 15  ( 1 ) " "       " "       " "       " "       " "       "*"
## 16  ( 1 ) " "       " "       " "       " "       " "       "*"
## 17  ( 1 ) "*"       " "       " "       " "       " "       "*"
## 18  ( 1 ) "*"       " "       " "       " "       " "       "*"
## 19  ( 1 ) "*"       " "       " "       " "       " "       "*"
## 20  ( 1 ) "*"       " "       " "       " "       " "       "*"
## 21  ( 1 ) "*"       " "       " "       " "       " "       "*"
## 22  ( 1 ) "*"       " "       " "       " "       " "       "*"
## 23  ( 1 ) "*"       " "       " "       " "       " "       "*"
```

```
## 24  ( 1 ) "*"       " "       " "       "*"       " "       "*"
## 25  ( 1 ) "*"       " "       "*"       "*"       " "       "*"
## 26  ( 1 ) "*"       " "       "*"       "*"       "*"       "*"
## 27  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
## 28  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
##           PAY_AMT2 PAY_AMT3 PAY_AMT4 PAY_AMT5 PAY_AMT6 ed.other
## 1  ( 1 )  " "      " "      " "      " "      " "      " "
## 2  ( 1 )  " "      " "      " "      " "      " "      " "
## 3  ( 1 )  " "      " "      " "      " "      " "      " "
## 4  ( 1 )  " "      " "      " "      " "      " "      " "
## 5  ( 1 )  " "      " "      " "      " "      " "      " "
## 6  ( 1 )  " "      " "      " "      " "      " "      " "
## 7  ( 1 )  " "      " "      " "      " "      " "      " "
## 8  ( 1 )  " "      " "      " "      " "      " "      " "
## 9  ( 1 )  " "      " "      " "      " "      " "      " "
## 10  ( 1 ) " "      " "      " "      " "      " "      " "
## 11  ( 1 ) " "      " "      " "      " "      " "      " "
## 12  ( 1 ) "*"      " "      " "      " "      " "      " "
## 13  ( 1 ) "*"      " "      " "      " "      " "      " "
## 14  ( 1 ) "*"      " "      "*"      " "      " "      " "
## 15  ( 1 ) "*"      " "      " "      " "      " "      " "
## 16  ( 1 ) "*"      " "      "*"      " "      " "      " "
## 17  ( 1 ) "*"      " "      "*"      " "      " "      " "
## 18  ( 1 ) "*"      " "      "*"      "*"      " "      " "
## 19  ( 1 ) "*"      " "      "*"      "*"      " "      " "
## 20  ( 1 ) "*"      "*"      "*"      "*"      " "      " "
## 21  ( 1 ) "*"      "*"      "*"      "*"      "*"      " "
## 22  ( 1 ) "*"      "*"      "*"      "*"      "*"      "*"
## 23  ( 1 ) "*"      "*"      "*"      "*"      "*"      "*"
## 24  ( 1 ) "*"      "*"      "*"      "*"      "*"      "*"
## 25  ( 1 ) "*"      "*"      "*"      "*"      "*"      "*"
## 26  ( 1 ) "*"      "*"      "*"      "*"      "*"      "*"
## 27  ( 1 ) "*"      "*"      "*"      "*"      "*"      "*"
## 28  ( 1 ) "*"      "*"      "*"      "*"      "*"      "*"
##           ed.high_school ed.school ed.university marriage.married
## 1  ( 1 )  " "            " "       " "           " "
## 2  ( 1 )  " "            " "       " "           " "
## 3  ( 1 )  " "            " "       " "           " "
## 4  ( 1 )  " "            " "       " "           " "
## 5  ( 1 )  " "            " "       " "           " "
## 6  ( 1 )  " "            "*"       " "           " "
## 7  ( 1 )  " "            "*"       " "           " "
## 8  ( 1 )  " "            "*"       " "           " "
## 9  ( 1 )  " "            "*"       " "           "*"
## 10  ( 1 ) " "            "*"       " "           "*"
## 11  ( 1 ) " "            "*"       " "           "*"
## 12  ( 1 ) " "            "*"       " "           "*"
## 13  ( 1 ) "*"            "*"       " "           "*"
## 14  ( 1 ) "*"            "*"       " "           "*"
## 15  ( 1 ) "*"            "*"       " "           "*"
## 16  ( 1 ) "*"            "*"       " "           "*"
## 17  ( 1 ) "*"            "*"       " "           "*"
## 18  ( 1 ) "*"            "*"       " "           "*"
## 19  ( 1 ) "*"            "*"       " "           "*"
```

