STAT 33B Homework 7

Ming Fong (3035619833)

Dec 10, 2020

This homework is due **Dec 10, 2020** by 11:59pm PT.

Homeworks are graded for correctness.

As you work, write your answers in this notebook. Answer questions with complete sentences, and put code in code chunks. You can make as many new code chunks as you like.

Please do not delete the exercises already in this notebook, because it may interfere with our grading tools.

You need to submit your work in two places:

- Submit this Rmd file with your edits on bCourses.
- Knit and submit the generated PDF file on Gradescope.

If you have any last-minute trouble knitting, **DON'T PANIC**. Submit your Rmd file on time and follow up in office hours or on Piazza to sort out the PDF.

SpamAssassin Email Data

The SpamAssassin Email Data set is a collection of email messages used to train the SpamAssassin software to detect spam. The email messages are divided into legitimate "ham" emails and illegitimate "spam" emails. Each email is in a separate plain text file.

In this assignment, you'll only use a collection of "ham" emails. You can find the emails in the file `emails.zip` on the bCourse. You will need to unzip the file before proceeding with Exercise 1.

This data set is originally from the Apache SpamAssassin project.

Exercise 1

The `readLines` function reads lines of text from a file and returns them in a character vector with one element for each line. The first argument is the path to the file. By default, the function will read all of the lines in the file.

Write a function `read_email` that reads all of the text in a single email file. Your function should have a parameter `file` to set the path to the file. Your function should collapse all of the lines in the file into a single string with lines separated by the newline character `\n`.

Show that your function works for 3 of the email files.

Hint: The `paste` function is relevant here.

YOUR ANSWER GOES HERE:

```
read_email = function(file) {
    lines = readLines(file)
    paste(lines, sep = "", collapse = "\n")
}
```

```
x = read_email("data/emails/easy_ham/0001.ea7e79d3153e7469e7a9c3e0af6a357e")
y = read_email("data/emails/easy_ham/0002.b3120c4bcbf3101e661161ee7efcb8bf")
z = read_email("data/emails/easy_ham/0003.acfc5ad94bbd27118a0d8685d18c89dd")
```

Exercise 2

Write a function `read_email_all` that reads all of the files in the email directory and returns a character vector with one element for each email. Your function should have a parameter `dir` to set the path to the directory. Your function should call the `read_email` function from Exercise 1.

Make sure not to put other files in the email directory!

After writing your function, use it to read all of the email files into a character vector called `emails`.

How many email files are there?

Hint: The `list.files` function is relevant here.

YOUR ANSWER GOES HERE:

```
read_email_all = function(dir) {
    files = list.files(dir, full.names = TRUE)
    vapply(files, read_email, "a")
}

emails = read_email_all("data/emails/easy_ham")
length(emails)
```

```
## [1] 2551
```

Exercise 3

Use stringr and regular expressions to write a function `extract_email_addr` that extracts all email addresses from a character vector. Your function should have a parameter `x` for the character vector, and should return a character vector with one element for each email address (duplicates are okay).

For simplicity, you can assume the formatting rules for email addresses are that they:

1. Must contain exactly one at-symbol `@`
2. Can also contain any number of letters, numbers, or characters in `._-`

Test your function on some made up strings and also on one of the email messages.

Hint: Using character classes `[ ]` in the regex pattern is important here.

Note: It's not necessary for this exercise, but if you're curious about the actual rules for email addresses, see Section 3.4.1 of RFC 5322, or this Wikipedia article.

YOUR ANSWER GOES HERE:

```
library(stringr)
extract_email_addr = function(x) {
    unlist(str_extract_all(x, "[[:alnum:]|.|_|-]+@[[:alnum:]|\\.|_|-]+"))
}
head(extract_email_addr(emails[1]))
```

```
## [1] "exmh-workers-admin@redhat.com"  "exmh-workers-admin@example.com"
## [3] "zzzz@localhost.netnoteinc.com"  "zzzz@localhost"
## [5] "zzzz@localhost"                 "zzzz-exmh@example.com"
```

Exercise 4

Using your `extract_email_addr` function and the email message data:

1. How many different email addresses appear in the emails?
2. Which 5 email addresses appear the most?

YOUR ANSWER GOES HERE:

```
addresses = unlist(sapply(emails, extract_email_addr), use.names = FALSE)
addresses_count = table(addresses)
# Number of unique addresses
length(addresses_count)
```

```
## [1] 3958
```

```
# Top 5 addresses
head(sort(addresses_count, decreasing = TRUE), 5)
```

```
## addresses
##               jm@localhost        fork-admin@xent.com
##                       4136                       2948
##          fork@example.com      fork-request@xent.com
##                       2774                       2107
## yyyy@localhost.example.com
##                       1699
```

Exercise 5

The part of an email address after the `@` is called the *domain*. The domain refers to a website, so it usually contains at least one dot. For example, Cal email addresses use the domain `berkeley.edu`, which is also the Cal website address.

Which 10 domains are the most common among the email addresses you extracted from the email data?

How many domains in the email addresses end in `.edu`?

YOUR ANSWER GOES HERE:

```
domains = substring(str_extract(addresses, "@.+"), 2)
domains_count = sort(table(domains), decreasing = TRUE)
# 10 most common domains
head(domains_count, 10)
```

```
## domains
##             example.com                xent.com               localhost
##                    6887                    6056                    5392
##           freshrpms.net              jmason.org example.sourceforge.net
##                    3643                    2854                    2068
##   localhost.example.com   lists.sourceforge.net              redhat.com
##                    1838                    1716                    1331
##                linux.ie
##                    1081
```

```
edu_domains = domains[str_detect(domains, "\\.edu$")]
# Number of .edu domains
edu_domains_count = table(edu_domains)
length(edu_domains_count)
```

```
## [1] 53
```