EDS 231: Text and Sentiment Analysis for Environmental Problems - Text Data in R

Juliet Cohen

4/12/2022

```
library(jsonlite) #convert results from API queries into R-friendly formats
library(tidyverse)
library(tidytext) # text data management and analysis
library(ggplot2) # plot word frequencies and publication dates
```

```
# the from JSON flatten the JSON object, then convert to a data frame
query_results <- fromJSON("http://api.nytimes.com/svc/search/v2/articlesearch.json?q=cat&api-key=IpTnsN
# q = wildlife is the word we are looking for
# flatten = true unnests the JSON structure so we can work with the data

class(query_results) #what type of object is the query?
```

```
## [1] "list"
```

```
# list

# convert the list to a df
query_results <- query_results %>%
  data.frame()

# Inspect our data
class(query_results) #now what is it? # df
```

```
## [1] "data.frame"
```

```
dim(query_results) # how big is it? 10 rows (articles), 33 columns (variables/fields for each article o
```

```
## [1] 10 33
```

```
names(query_results) # what variables are we working with? these are the variables
```

```
##  [1] "status"
##  [2] "copyright"
##  [3] "response.docs.abstract"
##  [4] "response.docs.web_url"
##  [5] "response.docs.snippet"
##  [6] "response.docs.lead_paragraph"
##  [7] "response.docs.print_section"
##  [8] "response.docs.print_page"
##  [9] "response.docs.source"
## [10] "response.docs.multimedia"
## [11] "response.docs.keywords"
## [12] "response.docs.pub_date"
```

```
## [13] "response.docs.document_type"
## [14] "response.docs.news_desk"
## [15] "response.docs.section_name"
## [16] "response.docs.type_of_material"
## [17] "response.docs._id"
## [18] "response.docs.word_count"
## [19] "response.docs.uri"
## [20] "response.docs.subsection_name"
## [21] "response.docs.headline.main"
## [22] "response.docs.headline.kicker"
## [23] "response.docs.headline.content_kicker"
## [24] "response.docs.headline.print_headline"
## [25] "response.docs.headline.name"
## [26] "response.docs.headline.seo"
## [27] "response.docs.headline.sub"
## [28] "response.docs.byline.original"
## [29] "response.docs.byline.person"
## [30] "response.docs.byline.organization"
## [31] "response.meta.hits"
## [32] "response.meta.offset"
## [33] "response.meta.time"
```

```
# the periods are bc it used to be a JSON object
```

Create a query for the word "cat" spanning from 1 year ago to today (April 9th, 2022)

```
term <- "cat" # Need to use + to string together separate words
begin_date <- "20210409"
end_date <- "20220409"

#construct the query url using API operators
baseurl <- paste0("http://api.nytimes.com/svc/search/v2/articlesearch.json?q=",term,
                  "&begin_date=",begin_date,"&end_date=",end_date,
                  "&facet_filter=true&api-key=","IpTnsNntENJKZIG3hUdas1S5PWeEt1ko", sep="")

#nexamine our query url
baseurl
```

```
## [1] "http://api.nytimes.com/svc/search/v2/articlesearch.json?q=cat&begin_date=20210409&end_date=2022
```

```
# initialQuery$response$meta$hits[1]
# there are 1209 hits for cat
# 1209 / 120.9 = 10

# this code allows for obtaining multiple pages of query results
initialQuery <- fromJSON(baseurl)
maxPages <- round((initialQuery$response$meta$hits[1] / 120.9))
maxPages # this will give us 10 pages max
```

```
## [1] 10
```

```
pages <- list()
for(i in 0:maxPages){
  nytSearch <- fromJSON(paste0(baseurl, "&page=", i), flatten = TRUE) %>% data.frame()
  message("Retrieving page ", i)
  pages[[i+1]] <- nytSearch
  Sys.sleep(6)
```

```
}

class(nytSearch)
```

```
## [1] "data.frame"
```

