Topic 2: Text Data in R

Alexandra Yousefivand

4/6/2022

```
# create an object called x with the results of our query ("haaland" --> "katrina")
# the from JSON flatten the JSON object, then convert to a data frame
t <- fromJSON("http://api.nytimes.com/svc/search/v2/articlesearch.json?q=katrina&api-key=NTKBHbsb6XFEkG
```

```
class(t) #what type of object is t?
```

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

```
t <- t %>%
  data.frame()


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

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

```
dim(t) # how big is it?
```

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

```
names(t) # what variables are we working with?
```

```
##  [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.source"
##  [8] "response.docs.multimedia"
##  [9] "response.docs.keywords"
## [10] "response.docs.pub_date"
## [11] "response.docs.document_type"
## [12] "response.docs.news_desk"
## [13] "response.docs.section_name"
## [14] "response.docs.type_of_material"
## [15] "response.docs._id"
## [16] "response.docs.word_count"
```

```
## [17] "response.docs.uri"
## [18] "response.docs.print_section"
## [19] "response.docs.print_page"
## [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"
```

```
# t <- readRDS("nytDat.rds") #in case of API emergency :)
```

```
t$response.docs.snippet[9]

#assign a snippet to x to use as fodder for stringr functions.  You can follow along using the sentence

x <- "The ruin of a region and the historic city of New Orleans could not be more important, and the tar

# tolower(x)
# str_split(x, ','); str_split(x, 't')
# str_replace(x, 'historic', 'without precedent')
# str_replace(x, ' ', '_') # first one
# str_replace_all(x, ' ', '_') # how do we replace all of them?
#
# str_detect(x, 't'); str_detect(x, 'tive') ### is pattern in the string? T/F
# str_locate(x, 't'); str_locate_all(x, 'as')
```

```
term <- "Katrina" # Need to use + to string together separate words
begin_date <- "20050823" # start of Hurricane Katrina
end_date <- "20050906" # two weeks later

#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=","NTKBHbsb6XFEkGymGumAiba7n3uBvs8V", sep="")

baseurl #examine our query url
```

```
## [1] "http://api.nytimes.com/svc/search/v2/articlesearch.json?q=Katrina&begin_date=20050823&end_date=
```

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

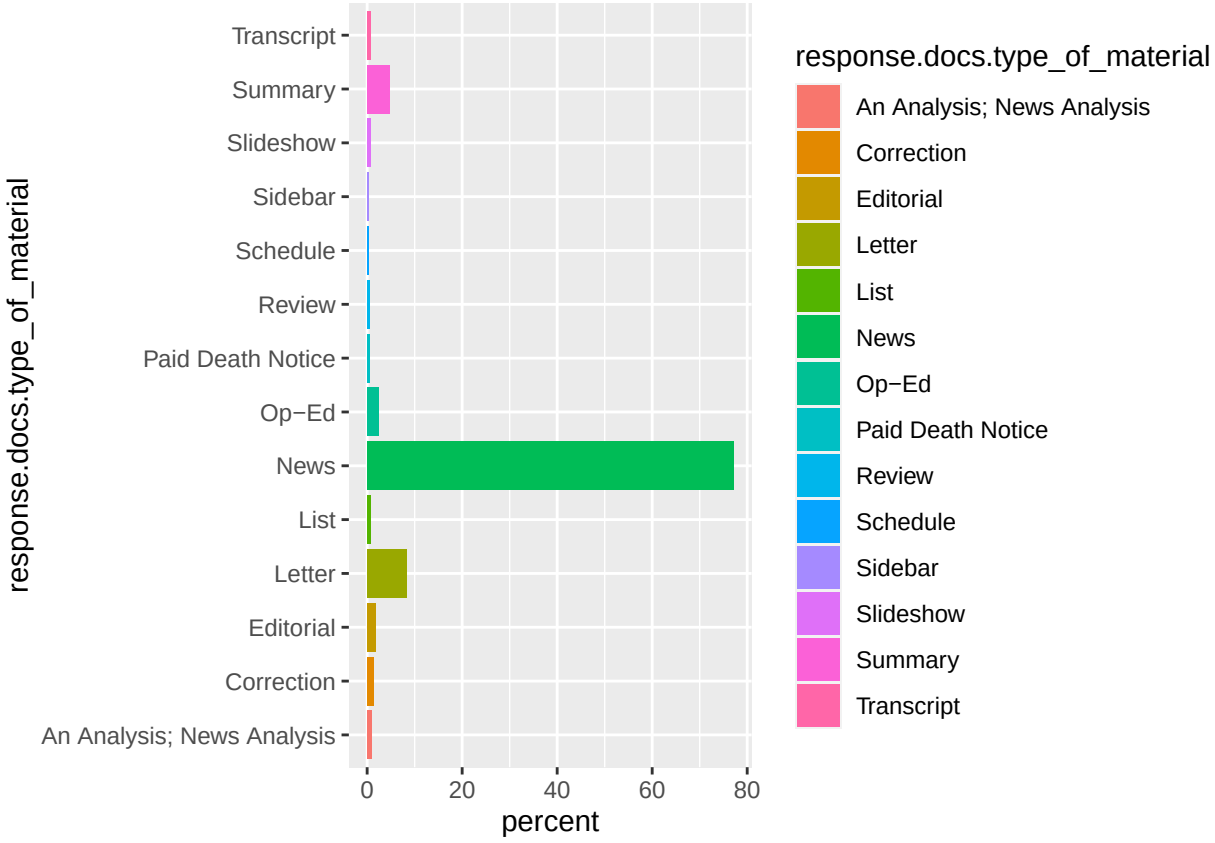
```
pages <- list()
for(i in 0:maxPages){
  nytSearch <- fromJSON(paste0(baseurl, "&page=", i), flatten = TRUE) %>% data.frame()
  message("Retrieving page ", i)
  pages[[i+1]] <- nytSearch
  Sys.sleep(6)
}
```

