# Topic 3: Sentiment Analysis I

Halina Do-Linh

04/19/2022

Using the Intergovernmental Panel on Climate Change (IPCC) Nexis Uni dataset, I was able to recreate Figure 1A from *Public Perceptions of Aquaculture: Evaluating Spatiotemporal Patterns of Sentiment around the World*, Froehlich et al. First, I read in the data and created a data frame that contains the necessary data I need: `element_id`, `date`, and `headline`.

```
# downloaded data from eds 231 github
# don't have to specify file in here() bc list.files automatically looks
# for the files based on the pattern
ipcc_files <- list.files(pattern = ".docx",
                         path = here("hw/ipcc_data"),
                         full.names = TRUE,
                         recursive = TRUE,
                         ignore.case = TRUE)

ipcc_data <- lnt_read(ipcc_files) # class 'LNToutput'

# pull out necessary data
meta_ipcc <- ipcc_data@meta
articles_ipcc <- ipcc_data@articles
paragraphs_ipcc <- ipcc_data@paragraphs

# create tibble
ipcc_data2 <- tibble(element_id = seq(1:length(meta_ipcc$Headline)),
                     date = meta_ipcc$Date,
                     headline = meta_ipcc$Headline)
```

Next I obtain all the headline sentences using `get_sentences()`, which resulted in a total of 100 lists. I then use `sentiment()` to approximate a polarity score of each sentence ranging from -1 to 1, or negative to positive. I join the data frame of headlines I created in the code chunk above with the data frame of sentiment polarity scores. My joined data frame has a length of 109 because some headlines are made up of multiple sentences. `sentiment_by` provides a data frame that has estimated the sentiment grouped by `headline`.

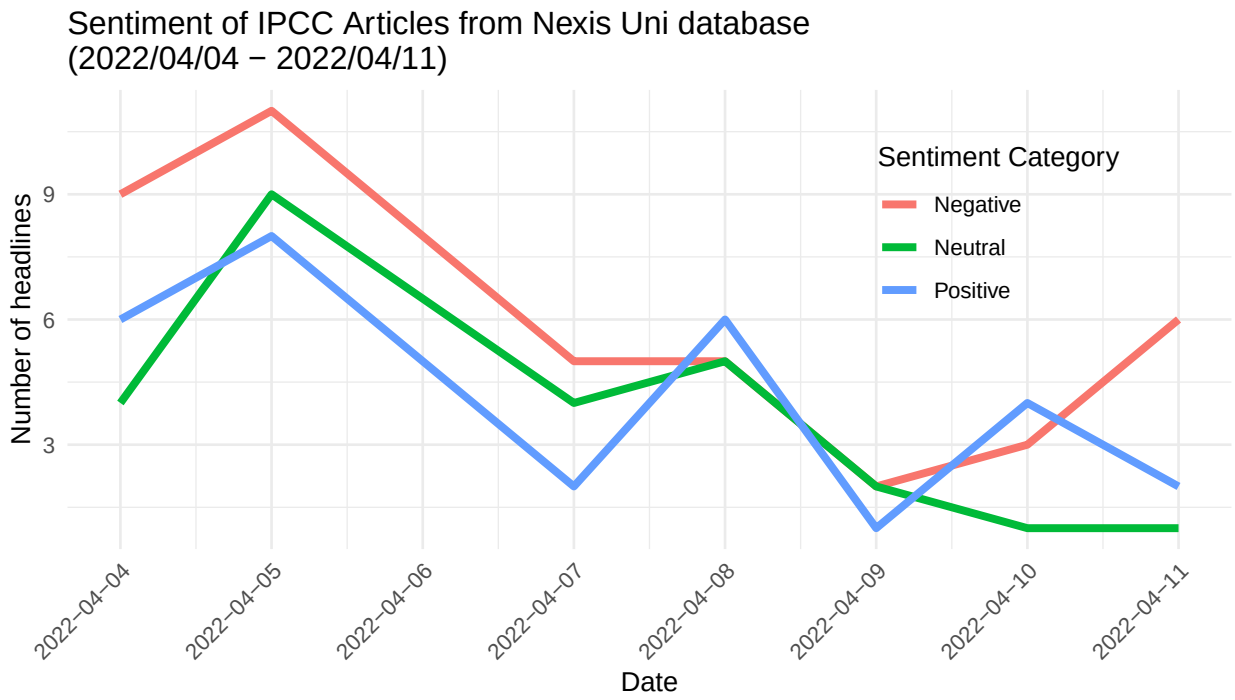
```
ipcc_text <- get_sentences(ipcc_data2$headline)
sent_ipcc <- sentiment(ipcc_text)

sent_ipcc_df <- inner_join(ipcc_data2, sent_ipcc, by = "element_id")

sentiment_ipcc <- sentiment_by(sent_ipcc_df$headline)

sent_ipcc_df %>% arrange(sentiment_ipcc)
```

