Applied Statistical Programming - Spring 2022

```
knitr::opts_chunk$set(echo = TRUE)
```

Problem Set 3

Due Wednesday, March 16, 10:00 AM (Before Class)

Instructions

1. The following questions should each be answered within an Rmarkdown file. Be sure to provide many comments in your code blocks to facilitate grading. Undocumented code will not be graded.
2. Work on git. Continue to work in the repository you forked from https://github.com/johnsontr/AppliedStatisticalProgramming2022 and add your code for Problem Set 4. Commit and push frequently. Use meaningful commit messages because these will affect your grade.
3. You may work in teams, but each student should develop their own Rmarkdown file. To be clear, there should be no copy and paste. Each keystroke in the assignment should be your own.
4. For students new to programming, this may take a while. Get started.

`tidyverse`

Your task in this problem set is to combine two datasets in order to observe how many endorsements each candidate received using only `dplyr` functions. Use the same Presidential primary polls that were used for the in class worksheets on February 28 and March 2.

```
# Change eval=FALSE in the code block. Install packages as appropriate.
#install.packages("fivethirtyeight")
library(fivethirtyeight)
```

```
## Warning: package 'fivethirtyeight' was built under R version 4.0.5
```

```
## Some larger datasets need to be installed separately, like senators and
## house_district_forecast. To install these, we recommend you install the
## fivethirtyeightdata package by running:
## install.packages('fivethirtyeightdata', repos =
## 'https://fivethirtyeightdata.github.io/drat/', type = 'source')
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.0.5
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.5     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
```

```
## Warning: package 'ggplot2' was built under R version 4.0.5
```

```
## Warning: package 'tibble' was built under R version 4.0.5
```

```
## Warning: package 'tidyr' was built under R version 4.0.5

## Warning: package 'purrr' was built under R version 4.0.4

## Warning: package 'dplyr' was built under R version 4.0.5

## Warning: package 'forcats' was built under R version 4.0.4

## -- Conflicts --------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
# URL to the data that you've used.
url <- 'https://jmontgomery.github.io/PDS/Datasets/president_primary_polls_feb2020.csv'
polls <- read_csv(url)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_character(),
##   question_id = col_double(),
##   poll_id = col_double(),
##   cycle = col_double(),
##   pollster_id = col_double(),
##   sponsor_ids = col_number(),
##   pollster_rating_id = col_double(),
##   sample_size = col_double(),
##   sponsor_candidate = col_logical(),
##   internal = col_logical(),
##   partisan = col_logical(),
##   tracking = col_logical(),
##   nationwide_batch = col_logical(),
##   candidate_id = col_double(),
##   pct = col_double()
## )
## i Use `spec()` for the full column specifications.

## Warning: 122 parsing failures.
##  row      col           expected actual
## 1394 partisan 1/0/T/F/TRUE/FALSE    DEM 'https://jmontgomery.github.io/PDS/Datasets/president_primary
## 1395 partisan 1/0/T/F/TRUE/FALSE    DEM 'https://jmontgomery.github.io/PDS/Datasets/president_primary
## 1396 partisan 1/0/T/F/TRUE/FALSE    DEM 'https://jmontgomery.github.io/PDS/Datasets/president_primary
## 1397 partisan 1/0/T/F/TRUE/FALSE    DEM 'https://jmontgomery.github.io/PDS/Datasets/president_primary
## 1398 partisan 1/0/T/F/TRUE/FALSE    DEM 'https://jmontgomery.github.io/PDS/Datasets/president_primary
## .... ........ .................. ...... ..................................................
## See problems(...) for more details.
```

```
Endorsements <- endorsements_2020 # from the fiverthirtyeight package
```

First, create two new objects `polls` and `Endorsements`. Then complete the following.

