Topic 5: Word Relationships EPA Reports on EJ

Halina Do-Linh

05/03/2022

Here I am creating an initial corpus, amending stop words, and converting the data into tidy format.

```
# intial corpus
epa_corpus <- corpus(x = ej_pdf,
                     text_field = "text")
summary(epa_corpus)
```

```
## 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  3973  30564       653  EPA   EJ    2017
##  EPA_EJ_2018.pdf  2774  16658       447  EPA   EJ    2018
##  EPA_EJ_2019.pdf  3773  22648       672  EPA   EJ    2019
##  EPA_EJ_2020.pdf  4493  30523       987  EPA   EJ    2020
```

```
# amending stop words
more_stops <-
  c("2015",
    "2016",
    "2017",
    "2018",
    "2019",
    "2020",
    "www.epa.gov",
    "https")
add_stops <- tibble(word = c(stop_words$word, more_stops))
# use stop vector with quanteda tools
stop_vec <- as_vector(add_stops)
```

```
# tidy format
tidy_text <- tidy(epa_corpus)

# adding stop words
words <- tidy_text %>%
  mutate(year = as.factor(docvar3)) %>%
  unnest_tokens(word, text) %>%
  anti_join(add_stops,  by = 'word') %>%
  select(-docvar3)
```

```r
# quanteda word relationship tools
tokens <- tokens(epa_corpus,
                 remove_punct = TRUE)
tokens_1 <- tokens_select(tokens,
                          min_nchar = 3)
tokens_1 <- tokens_tolower(tokens_1)
tokens_1 <- tokens_remove(tokens_1,
                          pattern = (stop_vec))
# create document feature matrix
dfm <- dfm(tokens_1)

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
```

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?

```r
# most freq trigrams
tokens_3 <- tokens_ngrams(tokens_1, n = 3)
dfm3 <- dfm(tokens_3)
dfm3 <- dfm_remove(dfm3, pattern = c(stop_vec))
freq_words3 <- textstat_frequency(dfm3, n = 20)
freq_words3$token <- rep("trigram", 20)

tstat_freq3 <- textstat_frequency(dfm3, n = 5, groups = year)
head(tstat_freq3, 10)
```

```
##                          feature frequency rank docfreq group
## 1        justice_fy2017_progress        51    1       1    EJ
## 2         fy2017_progress_report        51    1       1    EJ
## 3     environmental_public_health        50    3       6    EJ
## 4   environmental_justice_fy2017        50    3       1    EJ
## 5 national_environmental_justice        37    5       6    EJ
```

```r
# most freq bigrams
tokens_2 <- tokens_ngrams(tokens_1, n = 2)
dfm2 <- dfm(tokens_2)
dfm2 <- dfm_remove(dfm2, pattern = c(stop_vec))
freq_words2 <- textstat_frequency(dfm2, n = 20)
freq_words2$token <- rep("bigram", 20)

tstat_freq2 <- textstat_frequency(dfm2, n = 5, groups = year)
head(tstat_freq2, 10)
```

```
##                  feature frequency rank docfreq group
```

```
## 1 environmental_justice       556     1        6     EJ
## 2  technical_assistance       139     2        6     EJ
## 3        drinking_water       133     3        6     EJ
## 4         public_health       123     4        6     EJ
## 5       progress_report       108     5        6     EJ
```

Answer: The most frequent trigrams do not seem more informative than the most frequent bigrams. One of the top trigrams is `fy2017_progress_report` which is not informative at all. Because of this, I would say the bigrams are the more informative n-grams.

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!

Here I am tokenizing the paragraphs from my tidy corpus, and then tokenizing the paragraphs by words.

