Trees

Akhil Gupta

May 13, 2017

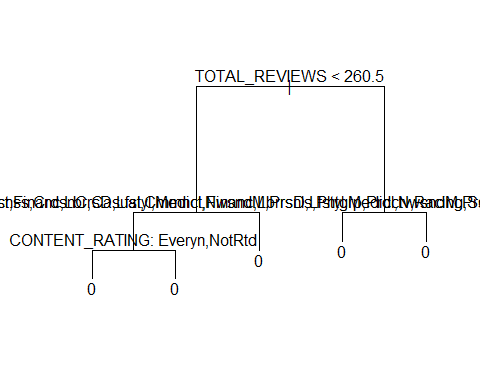
# Uploading the file into R  
  
setwd("C:/Users/gupta/DataMiningProject")  
project <- read.csv("app\_metadata\_cleaned\_removed\_min\_downloads\_above\_5m.csv")  
  
# creating duplicate copy of project  
data <- project  
class(project$PRICE)

## [1] "numeric"

# removing not required files  
data$APP\_ID <- NULL  
data$APP\_NAME <- NULL  
data$DOWNLOADS <- NULL  
data$CURRENT\_VERSION <- NULL  
data$LASTUPDATED <- NULL  
data$DEVELOPER\_SITE <- NULL  
data$DEVELOPER\_CONTACT <- NULL  
data$DEVELOPER\_NAME <- NULL  
data$MIN\_REQUIRED\_ANDROID <- NULL  
  
# creating factors of independent variables and changing variables from integer to numeric  
data$CATEGORY <- as.factor(data$CATEGORY)  
data$PRICE <- as.numeric(data$PRICE)  
data$CONTENT\_RATING <- as.factor(data$CONTENT\_RATING)  
data$DOWNLOAD\_MIN <- as.numeric(data$DOWNLOAD\_MIN)  
data$DOWNLOAD\_MAX <- as.numeric(data$DOWNLOAD\_MAX)  
data$SIZE\_MEGABYTES <- as.numeric(data$SIZE\_MEGABYTES)  
data$MIN\_REQ\_ANDROID\_FIRST <- as.factor(data$MIN\_REQ\_ANDROID\_FIRST)  
data$TOTAL\_REVIEWS <- as.numeric(data$TOTAL\_REVIEWS)  
data$AVERAGE\_RATING <- as.numeric(data$AVERAGE\_RATING)  
data$X5RATING <- as.numeric(data$X5RATING)  
data$X4RATING <- as.numeric(data$X4RATING)  
data$X3RATING <- as.numeric(data$X3RATING)  
data$X2RATING <- as.numeric(data$X2RATING)  
data$X1RATING <- as.numeric(data$X1RATING)  
data$spam <- as.factor(data$spam)  
  
#setting seed and dividing the dataset into Training and Test  
set.seed(12345)  
train <- sample(nrow(data),0.7\*nrow(data))  
project\_training <- data[train,]  
project\_Validation <- data[-train,]  
  
library(tree)  
library(ISLR)  
  
  
  
#Under Sampling Data  
#Taking all the observations with dependent variable = 1  
train\_under <- project\_training[project\_training$spam==1,]  
  
#Randomly select observations with dependent variable = 0  
zeroObs <- project\_training[project\_training$spam==0,]  
set.seed(123457)  
rearrangedZeroObs <- zeroObs[sample(nrow(zeroObs), length(train\_under$spam)),]  
  
#Appending rows of randomly selected 0s in our undersampled data frame  
train\_under <- rbind(train\_under, rearrangedZeroObs)  
  
  
  
###############################sampled tree#########################################  
# creating the classification tree  
tree.project <- tree(spam ~ CATEGORY+PRICE+CONTENT\_RATING+DOWNLOAD\_MAX+DOWNLOAD\_MIN+TOTAL\_REVIEWS+AVERAGE\_RATING+X5RATING+X4RATING+X3RATING+X2RATING+X1RATING,data = project\_training)  
  
summary(tree.project)

##   
## Classification tree:  
## tree(formula = spam ~ CATEGORY + PRICE + CONTENT\_RATING + DOWNLOAD\_MAX +   
## DOWNLOAD\_MIN + TOTAL\_REVIEWS + AVERAGE\_RATING + X5RATING +   
## X4RATING + X3RATING + X2RATING + X1RATING, data = project\_training)  
## Variables actually used in tree construction:  
## [1] "TOTAL\_REVIEWS" "CATEGORY" "CONTENT\_RATING"  
## Number of terminal nodes: 5   
## Residual mean deviance: 0.4697 = 2510 / 5343   
## Misclassification error rate: 0.07386 = 395 / 5348