Here I am assigning each polarity score a sentiment category of Positive, Neutral, or Negative. Then I grouped by date and sentiment category to find the total counts of each sentiment category by date. I need these categories to recreate the Froehlich et al plot.

```
sent_ipcc_aggregate <- sent_ipcc_df %>%
  mutate(sentiment_category = case_when(sentiment > 0 ~ "Positive",
                                        sentiment == 0 ~ "Neutral",
                                        sentiment < 0 ~ "Negative")) %>%
  group_by(date, sentiment_category) %>%
  summarize(count = n())
```

Now that I have my data in the right format, I can recreate the plot.

```
ggplot(data = sent_ipcc_aggregate,
       aes(x = date, y = count, color = sentiment_category)) +
  geom_line(size = 1.5) +
  theme_minimal() +
  theme(legend.position = c(0.8, 0.7),
        axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_x_date(date_labels = "%Y-%m-%d",
               breaks = unique(sent_ipcc_aggregate$date)) +
  labs(title = "Sentiment of IPCC Articles from Nexis Uni database \n(2022/04/04 - 2022/04/11)",
       x = "Date",
       y = "Number of headlines",
       color = "Sentiment Category")
```




From the Nexis Uni database, I chose the key search term **"carbon management"**.

```
my_files <- list.files(pattern = ".docx",
                       path = here("hw/data"),
```

```
                        full.names = TRUE,
                        recursive = TRUE,
                        ignore.case = TRUE)

# used exclude lines to remove unnecessary lines
data <- lnt_read(my_files,
                 exclude_lines = "^LOAD-DATE: |^UPDATE: |^GRAFIK: |^GRAPHIC: |^DATELINE:
                 |^Graphic |^CONTACT: |^Contact |^Email")
```

Here I am extracting the metadata, articles, and paragraphs from the LNT object and converting them into data frames. I had to create two separate data frames and then join the data together because the paragraph data did not have date or other metadata.

```
meta_df <- data@meta
articles_df <- data@articles
paragraphs_df <- data@paragraphs

# metadata like date and headline
data2 <- tibble(element_id = seq(1:length(meta_df$Headline)),
                date = meta_df$Date,
                headline = meta_df$Headline)

# text data from paragraphs
paragraphs_data <- tibble(element_id = paragraphs_df$Art_ID,
                          text = paragraphs_df$Paragraph)

# have to join because paragraphs_df doesn't contain date or other metadata
data3 <- inner_join(data2, paragraphs_data, by = "element_id")
```

Here I am doing some cleaning of the text (this feels like it could be an endless task).

```
data3_clean <- data3 %>%
  # remove short values i.e. link to story, visit us on social media, etc.
  filter(nchar(text) > 50) %>%
  # remove websites
  mutate(text = str_remove_all(string = text,
                               pattern = "\\S*.com\\S*")) %>%
  # remove email addresses
  mutate(text = str_remove_all(string = text,
                               pattern = "\\S*@\\S*")) %>%
  # remove dashes at the start of a string
  mutate(text = str_remove_all(string = text,
                                pattern = "^-")) %>%
  # remove short values after removing websites, addresses, etc
  filter(nchar(text) > 50)

# Note(HD): \\S* match any number of non-space characters
```

To finish cleaning the data, I create a data frame of stop words to remove from the text.

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

Now I can unnest the text to word-level tokens and remove stop words.