```
## 20  ( 1 ) "*"            "*"       " "           "*"
## 21  ( 1 ) "*"            "*"       " "           "*"
## 22  ( 1 ) "*"            "*"       " "           "*"
## 23  ( 1 ) "*"            "*"       " "           "*"
## 24  ( 1 ) "*"            "*"       " "           "*"
## 25  ( 1 ) "*"            "*"       " "           "*"
## 26  ( 1 ) "*"            "*"       " "           "*"
## 27  ( 1 ) "*"            "*"       " "           "*"
## 28  ( 1 ) "*"            "*"       "*"           "*"
##           marriage.single marriage.others
## 1  ( 1 )  " "              " "
## 2  ( 1 )  " "              " "
## 3  ( 1 )  " "              " "
## 4  ( 1 )  " "              " "
## 5  ( 1 )  " "              " "
## 6  ( 1 )  " "              " "
## 7  ( 1 )  " "              " "
## 8  ( 1 )  " "              " "
## 9  ( 1 )  " "              " "
## 10  ( 1 ) " "              " "
## 11  ( 1 ) " "              " "
## 12  ( 1 ) " "              " "
## 13  ( 1 ) " "              " "
## 14  ( 1 ) " "              " "
## 15  ( 1 ) "*"              "*"
## 16  ( 1 ) "*"              "*"
## 17  ( 1 ) "*"              "*"
## 18  ( 1 ) "*"              "*"
## 19  ( 1 ) "*"              "*"
## 20  ( 1 ) "*"              "*"
## 21  ( 1 ) "*"              "*"
## 22  ( 1 ) "*"              "*"
## 23  ( 1 ) "*"              "*"
## 24  ( 1 ) "*"              "*"
## 25  ( 1 ) "*"              "*"
## 26  ( 1 ) "*"              "*"
## 27  ( 1 ) "*"              "*"
## 28  ( 1 ) "*"              "*"
```

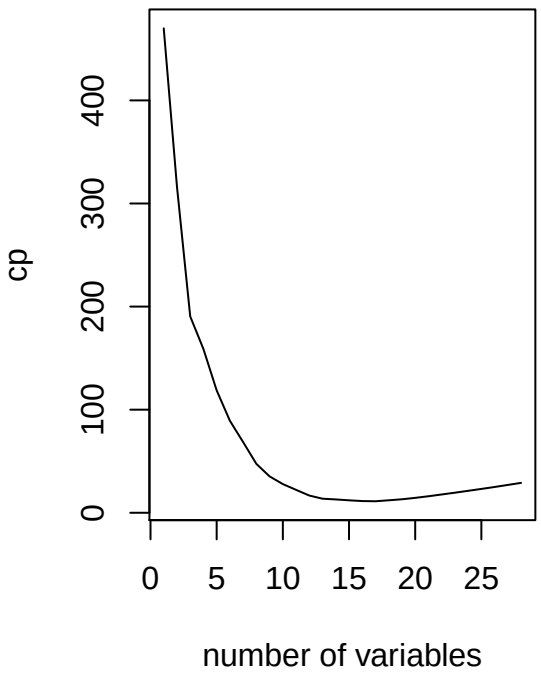
```
reg.sumary = summary(muestra_regsubsets)
```

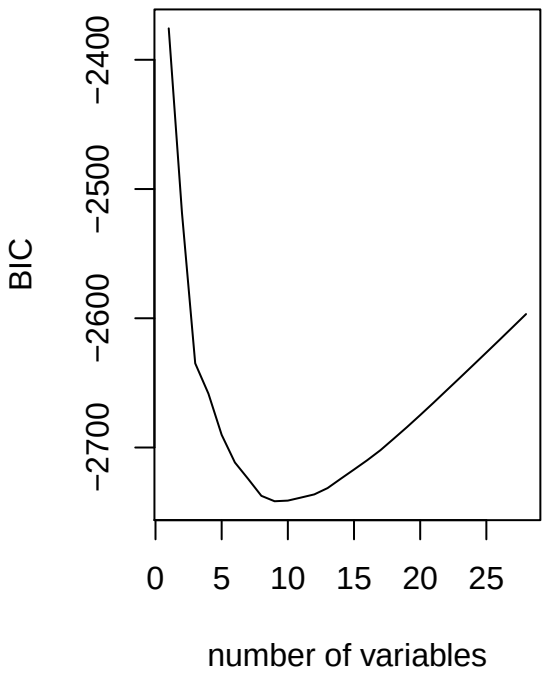
Ahora, calcularemos los valores mínimos de los criterios cp y BIC. Utilizaremos estos dos valores para probar diferentes modelos.

```
par(mfrow=c(1,2))
plot(reg.sumary$cp, xlab="number of variables", ylab="cp", type="l")
which.min(reg.sumary$cp)
```

```
## [1] 17
```

```
plot(reg.sumary$bic, xlab="number of variables", ylab="BIC", type="l")
```






```
which.min(reg.sumary$bic)
```

```
## [1] 9
```

```
par(mfrow=c(1,1) )
```

Según los resultados obtenidos, los dos mejores conjuntos de datos son de tamaño 17 y de tamaño 9. Tomaremos los atributos en el orden que aparecen en la salida de la función regsubsets.

Modelos lineales.

Para hacer más sencillo el cálculo del error teniendo un modelo, crearemos funciones para calcular el error, para calcular el conjunto de datos precedidos, y otra que las englobe.

```
# Función para calcular la solución dada una predicción.
calculateSol = function(x){
  prediction.model = rep(0,length(x))
  prediction.model[x >= 0.5] = 1

  prediction.model
}

# Función para calcular el Error.
calculateErrorClasification = function(calculated.solution, real.sol){
  er = sum(calculated.solution != real.sol)/length(calculated.solution)
  er
}
```

```r
# Función que calcula el Error pasándole la predicción.
calculateError = function(model.prediction, labels){
  pred = calculateSol(model.prediction)
  return(calculateErrorClasification(pred, labels))
}
```

Transformamos los valores de test.

```r
testTransformado = reemplazarCol(credit_card.test)
testTransformado = predict(trans, testTransformado)
summary(testTransformado)
```

