Topic 3 - Sentiment Analysis I

Alex Vand

4/13/2022

Overview

Sentiment analysis is a tool for assessing the mood of a piece of text. For example, we can use sentiment analysis to understand public perceptions of topics in environmental policy like energy, climate, and conservation.

```
library(tidyr) #text analysis in R
library(lubridate) #working with date data
library(pdftools) #read in pdfs
library(tidyverse)
library(tidytext)
library(here)
library(LexisNexisTools) #Nexis Uni data wrangling
library(sentimentr)
library(readr)
```

Intro to sentiment analysis example

For this introductory example, I selected a text excerpt from the National Book Award winning book, *The Overstory* by Richard Powers.

The excerpt is in .pdf format, so we'll need a tool (the pdftools package) to help us deal with that.

```
over <- pdf_text('overstory_exerpt.pdf')
over_df <- data.frame(text = over) %>% #create 1-column df with 'text' variable
  mutate(page = 1:n()) #add a page number variable, 'page'
#examine the beginning of the data frame
over_text <- over_df %>%
  filter(page %in% 8:41)%>%
  mutate(text = str_split(text, '\n')) %>% #this splits by page.
  unnest(text) %>%  #this splits by line
  mutate(line = str_to_lower(text)) #and convert to all lower case
#write_csv(over_text, "dat/over_text.csv")
#Note: \n, used above, is an example of an "escape sequence", which allow you to include characters tha
```

We'll start by using the Bing sentiment analysis lexicon.

```
bing_sent <- get_sentiments('bing') #grab the bing sentiment lexicon from tidytext
head(bing_sent, n = 20)
```

```
## # A tibble: 20 x 2
##    word          sentiment
##    <chr>         <chr>
##  1 2-faces       negative
##  2 abnormal      negative
##  3 abolish       negative
##  4 abominable    negative
##  5 abominably    negative
##  6 abominate     negative
##  7 abomination   negative
##  8 abort         negative
##  9 aborted       negative
## 10 aborts        negative
## 11 abound        positive
## 12 abounds       positive
## 13 abrade        negative
## 14 abrasive      negative
## 15 abrupt        negative
## 16 abruptly      negative
## 17 abscond       negative
## 18 absence       negative
## 19 absent-minded negative
## 20 absentee      negative
```

Here is the starting point for reading in the data as a .csv. We need to unnest the text to the word level so we can label the individual sentiment words. Let's also remove stop words as standard text cleaning procedure. Note: Not every English word is in the lexicons because many English words are pretty neutral.

```
over_text <- read_csv('over_text.csv')
#unnest to word-level tokens, remove stop words, and join sentiment words
 text_words <- over_text  %>%
  unnest_tokens(output = word, input = text, token = 'words')

 sent_words <- text_words%>% #break text into individual words
  anti_join(stop_words, by = 'word') %>% #returns only the rows without stop words
  inner_join(bing_sent, by = 'word') #joins and retains only sentiment words
```

