Homework 2

PSTAT131-231

Spam email classification

Background: The `spam` data are publicly available on the UCI Machine Learning repository. They comprise 2788 non-spam emails from saved work and personal emails, and 1813 emails filed as spam. Each observation in the dataset corresponds to one email.

The variables in the dataset comprise information about the frequencies of words and characters in the emails, and are as follows.

- 48 word frequency variables named `wf_WORD`, calculated as the percentage of words in the e-mail that match `WORD`.
- 6 character frequency variables named `cf_.` and `cf_1`, . . ., `cf_5` for non-alphanumeric characters, calculated as the percentage of characters in the e-mail that match the given character.
- 3 all-capital letter variables: `allcaps_avgrun` gives the average length of strings of consecutive captial letters; `allcaps_maxrun` gives the longest such run; `total_caps` gives the total number of capital letters in the email.
- A factor indicating whether the e-mail was considered spam (1) or not (0), i.e. unsolicited commercial e-mail.

Objective: You will build models to classify emails as spam and non-spam. This task has applications in developing spam filters.

```r
# spam dataset
load('data/spam.RData')

# center and scale features
spam <- spam %>% mutate(across(-spam, scale))
```

Throughout, you'll compare classification models based on estimates of misclassification error. As a preliminary step, create a random partition of the dataset into training and test sets. You'll use the *same* partitioning to ensure a fair comparison of models (this avoids confounding variability due to random partitioning with variability due to model).

```r
# partition into training and test sets
set.seed(41221)
spam_part <- spam %>% resample_partition(p = c('train' = 0.8, 'test' = 0.2))
```

Part 1: Exploratory plots

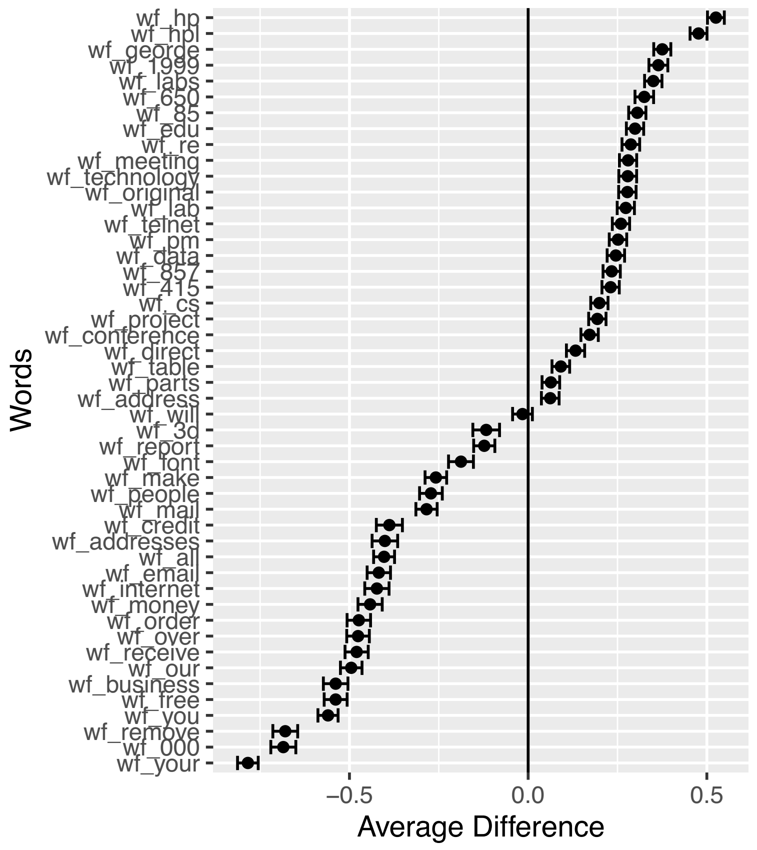
Question 1 (a)