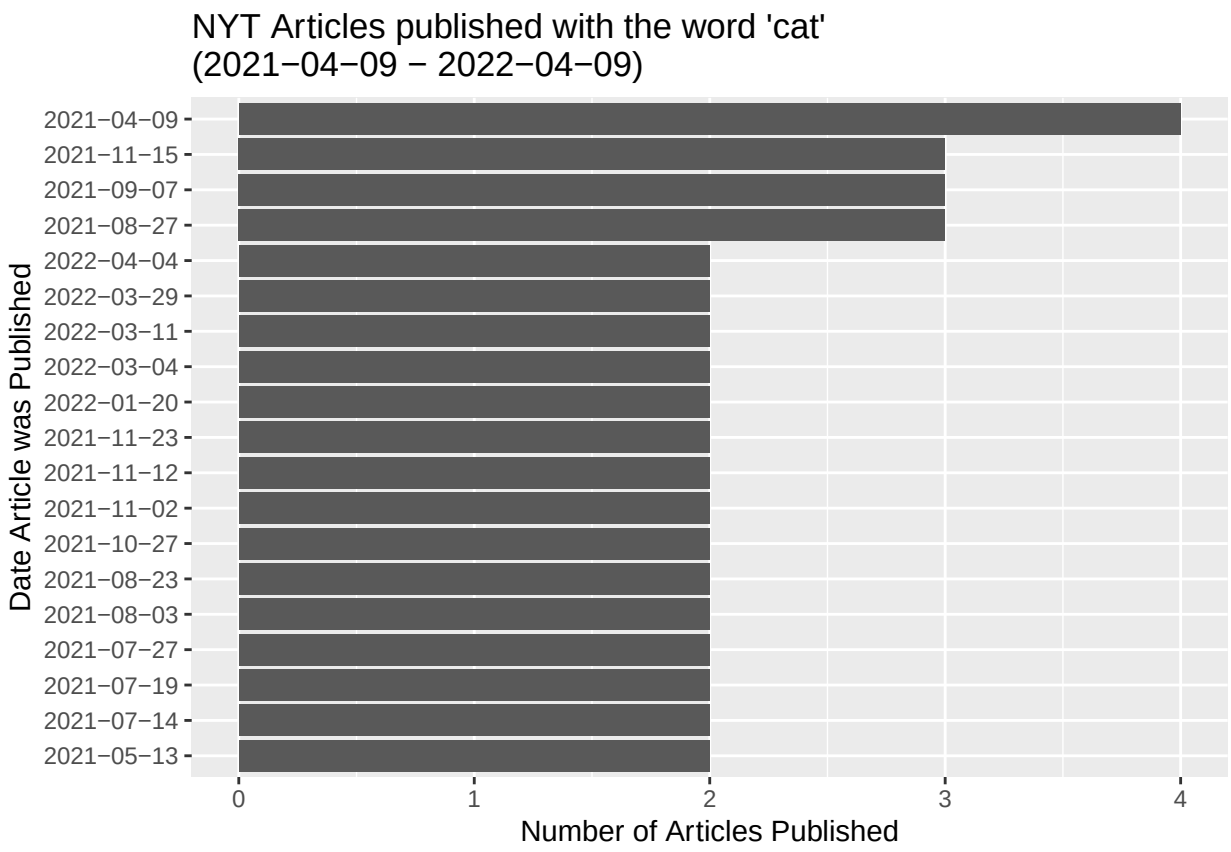
```
# need to bind the pages and create a tibble from nytDa
nytDat_cat <- rbind_pages(pages)
#nytDat_wildlife <- read.csv("nytDat.csv")
dim(nytDat_cat)
```

```
## [1] 110  33
```

```
# 110 rows and 33 cols
```

Publications Per Day

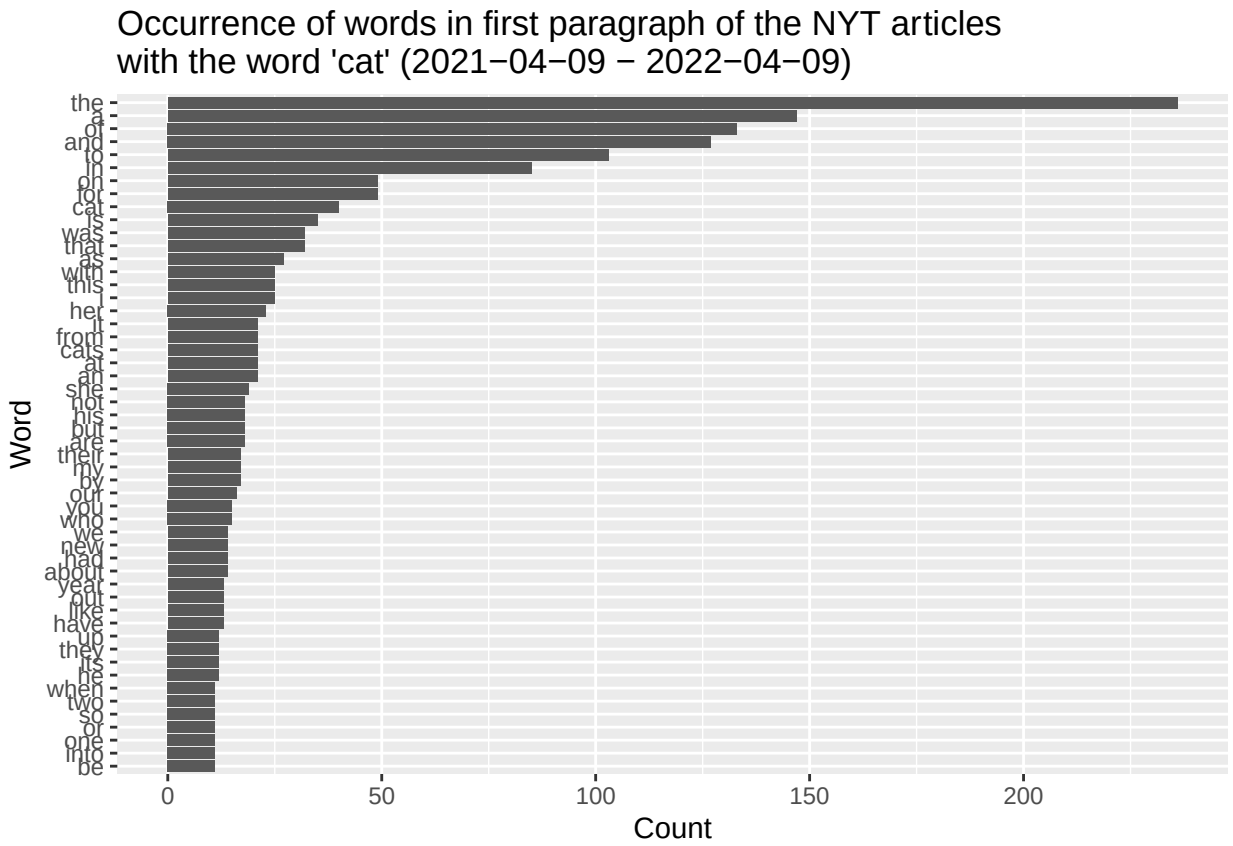
```
nytDat_cat %>%
  mutate(pubDay=gsub("T.*","",response.docs.pub_date)) %>%
  group_by(pubDay) %>%
  summarise(count=n()) %>%
  filter(count >= 2) %>%
  ggplot() +
  geom_bar(aes(x = reorder(pubDay, count), y=count), stat="identity") + coord_flip() +
  ylab("Number of Articles Published") +
  xlab("Date Article was Published") +
  ggtitle("NYT Articles published with the word 'cat'\n(2021-04-09 - 2022-04-09)")
```



