reso_practica4_clasificacion.R

gemamaria

2024-10-08

```
library(ISLR)
data("Default")
 ?Default
attach(Default)
table(default) #esta tabla es la que vamos a "perseguir" con los diferentes modelos.
```

```
## default
##   No  Yes
## 9667  333
```

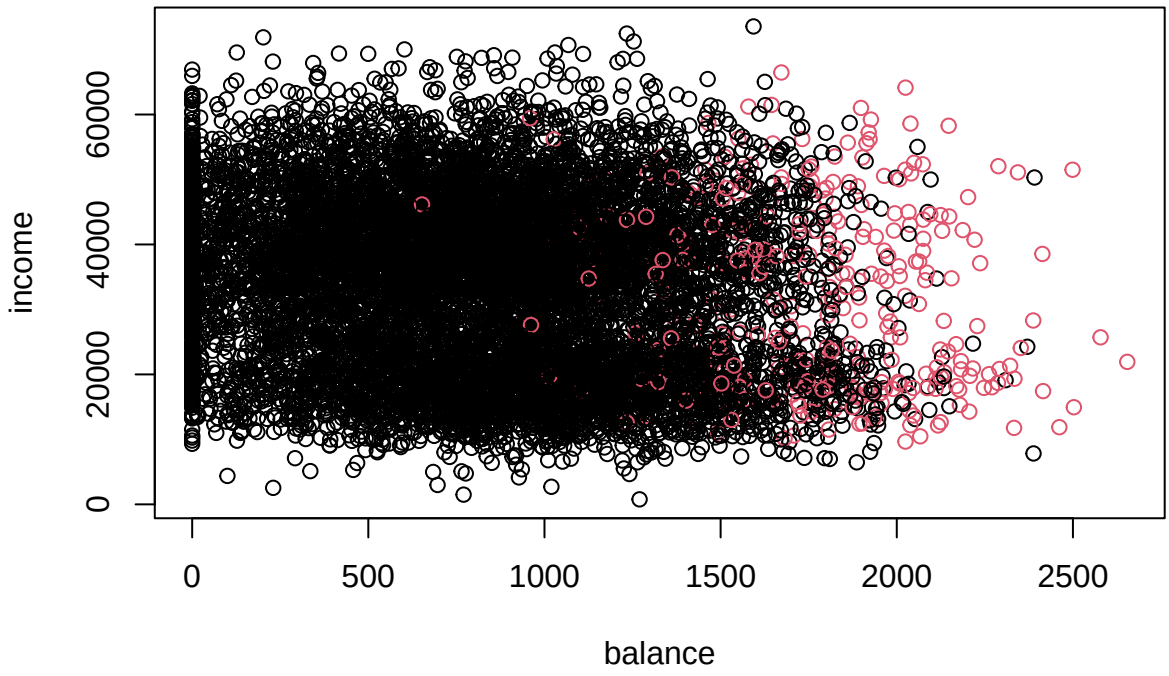
```
#View(Default)

