PREDICCIÓN

PRÁCTICA 2

LOAN-STATS

Ana Arias Botey

Índice

1. Introducción

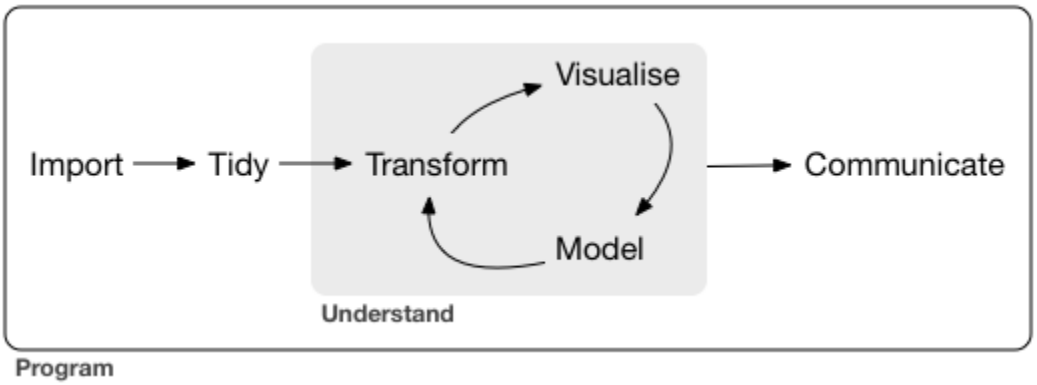
Ésta práctica consiste en realizar un análisis de regresión logística con el objetivo de predecir si los préstamos concedidos han sido pagados (fully paid-1) o si por el contrario, no han sido pagados (charged off-0).

Para realizar éste análisis se ha tenido en cuenta el tratamiento de datos y las explicaciones que se realizan en el artículo *"Evaluating Credit Risk and loan performance in online Peer-to-Peer"*. Como el análisis que se lleva a cabo en éste artículo es durante el periodo 2007-2012, se tomará la base de datos 2007-2011 de la página web https://www.lendingclub.com/info/download-data.action (loan data).

¿Qué es Peer-to-Peer?

El *P2P* establece una conexión directa entre ordenadores, sin necesidad de un servicio intermedio, siendo una serie de nodos que se comportan como iguales entre sí. Es decir, actúan simultáneamente como clientes y servidores respecto a los demás nodos de la red. Las redes P2P permiten el intercambio directo de información, en cualquier formato, entre los ordenadores interconectados. Dichas redes son útiles para diversos propósitos: En el artículo, éste métdo se emplea para préstamos donde intervienen prestamistas y prestatarios.

Pasos a seguir en el análisis:



2. Importación

Primero descargo la base de datos de loanStats para el año 2007-2011 de la página web https://www.lendingclub.com/info/download-data.action.

El fichero .csv tiene 145 variables con un total de 42585 observaciones.

```
rm(list=ls())

setwd("C:/Users/usuario/Desktop/prediccion/MDSF_Prediccion-
master/Clase03/practica2prediccion")
loadstats<-read.csv("LoanStats3a.csv", header=T, sep="|", dec=".", fill =
T)

head(loadstats$grade) #grado del riesgo del cr?dito
```

```
## [1] B C C C B A
## Levels:  0 1 2 A B C D E F G
```

- **variable dependiente:** "loan_status" . Tomará el valor 1 si es FullyPaid y 0 en el caso de charged off.
- **variables independientes o explicativas:** el resto de variables

3. Limpieza de datos

variable dependiente: loan_status

En un análisis de regresión logística, es necesario que la variable dependiente sea dicotómica, es decir que esté compuesta por unos y ceros. Para ello, se clasificarán los préstamos pagados (fullypaid) con un 1 y los no pagados (charged off) con un 0.

Para ver qué contiene la variable, se analiza de la siguiente forma:

```
plot(loadstats$loan_status)
```

Existe alguna observación (en total no llegan a 50 de las 43000) , que viene expresada con un dato que no es ni fully paid ni charged off (f, "", Aug-2010,...).

Debido a que éstas variables no son demasiadas, he optado por eliminarlas:

```
table(loadstats$loan_status)
```

```
##
##
##                                                  119
##                                             Aug-2010
##                                                    1
##                                           Charged Off
##                                                 5661
##                                             Dec-2010
##                                                    2
##                                             Dec-2011
##                                                    1
## Does not meet the credit policy. Status:Charged Off
##                                                  761
##  Does not meet the credit policy. Status:Fully Paid
##                                                 1983
##                                                    f
##                                                    7
##                                             Feb-2011
##                                                    1
##                                          Fully Paid
##                                                34042
##                                             Jul-2010
##                                                    1
##                                             Mar-2011
##                                                    2
##                                             May-2011
##                                                    1
##                                             Nov-2011
##                                                    1
##                                             Oct-2010
##                                                    1
##                                             Sep-2011
##                                                    1

loadstats<-loadstats[loadstats$loan_status!="f",]
loadstats<-loadstats[loadstats$loan_status!="",]
loadstats<-loadstats[loadstats$loan_status!="Aug-2010",]
loadstats<-loadstats[loadstats$loan_status!="Dec-2010",]
loadstats<-loadstats[loadstats$loan_status!="Dec-2011",]
loadstats<-loadstats[loadstats$loan_status!="Feb-2011",]
loadstats<-loadstats[loadstats$loan_status!="Jul-2010",]
loadstats<-loadstats[loadstats$loan_status!="Mar-2011",]
loadstats<-loadstats[loadstats$loan_status!="May-2011",]
loadstats<-loadstats[loadstats$loan_status!="Nov-2011",]
loadstats<-loadstats[loadstats$loan_status!="Oct-2010",]
loadstats<-loadstats[loadstats$loan_status!="Sep-2011",]
table(loadstats$loadstats.loan_status)

## < table of extent 0 >
```

