Topic 4: Sentiment Analysis II

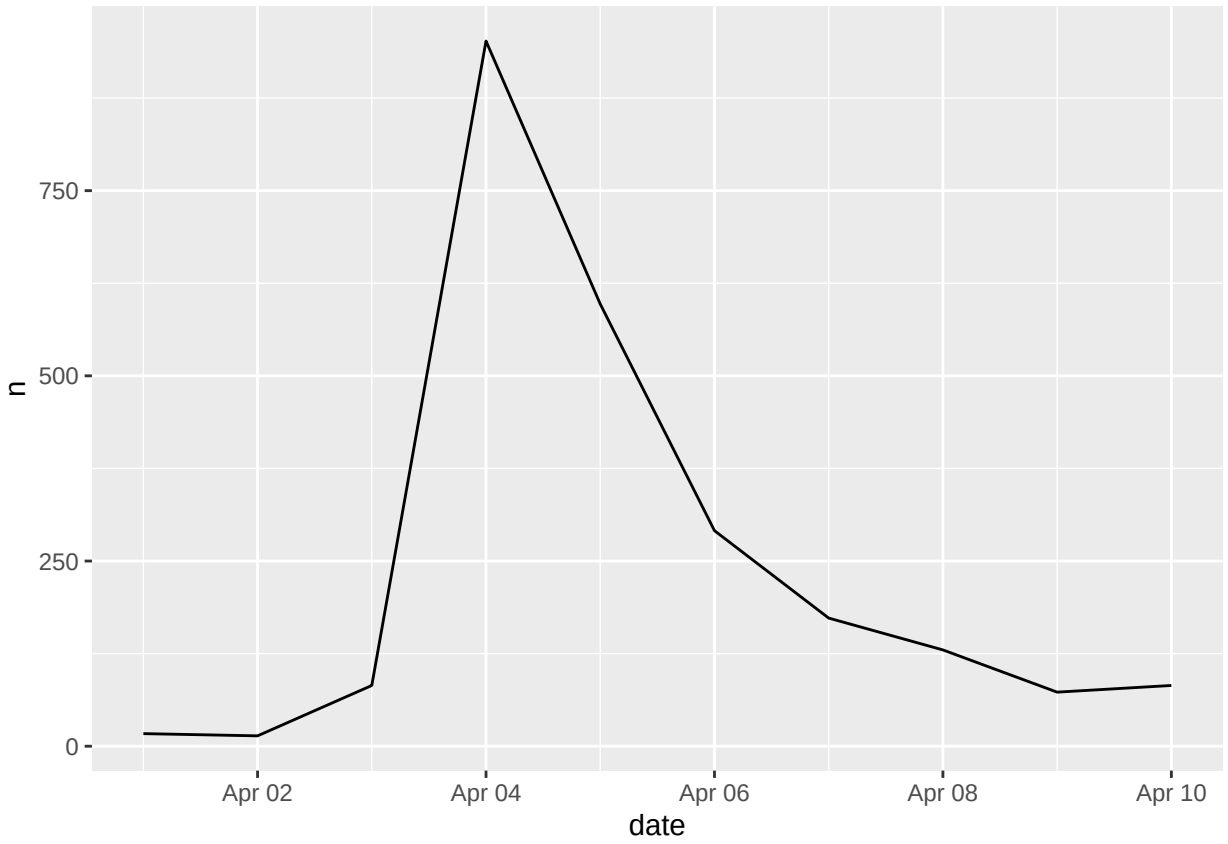
```
library(quanteda)
#devtools::install_github("quanteda/quanteda.sentiment") #not available currently through CRAN
library(quanteda.sentiment)
library(quanteda.textstats)
library(tidyverse)
library(tidytext)
library(lubridate)
library(wordcloud) #visualization of common words in the data set
library(reshape2)
library(sentimentr) #for question 5
```

IPCC Report Twitter Last week we used the tidytext approach to sentiment analysis for Nexis Uni .pdf data on coverage of the recent IPCC report. This week we will look at the conversation on Twitter about the same report. We'll start with the familiar tidy approach, and then introduce the quanteda package later.

```
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'))


head(tweets$text, n = 10)
```

```
##  [1] "thank you, followers, for the great photo suggestions for our upcoming IPCC report - on Monday
##  [2] "Greenpeace: The real solution to the climate crisis will require a rapid transition away from
##  [3] "Governments have a responsibility to ensure that #IPCCReport is grounded in rapid phaseout of
not #FalseClimateSolutions. \n\nRead more in our open letter: https://t.co/4larBPgeba https://t.co/Fv1Op
##  [4] "Next week, the IPCC will publish a new report detailing their new models and policy pathways.
##  [5] "Live stream of virtual IPCC press conference releasing the report on mitigation of climate char
##  [6] "Attention journalists: The deadline for embargoed materials for the upcoming @IPCC_CH report or
##  [7] "The IPCC Report and "The Physics of Climate Change" https://t.co/xnxP3fup2a"
##  [8] "With time running short and most of the Summary for Policymakers yet to be approved, #IPCC Worl
##  [9] "A helpful perspective on how to talk about the scenarios discussed in the forthcoming IPCC repo
## [10] "The private sector is an integral component of the water cycle and has much to lose as critical
```