### Observemos los datos:
names(Default)
```

```
## [1] "default" "student" "balance" "income"
```

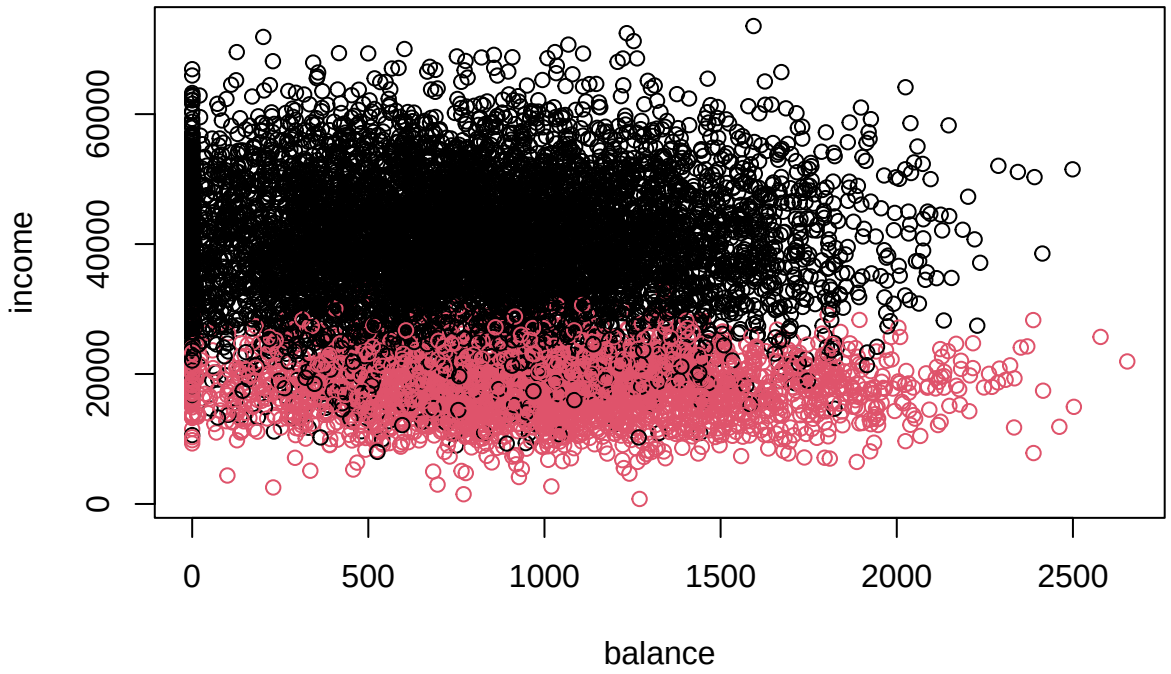
```
dim(Default)
```

```
## [1] 10000     4
```

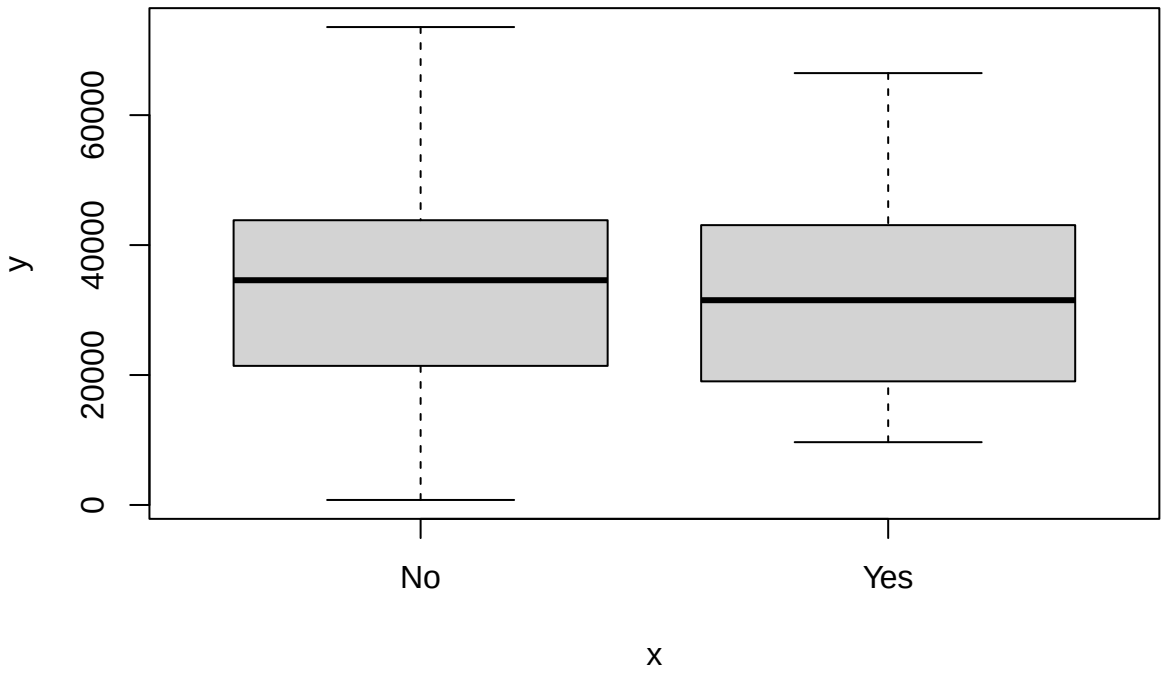
```
plot(balance,income,col=default) #col=colorear
```