Word Frequency Plot (using the first paragraph)

```
#names(nytDat_cat)
# we want to use col 6 for the lead paragraph

# create a list that is the column of the first paragraph of each article
first_paragraph <- names(nytDat_cat)[6]
# The 6th column, "response.doc.lead_paragraph", is the one we want here

tokenized <- nytDat_cat %>%
# create an df object of the nytData_cat df PLUS a column where each row is a word that was present in
  unnest_tokens(word, first_paragraph)

#tokenized[,34]
```

```
tokenized %>%
  count(word, sort = TRUE) %>%
  filter(n > 10) %>% #illegible with all the words displayed
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL) +
  xlab("Count") +
  ylab("Word") +
  ggtitle("Occurrence of words in first paragraph of the NYT articles\nwith the word 'cat' (2021-04-09 -
```

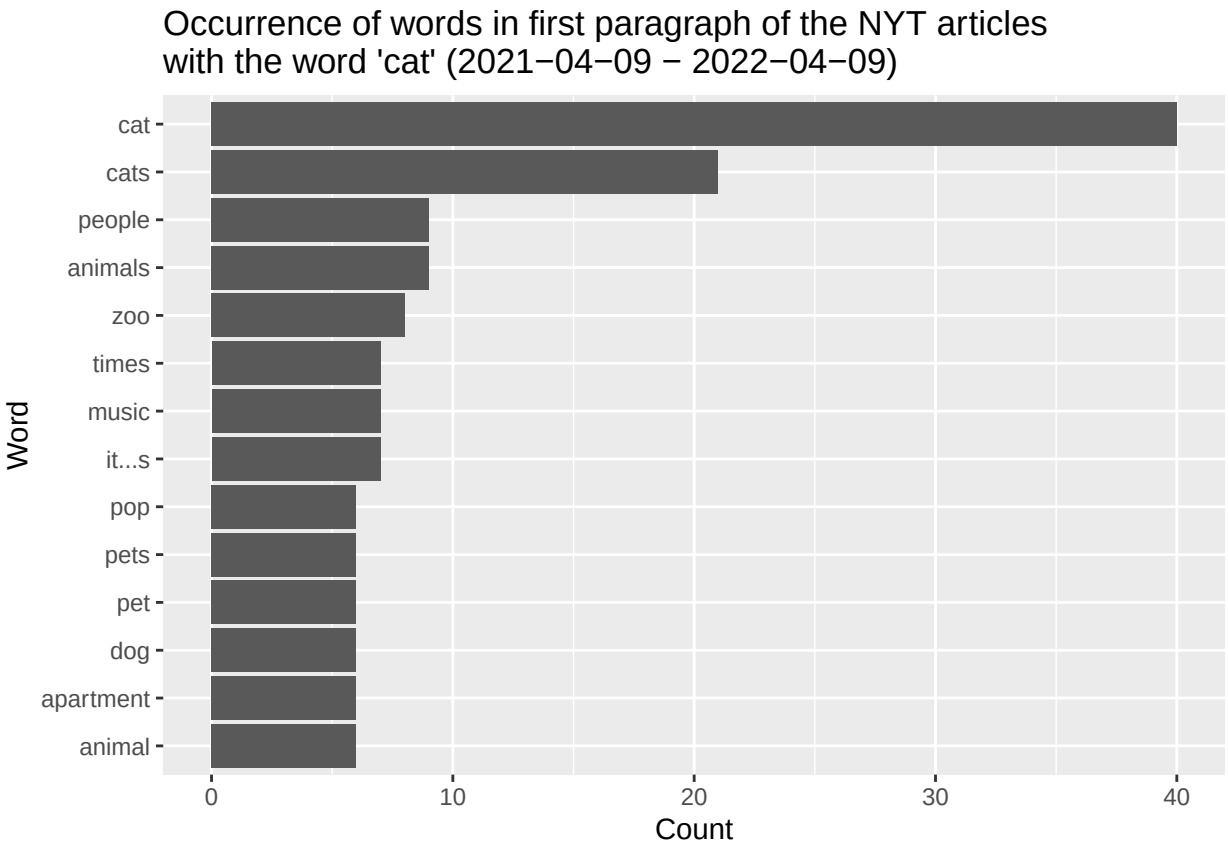



```
# determine stop words
data(stop_words)
```

```
stop_words
```

```
## # A tibble: 1,149 x 2
##    word        lexicon
##    <chr>       <chr>
##  1 a           SMART
##  2 a's         SMART
##  3 able        SMART
##  4 about       SMART
##  5 above       SMART
##  6 according   SMART
##  7 accordingly SMART
##  8 across      SMART
##  9 actually    SMART
## 10 after       SMART
## # ... with 1,139 more rows
```