#plotting the tree  
plot(tree.project)  
text(tree.project, pretty=6)



# Creating confusion matrix of pridcted values in test data set   
tree.predict <- predict(tree.project, project\_Validation, type = "class")  
confmatrix <- table (project\_Validation$spam,tree.predict)  
confmatrix

## tree.predict  
## 0 1  
## 0 2136 0  
## 1 156 0

#calculating the accuracy  
accuracy <- (confmatrix[1,1]+confmatrix[2,2])/sum(confmatrix)  
accuracy

## [1] 0.9319372

# calculating the sensitivity  
sensitivity <- (confmatrix[2,2]/(confmatrix[2,2]+confmatrix[2,1]))  
sensitivity

## [1] 0

# Cross validation technique to find the best tree  
set.seed(123)  
cv.credit <- cv.tree(tree.project,FUN =prune.misclass, K=10)  
names(cv.credit)

## [1] "size" "dev" "k" "method"

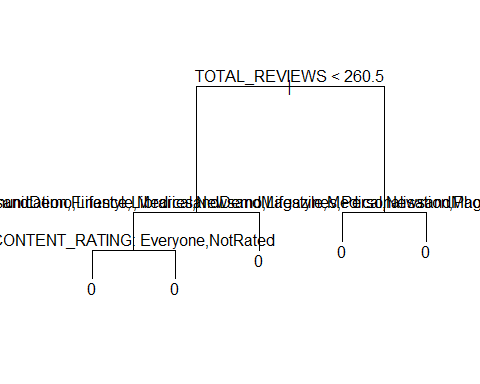
cv.credit

## $size  
## [1] 5 1  
##   
## $dev  
## [1] 395 395  
##   
## $k  
## [1] -Inf 0  
##   
## $method  
## [1] "misclass"  
##   
## attr(,"class")  
## [1] "prune" "tree.sequence"

#pruning the tree with best no. of terminal nodes  
prune.credit <- prune.misclass(tree.project, best =8)

## Warning in prune.tree(tree = tree.project, best = 8, method = "misclass"):  
## best is bigger than tree size

plot(prune.credit)  
text(prune.credit, pretty=0)



summary(prune.credit)

##   
## Classification tree:  
## tree(formula = spam ~ CATEGORY + PRICE + CONTENT\_RATING + DOWNLOAD\_MAX +   
## DOWNLOAD\_MIN + TOTAL\_REVIEWS + AVERAGE\_RATING + X5RATING +   
## X4RATING + X3RATING + X2RATING + X1RATING, data = project\_training)  
## Variables actually used in tree construction:  
## [1] "TOTAL\_REVIEWS" "CATEGORY" "CONTENT\_RATING"  
## Number of terminal nodes: 5   
## Residual mean deviance: 0.4697 = 2510 / 5343   
## Misclassification error rate: 0.07386 = 395 / 5348

# Creating confusion matrix of pridcted values in test data set with pruned tree  
tree.predict1 <- predict(prune.credit, project\_Validation, type = "class")  
confmatrix1 <- table (tree.predict1, project\_Validation$spam)  
confmatrix1

##   
## tree.predict1 0 1  
## 0 2136 156  
## 1 0 0

#calculating accuracy of pruned tree  
accuracy1 <- (confmatrix1[1,1]+confmatrix1[2,2])/sum(confmatrix1)  
accuracy1

## [1] 0.9319372

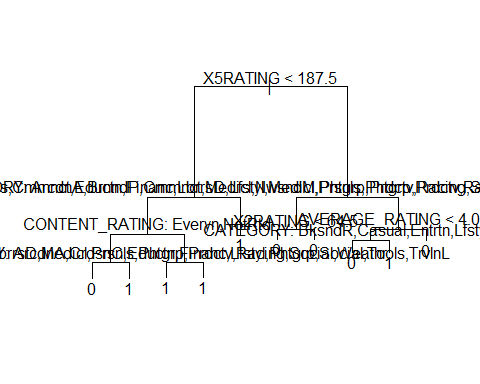
# calculating the sensitivity of pruned tree  
sensitivity1 <- (confmatrix1[2,2]/(confmatrix1[2,2]+confmatrix1[2,1]))  
sensitivity1

## [1] NaN

##############################under sampled treee#####################################  
  
# creating the classification tree  
tree.project <- tree(spam ~ CATEGORY+PRICE+CONTENT\_RATING+DOWNLOAD\_MAX+DOWNLOAD\_MIN+TOTAL\_REVIEWS+AVERAGE\_RATING+X5RATING+X4RATING+X3RATING+X2RATING+X1RATING,data = train\_under)  
  
summary(tree.project)