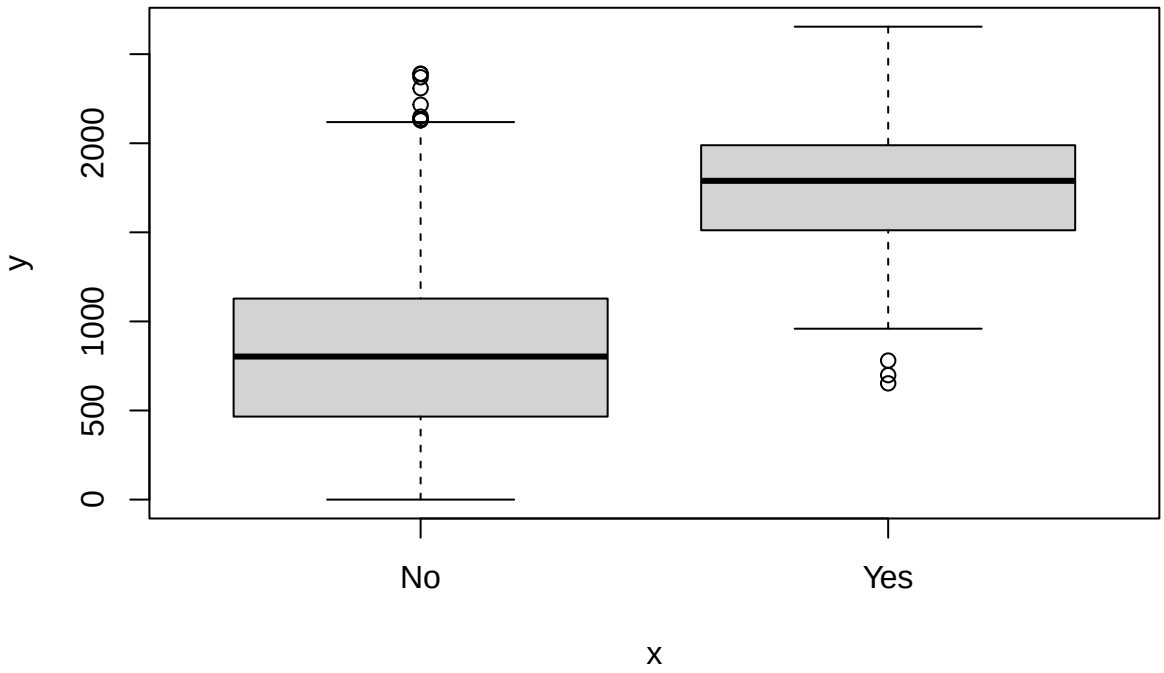
```
plot(balance,income,col=student) #col=colorear
```



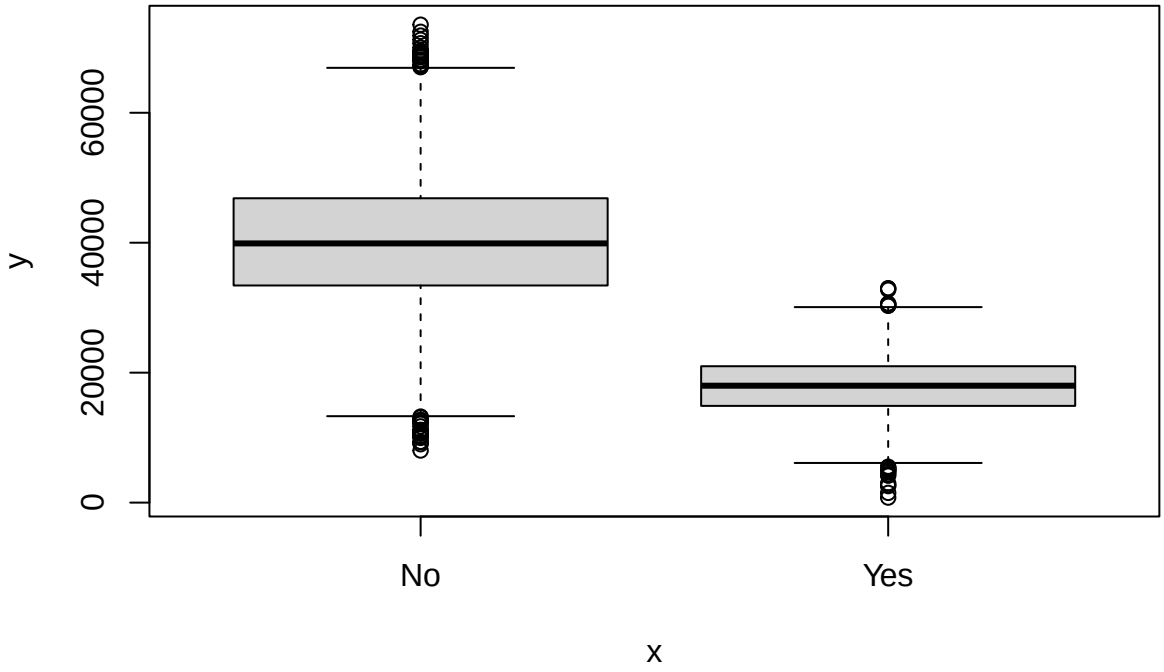

```
plot(default,income)
```



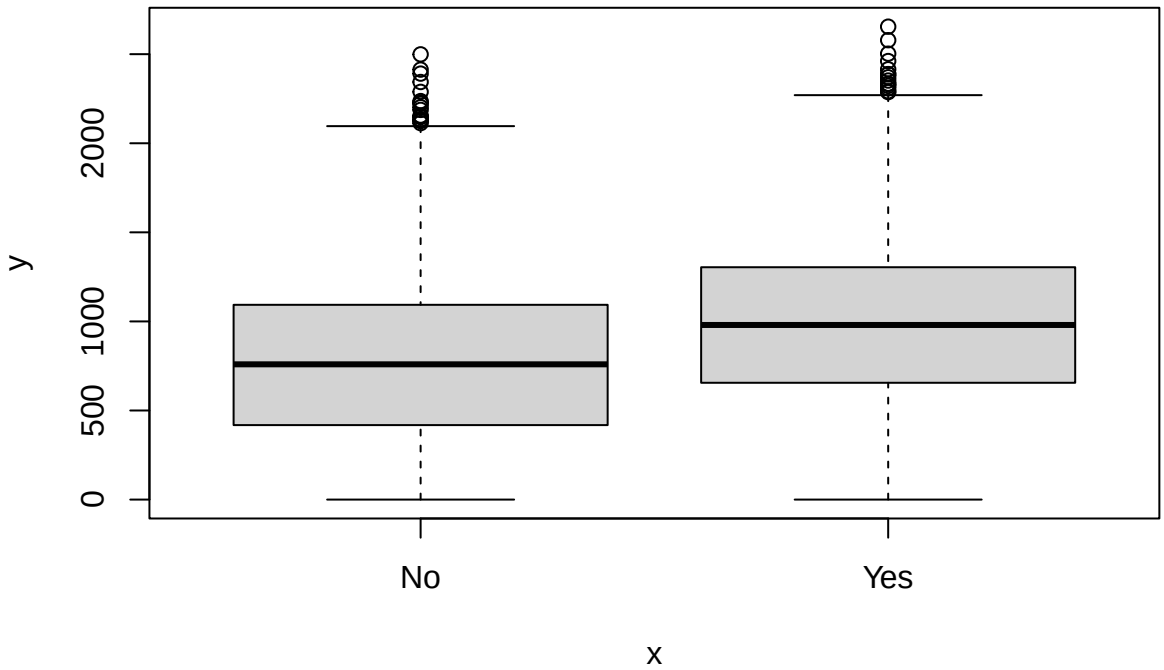

```
plot(default,balance)
```




```
plot(student,income)
```




```
plot(student,balance)
```




