Assignment 4: Sentiment Analysis II

Halina Do-Linh

04/25/2022

First round of tidying: I created the data frame needed to do the sentiment analysis.

```
ipcc_twitter_clean <- tibble(id = seq(1:length(ipcc_twitter_data$title)),
                             date = as.Date(ipcc_twitter_data$date, "%m/%d/%y"),
                             text = ipcc_twitter_data$title)
```

Here is the cleaning we did from the lab in class.

```
# remove URLs from the tweets
ipcc_twitter_clean$text <- gsub("http[^[:space:]]*", "",
                                ipcc_twitter_clean$text)

# make all the text lowercase
ipcc_twitter_clean$text <- str_to_lower(ipcc_twitter_clean$text)
```

Remove mentions of twitter accounts from the text field of the tweets tibble.

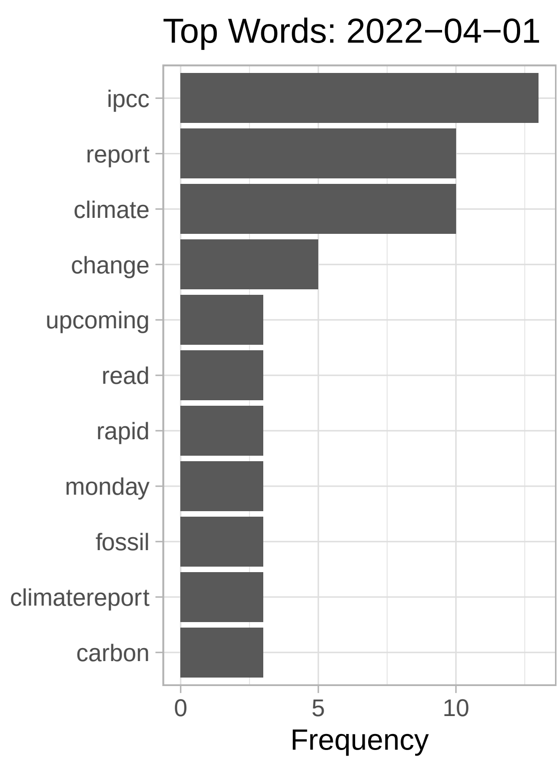
```
# removing mentions from tweets
ipcc_twitter_clean$text <- gsub("@\\w+", "", ipcc_twitter_clean$text)
```

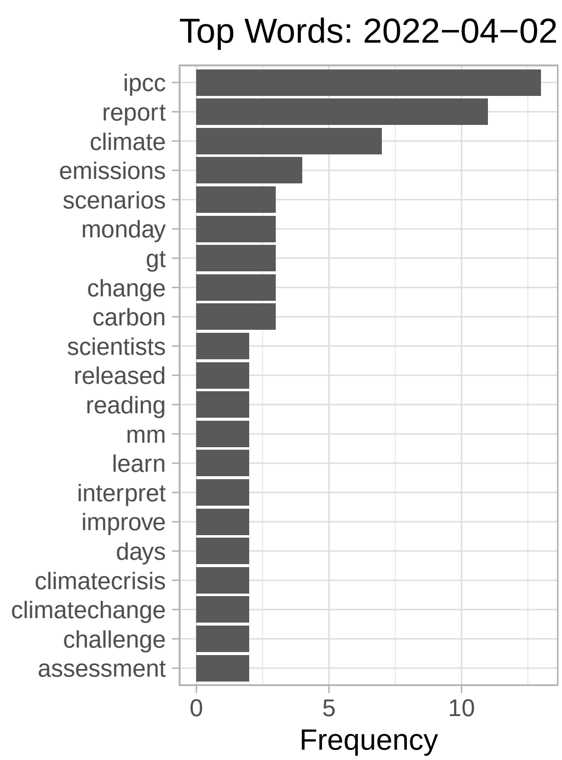
Now I'm going to load the sentiment lexicons and tokenize the tweets.

