ProjectTwo

Gauri Patil

3/15/2021

**Load library**

library(rpart)  
library(rpart.plot)  
library(mlbench)  
library(party)

library(caret)

library(klaR)

library(e1071)  
library(nnet)  
library(klaR)  
library(randomForest)

library(varhandle)  
library(ROCR)

**Load Data**

data("BreastCancer")  
summary(BreastCancer)

## Id Cl.thickness Cell.size Cell.shape Marg.adhesion  
## Length:699 1 :145 1 :384 1 :353 1 :407   
## Class :character 5 :130 10 : 67 2 : 59 2 : 58   
## Mode :character 3 :108 3 : 52 10 : 58 3 : 58   
## 4 : 80 2 : 45 3 : 56 10 : 55   
## 10 : 69 4 : 40 4 : 44 4 : 33   
## 2 : 50 5 : 30 5 : 34 8 : 25   
## (Other):117 (Other): 81 (Other): 95 (Other): 63   
## Epith.c.size Bare.nuclei Bl.cromatin Normal.nucleoli Mitoses   
## 2 :386 1 :402 2 :166 1 :443 1 :579   
## 3 : 72 10 :132 3 :165 10 : 61 2 : 35   
## 4 : 48 2 : 30 1 :152 3 : 44 3 : 33   
## 1 : 47 5 : 30 7 : 73 2 : 36 10 : 14   
## 6 : 41 3 : 28 4 : 40 8 : 24 4 : 12   
## 5 : 39 (Other): 61 5 : 34 6 : 22 7 : 9   
## (Other): 66 NA's : 16 (Other): 69 (Other): 69 (Other): 17   
## Class   
## benign :458   
## malignant:241   
##   
##   
##   
##   
##

**Clean Data**

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

**Partition Data**

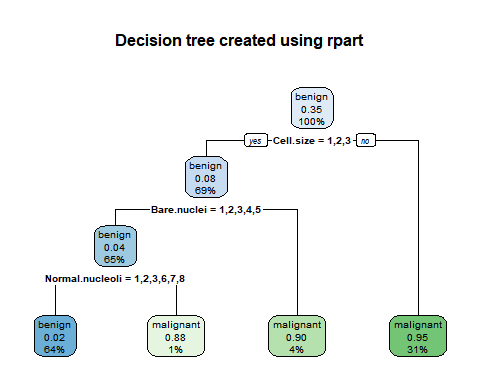
#Create partition  
index <- sample(2, nrow(BreastCancer), replace = TRUE, prob=c(0.8, 0.2))  
training <- BreastCancer[index == 1,]  
testing <- BreastCancer[index == 2,]

**Recursive Partitioning and Regression Tree**

# create model using recursive partitioning on the training data set  
x.rp <- rpart(Class ~ ., data=training, method = "class")  
  
# predict classes for the evaluation data set  
x.rp.pred <- predict(x.rp, type="class", newdata=testing)  
  
# score the evaluation data set (extract the probabilities)  
x.rp.prob <- predict(x.rp, type="prob", newdata=testing)  
  
#confusion matrix  
confusionMatrix(x.rp.pred,testing$Class)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction benign malignant  
## benign 81 0  
## malignant 3 42  
##   
## Accuracy : 0.9762   
## 95% CI : (0.932, 0.9951)  
## No Information Rate : 0.6667   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9474   
##   
## Mcnemar's Test P-Value : 0.2482   
##   
## Sensitivity : 0.9643   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.9333   
## Prevalence : 0.6667   
## Detection Rate : 0.6429   
## Detection Prevalence : 0.6429   
## Balanced Accuracy : 0.9821   
##   
## 'Positive' Class : benign   
##

