DS-GA 1015, Text as Data
Prof. Andrew Halterman
Assignment date: Monday, February 7, 2022

Homework 1

This homework must be turned in on NYU Brightspace by **Monday, February 21, 2022, at 4pm**. Late work may be turned in up to 2 days late and will incur penalties of the equivalent of one third of a letter grade per day late.

It must be your own work, and your own work only—you must not copy anyone's work, or allow anyone to copy yours. This extends to writing code. You may consult with others, but when you write up, you must do so alone.

Your homework submission must be a PDF or HTML report, containing all written answers and code, generated from `RMarkdown`. **Raw `.R` or `.Rmd` files will not be accepted.**

Please remember the following:

- Each question part should be clearly labeled in your submission.
- Do not include written answers as code comments. We will not grade code comments.
- The code used to obtain the answer for each question part should accompany the written answer.
- **Your code must be included in full, such that your understanding of the problems can be assessed.**

  Please consult "RMarkdown Basics" in *NYUBrightspace/Assignments* for help with RMarkdown. You can also use the "Sample RMarkdown HW" template to get started. Using this template is not required.

---

1. First we'll use the data from the U.S. inaugural addresses available in `quanteda`. Let's first look at the inaugural addresses given by Ronald Reagan in 1981 and 1985.

   You can find the speeches in the "data_corpus_inaugural" corpus of the `quanteda` package.

   To find the subset of the speeches given by Ronald Reagan you can use the following command:

   ```
   speeches <- corpus_subset(data_corpus_inaugural, President == "Reagan")
   ```

   (a) Write a function in R to calculate the type-token ratio (TTR) of each of these speeches and report your findings.

(b) Create a document feature matrix of the two speeches, with no pre-processing other than to remove the punctuation–be sure to check the options on “dfm” in R as appropriate. Calculate the cosine similarity between the two documents with `quanteda`. Report your findings.

2. Consider different preprocessing choices you could make. For each of the following parts of this question, you have three tasks: (i) make a theoretical argument for how it should affect the TTR of each document and the similarity of the two documents (ii) re-do question (1a) with the preprocessing option indicated and (iii) redo question (1b) with the preprocessing option indicated.

   To be clear, you must repeat tasks (i-iii) for each pre-processing option below. You should remove punctuation in each step.

   (a) Stemming the words?

   (b) Removing stop words?

   (c) Converting all words to lowercase?

   (d) Does tf-idf weighting make sense here? Calculate it and explain why or why not.

3. Take the following two headlines:

```
"Nasa Mars rover: Perseverance robot all set for big test."

"NASA Lands Its Perseverance Rover on Mars."
```

   (a) Write code in R to calculate the Euclidean distance between these sentences **you can use base R, but you can’t use distance functions from `quanteda` or similar. You are allowed to use `quanteda` to tokenize or create a DFM.** Use whatever pre-processing of the text you want, but justify your choice. Report your findings.

   (b) Write code in R to calculate the Manhattan distance between these sentences. Report your findings.

   (c) Write code in R to calculate the cosine similarity between these sentences. Report your findings.

   (d) Manually calculate the Levenshtein distance between *robot* and *rover*. Report your findings.

4. One of the earliest and most famous applications of statistical textual analysis was to determine the authorship of texts. You now get to do the same! You will be using the `stylest` package. To get the texts for this exercise you will need the `gutenbergr` package.

   (a) First you will need to get the data from Project Gutenberg using their `gutenbergr` package. Download the first four novels for each of the following authors:

   - `Poe, Edgar Allan` (*The Fall of the House of Usher*, *The Masque of the Red Death*, *The Raven*, and *Eureka: A Prose Poem*)
   - `Twain, Mark` (*The Adventures of Tom Sawyer*, *Adventures of Huckleberry Finn*, *A Connecticut Yankee in King Arthur's Court*, and *Tom Sawyer Abroad*),
   - `Shelley, Mary Wollstonecraft` (*Frankenstein*, *Proserpine and Midas*, *Mathilda*, and *The Last Man*),
   - `Doyle, Arthur Conan` (*The Return of Sherlock Holmes*, *The Poison Belt*, *The Lost World*, and *A Study in Scarlet*).

   From each of these novels extract a short excerpt (e.g. 500 lines of text).

   You can use the following code in order to prepare your data for the following exercises.

```
library(gutenbergr)
library(stylest)

## Prepare data

n<-gutenberg_authors[,]
# list of authors
#author_list <- c("Poe, Edgar Allan", "Twain, Mark", "Shelley, Mary
Wollstonecraft","Doyle, Arthur Conan")
#Here a list of the gutenberg_id associated with the books is given below
book_list<-c(932,1064,1065,32037,74,76,86,91,84,6447,15238,18247,108,126,
139,244)

#Using the following command you can check the information associated
with the first four novels for each author
#The gutenberg_id above were obtained with the following command
#meta <- gutenberg_works(author == "Doyle, Arthur Conan") %>% slice(1:4)

# Prepare data function

# @param author_name: author's name as it would appear in gutenberg
# @param num_texts: numeric specifying number of texts to select
# @param num_lines: num_lines specifying number of sentences to sample

  meta <- gutenberg_works(gutenberg_id == book_list)
  meta <- meta %>% mutate(author = unlist(str_split(author, ","))[1]
 %>% tolower(.))

prepare_dt <- function(book_list, num_lines, removePunct = TRUE){
 meta <- gutenberg_works(gutenberg_id == book_list)
 meta <- meta %>% mutate(author = unlist(str_split(author, ","))[1]
 %>% tolower(.))
 texts <- lapply(book_list, function(x) gutenberg_download(x,
 mirror="http://mirrors.xmission.com/gutenberg/") %>%
          #select(text) %>%
          sample_n(500, replace=TRUE) %>%
          unlist() %>%
          paste(., collapse = " ") %>%
          str_replace_all(., "^ +| +$|( ) +", "\\1"))

 # remove apostrophes
 texts <- lapply(texts, function(x) gsub("‘|’", "", x))
 if(removePunct) texts <- lapply(texts, function(x)
 gsub("[^[:alpha:]]", " ", x))

 # remove all non-alpha characters
 output <- tibble(title = meta$title, author = meta$author, text =
 unlist(texts, recursive = FALSE))
}

# run function
set.seed(1984L)
texts_dt <- lapply(book_list, prepare_dt, num_lines = 500, removePunct = TRUE)
texts_dt <- do.call(rbind, texts_dt)

print(texts_dt$title)
print(texts_dt$author)
```