```
## ¿Conclusiones?
### una de las conclusiones que podemos sacar es que el estudiante más uso de
### la tarjeta, aunque tenga menos ingresos que el que no lo es;
### otra: el income está altamente relacionado con student.
### otra: ?

addmargins( table(default,student))
```

```
##         student
## default    No   Yes   Sum
##     No   6850  2817  9667
##     Yes   206   127   333
##     Sum  7056  2944 10000
```

```
prop.table( table(default,student))*100
```

```
##         student
## default    No   Yes
##     No  68.50 28.17
##     Yes  2.06  1.27
```

```
prop.table( table(default,student),2 )*100
```

```
##         student
## default        No       Yes
##     No  97.080499 95.686141
##     Yes  2.919501  4.313859
```

```
prop.table( table(default,student),1 )*100
```

```
##         student
## default       No      Yes
##     No  70.85963 29.14037
##     Yes 61.86186 38.13814
```

```
### ¿Conclusiones?

### Hagamos Logistic Regression para predecir Default (número de clases=2), with p=1, predictor: balanc
### Tal como se dijo en clase, la recta de regresión no parece lo más adecuado.
### Creamos primero dummy variables para la recta de regresión. Si no, da error:

lm(default~ balance)
```

```
## Warning in model.response(mf, "numeric"): using type = "numeric" with a factor
## response will be ignored
```

```
## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors
```

```
