Topic 4: Sentiment Analysis II

Felicia Cruz

4/20/2022

```r
library(quanteda)
library(quanteda.sentiment)
library(quanteda.textstats)
library(tidyverse)
library(tidytext)
library(lubridate)
library(wordcloud)
library(reshape2)
library(here)
library(kableExtra)
```

Load IPCC Report Twitter Data

```r
raw_tweets <- read.csv(here("labs", "data", "IPCC_tweets_April1-10_sample.csv"))

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'))
```

Clean tweets

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

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

# tokenize tweets to individual words
words <- tweets %>%
  select(id, date, text) %>%
  unnest_tokens(output = word, input = text, token = "words") %>%
  anti_join(stop_words, by = "word") %>%
  left_join(bing_sent, by = "word") %>%
  left_join(
    tribble(
      ~sentiment, ~sent_score,
      "positive", 1,
      "negative", -1),
    by = "sentiment")
```

Assignment

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.

```
tweets_new <- tweets # duplicate the original cleaned tweets df

tweets_new$text <- str_replace_all(tweets_new$text, "@[a-z,A-Z]*","")
head(tweets_new, 10)
```

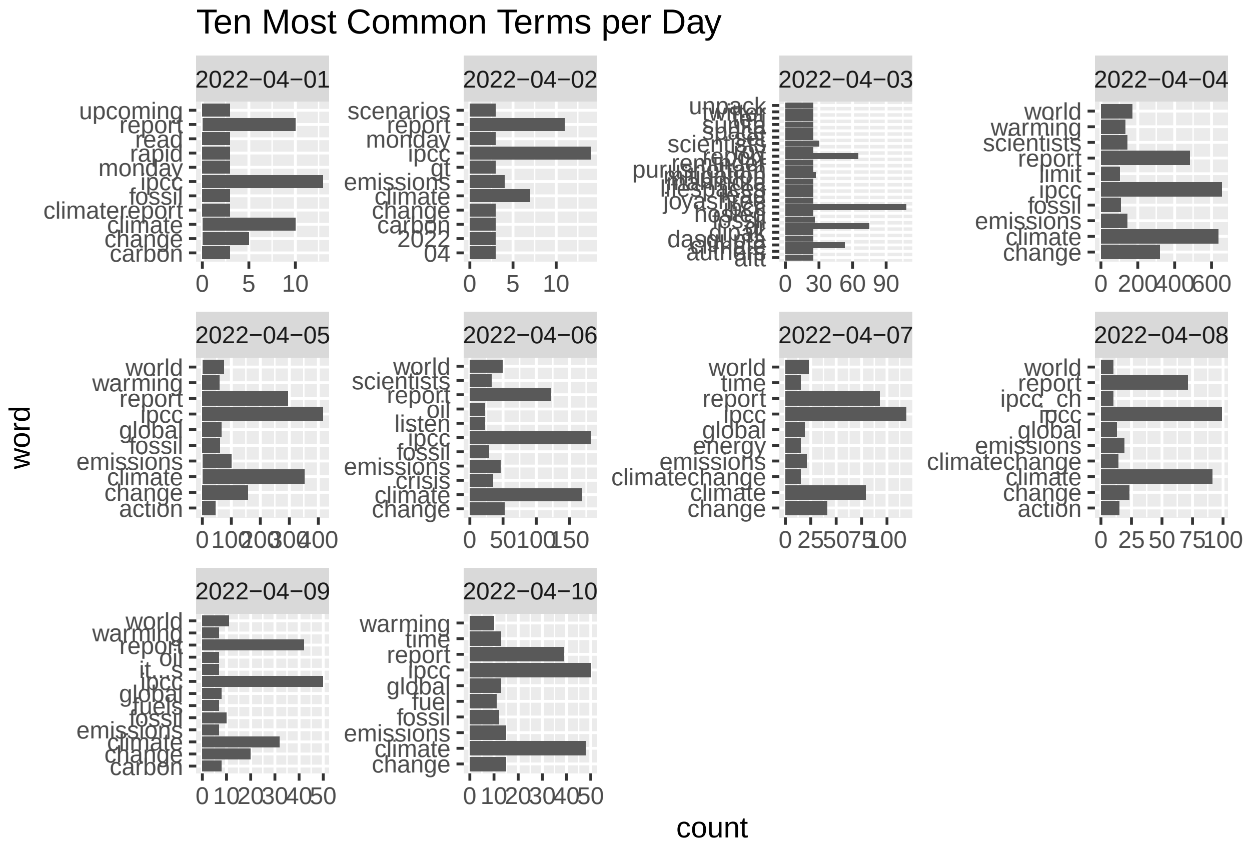
```
## # A tibble: 10 x 3
##    text                                                            id date
##    <chr>                                                        <int> <date>
##  1 "thank you, followers, for the great photo suggestions for ~     1 2022-04-01
##  2 "greenpeace: the real solution to the climate crisis will r~     2 2022-04-01
##  3 "governments have a responsibility to ensure that #ipccrepo~     3 2022-04-01
##  4 "next week, the ipcc will publish a new report detailing th~     4 2022-04-01
##  5 "live stream of virtual ipcc press conference releasing the~     5 2022-04-01
##  6 "attention journalists: the deadline for embargoed material~     6 2022-04-01
##  7 "the ipcc report and "the physics of climate change" "           7 2022-04-01
##  8 "with time running short and most of the summary for policy~     8 2022-04-01
##  9 "a helpful perspective on how to talk about the scenarios d~     9 2022-04-01
## 10 "the private sector is an integral component of the water c~    10 2022-04-01
```

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

