Text as Data HW 1

Vanessa (Ziwei) Xu and zx657

2/21/2022

```
# import libraries
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(pbapply)
library(stylest)
library(ggplot2)
library(stringr)
library(quanteda)
```

```
## Package version: 3.2.0
## Unicode version: 13.0
## ICU version: 69.1

## Parallel computing: 8 of 8 threads used.

## See https://quanteda.io for tutorials and examples.
```

```
library(gutenbergr)
library(quanteda.corpora)
library(quanteda.textstats)
library(quanteda.textplots)
```

Q1a

```
# your code to compute the answer
# ** Make sure your markdown document shows both CODE and OUTPUT **
speeches <- corpus_subset(data_corpus_inaugural, President == "Reagan") # punctuation included
tokenized_speeches <- tokens(speeches, remove_punct = TRUE) # punctuation excluded
# token <- tokens(text, remove_punct = TRUE)
TTR <- function(text) {
  TTR <- ntype(text) / ntoken(text)
  return(TTR)
}

cat("The type-token ratio (TTR) of the inaugural address given by Ronald Reagan in 1981 is", TTR(tokeni
```

```
## The type-token ratio (TTR) of the inaugural address given by Ronald Reagan in 1981 is 0.3680099
```

```
cat("The type-token ratio (TTR) of the inaugural address given by Ronald Reagan in 1985 is", TTR(tokeni
```

```
## The type-token ratio (TTR) of the inaugural address given by Ronald Reagan in 1985 is 0.3568643
```

TTR in 1981's speech given by Ronald Reagan is slightly higher than TTR in 1985, which means that a higher variety of unique words were used in 1981's speech compared to 1985's speech.

Q1b

```
# your code
RR_dfm <- dfm(tokenized_speeches, tolower = FALSE) # tolower is true default
textstat_simil(RR_dfm, method = "cosine")
```

```
## textstat_simil object; method = "cosine"
##             1981-Reagan 1985-Reagan
## 1981-Reagan       1.000       0.956
## 1985-Reagan       0.956       1.000
```

The cosine similarity between the two documents is 0.959, which indicates high similarity.

Q2a - Stemming the words

(i) Stemming the words should lower the TTR of each document because stemming reduces the number of unique words, which is the number of types. And stemming should increase the similarity of the two documents due to a decrease in variety of unique words used.

```
# Stemming the words
stemmed_speech <- tokens_wordstem(tokenized_speeches)  # stemmed
RR_dfm2a <- dfm_wordstem(RR_dfm) # matrix form
# ii: redo 1a
print("TTR of documents:")
```

```
## [1] "TTR of documents:"
```

```
TTR(stemmed_speech)
```

```
## 1981-Reagan 1985-Reagan
##   0.3322368   0.3178627
```

```
# iii: redo 1b
print("cosine similarity of two documents:")
```

```
## [1] "cosine similarity of two documents:"
```

```
textstat_simil(RR_dfm2a, method = "cosine")
```

```
## textstat_simil object; method = "cosine"
##             1981-Reagan 1985-Reagan
## 1981-Reagan       1.000       0.957
## 1985-Reagan       0.957       1.000
```

Q2b - Removing stop words

(i) Removing stop words should increase the TTR of each document a lot more than stemming should because removal of stop words drastically reduces the number of tokens compared to types, which does not decrease as much as tokens do. And removing stop words should reduce the similarity of the two documents due to a decrease in the number of repeated words that are very often used in both documents.

```
# Removing stop words
nostop_speech <- tokens_remove(tokenized_speeches, pattern = stopwords("english")) # stemmed
RR_dfm2b <- dfm(RR_dfm, tolower = FALSE, remove = stopwords("english")) # matrix form
```

```
## Warning: 'remove' is deprecated; use dfm_remove() instead
```

```
# ii: redo 1a
print("TTR of documents:")
```

```
## [1] "TTR of documents:"
```

```
TTR(nostop_speech)
```

```
## 1981-Reagan 1985-Reagan
##   0.6608544   0.6059908
```

```
# iii: redo 1b
print("cosine similarity of two documents:")
```

```
## [1] "cosine similarity of two documents:"
```

```
textstat_simil(RR_dfm2b, method = "cosine")
```

```
## textstat_simil object; method = "cosine"
##             1981-Reagan 1985-Reagan
## 1981-Reagan       1.000       0.668
## 1985-Reagan       0.668       1.000
```

