Project Based - Toward a Data Mining Portfolio

trieutran

2024-03-13

#install and load packages  
if(!require('mlbench')) {  
 install.packages('mlbench')  
 library('mlbench')  
}

#load data  
data(BreastCancer)  
summary(c)

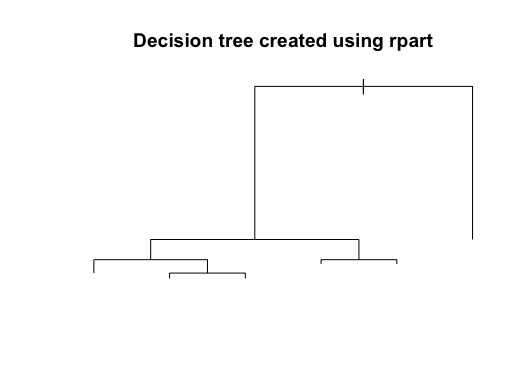
#> Error in object[[i]]: object of type 'builtin' is not subsettable

#remove rows with missing values  
BreastCancer <- na.omit(BreastCancer)  
# remove the unique identifier  
BreastCancer$Id <- NULL

# 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,]

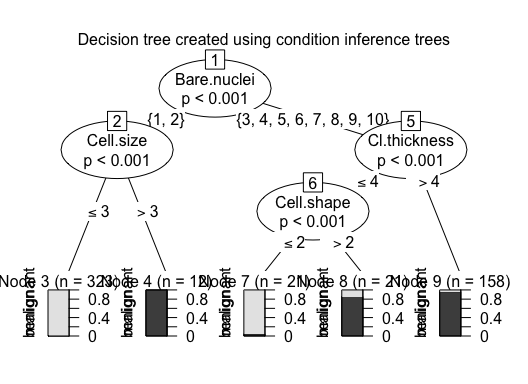
# 1. Decision tree

# create model using recursive partitioning on the training data set  
require(rpart)  
decisionTree <- rpart(Class ~ ., data=train)  
# predict classes for the evaluation data set  
decisionTree.pred <- predict(decisionTree, type="class", newdata=valid)  
# score the evaluation data set (extract the probabilities)  
decisionTree.prob <- predict(decisionTree, type="prob", newdata=valid)  
  
# view the decision tree  
plot(decisionTree, main="Decision tree created using rpart")



# 2. Conditional inference trees

# create model using conditional inference trees  
require(party)  
conditionalTree <- ctree(Class ~ ., data=train)  
conditionalTree.pred <- predict(conditionalTree, newdata=valid)  
conditionalTree.prob <- 1- unlist(treeresponse(conditionalTree, valid), use.names=F)[seq(1,nrow(valid)\*2,2)]  
plot(conditionalTree, main="Decision tree created using condition inference trees")



# 3. Random forest and bagging ensemble using conditional inference trees

# create model using random forest and bagging ensemble using conditional inference trees  
conditionalForest <- cforest(Class ~ ., data=train,   
 control = cforest\_unbiased(mtry = ncol(BreastCancer)-2))  
conditionalForest.pred <- predict(conditionalForest,   
 newdata=valid)  
conditionalForest.prob <- 1- unlist(treeresponse(conditionalForest, valid),  
 use.names=F)[seq(1,nrow(valid)\*2,2)]

# 4. Bagging

# create model using bagging (bootstrap aggregating)  
require(ipred)  
bag <- bagging(Class ~ ., data=train)  
bag.pred <- predict(bag, newdata=valid)  
bag.prob <- predict(bag, type="prob", newdata=valid)

# 5. Support vector machine

# create model using svm (support vector machine)  
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"))  
# 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 = train, cost=4, gamma=0.0625, probability = TRUE)  
x.svm.prob <- predict(x.svm, type="prob", newdata=valid, probability = TRUE)  
svm.pred <- predict(x.svm, newdata=valid)

# 6. Performance measures - Confusion Matrices

library(caret)  
# 1. Decision tree   
confusionMatrix(valid$Clas, decisionTree.pred)

#> Confusion Matrix and Statistics  
#>   
#> Reference  
#> Prediction benign malignant  
#> benign 86 6  
#> malignant 2 54  
#>   
#> Accuracy : 0.9459   
#> 95% CI : (0.8963, 0.9764)  
#> No Information Rate : 0.5946   
#> P-Value [Acc > NIR] : <2e-16   
#>   
#> Kappa : 0.8867   
#>   
#> Mcnemar's Test P-Value : 0.2888   
#>   
#> Sensitivity : 0.9773   
#> Specificity : 0.9000   
#> Pos Pred Value : 0.9348   
#> Neg Pred Value : 0.9643   
#> Prevalence : 0.5946   
#> Detection Rate : 0.5811   
#> Detection Prevalence : 0.6216   
#> Balanced Accuracy : 0.9386   
#>   
#> 'Positive' Class : benign   
#>

# 2. Conditional inference trees  
confusionMatrix(valid$Clas, conditionalTree.pred)

#> Confusion Matrix and Statistics  
#>   
#> Reference  
#> Prediction benign malignant  
#> benign 86 6  
#> malignant 2 54  
#>   
#> Accuracy : 0.9459   
#> 95% CI : (0.8963, 0.9764)  
#> No Information Rate : 0.5946   
#> P-Value [Acc > NIR] : <2e-16   
#>   
#> Kappa : 0.8867   
#>   
#> Mcnemar's Test P-Value : 0.2888   
#>   
#> Sensitivity : 0.9773   
#> Specificity : 0.9000   
#> Pos Pred Value : 0.9348   
#> Neg Pred Value : 0.9643   
#> Prevalence : 0.5946   
#> Detection Rate : 0.5811   
#> Detection Prevalence : 0.6216   
#> Balanced Accuracy : 0.9386   
#>   
#> 'Positive' Class : benign   
#>

