Homework 3 Report

Classification of liver malfunction severity (LDA)

Ian Effendi \ iae2784@rit.edu

October 12, 2021

Contents

Certification

I certify that I indeed finished reading Ch. 4 from *An Introduction to Statistical Learning*, by James Gareth, Daniela Witten, Trevor Hastie, Robert Tibshirani.

Overview

In this assignment we will:

- Perform exploratory data analysis (*EDA*) on the dataset.
- Fit and analyze a linear discriminant analysis (*LDA*) model on the dataset.
- Perform multiple cross-validation tasks at different k-fold values ($k = 3$, $k = 10$).

ELT

Much of the ***extract****,* ***load****, and* ***transform*** *(****ELT****) process from the previous report has been revised for this assignment. Notably, a* `make.dataset()` *function streamlines the process of parsing the source* `Liver.txt` *file into a compatible* `data.frame`*.*

```r
# Import the dataset as variable called "Liver".
invisible(setup.analysis(target = "Liver"))
```

```
## Use cached dataset? TRUE

## 'Liver' exists.

## Importing dataset...

## Parsing dataset from local file...

## Reading dataset from compressed cache file...

## Done.

## Dataset imported.

## Registered target dataset to global environment. Access using 'Liver'.

## Use 'Liver' to access underlying tbl_df.

## Dataset of type: 'tbl_df/tbl/data.frame'.

## tibble [345 x 7] (S3: tbl_df/tbl/data.frame)
##  $ blood.1 : int [1:345] 85 85 86 91 87 98 88 88 92 90 ...
##  $ blood.2 : int [1:345] 92 64 54 78 70 55 62 67 54 60 ...
##  $ blood.3 : int [1:345] 45 59 33 34 12 13 20 21 22 25 ...
##  $ blood.4 : int [1:345] 27 32 16 24 28 17 17 11 20 19 ...
##  $ blood.5 : int [1:345] 31 23 54 36 10 17 9 11 7 5 ...
##  $ drinks  : num [1:345] 0 0 0 0 0 0 0.5 0.5 0.5 0.5 ...
##  $ severity: Factor w/ 2 levels "Not Severe","Severe": 2 1 1 1 1 1 2 2 2 2 ...
```

The `setup.analysis()` function imports a dataset into the global environment with the name provided to the `target =` argument.

EDA

This section reviews the ***`Liver`*** *dataset. In an improvement over the previous report, it now incorporates an assessment of feature correlations.*

Response Encoding

```
# Recode the response.
liver <- Liver %>% make.response()
summary(liver)
```

```
##     blood.1          blood.2          blood.3
##  Min.   : 65.0   Min.   : 23.0   Min.   :  4.0
##  1st Qu.: 87.0   1st Qu.: 57.0   1st Qu.: 19.0
##  Median : 90.0   Median : 67.0   Median : 26.0
##  Mean   : 90.2   Mean   : 69.9   Mean   : 30.4
##  3rd Qu.: 93.0   3rd Qu.: 80.0   3rd Qu.: 34.0
##  Max.   :103.0   Max.   :138.0   Max.   :155.0
##     blood.4         blood.5          drinks
##  Min.   : 5.0   Min.   :  5.0   Min.   : 0.00
##  1st Qu.:19.0   1st Qu.: 15.0   1st Qu.: 0.50
##  Median :23.0   Median : 25.0   Median : 3.00
##  Mean   :24.6   Mean   : 38.3   Mean   : 3.46
##  3rd Qu.:27.0   3rd Qu.: 46.0   3rd Qu.: 6.00
##  Max.   :82.0   Max.   :297.0   Max.   :20.00
##        severity
##  Not Severe:200
##  Severe    :145
##
##
##
##
```