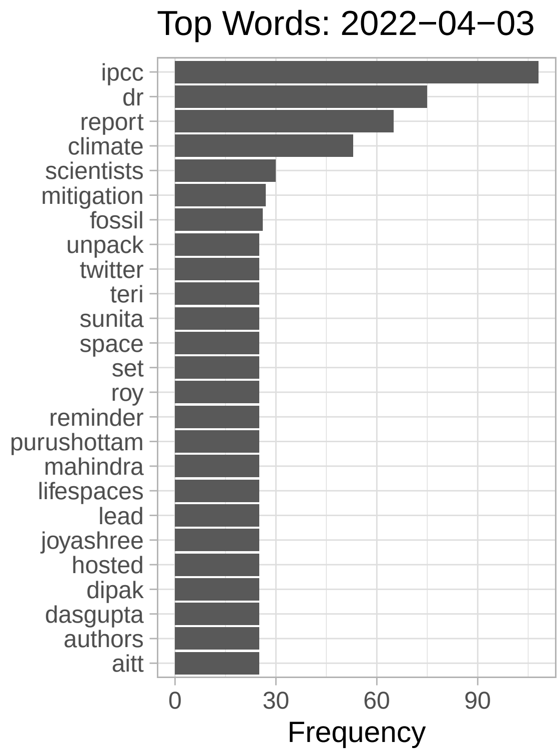
```
# load sentiment lexicons
bing_sent <- get_sentiments('bing')
nrc_sent <- get_sentiments('nrc')

ipcc_twitter_words <- ipcc_twitter_clean %>%
  select(id, date, text) %>%
  # tokenize tweets to individual words
  unnest_tokens(output = word,
                input = text,
                token = "words") %>%
  anti_join(stop_words, by = "word") %>%
  # remove digits
  mutate(word = str_remove_all(string = word, pattern = "[:digit:]")) %>%
  # remove empty values
  filter(word != "") %>%
  left_join(bing_sent, by = "word") %>%
  left_join(tribble(~ sentiment, ~ sent_score,
                    "positive", 1,
                    "negative",-1),
            by = "sentiment")
```

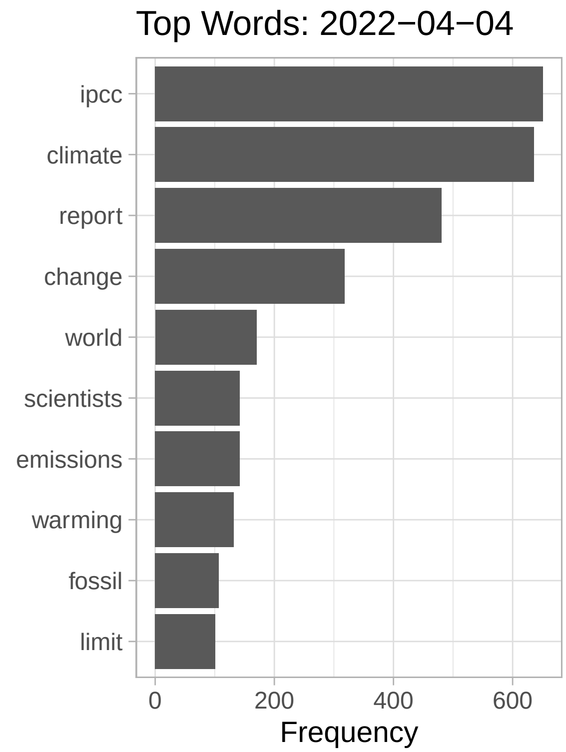
Before I move forward with sentiment analysis, I create a plot comparing the ten most common terms in the tweets per day. I noticed that some days have more than ten words because the same frequency occurs for those words on a specific day. Additionally, the words "ipcc", "climate", and "report" are the top 3 words for every day except for 2022-04-03 where the second more common word is "dr".

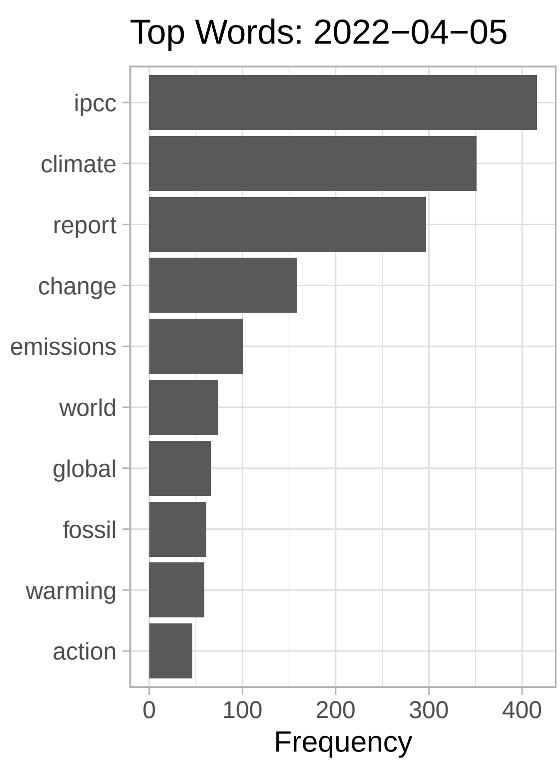
```
common_tweets <- ipcc_twitter_words %>%
  group_by(date) %>%
  summarize(freq_terms(word, 10))

dates <- unique(common_tweets$date)
plot_list = list()
for (i in seq_along(dates)){

  df <- common_tweets %>%
    filter(date == dates[i])

  p <- ggplot(data = df, aes(x = reorder(WORD, FREQ), y = FREQ)) +
    geom_bar(stat = "identity") +
    coord_flip() +
    theme_light() +
    labs(title = paste("Top Words:", dates[[i]]),
       x = NULL,
       y = "Frequency")

  plot_list[[i]] = p

  print(p)
}
```

