STAT-627 Project

Danny Tapp

Question 2:

How well do the variables age, sex, race/ethnicity, income, and education level predict the percentage of adults who achieve at least 150 minutes a week of moderate-intensity aerobic physical activity across US states?

```r
library(tidyverse)

### Load in data
food <- read_csv("food.csv")
```

```r
### Get the question about 150+ min of exercise a week and the necessary variables
df_model <- food |>
  filter(
    TopicID == "PA1",
    QuestionID == "Q043",
    !is.na(Data_Value),
    StratificationCategory1 %in% c("Age (years)", "Sex", "Race/Ethnicity", "Income", "Educ
  )

### Pivot wider so each stratum becomes it own column
df_wide <- df_model |>
  dplyr::select(LocationAbbr, YearStart,
         StratificationCategory1, Stratification1, Data_Value) |>
  unite(var, StratificationCategory1, Stratification1, sep = "_") |>
  pivot_wider(names_from = var, values_from = Data_Value)
```

```r
### Impute missing values using the median of each column
df_imp <- df_wide |>
  mutate(across(where(is.numeric), ~ replace_na(.x, median(.x, na.rm = TRUE))))
```

```r
### Extract the overall percentage values
df_overall <- food |>
  filter(
    TopicID == "PA1",
    QuestionID == "Q043",
    is.na(StratificationCategory1) |
      StratificationCategory1 %in% c("Overall", "Total") |
      Stratification1 %in% c("Overall", "Total", "OVR")
  ) |>
  dplyr::select(LocationAbbr, YearStart, PercentOverall = Data_Value)

### Join the two data sets
df_model2 <- df_imp |>
  left_join(df_overall, by = c("LocationAbbr", "YearStart"))
```

```r
### Predictor Matrix
X <- df_model2 |>
  select(-LocationAbbr, -YearStart, -PercentOverall) |>
  as.matrix()

### Response
y <- df_model2$PercentOverall
```

```r
library(glmnet)
```

```
Loading required package: Matrix


Attaching package: 'Matrix'

The following objects are masked from 'package:tidyr':

    expand, pack, unpack

Loaded glmnet 4.1-10
```

```r
### Standardize predictor matrix for ridge
X <- scale(X)
```

```r
set.seed(123)

### 10-fold cross-validation ridge regression
cv_ridge <- cv.glmnet(
  X, y,
  alpha = 0,
  nfolds = 10
)

### Lambda that minimizes cross-validation error
best_lambda <- cv_ridge$lambda.min

best_lambda
```

```
[1] 0.6391113
```

