Árboles de decisión. Detección de churn. Caso Telecomunicaciones

Adolfo Sánchez Burón

Introducción al análisis de Árboles de decisión

Los métodos basados en árboles de decisión son algoritmos que estratifican el espacio del predictor en un número simple de regiones, observando en cada una de ellas la media o moda. Esto hace que la estructura resultante sea asemejada a un árbol, similar a un diagrama de flujo, donde caben destacar los siguientes conceptos:

- **Nodo raíz**: que representa a toda la población a partir de al cual se comenzará a dividir en grupos homogéneos.
- **División**: proceso mediante el cual un nodo se divide en dos o más nodos en función de una característica determinada.
- **Nodo terminal** o **nodo hoja**: son los nodos sin hijos donde se termina le proceso e división.
- **Poda**: proceso mediante el que vamos especificando condiciones para reducir la profundidad del árbol. Lo normal es que empecemos solicitando un árbol muy profundo, sin restricciones y posteriormente vayamos aplicando el proceso de poda, entre otras cosas, para que sea más interpretable.
- **Rama** o **subárbol**: cada subsección del árbol de decisión.
- **Nodo principal**: nodo a partir del cual se divide en otros subnodos.

- **Nodos hijo**: subnodos que nacen de un nodo principal.

Frente a otros modelos caben destacar dos ventajas:

- Los AD tiene considerables ventajas frente a los modelos lineales (como regresión lineal o regresión logística).
- Es realmente sencilla su interpretación frente a los modelos de ensemble, como randon forest, baaging o boosting.

Hay dos tipos de algoritmos para árboles de decisión: los de regresión y los de clasificación.

- Los **AD de regresión** predicen una TARGET cuantitativa.
- Mientras que los **AD de clasificación** predicen la probabilidad de que un caso pertenecerá a una categoría determinada (esto es, la TARGET es categórica).

Características del caso

El caso empleado en este análisis es el 'Telco Customer Churn', que puede descargarse el dataset original de Kaggle (https://www.kaggle.com/blastchar/telco-customer-churn). 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 prepardo 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) la empresa. La explicación de esta conducta estará basada en toda una serie de variables predictoras que se pueden clasificar en cuatro grupos:

- Churn: la variable TARGET, con puntuaciones de 0 (no abandonó la empresa) y 1 (sí abandonó la empresa)
- Servicios contratados: phone, multiple lines, internet, online security, online backup, device protection, tech support, and streaming TV and movies.
- Información sobre cuentas dle cliente: how long they've been a customer, contract, payment method, paperless billing, monthly charges, and total charges.
- Variables demográficas: gender, age range, and if they have partners and dependents

Tras estudiar el peso predictivo de estas variables sobre la TARGET, finalmente se redujo el número de predictores a 6: Internet Service, Contract, Payment Method, tenure, Monthly Charges y Total Charges.

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. 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 con árboles de decisión

Por motivos didácticos, se dividirá en seis pasos:

- Paso 1. Primer modelo
- Paso 2. Segundo modelo
- Paso 3. Interpretación del árbol
- Paso 4. Predict
- Paso 5. Umbrales
- Paso 6. Matriz de confusión
- Paso 7. Métricas definitivas

---

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(ggplot2) #para gráficos
library(ROCR) #para evaluar modelos
library(DataExplorer) #para realizar el análisis descriptivo con gráficos
library(rpart) #para crear arboles de decisión
library(rpart.plot) #para el gráfico del árbol
```

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).

```
# Importamos los datos y los incluimos en un data frame llamado df1
df1 <- fread("TelcoChurn.csv")
```

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

---

2. Análisis descriptivo

2.1. Análisis inicial

```
head(df1) #con esta función podemos ver la estructura de los primeros 6 casos
```

```
##    InternetService       Contract             PaymentMethod tenure_DISC
## 1:             DSL Month-to-month          Electronic check     Grupo 1
## 2:             DSL       One year              Mailed check     Grupo 2
## 3:             DSL Month-to-month              Mailed check     Grupo 1
## 4:             DSL       One year Bank transfer (automatic)     Grupo 3
## 5:     Fiber optic Month-to-month          Electronic check     Grupo 1
## 6:     Fiber optic Month-to-month          Electronic check     Grupo 1
##    MonthlyCharges_DISC TotalCharges_DISC TARGET
## 1:             Grupo 1           Grupo 1     No
## 2:             Grupo 2           Grupo 3     No
## 3:             Grupo 2           Grupo 1     Si
## 4:             Grupo 2           Grupo 3     No
## 5:             Grupo 3           Grupo 1     Si
## 6:             Grupo 4           Grupo 2     Si
```

```
str(df1) #mostrar la estructura del dataset y los tipos de variables
```

```
## Classes 'data.table' and 'data.frame':   7032 obs. of  7 variables:
##  $ InternetService    : chr  "DSL" "DSL" "DSL" "DSL" ...
##  $ Contract           : chr  "Month-to-month" "One year" "Month-to-month" "One yea
r" ...
##  $ PaymentMethod      : chr  "Electronic check" "Mailed check" "Mailed check" "Bank
transfer (automatic)" ...
##  $ tenure_DISC        : chr  "Grupo 1" "Grupo 2" "Grupo 1" "Grupo 3" ...
##  $ MonthlyCharges_DISC: chr  "Grupo 1" "Grupo 2" "Grupo 2" "Grupo 2" ...
##  $ TotalCharges_DISC  : chr  "Grupo 1" "Grupo 3" "Grupo 1" "Grupo 3" ...
##  $ TARGET             : chr  "No" "No" "Si" "No" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

