~~~
---
title: 'Homework 2: text processing'
author: "Zhuo Zhang"
date: "2024/02/07"
output: pdf_document
---

```{r}
## Load packages
pkgTest <- function(pkg){
  new.pkg <- pkg[!(pkg %in% installed.packages()[,  "Package"])]
  if (length(new.pkg))
    install.packages(new.pkg,  dependencies = TRUE)
  sapply(pkg,  require,  character.only = TRUE)
}

lapply(c("tidyverse",
         "guardianapi", # for working with the Guardian's API
         "quanteda", # for QTA
         "quanteda.textstats", # more Quanteda!
         "quanteda.textplots", # even more Quanteda!
         "readtext", # for reading in text data
         "stringi", # for working with character strings
         "textstem" # an alternative method for lemmatizing
       ), pkgTest)

```



```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

## Overview

The second homework assignment covers text processing and associated skills, including textual
statistics and dictionary methods.

## Analysis of tweets during a political crisis

We will start with a dataset that contains almost 900 tweets that were published by four central
figures in American politics around the time of the onset of an impeachment inquiry: Pres. Donald
Trump, Rudy Giuliani, Speaker of the House Rep. Nancy Pelosi, and Chair of the House Intelligence
Committee Rep. Adam Schiff.

First, read in the spreadsheet of tweets into R and then use the `str` and `head` functions to
describe the variables and contents of the dataset.  Be sure that the file is in the same folder as
this homework RMarkdown file.

```{r}
data <- read.csv("us_tweets.csv",
                 stringsAsFactors=FALSE,
                 encoding = "utf-8")
```

Print the number of tweets that are in this dataset.

```{r}
print(nrow(data))
```

Create a new dataframe that only includes original tweets (i.e. remove retweets).

```{r}
#ndata <- data[which(),] # Fill in the gaps
ndata <- data[!grepl("^RT", data$text), ]

```
~~~

Create a smaller dataframe that only includes tweets by Donald Trump.

```{r}
#trump <- ndata[which(), ]# Fill in the gaps
trump <- ndata[ndata$screen_name == "realDonaldTrump", ]

```

How many tweets include an exclamation mark? In how many tweets did Trump mention words related to "winning", "employment", "immigration" or "hoax"? Use regular expressions when searching the tweets; you may also wish to wrap your search term between word anchor boundaries (`\\b`). For instance, for the term health: `"\\bhealth\\b"`

```{r}
#sum(grepl("", trump$text, ignore.case = TRUE)) # Adapt this code as needed
exclamation_count <- sum(grepl("!", trump$text))
winning_employment_immigration_hoax_count <-
sum(grepl("\\bwinning\\b|\\bemployment\\b|\\bimmigration\\b|\\bhoax\\b", trump$text, ignore.case =
TRUE))

cat("Number of tweets with exclamation mark:", exclamation_count, "\n")
cat("Number of tweets mentioning winning, employment, immigration, or hoax:",
winning_employment_immigration_hoax_count, "\n")

```

Corpus creation

Create a `corpus` and a `dfm` object with processed text (including collocations) using the dataframe generated in Question 1.1.

```{r}
library(quanteda)
# create corpus
corpus <- Corpus(VectorSource(ndata$text))
#corpus <- Corpus(ndata) # select the correct column here

# create tokens object
toks <- tokens(corpus,
               include_docvars = TRUE) %>%
  tokens_tolower() %>%
  tokens_remove(stopwords('english'), padding = TRUE) %>%
  tokens_remove('[\\p{P}\\p{S}]', valuetype = 'regex', padding = TRUE) %>%
  tokens_remove('amp', valuetype = 'fixed', padding = TRUE)

# detect collocations and merge with tokens object
col <- textstat_collocations(toks, method = "count", size = 2, min_count = 5, smoothing = 0.5)

toks <- tokens_compound(toks, pattern = col[col$z > 1.96, "feature"])
toks <- tokens_remove(tokens(toks), "")

# create dfm from tokens object
docfm <- dfm(toks,
             remove_numbers = TRUE,
             remove_punct = TRUE,
             remove_symbols = TRUE,
             remove_hyphens = TRUE,
             remove_separators = TRUE,
             remove_url = TRUE)

docfm <- dfm_select(docfm, pattern = stopwords("en"), selection = "remove")


```

Textual statistics

With the generated `dfm` object perform the following tasks:

- Create a frequency plot of the top 30 tokens for Trump and Pelosi.