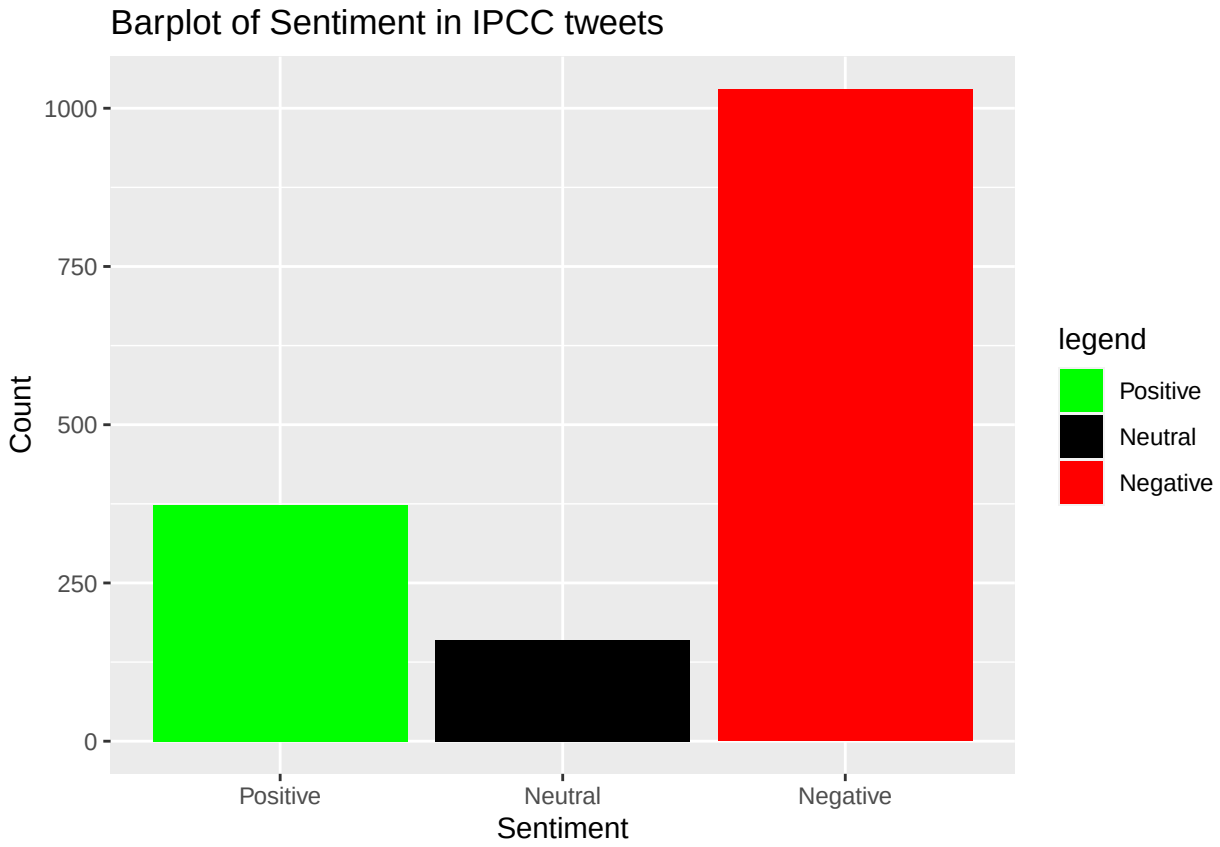
```
#simple plot of tweets per day
tweets %>%
  count(date) %>%
  ggplot(aes(x = date, y = n))+
  geom_line()
```



Question 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.

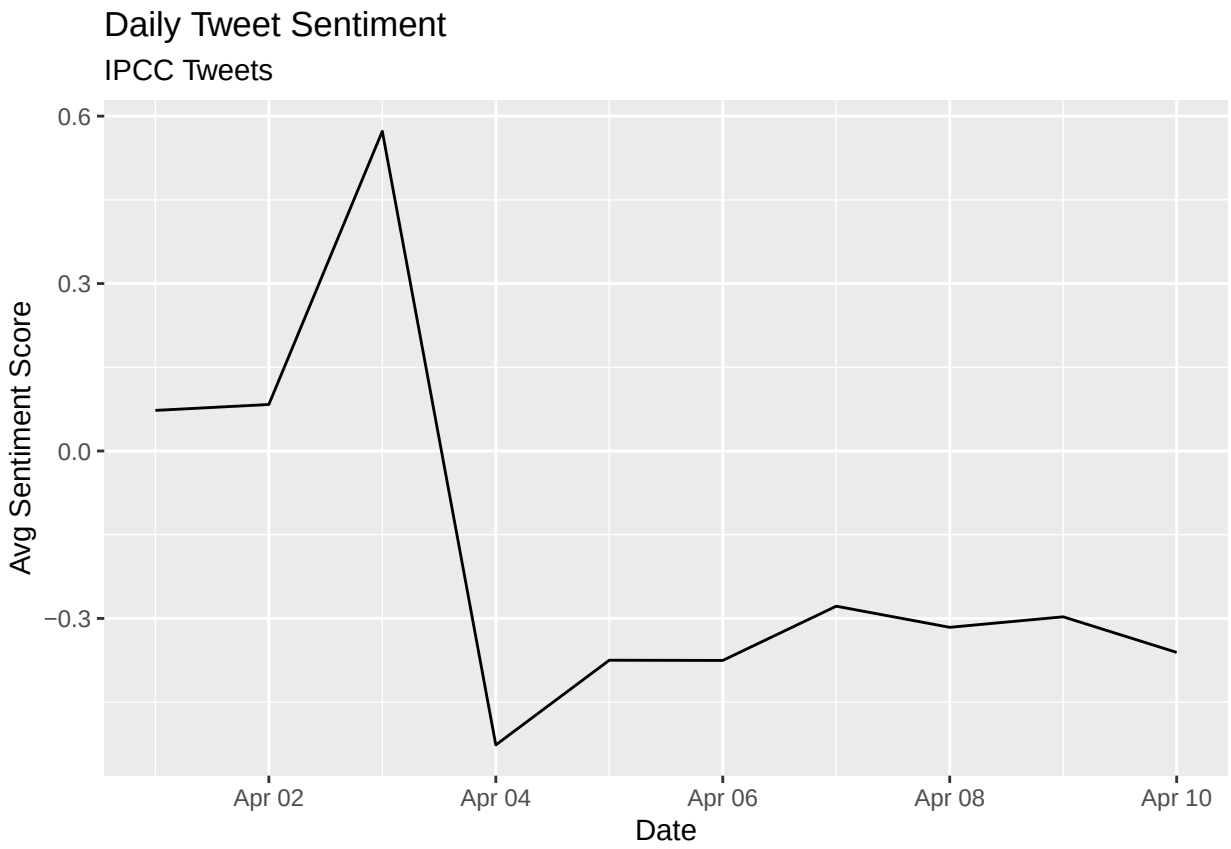
```
#let's clean up the URLs from the tweets
tweets$text <- gsub("http[^[:space:]]*", "",tweets$text)
tweets$text <- str_to_lower(tweets$text)

#Remove twitter account mentions from the text field of the tweets tibble.
tweets$text <- gsub("@[^[:space:]]*", "",tweets$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")
```

```r
#take average sentiment score by tweet
tweets_sent <- tweets %>%
  left_join(
    words %>%
      group_by(id) %>%
      summarize(
        sent_score = mean(sent_score, na.rm = T)),
    by = "id")

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

Sentiment <- c("Positive","Neutral","Negative")
Count <- c(positive,neutral,negative)
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")
```




```r
# tally sentiment score per day
daily_sent <- tweets_sent %>%
  group_by(date) %>%
  summarize(sent_score = mean(sent_score, na.rm = T))
```

```
daily_sent %>%
  ggplot( aes(x = date, y = sent_score)) +
  geom_line() +
    labs(x = "Date",
    y = "Avg Sentiment Score",
    title = "Daily Tweet Sentiment",
    subtitle = "IPCC Tweets")
```