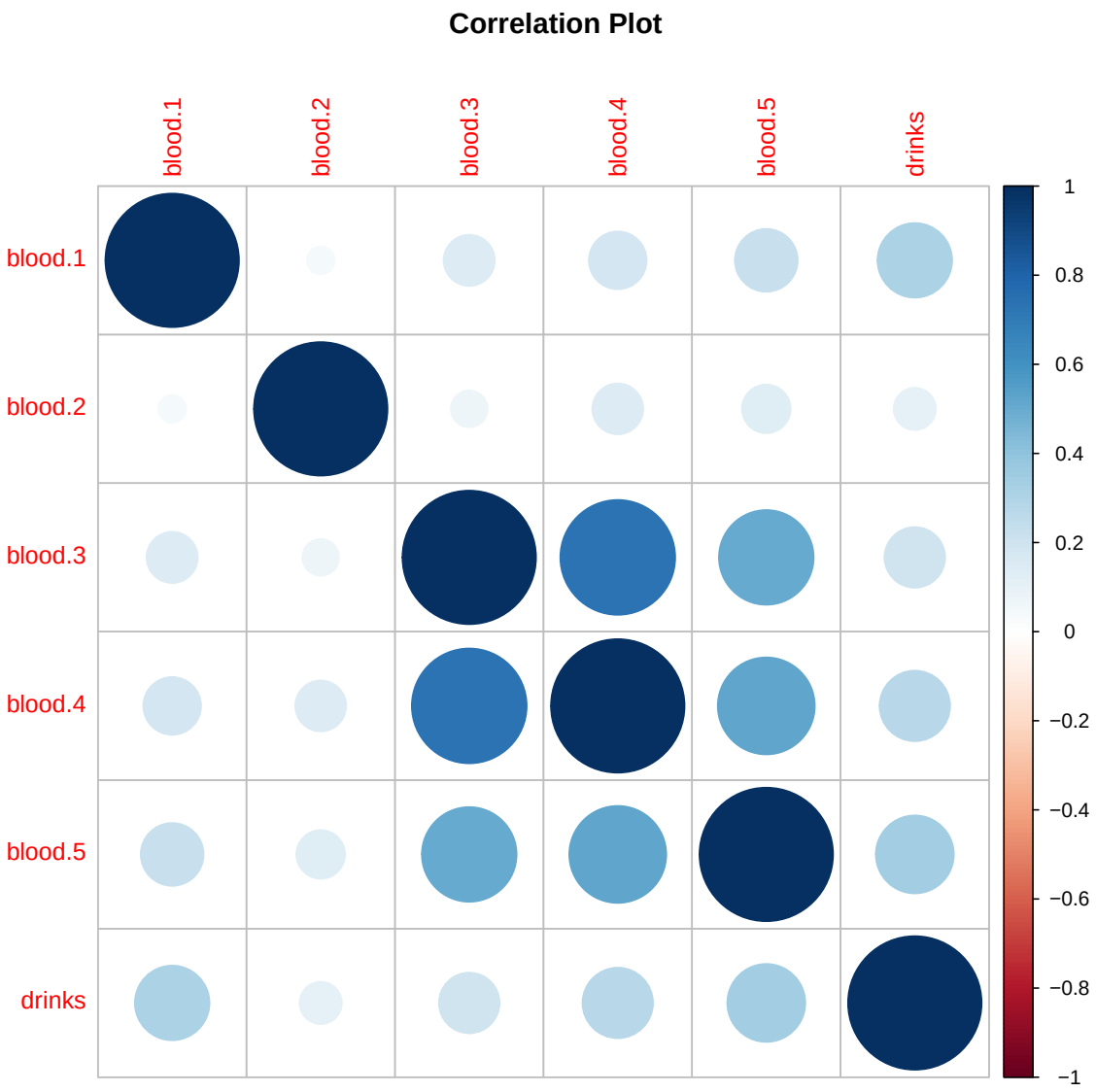
```
truth <- (liver$severity == "Severe")
```

Feature Correlations

```
liver.info <- analysis.eda(Liver, truth)
summary(liver.info)
```

```
##  n_samples n_features
##        345          7
## Label counts:
##   pos   neg total
##   145   200   345
## Class prior probabilities:
##    pos    neg
## 0.4203 0.5797
```

```
# Plot using corrplot::corrplot wrapper.
liver.corrplot <- corr.plot(
  liver, sig.level = 0.05, insig = "blank",
  title = "Correlation Plot",
  mar = c(1,1,2,1)
)
```



The data behind the above correlation plot is provided below:

```
## Feature correlation matrix:
##         blood.1 blood.2 blood.3 blood.4 blood.5 drinks
## blood.1  1.0000 0.04410 0.14770  0.1878  0.2223 0.3127
## blood.2  0.0441 1.00000 0.07621  0.1461  0.1331 0.1008
## blood.3  0.1477 0.07621 1.00000  0.7397  0.5034 0.2068
## blood.4  0.1878 0.14606 0.73967  1.0000  0.5276 0.2796
## blood.5  0.2223 0.13314 0.50344  0.5276  1.0000 0.3412
## drinks   0.3127 0.10080 0.20685  0.2796  0.3412 1.0000
```

