script\_ranger\_thr.R

Poitou

2020-06-24

library(ranger)  
library(randomForest)

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:ranger':  
##   
## importance

source('../utils/utils\_oblig.R')

## Loading required package: caret

## Loading required package: lattice

## Loading required package: ggplot2

##   
## Attaching package: 'ggplot2'

## The following object is masked from 'package:randomForest':  
##   
## margin

set.seed(117)  
  
script.name <- 'ranger\_thr'  
  
script.date <- 'v1'  
  
script.start <- Sys.time()  
  
print('Start')

## [1] "Start"

# leer el archivo dataset.csv de la carpeta  
  
dataset <- read.csv('../data/dataset.csv')  
  
# ver la estructura del dataset  
  
# str(dataset)  
  
# asignar el nombre del jugador como nombre de la fila  
  
rownames(dataset) <- dataset$CustomerID  
  
df <- dataset[,-1]  
  
df$ServiceArea <- NULL  
  
df <- na.roughfix(df)  
  
print('\*\* Distribucion a-priori de la variable a predecir')

## [1] "\*\* Distribucion a-priori de la variable a predecir"

print(prop.table(table(df$Churn)))

##   
## No Yes   
## 0.7114167 0.2885833

df.part <- train\_dev\_partition(df, p = 0.8)  
  
df.thr\_vec <- seq(0.2, 0.4, 0.025)  
  
df.fn\_summary <- function(data, lev = NULL, model = NULL) {  
 fn\_summaryUtilityThr(data, df.thr\_vec)  
}  
  
df.metric <- 'utility'  
  
df.form <- Churn ~ .  
  
print('\*\* GBM')

## [1] "\*\* GBM"

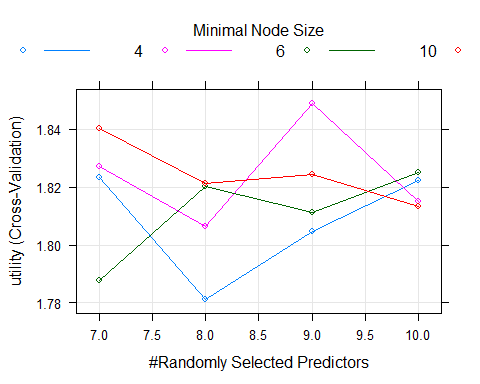
df.ranger.ctrl <- trainControl(method = 'cv',  
 number = 2,  
 verboseIter = TRUE,  
 classProbs = TRUE,  
 search = 'grid',  
 summaryFunction = df.fn\_summary)  
  
df.ranger.grid <- expand.grid(mtry = c(7, 8, 9, 10),   
 min.node.size = c(1, 4, 6, 10),  
 splitrule = 'gini')  
   
df.ranger <- train(form = df.form,   
 data = df.part$train,   
 method = 'ranger',   
 trControl = df.ranger.ctrl,  
 tuneGrid = df.ranger.grid,  
 verbose = FALSE,  
 metric = df.metric)