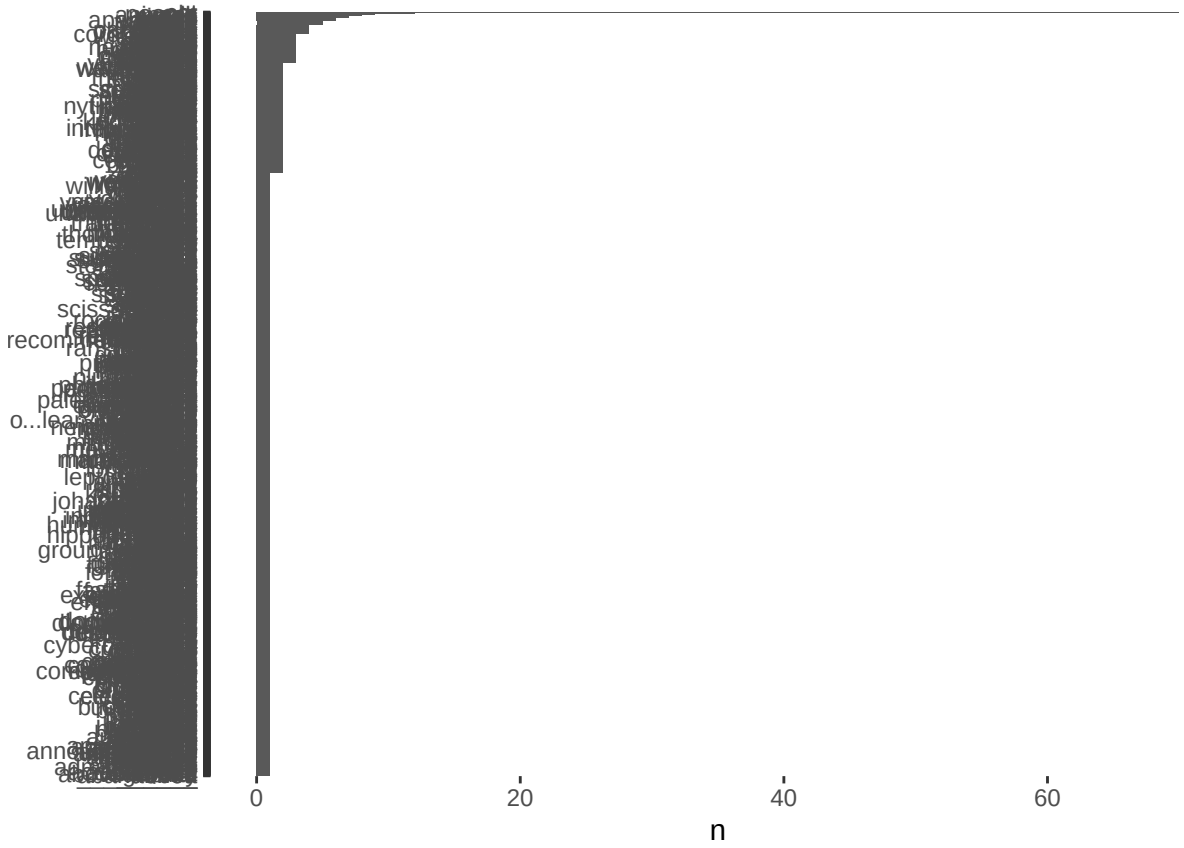
```
# remove stop words from the tokenized df
tokenized <- tokenized %>%
  anti_join(stop_words)

# repeat the same graph, but with the stop words removed and n > 5 rather than 10
tokenized %>%
  count(word, sort = TRUE) %>%
  filter(n > 5) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL) +
  xlab("Count") +
  ylab("Word") +
  ggtitle("Occurrence of words in first paragraph of the NYT articles\nwith the word 'cat' (2021-04-09 -
```



