DSC595_601_SP23_A2_ktsmith4

Keenan Smith

2/10/23

Built with R Version 4.2.2

Guidance on R tidy text exploration provided by Silge, J & Robinson, D (2022). Text Mining with R (1st ed.). O'Reilly. Link to Textbook

```r
here::i_am("content/exploratory_data_analysis/DSC595_601_SP23_A2_ktsmith4.qmd")
library(here)
library(tidyverse)
library(data.table)
library(dtplyr)
library(dplyr, warn.conflicts = FALSE)

library(parallel)

all_cores <- parallel::detectCores(logical = FALSE)
cl <- parallel::makePSOCKcluster(all_cores)
doParallel::registerDoParallel(cl)
```

ID and Messy Data

Since my data was acquired prior to this project. I have chosen to describe the data collection process. Several cleaning methods utilizing REGEX were used actually to clean link data to ensure that only links with actual text data was collected. The actual data was scraped using the `rvest` library which is modelled on the famous `beautifulsoup` library in Python. The scrape was done with cleaning in mind so only HTML `<p>` tags with the CSS article tags were collected into some additional article metadata. Prior to the large scale web scrape, testing functions were created and tested on 4 articles for each source. This seemed to be a good eyeball check in my estimation as a human can only compare so much. This allowed for checking to make sure that no extraneous data was collected in accordance with the article text data. Also, the 4 test articles were randomly selected to help catch faults in the HTML documents across a sitemap. This served as a further check to ensure that the article text data was gathered.

During the web scrape, a `trycatch()` function was utilized so that the scrape could continue if an article incurred an error. This helped to ensure the integrity of the scraped data. Only data with the appropriate CSS and HTML tags were selected.

One of the questions in data processing is what is the quality of the data? In this case, all of the data were selected from reputable think tanks that are widely utilized to discuss potential policy gains in several areas of politics. The only news source is Jacobin which is a prominent leftist news source. This was added since truly left think tanks do not widely exist in an influential capacity in the United States.

One oddity that was actually found during my initial test modeling of the data is that I did not removed the names of the organizations from which I was scraping. This led to optimal models selecting "Heritage Foundation" as one of the influencing identifiers out of 400 ngrams. In this run through the data, I will ensure that this is fixed prior to modeling. It will stay in during exploratory data analysis since I find it interesting to see which organizations use their names frequently.

Cleaning the Data

The following sections show pulling the data into R and then processing it into its final product. The `rvest` ingests text data row by row. Though R possesses the ability to analyze text data through grouping, I find that with the article data it is better to have one row, one article. Through the following steps I bring the data in, join it, add a bias classifier, and create the full corpus. I try to leave comments where applicable.

Function Block

```
corpus_concatenation <- function(df) {
  df |>
    lazy_dt() |> # helpful translator from tibble to datatable
    group_by(art_link,
             art_date,
             art_author,
             art_topic,
             art_title,
             art_source) |> # These are the metadata tags REFACTOR in future
    summarise(full_text = paste(text, collapse = " "),
              .groups = "drop") |> # combining using the summarise function
    as_tibble() |> # from lazy dt back to tibble
    mutate(art_source = as.factor(art_source)) # changing column type
}
```

