Regresión Logística y Näive bayes. Detección de churn bancario

Adolfo Sánchez Burón

Introducción al algoritmo de regresión logística

El Análisis de regresión logística es el indicado cuando queremos predecir una variable categórica binaria, en función de un conjunto de predictores. Cuando solo tenemos un predictor se le denomina RL simple, y cuando hay más de dos, RL múltiple.

La lógica de este estadístico es la siguiente: en función de una serie de predictores queremos predecir la probabilidad de que un caso pertenezca a una de las dos posibilidades de la variable binaria.

Si bien, la RL aporta una determinada probabilidad de ocurrencia para cada caso de pertenecer a una de las dos categorías, posteriormente se tendrá que establecer un límite (threshold) para determinar la pertenencia o no a una de esas dos categorías, el cual podrá ser más o menos restrictivo dependiendo de las características del estudio.

Por tanto, la función logística producirá una curva en S, indicando los valores 0 y 1, lo que se diferencia del análisis de regresión lineal, cuya variable dependiente es una variable cuantitativa continua, y reporta una linea recta.

Introducción al algoritmo de näive bayés

NB es un modelo predictivo de clasificación basado en el teorema de Bayés, el cual se basa en que la probabilidad de ocurrencia de cada variable es independiente de las demás. En otras palabras, la presencia de una variable no está relacionada con la probabilidad de ocurrencia de otra variable. Pero a medida que vamos teniendo más datos podemos asociar la presencia de dicha variable (independientemente de las demás) con la característica de clasificación.

Un ejemplo muy intuitivo lo describe JacobSoft (https://www.jacobsoft.com.mx/es_mx/clasificador-naive-bayes/) : una manzana puede ser clasificada como tal en función de una serie de características. Pero la presencia o ausencia de cada una de ellas es independiente (por ejemplo, color forma o diámetro). Es decir, cada característica contribuye de manera independiente a la probabilidad de que finalmente clasifiquemos un objeto como “manzana”.

El teorema de Bayés se representaría de la siguiente manera:

P(h|D) = P (D|h) * P(h) / P(D)

O en otros términos:

P(posteriori) = P(probabilidad condicional) * P(a priori) / P(total)

P(h) es la probabilidad a priori de obtener h sin ninguna observación.

P(D) es la probabilidad a priori de observar D, sin saber qué hipótesis se verifica.

P(h|D) es la probabilidad resultante o a posteriori de h, de que h sea cierta después de observar D.

P(D|h) es la probabilidad condicional de observar D en un universo donde se verifica la hipótesis h.

Se recomienda la lectura de:

Villalba,F. Capítulo 5 Naive Bayes- clasificación bayesiano ingenuo (https://fervilber.github.io/Aprendizaje-supervisado-en-R/ingenuo.html)

Navas,A. Algoritmo de Naive Bayes (https://rpubs.com/Kataniss/638768)

Características del caso

El caso empleado en este análisis es el ‘Predicting Churn for Bank Customers’, que puede descargarse el dataset original de Kaggle (https://www.kaggle.com/adammaus/predicting-churn-for-bank-customers). Este dataset ha sido previamente trabajado en cuanto a:

- análisis descriptivo
- limpieza de anomalías, missing y outliers
- peso predictivo de las variables mediante random forest
- discretización de las variables continuas para facilitar la interpretación posterior

Por lo que finalmente se emplea en este caso un dataset preparado para iniciar el análisis de RL, que puede descargarse de Github.

El objetivo del caso es predecir la probabilidad de que un determinado cliente puede abandonar (churn) el Banco. La explicación de esta conducta estará basada en toda una serie de variables predictoras: RowNumber, CustomerId, Surname, CreditScore, Geography, Gender, Age, Tenure, Balance, NumOfProducts, HasCrCard, IsActiveMember, EstimatedSalary y Exited.

Posteriormente se hará una descripción de las variables seleccionadas para el análisis y su composición.

Proceso

1. Entorno

El primer punto tratará sobre la preparación del entorno, donde se mostrará la descarga de las librerías empleadas y la importación de datos.

2. Análisis descriptivo

Se mostrarán y explicarán las funciones empleadas en este paso, dividiéndolas en tres grupos: Análisis inicial, Tipología de datos y Análisis descriptivo (gráficos).

3. Preparar modelización

Se preparará lo necesario para modelizar, mediante dos pasos:

a. Preparar funciones:

- Matriz de confusión
- Métricas
- Umbrales
- Curva ROC y AUC

b. Particiones del dataset en dos grupos: training (70%) y test (30%)

4. Modelización

Por motivos didácticos, se dividirán los algoritmos seleccionados en varios pasos.

---

1. Entorno

1.1. Instalar librerías

```
library(data.table) #para leer y escribir datos de forma rapida
library(dplyr) #para manipulación de datos
library(tidyr) #para manipulación de datos
library(ROCR) #para evaluar modelos
library(DataExplorer) #para realizar el análisis descriptivo con gráficos
library(e1071) #para modelizar con Näive Bayés
```

1.2. Importar datos

Como el dataset ha sido peviamente trabajado para poder modelizar directamente, si deseas seguir este tutorial, lo puedes descargar de GitHub (https://github.com/AdSan-R).

```
df <- read.csv("BankChurn2.csv")
```

```
options(scipen=999)
#Desactiva la notación científica
```

2. Análisis descriptivo

2.1. Análisis inicial

```
head(df) #ver la estructura de los primeros 6 casos
```

```
##   Geography IsActiveMember     RNumOfProducts TARGET CreditScoreR
## 1    France             Si        Un producto     Si   Medio-Alto
## 2     Spain             Si        Un producto     No   Medio-Alto
## 3    France             No Más de un producto     Si   Medio-Bajo
## 4    France             No Más de un producto     No   Medio-Alto
## 5     Spain             Si        Un producto     No         Alto
## 6     Spain             No Más de un producto     Si   Medio-Alto
##              AgeR    TenureR          BalanceR EstimatedSalaryR
## 1 De 41 a 50 años  Hasta 2,5       Hasta 80000       Medio-Alto
## 2 De 41 a 50 años  Hasta 2,5 De 80001 a 120000       Medio-Alto
## 3 De 41 a 50 años Más de 7,5     Más de 120001       Medio-Alto
## 4 De 31 a 40 años  Hasta 2,5       Hasta 80000       Medio-Alto
## 5 De 41 a 50 años  Hasta 2,5     Más de 120001       Medio-Bajo
## 6 De 41 a 50 años Más de 7,5 De 80001 a 120000             Alto
```

2.2. Tipología de datos

```
str(df) #mostrar la estructura del dataset y los tipos de variables
```

```
## 'data.frame':    10000 obs. of  9 variables:
##  $ Geography       : chr  "France" "Spain" "France" "France" ...
##  $ IsActiveMember  : chr  "Si" "Si" "No" "No" ...
##  $ RNumOfProducts  : chr  "Un producto" "Un producto" "Más de un producto" "Más de
un producto" ...
##  $ TARGET          : chr  "Si" "No" "Si" "No" ...
##  $ CreditScoreR    : chr  "Medio-Alto" "Medio-Alto" "Medio-Bajo" "Medio-Alto" ...
##  $ AgeR            : chr  "De 41 a 50 años" "De 41 a 50 años" "De 41 a 50 años" "De
31 a 40 años" ...
##  $ TenureR         : chr  "Hasta 2,5" "Hasta 2,5" "Más de 7,5" "Hasta 2,5" ...
##  $ BalanceR        : chr  "Hasta 80000" "De 80001 a 120000" "Más de 120001" "Hasta
80000" ...
##  $ EstimatedSalaryR: chr  "Medio-Alto" "Medio-Alto" "Medio-Alto" "Medio-Alto" ...
```