Model Analysis

Models will be tracked using a named list `MODELS`:

```
# Create a global `models` variable to keep track of models.
MODELS <<- list()
# Also, declares a reusable formula.
.FORMULA <<- severity ~ .
```

Baseline Model Analysis

```
# Fit LDA with priors calculated from the input sample.
# Equivalent to: MASS::lda(severity ~ ., data = liver)
# We use `.` instead of `baseline` for ease of typing.
MODELS$. <- fit.model(quote(liver), algorithm = lda, formula = .FORMULA)

# Temporary variable `model` for current analysed model:
model <- MODELS$.
```

```
## Baseline LDA Summary:
## Call:
## lda(severity ~ ., data = liver)
##
## Prior probabilities of groups:
## Not Severe     Severe
##     0.5797     0.4203
##
## Group means:
##            blood.1 blood.2 blood.3 blood.4 blood.5 drinks
## Not Severe   89.81   68.34   29.82   25.99   43.17  3.393
## Severe       90.63   71.98   31.21   22.79   31.54  3.541
##
## Coefficients of linear discriminants:
##              LD1
## blood.1  0.08312
## blood.2  0.02263
## blood.3  0.05893
## blood.4 -0.11812
## blood.5 -0.01497
## drinks   0.06061
```

```
## Baseline model error rate:
## 0.2957
##
## Baseline misclassification table:
##
##              Not Severe Severe
##   Not Severe        165     67
##   Severe             35     78
##
## Total misclassified: 102
```

Optimal Priors (for Total Error Rate)

We can calculate the model that minimizes the total error rate while explicitly setting the prior probability for the positive class. As a reminder:

$$p = P(\text{ Severe }),\ q = (1 - p) = P(\text{Not Severe})$$

```
# Prepare next analysis.
MODELS$min_error <- LDA.ANALYSIS$calc.optimal.prior.error.rates(
  liver, truth,
  formula = severity ~ .,
  from = 0.001, to = 0.999, m = 25,
  minimize_cost = FALSE, penalty = 10
)
```

```
## ---------------------------
## Finding optimal error rates with explicit priors.
## # Penalty factor applied to cost function: x10
## # Checking 25 pos class priors in range [0.001, 0.999]:
## tibble [25 x 2] (S3: tbl_df/tbl/data.frame)
##  $ p: num [1:25] 0.001 0.0426 0.0842 0.1258 0.1673 ...
##  $ q: num [1:25] 0.999 0.957 0.916 0.874 0.833 ...
## Summary of minimum error rate search:
## Min. total cost: 703 (Penalty = x10)
## Min. total error: 0.2899
## Min. total miss: 703
## Optimal q/p ratio: 1.39904038384646
## Optimal priors: p = 0.4168, q = 0.58317
## Summary:
## tibble [1 x 7] (S3: tbl_df/tbl/data.frame)
##  $ p    : num 0.417
##  $ q    : num 0.583
##  $ fp   : int 67
##  $ fn   : int 33
##  $ miss : int 100
##  $ cost : num 703
##  $ error: num 0.29
## ---------------------------
```

