dslusser_3

David Slusser

10/18/2020

R Markdown

This is homework assignment number 3, using naive bayes to predict if a flight will be on time or not

```
library(readr) # Need to load data
library(dplyr) # Selecting variables
```

```
## Warning: package 'dplyr' was built under R version 4.0.2
```

```
##
## 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
```

```
library(caret) # For splitting into training and validation
```

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

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

```
## Warning: package 'ggplot2' was built under R version 4.0.2
```

```
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 4.0.2
```

```
library(e1071) # For the Naive Bayes model
```

```
## Warning: package 'e1071' was built under R version 4.0.2
```

```
library(gmodels) # For counts table
```

```
## Warning: package 'gmodels' was built under R version 4.0.2
```

```
library(pROC) # For the ROC plot
```

```
## 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
```

```
flights <- read_csv("~/Desktop/School/Graduate/Machine Learning/HW/Data/FlightDelays.cs
v")
```

```
## Parsed with column specification:
## cols(
##   CRS_DEP_TIME = col_double(),
##   CARRIER = col_character(),
##   DEP_TIME = col_double(),
##   DEST = col_character(),
##   DISTANCE = col_double(),
##   FL_DATE = col_character(),
##   FL_NUM = col_double(),
##   ORIGIN = col_character(),
##   Weather = col_double(),
##   DAY_WEEK = col_double(),
##   DAY_OF_MONTH = col_double(),
##   TAIL_NUM = col_character(),
##   `Flight Status` = col_character()
## )
```

```
# We need to make variables factors
flights$DAY_WEEK <- cut(flights$DAY_WEEK,c(-Inf, 1, 2, 3, 4, 5, 6, Inf), # Convert day n
umber to days
                        labels=c("Monday", "Tuesday", "Wednesday" ,"Thursday", "Friday",
"Saturday", "Sunday"))

flightsModel <- flights %>%
  select(CRS_DEP_TIME, CARRIER, DEST, ORIGIN, DAY_WEEK, `Flight Status`) %>%
  rename(Status = `Flight Status`)

# Convert departure time into numeruc
flightsModel$CRS_DEP_TIME <- as.factor(flightsModel$CRS_DEP_TIME)

# We want the count and preportions for each airport for delayed
flightsModel %>%
  select(Status, ORIGIN) %>% # Select the variables needed
  group_by(ORIGIN, Status) %>% # Group by origin and then status to get the amount delay
ed/on time for airport
  summarise(Count = n()) %>% # Count the number of flights delayed/ontime at each airpor
t
  mutate(Freq = Count / sum(Count)) # Get the frequency of flights delayed/ontime at eac
h airport
```

```
## `summarise()` regrouping output by 'ORIGIN' (override with `.groups` argument)
```

ORIGIN <chr>	**Status** <chr>	**Count** <int>	**Freq** <dbl>
BWI	delayed	37	0.2551724
BWI	ontime	108	0.7448276
DCA	delayed	221	0.1613139
DCA	ontime	1149	0.8386861
IAD	delayed	170	0.2478134
IAD	ontime	516	0.7521866

6 rows

At BWI (Baltimore-Washington), there were 145 flights with 37 (25.5%) delayed and 108 (74.5%) on time At DCA (Reagan National), there were 1370 flights with 221 (16.1%) delayed and 1149 (83.9%) on time At IAD (Dulles), there were 686 fligths with 170 (24.8%) delayed and 516 (75.2%) on time

Overall, there were 2,201 flights with 428 (19.4)% delayed and 1773 (80.6%) on time

```
# We need to create and build the naive bayes model
# Start with training and validation


# Set seed and divide the data into training (60%) and validation (40%)
set.seed(1234)
Index_Train <- createDataPartition(flightsModel$Status, p= 0.6, list = FALSE)
Train <- flightsModel[Index_Train,]
```

```
## Warning: The `i` argument of ``[`()` can't be a matrix as of tibble 3.0.0.
## Convert to a vector.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

