Solución Ayudantía: Regresión Logística

Natalie Julian - Paula Muñoz

Ejercicio 1

La base de datos `heart.csv` contiene información de distintos pacientes y sus resultados de distintos exámenes cardiacos. Algunas de las variables de esta base están descritas en la siguiente tabla:

Variable	Descripción
`age`	Edad en años
`sex`	Sexo de la persona (1=Hombre, 0=Mujer)
`cp`	Tipo de dolor en el pecho (1= Angina típica, 2= Angina atípica, 3=No anginoso, 0=Asintomático)
`fbs`	Glicemia en ayunas (1=Mayor a 120 mg/dl, 0=Menor a 120 mg/dl)
`thall`	Presencia de defecto (1=Fijo, 2=Normal, 3=Reversible)
...	...

Considerando a los pacientes de esta base de datos y las variables de la tabla realice los siguientes pasos:

1.- Separe la base de datos en dos grupos, 70% de los datos para generar una base de entrenamiento y el esto para la base de validación.

```
# Cargo la librería tidyverse para usar %>% sin problemas
library(tidyverse)

### Cargo la base
corazon <- readr::read_csv("heart.csv")

### Observo la base
glimpse(corazon)
```

```
## Rows: 303
## Columns: 14
## $ age      <dbl> 63, 37, 41, 56, 57, 57, 56, 44, 52, 57, 54, 48, 49, 64, 58, 5~
## $ sex      <dbl> 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1~
## $ cp       <dbl> 3, 2, 1, 1, 0, 0, 1, 1, 2, 2, 0, 2, 1, 3, 3, 2, 2, 3, 0, 3, 0~
## $ trtbps   <dbl> 145, 130, 130, 120, 120, 140, 140, 120, 172, 150, 140, 130, 1~
## $ chol     <dbl> 233, 250, 204, 236, 354, 192, 294, 263, 199, 168, 239, 275, 2~
## $ fbs      <dbl> 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0~
## $ restecg  <dbl> 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1~
## $ thalachh <dbl> 150, 187, 172, 178, 163, 148, 153, 173, 162, 174, 160, 139, 1~
## $ exng     <dbl> 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0~
## $ oldpeak  <dbl> 2.3, 3.5, 1.4, 0.8, 0.6, 0.4, 1.3, 0.0, 0.5, 1.6, 1.2, 0.2, 0~
## $ slp      <dbl> 0, 0, 2, 2, 2, 1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 1, 2, 0, 2, 2, 1~
## $ caa      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0~
```

```
## $ thall    <dbl> 1, 2, 2, 2, 2, 1, 2, 3, 3, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3~
## $ output   <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
```

```r
### Cambio a factor las variables que voy a usar (cp, fbs y thall)
corazon <- corazon %>% mutate(cp = factor(cp),
                  thall = factor(thall),
                  fbs = factor(fbs))

### Chequeo que esté todo bien
glimpse(corazon)
```

```
## Rows: 303
## Columns: 14
## $ age      <dbl> 63, 37, 41, 56, 57, 57, 56, 44, 52, 57, 54, 48, 49, 64, 58, 5~
## $ sex      <dbl> 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1~
## $ cp       <fct> 3, 2, 1, 1, 0, 0, 1, 1, 2, 2, 0, 2, 1, 3, 3, 2, 2, 3, 0, 3, 0~
## $ trtbps   <dbl> 145, 130, 130, 120, 120, 140, 140, 120, 172, 150, 140, 130, 1~
## $ chol     <dbl> 233, 250, 204, 236, 354, 192, 294, 263, 199, 168, 239, 275, 2~
## $ fbs      <fct> 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0~
## $ restecg  <dbl> 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1~
## $ thalachh <dbl> 150, 187, 172, 178, 163, 148, 153, 173, 162, 174, 160, 139, 1~
## $ exng     <dbl> 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0~
## $ oldpeak  <dbl> 2.3, 3.5, 1.4, 0.8, 0.6, 0.4, 1.3, 0.0, 0.5, 1.6, 1.2, 0.2, 0~
## $ slp      <dbl> 0, 0, 2, 2, 2, 1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 1, 2, 0, 2, 2, 1~
## $ caa      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0~
## $ thall    <fct> 1, 2, 2, 2, 2, 1, 2, 3, 3, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3~
## $ output   <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
```

```r
# Luego separo la base en los dos grupos

n_filas <- dim(corazon)[1]

set.seed(2021)

ind_train <-  sample(1:n_filas, size = n_filas*0.7)

corazon_train = corazon[ind_train,] # Dejo sólo los de entrenamiento
corazon_test = corazon[-ind_train,] # Quito los de entrenamiento
```

2.- Ajuste un modelo para predecir el sexo de los pacientes a partir del tipo de dolor en el pecho presentado, la medición de la glicemia en ayunas y la presencia de un defecto.

```r
# cp: tipo de dolor, fbs: glicemia en ayundas, thall: presencia de
# defecto
modelo <- glm(sex ~ cp + fbs + thall, data = corazon_train,
              family = binomial(link = "logit"))

# Opción 1 para ver el modelo:

summary(modelo)$coefficients
```

```
##                Estimate Std. Error     z value   Pr(>|z|)
## (Intercept)  1.604824091  0.7830451  2.04946574 0.04041660
## cp1          0.336973503  0.4570390  0.73729698 0.46094177
## cp2         -0.008353147  0.4032192 -0.02071615 0.98347209
## cp3          0.582661910  0.7476327  0.77934247 0.43577800
```

```
## fbs1         0.449138531  0.4957085  0.90605366 0.36490745
## thall2      -1.595990588  0.7998460 -1.99537236 0.04600228
## thall3       0.475314226  0.8542761  0.55639416 0.57794143
```

```
# Opción 2:

broom::tidy(modelo)
```

```
## # A tibble: 7 x 5
##   term        estimate std.error statistic p.value
##   <chr>          <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)  1.60        0.783    2.05    0.0404
## 2 cp1          0.337       0.457    0.737   0.461
## 3 cp2         -0.00835     0.403   -0.0207  0.983
## 4 cp3          0.583       0.748    0.779   0.436
## 5 fbs1         0.449       0.496    0.906   0.365
## 6 thall2      -1.60        0.800   -2.00    0.0460
## 7 thall3       0.475       0.854    0.556   0.578
```

3.- Indique si el predictor es factor de riesgo, factor protector o no presenta efecto.

