DATA 622: Test 01

Shovan Biswas

10/15/2020

(A) Run Bagging (ipred package)

– sample with replacement

– estimate metrics for a model

– repeat as many times as specied and report the average

Bagging combines "weak" learners in a way that reduces their variance. In Bagging, a set of bootstrap samples are generated. The goal is to train the model and average them in regression and take the majority vote in classification. This makes Bagging parallelizable.

Load input dataset.csv

```r
mydata <- read.csv('./dataset.csv', head = TRUE, sep = ',', stringsAsFactors = TRUE)
head(mydata)
```

```
##   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
str(mydata)
```

```
## 'data.frame':    36 obs. of  3 variables:
##  $ X    : int  5 5 5 5 5 5 19 19 19 19 ...
##  $ 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 ...
```

Converting variable X to factor.

```r
mydata$X <- factor(mydata$X)
```

Data exploration

```r
print(paste0("Number of observations: ", dim(mydata)[1], "   Number of columns: ", dim(mydata)[2]))
```

```
## [1] "Number of observations: 36   Number of columns: 3"
```

Ratio of BLACK to BLUE in response variable label. The data is somewhat imbalanced.

```r
table(mydata$label)
```

```
##
## BLACK  BLUE
##    22    14
```

```r
summary(mydata)
```

```
##   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
xtabs(~label + X, data = mydata)
```

```
##        X
## label   5 19 35 51 55 63
##   BLACK 4  1  5  5  6  1
##   BLUE  2  5  1  1  0  5
```

```r
xtabs(~label + Y, data = mydata)
```

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

Split dataset

Splitting the dataset into train and test in 70/30 ratio.

```r
set.seed(423)

mydata_train_index <- sample(1:nrow(mydata), 0.30 * nrow(mydata), replace = F)
mydata_train <- mydata[-mydata_train_index, ]
mydata_test <- mydata[mydata_train_index, ]
```

```r
summary(mydata_train)
```

```
##    X      Y       label
##  5 :6   a:6   BLACK:16
##  19:3   b:4   BLUE :10
##  35:5   c:4
##  51:4   d:4
##  55:4   e:3
##  63:4   f:5
```

```r
summary(mydata_test)
```

```
##    X      Y       label
##  5 :0   a:0   BLACK:6
##  19:3   b:2   BLUE :4
##  35:1   c:2
##  51:2   d:2
##  55:2   e:3
##  63:2   f:1
```

Bagging model

```r
start_tm1 <- proc.time()

mydata_train_bag <- bagging(label ~ .,
                       data = mydata_train,
                       nbagg = 100,
                       coob = TRUE)

end_tm1 <- proc.time()

mydata_train_bag
```

```
##
## Bagging classification trees with 100 bootstrap replications
##
## Call: bagging.data.frame(formula = label ~ ., data = mydata_train,
##     nbagg = 100, coob = TRUE)
##
## Out-of-bag estimate of misclassification error:  0.3077
```

```r
diff_bagging <- end_tm1 - start_tm1
diff_bagging
```

```
##    user  system elapsed
##    0.25    0.00    0.25
```

```r
mydata_test_bag_pred <- predict(mydata_train_bag, mydata_test)

mydata_test_bag_pred_cm <- with(mydata_test, table(mydata_test_bag_pred, label))

cat("Confusion Matrix:")
```

```
## Confusion Matrix:
```

```r
cat('\n\n')
```

```r
mydata_test_bag_pred_cm
```

```
##                      label
## mydata_test_bag_pred BLACK BLUE
##                BLACK     5    0
##                BLUE      1    4
```

```r
cat("Original  labels:")
```

```
## Original  labels:
```

```r
mydata_test$label
```

```
##  [1] BLUE  BLACK BLACK BLACK BLACK BLACK BLACK BLUE  BLUE  BLUE
## Levels: BLACK BLUE
```

```r
cat('\n\n')
```

```r
cat("Predicted labels:")
```

```
## Predicted labels:
```

```r
mydata_test_bag_pred
```

```
##  [1] BLUE  BLACK BLACK BLACK BLUE  BLACK BLACK BLUE  BLUE  BLUE
## Levels: BLACK BLUE
```

Now, let's spend a moment on the prediction

We observe (above) that originally there were 6 BLACKs and 4 BLUEs. The columns of the Confusion Matrix (CM) show the actual labels. So, the first column of CM shows 6 (5 + 1) actual BLACKs and the second column shows 4 (4 + 0) actual BLUEs.

But these were predicted differently. Out of the 6 actual BLACKs, 5 were predicted as BLACKs and 1 was predicted as BLUE. So, 5 were predicted as TRUE and 1 was predicted as FALSE. This is clearly shown in the CM.

