560 Data Mining Course Project 2

Jing Zhang

2/23/2021

require(mlbench)  
library(tidyverse)

BreastCancer has 699 observations on 11 variables, one being a character variable, 9 being ordered or nominal, and 1 target class.

### Data preparation

Load Data

# 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 <- BreastCancer %>% dplyr::select(-Id) #why need to specify the package name to work?it's due to the package overlapping, MASS has select fun.   
  
summary(BreastCancer)

## Cl.thickness Cell.size Cell.shape Marg.adhesion Epith.c.size  
## 1 :139 1 :373 1 :346 1 :393 2 :376   
## 5 :128 10 : 67 2 : 58 2 : 58 3 : 71   
## 3 :104 3 : 52 10 : 58 3 : 58 4 : 48   
## 4 : 79 2 : 45 3 : 53 10 : 55 1 : 44   
## 10 : 69 4 : 38 4 : 43 4 : 33 6 : 40   
## 2 : 50 5 : 30 5 : 32 8 : 25 5 : 39   
## (Other):114 (Other): 78 (Other): 93 (Other): 61 (Other): 65   
## Bare.nuclei Bl.cromatin Normal.nucleoli Mitoses Class   
## 1 :402 3 :161 1 :432 1 :563 benign :444   
## 10 :132 2 :160 10 : 60 2 : 35 malignant:239   
## 2 : 30 1 :150 3 : 42 3 : 33   
## 5 : 30 7 : 71 2 : 36 10 : 14   
## 3 : 28 4 : 39 8 : 23 4 : 12   
## 8 : 21 5 : 34 6 : 22 7 : 9   
## (Other): 40 (Other): 68 (Other): 68 (Other): 17

#look at the class   
#split(names(BreastCancer), sapply(BreastCancer, function(x)(class(x))))

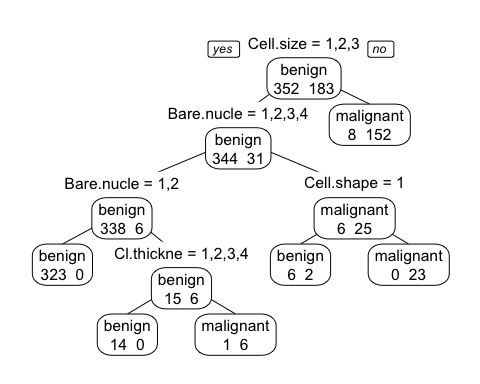
Split data

# partition the data set for 80% training and 20% evaluation  
set.seed(2)  
ind <- sample(2, nrow(BreastCancer), replace = TRUE, prob=c(0.8, 0.2))  
  
train <- BreastCancer[ind==1,]  
valid <- BreastCancer[ind==2,]

### Create multiple models using different classifiers/algorithms

1. decision tree

library(rpart)  
library(rpart.plot)  
x.rp <- rpart(Class ~ ., data=train)  
#plot(x.rp, main="Decision tree created using rpart") #?????   
prp(x.rp, type = 1, extra = 1, split.font = 1, varlen = -10)



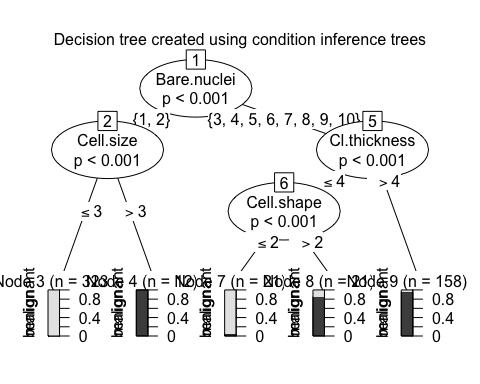
#prediction  
# predict classes for the evaluation data set  
x.rp.pred <- predict(x.rp, type="class", newdata=valid) # to ensemble  
# score the evaluation data set (extract the probabilities)  
x.rp.prob <- predict(x.rp, type="prob", newdata=valid)  
table(x.rp.pred,valid$Class)

