Homework 2: Classification Methods

Wen Li Teng

February 2 2020

Exploring Simulated Differences between LDA and QDA

```r
# Load packages
library(tidyverse)
```

```
## -- Attaching packages ------------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## -- Conflicts ---------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(rsample)
```

```
## Warning: package 'rsample' was built under R version 3.6.2
```

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.6.2
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(ggplot2)
```

```r
# 3(a) Repeat 1000 times

finderror <- function(rownum) {

  # 3(a)(i) Simulate a dataset
```

```r
set.seed(rownum)

n_obs <- 1000
min <- -1
max <- 1

q3_data <- tibble(X1 = runif(n_obs, min, max), X2 = runif(n_obs, min, max))

codeY <- function(simY) {
  if(simY >= 0)
    output <- TRUE
  if (simY < 0)
    output <- FALSE
  return(output)
}

q3_data <- q3_data %>%
  mutate (E = rnorm(1000, 0, 1)) %>%
  mutate (simY = X1 + X1^2 + X2 + X2^2 + E) %>%
  rowwise() %>%
  mutate(Y = codeY(simY))

# 3(a)(ii) Randomly split dataset
split <- initial_split(q3_data, prop = 0.7)
train <- training(split)
test <- testing(split)

# 3(a)(iii) Use training dataset to estimate LDA and QDA models
lda_q3 <- MASS::lda(Y ~ X1 + X1^2 + X2 + X2^2 + E, data = q3_data)
qda_q3 <- MASS::qda(Y ~ X1 + X1^2 + X2 + X2^2 + E, data = q3_data)

# 3(a)(iv) Calculate model's training and test error rate

# Training error rate
train_predicted_lda <- predict(lda_q3, newdata = train)
train_predicted_qda <- predict(qda_q3, newdata = train)
lda_cm_train <- table(train$Y, train_predicted_lda$class)
qda_cm_train <- table(train$Y, train_predicted_qda$class)

results_train <- as.data.frame(train) %>%
  mutate(lda.pred = (train_predicted_lda$class)) %>%
  mutate(qda.pred = (train_predicted_qda$class)) %>%
  summarize(lda.error = mean(Y != lda.pred),
            qda.error = mean(Y != qda.pred))

# Test error rate
test_predicted_lda <- predict(lda_q3, newdata = test)
test_predicted_qda <- predict(qda_q3, newdata = test)
lda_cm_test <- table(test$Y, test_predicted_lda$class)
qda_cm_test <- table(test$Y, test_predicted_qda$class)

results_test <- as.data.frame(test) %>%
  mutate(lda.pred = (test_predicted_lda$class)) %>%
    mutate(qda.pred = (test_predicted_qda$class)) %>%
    summarize(lda.error = mean(Y != lda.pred),
              qda.error = mean(Y != qda.pred))

  results <- list(results_train, results_test)

  return(results)
}
```

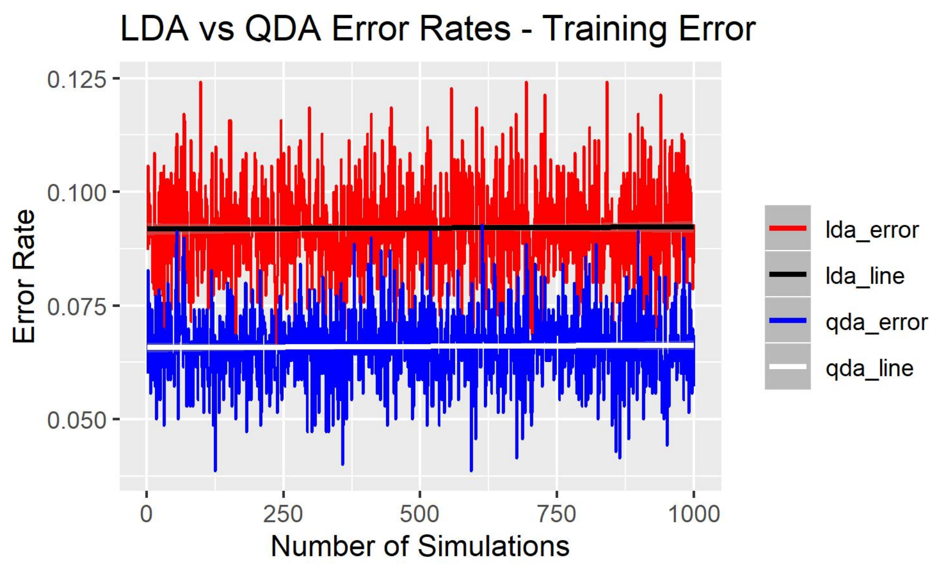
```r
sim <- 1:1000
lda_train_error <- vector("numeric", 1000)
qda_train_error <- vector("numeric", 1000)
lda_test_error <- vector("numeric", 1000)
qda_test_error <- vector("numeric", 1000)
results <- as.data.frame(cbind(sim,
                               lda_train_error,
                               qda_train_error,
                               lda_test_error,
                               qda_test_error))
for (i in 1:1000) {
  temp_result <- finderror(i)
  results$lda_train_error[i] <- temp_result[[1]][1]
  results$qda_train_error[i] <- temp_result[[1]][2]
  results$lda_test_error[i] <- temp_result[[2]][1]
  results$qda_test_error[i] <- temp_result[[2]][2]
  remove(temp_result)
}

results_new <- results
results_new$lda_train_error <- unlist(results_new$lda_train_error)
results_new$qda_train_error <- unlist(results_new$qda_train_error)
results_new$lda_test_error <- unlist(results_new$lda_test_error)
results_new$qda_test_error <- unlist(results_new$qda_test_error)


sim <- 1:1000
lda_error <- vector("numeric", 1000)
qda_error <- vector("numeric", 1000)
results <- as.data.frame(cbind(sim, lda_error, qda_error))
for (i in 1:1000) {
  temp_result <- finderror(i)
  results$lda_error[i] <- temp_result$lda.error
  results$qda_error[i] <- temp_result$qda.error
  remove(temp_result)
}
```

