Ensemble Models - Undersampling

Nick Wawee

3/26/2021

## Loading

rm(list = ls())  
knitr::opts\_chunk$set(echo = TRUE)  
source('/Users/nickwawee/Desktop/BGSU/MSA\_6440/Week\_2/myfunctions.R')  
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

library(ada)

## Loading required package: rpart

library(plyr)  
  
plot\_cm <- function(cm, plot\_title) {#https://stackoverflow.com/questions/23891140/r-how-to-visualize-confusion-matrix-using-the-caret-package  
  
 layout(matrix(c(1,1,2)))  
 par(mar=c(2,2,2,2))  
 plot(c(100, 345), c(300, 450), type = "n", xlab="", ylab="", xaxt='n', yaxt='n')  
 title(plot\_title, cex.main=2)  
  
 # create the matrix   
 rect(150, 430, 240, 370, col='#3F97D0')  
 text(195, 435, 'Class1', cex=1.2)  
 rect(250, 430, 340, 370, col='#F7AD50')  
 text(295, 435, 'Class2', cex=1.2)  
 text(125, 370, 'Predicted', cex=1.3, srt=90, font=2)  
 text(245, 450, 'Actual', cex=1.3, font=2)  
 rect(150, 305, 240, 365, col='#F7AD50')  
 rect(250, 305, 340, 365, col='#3F97D0')  
 text(140, 400, 'Class1', cex=1.2, srt=90)  
 text(140, 335, 'Class2', cex=1.2, srt=90)  
  
 # add in the cm results   
 res <- as.numeric(cm$table)  
 text(195, 400, res[1], cex=1.6, font=2, col='white')  
 text(195, 335, res[2], cex=1.6, font=2, col='white')  
 text(295, 400, res[3], cex=1.6, font=2, col='white')  
 text(295, 335, res[4], cex=1.6, font=2, col='white')  
  
 # add in the specifics   
 plot(c(100, 0), c(100, 0), type = "n", xlab="", ylab="", main = "DETAILS", xaxt='n', yaxt='n')  
 text(10, 85, names(cm$byClass[1]), cex=1.2, font=2)  
 text(10, 70, round(as.numeric(cm$byClass[1]), 3), cex=1.2)  
 text(30, 85, names(cm$byClass[2]), cex=1.2, font=2)  
 text(30, 70, round(as.numeric(cm$byClass[2]), 3), cex=1.2)  
 text(50, 85, names(cm$byClass[5]), cex=1.2, font=2)  
 text(50, 70, round(as.numeric(cm$byClass[5]), 3), cex=1.2)  
 text(70, 85, names(cm$byClass[6]), cex=1.2, font=2)  
 text(70, 70, round(as.numeric(cm$byClass[6]), 3), cex=1.2)  
 text(90, 85, names(cm$byClass[7]), cex=1.2, font=2)  
 text(90, 70, round(as.numeric(cm$byClass[7]), 3), cex=1.2)  
  
 # add in the accuracy information   
 text(30, 35, names(cm$overall[1]), cex=1.5, font=2)  
 text(30, 20, round(as.numeric(cm$overall[1]), 3), cex=1.4)  
 text(70, 35, names(cm$overall[2]), cex=1.5, font=2)  
 text(70, 20, round(as.numeric(cm$overall[2]), 3), cex=1.4)  
}

bank = read.csv('../data/processed/BankChurners\_filtered.csv', stringsAsFactors = T)  
  
# Creating dummy variables for Education  
bank$High\_School <- ifelse(bank$Education\_Level == 'High School', 1, 0)  
bank$College <- ifelse(bank$Education\_Level == 'College', 1, 0)  
bank$Graduate <- ifelse(bank$Education\_Level == 'Graduate', 1, 0)  
bank$Uneducated <- ifelse(bank$Education\_Level == 'Uneducated', 1, 0)  
bank$Post\_Graduate <- ifelse(bank$Education\_Level == 'Post-Graduate', 1, 0)  
bank$Doctorate <- ifelse(bank$Education\_Level == 'Doctorate', 1, 0)  
  