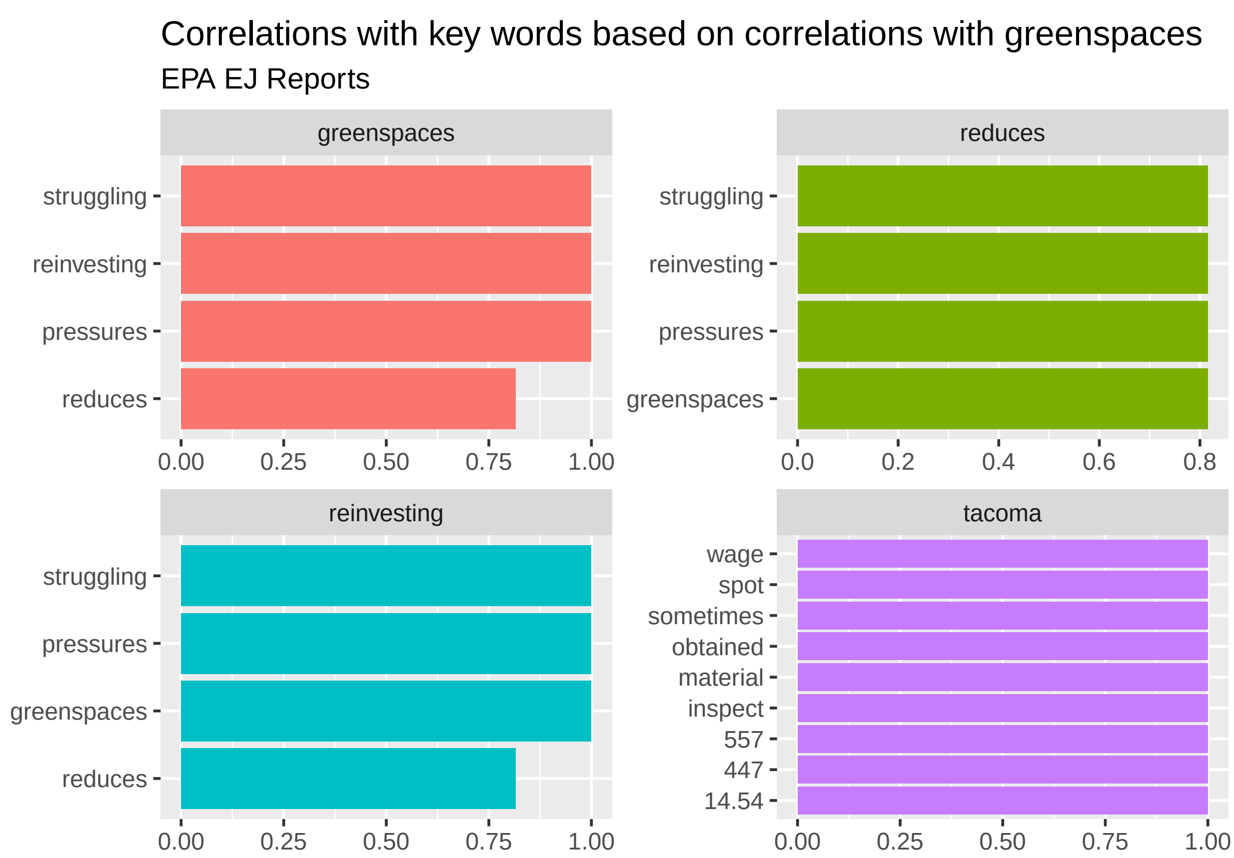
```r
# tokenize by paragraphs
paragraph_tokens <- unnest_tokens(tidy_text,
                                  output = paragraphs,
                                  input = text,
                                  token = "paragraphs")
# give each paragraph an id
paragraph_tokens <- paragraph_tokens %>%
  mutate(par_id = 1:n())
# tokenize paragraphs by words
paragraph_words <- unnest_tokens(paragraph_tokens,
                                 output = word,
                                 input = paragraphs,
                                 token = "words")
```

Here I am identifying which words tend to occur close together in the EPA reports, the word correlations, and chose "vegetation" as my focal word.

```r
# closely related pairs
word_pairs <- paragraph_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"))

# correlations
word_correlations <- paragraph_words %>%
  add_count(par_id) %>%
  filter(n >= 50) %>%
  select(-n) %>%
  pairwise_cor(word, par_id, sort = TRUE)

# focal word
greenspaces_correlations <- word_correlations %>%
  filter(item1 == "greenspaces") %>%
  mutate(n = 1:n())
```