##
## Call:
## lm(formula = default ~ balance)
##
## Coefficients:
## (Intercept)      balance
##   0.9248080    0.0001299
```

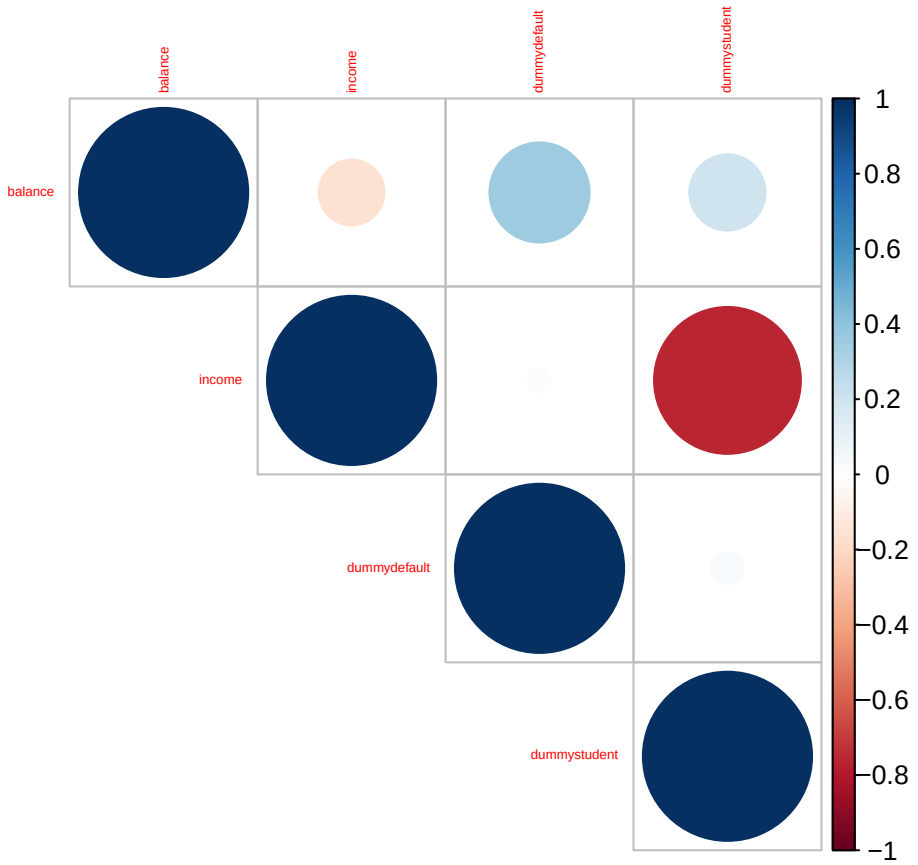
```
class(default)
```

```
## [1] "factor"
```

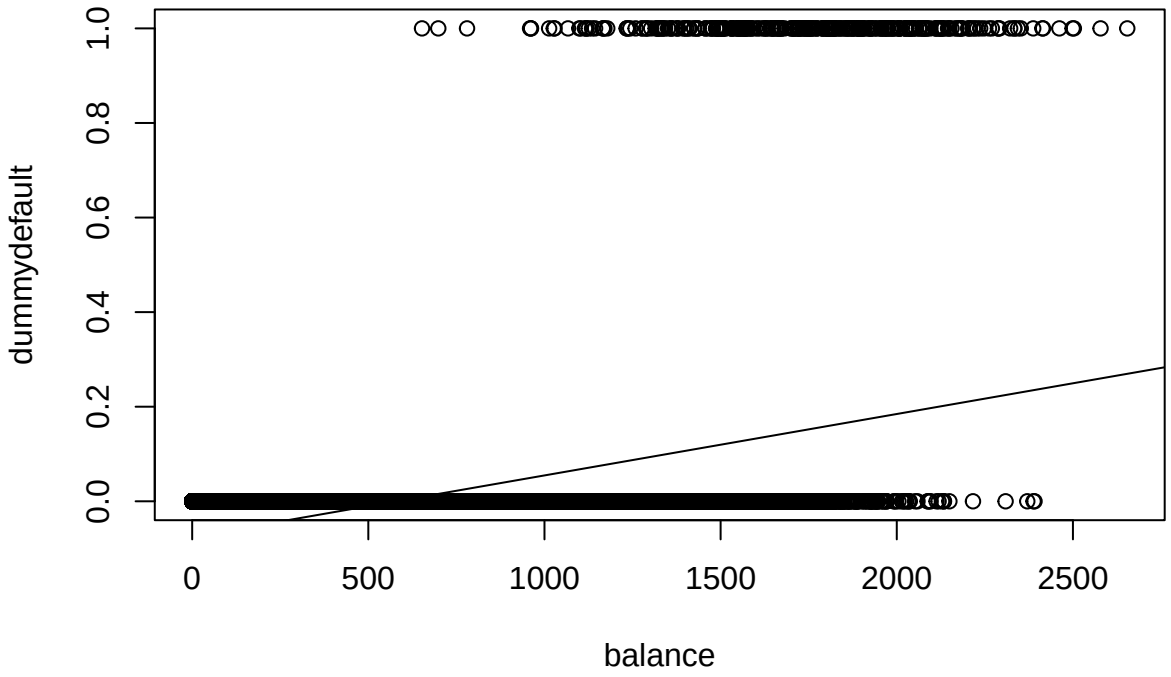
```
Default$dummydefault <- (default == "Yes")*1
Default$dummystudent <- (student == "Yes")*1
attach(Default)
```

```
## The following objects are masked from Default (pos = 3):
##
##     balance, default, income, student
```

```
# aprovechamos que tenemos todas numéricas para analizar la información que nos
# da la matriz de correlación

M<-cor(data.frame(balance,income,dummydefault,dummystudent))
corrplot::corrplot(M, type = "upper", tl.cex = 0.4)
```

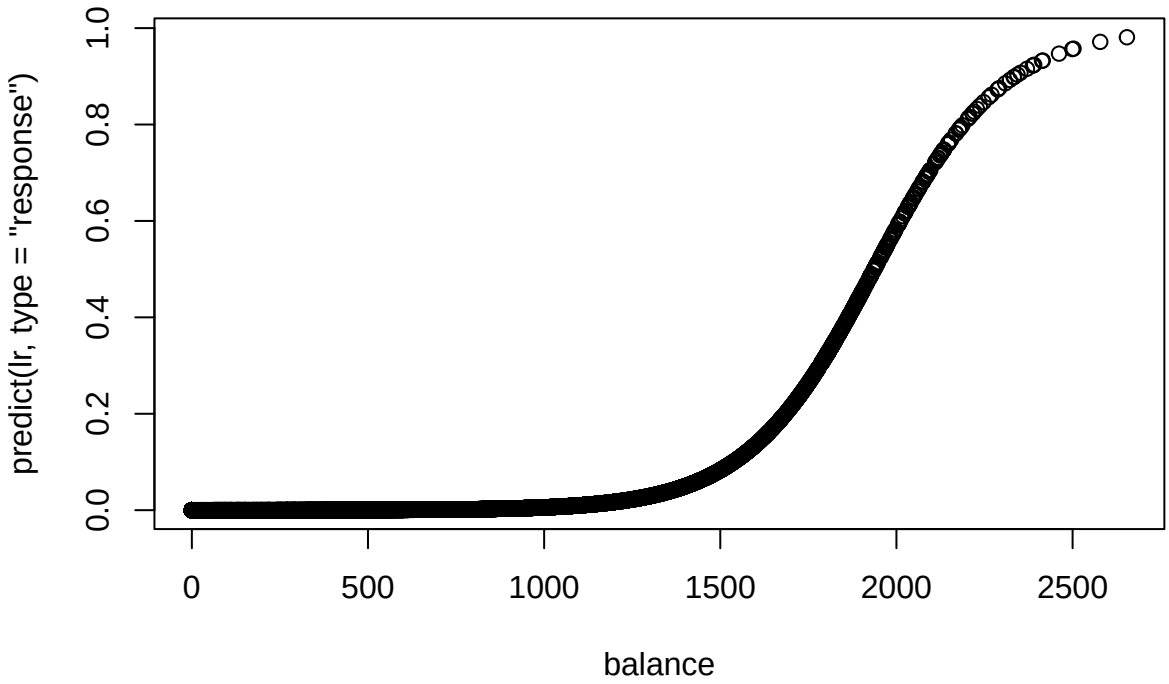



```
plot(balance,dummydefault) # no es un plot de cajas
abline(lm(dummydefault~balance))
```




```
## Para la regresión logística, no hace falta crear variables dummy
```

```r
lr<-glm(default~balance,family=binomial)

## Visualizamos la respuesta. Recordad que el modelo de regresión logística
## predice P(Y=1/X)

plot(balance, predict(lr, type = "response"))
```




```r
## nos aseguramos que 1 es default=Yes
contrasts(default)
```

```
##     Yes
## No    0
## Yes   1
```

```r
## lo que visualizamos en la gráfica es:

predict(lr, type = "response")[1:10]
```

```