Make a plot of the average difference in word frequencies between spam and non-spam emails for each word variable: plot the average difference on the horizontal axis and the word on the vertical axis. Order the words on the vertical axis by average frequency difference (see `fct_reorder`). Indicate the vertical line at 0 (`geom_vline`), and add error bars (`geom_errorbarh`). (*Hint*: first compute the mean and variance of the standardized frequencies for each word by spam classification; then compute the difference and a standard error. You will likely (but not necessarily) follow a select-gather-group-summarize-spread-mutate-plot pipeline.) Please ensure that the axes are appropriately labeled and legible.

```
# plot

#list of names
spam1 = spam %>%
  select(wf_make:wf_conference) %>%
  gather(name, value)

names = unique(spam1$name)

#calculating mean and variance
mv = spam %>%
  group_by(spam)%>%
  select(starts_with('wf')) %>%
  summarize(across(everything(), list(mean=mean, variance = var)))

#difference of mean and variance
diff_mv = mv[1,-1]-mv[2,-1]
mean_diff = diff_mv %>% select(contains('mean')) %>% gather(mean, values)
var_diff = diff_mv %>% select(contains('variance')) %>% gather(variance, values)

#variance of not spam and variance of spam

var1 = mv %>% filter(spam == 'not_spam') %>% select(contains('variance')) %>% t()
var1 = as.data.frame(var1)

var2 = mv %>% filter(spam == 'spam') %>% select(contains('variance')) %>% t()
var2 = as.data.frame(var2)
var2[,1]
```

```
##  [1] 1.034930e+00 7.309412e-02 9.092572e-01 2.529916e+00 1.105802e+00
##  [6] 1.382202e+00 2.136123e+00 1.845578e+00 1.621683e+00 9.589540e-01
## [11] 1.550485e+00 5.534903e-01 1.354729e+00 9.050045e-01 2.236886e+00
## [16] 1.505301e+00 1.989064e+00 1.572950e+00 7.788296e-01 2.391888e+00
## [21] 1.044752e+00 1.972791e+00 2.196587e+00 1.841068e+00 9.244875e-03
## [26] 1.255699e-02 9.836367e-05 3.200148e-01 4.542349e-04 5.115814e-02
## [31] 7.831403e-03 2.256118e-03 4.067875e-02 1.192328e-02 1.533093e-02
## [36] 1.345171e-01 4.018276e-01 5.197858e-02 4.243068e-02 1.888422e-01
## [41] 4.227615e-05 1.283626e-03 5.039481e-02 9.673503e-03 1.017873e-01
## [46] 2.160482e-02 5.889001e-02 8.810992e-03
```

```
#creating combined tibble
data = tibble(rows = c(1:48),
              name = names,
              mean = mean_diff$values,
              variance = var_diff$values,
              se = sqrt( (var1[,1]/table(spam$spam)[1]) + (var2[,1]/table(spam$spam)[2]) )
              )
data %>%
  ggplot(aes(mean, fct_reorder(name, mean))) +
  geom_point() +
  geom_vline(aes(xintercept = 0)) +
  geom_errorbarh(aes(xmax = mean + se, xmin = mean - se)) +
  xlab('Average Difference') +
  ylab('Words')
```



Question 1 (b)

Based on your plot, do you think that spam email is discernible from non-spam email from these features? Why or why not? Which variable(s) appear to be the strongest indicators of spam/non-spam? Which variables appear to be the weakest indicators?

Answer Based on my plot, I think that spam email is discernible from non-spam email because there are large differences between word frequencies in spam and not spam emails.

The strongest indicators of spam/not-spam are wf_remove, wf_000, and wf_your, because they had the largest differences in frequency averages, meaning that spam emails used these words a lot more in their emails than not-spam emails.

The weakest indicators of spam/not-spam are wf_report, wf_will, wf_address since the frequency difference between them are smallest, meaning that they are used a similar amount of times in spam and not-spam emails.

Part 2: Classification models

In this part you'll fit logistic regression and discriminant analysis classifiers.