```
##    LIMIT_BAL           SEX              AGE            PAY_1
##  Min.   : 49.50   Min.   :0.0000   Min.   :1.564   Min.   :-2.00000
##  1st Qu.: 82.29   1st Qu.:0.0000   1st Qu.:1.622   1st Qu.:-1.00000
##  Median :113.27   Median :0.0000   Median :1.657   Median : 0.00000
##  Mean   :111.07   Mean   :0.3936   Mean   :1.656   Mean   :-0.02011
##  3rd Qu.:132.00   3rd Qu.:1.0000   3rd Qu.:1.688   3rd Qu.: 0.00000
##  Max.   :193.37   Max.   :1.0000   Max.   :1.769   Max.   : 8.00000
##      PAY_2             PAY_3            PAY_4           PAY_5
##  Min.   :-2.0000   Min.   :-2.000   Min.   :-2.00   Min.   :-2.0000
##  1st Qu.:-1.0000   1st Qu.:-1.000   1st Qu.:-1.00   1st Qu.:-1.0000
##  Median : 0.0000   Median : 0.000   Median : 0.00   Median : 0.0000
##  Mean   :-0.1377   Mean   :-0.158   Mean   :-0.22   Mean   :-0.2626
##  3rd Qu.: 0.0000   3rd Qu.: 0.000   3rd Qu.: 0.00   3rd Qu.: 0.0000
##  Max.   : 7.0000   Max.   : 7.000   Max.   : 7.00   Max.   : 7.0000
##      PAY_6          BILL_AMT1         BILL_AMT2        BILL_AMT3
##  Min.   :-2.000   Min.   :-165580   Min.   :-67526   Min.   :-157264
##  1st Qu.:-1.000   1st Qu.:   3692   1st Qu.:  3024   1st Qu.:   2905
##  Median : 0.000   Median :  22984   Median : 21933   Median :  20309
##  Mean   :-0.296   Mean   :  51189   Mean   : 49383   Mean   :  46801
##  3rd Qu.: 0.000   3rd Qu.:  68619   3rd Qu.: 66224   3rd Qu.:  61433
##  Max.   : 7.000   Max.   : 621749   Max.   :577681   Max.   : 855086
##    BILL_AMT4        BILL_AMT5        BILL_AMT6         PAY_AMT1
##  Min.   :-17250   Min.   :-37594   Min.   :-39046   Min.   :     0
##  1st Qu.:  2444   1st Qu.:  1865   1st Qu.:  1229   1st Qu.:  1000
##  Median : 19063   Median : 18135   Median : 17132   Median :  2194
##  Mean   : 42821   Mean   : 40115   Mean   : 38695   Mean   :  5793
##  3rd Qu.: 56095   3rd Qu.: 51333   3rd Qu.: 50140   3rd Qu.:  5006
##  Max.   :565669   Max.   :587067   Max.   :527566   Max.   :873552
##     PAY_AMT2           PAY_AMT3         PAY_AMT4         PAY_AMT5
##  Min.   :      0.0   Min.   :     0   Min.   :     0   Min.   :     0.0
##  1st Qu.:    918.5   1st Qu.:   390   1st Qu.:   264   1st Qu.:   211.5
##  Median :   2057.5   Median :  1913   Median :  1549   Median :  1530.5
##  Mean   :   5763.2   Mean   :  5243   Mean   :  4903   Mean   :  4709.2
##  3rd Qu.:   5000.0   3rd Qu.:  4500   3rd Qu.:  4024   3rd Qu.:  4022.5
##  Max.   :1215471.0   Max.   :889043   Max.   :621000   Max.   :317077.0
##     PAY_AMT6      default.payment.next.month    ed.other
##  Min.   :     0   Min.   :0.0000             Min.   :0.000000
##  1st Qu.:   166   1st Qu.:0.0000             1st Qu.:0.000000
##  Median :  1500   Median :0.0000             Median :0.000000
##  Mean   :  5131   Mean   :0.2219             Mean   :0.004667
##  3rd Qu.:  4012   3rd Qu.:0.0000             3rd Qu.:0.000000
##  Max.   :351282   Max.   :1.0000             Max.   :1.000000
```

```
##  ed.high_school     ed.school      ed.university    marriage.married
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:1.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :1.0000   Median :1.0000   Median :0.0000   Median :0.0000
##  Mean   :0.6278   Mean   :0.9843   Mean   :0.4603   Mean   :0.4491
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##  marriage.single  marriage.others
##  Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.0000   1st Qu.:0.00000
##  Median :1.0000   Median :0.00000
##  Mean   :0.5381   Mean   :0.01078
##  3rd Qu.:1.0000   3rd Qu.:0.00000
##  Max.   :1.0000   Max.   :1.00000
```

Ahora, pasaremos a probar diferentes modelos lineales. Para ello utilizaremos la función glm, que utiliza regresión logística. Una vez hecho esto, nos quedaremos con el modelo que menor error fuera de la muestra obtenga y que sea más sencillo. Calcularemos también el error dentro de la muestra para comprobar que ambos valores son cercanos y que no hay sobreajuste en el modelo.

Por último, calcularemos la curva ROC y el área del modelo que hayamos seleccionado.

```
m1 = glm(default.payment.next.month~LIMIT_BAL+SEX+PAY_3+PAY_4+PAY_5+PAY_6+BILL_AMT1+BILL_AMT2+BILL_AMT3+
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
predtr.m1 = predict(m1, trainTransformado)
Ein.m1 = calculateError(predtr.m1, trainTransformado$default.payment.next.month)
Ein.m1
```

```
## [1] 0.2172857
```

```
pred.m1 = predict(m1, testTransformado)
Eout.m1 = calculateError(pred.m1, testTransformado$default.payment.next.month)
Eout.m1
```

```
## [1] 0.2196667
```

```
m2 = glm(default.payment.next.month~PAY_1+PAY_2+BILL_AMT1+LIMIT_BAL+AGE+ed.school+PAY_AMT1+PAY_5+marriag
predtr.m2 = predict(m2, trainTransformado)
Ein.m2 = calculateError(predtr.m2, trainTransformado$default.payment.next.month)
Ein.m2
```

```
## [1] 0.208
```

```
pred.m2 = predict(m2, testTransformado)
Eout.m2 = calculateError(pred.m2, testTransformado$default.payment.next.month)
Eout.m2
```

```
## [1] 0.2111111
```

```
m3= glm(default.payment.next.month~PAY_1+PAY_2+BILL_AMT1+LIMIT_BAL+AGE+ed.school+PAY_AMT1+PAY_5+marriage
predtr.m3 = predict(m3, trainTransformado)
Ein.m3 = calculateError(predtr.m3, trainTransformado$default.payment.next.month)
Ein.m3
```