##   
## x.rp.pred benign malignant  
## benign 86 2  
## malignant 6 54

# Leave-1-Out Cross Validation (LOOCV)  
#ans <- numeric(nrow(BreastCancer))  
#for (i in 1:nrow(BreastCancer)) {  
# mytree <- rpart(Class ~ ., BreastCancer[-i,])  
# mytree.pred <- predict(mytree,BreastCancer[i,],type="class")  
# ans[i] <- mytree.pred  
#}  
#ans <- factor(ans,labels=levels(BreastCancer$Class))  
#table(ans,BreastCancer$Class)  
#ans benign malignant  
# benign 430 20  
# malignant 14 219

1. conditional inference trees

require(party)  
x.ct <- ctree(Class ~ ., data=train)  
plot(x.ct, main="Decision tree created using condition inference trees")



x.ct.pred <- predict(x.ct, newdata=valid) #ensemble  
x.ct.prob <- 1- unlist(treeresponse(x.ct, valid), use.names=F)[seq(1,nrow(valid)\*2,2)]  
table(x.ct.pred,valid$Class)

##   
## x.ct.pred benign malignant  
## benign 86 2  
## malignant 6 54

1. random forest : an implementation of the random forest and bagging ensemble algorithms utilizing conditional inference trees as base learners.

x.cf <- cforest(Class ~ ., data=train, control = cforest\_unbiased(mtry = 9)) #?cforest\_unbiased, bagging? #500 trees  
  
x.cf.pred <- predict(x.cf, newdata=valid)  
x.cf.prob <- 1- unlist(treeresponse(x.cf, valid), use.names=F)[seq(1,nrow(valid)\*2,2)]  
table(x.cf.pred,valid$Class)

##   
## x.cf.pred benign malignant  
## benign 86 2  
## malignant 6 54

1. bagging (bootstrap aggregating)

# create model using bagging (bootstrap aggregating)  
require(ipred)  
x.ip <- bagging(Class ~ ., data=train) #Bagging classification trees with 25 bootstrap replications   
  
x.ip.pred <- predict(x.ip, newdata=valid)  
x.ip.prob <- predict(x.ip, type="prob", newdata=valid)  
table(x.ip.pred,valid$Class)

##   
## x.ip.pred benign malignant  
## benign 85 3  
## malignant 7 53

1. svm

require(e1071)  
# svm requires tuning  
x.svm.tune <- tune(svm, Class~., data = train,  
 ranges = list(gamma = 2^(-8:1), cost = 2^(0:4)),  
 tunecontrol = tune.control(sampling = "fix")) #why use these number to intialize ranges?  
# display the tuning results (in text format)  
x.svm.tune #note the gamma and cost

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: fixed training/validation set   
##   
## - best parameters:  
## gamma cost  
## 0.125 1  
##   
## - best performance: 0.005586592

# 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 = train, cost=1, gamma=0.03125, probability = TRUE) #  
  
x.svm.pred <- predict(x.svm, type="class", newdata=valid) #ensemble; only give the class  
x.svm.prob <- predict(x.svm, type="prob", newdata=valid, probability = TRUE) # has to include probability = TRUE while type="prob" is not needed  
#t <- attr(x.svm.prob, "probabilities") # only give the probabilities  
table(x.svm.pred,valid$Class)

##   
## x.svm.pred benign malignant  
## benign 87 1  
## malignant 5 55

1. naive bayes

library(klaR)  
x.nb <- NaiveBayes(Class~., data = train)  
x.nb.pred <- predict(x.nb,valid)$class #ensemble  
x.nb.prob <- predict(x.nb,valid)$posterior  
table(x.nb.pred,valid$Class)

##   
## x.nb.pred benign malignant  
## benign 87 0  
## malignant 5 56

1. neural network

library(nnet)  
x.nn <- nnet(Class~., data = train,size=1) #size? everytime, result changes, hidden layer, nodes?

