Topic 5 Word Relationships

Charles Hendrickson

5/3/2022

```
library(tidyr) #text analysis in R
library(pdftools)
library(lubridate) #working with date data
library(tidyverse)
library(tidytext)
library(readr)
library(quanteda)
library(readtext) #quanteda subpackage for reading pdf
library(quanteda.textstats)
library(quanteda.textplots)
library(ggplot2)
library(forcats)
library(stringr)
library(quanteda.textplots)
library(widyr)# pairwise correlations
library(igraph) #network plots
library(ggraph)
library(here)
```

#import EPA EJ Data

```
files <- list.files(path = here("data/"), pattern = "*pdf$", full.names = TRUE)

ej_reports <- lapply(files, pdf_text)

ej_pdf <- readtext(file = here("data/", "*.pdf"),
                   docvarsfrom = "filenames",
                   docvarnames = c("type", "year"),
                   sep = "_")
#creating an initial corpus containing our data
epa_corp <- corpus(x = ej_pdf, text_field = "text" )
summary(epa_corp)
```

```
## Corpus consisting of 6 documents, showing 6 documents:
##
##             Text Types Tokens Sentences type year docvar3
##  EPA_EJ_2015.pdf  2136   8944       263  EPA   EJ    2015
##  EPA_EJ_2016.pdf  1599   7965       176  EPA   EJ    2016
##  EPA_EJ_2017.pdf  2774  16658       447  EPA   EJ    2017
##  EPA_EJ_2018.pdf  3973  30564       653  EPA   EJ    2018
##  EPA_EJ_2019.pdf  3773  22648       672  EPA   EJ    2019
##  EPA_EJ_2020.pdf  4493  30523       987  EPA   EJ    2020
```

```
#I'm adding some additional, context-specific stop words to stop word lexicon
more_stops <-c("2015","2016", "2017", "2018", "2019", "2020", "www.epa.gov", "https")
add_stops<- tibble(word = c(stop_words$word, more_stops))
stop_vec <- as_vector(add_stops)
```

Now we'll create some different data objects that will set us up for the subsequent analyses

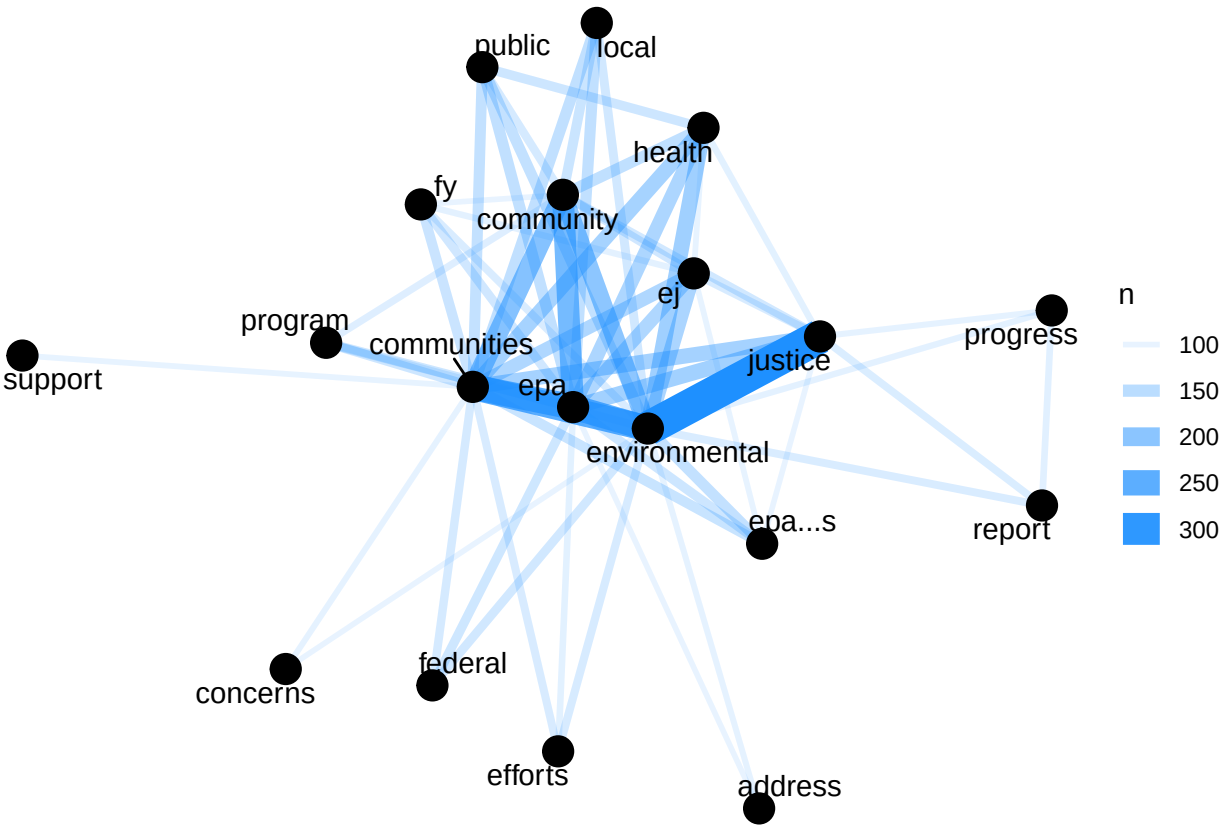
```
#convert to tidy format and apply my stop words
raw_text <- tidy(epa_corp)
#Distribution of most frequent words across documents
raw_words <- raw_text %>%
  mutate(year = as.factor(year)) %>%
  unnest_tokens(word, text) %>%
  anti_join(add_stops, by = 'word') %>%
  count(year, word, sort = TRUE)
#number of total words by document
total_words <- raw_words %>%
  group_by(year) %>%
  summarize(total = sum(n))
report_words <- left_join(raw_words, total_words)

par_tokens <- unnest_tokens(raw_text, output = paragraphs, input = text, token = "paragraphs")
par_tokens <- par_tokens %>%
 mutate(par_id = 1:n())
par_words <- unnest_tokens(par_tokens, output = word, input = paragraphs, token = "words")
```

Let's see which words tend to occur close together in the text. This is a way to leverage word relationships (in this case, co-occurence in a single paragraph) to give us some understanding of the things discussed in the documents.

```
word_pairs <- par_words %>%
  pairwise_count(word, par_id, sort = TRUE, upper = FALSE) %>%
  anti_join(add_stops, by = c("item1" = "word")) %>%
  anti_join(add_stops, by = c("item2" = "word"))
```

Now we can visualize

```
word_pairs %>%
  filter(n >= 100) %>%
  graph_from_data_frame() %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = n, edge_width = n), edge_colour = "dodgerblue") +
  geom_node_point(size = 5) +
  geom_node_text(aes(label = name), repel = TRUE,
                 point.padding = unit(0.2, "lines")) +
  theme_void()
```



Q2.