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

Now let's try a new type of text visualization: the wordcloud.

```
words %>%
   anti_join(stop_words) %>%
   count(word) %>%
   with(wordcloud(word, n, max.words = 100))
```

```
## Joining, by = "word"
```

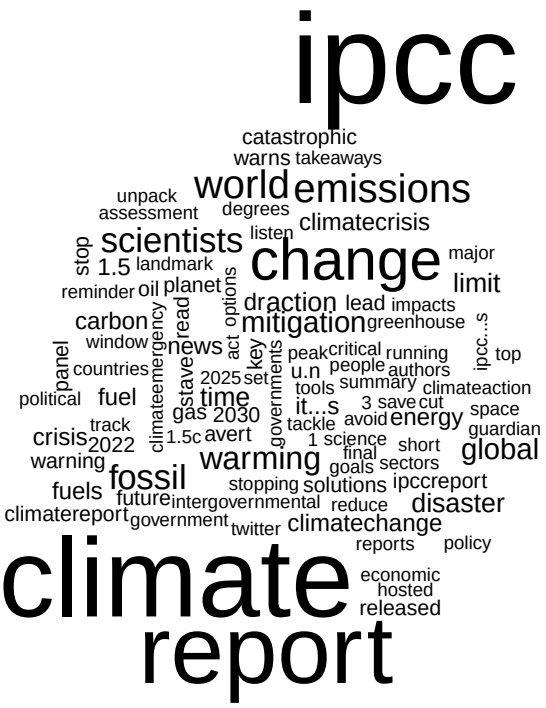
```
## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on 'it's'
## in 'mbcsToSbcs': dot substituted for <e2>
```

```
## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on 'it's'
## in 'mbcsToSbcs': dot substituted for <80>
```

```
## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on 'it's'
## in 'mbcsToSbcs': dot substituted for <99>
```

```
## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
```

```
## rotWord * : conversion failure on 'it’s' in 'mbcsToSbcs': dot substituted for
## <e2>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on 'it’s' in 'mbcsToSbcs': dot substituted for
## <80>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on 'it’s' in 'mbcsToSbcs': dot substituted for
## <99>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : font metrics unknown for Unicode character U+2019

## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## 'ipcc’s' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## 'ipcc’s' in 'mbcsToSbcs': dot substituted for <80>

## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## 'ipcc’s' in 'mbcsToSbcs': dot substituted for <99>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on 'ipcc’s' in 'mbcsToSbcs': dot substituted for
## <e2>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on 'ipcc’s' in 'mbcsToSbcs': dot substituted for
## <80>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on 'ipcc’s' in 'mbcsToSbcs': dot substituted for
## <99>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : font metrics unknown for Unicode character U+2019
```