Loading Text Data

Data is read in and combined column wise by using the `data.table` library in R. `data.table` is a very fast library for working with dataframes. There are a lot of transitions between datatables and tibbles (the tidyverse dataframe) because though datatables are faster, the tidyverse library of tools is wider and I know more of them. I am holding 1.4 GB of data in RAM, I want fast operations on things that need to be fast and I want deliberate operations on things that need to be deliberate.

```
# Define Bias Sources
left_wing <- c("Jacobin", "Brookings Institute")
right_wing <- c("Heritage Commentary", "Heritage Report", "American Mind")

# Left Wing Read In
text_jacobin <- read_rds(here("data", "text_jacobin.rds")) |> as.data.table()
text_brookings <- read_rds(here("data", "text_brooking.rds")) |> as.data.table()
```

```r
# Right Wing Read In
text_am_mind_feature <- read_rds(here("data", "text_am_mind_feature.rds")) |> as.data.table()
text_am_mind_memo <- read_rds(here("data", "text_am_mind_memo.rds")) |> as.data.table()
text_am_mind_salvo <- read_rds(here("data", "text_am_mind_salvo.rds")) |> as.data.table()
text_heritage_com <- read_rds(here("data", "text_heritage_com.rds")) |> as.data.table()
text_heritage_rep <- read_rds(here("data", "text_heritage_rep.rds")) |> as.data.table()

# Remove Loop Iteration Column from Large Datasets
text_jacobin <- text_jacobin[, i := NULL]
text_brookings <- text_brookings[, i := NULL]
text_heritage_com <- text_heritage_com[, i := NULL]
text_heritage_rep <- text_heritage_rep[, i := NULL]

# Combine Text Datasets into Full Corpus
text_full_corpus <- rbind(text_jacobin, text_brookings, text_am_mind_feature,
                          text_am_mind_memo, text_am_mind_salvo,
                          text_heritage_com, text_heritage_rep) |>
  as_tibble()

# Combine Text into One Row, One Article
text_condensed_full_corpus <- corpus_concatenation(text_full_corpus)

# Removing Individual Corpus from Memory
rm(text_jacobin, text_brookings, text_am_mind_feature, text_am_mind_memo,
   text_am_mind_salvo, text_heritage_com, text_heritage_rep)

# Add Bias Column
text_condensed_full_corpus <- text_condensed_full_corpus |>
  mutate(art_bias = case_when(
    art_source %in% left_wing ~ "left-wing",
    art_source %in% right_wing ~ "right-wing"
  ),
  art_bias = as.factor(art_bias))
# Write to Disk
write_rds(text_condensed_full_corpus, here("data", "text_full_corpus.rds"),
          "gz", compression = 9L)
# Removing non-condensed corpus
rm(text_full_corpus)
```

Exploratory Data Analysis

A majority of these techniques are using the code from Text Mining with R.

```r
# Reading the data back in after processing
text_condensed_full_corpus <- read_rds(here("data", "text_full_corpus.rds"))

# Load Text Specific Libraries
```

```r
library(tidytext)
# for future work
# library(text)

# stop words from the tidytext package
data("stop_words")
```

```r
# Initial look at word token data
bias_words <- text_condensed_full_corpus |>
  unnest_tokens(word, full_text) |>
  anti_join(stop_words) |>
  count(art_bias, word, sort = TRUE)
```

```
Joining with `by = join_by(word)`
```

```r
total_words <- bias_words |>
  group_by(art_bias) |>
  summarise(total = sum(n))

bias_words <- left_join(bias_words, total_words)
```

```
Joining with `by = join_by(art_bias)`
```

