DATA 621 - Homework 2

Fall 2020 - Business Analytics and Data Mining

Mohamed Thasleem, Kalikul Zaman

10/11/2020

Contents

Introduction

1. Data Download

Download the classification output data set

```
# Read in the data
file <- "https://raw.githubusercontent.com/mohamedthasleem/DATA621/master/HW2/classification-output-data
data <- read.csv(file)
head(data, 5)
```

```
##   pregnant glucose diastolic skinfold insulin  bmi pedigree age class
## 1        7     124        70       33     215 25.5    0.161  37     0
## 2        2     122        76       27     200 35.9    0.483  26     0
```

```
## 3        3     107        62       13      48 22.9    0.678  23     1
## 4        1      91        64       24       0 29.2    0.192  21     0
## 5        4      83        86       19       0 29.3    0.317  34     0
##   scored.class scored.probability
## 1            0         0.32845226
## 2            0         0.27319044
## 3            0         0.10966039
## 4            0         0.05599835
## 5            0         0.10049072
```

```
summary(data)
```

```
##     pregnant         glucose        diastolic        skinfold
##  Min.   : 0.000   Min.   : 57.0   Min.   : 38.0   Min.   : 0.0
##  1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 64.0   1st Qu.: 0.0
##  Median : 3.000   Median :112.0   Median : 70.0   Median :22.0
##  Mean   : 3.862   Mean   :118.3   Mean   : 71.7   Mean   :19.8
##  3rd Qu.: 6.000   3rd Qu.:136.0   3rd Qu.: 78.0   3rd Qu.:32.0
##  Max.   :15.000   Max.   :197.0   Max.   :104.0   Max.   :54.0
##     insulin            bmi           pedigree           age
##  Min.   :  0.00   Min.   :19.40   Min.   :0.0850   Min.   :21.00
##  1st Qu.:  0.00   1st Qu.:26.30   1st Qu.:0.2570   1st Qu.:24.00
##  Median :  0.00   Median :31.60   Median :0.3910   Median :30.00
##  Mean   : 63.77   Mean   :31.58   Mean   :0.4496   Mean   :33.31
##  3rd Qu.:105.00   3rd Qu.:36.00   3rd Qu.:0.5800   3rd Qu.:41.00
##  Max.   :543.00   Max.   :50.00   Max.   :2.2880   Max.   :67.00
##      class         scored.class    scored.probability
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.02323
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.11702
##  Median :0.0000   Median :0.0000   Median :0.23999
##  Mean   :0.3149   Mean   :0.1768   Mean   :0.30373
##  3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:0.43093
##  Max.   :1.0000   Max.   :1.0000   Max.   :0.94633
```

2. Confusion matrix

The data set has three key columns we will use: * **class:** the actual class for the observation * **scored.class:** the predicted class for the observation (based on a threshold of 0.5) * **scored.probability:** the predicted probability of success for the observation

Use the table() function to get the raw confusion matrix for this scored dataset. Make sure you understand the output. In particular, do the rows represent the actual or predicted class? The columns?

```
#raw confusion matrix
data %>% select(scored.class, class) %>%
  table()
```

```
##             class
## scored.class   0   1
##            0 119  30
##            1   5  27
```

3. Accuracy

Write a function that takes the data set as a dataframe, with actual and predicted classifications identified, and returns the accuracy of the predictions.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

```
accurary <- function(x){
  TP <- sum(x$class == 1 & x$scored.class == 1)
  TN <- sum(x$class == 0 & x$scored.class == 0)
  (TP + TN)/nrow(x)
}
accurary(data)
```

```
## [1] 0.8066298
```

4. Error rate

Write a function that takes the data set as a dataframe, with actual and predicted classifications identified, and returns the classification error rate of the predictions.

$$Classification\ Error\ Rate = \frac{FP + FN}{TP + FP + TN + FN}$$

Verify that you get an accuracy and an error rate that sums to one.

```
class_error_rate <- function(x){
  FP <- sum(x$class == 0 & x$scored.class == 1)
  FN <- sum(x$class == 1 & x$scored.class == 0)
  (FP + FN)/nrow(x)
}
class_error_rate(data)
```

```
## [1] 0.1933702
```

```
#Verify that you get an accuracy and an error rate that sums to one
accurary(data) + class_error_rate(data)
```

```
## [1] 1
```

5. Precision

Write a function that takes the data set as a dataframe, with actual and predicted classifications identified, and returns the precision of the predictions.

$$Precision = \frac{TP}{TP + FP}$$

```r
precision <- function(x){
  TP <- sum(x$class == 1 & x$scored.class == 1)
  FP <- sum(x$class == 0 & x$scored.class == 1)
  TP/(TP + FP)
}
precision(data)
```

```
## [1] 0.84375
```

6. Sensitivity

Write a function that takes the data set as a dataframe, with actual and predicted classifications identified, and returns the sensitivity of the predictions. Sensitivity is also known as recall.