```
Validation  <- flightsModel[-Index_Train,]

# Build a naive bayes classifier
nb_model <-naiveBayes(as.factor(Status) ~ as.factor(CRS_DEP_TIME) + CARRIER + DEST + ORI
GIN + DAY_WEEK,
                      data = Train) # This is the classifier with our categorical variab
les
nb_model # Produces the A-prori probabilities (Probabilities from deductive reasoning)
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##   delayed    ontime
## 0.1945496 0.8054504
##
## Conditional probabilities:
##          as.factor(CRS_DEP_TIME)
## Y                  600          630          640          645          700
##   delayed 0.0038910506 0.0038910506 0.0194552529 0.0038910506 0.0350194553
##   ontime  0.0140977444 0.0300751880 0.0075187970 0.0140977444 0.0404135338
##          as.factor(CRS_DEP_TIME)
## Y                  730          735          759          800          830
##   delayed 0.0077821012 0.0077821012 0.0000000000 0.0233463035 0.0077821012
##   ontime  0.0122180451 0.0065789474 0.0009398496 0.0159774436 0.0150375940
##          as.factor(CRS_DEP_TIME)
## Y                  840          845          850          900          925
##   delayed 0.0272373541 0.0000000000 0.0116731518 0.0311284047 0.0000000000
##   ontime  0.0244360902 0.0009398496 0.0150375940 0.0357142857 0.0018796992
##          as.factor(CRS_DEP_TIME)
## Y                  930         1000         1030         1039         1040
##   delayed 0.0038910506 0.0000000000 0.0272373541 0.0000000000 0.0038910506
##   ontime  0.0225563910 0.0122180451 0.0300751880 0.0028195489 0.0065789474
##          as.factor(CRS_DEP_TIME)
## Y                 1100         1130         1200         1230         1240
##   delayed 0.0116731518 0.0038910506 0.0000000000 0.0038910506 0.0194552529
##   ontime  0.0244360902 0.0122180451 0.0150375940 0.0159774436 0.0159774436
##          as.factor(CRS_DEP_TIME)
## Y                 1245         1300         1315         1330         1359
##   delayed 0.0466926070 0.0272373541 0.0038910506 0.0000000000 0.0116731518
##   ontime  0.0291353383 0.0545112782 0.0018796992 0.0084586466 0.0131578947
##          as.factor(CRS_DEP_TIME)
## Y                 1400         1430         1455         1500         1515
##   delayed 0.0116731518 0.0194552529 0.1011673152 0.0350194553 0.0077821012
##   ontime  0.0225563910 0.0206766917 0.0545112782 0.0328947368 0.0018796992
##          as.factor(CRS_DEP_TIME)
## Y                 1520         1525         1530         1600         1605
##   delayed 0.0000000000 0.0233463035 0.0233463035 0.0233463035 0.0000000000
##   ontime  0.0000000000 0.0065789474 0.0216165414 0.0169172932 0.0000000000
##          as.factor(CRS_DEP_TIME)
## Y                 1610         1630         1640         1645         1700
##   delayed 0.0038910506 0.0233463035 0.0116731518 0.0038910506 0.0233463035
##   ontime  0.0150375940 0.0281954887 0.0112781955 0.0159774436 0.0310150376
##          as.factor(CRS_DEP_TIME)
## Y                 1710         1715         1720         1725         1730
##   delayed 0.0116731518 0.0505836576 0.0311284047 0.0000000000 0.0194552529
##   ontime  0.0112781955 0.0197368421 0.0075187970 0.0009398496 0.0159774436
##          as.factor(CRS_DEP_TIME)
```

