STAT 231: Problem Set 6B

Jack Dove

due by 5 PM on Friday, October 9

This homework assignment is designed to help you futher ingest, practice, and expand upon the material covered in class over the past week(s). You are encouraged to work with other students, but all code and text must be written by you, and you must indicate below who you discussed the assignment with (if anyone).

Steps to proceed:

1. In RStudio, go to File > Open Project, navigate to the folder with the course-content repo, select the course-content project (course-content.Rproj), and click "Open"
2. Pull the course-content repo (e.g. using the blue-ish down arrow in the Git tab in upper right window)
3. Copy ps6B.Rmd from the course repo to your repo (see page 6 of the GitHub Classroom Guide for Stat231 if needed)
4. Close the course-content repo project in RStudio
5. Open YOUR repo project in RStudio
6. In the ps6B.Rmd file in YOUR repo, replace "YOUR NAME HERE" with your name
7. Add in your responses, committing and pushing to YOUR repo in appropriate places along the way
8. Run "Knit PDF"
9. Upload the pdf to Gradescope. Don't forget to select which of your pages are associated with each problem. *You will not get credit for work on unassigned pages (e.g., if you only selected the first page but your solution spans two pages, you would lose points for any part on the second page that the grader can't see).*

If you discussed this assignment with any of your peers, please list who here:

ANSWER:

Trump Tweets

David Robinson, Chief Data Scientist at DataCamp, wrote a blog post "Text analysis of Trump's tweets confirms he writes only the (angrier) Android half".

He provides a dataset with over 1,500 tweets from the account realDonaldTrump between 12/14/2015 and 8/8/2016. We'll use this dataset to explore the tweeting behavior of realDonaldTrump during this time period.

First, read in the file. Note that there is a `TwitteR` package which provides an interface to the Twitter web API. We'll use this R dataset David created using that package so that you don't have to set up Twitter authentication.

```
load(url("http://varianceexplained.org/files/trump_tweets_df.rda"))
```

A little wrangling to warm-up

1a. There are a number of variables in the dataset we won't need.

- First, confirm that all the observations in the dataset are from the screen-name `realDonaldTrump`.
- Then, create a new dataset called `tweets` that only includes the following variables:
- `text`
- `created`
- `statusSource`

```
library(mosaic)
```

```
## Loading required package: lattice

## Loading required package: ggformula

## Loading required package: ggstance

##
## Attaching package: 'ggstance'

## The following objects are masked from 'package:ggplot2':
##
##     geom_errorbarh, GeomErrorbarh

##
## New to ggformula?  Try the tutorials:
##  learnr::run_tutorial("introduction", package = "ggformula")
##  learnr::run_tutorial("refining", package = "ggformula")

## Loading required package: mosaicData

## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Registered S3 method overwritten by 'mosaic':
##   method                           from
##   fortify.SpatialPolygonsDataFrame ggplot2

##
## The 'mosaic' package masks several functions from core packages in order to add
## additional features.  The original behavior of these functions should not be affected by this.
##
## Note: If you use the Matrix package, be sure to load it BEFORE loading mosaic.

##
## Attaching package: 'mosaic'

## The following object is masked from 'package:Matrix':
##
##     mean

## The following objects are masked from 'package:dplyr':
##
##     count, do, tally

## The following object is masked from 'package:purrr':
##
##     cross

## The following object is masked from 'package:ggplot2':
##
##     stat

## The following objects are masked from 'package:stats':
##
##     binom.test, cor, cor.test, cov, fivenum, IQR, median, prop.test,
##     quantile, sd, t.test, var

## The following objects are masked from 'package:base':
##
##     max, mean, min, prod, range, sample, sum
```

```
library(tidyverse)
tally(~screenName, trump_tweets_df)
```

```
## screenName
## realDonaldTrump
##            1512
```

```r
#Tally confirms all 1512 tweets are from realDonaldTrump

tweets <- trump_tweets_df %>%
  select(text, created, statusSource)
```

1b. Using the `statusSource` variable, compute the number of tweets from each source. How many different sources are there? How often are each used?

> ANSWER: There are five sources: Instagram, Twitter Web Client, Twitter for iPad, Twitter for Android, and Twitter for iPhone. Trump predominantly uses the web client, Android and iPhone, led by 762 tweets from the Android.

```
tally(~statusSource, data=tweets)
```

```
## statusSource
##                          <a href="http://instagram.com" rel="nofollow">Instagram</a>
##                                                                                    1
##                   <a href="http://twitter.com" rel="nofollow">Twitter Web Client</a>
##                                                                                  120
##    <a href="http://twitter.com/#!/download/ipad" rel="nofollow">Twitter for iPad</a>
##                                                                                    1
## <a href="http://twitter.com/download/android" rel="nofollow">Twitter for Android</a>
##                                                                                  762
##   <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
##                                                                                  628
```

1c. We're going to compare the language used between the Android and iPhone sources, so only want to keep tweets coming from those sources. Explain what the `extract` function (from the `tidyverse` package) is doing below. (Note that "regex" stands for "regular expression".)