- Change the `Endorsements` variable name endorsee to `candidate_name`.
- Change the `Endorsements` dataframe into a `tibble` object.
- Filter the `poll` variable to only include the following 6 candidates: Amy Klobuchar, Bernard Sanders,Elizabeth Warren, Joseph R. Biden Jr., Michael Bloomberg, Pete Buttigieg **and** subset the dataset to the following five variables: `candidate_name, sample_size, start_date, party, pct`

- Compare the candidate names in the two datasets and find instances where the a candidates name is spelled differently i.e. Bernard vs. Bernie. Using only `dplyr` functions, make these the same across datasets.
- Now combine the two datasets by candidate name using `dplyr` (there will only be five candidates after joining).
- Create a variable which indicates the number of endorsements for each of the five candidates using `dplyr`.
- Plot the number of endorsement each of the 5 candidates have using `ggplot()`. Save your plot as an object `p`.
- Rerun the previous line as follows: `p + theme_dark()`. Notice how you can still customize your plot without rerunning the plot with new options.
- Now, using the knowledge from the last step change the label of the X and Y axes to be more informative, add a title. Save the plot in your forked repository.

```
#Change variable name, using rename function and store it in our original df
Endorsements <- Endorsements %>% rename(candidate_name = endorsee)

#Change to a tibble and check the class
Endorsements <- Endorsements %>% as_tibble()
class(Endorsements)
```

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

```
#Filter to the candidates with an in command of a vector of the candidates of interest
polls <- polls %>%
  filter(polls$candidate_name %in% c("Amy Klobuchar","Bernard Sanders", "Elizabeth Warren", "Joseph R. B
  select(candidate_name, sample_size, start_date, party, pct)
#build a table based on the candidate name to confirm only those 6 remain in our new object
table(polls$candidate_name)
```

```
##
##       Amy Klobuchar     Bernard Sanders    Elizabeth Warren Joseph R. Biden Jr.
##                 874                 960                  963                 971
##   Michael Bloomberg      Pete Buttigieg
##                 232                 890
```

```
#use summary to see that there are only 5 remaining variables as told to do
summary(polls)
```

```
##  candidate_name      sample_size     start_date          party
##  Length:4890        Min.   :   14   Length:4890        Length:4890
##  Class :character   1st Qu.:  447   Class :character   Class :character
##  Mode  :character   Median :  590   Mode  :character   Mode  :character
##                     Mean   : 1649
##                     3rd Qu.: 1002
##                     Max.   :17836
##       pct
##  Min.   : 0.00
##  1st Qu.: 5.00
##  Median :13.00
##  Mean   :14.33
##  3rd Qu.:21.00
##  Max.   :82.00
```

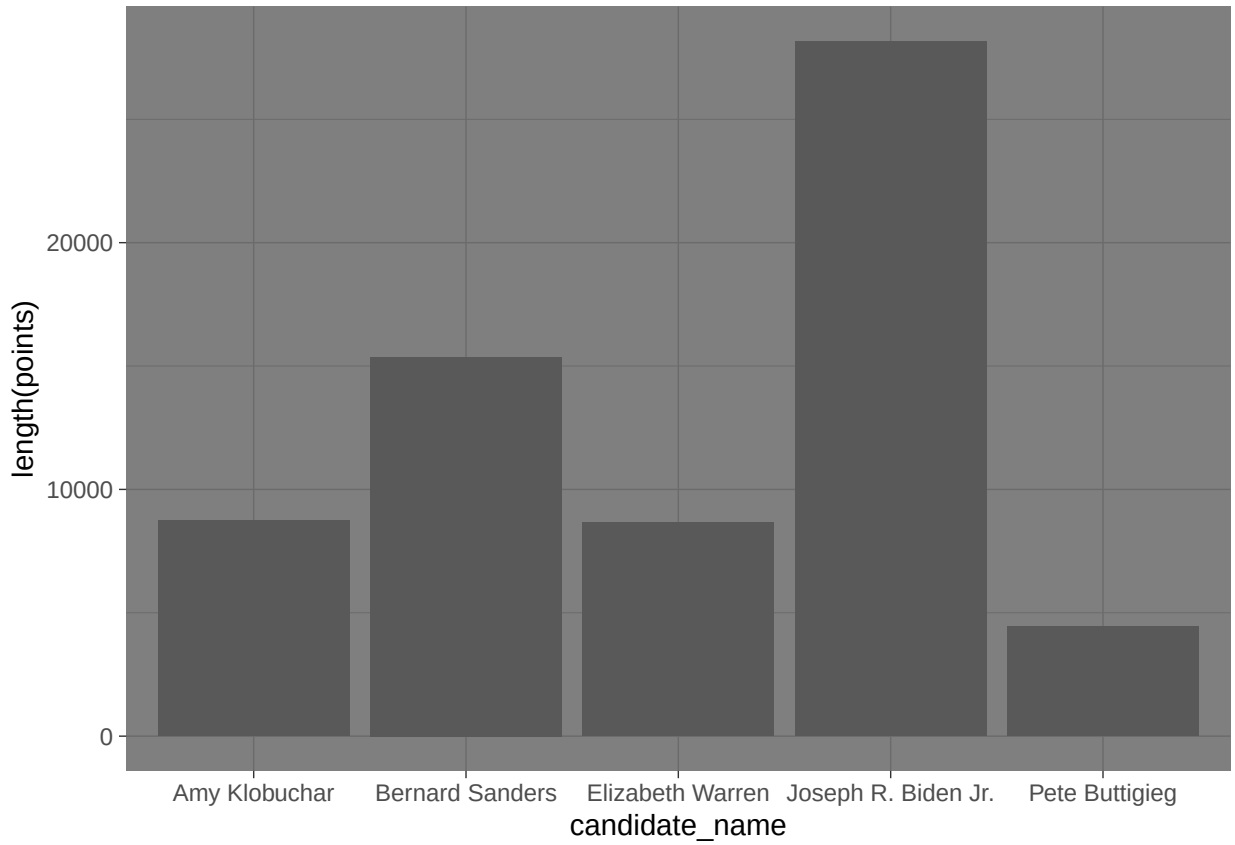
```r
#Check spelling of names in our Endorsements dataset
table(Endorsements$candidate_name)
```

```
##
##      Amy Klobuchar     Bernie Sanders      Beto O'Rourke        Cory Booker
##                 10                 16                  7                 20
##   Elizabeth Warren      Eric Swalwell         Jay Inslee          Joe Biden
##                  9                  1                  4                 29
##       John Delaney  John Hickenlooper      Julian Castro      Kamala Harris
##                  2                  1                  4                 31
## Kirsten Gillibrand     Pete Buttigieg      Steve Bullock
##                  1                  5                  3
```

