Assignment 3 - Flight Delay Problem

Steve Spence

10/17/2019

## Load Data Set and Libraries

First, we will load all of the packages that will be required for this problem. Specifically, “ISLR”, “caret”, “dplyr”, “e1071”, and “pROC” will be loaded for this problem.

Next, we will import the “FlightDelays” data set into the RStudio environment.

# Import data set from BlackBoard into the RStudio environment  
  
FlightDelays <- read.csv("FlightDelays.csv")

## Data Structure

A summary of the data set will be displayed to review the data set.

# Investigate the structure of the data set  
  
str(FlightDelays)

## 'data.frame': 2201 obs. of 13 variables:  
## $ CRS\_DEP\_TIME : int 1455 1640 1245 1715 1039 840 1240 1645 1715 2120 ...  
## $ CARRIER : Factor w/ 8 levels "CO","DH","DL",..: 5 2 2 2 2 2 2 2 2 2 ...  
## $ DEP\_TIME : int 1455 1640 1245 1709 1035 839 1243 1644 1710 2129 ...  
## $ DEST : Factor w/ 3 levels "EWR","JFK","LGA": 2 2 3 3 3 2 2 2 2 2 ...  
## $ DISTANCE : int 184 213 229 229 229 228 228 228 228 228 ...  
## $ FL\_DATE : Factor w/ 31 levels "01/01/2004","01/02/2004",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ FL\_NUM : int 5935 6155 7208 7215 7792 7800 7806 7810 7812 7814 ...  
## $ ORIGIN : Factor w/ 3 levels "BWI","DCA","IAD": 1 2 3 3 3 3 3 3 3 3 ...  
## $ Weather : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ DAY\_WEEK : int 4 4 4 4 4 4 4 4 4 4 ...  
## $ DAY\_OF\_MONTH : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ TAIL\_NUM : Factor w/ 549 levels "N10323","N10575",..: 526 263 382 350 385 374 241 227 246 372 ...  
## $ Flight.Status: Factor w/ 2 levels "delayed","ontime": 2 2 2 2 2 2 2 2 2 2 ...

# Investigate the summary of the data set  
  
summary(FlightDelays)

## CRS\_DEP\_TIME CARRIER DEP\_TIME DEST DISTANCE   
## Min. : 600 DH :551 Min. : 10 EWR: 665 Min. :169.0   
## 1st Qu.:1000 RU :408 1st Qu.:1004 JFK: 386 1st Qu.:213.0   
## Median :1455 US :404 Median :1450 LGA:1150 Median :214.0   
## Mean :1372 DL :388 Mean :1369 Mean :211.9   
## 3rd Qu.:1710 MQ :295 3rd Qu.:1709 3rd Qu.:214.0   
## Max. :2130 CO : 94 Max. :2330 Max. :229.0   
## (Other): 61   
## FL\_DATE FL\_NUM ORIGIN Weather   
## 1/22/2004 : 86 Min. : 746 BWI: 145 Min. :0.00000   
## 01/06/2004: 85 1st Qu.:2156 DCA:1370 1st Qu.:0.00000   
## 01/08/2004: 85 Median :2385 IAD: 686 Median :0.00000   
## 1/13/2004 : 85 Mean :3815 Mean :0.01454   
## 1/20/2004 : 85 3rd Qu.:6155 3rd Qu.:0.00000   
## 1/21/2004 : 85 Max. :7924 Max. :1.00000   
## (Other) :1690   
## DAY\_WEEK DAY\_OF\_MONTH TAIL\_NUM Flight.Status   
## Min. :1.000 Min. : 1.00 N225DL : 65 delayed: 428   
## 1st Qu.:2.000 1st Qu.: 8.00 N242DL : 56 ontime :1773   
## Median :4.000 Median :16.00 N223DZ : 50   
## Mean :3.905 Mean :16.02 N221DL : 45   
## 3rd Qu.:5.000 3rd Qu.:23.00 N241DL : 36   
## Max. :7.000 Max. :31.00 N722UW : 36   
## (Other):1913

We can see from the data set above, it appears that 428 out of 2,201 flights (Approximately 19.5%) are delayed historically. This can be compared against the final model at the end for a reality check.

Given in the problem statement, we will only be using five predictors: “DAY\_WEEK”, “DEP\_TIME”, “ORIGIN”, “DEST”, and “CARRIER” along with the dependent variable “Flight.Status”. Therefore, we will re-write the dataframe with these 6 variables.

Additionally, we will need to convert all variables to factors for Naive Bayes model.

