CUNY SPS
DATA 621 – Business Analytics and Data Mining
Homework# 2

Ahsanul Choudhury

March 18, 2018

Introduction

This is the homework 2 for DATA 621 on various classification metrics. For this homework we will be creating functions in R to carry out various calculations. We will also investigate some of the functions in packages such as **caret** and **pROC** to obtain the equivalent results. Finally, we will create graphical output that also can be used to evaluate the output of classification models, such as binary logistic regression. R code will be provided in Appendix.

Problem 1:

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

R function:

```
my_data <- read.csv('classification-output-data.csv', header=T)
head(my_data)
```

Problem 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?

A raw confusion matrix was created using R code, please see table on the next page.

R code:

```
t <- table(my_data$scored.class, my_data$class)
knitr::kable(t)
```

	0	1
0	119	30
1	5	27

The rows represent the predicted class and the columns represent the actual class.

Problem 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.

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

```
a_fun <- function(df){
  df <- as.data.frame(table(my_data$scored.class, my_data$class))
  tp <- df$Freq[4]; fp <- df$Freq[2]; fn <- df$Freq[3]; tn <- df$Freq[1]
  accuracy <- ((tp + tn)/(tp + fp + tn + fn))
  return (accuracy)
}
a_fun(df)
```

Accuracy = 0.8066298

Problem 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.

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

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

R code:

```
err_fun <- function(df){
  df <- as.data.frame(table(my_data$scored.class, my_data$class))
  tp <- df$Freq[4]; fp <- df$Freq[2]; fn <- df$Freq[3]; tn <- df$Freq[1]
  error <- ((fp + fn)/(tp + fp + tn + fn))
  return (error)
}
err_fun(df)
```

Classification Error Rate = 0.1933702

Verification:

Accuracy + Classification Error Rate = 0.8066298 + 0.1933702 = 1

Problem 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.

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

R function:

```
prec_fun <- function(df){
  df <- as.data.frame(table(my_data$scored.class, my_data$class))
  tp <- df$Freq[4]; fp <- df$Freq[2]
  prec <- tp/(tp + fp)
  return (prec)
}
prec_fun(df)
```

Precision = 0.84375

Problem 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.

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

R code:

```
sen_fun <- function(df){
  df <- as.data.frame(table(my_data$scored.class, my_data$class))
  tp <- df$Freq[4]; fn <- df$Freq[3]
  sen <- tp/(tp + fn)
  return (sen)
}
sen_fun(df)
```

Sensitivity = 0.4736842

Problem 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.

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

R code:

```
spec_fun <- function(df){
  df <- as.data.frame(table(my_data$scored.class, my_data$class))
  fp <- df$Freq[2]; tn <- df$Freq[1]
  spec <- tn/(tn + fp)
  return (spec)
}
spec_fun(df)
```

Specificity = 0.9596774

Problem 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.

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

R code:

```
f1_fun <- function(df){
  df <- as.data.frame(table(my_data$scored.class, my_data$class))
  tp <- df$Freq[4]; fp <- df$Freq[2]; fn <- df$Freq[3]
  prec <- tp/(tp + fp); sen <- tp/(tp + fn)
  f1 <- (2 * prec * sen) / (prec + sen)
  return (f1)
}
f1_fun(df)
```

F1 Score = 0.6067416

Problem 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 < a < 1 and 0 < b < 1 then ab < a.)

Since F1 score is a function of precision and sensitivity which are themselves bounded between 0 and 1, the solution to equation, F1 Score = $\frac{2 \times Precision \times Sensitivity}{Precision + Sensitivity}$ will always be between 0 and 1.

Example in R code:

```
set.seed(1023)
prec <- runif(50, 0, 1)
sen <- runif(50, 0, 1)
f1 <- (2 * prec * sen) / (prec + sen)
result <- summary(f1)
c(result[1], result[6])
```

We run a function 50 times with random precision and sensitivity between 0 and 1 and got following minimum and maximum for the equation which are between 0 and 1.

```
   Min.    Max.
0.03142 0.92510
```