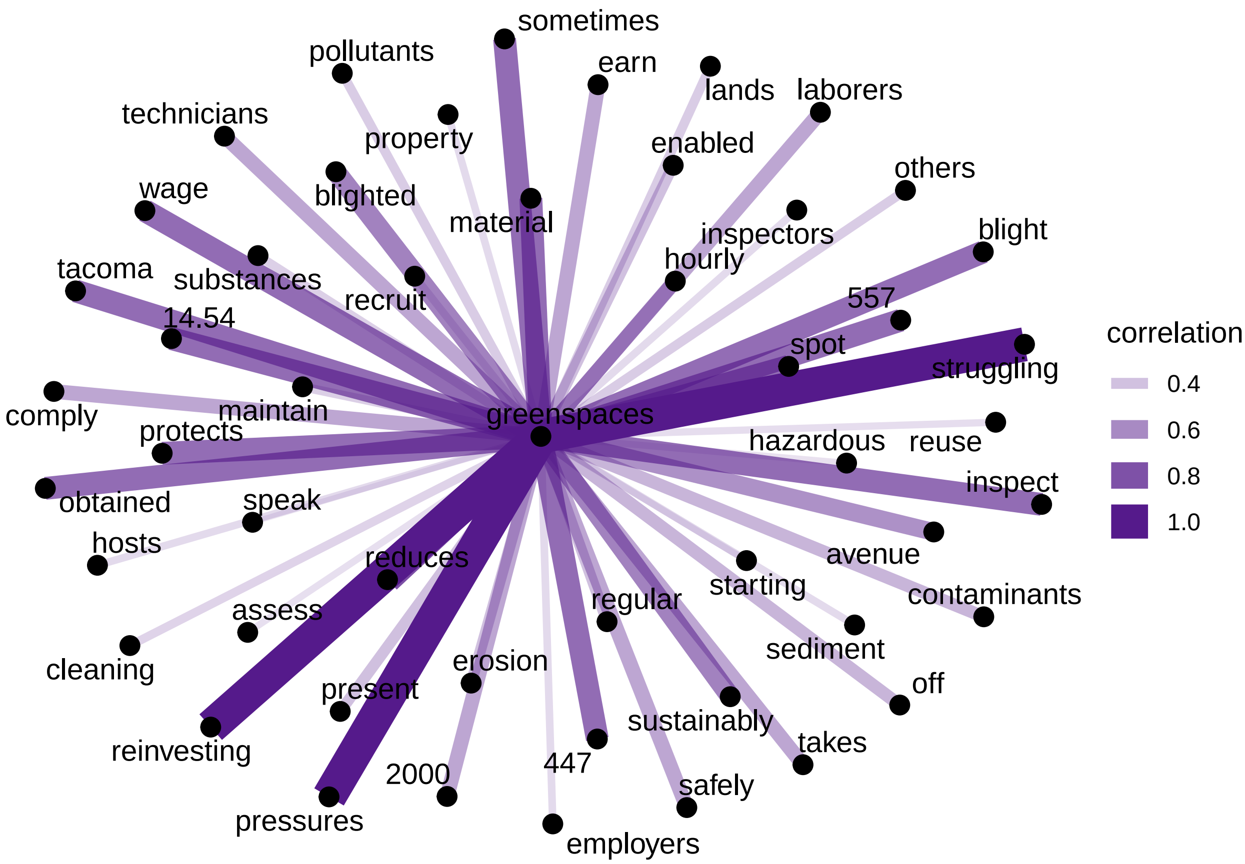
Here I am recreating the correlation table and network.

```r
# correlations
word_correlations %>%
  filter(item1 %in% c("greenspaces",
                      "reinvesting",
                      "reduces",
                      "tacoma")) %>%
  group_by(item1) %>%
  top_n(4) %>% # top 4 words
  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 based on correlations with greenspaces",
         subtitle = "EPA EJ Reports")
```




```r
# network
greenspaces_correlations  %>%
  filter(n <= 50) %>%
  graph_from_data_frame() %>%
  ggraph(layout = "fr") +
```

```r
  geom_edge_link(aes(edge_alpha = correlation, edge_width = correlation), edge_colour = "purple4",
                 size = 2) +
  geom_node_point(size = 3) +
  geom_node_text(aes(label = name), repel = TRUE,
                 point.padding = unit(0.2, "lines")) +
  theme_void()
```

