Topic 5: Word Relationships

Joe DeCesaro

2022-05-03

```
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

Setup

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

ej_reports <- lapply(X = files,
                     FUN = 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" )

#I'm adding some additional, context-specific stop words to stop word lexicon
```

```r
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

```r
#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")
```

```r
word_cors <- par_words %>%
  add_count(par_id) %>%
  filter(n >= 50) %>%
  select(-n) %>%
  pairwise_cor(word, par_id, sort = TRUE)
```

```r
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       127    1       1  2015
## 2    communities        99    2       1  2015
## 3            epa        92    3       1  2015
## 4        justice        84    4       1  2015
## 5      community        47    5       1  2015
```

```
## 6  environmental       109    1       1  2016
## 7    communities        85    2       1  2016
## 8        justice        71    3       1  2016
## 9            epa        48    4       1  2016
## 10       federal        31    5       1  2016
```

```
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")
```

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?

```
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")
```

```
freq_words2 <- data.frame(freq_words2) %>% select(feature, frequency, token)
freq_words3 <- data.frame(freq_words3) %>% select(feature, frequency, token)

head(freq_words2)
```

```
##                feature frequency  token
## 1 environmental_justice       556 bigram
## 2  technical_assistance       139 bigram
## 3        drinking_water       133 bigram
## 4         public_health       123 bigram
## 5       progress_report       108 bigram
## 6           air_quality        73 bigram
```

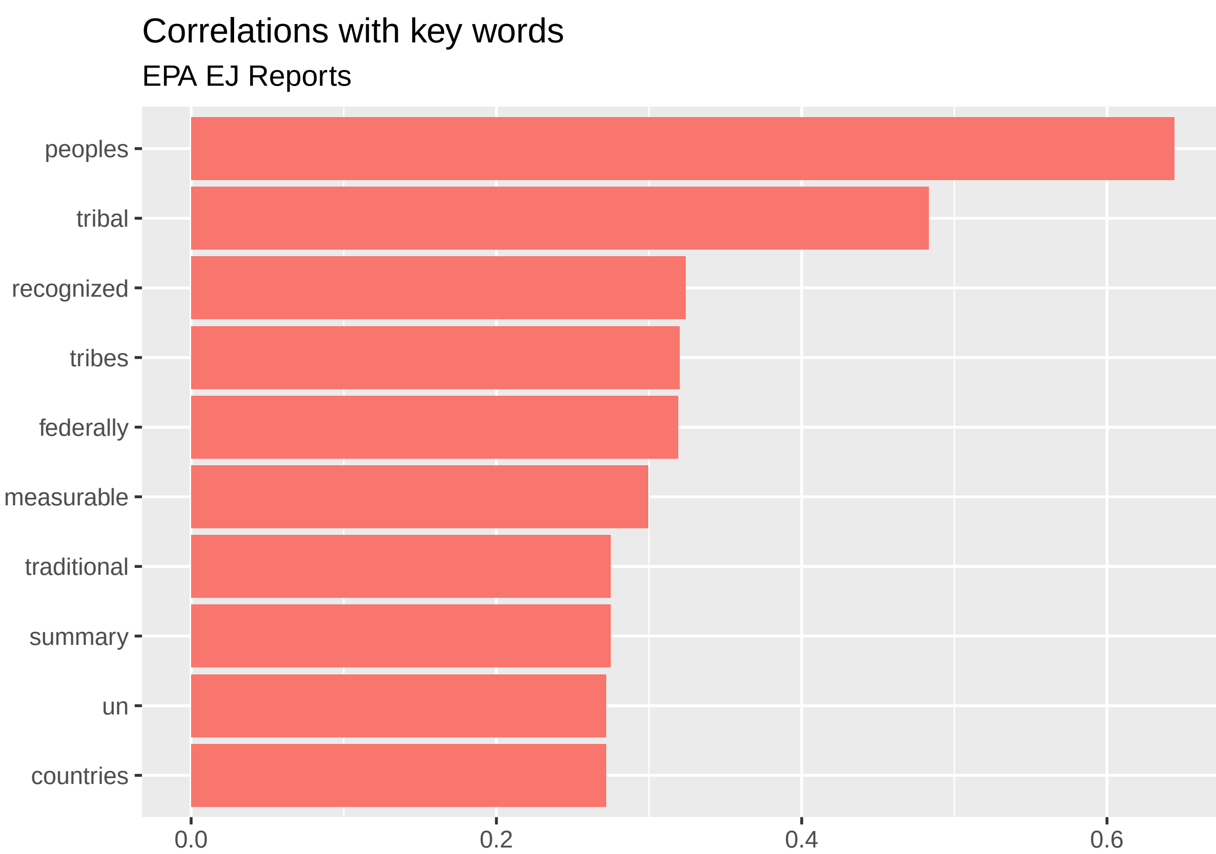
```
head(freq_words3)
```

