EDS 231 - Sentiment Analysis II - Twitter

Juliet Cohen

2022-04-26

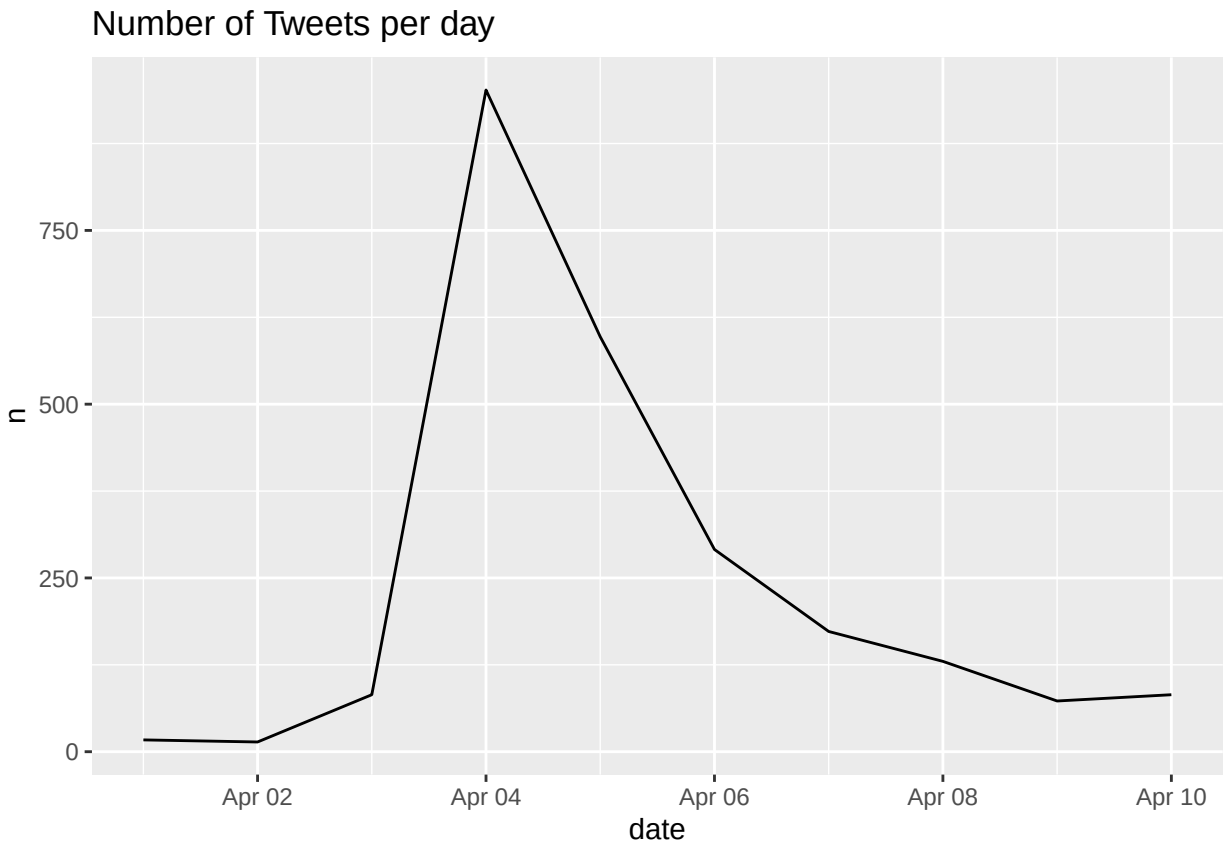
Read in Twitter data:

```
raw_tweets <- read.csv("/Users/juliet/Documents/MEDS/Text_Analysis/Text_Analysis/Assignment_3/IPCC_twee

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

# create a tibble from the tweet title and date columns
tweets <- tibble(text = data$Title,
                 id = seq(1:length(data$Title)),
                 date = as.Date(data$Date,'%m/%d/%y'))

# simple plot of tweets per day
tweets %>%
  count(date) %>%
  ggplot(aes(x = date, y = n))+
  geom_line() +
  labs(title = "Number of Tweets per day")
```



Data Cleaning

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.

