Data622 - Test1

Abdelmalek Hajjam

11/15/2020

```r
library(caret)
library(ipred)       # for fitting bagged decision trees
#library(bootstrap)

library(e1071)
library(tidyverse)
library(cvAUC)
library(pROC)

#library(bootstrap)
#  reading data
#data <- read.csv("HW1data.csv", header = TRUE)
data <- data.frame(
  X = as.factor(c(5, 5, 5, 5, 5, 5, 19, 19, 19, 19, 19, 19, 35, 35, 35, 35, 35, 35, 51, 51, 51, 51, 51,
  Y = c("a","b","c","d","e","f","a","b","c","d","e","f","a","b","c","d","e","f","a","b","c","d","e","f"
  label = c("BLUE","BLACK","BLUE","BLACK","BLACK","BLACK","BLUE","BLUE","BLUE","BLUE","BLACK","BLUE","BL
)
head(data)
```

```
##   X Y label
## 1 5 a  BLUE
## 2 5 b BLACK
## 3 5 c  BLUE
## 4 5 d BLACK
## 5 5 e BLACK
## 6 5 f BLACK
```

```r
dim(data)
```

```
## [1] 36  3
```

```r
str(data)
```

```
## 'data.frame':    36 obs. of  3 variables:
##  $ X    : Factor w/ 6 levels "5","19","35",..: 1 1 1 1 1 1 2 2 2 2 ...
##  $ Y    : Factor w/ 6 levels "a","b","c","d",..: 1 2 3 4 5 6 1 2 3 4 ...
##  $ label: Factor w/ 2 levels "BLACK","BLUE": 2 1 2 1 1 1 2 2 2 2 ...
```

```r
summary(data)
```

```
##    X       Y        label   
##  5 :6   a:6   BLACK:22  
##  19:6   b:6   BLUE :14  
##  35:6   c:6             
##  51:6   d:6             
##  55:6   e:6             
##  63:6   f:6             
```

```r
#Checking distibution in origanl data
prop.table(table(data$label)) * 100
```

```
## 
##    BLACK     BLUE 
## 61.11111 38.88889
```

Data Preparation

```r
set.seed(123456)
trainidx<-sample(1:nrow(data) , size=round(0.7*nrow(data)),replace=F)
training <- data[trainidx,]
testing <- data[-trainidx,]

summary(training)
```

```
##    X       Y        label   
##  5 :4   a:4   BLACK:15  
##  19:3   b:4   BLUE :10  
##  35:6   c:5             
##  51:3   d:5             
##  55:6   e:3             
##  63:3   f:4             
```

```r
summary(testing)
```

```
##    X       Y        label  
##  5 :2   a:2   BLACK:7  
##  19:3   b:2   BLUE :4  
##  35:0   c:1            
##  51:3   d:1            
##  55:0   e:3            
##  63:3   f:2            
```

```r
#Checking distibution in origanl data
prop.table(table(training$label)) * 100
```

```
## 
## BLACK  BLUE 
##    60    40
```

```r
prop.table(table(testing$label)) * 100
```

```
##
##    BLACK     BLUE
## 63.63636 36.36364
```

(A) Bagging

We refer to the documentation found here: https://cran.r-project.org/web/packages/ipred/ipred.pdf

Training the Model

```r
set.seed(98765)
#do 100 iterations
bgModel <- bagging(label ~ .,
                      data = training,
                      nbagg = 100,
                      coob = TRUE,
                   )
bgModel
```

```
##
## Bagging classification trees with 100 bootstrap replications
##
## Call: bagging.data.frame(formula = label ~ ., data = training, nbagg = 100,
##     coob = TRUE)
##
## Out-of-bag estimate of misclassification error:  0.28
```

Testing the Model

```r
Prediction <- predict(bgModel, testing)
with(testing, table(Prediction, label))
```

```
##           label
## Prediction BLACK BLUE
##      BLACK     5    0
##      BLUE      2    4
```

Model Metrics

```r
library(pROC)
cm <- table(Prediction, testing$label)
```

```r
tn <- cm[1,1]
fn <- cm[1,2]
fp <- cm[2,1]
tp <- cm[2,2]

pred_label <- ifelse(testing$label == 'BLUE', 1, 0)
# Area under the ROC curve (AUC)
auc <- auc(roc(Prediction, pred_label))

#Encapsulate those metrics in a simple procedure that we can call later
getMetrics <- function(tn, fn, fp, tp, auc) {

  tp.bg <- tp / (tp + fn)
  tn.bg <- tn / (tn + fp)
  fn.bg <- 1 - tp.bg
  fp.bg <- 1 - tn.bg
  acc <- (tp + tn) / (tp + tn + fp + fn)

  mytable <- matrix(c(tp.bg,tn.bg,fn.bg,fp.bg, auc, acc),ncol=1, byrow=TRUE)
  colnames(mytable) <- c("Value")
  rownames(mytable) <- c("TP","TN","FN", "FP", "AUC", "ACC")
  mytable
}

#call the above procedure
myMetrics <- getMetrics(tn, fn, fp, tp, auc)
myMetrics
```

```
##         Value
## TP  1.0000000
## TN  0.7142857
## FN  0.0000000
## FP  0.2857143
## AUC 0.8333333
## ACC 0.8181818
```