Puede observarse que todas son “chr”, esto es, “character”, por tanto, vamos a pasarlas a Factor.

```
df <- mutate_if(df, is.character, as.factor) #identifica todas las character y las pas
a a factores
#Sacamos la esructura
str(df)
```

```
## 'data.frame':    10000 obs. of  9 variables:
##  $ Geography       : Factor w/ 3 levels "France","Germany",..: 1 3 1 1 3 3 1 2 1 1
...
##  $ IsActiveMember  : Factor w/ 2 levels "No","Si": 2 2 1 1 2 1 2 1 2 2 ...
##  $ RNumOfProducts  : Factor w/ 2 levels "Más de un producto",..: 2 2 1 1 2 1 1 1 1
2 ...
##  $ TARGET          : Factor w/ 2 levels "No","Si": 2 1 2 1 1 2 1 2 1 1 ...
##  $ CreditScoreR    : Factor w/ 4 levels "Alto","Bajo",..: 3 3 4 3 1 3 1 2 4 3 ...
##  $ AgeR            : Factor w/ 5 levels "De 31 a 40 años",..: 2 2 2 1 2 2 2 4 2 4
...
##  $ TenureR         : Factor w/ 4 levels "De 2,6 a 5,0",..: 3 3 4 3 3 4 2 1 1 3 ...
##  $ BalanceR        : Factor w/ 3 levels "De 80001 a 120000",..: 2 1 3 2 3 1 2 1 3 3
...
##  $ EstimatedSalaryR: Factor w/ 5 levels "Alto","Bajo",..: 3 3 3 3 4 1 2 3 4 4 ...
```

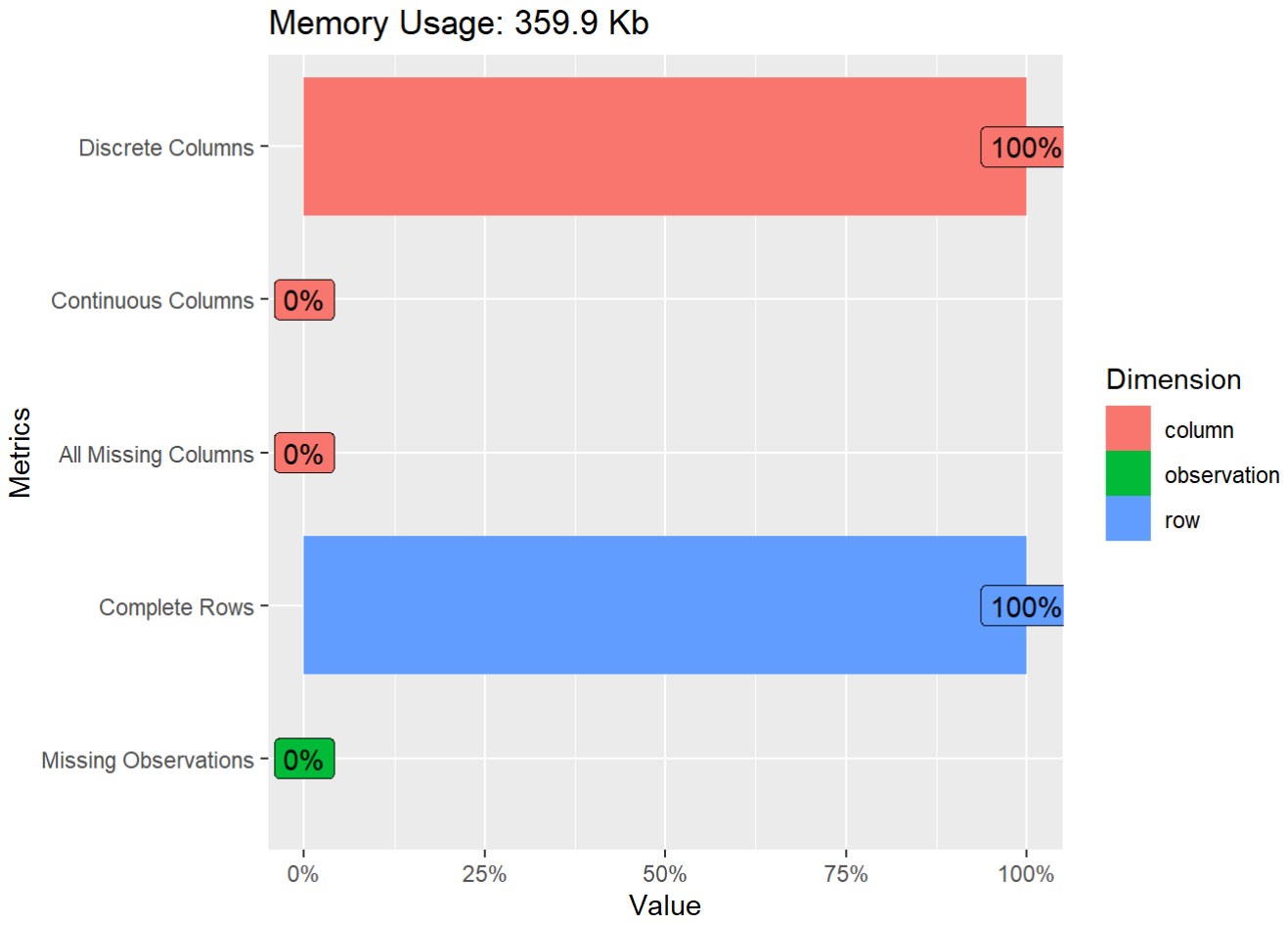
Ahora se puede observar que todas las variables son de tipo “Factor”

```
lapply(df,summary) #mostrar la distribución de frecuencias en cada categoría de todas
 las variables
```

```
## $Geography
##  France Germany   Spain
##    5014    2509    2477
##
## $IsActiveMember
##   No   Si
## 4849 5151
##
## $RNumOfProducts
## Más de un producto        Un producto
##               4916               5084
##
## $TARGET
##   No   Si
## 7963 2037
##
## $CreditScoreR
##       Alto       Bajo Medio-Alto Medio-Bajo
##       2270        357       4664       2709
##
## $AgeR
## De 31 a 40 años De 41 a 50 años De 51 a 65 años   Hasta 30 años   Más de 65años
##            4451            2320             797            1968             464
##
## $TenureR
## De 2,6 a 5,0 De 5,1 a 7,5    Hasta 2,5   Más de 7,5
##         3010         1995         2496         2499
##
## $BalanceR
## De 80001 a 120000       Hasta 80000     Más de 120001
##              2617              4202              3181
##
## $EstimatedSalaryR
##       Alto       Bajo Medio-Alto Medio-Bajo   Muy Alto
##       1989       1955       2029       2033       1994
```

