DATA 621 Homework 2

Critical Thinking Group 1

October 09, 2021

Contents

Prepared for:

Prof. Dr. Nasrin Khansari
City University of New York, School of Professional Studies - Data 621

DATA 621 – Business Analytics and Data Mining

Home Work 2

Prepared by:

Critical Thinking Group 1

Vic Chan
Gehad Gad
Evan McLaughlin
Bruno de Melo
Anjal Hussan
Zhouxin Shi
Sie Siong Wong

1. Data Source

```r
cm <- read.csv("https://raw.githubusercontent.com/ahussan/DATA_621_Group1/main/HW2/classification-outpu
```

2. Data Set and Confusion Matrix

From the confusion matrix table below, we have actual class or reference value versus predicted class value, and we assume *0* to be the non-event class and *1* to be positive class. Class values are display in columns, while predicted values are displayed in the rows. So there are 119 true negatives (TN), 27 true positives (TP), 30 false negatives (FN), and 5 false positives (FP).

```r
conf_mtx <- cm %>% dplyr::select(scored.class, class) %>% table()
conf_mtx
```

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

3. Function for Accuracy of Predictions

```r
accuracy <- function(x){

  numerator <- x[2,2] + x[1,1]
  denominator <- sum(x)
  return(numerator/denominator)
}
```

4. Function for Classification Error Rate of Predictions

```r
error_rate <- function(x){

  numerator <- x[1,2] + x[2,1]
  denominator <- sum(x)
  return(numerator/denominator)
}
```

We verify below that `accuracy` and an `error rate` sums to one.

```r
accuracy(conf_mtx) + error_rate(conf_mtx)
```

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

5. Function for Precisions of Predictions

```
precision <- function(x){

  numerator <- x[2,2]
  denominator <- x[2,2] + x[2,1]
  return(numerator/denominator)
}
```

6. Function for Sensitivity of Predictions

```
sensitivity <- function(x){

  numerator <- x[2,2]
  denominator <- x[2,2] + x[1,2]
  return(numerator/denominator)
}
```

7. Function for Specificity of Predictions

```
specificity <- function(x){

  numerator <- x[1,1]
  denominator <- x[1,1] + x[2,1]
  return(numerator/denominator)
}
```

8. F1 Score of Predictions

```
f1_score <- function(x){

  numerator <- 2 * precision(x) * sensitivity(x)
  denominator <- precision(x) + sensitivity(x)
  return(numerator/denominator)
}
```

9. Bound on the F1 score

We will assume that $a = Precision$ and $b = Sensitivity$. So, we re-rewrite:

$$F1Score = \frac{2 * a * b}{a + b}$$

To see the bounds of the f1 score we will be setting both a and b to their maximum and minimum.

Assuming that $a = 1$ and $b = 1$ we can see that the F1 score is equal to 1.

$$F1Score = \frac{2 * 1 * 1}{1 + 1} = 1$$

If we assume that a and b are approaching zero we can see that the f1 value will be positive non zero number. Since a and b are bounded between 0 and 1 this means that the F1 function will always be between 0 and 1 which means that $ab < a$ and $ab < b$.