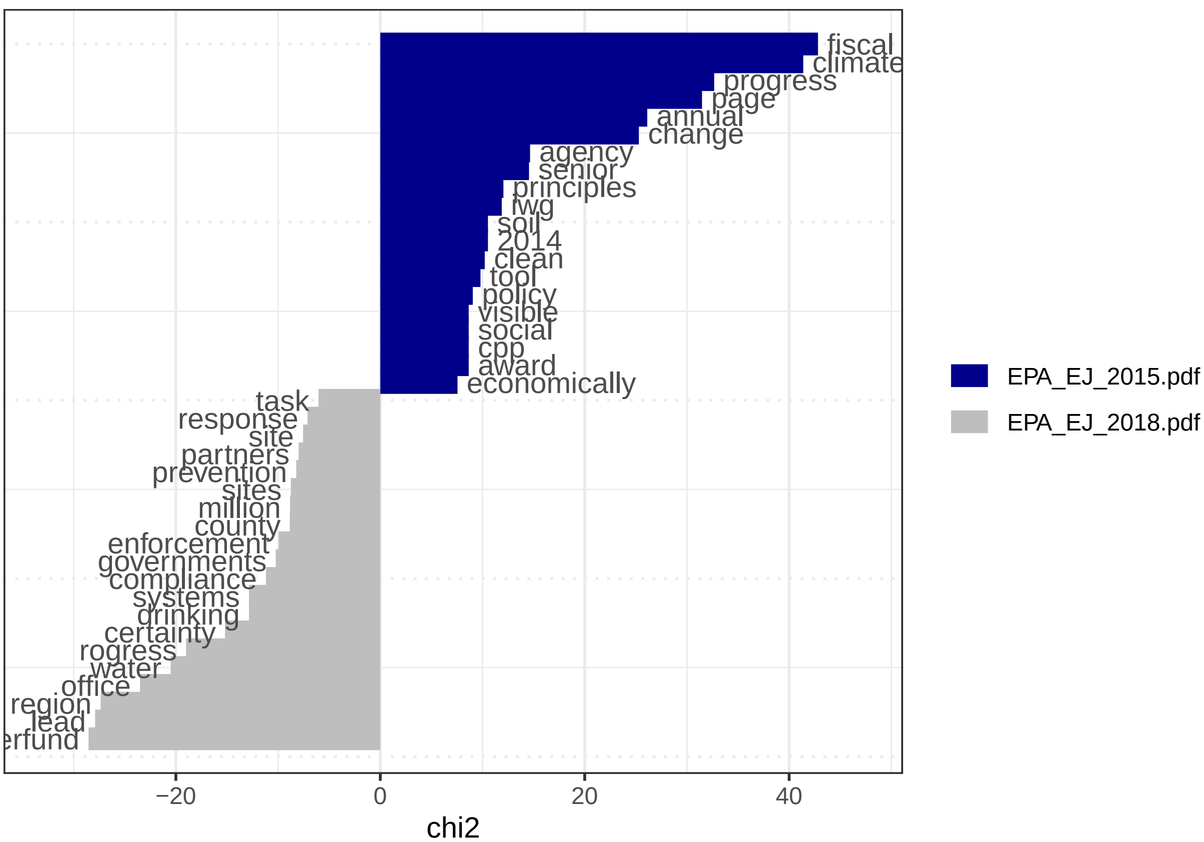


3. Write a function that allows you to conduct a keyness analysis to compare two individual EPA reports (hint: that means target and reference need to both be individual reports). Run the function on 3 pairs of reports, generating 3 keyness plots.