Choose a new focal term to replace "justice" and recreate the correlation table and network (see corr_paragraphs and corr_network chunks). Explore some of the plotting parameters in the cor_network chunk to see if you can improve the clarity or amount of information your plot conveys. Make sure to use a different color for the ties!

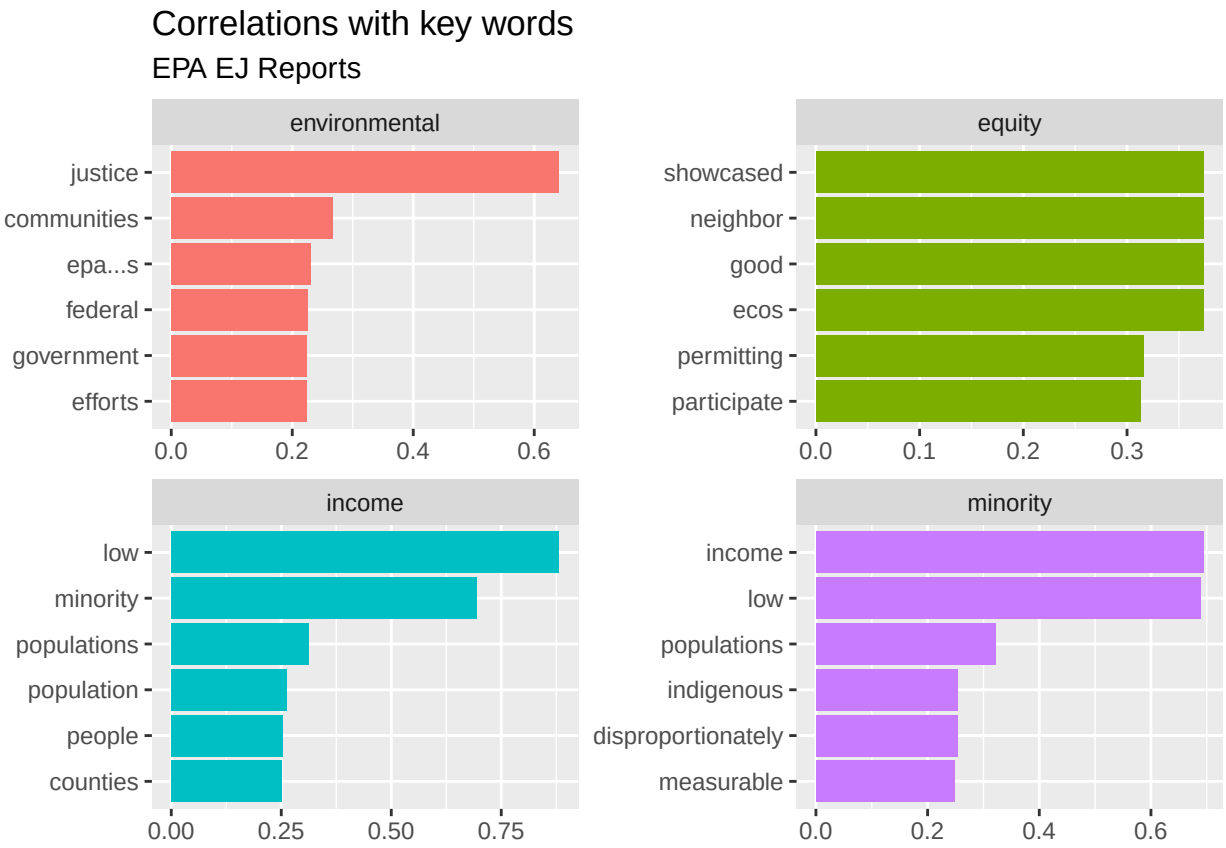
I chose to replace "justice" with "minority".

Hmm, interesting, but maybe we further subset the word pairs to get a cleaner picture of the most common ones by raising the cutoff for number of occurrences (n).

Pairs like "environmental" and "justice" are the most common co-occurring words, but that doesn't give us the full picture asthey're also the most common individual words. We can also look at correlation among words, which tells us how often they appear together relative to how often they appear separately.

```
word_cors <- par_words %>%
  add_count(par_id) %>%
  filter(n >= 50) %>%
  select(-n) %>%
  pairwise_cor(word, par_id, sort = TRUE)
just_cors <- word_cors %>%
  filter(item1 == "justice")
  word_cors %>%
  filter(item1 %in% c("environmental", "minority", "equity", "income"))%>%
  group_by(item1) %>%
  top_n(6) %>%
  ungroup() %>%
```

```
  mutate(item1 = as.factor(item1),
  name = reorder_within(item2, correlation, item1)) %>%
  ggplot(aes(y = name, x = correlation, fill = item1)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~item1, ncol = 2, scales = "free")+
  scale_y_reordered() +
  labs(y = NULL,
         x = NULL,
         title = "Correlations with key words",
         subtitle = "EPA EJ Reports")
```

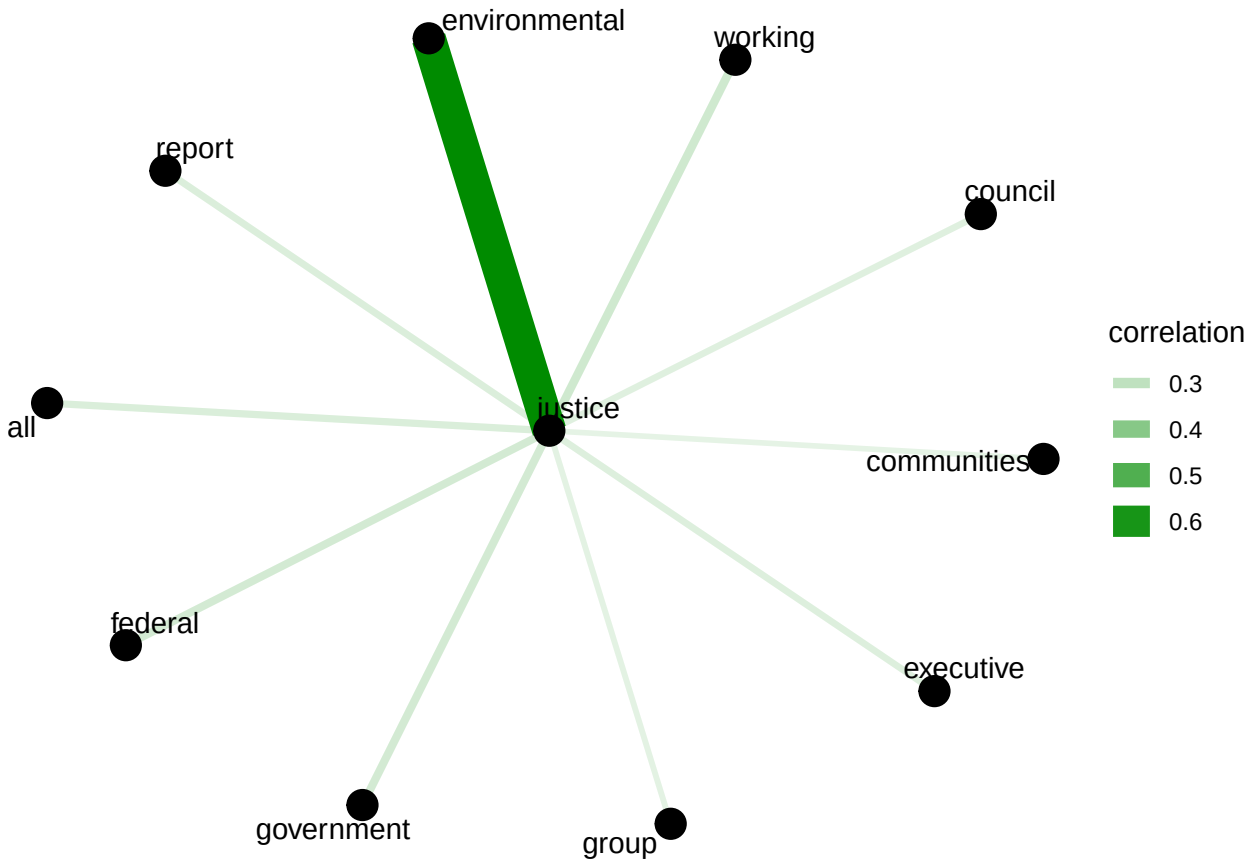



```
  #let's zoom in on just one of our key terms
   justice_cors <- word_cors %>%
  filter(item1 == "justice") %>%
   mutate(n = 1:n())
```