```
## Retrieving page 0

## Retrieving page 1

## Retrieving page 2

## Retrieving page 3

## Retrieving page 4

## Retrieving page 5

## Retrieving page 6

## Retrieving page 7

## Retrieving page 8

## Retrieving page 9

## Retrieving page 10

## Retrieving page 11

## Retrieving page 12

## Retrieving page 13

## Retrieving page 14

## Retrieving page 15

## Retrieving page 16

## Retrieving page 17

## Retrieving page 18

## Retrieving page 19
```

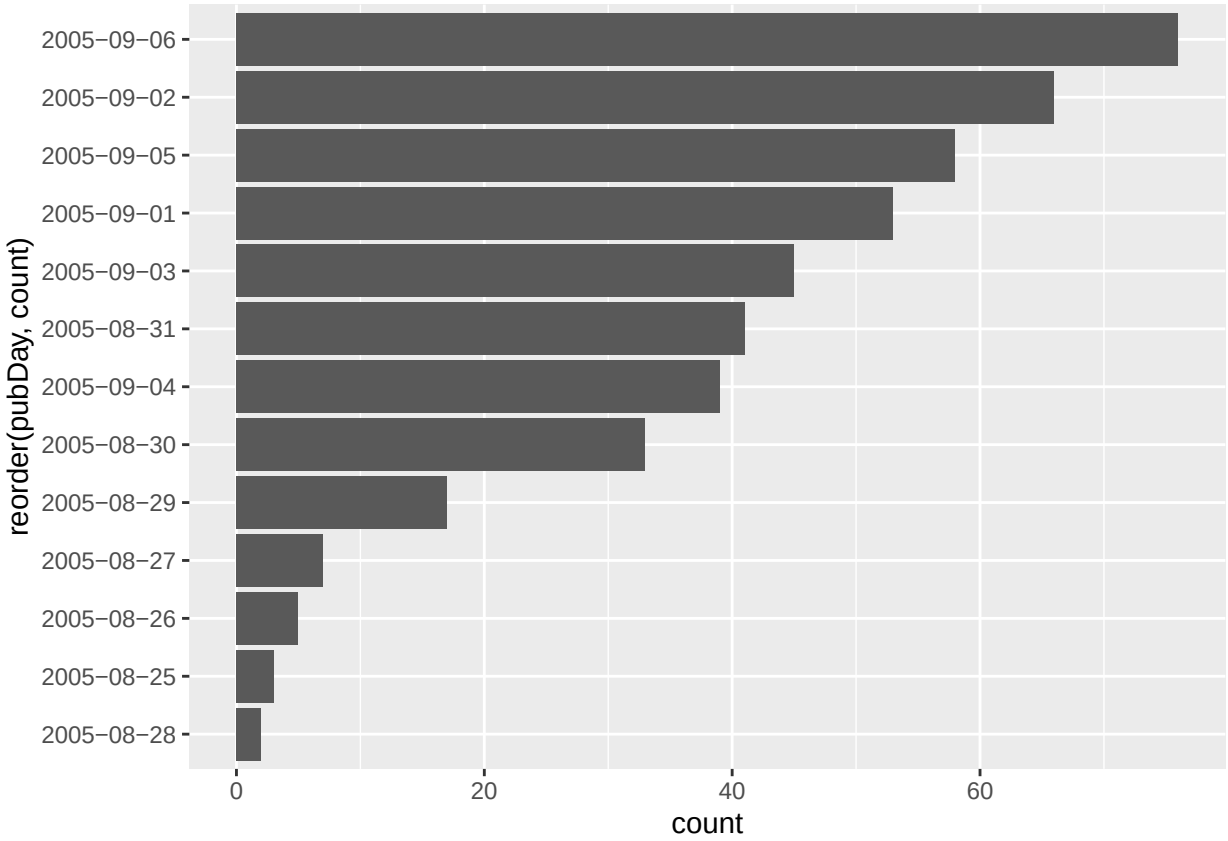
```
## Retrieving page 20

## Retrieving page 21

## Retrieving page 22

## Retrieving page 23

## Retrieving page 24

## Retrieving page 25

## Retrieving page 26

## Retrieving page 27

## Retrieving page 28

## Retrieving page 29

## Retrieving page 