(b) Next you will need to organize the data as required by the package. Create a table (i.e. a dataframe) with one column for the text excerpts and one column identifying the author of each excerpt (although not required to fit the model, also create a column for the title of the novel which the excerpt belongs to). Print the str() of your table.

(c) Now use the stylest_select_vocab function to select the terms you will include in your model. Note, this function allows you to include some pre-processing options. Justify any pre-processing choices you make. What percentile (of term frequency) has the best prediction rate? Also report the mean rate of incorrectly predicted speakers of held-out texts.

(d) Use your optimal percentile from above to subset the terms to be included in your model (this requires you use the stylest_terms function). Now go ahead and fit the model using `stylest_fit`. The output of this function includes information on the rate at which each author uses each term (the value is labeled `rate`). Report the top 5 terms (in terms of usage rate) for each author. Do these terms make sense?

(e) Choose any two authors, take the ratio of their rate vectors (make sure dimensions are in the same order) and arrange the resulting vector from largest to smallest values. What are the top 5 terms according to this ratio? How would you interpret this ordering?

(f) Load the mystery excerpt provided. According to your fitted model, who is the most likely author?

(g) Use `textstat_collocation` to inspect 2-grams with `min_count = 5` from your DFM of all 16 labeled novels. Report the 10 collocations with the largest λ value. Report the 10 collocations with the largest `count`. Discuss which set of n-grams is likely to be multi-word expressions.

5. For this question we will use the `sophistication` package discussed in the lab. The corpus for this exercise will be the UN data from quanteda (`data_corpus_ungd2017`).

   (a) Using the aforementioned corpus make snippets between 150 to 350 characters in length and clean the snippets (print the top 10).

   (b) Randomly sample 1000 snippets and use these to generate pairs for a minimum spanning tree. From these generate 10 gold pairs (print these—only each pair of text—in your HW). Without looking at the automated classification, read each pair and select whichever you think is "easiest" to read. Now compare your classification with those made by the package. What proportion of the ten gold pairs were you in agreement with the automated classification? Any reasons why you may have arrived at a different judgment?

6. Using Louisa May Alcott's "Little Women" (gutenberg_id = 514) and F. Scott Fitzgerald's "The Great Gatsby" (gutenberg_id = 64317), make a graph demonstrating Zipf's law. Include this graph and also discuss any pre-processing decisions you made.

7. Find the value of b that best fit the two works from the previous question to Heap's law, fixing $k = 44$. Report the value of b as well as any pre-processing decisions you made.

8. Both "Little Women" and "The Great Gatsby" broach the topic of *class*, but in very different ways. Choose a few Key Words in Context and discuss the different context in which those words are used by each author. Give a brief discussion of how the two works treat this theme differently.

9. Consider the bootstrapping of the texts we used to calculate the standard errors of the Flesch reading scores of Irish budget speeches in Recitation 4.

   (a) Obtain the UK Conservative Party's manifestos from `quanteda`. Generate estimates of the FRE scores of these manifestos over time (i.e. per year), using sentence-level bootstraps instead of the speech-level bootstraps used in Recitation 4. Report the bootstrapped estimates and standard errors in a table.

   You can use the following code to load and start using your data.

```
# load data
library(sophistication)
library(quanteda)
library(dplyr)
library(quanteda.corpora)
data("data_corpus_ukmanifestos")

manifestos <- corpus_subset(data_corpus_ukmanifestos, Party == "Con")

# tokenize by sentences
sent_tokens <- unlist(tokens(manifestos, what = "sentence",
include_docvars = TRUE))

# extract year metadata
yearnames <- list(unlist(names(sent_tokens)))
yearnames <- lapply(yearnames[[1]], function(x){strsplit(x, "_")[[1]][3]})
yearslist <- unlist(yearnames)

# create tibble
sentences_df <- tibble(text = sent_tokens, year = yearslist)

# filter out non-sentences (only sentences that end in sentence punctuation
sentences_df <- sentences_df[grepl( ("[\\.\\?\\!]$"), sentences_df$text), ]

# create quanteda corpus object
#sent_corp <- corpus(sentences_df$text)
#docvars(sent_corp, field = "Year") <- sentences_df$year
```

   (b) Compute the (non-bootstrapped) mean FRE score over time and report the results in a table. Discuss the contrast with the bootstrapped estimates from the previous section.

Hint: After you split up each speech into sentences, some of the sentences will begin with a number, or not be "sentences" at all (e.g. headings). Regular expressions are one way to remove this kind of text.