ASSIGNMENT 2

```
setwd("E:\\Ravin's Files\\DESKTOP\\UT Dallas\\Classes\\School Year 2022-2023\\Fall 2022\\Methods of Data Collection and Production")

# Data Method: Text mining

# File: textmining1.R

# Theme: Download text data from web and create wordcloud


# Install the easypackages package

install.packages("easypackages")

library(easypackages)


# Load multiple packages using easypackage function "packages"

# Download text data from website

WCLocation <-URLencode("http://www.historyplace.com/speeches/churchill-hour.htm")


# use htmlTreeParse function to read and parse paragraphs

doc.html<- htmlTreeParse(WCLocation, useInternal=TRUE)

WC <- unlist(xpathApply(doc.html, '//p', xmlValue))

WC

head(WC, 3)


# Vectorize wc

words.vec <- VectorSource(WC)


# Check the class of words.vec

class(words.vec)


# Create Corpus object for preprocessing

words.corpus <- Corpus(words.vec)
```

```
inspect(words.corpus)

# Turn all words to lower case
words.corpus <- tm_map(words.corpus, content_transformer(tolower))

# Remove punctuations, numbers
words.corpus <- tm_map(words.corpus, removePunctuation)
words.corpus <- tm_map(words.corpus, removeNumbers)

# How about stopwords, then uniform bag of words created

words.corpus <- tm_map(words.corpus, removeWords, stopwords("english"))

# Create Term Document Matrix

tdm <- TermDocumentMatrix(words.corpus)
inspect(tdm)

m <- as.matrix(tdm)
wordCounts <- rowSums(m)
wordCounts <- sort(wordCounts, decreasing=TRUE)
head(wordCounts)

# Create Wordcloud
cloudFrame<-data.frame(word=names(wordCounts),freq=wordCounts)

set.seed(1234)
wordcloud(cloudFrame$word,cloudFrame$freq)
```

```
wordcloud(names(wordCounts),wordCounts, min.freq=3,random.order=FALSE,
max.words=500,scale=c(3,.5), rot.per=0.35,colors=brewer.pal(8,"Dark2"))
```

Run the program on Winston Churchill's Finest Hour speech?

http://www.historyplace.com/speeches/churchill-hour.htm

ASSIGNMENT 3

```
# Data Methods: Social media (Twitter) data

# Sample program for using rtweet, sentiment analysis

# Use vignette("auth", package = "rtweet") for authentication

# Documentation: vignette("intro", package = "rtweet")

# GitHub: https://github.com/mkearney/rtweet

# [Bob Rudis 21 Recipes for Mining Twitter Data with rtweet](https://rud.is/books/21-recipes/)


install.packages(c("rtweet", "ggplot2"))

library(rtweet)


# Set up authentication using own Twitter account

# will save credentials to local drive as default.rds

auth_setup_default()


## search for 1000 tweets of "Joe Biden" in English

jbt <- rtweet::search_tweets(q = "JoeBiden", n = 500, lang = "en", retryonratelimit = TRUE)

## search for 1000 tweets of "Black Lives Matter" in English

blm <- rtweet::search_tweets(q = "BlackLivesMatter", n = 500, lang = "en", retryonratelimit = TRUE)

## search for 1000 tweets of "PLA Taiwam" in English

PLAt <- rtweet::search_tweets(q = "PLA Taiwan", n = 500, lang = "en", retryonratelimit = TRUE)
```

```
## search for 1000 tweets of "COVID vaccines" in English
COVID <- rtweet::search_tweets(q = "COVID vaccines", n = 500, lang = "en", retryonratelimit = TRUE)

# Get Joe Biden's most recent tweets
JBT = get_timelines("JoeBiden", n = 500)

## preview users data
users_data(JBT)

## Boolean search for large quantity of tweets (which could take a while)
jbt2 <- rtweet::search_tweets("JoeBiden", n = 5000,
                              retryonratelimit = TRUE)

## plot time series of tweets frequency
library(ggplot2)
ts_plot(jbt2, by = "mins") + theme_bw()
```