Q2c - Converting all words to lowercase

(i) Converting all words to lowercase should slightly decrease the TTR of each document because it slightly decreases the variety of unique words used. And converting to lowercase should slightly increase the similarity of the two documents because all forms (upper or lower case) of the same words are grouped together and decreases the chance of having more types than they should.

```
# Converting all words to lowercase
lower_speech <- tokens_tolower(tokenized_speeches, keep_acronyms = FALSE) # stemmed
RR_dfm2c <- dfm_tolower(RR_dfm, keep_acronyms = FALSE) # matrix form
# ii: redo 1a
print("TTR of documents:")
```

```
## [1] "TTR of documents:"
```

```
TTR(lower_speech)
```

```
## 1981-Reagan 1985-Reagan
##   0.3466283   0.3377535
```

```
# iii: redo 1b
print("cosine similarity of two documents:")
```

```
## [1] "cosine similarity of two documents:"
```

```
textstat_simil(RR_dfm2c, method = "cosine")
```

```
## textstat_simil object; method = "cosine"
##             1981-Reagan 1985-Reagan
## 1981-Reagan       1.000       0.959
## 1985-Reagan       0.959       1.000
```

Q2d

```
# your code
dfm_tfidf(RR_dfm)
```

```
## Document-feature matrix of: 2 documents, 1,450 features (37.59% sparse) and 4 docvars.
##              features
## docs          Senator Hatfield      Mr Chief Justice President Vice Bush
```

```
##   1981-Reagan       0  0.30103 0.90309     0       0         0    0    0
##   1985-Reagan       0  0       0           0       0         0    0    0
##              features
## docs          Mondale   Baker
##   1981-Reagan 0.30103 0.30103
##   1985-Reagan 0       0
## [ reached max_nfeat ... 1,440 more features ]
```

tf-idf weighting does not make much sense here because most of the words occur in both documents. This way, idf is pretty low and this term is not as informative.

Q3a

```
# your code
t1 = "Nasa Mars rover: Perseverance robot all set for big test."
t2 = "NASA Lands Its Perseverance Rover on Mars."
tokenized_t <- tokens(c(t1,t2), remove_punct = TRUE)
tokenized_t <- tokens_tolower(tokenized_t, keep_acronyms = FALSE) # pre-processed tokens
t_dfm <- dfm(tokenized_t) # pre-processed dfm
sqrt(sum((t_dfm[1,] - t_dfm[2,])^2)) # Euclidean distance
```

```
## [1] 3
```

I removed punctuation and changed to lower case when pre-processing text because I believe NASA and Nasa mean the same thing and punctuation removal does not change the meaning of the text under this context. However, I don't see why stop words and stemming are required here. The Euclidean distance is 3.

Q3b

```
# your code
sum(abs(t_dfm[1,] - t_dfm[2,]))
```

```
## [1] 9
```

The Manhattan distance is 9.

Q3c

```
# your code
sum(t_dfm[1,] * t_dfm[2,]) / (sqrt(sum((t_dfm[1,])^2))*sqrt(sum((t_dfm[2,])^2)))
```

```
## [1] 0.4780914
```

The cosine similarity is 0.478.