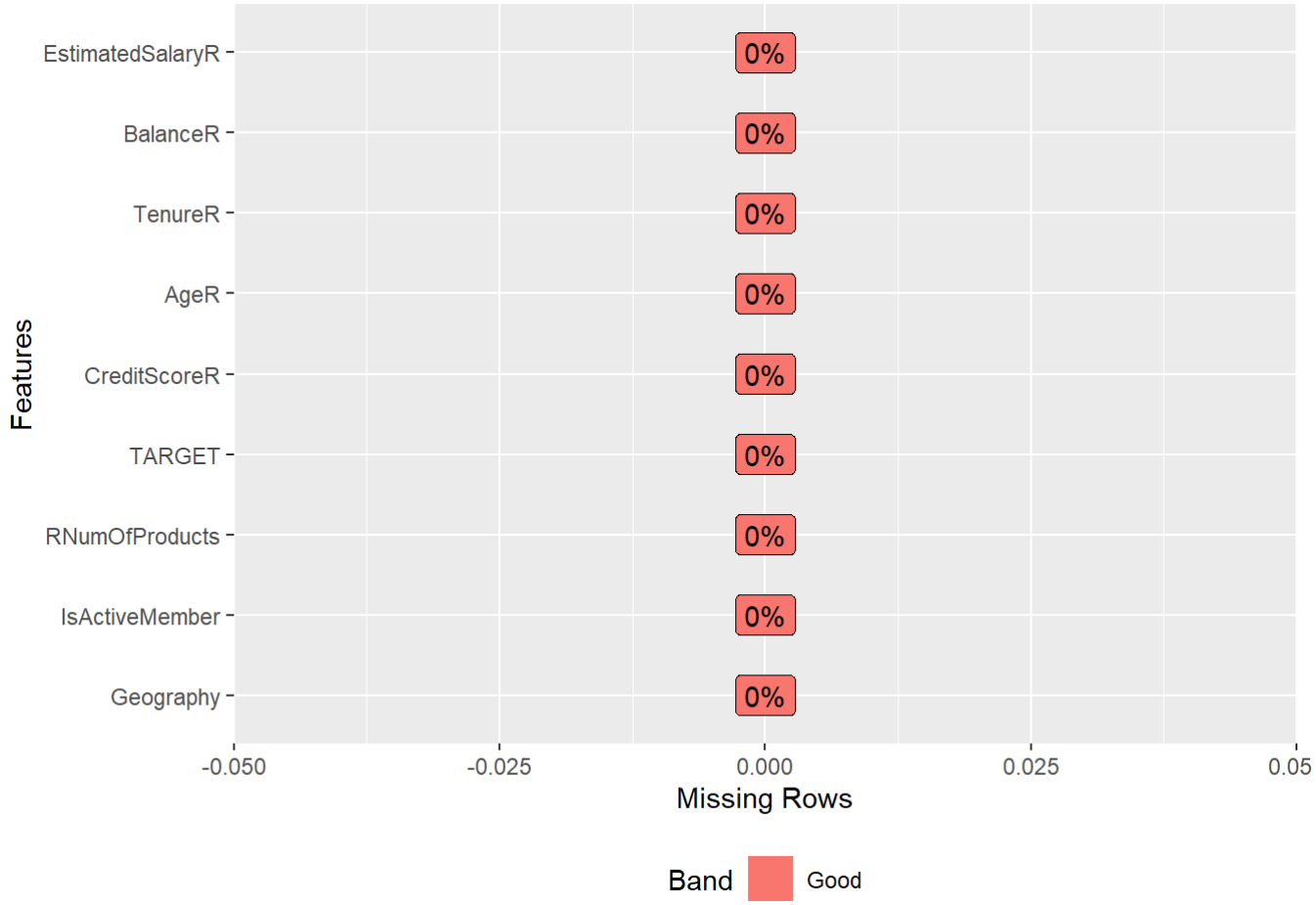
2.3. Análisis descriptivo (gráficos)

```
plot_intro(df) #gráfico para observar la distribución de variables y los casos missing por columnas, observaciones y filas
```

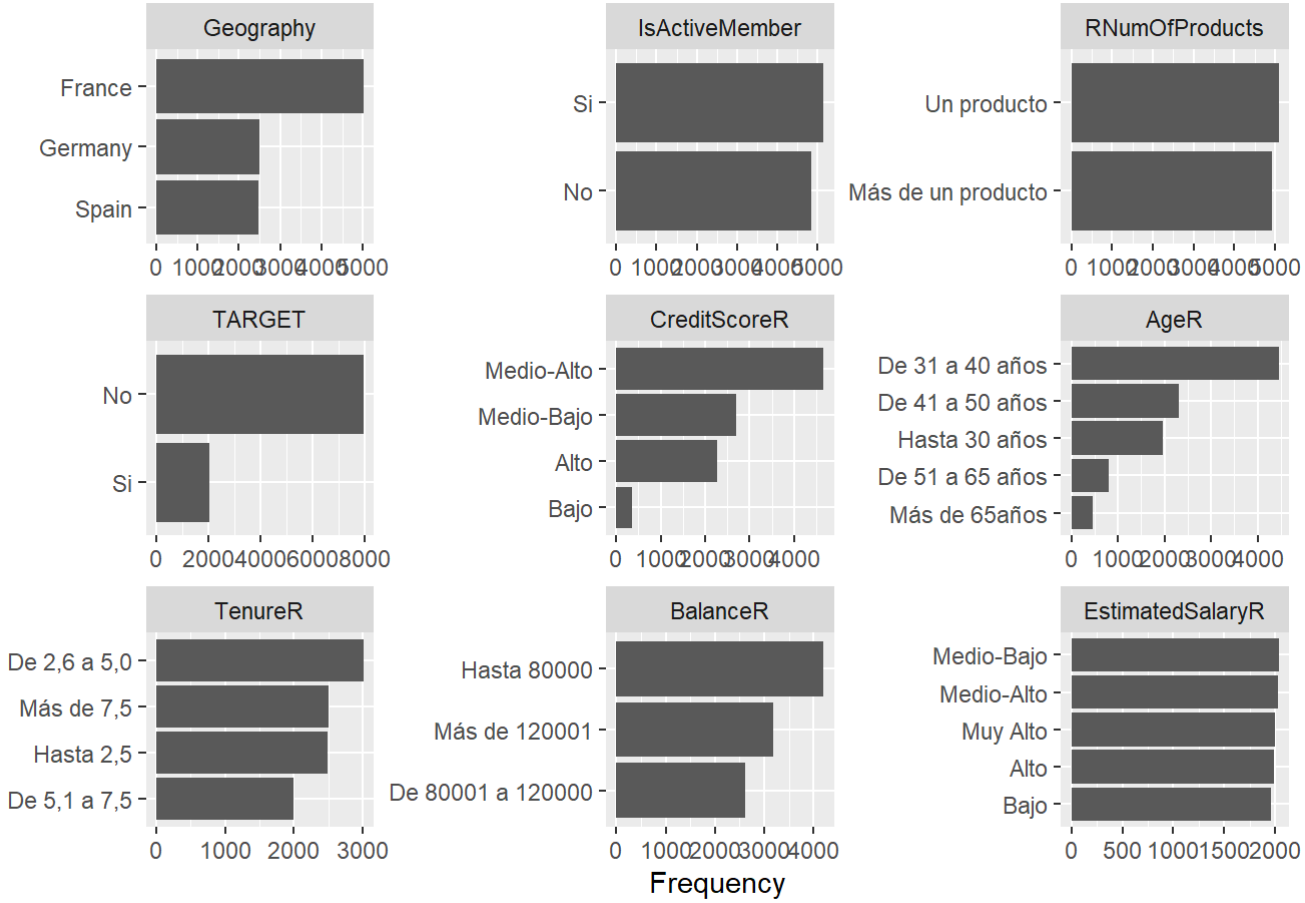


Como se ha trabajado previamente, no existen casos missing, por lo que podemos seguir el análisis descriptivo. De todas formas, aunque lógicamente no hace falta, por motivos didácticos sacamos un gráfico para corroborar lo anterior.

```
plot_missing(df)
```




```
plot_bar(df) #gráfico para observar la distribución de frecuencias en variables categóricas
```