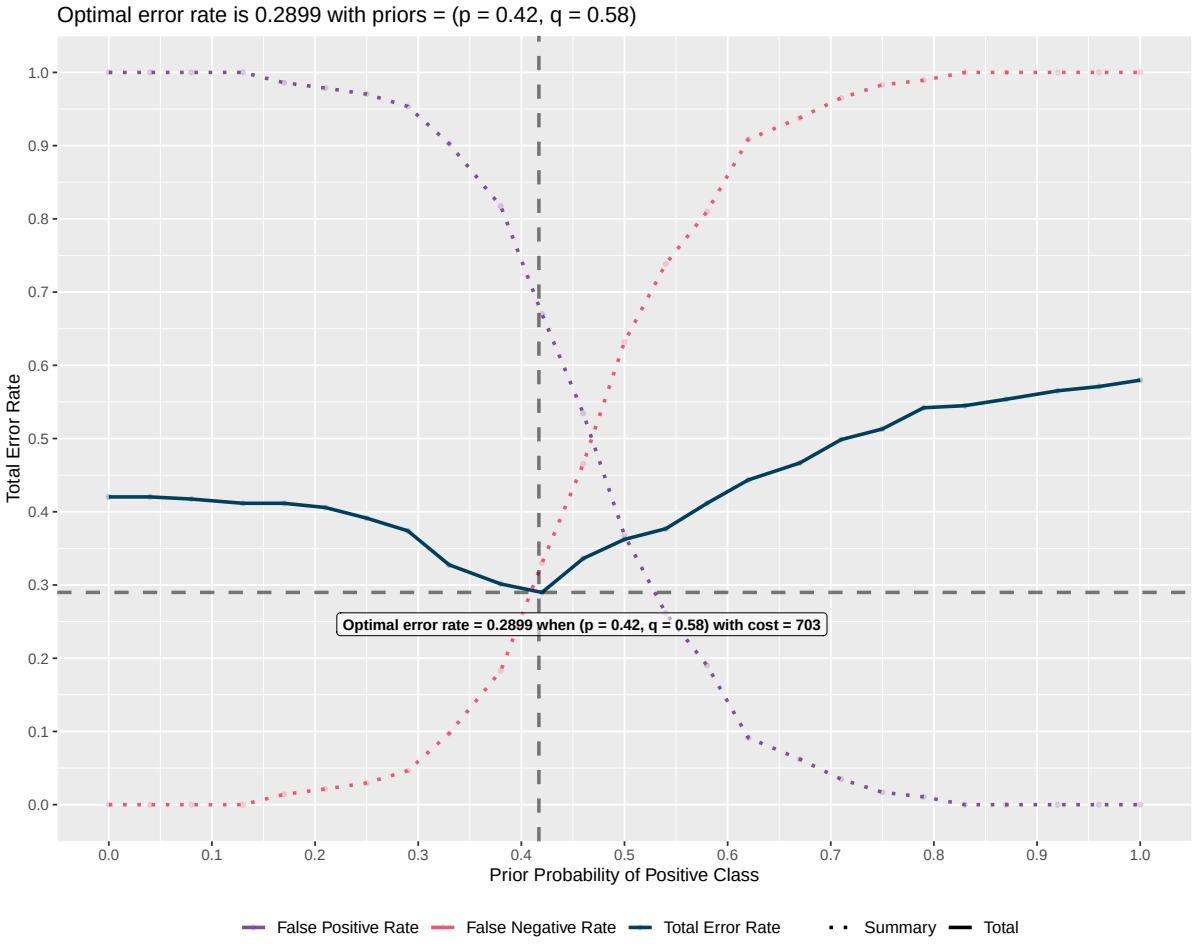
```
## ---------------------------
## Call:
## lda(severity ~ ., data = .data, prior = c(0.583166666666667,
## 0.416833333333333))
##
## Prior probabilities of groups:
## Not Severe     Severe
##     0.5832     0.4168
##
## Group means:
##            blood.1 blood.2 blood.3 blood.4 blood.5 drinks
```