Q3d

The minimum number of operations required to convert "robot" to "rover" is 3, which includes: 1. substitute "b" to "v" 2. substitute "o" to "e" 3. substitute "t" to "r"

Q4a & 4b

```
## 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 e
# 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)
```

```
## Warning in gutenberg_id == book_list: longer object length is not a multiple of
## shorter object length
```

```
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/guter

  # 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)
```

```
##  [1] "The Fall of the House of Usher"
```

```
##  [2] "The Masque of the Red Death"
##  [3] "The Raven"
##  [4] "Eureka: A Prose Poem"
##  [5] "The Adventures of Tom Sawyer"
##  [6] "Adventures of Huckleberry Finn"
##  [7] "A Connecticut Yankee in King Arthur's Court"
##  [8] "Tom Sawyer Abroad"
##  [9] "Frankenstein; Or, The Modern Prometheus"
## [10] "Proserpine and Midas"
## [11] "Mathilda"
## [12] "The Last Man"
## [13] "The Return of Sherlock Holmes"
## [14] "The Poison Belt"
## [15] "The Lost World"
## [16] "A Study in Scarlet"
```

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

```
##  [1] "poe"     "poe"     "poe"     "poe"     "twain"   "twain"   "twain"
##  [8] "twain"   "shelley" "shelley" "shelley" "shelley" "doyle"   "doyle"
## [15] "doyle"   "doyle"
```

Q4c

```
# your code
filter <- corpus::text_filter(drop_punct = TRUE, drop_number = TRUE, drop = stopwords("english"))
set.seed(1984L)
vocab_custom <- stylest_select_vocab(texts_dt$text, texts_dt$author, filter = filter, smooth = 1)
vocab_custom$cutoff_pct_best
```

```
## [1] 90
```

```
mean(vocab_custom$miss_pct)
```

```
## [1] 32.77778
```

90% (of term frequency) has the best prediction rate. The mean rate of incorrectly predicted speakers of held-out texts is 32.78%

Q4d

```
# your code
vocab_subset <- stylest_terms(texts_dt$text, texts_dt$author, vocab_custom$cutoff_pct_best, filter = fi
style_model <- stylest_fit(texts_dt$text, texts_dt$author, terms = vocab_subset, filter = filter)
authors <- unique(texts_dt$author)
term_usage <- style_model$rate
lapply(authors, function(x) head(term_usage[x,][order(-term_usage[x,])])) %>% setNames(authors)
```

```
## $poe
##       upon       door        one    chamber        now    nothing
## 0.01771408 0.01130497 0.01112694 0.01094891 0.01041481 0.00827844
##
## $twain
##          t          s        tom        got       said        see
## 0.03523261 0.01977572 0.01174420 0.01144113 0.01128959 0.01113805
##
## $shelley
##         one           s         now         may        love        life
## 0.009610478 0.008419799 0.007569315 0.007059024 0.006548733 0.006208539
##
## $doyle
##        said        upon         one           s          us         man
## 0.015829146 0.015326633 0.013316583 0.009798995 0.008626466 0.008626466
```

Some of the terms do make sense and some of them don't, such as "s" and "t".

Q4e

```
# your code
sort(term_usage["poe",]/term_usage["twain",], decreasing = TRUE)[1:5]
```

```
##     soul      thy    blood  chamber    fancy
## 88.11198 74.01406 50.51754 48.16788 45.81823
```

This means that Poe uses "soul" about 88 times more than Twain; Poe uses "thy" 74 times as Twain and so on.

Q4f

```
# your code
new_text <- readRDS("mystery_excerpt.rds")
pred <- stylest_predict(style_model, new_text)
pred$predicted
```

```
## [1] twain
## Levels: doyle poe shelley twain
```

```
pred$log_probs
```

```
## 1 x 4 Matrix of class "dgeMatrix"
##          doyle       poe   shelley         twain
## [1,] -15.56045 -48.35144 -34.53808 -1.746556e-07
```

The most likely author to this new excerpt is Twain.