En los gráficos anteriores pueden observarse las categorías de cada variable, algunas de ellas dicotomizadas previamente, por lo que haremos un repaso de cada una:

- TARGET: variable dependiente, dividida en dos niveles Sí hacen Cgurn y NO hacen churn. Se distribuye ne un 80%-20%.
- Geography: tres áreas georgráficas, France, Germany y Spain.
- IsActiveMember: dos niveles, Sí y NO, con distribución muy equilibrada.
- RNumOfProducts: variable recodificada en dos niveles debido a que alguna categoría mostraba una frecuencia muy reducida; Un producto o Más de un producto.
- CreditScoreR: variable discretizada por cuartiles en cuatro categorías; Bajo, Medio_Bajo, Medio-Alto y Alto.
- AgeR: variable discretizada desde la variable continua de Edad en cinco categorías,'Hasta 30 años', 'De 31 a 40 años', 'De 41 a 50 años', 'De 51 a 65 años' y 'Más de 65años'
- TenureR: variable recodificada a cuatro categorías; hasta 2,5, de 2,6 a 5, de 5,1 a 7,5 y más de 7,5.
- BalanceR: variable recodificada a 3 niveles; 'Hasta 80000', 'De 80001 a 120000' y 'Más de 120001'.
- EstimatedSalary: variable categorizada en cinco niveles, "Bajo", "Medio-Bajo", "Medio-Alto", "Alto", "Muy Alto"

Tras aplicar las técncias de Random forest e Information value para determinar el poder predictivo de las variables predictoras sobre la variable dependiente, se optó por eliminar del análisis a las variables: RowNumber, CustomerId, Surname (estas tres anteriores porque no aportan nada al análisis), Gender y HasCrCard.

3. Modelización

3.1. Preparar funciones

Tomadas del curso de Machine Learning Predictivo (https://www.datascience4business.com/08_mlc-salespage-b) de DS4B) :

- Matriz de confusión
- Métricas
- Umbrales
- Curva ROC y AUC

Función para la matriz de confusión

En esta función se prepara la matriz de confusión (ver en otro post), donde se observa qué casos coinciden entre la puntuación real (obtenida por cada sujeto) y la puntuación predicha ("scoring") por el modelo, estableciendo previmente un límite ("umbral") para ello.

```
confusion<-function(real,scoring,umbral){
  conf<-table(real,scoring>=umbral)
  if(ncol(conf)==2) return(conf) else return(NULL)
}
```

Funcion para métricas de los modelos

Los indicadores a observar serán:

- Acierto (accuracy) = (TRUE POSITIVE + TRUE NEGATIVE) / TODA LA POBLACIÓN
- Precisión = TRUE POSITIVE / (TRUE POSITIVE + FALSE POSITIVE)
- Cobertura (recall, sensitivity) = TRUE POSITIVE / (TRUE POSITIVE + FALSE NEGATIVE)
- F1 = 2* (precisión * cobertura) (precisión + cobertura)

```
metricas<-function(matriz_conf){
  acierto <- (matriz_conf[1,1] + matriz_conf[2,2]) / sum(matriz_conf) *100
  precision <- matriz_conf[2,2] / (matriz_conf[2,2] + matriz_conf[1,2]) *100
  cobertura <- matriz_conf[2,2] / (matriz_conf[2,2] + matriz_conf[2,1]) *100
  F1 <- 2*precision*cobertura/(precision+cobertura)
  salida<-c(acierto,precision,cobertura,F1)
  return(salida)
}
```

Función para probar distintos umbrales

Con esta función se analiza el efecto que tienen distintos umbrales sobre los indicadores de la matriz de confusión (precisión y cobertura). Lo que buscaremnos será aquél que maximice la relación entre cobertura y precisión (F1).

```
umbrales<-function(real,scoring){
  umbrales<-data.frame(umbral=rep(0,times=19),acierto=rep(0,times=19),precision=rep(0,
times=19),cobertura=rep(0,times=19),F1=rep(0,times=19))
  cont <- 1
  for (cada in seq(0.05,0.95,by = 0.05)){
    datos<-metricas(confusion(real,scoring,cada))
    registro<-c(cada,datos)
    umbrales[cont,]<-registro
    cont <- cont + 1
  }
  return(umbrales)
}
```

Funciones para calcular la curva ROC y el AUC

Por último, se preapra una función para calcular la curva ROC y el AUC.

- Curva ROC (Relative Operating Characteristic): resentación gráfica de la relación entre la cobertura (proporción de verdaderos positivos) y la especificidad (razón de falsos positivos). Muestra el rendimiento del modelo en todos los umbrales de clasificación.

- AUC (Area Under The Curve): mide el área que queda debajo de la curva. Indica en qué medida el modelo será capaz de clasificar adecuadamente. La AUC tiene un rango entre 0 y 1. Si es igual o cercano a 0.5, no tiene capacidad discriminativa.

```
roc<-function(prediction){
  r<-performance(prediction,'tpr','fpr')
  plot(r)
}

auc<-function(prediction){
  a<-performance(prediction,'auc')
  return(a@y.values[[1]])
}
```

3.2. Particiones de training (70%) y test (30%)

Se divide la muestra en dos partes:

1. Training o entrenamiento (70% de la muestra): servirá para entrenar al modelo de clasificación.
2. Test (30%): servirá para validar el modelo. La característica fundamental es que esta muestra no debe haber tenido contacto previamente con el funcionamiento del modelo.

```
# Lanzamos una semilla para que salgan siempre los mismos datos
set.seed(12345)

# Creamos los dataframes

# Generamos una variable aleatoria con una distribución 70-30
df$random<-sample(0:1,size = nrow(df),replace = T,prob = c(0.3,0.7))

train<-filter(df,random==1)
test<-filter(df,random==0)
#Eliminamos ya la random para que no moleste
df$random <- NULL
```