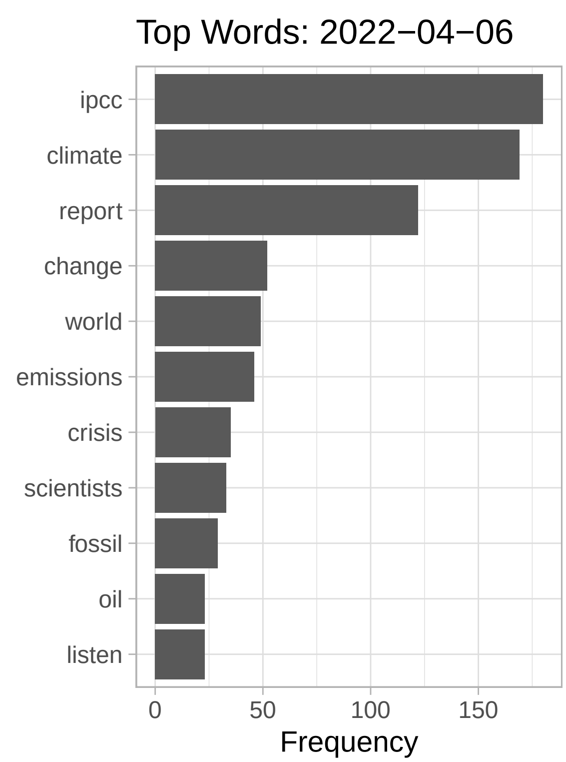


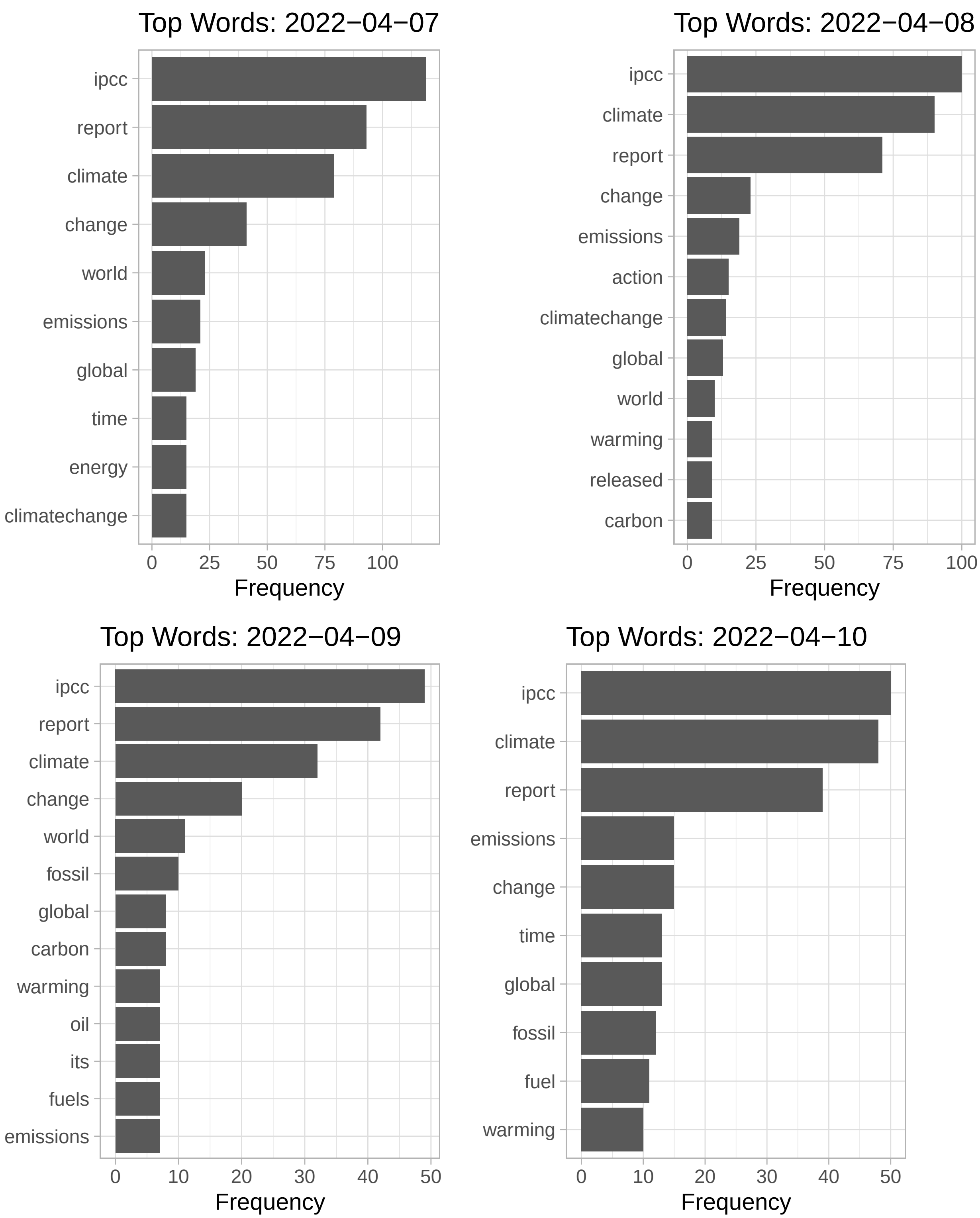



Top Words: 2022−04−03
ipcc
dr
report
climate
scientists
mitigation
fossil
unpack
twitter
teri
sunita
space
set
roy
reminder
purushottam
mahindra
lifespaces
lead
joyashree
hosted
dipak
dasgupta
authors
aitt
0
30
60
90
Frequency


Top Words: 2022−04−04
ipcc
climate
report
change
world
scientists
emissions
warming
fossil
limit
0
200
400
600
Frequency


Top Words: 2022−04−05
ipcc
climate
report
change
emissions
world
global
fossil
warming
action
0
100
200
300
400
Frequency


Top Words: 2022−04−06
ipcc
climate
report
change
world
emissions
crisis
scientists
fossil
oil
listen
0
50
100
150
Frequency



Top Words: 2022−04−07
ipcc
report
climate
change
world
emissions
global
time
energy
climatechange
0
25
50
75
100
Frequency
Top Words: 2022−04−08
ipcc
climate
report
change
emissions
action
climatechange
global
world
warming
released
carbon
0
25
50
75
100
Frequency
Top Words: 2022−04−09
ipcc
report
climate
change
world
fossil
global
carbon
warming
oil
its
fuels
emissions
0
10
20
30
40
50
Frequency
Top Words: 2022−04−10
ipcc
climate
report
emissions
change
time
global
fossil
fuel
warming
0
10
20
30
40
50
Frequency

Here I adjusted the wordcloud from lab so the coloring for positive and negative words are distinguishable.

```
ipcc_twitter_words %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("darkred", "#d8b365"),
                   max.words = 100)
```

To find the top 10 most tagged accounts in the data set, I first create a corpus using the `quanteda` package.

```
corpus <- corpus(ipcc_twitter_data$title) # enter quanteda
summary(corpus)
```

Then I tokenzie all the words in the corpus (which are tweets).

```
# tokenize the text so each tweet is a list of tokens
tokens <- tokens(corpus)
```

The tokenized words are pretty messy, so I need to do some cleaning before moving forward with my analysis. Here I am removing punctuation (okay to remove because "@" is a symbol), numbers and stop words.