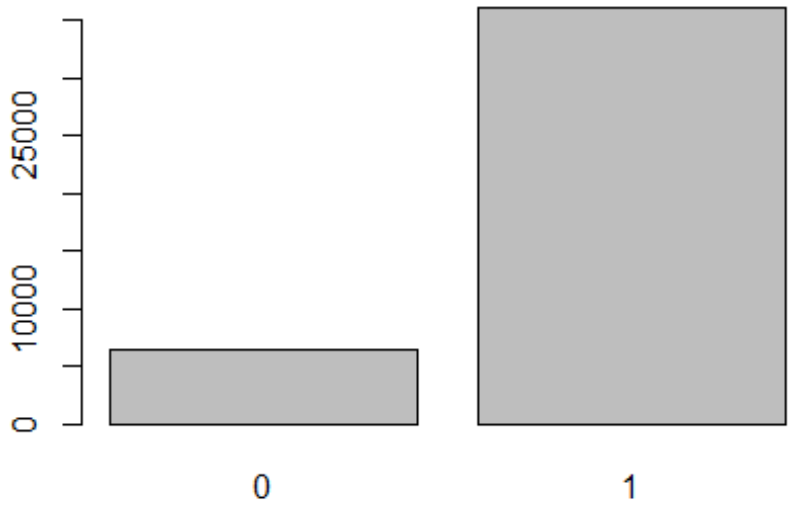
```r
loadstats$loan_status = gsub("Does not meet the credit policy.
Status:Charged Off", "Charged Off",loadstats$loan_status)
loadstats$loan_status = gsub("Does not meet the credit policy.
Status:Fully Paid", "Fully Paid",loadstats$loan_status)



FullyPaid<-loadstats$loan_status
table(FullyPaid)
```

```
## FullyPaid
## Charged Off  Fully Paid
##        6422       36025
```

```r
#plot(table(FullyPaid))
class(FullyPaid)
```

```
## [1] "character"
```

```r
FullyPaid <- factor(FullyPaid, labels=0:1)
plot(FullyPaid)
```



En el artículo "Evaluating Credit Risk and loan performance in online Peer-to-Peer" redacta que se escogen 13 variables a priori de interés en la muestra.

De éstas 13, para la estimación final del modelo de regresión empleará 4, ya que el resto no son significativas en el modelo. Se han intentado obtener estas 13 variables para seguir los mismos pasos que en el pdf, sin embargo, la variable FICO, que en el artículo es una variable cualitativa, no se proporciona en nuestros datos. Del resto de variables, algunas vienen con NA.

Por tanto, teniendo en cuenta las variables escogidas en el artículo y bajo mi criterio de cuáles podrían influir en la varible dependiente, cogeré las siguientes: dti, grade, int_rate, revol_util, FullyPaid(=loan_status), annual_inc, total_acc, loan_amnt, total_pymnt, total_rec_int.

```
loanStats1<-
data.frame(loadstats$dti,loadstats$grade,loadstats$int_rate,loadstats$rev
ol_util, FullyPaid, loadstats$annual_inc,loadstats$total_acc,
loadstats$loan_amnt, loadstats$total_pymnt, loadstats$total_rec_int)

head(loanStats1)
```

```
##   loadstats.dti loadstats.grade loadstats.int_rate
loadstats.revol_util
## 1         27.65               B             10.65%
83.7%
## 2             1               C             15.27%
9.4%
## 3          8.72               C             15.96%
98.5%
## 4            20               C             13.49%
21%
## 5         17.94               B             12.69%
53.9%
## 6          11.2               A              7.90%
28.3%
##   FullyPaid loadstats.annual_inc loadstats.total_acc
loadstats.loan_amnt
## 1         1                24000                   9
5000
## 2         0                30000                   4
2500
## 3         1                12252                  10
```

```
2400
## 4         1                49200                  37
10000
## 5         1                80000                  38
3000
## 6         1                36000                  12
5000
##   loadstats.total_pymnt loadstats.total_rec_int
## 1       5863.1551866952                  863.16
## 2               1014.53                  435.17
## 3       3005.6668441393                  605.67
## 4    12231.890000000902                 2214.92
## 5       4066.9081610817                 1066.91
## 6     5632.209999999401                  632.21
```

En R es importante saber qué tipo de objeto es cada variable, puesto que la mayoría de funciones lo tienen en cuenta. Para ver el tipo de un objeto, se puede utilizar la función class. Utilizamos la función sapply que toma como argumento el data.frame, y a cada elemento del data.frame (variables) le aplica la función que especifiquemos.

```
sapply(loanStats1, class)
```

```
##           loadstats.dti         loadstats.grade
loadstats.int_rate
##                "factor"                "factor"
"factor"
##    loadstats.revol_util               FullyPaid
loadstats.annual_inc
##                "factor"                "factor"
"factor"
##     loadstats.total_acc     loadstats.loan_amnt
loadstats.total_pymnt
##                "factor"                "factor"
"factor"
## loadstats.total_rec_int
##               "numeric"
```

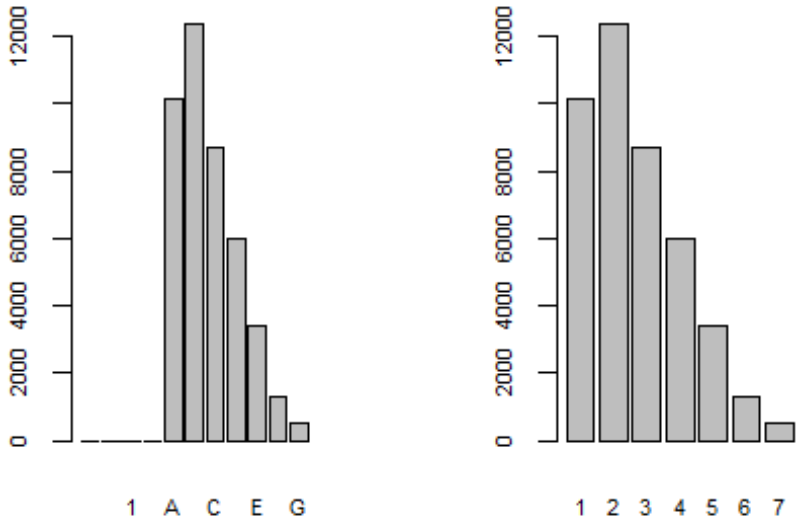
Es necesario cambiar los tipos de variables para el correcto análisis que tenemos, ya que la mayoría se han definido, por defecto como factor y sin embargo no lo son.