Q4g

```
# your code
collocations <- textstat_collocations(texts_dt$text, min_count = 5)
print("10 collocations with the largest lambda value:")
```

```
## [1] "10 collocations with the largest lambda value:"
```

```
collocations[order(collocations$lambda, decreasing = TRUE),]$collocation[1:10]
```

```
##  [1] "edgar allan"         "denser perfumed"     "whispering vows"
##  [4] "syllable expressing" "candelabrum amid"    "unseen censer"
##  [7] "allan poe"           "arabesque figures"   "densely crowded"
## [10] "unsuited limbs"
```

```
print("10 collocations with the largest count:")
```

```
## [1] "10 collocations with the largest count:"
```

```
collocations[order(collocations$count, decreasing = TRUE),]$collocation[1:10]
```

```
##  [1] "of the"   "in the"   "and the"  "to the"   "it was"   "on the"
##  [7] "of a"     "from the" "to be"    "that the"
```

Q5a

```
# your code
data(data_corpus_ungd2017, package = "quanteda.corpora")

# Make snippets of 1 sentence each, then clean them
snippetData <- snippets_make(data_corpus_ungd2017, nsentence = 1, minchar = 150, maxchar = 350)
```

```
## Error in snippets_make(data_corpus_ungd2017, nsentence = 1, minchar = 150, : could not find function
```

```
snippetData <- snippets_clean(snippetData)
```

```
## Error in snippets_clean(snippetData): could not find function "snippets_clean"
```

```
head(snippetData)
```

```
## Error in head(snippetData): object 'snippetData' not found
```

Q5b

```
# your code
testData <- sample_n(snippetData, 1000)
```

```
## Error in sample_n(snippetData, 1000): object 'snippetData' not found
```

```
snippetPairsMST <- pairs_regular_make(testData)
```

```
## Error in pairs_regular_make(testData): could not find function "pairs_regular_make"
```

```
pairs_regular_browse(snippetPairsMST)
```

```
## Error in pairs_regular_browse(snippetPairsMST): could not find function "pairs_regular_browse"
```

```
gold_questions <- pairs_gold_make(snippetPairsAll, n.pairs = 10)
```

```
## Error in pairs_gold_make(snippetPairsAll, n.pairs = 10): could not find function "pairs_gold_make"
```

Due to a failure to install sophistication package from my Mac with M1 chip, I used another computer to do this homework and run the code. However, I did the rest of my homework on my Mac so output for this question cannot be included.

