Applied Statistical Programming - Spring 2022

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.

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.

```
#Use rename function to change variable name
endorsements <- rename(Endorsements, candidate_name=endorsee)
#as_tibble  convert the data frame into tibble
endorsements <- as_tibble(endorsements)
#Endorsements is a tibble
class(endorsements)
```

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

```
#Use filter to subset the polls data and use select to only store the variables we are interested
subset_polls <- polls %>%  filter(candidate_name %in% c("Amy Klobuchar", "Bernard Sanders", "Elizabeth

distinct(subset_polls, candidate_name) #Unique values in candidate_name
```

```
## # A tibble: 6 x 1
##   candidate_name
##   <chr>
## 1 Bernard Sanders
## 2 Pete Buttigieg
## 3 Joseph R. Biden Jr.
## 4 Amy Klobuchar
## 5 Elizabeth Warren
## 6 Michael Bloomberg
```

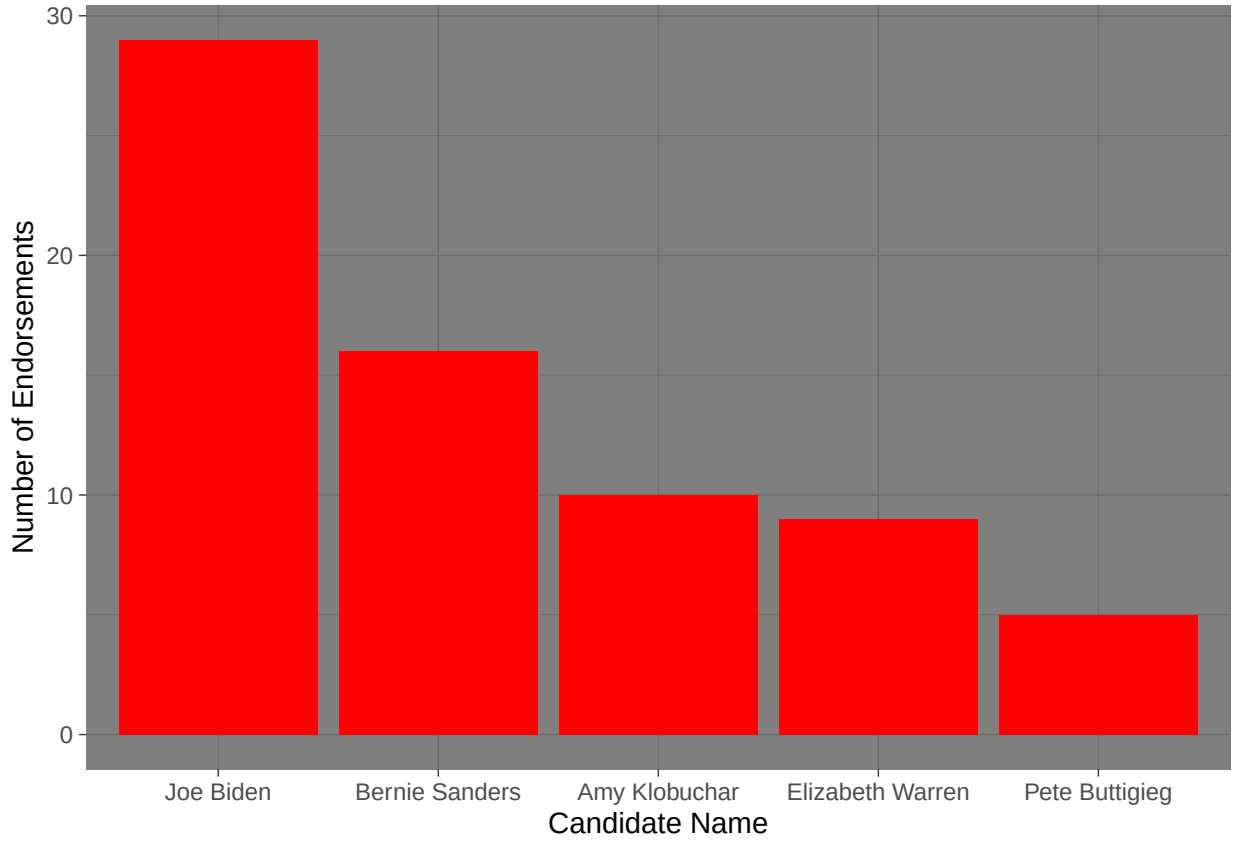
```
#In the same object as before, change names that are spelled differently in endorsements
subset_polls <- subset_polls %>% mutate(new_name = case_when(subset_polls$candidate_name == "Bernard Sa
                                        subset_polls$candidate_name == "Joseph R. Biden Jr." ~ "Joe Bid
                                        TRUE ~ as.character(subset_polls$candidate_name))) %>% rename(o

#Use inner_join to merge two tibbles/data frame
join <- inner_join(endorsements, subset_polls, by="candidate_name")
distinct(join, candidate_name)
```

```
## # A tibble: 5 x 1
##   candidate_name
##   <chr>
## 1 Joe Biden
## 2 Bernie Sanders
## 3 Amy Klobuchar
## 4 Elizabeth Warren
## 5 Pete Buttigieg
```

```
#Filter the five candidates that we are interested. Apply summarise function to collapse and count the
endorsements_count <- endorsements %>%   filter(candidate_name %in% c("Amy Klobuchar", "Bernie Sanders"
join <- left_join(join, endorsements_count, by= "candidate_name")
#Use ggplot to create the plot and customize it
```

```
library(ggplot2)
p <- ggplot(data=endorsements_count , aes(x=reorder(candidate_name, -count), y=count)) +
  geom_bar(stat="identity", fill="red")+ labs(x="Candidate Name", y="Number of Endorsements")
p + theme_dark()
```




```
ggsave("plot.pdf", p)
```

```
## 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:

