Naive_Bayes_FlightDelays

Sumanth

10/19/2019

NB: Predicting Delayed Flights

Predicting flight delays can be useful to a variety of organizations. Here, we look at five predictors. The outcome of interest is whether or not the flight is delayed (delayed here means arrived more than 15 minutes late). Our data consist of all flights from the Washington, DC area into the New York City area during January 2004. A record is a particular flight. The percentage of delayed flights among these 2201 flights is 19.5%. The data were obtained from the Bureau of Transportation Statistics (available on the web at www.transtats.bts.gov). The goal is to accurately predict whether or not a new flight (not in this dataset), will be delayed. The outcome variable is whether the flight was delayed, and thus it has two classes (1 = delayed and 0 = on time).

```r
library(e1071)
getwd()
```

```
## [1] "C:/Users/suman/Documents/Machine Learning/Assignments/Ass 3"
```

```r
FD <- read.csv("FlightDelays.csv")
# change numerical variables to categorical first
FD$DAY_WEEK <- factor(FD$DAY_WEEK)
FD$DEP_TIME <- factor(FD$DEP_TIME)
# create hourly bins departure time
FD$CRS_DEP_TIME <- factor(round(FD$CRS_DEP_TIME/100))
```

1. Divide the data into 60% training and 40% validation

```r
library(ISLR)
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```r
# Create training and validation sets.
FD_Pred <- FD[,c(10, 1, 8, 4, 2, 13)]
train.index <- createDataPartition(FD_Pred$Flight.Status,p=0.6,list = FALSE)
train.df <- FD_Pred[train.index, ]
valid.df <- FD_Pred[-train.index, ]
```

2. Run the Naive Bayes model to predict whether the flight is delayed or not. Use only categorical variables for the predictor variables. Note that Week and Time variables need to recoded as factors

```r
# run naive bayes
nb_delays <- naiveBayes(Flight.Status ~ ., data = train.df)
head(nb_delays)
```

```
## $apriori
## Y
## delayed  ontime
##     257    1064
##
## $tables
## $tables$DAY_WEEK
##          DAY_WEEK
## Y                  1          2          3          4          5          6
##   delayed 0.21011673 0.13618677 0.13618677 0.13229572 0.16342412 0.04669261
##   ontime  0.12406015 0.14849624 0.14661654 0.17763158 0.17481203 0.12593985
##          DAY_WEEK
## Y                  7
##   delayed 0.17509728
##   ontime  0.10244361
##
## $tables$CRS_DEP_TIME
##          CRS_DEP_TIME
## Y                  6          7          8          9         10         11
##   delayed 0.04669261 0.03501946 0.06614786 0.01945525 0.02723735 0.01167315
##   ontime  0.05921053 0.06109023 0.07518797 0.05733083 0.04981203 0.03853383
##          CRS_DEP_TIME
## Y                 12         13         14         15         16         17
##   delayed 0.04669261 0.03891051 0.05058366 0.17509728 0.07782101 0.14785992
##   ontime  0.06860902 0.06766917 0.06484962 0.11278195 0.08270677 0.09962406
##          CRS_DEP_TIME
## Y                 18         19         20         21
##   delayed 0.02723735 0.09338521 0.02334630 0.11284047
##   ontime  0.04417293 0.04041353 0.01785714 0.06015038
##
## $tables$ORIGIN
##          ORIGIN
## Y                BWI        DCA        IAD
##   delayed 0.08949416 0.52529183 0.38521401
##   ontime  0.06109023 0.63251880 0.30639098
##
## $tables$DEST
##          DEST
## Y               EWR       JFK       LGA
##   delayed 0.3852140 0.1750973 0.4396887
##   ontime  0.2941729 0.1748120 0.5310150
##
## $tables$CARRIER
##          CARRIER
## Y                  CO          DH          DL          MQ          OH
##   delayed 0.081712062 0.334630350 0.085603113 0.178988327 0.003891051
##   ontime  0.037593985 0.240601504 0.184210526 0.123120301 0.018796992
##          CARRIER
## Y                  RU          UA          US
##   delayed 0.210116732 0.015564202 0.089494163
##   ontime  0.177631579 0.015977444 0.202067669
##
##
## $levels
```

```
## [1] "delayed" "ontime"
##
## $isnumeric
##     DAY_WEEK CRS_DEP_TIME       ORIGIN         DEST      CARRIER
##        FALSE        FALSE        FALSE        FALSE        FALSE
##
## $call
## naiveBayes.default(x = X, y = Y, laplace = laplace)
```

3. Output both a counts table and a proportion table outlining how many and what proportion of flights were delayed and on-time at each of the three airports.

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

```
##
##                 EWR       JFK       LGA
##   delayed 0.3852140 0.1750973 0.4396887
##   ontime  0.2941729 0.1748120 0.5310150
```

```
nb_pred <- predict(nb_delays, newdata = valid.df, type = "raw")
head(nb_pred)
```

```
##        delayed    ontime
## [1,] 0.0781994 0.9218006
## [2,] 0.1637595 0.8362405
## [3,] 0.1505714 0.8494286
## [4,] 0.2787990 0.7212010
## [5,] 0.1246673 0.8753327
## [6,] 0.3714953 0.6285047
```

```
## predict class membership
pred_class <- predict(nb_delays, newdata = valid.df)
df <- data.frame(actual = valid.df$Flight.Status, predicted = pred_class, nb_pred)
```

4. Output the confusion matrix and ROC for the validation data

```
library(caret)
# training
pred_class1 <- predict(nb_delays, newdata = train.df)
confusionMatrix(pred_class1, train.df$Flight.Status)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction delayed ontime
##    delayed      33     32
##    ontime      224   1032
##
##                Accuracy : 0.8062
##                  95% CI : (0.7838, 0.8272)
##     No Information Rate : 0.8055
```

```
##     P-Value [Acc > NIR] : 0.489
##
##                   Kappa : 0.1372
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.12840
##             Specificity : 0.96992
##          Pos Pred Value : 0.50769
##          Neg Pred Value : 0.82166
##              Prevalence : 0.19455
##          Detection Rate : 0.02498
##    Detection Prevalence : 0.04921
##       Balanced Accuracy : 0.54916
##
##        'Positive' Class : delayed
##
```

```
# validation
pred_class2 <- predict(nb_delays, newdata = valid.df)
confusionMatrix(pred_class2, valid.df$Flight.Status)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction delayed ontime
##    delayed      17     26
##    ontime      154    683
##
##                Accuracy : 0.7955
##                  95% CI : (0.7673, 0.8216)
##     No Information Rate : 0.8057
##     P-Value [Acc > NIR] : 0.7917
##
##                   Kappa : 0.0876
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.09942
##             Specificity : 0.96333
##          Pos Pred Value : 0.39535
##          Neg Pred Value : 0.81601
##              Prevalence : 0.19432
##          Detection Rate : 0.01932
##    Detection Prevalence : 0.04886
##       Balanced Accuracy : 0.53137
##
##        'Positive' Class : delayed
##
```

```
#AUC Value and ROC Curves III
library(pROC)
```

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

```
##
## Attaching package: 'pROC'

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

```
roc(valid.df$Flight.Status, nb_pred[,2])
```

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

## Setting direction: controls < cases

##
## Call:
## roc.default(response = valid.df$Flight.Status, predictor = nb_pred[,     2])
##
## Data: nb_pred[, 2] in 171 controls (valid.df$Flight.Status delayed) < 709 cases (valid.df$Flight.Sta
## Area under the curve: 0.6516
```

```
plot.roc(valid.df$Flight.Status, nb_pred[,2])
```

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