## # weights: 83  
## initial value 365.397345   
## iter 10 value 51.028720  
## iter 20 value 21.484525  
## iter 30 value 7.922738  
## iter 40 value 6.258445  
## iter 50 value 6.215714  
## iter 60 value 6.213101  
## iter 70 value 6.212471  
## iter 80 value 6.212291  
## iter 90 value 6.212248  
## iter 100 value 6.212223  
## final value 6.212223   
## stopped after 100 iterations

x.nn.pred <- predict(x.nn,valid,type="class")  
x.nn.prob <- predict(x.nn,valid,type="raw") #is this the probability of "malignant"? yes?  
table(x.nn.pred,valid$Class)

##   
## x.nn.pred benign malignant  
## benign 85 3  
## malignant 7 53

1. Quadratic Discriminant Analysis

library(MASS)  
library(dplyr)  
train.num <- train %>% dplyr::select(-Class) %>% mutate\_if(is.factor,as.character)%>% mutate\_if(is.character,as.numeric)  
train.num$Class <- train$Class  
valid.num <- valid%>%dplyr::select(-Class) %>% mutate\_if(is.factor,as.character)%>% mutate\_if(is.character,as.numeric)  
valid.num$Class <- valid$Class  
  
x.qda <- qda(Class~., data = train.num) #qda, formula, right hand is non-factor  
x.qda.pred <- predict(x.qda, valid.num)$class  
x.qda.prob <- predict(x.qda, valid.num)$posterior #is posterior?   
table(x.qda.pred,valid.num$Class)

##   
## x.qda.pred benign malignant  
## benign 86 1  
## malignant 6 55

1. Regularised Discriminant Analysis

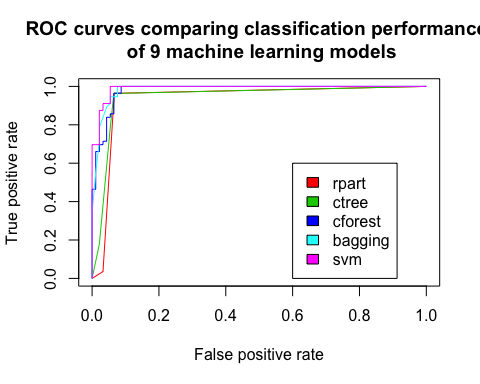
library(klaR)  
x.rda <- rda(Class~., data = train)  
x.rda.pred <- predict(x.rda, valid)$class  
x.rda.prob <- predict(x.rda, valid)$posterior  
table(x.rda.pred,valid$Class)

##   
## x.rda.pred benign malignant  
## benign 86 0  
## malignant 6 56

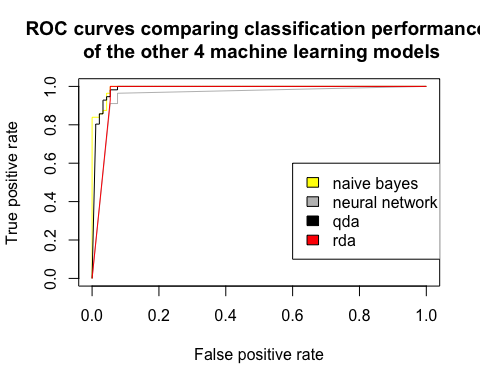
### Plot ROC curves to compare the performance of the individual classifiers.

#load the ROCR package which draws the ROC curves  
require(ROCR)  
  
# 1.rptree  
# create an ROCR prediction object from rpart() probabilities  
x.rp.prob.rocr <- prediction(x.rp.prob[,2], BreastCancer[ind == 2,'Class'])  
#x.rp.prob.rocr #Slot "n.pos": check and see if it identifies the positive class; compare with summary(valid$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")  
  
# 2.ctree  
x.ct.prob.rocr <- prediction(x.ct.prob, BreastCancer[ind == 2,'Class'])  
x.ct.perf <- performance(x.ct.prob.rocr, "tpr","fpr")  
  
# 3.cforest  
x.cf.prob.rocr <- prediction(x.cf.prob, BreastCancer[ind == 2,'Class'])  
x.cf.perf <- performance(x.cf.prob.rocr, "tpr","fpr")  
  
