adaboost

Nisha Iyer

April 2, 2016

We will now use the iris data set to demonstrate the AdaBoost ensemble algorithm.

library(adabag)

## Loading required package: rpart  
## Loading required package: mlbench  
## Loading required package: caret  
## Loading required package: lattice  
## Loading required package: ggplot2

library(randomForest)

## randomForest 4.6-12  
## Type rfNews() to see new features/changes/bug fixes.  
##   
## Attaching package: 'randomForest'  
##   
## The following object is masked from 'package:ggplot2':  
##   
## margin

data(iris)  
##adaBoost  
data(iris)  
head(iris)

## Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
## 1 5.1 3.5 1.4 0.2 setosa  
## 2 4.9 3.0 1.4 0.2 setosa  
## 3 4.7 3.2 1.3 0.2 setosa  
## 4 4.6 3.1 1.5 0.2 setosa  
## 5 5.0 3.6 1.4 0.2 setosa  
## 6 5.4 3.9 1.7 0.4 setosa

set.seed(123)  
split <- createDataPartition(y=iris$Species, p = 0.7, list=FALSE)  
train <- iris[split,]  
test<- iris[-split,]  
train$Species <- factor(train$Species)  
adaboost<-boosting(Species ~ . , data=train, boos=TRUE, mfinal=20, coeflearn='Breiman')  
summary(adaboost)

## Length Class Mode   
## formula 3 formula call   
## trees 20 -none- list   
## weights 20 -none- numeric   
## votes 315 -none- numeric   
## prob 315 -none- numeric   
## class 105 -none- character  
## importance 4 -none- numeric   
## terms 3 terms call   
## call 6 -none- call

# Above we use the Adaboost algorithm. M-final is the number of times the boosting algorithm is run. Breiman  
# as the 'coeflearn' suggests that we are using the M1 algorithm proposed by Breiman. The M1 algorithm  
# is a classification algorithm where each class can attain a weight of no more than 1/2.  
adaboost$importance

## Petal.Length Petal.Width Sepal.Length Sepal.Width   
## 67.471117 26.002360 6.526522 0.000000

#This gives importance of the variables.  
errorevol(adaboost,test)

## $error  
## [1] 0.06666667 0.06666667 0.06666667 0.04444444 0.06666667 0.06666667  
## [7] 0.06666667 0.06666667 0.06666667 0.06666667 0.06666667 0.06666667  
## [13] 0.06666667 0.06666667 0.06666667 0.06666667 0.06666667 0.06666667  
## [19] 0.06666667 0.06666667  
##   
## attr(,"class")  
## [1] "errorevol"

#This gives the error of the variables  
predictions <- predict(adaboost,test)  
predictions$confusion

## Observed Class  
## Predicted Class setosa versicolor virginica  
## setosa 15 0 0  
## versicolor 0 13 1  
## virginica 0 2 14

predictions$error

## [1] 0.06666667

#Let's compare AdaBoost to RandomForest. To do this, I will quickly run randomforest on the iris dataset.  
rf\_iris <- randomForest(Species ~ ., data = train)  
rf\_iris

##   
## Call:  
## randomForest(formula = Species ~ ., data = train)   
## Type of random forest: classification  
## Number of trees: 500  
## No. of variables tried at each split: 2  
##   
## OOB estimate of error rate: 3.81%  
## Confusion matrix:  
## setosa versicolor virginica class.error  
## setosa 35 0 0 0.00000000  
## versicolor 0 34 1 0.02857143  
## virginica 0 3 32 0.08571429

yhat\_iris <- predict(rf\_iris, test)  
#Random forest Confusion Matrix:  
table(yhat\_iris, test$Species)

##   
## yhat\_iris setosa versicolor virginica  
## setosa 15 0 0  
## versicolor 0 13 1  
## virginica 0 2 14

#AdaBoost Confusion Matrix:  
predictions$confusion

## Observed Class  
## Predicted Class setosa versicolor virginica  
## setosa 15 0 0  
## versicolor 0 13 1  
## virginica 0 2 14