30

## Retrieving page 31

## Retrieving page 32

## Retrieving page 33

## Retrieving page 34

## Retrieving page 35

## Retrieving page 36

## Retrieving page 37

## Retrieving page 38

## Retrieving page 39

## Retrieving page 40

## Retrieving page 41

## Retrieving page 42

## Retrieving page 43

## Retrieving page 44
```

```
nytDat <- rbind_pages(pages)
write_csv(nytDat, "nytDat.csv")
```

```
nytDat <- read.csv("nytDat.csv") # obtained from

nytDat %>%
  group_by(response.docs.type_of_material) %>%
  summarize(count=n()) %>%
  mutate(percent = (count / sum(count))*100) %>%
  ggplot() +
  geom_bar(aes(y=percent, x=response.docs.type_of_material, fill=response.docs.type_of_material), stat =
```

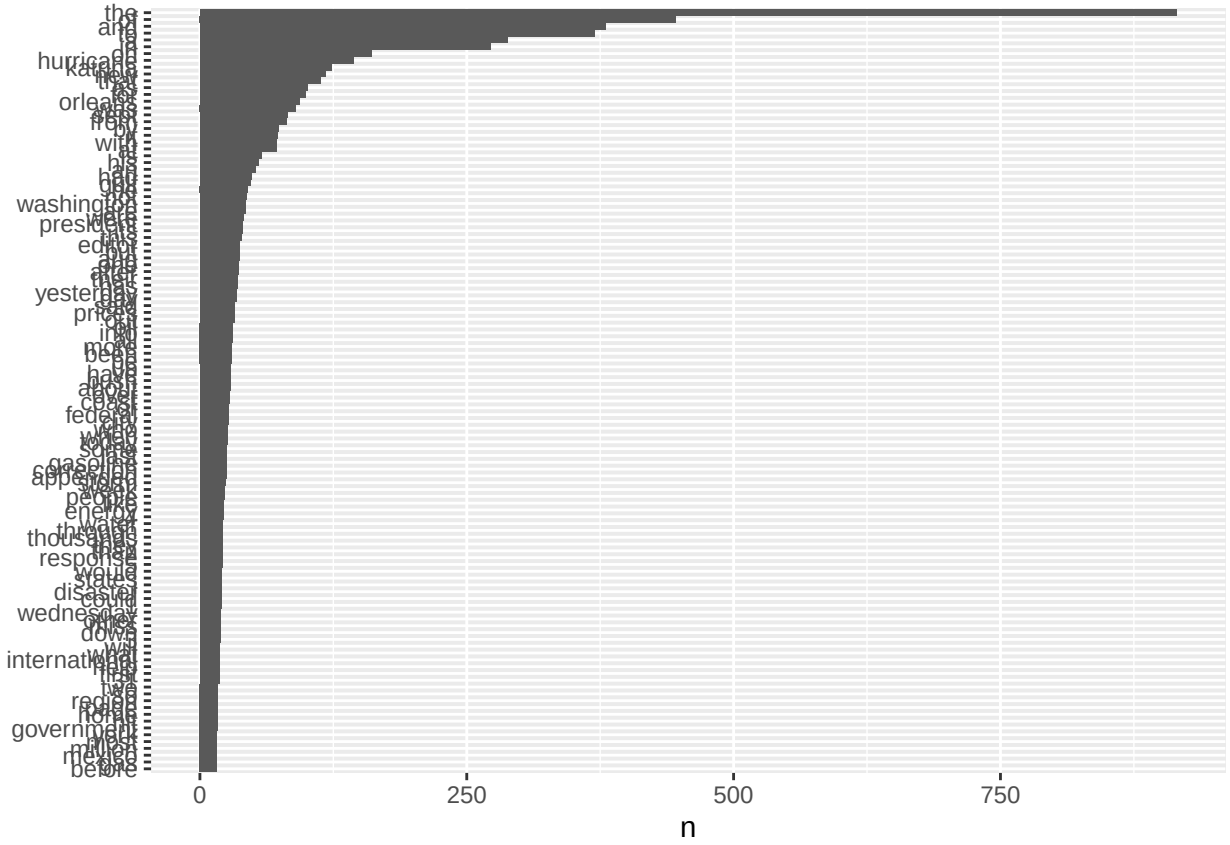