10. Write a function to write a ROC curve

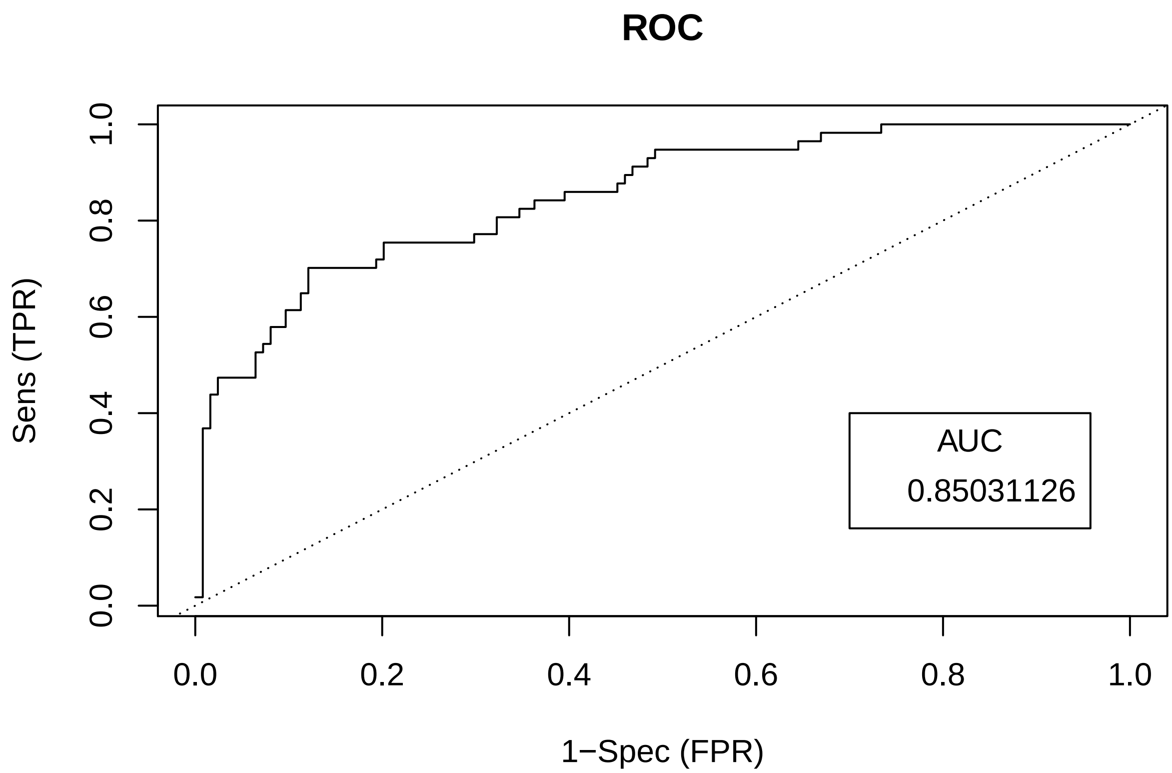
```
fetch_accuracy <- function(df){
  dt <- df %>% dplyr::select( scored.class, class ) %>% table()
  #positive
  TP <- dt[2,2]
  #negative
  TN <- dt[1,1]
  #false positive
  FP <- dt[2,1]
  #false negative
  FN <- dt[1,2]

  return(round((TP + TN)/(TP + FP + TN + FN),4))
}
#write function
fetch_roc_curve <- function(x,p){
    x <- x[order(p, decreasing=TRUE)]

    TP = cumsum(x)/sum(x)
    FP = cumsum(!x)/sum(!x)

    roc_df <- data.frame(TP, FP)
    auc <- sum(TP * c(diff(FP), 0)) + sum(c(diff(TP), 0) * c(diff(FP), 0))/2

    return(c(df=roc_df, auc = auc))
}
#apply function to dataset
roc_data <- fetch_roc_curve(cm$class, cm$scored.probability)
plot(roc_data[[2]],roc_data[[1]], type = 'l', main = "ROC",xlab="1-Spec (FPR)", ylab = "Sens (TPR)")
abline(0,1, lty=3)
legend(0.7,0.4, round(roc_data$auc,8), title = 'AUC')
```



This curve illustrates the true positive rate versus the false positive rate and enables us to assess the accuracy. We are thus able to classify our observations through the establishment of probability thresholds. AUC is a measurement of our model's suitability for determining positive and negative outcomes. Our relatively high AUC is a good "grade" for our desire to correctly guess outcomes.

11. Classification metrics output

Table 1: Summary of Classification Metrics

	Score
Accuracy	0.8066
Error Rate	0.1934
F1_score	0.6067
Precision	0.8438
Sensitivity	0.4737
Specificity	0.9597

12. Investigation of the `caret` R package.

We used the `caret` R package to calculate a confusion Matrix, sensitivity, and specificity for the data set. See below for the confusion matrix:

```
conf_mtx_caret <- confusionMatrix(factor(cm$scored.class), factor(cm$class),  positive = '1')

conf_mtx_caret$table
```

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

Result from the `caret` package is similar to our calculated matrix. For two classes, the `caret` function assumes as default that the class corresponding to an event is the first class level, in our case *0*. We changed the positive argument in the function, and assume the positive class to be *1*.

```
conf_mtx == conf_mtx_caret$table
```

```
##             class
## scored.class    0    1
##            0 TRUE TRUE
##            1 TRUE TRUE
```

