DATA 621 Business Analytics and Data Mining

Group 2 - Gabriel Campos, Melissa Bowman, Alexander Khaykin, & Jennifer Abinette

Last edited October 14, 2023

Homework #2 Assignment Requirements

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.

Supplemental Material

- Applied Predictive Modeling, Ch. 11 (provided as a PDF file).
- Web tutorials: http://www.saedsayad.com/model_evaluation_c.htm

Deliverables (100 Points)

- Upon following the instructions below, use your created R functions and the other packages to generate the classification metrics for the provided data set. A write-up of your solutions submitted in PDF format

Instructions

Complete each of the following steps as instructed:

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

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.

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

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.

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

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

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.

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

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.

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

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.

$$Specifity=\frac{TN}{TN+FN}$$

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.

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

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 $0 <$ < 1 and $0 <$ < 1 then $<$.)

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.

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

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?

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?

Data Exploration

Load the data

```
git_url<-
  "https://raw.githubusercontent.com/GitableGabe/Data621_Data/main/"

df_classif <-
  read.csv(paste0(git_url,"classification-output-data.csv"))
head(df_classif,n=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
```

Confusion Matrix

```
ls_class<-factor(df_classif$class)
ls_scr_class<-factor(df_classif$scored.class)
ls_sr_prb<-df_classif$scored.probability
```

```
(example<-confusionMatrix(data=ls_scr_class, reference = ls_class))
```

```
## 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
##              Prevalence : 0.6851
##          Detection Rate : 0.6575
##    Detection Prevalence : 0.8232
##       Balanced Accuracy : 0.7167
##
##        'Positive' Class : 0
##
```

```
(confusion_matrix <- table(ls_class,ls_scr_class))
```

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

True Positive

```
(tp <- confusion_matrix[1, 1])
```

```
## [1] 119
```

True Negative

```
(tn <- confusion_matrix[2, 2])
```

```
## [1] 27
```

False Positive

```
(fp <- confusion_matrix[1, 2])
```

```
## [1] 5
```

False Negative

```
(fn <- confusion_matrix[2, 1])
```

```
## [1] 30
```

Accuracy Function

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

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

Classification Error Rate Function

```
(classification_error_rate <- (fp + fn)/(tp + fp + tn + fn))
```

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

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

```
(accuracy + classification_error_rate)
```

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

Precision Function

```
(precision <- tp/(tp + fp))
```

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

Sensitivity Function

```
(sensitivity <- tp/(tp + fn))
```

```
## [1] 0.7986577
```

Specificity Function

```
(specificity <- tn/(tn + fp))
```

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

F1 score Function

```
(f1_score <- (2 * precision * sensitivity)/(precision + sensitivity))
```

```
## [1] 0.8717949
```