```
nytDat %>%
  mutate(pubDay=gsub("T.*","",response.docs.pub_date)) %>% # remove time component
  group_by(pubDay) %>%
  summarise(count=n()) %>%
  filter(count >= 2) %>%
  ggplot() +
  geom_bar(aes(x=reorder(pubDay, count), y=count), stat="identity") + coord_flip()
```



Paragraph

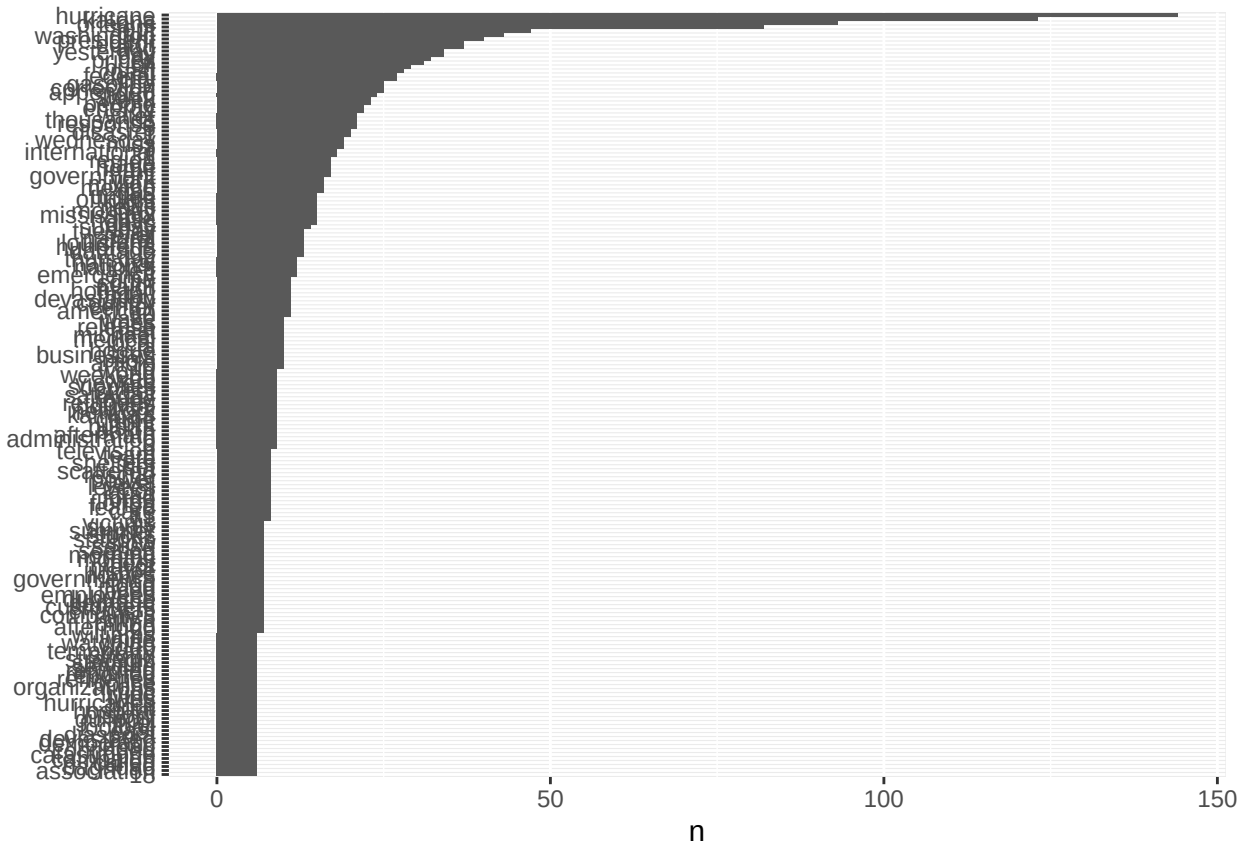
```
paragraph <- names(nytDat)[6] #The 6th column, "response.doc.lead_paragraph", is the one we want here.
tokenized <- nytDat %>%
  unnest_tokens(word, paragraph) # convert from text to tidy text format
                                 # change from paragraph to a collection of single words

tokenized %>%
  count(word, sort = TRUE) %>%
  filter(n > 15) %>% #illegible with all the words displayed
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```



