Leonid Rempel
ADEC7430.02 Big Data Econometrics
Individual Project 2
October 9, 2020

The base table function for creating a confusion matrix:

```
> table(data$scored.class, data$class, dnn = c('ScoredClass', 'ActualClass'))[2:1, 2:1]
           ActualClass
ScoredClass   1   0
          1  27   5
          0  30 119
```

The rows represent the class predicted by the model in question while the columns represent the actual class of those observations. I used the dnn option to label accordingly.

Next is a proof for number 9 which asks:

9. Let's consider the following question: 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 $ab < a$.)

The F1 score is made up from the measure of Precision - which I will call variable a - and from the measure of Specificity – which I will call variable b. Thus, F1 is rewritten as:

$$F1\ Score = \frac{2 * a * b}{a + b}$$

We know that both Specificity and Sensitivity have an upper bound at 1. This is possible because the model could, by chance or through its design, be perfect eliminating False Negatives and False Positives, thus we know that the F1 score is bounded at 1 because in this scenario, the F1 score simplifies to:

$$F1\ Score = \frac{2 * a * b}{a + b} = \frac{2 * 1 * 1}{1 + 1} = \frac{2}{2} = 1$$

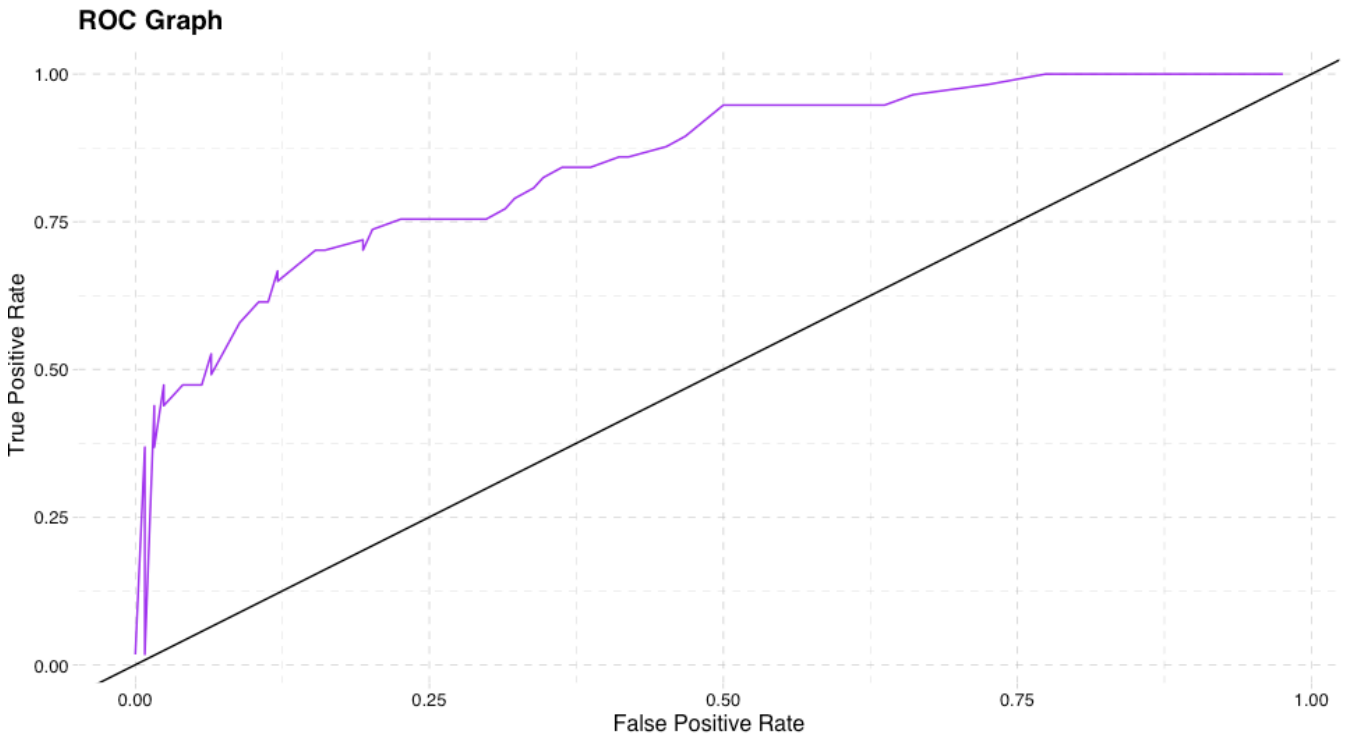
Alternatively, both Specificity and Sensitivity have a lower bound at 0. This may be the result of the model being so bad or unlucky that it either does not produce any True Positives or True Negatives, thus we know that the F1 score is also bounded at 0. This is because, in this scenario, as a and b approach 0, a*b converges to 0 faster than a+b giving us our lower bound of 0. If either one approaches 0 on its own, the F1 score simplifies to 0.

