Homework 2

Group 1

Contents

Prepared for:

Dr. Nathan Bastian
City University of New York, School of Professional Studies - Data 621

Prepared by:

Group 1

Senthil Dhanapal
Yadu Chittampalli
Christophe Hunt

1 Data Source

The data is a set of actual classes and predicted classes as provided by Dr. Nathan Bastian for this exercise. We uploaded the data to our public GitHub repository for ease of access.

2 Data Explained and Confusion Matrix

We will be using the following columns from the data source:

- 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

Below is the raw confusion matrix for our scored data set

	Predicted Failure	Predicted Success
Actual Failure	119	5
Actual Success	30	27

Here the rows represent the actual classes and the columns represent the predicted classes.

3 Function for Accuracy of Predictions

We developed a function that takes the data set as a dataframe, with actual and predicted classifications identified, and returns the accuracy of the predictions.

Accuracy is determined by the below equation:

$$Accuracy\ =\ \frac{True\ Positives\ +\ True\ Negatives}{True\ Positives\ +\ False\ Positives\ +\ True\ Negatives\ +\ False\ Negatives}$$

4 Function for Classification Error Rate of Predictions

We developed 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. It also verifies that the accuracy and an error rate sums to one.

Classification of Error Rate is determined by the below formula:

$$Classification\ Error\ Rate\ =\ \frac{False\ Positives\ +\ False\ Negatives}{True\ Positives\ +\ False\ Positives\ +\ True\ Negatives\ +\ False\ Negatives}$$

5 Function for Precisions of Predictions

Write a function that takes the data set as a dataframe, with actual and predicted classifications identified, and returns the precision of the predictions. - Senthil

Precision is determined by the below formula:

$$Precision = \frac{True\ Positives}{True\ Positives\ +\ False\ Positives}$$

6 Function for Sensitivity of Predictions

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. - Senthil

Sensitivity is determined by the below formula:

$$Sensitivity = \frac{True\ Positives}{True\ Positives\ +\ False\ Negatives}$$

7 Function for Specificity of Predictions

We developed a function that takes the data set as a dataframe, with actual and predicted classifications identified, and returns the specificity of the predictions.

Specificity is determined by the below formula:

$$Specificity = \frac{True\ Negative}{True\ Negatives\ +\ False\ Positives}$$

8 F1 Score of Predictions

We developed a function that takes the data set as a dataframe, with actual and predicted classifications identified, and returns the F1 score of the predictions.

```r
f1_score <- function(df, actual, prediction){
  require(scales)
  if (sum(colnames(df) %in% c(actual, prediction)) != 2) {
    return("One or more columns were not found, please verify selections")
  }

  m <- as.data.frame(table(df[[actual]], df[[prediction]]))

  true_positive  <- as.numeric(m$Freq[m$Var1 == 1 & m$Var2 == 1])
  false_positive <- as.numeric(m$Freq[m$Var1 == 0 & m$Var2 == 1])
  false_negative <- as.numeric(m$Freq[m$Var1 == 1 & m$Var2 == 0])

  precision <- true_positive/(true_positive + false_positive)
  sensitivity <- true_positive/(true_positive + false_negative)
  f1_score <- ((2 * precision * sensitivity) / (precision + sensitivity))
  return(f1_score)
  }
```

9 Bounds of F1 Score of Predictions

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 ????????- Christophe

10 Function for ROC curve

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. - Senthil

11 R Functions created and classification output

11.1 Accuracy of Predictions

```r
paste0("Accuracy of Predictions = ", percent(accuracy(df = scores, actual = "class",
    prediction = "scored.class")))
```

[1] "Accuracy of Predictions = 80.7%"

11.2 Classification Error Rate of Predictions

```r
paste0("Error Rate of Predictions = ", percent(errorrate(scores, "class", "scored.class")))
```

[1] "Error Rate of Predictions = 19.3%"

11.3 Precisions of Predictions

11.4 Sensitivity of Predictions

11.5 Specificity of Predictions

```r
paste0("Specificity of Predictions = ", percent(Specificity(df = scores, actual = "class",
    prediction = "scored.class")))
```

[1] "Specificity of Predictions = 96%"

11.6 F1 Score of Predictions

```
paste0("The F1 Score = ", f1_score(df = scores, actual = "class", prediction = "scored.class"))
```

[1] “The F1 Score = 0.606741573033708”

12 Investigation of `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?

```
0   1
```

0 119 5 1 30 27

Here, the accuracy is rounded up to 4 decimal places. The confusion matrix has rows that represent the predicted classes and columns that represent the actual classes.

[1] 0.7986577

[1] 0.84375

13 Investigation of the `pROC` package.

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

14 Appendix A

14.1 Session Info

- R version 3.3.1 (2016-06-21), `x86_64-w64-mingw32`
- Locale: `LC_COLLATE=English_United States.1252`, `LC_CTYPE=English_United States.1252`, `LC_MONETARY=English_United States.1252`, `LC_NUMERIC=C`, `LC_TIME=English_United States.1252`
- Base packages: base, datasets, graphics, grDevices, methods, stats, utils
- Other packages: caret 6.0-71, dplyr 0.5.0, formatR 1.4, ggplot2 2.1.0, knitr 1.14, lattice 0.20-34, pacman 0.4.1, pander 0.6.0, purrr 0.2.2, readr 1.0.0, scales 0.4.0, tibble 1.2, tidyr 0.6.0, tidyverse 1.0.0, xtable 1.8-2
- Loaded via a namespace (and not attached): assertthat 0.1, car 2.1-3, class 7.3-14, codetools 0.2-15, colorspace 1.2-6, DBI 0.5-1, digest 0.6.10, e1071 1.6-7, evaluate 0.9, foreach 1.4.3, grid 3.3.1, gtable 0.2.0, htmltools 0.3.5, iterators 1.0.8, lme4 1.1-12, magrittr 1.5, MASS 7.3-45, Matrix 1.2-7.1, MatrixModels 0.4-1, mgcv 1.8-15, minqa 1.2.4, munsell 0.4.3, nlme 3.1-128, nloptr 1.0.4, nnet 7.3-12, parallel 3.3.1, pbkrtest 0.4-6, plyr 1.8.4, quantreg 5.29, R6 2.2.0, Rcpp 0.12.7, reshape2 1.4.1, rmarkdown 1.0, SparseM 1.72, splines 3.3.1, stats4 3.3.1, stringi 1.1.2, stringr 1.1.0, tools 3.3.1, yaml 2.1.13