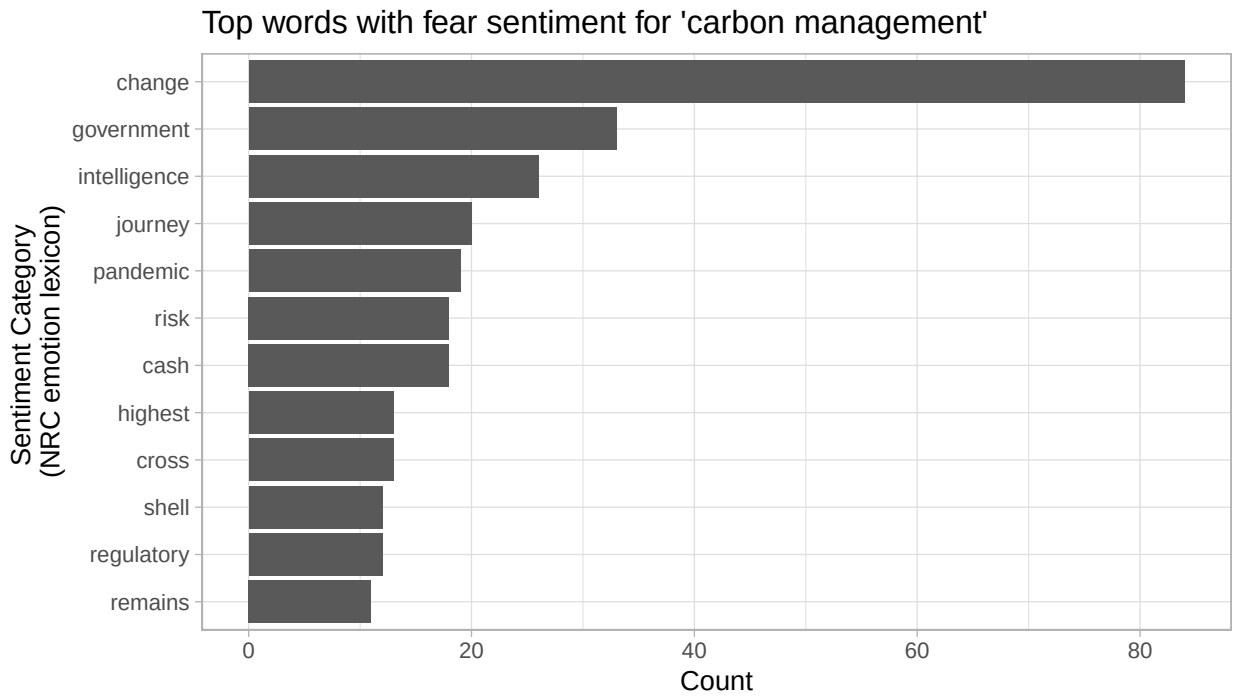
```
 text_words <- data3_clean  %>%
  unnest_tokens(output = word, input = text, token = 'words') %>%
  anti_join(custom_stop_words, by = "word")
```

To start my sentiment analysis, I am using the Bing sentiment analysis lexicon. I join the Bing lexicon with the cleaned and unnested data set to retain only sentiment words and a binary categorical sentiment value of positive or negative.

```
bing_sent <- get_sentiments('bing')

sent_words <- text_words %>% inner_join(bing_sent, by = 'word')
```

