# Xpbtid Example

Anon

2023-08-29

Load Libraries

```
library("rvest")
library("tidyverse")
library("ggplot2")
library("wordcloud")
library("tidytext")
library("tinytex")
library("syuzhet")
library("lubridate")
library("scales")
library("reshape2")
library("dplyr")
```

This scraping is getting all of the internal links

```
#Page 1 Scrape
pol_threads1 <- read_html("https://boards.4channel.org/pol/") %>%
  html_elements("a") %>%
  html_attr('href')

#Page 2 Scrape
pol_threads2 <- read_html("https://boards.4channel.org/pol/2") %>%
  html_elements("a") %>%
  html_attr('href')

#Page 3 Scrape
pol_threads3 <- read_html("https://boards.4channel.org/pol/3") %>%
  html_elements("a") %>%
  html_attr('href')

#Page 4 Scrape
pol_threads4 <- read_html("https://boards.4channel.org/pol/4") %>%
  html_elements("a") %>%
  html_attr('href')


#Page 5 Scrape
pol_threads5 <- read_html("https://boards.4channel.org/pol/5") %>%
  html_elements("a") %>%
  html_attr('href')

#Page 6 Scrape
pol_threads6 <- read_html("https://boards.4channel.org/pol/6") %>%
```

```
  html_elements("a") %>%
  html_attr('href')

#Page 7 Scrape
pol_threads7 <- read_html("https://boards.4channel.org/pol/7") %>%
  html_elements("a") %>%
  html_attr('href')

#Page 8 Scrape
pol_threads8 <- read_html("https://boards.4channel.org/pol/8") %>%
  html_elements("a") %>%
  html_attr('href')

#Page 9 Scrape
pol_threads9 <- read_html("https://boards.4channel.org/pol/9") %>%
  html_elements("a") %>%
  html_attr('href')

#Page 10 Scrape
pol_threads10 <- read_html("https://boards.4channel.org/pol/10") %>%
  html_elements("a") %>%
  html_attr('href')
```

Combining all of the threads into 1 vector.

```
df_pol <-      c(pol_threads1,
                 pol_threads2,
                 pol_threads3,
                 pol_threads4,
                 pol_threads5,
                 pol_threads6,
                 pol_threads7,
                 pol_threads8,
                 pol_threads9,
                 pol_threads10)
```

Tibble makes a table out of data from the scraped links.

```
pol_table <- tibble(txt = df_pol)
```

Choosing all of the links that look like: "4chan.org/pol/thread/this-is-a-thread/".

```
df_links <- pol_table %>%
  filter(str_detect(txt, "(thread/[0-9]{6,}[/][a-z]{1,})"))
```

Next step is appending on "https://boards.4chan.org/pol/" before the "thread/this-is-a-thread".

```
df_links$txt <- paste("https://boards.4chan.org/pol/", df_links$txt, sep = "")
```

This code will "apply" (like a loop function) the "read_html" "html_elements" and "html_attr" to each row in the data frame.

```
threads <- lapply(df_links$txt, function(x) {
  read_html(x) %>%
    html_elements("span") %>%
    html_attr("class")})
```

This gets all class elements from HTML and puts them under 1 column.

```
threads_rbind <- do.call(rbind.data.frame, threads)
threads_col <- tibble(txt = threads_rbind)
threads_col_all <- data.frame(txt = c(t(threads_col)), stringsAsFactors=FALSE)
```

Break up all of the sentences into single words. Also filters everything except: example -> "id_A1b2c45d".

```
tidy_pol_IDs <- threads_col_all %>%
  unnest_tokens(word, txt, format = "text") %>%
  filter(str_detect(word, "(id_[a-zA-z0-9]{8})"))
```

Import CSV as tidy_pol_IDs.

```
tidy_pol_IDs <- read.csv("~/Documents/Stats/4ChanScraper/Pol-IDs-Aug 29 2023 16:30:29.csv")
```

Shows the top 10 IDs, and displays number of their posts for all threads.

```
tidy_pol_sorted_IDs <- tidy_pol_IDs %>%
  count(word, sort = TRUE)
```

Useless code, but shows the bottom 10 IDs. A few hundred posters only post twice.

```
tidy_pol_bottom_10_IDs <- tidy_pol_IDs %>%
  count(word, sort = TRUE) %>%
  tail(50)
```

