Assignment 2: Sentiment Analysis 1

Benjamin Moscona

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
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(pdftools) #read in pdfs
```

```
## Using poppler version 22.02.0
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v dplyr   1.0.7
## v tibble  3.1.6     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
## v purrr   0.3.4

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x lubridate::as.difftime() masks base::as.difftime()
## x lubridate::date()        masks base::date()
## x dplyr::filter()          masks stats::filter()
## x lubridate::intersect()   masks base::intersect()
## x dplyr::lag()             masks stats::lag()
## x lubridate::setdiff()     masks base::setdiff()
## x lubridate::union()       masks base::union()
```

```
library(tidytext)
library(here)
```

```
## here() starts at /Users/benjaminmoscona/Documents/eds231_textSent
```

```
library(LexisNexisTools) #Nexis Uni data wrangling
```

```
## LexisNexisTools Version 0.3.5
```

```
library(sentimentr)
library(readr)
library(corpus)
```

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
```

```
my_files <- list.files(pattern = ".docx", path = "Data/",
                       full.names = TRUE, recursive = TRUE, ignore.case = TRUE)

dat <- lnt_read(my_files) #Object of class 'LNT output'
```

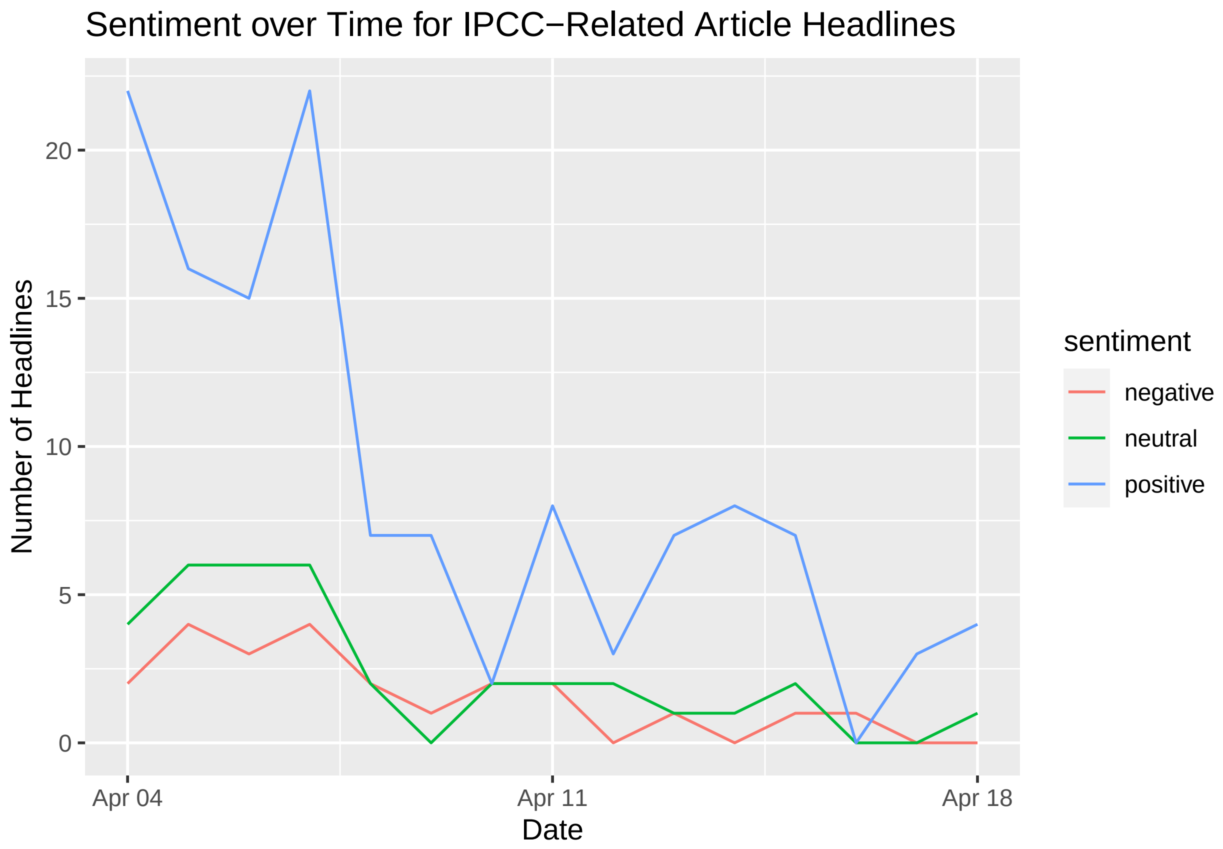
```
## Creating LNToutput from 5 files...

##  ...files loaded [2.26 secs]

##  ...articles split [2.56 secs]

##  ...lengths extracted [2.58 secs]

##  ...headlines extracted [2.58 secs]

##  ...newspapers extracted [2.59 secs]

##  ...dates extracted [2.68 secs]

##  ...authors extracted [2.69 secs]

##  ...sections extracted [2.70 secs]

##  ...editions extracted [2.70 secs]

## Warning in lnt_asDate(date.v, ...): More than one language was detected. The
## most likely one was chosen (English 87.8%)
##   ...dates converted [2.73 secs]

##   ...metadata extracted [2.74 secs]

##   ...article texts extracted [2.74 secs]

##   ...superfluous whitespace removed [2.92 secs]

## Elapsed time: 2.92 secs
```

```r
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_d
```

```
## Warning: `data_frame()` was deprecated in tibble 1.1.0.
## Please use `tibble()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```