La ventaja de usar `broom::tidy` es que nos entrega los datos del modelo como un tibble, lo que nos permite añadir una columna nueva con los resultados del OR respectivo.

```
broom::tidy(modelo) %>% mutate(OR = exp(estimate))
```

```
## # A tibble: 7 x 6
##   term        estimate std.error statistic p.value    OR
##   <chr>          <dbl>     <dbl>     <dbl>   <dbl> <dbl>
## 1 (Intercept)  1.60        0.783    2.05    0.0404 4.98
## 2 cp1          0.337       0.457    0.737   0.461  1.40
## 3 cp2         -0.00835     0.403   -0.0207  0.983  0.992
## 4 cp3          0.583       0.748    0.779   0.436  1.79
## 5 fbs1         0.449       0.496    0.906   0.365  1.57
## 6 thall2      -1.60        0.800   -2.00    0.0460 0.203
## 7 thall3       0.475       0.854    0.556   0.578  1.61
```

Son factor de riesgo cp1, cp3, fbs1 y thall3. Mientras que factores protector son cp2 y thall2.

4.- Genere la matriz de confusión para el modelo con un punto de corte de 0.7.

```
y_reales <- corazon_test$sex
probs <-  predict.glm(modelo, newdata = corazon_test, type = "response")

corte <-  0.7

y_predichos <- ifelse(probs >=corte, 1, 0)

MLmetrics::ConfusionMatrix(y_pred = y_predichos, y_true = y_reales)
```

```
##       y_pred
## y_true  0  1
##      0 26  7
##      1 24 34
```

5.- Calcule la sensibilidad, la especificidad y la exactitud del modelo.

```
MLmetrics::Sensitivity(y_pred = y_predichos, y_true = y_reales,
                       positive = 1)
```

```
## [1] 0.5862069
```

La sensibilidad nos muestra la proporción de los positivos capturados correctamente, en este caso es un valor un poco superior al 50%, por lo que sería un poco mejor que tirar a una moneda.

```
MLmetrics::Specificity(y_pred = y_predichos, y_true = y_reales,
                       positive = 1)
```

```
## [1] 0.7878788
```

La especificidad nos muestra la proporción de los negativos capturados correctamente, podemos apreciar que capturamos mejor los casos negativos que los positivos.

```
MLmetrics::Accuracy(y_pred = y_predichos, y_true = y_reales)
```

```
## [1] 0.6593407
```

La exactitud es la proporción de los casos clasificados correctamente, independiente de si son positivos o negativos. El valor es bastante bajo.

6.- Calcule el estadístico F_1.

```
MLmetrics::F1_Score(y_pred = y_predichos, y_true = y_reales,
                    positive = 1)
```

```
## [1] 0.6868687
```

El estadístico F_1 expresa de manera conjunta la sensibilidad y la precisión. El valor es también bastante bajo.

7.- Analice la bondad del ajuste.

Para analizar la bondad del ajuste usamos el Test Hosmer Lemeshow, donde la hipótesis nula H_0 es que no existe una diferencia entre los valores observados y los pronósticados en ningún grupo (recordemos que este test separa la muestra en diez grupos). Mientras que la hipótesis alternativa H_1 indica que existe diferencia en al menos un grupo.

```
DescTools::HosmerLemeshowTest(fit = probs, obs = y_reales)
```

```
## Warning in DescTools::HosmerLemeshowTest(fit = probs, obs = y_reales): Found
## only 6 different groups for Hosmer-Lemesho C statistic.
```

```
## $C
##
##  Hosmer-Lemeshow C statistic
##
```

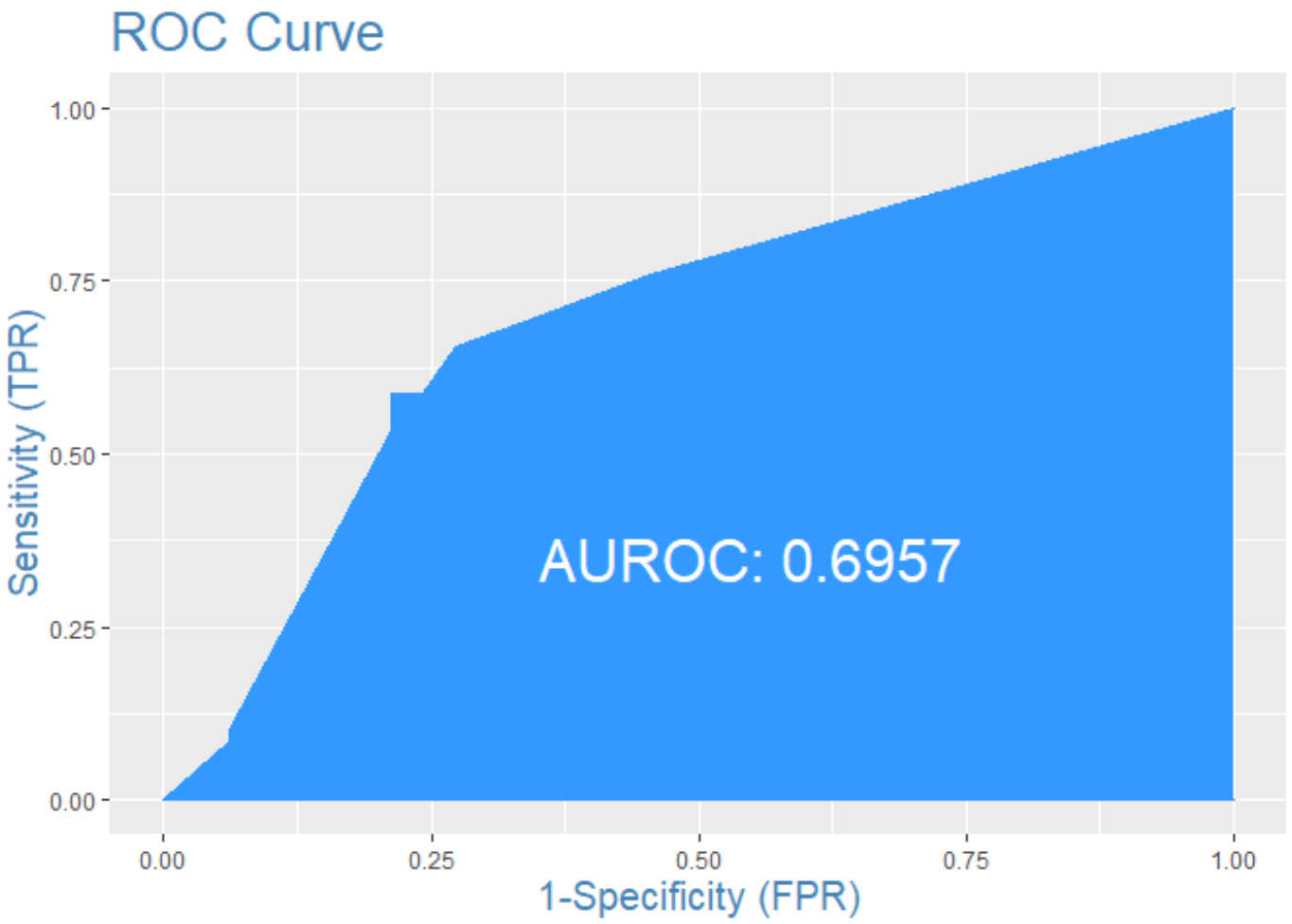
```
## data:  probs and y_reales
## X-squared = 5.1423, df = 4, p-value = 0.273
##
##
## $H
##
##  Hosmer-Lemeshow H statistic
##
## data:  probs and y_reales
## X-squared = 8.1578, df = 8, p-value = 0.4182
```

