Assignment-2

Anil Akyildirim, John K. Hancock, John Suh, Emmanuel Hayble-Gomes, Chunjie Nan

2/29/2020

Contents

Overview

In this homework assignment, you will work through various classification metrics. You will be asked to create functions in R to carry out the various calculations. You will also investigate some functions in packages that will let you obtain the equivalent results. Finally, you will create graphical output that also can be used to evaluate the output of classification models, such as binary logistic regression.

```
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
```

```
library(tidyr)
library(knitr)
library(zoo)
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

1.Download the classification output data set.

```r
git <- "https://raw.githubusercontent.com/nancunjie4560/Data621/master/classification-output-data.csv"
data <- read.csv(git)
head(data, 10)
```

```
##    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
## 6         1     100        74       12      46 19.5    0.149  28     0
## 7         9      89        62        0       0 22.5    0.142  33     0
## 8         8     120        78        0       0 25.0    0.409  64     0
## 9         1      79        60       42      48 43.5    0.678  23     0
## 10        2     123        48       32     165 42.1    0.520  26     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
## 6             0         0.05515460
## 7             0         0.10711542
## 8             0         0.45994744
## 9             0         0.11702368
## 10            0         0.31536320
```

2. 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?

```r
rcm<-table(data$scored.class,data$class)[2:1,2:1]
rcm
```

```
##
##       1   0
##   1  27   5
##   0  30 119
```

As the table shows, the rows are predicted classes and the comlumns are actual classes.

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

```r
predict_accuracy<- function(x){
a <- sum(x$class == 1 & x$scored.class == 1)
d <- sum(x$class == 0 & x$scored.class == 0)
  (a + d)/nrow(x)}
predict_accuracy(data)
```

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

4.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.

```r
predict_error_rate<-function(x){
b<-sum(data$class == 0 & data$scored.class == 1)
c<-sum(data$class == 1 & data$scored.class == 0)
  (b + c)/nrow(data)}
predict_error_rate(data)
```

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

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

```r
predict_precision<-function(x){
a <- sum(data$class == 1 & data$scored.class == 1)
b<-sum(data$class == 0 & data$scored.class == 1)
  a/(a+b)}
predict_precision(data)
```

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

6. 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.

```r
predict_sensitivity<-function(x){
a <- sum(data$class == 1 & data$scored.class == 1)
c<-sum(data$class == 1 & data$scored.class == 0)
  a/(a+c)}
predict_sensitivity(data)
```

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

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

```
predict_specificity<-function(x){
d <- sum(data$class == 0 & data$scored.class == 0)
b<-sum(data$class == 0 & data$scored.class == 1)
  d/(d+b)}
predict_specificity(data)
```

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

8. 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.