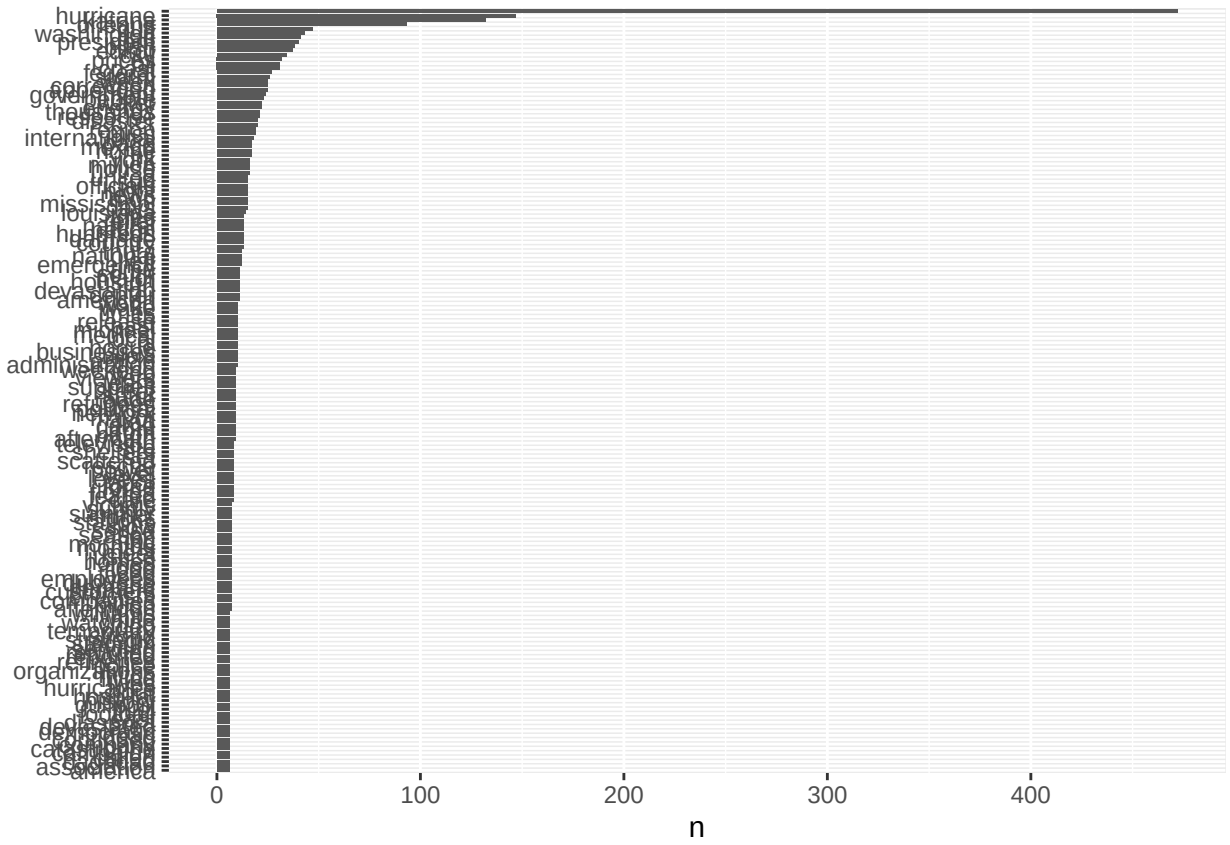

```
data(stop_words)

tokenized <- tokenized %>%
  anti_join(stop_words)
```

```
## Joining, by = "word"
```

```
tokenized %>%
  count(word, sort = TRUE) %>%
  filter(n > 5) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```



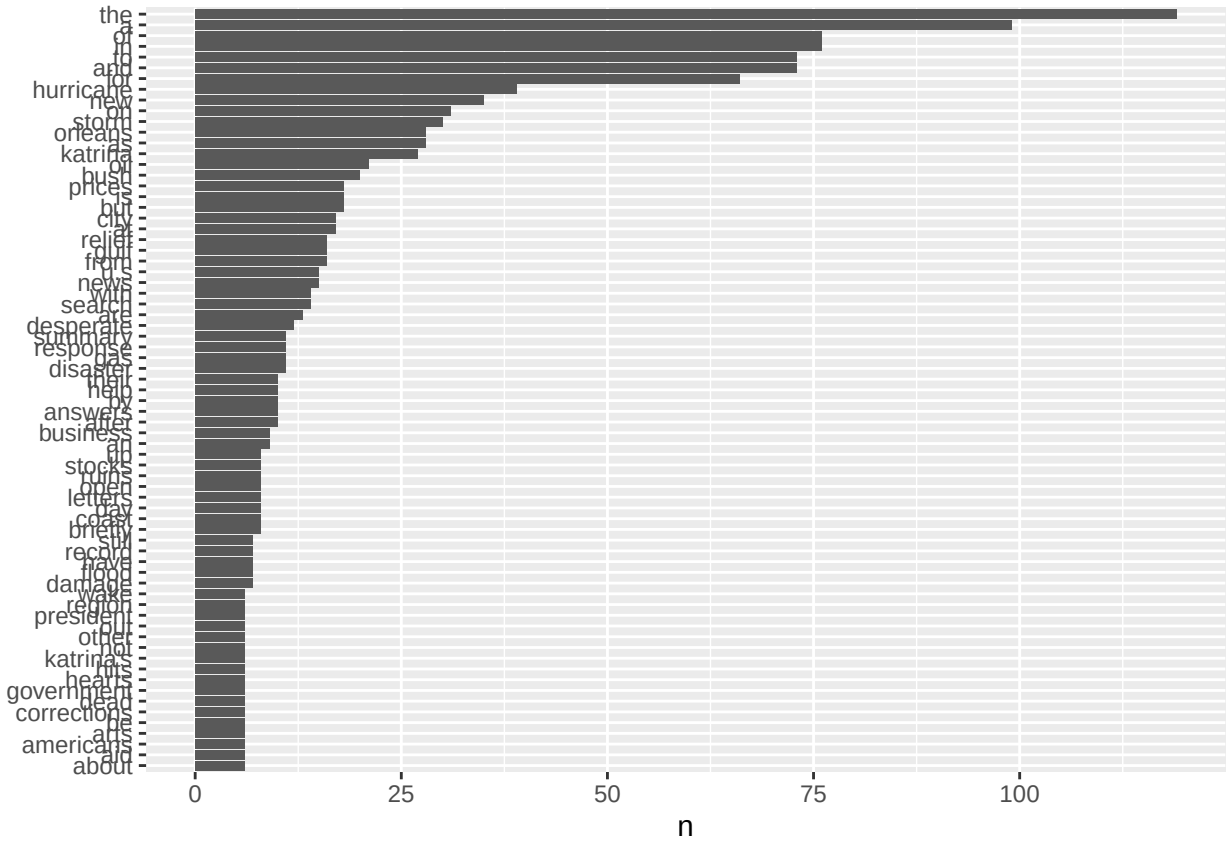