Not surprisingly, the correlation between "environmental" and "justice" is by far the highest, which makes sense given the nature of these reports. How might we visualize these correlations to develop of sense of the context in which justice is discussed here?

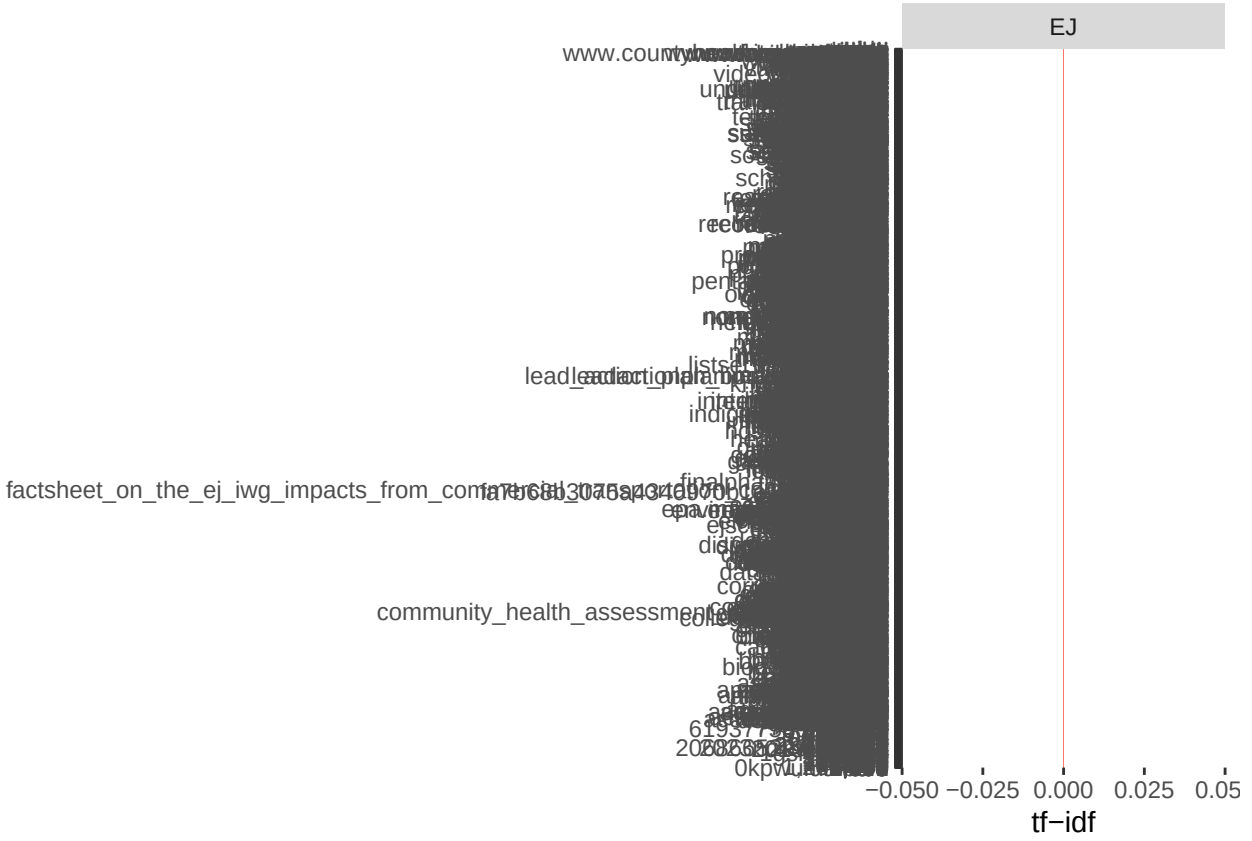
```
justice_cors  %>%
  filter(n <= 10) %>%
  graph_from_data_frame() %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = correlation, edge_width = correlation), edge_colour = "green4") +
  geom_node_point(size = 5) +
  geom_node_text(aes(label = name), repel = TRUE,
                 point.padding = unit(0.2, "lines")) +
```

```
  theme_void()
```



Now let's look at the tf-idf term we talked about. Remember, this statistic goes beyond simple frequency calculations within a document to control for overall commonality across documents

```
report_tf_idf <- report_words %>%
  bind_tf_idf(word, year, n) %>%
  select(-total) %>%
  arrange(desc(tf_idf))
report_tf_idf %>%
  group_by(year) %>%
  slice_max(tf_idf, n = 10) %>%
  ungroup() %>%
  filter(nchar(word) > 2)%>%
  ggplot(aes(tf_idf, fct_reorder(word, tf_idf), fill = year)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~year, ncol = 2, scales = "free") +
  labs(x = "tf-idf", y = NULL)
```



So that gives an idea which words are frequent and unique to certain documents.

Now let's switch gears to quanteda for some additional word relationship tools. We'll also get into some ways to assess the similarity of documents.

```
tokens <- tokens(epa_corp, remove_punct = TRUE)
toks1<- tokens_select(tokens, min_nchar = 3)
toks1 <- tokens_tolower(toks1)
toks1 <- tokens_remove(toks1, pattern = (stop_vec))
dfm <- dfm(toks1)
#first the basic frequency stat
tstat_freq <- textstat_frequency(dfm, n = 5, groups = year)
head(tstat_freq, 10)
```

```
##         feature frequency rank docfreq group
## 1 environmental      1088    1       6    EJ
## 2   communities       940    2       6    EJ
## 3           epa       929    3       6    EJ
## 4     community       744    4       6    EJ
## 5       justice       606    5       6    EJ
```

Q1.