```
##                          feature frequency   token
## 1        justice_fy2017_progress        51 trigram
## 2         fy2017_progress_report        51 trigram
## 3    environmental_public_health        50 trigram
## 4   environmental_justice_fy2017        50 trigram
## 5 national_environmental_justice        37 trigram
## 6   office_environmental_justice        32 trigram
```

The trigram most frequent "features" seem to mostly be official names of reports and does not seem to tell us anything more interesting about the documents than the bigrams. I think the bigrams are a more useful tool.

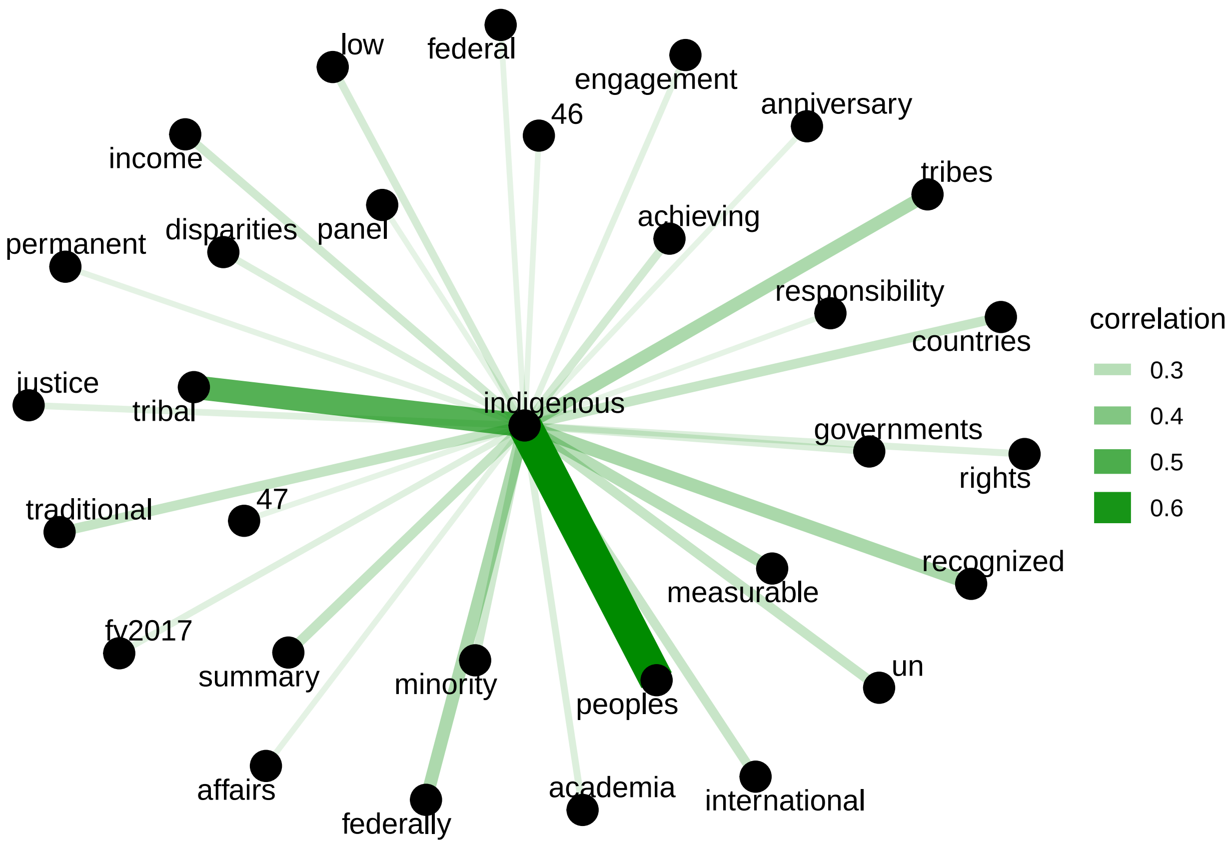
2. Choose a new focal term to replace "justice" and recreate the correlation table and network (see corr__paragraphs and corr__network chunks). Explore some of the plotting parameters in the cor__network chunk to see if you can improve the clarity or amount of information your plot conveys. Make sure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