##            1            2            3            4            5            6
## 1.305680e-03 2.112595e-03 8.594741e-03 4.344368e-04 1.776957e-03 3.704153e-03
##            7            8            9           10
## 2.211431e-03 2.016174e-03 1.383298e-02 2.366877e-05
```

```r
## Al igual que se hacía en el análisis de regresión lineal,
## podremos utilizar las funciones coef, summary, residuals, etc.

lr$coefficients
```

```
##   (Intercept)       balance
## -10.651330614   0.005498917
```

```r
summary(lr)$coef #Para su interpretación, vemos página 136 del libro.
```

```
##                   Estimate   Std. Error   z value      Pr(>|z|)
## (Intercept) -10.651330614 0.3611573721 -29.49221 3.623124e-191
```

```
## balance       0.005498917 0.0002203702  24.95309 1.976602e-137
```

```
## También se puede hacer predicciones. Por ejemplo:
predict(lr,data.frame(balance=1000),type = "response" )
```

```
##           1
## 0.005752145
```

```
## ojo! el modelo ya no es lineal
predict(lr,data.frame(balance=2000),type = "response" )
```

```
##         1
## 0.5857694
```

```
## Vemos que nos predice el modelo.
## Para ello, podemos "contar" cuantos tienen la probabilidad por encima de 0,5:
lr.probs <- predict(lr , type = "response") ##aquí van las probabilidades
lr.pred <- rep("No", 10000) ##creamos un vector con 10000 No'es
lr.pred[lr.probs > .5] = "Yes" ## y modificamos las posiciones donde la prob. sea >.5,

table(lr.pred)
```

```
## lr.pred
##   No  Yes
## 9858  142
```

```
table(lr.pred,default) ##¿Qué aprecias?