##   
## Classification tree:  
## tree(formula = spam ~ CATEGORY + PRICE + CONTENT\_RATING + DOWNLOAD\_MAX +   
## DOWNLOAD\_MIN + TOTAL\_REVIEWS + AVERAGE\_RATING + X5RATING +   
## X4RATING + X3RATING + X2RATING + X1RATING, data = train\_under)  
## Variables actually used in tree construction:  
## [1] "X5RATING" "CATEGORY" "CONTENT\_RATING" "X2RATING"   
## [5] "AVERAGE\_RATING"  
## Number of terminal nodes: 10   
## Residual mean deviance: 1.028 = 801.6 / 780   
## Misclassification error rate: 0.2494 = 197 / 790

#plotting the tree  
plot(tree.project)  
text(tree.project, pretty=6)



# Creating confusion matrix of pridcted values in test data set   
tree.predict <- predict(tree.project, project\_Validation, type = "class")  
confmatrix <- table (project\_Validation$spam,tree.predict)  
confmatrix

## tree.predict  
## 0 1  
## 0 1302 834  
## 1 42 114

#calculating the accuracy  
accuracy <- (confmatrix[1,1]+confmatrix[2,2])/sum(confmatrix)  
accuracy

## [1] 0.617801

# calculating the sensitivity  
sensitivity <- (confmatrix[2,2]/(confmatrix[2,2]+confmatrix[2,1]))  
sensitivity

## [1] 0.7307692

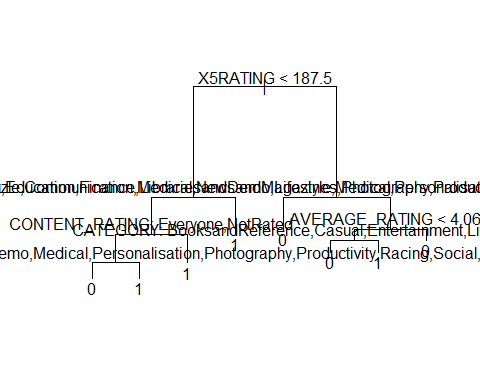
# Cross validation technique to find the best tree  
set.seed(123)  
cv.credit <- cv.tree(tree.project,FUN =prune.misclass, K=10)  
names(cv.credit)

## [1] "size" "dev" "k" "method"

cv.credit

## $size  
## [1] 10 8 7 5 4 2 1  
##   
## $dev  
## [1] 241 238 244 245 239 264 436  
##   
## $k  
## [1] -Inf 0.0 3.0 3.5 10.0 19.5 139.0  
##   
## $method  
## [1] "misclass"  
##   
## attr(,"class")  
## [1] "prune" "tree.sequence"

#pruning the tree with best no. of terminal nodes  
prune.credit <- prune.misclass(tree.project, best =8)  
plot(prune.credit)  
text(prune.credit, pretty=0)



summary(prune.credit)

##   
## Classification tree:  
## snip.tree(tree = tree.project, nodes = c(6L, 9L))  
## Variables actually used in tree construction:  
## [1] "X5RATING" "CATEGORY" "CONTENT\_RATING" "AVERAGE\_RATING"  
## Number of terminal nodes: 8   
## Residual mean deviance: 1.059 = 828.1 / 782   
## Misclassification error rate: 0.2494 = 197 / 790

# Creating confusion matrix of pridcted values in test data set with pruned tree  
tree.predict1 <- predict(prune.credit, project\_Validation, type = "class")  
confmatrix1 <- table (tree.predict1, project\_Validation$spam)  
confmatrix1

##   
## tree.predict1 0 1  
## 0 1302 42  
## 1 834 114

#calculating accuracy of pruned tree  
accuracy1 <- (confmatrix1[1,1]+confmatrix1[2,2])/sum(confmatrix1)  
accuracy1

## [1] 0.617801

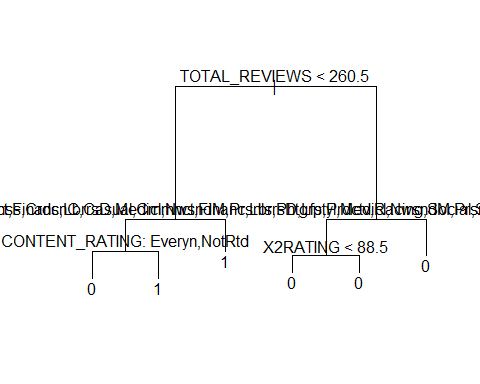
# calculating the sensitivity of pruned tree  
sensitivity1 <- (confmatrix1[2,2]/(confmatrix1[2,2]+confmatrix1[2,1]))  
sensitivity1