---

4. Modelización con regresión logística

Paso 1. Primer modelo

Primero vamos a hacer un modelo con todas las variables seleccionadas y lo incluimos en un objeto llamado “rl”

```
rl<- glm(TARGET ~ Geography + IsActiveMember + RNumOfProducts + CreditScoreR + AgeR +
TenureR + BalanceR + EstimatedSalaryR, train, family=binomial(link='logit'))
# glm: Lanzamos la función glm para entre un modelo de RL, de la familia "binomial, "l
ogit".
# TARGET ~ InternetService + Contract + PaymentMethod + tenure_DISC + MonthlyCharges_D
ISC + TotalCharges_DISC : es el modelo a entrenar, con VD la TARGET, y el resto son lo
s predictores.
# train: solo lo lanzamos con el df "train", para entrenar el modelo.

# summary(rl): función para ver los resultados del primer modelo
summary(rl)
```

```
##
## Call:
## glm(formula = TARGET ~ Geography + IsActiveMember + RNumOfProducts +
##     CreditScoreR + AgeR + TenureR + BalanceR + EstimatedSalaryR,
##     family = binomial(link = "logit"), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0379  -0.6461  -0.4111  -0.2393   2.8179
##
## Coefficients:
##                             Estimate Std. Error z value             Pr(>|z|)
## (Intercept)                -2.19651    0.15036 -14.608 < 0.0000000000000002 ***
## GeographyGermany            0.90959    0.08439  10.779 < 0.0000000000000002 ***
## GeographySpain              0.06236    0.08626   0.723                0.470
## IsActiveMemberSi           -1.02709    0.07059 -14.550 < 0.0000000000000002 ***
## RNumOfProductsUn producto   0.85920    0.07226  11.890 < 0.0000000000000002 ***
## CreditScoreRBajo            0.17267    0.18331   0.942                0.346
## CreditScoreRMedio-Alto     -0.08389    0.08613  -0.974                0.330
## CreditScoreRMedio-Bajo     -0.02948    0.09511  -0.310                0.757
## AgeRDe 41 a 50 años         1.32124    0.08026  16.462 < 0.0000000000000002 ***
## AgeRDe 51 a 65 años         2.38858    0.11063  21.590 < 0.0000000000000002 ***
## AgeRHasta 30 años          -0.49234    0.11876  -4.146            0.0000339 ***
## AgeRMás de 65años           1.35957    0.14843   9.160 < 0.0000000000000002 ***
## TenureRDe 5,1 a 7,5        -0.15560    0.09790  -1.589                0.112
## TenureRHasta 2,5           -0.01974    0.09039  -0.218                0.827
## TenureRMás de 7,5          -0.09996    0.08987  -1.112                0.266
## BalanceRHasta 80000        -0.04730    0.09286  -0.509                0.611
## BalanceRMás de 120001      -0.02331    0.08316  -0.280                0.779
## EstimatedSalaryRBajo       -0.04968    0.10742  -0.462                0.644
## EstimatedSalaryRMedio-Alto -0.08445    0.10570  -0.799                0.424
## EstimatedSalaryRMedio-Bajo -0.00144    0.10604  -0.014                0.989
## EstimatedSalaryRMuy Alto    0.02470    0.10524   0.235                0.814
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 7102.5  on 7007  degrees of freedom
## Residual deviance: 5681.4  on 6987  degrees of freedom
## AIC: 5723.4
##
## Number of Fisher Scoring iterations: 5
```

Revisamos la significatividad DE RL y mantenemos todas las variables que tengan tres estrellas en alguna categoría, Parece que solo cumplen esta condición: Geography, IsActiveMember, RNumOfProducts y AgeR.

Volvemos a modelizar solo con esas variables.

Paso 2. Segundo modelo

```
rl2<- glm(TARGET ~ Geography + IsActiveMember + RNumOfProducts + AgeR, train, family=b
inomial(link='logit'))
summary(rl2)
```

```
##
## Call:
## glm(formula = TARGET ~ Geography + IsActiveMember + RNumOfProducts +
##     AgeR, family = binomial(link = "logit"), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9887  -0.6388  -0.4063  -0.2589   2.7896
##
## Coefficients:
##                            Estimate Std. Error z value             Pr(>|z|)
## (Intercept)                -2.35228    0.08480 -27.738 < 0.0000000000000002 ***
## GeographyGermany            0.92536    0.07720  11.986 < 0.0000000000000002 ***
## GeographySpain              0.05992    0.08606   0.696                0.486
## IsActiveMemberSi           -1.02708    0.07042 -14.585 < 0.0000000000000002 ***
## RNumOfProductsUn producto   0.86650    0.06903  12.553 < 0.0000000000000002 ***
## AgeRDe 41 a 50 años         1.31745    0.08006  16.456 < 0.0000000000000002 ***
## AgeRDe 51 a 65 años         2.38879    0.11048  21.623 < 0.0000000000000002 ***
## AgeRHasta 30 años          -0.49095    0.11856  -4.141            0.0000346 ***
## AgeRMás de 65años           1.36134    0.14816   9.188 < 0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 7102.5  on 7007  degrees of freedom
## Residual deviance: 5689.0  on 6999  degrees of freedom
## AIC: 5707
##
## Number of Fisher Scoring iterations: 5
```

Vemos que ahora ya todas las variables tienen al menos una categoría con 3 estrellas de significación

Calculamos el pseudo R cuadrado de McFadden (“residual deviance” / “null deviance”): los resultados entre 0,2 y 0,4 indican un excelente ajuste del modelo.

```
pr2_rl <- 1 -(rl2$deviance / rl$null.deviance)
pr2_rl
```

```
## [1] 0.1990203
```

Paso 3. Predict

Aplicamos el modelo entrenado al conjunto de test (30%), generando un vector con las probabilidades en cada caso de ser 0 o 1.

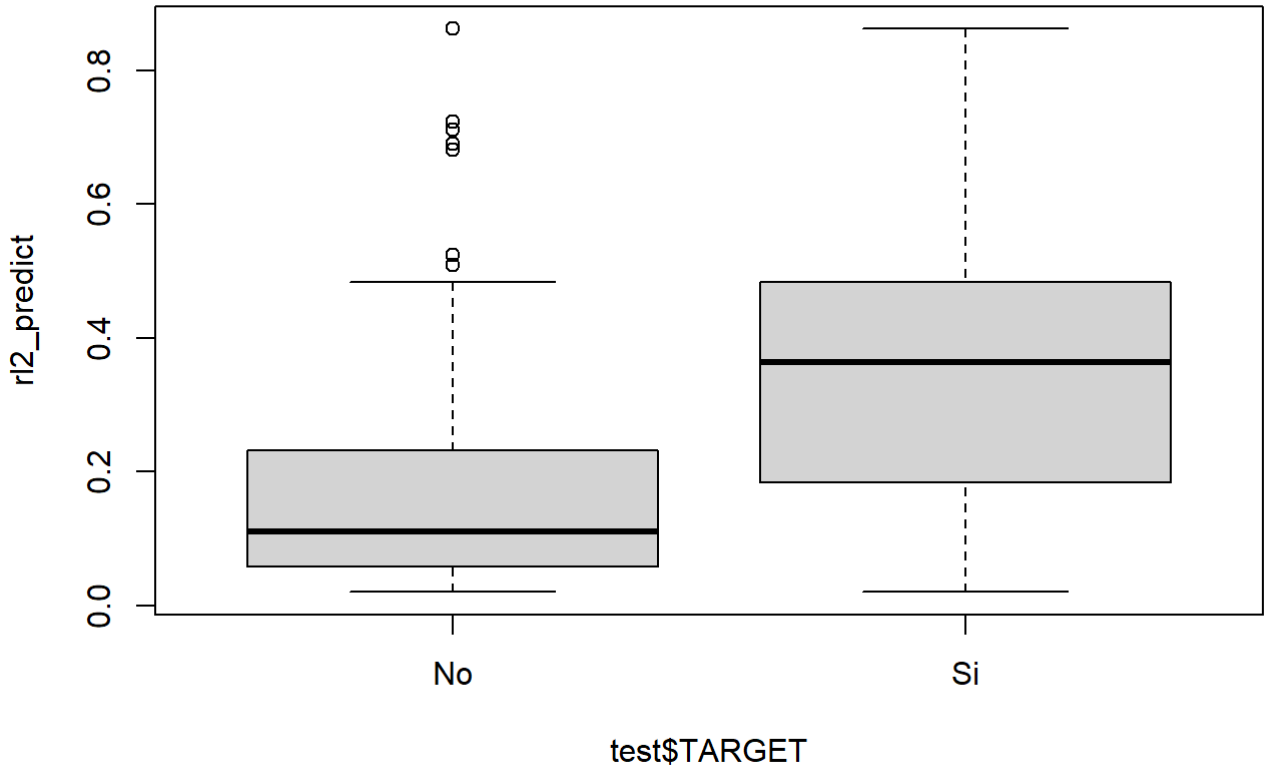
```
rl2_predict<-predict(rl2,test,type = 'response')
#type= 'response'. obtener de cada caso la probabilidqad de churn, lo que me permitirá posteriormente trabajar con umbrales
head(rl2_predict)
```

```
##         1         2         3         4         5         6
## 0.23229280 0.24314952 0.26214926 0.08688505 0.11285486 0.04725407
```

Lanzamos un "head" para ver los 6 primeros. Lo que quiere decir que: el sujeto 1 tendrá una probabilidad de clasificarse como 1 (Sí Churn) del 23,23%. El segundo de 24,31%, etc.