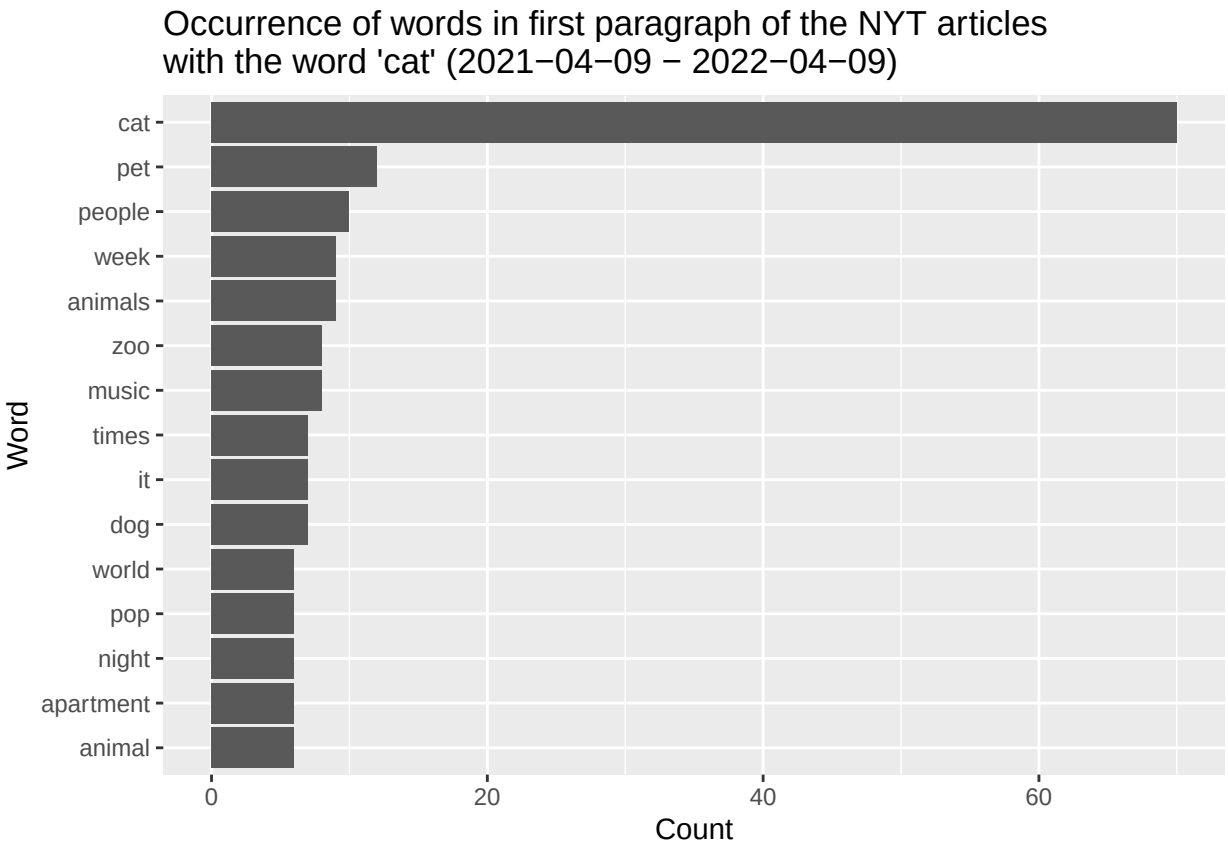
Explore the most common words

```
#inspect the list of tokens (words)
#tokenized$word
#words <- tokenized$word
#length(words) # 2063 words

clean_tokens <- str_replace_all(tokenized$word,"pet[a-z,A-Z]*","pet") # stem the word "furry" to "fur",
#clean_tokens <- str_replace_all(tokenized$word,"pet[a-z,A-Z]*","pet")

clean_tokens <- str_replace_all(clean_tokens,"cat[a-z,A-Z]*","cat") # stem the word "", which I expect

clean_tokens <- str_replace_all(clean_tokens,"sleep[a-z,A-Z]*","sleep") # stem the words "sleepy", "sle

clean_tokens <- str_remove_all(clean_tokens, "[:digit:]") # remove all numbers

clean_tokens <- gsub("'s", '', clean_tokens) # remove all 's at the end of words

tokenized$clean <- clean_tokens

tokenized %>%
 count(clean, sort = TRUE) %>%
# illegible with all the words displayed
 mutate(clean = reorder(clean, n)) %>%
 ggplot(aes(n, clean)) +
 geom_col() +
```

```
  labs(y = NULL)
```



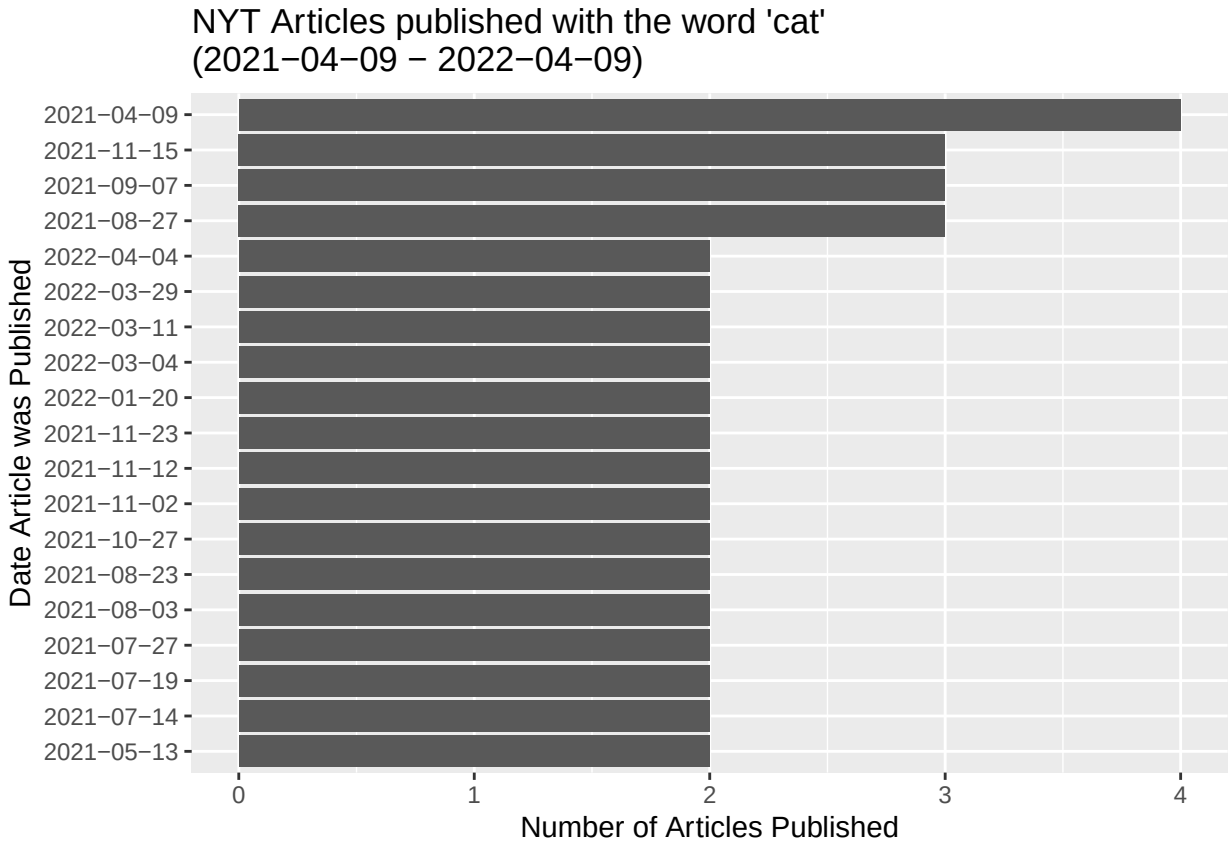

```
# remove the empty strings
tib <- subset(tokenized, clean!="")

# reassign
tokenized <- tib

# try again
tokenized %>%
  count(clean, sort = TRUE) %>%
  filter(n > 5) %>%
  mutate(clean = reorder(clean, n)) %>%
  ggplot(aes(n, clean)) +
  geom_col() +
  labs(y = NULL) +
  xlab("Count") +
  ylab("Word") +
  ggtitle("Occurrence of words in first paragraph of the NYT articles\nwith the word 'cat' (2021-04-09 -
```