# decision tree  
rpart.plot(x.rp, main="Decision tree created using rpart")

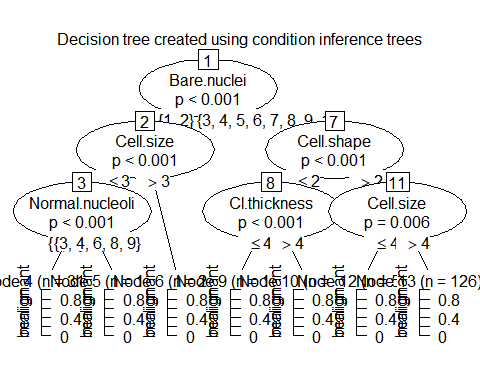


**Conditional Inference Tree**

# create model using conditional inference trees  
x.ct <- ctree(Class ~ ., data=training)  
# predict classes for the evaluation data set  
x.ct.pred <- predict(x.ct, newdata=testing)  
# score the evaluation data set (extract the probabilities)  
x.ct.prob <- 1- unlist(treeresponse(x.ct, testing), use.names=F)[seq(1,nrow(testing)\*2,2)]  
#confusion matrix  
confusionMatrix(x.ct.pred,testing$Class)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction benign malignant  
## benign 81 0  
## malignant 3 42  
##   
## Accuracy : 0.9762   
## 95% CI : (0.932, 0.9951)  
## No Information Rate : 0.6667   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9474   
##   
## Mcnemar's Test P-Value : 0.2482   
##   
## Sensitivity : 0.9643   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.9333   
## Prevalence : 0.6667   
## Detection Rate : 0.6429   
## Detection Prevalence : 0.6429   
## Balanced Accuracy : 0.9821   
##   
## 'Positive' Class : benign   
##

# decision tree  
plot(x.ct, main="Decision tree created using condition inference trees")



Cforest forest

# create model using random forest and bagging ensemble using conditional inference trees  
x.cf <- cforest(Class ~ ., data=training, control = cforest\_unbiased(mtry = ncol(BreastCancer)-2))  
# predict classes for the evaluation data set  
x.cf.pred <- predict(x.cf, newdata=testing)  
# score the evaluation data set (extract the probabilities)  
x.cf.prob <- 1- unlist(treeresponse(x.cf, testing), use.names=F)[seq(1,nrow(testing)\*2,2)]  
#confusion matrix  
confusionMatrix(x.cf.pred,testing$Class)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction benign malignant  
## benign 81 0  
## malignant 3 42  
##   
## Accuracy : 0.9762   
## 95% CI : (0.932, 0.9951)  
## No Information Rate : 0.6667   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9474   
##   
## Mcnemar's Test P-Value : 0.2482   
##   
## Sensitivity : 0.9643   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.9333   
## Prevalence : 0.6667   
## Detection Rate : 0.6429   
## Detection Prevalence : 0.6429   
## Balanced Accuracy : 0.9821   
##   
## 'Positive' Class : benign   
##

**Naive Bayes Model**

# create model using Naive Bayes   
x.nb <- NaiveBayes(Class ~ ., data=training)  
  
# predict classes for the evaluation data set  
x.nb.pred <- predict(x.nb, newdata=testing)  
  
# score the evaluation data set (extract the probabilities)  
x.nb.prob <- predict(x.nb, type="prob", newdata=testing)  
  
#confusion matrix  
confusionMatrix(x.nb.pred$class,testing$Class)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction benign malignant  
## benign 81 0  
## malignant 3 42  
##   
## Accuracy : 0.9762   
## 95% CI : (0.932, 0.9951)  
## No Information Rate : 0.6667   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9474   
##   
## Mcnemar's Test P-Value : 0.2482   
##   
## Sensitivity : 0.9643   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.9333   
## Prevalence : 0.6667   
## Detection Rate : 0.6429   
## Detection Prevalence : 0.6429   
## Balanced Accuracy : 0.9821   
##   
## 'Positive' Class : benign   
##

**Support Vector Machine Classifier**

# create model using svm  
x.svm <- svm(Class ~ ., data=training,cost=4, gamma=0.0625, probability = TRUE)  
  
# predict classes for the evaluation data set  
x.svm.pred <- predict(x.svm, newdata=testing)  
  
#probabilities  
x.svm.prob <- predict(x.svm, type="prob", newdata=testing, probability = TRUE)  
  
#confusion matrix  
confusionMatrix(x.svm.pred,testing$Class)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction benign malignant  
## benign 81 0  
## malignant 3 42  
##   
## Accuracy : 0.9762   
## 95% CI : (0.932, 0.9951)  
## No Information Rate : 0.6667   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9474   
##   
## Mcnemar's Test P-Value : 0.2482   
##   
## Sensitivity : 0.9643   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.9333   
## Prevalence : 0.6667   
## Detection Rate : 0.6429   
## Detection Prevalence : 0.6429   
## Balanced Accuracy : 0.9821   
##   
## 'Positive' Class : benign   
##