A continuación lanzamos un plot de caja y bigotes, para ver si discrimina bien entre las dos categorías, esto es, si la media de rl_predict de los clientes que sí contratan con la media de los que no contratan es diferente.

```
plot(rl2_predict~test$TARGET)
```




```
#COMPARA LA MEDIA DE rl_predict DE LOS CLIENTES QUE SÍ CONTRATAN CON LA MEDIA DE LOS QUE NO CONTRATAN.
#OBSERVAMOS LA GRÁFICA Y VEMOS QUE LA MEDIA DE LOS QUE SÍ Y LOS QUE NO ES MUY DIFERENTE, INCLUSO DISCRIMINA BIEN ENTRE LOS CUARTILES.
```

Se observa en la gráfica que la media de los que Sí y los que No es muy diferente, incluso discrimina bien entre los cuartiles.

Paso 4. Umbrales

Ahora tenemos que transformar la probabilidad obtenida en una decisión de si el cliente va a abandonar o no.

Con la función umbrales probamos diferentes cortes

```
umb_rl2<-umbrales(test$TARGET,rl2_predict)
umb_rl2
```

```
##    umbral  acierto precision cobertura        F1
## 1    0.05 34.72594  23.15036  96.51741 37.343600
## 2    0.10 56.45053  29.79215  85.57214 44.197002
## 3    0.15 65.14037  34.35277  80.09950 48.083624
## 4    0.20 73.36230  40.43393  67.99337 50.711194
## 5    0.25 77.23930  45.22059  61.19403 52.008457
## 6    0.30 81.14973  53.27731  52.57048 52.921536
## 7    0.35 81.14973  53.27731  52.57048 52.921536
## 8    0.40 82.21925  56.97446  48.09287 52.158273
## 9    0.45 82.08556  57.42794  42.95191 49.146110
## 10   0.50 83.42246  78.01047  24.70978 37.531486
## 11   0.55 83.15508  78.28571  22.71973 35.218509
## 12   0.60 83.15508  78.28571  22.71973 35.218509
## 13   0.65 83.15508  78.28571  22.71973 35.218509
## 14   0.70 81.95187  90.90909  11.60862 20.588235
## 15   0.75 80.54813  88.88889   3.98010  7.619048
## 16   0.80 80.54813  88.88889   3.98010  7.619048
## 17   0.85 80.54813  88.88889   3.98010  7.619048
## 18   0.90  0.90000   0.90000   0.90000  0.900000
## 19   0.95  0.95000   0.95000   0.95000  0.950000
```

Seleccionamos el umbral que maximiza la F1 (cuando empieza a decaer)

```
umbral_final_rl2<-umb_rl2[which.max(umb_rl2$F1),1]
umbral_final_rl2
```

```
## [1] 0.3
```

Como puede observarse en la tabla anterior, el indicador F1 crece a medida que los umbrales aumentan (esto es, se maximiza progresivamente la F1), pero llega a un punto que empieza a decrecer: umbral de 0.3

Paso 5. Matriz de confusión

Evaluamos la matriz de confusión y las métricas con el umbral optimizado

```
confusion(test$TARGET,rl2_predict,umbral_final_rl2)
```

```
##
## real FALSE TRUE
##   No  2111  278
##   Si   286  317
```

```
rl2_metricas<-filter(umb_rl2,umbral==umbral_final_rl2)
rl2_metricas
```

```
##   umbral  acierto precision cobertura       F1
## 1    0.3 81.14973  53.27731  52.57048 52.92154
```

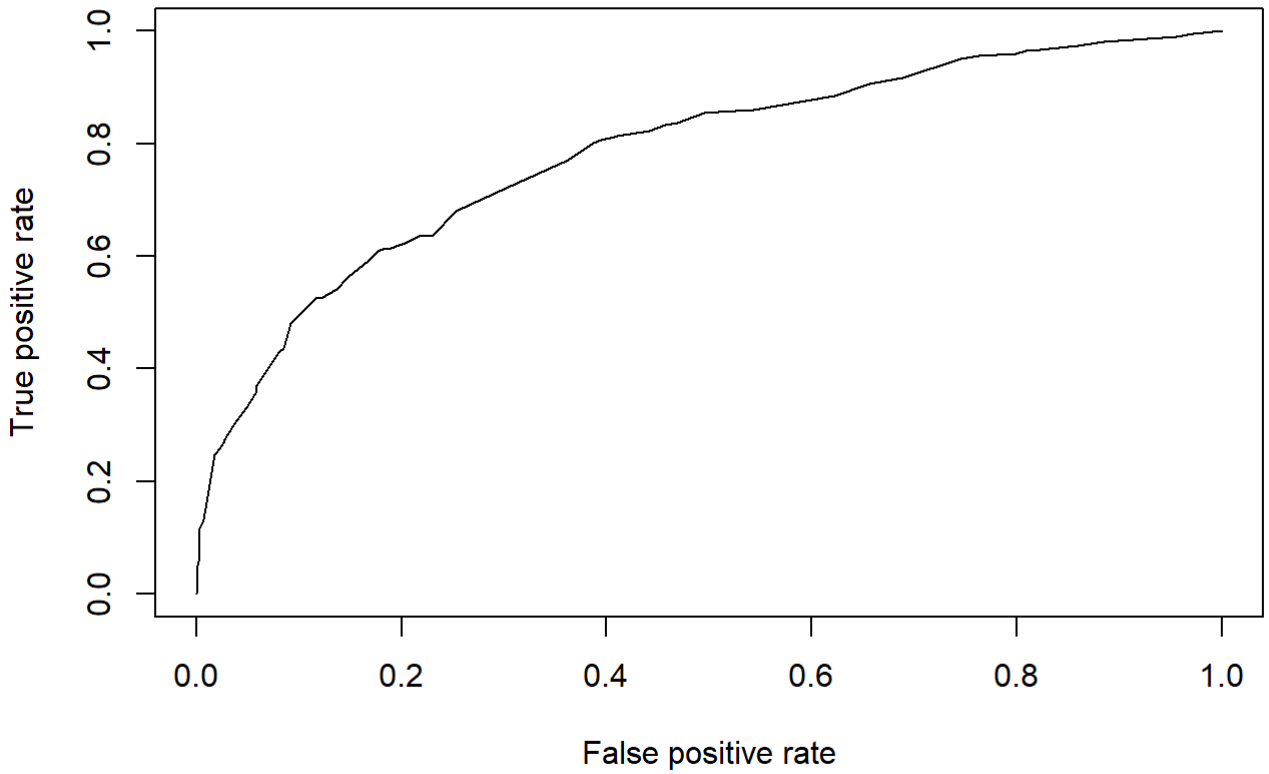
Observamos que para el umbral 0.3, tenemos un modelo con las métricas:

- acierto = 81.14973
- precision = 53.27731
- cobertura = 52.57048
- F1 = 52.92154

Paso 6. Métricas definitivas

Evaluamos la ROC

```
#creamos el objeto prediction
rl2_prediction<-prediction(rl2_predict,test$TARGET)
#visualizamos la ROC
roc(rl2_prediction)
```



En la curva ROC, la línea diagonal que divide el gráfico en dos partes iguales indica que el modelo no tiene ninguna capacidad predictiva. Todo el área que está por encima de esa diagonal hasta la curva, indica la capacidad predictiva del modelo. Lo ideal es que la curva suba rápido, ya que de esa manera hay más espacio ocupado por la curva y, por tanto, mejor es el modelo. Em este caso vemos que la curva no sube rápidamente, lo que da idea de un modelo con un ajuste solo reltivamente adecuado.