> ANSWER: The extract function looks for a string following "Twitter for," saving that result as a new variable called "source."

```
tweets2 <- tweets %>%
  extract(col = statusSource, into = "source"
          , regex = "Twitter for (.*)<"
          , remove = FALSE) %>%
  filter(source %in% c("Android", "iPhone"))
```

How does the language of the tweets differ by source?

2a. Create a word cloud for the top 50 words used in tweets sent from the Android. Create a second word cloud for the top 50 words used in tweets sent from the iPhone. How do these word clouds compare? (Are there some common words frequently used from both sources? Are the most common words different between the sources?)

Don't forget to remove stop words before creating the word cloud. Also remove the terms "https" and "t.co".

> ANSWER: Trump uses makeamericagreatagain disproportionately more on his iphone than on his android. He talks about his opponents more on his android, and more about himself (trump 2016, realDonaldTrump) on his iPhone. For the most part, however, he seems to display similar word trends between tweets from both sources.

```
stop_words <- tidytext::stop_words

word <- c("https", "t.co")
two <- data.frame(word)


stop_words <- stop_words %>%
  full_join(two, by= "word")
```

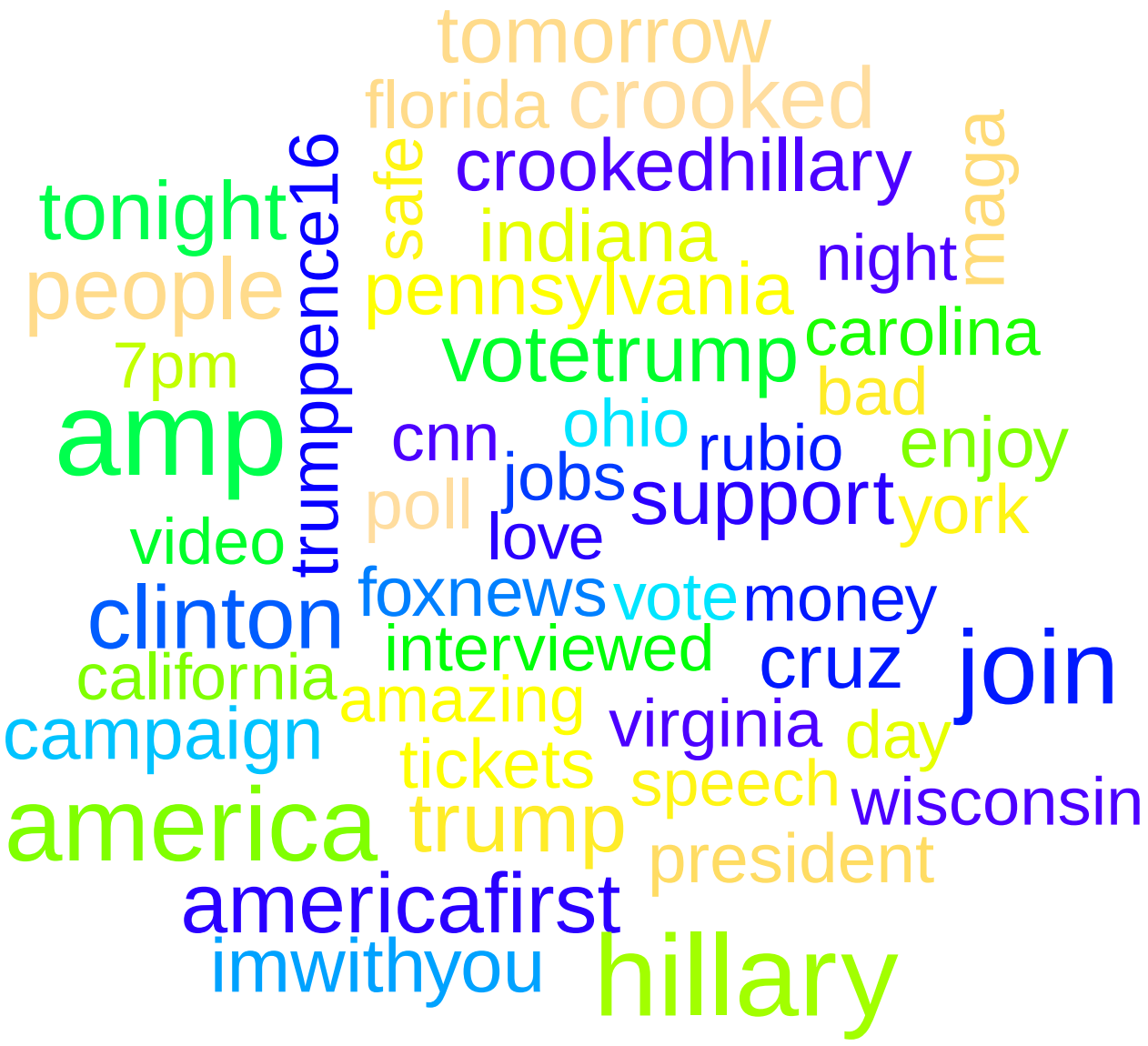
```
## Warning: Column `word` joining character vector and factor, coercing into
## character vector
```

```
#Android Words
trumpandroidtweets <- tweets2 %>%
  filter(source == "Android") %>%
  unnest_tokens(output = word, input = text, token = "words") %>%
  anti_join(stop_words, by = "word") %>%
  select(word)

#Android Word Frequency
trumpandroidtweetsfreq <- trumpandroidtweets %>%
  count(word, sort = TRUE) %>%
  select(word, n)

#iphone Words
trumpiphonetweets <- tweets2 %>%
  filter(source == "iPhone") %>%
  unnest_tokens(output = word, input = text, token = "words") %>%
  anti_join(stop_words, by = "word") %>%
  select(word)

#iphone Word Frequency
trumpiphonetweetsfreq <- trumpiphonetweets %>%
  count(word, sort = TRUE) %>%
  select(word, n)

#iphone word cloud
set.seed(1962)
wordcloud(trumpiphonetweetsfreq$word, trumpiphonetweetsfreq$n,
```

```
  max.words = 50,
  scale = c(10, 2),
  colors = topo.colors(n = 30),
  random.color = TRUE)
```

```
## Warning in wordcloud(trumpiphonetweetsfreq$word, trumpiphonetweetsfreq$n, :
## makeamericagreatagain could not be fit on page. It will not be plotted.
```