Question 2 (a)

Train a logistic regression model on the training partition and predict the class labels on the test partition using a probability threshold of $\hat{p} \geq 0.5$ to determine the class labels from the estimated class probabilities. Display a table of the misclassification error rates. Are the rates much different for spam and nonspam emails?

```
# fit logistic regression model on training partition
fit = glm(spam ~ ., data = spam_part$train, family = 'binomial')

# compute predicted probabilities on test partition
p_hat_glm = predict(fit, spam_part$test, type = 'response')

# convert to classes using probability threshold 0.5
y_hat_glm <- factor(p_hat_glm > 0.5, labels = c('not_spam', 'spam'))

# cross-tabulate with true labels
spam_part_test = as.data.frame(spam_part$test) %>% unnest(everything())
error_glm = table(spam_part_test$spam, y_hat_glm)
error_glm/rowSums(error_glm)
```

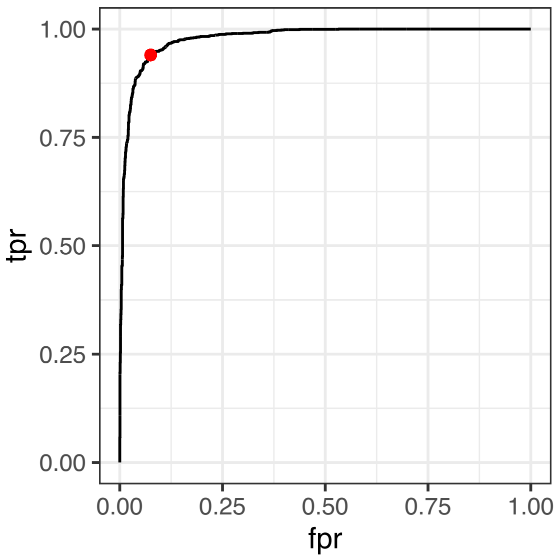
```
##           y_hat_glm
##              not_spam       spam
##   not_spam 0.94326241 0.05673759
##   spam     0.10924370 0.89075630
```

Answer The error rates are similar for spam (0.109) and nonspam emails (0.057).

Question 2 (b)

Construct an ROC curve for the logistic regression model and choose an optimal threshold. Show a plot of the curve with the true postitive and false positive rates at the optimal threshold indicated by a red point.

```
# store training labels for use in constructing ROC
spam_part_train <- as.data.frame(spam_part$train) %>% unnest(everything())
label = spam_part_train$spam

# compute predictions and performance metrics
preds_glm <- predict(fit, spam_part$train, type = 'response')
prediction_glm = prediction(predictions = preds_glm,
                            labels = label)
perf_glm = performance(prediction.obj = prediction_glm, 'tpr' , 'fpr')

# convert tpr and fpr to data frame and calculate youden statistic
rates_glm = tibble(fpr = slot(perf_glm, 'x.values'),
                   tpr = slot(perf_glm, 'y.values'),
                   thresh = slot(perf_glm, 'alpha.values')) %>%
  unnest(everything()) %>%
  mutate(youden = tpr-fpr)

# select optimal threshold
optimal_thresh <- rates_glm %>%
  slice_max(youden)

# plot
rates_glm %>%
  ggplot(aes(x=fpr, y = tpr)) +
  geom_line() +
  geom_point(aes(x=optimal_thresh$fpr, y = optimal_thresh$tpr), color = 'red') +
  theme_bw()
```



Question 2 (c)

Recalculate the misclassification errors after applying your optimal threshold, and display a table of the error rates. Store the table as `error_rates_glm` for future reference. How do the rates change after adjusting the threshold?

```
# convert to classes using optimal probability threshold
y_hat_glm2 <- factor(p_hat_glm > optimal_thresh$thresh, labels = c('not_spam', 'spam'))

# cross-tabulate with true labels
y2 = as.data.frame(spam_part$test)
error_rates_glm = table(y2$spam, y_hat_glm2)
error_rates_glm/rowSums(error_rates_glm)
```