Variable grado

En el artículo, se convierte la variable grado (formada por 7 tipos de grado: A, B, C, D, E, F, G) a una variable cualitativa del 1 al 7. Por lo que realizaré el mismo procedimiento.

Primero, elimino las observaciones que no tienen ninguno de éstos tipos de grados ("", 1, 2, 0), para después poner la variable grado como variable cualitativa.

```
loanStats1<-loanStats1[loanStats1$loadstats.grade!="",]
loanStats1<-loanStats1[loanStats1$loadstats.grade!=1,]
loanStats1<-loanStats1[loanStats1$loadstats.grade!=2,]
loanStats1<-loanStats1[loanStats1$loadstats.grade!=0,]

par(mfrow=c(1,2))
barplot(table(loanStats1$loadstats.grade), cex.names = 0.7, cex.axis =
0.7)

loanStats1$loadstats.grade <- factor(loanStats1$loadstats.grade,
labels=1:7)
levels(loanStats1$loadstats.grade)

## [1] "1" "2" "3" "4" "5" "6" "7"

head(loanStats1$loadstats.grade)

## [1] 2 3 3 3 2 1
## Levels: 1 2 3 4 5 6 7

barplot(table(loanStats1$loadstats.grade), cex.names = 0.7, cex.axis =
0.7)
```



Variable int_rate y revol_util

Es necesario quitar el símbolo de tanto por ciento y dividir los valores entre cien, para convertirlos en porcentajes. Posteriormente, se clasficarán en variables tipo numéricas.

```
typeof(loanStats1$loadstats.int_rate)

## [1] "integer"

loanStats1$loadstats.int_rate = gsub("%",
"",loanStats1$loadstats.int_rate)
head(loanStats1$loadstats.int_rate)

## [1] " 10.65" " 15.27" " 15.96" " 13.49" " 12.69" "  7.90"

loanStats1$loadstats.revol_util= gsub("%",
"",loanStats1$loadstats.revol_util)
head(loanStats1$loadstats.revol_util)

## [1] "83.7" "9.4"  "98.5" "21"   "53.9" "28.3"
```

Pongo la clase correcta a todas las variables

```
loanStats1$loadstats.dti<-as.numeric(paste(loanStats1$loadstats.dti))

## Warning: NAs introducidos por coerción

loanStats1$loadstats.int_rate<-
as.numeric(paste(loanStats1$loadstats.int_rate))/100
loanStats1$loadstats.revol_util<-
as.numeric(paste(loanStats1$loadstats.revol_util))/100
loanStats1$loadstats.annual_inc<-
as.numeric(paste(loanStats1$loadstats.annual_inc))
loanStats1$loadstats.total_acc<-
as.numeric(paste(loanStats1$loadstats.total_acc))

## Warning: NAs introducidos por coerción

loanStats1$loadstats.total_pymnt<-
as.numeric(paste(loanStats1$loadstats.total_pymnt))

## Warning: NAs introducidos por coerción

loanStats1$loadstats.loan_amnt<-
as.numeric(paste(loanStats1$loadstats.loan_amnt))
loanStats1$loadstats.total_rec_int<-
as.numeric(paste(loanStats1$loadstats.total_rec_int))

## Warning: NAs introducidos por coerción

loanStats1$FullyPaid<-as.numeric(paste(loanStats1$FullyPaid))
```

Por último, veamos un resúmen de los datos con summary:

```
sapply(loanStats1, class)

##           loadstats.dti         loadstats.grade
loadstats.int_rate
##               "numeric"                "factor"
"numeric"
##    loadstats.revol_util              FullyPaid
loadstats.annual_inc
##               "numeric"              "numeric"
"numeric"
##     loadstats.total_acc     loadstats.loan_amnt
loadstats.total_pymnt
##               "numeric"              "numeric"
"numeric"
## loadstats.total_rec_int
##               "numeric"
```

```r
loanStats1<-na.omit(loanStats1)
summary(loanStats1)
```

```
##  loadstats.dti   loadstats.grade loadstats.int_rate
loadstats.revol_util
##  Min.   : 0.00   1:10144         Min.   :0.0542     Min.   :0.0000
##  1st Qu.: 8.21   2:12336         1st Qu.:0.0962     1st Qu.:0.2570
##  Median :13.48   3: 8688         Median :0.1199     Median :0.4969
##  Mean   :13.38   4: 5976         Mean   :0.1216     Mean   :0.4911
##  3rd Qu.:18.69   5: 3365         3rd Qu.:0.1472     3rd Qu.:0.7270
##  Max.   :29.99   6: 1289         Max.   :0.2459     Max.   :1.1900
##                  7:  509
##    FullyPaid     loadstats.annual_inc loadstats.total_acc
##  Min.   :0.000   Min.   :   1896      Min.   : 1.00
##  1st Qu.:1.000   1st Qu.:  40000      1st Qu.:13.00
##  Median :1.000   Median :  59000      Median :20.00
##  Mean   :0.849   Mean   :  69164      Mean   :22.14
##  3rd Qu.:1.000   3rd Qu.:  82500      3rd Qu.:29.00
##  Max.   :1.000   Max.   :6000000      Max.   :90.00
##
##  loadstats.loan_amnt loadstats.total_pymnt loadstats.total_rec_int
##  Min.   :  500       Min.   :    0         Min.   :    0.0
##  1st Qu.: 5200       1st Qu.: 5472         1st Qu.:  657.8
##  Median : 9725       Median : 9693         Median : 1339.5
##  Mean   :11095       Mean   :12027         Mean   : 2241.8
##  3rd Qu.:15000       3rd Qu.:16429         3rd Qu.: 2804.2
##  Max.   :35000       Max.   :58886         Max.   :23886.5
##
```