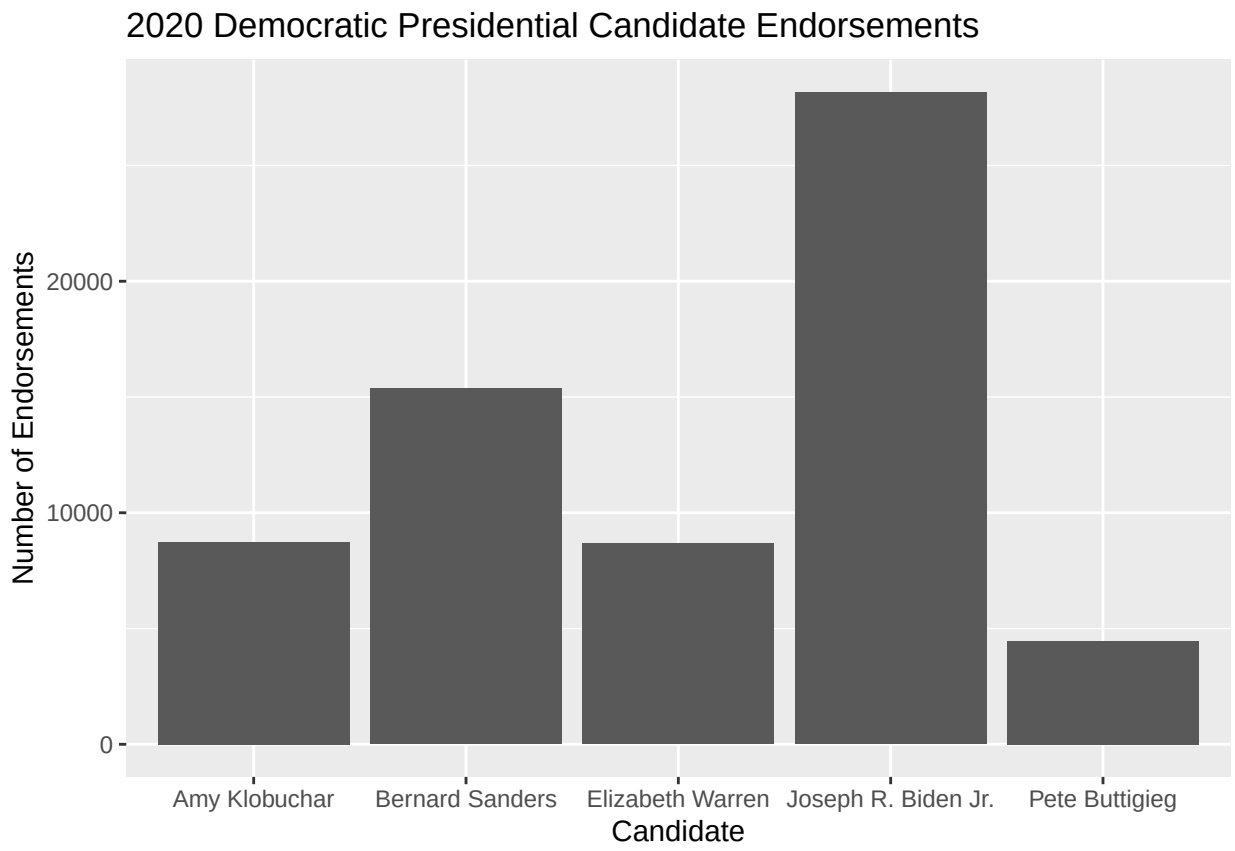
```r
#Here we can tell Amy Klobuchar, Elizabeth Warren, and Pete Buttigieg are spelled the same. Biden is "J

#Now we need to change these using recode
Endorsements <- Endorsements %>%
  mutate(candidate_name=recode(candidate_name, "Bernie Sanders" = "Bernard Sanders", "Joe Biden" = "Jos

#Combine them into a dataset using an inner join by candidate name, then verify only the 5 expected rem
pe_combined <- polls %>%
  inner_join(Endorsements, by="candidate_name")
table(pe_combined$candidate_name)
```

```
##
##       Amy Klobuchar     Bernard Sanders    Elizabeth Warren Joseph R. Biden Jr.
##                8740               15360                8667               28159
##      Pete Buttigieg
##                4450
```

```r
#Count the number of endorsements for each of the five
wide_pe_combined <- pe_combined %>%
  group_by(candidate_name) %>%
  summarize(length(points))

p <- ggplot(data = wide_pe_combined) + geom_col(aes(candidate_name,`length(points)`))

p + theme_dark()
```




```
p + labs(y = "Number of Endorsements", x = "Candidate", title = "2020 Democratic Presidential Candidate
```




```
ggsave("candidate_endorsements_2020.pdf",
       plot = last_plot())
```

```
## Saving 6.5 x 4.5 in image
```