```

```
##        default
## lr.pred   No  Yes
##     No  9625  233
##     Yes   42  100
```

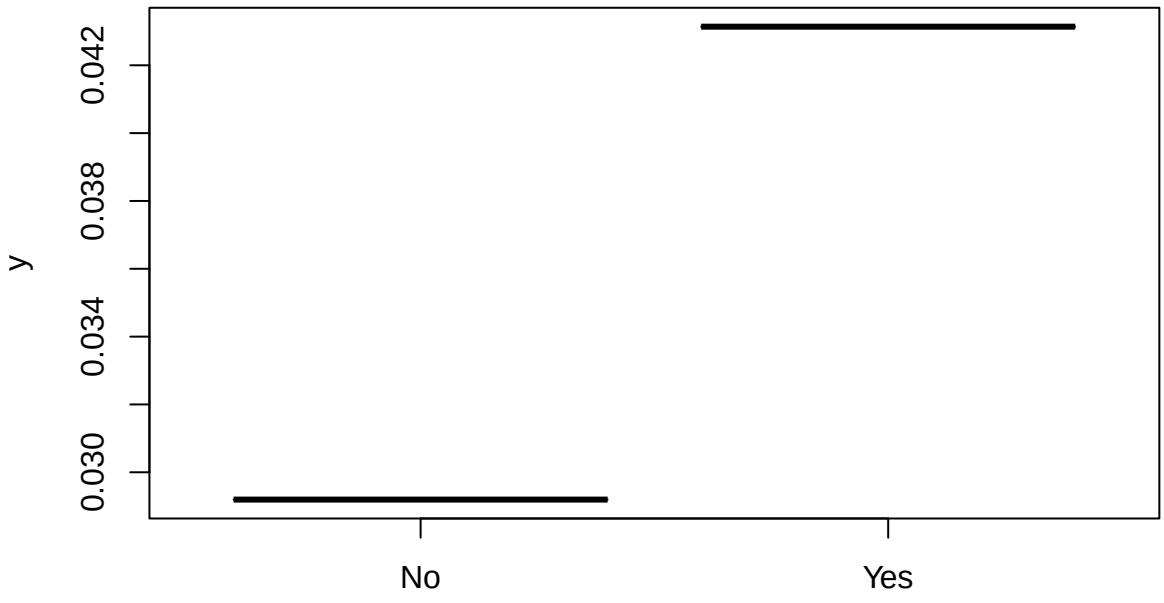
```
## Dicho esto, no olvidemos que hay más variables en la base de datos:
names(Default)
```

```
## [1] "default"      "student"      "balance"      "income"       "dummydefault"
## [6] "dummystudent"
```

```
#Si tenemos en cuenta el ser estudiante:

lrs<-glm(default~student,family=binomial)
lrs$coefficients
```

```
## (Intercept)  studentYes
##  -3.5041278   0.4048871
```

```
## Lógicamente, en predict tan sólo obtendremos dos datos:
## Pr(default=Yes|student=yes) y Pr(default=Yes|student=no)
plot(student,predict(lrs, type = "response"))
```




```
## Una pregunta que nos podríamos hacer es si este modelo nos da la misma información
## que las frecuencias condicionadas: la respuesta es sí.

predict(lrs, type = "response")[1:3]
```

```
##          1          2          3
## 0.02919501 0.04313859 0.02919501
```

```
prop.table( table(default,student),2 )*100
```

```
##        student
## default        No       Yes
##     No  97.080499 95.686141
##     Yes  2.919501  4.313859
```

```
## Parece que a un banquero le puede interesar, además del balance,
## si el futuro cliente es estudiante o no.

## Veamos que ocurre si tenemos todas las variables en cuenta.

## MULTIPLE LOGÍSTIC LINEAR REGRESION ##

lrT<-glm(default~balance + income +student,family=binomial)

summary(lrT)$coef
```

```
##                  Estimate   Std. Error    z value      Pr(>|z|)
## (Intercept) -1.086905e+01 4.922555e-01 -22.080088 4.911280e-108
## balance      5.736505e-03 2.318945e-04  24.737563 4.219578e-135
## income       3.033450e-06 8.202615e-06   0.369815  7.115203e-01
## studentYes  -6.467758e-01 2.362525e-01  -2.737646  6.188063e-03
```

```
## El coeficiente que acompaña a "Student" es negativo, lo cual es un poco sorprendente.
## Esto significa que si fijamos  balance e income,  el estudiante tiene menos
## probabilidad que defraude. Además la variable income parece que sobra (un z-value muy
```

```
## pequeño, Pr muy alta).

## Los gráficos y datos anteriores indican que income está altamente relacionado con student



## Estos plots indican que los ingresos de un estudiante son menores, con lo cual
## si el income aumenta, el coeficiente que acompaña a Student debe ser negativo.
## Se aprecia que Students tienen menos ingresos, mismas deudas, con
## lo cual es lógico que tengan más probabilidad de fraude cuando sólo se considera
## el predictor student. Lo más relevante: mismo income, mismo balance, el
## estudiante se comporta mejor que el no estudiante; sin embargo, el estudiante
## suele tener menos income y más balance que el que no lo es.

## De todas formas, nos interesa ver la efectividad del modelo.
## Para ello, podemos "contar" cuantos tienen la probabilidad por encima de 0,5:

lrT.probs <- predict(lrT , type = "response") ##aquí van las probabilidades
lrT.pred <- rep("No", 10000) ##creamos un vector con 10000 No'es
lrT.pred[lrT.probs > .5] = "Yes" ## y modificamos las posiciones donde la prob. sea >.5,

## por último, comparamos resultados:
table(lrT.pred)
```

```
## lrT.pred
##   No  Yes
## 9855  145
```

```
table(default)
```

```
## default
##   No  Yes
## 9667  333
```

```
## Observad que no es mucho mejor que la tabla con regresión logística con sólo default y balance

########## LINEAR DISCRIMINANT ANALYSIS (LDA) ##########

library(MASS)

lda1=lda(default~balance) #p=1, K=2

## Vemos que modelo obtenemos:

#### Prior probabilities of groups:
#######indican simplemente las estimaciones de p1_1=P(default=1) y pi_0=P(default=0) a partir de los da
#### Group means:
#######la media del balance con default=1 y default=0.