4. Estimación del modelo

Creo una muestra de entrenamiento y otra de validación:

```
set.seed(1234)
n=nrow(loanStats1)
id_train <- sample(1:n , 0.9*n)
credit.train = loanStats1[id_train,]
credit.test = loanStats1[-id_train,]
plot(table(credit.train$FullyPaid))
```

Estimo un primer modelo:

```
library(glmnet)

credit.glm0<-
glm(FullyPaid~loadstats.grade+loadstats.dti+loadstats.annual_inc+loadstat
s.total_pymnt+loadstats.loan_amnt,family=binomial,credit.train)

summary(credit.glm0)

##
## Call:
## glm(formula = FullyPaid ~ loadstats.grade + loadstats.dti +
loadstats.annual_inc +
##     loadstats.total_pymnt + loadstats.loan_amnt, family = binomial,
##     data = credit.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -6.0057   0.0684   0.2125   0.3543   5.1329
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)             2.635e+00  7.891e-02  33.395  < 2e-16 ***
## loadstats.grade2       -1.027e+00  6.872e-02 -14.942  < 2e-16 ***
## loadstats.grade3       -1.540e+00  7.066e-02 -21.801  < 2e-16 ***
## loadstats.grade4       -2.089e+00  7.528e-02 -27.747  < 2e-16 ***
## loadstats.grade5       -2.386e+00  9.087e-02 -26.252  < 2e-16 ***
## loadstats.grade6       -3.045e+00  1.330e-01 -22.901  < 2e-16 ***
## loadstats.grade7       -2.851e+00  2.087e-01 -13.661  < 2e-16 ***
## loadstats.dti          -1.196e-02  3.161e-03  -3.783 0.000155 ***
## loadstats.annual_inc    3.169e-06  6.174e-07   5.134 2.84e-07 ***
## loadstats.total_pymnt   7.638e-04  1.073e-05  71.200  < 2e-16 ***
```

```
## loadstats.loan_amnt   -6.953e-04  1.001e-05 -69.432  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 32358  on 38075  degrees of freedom
## Residual deviance: 15379  on 38065  degrees of freedom
## AIC: 15401
##
## Number of Fisher Scoring iterations: 7
```

Los coeficientes del modelo los muestra en formato tabular, añadiendo el error estándar, y el valor z (distr normal) que es el coeficiente dividido por el error.

Se muestran los símbolos "*" y ".", que indican la significación de los parámetros a diferentes niveles. A un nivel de significación 0,05 los parámetros asociados a las variables con "*" son significativamente distintos de 0, ya que el valor del estadístico de Wald es en valor absoluto mayore que el punto crítico z/2 = 1,96.

En éste caso, se puede observar como todas las variables empleadas son significativas. La variable cualitativa que se ha considerado: grado, es significativa para todos sus niveles.

Por último se muestra la devianza del modelo nulo (null deviance) y del modelo ajustado (Residual deviance), con sus respectivos grados de libertad, así como el valor del AIC (Criterio de información de Akaike).

Para acceder a los resultados del modelo, primero utilizo la función names() para saber cómo llamarlos:

```
names(credit.glm0)
```

```
##  [1] "coefficients"      "residuals"         "fitted.values"
##  [4] "effects"           "R"                 "rank"
##  [7] "qr"                "family"            "linear.predictors"
## [10] "deviance"          "aic"               "null.deviance"
## [13] "iter"              "weights"           "prior.weights"
## [16] "df.residual"       "df.null"           "y"
```

```
## [19] "converged"          "boundary"          "model"
## [22] "call"               "formula"           "terms"
## [25] "data"               "offset"            "control"
## [28] "method"             "contrasts"         "xlevels"
```

Los coeficientes del modelo 1 estimado son:

```
credit.glm0$coefficients

##           (Intercept)      loadstats.grade2      loadstats.grade3
##          2.635137e+00         -1.026770e+00         -1.540424e+00
##      loadstats.grade4      loadstats.grade5      loadstats.grade6
##         -2.088736e+00         -2.385573e+00         -3.044973e+00
##      loadstats.grade7         loadstats.dti  loadstats.annual_inc
##         -2.850872e+00         -1.195882e-02          3.169457e-06
## loadstats.total_pymnt   loadstats.loan_amnt
##          7.638494e-04         -6.953449e-04
```

$$\text{logit}[p(x)] = \ln\left[\frac{p(x)}{1} - p(x)\right]$$

$$= 2.635137e+00 - 1.026770e+00 * \text{grade2} - 1.540424e+00 * \text{grade3} - 2.088736e+00 * \text{grade4} - 2.385573e+00 * \text{grade5} - 3.044973e+00 * \text{grade6} - 2.850872e+00 * \text{grade7} - 1.195882e-02 * \text{dti} + 3.169457e-06 * \text{annual}_{\text{inc}} + 7.638494e-04 * \text{totalpymnt} - 6.953449e-04 * \text{loan\_amnt}$$

En cuanto a los coeficientes, la interpretación cambia respecto a un modelo de regresión lineal (lm). El modelo GLM no ajusta la variable respuesta sino una función de enlace. En el caso del modelo logit esta función es:

$$\text{logit}[p(x)] = \ln\left[\frac{p(x)}{1} - p(x)\right]$$

siendo p la probabilidad de que el individuo tome el valor "1" en la variable dicotómica.

Por tanto, para hallar la probabilidad de cada observación es necesitario hacer:

$$p(x) = \frac{1}{(1 - e^{-x})}$$

Las variables de grado toman el valor 1 para los individuos de ese grado (grado 1, grado 2,..., grado 7) y 0 para los que no pertenezcan a ese grado. Para los individuos del grado 1 el coeficiente del intercept es el logit, ya que como se puede observar en el summary del modelo, no aparece el grado 1. Ésto es debido a que en las vaiables cualitativas siempre aparecen todas las variables menos una.

```
# función invlogit para pasar de logit a probabilidades
invlogit <- function(x) {
1/(1 + exp(-x))
}
# aplicamos la funci?n invlogit al primer coeficiente del modelo
invlogit(coef(credit.glm0)[1])
```

```
## (Intercept)
##    0.933089
```

5. Contraste sobre los parámetros

Una vez estimado el modelo, nos interesa contrastar si los coeficientes estimados son significativamente distintos de 0. Es decir, si una determinada variable explicativa tiene un efecto significativo sobre la respuesta o no. Se utilizan los contrastes de hipótesis sobre los parámetros.