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

2.2. Tipología de datos

```
df1 <- mutate_if(df1, is.character, as.factor) #identifica todas las variables character y transformarlas en factores

str(df1) #estructura de la base de datos después de la transformación
```

```
## Classes 'data.table' and 'data.frame':   7032 obs. of  7 variables:
##  $ InternetService    : Factor w/ 3 levels "DSL","Fiber optic",..: 1 1 1 1 2 2 2 1 2 1 ...
##  $ Contract           : Factor w/ 3 levels "Month-to-month",..: 1 2 1 2 1 1 1 1 1 2 ...
##  $ PaymentMethod      : Factor w/ 4 levels "Bank transfer (automatic)",..: 3 4 4 1 3 3 2 4 3 1 ...
##  $ tenure_DISC        : Factor w/ 4 levels "Grupo 1","Grupo 2",..: 1 2 1 3 1 1 2 1 2 4 ...
##  $ MonthlyCharges_DISC: Factor w/ 4 levels "Grupo 1","Grupo 2",..: 1 2 2 2 3 4 3 1 4 2 ...
##  $ TotalCharges_DISC  : Factor w/ 4 levels "Grupo 1","Grupo 2",..: 1 3 1 3 1 2 3 1 3 3 ...
##  $ TARGET             : Factor w/ 2 levels "No","Si": 1 1 2 1 2 2 1 1 2 1 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

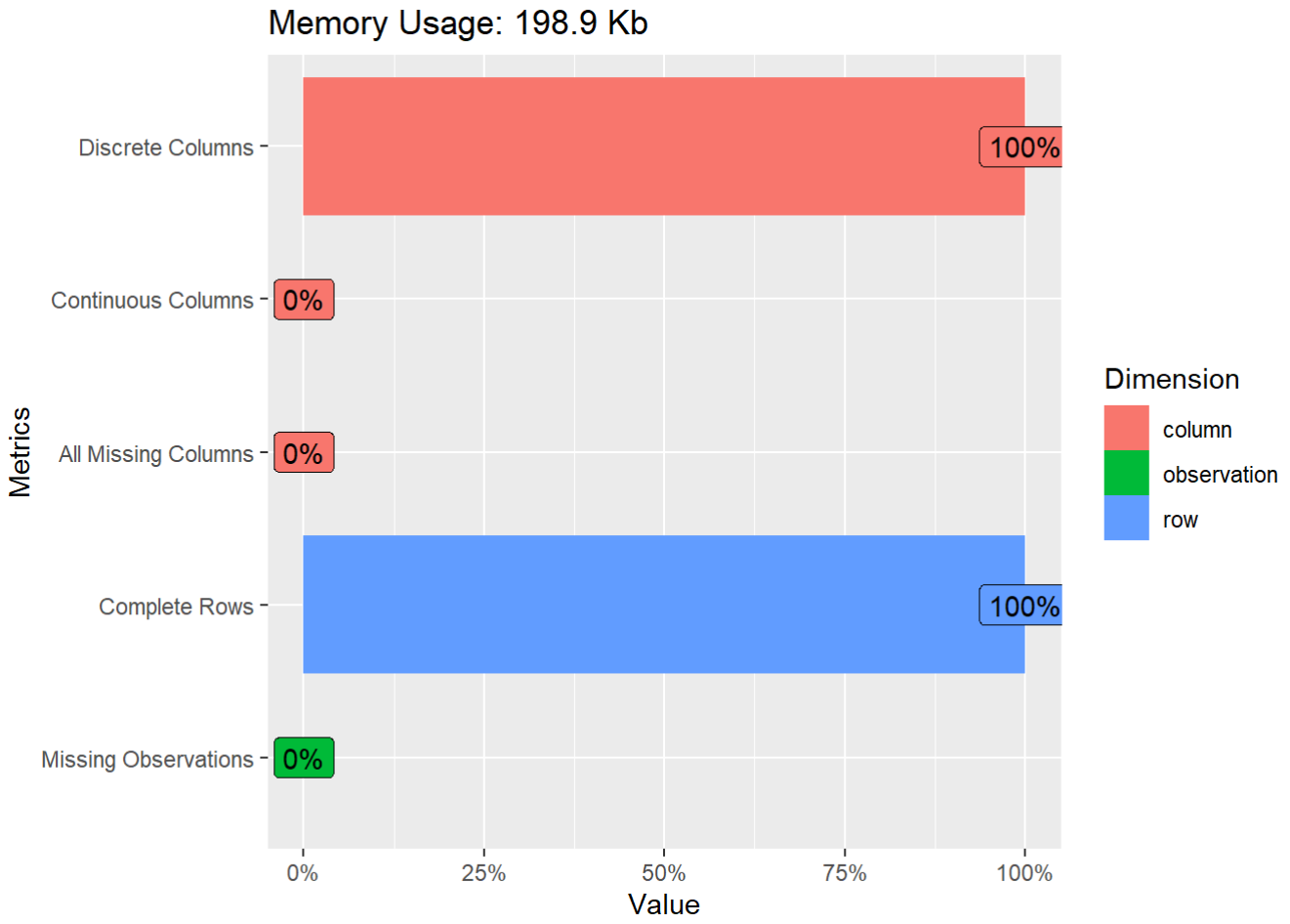
Ahora se puede observar que todas las variables son de tipo “Factor”

```
lapply(df1,summary) #mostrar la distribución de frecuencias en cada categoría de todas las variables
```