```r
# 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)

# paragraphs_dat$Text <- text_tokens(paragraphs_dat$Text)


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

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

clean_tokens <- str_replace_all(dat3$Headline,"(.*)(((1[0-2]|0?[1-9])\\/(3[01]|[12][0-9]|0?[1-9])\\/(?:

dat3$Headline <- clean_tokens

text_words <- dat3  %>%
  unnest_tokens(output = word, input = Headline, 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


sent_scores <- sent_words %>%
  drop_na(Date) %>%
  count(sentiment, element_id, Date) %>%
  spread(sentiment, n) %>%
  replace_na(list(positive = 0, negative = 0)) %>%
  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 %>%
  mutate(positive = ifelse(offset_score >= 8, 1, 0),
         negative = ifelse(offset_score <= -8, 1, 0),
         neutral = ifelse(offset_score > -8 & offset_score < 8, 1, 0)) %>%
  group_by(Date) %>%
  summarize(positive = sum(positive),
            negative = sum(negative),
            neutral = sum(neutral)) %>%
  pivot_longer(-Date, names_to = "sentiment", values_to = "Number of Headlines") %>%
  ggplot(aes(Date, `Number of Headlines`, color = sentiment)) + geom_line() +
  labs(title = "Sentiment over Time for IPCC-Related Article Headlines")
```



```
#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 = "Data/Articles/",
                       full.names = TRUE, recursive = TRUE, ignore.case = TRUE)

dat <- lnt_read(my_files) #Object of class 'LNT output'
```

```
## Warning in lnt_asDate(date.v, ...): More than one language was detected. The
## most likely one was chosen (English 84.75%)
```

```
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$

# 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)
```

```r
# paragraphs_dat$Text <- text_tokens(paragraphs_dat$Text)


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

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

clean_tokens <- str_replace_all(dat3$Text,"(.*)(((1[0-2]|0?[1-9])\\/(3[01]|[12][0-9]|0?[1-9])\\/(?:[0-9

dat3$Text <- clean_tokens
```

```r
#can we create a similar graph to Figure 3A from Froelich et al.?

text_words <- dat3  %>%
  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


sent_scores <- sent_words %>%
  drop_na(Date) %>%
  count(sentiment, element_id, Date) %>%
  spread(sentiment, n) %>%
  replace_na(list(positive = 0, negative = 0)) %>%
  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: 335 x 7
##    element_id Date       negative positive raw_score offset offset_score
##         <int> <date>        <dbl>    <dbl>     <dbl>  <dbl>        <dbl>
##  1        306 2022-04-18      230      453       223   22.1         201.
##  2        258 2022-04-05      240      454       214   22.1         192.
##  3        323 2022-04-15       48      257       209   22.1         187.
##  4          1 2022-04-12       48      256       208   22.1         186.
##  5        218 2022-04-06      247      452       205   22.1         183.
##  6        168 2022-04-07      243      447       204   22.1         182.
##  7        324 2022-04-15       56      238       182   22.1         160.
##  8        167 2022-04-05       57      238       181   22.1         159.
##  9        256 2022-04-06       91      270       179   22.1         157.
## 10        291 2022-04-04       94      272       178   22.1         156.
## # ... with 325 more rows
```