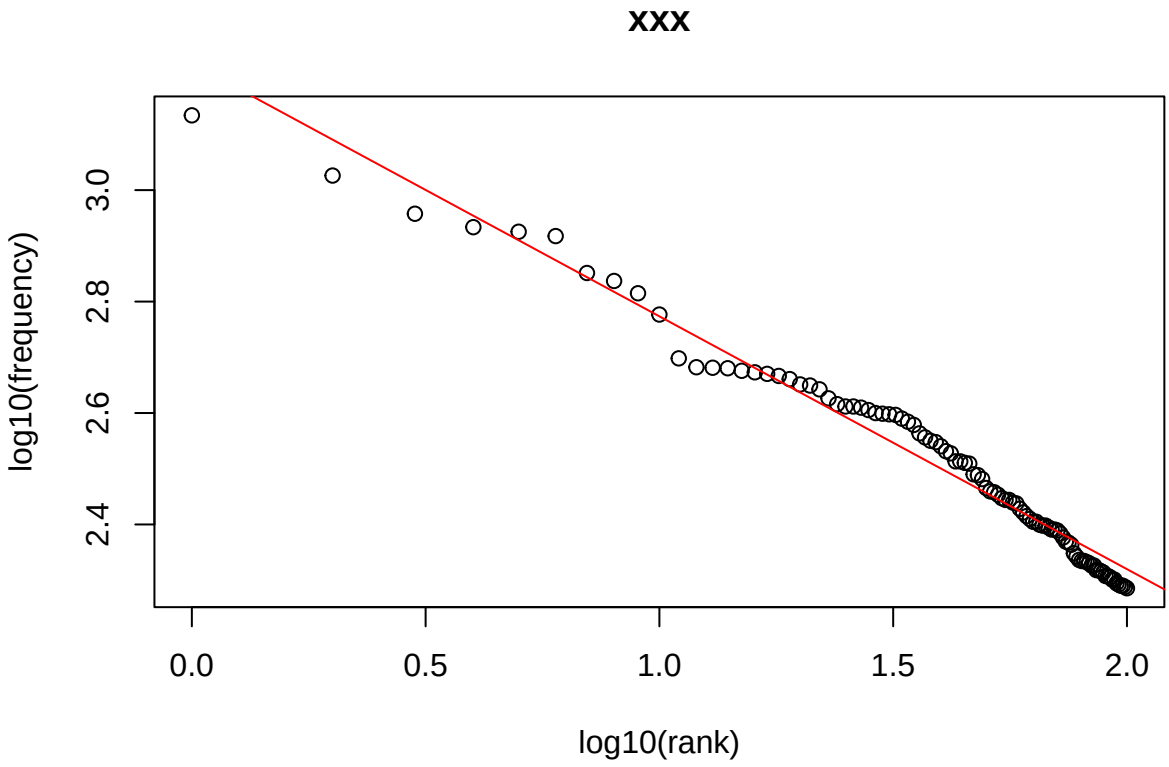
Q6

```
# your code

text_gg <- gutenberg_download(64317, mirror="http://mirrors.xmission.com/gutenberg/") %>% select(text) %
# text_gg <- tokens(text_gg, remove_punct = TRUE)
text_lw <- gutenberg_download(514, mirror="http://mirrors.xmission.com/gutenberg/") %>% select(text) %>%
# text_lw <- tokens(text_lw, remove_punct = TRUE)
gglw_dfm <- dfm(tokens(c(text_gg, text_lw)), remove_punct = TRUE, tolower = TRUE, remove = stopwords("en
```

```
## Warning: '...' should not be used for tokens() arguments; use 'tokens()' first.
```

```
## Warning: 'remove' is deprecated; use dfm_remove() instead
```

```
gglw_dfm <- dfm_wordstem(gglw_dfm) # matrix form


# Regression to check if slope is approx -1.0
regression <- lm(log10(topfeatures(gglw_dfm, 100)) ~ log10(1:100))
plot(log10(1:100), log10(topfeatures(gglw_dfm, 100)),
     xlab = "log10(rank)", ylab = "log10(frequency)", main = "xxx") + abline(regression, col = "red")
```




```
## integer(0)
```

I decided to download both texts with removal of punctuation and stop words, and I also applied stemming and changing to lower case only. This makes later calculations a lot easier and more efficient.

Q7

```
# your code
text_gglw <- tokens(c(text_gg,text_lw), remove_punct = TRUE)
num_tokens <- sum(lengths(text_gglw))
M <- nfeat(gglw_dfm)  # number of types
k <- 44
b <- log(M)/log(num_tokens) - log(k)/log(num_tokens)
b
```

```
## [1] 0.4275952
```

b is approximately 0.428 to 3 decimal places and I've removed punctuation as too much pre-processing results in weird tokens such as "f".