```
## $InternetService
##         DSL Fiber optic          No
##        2416        3096        1520
##
## $Contract
## Month-to-month       One year       Two year
##           3875           1472           1685
##
## $PaymentMethod
## Bank transfer (automatic)   Credit card (automatic)          Electronic check
##                      1542                      1521                      2365
##              Mailed check
##                      1604
##
## $tenure_DISC
## Grupo 1 Grupo 2 Grupo 3 Grupo 4
##    2723    1308    1182    1819
##
## $MonthlyCharges_DISC
## Grupo 1 Grupo 2 Grupo 3 Grupo 4
##    1758    1761    1755    1758
##
## $TotalCharges_DISC
## Grupo 1 Grupo 2 Grupo 3 Grupo 4
##    1758    1758    1758    1758
##
## $TARGET
##   No   Si
## 5163 1869
```

2.3. Análisis descriptivo (gráficos)

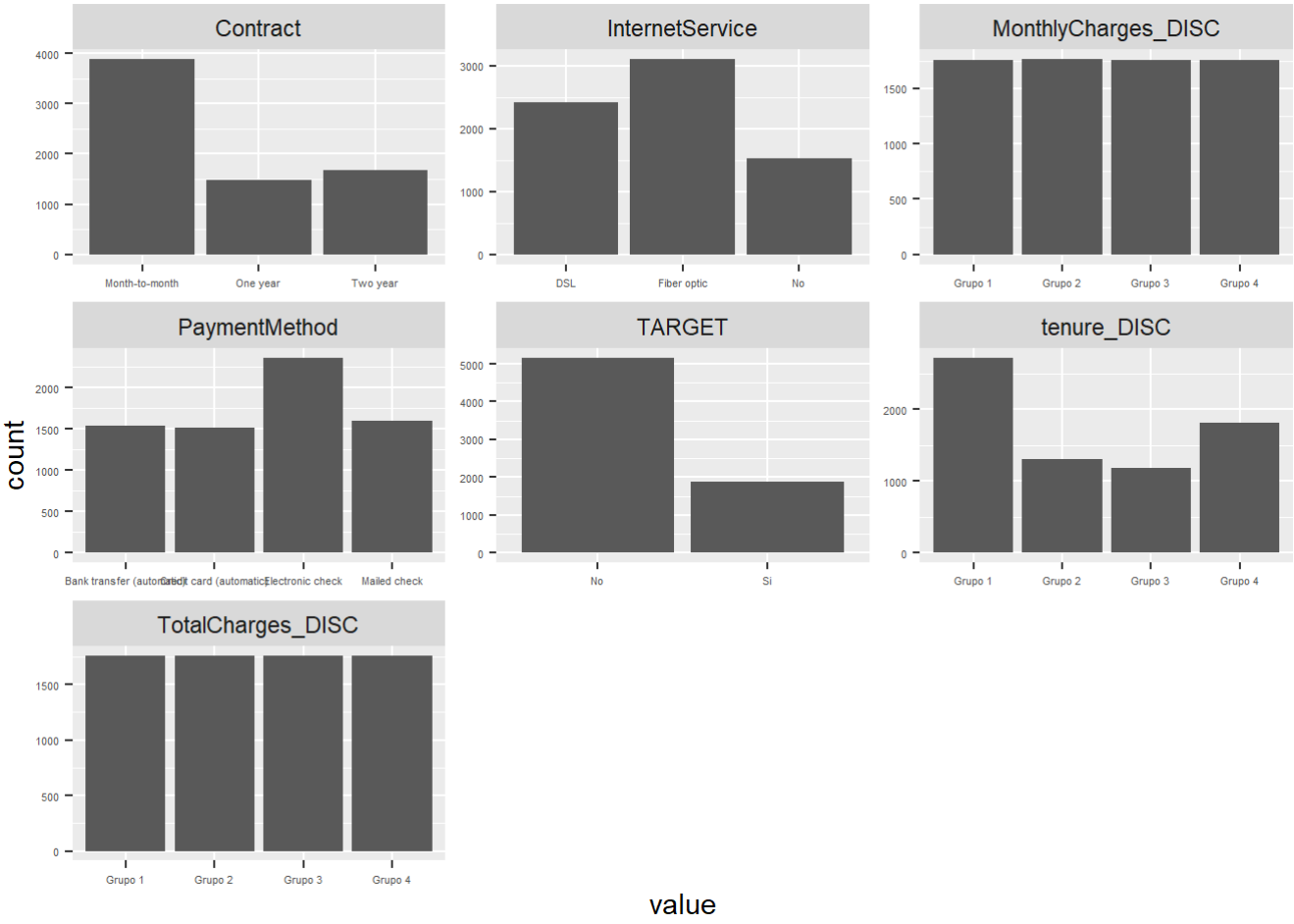
```
plot_intro(df1) #gráfico para observar la distribución de variables y los casos missin
g por columnas, observaciones y filas
```



Como se ha trabajado previamente, no existen casos missing, por lo que podemos seguir el análisis descriptivo

```
#Análisis visual de frecuencias de cada categoría por variable
df1 %>%
  gather() %>%
  ggplot(aes(value)) +
  geom_bar()+
  facet_wrap(~ key, scales = "free")+
  theme(axis.text=element_text(size=4))
```

```
## Warning: attributes are not identical across measure variables;
## they will be dropped
```



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:

- Internet Service: tiene tres niveles: DSL, Fiber optic, No.
- Contract (tipo de contrato): tiene tres niveles, Month-to-month, One year, Two years.
- Payment Method: con tres niveles, Bank transfer, Credit card, Electronic check.
- Tenure: variable originalmente cuantitativa, que se discretizó en cuatro categorías por cuartiles.
- Monthly Charges: se discretizó en cuatro categorías. Grupo 1 (<= 35.59), Grupo 2 (> 35.59 & <= 70.35), Grupo 3 (> 70.35 & <= 89.86), Grupo 4 (> 89.86).
- Total Charges: se discretizó en cuatro categorías. Grupo 1 (<= 401.4), Grupo 2 (> 401.4 & <= 1397.5), Grupo 3 (> 1397.5 & <= 3794.7), Grupo 4 (> 3794.7).
- TARGET: con dos niveles Sí han abandonado (churn), No han abandonado (churn).

Parece que la distribución de frecuencias en todas las variables es aceptable, incluso en la variable TARGET, que suele dar más problemas.

---

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
df1$random<-sample(0:1,size = nrow(df1),replace = T,prob = c(0.3,0.7))

train<-filter(df1,random==1)
test<-filter(df1,random==0)

#Eliminamos ya la random
df1$random <- NULL
```

---

4. Modelización con Árboles de decisión

Paso 1. Primer modelo

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

```
ar<-rpart(TARGET ~ InternetService + Contract + PaymentMethod + tenure_DISC + MonthlyC
harges_DISC + TotalCharges_DISC, train,           method = 'class',
          parms = list(
            split = "information"),
          control = rpart.control(cp = 0.00001))

#Lanzamos la runción "rpart2" para hallar el AD, que incluye>:
#El modelo a entrenar: TARGET ~ InternetService + Contract + PaymentMethod + tenure_DI
SC + MonthlyCharges_DISC + TotalCharges_DISC
#El df: "train"
#method = 'class'. Método de clasificación
#split: criterio de corte. Se tienen dos opciones: 'information' o 'gini'
#cp: criterio de complejidad. Se comienza emplenado un cp muy pequeño (esto es, dejamo
s que el árbol sea muy profundo), y posteriormente, se irán añadiendo condiciones más
 restrictivas (siendo el árbol menos profundo y más interpretable).

# Imprimimos los resultados
printcp(ar)
```