```
just_cors <- word_cors %>%
  filter(item1 == "indigenous")

word_cors %>%
  filter(item1 %in% c("indigenous"))%>%
  group_by(item1) %>%
  top_n(10) %>%
  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) +
  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
indigenous_cors <- word_cors %>%
```

```
  filter(item1 == "indigenous") %>%
  mutate(n = 1:n())

indigenous_cors  %>%
  filter(n <= 30) %>%
  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()
```

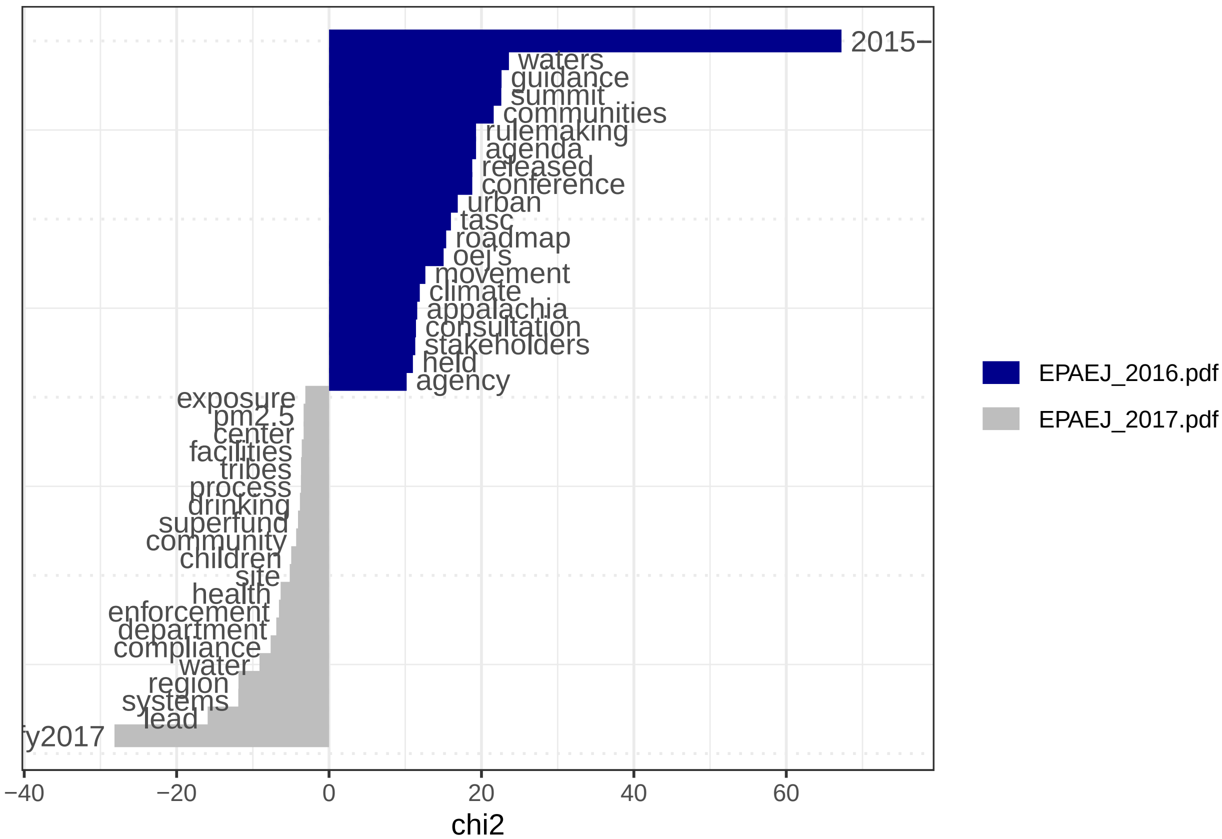