Calling `confusionMatrix` function can be used to generate statistics.

```
conf_mtx_caret
```

```
## 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
##
```

Using the function `byclass` argument, we can extract *sensitivity* and *specificity* measures and thus compare with our own functions.

```r
sens<-conf_mtx_caret$byClass[1]
spec<-conf_mtx_caret$byClass[2]

sensitivity(conf_mtx) == sens
```

```
## Sensitivity
##        TRUE
```

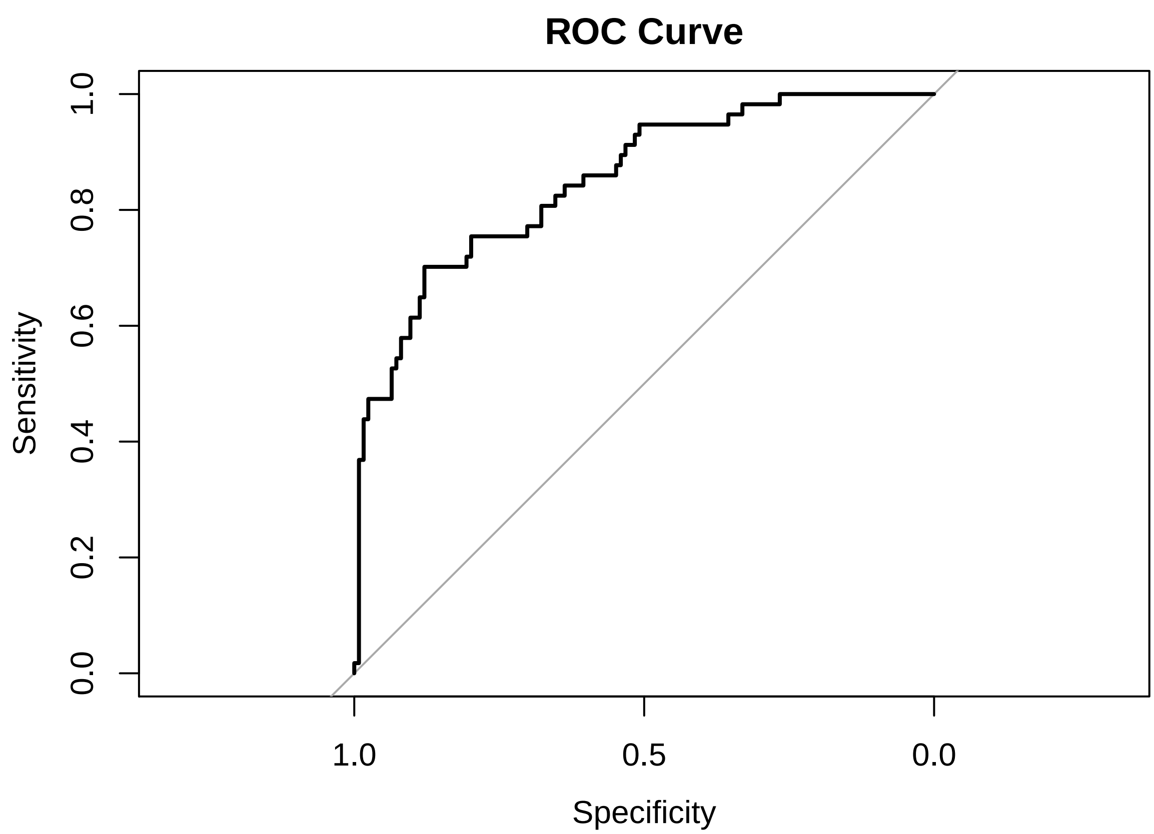
```r
specificity(conf_mtx) == spec
```

```
## Specificity
##        TRUE
```

13. Investigation of the `pROC` R package.

We used the `pROC` R package to generate an ROC curve for the data set.

```
rcurve <- roc(cm$class~cm$scored.probability)
plot(rcurve, main="ROC Curve")
```



Best Threshold value using pROC package is {Threshold = 0.375117,fpr = 0.120968,tpr = 0.701754}

Note: The second method (using auc) predicts better than first method (using distance from (0,1))