Counts all occurances of the unique ID showing up in threads

```
count_by_IDs <- tidy_pol_IDs %>%
  count(word, sort = TRUE) %>%
  tail(50)
```

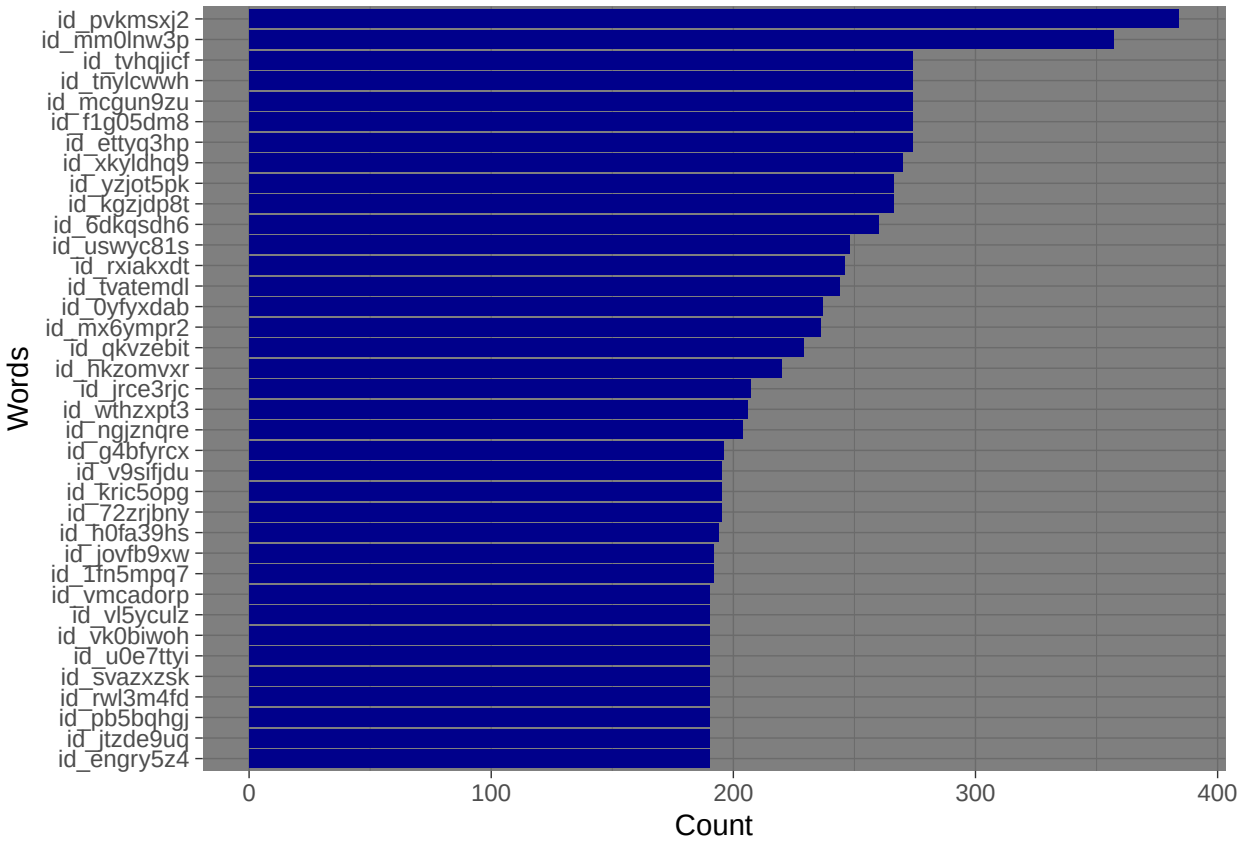
Combines all counts of each poster's post.

Example: Unique IDs posted twice, for a total of 500 times across all threads (i.e. 2bptid).

```
count_by_n <- tidy_pol_sorted_IDs %>%
  count(n, sort = TRUE)
```

Plots the biggest contributers for posts in all threads.

