EDS 231: Assignment 3

Scout Leonard

04/25/2022

Load Libraries

The code chunk below loads libraries used in this homework's R code.

```
packages=c("quanteda.sentiment",
           "quanteda.textstats",
           "tidyverse",
           "tidytext",
           "lubridate",
           "here",
           "wordcloud", #visualization of common words in the data set
           "reshape2",
           "knitr",
           "reshape",
           "patchwork",
           "MetBrewer",
           "dplyr",
           "sentimentr",
           "quanteda")

for (i in packages) {
  if (require(i,character.only=TRUE)==FALSE) {
    install.packages(i,repos='http://cran.us.r-project.org')
  }
  else {
    require(i,character.only=TRUE)
  }
}
```

Import Data

The code chunk below calls data from the GitHub of my EDS 242 instructor. The data is Twitter data that includes tweets about International Panel on Climate Change (IPCC) from select dates before and after the publication of the 2022 IPCC report. The chunk also does some cleaning for the first parts of the homework. `raw_tweets` is used throughout.

```
raw_tweets <- read.csv("https://raw.githubusercontent.com/MaRo406/EDS_231-text-sentiment/main/dat/IPCC_

dat<- raw_tweets[,c(4,6)] # Extract Date and Title fields

tweets <- tibble(text = dat$Title,
                 id = seq(1:length(dat$Title)),
```

```
                date = as.Date(dat$Date,'%m/%d/%y'))
```

Part 1

Think about how to further clean a twitter data set. Let's assume that the mentions of twitter accounts is not useful to us. Remove them from the text field of the tweets tibble.

Remove mentions and clean

The code chunk below cleans data containing tweet content by removing mentions, stop words, numbers, and meaningless URL bits so that the text analysis can focus on words meaningful to analyzing IPCC text and sentiment.

```
#remove mentions
tweets$text <- str_remove(tweets$text, "@[a-z,A-Z]*")
tweets$text <- str_remove(tweets$text, "[:digit:]")

#tokenise tweets and remove stop words
words <- tweets %>%
  select(id, date, text) %>%
  unnest_tokens(output = word, input = text, token = "words") %>%
  anti_join(stop_words, by = "word")

#clean tokens
##remove numbers
clean_tokens <- str_remove_all(words$word, "[:digit:]")
##remove mentions
clean_tokens <- str_remove_all(clean_tokens, "@[a-z,A-Z]*")
##removes appostrophes
clean_tokens <- gsub("'s", '', clean_tokens)
## remove weird twitter and internet things
clean_tokens <- str_remove_all(clean_tokens, "https")
clean_tokens <- str_remove_all(clean_tokens, "t.co")

words$clean <- clean_tokens

#remove the empty strings
tib <-subset(words, clean != "")

#reassign
words <- tib
```

Part 2

Compare the ten most common terms in the tweets per day. Do you notice anything interesting?

Count word use frequency and filter data for the top 10

```
#select the top 10 used words
words_freq <- words %>%
  group_by(clean) %>%
  summarise(freq = n()) %>%
   top_n(10) %>%
  select(clean)
```

```
## Selecting by freq
```