#### Recordad que en el LDA, se asume que las funciones de densidad de probabilidad
#### de cada clase son distribuciones normales, con la misma varianza (matriz de covarianza)

## Por curiosidad:

k1=subset.data.frame (Default,default=="Yes")
```

```r
sd(k1$balance)
```

```
## [1] 341.2668
```

```r
k0=subset.data.frame (Default,default=="No")
sd(k0$balance)
```

```
## [1] 456.4762
```

```r
lda1
```

```
## Call:
## lda(default ~ balance)
##
## Prior probabilities of groups:
##     No    Yes
## 0.9667 0.0333
##
## Group means:
##       balance
## No   803.9438
## Yes 1747.8217
##
## Coefficients of linear discriminants:
##                 LD1
## balance 0.002206916
```

```r
## La predicción se lleva a cabo con la función predict:

lda1.pred=predict(lda1)

names(lda1.pred) ##podemos obtener tres tipos de datos
```

```
## [1] "class"     "posterior" "x"
```

```r
lda1.pred$class[1:3] ##indica el valor de default asignado a cada observación-
```

```
## [1] No No No
## Levels: No Yes
```

```r
## las tablas nos indican que los resultandos no son óptimos.
table(lda1.pred$class,default)
```

```
##      default
##         No  Yes
##   No  9643  257
##   Yes   24   76
```

```r
table(lr.pred,default) #de hecho, un poco mejor.
```

```
##        default
## lr.pred   No  Yes
##     No  9625  233
##     Yes   42  100
```

```r
lda1.pred$posterior[1:3,] ##la columna k+1 de esta clase indica la probabilidad de default=k
```

```
##          No         Yes
## 1 0.9972130 0.002786981
```

```
## 2 0.9958358 0.004164240
## 3 0.9865931 0.013406929
```

```
lda1.pred$x[1:3] ##x contiene el discriminante lineal para k=1
```

```
## [1] -0.23359853 -0.04015369  0.52563067
```

```
## la observación 174 tiene como dato default=Yes. Observa que datos me da la lda.

############################################################
## Veamos ahora LDA con K=2, p=2:

lda2=lda(default~balance+student )

## Tenemos:
lda2
```

```
## Call:
## lda(default ~ balance + student)
##
## Prior probabilities of groups:
##     No    Yes
## 0.9667 0.0333
##
## Group means:
##       balance studentYes
## No   803.9438  0.2914037
## Yes 1747.8217  0.3813814
##
## Coefficients of linear discriminants:
##                     LD1
## balance     0.002244397
## studentYes -0.249059498
```

```
## prior probabilities: no cambian lógicamente
## group means: la media de balance de los que no defraudan y de los que sí.
## A grosso modo, estiman los mu_k.
## Los coefficients están relacionados con la función que hay que
## minimizar, delta_k.

## Sin embargo analicemos la "confusion matrix":
ldapredi=predict(lda2)
names(ldapredi)
```

```
## [1] "class"     "posterior" "x"
```

```
table(ldapredi$class,default ) #no mejora mucho la anterior. Es la tabla 4.4 del libro.
```

```
##      default
##         No  Yes
##   No  9644  252
##   Yes   23   81
```

```
## En los datos, hay 9644+23=9667 que no defrauden, y 252+81=333 que sí.
## El análisis detecta que 9644 +252=9896 como NO, y 23+81=104 como SÍ.
## Clasifica mal a 23+252=275 personas ( o sea, un 2.75% al haber 10000)
## Sin embargo, 333-81=252 personas que defraudan no han sido detectadas,
## es decir, un 75%  (252/333=0.75) de la gente que defrauda en la muestra no ha
```

```r
## sido detectaca. Nota: si se tiene en cuenta income, el análisis es casi idéntico.
## Conclusión: el análisis no ha sido muy efectivo.

## Por defecto, el método clasifica Default=1 las probabilidades mayores de 0.5:
sum(ldapredi$posterior[,2]>.5)
```

```
## [1] 104
```

```r
## Podemos modificar el umbral (threshold value)  a 0.2, es decir, si
## P(default=Yes|X)>2 => default =yes
sum(ldapredi$posterior[,2]>.2)
```

```
## [1] 430
```

```r
## el número de "defraudadores" es 430, que es más de los que hay pero
## a modos prácticos, beneficia el banquero (véase tabla 4.5 del libro)

#Por último, si queremos hacer predicciones en un nuevo conjunto de datos:
predict(lda2,newdata = data.frame( balance=c(1230,2200,700,1114), student=c("No", "No", "Yes", "Yes")),
```

```
## $class
## [1] No  Yes No  No
## Levels: No Yes
##
## $posterior
##          No         Yes
## 1 0.9679713 0.032028739
## 2 0.2396466 0.760353429
## 3 0.9983803 0.001619702
## 4 0.9887526 0.011247425
##
## $x
##          LD1
## 1  0.9590185
## 2  3.1360835
## 3 -0.4795714
## 4  0.4496090
```