```
words_summary <- words %>%
  group_by(date, word) %>%
  summarize(count = n())
```

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

```
words_summary %>%
  group_by(date) %>%
  slice_max(order_by = count, n = 10) %>%
  ggplot(aes(x = count, y = word)) +
  geom_col() +
  facet_wrap(~as.factor(date),
             scales = "free") +
  labs(title = "Ten Most Common Terms per Day")
```

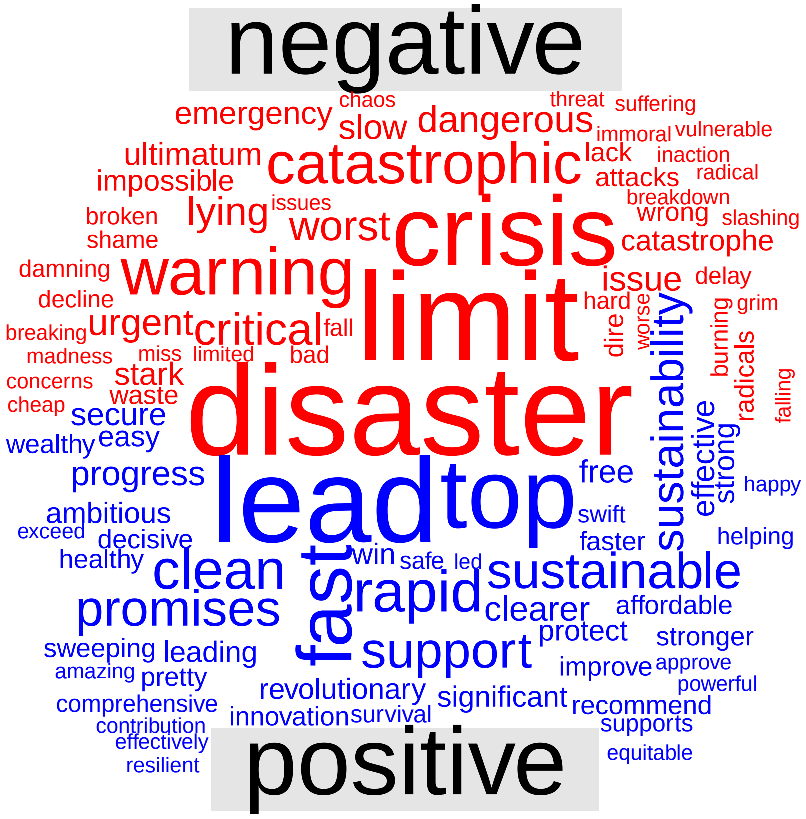


Looking at the most common words for each day, we can see that "world", "report", "ipcc", "emissions", "climate", and "change" are all top words for most of the days which makes sense. "ipcc" in particular is the most frequent for each day. On 4/3 there were a few ties for word frequency which is why that particular plot has more than 10 words on the y axis.