Out of the 4 actual BLUEs, none (or 0) were predicted as BLACK and 4 were predicted as BLUE. So, none were predicted as FALSE and all 4 were predicted as TRUE. This is clearly supported by the CM.

Here's a quick summary of the predictions in the language of TP, TN, FP, FN. Before, I begin, let me state that I decided to call BLACK the positive. So, BLUE is negative.

(The ideas used in the following, were based on the Confusion Matrix Wiki at https://en.wikipedia.org/wiki/Confusion_matrix)

5 BLACK (P) were predicted as BLACK (P) i.e. 5 positives were predicted as positive. So, it was a True Positive (TP == 5) 1 BLACK (P) was predicted as BLUE (N) i.e. 1 positive was predicted as negative. So, it was a False Negative (FN == 1)

0 BLUE (N) was predicted as BLACK (P) i.e. 0 negative was predicted as positive. So, it was a False positive (FP == 0) 4 BLUE (N) were predicted as BLUE (N) i.e. 4 negatives were predicted as negative. So, it was a False Negative (TN == 4)

In the following code chink, we'll use this knowledge to compute the rates (tpr, fpr etc).

```
acc <- sum(diag(mydata_test_bag_pred_cm)) / sum(mydata_test_bag_pred_cm)

tpr <- mydata_test_bag_pred_cm[1, 1] / (mydata_test_bag_pred_cm[1, 1] + mydata_test_bag_pred_cm[2, 1])
fpr <- mydata_test_bag_pred_cm[1, 2] / (mydata_test_bag_pred_cm[1, 2] + mydata_test_bag_pred_cm[2, 2])
fnr <- mydata_test_bag_pred_cm[2, 1] / (mydata_test_bag_pred_cm[2, 1] + mydata_test_bag_pred_cm[1, 1])
tnr <- mydata_test_bag_pred_cm[2, 2] / (mydata_test_bag_pred_cm[2, 2] + mydata_test_bag_pred_cm[1, 2])
```

```
auc <- auc(roc(mydata_test_bag_pred, ifelse(mydata_test$label == 'BLUE', 1, 0)))
```

```
## Setting levels: control = BLACK, case = BLUE

## Setting direction: controls < cases
```

```
mydata_test_bag_row <- c("Bagging model ", round(auc, 2), round(acc, 2), round(tpr, 2), round(fpr, 2), r
names(mydata_test_bag_row) <- c("Bagging model ", "AUC", "accuracy", "tpr", "fpr", "fnr", "tnr")
mydata_test_bag_row
```

```
##   Bagging model               AUC         accuracy              tpr
## "Bagging model "            "0.9"            "0.9"           "0.83"
##              fpr              fnr              tnr
##              "0"           "0.17"              "1"
```

(B) Run LOOCV (jacknife) for the same dataset

— iterate over all points

– keep one observation as test

– train using the rest of the observations

– determine test metrics

– aggregate the test metrics

end of loop

find the average of the test metric(s)

Compare (A), (B) above with the results you obtained in HW-1 and write 3 sentences explaining the

observed difference.

LOOCV or Leave-One-Out Cross-Validation procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model.