```
## [1] 0.2091905
```

```
pred.m3 = predict(m3, testTransformado)
Eout.m3 = calculateError(pred.m3, testTransformado$default.payment.next.month)
Eout.m3
```

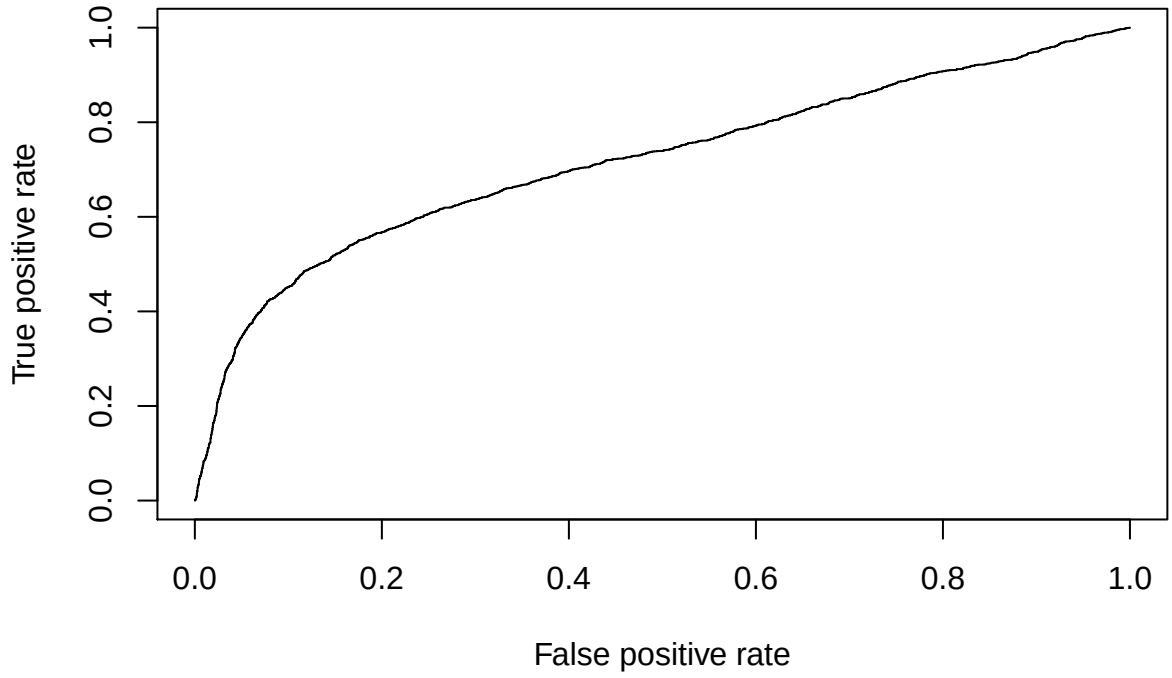
```
## [1] 0.2117778
```

Nuestro mejor modelo es el tercer modelo, ya que es el más simple de los 3 y obtiene resultados casi iguales que los anteriores. La diferencia entre el error dentro de la muestra y fuera de la muestra son muy parecidos, por lo tanto, el modelo no presenta sobreajuste.Ahora pasaremos a calcular la curva ROC de este modelo(utilizamos las probabilidades para calcular la curva ROC, para ello utilizamos la libreria ROCR. Primero calculamos las probabilidades del modelo y con estas dibujamos la curva ROC), y también calcularemos el área debajo de la curva.

```
linear.model = ROCR::prediction(pred.m3, testTransformado$default.payment.next.month)
perf = performance(linear.model, "tpr", "fpr")
auc.linear = performance(linear.model, measure = "auc")
print(auc.linear@y.values[[1]])
```

```
## [1] 0.7182492
```

```
plot(perf)
```



Modelos Boosting.

Para crear los modelos del boosting, utilizaremos la biblioteca ada, que contiene la función ada la cuál implementa el algoritmo adaboost. Uno de los parámetros de este algoritmo indica el número de clasficadores que se van a utilizar para separar los datos de la muestra; probaremos varios números de clasificadores. Nos quedaremos con el clasificador que menor error fuera de la muestra cometa. También calcularemos su curva ROC, y calcularemos cuál es el área de esta.

```
bt1 = ada::ada(default.payment.next.month~PAY_1+PAY_2+BILL_AMT1+LIMIT_BAL+AGE+ed.school+PAY_AMT1+PAY_5+
pred.bt1 = predict(bt1, trainTransformado)
Ein.bt1 = mean(pred.bt1 != trainTransformado$default.payment.next.month)
Ein.bt1
```

```
## [1] 0.1778095
```

```
predTs.bt1 = predict(bt1, testTransformado)
Eout.bt1 = mean(predTs.bt1 != testTransformado$default.payment.next.month)
Eout.bt1
```

```
## [1] 0.182
```

```
bt2 = ada::ada(formula = default.payment.next.month~PAY_1+PAY_2+BILL_AMT1+LIMIT_BAL+AGE+ed.school+PAY_A
pred.bt2 = predict(bt2, trainTransformado)
Ein.bt2 = mean(pred.bt2 != trainTransformado$default.payment.next.month)
Ein.bt2
```

```
## [1] 0.1780476
```

```
predTs.bt2 = predict(bt2, testTransformado)
Eout.bt2 = mean(predTs.bt2 != testTransformado$default.payment.next.month)
Eout.bt2
```

```
## [1] 0.1827778
```

```
bt3 = ada::ada(formula=default.payment.next.month~PAY_1+PAY_2+BILL_AMT1+LIMIT_BAL+AGE+ed.school+PAY_AMT
pred.bt3 = predict(bt3, trainTransformado)
Ein.bt3 = mean(pred.bt3 != trainTransformado$default.payment.next.month)
Ein.bt3
```