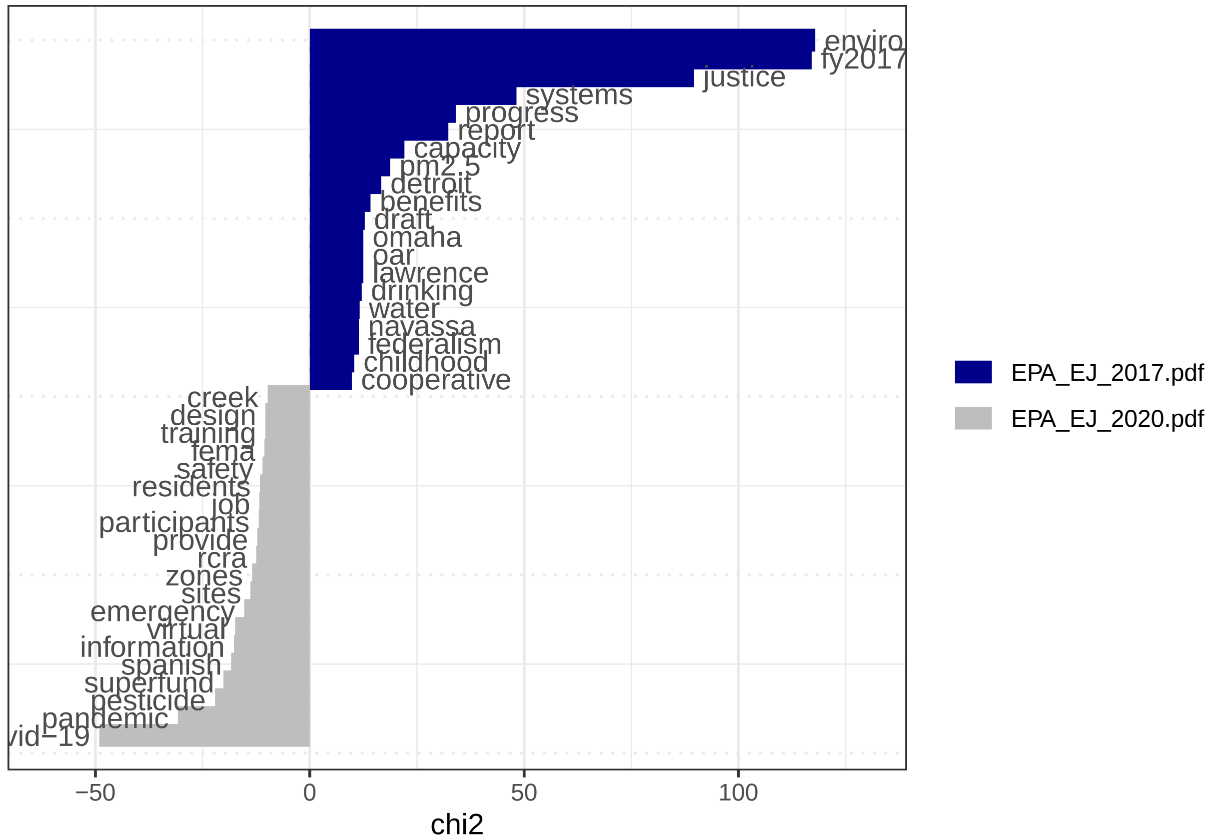
```r
dual_keyness <- function(years, target){

  # read in data
  files <- list.files(
    path = here::here("hw/epa_data"),
    pattern = "pdf$",
    full.names = TRUE)

  ej_reports <- lapply(files, pdf_text)

  # create df of all 6 PDf reports
  ej_pdf <- readtext(
    file = files,
    docvarsfrom = "filenames",
    docvarnames = c("type", "year"),
```

```r
    sep = "_") %>%
    filter(docvar3 %in% years)

  # creating an initial corpus
  epa_corp <- corpus(x = ej_pdf, text_field = "text")

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

  doc_freq_matrix <- dfm(tokens)

  keyness <- textstat_keyness(doc_freq_matrix,
                              target = target) # target refers to document you are comparing to
  textplot_keyness(keyness)
}
```