What are the most frequent trigrams in the dataset? How does this compare to the most frequent bigrams? Which n-gram seems more informative here, and why?

The three most frequent trigrams in the dataset are "justice_fy2017_progress", "fy2017_progress_report", and "environmental_public_health". The top three most

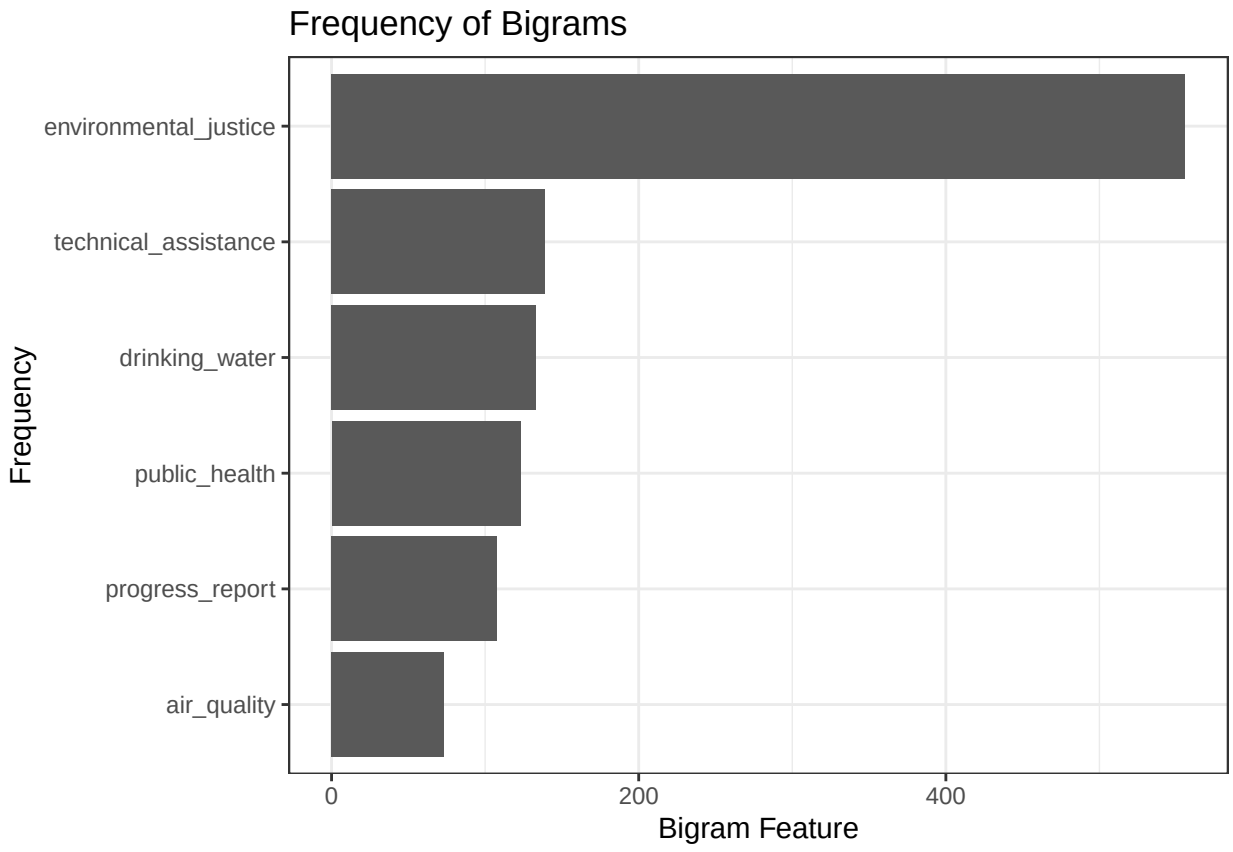
frequent bigrams are "environmental_justice", "technical_assistance", and "drinking_water".

Comparing the trigrams to bigrams, we can see that "environmental" is the most frequently used word in both and that the report called "fy2017" was only used commonly in trigrams and not in bigrams. Additionally, environmental justice was frquently used in trigrams, but with the report called "fy2017" included in the trigram.

Another useful word relationship concept is that of the n-gram, which essentially means tokenizing at the multi-word level