Q8

```
# your code
text_lw <- tokens(text_lw, remove_punct = TRUE)
text_gg <- tokens(text_gg, remove_punct = TRUE)
kwic_classlw <- kwic(text_lw, pattern = "class*", valuetype = "glob", window = 3)
kwic_classgg <- kwic(text_gg, pattern = "class*", valuetype = "glob", window = 3)
kwic_wealthlw <- kwic(text_lw, pattern = "wealth*", valuetype = "glob", window = 3)
kwic_wealthgg <- kwic(text_gg, pattern = "wealth*", valuetype = "glob", window = 3)
kwic_powerlw <- kwic(text_lw, pattern = "power*", valuetype = "glob", window = 3)
kwic_powergg <- kwic(text_gg, pattern = "power*", valuetype = "glob", window = 3)
kwic_elitlw <- kwic(text_lw, pattern = "elit*", valuetype = "glob", window = 3)
kwic_elitgg <- kwic(text_gg, pattern = "elit*", valuetype = "glob", window = 3)
kwic_classlw
```

```
## Keyword-in-context with 10 matches.
##   [text1, 34476]        attracts a certain |   class   | of people and
##   [text1, 40540] excellent being patriotic | classical | comical or dramatic
##   [text1, 89529]                 men of my |   class   | were heroes in
##   [text1, 96306]          baby Our drawing |   class   | breaks up next
##   [text1, 96593]           fourteen in the |   class   | but I dare
##   [text1, 99859]          belonged to that |   class   | of light literature
##  [text1, 144502]  statuesque attitudes and |  classic  | draperies But dear
##  [text1, 144737]               only had a | classical | nose and mouth
##  [text1, 174525]       and there's another |   class   | who can't ask
##  [text1, 175896]           and dismiss the |   class   | in metaphysics There
```

```
kwic_classgg
```

```
## Keyword-in-context with 1 match.
##  [text1, 33903] president of your | class | at Yale Tom
```

```
kwic_wealthlw
```

```
## Keyword-in-context with 7 matches.
##    [text1, 7612] she bequeaths untold | wealth | to the young
##   [text1, 38545]   name and boundless | wealth | in return for
##   [text1, 68305]     more than talent | wealth | or beauty And
##  [text1, 100746]  groceries and gowns | Wealth | is certainly a
##  [text1, 111613]          or women of | wealth | and position we
##  [text1, 170136]       but the better | wealth | of love confidence
##  [text1, 180199]         on the plain | wealth | on the poor
```

```
kwic_wealthgg
```

```
## Keyword-in-context with 4 matches.
##   [text1, 1610] family were enormously | wealthy | even in college
##   [text1, 1666]     own generation was | wealthy | enough to do
##  [text1, 17250]            son of some | wealthy | people in the
##  [text1, 39524]       and mystery that | wealth  | imprisons and preserves
```

```
kwic_powerlw
```

```
## Keyword-in-context with 35 matches.
##    [text1, 2298] gifted with dramatic |     power     | but was chosen
##    [text1, 6790]          well Or its |     power     | will vanish soon
##    [text1, 7369]    song of exquisite |     power     | and melody This
##   [text1, 26465]         charm of all |     power     | is modesty So
##   [text1, 30802]      depend on human |     power     | and wisdom His
##   [text1, 42417]           in her own |    powers     | and a friendly
##   [text1, 42641]     expressed of her |    powers     | Get what you
##   [text1, 44670]           a sense of |     power     | and independence better
##   [text1, 70422]    dreadful sense of | powerlessness | which comes to
##   [text1, 85487]          the love of |     power     | which sleeps in
##   [text1, 85666]          and her own |     power     | He was grave
##   [text1, 88266]     it possessed the |     power     | to lead him
##   [text1, 89487]  or the irresistible |     power     | of persuasion which
##   [text1, 90864]         will but the |     power     | to cook wholesome
##   [text1, 91574]        sample of its |    powers     | that made them
##   [text1, 94528]      and feeling her |     power     | used it as
##  [text1, 100693]       feel herself a |     power     | in the house
##  [text1, 108951]            it off My |    powers     | are great as
##  [text1, 109384]     situated she was |   powerless   | to check Jo
##  [text1, 130399] that money conferred |     power     | money and power
##  [text1, 130402]      power money and |     power     | therefore she resolved
##  [text1, 130654]    an influence more |   powerful    | over many than
##  [text1, 137938]   all his persuasive |    powers     | to bear as
##  [text1, 140414]          much of its |     power     | for Beth seemed
##  [text1, 145410]  delightful sense of |     power     | which comes when
##  [text1, 153791]         she knew her |     power     | and enjoyed the
##  [text1, 157874]   patience Which has |     power     | to sustain A
##  [text1, 160049]       absorb all his |    powers     | for years but
##  [text1, 164372]   with gratitude and |     power     | Other helps had
##  [text1, 164756]      argument in her |     power     | and the sisterly
##  [text1, 167336]      almost the only |     power     | that can part
##  [text1, 172334]       with a sweeter |     power     | than any other
##  [text1, 174257]          to have the |     power     | of giving freely
##  [text1, 181262]   by love's immortal |     power     | Nearest and dearest
##  [text1, 182295]          will take a |     power     | of money to
```

```
kwic_powergg
```

```
## Keyword-in-context with 6 matches.
##   [text1, 1572]       of the most | powerful | ends that ever
##   [text1, 1939] hide the enormous |  power   | of that body
##  [text1, 27895] expended your own |  powers  | of adjustment They
##  [text1, 36630]   movement of his | powerful | arms pushed his
##  [text1, 44650]      lot of brain |  power   | here He touched
##  [text1, 47104]    beyond my eyes |  power   | of correction So
```

```
kwic_elitlw
```

```
## Keyword-in-context with 0 matches.
```

```
kwic_elitgg
```

```
## Keyword-in-context with 0 matches.
```