```
## [1] 0.179381
```

```
predTs.bt3 = predict(bt3, testTransformado)
Eout.bt3 = mean(predTs.bt3 != testTransformado$default.payment.next.month)
Eout.bt3
```

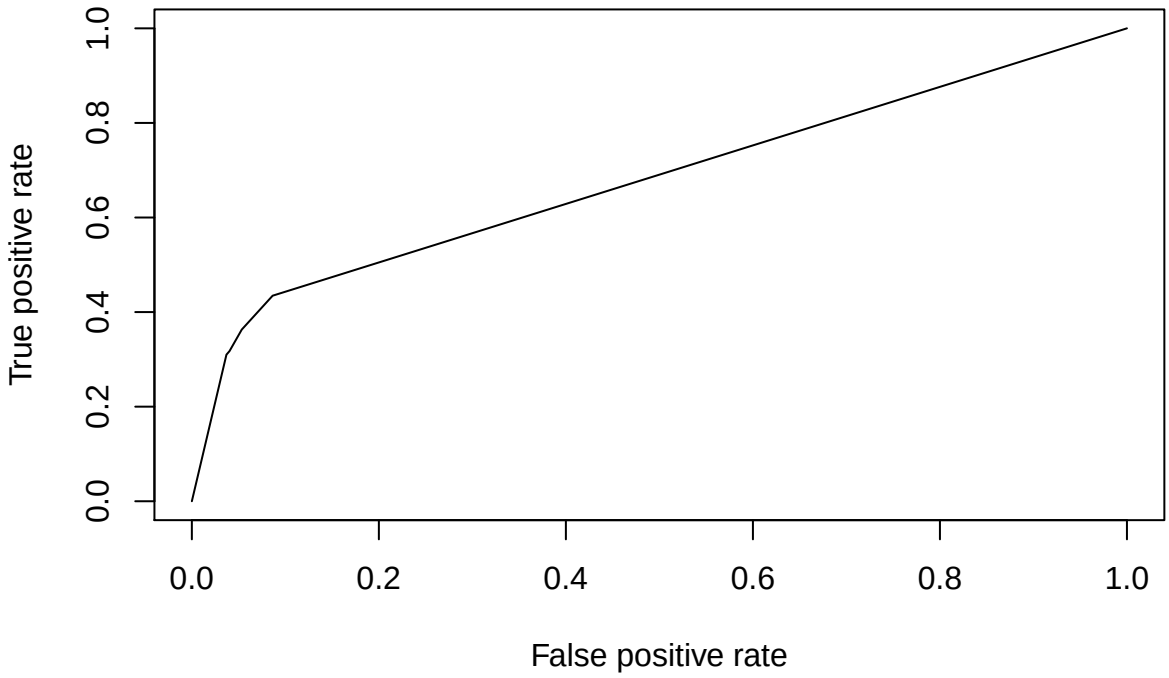
```
## [1] 0.1827778
```

Como los errores son todos muy parecidos entre ellos, nos quedaremos con el modelo más sencillo, el cuál es el modelo 3; ya que es el que menos clasificadores utiliza. Ahora calcularemos su curva ROC y el area debajo de esta.

```
predi = predict(bt3, testTransformado, type = "probs")
salida.boosting = predi[,2]
boosting.model = ROCR::prediction(salida.boosting, testTransformado$default.payment.next.month)
perf.boost = performance(boosting.model, "tpr", "fpr")
auc.boosting = performance(boosting.model, measure = "auc")
print(auc.boosting@y.values[[1]])
```

```
## [1] 0.6797438
```

```
plot(perf.boost)
```



Modelos Redes Neuronales

Para crear los modelos del Redes Neuronales, utilizaremos la biblioteca neuralnet, que contiene la función neuralnet la cuál implementa el algoritmo backpropagation. Los parametros utilizados en este algoritmo son: hidden = indica el número de capas ocultas y los nodos que contiene en cada capa que se van a utilizar para separar los datos de la muestra; probaremos con distintas capas; stepmax = fija el número máximo de pasos para el entrenamiento de las redes neuronales; threshold = es el umbral que limita el valor de las derivadas parciales de la función de error usandolo como criterio de parada. Nos quedaremos con la red que menor error fuera de la muestra cometa. También calcularemos su curva ROC, y calcularemos cuál es el área de esta.

```
nn1 = neuralnet(formula = default.payment.next.month~PAY_1+PAY_2+BILL_AMT1+LIMIT_BAL+AGE+ed.school+PAY_A

plot(nn1)


n <- c("PAY_1", "PAY_2", "BILL_AMT1", "LIMIT_BAL","AGE","ed.school","PAY_AMT1","PAY_5","marriage.married

pred.nn1 = compute(nn1, trainTransformado[names(trainTransformado) %in% n])

pred.nn1_ = pred.nn1$net.result*(max(pred.nn1$net.result)-min(pred.nn1$net.result))+min(pred.nn1$net.res

calculateError(pred.nn1_,trainTransformado$default.payment.next.month)
```

```
## [1] 0.2209047619
```

```
pred.nn1Ts = compute(nn1, testTransformado[names(testTransformado) %in% n])

pred.nn1_test = pred.nn1Ts$net.result*(max(pred.nn1Ts$net.result)-min(pred.nn1Ts$net.result))+min(pred.n


calculateError(pred.nn1_test,testTransformado$default.payment.next.month)
```

```
## [1] 0.2218888889
```

```
nn2 = neuralnet(formula = default.payment.next.month~PAY_1+PAY_2+BILL_AMT1+LIMIT_BAL+AGE+ed.school+PAY_A

plot(nn2)

pred.nn2 = compute(nn2, trainTransformado[names(trainTransformado) %in% n])

pred.nn2_ = pred.nn2$net.result*(max(pred.nn2$net.result)-min(pred.nn2$net.result))+min(pred.nn2$net.res

calculateError(pred.nn2_,trainTransformado$default.payment.next.month)
```

```
## [1] 0.3118571429
```

```
pred.nn2Ts = compute(nn2, testTransformado[names(testTransformado) %in% n])

pred.nn2_test = pred.nn2Ts$net.result*(max(pred.nn2Ts$net.result)-min(pred.nn2Ts$net.result))+min(pred.n


calculateError(pred.nn2_test,testTransformado$default.payment.next.month)
```

```
## [1] 0.2505555556
```

```
nn3 = neuralnet(formula = default.payment.next.month~PAY_1+PAY_2+BILL_AMT1+LIMIT_BAL+AGE+ed.school+PAY_A

plot(nn3)

pred.nn3 = compute(nn3, trainTransformado[names(trainTransformado) %in% n])

pred.nn3_ = pred.nn3$net.result*(max(pred.nn3$net.result)-min(pred.nn3$net.result))+min(pred.nn3$net.res