```
library(tidyverse)
library(tm)
```

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

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

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

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
library(readr)
trump_tweets_url <- 'https://politicaldatascience.com/PDS/Datasets/trump_tweets.csv'
tweets <- read_csv(trump_tweets_url)
```

```
## Rows: 32974 Columns: 6

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): source, text, created_at
## dbl (2): retweet_count, favorite_count
## lgl (1): is_retweet

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

- 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 retweents) 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.

```
#Use the separate function to separate the two values in two variables. Then use the as.Date function t
s_tweets <- tweets %>% separate(created_at, into = c("sdate", "stime"), sep = " ") %>% mutate(dates = as
#Report the ranges of dates with summarise function and min and max arguments
s_tweets %>% summarise(min = min(dates),
            max = max(dates))
```

```
## # A tibble: 1 x 2
##   min        max
##   <date>     <date>
## 1 2014-01-01 2020-02-14
```

```
#Use slice_max and filter to show the top 5 most popular and retweeted tweets.
top5_rt <- s_tweets %>% filter(is_retweet==FALSE) %>% slice_max(retweet_count, n = 5)
top5_fav <- s_tweets %>% filter(is_retweet==FALSE) %>% slice_max(favorite_count, n = 5)
knitr::kable(top5_rt$text)
```

x
#FraudNewsCNN #FNN https://t.co/WYUnHjjUjg
TODAY WE MAKE AMERICA GREAT AGAIN!
Why would Kim Jong-un insult me by calling me “old” when I would NEVER call him “short and fat?”
Oh well I try so hard to be his friend - and maybe someday that will happen!
A$APRockyreleasedfromprisonandonhiswayhometotheUnitedStatesfromSweden.ItwasaRockyWeekgethomeASAPA$AP!
Such a beautiful and important evening! The forgotten man and woman will never be forgotten again. We will all come together as never before

```
knitr::kable(top5_fav$text)
```

x
A$APRockyreleasedfromprisonandonhiswayhometotheUnitedStatesfromSweden.ItwasaRockyWeekgethomeASAPA$AP!
https://t.co/VXeKiVzpTf
All is well! Missiles launched from Iran at two military bases located in Iraq. Assessment of casualties & damages taking place now. So far so good! We have the most powerful and well equipped military anywhere in the world by far! I will be making a statement tomorrow morning.
MERRY CHRISTMAS!
Kobe Bryant despite being one of the truly great basketball players of all time was just getting started in life. He loved his family so much and had such strong passion for the future. The loss of his beautiful daughter Gianna makes this moment even more devastating....

```
s_tweets$text <- gsub(" ?(f|ht)tp(s?)://(.*)[.][a-z]+", "", s_tweets$text) #Remove URLS from the begini
#Use Vcorpus function to create a corpus of Trump's tweets
Corpus_trump <- VCorpus(VectorSource(s_tweets$text))
inspect(Corpus_trump[[1]])
```