The stemmed word "cat" appears the most by a large margin. The second most common word is "pet". The third most common word is "people".

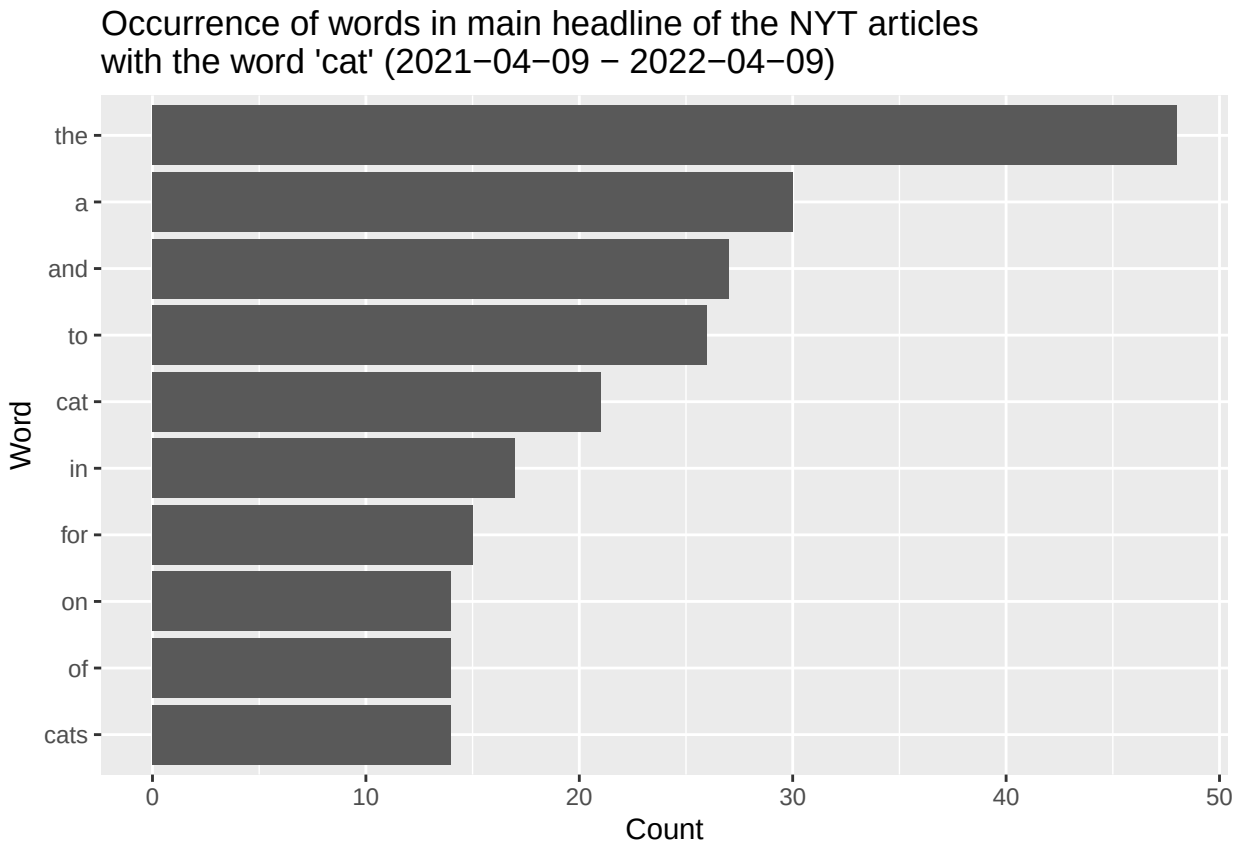
Recreate the publications per day and word frequency plots using the headlines variable (response.docs.headline.main). Compare the distributions of word frequencies between the first paragraph and headlines. Do you see any difference?

```
nytDat_cat %>%
  mutate(pubDay=gsub("T.*","",response.docs.pub_date)) %>%
  group_by(pubDay) %>%
  summarise(count=n()) %>%
  filter(count >= 2) %>%
  ggplot() +
  geom_bar(aes(x = reorder(pubDay, count), y=count), stat="identity") + coord_flip() +
  ylab("Number of Articles Published") +
  xlab("Date Article was Published") +
  ggtitle("NYT Articles published with the word 'cat'\n(2021-04-09 - 2022-04-09)")
```




```
#names(nytDat_cat)

# create a list that is the column of the headline of each article
main_headline <- names(nytDat_cat)[21]
# The 21st column, "response.docs.headline.main", is the one we want here

tokenized2 <- nytDat_cat %>%
  unnest_tokens(word, main_headline)

#tokenized2[,34]
```

```
tokenized2 %>%
  count(word, sort = TRUE) %>%
  filter(n > 10) %>% #illegible with all the words displayed
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL) +
  xlab("Count") +
  ylab("Word") +
  ggtitle("Occurrence of words in main headline of the NYT articles\nwith the word 'cat' (2021-04-09 - 1
```