# Creating dummy variables for Marital Status  
bank$Married <- ifelse(bank$Marital\_Status == 'Married', 1, 0)  
bank$Single <- ifelse(bank$Marital\_Status == 'Single', 1, 0)  
bank$Divorced <- ifelse(bank$Marital\_Status == 'Divorced', 1, 0)  
  
# Creating dummy variables for Gender  
bank$Male <- ifelse(bank$Gender == 'M', 1, 0)  
bank$Female <- ifelse(bank$Gender == 'F', 1, 0)  
  
# Creating dummy variables for Income  
bank$Income\_5 <- ifelse(bank$Income\_Category == '$120K +', 1, 0)  
bank$Income\_4 <- ifelse(bank$Income\_Category == '$80K - $120K', 1, 0)  
bank$Income\_3 <- ifelse(bank$Income\_Category == '$60K - $80K', 1, 0)  
bank$Income\_2 <- ifelse(bank$Income\_Category == '$40K - $60K', 1, 0)  
bank$Income\_1 <- ifelse(bank$Income\_Category == 'Less than $40K', 1, 0)  
  
#Creating dummy variables for Card category  
bank$Blue\_Card <- ifelse(bank$Card\_Category == 'Blue', 1, 0)  
bank$Gold\_Card <- ifelse(bank$Card\_Category == 'Gold', 1, 0)  
bank$Plat\_Card <- ifelse(bank$Card\_Category == 'Platinum', 1, 0)  
bank$Silver\_Card <- ifelse(bank$Card\_Category == 'Silver', 1, 0)  
  
bank <- bank[,c(-1,-2,-5,-7:-10,-39:-42)]  
  
RNGkind (sample.kind = "Rounding")

## Warning in RNGkind(sample.kind = "Rounding"): non-uniform 'Rounding' sampler  
## used

set.seed(0)  
  
dfls = partition.2(bank, 0.7)  
  
test.data = dfls$data.test  
training.data = dfls$data.train  
  
  
# Undersampling  
set.seed(0)  
all\_exist <- training.data[which(training.data$Attrition\_Flag == 'Existing Customer'),]  
all\_attrit <- training.data[which(training.data$Attrition\_Flag == 'Attrited Customer'),]  
  
random <- sample(1:nrow(all\_exist), nrow(all\_attrit), replace = FALSE)  
train.data <- all\_exist[random,]  
  
training.under <- rbind(all\_attrit, train.data)  
table(training.under$Attrition\_Flag)

##   
## Attrited Customer Existing Customer   
## 776 776

## Random Forest

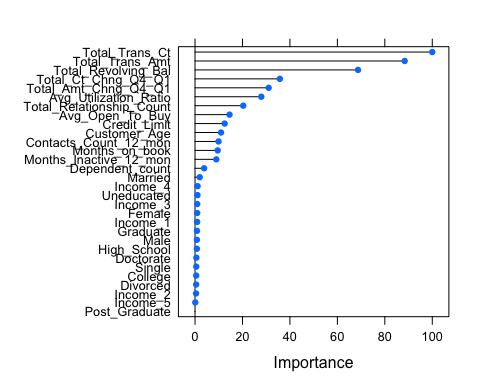
set.seed(0)  
modelLookup("rf")

## model parameter label forReg forClass probModel  
## 1 rf mtry #Randomly Selected Predictors TRUE TRUE TRUE

train\_control <- trainControl(method="cv", number=10)  
rf <- train(Attrition\_Flag ~ ., data = training.under, method = "rf", ntree = 50,rControl = train\_control, tuneGrid = expand.grid(mtry = c(4, 5, 6, 7, 8)), metric = 'Kappa')  
  
  
print(rf)

