STAT 745: Homework 2

Ian Effendi (iae2784@rit.edu)

9/16/2021

Contents

```r
# Required packages
require(knitr)
require(rmarkdown)
require(tidyverse) # readr, tibble, ggplot2
```

Overview

This section provides an overview of the document.

For this assignment, we will be exploring the `Livers.txt` data set, detailing various liver and blood conditions of 345 men, with the goal of predicting liver disorder severity.

Preprocess

In this section, I clean up the `Liver.txt` data source and output an intermediary data file to make it easier to perform classification with the data.

Contents of `Liver.txt`

1. Read in the data from the text file "`Liver.txt`".

The data is stored in a raw text file with comma-separated values. We can use `readr::read_csv()` on this file as-is to get our data into R. Additionally, the `read_csv()` accepts a `col_types` parameter that allows us to designate the types for each of the features.

```
# Read raw text file into a tibble.
Liver.txt <- as_tibble(read_csv(
  # File is located in the data/raw folder.
  file = "data/raw/Liver.txt",
  # First row is not a header row, so we should set `col_names` to FALSE.
  col_names = FALSE,
  # The column type choices are described below.
  col_types = cols(
    # 1 through 5 are quantitative.
    X1 = col_integer(),
    X2 = col_integer(),
    X3 = col_integer(),
    X4 = col_integer(),
    X5 = col_integer(),
    # No. of alcoholic beverages drunk.
    X6 = col_double(),
    # Severity group of the liver condition.
    X7 = col_factor(),
  )
))

# Preview the first 10 rows from the data set.
Liver.txt
#>  # A tibble: 345 x 7
#>        X1    X2    X3    X4    X5    X6 X7
#>     <int> <int> <int> <int> <int> <dbl> <fct>
#>   1    85    92    45    27    31    0   1
#>   2    85    64    59    32    23    0   2
#>   3    86    54    33    16    54    0   2
#>   4    91    78    34    24    36    0   2
#>   5    87    70    12    28    10    0   2
#>   6    98    55    13    17    17    0   2
#>   7    88    62    20    17     9    0.5 1
#>   8    88    67    21    11    11    0.5 1
#>   9    92    54    22    20     7    0.5 1
#>  10    90    60    25    19     5    0.5 1
#>  # ... with 335 more rows
```

We can see the tibble contains 345 rows and a total of 7 columns, which matches our expectations from the assignment details.

Clean `Liver.txt`

The assignment document provides some information about the features in the data set. We know that:

- Variables 1 through 5 are quantitative results of blood tests sensitive to liver disorders.
  - Inference: These are stored as whole integers in the original file, so a `<int>` type seems appropriate.
  - Concern: We don't really know what units these are stored with; if they are percentages, some level of scaling or data transformation may have been appropriate.
- Variable 6 is the number of alcoholic beverages drunk per day.
  - Inference: This is stored as a fractional numeric, so despite being described as something that I would assume is a discrete integer, we'll store this as a `<dbl>` as well.
- Variable 7 is the response, demarcating the liver malfunctioning group.
  - Inference: This is a factor. Given this is a classification problem, we should store this as a categorical `<fctr>` type.

```
# Normalize column names to lowercase.
Liver.1 <- rename_with(Liver.txt, tolower)

# Rename some column names to be more descriptive.
# Uses `dplyr::rename(.data, new.name = old.name)

# x1 through x5 will have similar naming convention
# as they are blood test results sensitive to liver conditions.
Liver.1 <- rename(Liver.1, blood.1 = x1)
Liver.1 <- rename(Liver.1, blood.2 = x2)
Liver.1 <- rename(Liver.1, blood.3 = x3)
Liver.1 <- rename(Liver.1, blood.4 = x4)
Liver.1 <- rename(Liver.1, blood.5 = x5)

# x6 -> drinks
Liver.1 <- rename(Liver.1, drinks = x6)

# x7 -> severity
Liver.1 <- rename(Liver.1, severity = x7)

# Preview changes to column names.
names(Liver.1)
#>  [1] "blood.1"  "blood.2"  "blood.3"  "blood.4"  "blood.5"
#>  [6] "drinks"   "severity"
```

Now that the fields are named appropriately, I want to finish cleaning the data up by encoding the `severity` factor as a binary field.