14.2 Data Table

class	scored.class	scored.probability
0	0	0.3284523
0	0	0.2731904
1	0	0.1096604
0	0	0.0559984
0	0	0.1004907
0	0	0.0551546
0	0	0.1071154
0	0	0.4599474
0	0	0.1170237
0	0	0.3153632
0	0	0.1251892
0	0	0.2706248
0	0	0.2098096
0	0	0.0935859
1	1	0.8848457
1	0	0.3966522
0	1	0.8913949
1	1	0.5345490
1	1	0.9463342
0	0	0.1449162
0	0	0.2176380
0	0	0.0752136
0	0	0.0884325
0	0	0.3034682
1	1	0.7244800
1	0	0.2749737

class	scored.class	scored.probability
1	0	0.4248648
1	0	0.4309255
0	0	0.0232280
0	0	0.0459608
0	0	0.1279853
0	0	0.2993371
1	0	0.4590950
0	0	0.1047958
1	1	0.8630918
1	1	0.6399750
0	0	0.3581843
0	0	0.3721647
1	1	0.8111032
0	0	0.1681274
0	0	0.1512780
0	0	0.1070070
0	0	0.1879614
0	0	0.1371971
0	0	0.3004749
0	0	0.1368871
0	0	0.0978691
0	0	0.0629070
1	0	0.2694193
0	0	0.4885428
0	0	0.3717580
1	0	0.0994799
0	0	0.0865623
0	0	0.1752894
1	0	0.4693790
1	1	0.6165544
0	0	0.0998279
1	1	0.6891771
0	0	0.2552862
1	1	0.8505433
1	0	0.1688625
0	0	0.0972415
0	0	0.2483691
0	0	0.1815610
0	0	0.2399936
0	0	0.4045589
0	0	0.3583026
0	0	0.1506308
1	0	0.4865383
1	1	0.6150348
0	0	0.3544895
1	0	0.1731396
1	0	0.2596111
1	1	0.6989399
0	0	0.2986016
0	0	0.1042313
0	0	0.2110528
0	0	0.0654278

class	scored.class	scored.probability
0	0	0.0505143
0	0	0.1086797
0	0	0.0786895
1	1	0.6812876
1	0	0.3771203
0	0	0.1627077
0	0	0.3521508
1	0	0.4754964
0	0	0.1356581
0	0	0.1339198
0	1	0.5224711
0	0	0.2593819
0	0	0.0994674
0	0	0.1271923
1	0	0.3764462
0	1	0.5208823
1	1	0.7605921
1	0	0.2089265
1	0	0.2333521
0	0	0.2059407
0	0	0.1156596
0	0	0.0839931
1	0	0.1177312
1	1	0.7170364
0	0	0.1292900
0	0	0.4368037
0	0	0.2815537
1	1	0.5919838
1	1	0.8472932
0	0	0.3151556
0	0	0.1373101
0	0	0.1680963
0	0	0.0506711
0	0	0.4983591
1	0	0.4548143
0	0	0.4504421
1	0	0.1814974
0	0	0.2942037
1	0	0.4094483
1	0	0.3167683
0	0	0.1969549
0	0	0.0627996
1	1	0.8833591
0	0	0.0993645
1	0	0.4088370
0	0	0.3624922
0	0	0.0799184
1	1	0.6172762
0	0	0.2235817
0	0	0.3013863
0	0	0.0661091
1	0	0.1670290

class	scored.class	scored.probability
0	0	0.2970645
0	1	0.6276502
0	0	0.2036287
0	0	0.4574735
0	0	0.3722721
1	1	0.6357807
0	0	0.0837734
0	0	0.1519378
0	0	0.0532099
1	1	0.5486644
0	0	0.4946261
0	0	0.2353255
0	0	0.1831519
0	0	0.0641505
0	0	0.0859556
0	0	0.3737878
0	0	0.4128094
1	1	0.8304976
0	0	0.1314538
0	0	0.0661406
0	0	0.1009617
0	0	0.0286396
0	0	0.2696404
1	0	0.3281420
0	0	0.1493509
1	0	0.4555714
0	0	0.0809498
0	0	0.0347143
1	1	0.6614750
0	0	0.0659893
0	0	0.1397903
0	0	0.0474267
0	0	0.0266070
1	1	0.7825946
0	0	0.1418285
0	0	0.2850303
1	0	0.3388554
1	0	0.1626446
0	1	0.5649062
0	0	0.0562242
0	0	0.1891168
0	0	0.1707249
0	0	0.1608049
1	0	0.2457727
0	0	0.1099905
1	1	0.6764516
0	0	0.3114196
1	1	0.7072096
1	1	0.8882766
0	0	0.4224679
0	0	0.1199810

14.3 R source code

Please see Homework 2.rmd on GitHub for source code.

https://github.com/ChristopheHunt/DATA-621-Group-1/blob/master/Homework%202/Homework%202.
Rmd.