```
#inspect the list of tokens (words)
# tokenized$word

clean_tokens <- str_replace_all(tokenized$word,"hurricane*", "hurricane") %>%
                  str_replace_all("gasoline", "gas")       %>%
                  str_remove_all("[:digit:]") %>%  #remove all numbers
                  str_remove_all("day$")      %>%  #remove days of the week
                  str_remove_all("'s")        %>%  #remove Katrina's
                  str_remove_all("'s")        %>%  #remove Katrina's different font
                  str_remove_all("aug")       %>%  #remove month august
                  str_remove_all("sept")      %>%  #remove month september
                  str_remove_all("yester")    %>%  #remove yesterday
                  str_remove_all("wednes")         #remove wednesday


tokenized$clean <- clean_tokens

tokenized %>%
  count(clean, sort = TRUE) %>%
  filter(n > 5) %>% #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 <- tokenized %>%
          subset(clean!= "") %>%
          subset(clean!=".") %>%
          subset(clean!="a") %>%
          subset(clean!="katrina") %>%
          subset(clean!="week")

#reassign
tokenized <- tib

#try again
plot_paragraph <- tokenized %>%
                  count(clean, sort = TRUE) %>%
                  filter(n > 15) %>% #illegible with all the words displayed
                  mutate(clean = reorder(clean, n)) %>%
                  ggplot(aes(n, clean)) +
                    geom_col() +
                    labs(y = NULL) +
                    ggtitle("Paragraph")
```

Headline

```
headline <- names(nytDat)[20] #24th column "response.docs.headline.print_headline"
tokenized <- nytDat %>%
  unnest_tokens(word, headline) # convert from text to tidy text format
                                # change from headline to a collection of single words

tokenized %>%
  count(word, sort = TRUE) %>%
  filter(n > 5) %>% #illegible with all the words displayed
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```

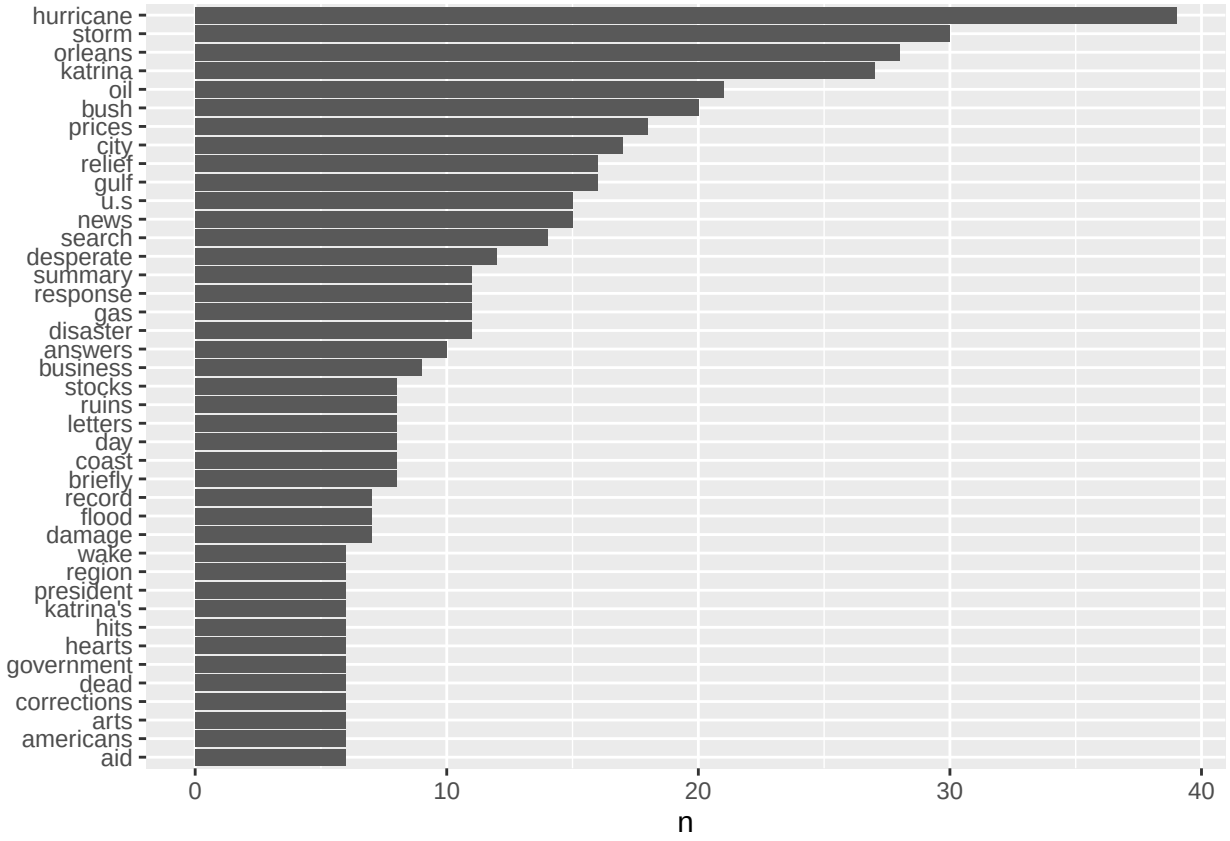



```
data(stop_words)

tokenized <- tokenized %>%
  anti_join(stop_words)
```

```
## Joining, by = "word"
```