3. Write a function that allows you to conduct a keyness analysis to compare two individual EPA reports (hint: that means target and reference need to both be individual reports). Run the function on 3 pairs of reports, generating 3 keyness plots.

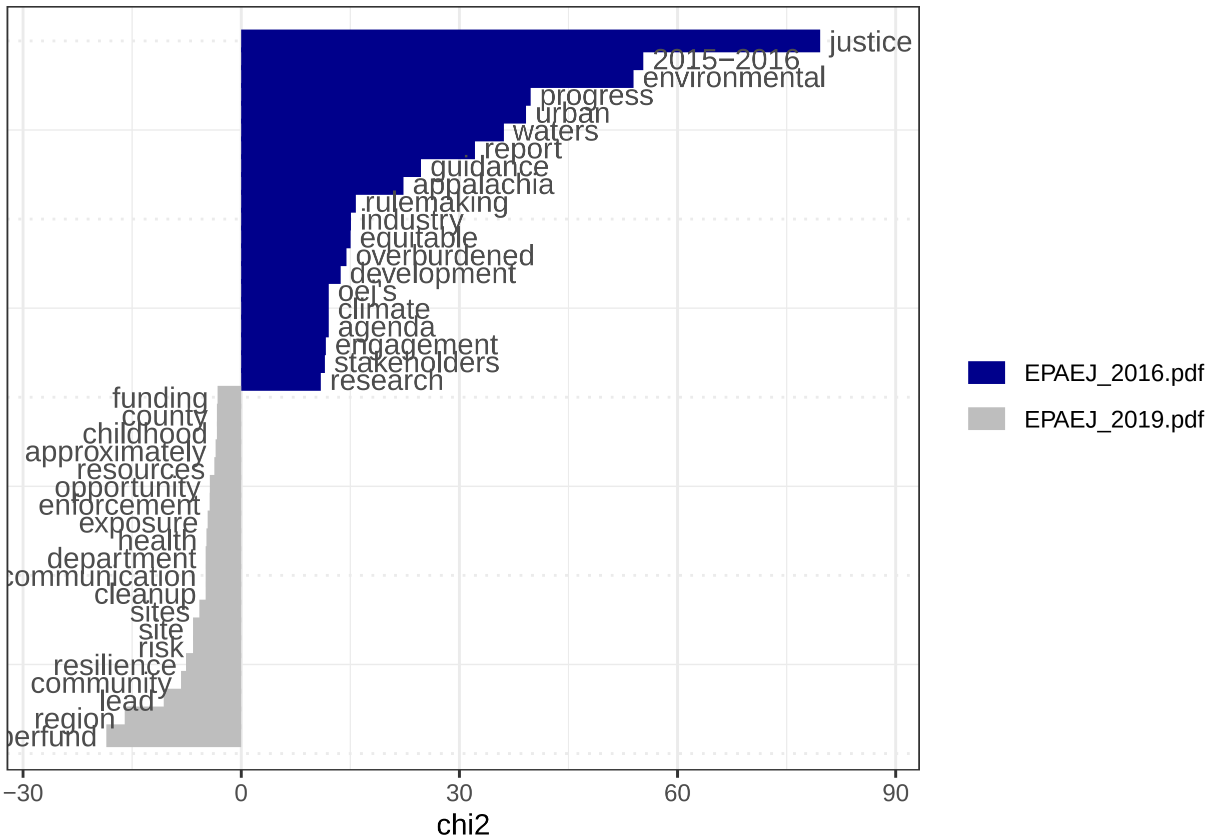
```r
#Distribution of most frequent words across documents
raw_words <- raw_text %>%
  mutate(year = as.factor(year)) %>%
  unnest_tokens(word, text) %>%
  anti_join(add_stops, by = 'word') %>%
  count(year, word, sort = TRUE)
```