Con el valor entregado no rechazamos la hipótesis nula, por lo que no habría diferencia entre los valores observados y los pronósticados. Teniendo un buen ajuste.

8.- Analice el poder predictivo del modelo.

Primero analizaremos la curva ROC.

```
InformationValue::plotROC(actuals = y_reales, predictedScores = probs)
```



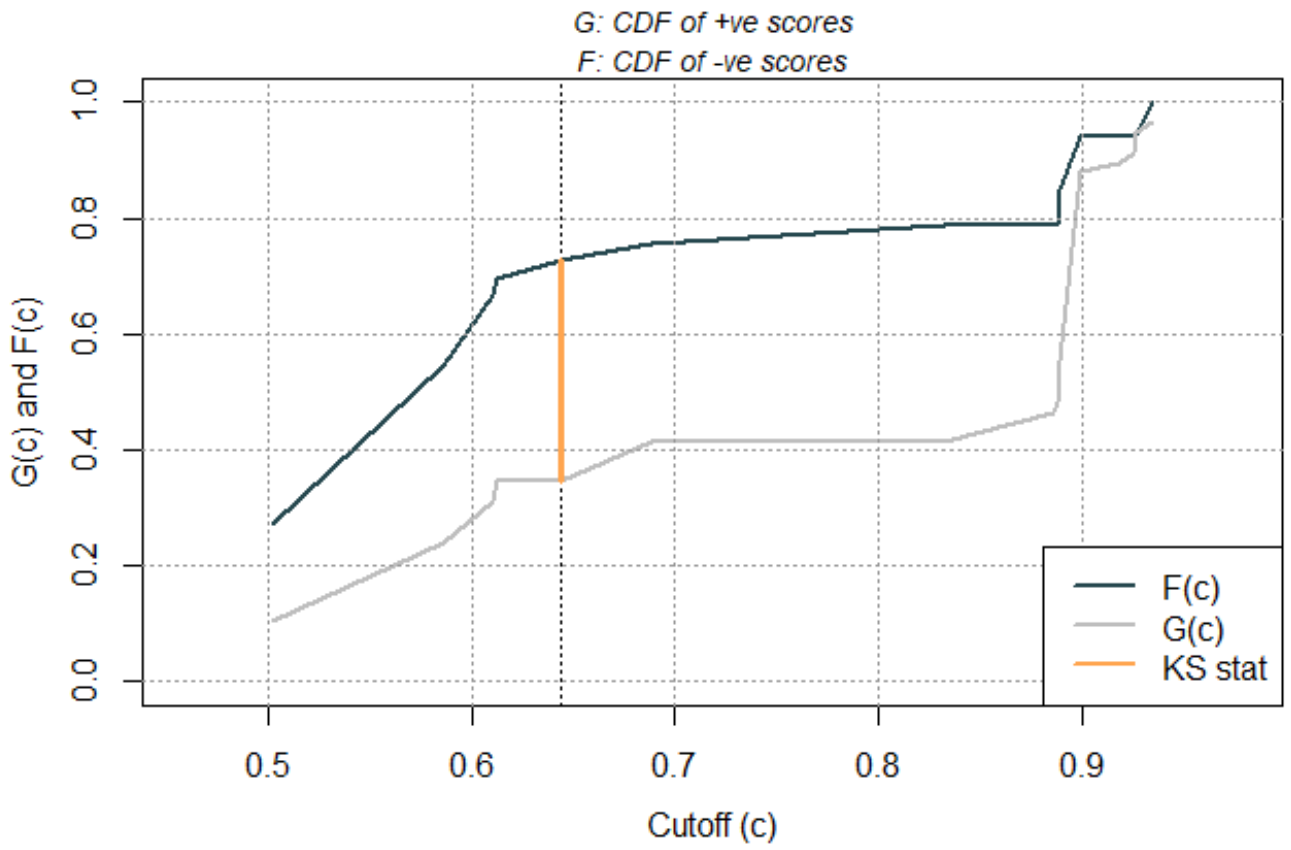
El valor del AUC es 0.6957, lo que corresponde a un valor aceptable. Recordemos que

Valor AUC	Poder predictivo
+90	Sospechoso
75 - 90	Buen Ajuste

Valor AUC	Poder predictivo
60 - 75	Aceptable
50 - 60	Malo
=50	No sirve

Veamos ahora Kolmogorox- Smirnov.

```
ROCit::ksplot(ROCit::rocit(score = probs, class = y_reales))$'KS stat'
```




```
## [1] 0.3824451
```

El valor es 0.3824451, siendo un poder predictivo regular. Recordemos que

Kolmogorox- Smirnov	Poder predictivo
+75	Sospechoso

Kolmogorox- Smirnov	Poder predictivo
60 - 75	Muy Bueno
40 - 60	Bueno
20 - 40	Regular
<20	Malo

Finalmente veremos el Índice de Gini.

```
MLmetrics::Gini(y_true = y_reales, y_pred = probs)
```

```
## [1] 0.4242424
```

Un valor cercano a cero nos indica que el modelo no es capaz de distinguir entre los casos positivos o los negativos, mientras que uno cercano a uno indica lo contrario, que sí hay poder predictivo. En este caso tenemos un valor bastante intermedio, por lo que el poder predictivo sería regular.

Podemos ver en este ejercicio que tomar sólo un indicador no es suficiente para hacer un buen análisis. El Test de Hosmer Lemeshow nos indicó que el ajuste era bueno, mientras que el poder predictor fue solamente regular o aceptable. Por otro lado, el modelo distingue mejor los casos negativos que los positivos. La utilidad de él, por lo tanto, dependerá de lo que se esté buscando.