## + Fold1: mtry= 7, min.node.size= 1, splitrule=gini   
## - Fold1: mtry= 7, min.node.size= 1, splitrule=gini   
## + Fold1: mtry= 8, min.node.size= 1, splitrule=gini   
## - Fold1: mtry= 8, min.node.size= 1, splitrule=gini   
## + Fold1: mtry= 9, min.node.size= 1, splitrule=gini   
## - Fold1: mtry= 9, min.node.size= 1, splitrule=gini   
## + Fold1: mtry=10, min.node.size= 1, splitrule=gini   
## - Fold1: mtry=10, min.node.size= 1, splitrule=gini   
## + Fold1: mtry= 7, min.node.size= 4, splitrule=gini   
## - Fold1: mtry= 7, min.node.size= 4, splitrule=gini   
## + Fold1: mtry= 8, min.node.size= 4, splitrule=gini   
## - Fold1: mtry= 8, min.node.size= 4, splitrule=gini   
## + Fold1: mtry= 9, min.node.size= 4, splitrule=gini   
## - Fold1: mtry= 9, min.node.size= 4, splitrule=gini   
## + Fold1: mtry=10, min.node.size= 4, splitrule=gini   
## - Fold1: mtry=10, min.node.size= 4, splitrule=gini   
## + Fold1: mtry= 7, min.node.size= 6, splitrule=gini   
## - Fold1: mtry= 7, min.node.size= 6, splitrule=gini   
## + Fold1: mtry= 8, min.node.size= 6, splitrule=gini   
## - Fold1: mtry= 8, min.node.size= 6, splitrule=gini   
## + Fold1: mtry= 9, min.node.size= 6, splitrule=gini   
## - Fold1: mtry= 9, min.node.size= 6, splitrule=gini   
## + Fold1: mtry=10, min.node.size= 6, splitrule=gini   
## - Fold1: mtry=10, min.node.size= 6, splitrule=gini   
## + Fold1: mtry= 7, min.node.size=10, splitrule=gini   
## - Fold1: mtry= 7, min.node.size=10, splitrule=gini   
## + Fold1: mtry= 8, min.node.size=10, splitrule=gini   
## - Fold1: mtry= 8, min.node.size=10, splitrule=gini   
## + Fold1: mtry= 9, min.node.size=10, splitrule=gini   
## - Fold1: mtry= 9, min.node.size=10, splitrule=gini   
## + Fold1: mtry=10, min.node.size=10, splitrule=gini   
## - Fold1: mtry=10, min.node.size=10, splitrule=gini   
## + Fold2: mtry= 7, min.node.size= 1, splitrule=gini   
## - Fold2: mtry= 7, min.node.size= 1, splitrule=gini   
## + Fold2: mtry= 8, min.node.size= 1, splitrule=gini   
## - Fold2: mtry= 8, min.node.size= 1, splitrule=gini   
## + Fold2: mtry= 9, min.node.size= 1, splitrule=gini   
## - Fold2: mtry= 9, min.node.size= 1, splitrule=gini   
## + Fold2: mtry=10, min.node.size= 1, splitrule=gini   
## - Fold2: mtry=10, min.node.size= 1, splitrule=gini   
## + Fold2: mtry= 7, min.node.size= 4, splitrule=gini   
## - Fold2: mtry= 7, min.node.size= 4, splitrule=gini   
## + Fold2: mtry= 8, min.node.size= 4, splitrule=gini   
## - Fold2: mtry= 8, min.node.size= 4, splitrule=gini   
## + Fold2: mtry= 9, min.node.size= 4, splitrule=gini   
## - Fold2: mtry= 9, min.node.size= 4, splitrule=gini   
## + Fold2: mtry=10, min.node.size= 4, splitrule=gini   
## - Fold2: mtry=10, min.node.size= 4, splitrule=gini   
## + Fold2: mtry= 7, min.node.size= 6, splitrule=gini   
## - Fold2: mtry= 7, min.node.size= 6, splitrule=gini   
## + Fold2: mtry= 8, min.node.size= 6, splitrule=gini   
## - Fold2: mtry= 8, min.node.size= 6, splitrule=gini   
## + Fold2: mtry= 9, min.node.size= 6, splitrule=gini   
## - Fold2: mtry= 9, min.node.size= 6, splitrule=gini   
## + Fold2: mtry=10, min.node.size= 6, splitrule=gini   
## - Fold2: mtry=10, min.node.size= 6, splitrule=gini   
## + Fold2: mtry= 7, min.node.size=10, splitrule=gini   
## - Fold2: mtry= 7, min.node.size=10, splitrule=gini   
## + Fold2: mtry= 8, min.node.size=10, splitrule=gini   
## - Fold2: mtry= 8, min.node.size=10, splitrule=gini   
## + Fold2: mtry= 9, min.node.size=10, splitrule=gini   
## - Fold2: mtry= 9, min.node.size=10, splitrule=gini   
## + Fold2: mtry=10, min.node.size=10, splitrule=gini   
## - Fold2: mtry=10, min.node.size=10, splitrule=gini   
## Aggregating results  
## Selecting tuning parameters  
## Fitting mtry = 9, splitrule = gini, min.node.size = 4 on full training set