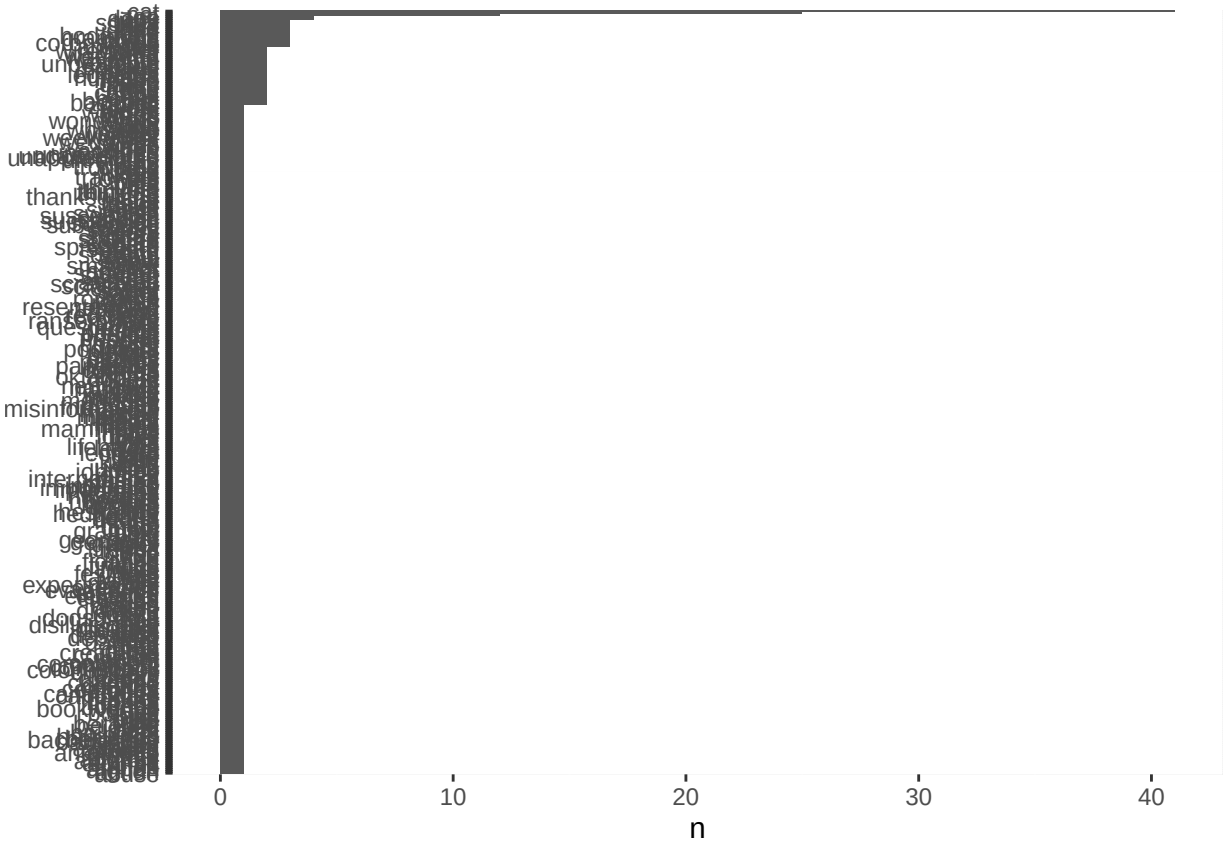



```
# determine stop words
data(stop_words)
stop_words
```

```
## # A tibble: 1,149 x 2
##    word        lexicon
##    <chr>       <chr>
##  1 a           SMART
##  2 a's         SMART
##  3 able        SMART
##  4 about       SMART
##  5 above       SMART
##  6 according   SMART
##  7 accordingly SMART
##  8 across      SMART
##  9 actually    SMART
## 10 after       SMART
## # ... with 1,139 more rows
```