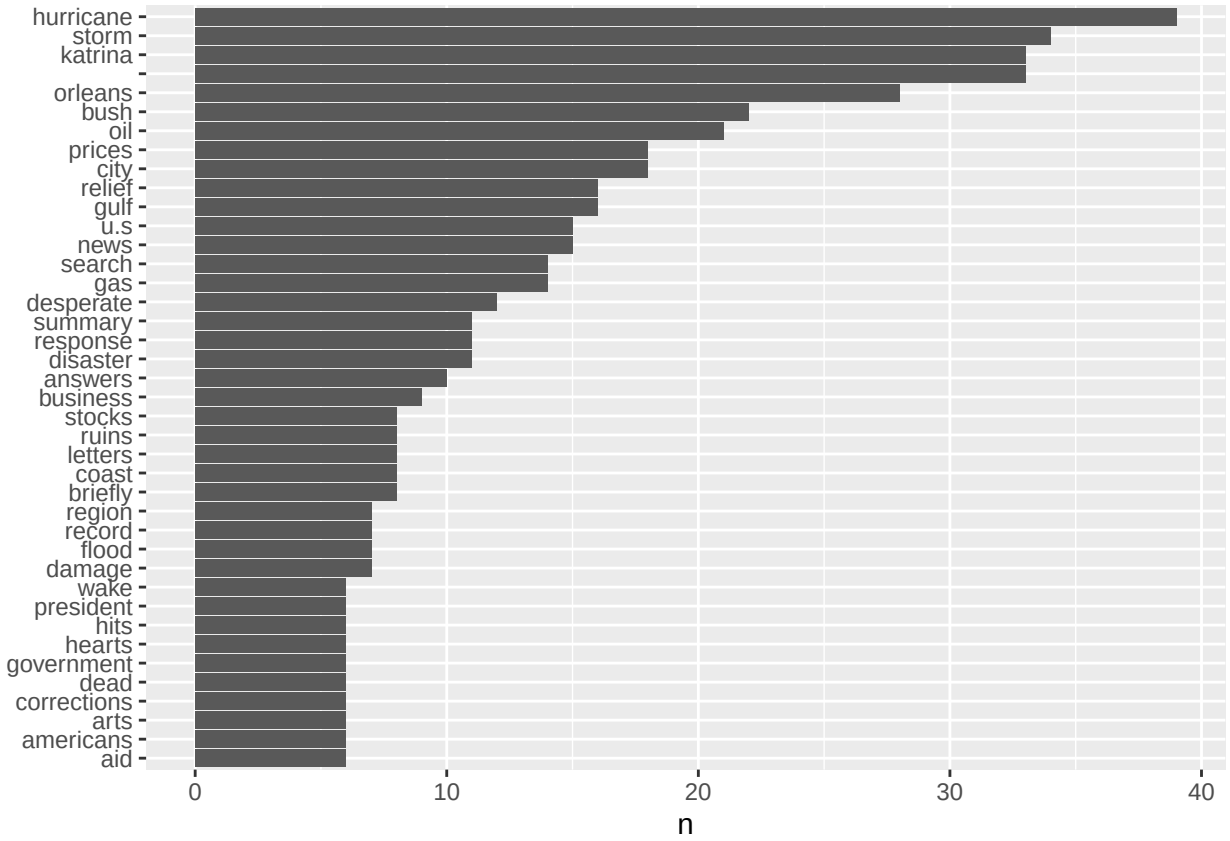
```
tokenized %>%
  count(word, sort = TRUE) %>%
  filter(n > 5) %>%
  mutate(word = reorder(word, n)) %>%
```

```
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```




