Tarea 3 - Minería de datos predictiva

José Manuel Alvarez García

Octubre 29, 2016

Para instalar el paquete knit se debe ejecutar lo siguiente:

```
knitr::opts_chunk$set(echo = TRUE)
install = function(pkg)
{
  # Si ya está instalado, no lo instala.
  if (!require(pkg, character.only = TRUE)) {
    install.packages(pkg)
    if (!require(pkg, character.only = TRUE)) stop(paste("load failure:", pkg))
  }
}
```

Inicialmente se debe estar posicionado en el directorio de la tarea:

```
setwd("C:/Users/José Manuel/Documents/ICD/mineria-de-datos-predictiva-josemalvarezg1")
```

Pre-Procesamiento

Inicialmente se debe leer el dataset adult.data:

```
adultsData = read.csv(file = "../data/adult.data", header = F)
```

Luego se debe leer el dataset adult.test:

```
adultsTest = read.csv(file = "../data/adult.test", header = F)
```

Se trabajará con el dataset adult, el cual está compuesto por los dos anteriores:

```
adults <- rbind(adultsData, adultsTest)
```

Se identificarán las columnas del dataset de la siguiente manera:

```
colnames(adults) <- c("Edad", "Tipo_empleo", "Peso_final", "Educación", "Número_educación", "Estado_civ
```

Algunos elementos de la columna Clase tienen un punto al final; y en realidad, sólo deberían existir dos clases (<=50 y >50), por lo que, de manera de limpieza del dataset, a esos elementos se le eliminará el punto:

```
adults$Clase <- sub("50K.", "50K", adults$Clase)
```

Se ignoran los registros que contienen valores desconocidos:

```r
adults <- na.omit(adults[1:15])
```

Si se desea realizar un análisis exploratorio para estudiar más a fondo el dataset se debe realizar lo siguiente:

```r
library(FactoMineR)
#Se muestras valores de interés del dataset
# head(adults)
# dim(adults)
# names(adults)
# str(adults)
# attributes(adults)
# summary(adults)
# pca <- PCA(adults)
```

Árboles de Decisión

Se cargan las bibliotecas para la utilización del método de clasificación de árboles de decisión:

```r
library("rpart")
library("rpart.plot")
```

Se guarda el número de registros del dataset:

```r
n <- nrow(adults)
```

Para no depender de un orden en específico y hacer el entrenamiento más aleatorio, se barajea el dataset:

```r
shuffled <- adults[sample(n),]
```

Como se vio en clases, se calculará un índice de training del 70%:

```r
train_indices <- 1:round(0.7 * n)
```

Se obtiene el índice de testing:

```r
test_indices <- (round(0.7 * n) + 1):n
```

Se obtiene el conjunto de training:

```r
train <- shuffled[train_indices,]
```

Se obtiene el conjunto de testing:

```r
test <- shuffled[test_indices,]
```

Se obtiene el árbol de decisión en base a la columna Clase del dataset:

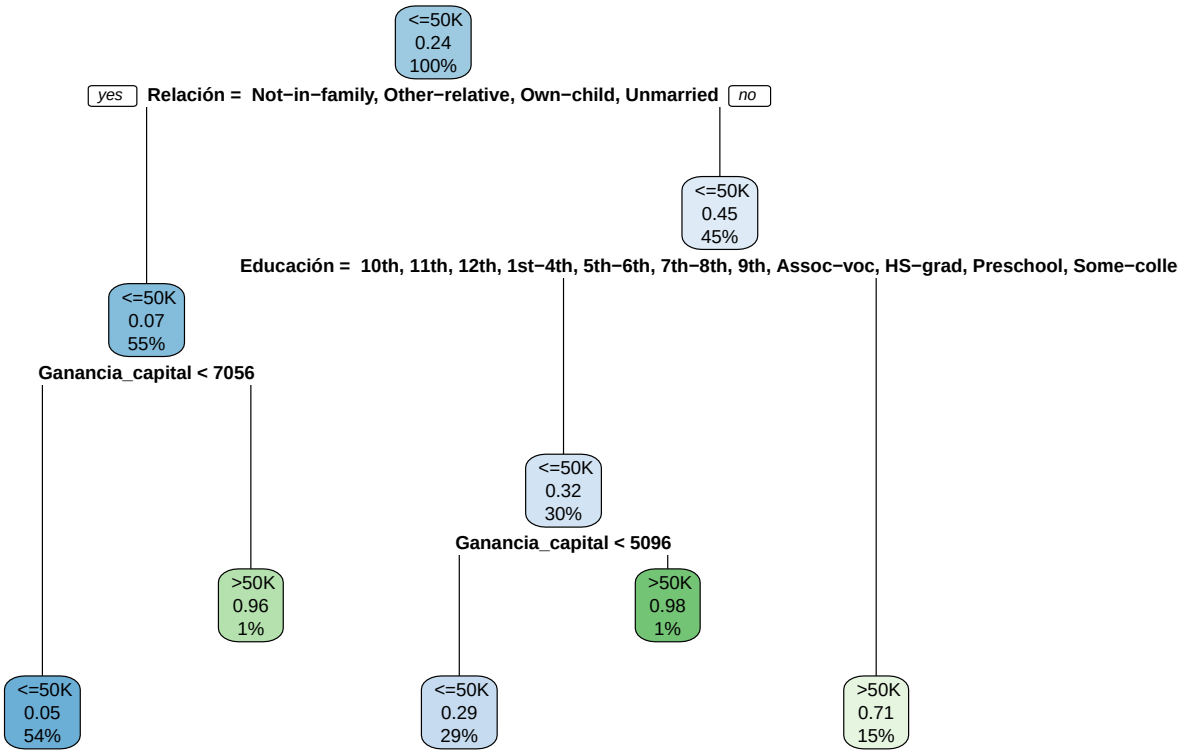
```
tree <- rpart(Clase ~ ., train, method = "class")
```

