Machine Learning - Assignment 3

Tom Scarberry

10/13/2020

##load Flight Delays data and view summary information and load R libraries

```
Delays.all<-read.csv("FlightDelays.csv")
summary(Delays.all)
```

```
##   CRS_DEP_TIME    CARRIER            DEP_TIME        DEST
##  Min.   : 600   Length:2201        Min.   :  10   Length:2201
##  1st Qu.:1000   Class :character   1st Qu.:1004   Class :character
##  Median :1455   Mode  :character   Median :1450   Mode  :character
##  Mean   :1372                      Mean   :1369
##  3rd Qu.:1710                      3rd Qu.:1709
##  Max.   :2130                      Max.   :2330
##     DISTANCE       FL_DATE             FL_NUM        ORIGIN
##  Min.   :169.0   Length:2201         Min.   : 746   Length:2201
##  1st Qu.:213.0   Class :character    1st Qu.:2156   Class :character
##  Median :214.0   Mode  :character    Median :2385   Mode  :character
##  Mean   :211.9                       Mean   :3815
##  3rd Qu.:214.0                       3rd Qu.:6155
##  Max.   :229.0                       Max.   :7924
##     Weather           DAY_WEEK      DAY_OF_MONTH     TAIL_NUM
##  Min.   :0.00000   Min.   :1.000   Min.   : 1.00   Length:2201
##  1st Qu.:0.00000   1st Qu.:2.000   1st Qu.: 8.00   Class :character
##  Median :0.00000   Median :4.000   Median :16.00   Mode  :character
##  Mean   :0.01454   Mean   :3.905   Mean   :16.02
##  3rd Qu.:0.00000   3rd Qu.:5.000   3rd Qu.:23.00
##  Max.   :1.00000   Max.   :7.000   Max.   :31.00
##  Flight.Status
##  Length:2201
##  Class :character
##  Mode  :character
##
##
##
```

```
library(caret)
```

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

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

```
library(ISLR)
library(naivebayes)
```

```
## naivebayes 0.9.7 loaded
```

Create new data frame with five predictors and identify the variable types to check if factors or other variable types

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
Delays<-select(Delays.all, Flight.Status, DAY_WEEK, CRS_DEP_TIME, ORIGIN,DEST, CARRIER)
summary(Delays)
```

```
##  Flight.Status         DAY_WEEK      CRS_DEP_TIME     ORIGIN
##  Length:2201        Min.   :1.000   Min.   : 600   Length:2201
##  Class :character   1st Qu.:2.000   1st Qu.:1000   Class :character
##  Mode  :character   Median :4.000   Median :1455   Mode  :character
##                     Mean   :3.905   Mean   :1372
##                     3rd Qu.:5.000   3rd Qu.:1710
##                     Max.   :7.000   Max.   :2130
##      DEST             CARRIER
##  Length:2201        Length:2201
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

```
str(Delays)
```

```
## 'data.frame':    2201 obs. of  6 variables:
##  $ Flight.Status: chr  "ontime" "ontime" "ontime" "ontime" ...
##  $ 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       : chr  "BWI" "DCA" "IAD" "IAD" ...
##  $ DEST         : chr  "JFK" "JFK" "LGA" "LGA" ...
##  $ CARRIER      : chr  "OH" "DH" "DH" "DH" ...
```

Create bins for flight departure times

```
Delays.bin<-Delays
Delays.bin[,3]<-round(Delays$CRS_DEP_TIME/100, digit=0)
str(Delays.bin$CRS_DEP_TIME)
```

```
##  num [1:2201] 15 16 12 17 10 8 12 16 17 21 ...
```