To clean the data, I removed website links, emojis, @ symbols, and # symbols because mentions of twitter accounts are not useful to us, plus I converted all text to lower case and removed "'s", "'t", and digits.

```
# let's clean up the URLs from the tweets
tweets$text <- gsub("http[^[:space:]]*", "", tweets$text)

# remove emojis
tweets$text <- iconv(tweets$text, "latin1", "ASCII", sub="")

# remove @ and the name of the account tagged because we dont need tagged people in this analysis, just
tweets$text <- gsub("@[^[:space:]]*", "", tweets$text)

# remove # but keep the word following
tweets$text <- gsub("#", "", tweets$text)

# remove the @ symbol and the account that follows
tweets$text <- gsub("@[a-z,A-Z]*", "", tweets$text)

# convert all text to lower case
tweets$text <- str_to_lower(tweets$text)
```

```
# remove 's
tweets$text <- gsub("'s", "", tweets$text)

# remove 't
tweets$text <- gsub("'t", "", tweets$text)

# remove digits
tweets$text <- str_remove_all(tweets$text, "[:digit:]")

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

# tokenize tweets to individual words so they will be 1 word per row
words <- tweets %>%
  select(id, date, text) %>%
  unnest_tokens(output = word, input = text, token = "words") %>%
  anti_join(stop_words, by = "word") %>% # remove stop words
  left_join(bing_sent, by = "word") %>% # join the words to the sentiment words (label with a sentiment)
  left_join(
    tribble(
      ~sentiment, ~sent_score,
      "positive", 1,
      "negative", -1),
    by = "sentiment")

# the new sentiment score column is numerical, it is how we assign sentiment to words
# besides just pos/neg/neutral/etc.
```

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

I noticed that there were much higher word counts on the dates on April 4th, followed by April 5th and 6th, and the word "ipcc" and "climate" appear on all days. I also noticed that the word "fossil" appears as a top word on many of the days. Overall, the words are more practical, scientific words rather than words that are strongly associated with emotions.

By looking at the total top 10 words over all these days, we can see that they are all reflective of what I would expect to see for this topic.

```
# examine trends of the top 10 words per day
top_daily_words <- words %>%
  group_by(date, word) %>%
  summarize(count = n()) %>%
  arrange(desc(count)) %>%
  slice(1:10)
```

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

```
# look for the individual words "ipcc", "fossil", "time", and "climate"
ipcc_mentions <- top_daily_words %>%
  filter(word == "ipcc")

fossil_mentions <- top_daily_words %>%
```

```
  filter(word == "fossil")

time_mentions <- top_daily_words %>%
  filter(word == "time")

climate_mentions <- top_daily_words %>%
  filter(word == "climate")

top_daily_words_table <- kable(top_daily_words,
                          caption = "Top 10 Daily Words by Day")
top_daily_words_table
```

```
# examine trends of the top 10 words total
top_10_total <- words %>%
  group_by(word) %>%
  summarize(count = n()) %>%
  slice_max(count, n = 10)

top_10_total_table <- kable(top_10_total,
                          caption = "Top 10 Daily Words Total")
top_10_total_table
```

```
# plot the top 10 words total
ggplot(data = top_10_total, aes(y = reorder(word, count), x = count)) +
  geom_col(aes(fill = word, width = .5)) +
  theme_classic() +
  labs(title = "Top 10 Words Total\nApril 1st, 2022 - April 10th, 2022",
       x = "Count",
       y = "Word")
```