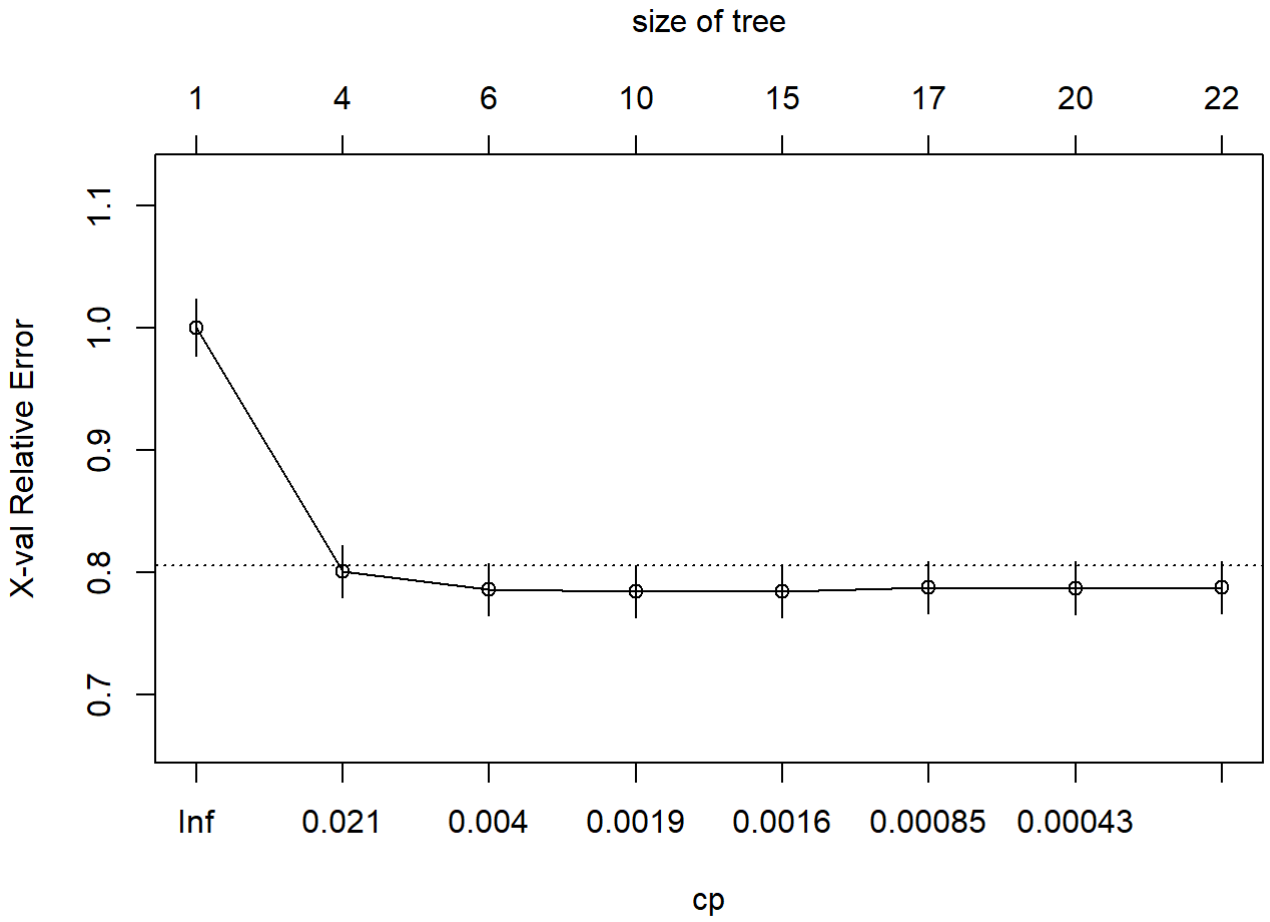
```
##
## Classification tree:
## rpart(formula = TARGET ~ InternetService + Contract + PaymentMethod +
##     tenure_DISC + MonthlyCharges_DISC + TotalCharges_DISC, data = train,
##     method = "class", parms = list(split = "information"), control = rpart.control
(cp = 0.00001))
##
## Variables actually used in tree construction:
## [1] Contract            InternetService     MonthlyCharges_DISC
## [4] PaymentMethod       tenure_DISC         TotalCharges_DISC
##
## Root node error: 1353/4934 = 0.27422
##
## n= 4934
##
##           CP nsplit rel error  xerror     xstd
## 1 0.05691057      0   1.00000 1.00000 0.023161
## 2 0.00739098      3   0.79084 0.80044 0.021488
## 3 0.00221729      5   0.77605 0.78566 0.021344
## 4 0.00162602      9   0.76718 0.78418 0.021330
## 5 0.00147820     14   0.75905 0.78418 0.021330
## 6 0.00049273     16   0.75610 0.78788 0.021366
## 7 0.00036955     19   0.75462 0.78714 0.021359
## 8 0.00001000     21   0.75388 0.78788 0.021366
```

Interpretación de la tabla:

a. Observamos la columna xerror (error de validación cruzada), la cual se va reduciendo hasta que empieza a crecer.

b. Tomamos el datos de CP (criterior de complejidad) de ese nivel para incluirlo posteriormente al modelo.

```
# Sacamos la representación gráfica de la tabla anterior (aunque en ocasiones no se ob
serva con nitidez)
plotcp(ar)
```



Se observa que el error de validación cruzada (xerror = 0.78418) minimiza aproximadamente en un cp = 0.0016 de complejidad.Por tanto, los dos siguientes pasos son:

c. Generamos un nuevo árbol con ese parámetro, esto es más restrictivo.

d. Además, le añadimos otro parámetro de restricción: el número de niveles, que no tenga mas de 7 niveles de profundidad (maxdepth = 7), aunque no sabemos dónde va a parar antes, por el cp o por el número de niveles.