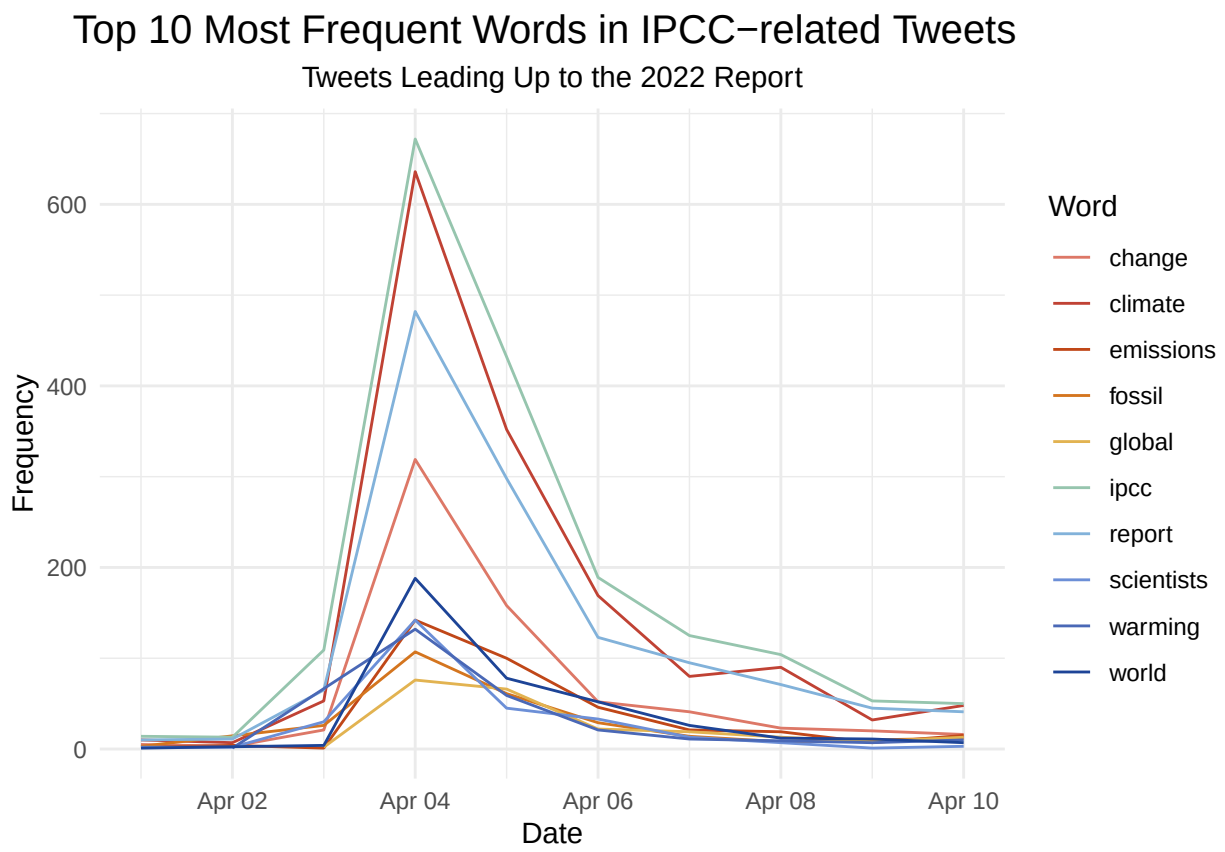
```
#join with full dataset to return frquency of those words / day
top_10_words <- inner_join(words_freq, words, by = "clean") %>%
  group_by(date, clean) %>%
  summarize(freq = n())
```

```
## `summarise()` has grouped output by 'date'. You can override using the
## `.groups` argument.
```

Plot use of top 10 words over time

The code chunk below plots a time series of how the top 10 words in IPCC tweet data changes over time - before, during, and after publication of the 2022 report.

```
ggplot(data = top_10_words, aes(x = date, y = freq)) +
  geom_line(aes(color = clean)) +
  theme_minimal() +
  labs(title = "Top 10 Most Frequent Words in IPCC-related Tweets",
       subtitle = "Tweets Leading Up to the 2022 Report",
       x = "Date",
       y = "Frequency",
       color = "Word") +
  theme(plot.title = element_text(hjust = 0.7, size = 15),
        plot.subtitle = element_text(hjust = 0.5)) +
  scale_color_manual(values = met.brewer("Nizami", n = 10))
```



I notice that there is a spike in all the top 10 words on April 4th, when the report is published. I assume this is because there is lots of coverage on this day compared to others.

"IPCC", "report", and "climate" are all the most frequent words the day the report is published and tend to stay highest for several days after, even as the frequency of all words declines.