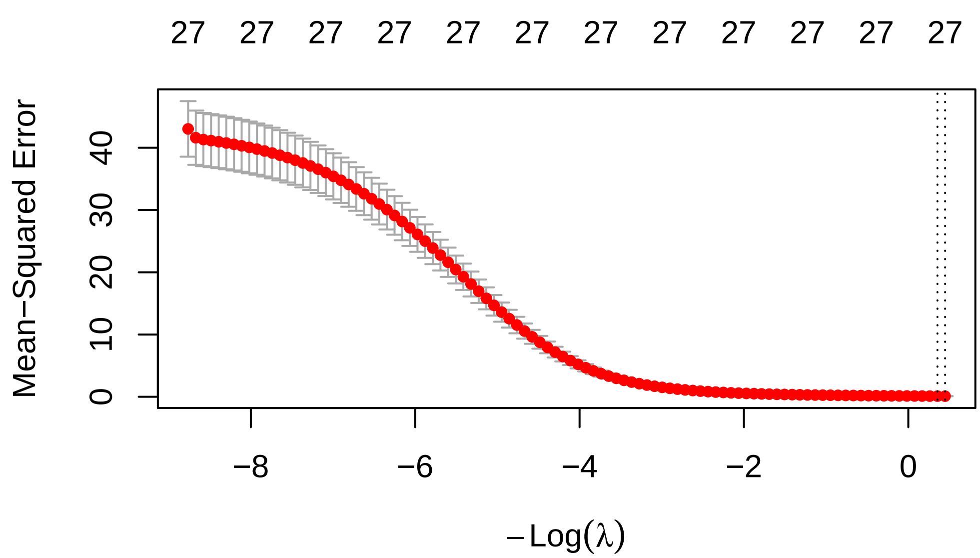
```r
### Fit ridge regression model using best lambda
ridge_final <- glmnet(
  X, y,
  alpha = 0,
  lambda = best_lambda
)

coef(ridge_final)
```

```
28 x 1 sparse Matrix of class "dgCMatrix"
                                                   s0
(Intercept)                               52.313836478
Income_$15,000 - $24,999                   0.082570466
Income_$25,000 - $34,999                   0.037845404
Income_$35,000 - $49,999                   0.117056342
Income_$50,000 - $74,999                   0.159823257
Income_$75,000 or greater                  0.280781509
Age (years)_18 - 24                        0.290150181
Race/Ethnicity_2 or more races             0.015693338
Age (years)_25 - 34                        0.504075021
Age (years)_35 - 44                        0.534949137
Age (years)_45 - 54                        0.669258522
Age (years)_55 - 64                        0.724984474
Age (years)_65 or older                    0.641806827
```

```
Race/Ethnicity_American Indian/Alaska Native -0.031537102
Education_College graduate                    0.446487837
Income_Data not reported                      0.312152639
Sex_Female                                    0.686421573
Education_High school graduate                0.327526534
Race/Ethnicity_Hispanic                      -0.004699734
Income_Less than $15,000                      0.082084394
Education_Less than high school               0.149554801
Sex_Male                                      0.512572179
Race/Ethnicity_Non-Hispanic Black             0.003946892
Race/Ethnicity_Non-Hispanic White             0.162085726
Education_Some college or technical school    0.303347341
Race/Ethnicity_Other                         -0.012596834
Race/Ethnicity_Asian                         -0.006700693
Race/Ethnicity_Hawaiian/Pacific Islander      0.025866220
```

```r
plot(cv_ridge)
```




```r
### Mean CV error for each lambda
cv_ridge$cvm
```

```
  [1] 43.03147921 41.61640129 41.31845795 41.15142142 40.96928446 40.77080540
  [7] 40.55466370 40.31946070 40.06372089 39.78589457 39.48436235 39.15744176
 [13] 38.80339634 38.42044758 38.00677926 37.56034756 37.07974591 36.56306982
 [19] 36.00861001 35.41478977 34.78018375 34.10356408 33.38395063 32.62066481
 [25] 31.81338585 30.96220808 30.06769747 29.13094518 28.15361554 27.13798555
 [31] 26.08697286 25.00414913 23.89372886 22.76043422 21.60979309 20.44769241
 [37] 19.28045962 18.11472645 16.95729698 15.81500470 14.69456192 13.60240772
 [43] 12.54456112 11.52648559 10.55297068  9.62803545  8.75485708  7.93572708
 [49]  7.17203622  6.46428876  5.81214572  5.21427544  4.66419219  4.16965332
 [55]  3.72414897  3.32399594  2.96586103  2.64627074  2.36174634  2.10903413
 [61]  1.88622613  1.69016382  1.51647600  1.36278307  1.22704428  1.10739536
 [67]  1.00209975  0.90947168  0.82789297  0.75587360  0.69209089  0.63539743
 [73]  0.58481110  0.53949728  0.49874927  0.46196964  0.42865374  0.39837534
 [79]  0.37077411  0.34554477  0.32242758  0.30120022  0.28189063  0.26399008
 [85]  0.24725793  0.23182344  0.21756529  0.20459397  0.19236128  0.18089199
 [91]  0.17043307  0.16025013  0.15107611  0.14244222  0.13410294  0.12652571
 [97]  0.11957724  0.11270228  0.10664839  0.09982398
```

```r
### Lambda with lowest CV error
cv_ridge$lambda.min
```

```
[1] 0.6391113
```

```r
### Lambda within 1 se of min
cv_ridge$lambda.1se
```

```
[1] 0.7014238
```

```r
### Best lambda's MSE
best_lambda <- cv_ridge$lambda.min
best_mse <- cv_ridge$cvm[cv_ridge$lambda == best_lambda]
```

```r
best_lambda
```

```
[1] 0.6391113
```

```r
best_mse
```

```
[1] 0.09982398
```

```r
### Take model coefficients and put them into a vector with their names
coef_vec <- as.numeric(coef(ridge_final))
names_vec <- rownames(coef(ridge_final))

### DF of coefficients
coef_df <- data.frame(
  variable = names_vec,
  coefficient = coef_vec,
  row.names = NULL
)

### Remove intercept
coef_df <- coef_df |>
  dplyr::filter(variable != "(Intercept)")

### Top 10 predictors based on largest aboslute coefficients
top10 <- coef_df |>
  dplyr::arrange(desc(abs(coefficient))) |>
  dplyr::slice(1:10)

top10
```

```
                        variable coefficient
1             Age (years)_55 - 64   0.7249845
2                      Sex_Female   0.6864216
3             Age (years)_45 - 54   0.6692585
4         Age (years)_65 or older   0.6418068
5             Age (years)_35 - 44   0.5349491
6                        Sex_Male   0.5125722
7             Age (years)_25 - 34   0.5040750
8      Education_College graduate   0.4464878
9  Education_High school graduate   0.3275265
10       Income_Data not reported   0.3121526
```