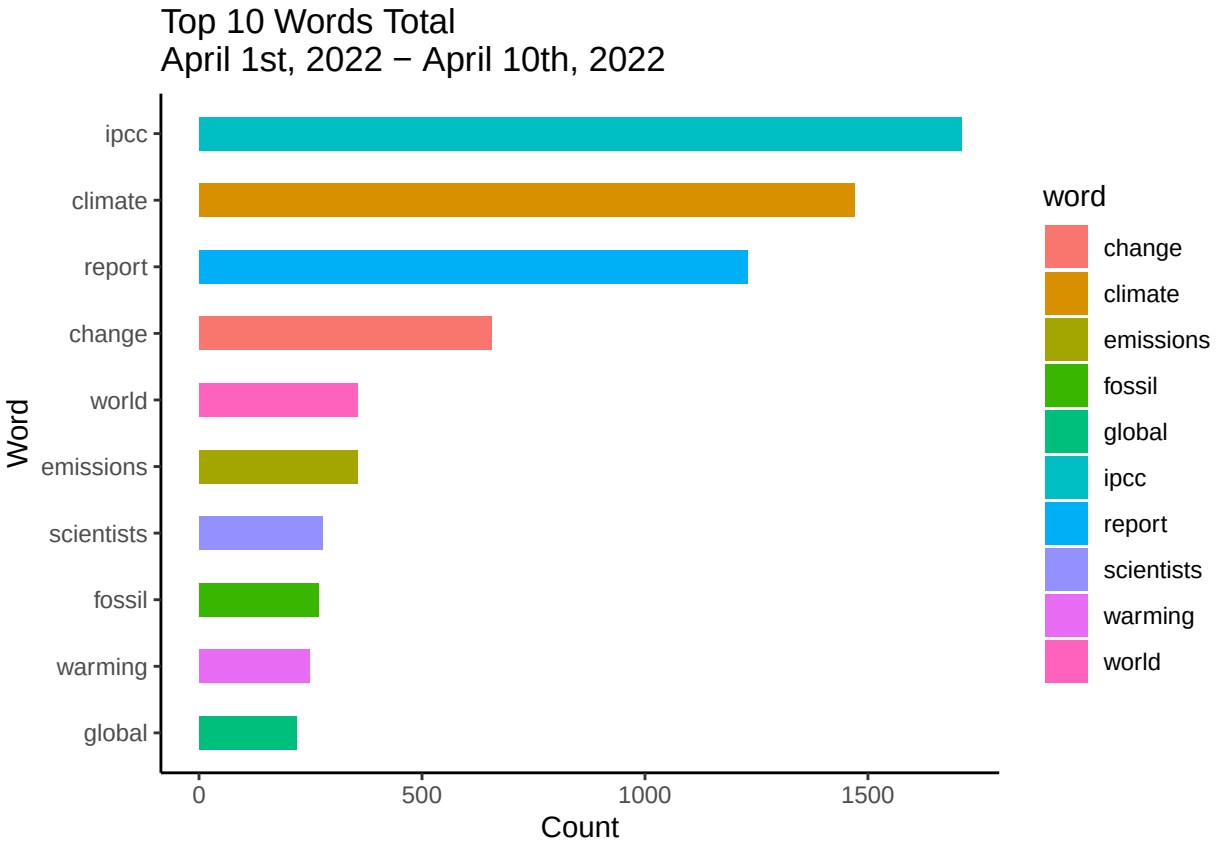
```
## Warning: Ignoring unknown aesthetics: width
```

Table 1: Top 10 Daily Words by Day

date	word	count
2022-04-01	ipcc	13
2022-04-01	report	10
2022-04-01	climate	9
2022-04-01	change	4
2022-04-01	carbon	3
2022-04-01	climatereport	3
2022-04-01	fossil	3
2022-04-01	monday	3
2022-04-01	rapid	3
2022-04-01	read	3
2022-04-02	ipcc	13
2022-04-02	report	11
2022-04-02	climate	7
2022-04-02	emissions	4
2022-04-02	carbon	3
2022-04-02	change	3
2022-04-02	gt	3
2022-04-02	monday	3
2022-04-02	scenarios	3
2022-04-02	assessment	2
2022-04-03	ipcc	107
2022-04-03	dr	75
2022-04-03	report	64
2022-04-03	climate	53
2022-04-03	scientists	30
2022-04-03	mitigation	27
2022-04-03	fossil	26
2022-04-03	aitt	25
2022-04-03	authors	25
2022-04-03	dasgupta	25
2022-04-04	ipcc	652
2022-04-04	climate	634
2022-04-04	report	480
2022-04-04	change	320
2022-04-04	world	174
2022-04-04	emissions	142
2022-04-04	scientists	141
2022-04-04	warming	132
2022-04-04	fossil	105
2022-04-04	limit	101
2022-04-05	ipcc	418
2022-04-05	climate	350
2022-04-05	report	297
2022-04-05	change	159
2022-04-05	emissions	98
2022-04-05	world	76
2022-04-05	global	66
2022-04-05	fossil	61
2022-04-05	warming	59
2022-04-05	action	46
2022-04-06	ipcc	180
2022-04-06	climate	169
2022-04-06	report	121
2022-04-06	change	52
2022-04-06	world	49