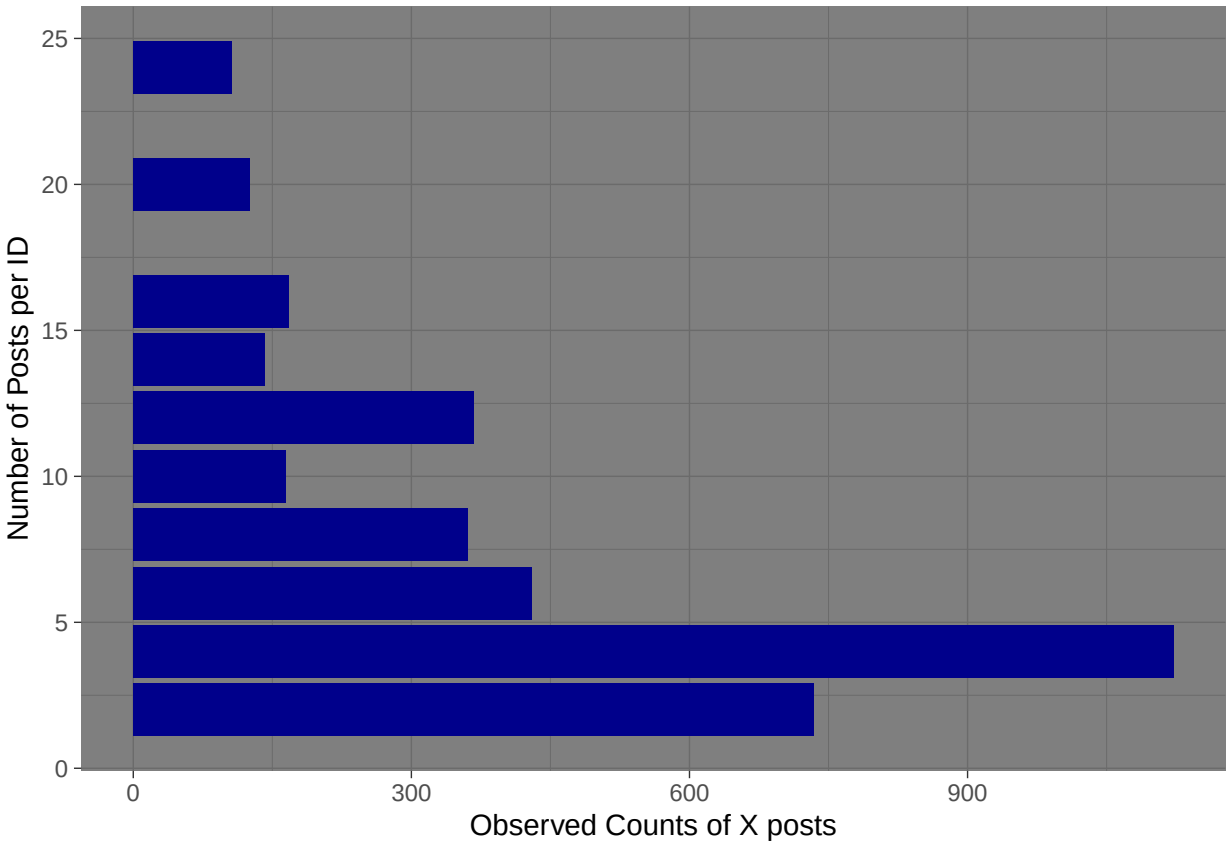
```
tidy_pol_sorted_IDs %>%
  top_n(30) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_col(fill="darkblue") +
  xlab("Words") +
  ylab("Count") +
  coord_flip() +
  theme_dark()
```




Plots total observation for X posts by this ID.

Example: plots X number of observations for 2 posts by this ID. Then it orders the top 20 frequencies starting from zero.

```
count_by_n %>%
  top_n(10) %>%
  mutate(reorder(n, nn)) %>%
  ggplot(aes(n, nn)) +
  geom_col(fill = "darkblue") +
  xlab("Number of Posts per ID") +
  ylab("Observed Counts of X posts") +
  coord_flip() +
  theme_dark()
```




Save the ID count data.

```
timestamp <- format(Sys.time(), "%b %d %Y %X")
filename <- paste0("~/Documents/Stats/4ChanScraper/Pol-IDs-",timestamp,".csv")
write.csv(tidy_pol_IDs, file = filename)
```