```
#inspect the list of tokens (words)
# tokenized$word

clean_tokens <- str_replace_all(tokenized$word,"hurricane*", "hurricane") %>%
                  str_replace_all("gasoline", "gas")       %>%
                  str_remove_all("[:digit:]") %>%  #remove all numbers
                  str_remove_all("day$")      %>%  #remove days of the week
                  str_remove_all("'s")        %>%  #remove Katrina's
                  str_remove_all("'s")        %>%  #remove Katrina's different font
                  str_remove_all("aug")       %>%  #remove month august
                  str_remove_all("sept")      %>%  #remove month september
                  str_remove_all("yester")    %>%  #remove yesterday
                  str_remove_all("wednes")    %>%  #remove wednesday
                  str_remove_all(" ")


tokenized$clean <- clean_tokens

tokenized %>%
  count(clean, sort = TRUE) %>%
  filter(n > 5) %>% #illegible with all the words displayed
```

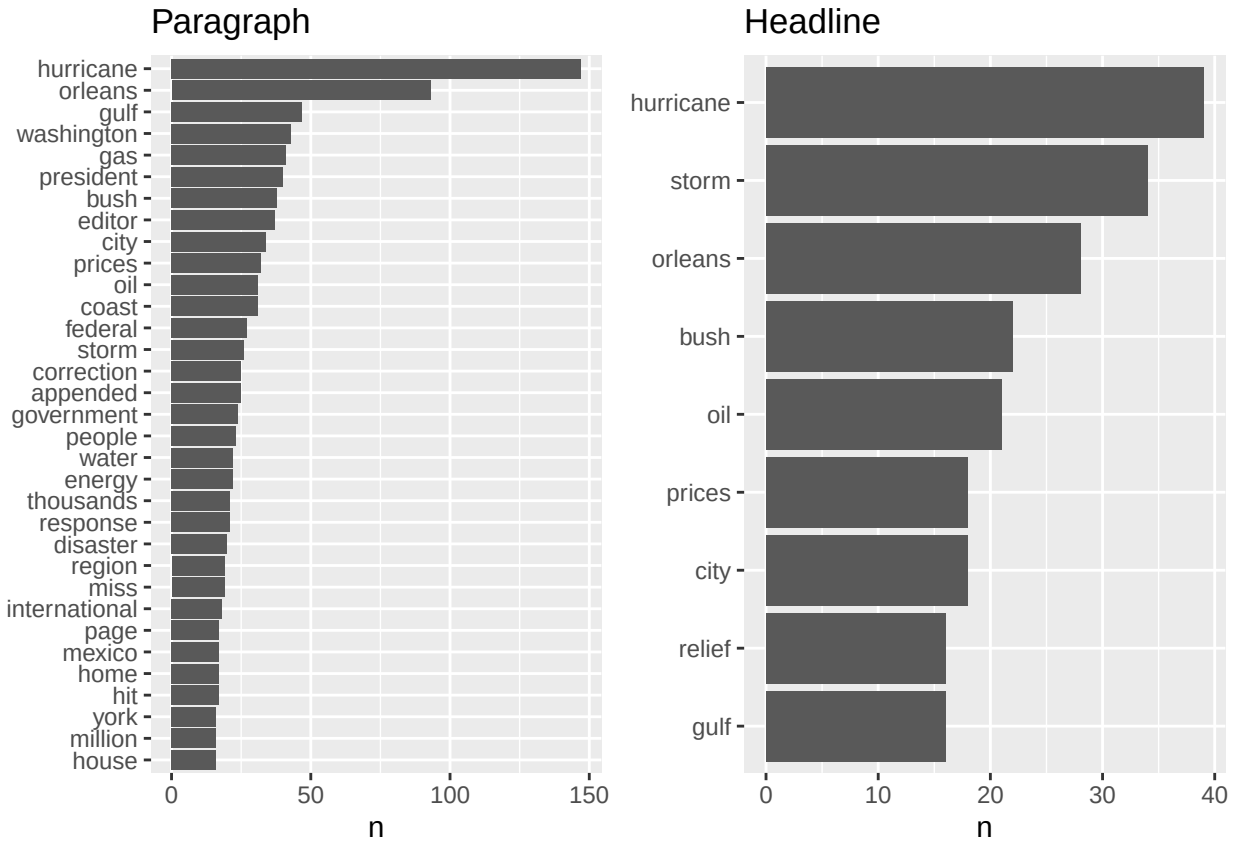
```
  mutate(clean = reorder(clean, n)) %>%
  ggplot(aes(n, clean)) +
    geom_col() +
    labs(y = NULL)
```




```
#remove the empty strings
tib <- tokenized %>%
          subset(clean!= "") %>%
          subset(clean!=".") %>%
          subset(clean!="a") %>%
          subset(clean!="katrina") %>%
          subset(clean!="week")

#reassign
tokenized <- tib

#try again
plot_headline <- tokenized %>%
                   count(clean, sort = TRUE) %>%
                   filter(n > 15) %>% #illegible with all the words displayed
                   mutate(clean = reorder(clean, n)) %>%
                   ggplot(aes(n, clean)) +
                     geom_col() +
                     labs(y = NULL) +
                     ggtitle("Headline")
```

Compare

```
ggarrange(plot_paragraph, plot_headline)
```



Many frequent paragraph words and headline words overlap. One interesting word "relief" often appears in the headlines, but not in the first paragraph.