Table 1: Severity Encoding Map

Status	Value	Code
severe	1	1
not severe	2	0

```
# Encode severity == 1 as 1, All other values as 0.
Liver.2 <- Liver.1 %>%
          mutate(severity = as.integer(ifelse(severity == 1, 1, 0)))

# Preview the Liver tibble.
Liver.2
#>  # A tibble: 345 x 7
#>     blood.1 blood.2 blood.3 blood.4 blood.5 drinks severity
#>       <int>   <int>   <int>   <int>   <int>  <dbl>    <int>
#>   1      85      92      45      27      31     0          1
#>   2      85      64      59      32      23     0          0
#>   3      86      54      33      16      54     0          0
#>   4      91      78      34      24      36     0          0
#>   5      87      70      12      28      10     0          0
#>   6      98      55      13      17      17     0          0
#>   7      88      62      20      17       9     0.5        1
#>   8      88      67      21      11      11     0.5        1
#>   9      92      54      22      20       7     0.5        1
#>  10      90      60      25      19       5     0.5        1
#>  # ... with 335 more rows
```

```
# Write the data to the `data/processed/` directory as `Liver.Rds`
write_rds(Liver.2, "data/processed/Liver.Rds")
```

Instead of using the `save()` function (which would save our entire environment), we'll save the single modified `Liver.2` object with `readr::write_rds()` wrapper for `saveRDS()`.

This method will allow us to import our intermediate data object into a new variable rather than dumping loose data objects into the environment, when using the `readRDS()` wrapper, `readr::read_rds()`.

Classification

2. Predict Variable 7 using logistic regression (using $p = P(\text{Group 1})$) with the remaining variables as predictors.

```
# Read the processed intermediate file.
Liver <- read_rds("data/processed/Liver.Rds")
Liver[1,]
#>  # A tibble: 1 x 7
#>    blood.1 blood.2 blood.3 blood.4 blood.5 drinks severity
#>      <int>   <int>   <int>   <int>   <int>  <dbl>    <int>
#>  1      85      92      45      27      31      0        1
```

Logistic Regression

severity can only take one of two values: 0 or 1. We denote $p = P(severity = 1)$ and we will fit a logistic regression model:

$$ln[\frac{p}{(1-p)}] = \beta_0 + \beta_1(blood.1) + \cdots + \beta_5(blood.5) + \beta_6(drinks)$$

We will predict $severity = 1$ if $p > p_0$ and as 0 otherwise.

```
# Define and fit a logistic regression model.
fit.1 <- glm(severity ~ ., data=Liver, family="binomial")
```

Model Analysis

```
# Summary fit function from class.
Summary.fit <- function(fit.obj) {
  mat <- summary(fit.obj)$coef
  mat[,3] <- round(mat[,3], 2)
  mat[,4] <- round(mat[,4], 2)
  return(mat)
}

# Show results.
Summary.fit(fit.1)

fit.1.residuals <- as_tibble(residuals.glm(fit.1))
fit.1.residuals
#>                 Estimate  Std. Error z value Pr(>|z|)
#>  (Intercept) -5.99025807 2.685457986   -2.23     0.03
#>  blood.1      0.06398364 0.029644744    2.16     0.03
#>  blood.2      0.01952514 0.006759465    2.89     0.00
#>  blood.3      0.06410602 0.012299568    5.21     0.00
#>  blood.4     -0.12319809 0.024273023   -5.08     0.00
#>  blood.5     -0.01894716 0.005601748   -3.38     0.00
#>  drinks       0.06808016 0.040379564    1.69     0.09
#>  # A tibble: 345 x 1
#>      value
#>      <dbl>
#>   1  1.09
#>   2 -1.22
#>   3 -1.05
#>   4 -1.13
#>   5 -0.521
#>   6 -1.08
#>   7  1.22
#>   8  0.874
#>   9  1.26
#>  10  1.12
#>  # ... with 335 more rows
```