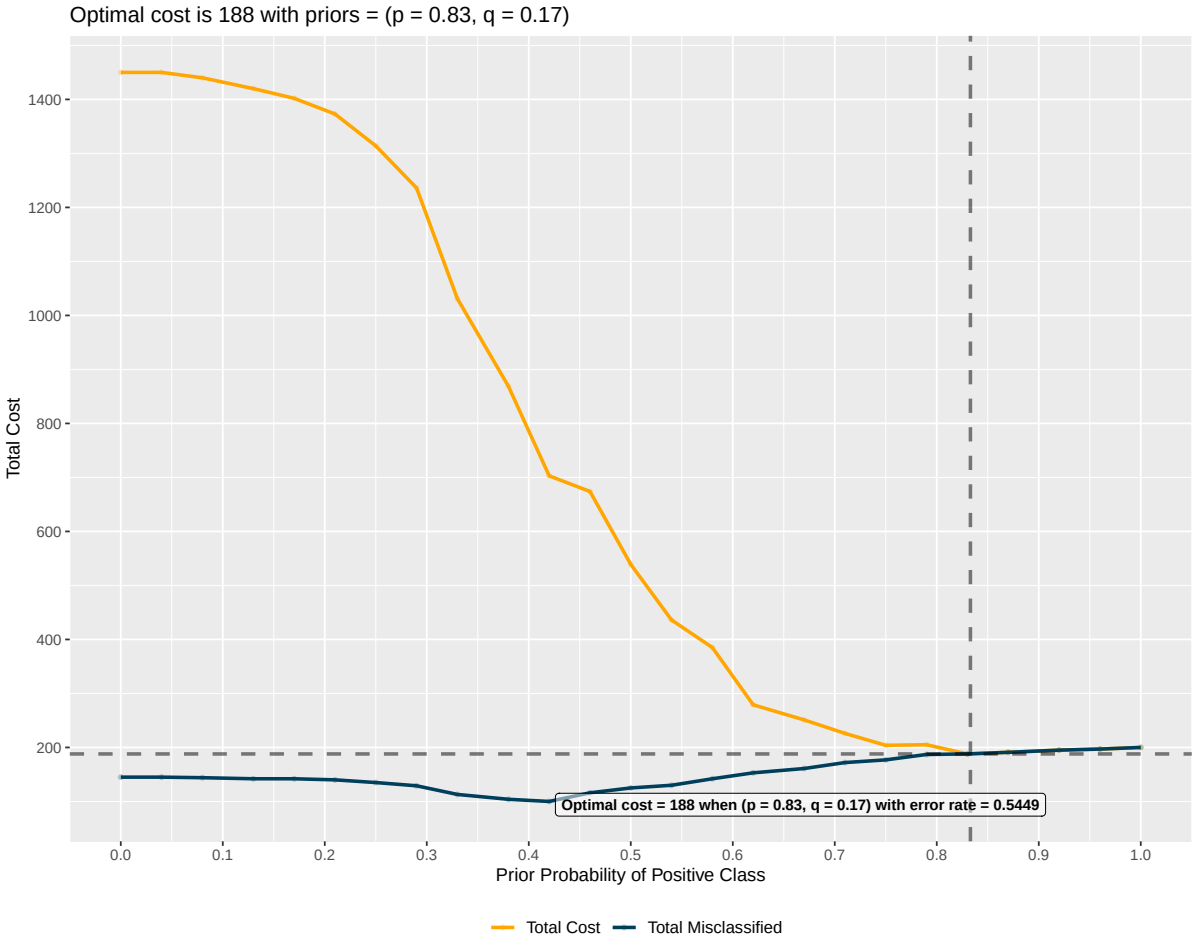
```
## Not Severe   89.81   68.34   29.82   25.99   43.17  3.393
## Severe       90.63   71.98   31.21   22.79   31.54  3.541
##
## Coefficients of linear discriminants:
##              LD1
## blood.1  0.08312
## blood.2  0.02263
## blood.3  0.05893
## blood.4 -0.11812
## blood.5 -0.01497
## drinks   0.06061
## Total cost of misclassification (Penalty x10): 703.00
## Total error rate (100 misclassifications): 0.29


##
##              Not Severe Severe
##   Not Severe        167     67
##   Severe             33     78
## --------------------------
```

Optimal Priors (for Total Cost)

Notice the current "cost" is a penalty factor of 10. The `penalty` value is a ratio that can be used to scale how a misclassification for the positive class affects the model.

We can also minimize our model based on the total cost. The following plots summarize both scenarios:


Optimal error rate is 0.2899 with priors = (p = 0.42, q = 0.58)
Total Error Rate
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Optimal error rate = 0.2899 when (p = 0.42, q = 0.58) with cost = 703
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Prior Probability of Positive Class
False Positive Rate
False Negative Rate
Total Error Rate
Summary
Total