```r
key_func <- function(target_rep, reference_rep){
  # read in the files
  files <- c(target_rep, reference_rep)
  reports <- lapply(X = files,
                    FUN = pdf_text)

  pdf_files <- readtext(file = files,
                        docvarsfrom = "filenames",
                        docvarnames = c("type", "year"),
                        sep = "_")

  #creating an initial corpus containing our data
  corp <- corpus(x = pdf_files, text_field = "text" )

  # tokenize
  tokens <- tokens(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)

  keyness <- textstat_keyness(dfm, target = 1)
  return(keyness)

}
```

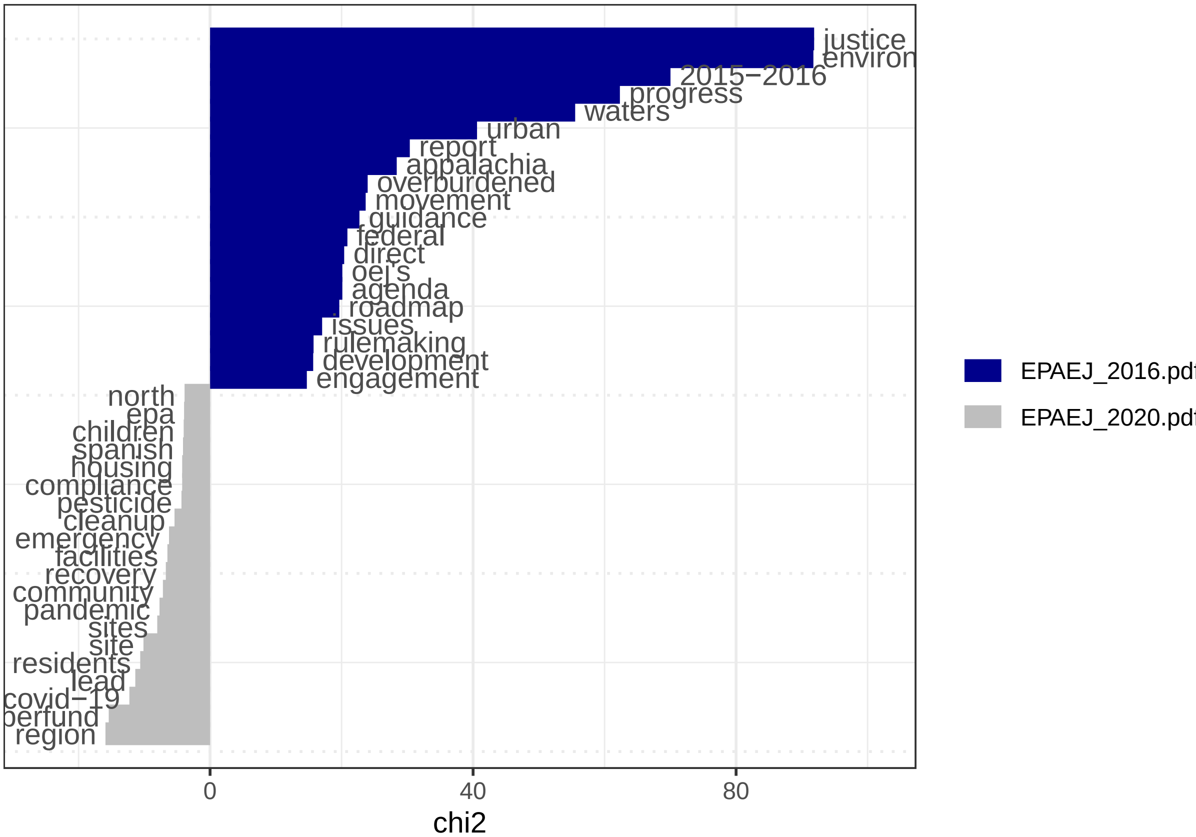
```
keyness1 <- key_func(target_rep = here("data", "EPAEJ_2016.pdf"),
                     reference_rep = here("data", "EPAEJ_2017.pdf"))

textplot_keyness(keyness1)
```



```r
keyness2 <- key_func(target_rep = here("data", "EPAEJ_2016.pdf"),
                     reference_rep = here("data", "EPAEJ_2019.pdf"))

textplot_keyness(keyness2)
```



```r
keyness3 <- key_func(target_rep = here("data", "EPAEJ_2016.pdf"),
                     reference_rep = here("data", "EPAEJ_2020.pdf"))

textplot_keyness(keyness3)
```



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

```
indigenous <- c("indigenous", "indigenous peoples", "peoples")
toks_inside <- tokens_keep(tokens, pattern = indigenous, window = 10)
toks_inside <- tokens_remove(toks_inside, pattern = indigenous) # remove the keywords
toks_outside <- tokens_remove(tokens, pattern = indigenous, 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       recognized 326.29503 0.000000e+00       15          13
## 2           tribes 317.31031 0.000000e+00       32          92
## 3           tribal 217.60905 0.000000e+00       39         208
## 4        federally 205.74679 0.000000e+00       10           9
## 5         minority 202.52656 0.000000e+00       21          61
## 6       low-income 166.64938 0.000000e+00       20          68
## 7      governments 139.76071 0.000000e+00       17          58
## 8         academia 106.13316 0.000000e+00        8          14
## 9  community-based  95.75908 0.000000e+00       12          41
## 10       permanent  86.82673 0.000000e+00        5           5
## 11   organizations  63.51566 1.554312e-15       16         107
## 12    collaborates  57.43890 3.486100e-14        4           5
## 13       panelists  54.13586 1.870726e-13        3           2
## 14     communities  51.05075 9.000578e-13       52         888
## 15      principles  48.97480 2.592704e-12        8          34
## 16            side  45.77579 1.325928e-11        4           7
## 17     protections  44.30239 2.813716e-11        3           3
## 18             and  36.37656 1.626472e-09      158        4464
## 19              un  32.02708 1.520380e-08        3           5
## 20             usg  32.02708 1.520380e-08        3           5
```

The tokens inside the 10-word window are the targets while the tokens outside the 10-word window are the references. The word “recognized” happens inside the 10 word window of indigenous words 15 times and outside of the window 13 times.