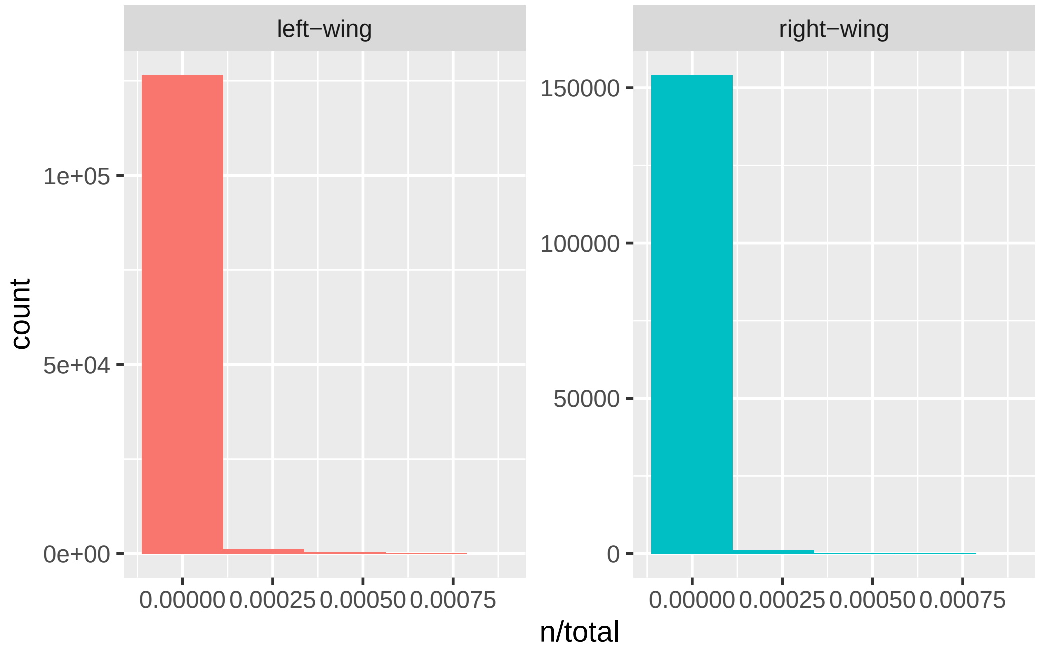
```r
head(bias_words, 25)
```

art_bias	word	n	total
right-wing	u.s	67890	9932377
right-wing	government	44784	9932377
right-wing	federal	39710	9932377
left-wing	people	39368	7919092
right-wing	percent	34360	9932377
left-wing	workers	31010	7919092
right-wing	congress	30361	9932377
left-wing	political	29096	7919092
right-wing	policy	28362	9932377
right-wing	tax	27453	9932377
right-wing	law	26869	9932377
right-wing	american	26798	9932377
right-wing	economic	25400	9932377
right-wing	united	25221	9932377
right-wing	security	24029	9932377
right-wing	national	23715	9932377
left-wing	time	23309	7919092
left-wing	government	23089	7919092
left-wing	public	22917	7919092

art_bias	word	n	total
right-wing	president	22611	9932377
right-wing	public	22591	9932377
left-wing	party	22416	7919092
right-wing	time	22403	9932377
right-wing	health	21897	9932377
right-wing	people	21835	9932377

Term Frequency

```
ggplot(bias_words, aes(n/total, fill = art_bias)) +
  geom_histogram(show.legend = FALSE, bins = 5) +
  xlim(NA, 0.0009) +
  facet_wrap(~art_bias, ncol = 2, scales = "free_y")
```