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

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

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



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.

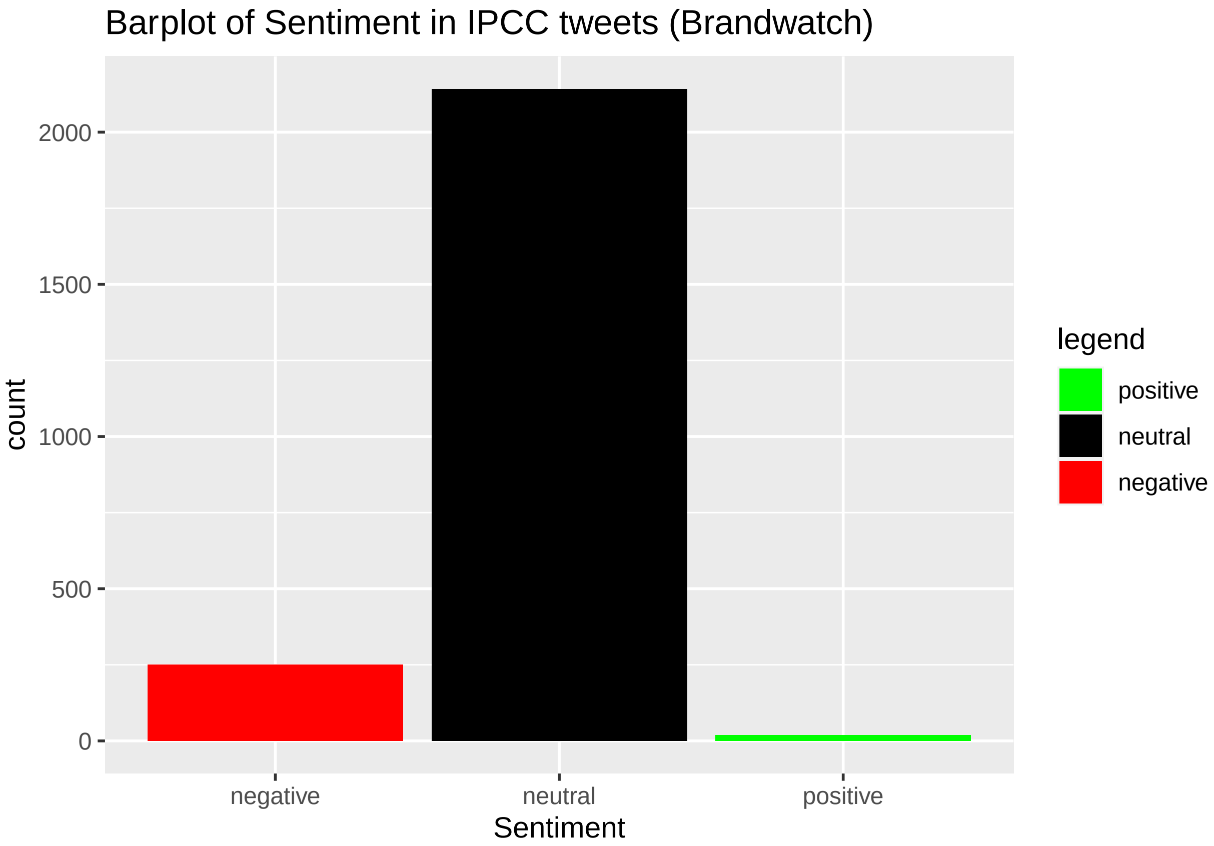
```
corpus <- corpus(dat$Title) # create corpus
```

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

dfm_account<- dfm(account_tweets)

tstat_freq <- textstat_frequency(dfm_account, n = 100) %>%
  head(10)

# find the top ten most tagged accounts
tstat_freq$feature
```

```
##  [1] "@ipcc_ch"          "@logicalindians"  "@antonioguterres" "@nytimes"
##  [5] "@yahoo"            "@potus"           "@un"              "@youtube"
##  [9] "@conversationedu" "@ipcc"
```

