Data621_HW2

William Outcault, Kevin Potter, Mengqin Cai, Philip Tanofsky, Robert Welk, Zhi Ying Chen

10/6/2020

```
library(ggplot2)
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.5.3
```

```
## -- Attaching packages ---------------------------------------------------------- tidyverse 1.3.0
```

```
## v tibble  3.0.1     v dplyr   0.8.5
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
## v purrr   0.3.4
```

```
## Warning: package 'tibble' was built under R version 3.5.3
```

```
## Warning: package 'tidyr' was built under R version 3.5.3
```

```
## Warning: package 'readr' was built under R version 3.5.3
```

```
## Warning: package 'purrr' was built under R version 3.5.3
```

```
## Warning: package 'dplyr' was built under R version 3.5.3
```

```
## Warning: package 'forcats' was built under R version 3.5.3
```

```
## -- Conflicts ------------------------------------------------------------------- tidyverse_conflicts()
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(dplyr)
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.5.3
```

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

```
## Warning: package 'lattice' was built under R version 3.5.3
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(pROC)
```

```
## Warning: package 'pROC' was built under R version 3.5.3
```

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

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

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

```
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 3.5.3
```

Questions 1

Download the classification output data set (attached in Blackboard to the assignment).

```
df<-read.csv("https://raw.githubusercontent.com/DaisyCai2019/NewData/master/classification-output-data.
head(data)
```

```
##
## 1 function (..., list = character(), package = NULL, lib.loc = NULL,
## 2     verbose = getOption("verbose"), envir = .GlobalEnv)
## 3 {
## 4     fileExt <- function(x) {
## 5         db <- grepl("\\\\.[^.]+\\\\.(gz|bz2|xz)$", x)
## 6         ans <- sub(".*\\\\.", "", x)
```

Questions 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?

```
df.subset <- df %>% select(class, scored.class, scored.probability)
df.subset$class <- as.factor(df.subset$class)
df.subset$scored.class <- as.factor(df.subset$scored.class)
```

After setting the column values to factors we can use the table function with the actual class values shown in the rows and predicted class values shown in the columns. The threshold used to make the predictions was 0.5.

```
table(df.subset$scored.class,df.subset$class)
```

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

The row represent the predicted data and columns represent the actual data.

Questions 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.

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

```
calculate_prediction_accuracy <- function(df){
tabl <- table(df$class, df$scored.class)
num <- tabl[1]+tabl[4]
den <- sum(tabl)
return(num/den)
}
calculate_prediction_accuracy(df)
```

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

Questions 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.

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

```
calculate_class_error_rate <- function(df){
tabl <- table(df$class, df$scored.class)
num <- tabl[2]+tabl[3]
den <- sum(tabl)
return(num/den)
}
calculate_class_error_rate(df)
```

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

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

```
calculate_class_error_rate(df) + calculate_prediction_accuracy(df) == 1
```

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

Questions 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.

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

```
calculate_prediction_precision <- function(df){
tabl <- table(df$class, df$scored.class)
num <- tabl[4]
den <- tabl[4] + tabl[3]
return(num/den)
}
calculate_prediction_precision(df)
```

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

Questions 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.

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

```
calculate_prediction_sensitivity <- function(df){
tabl <- table(df$class, df$scored.class)
```

```
num <- tabl[4]
den <- tabl[4] + tabl[2]
return(num/den)
}
calculate_prediction_sensitivity(df)
```

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

Questions 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.

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

```
calculate_prediction_specificity <- function(df){
tabl <- table(df$class, df$scored.class)
num <- tabl[1]
den <- tabl[1] + tabl[3]
return(num/den)
}
calculate_prediction_specificity(df)
```

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

Questions 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.