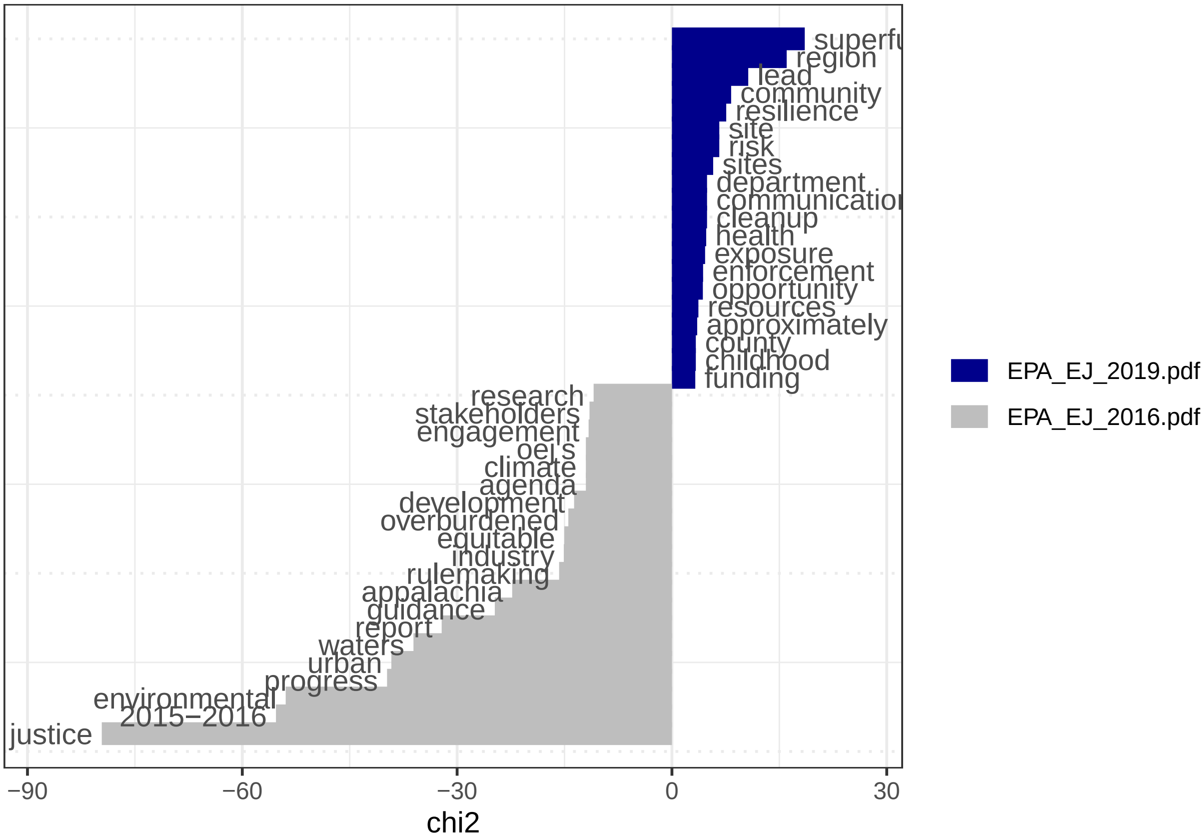
```r
dual_keyness(years = c(2015, 2018), target = 1)
```



```r
dual_keyness(years = c(2017, 2020), target = 1)
```




```r
dual_keyness(years = c(2016, 2019), target = 2)
```



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

Answer: The target is the list of words within a 10 word window based on the key terms I've chosen which is "air" and "air quality". The reference is the list of all other words outside of the 10 word window.

```
air <- c("air", "air quality")

toks_inside <- tokens_keep(tokens_1, pattern = air, window = 10) %>%
  tokens_remove(pattern = air) # remove the keywords

toks_outside <- tokens_remove(tokens_1, pattern = air, window = 10)
```

```
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, 20)
```

```
##           feature      chi2            p n_target n_reference
## 1         quality 512.53160 0.000000e+00       85          66
## 2       pollution 202.64548 0.000000e+00       42          46
## 3           clean 169.68781 0.000000e+00       39          49
```

```
## 4       radiation 145.52291 0.000000e+00       13           0
## 5           pm2.5 141.16927 0.000000e+00       20          10
## 6         ambient 134.64201 0.000000e+00       14           2
## 7      monitoring 105.02684 0.000000e+00       24          28
## 8       standards 102.24938 0.000000e+00       24          29
## 9      pollutants 100.46687 0.000000e+00       16          10
## 10            oar  76.00121 0.000000e+00       12           7
## 11    particulate  66.70720 3.330669e-16       12           9
## 12      emissions  66.45618 3.330669e-16       20          32
## 13           land  64.56925 8.881784e-16       20          33
## 14       particle  61.97559 3.441691e-15        9           4
## 15       monitors  60.52323 7.216450e-15        6           0
## 16           fine  51.63956 6.667999e-13        8           4
## 17 non-attainment  48.45544 3.378742e-12        5           0
## 18        chelsea  47.03849 6.960654e-12        7           3
## 19            act  45.11205 1.860767e-11       23          63
## 20          noise  38.90749 4.443717e-10        5           1
```