Ensemble Learning

David Teran

Created by David Teran on March 30, 2023

##Ensemble Learning

This R Notebook will be using classification data from a previous R project. With that data, we will be making improvements on the data using 2 of the 4 ensemble methods of learning. The 4 methods are Random Forest, Bagging, AdaBoost, and XGBoost. For this project, the Random Forest and AdaBoost will be used to see if the accuracy of the data can be improved.

First, the classification dataset from a previous R project must be read in and cleaned of any NA or zero values, before attempting to run any of the ensemble learning methods.

```
#Read the dataset in
current_path = rstudioapi::getActiveDocumentContext()$path
setwd(dirname(current_path ))
classData1 <- read.csv("train.csv")
classData2 <- read.csv("test.csv")
classData <- rbind(classData1, classData2)

#Summary of NA or 0 values
print(colSums(is.na(classData)))
```

```
##                                X                               id
##                                0                                0
##                           Gender                    Customer.Type
##                                0                                0
##                              Age                    Type.of.Travel
##                                0                                0
##                            Class                   Flight.Distance
##                                0                                0
##            Inflight.wifi.service Departure.Arrival.time.convenient
##                                0                                0
##           Ease.of.Online.booking                     Gate.location
##                                0                                0
##                   Food.and.drink                   Online.boarding
##                                0                                0
##                     Seat.comfort           Inflight.entertainment
##                                0                                0
##                 On.board.service                 Leg.room.service
##                                0                                0
##                 Baggage.handling                  Checkin.service
##                                0                                0
##                 Inflight.service                      Cleanliness
##                                0                                0
##       Departure.Delay.in.Minutes          Arrival.Delay.in.Minutes
```

```
##                                 0                                 393
##                      satisfaction
##                                 0
```

```
print(sapply(classData, function(y) sum(length(which(y==0)))))
```

```
##                                 X                                id
##                                 2                                 0
##                            Gender                     Customer.Type
##                                 0                                 0
##                               Age                    Type.of.Travel
##                                 0                                 0
##                             Class                   Flight.Distance
##                                 0                                 0
##             Inflight.wifi.service Departure.Arrival.time.convenient
##                              3916                              6681
##            Ease.of.Online.booking                     Gate.location
##                              5682                                 1
##                     Food.and.drink                    Online.boarding
##                               132                              3080
##                      Seat.comfort            Inflight.entertainment
##                                 1                                18
##                  On.board.service                  Leg.room.service
##                                 5                               598
##                  Baggage.handling                   Checkin.service
##                                 0                                 1
##                  Inflight.service                       Cleanliness
##                                 5                                14
##        Departure.Delay.in.Minutes          Arrival.Delay.in.Minutes
##                             73356                             72753
##                      satisfaction
##                                 0
```