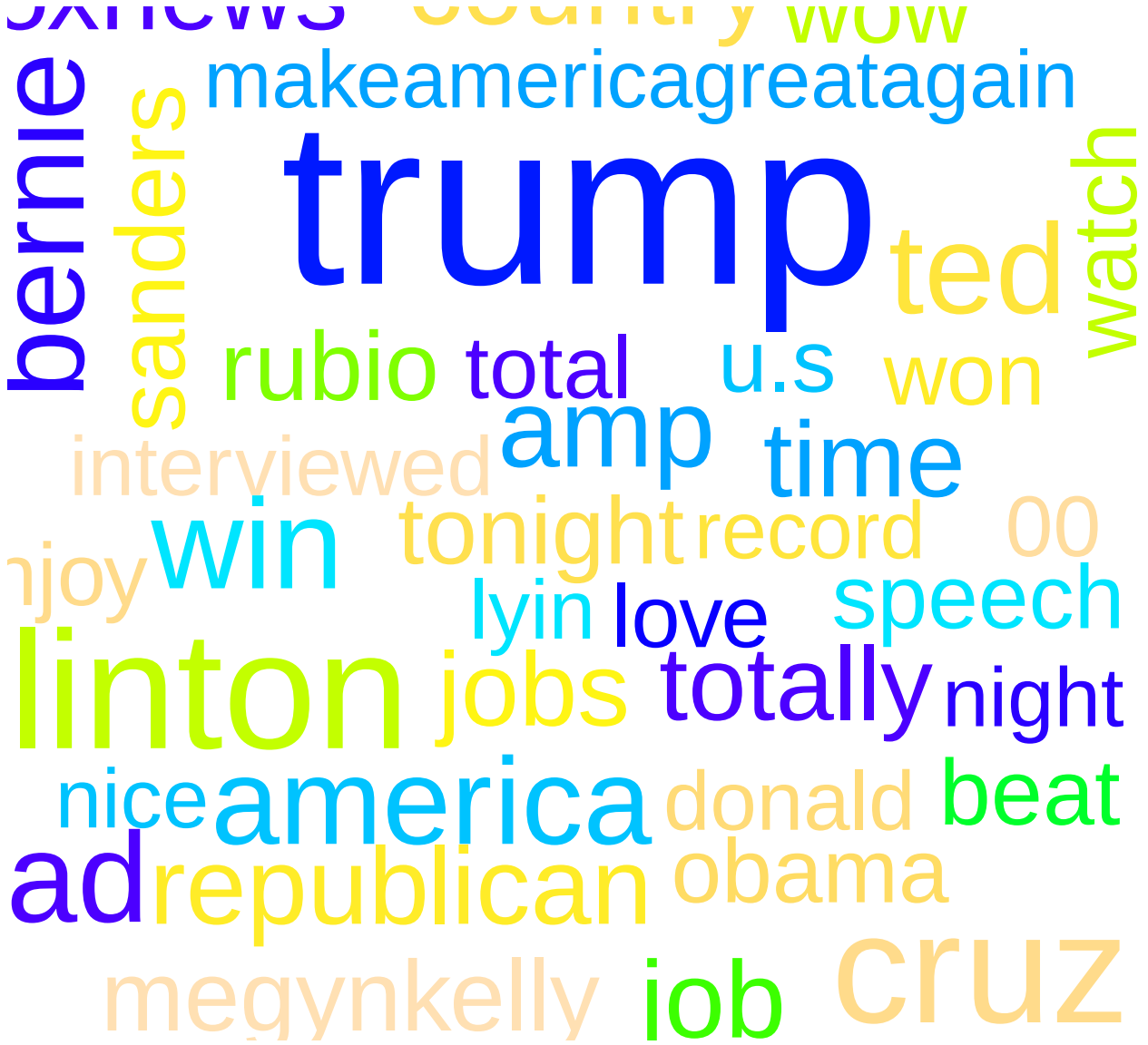
```
## Warning in wordcloud(trumpiphonetweetsfreq$word, trumpiphonetweetsfreq$n, :
## trump2016 could not be fit on page. It will not be plotted.
```




```
#Android Word Cloud
set.seed(1962)
wordcloud(trumpandroidtweetsfreq$word, trumpandroidtweetsfreq$n,
  max.words = 50,
  scale = c(10, 2),
```

```
  colors = topo.colors(n = 30),
  random.color = TRUE)
```

```
## Warning in wordcloud(trumpandroidtweetsfreq$word, trumpandroidtweetsfreq$n, :
## hillary could not be fit on page. It will not be plotted.

## Warning in wordcloud(trumpandroidtweetsfreq$word, trumpandroidtweetsfreq$n, :
## crooked could not be fit on page. It will not be plotted.

## Warning in wordcloud(trumpandroidtweetsfreq$word, trumpandroidtweetsfreq$n, :
## people could not be fit on page. It will not be plotted.

## Warning in wordcloud(trumpandroidtweetsfreq$word, trumpandroidtweetsfreq$n, :
## president could not be fit on page. It will not be plotted.

## Warning in wordcloud(trumpandroidtweetsfreq$word, trumpandroidtweetsfreq$n, :
## realdonaldtrump could not be fit on page. It will not be plotted.

## Warning in wordcloud(trumpandroidtweetsfreq$word, trumpandroidtweetsfreq$n, :
## vote could not be fit on page. It will not be plotted.