**Neural Net Model**

# create model using neural net  
x.nnet <- nnet(Class ~ ., training, size=1)

## # weights: 83  
## initial value 353.802157   
## iter 10 value 43.412573  
## iter 20 value 7.638217  
## iter 30 value 6.944158  
## iter 40 value 6.922788  
## iter 50 value 6.904785  
## iter 60 value 6.897861  
## iter 70 value 6.894372  
## iter 80 value 6.891611  
## iter 90 value 6.889971  
## iter 100 value 6.889097  
## final value 6.889097   
## stopped after 100 iterations

# predict classes for the evaluation data set  
x.nnet.pred <- predict(x.nnet,testing,type="class")  
#confusion matrix  
confusionMatrix(as.factor(x.nnet.pred),testing$Class)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction benign malignant  
## benign 81 2  
## malignant 3 40  
##   
## Accuracy : 0.9603   
## 95% CI : (0.9098, 0.987)  
## No Information Rate : 0.6667   
## P-Value [Acc > NIR] : 5.39e-16   
##   
## Kappa : 0.9112   
##   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.9643   
## Specificity : 0.9524   
## Pos Pred Value : 0.9759   
## Neg Pred Value : 0.9302   
## Prevalence : 0.6667   
## Detection Rate : 0.6429   
## Detection Prevalence : 0.6587   
## Balanced Accuracy : 0.9583   
##   
## 'Positive' Class : benign   
##

**Cross Validation R tree**

# Leave-1-Out Cross Validation (LOOCV)  
x.cv.pred <- numeric(length(testing[,10]))  
x.cv.prob <- numeric(length(testing[,10]))  
  
for (i in 1:length(training[,10])) {  
 x1.tree <- rpart(Class ~ ., training[-i,])  
}  
  
for (i in 1:length(testing[,10])) {  
 x1.pred <- predict(x1.tree,testing[i,],type="class")  
 # score the evaluation data set (extract the probabilities)  
 x1.prob <- predict(x1.tree,testing[i,],type="class")  
 x.cv.pred[i] <- x1.pred  
 x.cv.prob[i] <- x1.prob  
}  
  
x.cv.pred <- factor(x.cv.pred,labels=levels(testing$Class))  
#Confusion matrix  
confusionMatrix(x.cv.pred,testing$Class)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction benign malignant  
## benign 81 0  
## malignant 3 42  
##   
## Accuracy : 0.9762   
## 95% CI : (0.932, 0.9951)  
## No Information Rate : 0.6667   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9474   
##   
## Mcnemar's Test P-Value : 0.2482   
##   
## Sensitivity : 0.9643   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.9333   
## Prevalence : 0.6667   
## Detection Rate : 0.6429   
## Detection Prevalence : 0.6429   
## Balanced Accuracy : 0.9821   
##   
## 'Positive' Class : benign   
##

**QDA**

training1<-training  
for (i in 1:ncol(training1)){  
 training1[,i] <- as.integer(training1[,i])  
}  
  
testing1<-testing  
for (i in 1:ncol(testing1)){  
 testing1[,i] <- as.integer(testing1[,i])  
}  
  
#convert class to binary  
training1$Class<-ifelse(as.integer(training1$Class)==2,1,0)  
testing1$Class<-ifelse(as.integer(testing1$Class)==2,1,0)  
#Run QDA model  
x.qda <- qda(Class ~ ., data=training1)  
#predict using training dataset   
x.qda.pred <- predict(x.qda, testing1)  
#confusion matrix  
confusionMatrix(x.qda.pred$class,as.factor(testing1$Class))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 81 0  
## 1 3 42  
##   
## Accuracy : 0.9762   
## 95% CI : (0.932, 0.9951)  
## No Information Rate : 0.6667   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9474   
##   
## Mcnemar's Test P-Value : 0.2482   
##   
## Sensitivity : 0.9643   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.9333   
## Prevalence : 0.6667   
## Detection Rate : 0.6429   
## Detection Prevalence : 0.6429   
## Balanced Accuracy : 0.9821   
##   
## 'Positive' Class : 0   
##