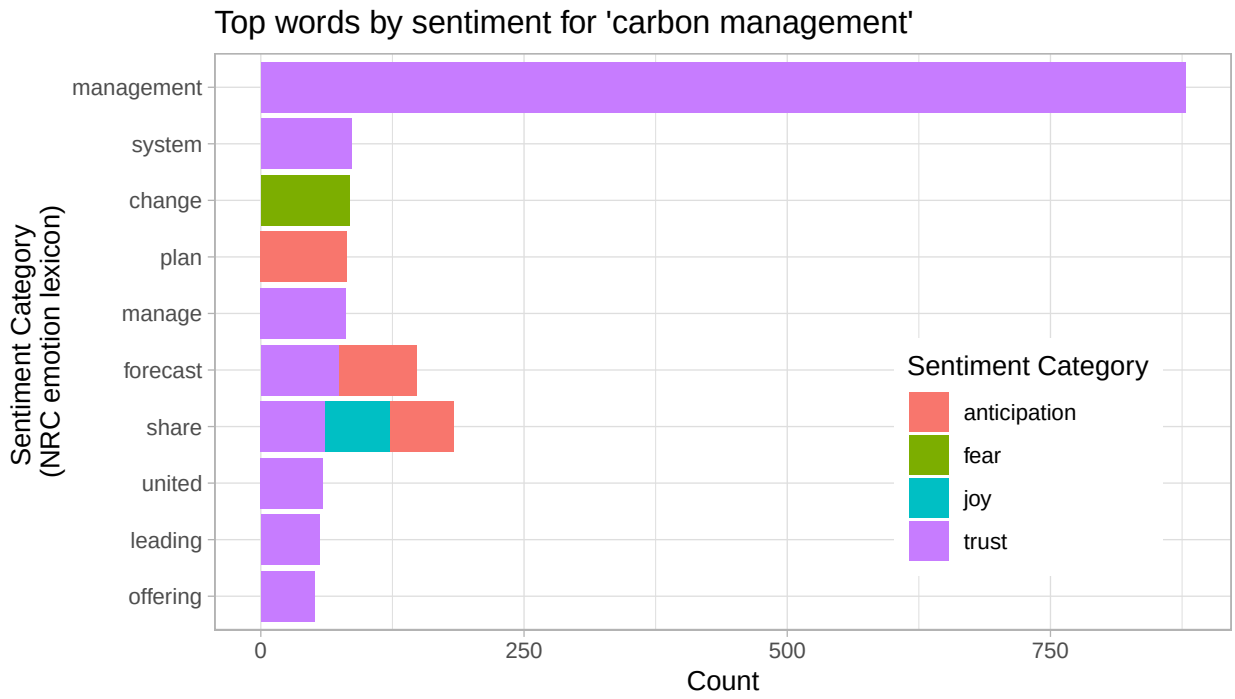
To obtain more specific sentiment values, I use the NRC emotion lexicon and filtered out the binary positive and negative sentiments. I did some exploratory analysis with the sentiment "fear" and found the top "fear" related words for carbon management. I plotted these results. I thought it was interesting that the top three words were change, government, and intelligence.

```
# obtaining nrc sentiments remove positive and negative
nrc_sent <- get_sentiments('nrc') %>%
  filter(!sentiment %in% c("positive", "negative"))

# exploratory analysis with fear sentiment
nrc_fear <- get_sentiments("nrc") %>%
  filter(sentiment == "fear")

# most common words for fear sentiment
fear_words <- data3_clean  %>%
  unnest_tokens(output = word, input = text, token = 'words') %>%
  inner_join(nrc_fear) %>%
  count(word, sort = TRUE) %>%
  filter(n > 10)

ggplot(data = fear_words, aes(x = reorder(word, n), y = n)) +
  geom_bar(stat = 'identity') +
  coord_flip() +
  theme_light() +
  labs(title = "Top words with fear sentiment for 'carbon management'",
       y = "Count",
       x = "Sentiment Category \n(NRC emotion lexicon)")
```




I also found the most common sentiment words in the data set and plotted these results. With more time, I would stem the word management and manage.