Además cabe mencionar que en este ejercicio se indicó qué variables usar para el modelo y el punto de corte. En el ejercicio 2 se verá cómo seleccionar las variables mediante el método backward y cómo obtener un punto de corte óptimo.

Ejercicio 2

El paquete `ISLR` posee la base de datos `Default` que contiene información de la tarjeta de crédito de distintos clientes. Las variables de esta base están descritas en la siguiente tabla:

Variable	Descripción
`default`	El cliente incumplió el pago de su cuota (1 = Sí, 0 = No)
`student`	El cliente corresponde a un estudiante (1= Sí, 0 = No)
`balance`	Saldo promedio que le queda al cliente en su tarjeta después de realizar el pago mensual
`income`	Ingreso del cliente

```
library(ISLR)

head(Default, 10)
```

```
##    default student   balance    income
## 1       No      No  729.5265 44361.625
## 2       No     Yes  817.1804 12106.135
## 3       No      No 1073.5492 31767.139
## 4       No      No  529.2506 35704.494
## 5       No      No  785.6559 38463.496
## 6       No     Yes  919.5885  7491.559
## 7       No      No  825.5133 24905.227
## 8       No     Yes  808.6675 17600.451
```

```
## 9       No      No 1161.0579 37468.529
## 10      No      No    0.0000 29275.268
```

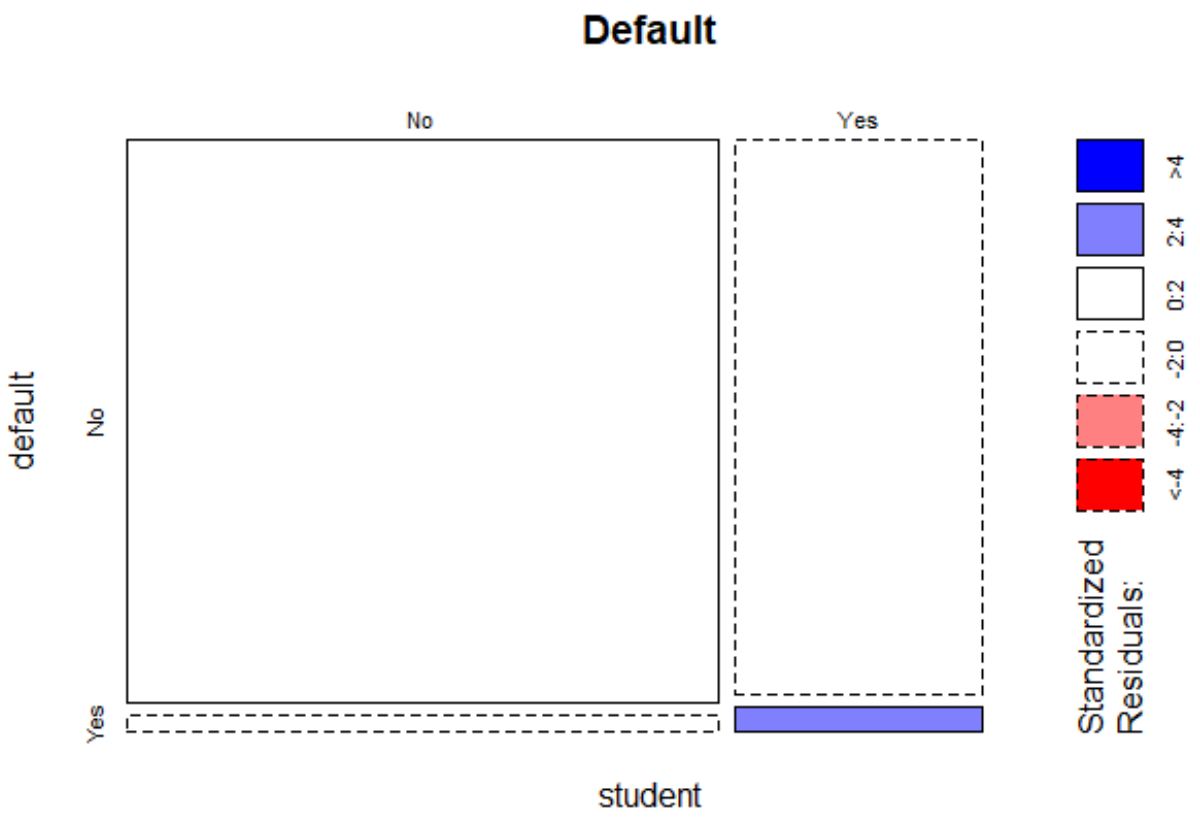
a) Analice la asociación/relación de la variable respuesta default con las variables:

- Student
- Balance
- Income

```
#Student

#H_0: Son independientes
#H_1: No son independientes

chisq.test(Default$student, Default$default)
```