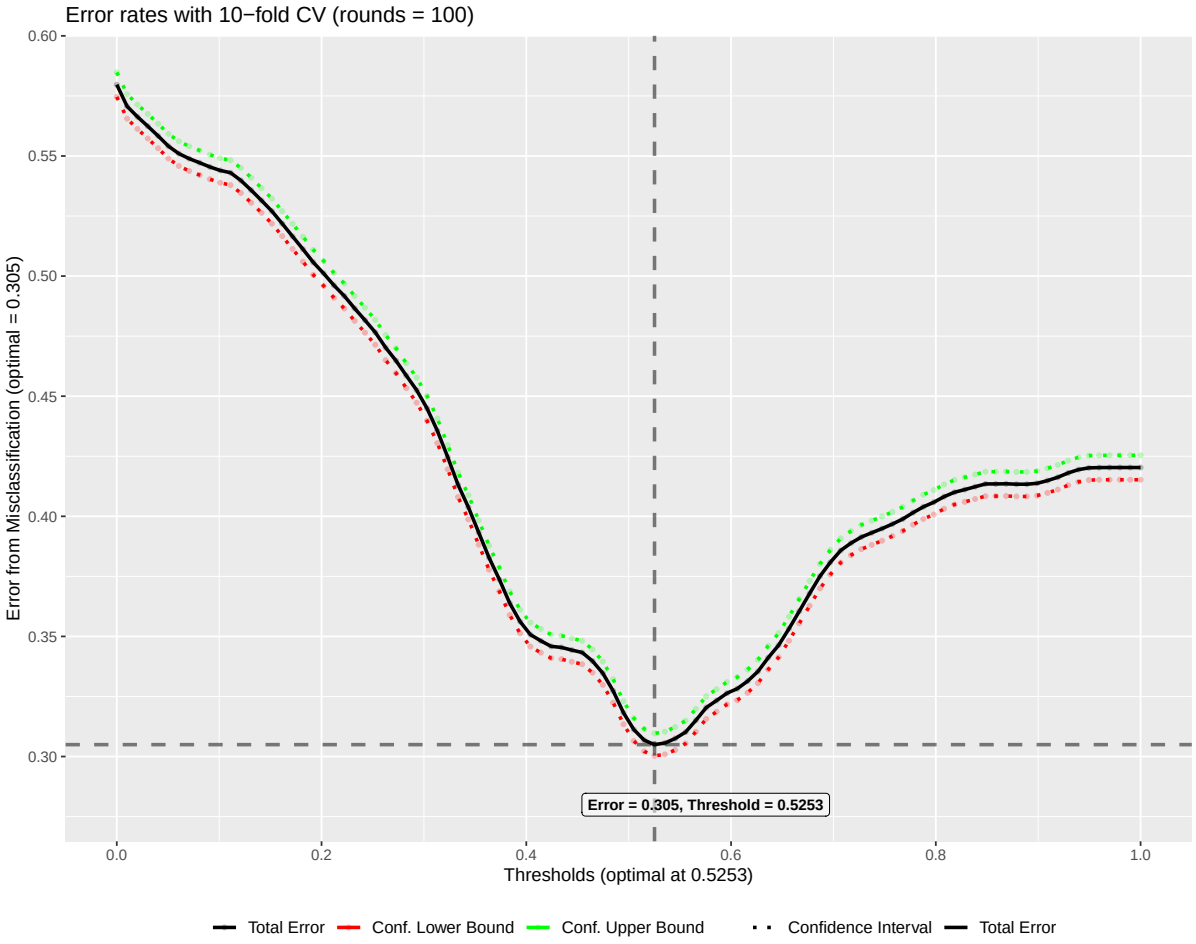

```
## ----------------------------
## Generating plot with priors of range [0.001, 0.999]
## # Searching for optimal values by minimizing total error rate.
## # Found optimal error rate from dataset: 0.2899
## # Total cost at optimal error rate: 703.00
## # Optimal priors for minimized total error rate: (p = 0.42, q = 0.58)
## # Cost calculated with penatly = 10
## ----------------------------
## ----------------------------
## Generating plot with priors of range [0.001, 0.999]
## # Searching for optimal values by minimizing total cost.
## # Found optimal cost from dataset: 188.00
## # Total error rate at optimal cost: 0.5449
## # Optimal priors for minimized cost: (p = 0.83, q = 0.17)
## # Cost calculated with penatly = 10
## ----------------------------
```

Cross-Validation Analysis

This section is a revision of the logistic regression CV. I was unable to complete the LDA CV analysis but I had reworked this analysis from my previous report in order to better understand how to accomplish the assignment. I hope to submit a revision before solutions are released for homework 3; the work is simply in progress.