A few keywords that I thought of was “class”, “wealth”, “power”, and “elite”. A difference of word choice can be told from the kwic command results. From my observations, the description of “power” is a lot more subtle in Little Women and a lot stronger and straightforward in The Great Gatsbby. Little Women also emphasizes on the power of women.

Q9a

```
# load data
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

iters <- 10
boot_flesch <- function(year_data){
  N <- nrow(year_data)
  bootstrap_sample <- corpus_sample(corpus(c(year_data$text)), size = N, replace = TRUE)
  bootstrap_sample<- as.data.frame(as.matrix(bootstrap_sample))
  readability_results <- textstat_readability(bootstrap_sample$V1, measure = "Flesch")
  return(mean(readability_results$Flesch))
}

boot_flesch_by_year <- pblapply(unique(yearslist), function(x){
  sub_data <- sentences_df %>% filter(year == x)
  output_flesch <- lapply(1:iters, function(i) boot_flesch(sub_data))
  return(unlist(output_flesch))
})
names(boot_flesch_by_year) <- unique(yearslist)
View(boot_flesch_by_year)

# compute mean and std.errors
year_means <- lapply(boot_flesch_by_year, mean) %>% unname() %>% unlist()
year_ses <- lapply(boot_flesch_by_year, sd) %>% unname() %>% unlist() # bootstrap standard error = samp
year_means
```

```
##  [1] 49.35105 44.45550 52.91909 48.74453 48.02025 45.14278 45.61723 46.20989
##  [9] 42.42979 47.62509 47.40783 46.54466 46.28899 50.20381 47.99792 49.42894
```

```
year_ses
```

```
##  [1] 1.4509173 0.9521737 1.5475880 0.8505582 1.1989218 1.1204508 1.7604011
##  [8] 0.9605794 0.8973020 0.3440682 1.1013353 0.6692886 0.7733140 0.5766891
## [15] 0.6716944 1.1269684
```

Q9b

```
# your code
flesch_point <- sentences_df$text %>% textstat_readability(measure = "Flesch") %>%
  group_by(sentences_df$year) %>%
  summarise(mean_flesch = mean(Flesch)) %>%
  setNames(c("year", "mean")) %>% arrange(year)

flesch_point
```

```
## # A tibble: 16 x 2
##    year   mean
##    <chr> <dbl>
##  1 1945   49.0
##  2 1950   43.9
##  3 1951   52.0
##  4 1955   49.1
##  5 1959   48.4
##  6 1964   45.8
##  7 1966   46.3
##  8 1970   46.1
##  9 1974   42.3
## 10 1979   47.5
## 11 1983   47.7
## 12 1987   46.7
## 13 1992   46.4
## 14 1997   49.9
## 15 2001   48.1
## 16 2005   49.5
```

There are slight deviations between bootstrapped and unbootstrapped estimates of FRE score over time.