5.1. Contraste condicional de razón de verosimilitud

Las hipótesis de este contraste son:

$$H0 : \beta r = 0$$

$$H1 : \beta r \neq 0$$

Se comparan dos modelos: uno dónde se haya estimado el parámetro βr, frente a otro modelo que se diferencie del primero en que no esté dicho parámetro, pero si el resto de los parámetros. Es decir, comparamos modelos anidados. El test que se utiliza es el test condicional de razón de verosimilitudes.

Se empleará la función anova del paquete car, la cual realiza los contrastes condicionales de razón de verosimilitud sobre los parámetros asociados a cada una de las variables del modelo, sin necesidad de especificar los distintos modelos.

```
library(car)

#anova(modelo1, modelo2, test = "Chisq") #anova sin utilizar paquete car
Anova(credit.glm0) #utilizando paquete car

## Analysis of Deviance Table (Type II tests)
##
## Analysis of Deviance Table (Type II tests)
##
## Response: FullyPaid
##                         LR Chisq Df Pr(>Chisq)
## loadstats.grade          1485.1  6  < 2.2e-16 ***
## loadstats.dti              14.3  1  0.0001526 ***
## loadstats.annual_inc       28.6  1  8.902e-08 ***
## loadstats.total_pymnt   15233.2  1  < 2.2e-16 ***
## loadstats.loan_amnt     10870.0  1  < 2.2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Se observa como todas las variables elegidas son altamente significativas en nuestro modelo.

6. Intervalos de confianza para los parámetros

Los intervalos de confianza más que dar una respuesta positiva o negativa sobre si un determinado parámetro es significativo, calculan un intervalo de valores plausibles.

6.1. Intervalos de confianza para los parámetros basados en el test de Wald

Se calculan con la función confint.default, que toma por defecto un nivel de confianza $1-\alpha = 0{,}95$. Pero en clase hemos visto que éste intervalo ed confianza es muy grande, por lo se se utilizará level=0.75.

Si el intervalo de confianza incluye el 0, significa que al nivel α elegido no se podría rechazar la hipótesis nula de que $\beta r = 0$

```
confint.default(credit.glm0, level = (0.75))
```

```
##                               12.5 %        87.5 %
## (Intercept)             2.544366e+00  2.725908e+00
## loadstats.grade2       -1.105821e+00 -9.477195e-01
## loadstats.grade3       -1.621705e+00 -1.459143e+00
## loadstats.grade4       -2.175331e+00 -2.002141e+00
## loadstats.grade5       -2.490108e+00 -2.281037e+00
## loadstats.grade6       -3.197928e+00 -2.892018e+00
## loadstats.grade7       -3.090932e+00 -2.610812e+00
## loadstats.dti          -1.559515e-02 -8.322491e-03
## loadstats.annual_inc    2.459259e-06  3.879655e-06
## loadstats.total_pymnt   7.515082e-04  7.761907e-04
## loadstats.loan_amnt    -7.068653e-04 -6.838245e-04
```

6.2. Intervalos de confianza para los e^r

Debido a la interpretación de los parámetros en los modelos de regresión logística, se suelen calcular los intervalos de confianza para los exponenciales de los parámetros, que se corresponden con los cocientes de ventajas. En estos casos, el contraste asociado se define como

$$H0 : e^{\beta r} = 1$$

$$H1 : e^{\beta r} \neq 1$$

Y los intervalos de confianza se obtienen a partir de los intervalos anteriores, sin más que aplicarles la función e x , el cálculo para los intervalos de confianza de los e βr sería

```
exp(confint.default(credit.glm0, level = 0.75))
```

```
##                              12.5 %      87.5 %
## (Intercept)           12.73515335 15.27027895
## loadstats.grade2       0.33093907  0.38762401
## loadstats.grade3       0.19756157  0.23243545
## loadstats.grade4       0.11357057  0.13504581
## loadstats.grade5       0.08290097  0.10217823
## loadstats.grade6       0.04084677  0.05546416
## loadstats.grade7       0.04545959  0.07347485
## loadstats.dti          0.98452583  0.99171205
## loadstats.annual_inc   1.00000246  1.00000388
## loadstats.total_pymnt  1.00075179  1.00077649
## loadstats.loan_amnt    0.99929338  0.99931641
```

Tanto en el anterior contraste de los betas, como en éste de los exponentes de los betas, se concluye que ninguno de los parámetros contiene ni al cero (para beta) ni al uno (para éste).

Por tanto, se conclute que ninguna de las variables que se han escogido para el modelo son prescindibles.

Ahora, guardo en variables los residuos para después poderlos utilizar en los sucesivos tests y gráficamente veo cómo se comportan:

```
res.p <- residuals(credit.glm0, type = "pearson")
head(res.p)
```

```
##        4815       26355       25803       26400       36491
27119
## 0.001696087 0.198067821 0.167224320 0.423428521 0.214256560
0.102438726
```

```
res.p.std <- rstandard(credit.glm0, type = "pearson")
head(res.p.std)
```

```
##        4815       26355       25803       26400       36491
27119
## 0.001696087 0.198083849 0.167235510 0.423508830 0.214278726
0.102441167
```

```
res.d <- residuals(credit.glm0, type = "deviance")
head(res.d)
```

```
##        4815       26355       25803       26400       36491
27119
## 0.002398627 0.277419839 0.234862176 0.574306564 0.299610996
0.144492325
```

```
res.dev.std <- rstandard(credit.glm0, type = "deviance")
head(res.dev.std)
```