Text-as-Data with `tidyverse`

For this question you will be analyzing Tweets from President Trump for various characteristics. Load in the following packages and data:

```
# Change eval=FALSE in the code block. Install packages as appropriate.
library(tidyverse)
#install.packages('tm')
library(tm)
```

```
## Warning: package 'tm' was built under R version 4.0.5
```

```
## Loading required package: NLP
```

```
##
## Attaching package: 'NLP'
```

```
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
#install.packages('lubridate')
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 4.0.5
##
## Attaching package: 'lubridate'
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
#install.packages('wordcloud')
library(wordcloud)
## Warning: package 'wordcloud' was built under R version 4.0.5
## Loading required package: RColorBrewer
library(knitr)
## Warning: package 'knitr' was built under R version 4.0.5
library(broom)
library(tidytext)
## Warning: package 'tidytext' was built under R version 4.0.5
trump_tweets_url <- 'https://politicaldatascience.com/PDS/Datasets/trump_tweets.csv'
tweets <- read_csv(trump_tweets_url)
##
## -- Column specification --------------------------------------------------------
## cols(
##   source = col_character(),
##   text = col_character(),
##   created_at = col_character(),
##   retweet_count = col_double(),
##   favorite_count = col_double(),
##   is_retweet = col_logical()
## )
```