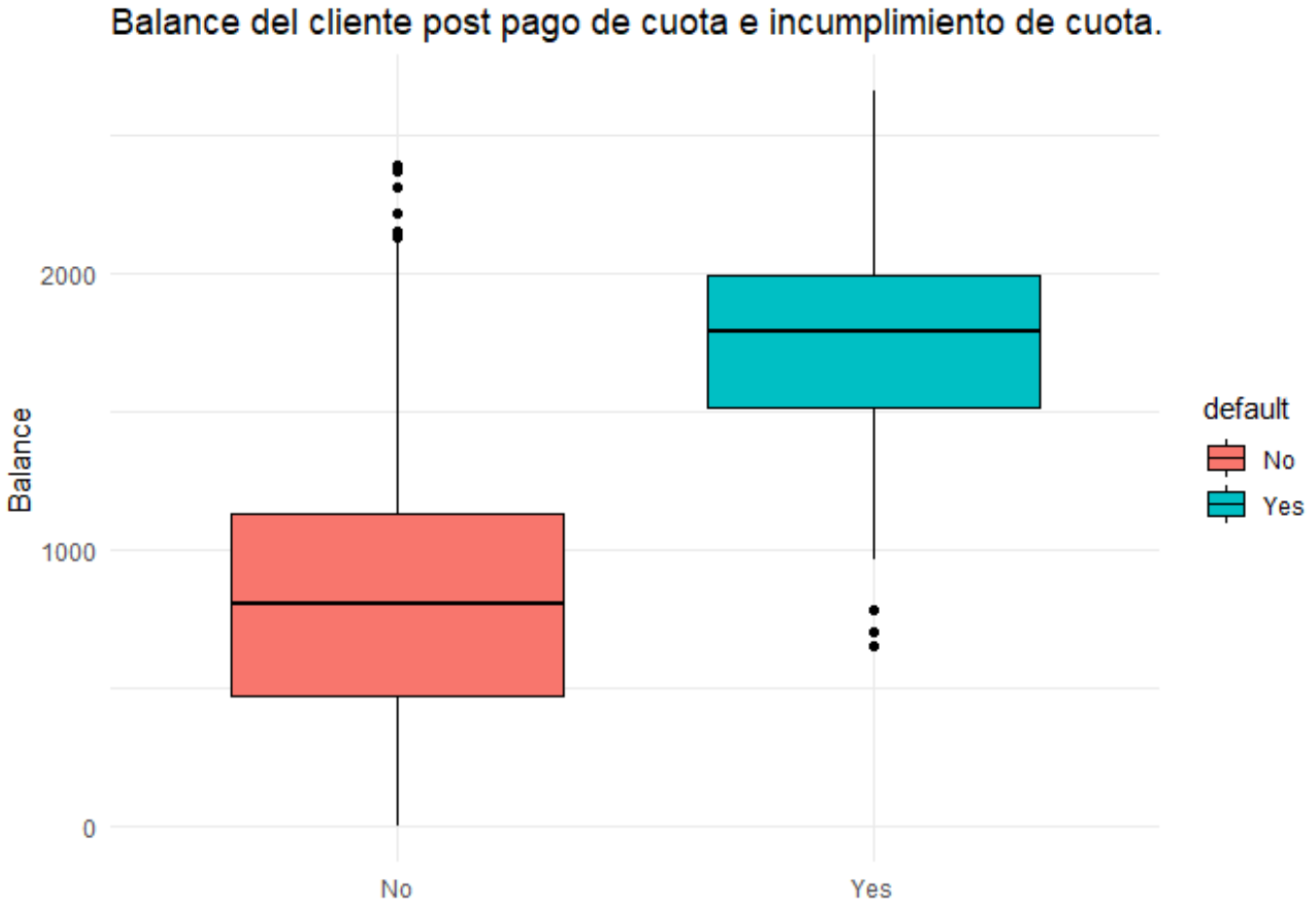
```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  Default$student and Default$default
## X-squared = 12.117, df = 1, p-value = 0.0004997
```

```
mosaicplot(~student+default, data=Default, shade=TRUE)
```




```
#Se rechaza la independencia entre las variables. Se esperaría que
#student fuera una variable significativa en el modelo.

#Balance

library(ggpubr)

ggboxplot(Default, y="balance", x="default", fill="default")+
```

```
  xlab("")+
  ylab("Balance")+
  ggtitle("Balance del cliente post pago de cuota e incumplimiento de cuota.")+
  theme_minimal()
```

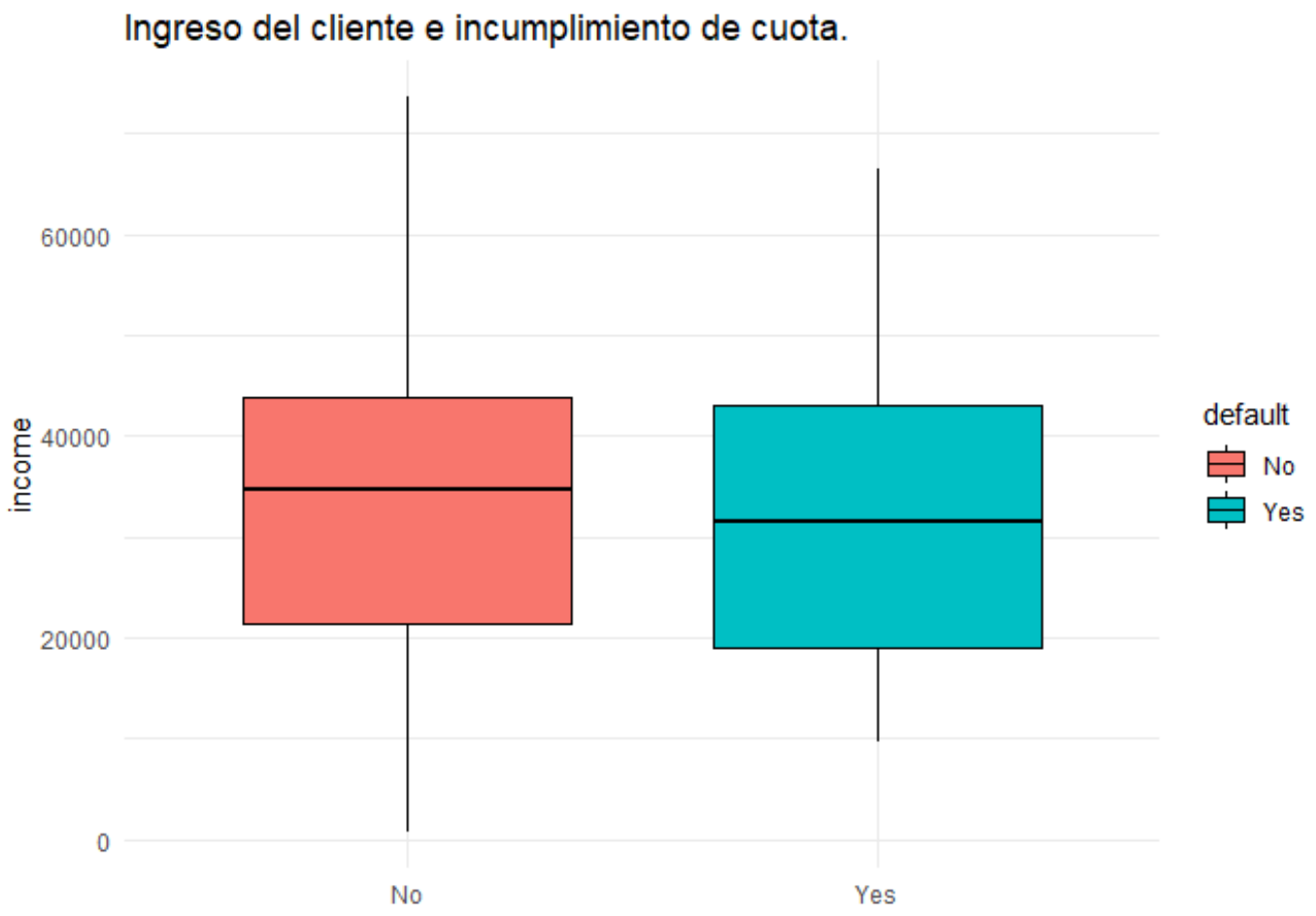



```
#Pareciera que a mayor monto en la cuenta post pago,
#más probable sería el incumplimiento
```

```r
anova(aov(balance~default, data=Default))
```

```
## Analysis of Variance Table
##
## Response: balance
##             Df     Sum Sq   Mean Sq F value    Pr(>F)
## default      1  286792390 286792390  1396.8 < 2.2e-16 ***
## Residuals 9998 2052775499    205319
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#Diferencias entre ambos grupos son significativas

#Income

ggboxplot(Default, y="income", x="default", fill="default")+
  xlab("")+
  ylab("income")+
  ggtitle("Ingreso del cliente e incumplimiento de cuota.")+
  theme_minimal()
```




```
#El grupo de no incumplimiento parecería tener mayores ingresos
#(leve diferencia)

anova(aov(income~default, data=Default))
```

```
## Analysis of Variance Table
##
```