```
predict_F1Score<-function(x){
(2*predict_precision(x)*predict_sensitivity(x))/(predict_precision(x)+predict_sensitivity(x))}
predict_F1Score(data)
```

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

9. 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.

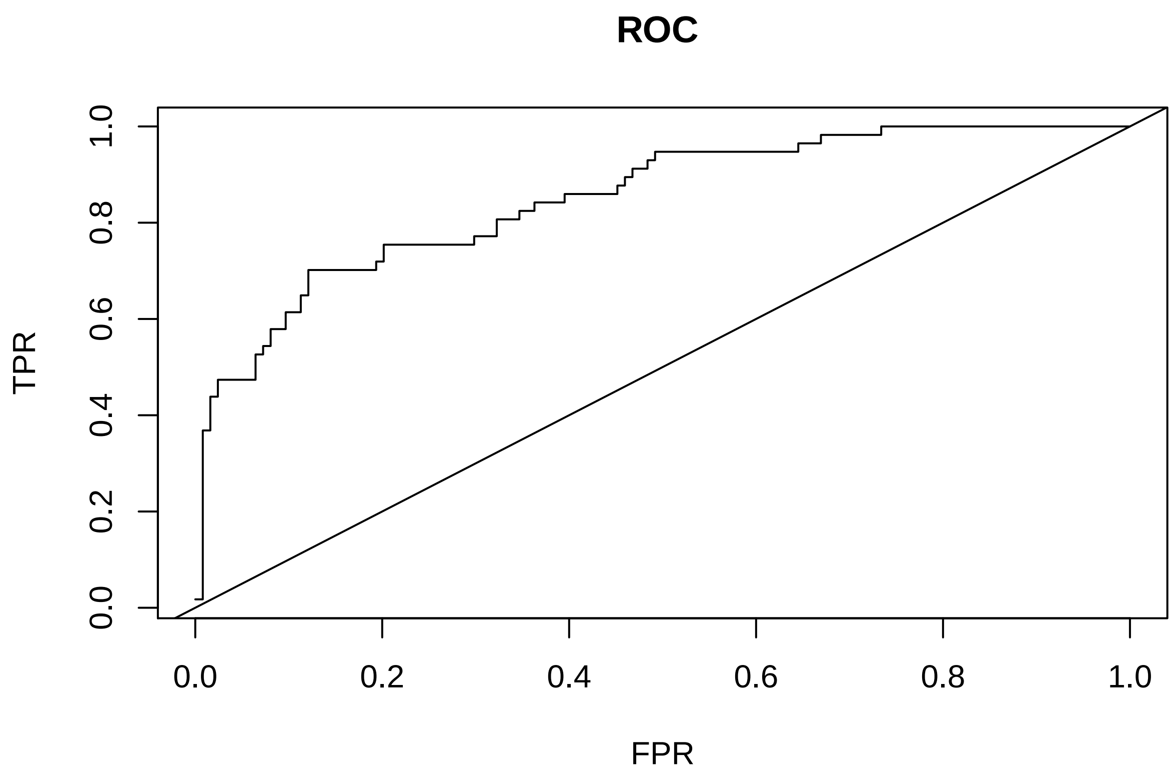
We calculate the F1 socre with precision and the sensitivity. The precision and the sensitivity are bounded between 0 and 1, and any calculation with numbers bounded between 0 and 1 is also results bounded between 0 and 1. Therefore, F1 score will be between 0 and 1.

10. 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.

```
ROC <- function(x, y){
  x <- x[order(y, decreasing = TRUE)]
  TPR <- cumsum(x) / sum(x)
  FPR <- cumsum(!x) / sum(!x)
  df <- data.frame(TPR, FPR, x)

  FPR_df <- c(diff(df$FPR), 0)
  TPR_df <- c(diff(df$TPR), 0)
  area_under_curve <- sum(df$TPR * FPR_df) + sum(TPR_df * FPR_df)/2
  print(area_under_curve)

  plot(df$FPR, df$TPR, type = "l",
       main = "ROC ",
       xlab = "FPR",
       ylab = "TPR")
  abline(a = 0, b = 1)

}
```

```r
ROC(data$class,data$scored.probability)
```

```
## [1] 0.8503113
```



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

```r
all_metrics <- c(predict_accuracy(data), predict_error_rate(data), predict_precision(data), predict_sens

names(all_metrics) <- c("Accuracy", "Error Rate", "Precision", "Sensitivity", "Specificity", "F1 Score"]

kable(all_metrics, col.names = "Metrics")
```

	Metrics
Accuracy	0.8066298
Error Rate	0.1933702
Precision	0.8437500
Sensitivity	0.4736842
Specificity	0.9596774
F1 Score	0.6067416

12. 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?