Table 2: Top 10 Daily Words Total

word	count
ipcc	1709
climate	1470
report	1231
change	657
world	356
emissions	355
scientists	276
fossil	268
warming	248
global	219



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

```
# sentiment wordcloud
words %>%
  # attach sentiments
inner_join(get_sentiments("bing")) %>%
  # count the sentiment words, showing largest groups at the top
count(word, sentiment, sort = TRUE) %>%
  # aggregate data by sentiment
acast(word ~ sentiment, value.var = "n", fill = 0) %>%
```

```
comparison.cloud(colors = c("firebrick", "forestgreen"),
                 max.words = 100,
                 title.colors = c("firebrick", "forestgreen"))
```

```
## Joining, by = c("word", "sentiment")
```

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.

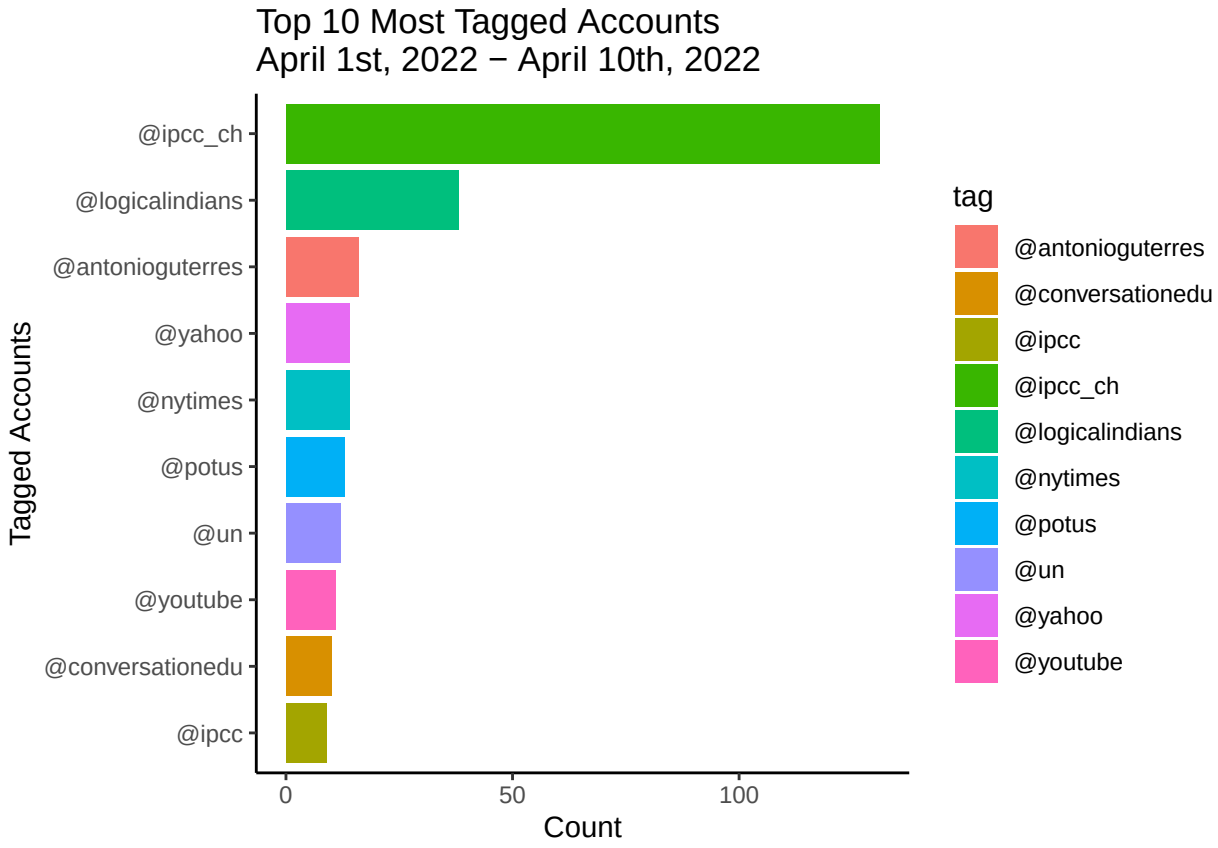
```
# import the corpus for this data
corpus <- corpus(data$Title)
#summary(corpus)

# make the corpus into a tokens object, without punctuation, do not remove numbers because they might b
tokens <- tokens(corpus, remove_punct = TRUE)

# only keep those that have tagged accounts, then convert that into a document-feature matrix and renam
tags <- tokens %>%
  tokens_keep(pattern = "@*") %>%
  dfm() %>%
  textstat_frequency(n = 10) %>%
  rename(tag = feature)

ggplot(data = tags, aes(x = frequency, y = reorder(tag, frequency))) +
  geom_col(aes(fill = tag)) +
  theme_classic() +
```

```
  labs(x = "Count",
       y = "Tagged Accounts",
       title = "Top 10 Most Tagged Accounts\nApril 1st, 2022 - April 10th, 2022")
```