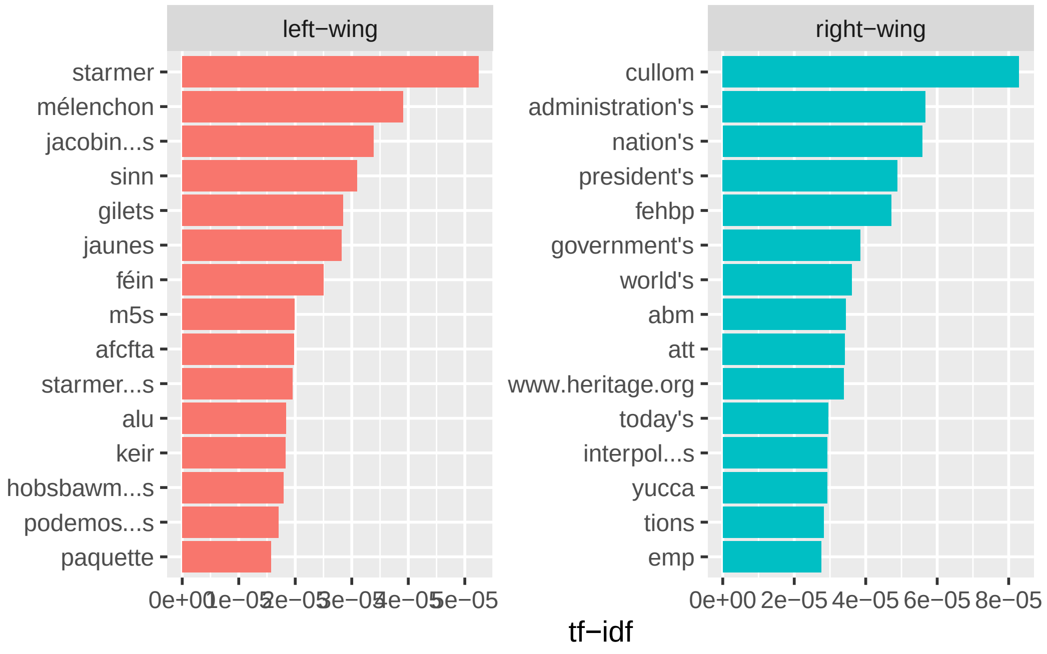
```
bias_tf_idf <- bias_words |>
  bind_tf_idf(word, art_bias, n)

bias_tf_idf |>
  select(-total) |>
  arrange(desc(tf_idf)) |>
  head(25)
```

art_bias	word	n	tf	idf	tf_idf
right-wing	cullom	1188	0.0001196	0.6931472	8.29e-05
right-wing	administration's	813	0.0000819	0.6931472	5.67e-05

art_bias	word	n	tf	idf	tf_idf
right-wing	nation’s	800	0.0000805	0.6931472	5.58e-05
left-wing	starmer	599	0.0000756	0.6931472	5.24e-05
right-wing	president’s	700	0.0000705	0.6931472	4.89e-05
right-wing	fehbp	675	0.0000680	0.6931472	4.71e-05
left-wing	mélenchon	446	0.0000563	0.6931472	3.90e-05
right-wing	government’s	551	0.0000555	0.6931472	3.85e-05
right-wing	world’s	517	0.0000521	0.6931472	3.61e-05
right-wing	abm	493	0.0000496	0.6931472	3.44e-05
right-wing	att	490	0.0000493	0.6931472	3.42e-05
left-wing	jacobin’s	387	0.0000489	0.6931472	3.39e-05
right-wing	www.heritage.org	485	0.0000488	0.6931472	3.38e-05
left-wing	sinn	353	0.0000446	0.6931472	3.09e-05
right-wing	today’s	423	0.0000426	0.6931472	2.95e-05
right-wing	interpol’s	419	0.0000422	0.6931472	2.92e-05
right-wing	yucca	418	0.0000421	0.6931472	2.92e-05
left-wing	gilets	325	0.0000410	0.6931472	2.84e-05
right-wing	tions	405	0.0000408	0.6931472	2.83e-05
left-wing	jaunes	322	0.0000407	0.6931472	2.82e-05
right-wing	emp	394	0.0000397	0.6931472	2.75e-05
right-wing	sogi	392	0.0000395	0.6931472	2.74e-05
right-wing	iran’s	388	0.0000391	0.6931472	2.71e-05
right-wing	people’s	382	0.0000385	0.6931472	2.67e-05
left-wing	féin	286	0.0000361	0.6931472	2.50e-05

```r
bias_tf_idf |>
  group_by(art_bias) |>
  slice_max(tf_idf, n = 15) |>
  ungroup() |>
  ggplot(aes(tf_idf, forcats::fct_reorder(word, tf_idf), fill = art_bias)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~art_bias, ncol = 2, scales = "free") +
  labs(x = "tf-idf", y = NULL)
```



After looking at the initial word token tf-idf data, we can see that there are alot of proper nouns and use of the thinktank or websites name. There are some interesting things to be seen in the initial data. Since this is a helpful technique for language classification, I plan on using this as the initial modeling vectorization for my non-deep learning modeling.