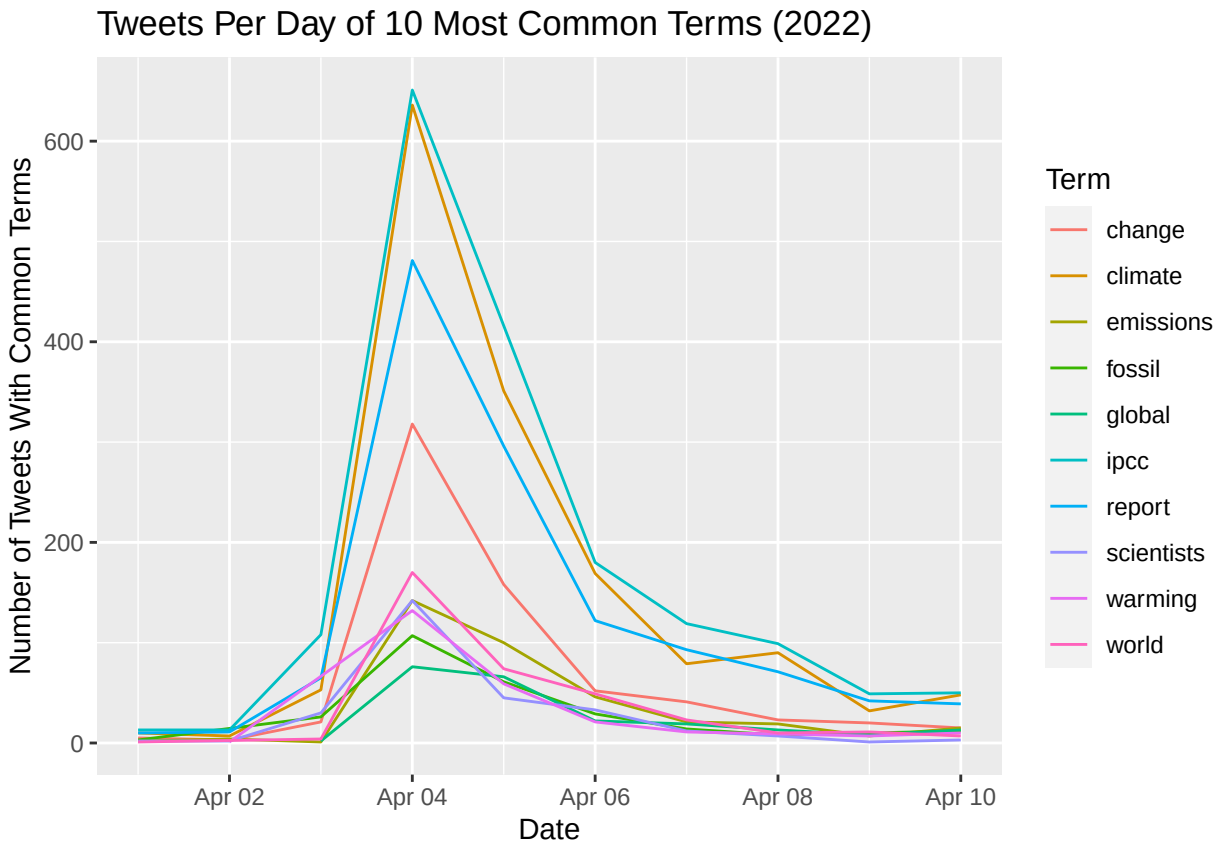



```
words %>%
inner_join(get_sentiments("bing")) %>%
count(word, sentiment, sort = TRUE) %>%
acast(word ~ sentiment, value.var = "n", fill = 0) %>%
comparison.cloud(colors = c("red", "green"),
                 max.words = 100)
```

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

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

```
#Word count
text_wordcounts <- words  %>% count(word, sort = TRUE)

#Get only the ten most common terms
text_wordcounts <- text_wordcounts[1:10, ]

#select terms in words df that match the 10 most common terms
common_terms <- words %>%
  subset(word %in% text_wordcounts$word)

# Group by date
common_terms_date <- common_terms %>%
  group_by(date) %>%
  count(word)

#Plot of ten most common terms in the tweets per day
ggplot(data = common_terms_date, aes(x = date, y = n)) +
  geom_line(aes(color = word)) +
  labs(title = "Tweets Per Day of 10 Most Common Terms (2022)",
       x = "Date",
       y = "Number of Tweets With Common Terms") +
  labs(color = "Term")
```



The thing I notice is that all of the most common words sem to be pretty straight forward and I could have guessed them. Id be interested to see what they are if you take out "ipcc", "report", and other words like that

The quanteda package quanteda is a package (actually a family of packages) full of tools for conducting text analysis. quanteda.sentiment (not yet on CRAN, download from github) is the quanteda modular package for conducting sentiment analysis.

quanteda has its own built in functions for cleaning text data. Let's take a look at some. First we have to clean the messy tweet data:

```
corpus <- corpus(dat$Title) #enter quanteda
summary(corpus)
```

```
## Corpus consisting of 2411 documents, showing 100 documents:
##
##     Text Types Tokens Sentences
##    text1    43     53         2
##    text2    37     42         2
##    text3    31     32         2
##    text4    42     49         3
##    text5    21     25         2
##    text6    30     33         1
##    text7    10     12         1
##    text8    40     42         2
##    text9    16     17         1
##   text10    36     42         2
##   text11    16     16         1
##   text12    34     44         6
```

```
##   text13    35     46         3
##   text14    46     52         2
##   text15    42     51         1
##   text16     7      7         1
##   text17    42     48         2
##   text18    17     17         2
##   text19    43     60         1
##   text20    27     34         3
##   text21    40     43         3
##   text22    44     50         3
##   text23    28     30         2
##   text24    35     38         3
##   text25    36     41         3
##   text26    37     43         4
##   text27    21     23         1
##   text28    29     31         1
##   text29    12     13         1
##   text30    45     47         2
##   text31    38     42         1
##   text32    31     36         1
##   text33    14     14         1
##   text34    41     49         1
##   text35     7      7         1
##   text36    44     54         2
##   text37    26     28         1
##   text38    13     13         1
##   text39    13     13         1
##   text40    31     37         2
##   text41    47     54         4
##   text42    38     46         1
##   text43    42     46         2
##   text44    22     24         2
##   text45    38     46         1
##   text46    16     16         1
##   text47    30     32         1
##   text48    17     17         1
##   text49    13     13         1
##   text50    23     23         1
##   text51    23     25         1
##   text52    25     27         1
##   text53    13     13         1
##   text54    34     35         3
##   text55    38     46         1
##   text56    38     46         1
##   text57    38     46         1
##   text58    38     46         1
##   text59    38     46         1
##   text60    38     46         1
##   text61    19     19         2
##   text62    17     18         1
##   text63    11     11         1
##   text64    13     13         1
##   text65    14     16         1
##   text66    12     12         2
```

```
##   text67    18     18         1
##   text68    38     46         1
##   text69    15     16         1
##   text70    12     13         1
##   text71    30     35         2
##   text72    22     23         1
##   text73    38     46         1
##   text74    39     46         1
##   text75    13     13         1
##   text76    32     35         1
##   text77    38     46         1
##   text78    39     45         2
##   text79    38     46         1
##   text80    36     41         1
##   text81    33     33         2
##   text82    18     19         1
##   text83    38     46         1
##   text84    38     46         1
##   text85    38     46         1
##   text86    39     43         2
##   text87    13     13         1
##   text88    13     13         1
##   text89    38     46         1
##   text90    38     46         1
##   text91    38     46         1
##   text92    40     43         1
##   text93    11     11         1
##   text94    41     49         1
##   text95    38     46         1
##   text96    15     15         1
##   text97    29     31         1
##   text98    11     11         1
##   text99    13     13         1
##  text100    38     46         1
```

```
tokens <- tokens(corpus) #tokenize the text so each doc (page, in this case) is a list of tokens (words

#examine the uncleaned version
tokens
```

```
## Tokens consisting of 2,411 documents.
## text1 :
##  [1] "thank"       "you"         ","           "followers"   ","
##  [6] "for"         "the"         "great"       "photo"       "suggestions"
## [11] "for"         "our"
## [ ... and 41 more ]
##
## text2 :
##  [1] "Greenpeace" ":"          "The"        "real"       "solution"
##  [6] "to"         "the"        "climate"    "crisis"     "will"
## [11] "require"    "a"
## [ ... and 30 more ]
##
## text3 :
##  [1] "Governments"    "have"           "a"              "responsibility"
```

```
##  [5] "to"             "ensure"         "that"           "#IPCCReport"
##  [9] "is"             "grounded"       "in"             "rapid"
## [ ... and 20 more ]
##
## text4 :
##  [1] "Next"      "week"      ","         "the"       "IPCC"      "will"
##  [7] "publish"   "a"         "new"       "report"    "detailing" "their"
## [ ... and 37 more ]
##
## text5 :
##  [1] "Live"       "stream"     "of"         "virtual"    "IPCC"
##  [6] "press"      "conference" "releasing"  "the"        "report"
## [11] "on"         "mitigation"
## [ ... and 13 more ]
##
## text6 :
##  [1] "Attention"   "journalists" ":"           "The"         "deadline"
##  [6] "for"         "embargoed"   "materials"   "for"         "the"
## [11] "upcoming"    "@IPCC_CH"
## [ ... and 21 more ]
##
## [ reached max_ndoc ... 2,405 more documents ]
```

```
#clean it up
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) #stem words down to their base form for comparisons across tense and

tokens <- tokens_tolower(tokens)
```