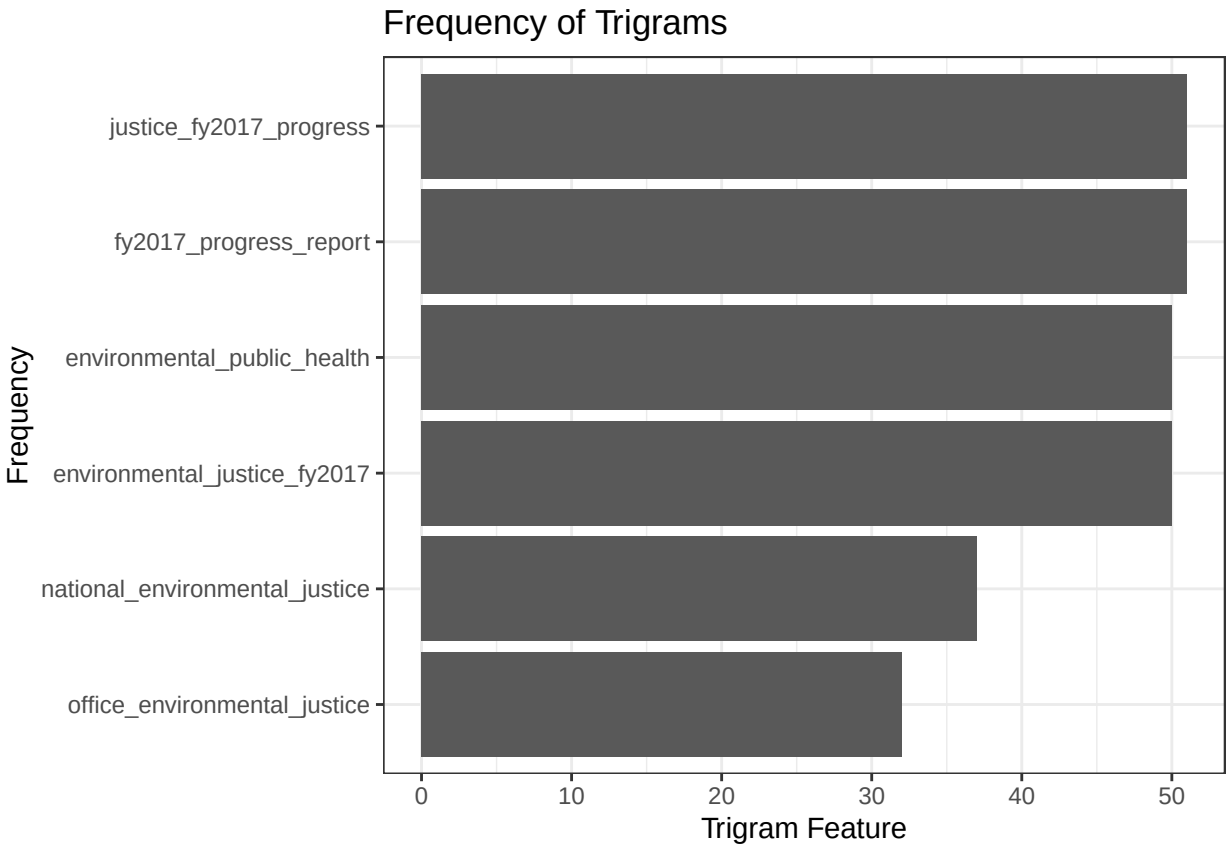
```
# Find the most frequent bigrams
toks2 <- tokens_ngrams(toks1, n=2)
dfm2 <- dfm(toks2)
dfm2 <- dfm_remove(dfm2, pattern = c(stop_vec))
freq_words2 <- textstat_frequency(dfm2, n=20)
freq_words2$token <- rep("bigram", 20)
#tokens1 <- tokens_select(tokens1,pattern = stopwords("en"), selection = "remove")
```

```
# Reorder bigram frequency by greatest to least
freq_words2_ordered <- freq_words2 %>% mutate(feature = reorder(feature, frequency))

# Visualize the most frequent bigrams
  ggplot(data = freq_words2_ordered, aes(x = feature, y = frequency)) +
  geom_col() +
  coord_flip() +
  labs(title = "Frequency of Bigrams",
       x = "Frequency",
       y = "Bigram Feature") +
    theme_bw()
```



```r
# Find the most frequent trigrams
toks3 <- tokens_ngrams(toks1, n=3)
dfm3 <- dfm(toks3)
dfm3 <- dfm_remove(dfm3, pattern = c(stop_vec))
freq_words3 <- textstat_frequency(dfm3, n=20)
freq_words3$token <- rep("trigram", 20)
#tokens1 <- tokens_select(tokens1,pattern = stopwords("en"), selection = "remove")
```

```r
# Reorder trigram frequency by greatest to least
freq_words3_ordered <- freq_words3 %>% mutate(feature = reorder(feature, frequency))

# Visualize the most frequent trigrams
  ggplot(data = freq_words3_ordered, aes(x = feature, y = frequency)) +
  geom_col() +
  coord_flip() +
  labs(title = "Frequency of Trigrams",
       x = "Frequency",
       y = "Trigram Feature") +
    theme_bw()
```



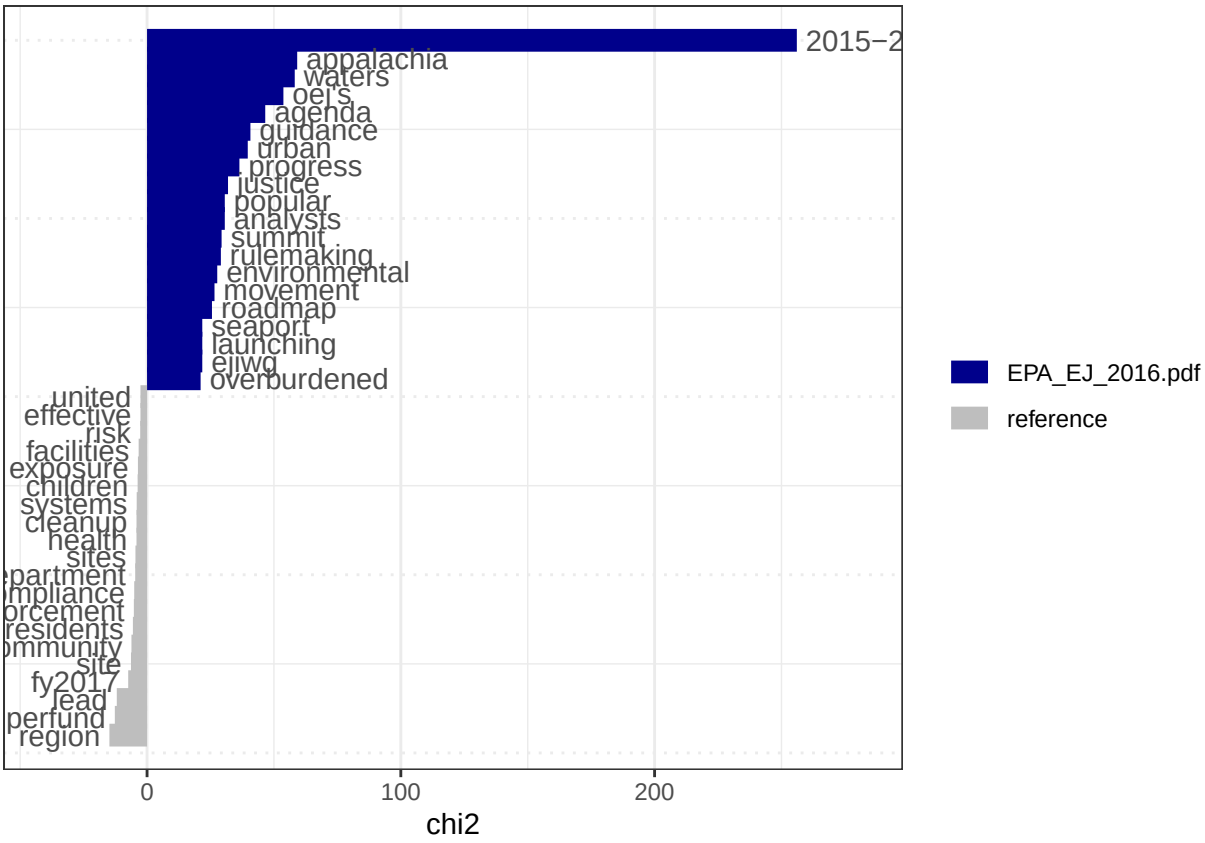
Q3.

Write a function that allows you to conduct a keyness analysis to compare two individual EPA reports (hint: that means target and reference need to both be individual reports). Run the function on 3 pairs of reports, generating 3 keyness plots.