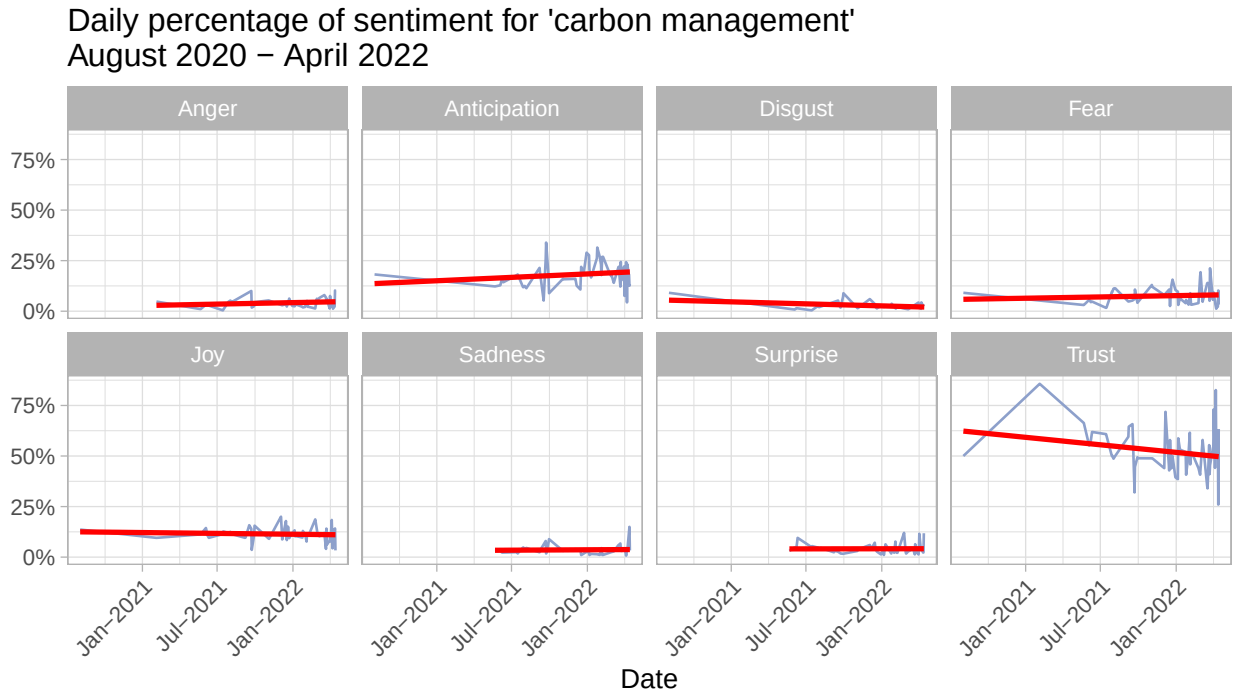
```
# most common words by sentiment
nrc_word_counts <- text_words %>%
  inner_join(nrc_sent) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup() %>%
  filter(n > 50)

ggplot(data = nrc_word_counts, aes(x = reorder(word, n), y = n, fill = sentiment)) +
  geom_bar(stat = 'identity') +
  coord_flip() +
  theme_light() +
  theme(legend.position = c(0.8, 0.3)) +
  labs(title = "Top words by sentiment for 'carbon management'",
       y = "Count",
       x = "Sentiment Category \n(NRC emotion lexicon)",
       fill = "Sentiment Category")
```




Lastly, I am going to plot the amount of emotion words (all eight from the NRC) as a percentage of all the emotion words used each day (aggregate text from articles published on the same day) to show how the emotion words change over time.

To do this I joined the NRC lexicon data frame with my unnested paragraph words and then calculate the total words per day and the percentage of all emotion words per day.

```
emotion_plot <- text_words %>%
  inner_join(nrc_sent, by = "word") %>%
  group_by(date) %>%
  count(sentiment) %>%
  mutate(total_words_day = sum(n)) %>%
  mutate(pct_sent_day = n / total_words_day)

ggplot(data = emotion_plot,
       aes(x = date,
           y = pct_sent_day)) +
  geom_line(color = "#8da0cb",
            size = .5) +
  geom_smooth(method = "lm", se = F, color = "red") +
  scale_x_date(date_labels = "%b-%Y") +
  scale_y_continuous(labels = scales::percent) +
  theme_light() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  facet_wrap(~ str_to_title(sentiment), ncol = 4) +
  labs(title = "Daily percentage of sentiment for 'carbon management' \nAugust 2020 - April 2022",
       x = "Date",
       y = NULL)
```




My plot shows that the emotions **trust** and **anticipation** are changing the most over time. In the past year and a half, trust has gone down, while anticipation has gone up. This is an interesting trend for both sentiments and without very much knowledge on carbon management, it aligns with our current reality. As the climate crisis ramps up, the technological advancements in carbon capture appear to be more of a fantasy than a reality. There has been slow progress for renewable energy products and production despite the increasing threats of climate change. And this juxtaposing scenario fuels distrust. Anticipation increasing over time is also believable because of the crucial need for climate solutions, and we are all impatience for these solutions to finally arrive and be implemented.