```r
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 <- text_words  %>%
  inner_join(nrc_fear) %>%
  count(word, sort = TRUE)
```

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

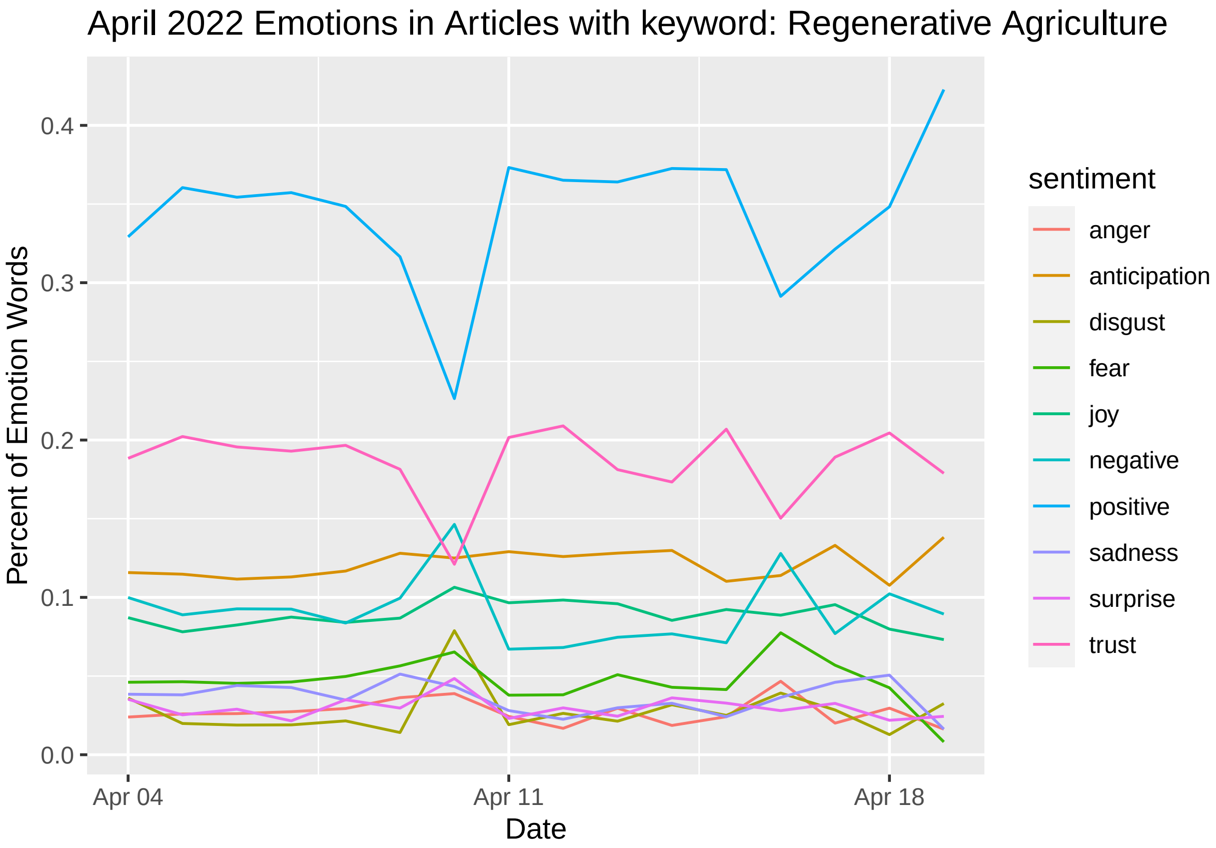
```
nrc_word_counts <- text_words %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

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

```
book_sent_counts <- text_words %>%
        drop_na(Date) %>%
        group_by(element_id, Date) %>%
        # mutate(page_num = 1:n(),
        #        index = round(page_num / n(), 2)) %>%
        #unnest_tokens(word, line) %>%
        inner_join(get_sentiments("nrc")) %>%
        group_by(sentiment, Date) %>%
        count(sentiment, sort = TRUE) %>%
        ungroup() %>%
  group_by(Date) %>%
  mutate(tot = sum(n),
         pct = n/tot)
```

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

```
book_sent_counts %>%
  ggplot(aes(Date, pct, color = sentiment)) + geom_line() +
  labs(y = "Percent of Emotion Words", title = "April 2022 Emotions in Articles with keyword: Regenerati
```



```
# book_sent_counts %>%
#   group_by(sentiment, Date) %>%
#   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)
```

Positive and negative as a percent of emotions run opposite of each other, which is reassurring from a robustness standpoint, even though the sentiment labels are not exclusive. Positivity still dominates the other sentiments. I would want to see how this changes over a longer period of time. In this graph, we have 500 articles in April 2022. I would love to use the NEXIS API to download the full set of 7000 articles over the past 5 years. Aroun April 11th, there was a large drop in positivity. I checked articles around this date and saw that it might have been driven down by a low earnings report from Ingredion, which mentions regenerative agriculture.