# 4.bagging  
x.ip.prob.rocr <- prediction(x.ip.prob[,2], BreastCancer[ind == 2,'Class'])  
x.ip.perf <- performance(x.ip.prob.rocr, "tpr","fpr")  
  
# 5.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")  
  
# 6.nb ### calculate the prob   
x.nb.prob.rocr <- prediction(x.nb.prob[,2], BreastCancer[ind == 2,'Class'])  
x.nb.perf <- performance(x.nb.prob.rocr, "tpr","fpr")  
  
# 7.nn  
x.nn.prob.rocr <- prediction(x.nn.prob, BreastCancer[ind == 2,'Class'])  
x.nn.perf <- performance(x.nn.prob.rocr, "tpr","fpr")  
  
# 8.qda  
x.qda.prob.rocr <- prediction(x.qda.prob[,2], BreastCancer[ind == 2,'Class'])  
x.qda.perf <- performance(x.qda.prob.rocr, "tpr","fpr")  
  
# 9.rda  
x.rda.prob.rocr <- prediction(x.rda.prob[,2], BreastCancer[ind == 2,'Class'])  
x.rda.perf <- performance(x.rda.prob.rocr, "tpr","fpr")

####### plot  
# Output the plot to a PNG file for display on web. To draw to the screen,   
# comment this line out.  
#png(filename="roc\_curve\_models1.png", width=700, height=700)  
  
#par(mfrow=c(1,2))  
plot(x.rp.perf, col=2, main="ROC curves comparing classification performance \n of 9 machine learning models") #   
legend(0.6, 0.6, c('rpart', 'ctree', 'cforest','bagging','svm'), 2:6)# Draw a legend.  
plot(x.ct.perf, col=3, add=TRUE)# add=TRUE draws on the existing chart #has to be run together.  
plot(x.cf.perf, col=4, add=TRUE)  
plot(x.ip.perf, col=5, add=TRUE)  
plot(x.svm.perf, col=6, add=TRUE)



# Close and save the PNG file.  
#dev.off()  
  
#png(filename="roc\_curve\_models2.png", width=700, height=700)  
plot(x.nb.perf, col=7, main="ROC curves comparing classification performance \n of the other 4 machine learning models")  
legend(0.6, 0.6, c('naive bayes', 'neural network', 'qda','rda'), 7:10)  
plot(x.nn.perf, col=8, add=TRUE)  
plot(x.qda.perf, col=9, add=TRUE)  
plot(x.rda.perf, col=10, add=TRUE)



#dev.off()

### Ensemble: combine all the nine classifiers and generate the final prediction based on the majority rule.

classifier <- data.frame(cbind(x.rp.pred, x.ct.pred, x.cf.pred, x.ip.pred, x.svm.pred, x.nb.pred,x.nn.pred,x.qda.pred,x.rda.pred))  
  
names(classifier) <-c('recursive.tree','conditional.inference.tree','random.forest','bootstrap','svm','naive.bayes','neutral.network','qda','rda')  
levels(classifier$neutral.network) =c('1','2')  
  
classifier <-classifier%>% sapply(FUN = function(x)(ifelse(x=='1',0,1)))  
classifier<- addmargins(classifier, margin = 2) # table/arragy, margin =2 aggregate by col   
classifier <- data.frame(classifier)  
classifier$predition <- ifelse(classifier$Sum >=5, 'malignant','benign')  
  
#confusion matrix   
library(caret)  
confusionMatrix(as.factor(classifier$predition), valid$Class, positive = 'malignant')

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction benign malignant  
## benign 86 1  
## malignant 6 55  
##   
## Accuracy : 0.9527   
## 95% CI : (0.905, 0.9808)  
## No Information Rate : 0.6216   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9012   
##   
## Mcnemar's Test P-Value : 0.1306   
##   
## Sensitivity : 0.9821   
## Specificity : 0.9348   
## Pos Pred Value : 0.9016   
## Neg Pred Value : 0.9885   
## Prevalence : 0.3784   
## Detection Rate : 0.3716   
## Detection Prevalence : 0.4122   
## Balanced Accuracy : 0.9585   
##   
## 'Positive' Class : malignant   
##