Now we clean the data and drop any NA's and 0' along with any unneeded columns. Also, any data must also be normalized to properly run the algorithms on it.

```
#Drop unneeded Columns
classData <- classData[,-1:-2]

#Preparing Data
head(classData)
```

```
##   Gender     Customer.Type Age  Type.of.Travel    Class Flight.Distance
## 1   Male    Loyal Customer  13 Personal Travel Eco Plus             460
## 2   Male disloyal Customer  25 Business travel Business             235
## 3 Female    Loyal Customer  26 Business travel Business            1142
## 4 Female    Loyal Customer  25 Business travel Business             562
## 5   Male    Loyal Customer  61 Business travel Business             214
## 6 Female    Loyal Customer  26 Personal Travel      Eco            1180
##   Inflight.wifi.service Departure.Arrival.time.convenient
## 1                     3                                4
## 2                     3                                2
## 3                     2                                2
```

```
## 4                    2                                5
## 5                    3                                3
## 6                    3                                4
##   Ease.of.Online.booking Gate.location Food.and.drink Online.boarding
## 1                      3             1              5               3
## 2                      3             3              1               3
## 3                      2             2              5               5
## 4                      5             5              2               2
## 5                      3             3              4               5
## 6                      2             1              1               2
##   Seat.comfort Inflight.entertainment On.board.service Leg.room.service
## 1            5                      5                4                3
## 2            1                      1                1                5
## 3            5                      5                4                3
## 4            2                      2                2                5
## 5            5                      3                3                4
## 6            1                      1                3                4
##   Baggage.handling Checkin.service Inflight.service Cleanliness
## 1                4               4                5           5
## 2                3               1                4           1
## 3                4               4                4           5
## 4                3               1                4           2
## 5                4               3                3           3
## 6                4               4                4           1
##   Departure.Delay.in.Minutes Arrival.Delay.in.Minutes            satisfaction
## 1                         25                       18 neutral or dissatisfied
## 2                          1                        6 neutral or dissatisfied
## 3                          0                        0               satisfied
## 4                         11                        9 neutral or dissatisfied
## 5                          0                        0               satisfied
## 6                          0                        0 neutral or dissatisfied
```

```
data(classData)
```

```
## Warning in data(classData): data set 'classData' not found
```

```
names(classData)
```

```
##  [1] "Gender"                          "Customer.Type"
##  [3] "Age"                             "Type.of.Travel"
##  [5] "Class"                           "Flight.Distance"
##  [7] "Inflight.wifi.service"           "Departure.Arrival.time.convenient"
##  [9] "Ease.of.Online.booking"          "Gate.location"
## [11] "Food.and.drink"                  "Online.boarding"
## [13] "Seat.comfort"                    "Inflight.entertainment"
## [15] "On.board.service"                "Leg.room.service"
## [17] "Baggage.handling"                "Checkin.service"
## [19] "Inflight.service"                "Cleanliness"
## [21] "Departure.Delay.in.Minutes"      "Arrival.Delay.in.Minutes"
## [23] "satisfaction"
```

```r
#Convert data columns to numeric data
classData$Gender <- ifelse(classData$Gender=="Female", 1, 0)
classData$Customer.Type <- ifelse(classData$Customer.Type=="Local Customer", 1, 0)
classData$Type.of.Travel <- ifelse(classData$Type.of.Travel=="Business travel", 1, 0)
classData$Class[classData$Class == "Eco"] <- 0
classData$Class[classData$Class == "Eco Plus"] <- 1
classData$Class[classData$Class == "Business"] <- 2

#Remove 0's
classData <- na.omit(classData) #Clear missing data
classData <- classData[!(is.na(classData$Arrival.Delay.in.Minutes)),]

#Normalizing Data
classData$Class <- as.numeric(classData$Class)
classData$satisfaction <- as.factor(classData$satisfaction)


classData <- classData[!(classData$Gate.location==0),]
classData <- classData[!(classData$Food.and.drink==0),]
classData <- classData[!(classData$Online.boarding==0),]
classData <- classData[!(classData$Seat.comfort==0),]
classData <- classData[!(classData$Inflight.entertainment==0),]
classData <- classData[!(classData$On.board.service==0),]
classData <- classData[!(classData$Leg.room.service==0),]
classData <- classData[!(classData$Checkin.service==0),]
classData <- classData[!(classData$Inflight.service==0),]
classData <- classData[!(classData$Cleanliness==0),]
classData <- classData[!(classData$Departure.Arrival.time.convenient==0),]
classData <- classData[!(classData$Departure.Delay.in.Minutes==0),]
classData <- classData[!(classData$Arrival.Delay.in.Minutes==0),]
classData <- classData[!(classData$Inflight.wifi.service==0),]
classData <- classData[!(classData$Ease.of.Online.booking==0),]

#Convert to Factors
classData$satisfaction<-as.factor(classData$satisfaction)
```

Once the data has been cleared of NA and 0 values, the data is then split into train/test data using an 80/20 split.

```r
set.seed(1234)
i <- sample(1:nrow(classData), nrow(classData)*0.80, replace=FALSE)
train <- classData[i,]
test <- classData[-i,]
```

Having split the data, we then run logistic regression to set up the baseline for the accuracy and Matthew's Correlation Coefficient (mcc) to account for any differences in class distribution.

```r
glmPass <- glm(satisfaction~., data=train, family="binomial")
summary(glmPass)
```

```
##
## Call:
```

```
## glm(formula = satisfaction ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.1924  -0.3158  -0.0750   0.2952   4.2442
##
## Coefficients: (1 not defined because of singularities)
##                                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)                      -1.488e+01  1.987e-01 -74.888  < 2e-16 ***
## Gender                           -8.355e-02  4.043e-02  -2.066   0.0388 *
## Customer.Type                           NA         NA      NA       NA
## Age                               2.019e-02  1.496e-03  13.502  < 2e-16 ***
## Type.of.Travel                    2.802e+00  7.362e-02  38.058  < 2e-16 ***
## Class                             5.316e-01  2.703e-02  19.670  < 2e-16 ***
## Flight.Distance                   2.742e-04  2.164e-05  12.669  < 2e-16 ***
## Inflight.wifi.service             6.139e-01  2.410e-02  25.468  < 2e-16 ***
## Departure.Arrival.time.convenient -2.523e-01  2.449e-02 -10.304  < 2e-16 ***
## Ease.of.Online.booking            2.851e-01  2.572e-02  11.084  < 2e-16 ***
## Gate.location                    -2.318e-01  2.354e-02  -9.846  < 2e-16 ***
## Food.and.drink                   -1.289e-02  2.099e-02  -0.614   0.5391
## Online.boarding                   9.371e-01  2.330e-02  40.217  < 2e-16 ***
## Seat.comfort                      7.006e-02  2.414e-02   2.902   0.0037 **
## Inflight.entertainment            2.672e-01  2.749e-02   9.722  < 2e-16 ***
## On.board.service                  3.199e-01  2.095e-02  15.265  < 2e-16 ***
## Leg.room.service                  2.838e-01  1.888e-02  15.035  < 2e-16 ***
## Baggage.handling                  1.208e-01  2.419e-02   4.994 5.90e-07 ***
## Checkin.service                   3.072e-01  1.770e-02  17.353  < 2e-16 ***
## Inflight.service                  1.466e-01  2.470e-02   5.934 2.96e-09 ***
## Cleanliness                       2.079e-01  2.406e-02   8.642  < 2e-16 ***
## Departure.Delay.in.Minutes        5.125e-04  1.356e-03   0.378   0.7055
## Arrival.Delay.in.Minutes         -1.240e-03  1.329e-03  -0.933   0.3509
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 42043  on 31926  degrees of freedom
## Residual deviance: 16220  on 31905  degrees of freedom
## AIC: 16264
##
## Number of Fisher Scoring iterations: 7
```