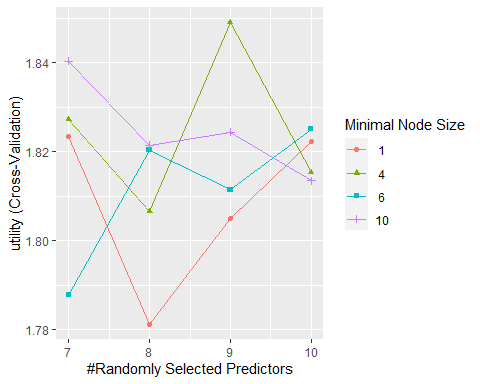
print(df.ranger)

## Random Forest   
##   
## 38400 samples  
## 55 predictor  
## 2 classes: 'No', 'Yes'   
##   
## No pre-processing  
## Resampling: Cross-Validated (2 fold)   
## Summary of sample sizes: 19199, 19201   
## Resampling results across tuning parameters:  
##   
## mtry min.node.size utility prob\_thr  
## 7 1 1.823386 0.3000   
## 7 4 1.827252 0.3000   
## 7 6 1.787696 0.2875   
## 7 10 1.840339 0.2875   
## 8 1 1.781160 0.3000   
## 8 4 1.806446 0.2875   
## 8 6 1.820312 0.3000   
## 8 10 1.821446 0.3000   
## 9 1 1.804870 0.3000   
## 9 4 1.849051 0.2875   
## 9 6 1.811395 0.3125   
## 9 10 1.824376 0.3125   
## 10 1 1.822305 0.3000   
## 10 4 1.815210 0.3125   
## 10 6 1.825028 0.3000   
## 10 10 1.813425 0.3000   
##   
## Tuning parameter 'splitrule' was held constant at a value of gini  
## utility was used to select the optimal model using the largest value.  
## The final values used for the model were mtry = 9, splitrule = gini  
## and min.node.size = 4.

plot(df.ranger)



ggplot(df.ranger)



df.ranger.model <- df.ranger$finalModel  
  
df.ranger.results <- fn\_results(df.ranger)  
  
print('Umbral')

## [1] "Umbral"

print(df.ranger.results$prob\_thr)

## [1] 0.2875

print('Utilidad en train')

## [1] "Utilidad en train"

print(df.ranger.results$utility)

## [1] 1.849051

print('Utilidad en dev')

## [1] "Utilidad en dev"

df.ranger.dev.prob <- predict(df.ranger, newdata = df.part$dev, type = 'prob')  
df.ranger.dev.pred <- fn\_pred(df.ranger.dev.prob, thr = df.ranger.results$prob\_thr)  
  
df.ranger.dev.utility <- fn\_utility(df.ranger.dev.pred, df.part$dev$Churn)  
  
print(df.ranger.dev.utility)

## [1] 1.846771

print('Matriz de confusion en dev')

## [1] "Matriz de confusion en dev"

df.ranger.dev.cm <- conf\_matrix(df.ranger.dev.pred, df.part$dev$Churn)  
  
print(df.ranger.dev.cm)

## Reference  
## Prediction No Yes  
## No 3688 805  
## Yes 3169 1938

print('\*\* Generacion de la prediccion sobre test sample')

## [1] "\*\* Generacion de la prediccion sobre test sample"

test\_sample <- read.csv('../data/test\_sample.csv')  
rownames(test\_sample) <- test\_sample$CustomerID  
test\_sample$CustomerID <- NULL  
test\_sample$ServiceArea <- NULL  
test\_sample <- na.roughfix(test\_sample)  
  
file\_id <- paste0(c(script.name, script.date), collapse = ' ')  
  
gen\_prediction(df.ranger, test\_sample, prob\_thr = df.ranger.results$prob\_thr, id = file\_id)  
  
print('Done')

## [1] "Done"

script.done <- Sys.time()  
  
print(script.done - script.start)

## Time difference of 27.36314 mins