10-fold CV

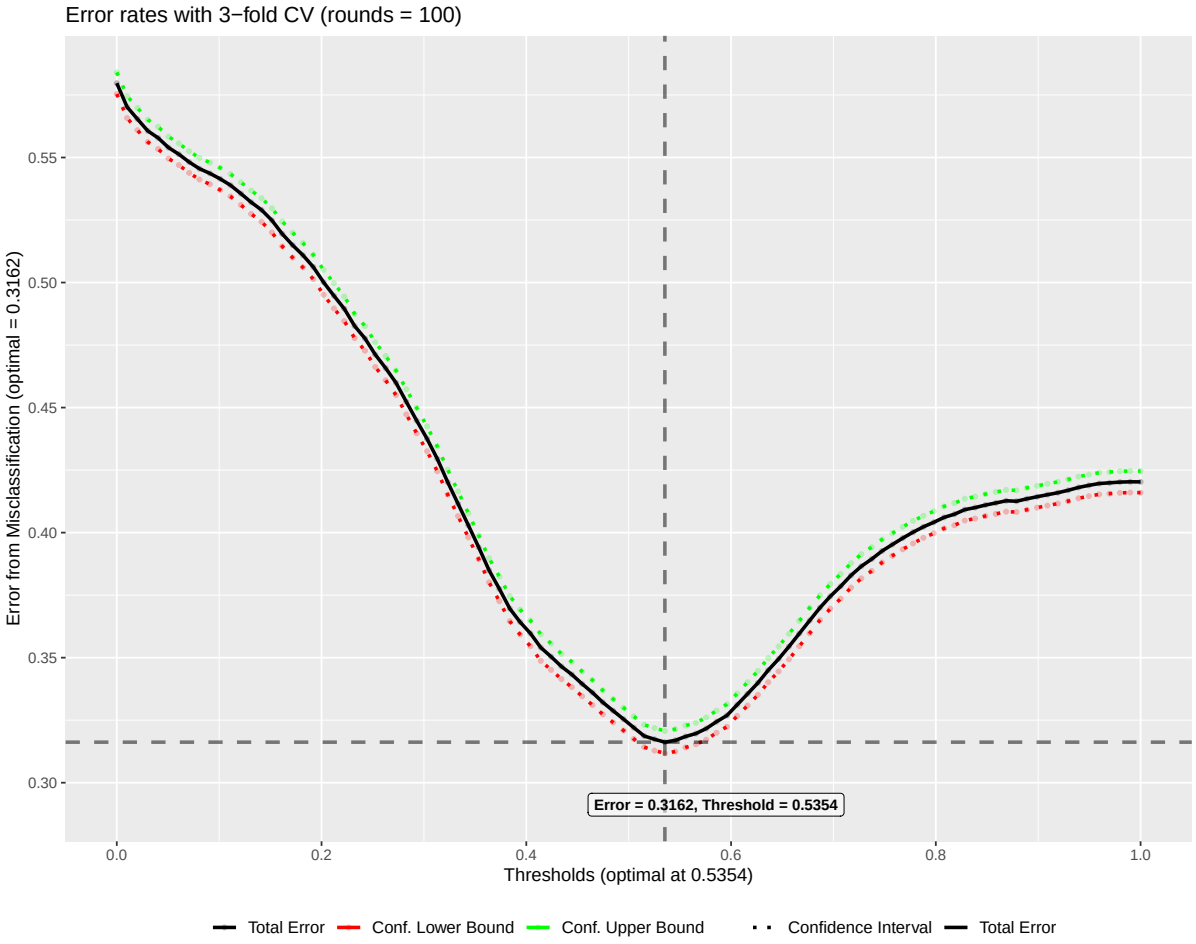
```
## ---------------------------
## Performing 10-fold CV:
## # Samples: 345
## # Thresholds: 100
## # k folds: 10 || # rounds: 100
## ---------------------------
## tibble [100 x 4] (S3: tbl_df/tbl/data.frame)
##  $ total.err : num [1:100] 0.58 0.571 0.566 0.562 0.558 ...
##  $ ci.low    : num [1:100] 0.575 0.566 0.561 0.557 0.553 ...
##  $ ci.high   : num [1:100] 0.585 0.576 0.571 0.567 0.563 ...
##  $ thresholds: num [1:100] 0 0.0101 0.0202 0.0303 0.0404 ...
## tibble [300 x 4] (S3: tbl_df/tbl/data.frame)
##  $ thresholds: num [1:300] 0 0 0 0.0101 0.0101 ...
##  $ category  : chr [1:300] "total.err" "ci.low" "ci.high" "total.err" ...
##  $ error     : num [1:300] 0.58 0.575 0.585 0.571 0.566 ...
##  $ linetype  : logi [1:300] TRUE FALSE FALSE TRUE FALSE FALSE ...
## ---------------------------
## Calculating 10-fold CV table:
## # Samples: 345
## # Threshold: 1
## # k folds: 10 || # rounds: 100
## ---------------------------
##                       Length Class  Mode
## results                  2   -none- list
## plot                     2   -none- list
## optimal_threshold        1   -none- numeric
## confusion_mat         4000   -none- numeric
## confusion_mat_summary    2   -none- list
```




```
## $plot
##
## $optimal_threshold
## [1] 0.5253

## Optimal threshold: 0.525252525252525
## $mean
##        [,1]  [,2]
## [1,] 167.61 73.08
## [2,]  32.39 71.92
##
## $sterr
##        [,1]   [,2]
## [1,] 0.1974 0.2038
## [2,] 0.1974 0.2038
```