```r
# 3(b) Report results in tabular and graphical form

error_mean <- results_new %>%
  summarize(mean_lda_train_error = mean(lda_train_error),
            mean_qda_train_error = mean (qda_train_error),
            mean_lda_test_error = mean(lda_test_error),
            mean_qda_test_error = mean (qda_test_error)) %>%
  mutate(train_diff = mean_lda_train_error - mean_qda_train_error,
         test_diff = mean_lda_test_error - mean_qda_test_error)

write.csv(error_mean, "data/error_mean.csv", row.names = F)
```

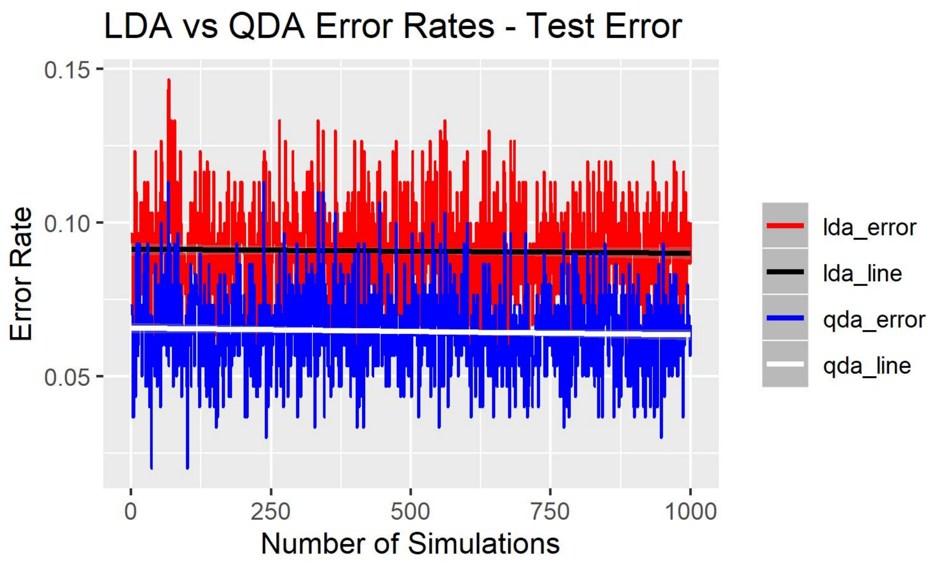
```r
error_mean <- read.csv("data/error_mean.csv")
error_mean
```

```
##   mean_lda_train_error mean_qda_train_error mean_lda_test_error
## 1           0.09202429           0.06599143          0.09064333
##   mean_qda_test_error train_diff  test_diff
## 1             0.06457 0.02603286 0.02607333
```

```r
ggplot(results_new, aes(x = sim)) +
  geom_line(aes(y = lda_train_error, colour = "lda_error")) +
  geom_smooth(aes(y = lda_train_error, colour = "lda_line"), method = "lm") +
  geom_line(aes(y = qda_train_error, colour = "qda_error")) +
  geom_smooth(aes(y = qda_train_error, colour = "qda_line"), method = "lm") +
  scale_colour_manual("", values = c("lda_error"="red",
                                     "lda_line" = "black",
                                     "qda_error"="blue",
                                     "qda_line" = "white")) +
  labs(title = "LDA vs QDA Error Rates - Training Error",
       x = "Number of Simulations",
       y = "Error Rate")
ggsave(filename="plots/plot1.jpeg", width = 5, height = 3)
```



```r
ggplot(results_new, aes(x = sim)) +
  geom_line(aes(y = lda_test_error, colour = "lda_error")) +
  geom_smooth(aes(y = lda_test_error, colour = "lda_line"), method = "lm") +
  geom_line(aes(y = qda_test_error, colour = "qda_error")) +
  geom_smooth(aes(y = qda_test_error, colour = "qda_line"), method = "lm") +
  scale_colour_manual("", values = c("lda_error"="red",
                                           "lda_line" = "black",
                                           "qda_error"="blue",
                                           "qda_line" = "white")) +
  labs(title = "LDA vs QDA Error Rates - Test Error",
       x = "Number of Simulations",
       y = "Error Rate")
ggsave(filename="plots/plot2.jpeg", width = 5, height = 3)
```



If the Bayes decision boundary is linear, we should expect QDA to perform better on the training and test sets. As mentioned above, according to James et al. (2013), the bias-variance trade-off is important to considering whether LDA or QDA will perform better. LDA is less flexible. It has lower variance but higher bias. QDA is more flexible. It has higher variance but lower variance. Consequently, QDA performs better on the training set by fitting it closely. At the same time, QDA performs better on the test set by approximating the non-linear decision boundary.