calculateError(pred.nn3_,trainTransformado$default.payment.next.month)
```

```
## [1] 0.2223809524
```

```
pred.nn3Ts = compute(nn3, testTransformado[names(testTransformado) %in% n])

pred.nn3_test = pred.nn3Ts$net.result*(max(pred.nn3Ts$net.result)-min(pred.nn3Ts$net.result))+min(pred.n


calculateError(pred.nn3_test,testTransformado$default.payment.next.month)
```

```
## [1] 0.2228888889
```

```
nn4 = neuralnet(formula = default.payment.next.month~PAY_1+PAY_2+BILL_AMT1+LIMIT_BAL+AGE+ed.school+PAY_A
plot(nn4)

pred.nn4 = compute(nn4, trainTransformado[names(trainTransformado) %in% n])

pred.nn4_ = pred.nn4$net.result*(max(pred.nn4$net.result)-min(pred.nn4$net.result))+min(pred.nn4$net.res
```

```
calculateError(pred.nn4_,trainTransformado$default.payment.next.month)
```

```
## [1] 0.2251428571
```

```
pred.nn4Ts = compute(nn4, testTransformado[names(testTransformado) %in% n])

pred.nn4_test = pred.nn4Ts$net.result*(max(pred.nn4Ts$net.result)-min(pred.nn4Ts$net.result))+min(pred.n


calculateError(pred.nn4_test,testTransformado$default.payment.next.month)
```

```
## [1] 0.2252222222
```

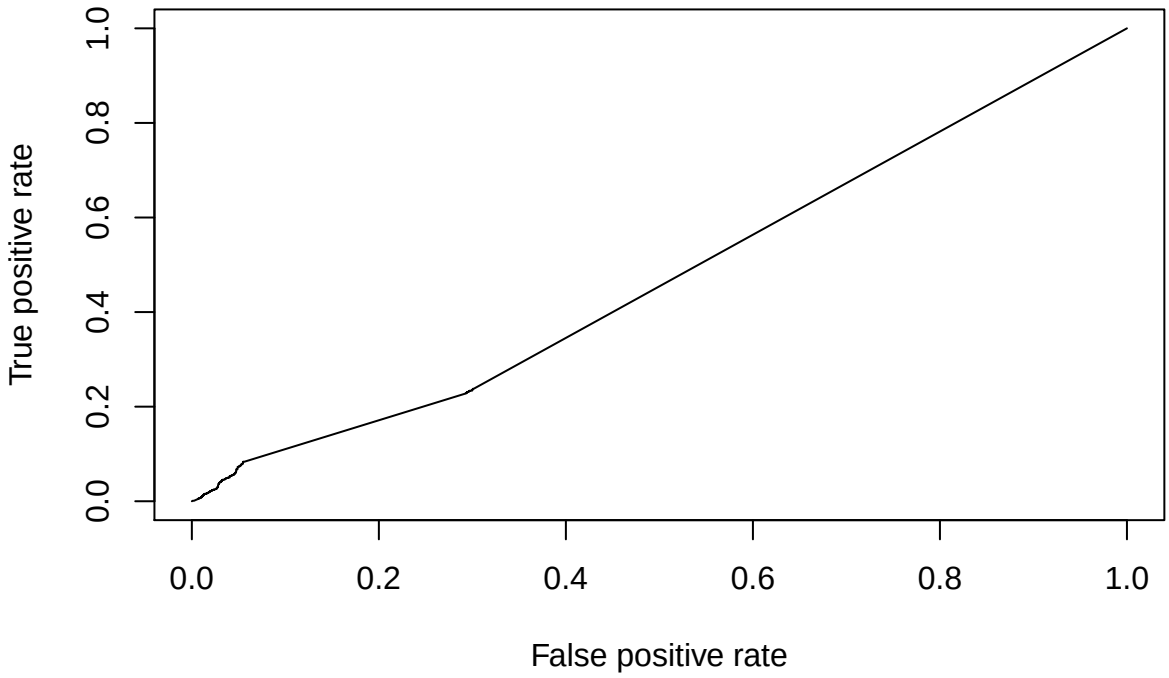
Podemos observar que los errores son casi iguales y se diferencian muy poco. Esto es debido a que por la necesidad de eficiencia en tiempo hemos aunmentado mucho el valor de la variable threshold, cuyo valor mejoraría bastante el error con un [0,1,0.01], lo que creemos que repercute en el error que comete. Dado que las redes neuronales necesitan mas tiempo de entrenamiento que el que les hemos podido proporcionar. Por último creemos que los errores son muy parecidos entre los modelos puede ser debido, además de que el umbral threshold es muy alto, las capas ocultas son muy parecidas.

En este caso, el mejor modelo es el tercero, ya que es el que tiene una cantidad muy pequeña menos de error y consume menos tiempo de ejecución. Ahora calcularemos su curva ROC.

```
probsnn = compute(nn3, testTransformado[names(testTransformado) %in% n])
probsnn.pred = probsnn$net.result
nn.model = ROCR::prediction(probsnn.pred, testTransformado$default.payment.next.month)
perfnn = performance(nn.model, "tpr", "fpr")
auc.nn = performance(nn.model, measure = "auc")
print(auc.nn@y.values[[1]])
```

```
## [1] 0.4732142838
```

```
plot(perfnn)
```



Modelos con clasificador SVM.

Para utilizar el clasificador SVM, utilizaremos el paquete *e1071*, dentro de este se encuentra la función svm, esta por defecto utiliza un kernel gaussiano. La fórmula que utilizaremos será la misma que las anteriormente utilizadas. Probaremos con diferentes valores para este parámetro y nos quedaremos con el mejor de ellos.

Después, calcularemos la curva ROC del modeo elegido.