(B) LOOCV(JackKnife)

This code is from Professor Raman Rmd in Learning Module M11.

```r
N<-nrow(training)
training$label <- ifelse(training$label == 'BLUE', 1, 0)
cv_df  <- do.call('rbind',lapply(1:N,FUN=function(idx,data=training) { ### Iterate Over All Points
   ### Keep One Observation as Test
   m <- naiveBayes(label~., data = data[-idx,])
   ### Train Using the Rest of Observations, predict that one observation
   p <- predict(m, data[idx,-c(3)], type='raw')
   # NB returns the probabilities of the classes,
   # as per Bayesian Classifier, we take the classs with the higher probability
   pc <- unlist(apply(round(p), 1, which.max))-1
```

```r
  list("fold"=idx, "m"=m, "predicted"=pc, "actual" = data[idx,c(3)])
 }
))
```

Training Accuray

```r
cv_df<-as.data.frame(cv_df)
loocv_tbl<-table(as.numeric(cv_df$actual),as.numeric(cv_df$predicted))
(loocv_caret_cfm<-caret::confusionMatrix(loocv_tbl))
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 12  3
##   1  5  5
##
##                Accuracy : 0.68
##                  95% CI : (0.465, 0.8505)
##     No Information Rate : 0.68
##     P-Value [Acc > NIR] : 0.5943
##
##                   Kappa : 0.3103
##
##  Mcnemar's Test P-Value : 0.7237
##
##             Sensitivity : 0.7059
##             Specificity : 0.6250
##          Pos Pred Value : 0.8000
##          Neg Pred Value : 0.5000
##              Prevalence : 0.6800
##          Detection Rate : 0.4800
##    Detection Prevalence : 0.6000
##       Balanced Accuracy : 0.6654
##
##        'Positive' Class : 0
##
```

Tesing The Model

```r
testing$label <- ifelse(testing$label == 'BLUE', 1, 0)
cv_df <- data.frame(cv_df)
df.perf<-as.data.frame(do.call('cbind',lapply(cv_df$m,FUN=function(m,data=testing)
{
   ### Determine Test Metrics
  v <- predict(m,data[,-c(3)],type='raw')
  lbllist <- unlist(apply(round(v), 1, which.max))-1

}
```

```
  )))

### Aggregate Test Metrics
np <- ncol(df.perf)
predclass <- unlist(apply(df.perf,1,FUN=function(v){ ifelse(sum(v[2:length(v)])/np<0.5,0,1)}))
loocvtbl <- table(testing[,3], predclass)
(loocv_cfm<-caret::confusionMatrix(loocvtbl))
```

```
## Confusion Matrix and Statistics
##
##    predclass
##     0 1
##   0 5 2
##   1 0 4
##
##                Accuracy : 0.8182
##                  95% CI : (0.4822, 0.9772)
##     No Information Rate : 0.5455
##     P-Value [Acc > NIR] : 0.0615
##
##                   Kappa : 0.6452
##
##  Mcnemar's Test P-Value : 0.4795
##
##             Sensitivity : 1.0000
##             Specificity : 0.6667
##          Pos Pred Value : 0.7143
##          Neg Pred Value : 1.0000
##              Prevalence : 0.4545
##          Detection Rate : 0.4545
##    Detection Prevalence : 0.6364
##       Balanced Accuracy : 0.8333
##
##        'Positive' Class : 0
##
```

Conclusion

Both bagging and LOOCV performed very well and are both have the same accuracy of 0.81. In my Homework1, my weak learners LR and KNN both had an accuracy of .64 and .73 respectively. Therefore performed poorly comparing to bagging and LOOCV. We were then able to increase the accuracy and obtain a better model using these 2 methods. Naive Bayes on the other hand had a better accuracy .82 in my Homework1, but had an AUC of 0.80 which is less than our AUC for Bagging which was .83. Bagging was then a better Model. Bagging is a model that is less susceptible to overfitting than the individual models we've fit. LOOCV cross validation, on the other hand, is used to estimate the out of sample accuracy.