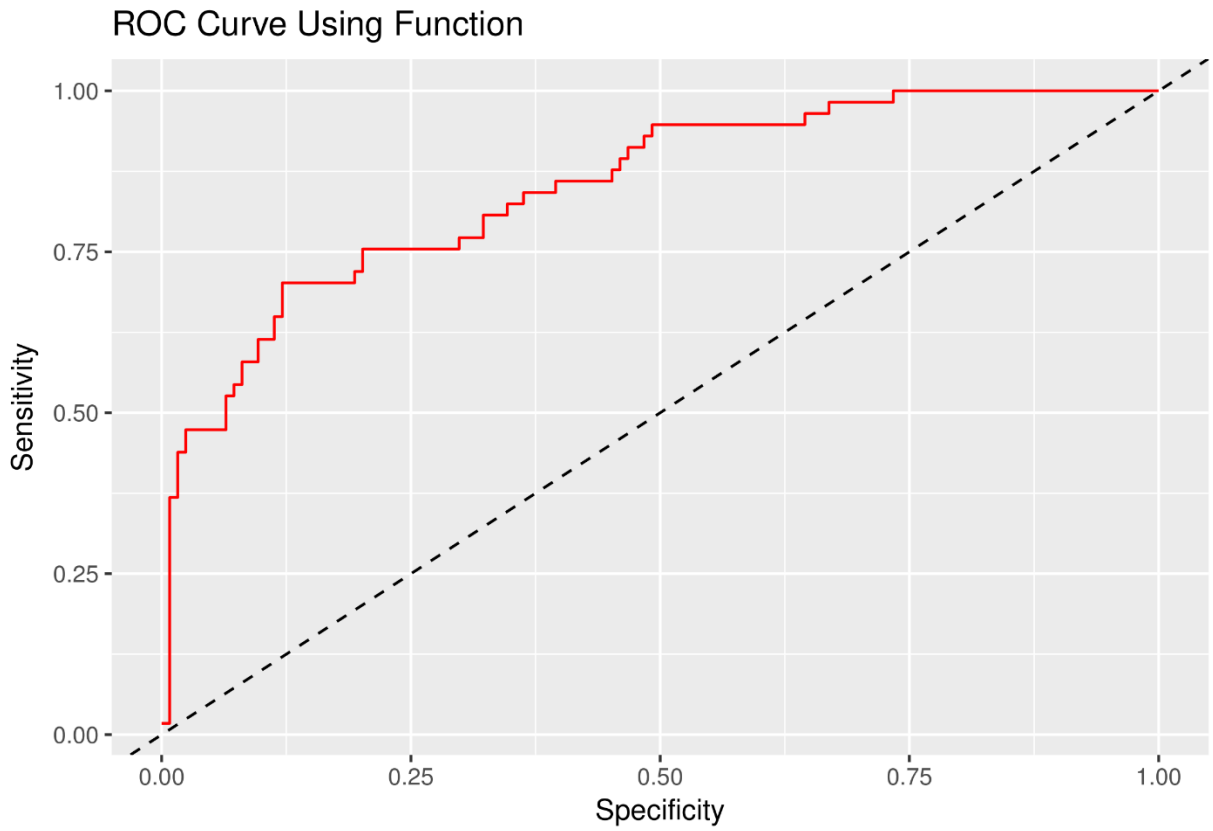
Problem 10

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.

R code:

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

  dFPR <- c(diff(df$FPR), 0)
  dTPR <- c(diff(df$TPR), 0)
  AUC <- sum(df$TPR * dFPR) + sum(dTPR * dFPR) / 2

  results <- list(df, AUC)
  return(results)
}

my_roc_auc <- my_roc(my_data$class, my_data$scored.probability)
my_roc_results <- my_roc_auc[[1]]
auc <- my_roc_auc[[2]]

ggplot(my_roc_results, aes(FPR, TPR)) +
  geom_line(color='red') +
  labs(title = "ROC Curve Using Function" , x = "Specificity", y = "Sensitivity") +
  geom_abline(linetype=2)
```

Plot:



AUC = 0.8503113

Problem 11

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

R code:

```
Classification <- c('Accuracy','Classification Error Rate', 'Precision',
                    'Sensitivity','Specificity', 'F1 Score')
Results <- round(c(a_fun(df), err_fun(data), prec_fun(df), sen_fun(df),
                   spec_fun(df), f1_fun(df)),4)
df1 <- as.data.frame(cbind(Classification, Results))
knitr::kable(df1)
```

Classification metrics:

Classification	Results
Accuracy	0.8066
Classification Error Rate	0.1934
Precision	0.8438
Sensitivity	0.4737
Specificity	0.9597
F1 Score	0.6067

Problem 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?

R code:

```
confusionMatrix(my_data$scored.class, my_data$class, positive="1")
```

Results:

```
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 : 0.4737
            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
```

The caret package result and my results are the same.

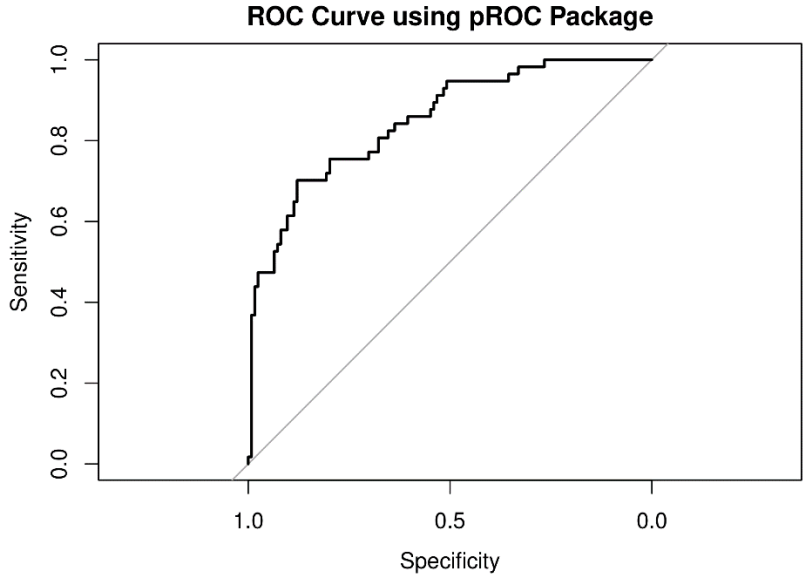
Problem 13

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

R code:

```
r_curve <- roc(my_data$class~my_data$scored.probability)
plot(r_curve, main= "ROC Curve using pROC Package")
```

Plot:




```
auc(roc(my_data$class, my_data$scored.probability))
```

```
## Area under the curve: 0.8503
```

The results are exact match with my functions.

Appendix:

R code

Load libraries

```
if (!require('ggplot2')) (install.packages('ggplot2'))

if (!require('caret')) (install.packages('caret'))

if (!require('pROC')) (install.packages('pROC'))
```

Problem 1

```
my_data <- read.csv('classification-output-data.csv', header=T)

head(my_data)
```

Problem 2

```
t <- table(my_data$scored.class, my_data$class)

knitr::kable(t)
```

Problem 3

```
a_fun <- function(df){

  df <- as.data.frame(table(my_data$scored.class, my_data$class))

  tp <- df$Freq[4]; fp <- df$Freq[2]; fn <- df$Freq[3]; tn <- df$Freq[1]

  accuracy <- ((tp + tn)/(tp + fp + tn + fn))

  return (accuracy)

}
```

```
a_fun(df)
```

Problem 4

```
err_fun <- function(df){
  df <- as.data.frame(table(my_data$scored.class, my_data$class))
  tp <- df$Freq[4]; fp <- df$Freq[2]; fn <- df$Freq[3]; tn <- df$Freq[1]
  error <- ((fp + fn)/(tp + fp + tn + fn))
  return (error)
}
err_fun(df)
```

Problem 5

```
prec_fun <- function(df){
  df <- as.data.frame(table(my_data$scored.class, my_data$class))
  tp <- df$Freq[4]; fp <- df$Freq[2]
  prec <- tp/(tp + fp)
  return (prec)
}
prec_fun(df)
```

Problem 6

```
sen_fun <- function(df){
```

```
  df <- as.data.frame(table(my_data$scored.class, my_data$class))

  tp <- df$Freq[4]; fn <- df$Freq[3]

  sen <- tp/(tp + fn)

  return (sen)

}

sen_fun(df)
```

Problem 7

```
spec_fun <- function(df){

  df <- as.data.frame(table(my_data$scored.class, my_data$class))

  fp <- df$Freq[2]; tn <- df$Freq[1]

  spec <- tn/(tn + fp)

  return (spec)

}

spec_fun(df)
```

Problem 8

```
f1_fun <- function(df){

  df <- as.data.frame(table(my_data$scored.class, my_data$class))

  tp <- df$Freq[4]; fp <- df$Freq[2]; fn <- df$Freq[3]

  prec <- tp/(tp + fp); sen <- tp/(tp + fn)
```

```
 f1 <- (2 * prec * sen) / (prec + sen)
 return (f1)
}
f1_fun(df)
```

Problem 9

```
set.seed(1023)
prec <- runif(50, 0, 1)
sen <- runif(50, 0, 1)
f1 <- (2 * prec * sen) / (prec + sen)
result <- summary(f1)
c(result[1], result[6])
```

Problem 10

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

 dFPR <- c(diff(df$FPR), 0)
 dTPR <- c(diff(df$TPR), 0)
 AUC <- sum(df$TPR * dFPR) + sum(dTPR * dFPR) / 2
```

```
  results <- list(df, AUC)
  return(results)
}

my_roc_auc <- my_roc(my_data$class, my_data$scored.probability)
my_roc_results <- my_roc_auc[[1]]
auc <- my_roc_auc[[2]]

ggplot(my_roc_results, aes(FPR, TPR)) +
  geom_line(color='red') +
  labs(title = "ROC Curve Using Function" , x = "Specificity", y = "Sensitivity") +
  geom_abline(linetype=2)

auc
```

Problem 11

```
Classification <- c('Accuracy','Classification Error Rate', 'Precision',
                    'Sensitivity','Specificity', 'F1 Score')
Results <- round(c(a_fun(df), err_fun(data), prec_fun(df), sen_fun(df),
```

```
            spec_fun(df), f1_fun(df)),4)

df1 <- as.data.frame(cbind(Classification, Results))

knitr::kable(df1)
```

Problem 12

```
confusionMatrix(my_data$scored.class, my_data$class, positive="1")
```

Problem 13

```
r_curve <- roc(my_data$class~my_data$scored.probability)

plot(r_curve, main= "ROC Curve using pROC Package")



auc(roc(my_data$class, my_data$scored.probability))
```

Reference:

http://blog.revolutionanalytics.com/2016/08/roc-curves-in-two-lines-of-code.html

http://blog.revolutionanalytics.com/2016/11/calculating-auc.html

GitHub URL: https://github.com/choudhury1023/DATA-621/tree/master/HW2