$F1Score = \frac{2*Precision*Sensitivity}{Precision+Sensitivity}$

```
f1_score <- function(df){
(2*calculate_prediction_precision(df)*calculate_prediction_sensitivity(df))/(calculate_prediction_preci
}
f1_score(df)
```

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

Questions 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.(Hint: If 0 0

[1] TRUE

‘‘‘

```
max(z) < 1
```

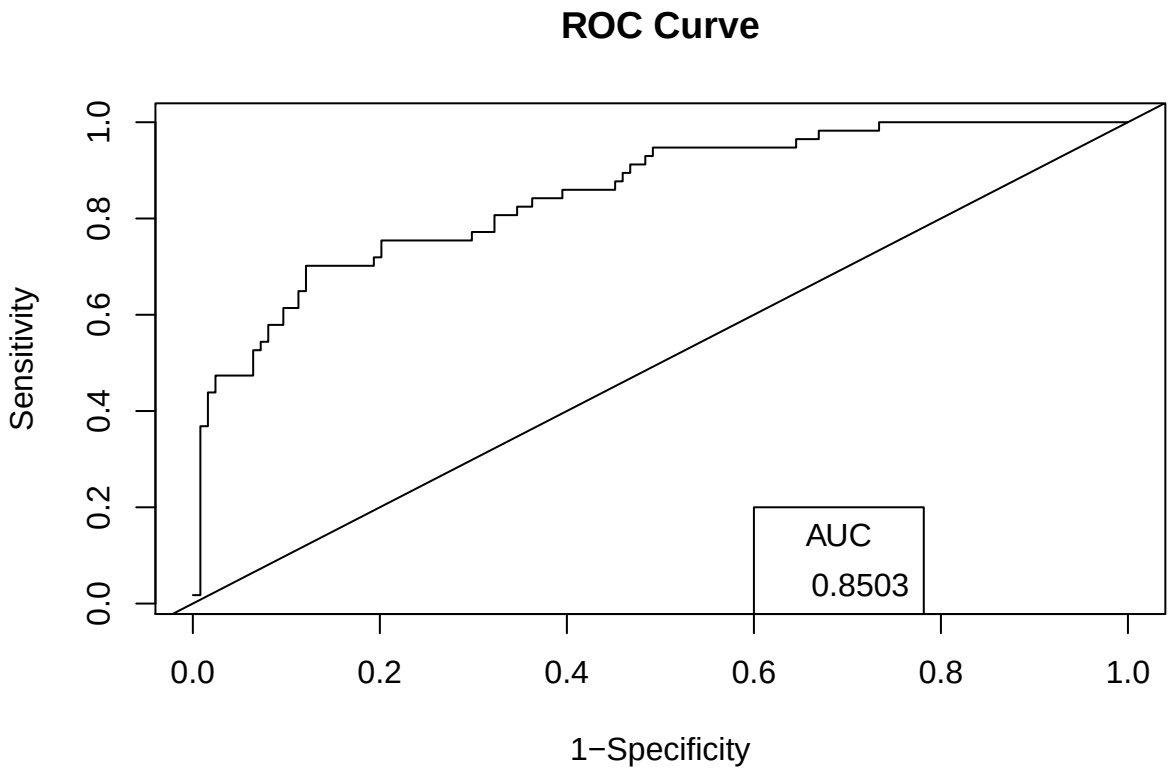
```
## [1] TRUE
```

The result show the max of F1 Score is less than 1 and larger than 0.

Questions 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(class, probability){
class = class[order(probability, decreasing=TRUE)]
result =data.frame(TPR=cumsum(class)/sum(class), FPR=cumsum(!class)/sum(!class), class)
FPR_df = c(diff(result$FPR), 0)
TPR_df = c(diff(result$TPR), 0)
AUC = round(sum(result$TPR * FPR_df) + sum(TPR_df * FPR_df)/2,4)
plot(result$FPR,result$TPR,type="l",main ="ROC Curve",ylab="Sensitivity",xlab="1-Specificity")
abline(a=0,b=1)
legend(.6,.2,AUC,title = "AUC")
}
ROC(df$class,df$scored.probability)
```



Questions 11

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