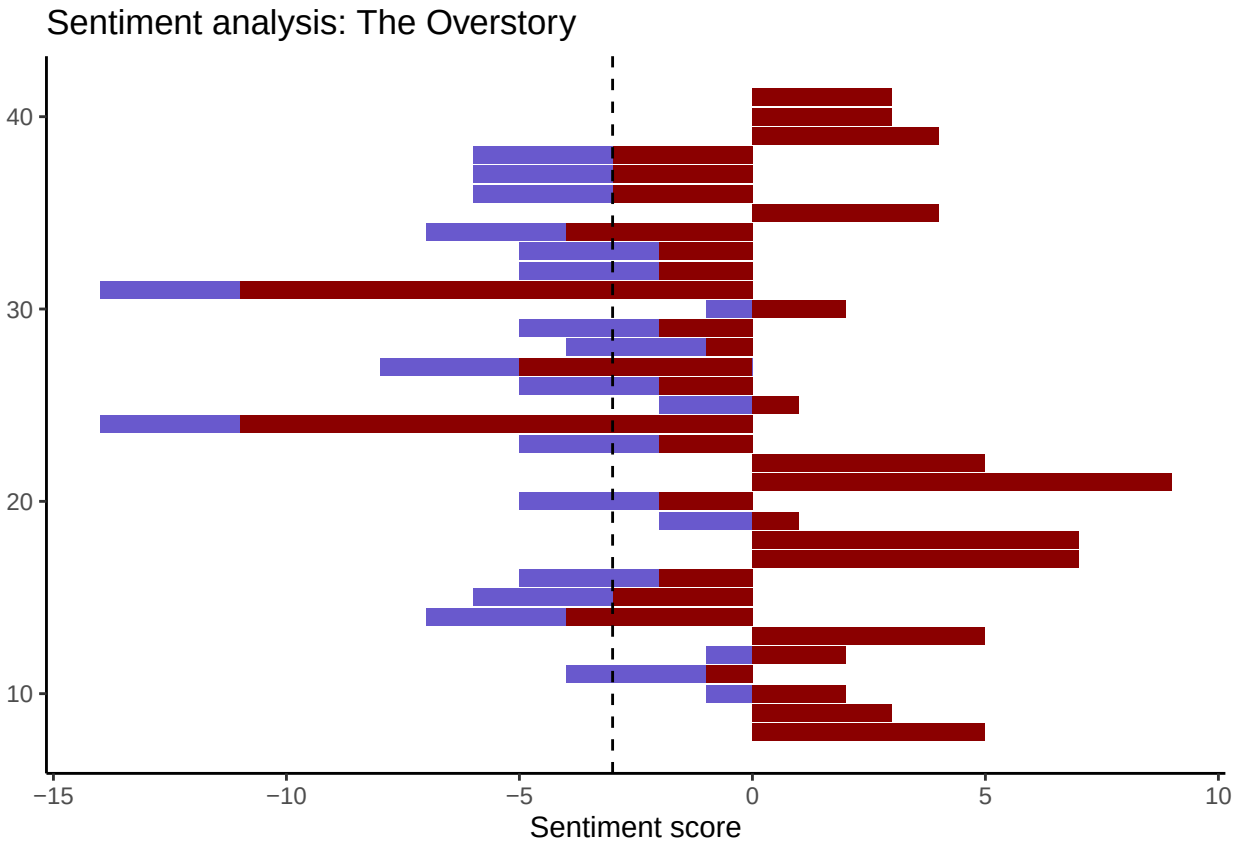
Create a sentiment score by counting the number of sentiment words occurring per page. We can center the scores around an offset point equal to the average page sentiment score. This lets us measure the sentiment of a given page relative to the overall sentiment of the book.

```
sent_scores <- sent_words %>%
  count(sentiment, page) %>%
  spread(sentiment, n) %>%
  mutate(raw_score = positive - negative, #single sentiment score per page
  offset = mean(positive - negative), #what is the average sentiment per page?
  offset_score = (positive - negative) - offset) %>% #how does this page's sentiment compare to that of
  arrange(desc(raw_score))
sent_scores
```

```
## # A tibble: 34 x 6
##     page negative positive raw_score offset offset_score
##    <dbl>    <int>    <int>     <int>  <dbl>        <dbl>
```

```
##  1    21        6       12         6     -3            9
##  2    17        4        8         4     -3            7
##  3    18        9       13         4     -3            7
##  4     8        1        3         2     -3            5
##  5    13        6        8         2     -3            5
##  6    22       10       12         2     -3            5
##  7    35       10       11         1     -3            4
##  8    39        4        5         1     -3            4
##  9     9        3        3         0     -3            3
## 10    40       14       14         0     -3            3
## # ... with 24 more rows
```

```
ggplot(sent_scores, aes(x = page)) +
  theme_classic() +
  geom_bar(aes(y = raw_score), stat = 'identity', fill = 'slateblue3') +
  geom_bar(aes(y = offset_score), stat = 'identity', fill = 'red4') +
  geom_hline(yintercept = sent_scores$offset[1], linetype = 'dashed', size = .5) +
  coord_flip() +
  theme(axis.title.y = element_blank()) +
  labs(title = 'Sentiment analysis: The Overstory',
       y = 'Sentiment score')
```



Origin of the NRC lexicon "These guys selected about 10,000 words from an existing thesaurus... and then created a set of five questions to ask about each word that would reveal the emotions and polarity associated with it. That's a total of over 50,000 questions.

