Final Project

Cameron Marsden

2023-03-20

#load the mlbench package which has the BreastCancer data set  
require(mlbench)

## Loading required package: mlbench

# if you don't have any required package, use the install.packages() command  
# load the data set  
data(BreastCancer)  
# some algorithms don't like missing values, so remove rows with missing values  
BreastCancer <- na.omit(BreastCancer)   
# remove the unique identifier, which is useless and would confuse the machine learning algorithms  
BreastCancer$Id <- NULL   
# partition the data set for 80% training and 20% evaluation (adapted from ?randomForest)  
set.seed(2)  
  
ind <- sample(2, nrow(BreastCancer), replace = TRUE, prob=c(0.8, 0.2))  
  
test.rows <- nrow(BreastCancer[ind==2,])  
results <- data.frame(rp = rep(0, test.rows), ct = rep(0, test.rows), cf = rep(0, test.rows), ip = rep(0, test.rows), svm = rep(0, test.rows), vote = rep(0, test.rows))  
  
# create model using recursive partitioning on the training data set  
library(rpart)  
x.rp <- rpart(Class ~ ., data=BreastCancer[ind == 1,])  
# predict classes for the evaluation data set  
x.rp.pred <- predict(x.rp, type="class", newdata=BreastCancer[ind == 2,])  
# score the evaluation data set (extract the probabilities)  
x.rp.prob <- predict(x.rp, type="prob", newdata=BreastCancer[ind == 2,])  
  
# add results to results df  
results$rp <- ifelse(x.rp.pred == "malignant", 1, 0)  
  
# To view the decision tree, uncomment this line.  
#plot(x.rp, main="Decision tree created using rpart")  
  
# create model using conditional inference trees  
library(party)

## Loading required package: grid

## Loading required package: mvtnorm

## Loading required package: modeltools

## Loading required package: stats4

## Loading required package: strucchange

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

## Loading required package: sandwich

x.ct <- ctree(Class ~ ., data=BreastCancer[ind == 1,])  
x.ct.pred <- predict(x.ct, newdata=BreastCancer[ind == 2,])  
x.ct.prob <- 1- unlist(treeresponse(x.ct, BreastCancer[ind == 2,]), use.names=F)[seq(1,nrow(BreastCancer[ind == 2,])\*2,2)]  
  
# add results to results df  
results$ct <- ifelse(x.ct.pred == "malignant", 1, 0)  
  
# To view the decision tree, uncomment this line.  
# plot(x.ct, main="Decision tree created using condition inference trees")  
  
# create model using random forest and bagging ensemble using conditional inference trees  
x.cf <- cforest(Class ~ ., data=BreastCancer[ind == 1,], control = cforest\_unbiased(mtry = ncol(BreastCancer)-2))  
x.cf.pred <- predict(x.cf, newdata=BreastCancer[ind == 2,])  
x.cf.prob <- 1- unlist(treeresponse(x.cf, BreastCancer[ind == 2,]), use.names=F)[seq(1,nrow(BreastCancer[ind == 2,])\*2,2)]  
  
# add results to results df  
results$cf <- ifelse(x.cf.pred == "malignant", 1, 0)  
  
# create model using bagging (bootstrap aggregating)  
library(ipred)  
x.ip <- bagging(Class ~ ., data=BreastCancer[ind == 1,])  
x.ip.prob <- predict(x.ip, type="prob", newdata=BreastCancer[ind == 2,])  
  
# add results to results df  
results$ip <- ifelse(as.data.frame(x.ip.prob)$benign < as.data.frame(x.ip.prob)$malignant, 1, 0)  
  
# create model using svm (support vector machine)  
library(e1071)  
  
# svm requires tuning  
x.svm.tune <- tune(svm, Class~., data = BreastCancer[ind == 1,],  
 ranges = list(gamma = 2^(-8:1), cost = 2^(0:4)),  
 tunecontrol = tune.control(sampling = "fix"))  
# display the tuning results (in text format)  
x.svm.tune

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: fixed training/validation set   
##   
## - best parameters:  
## gamma cost  
## 0.0625 1  
##   
## - best performance: 0.02234637