```
svm1 = e1071::svm(formula=default.payment.next.month~PAY_1+PAY_2+BILL_AMT1+LIMIT_BAL+AGE+ed.school+PAY_

pred.svm1 = predict(svm1, trainTransformado)
predTs.svm1 = predict(svm1, testTransformado)
Ein.svm1 = mean(pred.svm1 != trainTransformado$default.payment.next.month)
Eout.svm1 = mean(predTs.svm1 != testTransformado$default.payment.next.month)
Ein.svm1
```

```
## [1] 0.1755238095
```

```
Eout.svm1
```

```
## [1] 0.1814444444
```

```
svm2 = e1071::svm(formula=default.payment.next.month~PAY_1+PAY_2+BILL_AMT1+LIMIT_BAL+AGE+ed.school+PAY_

pred.svm2 = predict(svm2, trainTransformado)
predTs.svm2 = predict(svm2, testTransformado)
Ein.svm2 = mean(pred.svm2 != trainTransformado$default.payment.next.month)
```

```
Eout.svm2 = mean(predTs.svm2 != testTransformado$default.payment.next.month)
Ein.svm2
```

```
## [1] 0.1779047619
```

```
Eout.svm2
```

```
## [1] 0.1818888889
```

```
svm3 = e1071::svm(formula=default.payment.next.month~PAY_1+PAY_2+BILL_AMT1+LIMIT_BAL+AGE+ed.school+PAY_

pred.svm3 = predict(svm3, trainTransformado)
predTs.svm3 = predict(svm3, testTransformado)
Ein.svm3 = mean(pred.svm3 != trainTransformado$default.payment.next.month)
Eout.svm3 = mean(predTs.svm3 != testTransformado$default.payment.next.month)
Ein.svm3
```

```
## [1] 0.1557142857
```

```
Eout.svm3
```

```
## [1] 0.1851111111
```

```
svm4 = e1071::svm(formula=default.payment.next.month~PAY_1+PAY_2+BILL_AMT1+LIMIT_BAL+AGE+ed.school+PAY_

pred.svm4 = predict(svm4, trainTransformado)
predTs.svm4 = predict(svm4, testTransformado)
Ein.svm4 = mean(pred.svm4 != trainTransformado$default.payment.next.month)
Eout.svm4 = mean(predTs.svm4 != testTransformado$default.payment.next.month)
Ein.svm4
```

```
## [1] 0.09571428571
```

```
Eout.svm4
```

```
## [1] 0.2112222222
```

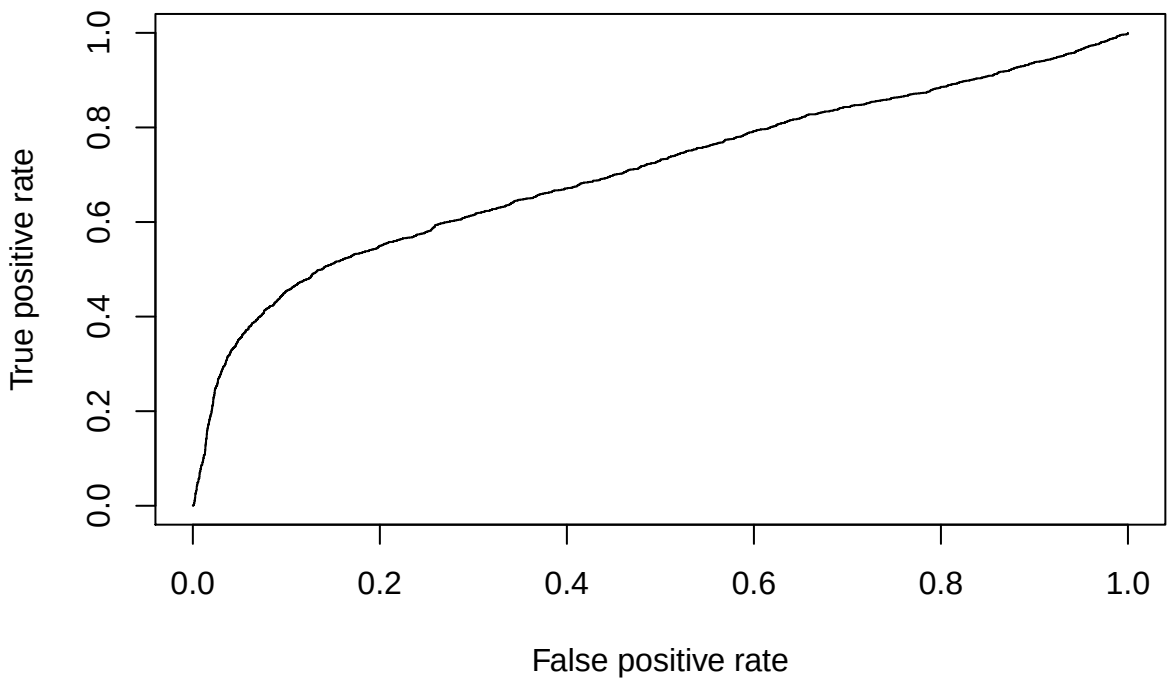
Para estos cuatro modelos, el mejor de todos ellos es el primero, ya que tiene el error fuera de la muestra más bajo. Además, la diferencia entre el error dentro de la muestra y el error fuera de la muestra no es grande por lo que sabemos que el modelo no tiene sobreajuste.

Ahora, calcularemos las probabilidades del modelo que hemos elegido y calcularemos su curva ROC y su área.

```
s.svm = e1071::svm(formula=default.payment.next.month~PAY_1+PAY_2+BILL_AMT1+LIMIT_BAL+AGE+ed.school+PAY
pred.s = predict(s.svm, testTransformado, probability=T)
probs.svm = attr(pred.s, "probabilities")[,1]
svm.model = ROCR::prediction(probs.svm, testTransformado$default.payment.next.month)
perf = performance(svm.model, "tpr", "fpr")
auc.svm = performance(svm.model, measure = "auc")
print(auc.svm@y.values[[1]])
```

```
## [1] 0.7062206547
```

```
plot(perf)
```



Modelos con randomForest

Para los modelos de Random Forest, utilizaremos la función randomForest. Utilizaremos diferentes valores para el número de árboles que usa este método; el número de nodos será siempre el mismo, 400. Una vez hayamos evaluado todos los modelos, elegiremos el que menor error fuera de la muestra cometa y calcularemos la curva ROC de este.