With the Logistic regression model created, along with a summary printed, we can check for accuracy and mcc.

```
library(mltools)
probs <- predict(glmPass, newdata=test, type="response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
pred <- ifelse(probs>0.5, 2, 1)
accuracy <- mean(pred==as.integer(test$satisfaction))
mcc1 <- mcc(pred, as.integer(test$satisfaction))
print(paste("accuracy=", accuracy))
```

```
## [1] "accuracy= 0.903407667251315"
```

```
print(paste("mcc=", mcc1))
```

```
## [1] "mcc= 0.791512285107994"
```

From here, we can run the random forest algorithm to see if the accuracy and mcc values improve with the current logistic regression as the baseline.

```
library(randomForest)
```

```
## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.
```

```
set.seed(1234)
randomF1 <- randomForest(satisfaction~., data=train, importance=TRUE)
summary(randomF1)
```

```
##                 Length Class  Mode
## call               4  -none- call
## type               1  -none- character
## predicted      31927  factor numeric
## err.rate        1500  -none- numeric
## confusion          6  -none- numeric
## votes          63854  matrix numeric
## oob.times      31927  -none- numeric
## classes            2  -none- character
## importance        88  -none- numeric
## importanceSD      66  -none- numeric
## localImportance    0  -none- NULL
## proximity          0  -none- NULL
## ntree              1  -none- numeric
## mtry               1  -none- numeric
## forest            14  -none- list
## y              31927  factor numeric
## test               0  -none- NULL
## inbag              0  -none- NULL
## terms              3  terms  call
```

```
pred <- predict(randomF1, newdata=test, type="response")
accuracyF1 <- mean(pred==test$satisfaction)
mccF1 <- mcc(factor(pred), test$satisfaction)
print(paste("accuracy=", accuracyF1))
```

```
## [1] "accuracy= 0.961789025306941"
```

```
print(paste("mcc=", mccF1))
```

```
## [1] "mcc= 0.917357766982803"
```

With the random forest ensemble learning model completed, there is a difference between logistic regression and random forest, where random forest does have a higher accuracy and mcc value.

Next is to try increasing the accuracy and mcc values with AdaBoost.

```
library(adabag)
```

```
## Loading required package: rpart
```

```
## Loading required package: caret
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:randomForest':
##
##     margin
```

```
## Loading required package: lattice
```

```
## Loading required package: foreach
```

```
## Loading required package: doParallel
```

```
## Loading required package: iterators
```

```
## Loading required package: parallel
```

```
library(ggplot2)
adaBoostData <- boosting(satisfaction~., data=train, boos=TRUE, mfinal=20, coeflearn='Breiman')
summary(adaBoostData)
```

```
##            Length Class   Mode
## formula         3  formula call
## trees          20  -none-  list
## weights        20  -none-  numeric
## votes       63854  -none-  numeric
## prob        63854  -none-  numeric
## class       31927  -none-  character
## importance     22  -none-  numeric
## terms           3  terms   call
## call            6  -none-  call
```

```r
pred <- predict(adaBoostData, newdata=test, type="response")
accAda <- mean(pred$class==test$satisfaction)
mccAda <- mcc(factor(pred$class), test$satisfaction)
print(paste("accuracy=", accAda))
```

```
## [1] "accuracy= 0.948759709346029"
```

```r
print(paste("mcc=", mccAda))
```

```
## [1] "mcc= 0.889097452702984"
```

From here, we can see that AdaBoost does fairly well, while still better than logistic regression. It is much faster in runtime compared to random forest, which does take longer than either logistic regression and AdaBoost

Overall, ensemble learning does provide a better experience in obtaining accuracy with some additional efficiency. The difference between the three is that AdaBoost is faster than logistic regression and random forest. As for accuracy and mcc values, random forest seems to do better than both logistic regression and AdaBoost, with AdaBoost having a lower mcc value than random forest, but still better than linear regression.