This tf-idf chart also shows that their is a global nature to these proper nouns. This may be due to the nature of the sources selected and the amount of the corpus that is contributing to each side. I am currently working to expand the corpus to include more American centric think tanks. This will hopefully dull some of the global nature of the proper nouns used. However, this could lead to *different* questions like "do left wing sites choose to cover more international events than right wing?"

Fortunately, these data don't seem to have high valuation for numbers. This is different from my experience using the `NLTK` framework in Python on my initial look. However, I did my exploratory data analysis post modeling and under different conditions aka the deliverable was different.

n-grams

```
bias_bigrams <- text_condensed_full_corpus |>
  unnest_tokens(bigram, full_text, token = "ngrams", n = 2) |>
  filter(!is.na(bigram))

bigrams_separated <- bias_bigrams |>
  lazy_dt() |>
  separate(bigram, c("word1", "word2"), sep = " ")

bigrams_filtered <- bigrams_separated |>
  filter(!word1 %in% stop_words$word) |>
  filter(!word2 %in% stop_words$word)
```

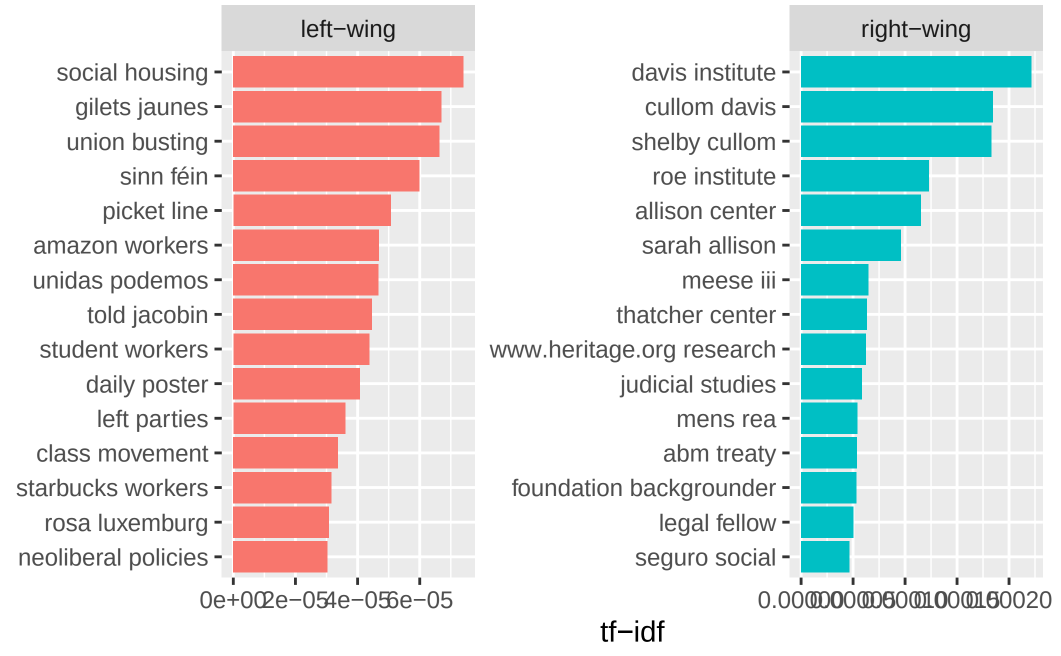
```r
bigrams_united <- bigrams_filtered |>
  mutate(bigram = paste(word1, word2, sep = " ")) |>
  as_tibble()
```

```r
bigram_tf_idf <- bigrams_united |>
  count(art_bias, bigram) |>
  bind_tf_idf(bigram, art_bias, n) |>
  arrange(desc(tf_idf))

head(bigram_tf_idf, 25)
```