Part 3

Adjust the wordcloud in the "wordcloud" chunk by coloring the positive and negative words so they are identifiable.

Plot wordcloud

```
words %>%
inner_join(get_sentiments("bing")) %>%
count(word, sentiment, sort = TRUE) %>%
acast(word ~ sentiment, value.var = "n", fill = 0) %>%
comparison.cloud(colors = c("darkslateblue", "goldenrod"),
                 max.words = 100)
```

Part 4

Let's say we are interested in the most prominent entities in the Twitter discussion. Which are the top 10 most tagged accounts in the data set. Hint: the "explore_hashtags" chunk is a good starting point.

The code chunk below tokenizes words from tweets and removes puctuation except for the "@" symbol, so that I can count the most frequntly metioned Twitter accounts in the IPCC tweets. The 10 most frequent accounts are printed in the table below.

```
corpus <- corpus(dat$Title)

#tokenize words from Tweets
tokens <- tokens(corpus)

#clean up tokens
tokens <- tokens(tokens, remove_punct = TRUE,
                         remove_numbers = TRUE)

tokens <- tokens_select(tokens, stopwords('english'),selection='remove') #stopwords lexicon built in to

tokens <- tokens_wordstem(tokens)

tokens <- tokens_tolower(tokens)

at_tweets <- tokens(corpus, remove_punct = TRUE) %>%
               tokens_keep(pattern = "@*")

dfm_at<- dfm(at_tweets) #document feature matrix

tstat_freq <- textstat_frequency(dfm_at, n = 100)

#tidytext gives us tools to convert to tidy from non-tidy formats
at_tib<- tidy(dfm_at)

#print table of 10 most mentioned accounts
at_tib %>%
   group_by(term) %>%
  summarise(n()) %>%
  top_n(10) %>%
  kable()
```

term	n()
@antonioguterres	16
@conversationedu	10
@ipcc	9
@ipcc_ch	131
@logicalindians	38
@nytimes	14
@potus	13
@un	12
@yahoo	14
@youtube	11

Part 5

The Twitter data download comes with a variable called "Sentiment" that must be calculated by Brandwatch. Use your own method to assign each tweet a polarity score (Positive, Negative, Neutral) and compare your classification to Brandwatch's (hint: you'll need to revisit the "raw_tweets" data frame).

Use my method to assign sentiment

In the code chunk below, I get the sentiment of the raw tweets and add those sentiments to the raw tweets dataframe:

```
#give raw tweets element id
raw_tweets_trimmed <- data.frame(element_id = seq(1:length(raw_tweets$Title)),
                        Date = raw_tweets$Date,
                        Title = raw_tweets$Title)

#make a character vector to get sentiments from
tweets_text <- raw_tweets_trimmed$Title

#calculate the mean sentiment based on all sentences sentences in single tweet
tweet_sent <- sentiment(tweets_text) %>%
  group_by(element_id) %>%
  summarise(mean_sent = mean(sentiment))

#join with sentence data
tweet_sent <- inner_join(raw_tweets_trimmed, tweet_sent, by = "element_id")
```

Next, I categorize the mean sentiment value for all of the sentences in a tweet as positive, negative, or neutral. I add these categories to my original dataframe of raw tweets so that I can compare my categorization method with the Brandwatch classification results.

```
# catgorize for ggplot
tweet_cat <- tweet_sent %>%
  mutate(sent_cat = case_when(
    mean_sent < 0 ~ "negative",
    mean_sent == 0 ~ "neutral",
    mean_sent > 0 ~ "positive"
  ))

raw_tweets <- raw_tweets %>%
  mutate(element_id = seq(1:length(Title)))

all_sent <- full_join(tweet_cat, raw_tweets, by = "element_id", copy = TRUE) %>%
  mutate_all(na_if, "")
```