Thus the F1 score is bounded between 0 and 1 due to the underlying measures of Specificity and Sensitivity being bounded between 0 and 1.

The classification metrics as generated by my own functions are presented below:

```
"Area Under the Curve: 0.829937747594793"
"Accuracy: 0.806629834254144"
"Classification Error Rate: 0.193370165745856"
"Precision: 0.84375"
"Sensitivity: 0.473684210526316"
"Specificity: 0.959677419354839"
"F1 Score: 0.606741573033708"
```

The ROC Curve as generated by me through ggplot2 is presented below:



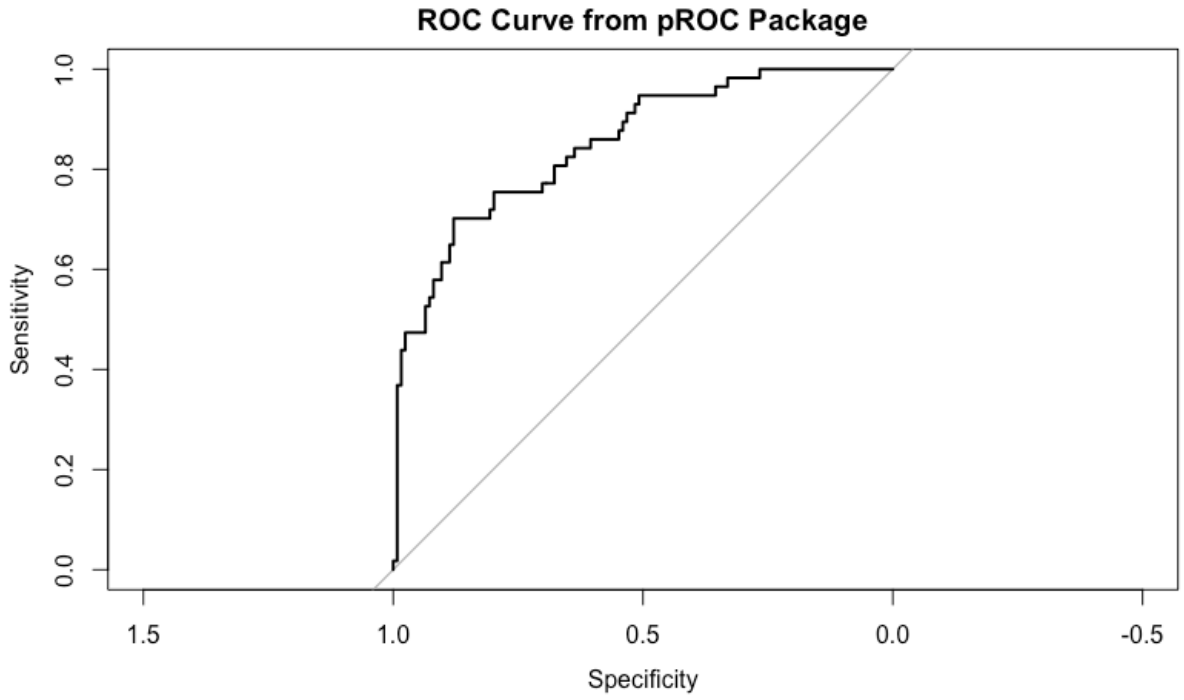
Here is the confusion matrix output as generated through the caret package:

```
Confusion Matrix and Statistics

          class
scored.class   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
```

Below is the additional Caret output

```
> sensitivity(df_conf_table)
[1] 0.4736842
> specificity(df_conf_table)
[1] 0.9596774
```

Comparing the two methods, I can see that my functions produced similar results to the Caret package. Below is the ROC produced by Caret and the calculated AUC:




```
> auc(roc(data$class, data$scored.probability))
Setting levels: control = 0, case = 1
Setting direction: controls < cases
Area under the curve: 0.8503
```

The AUC I produced is not too far off from the Caret AUC and my graph is similar to Caret's – the only difference is in the steps used (I don't think Caret used 0.01 steps and its graph is a bit more zoomed out than mine).