ASSIGNMENT 4

```
# Sample program for using quanteda for text modeling and analysis
# Use vignette("auth", package = "rtweet") for authentication
# Documentation: vignette("quickstart", package = "quanteda")
# Website: https://quanteda.io/
install.packages(c("quanteda","quanteda.textmodels","quanteda.textplots","quanteda.textstats"))

library(quanteda)
library(quanteda.textmodels)
library(quanteda.textplots)
library(quanteda.textstats)
library(readr)
```

```
library(ggplot2)

# Twitter data about President Biden and Xi summit in Novemeber 2021

# Do some background search/study on the event

#

summit <-
read_csv("https://raw.githubusercontent.com/datageneration/datamethods/master/textanalytics/summit_11162021.csv")

View(summit)


sum_twt = summit$text

toks = tokens(sum_twt)

sumtwtdfm <- dfm(toks)

class(sumtwtdfm)

# Latent Semantic Analysis

sum_lsa <- textmodel_lsa(sumtwtdfm)

summary(sum_lsa)

class(sum_lsa)

library(tidyverse)

tweet_dfm <- tokens(sum_twt, remove_punct = TRUE) %>%

  dfm()

head(tweet_dfm)

tag_dfm <- dfm_select(tweet_dfm, pattern = "#*")

toptag <- names(topfeatures(tag_dfm, 50))

head(toptag, 10)



tag_fcm <- fcm(tag_dfm)

head(tag_fcm)

topgat_fcm <- fcm_select(tag_fcm, pattern = toptag)
```

```
textplot_network(topgat_fcm, min_freq = 50, edge_alpha = 0.8, edge_size = 5)

user_dfm <- dfm_select(tweet_dfm, pattern = "@*")

topuser <- names(topfeatures(user_dfm, 50))

head(topuser, 20)

user_fcm <- fcm(user_dfm)

head(user_fcm, 20)


user_fcm <- fcm_select(user_fcm, pattern = topuser)

textplot_network(user_fcm, min_freq = 20, edge_color = "firebrick", edge_alpha = 0.8, edge_size = 5)



# Wordcloud

# based on US presidential inaugural address texts, and metadata (for the corpus), from 1789 to present.

dfm_inaug <- corpus_subset(data_corpus_inaugural, Year <= 1826) %>%

  dfm(remove = stopwords('english'), remove_punct = TRUE) %>%

  dfm_trim(min_termfreq = 10, verbose = FALSE)



set.seed(100)

textplot_wordcloud(dfm_inaug)

inaug_speech = data_corpus_inaugural


corpus_subset(data_corpus_inaugural,

            President %in% c("Trump", "Obama", "Bush")) %>%

  tokens(remove_punct = TRUE) %>%

  tokens_remove(stopwords("english")) %>%

  dfm() %>%

  dfm_group(groups = President) %>%
```

```
  dfm_trim(min_termfreq = 5, verbose = FALSE) %>%
  textplot_wordcloud(comparison = TRUE)


textplot_wordcloud(dfm_inaug, min_count = 10,
                   color = c('red', 'pink', 'green', 'purple', 'orange', 'blue'))


data_corpus_inaugural_subset <-
  corpus_subset(data_corpus_inaugural, Year > 1949)
kwic(tokens(data_corpus_inaugural_subset), pattern = "american") %>%
  textplot_xray()


textplot_xray(
  kwic(data_corpus_inaugural_subset, pattern = "american"),
  kwic(data_corpus_inaugural_subset, pattern = "people"),
  kwic(data_corpus_inaugural_subset, pattern = "communist")
)


theme_set(theme_bw())
g <- textplot_xray(
  kwic(toks, pattern = "american"),
  kwic(toks, pattern = "people"),
  kwic(toks, pattern = "communist")
)
g + aes(color = keyword) +
  scale_color_manual(values = c("blue", "red", "green")) +
```

```
  theme(legend.position = "none")



features_dfm_inaug <- textstat_frequency(dfm_inaug, n = 100)

# Sort by reverse frequency order
features_dfm_inaug$feature <- with(features_dfm_inaug, reorder(feature, -frequency))

ggplot(features_dfm_inaug, aes(x = feature, y = frequency)) +
  geom_point() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))



# Get frequency grouped by president
freq_grouped <- textstat_frequency(dfm(tokens(data_corpus_inaugural_subset)),
                                   groups = data_corpus_inaugural_subset$President)

# Filter the term "american"
freq_american <- subset(freq_grouped, freq_grouped$feature %in% "american")

ggplot(freq_american, aes(x = group, y = frequency)) +
  geom_point() +
  scale_y_continuous(limits = c(0, 14), breaks = c(seq(0, 14, 2))) +
  xlab(NULL) +
  ylab("Frequency") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