Plot both methods and compare

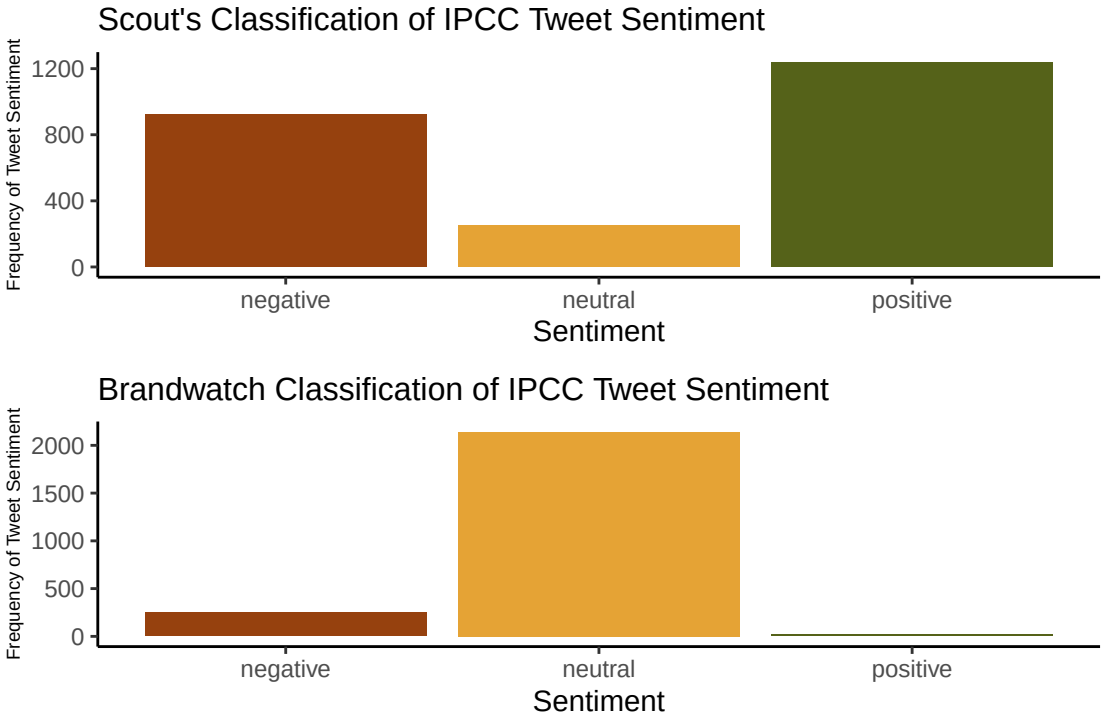
Finally, I plot the frequency of IPCC tweets classified as positive, negative, or neutral using the two methods, mine and Brandwatch:

```
p1 <- ggplot(data = all_sent, aes(x = sent_cat)) +
  geom_bar(stat = "count",
           aes(fill = sent_cat),
           show.legend = FALSE) +
  theme_classic() +
  labs(title = "Scout's Classification of IPCC Tweet Sentiment",
```

```
       x = "Sentiment",
       y = "Frequency of Tweet Sentiment") +
  theme(plot.title = element_text(size = 12),
        axis.title.y = element_text(size = 7)) +
  scale_fill_manual(values = met.brewer("Degas", 3))

p2 <- ggplot(data = all_sent, aes(x = Sentiment)) +
  geom_bar(stat = "count",
           aes(fill = Sentiment),
           show.legend = FALSE) +
  theme_classic() +
  theme(plot.title = element_text(size = 12),
        axis.title.y = element_text(size = 7)) +
  labs(title = "Brandwatch Classification of IPCC Tweet Sentiment",
       y = "Frequency of Tweet Sentiment") +
  scale_fill_manual(values = met.brewer("Degas", n = 3))

p1 / p2
```



The resulting figure shows that the Brandwatch sentiment analysis classification of tweet sentiment assigned many more tweets as "neutral" compared to my method, which assigns tweets as negative or positive in a less conservative way.