R Code used:

```
data <- read.csv("classification-output-data.csv")
RNGversion('3.5.3')

library(tidyverse)

# Creating the Confusion Matrix
table(data$scored.class, data$class, dnn = c('ScoredClass', 'ActualClass'))[2:1, 2:1]

accuracy_score <- function(df, class, scored) {
  conf <- with(df, table(scored, class))
  return((conf[1] + conf[4])/(conf[1] + conf[3] + conf[4] + conf[2]))
}
data_acc <- accuracy_score(data, class, scored.class)

classification_error_score <- function(df, class, scored) {
  conf <- with(df, table(scored, class))
  return((conf[3] + conf[2])/(conf[4] + conf[3] + conf[1] + conf[2]))
}
data_classif <- classification_error_score(data, class, scored.class)

precision_score <- function(df, class, scored) {
  conf <- with(df, table(scored, class))
  return((conf[4])/(conf[4] + conf[2]))
}
data_prec <- precision_score(data, class, scored.class)

sensitivity_score <- function(df, class, scored) {
  conf <- with(df, table(scored, class))
  return((conf[4])/(conf[4] + conf[3]))
}
data_sens <- sensitivity_score(data, class, scored.class)

specificity_score <- function(df, class, scored) {
  conf <- with(df, table(scored, class))
  return((conf[1])/(conf[1] + conf[2]))
}
data_spec <- specificity_score(data, class, scored.class)

f1_score <- function(df, class, scored) {
  prec.score <- precision_score(df, class, scored)
  sens.score <- sensitivity_score(df, class, scored)
  return((2 * prec.score * sens.score)/(prec.score + sens.score))
}
data_f1 <- f1_score(data, class, scored.class)
```

```
# Question 9 to be answered on pdf

library(ggplot2)
library(ggthemes)

roc_curve <- function(classification, prob) {

  random_guess_line <- seq(0, 1, by=0.01)
  TPRs <- c()
  FPRs <- c()

  for (i in 1:length(random_guess_line)){

    scores <- ifelse(prob >= random_guess_line[i], 1, 0)
    scored_df <- data.frame(scored.class = scores, class = classification)
    conf <- with(scored_df, table(scored.class, class))

    if (length(margin.table(conf,1)) == 1) {
       next
    }
      else {
      TPRs[i] <- (conf[4])/(conf[4] + conf[3])
      FPRs[i] <- (conf[2])/(conf[2] + conf[1])
    }

  }

  df_plot <- data.frame(Sens = TPRs, FalseP = FPRs)
  # Making sure we have complete cases
  df_plot <- df_plot[complete.cases(df_plot),]


  area_under_curve <- sum(abs(diff(df_plot$FalseP)) * df_plot$Sens)

  ROC_plot <- ggplot(df_plot, aes(x = FalseP, y = Sens)) +
    theme_pander() +
    geom_line(col = 'purple', alpha = 2) +
    geom_abline(intercept = 0, slope = 1) +
    labs(title="ROC Graph",
         x = "False Positive Rate",
         y = "True Positive Rate")

  # We have to return the plot as a list because its ggplot
  return(list(ROC_plot, area_under_curve))
}
```

```
ROC_list <- roc_curve(data$class, data$scored.probability)
ROC_plot <- ROC_list[[1]]

# Plotting the ROC Graph
ROC_plot

# AUF calculation
area_under_curve <- ROC_list[[2]]

metrics <- c(paste0("Area Under the Curve: ", area_under_curve),
             paste0("Accuracy: ", data_acc),
             paste0('Classification Error Rate: ', data_classif),
             paste0('Precision: ', data_prec),
             paste0('Sensitivity: ', data_sens),
             paste0('Specificity: ', data_spec),
             paste0('F1 Score: ', data_f1))

for (i in metrics) {
  print(i)
}

#############################################################################
# Investigating Caret
library(caret)

df_conf_table <- with(data, table(scored.class, class)[2:1,2:1])
confusionMatrix(df_conf_table)
sensitivity(df_conf_table)
specificity(df_conf_table)

library(pROC)
plot(roc(data$class, data$scored.probability), main="ROC Curve from pROC Package")
auc(roc(data$class, data$scored.probability))
```