Misclassification Error Analysis

3. Minimize the total error of misclassification by changing the threshold for p. Show the results graphically, as we did in class, and also show the optimal values, including the misclassification table for that optimal case.

```
# Make the prediction needed with threshold set to p = 0.5.
p.threshold <- 0.5
fit.1.prediction <- (fit.1$fitted > p.threshold)

# Get the misclassification table.
(Classif.table <- table(fit.1.prediction, Liver$severity))

# Calculate the error rate.
Error.rate.fit <- function(mat) {
  # Assume a 2 by 2 table.
  (mat[1,2] + mat[2,1]) / sum(mat)
}

# Misclassification error rate:
Error.rate.fit(Classif.table)

# Normal 'at random' error rate:
sum(Liver$severity==0)/length(Liver$severity)
#>
#>  fit.1.prediction   0   1
#>             FALSE 165  67
#>             TRUE   35  78
#>  [1] 0.2956522
#>  [1] 0.5797101
```

Crossvalidation

$k = 10$-fold Crossvalidation

4. Repeat Step 3 with 10-fold CV. Use 100 rounds. Calculate means and their standard errors of the minimized total error of misclassification.

$k = 3$-fold Crossvalidation

5. Repeat Step 3 with 3-fold CV. Use 100 rounds. Calculate means and their standard errors of the minimized total error of misclassification.

Session Information

This report was generated by the `vignettes/hw2_notebook.Rmd` file. The setup chunk for this document sets the root directory to the project root directory using the `rprojroot` package; all file paths are relative to the project root.

```
#> R version 4.1.1 (2021-08-10)
#> Platform: x86_64-w64-mingw32/x64 (64-bit)
#> Running under: Windows 10 x64 (build 19043)
#>
#> Matrix products: default
#>
#> locale:
#> [1] LC_COLLATE=English_United States.1252
#> [2] LC_CTYPE=English_United States.1252
#> [3] LC_MONETARY=English_United States.1252
#> [4] LC_NUMERIC=C
#> [5] LC_TIME=English_United States.1252
#>
#> attached base packages:
#> [1] stats     graphics  grDevices utils     datasets
#> [6] methods   base
#>
#> other attached packages:
#>  [1] forcats_0.5.1   stringr_1.4.0   dplyr_1.0.7
#>  [4] purrr_0.3.4     readr_2.0.1     tidyr_1.1.3
#>  [7] tibble_3.1.4    ggplot2_3.3.5   tidyverse_1.3.1
#> [10] rmarkdown_2.10  rprojroot_2.0.2 knitr_1.34
#>
#> loaded via a namespace (and not attached):
#>  [1] tidyselect_1.1.1 xfun_0.25        haven_2.4.3
#>  [4] colorspace_2.0-2 vctrs_0.3.8      generics_0.1.0
#>  [7] htmltools_0.5.2  yaml_2.2.1       utf8_1.2.2
#> [10] rlang_0.4.11     pillar_1.6.2     glue_1.4.2
#> [13] withr_2.4.2      DBI_1.1.1        bit64_4.0.5
#> [16] dbplyr_2.1.1     modelr_0.1.8     readxl_1.3.1
#> [19] lifecycle_1.0.0  munsell_0.5.0    gtable_0.3.0
#> [22] cellranger_1.1.0 rvest_1.0.1      evaluate_0.14
#> [25] tzdb_0.1.2       fastmap_1.1.0    parallel_4.1.1
#> [28] fansi_0.5.0      highr_0.9        broom_0.7.9
#> [31] Rcpp_1.0.7       scales_1.1.1     backports_1.2.1
#> [34] vroom_1.5.4      jsonlite_1.7.2   bit_4.0.4
#> [37] fs_1.5.0         hms_1.1.0        digest_0.6.27
#> [40] stringi_1.7.4    grid_4.1.1       cli_3.0.1
#> [43] tools_4.1.1      magrittr_2.0.1   crayon_1.4.1
#> [46] pkgconfig_2.0.3  ellipsis_0.3.2   xml2_1.3.2
#> [49] reprex_2.0.1     lubridate_1.7.10 rstudioapi_0.13
#> [52] assertthat_0.2.1 httr_1.4.2       R6_2.5.1
#> [55] compiler_4.1.1
```