```
n <- c("PAY_1", "PAY_2", "BILL_AMT1", "LIMIT_BAL","AGE","ed.school","PAY_AMT1","PAY_5","marriage.marrie
rf = randomForest::randomForest(x=trainTransformado[, names(trainTransformado)%in%n], y=as.factor(train

pred.rf1 = predict(rf, trainTransformado)
Ein.rf1 = mean( pred.rf1 != trainTransformado$default.payment.next.month)
Ein.rf1
```

```
## [1] 0.1613333333
```

```
predTs.rf1 = predict(rf, testTransformado)
Eout.rf1 = mean( predTs.rf1 != testTransformado$default.payment.next.month)
Eout.rf1
```

```
## [1] 0.1824444444
```

```
rf2 = randomForest::randomForest(x=trainTransformado[, names(trainTransformado)%in%n], y=as.factor(trai

pred.rf2 = predict(rf2, trainTransformado)
```

```
Ein.rf2 = mean( pred.rf2 != trainTransformado$default.payment.next.month)
Ein.rf2
```

```
## [1] 0.161
```

```
predTs.rf2 = predict(rf2, testTransformado)
Eout.rf2 = mean( predTs.rf2 != testTransformado$default.payment.next.month)
Eout.rf2
```

```
## [1] 0.182
```

```
rf3 = randomForest::randomForest(x=trainTransformado[, names(trainTransformado)%in%n], y=as.factor(trai

pred.rf3 = predict(rf3, trainTransformado)
Ein.rf3 = mean( pred.rf3 != trainTransformado$default.payment.next.month)
Ein.rf3
```

```
## [1] 0.1608571429
```

```
predTs.rf3 = predict(rf3, testTransformado)
Eout.rf3 = mean( predTs.rf3 != testTransformado$default.payment.next.month)
Eout.rf3
```

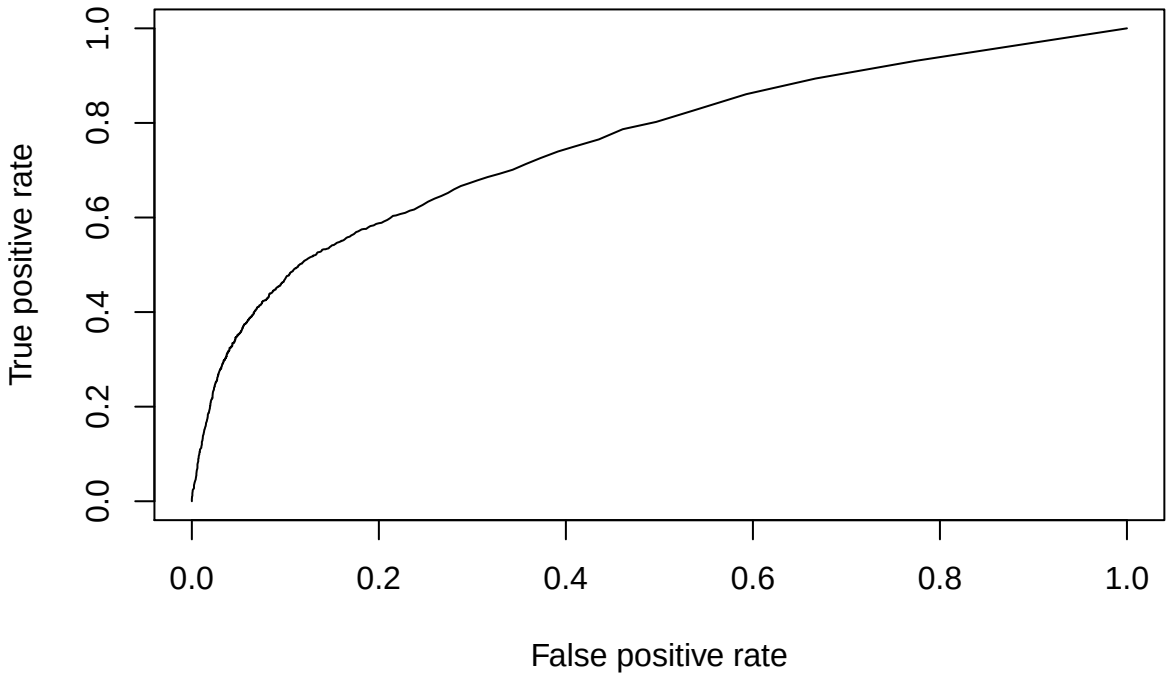
```
## [1] 0.1825555556
```

El primer modelo es el mejor de los 3, ya que comete un error muy parecido al resto de los modelos utilizando menos árboles para ello. Ahora calcularemos la curva ROC y su área debajo de la curva.

```
probs = predict(rf, testTransformado, type="prob")
probs.pred = probs[,2]
rf.model = ROCR::prediction(probs.pred, testTransformado$default.payment.next.month)
perf = performance(rf.model, "tpr", "fpr")
auc.rf = performance(rf.model, measure = "auc")
print(auc.rf@y.values[[1]])
```

```
## [1] 0.7555162531
```

```
plot(perf)
```



Conclusiones.

Los resultados obtenidos por los clasificadores son los siguientes:

Área debajo de la curva ROC Clasificador: Área:
Lineal: 0.7182 Boosting: 0.6797 SVM: 0.7062 Redes Neuronales: 0.5492 Random Forest: 0.7523

Como podemos ver, el clasificador que mejores resultados obtiene es el Random Forest, el siguiente mejor clasificador es el lineal. Por lo tanto el mejor modelo posible es el que utiliza Random Forest, pero el modelo lineal también obtiene bastante buenos resultados, por lo también podríamos considerar utilizar este modelo ya que es más sencillo.

Además, creemos que un modelo de redes neuronales con los parámetros adecuados, y con la el tiempo suficiente para entrenarlas; podría obtener unos resultados parecidos a los que se obtienen con el resto de los modelos.