They then asked these questions to over 2000 people, or Turkers, on Amazon's Mechanical Turk website, paying 4 cents for each set of properly answered questions.

The result is a comprehensive word-emotion lexicon for over 10,000 words."

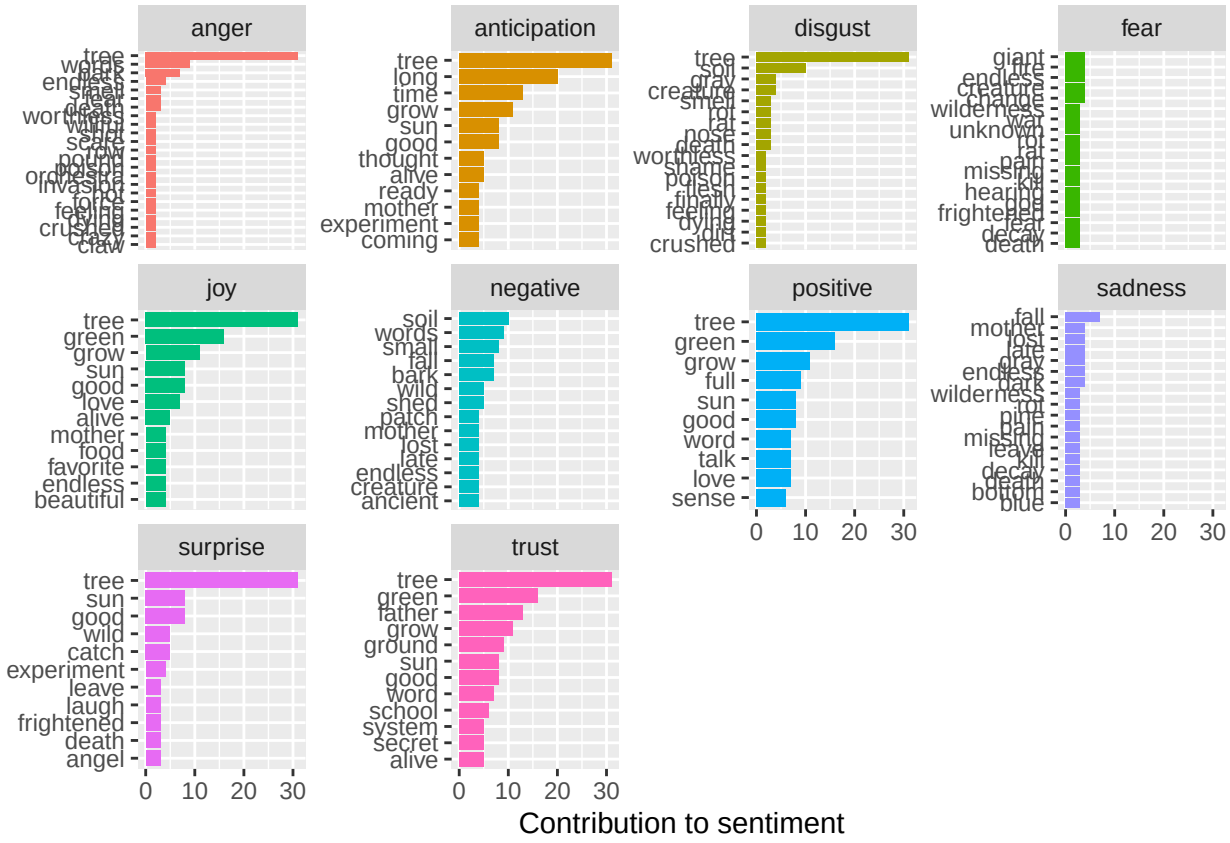
Let's take a look at the most common sentiment words in the data set

```
nrc_sent <- get_sentiments('nrc') #requires downloading a large dataset via prompt
nrc_fear <- get_sentiments("nrc") %>%
  filter(sentiment == "fear")
#most common words by sentiment
fear_words <- over_text  %>%
  unnest_tokens(output = word, input = text, token = 'words') %>%
  inner_join(nrc_fear) %>%
  count(word, sort = TRUE)
```

```
nrc_word_counts <- text_words %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

Let's break it out and plot the contributions by particular words different sentiment categories