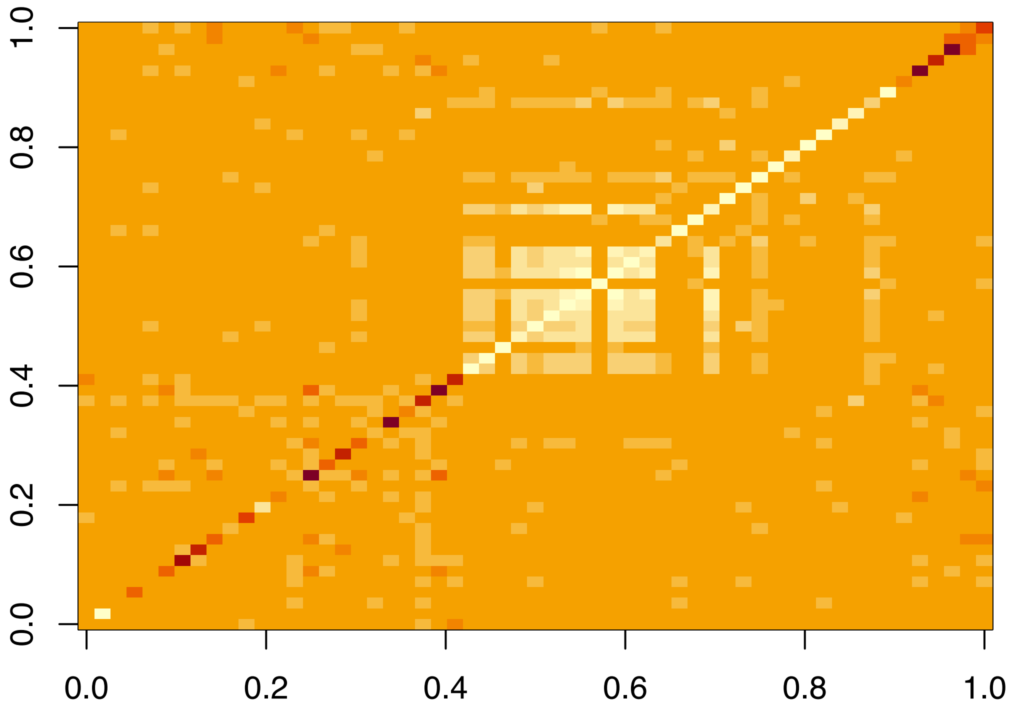
```
##           y_hat_glm2
##              not_spam       spam
##   not_spam 0.92375887 0.07624113
##   spam     0.08403361 0.91596639
```

Answer We can see that after changing using the optimal threshold, the not_spam column slightly decreased and the spam column slightly increased.

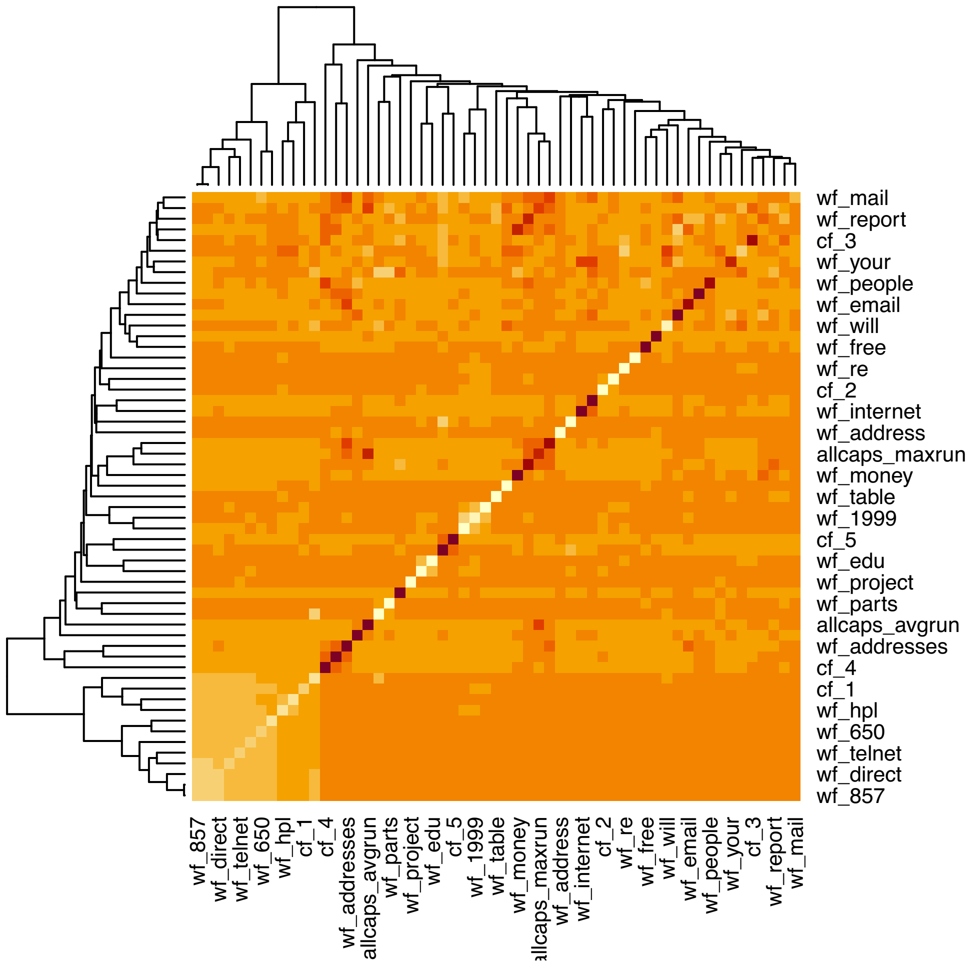
Question 2 (d)