Question 5:

Can the percentage of adults in a state who engage in healthy behaviors (exercising and eating fruits/vegetables) be used to classify whether a state's obesity rate is above or below the national median?

```
### Get questions for fruit intake, vegetable intake, and obesity
### Get overall values
veggie_model <- food |>
  filter(
    QuestionID %in% c("Q018", "Q019", "Q036"),
    StratificationCategoryId1 == "OVR",
    !is.na(Data_Value)
  )
```

```
### Rename to make it easier to distinguish variables
veggie_clean <- veggie_model |>
  mutate(
    Var = case_when(
      QuestionID == "Q018" ~ "FruitUnhealthy",
      QuestionID == "Q019" ~ "VegUnhealthy",
      QuestionID == "Q036" ~ "Obesity"
    )
  )
```

```
### Pivot wider so each row is a state-year with all 3 variables
veggie_wide <- veggie_clean |>
  dplyr::select(LocationAbbr, YearStart, Var, Data_Value) |>
  pivot_wider(
    names_from = Var,
    values_from = Data_Value
  )
```

```
### Change to percent healthy
veggie_wide <- veggie_wide |>
  mutate(
    FruitHealthy = 100 - FruitUnhealthy,
    VegHealthy = 100 - VegUnhealthy
  )
```

```r
### Get national median obesity rate
national_median <- veggie_wide |>
  filter(LocationAbbr == "US") |>
  summarize(med = median(Obesity, na.rm = TRUE)) |>
  pull(med)
```

```r
### Classify which states are above or below the national median obesity
veggie_wide <- veggie_wide |>
  mutate(
    ObesityClass = ifelse(
      Obesity > national_median,
      "High",
      "Low"
    )
  )
```

```r
### Remove US overall
veggie_model_ready <- veggie_wide |>
  filter(!LocationAbbr == "US")

veggie_lqda <- veggie_model_ready |>
  dplyr::select(LocationAbbr:ObesityClass) |>
  drop_na()
```

```r
library(MASS)
```

```

Attaching package: 'MASS'

The following object is masked from 'package:dplyr':

    select
```

```r
### Fit LDA model
lda_fit <- lda(
  ObesityClass ~ FruitHealthy + VegHealthy,
  data = veggie_lqda
)
```

```r
lda_fit
```

```
Call:
lda(ObesityClass ~ FruitHealthy + VegHealthy, data = veggie_lqda)

Prior probabilities of groups:
     High       Low 
0.6772152 0.3227848 

Group means:
     FruitHealthy VegHealthy
High     58.60000   79.14299
Low      64.78039   81.69608

Coefficients of linear discriminants:
                     LD1
FruitHealthy  0.29609364
VegHealthy   -0.07452308
```

```r
### LDA coefficients
lda_fit$scaling
```

```
                     LD1
FruitHealthy  0.29609364
VegHealthy   -0.07452308
```

```r
### Look at averages between classes
veggie_lqda |>
  group_by(ObesityClass) |>
  summarise(
    n = n(),
    FruitHealthy = mean(FruitHealthy, na.rm = TRUE),
    VegHealthy   = mean(VegHealthy, na.rm = TRUE),
    Obesity      = mean(Obesity, na.rm = TRUE)
  )
```

```
# A tibble: 2 x 5
  ObesityClass     n FruitHealthy VegHealthy Obesity
  <chr>        <int>        <dbl>      <dbl>   <dbl>
```

```
1 High            107          58.6        79.1     34.4
2 Low              51          64.8        81.7     27.4
```

```r
### Priors
lda_fit$prior
```

```
     High       Low 
0.6772152 0.3227848 
```

```r
### Group means for each predictor by class
lda_fit$means
```

```
     FruitHealthy VegHealthy
High     58.60000   79.14299
Low      64.78039   81.69608
```

```r
### Fit QDA model
qda_fit <- qda(
  ObesityClass ~ FruitHealthy + VegHealthy,
  data = veggie_lqda
)

qda_fit
```

```
Call:
qda(ObesityClass ~ FruitHealthy + VegHealthy, data = veggie_lqda)

Prior probabilities of groups:
     High       Low 
0.6772152 0.3227848 

Group means:
     FruitHealthy VegHealthy
High     58.60000   79.14299
Low      64.78039   81.69608
```

```r
### Priors
qda_fit$prior
```

```
     High       Low 
0.6772152 0.3227848 
```

```r
### ### Group means for each predictor by class
qda_fit$means
```

```
     FruitHealthy VegHealthy
High     58.60000   79.14299
Low      64.78039   81.69608
```

```r
### Predict both LDA and QDA
lda_pred <- predict(lda_fit)
qda_pred <- predict(qda_fit)
```

```r
### LDA Confusion matrix
table(veggie_lqda$ObesityClass, lda_pred$class)
```

```
      
       High Low
  High   94  13
  Low    15  36
```

```r
### Classification rate
mean(lda_pred$class == veggie_lqda$ObesityClass)
```

```
[1] 0.8227848
```

```r
### QDA Confusion matrix
table(veggie_lqda$ObesityClass, qda_pred$class)
```

```
      
       High Low
  High   86  21
  Low     9  42
```