```
##        4815       26355       25803       26400       36491
27119
## 0.002398627 0.277442288 0.234877893 0.574415489 0.299641992
0.144495768
```

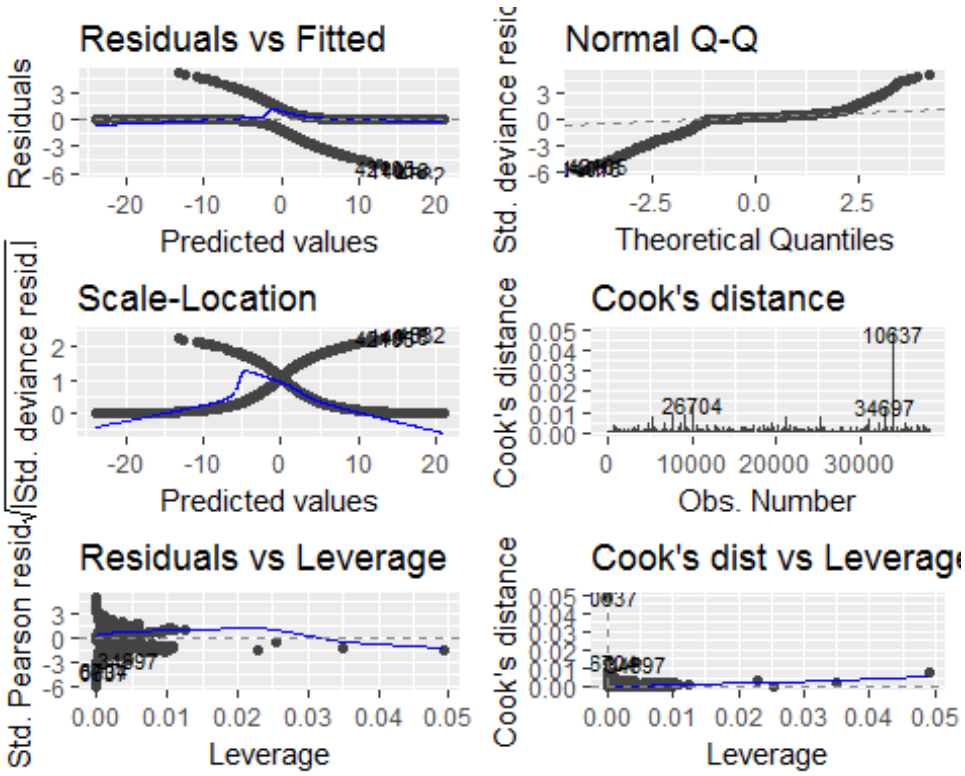
```
#install.packages("ggfortify")
library(ggfortify)
```

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

```
## Warning: package 'ggplot2' was built under R version 3.4.2
```

```
autoplot(credit.glm0, which = 1:6, label.size = 3)
```

```
## Warning: package 'bindrcpp' was built under R version 3.4.2
```


Residuals vs Fitted
Residuals
Predicted values
Normal Q-Q
Std. deviance resic
Theoretical Quantiles
Scale-Location
√|Std. deviance resid.|
Predicted values
Cook's distance
Cook's distance
10637
26704
34697
Obs. Number
Residuals vs Leverage
Std. Pearson resid.
Leverage
Cook's dist vs Leverage
Cook's distance
Leverage

7. Medidas de bondad del ajuste

Cuando los datos están en forma binaria, una manera de detectar la falta de ajuste es realizando contrastes condicionales de razón de verosimilitudes entre modelos anidados.

Otra opción es utilizar el estadístico de Hosmer-Lemeshow, que realiza una partición de los datos en base a las probabilidades predichas, y analiza posteriormente la tabla de contingencia resultante a través de su estadístico X^2 asociado. Otras analizan el poder predictivo del modelo mediante la tasa de clasificaciones correctas o el análisis de curvas ROC.

7.1. Estadístico G^2 de Wilks de razón de verosimilitudes.

El resultado deseado en éste test es si se rechaza o no la hipótesis nula de que el modelo se ajusta globalmente bien a los datos.

```
#grados de libertad
credit.glm1$df.residual
## [1] 38065
```

A pesar de que aparece en el summary del modelo credit.glm0, la forma de calcular la desviacion de los residuos es:

```
(residuos.deviance <- sum(residuals(credit.glm1, type = "deviance")^2))
## [1] 30650.46  # se observa que sale el mismo resultado que en el
                        # summary
pchisq(residuos.deviance, 38065)
## [1] 0.000000637
1 - pchisq(residuos.deviance, 38065)
## [1] 1
```

Al ser mayor de 0.05 (para un nivel de confianza del 95 %), no se rechaza hipótesis nula de que el modelo se ajusta globalmente bien a los datos (el coeficiente *pchisq(residuos.deviance, 38097)* es un número muy cercano a cero y por eso el resultado de la diferencia lo considera como 1).

7.2. Contraste basado en el estadístico de Hosmer-Lemeshow (utilizado en el artículo)

Se crean grupos de la variable respuesta en base a las probabilidades estimadas por el modelo, y se comparan las frecuencias de éxito observadas con las estimadas, mediante el estadístico usual X^2 de Pearson. Se cran 10 grupos eligiendo los puntos de corte de las probabilidades estimadas en intervalos de igual amplitud.

```
yhat.corte <- cut(yhat, breaks = 10, include.lowest = TRUE)
# los intervalos tienen igual amplitud pero el n?mero de individuos en
# cada uno var?a
table(yhat.corte)

## yhat.corte
## [-0.001,0.1]    (0.1,0.2]    (0.2,0.3]    (0.3,0.4]    (0.4,0.5]
##         2364          558          438          363          423
##    (0.5,0.6]    (0.6,0.7]    (0.7,0.8]    (0.8,0.9]      (0.9,1]
##          452          715         1343         4374        27046

a<-fitted.values(credit.glm0)
a<-a[a<=0]
a  ###observación: aunque el primer intervalo sea entre -0.001,0.1, se
#comprueba que no existe ninguna probabilidad menor que uno, de lo
#contrario, sería una incongruencia ya que las probabilidades nunca
#pueden ser menores que cero!

## NULL
```

La función cut divide los valores de una variable numérica en intervalos, y con el argumento breaks divide la variable en intervalos de igual amplitud. El resultado de aplicar cut a un vector numérico es una variable categórica (factor en R).

La tabla de los valores observados. Las columnas V1 e y son respectivamente el número de individuos en cada grupo que tienen valor fullyPaid=0 y Fullypaid=1.