```
metrics <- c(calculate_prediction_accuracy(df), calculate_class_error_rate(df), calculate_prediction_pr
names(metrics) <- c("Accuracy", "Classification Error Rate", "Precision", "Sensitivity", "Specificity",
kable(metrics, col.names = "Metrics")
```

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

Questions 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(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.5.3
```

```
df$scored.class <- as.factor(df$scored.class)
df$class <- as.factor(df$class)
```

```
confusionMatrix(df$scored.class, df$class, mode = 'everything')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 119  30
##          1   5  27
##
##                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 : 0.9597
##             Specificity : 0.4737
##          Pos Pred Value : 0.7987
##          Neg Pred Value : 0.8438
##               Precision : 0.7987
##                  Recall : 0.9597
##                      F1 : 0.8718
##              Prevalence : 0.6851
##          Detection Rate : 0.6575
##    Detection Prevalence : 0.8232
##       Balanced Accuracy : 0.7167
##
##        'Positive' Class : 0
##
```

```
sensitivity(df$scored.class, df$class)
```

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

```
specificity(df$scored.class, df$class)
```

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

The values from the built in function are almost the same with the function we created before.

Questions 13

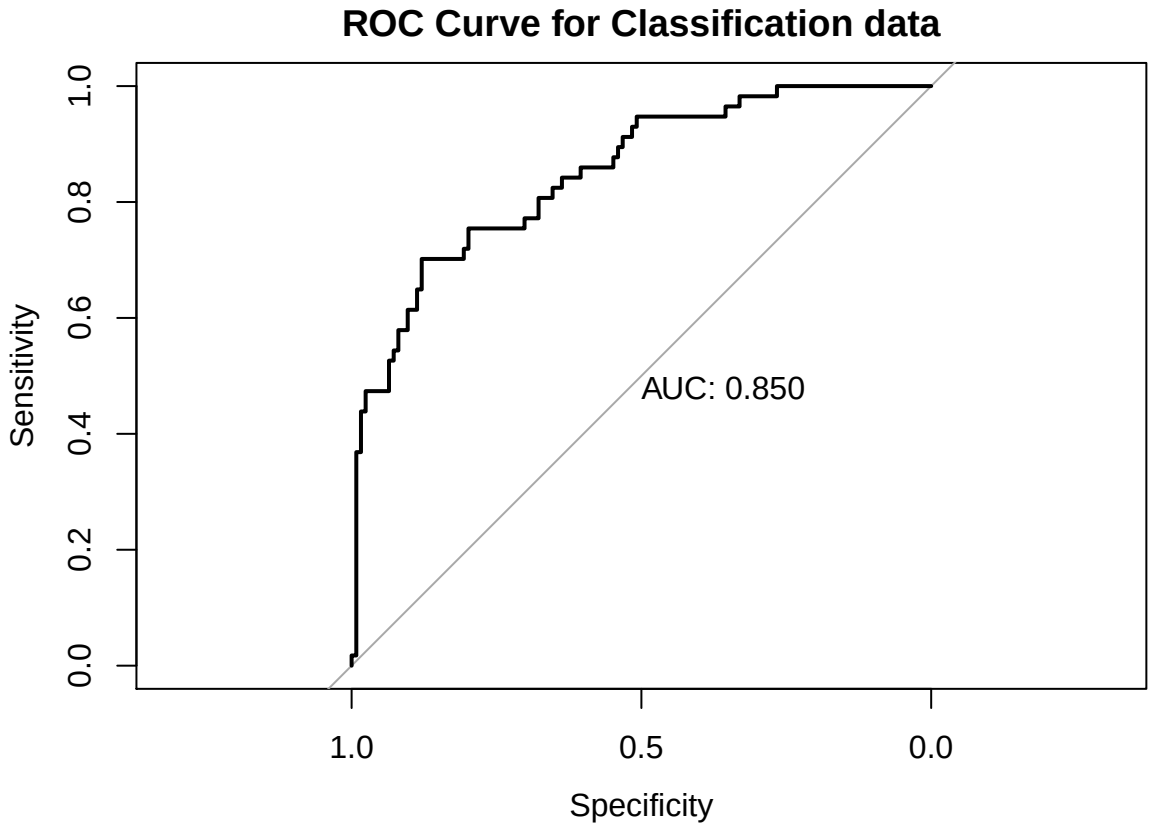
Investigate the pROC package. Use it to generate an ROC curve for the data set. How do the results compare with your own functions?

```
roc_new <- roc(df$class, df$scored.probability)
```

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

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

```
plot(roc_new, main="ROC Curve for Classification data", print.auc = TRUE)
```



The ROC graphs from the pROC package and the one we generate are very similar.