Convert all data variables to factors for Naive Bayes model

```
Delays$Flight.Status<-as.factor(Delays$Flight.Status)
Delays$DAY_WEEK<-as.factor(Delays$DAY_WEEK)
Delays$CRS_DEP_TIME<-as.factor(Delays.bin$CRS_DEP_TIME)
Delays$ORIGIN<-as.factor(Delays$ORIGIN)
Delays$DEST<-as.factor(Delays$DEST)
Delays$CARRIER<-as.factor(Delays$CARRIER)
summary(Delays)
```

```
##  Flight.Status  DAY_WEEK  CRS_DEP_TIME  ORIGIN      DEST         CARRIER
##  delayed: 428   1:308    15     : 292   BWI: 145   EWR: 665   DH     :551
##  ontime :1773   2:307    17     : 241   DCA:1370   JFK: 386   RU     :408
##                 3:320    16     : 178   IAD: 686   LGA:1150   US     :404
##                 4:372    8      : 164                         DL     :388
##                 5:391    12     : 142                         MQ     :295
##                 6:250    21     : 137                         CO     : 94
##                 7:253    (Other):1047                         (Other): 61
```

Divide the data into 60% training and 40% validation data sets

```
set.seed(20)
train.index<-createDataPartition(Delays$Flight.Status,p=0.6, list=FALSE)
train.data<-Delays[train.index,]
validation.data<-Delays[-train.index,]
summary(train.data$Flight.Status)
```

```
## delayed  ontime
##     257    1064
```

```
summary(validation.data$Flight.Status)
```

```
## delayed  ontime
##     171     709
```

Build Naive Bayes model with training data

```r
model.train<-naive_bayes(Flight.Status~DAY_WEEK+CRS_DEP_TIME+ORIGIN+DEST+CARRIER,data=train.data)
model.train
```

```
##
## ================================== Naive Bayes ==================================
##
##  Call:
## naive_bayes.formula(formula = Flight.Status ~ DAY_WEEK + CRS_DEP_TIME +
##     ORIGIN + DEST + CARRIER, data = train.data)
##
## ---------------------------------------------------------------------------------
##
## Laplace smoothing: 0
##
## ---------------------------------------------------------------------------------
##
##  A priori probabilities:
##
##   delayed    ontime
## 0.1945496 0.8054504
##
## ---------------------------------------------------------------------------------
##
##  Tables:
##
## ---------------------------------------------------------------------------------
##  ::: DAY_WEEK (Categorical)
## ---------------------------------------------------------------------------------
##
## DAY_WEEK    delayed     ontime
##        1 0.21789883 0.11748120
##        2 0.12840467 0.13909774
##        3 0.10894942 0.14191729
##        4 0.13229572 0.17387218
##        5 0.16731518 0.17387218
##        6 0.07392996 0.13627820
##        7 0.17120623 0.11748120
##
## ---------------------------------------------------------------------------------
##  ::: CRS_DEP_TIME (Categorical)
## ---------------------------------------------------------------------------------
##
## CRS_DEP_TIME    delayed     ontime
##            6  0.04280156 0.06390977
##            7  0.07003891 0.06203008
##            8  0.06225681 0.07706767
##            9  0.01556420 0.06109023
##           10 0.02723735 0.04793233
##           11 0.01945525 0.03289474
##           12 0.05447471 0.07988722
##           13 0.04280156 0.06860902
##           14 0.05447471 0.04793233
##           15 0.17898833 0.11090226
```

```
##           16 0.08560311 0.08646617
##           17 0.14785992 0.10056391
##           18 0.01945525 0.03947368
##           19 0.08560311 0.04041353
##           20 0.01167315 0.02913534
##           21 0.08171206 0.05169173
##
## ------------------------------------------------------------------------------
##  ::: ORIGIN (Categorical)
## ------------------------------------------------------------------------------
##
## ORIGIN    delayed     ontime
##    BWI 0.07003891 0.05733083
##    DCA 0.52140078 0.64849624
##    IAD 0.40856031 0.29417293
##
## ------------------------------------------------------------------------------
##  ::: DEST (Categorical)
## ------------------------------------------------------------------------------
##
## DEST    delayed    ontime
##   EWR 0.3618677 0.2875940
##   JFK 0.1984436 0.1635338
##   LGA 0.4396887 0.5488722
##
## ------------------------------------------------------------------------------
##  ::: CARRIER (Categorical)
## ------------------------------------------------------------------------------
##
## CARRIER     delayed      ontime
##      CO 0.054474708 0.036654135
##      DH 0.330739300 0.231203008
##      DL 0.120622568 0.191729323
##      MQ 0.206225681 0.114661654
##      OH 0.003891051 0.014097744
##      RU 0.206225681 0.180451128
##      UA 0.011673152 0.016917293
##      US 0.066147860 0.214285714
##
## ------------------------------------------------------------------------------
```