```
y <- credit.train$FullyPaid
(obs <- xtabs(cbind(1 - y, y) ~ yhat.corte))
```

```
##
## yhat.corte         V1     y
##   [-0.001,0.1]  2239   125
##   (0.1,0.2]      470    88
##   (0.2,0.3]      365    73
##   (0.3,0.4]      292    71
##   (0.4,0.5]      315   108
##   (0.5,0.6]      291   161
##   (0.6,0.7]      343   372
##   (0.7,0.8]      409   934
##   (0.8,0.9]      469  3905
##   (0.9,1]        567 26479
```

La tabla de valores esperados

```
(expect <- xtabs(cbind(1 - yhat, yhat) ~ yhat.corte))
```

```
##
## yhat.corte              V1        yhat
##   [-0.001,0.1]  2313.80768    50.19232
##   (0.1,0.2]      476.30075    81.69925
##   (0.2,0.3]      329.28881   108.71119
##   (0.3,0.4]      236.02956   126.97044
##   (0.4,0.5]      232.98079   190.01921
##   (0.5,0.6]      202.58943   249.41057
##   (0.6,0.7]      246.89286   468.10714
##   (0.7,0.8]      325.89299  1017.10701
##   (0.8,0.9]      609.85072  3764.14928
##   (0.9,1]        786.36641 26259.63359
```

Aplicanto el test de hoslem se observa que se acepta la hipótesis nula:

```
library(ResourceSelection)

## Warning: package 'ResourceSelection' was built under R version 3.4.2



## ResourceSelection 0.3-2   2017-02-28

hl <- hoslem.test(credit.train$FullyPaid, fitted(credit.glm0), g=10)
hl

##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  credit.train$FullyPaid, fitted(credit.glm0)
## X-squared = 1649.7, df = 8, p-value < 2.2e-01
```

Por tanto, el modelo se ajusta globalmente a los datos.

7.3. Medidas basadas en la tabla de clasificación. Curvas ROC

```
prediccion <- ifelse(fitted.values(credit.glm1) >= 0.5, 1, 0)


table(credit.train$FullyPaid, prediccion)

##    prediccion
##         0     1
##   0  3681  2079
##   1   465 31851

tabla.clasif <- table(credit.train$FullyPaid, prediccion)
tcc <- 100 * sum(diag(tabla.clasif))/sum(tabla.clasif)
tcc

## [1] 93.31863
```

Si elegimos como punto de corte p = 0,5 el modelo clasifica correctamente al 93.31863% de los individuos. Además, se bserva en la tabla que de 4146 ceros (charged off- préstamos no devueltos), 3681 se clasifican bien que son un porcentaje

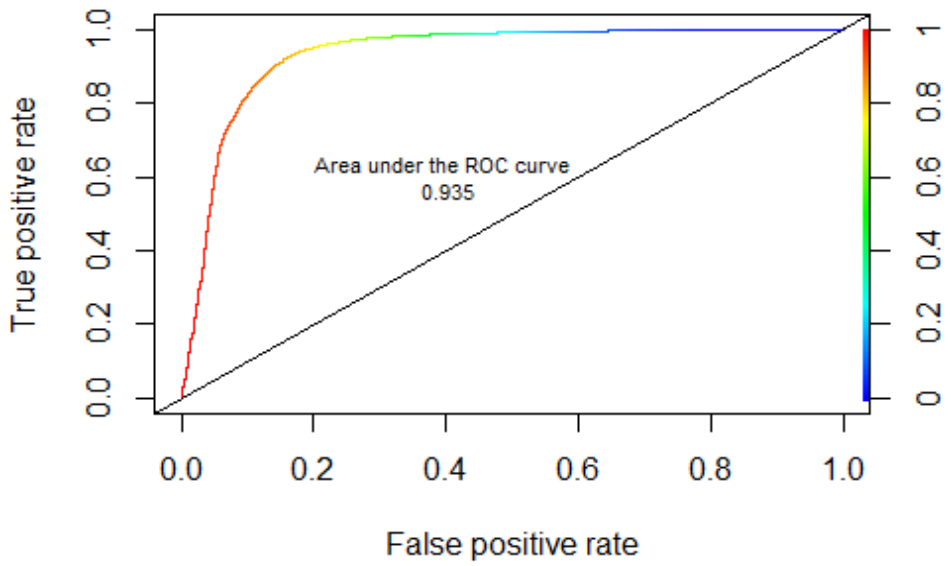
de 88.9% y de 33940 unos (prestamos pagados-fully paid) 31851 son clasificados bien (93.84%)

```
library(ROCR)

## Warning: package 'ROCR' was built under R version 3.4.2

## Loading required package: gplots

## Warning: package 'gplots' was built under R version 3.4.2

##
## Attaching package: 'gplots'

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

pred <- prediction(fitted.values(credit.glm0), credit.train$FullyPaid)
```

La función prediction calcula los valores de verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos para diferentes puntos de corte. Requiere como argumentos las probabilidades estimadas y las etiquetas de la variable respuesta en los datos (en este caso ceros y unos).

```
AUC <- performance(pred, "auc")
AUC@y.name

## [1] "Area under the ROC curve"


AUC@y.values

## [[1]]
## [1] 0.9345633

perf2 <- performance(pred, "tpr", "fpr")
plot(perf2, colorize = TRUE) # mostramos colores seg?n el punto de corte
# Añadimos la recta y=x que ser?a la correspondiente al peor clasificador
abline(a = 0, b = 1)
# a?adimos el valor del ?rea bajo la curva
```

```
text(0.4, 0.6, paste(AUC@y.name, "\n", round(unlist(AUC@y.values), 3)),
cex = 0.7)
```



8. Cross validation

En el apartado 3, al comenzar la estimación del modelo, se seleccionó una muestra de entrenamiento (90%) y otra muestra de validación, que tiene el (10% 4231 observaciones aleatorias). En éste apartado, se comprobará si el modelo es adecuado también para otras muestras. Para ello, se volverá a estimar el mismo modelo sobre la muestra de validación y se comprobará que, en efecto, el modelo estimado es válido para ésta muestra también:

```
head(credit.test)
```

```
##    loadstats.dti loadstats.grade loadstats.int_rate
loadstats.revol_util
## 3           8.72               3             0.1596
0.985
## 5          17.94               2             0.1269
0.539
## 7          23.51               3             0.1596
0.856
## 14         12.56               2             0.0991
0.431
## 17         18.60               3             0.1527
0.702
## 29          5.63               2             0.1171
0.377
##    FullyPaid loadstats.annual_inc loadstats.total_acc
loadstats.loan_amnt
## 3          1                12252                  10
2400
## 5          1                80000                  38
3000
## 7          1                47004                  11
7000
## 14         1                15000                  11
3000
## 17         1                42000                  28
10000
## 29         1               106000                  44
4000
##    loadstats.total_pymnt loadstats.total_rec_int
## 3               3005.667                605.67
## 5               4066.908               1066.91
## 7              10137.840               3137.84
## 14              3480.270                480.27
```

```
## 17             12527.150                  2527.15
## 29              4486.294                   486.29

dmensionCreditTest<-dim(credit.test); dmensionCreditTest

## [1] 4231   10

# Estimamos el modelo
credit.glm1<-
glm(FullyPaid~loadstats.grade+loadstats.dti+loadstats.annual_inc+loadstat
s.total_pymnt+loadstats.loan_amnt,family=binomial,credit.test)

summary(credit.glm1)

##
## Call:
## glm(formula = FullyPaid ~ loadstats.grade + loadstats.dti +
loadstats.annual_inc +
##     loadstats.total_pymnt + loadstats.loan_amnt, family = binomial,
##     data = credit.test)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -5.8105   0.0651   0.2115   0.3503   4.0965
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)             2.636e+00  2.391e-01  11.026  < 2e-16 ***
## loadstats.grade2       -9.891e-01  2.002e-01  -4.940 7.79e-07 ***
## loadstats.grade3       -1.447e+00  2.085e-01  -6.938 3.98e-12 ***
## loadstats.grade4       -2.020e+00  2.262e-01  -8.930  < 2e-16 ***
## loadstats.grade5       -2.617e+00  2.716e-01  -9.635  < 2e-16 ***
## loadstats.grade6       -2.279e+00  4.290e-01  -5.312 1.08e-07 ***
## loadstats.grade7       -3.771e+00  5.704e-01  -6.611 3.81e-11 ***
## loadstats.dti          -2.220e-02  9.604e-03  -2.311   0.0208 *
## loadstats.annual_inc    4.358e-06  2.118e-06   2.058   0.0396 *
## loadstats.total_pymnt   8.130e-04  3.456e-05  23.527  < 2e-16 ***
## loadstats.loan_amnt    -7.462e-04  3.255e-05 -22.924  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 3560.8  on 4230  degrees of freedom
## Residual deviance: 1682.5  on 4220  degrees of freedom
## AIC: 1704.5
##
## Number of Fisher Scoring iterations: 7

prediccion1 <- ifelse(fitted.values(credit.glm1) >= 0.5, 1, 0)
table(credit.test$FullyPaid, prediccion1)
```

```
##    prediccion1
##        0    1
##   0  405  225
##   1   48 3553
```

```
tabla.clasif1 <- table(credit.test$FullyPaid, prediccion1)
tcc1 <- 100 * sum(diag(tabla.clasif1))/sum(tabla.clasif1)
tcc1
```

```
## [1] 93.54762
```

Se puede observar cómo todas las variables elegidas anteriormente, también son significativas para ésta muestra y que el porcentaje de aciertos es aproximadamente el mismo, 93.547%.

Por tanto, se conluye que el modelo estimado para predecir la varible dependiente fullypaid(1 si el préstamo es pagado y 0 en caso contrario) es adecuado.

9. Conclusiones

El modelo de regresión logística estimado para explicar la variable loan_status predice un 93.34% de los casos de forma correcta y viene dado por la ecuación:

$$\text{logit}[p(x)] = \ln\left[\frac{p(x)}{1} - p(x)\right]$$
$$= 2.635137e+00 - 1.026770e+00 * \text{grade2} - 1.540424e+00 * \text{grade3} - 2.088736e+00 * \text{grade4} - 2.385573e+00 * \text{grade5} - 3.044973e+00 * \text{grade6} - 2.850872e+00 * \text{grade7} - 1.195882e-02 * \text{dti} + 3.169457e-06 * \text{annual}_{\text{inc}} + 7.638494e-04 * \text{totalpymnt} - 6.953449e-04 * \text{loan\_amnt}$$

En función del signo que toman los coeficientes estimados beta, las variables se pueden agrupar en dos grupos: las que presentan un efecto positivo incrementando la probabilidad de que se pague el préstamo y otras de efecto negativo que reducen esa probabilidad.

Las variables de efecto negativo se pueden interpretar de la siguiente manera:

Dentro de aquellas quepresentan un efecto negativo está la variable grado, en general en todos los posibles niveles de riesgo que tiene, incrementándose el número negativo en función de que aumente el riesgo (del grado 1 al 7). Esto tiene sentido, pues a mayor riesgo la probabilidad de pago del préstamo se va reduciendo en mayor cuantía.

La variable dti representa la deuda respecto a los ingresos del prestatario. Esta variable también tiene coeficiente negativo, es decir que cuando aumenta la deuda o se reducen los ingresos del prestatario, se reduce la probabilidad de pag del préstamo.

La variable loan_amnt indica la cantidad solicitada por el prestamista. Tiene influencia negativa pues cuanto mayor sea la cantidad prestada se reduce la probabilidad del pago.

Las variables de efecto positivo son las siguientes:

annual_inc: ingresos anuales del prestatario. Esta variable tiene un efecto positivo, a medida que aumentan los ingresos del prestatario, la probabilidad de devolder el préstamo es mayor.

total_pymnt: importe recibido a la fecha de la cantidad total prestada. Tiene un efecto positivo pues si el importe recibido es mayor, incrementará la probabilidad de devolución del préstamo.