We can use the kwic function (keywords-in-context) to briefly examine the context in which certain words or patterns appear.

```
head(kwic(tokens, pattern = "climate", window = 3))
```

```
## Keyword-in-context with 6 matches.
##    [text2, 4]    greenpeace real solution | climate |
##   [text2, 17]        upcoming#ipcc report | climate |
##   [text5, 10] releasing report mitigation | climate |
##    [text6, 9]     upcoming@ipcc_ch report | climate |
##    [text7, 4]         ipcc report physics | climate |
##  [text10, 10]          much lose critical | climate |
##
##  crisis require rapid
##  solutions set publication
##  change a.m gmt
##  mitigation extended today
##  change https://t.co/xnxp3fup2a
##  water risks grow
```

```
head(kwic(tokens, pattern = phrase("climate change"), window = 3))
```

```
## Keyword-in-context with 6 matches.
```

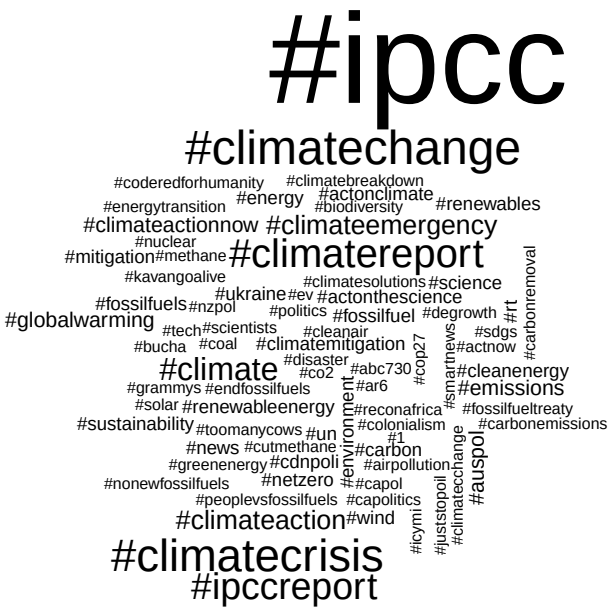
```
##   [text5, 10:11] releasing report mitigation | climate change |
##     [text7, 4:5]         ipcc report physics | climate change |
##  [text14, 15:16]         avert worst effects | climate change |
##  [text15, 10:11]   s#climatereport emissions | climate change |
##    [text20, 1:2]                             | climate change |
##    [text24, 6:7]      report revealed threat | climate change |
##
##  a.m gmt o
##  https://t.co/xnxp3fup2a
##  anyone think revolution
##  meenakshi raman@sahabatalammsia
##  want learn 100s
##  team weighed findings
```

Question 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.

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

dfm_hash<- dfm(hash_tweets)