Se obtienen todos los valores probables y se guardan para ser utilizados posteriormente:

```
all_prob <- predict(tree,test,type="prob")
prob <- all_prob[,2]
```

Se grafica el árbol de decisión:

```
rpart.plot(tree)
```



Curvas ROC

Se cargan las biblioteca ROCR para la utilización de curvas ROC:

```
library("ROCR")
```

Se obtiene el árbol de decisión nuevamente y se guarda la predicciónd el mismo:

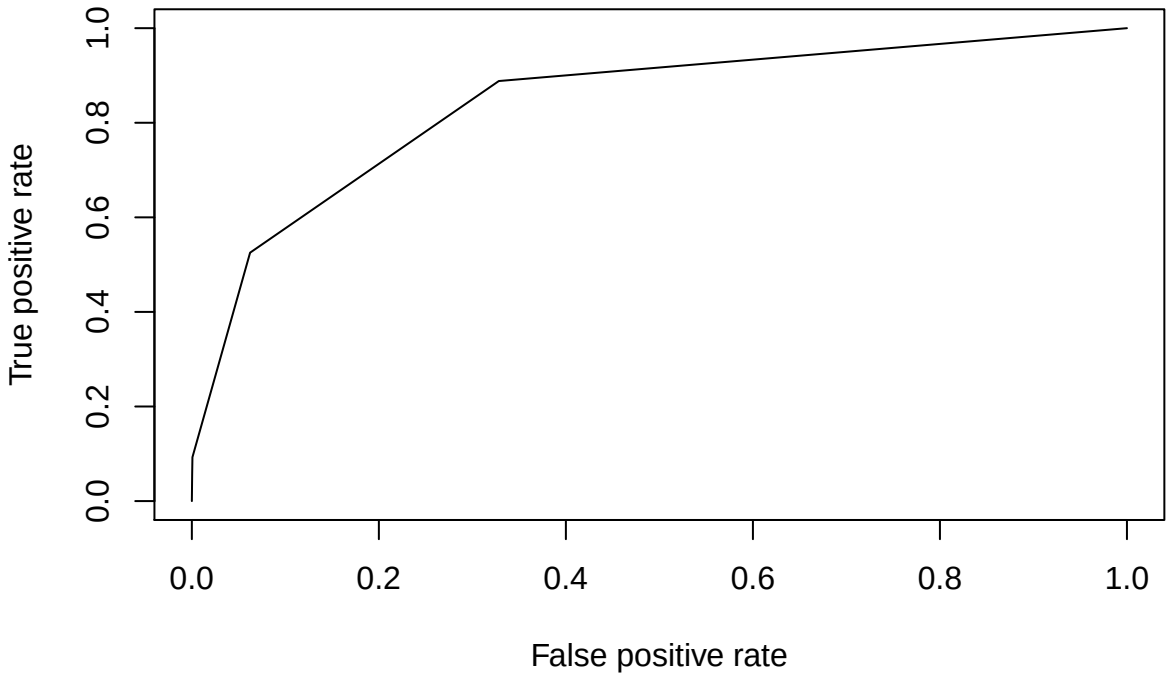
```
set.seed(1)
tree <- rpart(Clase ~ ., train, method = "class")
prob <- predict(tree, test, type = "prob")[,2]
prob_tree <- prob
pred <- prediction(prob,test$Clase)
```

Se obtiene la tasa de verdaderos y falsos positivos:

```
perf <- performance(pred,"tpr","fpr")
```

Se grafica la tasa anterior:

```
plot(perf)
```



Se obtiene el área bajo la curva:

```
set.seed(1)
tree <- rpart(Clase ~ ., train, method = "class")
prob <- predict(tree, test, type = "prob")[,2]
prob_curve <- prob
pred <- prediction(prob,test$Clase)
perf <- performance(pred,"auc")
```

Se obtiene el porcentaje de precisión:

```
perf@y.values[[1]] * 100
```

```
## [1] 84.12611
```

Se comparan los métodos:

```
pred_tree <- prediction(prob_tree,test$Clase)
pred_curve <- prediction(prob_curve,test$Clase)
perf_tree <- performance(pred_tree,"tpr","fpr")
perf_curve <- performance(pred_curve,"tpr","fpr")
```

Se grafica el desempeño de ambos métodos

```
plot(perf_tree)
plot(perf_curve)
```