balance\_data.R

SANGHOOJEFFREY

Thu Jun 28 03:13:26 2018

if(!require(rpart)) install.packages("rpart", repos = "http://cran.us.r-project.org"); library(rpart)

## Loading required package: rpart

# randomforest, svm, decision tree 등의 machine learning(기계학습)을 현실적으로 적용시키는데  
# 가장 많이 발생하는 문제는 클래스의 불균형이다.  
  
# 분류에 해당하는 데이터의 비율이 반반이 아닌 경우 훈련 데이터 내 비율이 높은 분류 쪽으로 결과를 내놓는 모델을 만든다.  
# 예) 돌발홍수가 발생한 자료 50, 발생되지 않은 자료 950개인 경우  
# 모형에서 모두 돌발홍수가 발생되지 않는다로 예측하더라도 예측정확도는 95%임.  
# 따라서 분류를 잘하는 모형을 개발하기 위해선 데이터의 비율을 비슷하게 만들 필요가 있다.  
  
if(!require(mlbench)) install.packages("mlbench", repos = "http://cran.us.r-project.org"); library(mlbench)

## Loading required package: mlbench

data(BreastCancer)  
  
table(BreastCancer$Class)

##   
## benign malignant   
## 458 241

# 양성 benign이 458개, 악성 malignant가 241개  
# 그냥 모형을 세우개 되면 benign을 잘 예측하는 모형을 만들 가능성이 높다.   
  
# 클래스 불균형을 해결하기 위한 방법은 관찰 데이터가 적은 쪽에 더 큰 가중치(Weight)를 주는 방버  
# 데이터가 적은 쪽으로 잘못 불류했을 때 더 많은 비용(cost 또는 loss)을 부과하는 방법 등이 있다.   
  
# 또 방법으로 훈련데이터를 동일하게 만드는 방법이 있다.  
  
# 업샘플링, 다운샘플링  
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org"); library(caret)

## Loading required package: caret

## Warning: package 'caret' was built under R version 3.4.4

## Loading required package: lattice

## Loading required package: ggplot2

x <- upSample(subset(BreastCancer, select= -Class),BreastCancer$Class)  
table(x$Class)

##   
## benign malignant   
## 458 458

# x내의 행의 상당수는 중복되어 생성되어 있다.  
   
y <- downSample(subset(BreastCancer, select= -Class),BreastCancer$Class)  
table(y$Class)

##   
## benign malignant   
## 241 241

# y 의 Class가 benign인 경우 일부가 임의로 제거되었다.   
  
# UpSample으로 자료를 생성한 경우와 그냥 사용했을 경우 모형성능은 정말 차이가 날까?  
  
data <- subset(BreastCancer, select=-Id)  
set.seed(124)  
parts <- createDataPartition(data$Class, p=0.8) # 80%는 훈련, 20%는 테스트 데이터  
data.train <- data[parts$Resample1,]  
data.test <- data[-parts$Resample1,]  
m.rpart <- rpart(Class ~., data=data.train)  
confusionMatrix(data.test$Class, predict(m.rpart, newdat=data.test, type="class"))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction benign malignant  
## benign 86 5  
## malignant 5 43  
##   
## Accuracy : 0.9281   
## 95% CI : (0.8717, 0.965)  
## No Information Rate : 0.6547   
## P-Value [Acc > NIR] : 2.775e-14   
##   
## Kappa : 0.8409   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.9451   
## Specificity : 0.8958   
## Pos Pred Value : 0.9451   
## Neg Pred Value : 0.8958   
## Prevalence : 0.6547   
## Detection Rate : 0.6187   
## Detection Prevalence : 0.6547   
## Balanced Accuracy : 0.9204   
##   
## 'Positive' Class : benign   
##

data2 <- upSample(subset(data.train, select= -Class),data.train$Class)  
m.rpart2 <- rpart(Class ~., data=data2)  
confusionMatrix(data.test$Class, predict(m.rpart2, newdat=data.test, type="class"))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction benign malignant  
## benign 86 5  
## malignant 2 46  
##   
## Accuracy : 0.9496   
## 95% CI : (0.899, 0.9795)  
## No Information Rate : 0.6331   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.8902   
## Mcnemar's Test P-Value : 0.4497   
##   
## Sensitivity : 0.9773   
## Specificity : 0.9020   
## Pos Pred Value : 0.9451   
## Neg Pred Value : 0.9583   
## Prevalence : 0.6331   
## Detection Rate : 0.6187   
## Detection Prevalence : 0.6547   
## Balanced Accuracy : 0.9396   
##   
## 'Positive' Class : benign   
##

# SMOTE함수  
# 인접값들을 찾아 추가하는 방법으로, 비율이 낮은 분류의 데이터를 추가로 생성하거나 높은 쪽ㅇ 데이터를 적게 샘플링 해준다.  
  
if(!require(DMwR)) install.packages("DMwR", repos = "http://cran.us.r-project.org"); library(DMwR)

## Loading required package: DMwR

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,  
## logical.return = TRUE, : there is no package called 'DMwR'

## Installing package into 'C:/Users/SANGHOOJEFFREY/Documents/R/win-library/3.4'  
## (as 'lib' is unspecified)

## package 'DMwR' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\SANGHOOJEFFREY\AppData\Local\Temp\RtmpIHijIW\downloaded\_packages