# Isolate the 6 variables previously discussed  
  
FlightDelays <- FlightDelays[ , c(10, 1, 8, 4, 2, 13)]  
  
# Review the new structure of the dataset  
  
str(FlightDelays)

## 'data.frame': 2201 obs. of 6 variables:  
## $ DAY\_WEEK : int 4 4 4 4 4 4 4 4 4 4 ...  
## $ CRS\_DEP\_TIME : int 1455 1640 1245 1715 1039 840 1240 1645 1715 2120 ...  
## $ ORIGIN : Factor w/ 3 levels "BWI","DCA","IAD": 1 2 3 3 3 3 3 3 3 3 ...  
## $ DEST : Factor w/ 3 levels "EWR","JFK","LGA": 2 2 3 3 3 2 2 2 2 2 ...  
## $ CARRIER : Factor w/ 8 levels "CO","DH","DL",..: 5 2 2 2 2 2 2 2 2 2 ...  
## $ Flight.Status: Factor w/ 2 levels "delayed","ontime": 2 2 2 2 2 2 2 2 2 2 ...

Now that we have the 6 varaibles in question, we must ensure all of them are converted to factors for proper use in the Naive Bayes algorithm. As shown above, “DAY\_WEEK” and “CRS\_DEP\_TIME” are the two remaining variables that need to be converted to factors.

Additionally, “CRS\_DEP\_TIME” will also need to be converted to a factor with 16 time ranges (as stated in the initial problem statement). These time ranges will stretch from “6:00am to 7:00am” departure time to the final time range of “9:00pm to 10:00pm”. These will correlate to factor level 1 to 16, respectively.

# Convert "DAY\_WEEK" to a factor for Naives Bayes  
  
FlightDelays$DAY\_WEEK <- as.factor(FlightDelays$DAY\_WEEK)  
  
# Convert "CRS\_DEP\_TIME" to a factor with 16 time ranges and labels  
  
FlightDelays$CRS\_DEP\_TIME <- cut(FlightDelays$CRS\_DEP\_TIME, breaks = c(600, 700, 800, 900, 1000, 1100, 1200, 1300, 1400, 1500, 1600, 1700, 1800, 1900, 2000, 2100, 2200))  
  
# Verify the new structure of the data set  
  
str(FlightDelays)

## 'data.frame': 2201 obs. of 6 variables:  
## $ DAY\_WEEK : Factor w/ 7 levels "1","2","3","4",..: 4 4 4 4 4 4 4 4 4 4 ...  
## $ CRS\_DEP\_TIME : Factor w/ 16 levels "(600,700]","(700,800]",..: 9 11 7 12 5 3 7 11 12 16 ...  
## $ ORIGIN : Factor w/ 3 levels "BWI","DCA","IAD": 1 2 3 3 3 3 3 3 3 3 ...  
## $ DEST : Factor w/ 3 levels "EWR","JFK","LGA": 2 2 3 3 3 2 2 2 2 2 ...  
## $ CARRIER : Factor w/ 8 levels "CO","DH","DL",..: 5 2 2 2 2 2 2 2 2 2 ...  
## $ Flight.Status: Factor w/ 2 levels "delayed","ontime": 2 2 2 2 2 2 2 2 2 2 ...

## Data Preprocessing

To begin preprocessing data, we must split the data set into 60% training and 40% validation, per the problem description.

# Set the seed for randomized functions  
  
set.seed(102019)  
  
# Split the data into 60% training data and 40% validation data  
  
FlightDelaysIndex <- createDataPartition(FlightDelays$DAY\_WEEK, p=0.4, list = F)  
  
FlightDelaysValidation <- FlightDelays[FlightDelaysIndex,]  
  
FlightDelaysTrain <- FlightDelays[-FlightDelaysIndex,]

## Counts and Proportion Table by Airport

Assignment calls for a table containing a count and proportion of delayed flights by airport. For this, the “dplyr” package and summarise function will be used to return these values.

# Summarise flight delay statisitics by airport  
  
FlightDelays %>%  
 group\_by(Origin\_Airport = ORIGIN, Flight\_Status = Flight.Status) %>%  
 summarise(Count\_of\_Flights = n()) %>%  
 mutate(Proportion\_for\_Airport = 100\*(Count\_of\_Flights / sum(Count\_of\_Flights)))

## # A tibble: 6 x 4  
## # Groups: Origin\_Airport [3]  
## Origin\_Airport Flight\_Status Count\_of\_Flights Proportion\_for\_Airport  
## <fct> <fct> <int> <dbl>  
## 1 BWI delayed 37 25.5  
## 2 BWI ontime 108 74.5  
## 3 DCA delayed 221 16.1  
## 4 DCA ontime 1149 83.9  
## 5 IAD delayed 170 24.8  
## 6 IAD ontime 516 75.2

From this data table, we can see the total number of flights from each airport and the proportion of them that were delayed for each individual airport.