# 3. Random forest and bagging ensemble using conditional inference trees  
confusionMatrix(valid$Clas, conditionalForest.pred)

#> Confusion Matrix and Statistics  
#>   
#> Reference  
#> Prediction benign malignant  
#> benign 86 6  
#> malignant 2 54  
#>   
#> Accuracy : 0.9459   
#> 95% CI : (0.8963, 0.9764)  
#> No Information Rate : 0.5946   
#> P-Value [Acc > NIR] : <2e-16   
#>   
#> Kappa : 0.8867   
#>   
#> Mcnemar's Test P-Value : 0.2888   
#>   
#> Sensitivity : 0.9773   
#> Specificity : 0.9000   
#> Pos Pred Value : 0.9348   
#> Neg Pred Value : 0.9643   
#> Prevalence : 0.5946   
#> Detection Rate : 0.5811   
#> Detection Prevalence : 0.6216   
#> Balanced Accuracy : 0.9386   
#>   
#> 'Positive' Class : benign   
#>

# 4. Bagging  
confusionMatrix(valid$Clas, bag.pred)

#> Confusion Matrix and Statistics  
#>   
#> Reference  
#> Prediction benign malignant  
#> benign 86 6  
#> malignant 3 53  
#>   
#> Accuracy : 0.9392   
#> 95% CI : (0.8877, 0.9718)  
#> No Information Rate : 0.6014   
#> P-Value [Acc > NIR] : <2e-16   
#>   
#> Kappa : 0.8721   
#>   
#> Mcnemar's Test P-Value : 0.505   
#>   
#> Sensitivity : 0.9663   
#> Specificity : 0.8983   
#> Pos Pred Value : 0.9348   
#> Neg Pred Value : 0.9464   
#> Prevalence : 0.6014   
#> Detection Rate : 0.5811   
#> Detection Prevalence : 0.6216   
#> Balanced Accuracy : 0.9323   
#>   
#> 'Positive' Class : benign   
#>

# 5. Support vector machine   
confusionMatrix(valid$Clas, svm.pred)

#> Confusion Matrix and Statistics  
#>   
#> Reference  
#> Prediction benign malignant  
#> benign 87 5  
#> malignant 2 54  
#>   
#> Accuracy : 0.9527   
#> 95% CI : (0.905, 0.9808)  
#> No Information Rate : 0.6014   
#> P-Value [Acc > NIR] : <2e-16   
#>   
#> Kappa : 0.9005   
#>   
#> Mcnemar's Test P-Value : 0.4497   
#>   
#> Sensitivity : 0.9775   
#> Specificity : 0.9153   
#> Pos Pred Value : 0.9457   
#> Neg Pred Value : 0.9643   
#> Prevalence : 0.6014   
#> Detection Rate : 0.5878   
#> Detection Prevalence : 0.6216   
#> Balanced Accuracy : 0.9464   
#>   
#> 'Positive' Class : benign   
#>

# 7. ROC curves to compare the performance of the 5 classifiers

# plot ROC curves to compare the performance of the individual classifiers  
png(filename="/Users/trieutlh/Library/CloudStorage/OneDrive-UW/TBANLT 560/extraCredit/figures/roc\_curve\_5\_models.png",  
 width=700, height=500)  
  
# load the ROCR package which draws the ROC curves  
require(ROCR)  
  
# create an ROCR prediction object from rpart() probabilities  
decisionTree.prob.rocr <- prediction(decisionTree.prob[,2], valid$Class)  
# prepare an ROCR performance object for ROC curve (tpr=true positive rate, fpr=false positive rate)  
decisionTree.perf <- performance(decisionTree.prob.rocr, "tpr","fpr")  
# plot it  
plot(decisionTree.perf, col=2, main="ROC curves comparing classification performance of 5 ML models")  
  
# Draw a legend.  
legend(0.6, 0.6, c('rpart', 'ctree', 'cforest','bagging','svm'), 2:6)  
  
# ctree  
conditionalTree.prob.rocr <- prediction(conditionalTree.prob, BreastCancer[ind == 2,'Class'])  
conditionalTree.perf <- performance(conditionalTree.prob.rocr, "tpr","fpr")  
plot(conditionalTree.perf, col=3, add=TRUE)  
  
# cforest  
conditionalForest.prob.rocr <- prediction(conditionalForest.prob, BreastCancer[ind == 2,'Class'])  
conditionalForest.perf <- performance(conditionalForest.prob.rocr, "tpr","fpr")  
plot(conditionalForest.perf, col=4, add=TRUE)  
  
# bagging  
bag.prob.rocr <- prediction(bag.prob[,2], BreastCancer[ind == 2,'Class'])  
bag.perf <- performance(bag.prob.rocr, "tpr","fpr")  
plot(bag.perf, col=5, add=TRUE)  
  
# svm  
svm.prob.rocr <- prediction(attr(x.svm.prob, "probabilities")[,2], BreastCancer[ind == 2,'Class'])  
svm.perf <- performance(svm.prob.rocr, "tpr","fpr")  
plot(svm.perf, col=6, add=TRUE)  
  
# Close and save the PNG file.  
dev.off()

#> quartz\_off\_screen   
#> 2