**Regularised Discriminant Analysis**

#Build model  
x.rda <- rda(Class ~ ., training)  
#Predict using validation data  
x.rda.pred <- predict(x.rda, testing)  
#Confusion matrix  
confusionMatrix(x.rda.pred$class,testing$Class)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction benign malignant  
## benign 81 0  
## malignant 3 42  
##   
## Accuracy : 0.9762   
## 95% CI : (0.932, 0.9951)  
## No Information Rate : 0.6667   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9474   
##   
## Mcnemar's Test P-Value : 0.2482   
##   
## Sensitivity : 0.9643   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.9333   
## Prevalence : 0.6667   
## Detection Rate : 0.6429   
## Detection Prevalence : 0.6429   
## Balanced Accuracy : 0.9821   
##   
## 'Positive' Class : benign   
##

**Random Forests**

#Build model  
x.rf <- randomForest(Class ~ ., training)  
#Predict using validation data  
x.rf.pred <- predict(x.rf, testing)  
#confusion matrix  
confusionMatrix(x.rf.pred, testing$Class)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction benign malignant  
## benign 81 0  
## malignant 3 42  
##   
## Accuracy : 0.9762   
## 95% CI : (0.932, 0.9951)  
## No Information Rate : 0.6667   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9474   
##   
## Mcnemar's Test P-Value : 0.2482   
##   
## Sensitivity : 0.9643   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.9333   
## Prevalence : 0.6667   
## Detection Rate : 0.6429   
## Detection Prevalence : 0.6429   
## Balanced Accuracy : 0.9821   
##   
## 'Positive' Class : benign   
##

**Ensemble output using models**

#Add all prediction in a table  
predall<-data.frame(x.rp.pred,x.ct.pred,x.cf.pred,x.nb.pred$class,x.svm.pred,x.nnet.pred,x.cv.pred,x.rda.pred$class,x.rf.pred)  
#convert to binary  
for (i in 1:ncol(predall)){  
 predall[,i] <- ifelse(predall[,i]=="benign",0,1)  
}  
#unfactor to calcuate rowsums  
predall$x.qda.pred<-unfactor(x.qda.pred$class)  
#calculate rowsums  
predall$sum<-rowSums(predall)  
#Majority Rule   
predall$majority<-ifelse(predall$sum>=6, "malignant", "benign")  
#Confusion matrix   
confusionMatrix(as.factor(predall$majority), testing$Class)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction benign malignant  
## benign 81 0  
## malignant 3 42  
##   
## Accuracy : 0.9762   
## 95% CI : (0.932, 0.9951)  
## No Information Rate : 0.6667   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9474   
##   
## Mcnemar's Test P-Value : 0.2482   
##   
## Sensitivity : 0.9643   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.9333   
## Prevalence : 0.6667   
## Detection Rate : 0.6429   
## Detection Prevalence : 0.6429   
## Balanced Accuracy : 0.9821   
##   
## 'Positive' Class : benign   
##

ROC curve

# create an ROCR prediction object from rpart() probabilities  
x.rp.prob.rocr <- prediction(x.rp.prob[,2], testing$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','naive','svm','LOOCV'), 2:6)  
  
# ctree  
x.ct.prob.rocr <- prediction(x.ct.prob, testing$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, testing$Class)  
x.cf.perf <- performance(x.cf.prob.rocr, "tpr","fpr")  
plot(x.cf.perf, col=4, add=TRUE)  
  
#naive bayes  
# create an ROCR prediction object from nb probabilities  
x.nb.prob.rocr <- prediction(x.nb.prob$posterior[,2], testing$Class)  
x.nb.perf <- performance(x.nb.prob.rocr, "tpr","fpr")  
plot(x.nb.perf, col=5, add=TRUE)  
  
# svm  
x.svm.prob.rocr <- prediction(attr(x.svm.prob, "probabilities")[,2], testing$Class)  
x.svm.perf <- performance(x.svm.prob.rocr, "tpr","fpr")  
plot(x.svm.perf, col=6, add=TRUE)  
  
# cross validation  
x.cv.prob.rocr <- prediction(x.cv.prob, testing$Class)  
x.cv.perf <- performance(x.cv.prob.rocr, "tpr","fpr")  
plot(x.cv.perf, col=7, add=TRUE)