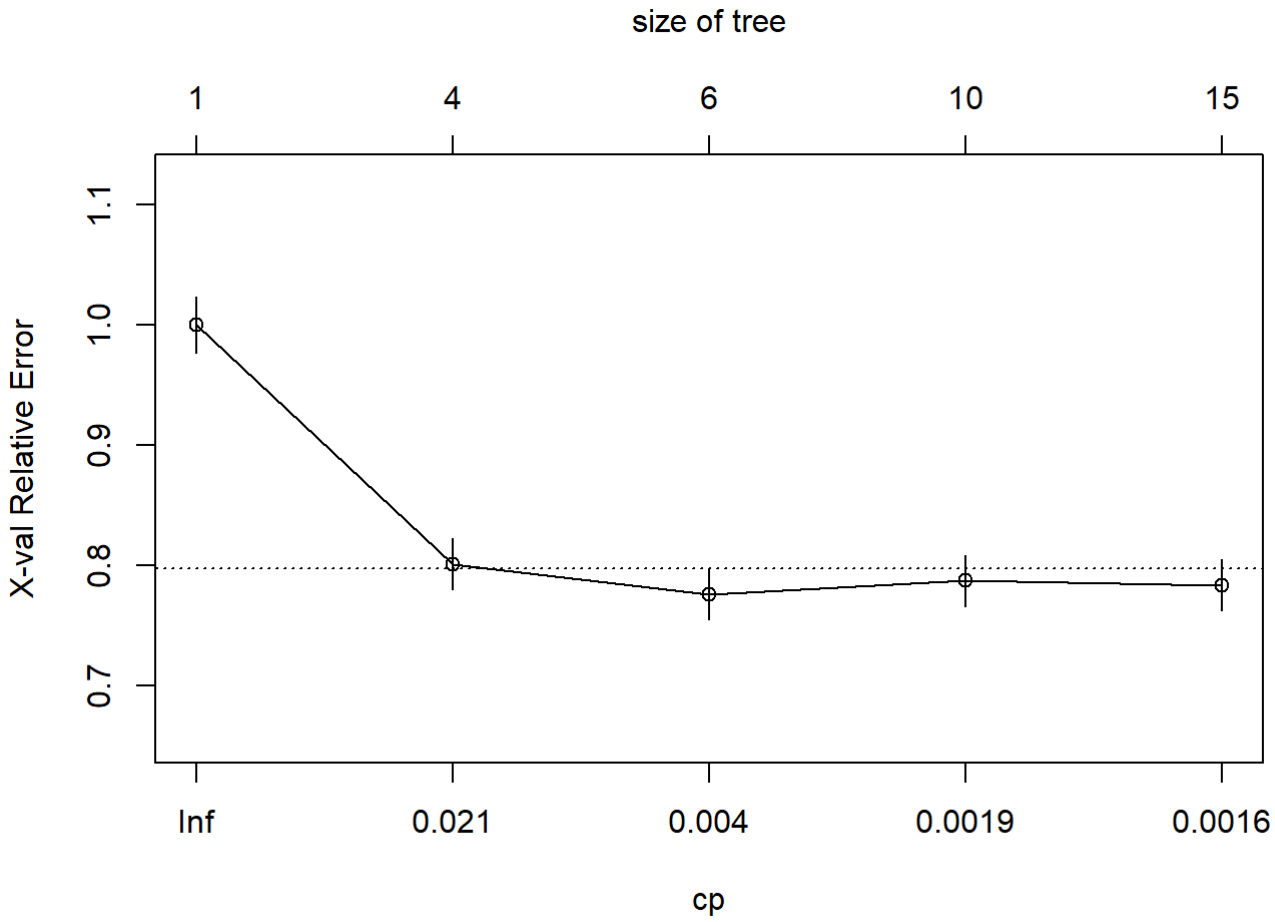
Paso 2. Segundo modelo

```
ar2<-rpart(TARGET ~ InternetService + Contract + PaymentMethod + tenure_DISC + Monthly
Charges_DISC + TotalCharges_DISC, train, method = 'class', parms = list(
  split = "information"),
  control = rpart.control(cp = 0.0016,maxdepth = 7))

printcp(ar2)
```

```
##
## Classification tree:
## rpart(formula = TARGET ~ InternetService + Contract + PaymentMethod +
##     tenure_DISC + MonthlyCharges_DISC + TotalCharges_DISC, data = train,
##     method = "class", parms = list(split = "information"), control = rpart.control
(cp = 0.0016,
##         maxdepth = 7))
##
## Variables actually used in tree construction:
## [1] Contract            InternetService     MonthlyCharges_DISC
## [4] PaymentMethod       tenure_DISC         TotalCharges_DISC
##
## Root node error: 1353/4934 = 0.27422
##
## n= 4934
##
##          CP nsplit rel error  xerror     xstd
## 1 0.0569106      0   1.00000 1.00000 0.023161
## 2 0.0073910      3   0.79084 0.80118 0.021496
## 3 0.0022173      5   0.77605 0.77605 0.021249
## 4 0.0016260      9   0.76718 0.78714 0.021359
## 5 0.0016000     14   0.75905 0.78344 0.021322
```

Se observa que xerror no asciendo y que se corta en el nivel de profundidad de 5.

```
#Observamos gráficamente el resultado anterior
plotcp(ar2)
```



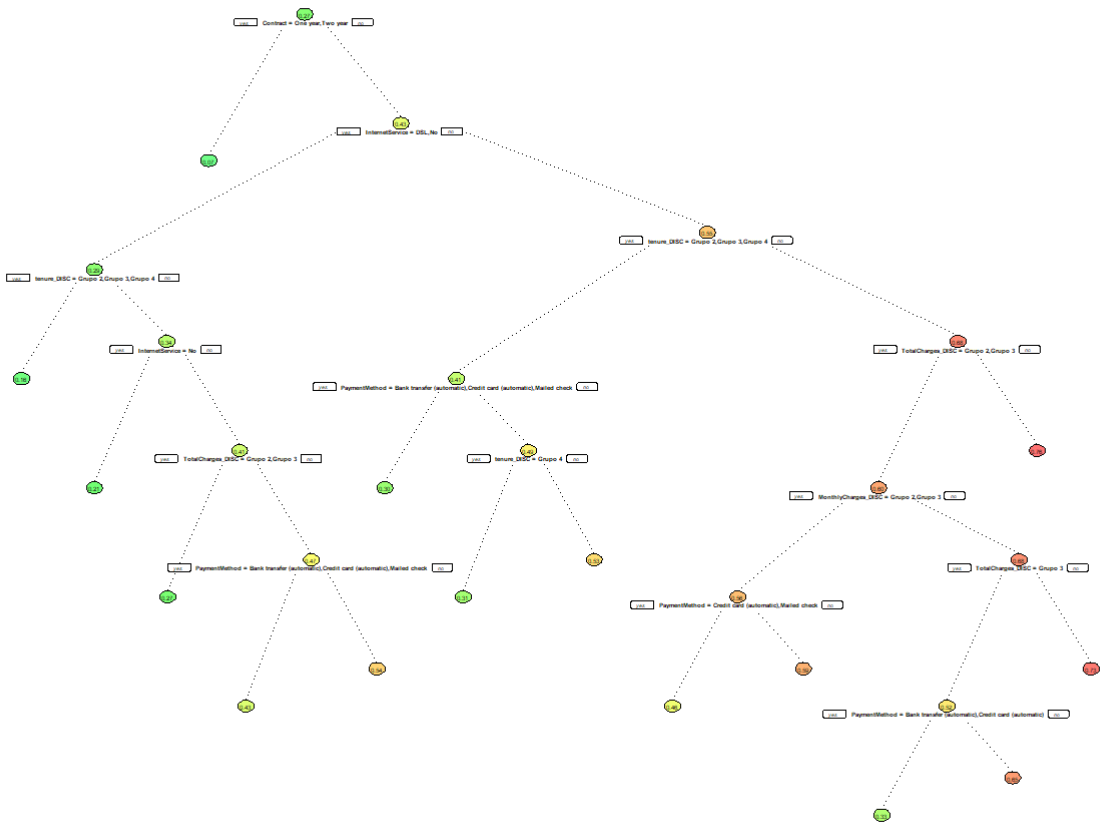
Parece que el árbol es bastante estable, por lo que pasamos a interpretarlo.