```
book_sent_counts <- text_words %>%
        group_by(page) %>%
        # mutate(page_num = 1:n(),
        #        index = round(page_num / n(), 2)) %>%
        #unnest_tokens(word, line) %>%
        inner_join(get_sentiments("nrc")) %>%
        group_by(sentiment) %>%
        count(word, sentiment, sort = TRUE) %>%
        ungroup()
book_sent_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```



Introduction to the Nexis Uni data source

```
#setwd("/Users/mateorobbins/Desktop/Git/test/nexis_dat")
#to follow along with this example, download this .docx to your working directory:
#https://github.com/MaRo406/EDS_231-text-sentiment/blob/main/nexis_dat/Nexis_IPCC_Results.docx
my_files <- list.files(pattern = ".docx",
                       path = getwd(),
                       full.names = TRUE,
                       recursive = TRUE,
                       ignore.case = TRUE)
dat <- lnt_read(my_files) #Object of class 'LNT output'
meta_df <- dat@meta
articles_df <- dat@articles
paragraphs_df <- dat@paragraphs
dat2<- data_frame(element_id = seq(1:length(meta_df$Headline)),
                  Date = meta_df$Date,
                  Headline = meta_df$Headline)

#May be of use for assignment: using the full text from the articles
paragraphs_dat <- data_frame(element_id = paragraphs_df$Art_ID,
                             Text  = paragraphs_df$Paragraph)

dat3 <- inner_join(dat2,paragraphs_dat, by = "element_id")
```