Métricas definitivas

```
rl2_metricas<-cbind(rl2_metricas,AUC=round(auc(rl2_prediction),2)*100)
print(t(rl2_metricas))
```

```
##               [,1]
## umbral     0.30000
## acierto   81.14973
## precision 53.27731
## cobertura 52.57048
## F1        52.92154
## AUC       78.00000
```

Obtenemos las métricas definitivas añadiendo la métrica AUC, que indica el porcentaje de predicción del modelo, un 78%, lo que indica que es un modelo relativamente bueno.

4. Modelización con Näive Bayes

Paso 1. Primer modelo

```
nb <- naiveBayes(TARGET ~ Geography + IsActiveMember + RNumOfProducts + CreditScoreR +
AgeR + TenureR + BalanceR + EstimatedSalaryR, data= train)
nb
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##        No        Si
## 0.7953767 0.2046233
##
## Conditional probabilities:
##     Geography
## Y       France   Germany     Spain
##   No 0.5238608 0.2136706 0.2624686
##   Si 0.3870293 0.4030683 0.2099024
##
##     IsActiveMember
## Y           No        Si
##   No 0.4420524 0.5579476
##   Si 0.6457462 0.3542538
##
##     RNumOfProducts
## Y    Más de un producto Un producto
##   No          0.5405454   0.4594546
##   Si          0.3179916   0.6820084
##
##     CreditScoreR
## Y          Alto       Bajo Medio-Alto Medio-Bajo
##   No 0.22604952 0.03372802 0.47111590 0.26910657
##   Si 0.22594142 0.04323570 0.45327755 0.27754533
##
##     AgeR
## Y    De 31 a 40 años De 41 a 50 años De 51 a 65 años Hasta 30 años
##   No      0.49210621      0.19411554      0.04377467    0.22694654
##   Si      0.25871688      0.38772664      0.21966527    0.07391911
##     AgeR
## Y    Más de 65años
##   No    0.04305705
##   Si    0.05997211
##
##     TenureR
## Y    De 2,6 a 5,0 De 5,1 a 7,5 Hasta 2,5 Más de 7,5
##   No    0.2994259    0.2059562 0.2445282  0.2500897
##   Si    0.3089261    0.1799163 0.2545328  0.2566248
##
##     BalanceR
## Y    De 80001 a 120000 Hasta 80000 Más de 120001
##   No         0.2452458   0.4546107     0.3001435
##   Si         0.3291492   0.3026499     0.3682008
##
```

```
##     EstimatedSalaryR
## Y         Alto      Bajo Medio-Alto Medio-Bajo  Muy Alto
##   No 0.2020093 0.1916039  0.2099031  0.1998565 0.1966272
##   Si 0.1987448 0.1931660  0.1994421  0.1973501 0.2112971
```

Paso 2. Predict

Aplicamos el modelo entrenado al conjunto de test (30%), generando un vector con las probabilidades en cada caso de ser 0 o 1.

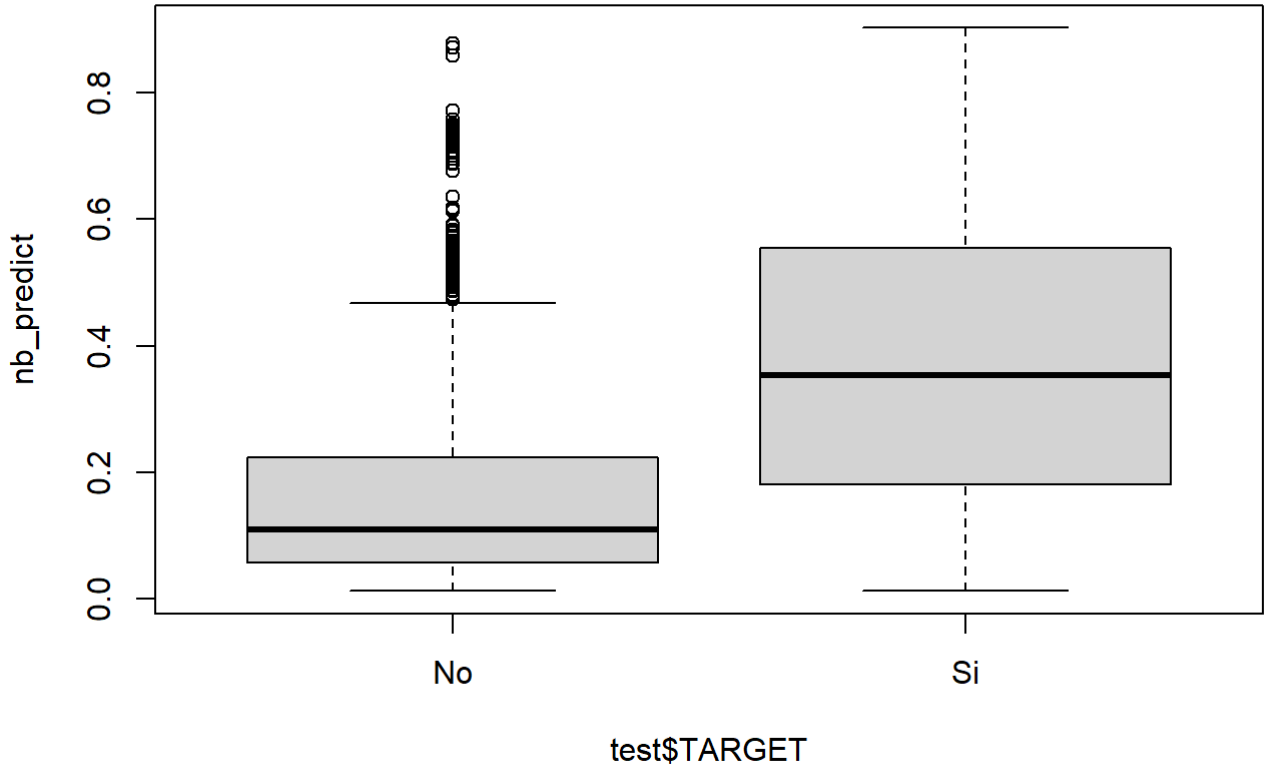
```
nb_predict<-predict(nb, test, type = 'raw')[,2]
#Se emplea type= 'raw' para sacar las probabilidades de que cada sujeto sea SÍ churn O NO churn.

#Sacamos los primeros 6 casos
head(nb_predict)
```

```
## [1] 0.18478477 0.33094617 0.28696277 0.05159321 0.15453458 0.06610520
```

Lanzamos un “head” para ver los 6 primeros. Lo que quiere decir que: el sujeto 1 tendrá una probabilidad de clasificarse como 1 (Sí Churn) del 18,48%. El segundo de 33,09%, etc.