In this exercise, I used the algorithm that was taught in class M11. However, in heart dataset since **target** field was numeric, and the field **labels** in my current dataset.csv is not numeric, I will substitute mydata_train$labels to binary values 1 or 0 – 1 for BLACK and 0 for BLUE.

```
cat("Current content:")
```

```
## Current content:
```

```
mydata_train$label
```

```
##  [1] BLUE  BLACK BLUE  BLACK BLACK BLACK BLUE  BLUE  BLUE  BLACK BLACK BLUE
## [13] BLACK BLACK BLACK BLUE  BLACK BLACK BLACK BLACK BLACK BLACK BLACK BLUE
## [25] BLUE  BLUE
## Levels: BLACK BLUE
```

```
mydata_train$label <- ifelse(mydata_train$label == 'BLACK', 1, 0)
```

```
cat("Content after alteration:")
```

```
## Content after alteration:
```

```r
mydata_train$label
```

```
##  [1] 0 1 0 1 1 1 0 0 0 1 1 0 1 1 1 0 1 1 1 1 1 1 1 0 0 0
```

```r
N <- nrow(mydata_train)

start_tm2 <- proc.time()

cv_df <- do.call('rbind', lapply(1:N, FUN = function(idx, data = mydata_train) {


   m <- naiveBayes(label~., data = data[-idx,])


   p <- predict(m, data[idx, -c(3)], type = 'raw')

   pc <- unlist(apply(round(p), 1, which.max)) - 1

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

end_tm2 <- proc.time()
```

```r
diff_LOOCV <- end_tm2 - start_tm2
diff_LOOCV
```

```
##    user  system elapsed
##    0.04    0.00    0.05
```

```r
cv_df
```

```
##       fold m      predicted actual
##  [1,] 1    List,5 1         0
##  [2,] 2    List,5 0         1
##  [3,] 3    List,5 1         0
##  [4,] 4    List,5 1         1
##  [5,] 5    List,5 0         1
##  [6,] 6    List,5 1         1
##  [7,] 7    List,5 0         0
##  [8,] 8    List,5 0         0
##  [9,] 9    List,5 1         0
## [10,] 10   List,5 1         1
## [11,] 11   List,5 0         1
## [12,] 12   List,5 1         0
## [13,] 13   List,5 1         1
## [14,] 14   List,5 1         1
## [15,] 15   List,5 1         1
## [16,] 16   List,5 1         0
## [17,] 17   List,5 1         1
## [18,] 18   List,5 1         1
```

```
## [19,] 19   List,5 1         1
## [20,] 20   List,5 1         1
## [21,] 21   List,5 1         1
## [22,] 22   List,5 1         1
## [23,] 23   List,5 0         1
## [24,] 24   List,5 0         0
## [25,] 25   List,5 1         0
## [26,] 26   List,5 1         0
```

```
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  3  7
##   1  4 12
##
##                Accuracy : 0.5769
##                  95% CI : (0.3692, 0.7665)
##     No Information Rate : 0.7308
##     P-Value [Acc > NIR] : 0.9725
##
##                   Kappa : 0.053
##
##  Mcnemar's Test P-Value : 0.5465
##
##             Sensitivity : 0.4286
##             Specificity : 0.6316
##          Pos Pred Value : 0.3000
##          Neg Pred Value : 0.7500
##              Prevalence : 0.2692
##          Detection Rate : 0.1154
##    Detection Prevalence : 0.3846
##       Balanced Accuracy : 0.5301
##
##        'Positive' Class : 0
##
```

```
mydata_test$label <- ifelse(mydata_test$label == 'BLACK', 1, 0)

tstcv.perf <- as.data.frame(do.call('cbind', lapply(cv_df$m, FUN = function(m, data = mydata_test) {
  v <- predict(m, data[, -c(3)], type = 'raw')
  lbllist <- unlist(apply(round(v), 1, which.max)) - 1
}
  )))

np <- ncol(tstcv.perf)
```

```r
predclass <- unlist(apply(tstcv.perf, 1, FUN = function(v) {
   ifelse(sum(v[2:length(v)]) / np < 0.5, 0, 1)
   }
))

loocvtbl <- table(mydata_test[, c(3)], predclass)

(loocv_cfm <- caret::confusionMatrix(loocvtbl))
```

```
## Confusion Matrix and Statistics
##
##    predclass
##     0 1
##   0 4 0
##   1 1 5
##
##                Accuracy : 0.9
##                  95% CI : (0.555, 0.9975)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : 0.01074
##
##                   Kappa : 0.8
##
##  Mcnemar's Test P-Value : 1.00000
##
##             Sensitivity : 0.8000
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.8333
##              Prevalence : 0.5000
##          Detection Rate : 0.4000
##    Detection Prevalence : 0.4000
##       Balanced Accuracy : 0.9000
##
##        'Positive' Class : 0
##
```

Stats of Homework01 are as follows:

(Please refer my Homework01 at https://github.com/ShovanBiswas/DATA622/blob/main/Homework01/Data621-HomeWork01.pdf)

ALGO AUC ACCURACY TPR FPR TNR FNR

1 LR_test 0.84375 70 85.71 66.67 33.33 14.29

2 NB_test 0.5625 0.6 62.5 50 50 37.5

3 KNN_test3 0.8125 0.7 62.5 0 100 37.5

4 KNN_test5 0.8125 0.7 62.5 0 100 37.5

```
cat('Bagging model:\n')
```

```
## Bagging model:
```

```
mydata_test_bag_row
```

```
##   Bagging model                 AUC         accuracy              tpr
## "Bagging model "              "0.9"            "0.9"           "0.83"
##              fpr                fnr              tnr
##              "0"             "0.17"              "1"
```

```
cat('LOOCV:\n')
```

```
## LOOCV:
```

```
loocv_cfm$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##     0.90000000     0.80000000     0.55498388     0.99747142     0.50000000
## AccuracyPValue  McnemarPValue
##     0.01074219     1.00000000
```

Conclusion

1) For both Bagging and LOOCV, the accuracies are 0.9.
2) The accuracies improved from Homework 01.
3) The system time used by both Bagging and LOOCV are almost the same (see below).

```
## Bagging:
```

```
##    user  system elapsed
##    0.25    0.00    0.25
```

```
## LOOCV:
```

```
##    user  system elapsed
##    0.04    0.00    0.05
```