art_bias	bigram	n	tf	idf	tf_idf
right-wing	davis institute	1427	0.0003196	0.6931472	0.0002215
right-wing	cullom davis	1188	0.0002660	0.6931472	0.0001844
right-wing	shelby cullom	1179	0.0002640	0.6931472	0.0001830
right-wing	roe institute	792	0.0001774	0.6931472	0.0001229
right-wing	allison center	741	0.0001659	0.6931472	0.0001150
right-wing	sarah allison	619	0.0001386	0.6931472	0.0000961
left-wing	social housing	354	0.0001069	0.6931472	0.0000741
left-wing	gilets jaunes	320	0.0000966	0.6931472	0.0000670
left-wing	union busting	317	0.0000957	0.6931472	0.0000664
right-wing	meese iii	418	0.0000936	0.6931472	0.0000649
right-wing	thatcher center	406	0.0000909	0.6931472	0.0000630
right-wing	www.heritage.org research	402	0.0000900	0.6931472	0.0000624
left-wing	sinn féin	286	0.0000864	0.6931472	0.0000599
right-wing	judicial studies	378	0.0000847	0.6931472	0.0000587
right-wing	mens rea	350	0.0000784	0.6931472	0.0000543
right-wing	abm treaty	344	0.0000770	0.6931472	0.0000534
right-wing	foundation backgrounder	342	0.0000766	0.6931472	0.0000531
left-wing	picket line	242	0.0000731	0.6931472	0.0000507
right-wing	legal fellow	324	0.0000726	0.6931472	0.0000503
left-wing	amazon workers	224	0.0000676	0.6931472	0.0000469
left-wing	unidas podemos	223	0.0000673	0.6931472	0.0000467
right-wing	seguro social	298	0.0000667	0.6931472	0.0000463
right-wing	yucca mountain	296	0.0000663	0.6931472	0.0000459
right-wing	gender dysphoria	288	0.0000645	0.6931472	0.0000447
left-wing	told jacobin	213	0.0000643	0.6931472	0.0000446

```r
bigram_tf_idf |>
  group_by(art_bias) |>
  slice_max(tf_idf, n = 15) |>
  ungroup() |>
  ggplot(aes(tf_idf, forcats::fct_reorder(bigram, tf_idf), fill = art_bias)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~art_bias, ncol = 2, scales = "free") +
  labs(x = "tf-idf", y = NULL)
```



n-grams characterize speech in phrases. In TF-IDF, this provides context. When a word token might capture one of the words in a proper noun or part of speech, n-grams allow us to see more of that context. This is a lighter approximation of word embedding, but where embedding and transformers are far more mathematically and theory based, n-gram analysis in this case is more based on intuition and subject matter expertise. Some of the bigrams present in the figure show that there are non-proper nouns that show up in the analysis. We could use these to draw some conclusions about what certain organizations like to talk about or what they don't like to talk about. It is certainly interesting that the Right-wing has a top 15 that is mainly proper nouns, whereas the left has more sayings or talking points. I don't want to draw too many conclusions however, because that is not the current scope of the question, the current is whether political speech can classify bias. Another interesting question would be whether or not one side spends time addressing the other side or vice versa.

NLP Technique

The NLP technique that I am using to solve my question is modeling via TF-idf vectorization. I think this suites my question well and I have already seen some promising initial tests prior to this class. My end goal of this capstone is to ultimately build a political lexicon by which to classify speech in the future.

As this is part of a larger capstone project, I am currently adding and expanding my corpus. I am collecting links and will be scraping for more data soon. With the size of the current corpus, I may need to look at new data management techniques to speed the process along for exploratory data analysis. Luckily, with `dtplyr` I can do many of the data manipulation steps neccessary, but I may need to look at additional parallel processing to help.

I will also be expanding this out to look at deep learning techniques and embeddings. I already have some code which I can use to create my own embeddings specific to this corpus. I plan on looking at a bit more vectorization techniques prior to actual modeling.