## [1] 0.1202532

###################################tree on over sampled data ###########################################  
  
#oversampling the data  
train\_over <- project\_training[project\_training$spam==1,]  
train\_1 <- train\_over  
for (i in seq(from=1, to=6, by=1)){  
 train\_over <- rbind(train\_over, train\_1)  
}  
  
train\_oversampling <- rbind(project\_training, train\_over)  
# running classification tree on oversampled data  
tree.project <- tree(spam ~ CATEGORY+PRICE+CONTENT\_RATING+DOWNLOAD\_MAX+DOWNLOAD\_MIN+TOTAL\_REVIEWS+AVERAGE\_RATING+X5RATING+X4RATING+X3RATING+X2RATING+X1RATING,data = train\_oversampling)  
summary(tree.project)

##   
## Classification tree:  
## tree(formula = spam ~ CATEGORY + PRICE + CONTENT\_RATING + DOWNLOAD\_MAX +   
## DOWNLOAD\_MIN + TOTAL\_REVIEWS + AVERAGE\_RATING + X5RATING +   
## X4RATING + X3RATING + X2RATING + X1RATING, data = train\_oversampling)  
## Variables actually used in tree construction:  
## [1] "TOTAL\_REVIEWS" "CATEGORY" "CONTENT\_RATING" "X2RATING"   
## Number of terminal nodes: 6   
## Residual mean deviance: 1.118 = 9060 / 8107   
## Misclassification error rate: 0.2966 = 2406 / 8113

#plotting the tree  
plot(tree.project)  
text(tree.project, pretty=6)



# Creating confusion matrix   
tree.predict <- predict(tree.project, project\_Validation, type = "class")  
confmatrix <- table (tree.predict, project\_Validation$spam)  
confmatrix

##   
## tree.predict 0 1  
## 0 1489 57  
## 1 647 99

# calculating accuracy  
accuracy <- (confmatrix[1,1]+confmatrix[2,2])/sum(confmatrix)  
accuracy

## [1] 0.6928447

# calculating the sensitivity  
sensitivity <- (confmatrix[2,2]/(confmatrix[2,2]+confmatrix[2,1]))  
sensitivity

## [1] 0.1327078

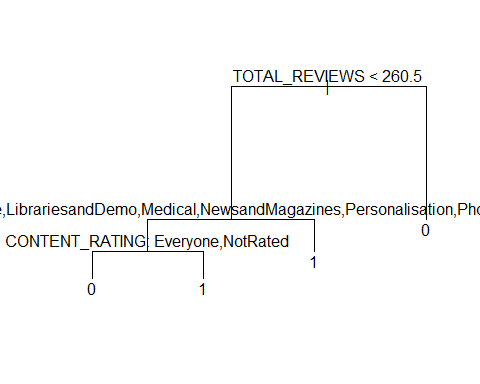
#running cross validation technique to fnd best tree  
set.seed(12345)  
cv.credit <- cv.tree(tree.project,FUN =prune.misclass, K=10)  
names(cv.credit)

## [1] "size" "dev" "k" "method"

cv.credit

## $size  
## [1] 6 4 3 1  
##   
## $dev  
## [1] 2644 2644 2850 3160  
##   
## $k  
## [1] -Inf 0 164 295  
##   
## $method  
## [1] "misclass"  
##   
## attr(,"class")  
## [1] "prune" "tree.sequence"

#pruning the tree  
prune.credit <- prune.misclass(tree.project, best =4)  
plot(prune.credit)  
text(prune.credit, pretty=0)



summary(prune.credit)

##   
## Classification tree:  
## snip.tree(tree = tree.project, nodes = 3L)  
## Variables actually used in tree construction:  
## [1] "TOTAL\_REVIEWS" "CATEGORY" "CONTENT\_RATING"  
## Number of terminal nodes: 4   
## Residual mean deviance: 1.165 = 9446 / 8109   
## Misclassification error rate: 0.2966 = 2406 / 8113

# calcuate confusion matrix  
tree.predict1 <- predict(prune.credit, project\_Validation, type = "class")  
confmatrix1 <- table (tree.predict1, project\_Validation$spam)  
confmatrix1

##   
## tree.predict1 0 1  
## 0 1489 57  
## 1 647 99

# calculating accuracy of pruned tree  
accuracy1 <- (confmatrix1[1,1]+confmatrix1[2,2])/sum(confmatrix1)  
accuracy1

## [1] 0.6928447

# calculating the sensitivity of pruned tree  
sensitivity <- (confmatrix[2,2]/(confmatrix[2,2]+confmatrix[2,1]))  
sensitivity

## [1] 0.1327078

knitr::opts\_chunk$set(echo = TRUE)