3-fold CV

```
## ---------------------------
## Performing 3-fold CV:
## # Samples: 345
## # Thresholds: 100
## # k folds: 3 || # rounds: 100
## ---------------------------
## tibble [100 x 4] (S3: tbl_df/tbl/data.frame)
##  $ total.err : num [1:100] 0.58 0.57 0.565 0.561 0.558 ...
##  $ ci.low    : num [1:100] 0.575 0.566 0.561 0.556 0.553 ...
##  $ ci.high   : num [1:100] 0.584 0.575 0.57 0.565 0.562 ...
##  $ thresholds: num [1:100] 0 0.0101 0.0202 0.0303 0.0404 ...
## tibble [300 x 4] (S3: tbl_df/tbl/data.frame)
##  $ thresholds: num [1:300] 0 0 0 0.0101 0.0101 ...
##  $ category  : chr [1:300] "total.err" "ci.low" "ci.high" "total.err" ...
##  $ error     : num [1:300] 0.58 0.575 0.584 0.57 0.566 ...
##  $ linetype  : logi [1:300] TRUE FALSE FALSE TRUE FALSE FALSE ...
## ---------------------------
## Calculating 3-fold CV table:
## # Samples: 345
## # Threshold: 1
## # k folds: 3 || # rounds: 100
## ---------------------------
##                       Length Class  Mode
## results                  2   -none- list
## plot                     2   -none- list
## optimal_threshold        1   -none- numeric
## confusion_mat         1200   -none- numeric
## confusion_mat_summary    2   -none- list
```




```
## $plot
##
## $optimal_threshold
## [1] 0.5354

## Optimal threshold: 0.535353535353535
## $mean
##        [,1]  [,2]
## [1,] 168.56 76.46
## [2,]  31.44 68.54
##
## $sterr
##        [,1]   [,2]
## [1,] 0.3436 0.3301
## [2,] 0.3436 0.3301
```

Session Information

This document was generated from an R Markdown Notebook (See the ***vignettes/HW3_report.Rmd*** *in the project's sub-directory). The setup chunk for this document sets the root directory to the project root directory using the* ***here*** *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 datasets  utils
## [6] methods   base
##
## other attached packages:
##  [1] dplyr_1.0.7    tidyr_1.1.4    tibble_3.1.5
##  [4] forcats_0.5.1  ggplot2_3.3.5  foreach_1.5.1
##  [7] magrittr_2.0.1 MASS_7.3-54    mime_0.12
## [10] markdown_1.1   rmarkdown_2.11 knitr_1.36
##
## loaded via a namespace (and not attached):
##  [1] highr_0.9        jquerylib_0.1.4  compiler_4.1.1
##  [4] pillar_1.6.3     iterators_1.0.13 tools_4.1.1
##  [7] corrplot_0.90    bit_4.0.4        digest_0.6.28
## [10] jsonlite_1.7.2   evaluate_0.14    lifecycle_1.0.1
## [13] gtable_0.3.0     pkgconfig_2.0.3  rlang_0.4.11
## [16] cli_3.0.1        rstudioapi_0.13  yaml_2.2.1
## [19] xfun_0.26        fastmap_1.1.0    stringr_1.4.0
## [22] withr_2.4.2      hms_1.1.1        generics_0.1.0
## [25] vctrs_0.3.8      bit64_4.0.5      rprojroot_2.0.2
## [28] grid_4.1.1       tidyselect_1.1.1 glue_1.4.2
## [31] here_1.0.1       R6_2.5.1         fansi_0.5.0
## [34] vroom_1.5.5      farver_2.1.0     tzdb_0.1.2
## [37] readr_2.0.2      purrr_0.3.4      scales_1.1.1
## [40] codetools_0.2-18 htmltools_0.5.2  ellipsis_0.3.2
## [43] colorspace_2.0-2 renv_0.14.0      labeling_0.4.2
## [46] utf8_1.2.2       stringi_1.7.5    munsell_0.5.0
## [49] crayon_1.4.1
```