```
dfm_rel_freq <- dfm_weight(dfm(tokens(data_corpus_inaugural_subset)), scheme = "prop") * 100
head(dfm_rel_freq)

rel_freq <- textstat_frequency(dfm_rel_freq, groups = dfm_rel_freq$President)

# Filter the term "american"
rel_freq_american <- subset(rel_freq, feature %in% "american")

ggplot(rel_freq_american, aes(x = group, y = frequency)) +
  geom_point() +
  scale_y_continuous(limits = c(0, 0.7), breaks = c(seq(0, 0.7, 0.1))) +
  xlab(NULL) +
  ylab("Relative frequency") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))



dfm_weight_pres <- data_corpus_inaugural %>%
  corpus_subset(Year > 2000) %>%
  tokens(remove_punct = TRUE) %>%
  tokens_remove(stopwords("english")) %>%
  dfm() %>%
  dfm_weight(scheme = "prop")

# Calculate relative frequency by president
freq_weight <- textstat_frequency(dfm_weight_pres, n = 10,
                                  groups = dfm_weight_pres$President)

ggplot(data = freq_weight, aes(x = nrow(freq_weight):1, y = frequency)) +
  geom_point() +
```

```
  facet_wrap(~ group, scales = "free") +
  coord_flip() +
  scale_x_continuous(breaks = nrow(freq_weight):1,
                     labels = freq_weight$feature) +
  labs(x = NULL, y = "Relative frequency")



# Only select speeches by Obama and Trump
pres_corpus <- corpus_subset(data_corpus_inaugural,
                             President %in% c("Obama", "Trump"))

# Create a dfm grouped by president
pres_dfm <- tokens(pres_corpus, remove_punct = TRUE) %>%
  tokens_remove(stopwords("english")) %>%
  tokens_group(groups = President) %>%
  dfm()

# Calculate keyness and determine Trump as target group
result_keyness <- textstat_keyness(pres_dfm, target = "Trump")

# Plot estimated word keyness
textplot_keyness(result_keyness)

# Plot without the reference text (in this case Obama)
textplot_keyness(result_keyness, show_reference = FALSE)
```

```
# Transform corpus to dfm
data(data_corpus_irishbudget2010, package = "quanteda.textmodels")
ie_dfm <- dfm(tokens(data_corpus_irishbudget2010))

# Set reference scores
refscores <- c(rep(NA, 4), 1, -1, rep(NA, 8))

# Predict Wordscores model
ws <- textmodel_wordscores(ie_dfm, y = refscores, smooth = 1)

# Plot estimated word positions (highlight words and print them in red)
textplot_scale1d(ws,
                 highlighted = c("minister", "have", "our", "budget"),
                 highlighted_color = "red")


# Get predictions
pred <- predict(ws, se.fit = TRUE)

# Plot estimated document positions and group by "party" variable
textplot_scale1d(pred, margin = "documents",
                 groups = docvars(data_corpus_irishbudget2010, "party"))

# Plot estimated document positions using the LBG transformation and group by "party" variable

pred_lbg <- predict(ws, se.fit = TRUE, rescaling = "lbg")

textplot_scale1d(pred_lbg, margin = "documents",
                 groups = docvars(data_corpus_irishbudget2010, "party"))
```

```
# Estimate Wordfish model
wf <- textmodel_wordfish(dfm(tokens(data_corpus_irishbudget2010)), dir = c(6, 5))

# Plot estimated word positions
textplot_scale1d(wf, margin = "features",
        highlighted = c("government", "global", "children",
                "bank", "economy", "the", "citizenship",
                "productivity", "deficit"),
        highlighted_color = "red")


# Plot estimated document positions
textplot_scale1d(wf, groups = data_corpus_irishbudget2010$party)


# Transform corpus to dfm
ie_dfm <- dfm(tokens(data_corpus_irishbudget2010))

# Run correspondence analysis on dfm
ca <- textmodel_ca(ie_dfm)

# Plot estimated positions and group by party
textplot_scale1d(ca, margin = "documents",
        groups = docvars(data_corpus_irishbudget2010, "party"))
```

ASSIGNMENT ???