$$Sensitivity = \frac{TP}{TP + FN}$$

```r
sensitivity <- function(x){
  TP <- sum(x$class == 1 & x$scored.class == 1)
  FN <- sum(x$class == 1 & x$scored.class == 0)
  TP/(TP + FN)
}
sensitivity(data)
```

```
## [1] 0.4736842
```

7. Specificity

Write a function that takes the data set as a dataframe, with actual and predicted classifications identified, and returns the specificity of the predictions.

$$Specificity = \frac{TN}{TN + FP}$$

```r
specificity <- function(x){
  TN <- sum(x$class == 0 & x$scored.class == 0)
  FP <- sum(x$class == 0 & x$scored.class == 1)
  TN/(TN + FP)
}
specificity(data)
```

```
## [1] 0.9596774
```

8. F1 Score

Write a function that takes the data set as a dataframe, with actual and predicted classifications identified, and returns the F1 score of the predictions.

$$F1\,Score = \frac{2 \times Precision \times Sensitivity}{Precision + Sensitivity}$$

```r
f1_score <- function(x){
  (2*precision(x)*sensitivity(x))/(precision(x)+sensitivity(x))
}
f1_score(data)
```

```
## [1] 0.6067416
```

9. F1 Score boundry

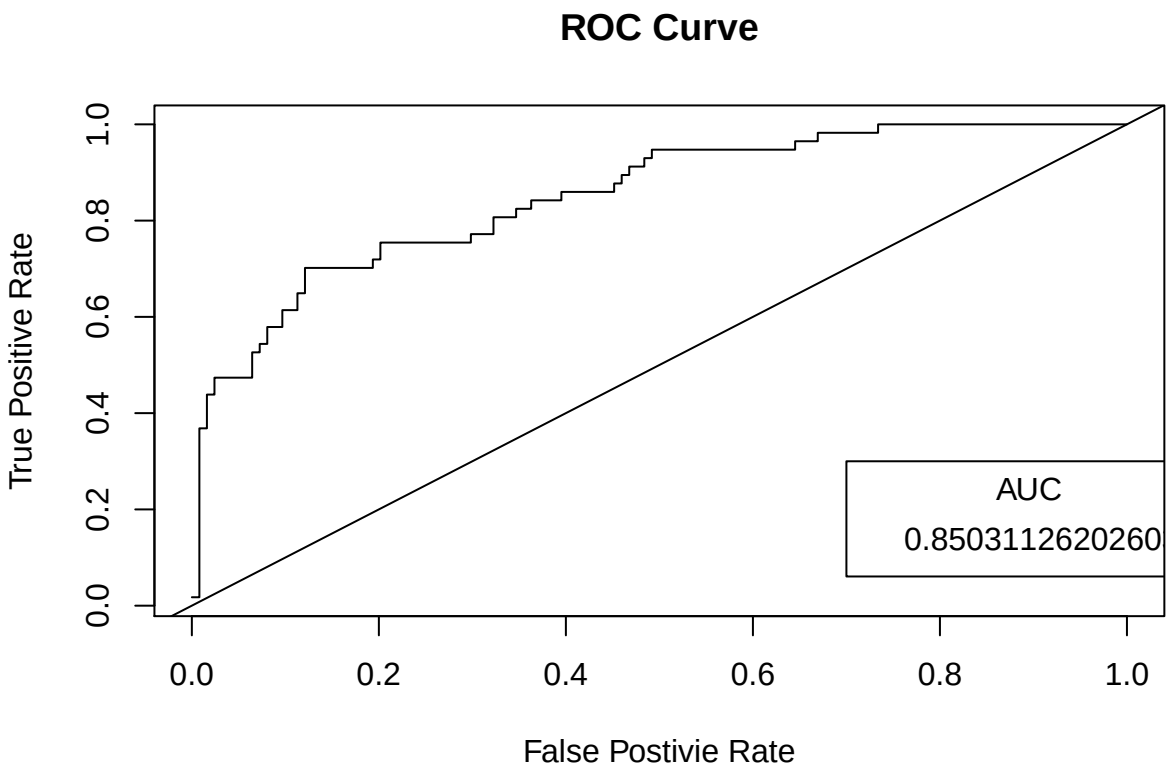
Before we move on, let's consider a question that was asked: What are the bounds on the F1 score? Show that the F1 score will always be between 0 and 1.

If the False Positive (FP) or False Negative (FN) is close to 1, the F1 Score will be close to 0,In other hand if the False Positive(FP) and False Negative (FN) are close to 0 , then the F1 score would be close to 1. Both Precision and Sensitivity used to calculate F1 score are bounded between 0 and 1 and the score will always be close to 0 and 1

10. ROC curve

Write a function that generates an ROC curve from a data set with a true classification column (class in our example) and a probability column (scored.probability in our example). Your function should return a list that includes the plot of the ROC curve and a vector that contains the calculated area under the curve (AUC). Note that I recommend using a sequence of thresholds ranging from 0 to 1 at 0.01 intervals.

```r
roc <- function(x, y){
  x <- x[order(y, decreasing = TRUE)]
  TPR <- cumsum(x) / sum(x)
  FPR <- cumsum(!x) / sum(!x)
  xy <- data.frame(TPR, FPR, x)
  FPR1 <- c(diff(xy$FPR), 0)
  TPR1 <- c(diff(xy$TPR), 0)
  AUC <- sum(xy$TPR * FPR1) + sum(TPR1 * FPR1)/2
  plot(xy$FPR, xy$TPR, type = "l", main = "ROC Curve", xlab = "False Postivie Rate"
       , ylab = "True Positive Rate")
  abline(a = 0, b = 1)
  legend(.7, .3, AUC, title = "AUC")
}

roc(data$class,data$scored.probability)
```