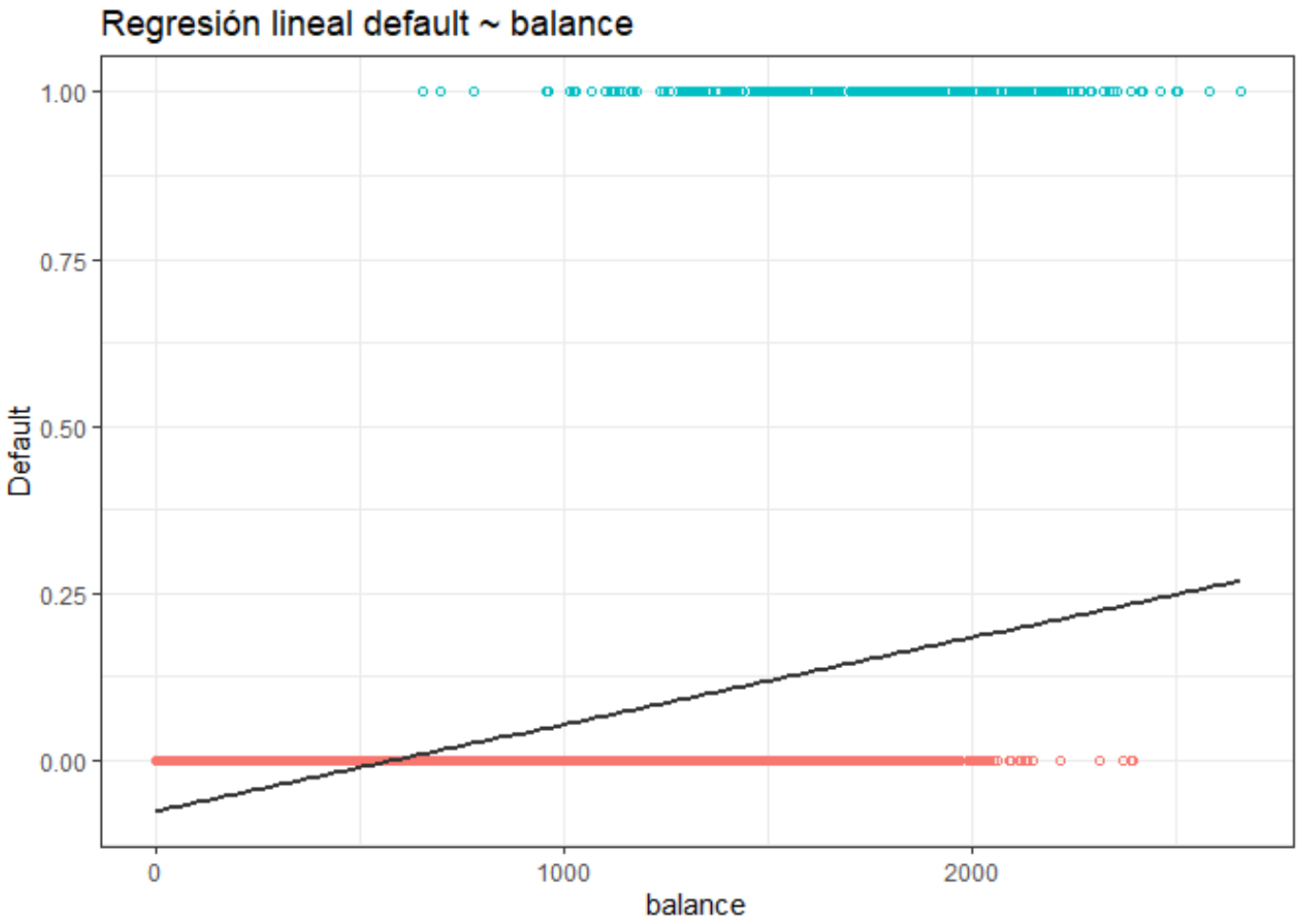
```
## Response: income
##              Df     Sum Sq   Mean Sq F value  Pr(>F)
## default       1 7.0228e+08 702276944  3.9495 0.04691 *
## Residuals  9998 1.7778e+12 177813503
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#Diferencia entre los grupos sería significativa al 5%
#pero está al borde!
```