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).

```
# tokenize tweets to individual words so they will be 1 word per row
sentiment <- tweets %>%
  select(id, date, text) %>%
  unnest_tokens(output = word, input = text, token = "words") %>%
  anti_join(stop_words, by = "word") %>% # remove stop words
  left_join(bing_sent, by = "word") %>% # join the words to the sentiment words (label with a sentiment
  left_join(
    tribble(
      ~sentiment, ~sent_score,
      "positive", 1,
      "negative", -1),
    by = "sentiment")

my_sentiment <- get_sentences(data$Title) %>%
  sentiment() %>%
  group_by(element_id) %>%
  summarize(sentiment_score = mean(sentiment)) %>%
```

```
  mutate(sentiment = case_when(
    sentiment_score < 0 ~ "negative",
    sentiment_score == 0 ~ "neutral",
    sentiment_score > 0 ~ "positive")) %>%
  group_by(sentiment) %>%
  summarize(count = n()) %>%
  mutate(color = c("firebrick", "purple", "forestgreen"))

my_plot <- ggplot(data = my_sentiment, aes(x = count, y = reorder(sentiment, count), fill = color)) +
  geom_col(stat="identity") +
  xlim(0, 2400) +
  scale_fill_identity() +
  labs(x = "",
       y = "",
       subtitle = "My Sentiment Analysis")
```

```
## Warning: Ignoring unknown parameters: stat
```

```
# Brandwatch sentiment
raw_tweets <- read.csv("/Users/juliet/Documents/MEDS/Text_Analysis/Text_Analysis/Assignment_3/IPCC_twee

# Extract Date and Title fields
data <- raw_tweets[,c(4,6, 10:11)]

clean_tweets <- tibble(id = seq(1:length(data$Title)),
                text = data$Title,
                date = as.Date(data$Date,'%m/%d/%y'),
                sentiment = data$Sentiment,
                emotion = data$Emotion)

brandwatch_sentiment <- clean_tweets %>%
  filter(sentiment %in% c("positive", "negative", "neutral")) %>%
  group_by(sentiment) %>%
  summarize(count = n()) %>%
  mutate(color = c("firebrick", "purple", "forestgreen"))

brandwatch_plot <- ggplot(data = brandwatch_sentiment, aes(x = count, y = reorder(sentiment, count), fi
  geom_col() +
  xlim(0, 2400) +
  scale_fill_identity() +
  labs(x = "",
       y = "",
       subtitle = "Brandwatch Sentiment Analysis")

# use patchwork to combine plots
combined_plots <- (brandwatch_plot / my_plot) +
  plot_annotation(title = "Tweet Sentiment: My Analysis versus Brandwatch Analysis") +
  plot_layout(guides = "collect")

combined_plots
```

Tweet Sentiment: My Analysis versus Brandwatch Analysis