tstat_freq <- textstat_frequency(dfm_hash, n = 100)
head(tstat_freq, 10)
```

```
##              feature frequency rank docfreq group
## 1              #ipcc       464    1     460   all
## 2     #climatechange       137    2     135   all
## 3     #climatecrisis       118    3     117   all
## 4     #climatereport        97    4      97   all
## 5        #ipccreport        87    5      87   all
## 6           #climate        68    6      67   all
## 7  #climateemergency        45    7      45   all
## 8     #climateaction        44    8      44   all
## 9     #globalwarming        24    9      24   all
## 10 #climateactionnow        23   10      23   all
```

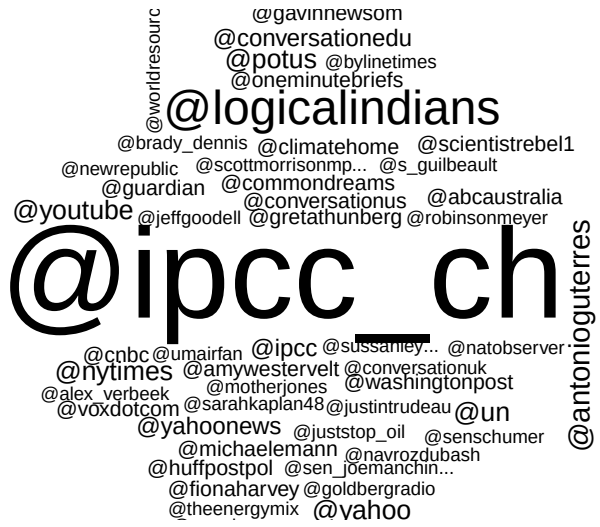
```
#tidytext gives us tools to convert to tidy from non-tidy formats
hash_tib<- tidy(dfm_hash)

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



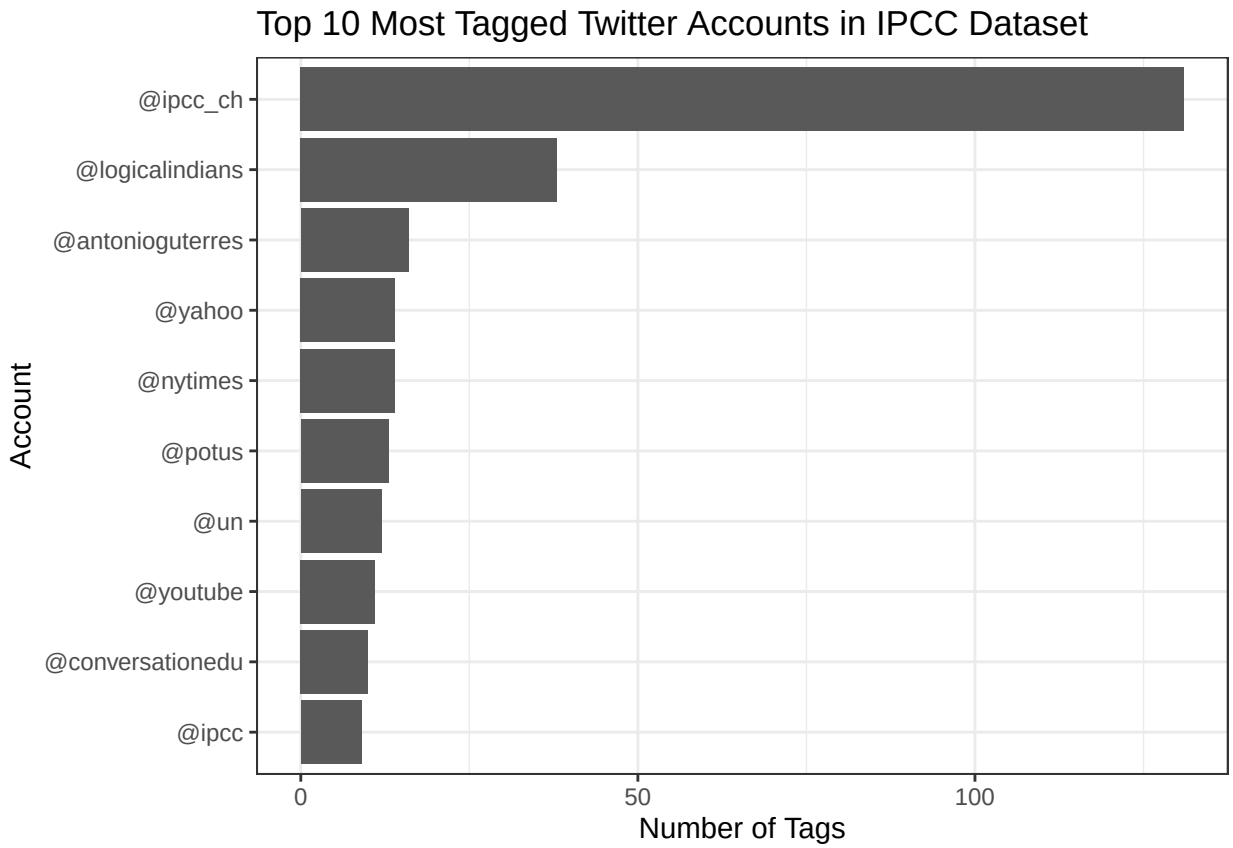

```
#this is where the hw answer starts
tagged_tweets <- tokens(corpus, remove_punct = TRUE) %>%
                 tokens_keep(pattern = "@*")
dfm_tagged<- dfm(tagged_tweets)
tstat_freq <- textstat_frequency(dfm_tagged, n = 100)
tstat_freq <- tstat_freq[1:10, ]
#tidytext gives us tools to convert to tidy from non-tidy formats
tagged_tib<- tidy(dfm_tagged)
tagged_tib %>%
   count(term) %>%
   with(wordcloud(term, n, max.words = 50))
```

```
## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## '@scottmorrisonmp' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## '@scottmorrisonmp' in 'mbcsToSbcs': dot substituted for <81>

## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## '@scottmorrisonmp' in 'mbcsToSbcs': dot substituted for <a9>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on '@scottmorrisonmp' in 'mbcsToSbcs': dot
## substituted for <e2>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on '@scottmorrisonmp' in 'mbcsToSbcs': dot
## substituted for <81>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on '@scottmorrisonmp' in 'mbcsToSbcs': dot
## substituted for <a9>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : font metrics unknown for Unicode character U+2069

## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## '@sussanley' in 'mbcsToSbcs': dot substituted for <e2>
```

```
## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## '@sussanley' in 'mbcsToSbcs': dot substituted for <81>

## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## '@sussanley' in 'mbcsToSbcs': dot substituted for <a9>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on '@sussanley' in 'mbcsToSbcs': dot substituted
## for <e2>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on '@sussanley' in 'mbcsToSbcs': dot substituted
## for <81>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on '@sussanley' in 'mbcsToSbcs': dot substituted
## for <a9>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : font metrics unknown for Unicode character U+2069

## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## '@sen_joemanchin' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## '@sen_joemanchin' in 'mbcsToSbcs': dot substituted for <81>

## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## '@sen_joemanchin' in 'mbcsToSbcs': dot substituted for <a9>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt
## = rotWord * : conversion failure on '@sen_joemanchin' in 'mbcsToSbcs': dot
## substituted for <e2>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt
## = rotWord * : conversion failure on '@sen_joemanchin' in 'mbcsToSbcs': dot
## substituted for <81>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt
## = rotWord * : conversion failure on '@sen_joemanchin' in 'mbcsToSbcs': dot
## substituted for <a9>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : font metrics unknown for Unicode character U+2069
```




```
# Plot Top 10 Most Tagged Twitter Accounts in IPCC Dataset
ggplot(data = tstat_freq, aes(y = reorder(feature, -rank), x = frequency)) +
  geom_bar(stat = "identity") +
  labs(title = "Top 10 Most Tagged Twitter Accounts in IPCC Dataset",
       x = "Number of Tags",
       y = "Account") +
  theme_bw()
```