Paso 3. Interpretación del árbol

Seguimos tres pasos:

A. Creación del gráfico del árbol

```
rpart.plot(ar2,type=2,extra = 7, under = TRUE,under.cex = 0.7,fallen.leaves=F,gap = 0,
cex=0.2,yesno = 2,box.palette = "GnYlRd",branch.lty = 3)
```



En muchas ocasiones el árbol no es fácilmente visible si no se amplía considerablemente, lo cual podemos hacerlo exportándolo a pdf o en un power point. Si observamos la parte superior del árbol podremos ver estos nodos y divisiones:

- La variable con mayor capacidad discriminativa es Contrato de uno o dos años (el 27% de clientes está en este nodo).
- Los que SÍ tienen este contrato tienen una probabilidad de churn del 7%. Este es un nodo terminal ya que no tiene más divisiones. Desde un punto de vista de negocio viene a decir que son clientes bastante poco proclives al churn.
- Los que NO tienen este contrato tienen la probailidad de churn del 43%.

B. Reglas del árbol

Sacamos las reglas de división del árbol, necesarias para hacer una implantación de negocio posterior

```
#Se emplea la función "rpart.rules"
#style: sirve para que la salida sea mas legible
#cover = T: añade el % de casos que aplica la regla
rpart.rules(ar2,style = 'tall',cover = T)
```

```
## TARGET is 0.07 with cover 44% when
##     Contract is One year or Two year
##
## TARGET is 0.16 with cover 7% when
##     Contract is Month-to-month
##     tenure_DISC is Grupo 2 or Grupo 3 or Grupo 4
##     InternetService is DSL or No
##
## TARGET is 0.21 with cover 6% when
##     Contract is Month-to-month
##     tenure_DISC is Grupo 1
##     InternetService is No
##
## TARGET is 0.27 with cover 4% when
##     Contract is Month-to-month
##     tenure_DISC is Grupo 1
##     InternetService is DSL
##     TotalCharges_DISC is Grupo 2 or Grupo 3
##
## TARGET is 0.30 with cover 6% when
##     Contract is Month-to-month
##     tenure_DISC is Grupo 2 or Grupo 3 or Grupo 4
##     InternetService is Fiber optic
##     PaymentMethod is Bank transfer (automatic) or Credit card (automatic) or Mailed
check
##
## TARGET is 0.31 with cover 1% when
##     Contract is Month-to-month
##     tenure_DISC is Grupo 4
##     InternetService is Fiber optic
##     PaymentMethod is Electronic check
##
## TARGET is 0.33 with cover 0% when
##     Contract is Month-to-month
##     tenure_DISC is Grupo 1
##     InternetService is Fiber optic
##     PaymentMethod is Bank transfer (automatic) or Credit card (automatic)
##     TotalCharges_DISC is Grupo 3
##     MonthlyCharges_DISC is Grupo 4
##
## TARGET is 0.43 with cover 5% when
##     Contract is Month-to-month
##     tenure_DISC is Grupo 1
##     InternetService is DSL
##     PaymentMethod is Bank transfer (automatic) or Credit card (automatic) or Mailed
check
##     TotalCharges_DISC is Grupo 1
##
## TARGET is 0.46 with cover 1% when
##     Contract is Month-to-month
##     tenure_DISC is Grupo 1
```

```
##     InternetService is Fiber optic
##     PaymentMethod is Credit card (automatic) or Mailed check
##     TotalCharges_DISC is Grupo 2 or Grupo 3
##     MonthlyCharges_DISC is Grupo 2 or Grupo 3
##
## TARGET is 0.53 with cover 7% when
##     Contract is Month-to-month
##     tenure_DISC is Grupo 2 or Grupo 3
##     InternetService is Fiber optic
##     PaymentMethod is Electronic check
##
## TARGET is 0.54 with cover 3% when
##     Contract is Month-to-month
##     tenure_DISC is Grupo 1
##     InternetService is DSL
##     PaymentMethod is Electronic check
##     TotalCharges_DISC is Grupo 1
##
## TARGET is 0.59 with cover 4% when
##     Contract is Month-to-month
##     tenure_DISC is Grupo 1
##     InternetService is Fiber optic
##     PaymentMethod is Bank transfer (automatic) or Electronic check
##     TotalCharges_DISC is Grupo 2 or Grupo 3
##     MonthlyCharges_DISC is Grupo 2 or Grupo 3
##
## TARGET is 0.65 with cover 1% when
##     Contract is Month-to-month
##     tenure_DISC is Grupo 1
##     InternetService is Fiber optic
##     PaymentMethod is Electronic check or Mailed check
##     TotalCharges_DISC is Grupo 3
##     MonthlyCharges_DISC is Grupo 4
##
## TARGET is 0.73 with cover 2% when
##     Contract is Month-to-month
##     tenure_DISC is Grupo 1
##     InternetService is Fiber optic
##     TotalCharges_DISC is Grupo 2
##     MonthlyCharges_DISC is Grupo 4
##
## TARGET is 0.76 with cover 8% when
##     Contract is Month-to-month
##     tenure_DISC is Grupo 1
##     InternetService is Fiber optic
##     TotalCharges_DISC is Grupo 1
```

Interpretación:

Vemos el primero:

```
TARGET is 0.07 with cover 44% when
        Contract is One year or Two year
```

- Grupo con scoring (probabilidad de hacer churn) es del 7%.
- Compuesto por el 44% de los clientes
- Regla: tienen contrato de 1 o 2 años
- Interpretación de negocio: en principio, con estos clientes no es necesaria ninguna acción extra para evitar churn. Son clientes bastante fieles.