# If the tuning results are on the margin of the parameters (e.g., gamma = 2^-8),   
# then widen the parameters.  
# I manually copied the cost and gamma from console messages above to parameters below.  
x.svm <- svm(Class~., data = BreastCancer[ind == 1,], cost=4, gamma=0.0625, probability = TRUE)  
x.svm.prob <- predict(x.svm, type="prob", newdata=BreastCancer[ind == 2,], probability = TRUE)  
  
  
# add results to results df  
results$svm <- ifelse(as.data.frame(attr(x.svm.prob, "probabilities"))$benign < as.data.frame(attr(x.svm.prob, "probabilities"))$malignant, 1, 0)  
  
  
  
## ENSEMBLE CREATION  
# Mode function will find the mode of a row  
Mode <- function(x) {  
 ux <- unique(x)  
 ux[which.max(tabulate(match(x, ux)))]  
}  
  
# Find the most common number in each row  
for (i in 1:nrow(results)){  
 results[i, "vote"] <- Mode(results[i,])  
}  
  
# Convert vote into factor  
results$vote <- as.factor(ifelse(results$vote == 1, "malignant", "benign"))  
  
# Here is the accuracy of the ensemble. 1 means "malignant"  
library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

confusionMatrix(results$vote, BreastCancer[ind == 2, "Class"], positive = "malignant")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction benign malignant  
## benign 86 2  
## malignant 6 54  
##   
## Accuracy : 0.9459   
## 95% CI : (0.8963, 0.9764)  
## No Information Rate : 0.6216   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.8867   
##   
## Mcnemar's Test P-Value : 0.2888   
##   
## Sensitivity : 0.9643   
## Specificity : 0.9348   
## Pos Pred Value : 0.9000   
## Neg Pred Value : 0.9773   
## Prevalence : 0.3784   
## Detection Rate : 0.3649   
## Detection Prevalence : 0.4054   
## Balanced Accuracy : 0.9495   
##   
## 'Positive' Class : malignant   
##

##  
## plot ROC curves to compare the performance of the individual classifiers  
##  
  
# Output the plot to a PNG file for display on web. To draw to the screen,   
# comment this line out.  
png(filename="roc\_curve\_5\_models.png", width=700, height=700)  
  
# load the ROCR package which draws the ROC curves  
require(ROCR)

## Loading required package: ROCR

# create an ROCR prediction object from rpart() probabilities  
x.rp.prob.rocr <- prediction(x.rp.prob[,2], BreastCancer[ind == 2,'Class'])  
# prepare an ROCR performance object for ROC curve (tpr=true positive rate, fpr=false positive rate)  
x.rp.perf <- performance(x.rp.prob.rocr, "tpr","fpr")  
# plot it  
plot(x.rp.perf, col=2, main="ROC curves comparing classification performance of five machine learning models")  
  
# Draw a legend.  
legend(0.6, 0.6, c('rpart', 'ctree', 'cforest','bagging','svm'), 2:6)  
  
# ctree  
x.ct.prob.rocr <- prediction(x.ct.prob, BreastCancer[ind == 2,'Class'])  
x.ct.perf <- performance(x.ct.prob.rocr, "tpr","fpr")  
# add=TRUE draws on the existing chart   
plot(x.ct.perf, col=3, add=TRUE)  
  
  
# cforest  
x.cf.prob.rocr <- prediction(x.cf.prob, BreastCancer[ind == 2,'Class'])  
x.cf.perf <- performance(x.cf.prob.rocr, "tpr","fpr")  
plot(x.cf.perf, col=4, add=TRUE)  
  
# bagging  
x.ip.prob.rocr <- prediction(x.ip.prob[,2], BreastCancer[ind == 2,'Class'])  
x.ip.perf <- performance(x.ip.prob.rocr, "tpr","fpr")  
plot(x.ip.perf, col=5, add=TRUE)  
# svm  
x.svm.prob.rocr <- prediction(attr(x.svm.prob, "probabilities")[,2], BreastCancer[ind == 2,'Class'])  
x.svm.perf <- performance(x.svm.prob.rocr, "tpr","fpr")  
  
plot(x.svm.perf, col=6, add=TRUE)  
  
# Close and save the PNG file.  
dev.off()

## png   
## 2