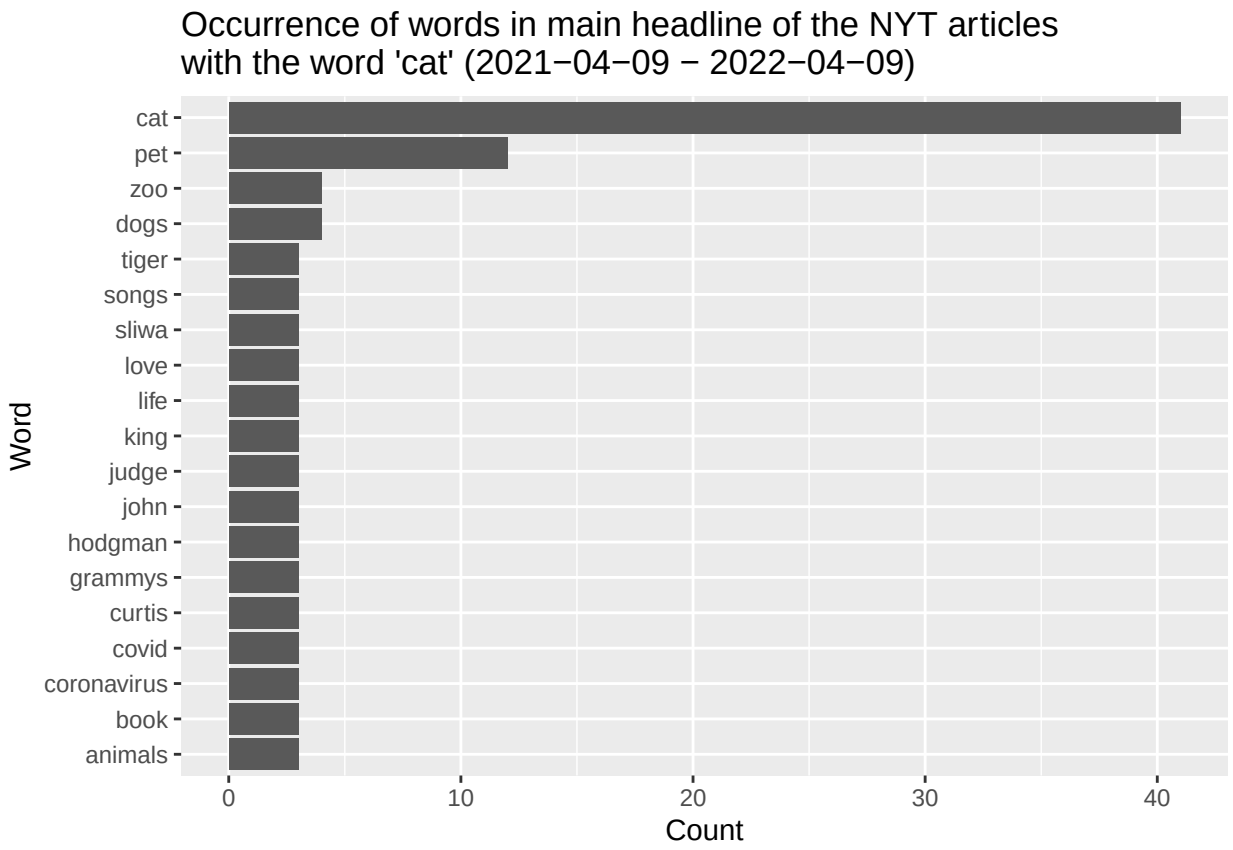
```
# remove stop words from the tokenized df
tokenized2 <- tokenized2 %>%
  anti_join(stop_words)

clean_tokens2 <- str_replace_all(tokenized2$word,"pet.*","pet") # stem the word "pets" to "pet", I expe

clean_tokens2 <- str_replace_all(clean_tokens2,"cat[a-z,A-Z]*","cat") # stem the word "cats" to "cat",

clean_tokens2 <- str_replace_all(clean_tokens2,"sleep[a-z,A-Z]*","sleep") # stem the words "sleepy", "s
```

```
clean_tokens2 <- str_remove_all(clean_tokens2, "[:digit:]") # remove all numbers

clean_tokens2 <- gsub("'s", '', clean_tokens2) # remove all 's
clean_tokens2 <- gsub("'s", '', clean_tokens2) # remove all 't

tokenized2$clean <- clean_tokens2

tokenized2 %>%
  count(clean, sort = TRUE) %>%
 #illegible with all the words displayed
  mutate(clean = reorder(clean, n)) %>%
  ggplot(aes(n, clean)) +
  geom_col() +
  labs(y = NULL)
```




```
# remove the empty strings
tib <- subset(tokenized2, clean!="")

# reassign
tokenized2 <- tib

# repeat the same graph, but with the stop words removed and n > 3 rather than 10
tokenized2 %>%
  count(clean, sort = TRUE) %>%
  filter(n > 2) %>%
  mutate(clean = reorder(clean, n)) %>%
```

```
ggplot(aes(n, clean)) +
geom_col() +
labs(y = NULL) +
xlab("Count") +
ylab("Word") +
ggtitle("Occurrence of words in main headline of the NYT articles\nwith the word 'cat' (2021-04-09 - 1
```



For the main headline, the three most common words in order from most common to least common (with stemming) are “cat”, “pet”, and “zoo”. The first two match the same most common words in the first paragraph (with stemming), but the third most common word differs, because here is it “zoo” instead of “people”. In the first paragraph, “zoo” is the sixth most common word after stemming. The most common words in the headlines seem to be more attention-grabbing, such as “covid” and “grammys”, while the words in the first paragraph are more general like “apartment”.