Vemos el último:

```
TARGET is 0.76 with cover 8% when
    Contract is Month-to-month
    tenure_DISC is Grupo 1
    InternetService is Fiber optic
    TotalCharges_DISC is Grupo 1
```

- Grupo con scoring (probabilidad de hacer churn) es del 76%.
- Compuesto por el 8% de los clientes
- Regla: tienen
    a. contrato de mes a mes
    b. Son del grupo 1 de tenure
    c. Tienen de servicio de internet fibra óptica
    d. En TotalCharges son del gruo 1
- Interpretación de negocio: en principio, estos clientes son muy poco fieles y con alta tendencia al churn.

C. Introducir datos en un df

Llevamos el nodo final de cada cliente a un data.frame para poder hacer una explotacion posterior (por ejemplo para saber las características de cada nodo, como edad, etc.)

```
#Se usa el predict específico de rpart y con el parámetro nn
ar2_numnodos<-rpart.predict(ar2,test,nn = T)

head(ar2_numnodos)
```

```
##          No         Si  nn
## 1 0.4638554 0.53614458 111
## 2 0.9268966 0.07310345   2
## 3 0.5684647 0.43153527 110
## 4 0.9268966 0.07310345   2
## 5 0.4695122 0.53048780  59
## 6 0.9268966 0.07310345   2
```

INTERPRETACIÓN

El cliente 1 tiene una probabilidad de churn del 53,61% y cae en el nodo 111.

El cliente 2 tiene una probabilidad de churn del 7,31% y cae en el nodo 2.

Paso 4. Predict

Vamos a calcular los scorings y evaluar el modelo

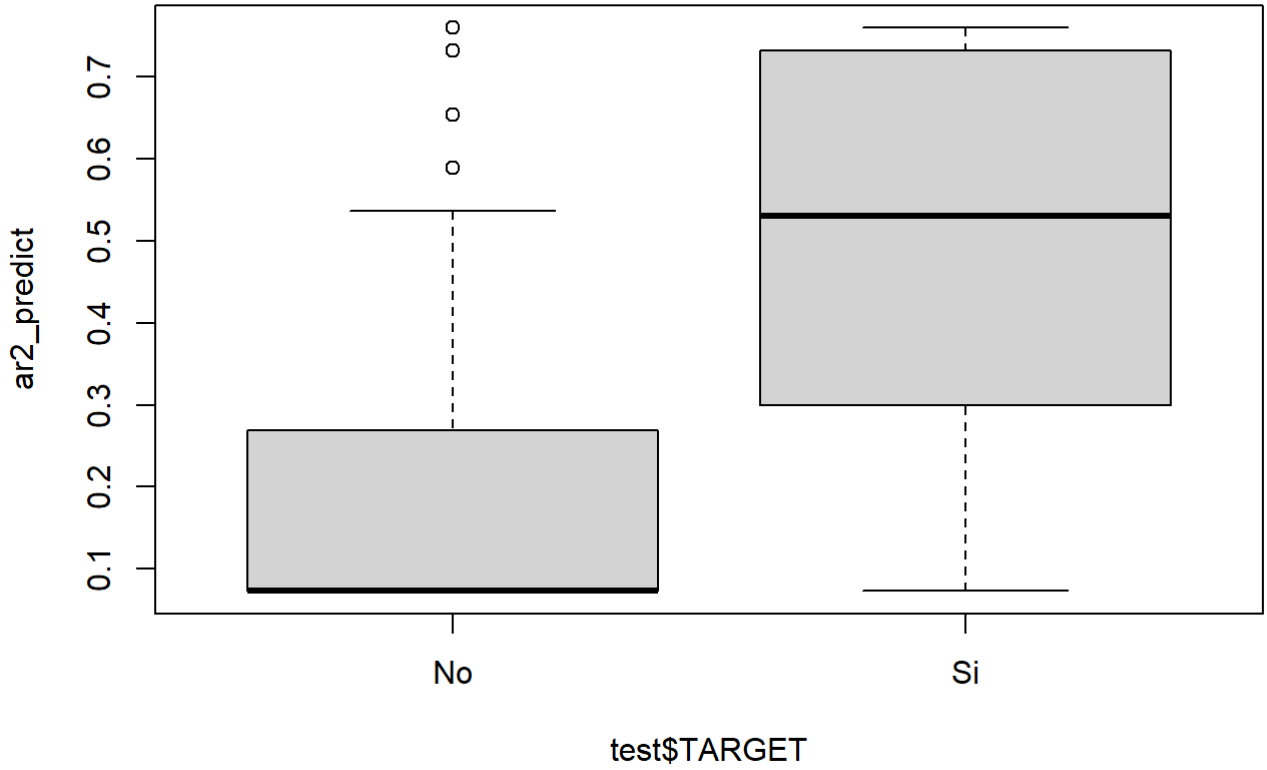
```
ar2_predict<-predict(ar2,test,type = 'prob')[,2]
#Samos el predict para el modelo "ar2":
#Con el data frame "test"
#Se utiliza "type=prob" (que reporta la probabilidad para cada caso)
#Se le incluye [,2], es decir, la segunda columna, ya que interesa la probabilidad de
 que sea 1 (que haga churn)

#Sacamos los 6 primeros casos
head(ar2_predict)
```

```
##         1         2         3         4         5         6
## 0.53614458 0.07310345 0.43153527 0.07310345 0.53048780 0.07310345
```

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 53,61%. El segundo de 7,31%, etc.