## Warning: package 'DMwR' was built under R version 3.4.4

## Loading required package: grid

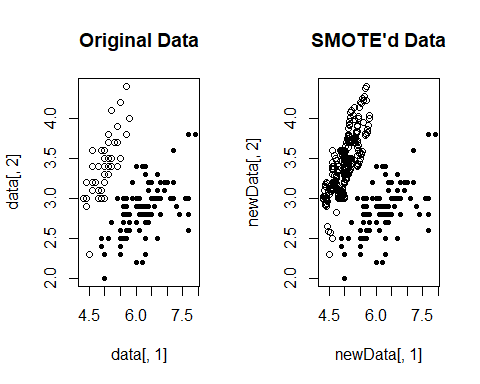
## A small example with a data set created artificially from the IRIS  
## data   
data(iris)  
data <- iris[, c(1, 2, 5)]  
data$Species <- factor(ifelse(data$Species == "setosa","rare","common"))   
## checking the class distribution of this artificial data set  
table(data$Species)

##   
## common rare   
## 100 50

## now using SMOTE to create a more "balanced problem"  
newData <- SMOTE(Species ~ ., data, perc.over = 600,perc.under=100)  
table(newData$Species)

##   
## common rare   
## 300 350

## Checking visually the created data  
## Not run:   
par(mfrow = c(1, 2))  
plot(data[, 1], data[, 2], pch = 19 + as.integer(data[, 3]),  
 main = "Original Data")  
plot(newData[, 1], newData[, 2], pch = 19 + as.integer(newData[,3]),  
 main = "SMOTE'd Data")



## End(Not run)  
  
## Now an example where we obtain a model with the "balanced" data  
classTree <- SMOTE(Species ~ ., data, perc.over = 600,perc.under=100,  
 learner='rpartXse',se=0.5)  
## check the resulting classification tree  
classTree

## n= 650   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 650 300 rare (0.46153846 0.53846154)   
## 2) Sepal.Length>=5.499929 305 24 common (0.92131148 0.07868852)   
## 4) Sepal.Width< 3.819972 286 5 common (0.98251748 0.01748252)   
## 8) Sepal.Width< 3.45 271 0 common (1.00000000 0.00000000) \*  
## 9) Sepal.Width>=3.45 15 5 common (0.66666667 0.33333333)   
## 18) Sepal.Length>=6.45 10 0 common (1.00000000 0.00000000) \*  
## 19) Sepal.Length< 6.45 5 0 rare (0.00000000 1.00000000) \*  
## 5) Sepal.Width>=3.819972 19 0 rare (0.00000000 1.00000000) \*  
## 3) Sepal.Length< 5.499929 345 19 rare (0.05507246 0.94492754)   
## 6) Sepal.Width< 2.759635 21 4 common (0.80952381 0.19047619)   
## 12) Sepal.Length>=4.7 17 0 common (1.00000000 0.00000000) \*  
## 13) Sepal.Length< 4.7 4 0 rare (0.00000000 1.00000000) \*  
## 7) Sepal.Width>=2.759635 324 2 rare (0.00617284 0.99382716) \*

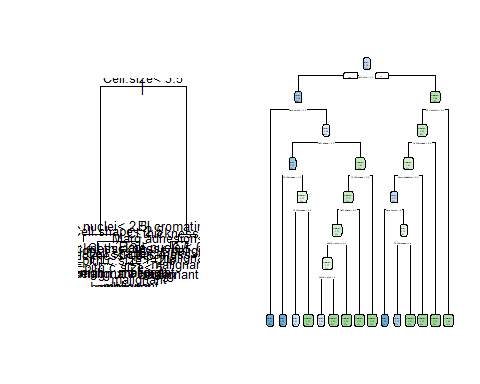
## The tree with the unbalanced data set would be  
rpartXse(Species ~ .,data,se=0.5)

## n= 150   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 150 50 common (0.66666667 0.33333333)   
## 2) Sepal.Length>=5.45 98 5 common (0.94897959 0.05102041)   
## 4) Sepal.Width< 3.45 90 0 common (1.00000000 0.00000000) \*  
## 5) Sepal.Width>=3.45 8 3 rare (0.37500000 0.62500000)   
## 10) Sepal.Length>=6.5 3 0 common (1.00000000 0.00000000) \*  
## 11) Sepal.Length< 6.5 5 0 rare (0.00000000 1.00000000) \*  
## 3) Sepal.Length< 5.45 52 7 rare (0.13461538 0.86538462)   
## 6) Sepal.Width< 2.8 7 1 common (0.85714286 0.14285714) \*  
## 7) Sepal.Width>=2.8 45 1 rare (0.02222222 0.97777778) \*

######################################################################################  
if(!require(rpart)) install.packages("rpart");library(rpart)  
if(!require(rpart.plot)) install.packages("rpart.plot");library(rpart.plot)

## Loading required package: rpart.plot

set.seed(42)  
index <- createDataPartition(BreastCancer$Class, p = 0.7, list = FALSE)  
  
train\_data <- BreastCancer[index, -1]  
test\_data <- BreastCancer[-index, -1]  
  
train\_data[,1:9] <- apply(train\_data[, 1:9], 2, function(x) as.numeric(as.character(x)))  
test\_data[,1:9] <- apply(test\_data[, 1:9], 2, function(x) as.numeric(as.character(x)))  
  
  
set.seed(42)  
fit <- rpart(Class ~ .,  
 data = train\_data,  
 method = "class",  
 control = rpart.control(xval = 10,   
 minbucket = 2,   
 cp = 0),   
 parms = list(split = "information"))  
  
plot(fit)  
text(fit, cex=0.8)  
  
rpart.plot(fit)



# logistic regression으로 세운다면 그리고 그 결과  
# randomforest 로 세운다면  
# SVM으로 세운다면  
  
# UpSample된 자료로 rpart, randomforest을 돌려보자.