11. Classification metrics

Use your created R functions and the provided classification output data set to produce all of the classification metrics discussed above.

```
metrics <- c(accurary(data), class_error_rate(data), precision(data), sensitivity(data),
             specificity(data), f1_score(data))
names(metrics) <- c("Accuracy", "CFR", "Precision", "Sensitivity", "Specificity", "F1 Score")
metrics
```

```
##    Accuracy         CFR   Precision Sensitivity Specificity    F1 Score
##   0.8066298   0.1933702   0.8437500   0.4736842   0.9596774   0.6067416
```

12. Caret package

Investigate the caret package. In particular, consider the functions confusionMatrix, sensitivity, and specificity. Apply the functions to the data set. How do the results compare with your own functions?

```
b <- data %>%
  select(scored.class, class) %>%
  mutate(scored.class = as.factor(scored.class),
         class = as.factor(class))
```

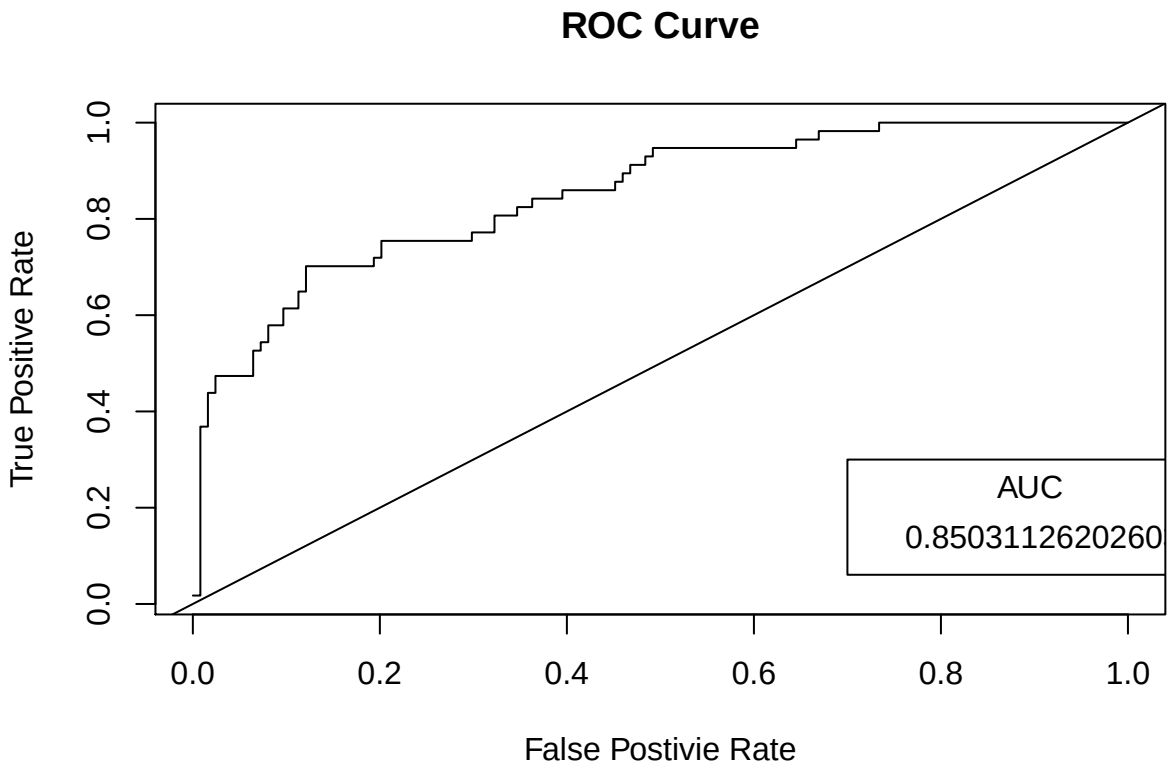
```r
c <- confusionMatrix(b$scored.class, b$class, positive = "1")

caret_package <- c(c$overall["Accuracy"], c$byClass["Sensitivity"], c$byClass["Specificity"])
function1 <- c(accurary(data), sensitivity(data), specificity(data))
d <- cbind(caret_package, function1)
d
```

```
##             caret_package function1
## Accuracy        0.8066298 0.8066298
## Sensitivity     0.4736842 0.4736842
## Specificity     0.9596774 0.9596774
```

13. pROC package

Investigate the pROC package. Use it to generate an ROC curve for the data set. How do the results compare with your own functions?

```r
#The results same
roc(data$class,data$scored.probability)
```



It appears to be both in-built pROC and own function has almost similar output value.