- First separate the `created_at` variable into two new variables where the date and the time are in separate columns. After you do that, then report the range of dates that is in this dataset.
- Using `dplyr` subset the data to only include original tweets (remove retweets) and show the text of the President's **top 5** most popular and most retweeted tweets. (Hint: The `match` function can help you find the index once you identify the largest values.)
- Create a *corpus* of the tweet content and put this into the object `Corpus` using the `tm` (text mining) package. (Hint: Do the assigned readings.)
- Remove extraneous whitespace, remove numbers and punctuation, convert everything to lower case and remove 'stop words' that have little substantive meaning (the, a, it).
- Now create a `wordcloud` to visualize the top 50 words the President uses in his tweets. Use only words that occur at least three times. Display the plot with words in random order and use 50 random colors. Save the plot into your forked repository.
- Create a *document term matrix* called `DTM` that includes the argument `control = list(weighting = weightTfIdf)`
- Finally, report the 50 words with the the highest tf.idf scores using a lower frequency bound of .8.

```r
#library a package necessary for date conversion
library(anytime)
```

```
## Warning: package 'anytime' was built under R version 4.0.5
```

```r
#split the data using mutate and str_split
created_split <- tweets %>% mutate(date_created = str_split(created_at, pattern = " "))
#convert it to the correct col/row orientation
time <- t(as.data.frame(created_split$date_created))
#name columns
colnames(time) <- c("date","time")
#combine into our original dataset
tweets <- cbind(tweets,time)
#convert to date format for finding the oldest and newest
tweets$date <- anydate(tweets$date)
#oldest tweet day
min(tweets$date)
```

```
## [1] "2014-01-01"
```

```r
#newest tweet day
max(tweets$date)
```

```
## [1] "2020-02-14"
```

```r
#So the range of dates is 1/1/2014 to 2/14/2020

#Now to subset only to tweets
tweets_only <- tweets %>% filter(is_retweet == FALSE)
#verify our result
table(tweets_only$is_retweet)
```

```
##
## FALSE
## 30199
```

```r
#get the most popular
most_liked <- tweets_only %>%
  slice_max(favorite_count, n=5)
most_liked$text
```

```
## [1] "A$AP Rocky released from prison and on his way home to the United States from Sweden. It was a l
## [2] "https://t.co/VXeKiVzpTf"
## [3] "All is well! Missiles launched from Iran at two military bases located in Iraq. Assessment of ca
## [4] "MERRY CHRISTMAS!"
## [5] "Kobe Bryant despite being one of the truly great basketball players of all time was just gettin
```

```r
#Get the most retweeted
most_retweeted <- tweets_only %>%
  slice_max(retweet_count, n=5)
most_retweeted$text
```

```
## [1] "#FraudNewsCNN #FNN https://t.co/WYUnHjjUjg"
## [2] "TODAY WE MAKE AMERICA GREAT AGAIN!"
## [3] "Why would Kim Jong-un insult me by calling me \"old\" when I would NEVER call him \"short and fa
## [4] "A$AP Rocky released from prison and on his way home to the United States from Sweden. It was a l
## [5] "Such a beautiful and important evening! The forgotten man and woman will never be forgotten aga
```

```r
#Now for the corpus
#Create the corpus
Corpus <- VCorpus(VectorSource(tweets_only$text))
#check that it is working
writeLines(head(strwrap(Corpus[[420]]),10))
```

```
## Amazon Calls 'Profiles in Corruption' the 'Most Anticipated' Nonfiction
## Book https://t.co/sOQcs3OHS4 via @BreitbartNews
```

```r
#Now to clean the tweets; I'm splitting it for processing power, but keeping some together for efficien
#first I'll remove all URLs
removepattern <- content_transformer(function(x,pattern){
  return(gsub(pattern,"",x))
})

Corpus <- tm_map(Corpus, removepattern, "?(f|ht)(tp)(s?)(://)(.*)(.|/])(.*)")