Output both a count table and proportion table of how many and what proportion of flights were delayed and ontime at the three airports for training data set

```
table(train.data$Flight.Status, train.data$DEST)
```

```
##
##           EWR JFK LGA
##   delayed  93  51 113
##   ontime  306 174 584
```

```
prop.table(table(train.data$Flight.Status, train.data$DEST), margin = 1)
```

```
##
##                 EWR       JFK       LGA
##   delayed 0.3618677 0.1984436 0.4396887
##   ontime  0.2875940 0.1635338 0.5488722
```

LGA has the highest percentage delays with 44% of the total flights delayed and the highest total delays at 113, but LGA also has the highest total number of flights of the three airports for the training data set.

Output both a count table and proportion table of how many and what proportion of flights were delayed and ontime at the three airports for entire data set

```
table(Delays$Flight.Status, Delays$DEST)
```

```
##
##           EWR JFK LGA
##   delayed 161  84 183
##   ontime  504 302 967
```

```
prop.table(table(Delays$Flight.Status, Delays$DEST), margin = 1)
```

```
##
##                 EWR       JFK       LGA
##   delayed 0.3761682 0.1962617 0.4275701
##   ontime  0.2842640 0.1703328 0.5454033
```

LGA has the highest percentage delays with 43% of the total flights delayed and the highest total delays at 183, but LGA also has the highest total number of flights of the three airports for the total data set.

Run the Naive Bayes model with the validation data

View Confusion matrix

```
predict.delays<-predict(model.train,validation.data)
```

```
## Warning: predict.naive_bayes(): more features in the newdata are provided as
## there are probability tables in the object. Calculation is performed based on
## features to be found in the tables.
```

```
library(gmodels)
CrossTable(x=validation.data$Flight.Status, y=predict.delays,prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  880
##
##
##                                 | predict.delays
## validation.data$Flight.Status |   delayed |    ontime | Row Total |
## ------------------------------|-----------|-----------|-----------|
##                       delayed |        18 |       153 |       171 |
##                               |     0.105 |     0.895 |     0.194 |
##                               |     0.419 |     0.183 |           |
##                               |     0.020 |     0.174 |           |
## ------------------------------|-----------|-----------|-----------|
##                        ontime |        25 |       684 |       709 |
##                               |     0.035 |     0.965 |     0.806 |
##                               |     0.581 |     0.817 |           |
##                               |     0.028 |     0.777 |           |
## ------------------------------|-----------|-----------|-----------|
##                  Column Total |        43 |       837 |       880 |
##                               |     0.049 |     0.951 |           |
## ------------------------------|-----------|-----------|-----------|
##
##
```

Accuracy for the is 80% for the model, which is not very accurate for a Confusion Matrix. The model has low precision (82%) and high recall (sensitivity) with a score of (96%). The model predicts most flights as on time because that is the higher percentage likelyhood of occuring and thus the model performs poorly on correctly identifying delayed flights. This will lead to flyers expecting to be on time, but becoming unhappy when the flight is delayed.

Convert the model to raw prediction to output for the ROC graph for the validation data

```
predict.delays.raw<-predict(model.train,validation.data, type="prob")
```

```
## Warning: predict.naive_bayes(): more features in the newdata are provided as
## there are probability tables in the object. Calculation is performed based on
## features to be found in the tables.
```

```r
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following object is masked from 'package:gmodels':
##
##     ci

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
summary(predict.delays.raw)
```

```
##     delayed             ontime
##  Min.   :0.006594   Min.   :0.3610
##  1st Qu.:0.071554   1st Qu.:0.6654
##  Median :0.188468   Median :0.8115
##  Mean   :0.216632   Mean   :0.7834
##  3rd Qu.:0.334643   3rd Qu.:0.9284
##  Max.   :0.639035   Max.   :0.9934
```

```r
head(predict.delays.raw)
```

```
##        delayed    ontime
## [1,] 0.1082288 0.8917712
## [2,] 0.2025157 0.7974843
## [3,] 0.3007317 0.6992683
## [4,] 0.1425233 0.8574767
## [5,] 0.2635845 0.7364155
## [6,] 0.3944768 0.6055232
```

```r
roc(validation.data$Flight.Status,predict.delays.raw[,2])
```