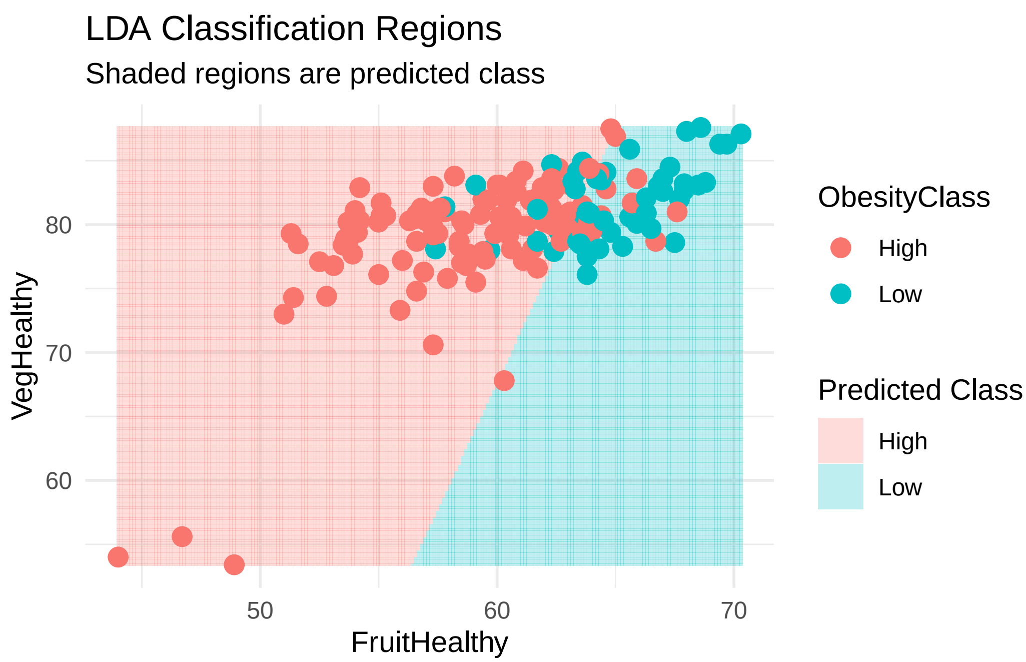
```r
### Classification Rate
mean(qda_pred$class == veggie_lqda$ObesityClass)
```

```
[1] 0.8101266
```

```r
### Grid for data points
grid <- expand.grid(
  FruitHealthy = seq(min(veggie_lqda$FruitHealthy), max(veggie_lqda$FruitHealthy), length.
  VegHealthy   = seq(min(veggie_lqda$VegHealthy),   max(veggie_lqda$VegHealthy),   length.
)

# Get LDA posterior for each grid point
grid$pred <- predict(lda_fit, newdata = grid)$class

### Plot
ggplot() +
  geom_tile(data = grid, aes(FruitHealthy, VegHealthy, fill = pred),
            alpha = 0.25) +
  geom_point(data = veggie_lqda,
             aes(FruitHealthy, VegHealthy, color = ObesityClass), size = 3) +
  labs(title = "LDA Classification Regions",
       subtitle = "Shaded regions are predicted class",
       fill = "Predicted Class") +
  theme_minimal()
```


LDA Classification Regions
Shaded regions are predicted class
VegHealthy
FruitHealthy
60
70
80
50
60
70
ObesityClass
High
Low
Predicted Class
High
Low