```
## Y                  1800         1830         1900         1930         2000
##   delayed 0.0038910506 0.0350194553 0.0700389105 0.0116731518 0.0155642023
##   ontime  0.0150375940 0.0272556391 0.0366541353 0.0093984962 0.0140977444
##          as.factor(CRS_DEP_TIME)
## Y                  2030         2100         2120         2130
##   delayed 0.0116731518 0.0116731518 0.0700389105 0.0000000000
##   ontime  0.0112781955 0.0178571429 0.0328947368 0.0009398496
##
##          CARRIER
## Y                 CO         DH         DL         MQ         OH         RU
##   delayed 0.03501946 0.33852140 0.12451362 0.15953307 0.01167315 0.23735409
##   ontime  0.03571429 0.22744361 0.19924812 0.11842105 0.01691729 0.18984962
##          CARRIER
## Y                 UA         US
##   delayed 0.01167315 0.08171206
##   ontime  0.01503759 0.19736842
##
##          DEST
## Y               EWR       JFK       LGA
##   delayed 0.3657588 0.2178988 0.4163424
##   ontime  0.2913534 0.1682331 0.5404135
##
##          ORIGIN
## Y                BWI        DCA        IAD
##   delayed 0.08560311 0.48249027 0.43190661
##   ontime  0.06484962 0.64473684 0.29041353
##
##          DAY_WEEK
## Y             Monday    Tuesday  Wednesday   Thursday     Friday   Saturday
##   delayed 0.20622568 0.13229572 0.13618677 0.14396887 0.17509728 0.04280156
##   ontime  0.13721805 0.12593985 0.14379699 0.15601504 0.19360902 0.13157895
##          DAY_WEEK
## Y             Sunday
##   delayed 0.16342412
##   ontime  0.11184211
```

```
# Predict the status of the flight using the model and the validation data set
predicted_status <-predict(nb_model, Validation) # this provides the predicted label (de
layed/on time) where
                                           # P > 0.5 is the cutoff

# Show the confusion matrix of the flight status
CrossTable(x = Validation$Status, y = predicted_status, prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  880
##
##
##                   | predicted_status
## Validation$Status |   delayed |    ontime | Row Total |
## ------------------|-----------|-----------|-----------|
##           delayed |         7 |       164 |       171 |
##                   |     0.041 |     0.959 |     0.194 |
##                   |     0.259 |     0.192 |           |
##                   |     0.008 |     0.186 |           |
## ------------------|-----------|-----------|-----------|
##            ontime |        20 |       689 |       709 |
##                   |     0.028 |     0.972 |     0.806 |
##                   |     0.741 |     0.808 |           |
##                   |     0.023 |     0.783 |           |
## ------------------|-----------|-----------|-----------|
##      Column Total |        27 |       853 |       880 |
##                   |     0.031 |     0.969 |           |
## ------------------|-----------|-----------|-----------|
##
##
```

```
# For the PROC we want the probabilities of each, so the predicted status we need the ra
w probabilities
predicted_status_prob <-predict(nb_model, Validation, type = "raw") # type = "raw" provi
des prob of each
head(predicted_status_prob) # gives first couple rows of the predicted prob. delayed is
 column 1 and
```

```
##        delayed    ontime
## [1,] 0.2082043 0.7917957
## [2,] 0.2433158 0.7566842
## [3,] 0.2754174 0.7245826
## [4,] 0.2754174 0.7245826
## [5,] 0.3898837 0.6101163
## [6,] 0.3898837 0.6101163
```