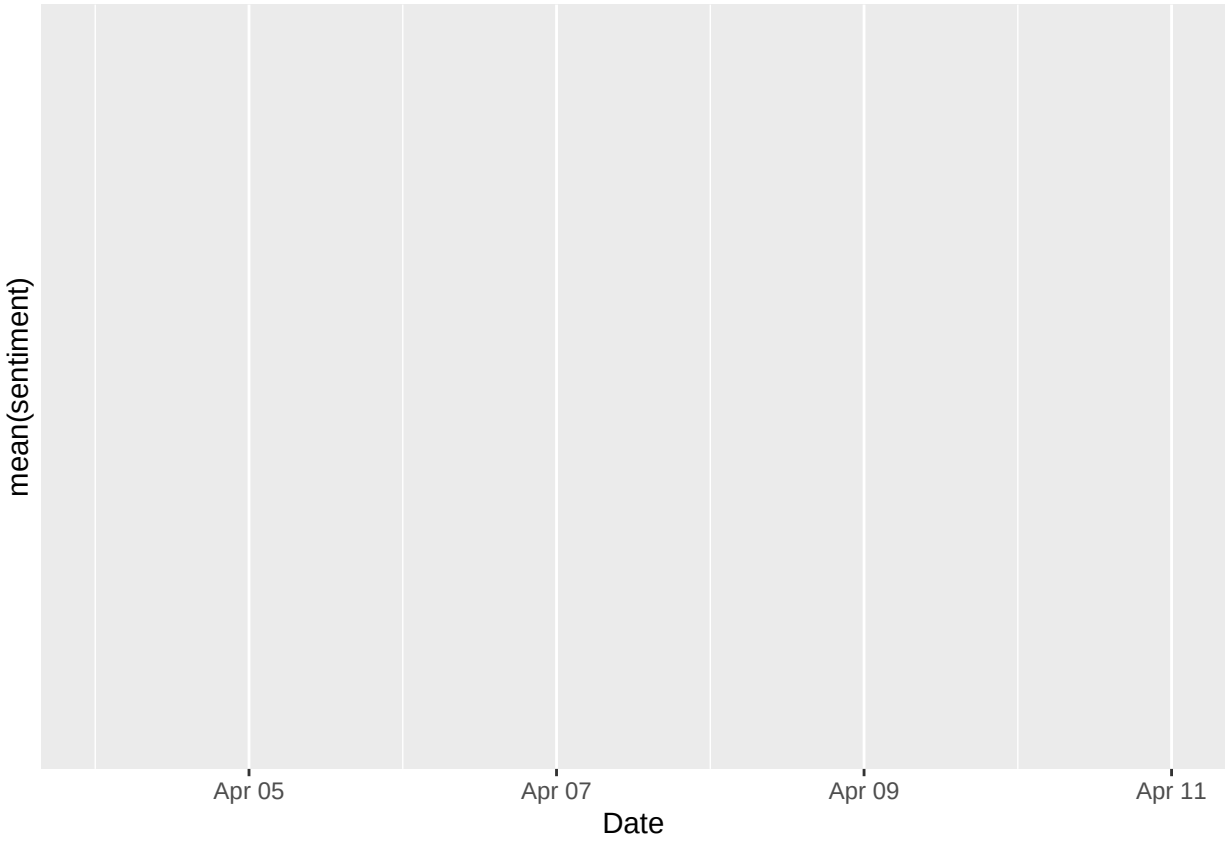
```
#can we create a similar graph to Figure 3A from Froelich et al.?
mytext <- get_sentences(dat2$Headline)
sent <- sentiment(mytext)
sent_df <- inner_join(dat2, sent, by = "element_id")
sentiment <- sentiment_by(sent_df$Headline)
sent_df %>%
  arrange(sentiment)
```

```
## # A tibble: 109 x 6
##    element_id Date       Headline               sentence_id word_count sentiment
##         <int> <date>     <chr>                        <int>      <int>     <dbl>
##  1         66 2022-04-04 Scientists risk arres~           1          7    -0.756
##  2         91 2022-04-07 The 'climate change' ~           1          9    -0.75
##  3         28 2022-04-09 The Dread 1.5 Degree ~           1          6    -0.714
##  4         43 2022-04-06 India's banks unprepa~           1          7    -0.510
##  5         34 2022-04-08 Dangerous radicals ar~           1          6    -0.449
##  6         14 2022-04-04 'Now or never' to avo~           1          8    -0.442
##  7         78 2022-04-07 Statewide Gas Ban Bil~           1         10    -0.427
##  8         50 2022-04-04 Guardian: Media 'Bare~           1          8    -0.407
##  9         62 2022-04-06 Governor Youngkin's I~           1         11    -0.377
## 10          7 2022-04-05 Narrow path to avoid ~           1          8    -0.354
## # ... with 99 more rows
```

```
sent_df$polarity <- ifelse(sent_df$sentiment <0, -1,
                      ifelse(sent_df$sentiment > 0, 1, 0))
```

```
#Froelich et al plot: mean sentiment by day
#summarize plot x = day, y = sentiment

ggplot(data = dat3, aes(x = Date,
                        y = mean(sentiment))) +
        geom_line()
```




```
custom_stop_words <- bind_rows(tibble(word = c("your_word"),
                                      lexicon = c("custom")),
                               stop_words)
```

Assignment (Due 4/19 by 5PM)

1. Access the Nexis Uni database through the UCSB library: https://www.library.ucsb.edu/research/db/211
2. Choose a key search term or terms to define a set of articles.
3. Use your search term along with appropriate filters to obtain and download a batch of around 100 full text search results (.docx).
4. Read your Nexis article document into RStudio.
5. This time use the full text of the articles for the analysis. First clean any artifacts of the data collection process (hint: this type of thing should be removed: "Apr 04, 2022 (Biofuels Digest: http://www.biofuelsdigest.com/ Delivered by Newstex"))
6. Explore your data a bit and try to replicate some of the analyses above presented in class if you'd like (not necessary).
7. Plot the amount of emotion words (the 8 from nrc) as a percentage of all the emotion words used each day (aggregate text from articles published on the same day). How does the distribution of emotion words change over time? Can you think of any reason this would be the case?