#Next, I will make transform everything to lower case, remove whitespace, and stopwords
Corpus <- Corpus %>%
  tm_map(content_transformer(tolower)) %>%
  tm_map(stripWhitespace) %>%
  tm_map(removeWords, stopwords("english"))

#next I'll remove punctuation and numbers
Corpus <- Corpus %>%
  tm_map(removePunctuation) %>%
  tm_map(removeNumbers)

#Test
writeLines(head(strwrap(Corpus[[420]]),10))
```

```
## amazon calls profiles corruption anticipated nonfiction book
```

```r
#Make the word cloud with random color and order
set.seed(125)
wc <- wordcloud(Corpus, min.freq = 3,
                random.order = T,
                random.color = T,
                max.words = 50,
                colors = sample(colors()))
```

```
## Warning in wordcloud(Corpus, min.freq = 3, random.order = T, random.color = T, :
## realdonaldtrump could not be fit on page. It will not be plotted.
```


```
#Document Term matrix
dtm <- DocumentTermMatrix(Corpus, control = list(weighting = weightTfIdf, global=c(0.8,Inf)))
```

```
## Warning in weighting(x): empty document(s): 20 97 107 111 118 122 126 128 129
## 134 143 192 193 194 195 198 199 204 227 230 232 233 244 245 255 275 295 312 313
## 315 368 384 401 427 453 456 457 458 463 479 480 482 528 529 542 555 565 660 661
## 683 684 685 687 692 741 769 784 785 795 829 832 833 897 945 946 979 1006 1041
## 1049 1051 1054 1062 1065 1089 1090 1091 1105 1114 1116 1139 1145 1160 1172 1173
## 1181 1189 1192 1193 1195 1199 1207 1208 1209 1211 1213 1215 1216 1234 1235 1236
## 1251 1253 1258 1259 1260 1261 1262 1328 1378 1379 1382 1383 1384 1393 1457 1477
## 1579 1599 1711 1811 1812 1824 1841 1850 1907 1924 2000 2002 2004 2019 2020 2027
## 2032 2040 2045 2092 2094 2138 2153 2170 2201 2204 2205 2206 2252 2253 2326 2327
## 2346 2385 2386 2403 2447 2489 2490 2513 2514 2515 2528 2539 2541 2568 2578 2584
## 2611 2676 2693 2696 2718 2720 2734 2735 2737 2761 2809 2818 2826 2827 2828 2831
## 2832 2833 2888 2889 2904 2905 2907 2910 2926 2928 2945 2955 2959 2963 2973 2984
## 2985 2987 2989 2990 2999 3020 3021 3064 3136 3207 3211 3283 3292 3319 3323 3324
## 3372 3429 3555 3566 3575 3627 3633 3794 3902 3903 3917 4077 4164 4169 4171 4201
## 4263 4270 4287 4289 4330 4334 4342 4343 4384 4400 4402 4427 4468 4530 4700 4701
## 4789 4837 4845 4863 4871 4897 4912 4973 4974 4979 4984 4998 5058 5078 5191 5192
## 5199 5200 5263 5264 5298 5381 5417 5421 5464 5508 5697 5703 5970 6214 6224 6227
## 6255 6374 6460 6506 6507 6515 6516 6521 6524 6525 6544 6545 6560 6561 6563 6575
## 6577 6590 6596 6597 6600 6603 6632 6634 6657 6659 6661 6662 6664 6675 6692 6767
## 6859 6883 6884 6971 7003 7004 7019 7020 7022 7034 7035 7054 7056 7108 7138 7197
## 7271 7282 7296 7308 7353 7380 7430 7449 7473 7475 7850 8037 8152 8162 8223 8224
## 8225 8226 8275 8284 8294 8295 8296 8308 8320 8321 8360 8378 8389 8394 8410 8417
## 8429 8430 8436 8451 8453 8464 8474 8488 8542 8560 8671 8718 8758 8770 8771 8772
```

```
## 8778 8779 8781 8782 8804 8822 8826 8835 8899 8946 9046 9063 9209 9270 9424 9511
## 9577 9619 10002 10098 12142 15738 15780 16492 17385 17532 20122 20747 23819
## 26332
```

```
#Get the top 50:
#Start by turning it into a data frame
dtm <- tidy(dtm)
#calculate the scores
dtm_tfidf <- bind_tf_idf(dtm, term = term, document = document, n = count)
top50_terms <- dtm_tfidf %>%
  group_by(term) %>%
  filter(tf_idf == max(tf_idf)) %>%
  ungroup() %>%
  slice_max(tf_idf, n=50)

head(top50_terms)
```

```
## # A tibble: 6 x 6
##   document term              count    tf   idf tf_idf
##   <chr>    <chr>             <dbl> <dbl> <dbl>  <dbl>
## 1 519      winred             14.9     1  10.3   10.3
## 2 578      antibenghazi       14.9     1  10.3   10.3
## 3 580      antibengahzi       14.9     1  10.3   10.3
## 4 1249     newhoaxsameswamp   14.9     1  10.3   10.3
## 5 1555     iranintlar         14.9     1  10.3   10.3
## 6 2493     donothingdemocrats 14.9     1  10.3   10.3
```