```
                            # on time is column 2

# Get the ROC curve
# This uses the pROC package
roc(Validation$Status, predicted_status_prob[,2]) # Prob that the flight is ontime
```

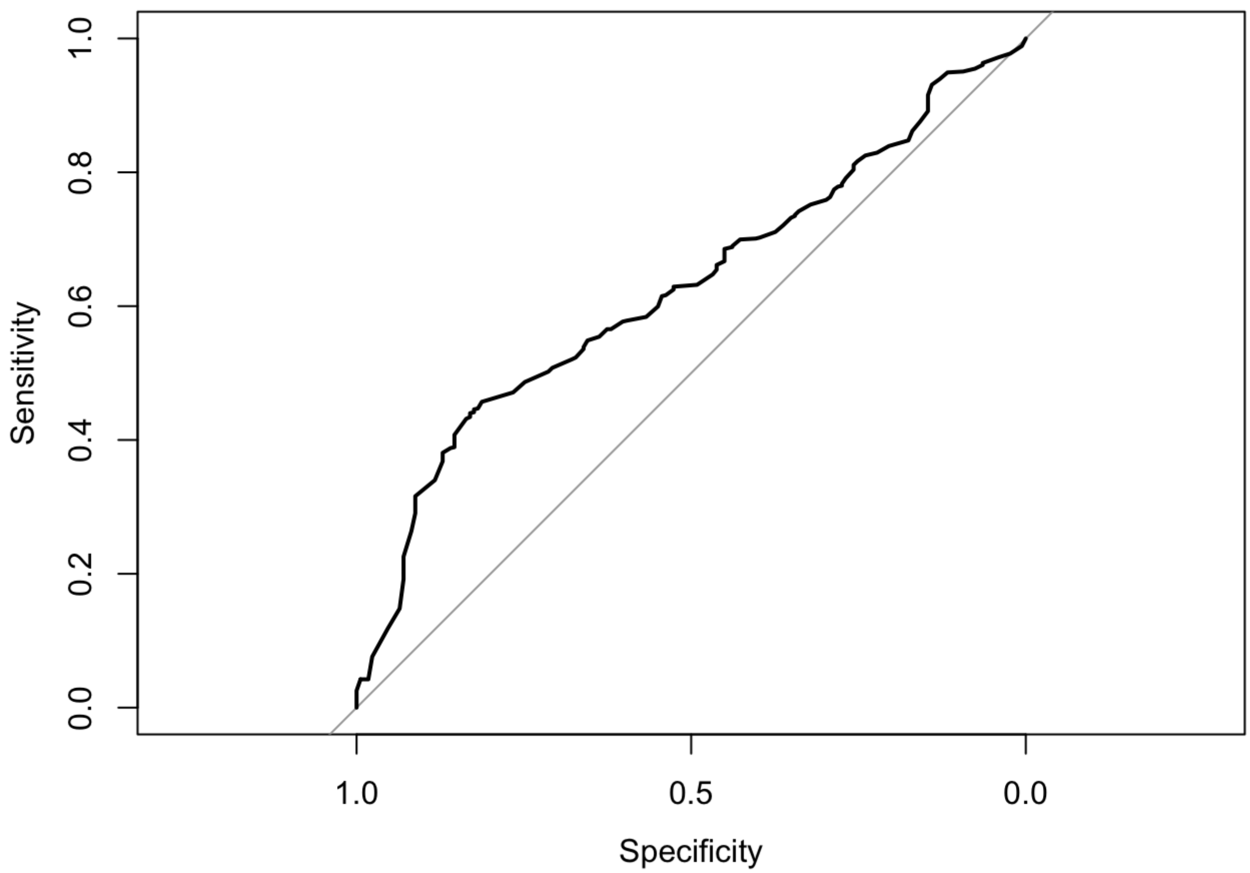
```
## Setting levels: control = delayed, case = ontime
```

```
## Setting direction: controls < cases
```

```
##
## Call:
## roc.default(response = Validation$Status, predictor = predicted_status_prob[,     2])
##
## Data: predicted_status_prob[, 2] in 171 controls (Validation$Status delayed) < 709 ca
ses (Validation$Status ontime).
## Area under the curve: 0.626
```

```
plot.roc(Validation$Status, predicted_status_prob[,2]) # Plot of curve that the flight i
s ontime
```

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



This cross table shows us that we have 184 misclassifications

We find the Area under the curve (AUC) is 62.6%

Better curves are closer to the top left corner, but as our ROC is closer to the dashed line, our model is not as strong (also seen with the lower AUC)