## Random Forest   
##   
## 1552 samples  
## 30 predictor  
## 2 classes: 'Attrited Customer', 'Existing Customer'   
##   
## No pre-processing  
## Resampling: Bootstrapped (25 reps)   
## Summary of sample sizes: 1552, 1552, 1552, 1552, 1552, 1552, ...   
## Resampling results across tuning parameters:  
##   
## mtry Accuracy Kappa   
## 4 0.9122117 0.8242793  
## 5 0.9164636 0.8327629  
## 6 0.9223243 0.8445256  
## 7 0.9274030 0.8546678  
## 8 0.9305757 0.8610205  
##   
## Kappa was used to select the optimal model using the largest value.  
## The final value used for the model was mtry = 8.

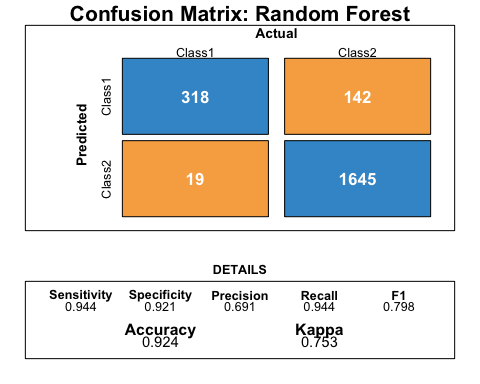
plot(varImp(rf))



rf$finalModel

##   
## Call:  
## randomForest(x = x, y = y, ntree = 50, mtry = param$mtry, rControl = ..2)   
## Type of random forest: classification  
## Number of trees: 50  
## No. of variables tried at each split: 8  
##   
## OOB estimate of error rate: 6.77%  
## Confusion matrix:  
## Attrited Customer Existing Customer class.error  
## Attrited Customer 723 53 0.06829897  
## Existing Customer 52 724 0.06701031

# get prediction on the test data  
pred.test.rf = predict(rf$finalModel, test.data, type = 'class')  
  
# create confusion matrix  
cm = confusionMatrix(pred.test.rf, test.data$Attrition\_Flag, positive = "Attrited Customer")  
  
cm\_plot = plot\_cm(cm, 'Confusion Matrix: Random Forest' )



cm\_plot

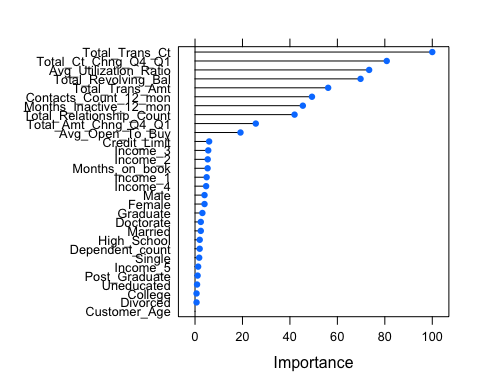
## NULL

## ADABoost

modelLookup("ada")

## model parameter label forReg forClass probModel  
## 1 ada iter #Trees FALSE TRUE TRUE  
## 2 ada maxdepth Max Tree Depth FALSE TRUE TRUE  
## 3 ada nu Learning Rate FALSE TRUE TRUE

set.seed(0)  
train\_control <- trainControl(method="cv", number=10)  
  
tgrid <- expand.grid(iter = c(150),   
 maxdepth = c(6, 7, 8),   
 nu = c(0.12, 0.15, 0.18))  
ada\_m <- train(Attrition\_Flag~ . , data = training.under, method = "ada", metric = "Kappa",  
 trControl = train\_control, tuneGrid = tgrid)  
plot(varImp(ada\_m))



# get prediction on the test data  
pred.test.ada = predict(ada\_m$finalModel, test.data)  
  
# create confusion matrix  
cm = confusionMatrix(pred.test.ada, test.data$Attrition\_Flag, positive = "Attrited Customer")  
  
cm\_plot = plot\_cm(cm, 'Confusion Matrix: AdaBoost' )