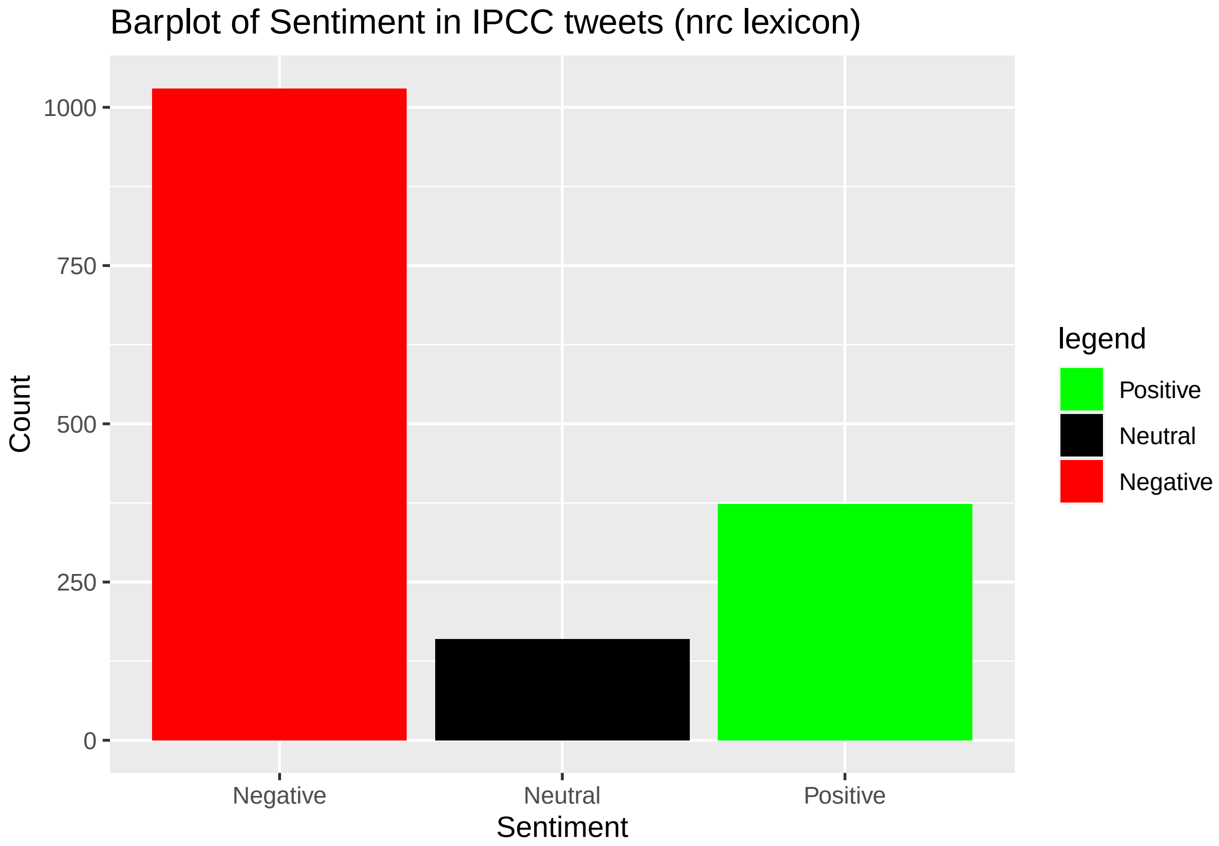
5. The Twitter data download comes with a variable called "Sentiment" that must be calculated by Brandwatch. Use your own method to assign each tweet a polarity score (Positive, Negative, Neutral) and compare your classification to Brandwatch's (hint: you'll need to revisit the "raw_tweets" data frame).

```
# bar graph of Sentiment column in raw_tweets df (Brandwatch)

raw_tweets %>%
  group_by(Sentiment) %>%
  summarize(count = n()) %>%
  ggplot(aes(x=Sentiment,y=count))+
  geom_bar(stat = "identity", aes(fill = Sentiment)) +
  scale_fill_manual("legend", values = c("positive" = "green", "neutral" = "black", "negative" = "red")
  ggtitle("Barplot of Sentiment in IPCC tweets (Brandwatch)")
```



```
# bar graph of the sentiments using nrc lexicon

words_5 <- tweets %>%
  select(id, date, text) %>%
  unnest_tokens(output = word, input = text, token = "words") %>%
  anti_join(stop_words, by = "word") %>%
  left_join(nrc_sent, by = "word") %>%
  left_join(
    tribble(
      ~sentiment, ~sent_score,
      "positive", 1,
      "negative", -1),
    by = "sentiment")

#take average sentiment score by tweet
```

```
tweets_sent_5 <- tweets %>%
  left_join(
    words %>%
      group_by(id) %>%
      summarize(
        sent_score = mean(sent_score, na.rm = T)),
    by = "id")

neutral <- length(which(tweets_sent_5$sent_score == 0))
positive <- length(which(tweets_sent_5$sent_score > 0))
negative <- length(which(tweets_sent_5$sent_score < 0))

Sentiment <- c("Negative","Neutral","Positive")
Count <- c(negative,neutral,positive)
output <- data.frame(Sentiment,Count)
output$Sentiment<-factor(output$Sentiment,levels=Sentiment)
ggplot(output, aes(x=Sentiment,y=Count))+
  geom_bar(stat = "identity", aes(fill = Sentiment))+
  scale_fill_manual("legend", values = c("Positive" = "green", "Neutral" = "black", "Negative" = "red"))
  ggtitle("Barplot of Sentiment in IPCC tweets (nrc lexicon)")
```



From these two bar graphs we can see that Brandwatch produces many neutral scores and very few positive scores, while my method, which used the nrc lexicon, produced a small amount of neutral scores, more positive scors, but a majority of negative scores.