A continuación lanzamos un plot de caja y bigotes, para ver si discrimina bien entre las dos categorías, esto es, si la media de rl_predict de los clientes que sí contratan con la media de los que no contratan es diferente.

```
plot(nb_predict~test$TARGET)
```



Se observa en la gráfica que la media de los que Sí y los que No es muy diferente, incluso discrimina bien entre los cuartiles.

Paso 3. Umbrales

Ahora tenemos que transformar la probabilidad obtenida en una decisión de si el cliente va a abandonar o no.

Con la función umbrales probamos diferentes cortes

```
umb_nb<-umbrales(test$TARGET,nb_predict)
umb_nb
```

```
##    umbral  acierto precision  cobertura         F1
## 1    0.05 35.59492  23.37088 96.3515755 37.6173519
## 2    0.10 53.74332  28.55574 86.2354892 42.9042904
## 3    0.15 64.47193  33.61823 78.2752902 47.0353762
## 4    0.20 69.71925  36.76856 69.8175788 48.1693364
## 5    0.25 76.73797  44.44444 61.6915423 51.6666667
## 6    0.30 78.20856  46.62999 56.2189055 50.9774436
## 7    0.35 81.08289  53.23993 50.4145937 51.7887564
## 8    0.40 81.61765  55.57895 43.7810945 48.9795918
## 9    0.45 82.15241  58.43521 39.6351575 47.2332016
## 10   0.50 82.48663  61.12676 35.9867330 45.3027140
## 11   0.55 83.22193  73.27189 26.3681592 38.7804878
## 12   0.60 82.65374  75.92593 20.3980100 32.1568627
## 13   0.65 82.48663  76.87075 18.7396352 30.1333333
## 14   0.70 82.31952  77.61194 17.2470978 28.2225237
## 15   0.75 80.91578  88.09524  6.1359867 11.4728682
## 16   0.80 80.54813  88.88889  3.9800995  7.6190476
## 17   0.85 80.51471  88.46154  3.8142620  7.3131955
## 18   0.90 79.87968 100.00000  0.1658375  0.3311258
## 19   0.95  0.95000   0.95000  0.9500000  0.9500000
```

Seleccionamos el umbral que maximiza la F1 (cuando empieza a decaer)

```
umbral_final_nb<-umb_nb[which.max(umb_nb$F1),1]
umbral_final_nb
```

```
## [1] 0.35
```

Como puede observarse en la tabla anterior, el indicador F1 crece a medida que los umbrales aumentan (esto es, se maximiza progresivamente la F1), pero llega a un punto que empieza a decrecer: umbral de 0.35

Paso 4. Matriz de confusión

Evaluamos la matriz de confusión y las métricas con el umbral optimizado

```
confusion(test$TARGET,nb_predict,umbral_final_nb)
```

```
##
## real FALSE TRUE
##   No  2122  267
##   Si   299  304
```

```
nb_metricas<-filter(umb_nb,umbral==umbral_final_nb)
nb_metricas
```

```
##   umbral  acierto precision cobertura       F1
## 1   0.35 81.08289  53.23993  50.41459 51.78876
```

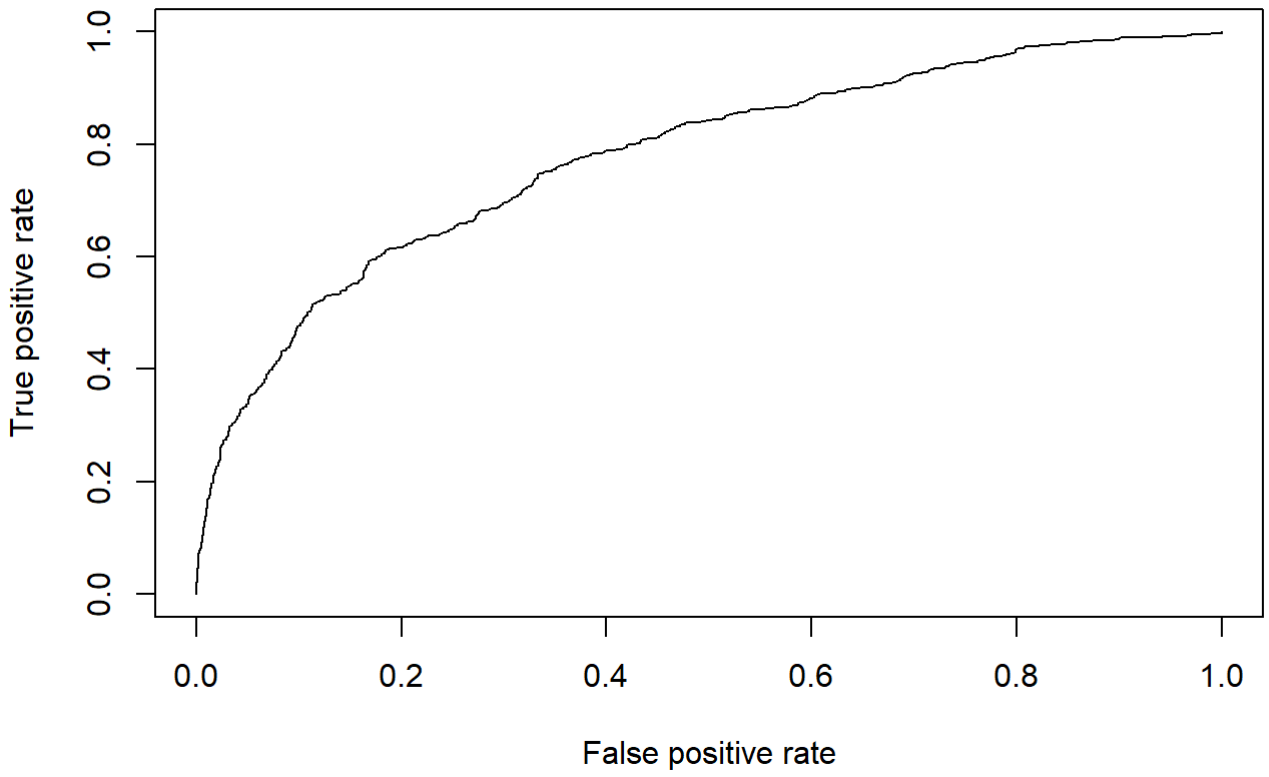
Observamos que para el umbral 0.35, tenemos un modelo con las métricas:

- acierto = 81.08289
- precision = 53.23993
- cobertura = 50.41459
- F1 = 51.78876

Paso 5. Métricas definitivas

Evaluamos la ROC

```
#creamos el objeto prediction
nb_prediction<-prediction(nb_predict,test$TARGET)
#visualizamos la ROC
roc(nb_prediction)
```



En la curva ROC, la línea diagonal que divide el gráfico en dos partes iguales indica que el modelo no tiene ninguna capacidad predictiva. Todo el área que está por encima de esa diagonal hasta la curva, indica la capacidad predictiva del modelo.

Métricas definitivas

```
nb_metricas<-cbind(nb_metricas,AUC=round(auc(nb_prediction),2)*100)
print(t(nb_metricas))
```

```
##               [,1]
## umbral     0.35000
## acierto   81.08289
## precision 53.23993
## cobertura 50.41459
## F1        51.78876
## AUC       78.00000
```

Obtenemos las métricas definitivas añadiendo la métrica AUC, que indica el porcentaje de predicción del modelo, un 78%, lo que indica que es un modelo relativamente aceptable.