Extract the feature covariance matrices for spam emails and non-spam emails from the training partition. Construct a heatmap of the difference between the covariance matrices (see `image()` or `geom_raster()`). Do the covariance matrices seem similar? Which discriminant method (linear or quadratic) seems more appropriate?

```r
# feature covariance matrix for spam group
spam_part_train = as.data.frame(spam_part$train) %>% unnest(everything())
spam_cov <- spam_part_train %>%
  filter(spam == 'spam') %>%
  select(-spam) %>%
  cov()

# feature covariance matrix for nonspam group
spam_ncov <- spam_part_train %>%
  filter(spam == 'not_spam') %>%
  select(-spam) %>%
  cov()

# heatmap of differences
diffheatmap <- spam_cov - spam_ncov
image(diffheatmap)
```



```r
heatmap(diffheatmap)
```



Answer From the heatmap we can see that many of the differences between the the 2 covariance matrices are close to 0. Therefore, we can say that the 2 matrices are similar and because of that similarly, we can use a linear discriminant analysis method to analyzed this further.

Question 2 (e)

Assume that the covariance matrices are similar enough that the LDA assumption is not unreasonable. Fit an LDA classifier to the training partition and predict class labels on the test partition using a probability threshold of 0.5. Compute the misclassification errors, and display the error rates as a table

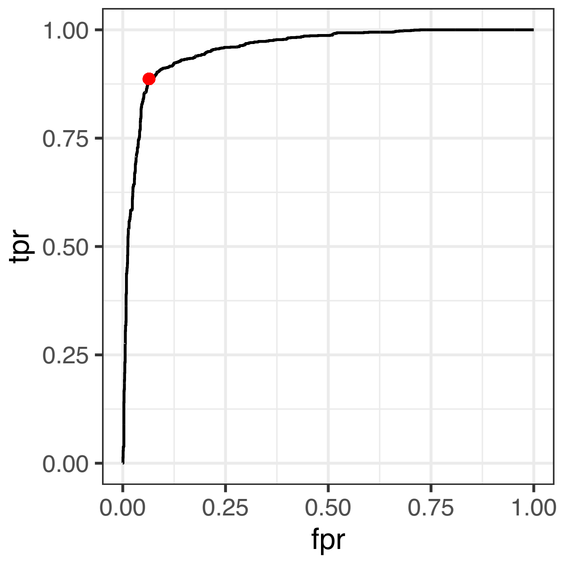
```
# fit lda model on training partition
lda_mod <- MASS::lda(spam ~ ., data = spam_part$train, method = 'mle')

# predict on test partition
lda_predict <- predict(lda_mod, spam_part$test)

# compute misclassification errors
spam.test <- as.data.frame(spam_part$test)
errors_lda <- table(spam.test$spam, lda_predict$class)
errors_lda / rowSums(errors_lda)
```

```
##
##                not_spam       spam
##   not_spam 0.94858156 0.05141844
##    spam     0.17086835 0.82913165
```

Question 2 (f)