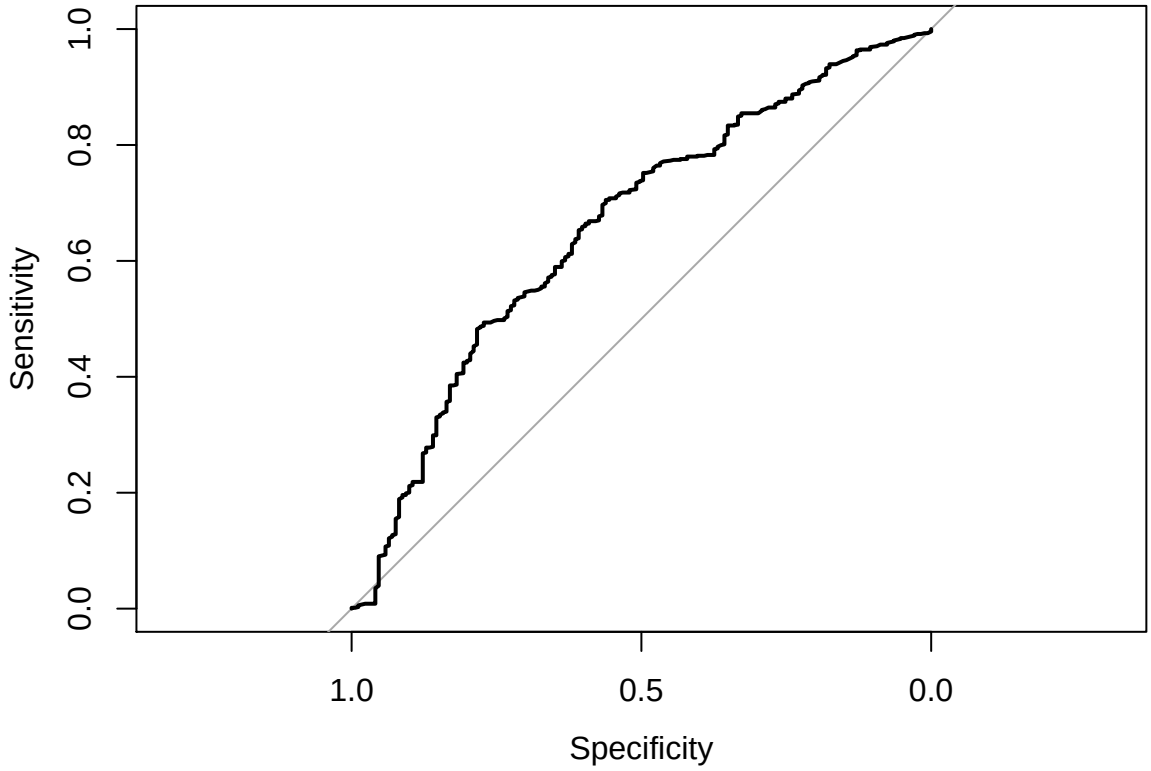
```
## Setting levels: control = delayed, case = ontime

## Setting direction: controls < cases

##
## Call:
## roc.default(response = validation.data$Flight.Status, predictor = predict.delays.raw[,     2])
##
## Data: predict.delays.raw[, 2] in 171 controls (validation.data$Flight.Status delayed) < 709 cases (va
## Area under the curve: 0.658
```

```
plot.roc(validation.data$Flight.Status,predict.delays.raw[,2])
```

```
## Setting levels: control = delayed, case = ontime
## Setting direction: controls < cases
```



AUC value is 0.658 and ROC plot is shown. The AUC value is better closer to 1 and the ROC plot is best if it plots the curve close to the top left corner of the chart.