```
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```
confusionMatrix(rcm)
```

```
## Confusion Matrix and Statistics
##
##
##       1   0
##   1  27   5
##   0  30 119
##
##                Accuracy : 0.8066
##                  95% CI : (0.7415, 0.8615)
##     No Information Rate : 0.6851
##     P-Value [Acc > NIR] : 0.0001712
##
##                   Kappa : 0.4916
##
##  Mcnemar's Test P-Value : 4.976e-05
##
##             Sensitivity : 0.4737
##             Specificity : 0.9597
##          Pos Pred Value : 0.8438
##          Neg Pred Value : 0.7987
##              Prevalence : 0.3149
##          Detection Rate : 0.1492
##    Detection Prevalence : 0.1768
##       Balanced Accuracy : 0.7167
##
##        'Positive' Class : 1
##
```

```
predict_sensitivity(data)
```

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

```
predict_specificity(data)
```

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

According to the Confusion Matrix, we can conclude that the two results are matched.

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

```
library(pROC)
```

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

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

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

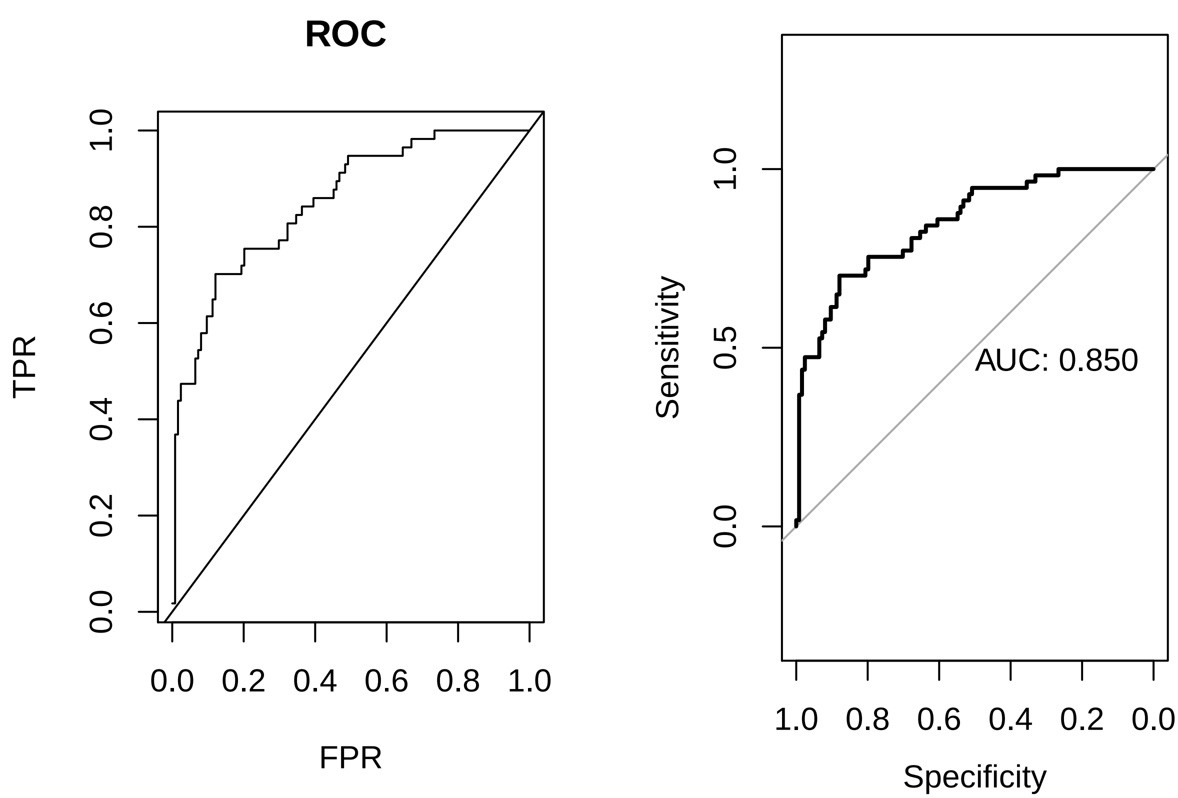
```
par(mfrow=c(1,2))
ROC(data$class, data$scored.probability)
```

```
## [1] 0.8503113
```

```
plot(roc(data$class,data$scored.probability),print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

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



According to the graph above, it shows the results are the same for ROC with 0.850.