Create the sparse matrix representation known as the document-feature matrix. quanteda's textstat_polarity function has multiple ways to combine polarity to a single score. The sent_logit value to fun argument is the log of (pos/neg) counts.

```
dfm <- dfm(tokens)

topfeatures(dfm, 12)
```

```
##    climate       ipcc     report     change        now      #ipcc      world
##       1396       1243       1225        651        505        464        346
##  emissions      never        new     latest scientists
##        333        291        279        279        274
```

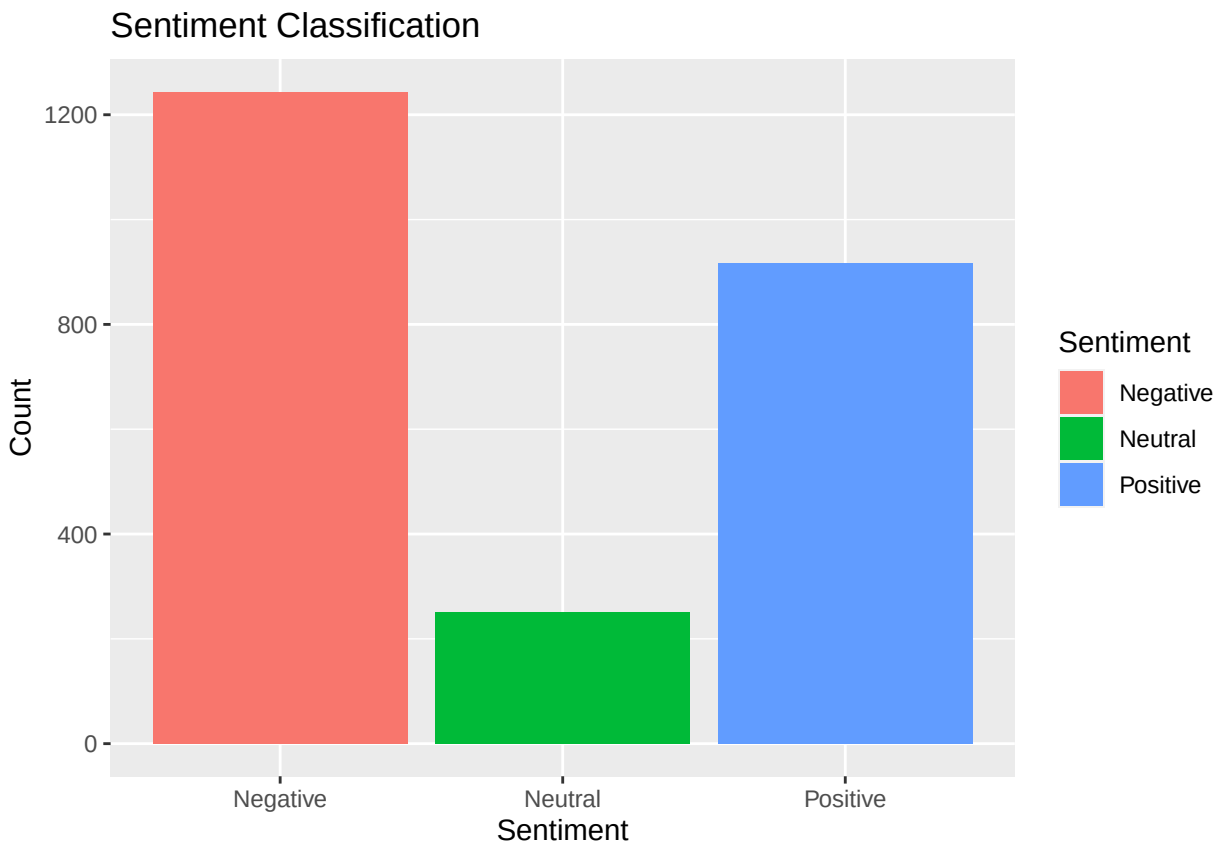
```
dfm.sentiment <- dfm_lookup(dfm, dictionary = data_dictionary_LSD2015)

head(textstat_polarity(tokens, data_dictionary_LSD2015, fun = sent_logit))
```

```
##   doc_id sentiment
## 1  text1  2.197225
## 2  text2 -1.098612
## 3  text3  1.945910
## 4  text4  0.000000
## 5  text5  1.098612
## 6  text6  1.098612
```