Vemos la capacidad de discriminación entre los dos niveles de la TARGET

```
plot(ar2_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 5. Umbrales

Ahora vamos a transformar la probabilidad obtenida en otra dicotómica, es decir, cuando el cliente SÍ hara churn o NO.

Con la función umbrales probamos diferentes cortes

```
umb_ar2<-umbrales(test$TARGET,ar2_predict)
umb_ar2
```

```
##    umbral  acierto precision cobertura       F1
## 1    0.05  0.05000   0.05000   0.05000  0.05000
## 2    0.10 66.15825  41.30824  89.34109 56.49510
## 3    0.15 66.15825  41.30824  89.34109 56.49510
## 4    0.20 71.35367  45.64103  86.24031 59.69148
## 5    0.25 74.49952  48.88628  80.81395 60.92038
## 6    0.30 78.83699  55.64263  68.79845 61.52513
## 7    0.35 79.55195  57.19008  67.05426 61.73060
## 8    0.40 79.55195  57.19008  67.05426 61.73060
## 9    0.45 80.60057  60.42065  61.24031 60.82772
## 10   0.50 80.60057  60.96579  58.72093 59.82231
## 11   0.55 80.45758  66.66667  41.08527 50.83933
## 12   0.60 79.93327  72.09302  30.03876 42.40766
## 13   0.65 79.93327  72.09302  30.03876 42.40766
## 14   0.70 79.83794  73.36683  28.29457 40.83916
## 15   0.75 78.59867  71.06918  21.89922 33.48148
## 16   0.80  0.80000   0.80000   0.80000  0.80000
## 17   0.85  0.85000   0.85000   0.85000  0.85000
## 18   0.90  0.90000   0.90000   0.90000  0.90000
## 19   0.95  0.95000   0.95000   0.95000  0.95000
```

Seleccionamos automáticamente el mejor umbral

```
umbral_final_ar2<-umb_ar2[which.max(umb_ar2$F1),1]
umbral_final_ar2
```

```
## [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 6. Matriz de confusión

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

```
confusion(test$TARGET,ar2_predict,umbral_final_ar2)
```

```
##
## real FALSE TRUE
##   No  1323  259
##   Si   170  346
```

```
ar2_metricas<-filter(umb_ar2,umbral==umbral_final_ar2)
ar2_metricas
```

```
##   umbral  acierto precision cobertura      F1
## 1   0.35 79.55195  57.19008  67.05426 61.7306
```

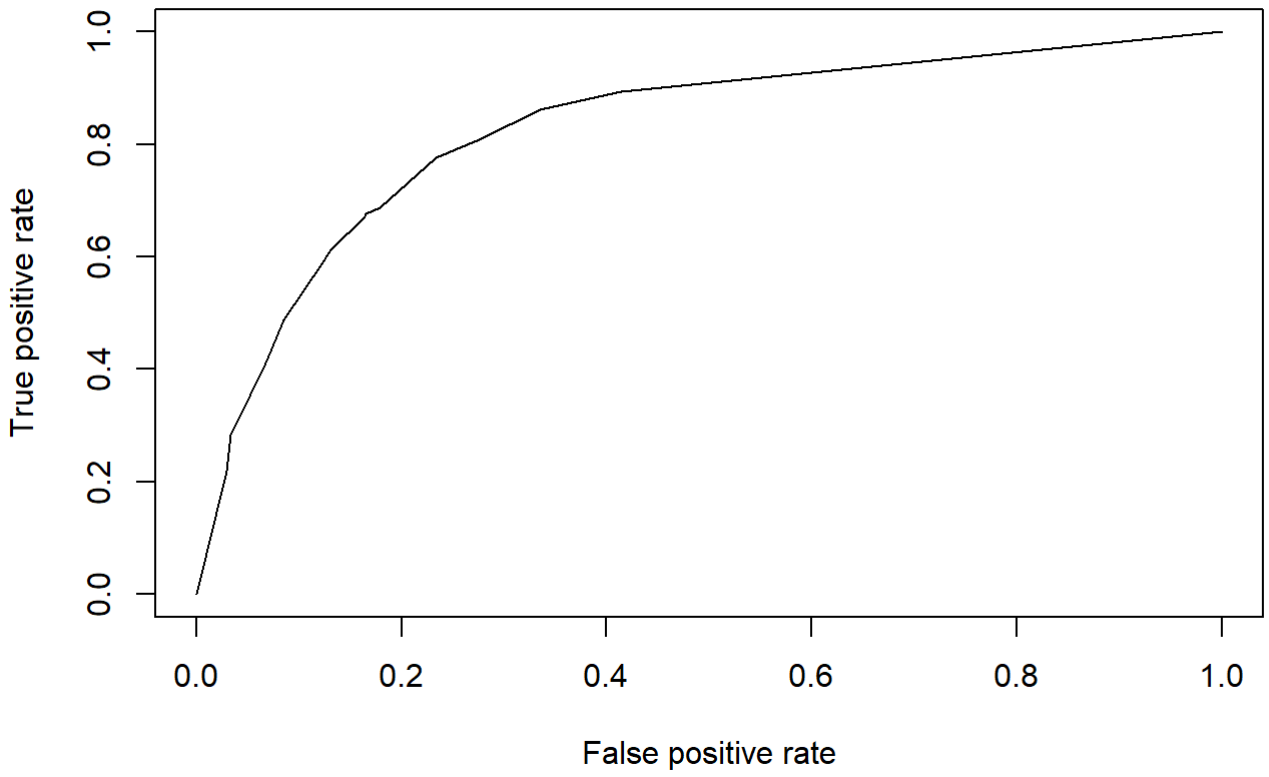
Observamos que para el umbral 0.35, tenemos un modelo con las métricas:

- acierto = 79.55195
- precision = 57.19008
- cobertura = 67.05426
- F1 = 61.7306

Paso 7. Métricas definitivas

Evaluamos la ROC

```
#creamos el objeto prediction
ar2_prediction<-prediction(ar2_predict,test$TARGET)
#visualizamos la ROC (lo ideal es que la curva suba rápido)
roc(ar2_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

```
ar2_metricas<-cbind(ar2_metricas,AUC=round(auc(ar2_prediction),2)*100)
print(t(ar2_metricas))
```

```
##               [,1]
## umbral     0.35000
## acierto   79.55195
## precision 57.19008
## cobertura 67.05426
## F1        61.73060
## AUC       83.00000
```

Obtenemos las métricas definitivas añadiendo la métrica AUC, que indica el porcentaje de predicción del modelo, un 83%, lo que indica que es un buen modelo.