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

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

tokens <- tokens_tolower(tokens)
```

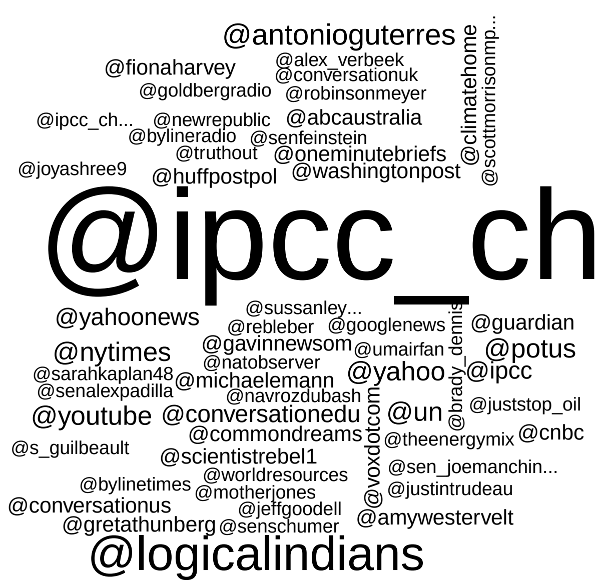
After cleaning the corpus, I use `tokens_keep()` to find all the mentions and then create a document-feature matrix (dfm) of the tokens with a tagged account. Then I use `textstat_frequency()` to create a data frame of all the tagged accounts and how frequent they are.

```r
mention_tweets <- tokens(corpus) %>% tokens_keep(pattern = "@*")

dfm_mention <- dfm(mention_tweets)

mention_freq <- textstat_frequency(dfm_mention, n = 100)
head(mention_freq, 10)
```

```
##              feature frequency rank docfreq group
## 1           @ipcc_ch       131    1     131   all
## 2    @logicalindians        38    2      38   all
## 3  @antonioguterres         16    3      16   all
## 4           @nytimes        14    4      14   all
## 5             @yahoo        14    4      14   all
## 6             @potus        13    6      13   all
## 7                @un        12    7      12   all
## 8           @youtube        11    8      11   all
## 9   @conversationedu        10    9      10   all
## 10             @ipcc         9   10       9   all
```

Lastly, I created a wordcloud of all the mentions.

```r
# tidytext gives us tools to convert to tidy from non-tidy formats
mention_tib <- tidy(dfm_mention)

mention_tib %>%
  count(term) %>%
  with(wordcloud(term, n, max.words = 100))
```



Here I am calling in the "raw tweets" with the sentiment scores calculated by Brandwatch.

```
# read in data and basic cleaning
ipcc_twitter_sent <- read_csv(here("hw/data/IPCC_tweets_April1-10_sample.csv")) %>%
  clean_names() %>%
  select(c("date", "title", "sentiment"))
```

Here I calculated a polarity score and assigned each tweet a polarity of Positive, Negative, or Neutral.

```
# take average sentiment score by tweet
my_sent <- ipcc_twitter_clean %>%
  get_sentences(text) %>% # puts data in right format to use sentiment()
  sentiment() %>%
  group_by(id) %>%
  summarize(sent_score = mean(sentiment))

# assign scores a category of positive, neutral, or negative
my_sent_classified <- my_sent %>%
  mutate(sent_score = case_when(sent_score > 0 ~ "positive",
                               sent_score == 0 ~ "neutral",
                               sent_score < 0 ~ "negative"))
```

I created two plots to show the comparison between the two sentiment classifications.

```
# brandwatch wrangling and plot
brandwatch_sent <- ipcc_twitter_sent %>%
  group_by(sentiment) %>%
  summarize(count = n()) %>%
  ggplot(aes(x = reorder(sentiment, count), y = count,
             fill = sentiment)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  theme_classic() +
  theme(legend.position = "bottom") +
  labs(title = "Brandwatch Sentiment",
       x = "Sentiment",
       y = "Count") +
  geom_text(aes(x = sentiment, y = count,
            label = count),
            nudge_y = 80) +
  scale_fill_manual(name = NULL,
                    values = c("#d8b365", "lightgray", "#5ab4ac"))

# my calculate sentiment wrangling and plot
my_sent_plot <- my_sent_classified %>%
  group_by(sent_score) %>%
  summarize(count = n()) %>%
  ggplot(aes(x = reorder(sent_score, count), y = count,
             fill = sent_score)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  theme_classic() +
  theme(legend.position = "none") +
```

```
  labs(title = "My Calculated Sentiment",
       x = "Sentiment",
       y = "Count") +
  geom_text(aes(x = sent_score, y = count,
            label = count),
            nudge_y = 50) +
  scale_fill_manual(name = "Sentiment",
                    values = c("#d8b365", "lightgray", "#5ab4ac"))

brandwatch_sent / my_sent_plot
```