b) Realice un gráfico que muestre los puntos y el comportamiento de una regresión lineal usando como predictor a Balance. Realice lo mismo pero con un modelo de regresión logística. Explique por qué no sería correcto utilizar una regresión lineal, explique también la idea de la regresión logística. Comente.

```
library(ggplot2)

ggplot(data = Default, aes(x = balance, y = ifelse(default=="Yes", 1, 0))) +
  geom_point(aes(color = as.factor(ifelse(default=="Yes", 1, 0))), shape = 1) +
  geom_smooth(method = "lm", color = "gray20", se = FALSE) +
  theme_bw()  +
  labs(title = "Regresión lineal default ~ balance",
       y = "Default") +
  theme(legend.position = "none")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



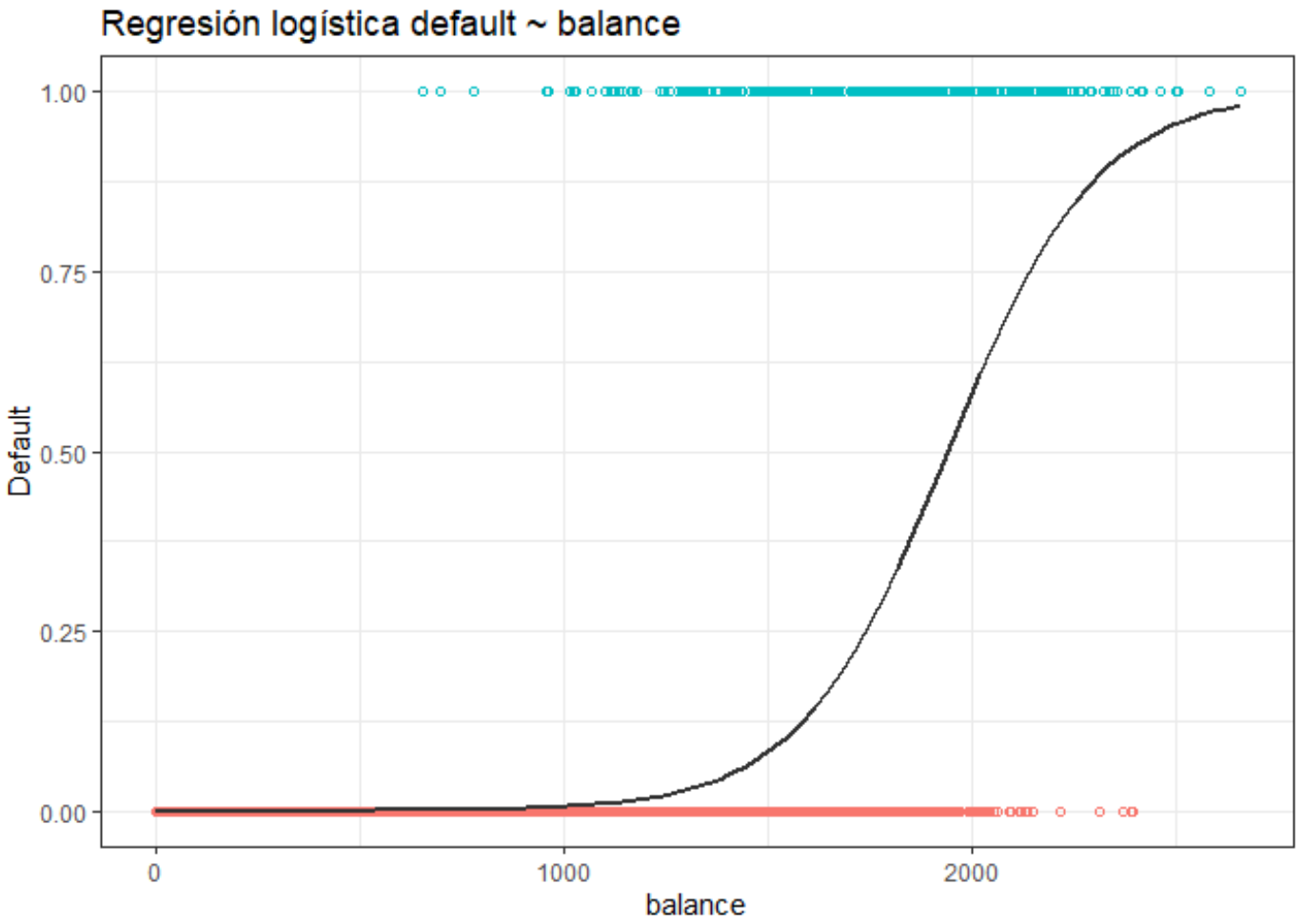
De partida, la variable respuesta es dicotómica, no es numérica, no se cumpliría el supuesto de que $Y|X \sim N()$, por lo tanto habría un serio problema de sustento teórico del modelo.

Además, la regresión te puede dar cualquier cosa, valores sobre 1 y valores #bajo 0, lo cual no tiene mucho sentido en este contexto.

Finalmente, queda muy poco claro cómo interpretar los coeficientes en una regresión lineal, mirando el gráfico, ¿cómo se interpreta el intercepto? ¿y la pendiente? ¿qué significa?

```r
ggplot(data = Default, aes(x = balance, y = ifelse(default=="Yes", 1, 0))) +
  geom_point(aes(color = as.factor(ifelse(default=="Yes", 1, 0))), shape = 1) +
  geom_smooth(method = "glm",
              method.args = list(family = "binomial"),
              color = "gray20",
              se = FALSE) +
  labs(title = "Regresión logística default ~ balance",
       y = "Default") +
  theme_bw() +
  theme(legend.position = "none")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



La regresión logística entrega la probabilidad de ser 1 (éxito o evento), es necesario determinar el punto de corte, desde el cual se clasificará como 1 o 0. Al modelar probabilidades no se escapa de 1 y 0.

c) Obtenga set de entrenamiento y prueba al 70%-30% respectivamente. Luego utilice backward para establecer el modelo final. Comente cuál sería la variable que menos aporta al modelo.

```r
#Se obtienen los set de entrenamiento y testeo:

set.seed(2021)
ind_train <- sample(1:nrow(Default), size = 0.7*nrow(Default), replace = FALSE)

library(dplyr)

data_train <- Default %>%
  slice(ind_train)

data_test <- Default %>%
  slice(-ind_train)

model.1<-glm(default ~ ., data = data_train, family = binomial(link = "logit"))

model.backward<-step(model.1, birection = "backward")
```

```
## Start:  AIC=1143.43
## default ~ student + balance + income
##
##           Df Deviance    AIC
## - income   1   1135.9 1141.9
## <none>         1135.4 1143.4
## - student  1   1141.4 1147.4
## - balance  1   2038.5 2044.5
##
## Step:  AIC=1141.87
## default ~ student + balance
##
```

```
##           Df Deviance    AIC
## <none>         1135.9 1141.9
## - student  1   1158.9 1162.9
## - balance  1   2039.6 2043.6
```

```
model.backward$formula #no aparece income, sería la que menos aporta y por eso se quitó
```

```
## default ~ student + balance
```

d) Interprete los coeficientes del modelo anterior. ¿Qué variables se relacionan con una mayor probabilidad de incumplimiento?

```
coef(model.backward)
```

```
##  (Intercept)   studentYes      balance
## -10.32619865  -0.81691509   0.00549505
```

Ser estudiante se asocia con menor probabilidad de incumplimiento que no serlo a mayor balance mayor probabilidad de incumplimiento (visto en el gráfico).

e) Con la función optimalCutoff() del paquete *InformationValue*, determine cuál es el punto de corte óptimo para predecir si un cliente presentó incumplimiento o no. Evalúe la bondad de ajuste del modelo con el punto de corte encontrado, obtenga:

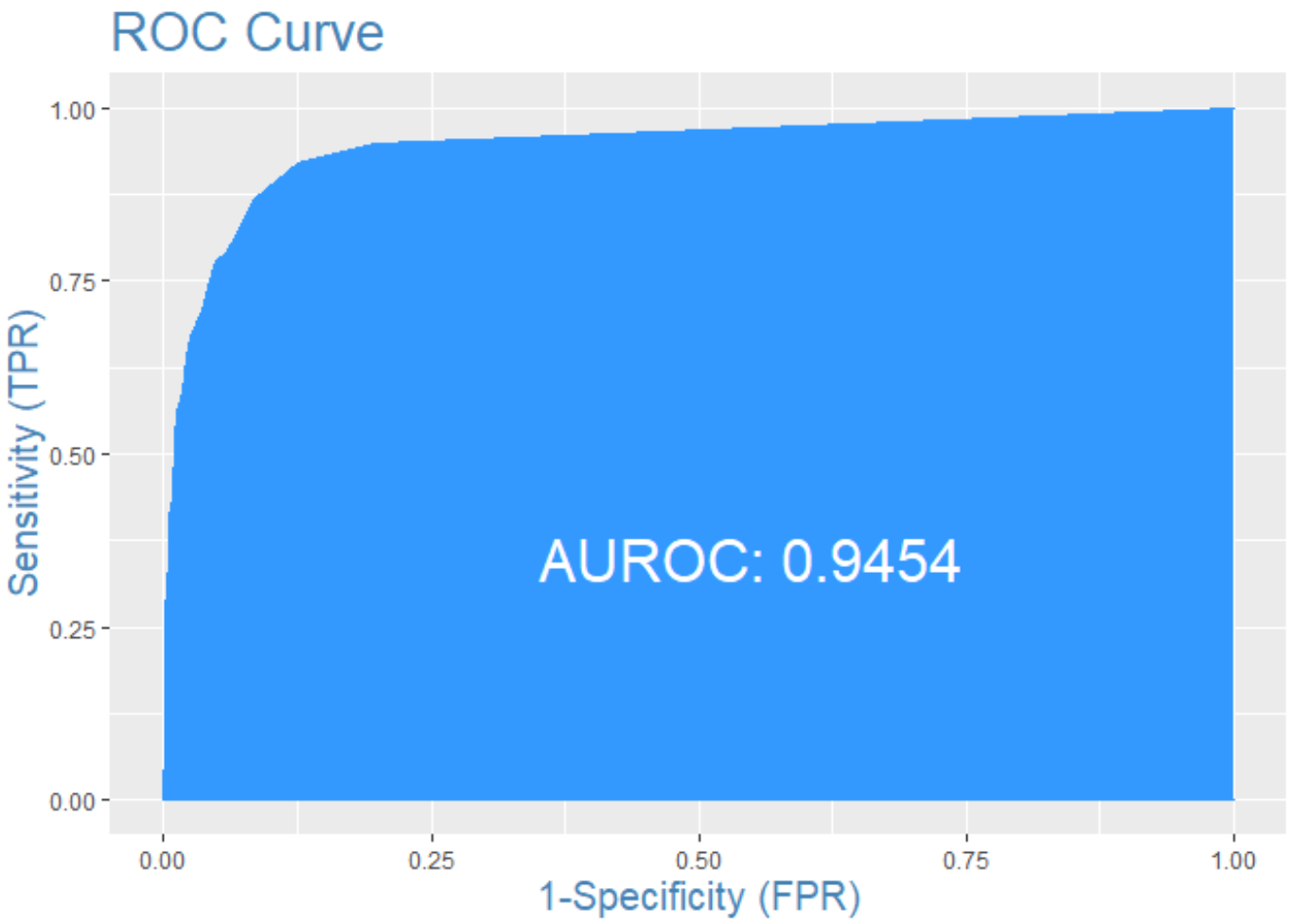
- Curva ROC: ¿En base al área de la curva ROC, usted diría que el modelo es bueno?

- Matriz de confusión: Determine VP, VN, FN, FP y comente qué estaría pasando en este modelo.
- Sensibilidad
- Especificidad
- Precisión

```
library(InformationValue)


predictedscores<-plogis(predict(model.backward, data_test)) #Lo que necesitamos es determina

corte <- optimalCutoff(data_test$default, predictedscores) #Punto de corte

corte
```

```
## [1] 0.009007329
```

```
#Se elige por default el punto de corte que optimiza la tasa de clasificación erronea
#(misclasserror) (ver help de la función)

#Curva ROC
plotROC(ifelse(data_test$default=="Yes", 1, 0), predictedscores)
```




```
#El área está cerca de 1, por lo tanto, A PRIORI viendo sólo el área,
#posee un buen valor para esta métrica.

#Matriz de confusión
confusionMatrix(ifelse(data_test$default=="Yes", 1, 0), predictedscores, threshold = corte)
```

```
##      0  1
## 0 2062  3
## 1  838 97
```

```
#Lo que podemos ver es que hay demasiados casos negativos por sobre los positivos
#el modelo podría entregar puros "No" y le achuntaría a casi todos los registros
#hay que tener cuidado con los casos desbalanceados.

#Sensibilidad: TP/(TP+FN)=97/(97+3)
sensitivity(ifelse(data_test$default=="Yes", 1, 0), predictedscores, threshold = corte)
```

```
## [1] 0.97
```

```
##Indica de todos los positivos, cuántos fueron correctamente predichos
#pero notar que son pocos los positivos!

#Especificidad: TN/(TN+FP)=2062/(2062+838)
specificity(ifelse(data_test$default=="Yes", 1, 0), predictedscores, threshold = corte)
```

```
## [1] 0.7110345
```

```
##Indica de todos los negativos, cuántos fueron correctamente predichos
#Es menor la Especificidad, esto porque hubieron 838 casos donde el modelo
#predijo que cliente incumple y no incumple.
```

```r
#Precisión: TP/(TP+FP)=97/(97+838)
precision(ifelse(data_test$default=="Yes", 1, 0), predictedscores, threshold = corte)
```

```
## [1] 0.1037433
```

```r
##Indica de todos los positivos predichos, cuántos realmente eran positivos

#El modelo entrega demasiadas predicciones positivas de las que son!
#Precisión muy baja.


#Conclusión: Deben mirarse todas las métricas, no sólo una! Pues todas
#nos ayudan a comprender el rendimiento del modelo a nivel más macro.


#También se podría probar otros puntos de corte e ir viendo
#cómo cambian las métricas :D
```