Now we can upgrade that by using all of the frequencies for each word in each document and calculating a

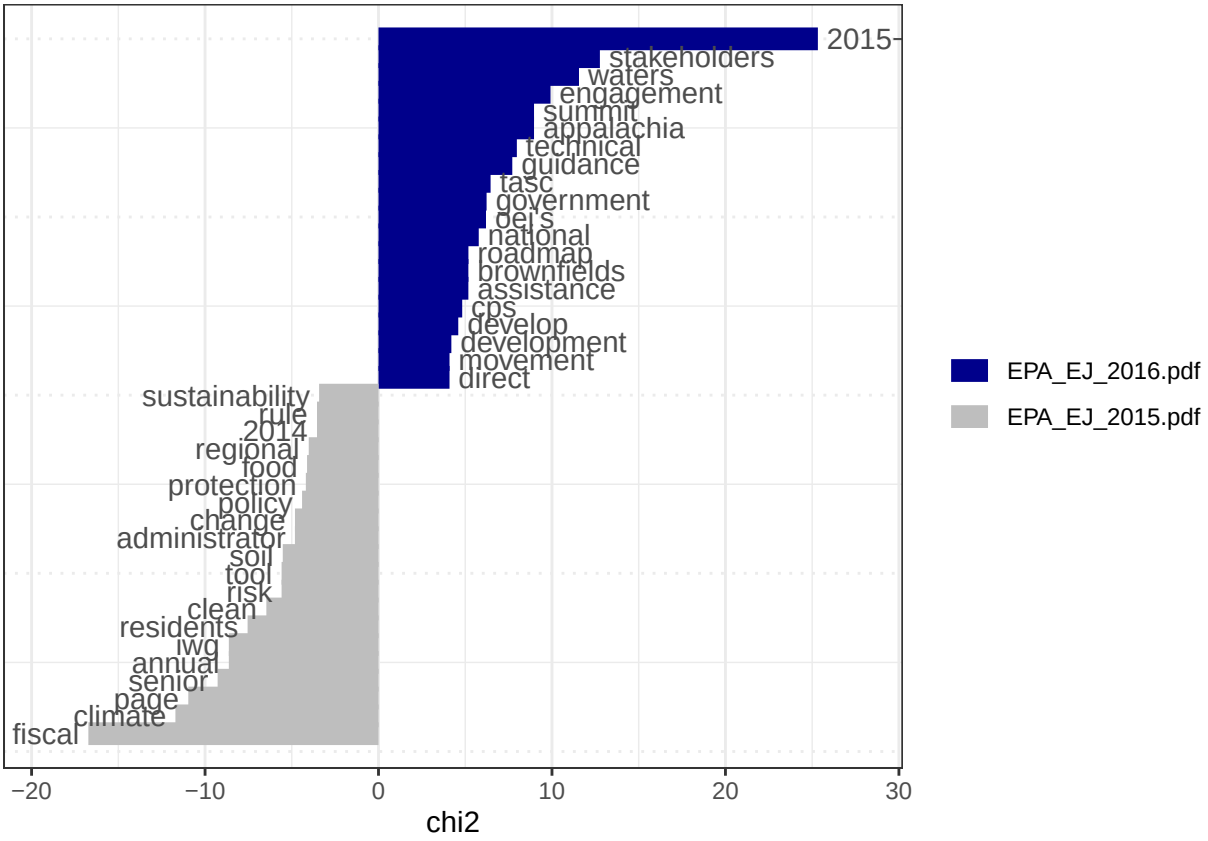
chi-square to see which words occur significantly more or less within a particular target document

```
keyness <- textstat_keyness(dfm, target = 2)
textplot_keyness(keyness)
```

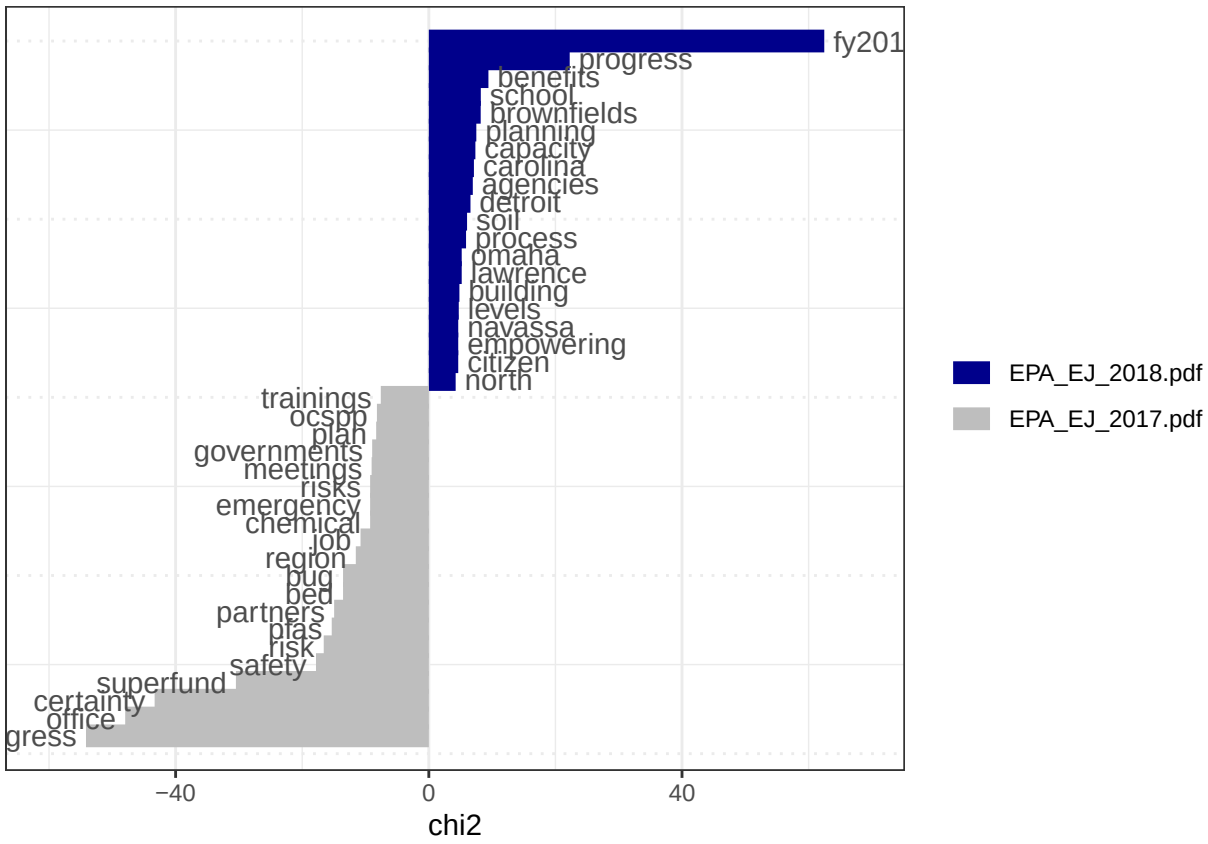



```
keyness_function <- function(EPA_report1, EPA_report2) {

files = c(EPA_report1, EPA_report2)

ej_reports <- lapply(files, pdf_text)

ej_pdf <- readtext(file = files,
                   docvarsfrom = "filenames",
                   docvarnames = c("type", "year"),
                   sep = "_")
#creating an initial corpus containing our data
epa_corp <- corpus(x = ej_pdf, text_field = "text" )

tokens <- tokens(epa_corp, remove_punct = TRUE)
toks1<- tokens_select(tokens, min_nchar = 3)
toks1 <- tokens_tolower(toks1)
toks1 <- tokens_remove(toks1, pattern = (stop_vec))
dfm <- dfm(toks1)

keyness <- textstat_keyness(dfm, target = 2)
return(textplot_keyness(keyness))
}
```