```{r}
library(ggplot2)

# Trump plot
dfm_trump <- dfm_subset(docfm, screen_name == "realDonaldTrump")
dfm_freq_trump <- textstat_frequency(dfm_trump, n = 30)

ggplot(dfm_freq_trump, aes(x = reorder(feature, -frequency), y = frequency)) +
  ggtitle("Top 30 Tokens for Trump") +
  geom_col() +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

# Pelosi plot
dfm_pelosi <- dfm_subset(docfm, screen_name == "SpeakerPelosi")
dfm_freq_pelosi <- textstat_frequency(dfm_pelosi, n = 30)

ggplot(dfm_freq_pelosi, aes(x = reorder(feature, -frequency), y = frequency)) +
  ggtitle("Top 30 Tokens for Pelosi") +
  geom_col() +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))


# Pelosi plot
# your code  here
```

- Determine the "key" terms that Trump and Pelosi are more likely to tweet. Plot your results.

```{r}
# Determine the "key" terms that Trump and Pelosi are more likely to tweet. Plot your results.

# Subset the document-feature matrix (dfm) for comparison
dfm_comparison <- dfm_subset(docfm, c("realDonaldTrump", "NancyPelosi"))

# Group the dfm for keyness analysis
set.seed(1234)
dfm_keyness <- dfm_group(dfm_comparison, groups = "screen_name")

# Compute keyness statistics
keyness_stat <- textstat_keyness(dfm_keyness, target = "realDonaldTrump")

# Plot the keyness results
textplot_keyness(keyness_stat, labelsize = 3)

# Trump keyness
head(keyness_stat, 30)

# Pelosi keyness
keyness_stat_pelosi <- textstat_keyness(dfm_keyness, target = "NancyPelosi")
head(keyness_stat_pelosi, 30)

```

```{r}
#Trump keyness
head(keyness_stat, 30)
```

```{r}
# Pelosi keyness
# your code here
```

- Perform a keyword in context analysis using your `corpus` object for some of the most distinct keywords from both Trump and Pelosi. *Hint: remember to use the `phrase` function in the `pattern` argument of `kwic`*

```{r}
# Trump
# Perform a keyword in context analysis using your corpus object for some of the most distinct
keywords from both Trump and Pelosi.

# Trump
trump_corp <- corpus_subset(corpus, screen_name %in% "realDonaldTrump")
trump_kwic1 <- kwic(trump_corp, pattern = phrase("witch hunt"), window = 5, case_insensitive = TRUE)
trump_kwic2 <- kwic(trump_corp, pattern = phrase("great"), window = 5, case_insensitive = TRUE)
trump_kwic3 <- kwic(trump_corp, pattern = phrase("ukraine"), window = 5, case_insensitive = TRUE)
head(trump_kwic1)
# Pelosi
pelosi_corp <- corpus_subset(corpus, screen_name %in% "NancyPelosi")
pelosi_kwic1 <- kwic(pelosi_corp, pattern = phrase("keyword1"), window = 5, case_insensitive = TRUE)
pelosi_kwic2 <- kwic(pelosi_corp, pattern = phrase("keyword2"), window = 5, case_insensitive = TRUE)
pelosi_kwic3 <- kwic(pelosi_corp, pattern = phrase("keyword3"), window = 5, case_insensitive = TRUE)
head(pelosi_kwic1)

# Your code here
```

Dictionary methods

Conduct a sentiment analysis of Trump's tweets using the Lexicon Sentiment Dictionary. Plot net sentiment over the entire sample period and interpret the results.

```{r}
# Conduct a sentiment analysis of Trump's tweets using the Lexicon Sentiment Dictionary. Plot net
sentiment over the entire sample period and interpret the results.

# Assuming dfm_trump is the dfm object for Trump's tweets

sent_dfm <- dfm(trump_corp, dictionary = data_dictionary_LSD2015[1:2])

# Calculate proportions of positive and negative sentiment
docvars(dfm_trump, "prop_negative") <- as.numeric(sent_dfm[, "negative"] / ntoken(trump_corp))
docvars(dfm_trump, "prop_positive") <- as.numeric(sent_dfm[, "positive"] / ntoken(trump_corp))

# Calculate net sentiment
docvars(dfm_trump, "net_sentiment") <- docvars(dfm_trump, "prop_positive") - docvars(dfm_trump,
"prop_negative")

# Parse the date using lubridate package
docvars(dfm_trump, "date2") <- as.Date(docvars(dfm_trump, "date"))

# Plot net sentiment over time
sent_plot <- ggplot(dfm_trump, aes(x = date2, y = net_sentiment)) +
  geom_smooth() +
  theme_minimal()

sent_plot

```

What can we learn about the political communication surrounding the political crisis based on the above results?