Construct an ROC curve for the LDA model and choose an optimal threshold. Show a plot of the curve with the true postitive and false positive rates at the optimal threshold indicated by a red point.

```
# store training labels for use in constructing ROC
labels <- spam_part$train %>%
as.data.frame() %>%
dplyr::select(spam)

# compute estimated probabilities on the training partition
lda_preds <- predict(lda_mod, spam_part$train)

# compute predictions and performance metrics
prediction_lda <- prediction(predictions = lda_preds$posterior[, 2],
labels = labels)
perf_lda = performance(prediction.obj = prediction_lda, 'tpr' , 'fpr')

# convert tpr and fpr to data frame and calculate youden statistic
rates_lda = tibble(fpr = slot(perf_lda, 'x.values'),
                    tpr = slot(perf_lda, 'y.values'),
                    thresh = slot(perf_lda, 'alpha.values')) %>%
  unnest(everything()) %>%
  mutate(youden = tpr-fpr)

# select optimal threshold
optimal_thresh2 <- rates_lda %>%
  slice_max(youden)

# plot
rates_lda %>%
  ggplot(aes(x=fpr, y = tpr)) +
  geom_line() +
  geom_point(aes(x=optimal_thresh2$fpr, y = optimal_thresh2$tpr), color = 'red') +
  theme_bw()
```



Question 2 (g)

Recalculate the misclassification errors after applying your optimal threshold, and display a table of the error rates. Store the table as `error_rates_lda` for future reference.

```
# convert to classes using optimal probability threshold
y_hat_lda <- factor(lda_predict$posterior[,2]  > optimal_thresh2$thresh, labels = c('not_spam', 'spam'))

# cross-tabulate with true labels
y3 = as.data.frame(spam_part$test)
error_rates_lda = table(y3$spam, y_hat_lda)
error_rates_lda/rowSums(error_rates_lda)
```

```
##           y_hat_lda
##              not_spam       spam
##   not_spam 0.92021277 0.07978723
##   spam     0.09803922 0.90196078
```

Question 2 (h)

Compute the total misclassification error for each model. Which method performs best?

```
# total misclassification error
glm_compare_error = table(spam_part_test$spam, y_hat_glm)
glm_compare_error/rowSums(glm_compare_error)
```

```
##           y_hat_glm
##              not_spam       spam
##   not_spam 0.94326241 0.05673759
##   spam     0.10924370 0.89075630
```

```
mean(spam_part_test$spam != y_hat_glm)
```

```
## [1] 0.07709012
```

```
lda_compare_error = table(spam_part_test$spam, y_hat_lda)
lda_compare_error/rowSums(lda_compare_error)
```

```
##           y_hat_lda
##              not_spam       spam
##   not_spam 0.92021277 0.07978723
##   spam     0.09803922 0.90196078
```

```
mean(spam_part_test$spam != y_hat_lda)
```

```
## [1] 0.08686211
```

Answer From this we can see that the the glm model (logistic regression model) does a better job at predicting because it has a lower misclassifaction error than the lda (linear discriminant model).

Theory

Part 3: Logistic regression

Question 3 (a)

Let:

$$p = \frac{1}{1 + \exp\{-\mathbf{x}'\beta\}}$$

Show:

$$\log\left(\frac{p}{1-p}\right) = \mathbf{x}'\beta$$

<u>Proof</u>

$$p(1+e^{-x'\beta}) = 1$$

$$p + pe^{-x'\beta} = 1$$

$$pe^{-x'\beta} = 1-p$$

$$\frac{p}{1-p} = e^{x'\beta}$$

$$\log\left(\frac{1-p}{p}\right) = x'\beta \checkmark$$

Question 3 (b)

Let $Z = \mathbf{x}'\beta + \epsilon$ where $\epsilon \sim \text{logistic}(0,1)$ and $\mathbf{x}$ is fixed. Define $Y = I(Z > 0)$. Show that:

$$P(Y = 1) = \frac{1}{1 + \exp\{-\mathbf{x}'\beta\}}$$

Show that:

$$P(Y=1) = \frac{1}{1+\exp\{-x'\beta\}}$$

$$P(Y=1) = P(Z>0)$$

$$= P(x'\beta + \varepsilon > 0)$$

$$= P(\varepsilon > -x'\beta)$$ and Some $\varepsilon \sim \log(0,1)$

$$= P(\varepsilon > -x'\beta)$$

$$= 1 - \frac{1}{1+e^{x'\beta}} = \frac{e^{x'\beta}}{1+e^{x'\beta}}$$

$$= \frac{1}{e^{-x'\beta}+1} \checkmark$$