“BWI” had the lowest number of flights during this time period, but it also had the highest percentage of them being delayed.

## Create Naive Bayes Model

Now that we have the training and validation data properly prepared, the Naive Bayes model can be created from the training data and then ran on the validation data.

# Create Naive Bayes model from the training data set  
  
NB\_Model <- naiveBayes(Flight.Status ~ ., data = FlightDelaysTrain)  
  
# Use Model on the validation data set to predict if flights will be delayed  
  
FlightDelaysValidation\_Predicted <- predict(NB\_Model, FlightDelaysValidation)

## Confusion Matrix and ROC for Validation Data

A confusion matrix will be created to determine the accuracy of the model with varying statistics.

# Create confusion matrix for the label outputs from the Naive Bayes model  
  
CrossTable(x = FlightDelaysValidation$Flight.Status, y = FlightDelaysValidation\_Predicted, prop.chisq = FALSE)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Row Total |  
## | N / Col Total |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 883   
##   
##   
## | FlightDelaysValidation\_Predicted   
## FlightDelaysValidation$Flight.Status | delayed | ontime | Row Total |   
## -------------------------------------|-----------|-----------|-----------|  
## delayed | 17 | 173 | 190 |   
## | 0.089 | 0.911 | 0.215 |   
## | 0.500 | 0.204 | |   
## | 0.019 | 0.196 | |   
## -------------------------------------|-----------|-----------|-----------|  
## ontime | 17 | 676 | 693 |   
## | 0.025 | 0.975 | 0.785 |   
## | 0.500 | 0.796 | |   
## | 0.019 | 0.766 | |   
## -------------------------------------|-----------|-----------|-----------|  
## Column Total | 34 | 849 | 883 |   
## | 0.039 | 0.961 | |   
## -------------------------------------|-----------|-----------|-----------|  
##   
##

From this confusion matrix, we can state that:

The sensitivity (true positive rate) of the model is: 676/693 = 97.5% The specificity (true negative rate) of the model is: 17/190 = 9.0%

Next, there will be the creation of the ROC cure and return the AUC value for the validation data on this model.

Before the curve is plotted out, there must be another model ran with the raw probabilities listed.

# Re-run the model to return the raw probabilities  
  
FlightDelaysValidation\_Predicted2 <- predict(NB\_Model, FlightDelaysValidation, type = "raw")  
  
# Return the first few values in the output table  
  
head(FlightDelaysValidation\_Predicted2)

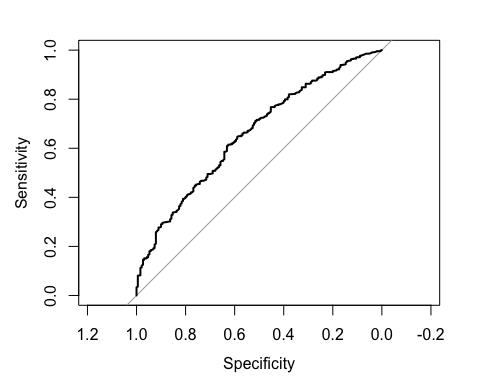
## delayed ontime  
## [1,] 0.13747017 0.8625298  
## [2,] 0.20216762 0.7978324  
## [3,] 0.36800677 0.6319932  
## [4,] 0.29987116 0.7001288  
## [5,] 0.03522145 0.9647786  
## [6,] 0.09009041 0.9099096

Next, the curves can now be plotted out.

# Creating the ROC curve for the model  
  
roc(FlightDelaysValidation$Flight.Status, FlightDelaysValidation\_Predicted2[,2])

##   
## Call:  
## roc.default(response = FlightDelaysValidation$Flight.Status, predictor = FlightDelaysValidation\_Predicted2[, 2])  
##   
## Data: FlightDelaysValidation\_Predicted2[, 2] in 190 controls (FlightDelaysValidation$Flight.Status delayed) < 693 cases (FlightDelaysValidation$Flight.Status ontime).  
## Area under the curve: 0.6588

# Plot the ROC curve  
  
plot.roc(FlightDelaysValidation$Flight.Status, FlightDelaysValidation\_Predicted2[,2])



From the AUC value of 0.6588, it can be stated that the model is better than chance guessing; however, the model still has some room for improvement to increase this AUC value closer and closer to nearly 1.0.