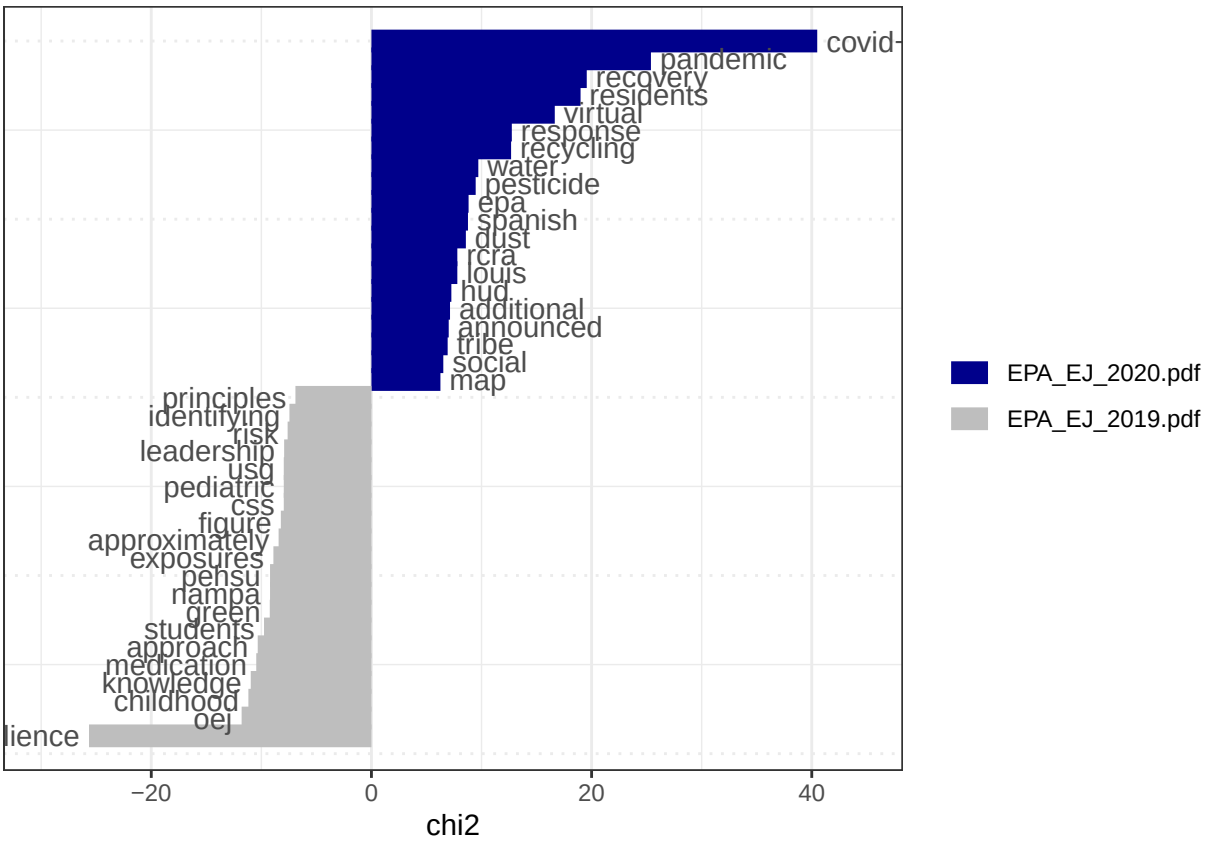
```
# Plot of keyness analysis to compare two individual EPA reports
keyness_function(EPA_report1 = "data/EPA_EJ_2015.pdf" , EPA_report2 = "data/EPA_EJ_2016.pdf")
```




```
# Plot of keyness analysis to compare two individual EPA reports

keyness_function(EPA_report1 = "data/EPA_EJ_2017.pdf" , EPA_report2 = "data/EPA_EJ_2018.pdf")
```

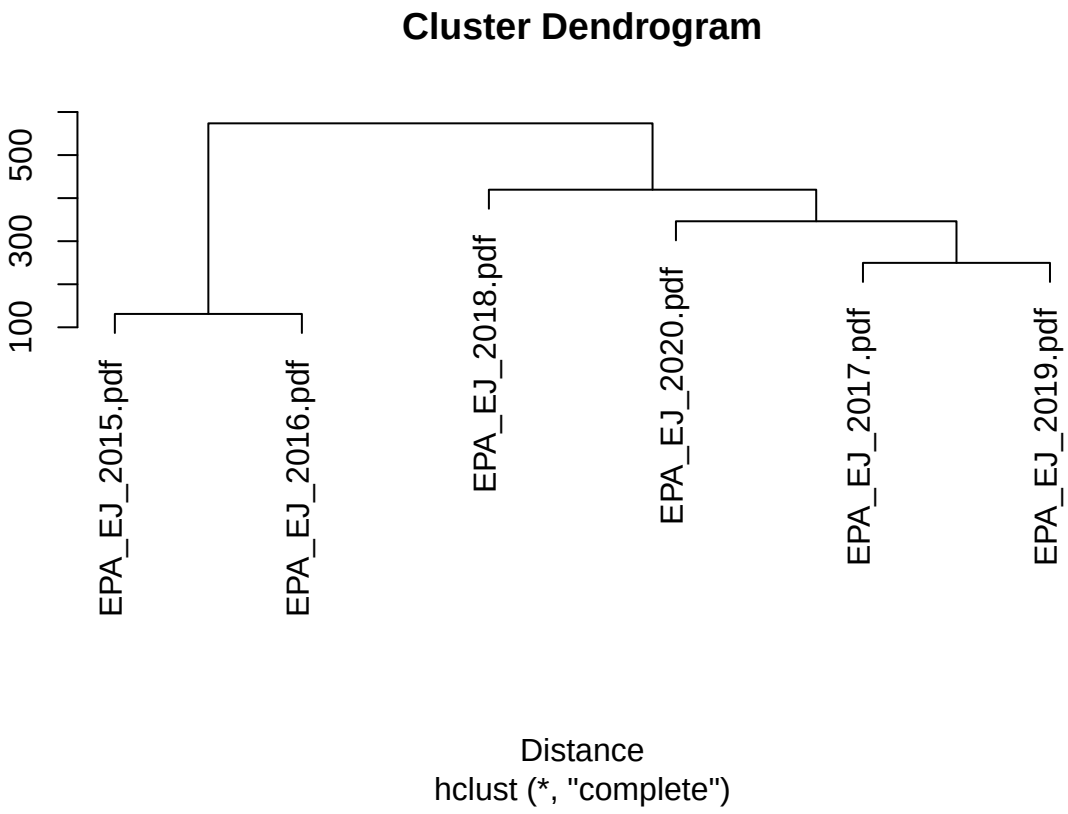



```
# Plot of keyness analysis to compare two individual EPA reports

keyness_function(EPA_report1 = "data/EPA_EJ_2019.pdf" , EPA_report2 = "data/EPA_EJ_2020.pdf")
```



And finally, we can run a hierarchical clustering algorithm to assess document similarity. This tends to be more informative when you are dealing with a larger number of documents, but we'll add it here for future reference.

```
dist <- as.dist(textstat_dist(dfm))
clust <- hclust(dist)
plot(clust, xlab = "Distance", ylab = NULL)
```



Q4.