```
## <<PlainTextDocument>>
## Metadata:  7
## Content:  chars: 120
##
## RT @DailyCaller: ‘Why Would I Not:’ Chiefs’ Bashaud Breeland Looking Forward To WH Visit After Super
```

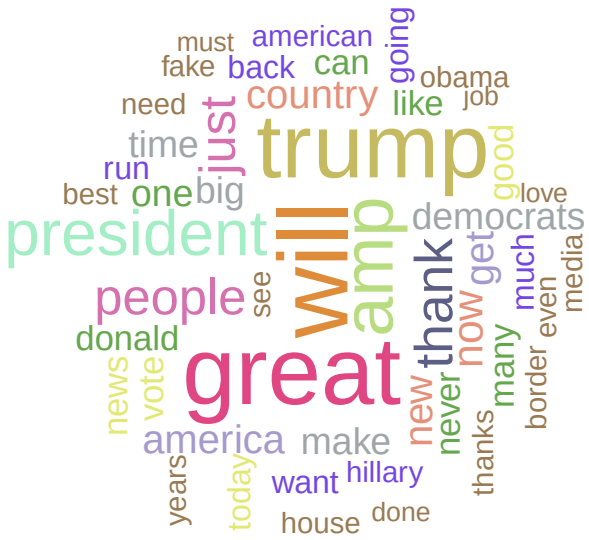
```
Corpus_trump  <- Corpus_trump %>%  tm_map(stripWhitespace) %>% #remove whitespace
                  tm_map(content_transformer(tolower)) %>% #to lower case all letters
                   tm_map(removeWords, stopwords("english")) %>% #remove stop words
                    tm_map(removeNumbers) %>% #remove numbers
                      tm_map(removePunctuation) #remove punctuation

library(randomcoloR)
```

```
## Warning: package 'randomcoloR' was built under R version 4.1.3
```

```
n <- 50
palette <- distinctColorPalette(n)
set.seed(1234) # for reproducibility
#Use wordcloud function to visualize the top 50 words
wordcloud(Corpus_trump, min.freq = 3,max.words=50, random.order=FALSE, rot.per=0.35,colors=palette)
```

```
## Warning in wordcloud(Corpus_trump, min.freq = 3, max.words = 50, random.order =
## FALSE, : realdonaldtrump could not be fit on page. It will not be plotted.
```




```
#Create a dtm
dtm <- TermDocumentMatrix(Corpus_trump, control = list(weighting= weightTfIdf))
```

```
## Warning in weighting(x): empty document(s): 1233 1235 3579 10943 14686
```

```
dtm_sparse <- removeSparseTerms(dtm, .99)
inspect(dtm_sparse)
```

```
## <<TermDocumentMatrix (terms: 160, documents: 32974)>>
## Non-/sparse entries: 124944/5150896
## Sparsity           : 98%
## Maximal term length: 21
## Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
## Sample             :
##                 Docs
## Terms            11853 12213 12462 1422 2229 25452 4969 5973 8939 9443
##   america            0     0     0    0    0     0    0    0    0    0
##   amp                0     0     0    0    0     0    0    0    0    0
##   great              0     0     0    0    0     0    0    0    0    0
##   just               0     0     0    0    0     0    0    0    0    0
##   people             0     0     0    0    0     0    0    0    0    0
##   president          0     0     0    0    0     0    0    0    0    0
##   realdonaldtrump    0     0     0    0    0     0    0    0    0    0
##   thank              0     0     0    0    0     0    0    0    0    0
##   trump              0     0     0    0    0     0    0    0    0    0
##   will               0     0     0    0    0     0    0    0    0    0
```

```
#Find the 50 words with highest tf.idf
dtm_matrix <- as.matrix(dtm_sparse)
dim(dtm_matrix)
```

```
## [1]   160 32974
```

```
#Sum all the words throughout the documents and in this sense, we can calculate the 50 top words
topwords <- data.frame(word=rownames(dtm_matrix), score=rowSums(dtm_matrix)) %>% slice_max(score, n =50)
knitr::kable(topwords)
```

	word	score
realdonaldtrump	realdonaldtrump	1924.1117
thank	thank	1455.7853
trump	trump	1320.3438
great	great	1300.2197
will	will	1193.4731
president	president	1001.3427
amp	amp	786.8330
america	america	712.7308
just	just	659.4659
people	people	613.6201
donald	donald	587.9756
new	new	576.8868
make	make	571.3972
thanks	thanks	530.9927
country	country	526.8978
run	run	525.4277
get	get	515.3841

	word	score
vote	vote	513.7458
now	now	509.9307
time	time	472.2704
one	one	458.3435
can	can	457.1268
big	big	450.9199
like	like	446.0007
never	never	439.2362
good	good	436.9428
democrats	democrats	434.3261
love	love	431.1505
today	today	420.7009
foxandfriends	foxandfriends	411.4763
makeamericagreatagain	makeamericagreatagain	401.8382
see	see	395.2871
best	best	389.4289
need	need	384.5747
going	going	382.8899
via	via	372.7494
tonight	tonight	371.8533
back	back	369.7678
hillary	hillary	369.2157
foxnews	foxnews	367.8251
news	news	367.6559
want	want	365.7595
many	many	365.7567
american	american	365.1458
obama	obama	354.6385
much	much	350.5131
jobs	jobs	332.4459
job	job	330.4590
day	day	329.5398
true	true	319.3690