Stat 133, Fall 2014
Homework 6: Text Manipulation: Creating Spam-related Variables
Due: Thursday, Wednesday Nov 6, 11:55pm

About the Data

For this homework, you will be creating some basic spam filters for e-mail. The emails you will use for this assignment are in the list `Emails` stored as an .Rda file at
`http://stat.berkeley.edu/~nolan/stat133/data/Emails.rda`

Each element of the list `Emails` contains one e-mail message. Each e-mail is itself a list consisting of three elements:

- The element named "header" is a named character vector, where each name corresponds to a key in the email header and the value of the element corresponds to the text following the : in the key:value of the header.
- The element named "body" is itself a list, the first element of which is named "text" and contains the body of the email message. This element is a character vector, with one string per line in the email message.
  A second element, if it exists, is named "attachments." This element is a list containing one element per attachment. The individual attachment element is a list of two elements – one containing information about the format of the attachment and the other containing and the contents of the attachment.
- The element named spam is a logical vector of length 1 that indicates whether the message is spam (TRUE) or ham (FALSE).

The Assignment

Your task is to write four functions that return information about the header/body of an e-mail, and a fifth function that calls these three functions to create a basic spam filter.

On bSpace, turn in a .R file containing the code for these five functions. As before, name the file `LastnameFirstname_HW6.R`**. A skeleton file is provided for you and contains the names I want you to use for the input and output. Please keep these names as they are.**

Function spamCounts

`spamCounts()` takes as input two arguments: first, the subject of an e-mail as a character string (e.g., `Emails[[1]]$header["Subject"]`) and second, the body text of an e-mail as a character vector (e.g., `Emails[[1]]$body$text`). It returns a numeric vector `summaryStats` with three elements:

1. The number of exclamation marks in the subject.
2. The proportion of characters in the body of the email that are upper case. Exclude blanks, numbers, and punctuation when counting characters.
3. The proportion of characters in the subject that are upper case. Again, exclude blanks, numbers, and punctuation when counting characters.

Refer to the skeleton file for the names to give the input and output variables.

Function spamReplyTo

`spamReplyTo()` takes as input an e-mail address as a character string
(e.g., "`jane_doe@gmail.com`") and returns `TRUE` if it contains an underscore, and `FALSE` otherwise.
Refer to the skeleton file for the names to give the input and output variables.

Function spamSubject1

`spamSubject1()` takes takes as input the subject of an e-mail as a character string
(e.g., `Emails[[1]]$header["Subject"]`) and returns `TRUE` if the subject has punctuation or digits surrounded by characters. For example, the input "V?agra" and "Cia1is" should return `TRUE`, while "New!" should return `FALSE`.
Refer to the skeleton file for the names to give the input and output variables.

Function spamSubject2

`spamSubject2()` takes as input two arguments: first, the subject of an e-mail as a character string (e.g., `Emails[[1]]$header["Subject"]`) and second, the body text of an e-mail as a character vector (e.g., `Emails[[1]]$body$text`). It returns `TRUE` if the subject has the format "Re: something or other" but "something or other" does not appear in the body text, and `FALSE` otherwise. If the subject does not have the format "Re: something or other", return `NA`.
Refer to the skeleton file for the names to give the input and output variables.

Putting it all together: function spamFilter

Write a function `spamFilter()` to act as a simple spam filter. It should take as input four arguments: a list representing a single e-mail (e.g., `Emails[[1]]`); an upper threshold for the number of exclamation marks in the subject; upper thresholds for the proportions of uppercase characters in the body and subject of the e-mail.

An e-mail is marked as spam if any one of the conditions is met: the number of exclamation marks or the proportions of uppercase characters in the body or subject exceed their given thresholds, or any one of `spamSubject1()`, `spamSubject2()`, `spamReplyTo()` return `TRUE`.

`spamFilter()` should return `TRUE` to mark the e-mail as spam, and `FALSE` otherwise.

This function should call the other four functions `spamCounts()`, `spamSubject1()`, `spamSubject2()`, `spamReplyTo`.

Note: Not all e-mails will have a "Reply-To" field. "In Reply-To" is NOT the same as "Reply-To." As a result, you will not apply `spamReplyTo` to all e-mails.

Grading

I will source your code by running `source(''LastnameFirstname_HW6.R'')` This should produce four functions in my workspace, which I should be able to run on inputs with the format described above. Like the previous two assignments, points will be deducted if your code produces errors when run. You do NOT need to load the dataset `Emails.rda` at the top of the file; I'm just looking for these five functions.

I will run your functions on a diverse set of test cases. If your functions give me the expected output on these cases, you will get full points.