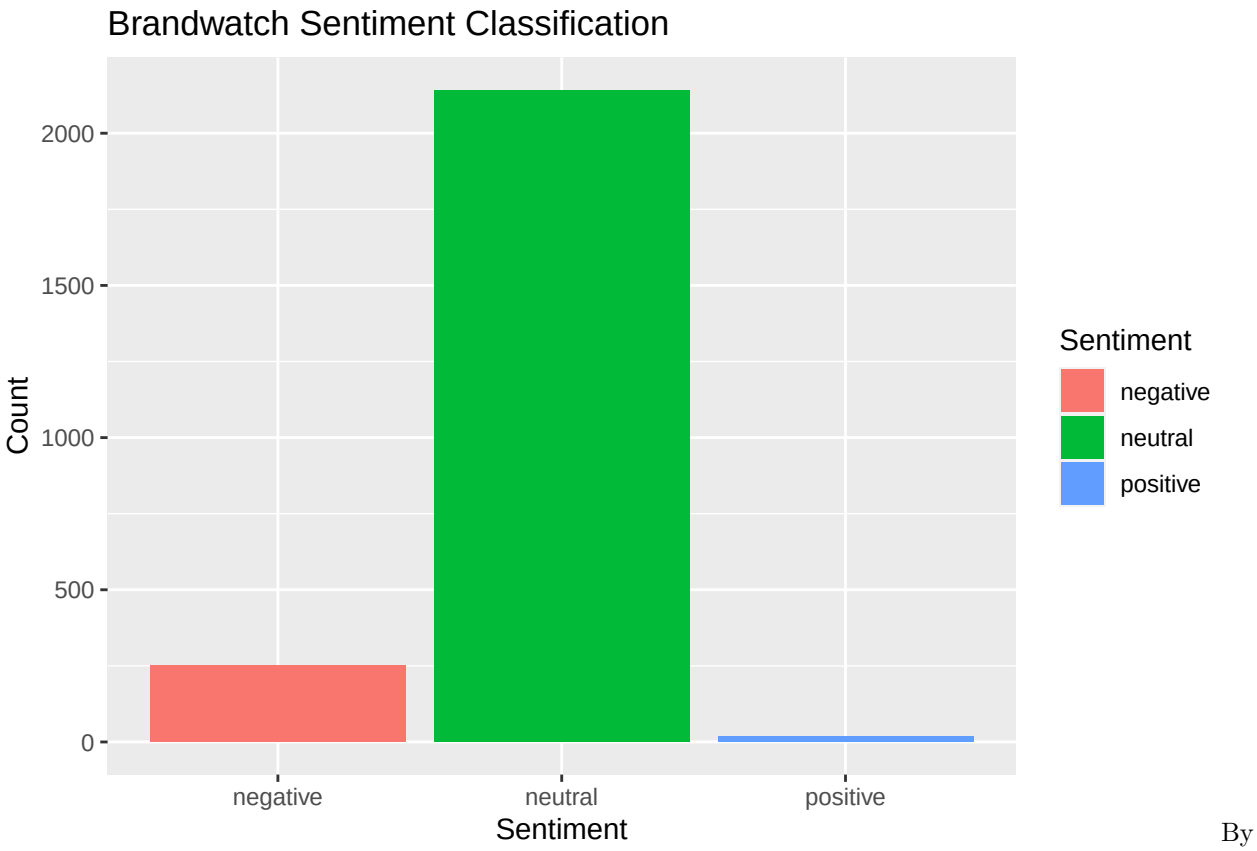
Question 5. The Twitter data download comes with a variable called "Sentiment" that must be calculated by Brandwatch. Use your own method to assign each tweet a polarity score (Positive, Negative, Neutral) and compare your classification to Brandwatch's (hint: you'll need to revisit the "raw_tweets"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

```
#data cleaning and wrangling

data_text <- raw_tweets[,c(4,6)]

tweets_text <- tibble(text = data_text$Title,
                      id = seq(1:length(data_text$Title)),
                     date = as.Date(data_text$Date,'%m/%d/%y'))

tweets_text$text <- gsub("http[^[:space:]]*", "",tweets_text$text)
tweets_text$text <- str_to_lower(tweets_text$text)

tweets_text$text <- gsub("@[^[:space:]]*", "",tweets_text$text)


mytext <- get_sentences(tweets_text$text) %>%
  sentiment() %>%
  rename(sentiment_score = sentiment) #uses the sentimentr package to get the sentiment based on polart

sentiment_assigned <- mytext %>%
  group_by(element_id) %>%
  summarise(sentiment_score = mean(sentiment_score))

sentiment_assigned$sentiment <- ifelse(sentiment_assigned$sentiment_score <0,
                                       "Negative",
                                       ifelse(sentiment_assigned$sentiment_score > 0, "Positive", "Neut



#Now we have to get the number of tweets with each sentiment type
raw_sentiment <- sentiment_assigned %>%
```

```
  group_by(sentiment) %>%
  summarise(sentiment_count = n())

#Plot the count per sentiment
ggplot(data = raw_sentiment, aes(x = Sentiment, y = sentiment_count)) +
        geom_bar(stat = "identity", aes(fill = Sentiment)) +
  labs(title = "Sentiment Classification",
        x= "Sentiment",
        y= "Count")
```




```
#Find the number of tweets with each sentiment type for Brandwatch
raw_sentiment <- raw_tweets %>%
  group_by(Sentiment) %>%
  summarise(sentiment_count = n())
#Plot the count per sentiment
ggplot(data = raw_sentiment, aes(x = Sentiment, y = sentiment_count)) +
        geom_bar(stat = "identity", aes(fill = Sentiment)) +
  labs(title = "Brandwatch Sentiment Classification",
        x= "Sentiment",
        y= "Count")
```



By doing it like this, and using the sentimentr package it is clear to see that there are more words assigned to either negative or positive when compared to the Brandwatch dataset. The overall trend is still the same, where its slightly more negative, however, there is a larger number of overall words in the negative and positive categories.