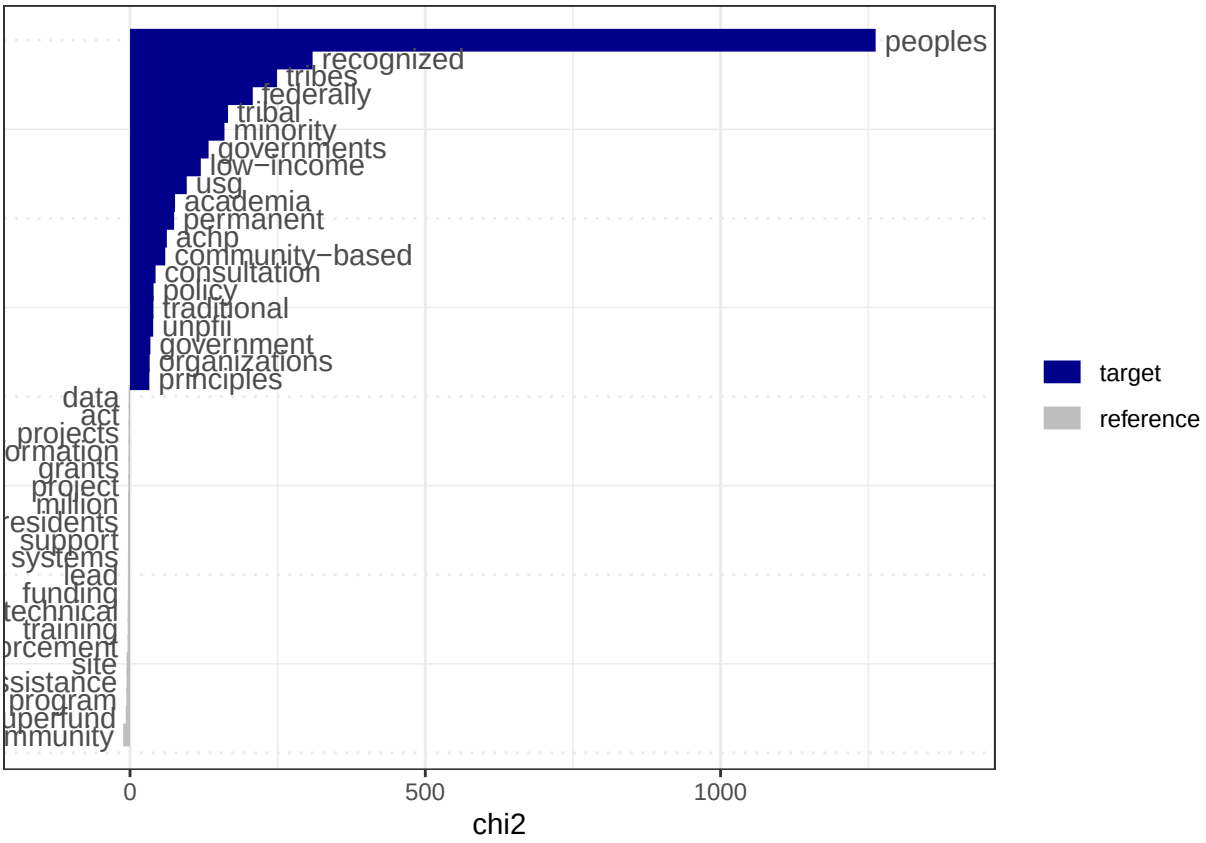
Select a word or multi-word term of interest and identify words related to it using windowing and keyness comparison. To do this you will create two objects: one containing all words occurring within a 10-word window of your term of interest, and the second object containing all other words. Then run a keyness comparison on these objects. Which one is the target, and which the reference? Hint

```
# We select two tokens objects for words inside and outside of the 10-word windows of the keywords (eu)

tokens <- tokens(epa_corp, remove_punct = TRUE)
toks1<- tokens_select(tokens, min_nchar = 3)
toks1 <- tokens_tolower(toks1)
toks1 <- tokens_remove(toks1, pattern = (stop_vec))

indigenous <- c("indigenous")
toks_inside <- tokens_keep(toks1, pattern = indigenous, window = 10)
toks_inside <- tokens_remove(toks_inside, pattern = indigenous) # remove the keywords
toks_outside <- tokens_remove(toks1, pattern = indigenous, window = 10)
```

```
# We compute words' association with the keywords using textstat_keyness().

dfmat_inside <- dfm(toks_inside)
dfmat_outside <- dfm(toks_outside)

tstat_key_inside <- textstat_keyness(rbind(dfmat_inside, dfmat_outside),
                                     target = seq_len(ndoc(dfmat_inside)))
head(tstat_key_inside, 10)
```

```
##         feature       chi2 p n_target n_reference
## 1       peoples 1262.56075 0       49           0
## 2    recognized  309.16345 0       19           9
## 3        tribes  248.78569 0       38          86
## 4    federally  207.86257 0       13           6
## 5       tribal  166.00369 0       47         200
## 6     minority  159.91760 0       25          57
## 7  governments  133.04273 0       22          53
## 8   low-income  119.84064 0       23          65
## 9          usg   96.04113 0        6           2
## 10    academia   76.27578 0        9          13
```

```
textplot_keyness(tstat_key_inside)
```



The target is all words occurring within a 10-word window, and the reference is all words occurring outside the 10-word window.

Assignment

1. What are the most frequent trigrams in the dataset? How does this compare to the most frequent bigrams? Which n-gram seems more informative here, and why?
2. Choose a new focal term to replace "justice" and recreate the correlation table and network (see corr_paragraphs and corr_network chunks). Explore some of the plotting parameters in the cor_network chunk to see if you can improve the clarity or amount of information your plot conveys. Make sure to use a different color for the ties!
3. Write a function that allows you to conduct a keyness analysis to compare two individual EPA reports (hint: that means target and reference need to both be individual reports). Run the function on 3 pairs of reports, generating 3 keyness plots.

4. Select a word or multi-word term of interest and identify words related to it using windowing and keyness comparison. To do this you will create two objects: one containing all words occurring within a 10-word window of your term of interest, and the second object containing all other words. Then run a keyness comparison on these objects. Which one is the target, and which the reference? Hint