## Warning in wordcloud(trumpandroidtweetsfreq$word, trumpandroidtweetsfreq$n, :
## nytimes could not be fit on page. It will not be plotted.

## Warning in wordcloud(trumpandroidtweetsfreq$word, trumpandroidtweetsfreq$n, :
## cnn could not be fit on page. It will not be plotted.

## Warning in wordcloud(trumpandroidtweetsfreq$word, trumpandroidtweetsfreq$n, :
## convention could not be fit on page. It will not be plotted.

## Warning in wordcloud(trumpandroidtweetsfreq$word, trumpandroidtweetsfreq$n, :
## campaign could not be fit on page. It will not be plotted.
```


wow
bernie
makeamericagreatagain
sanders
trump
ted
watch
rubio
total
u.s
won
amp
interviewed
time
win
tonight
record
00
joy
lyin
love
speech
linton
jobs
totally
night
nice
america
donald
beat
ad
republican
obama
cruz
megynkelly
job

2b. Create a visualization that compares the top 10 *bigrams* appearing in tweets by each source (that is, facet by source). After creating a dataset with one row per bigram, you should remove any rows that contain a stop word within the bigram.

How do the top used bigrams compare between the two sources?

> ANSWER: Similar to the word clouds, the top bigrams from the Android are related to Trump's political foes, while those from his iPhone are more self-centered.

```
tweets2_bigrams_freq <- tweets2 %>%
  group_by(source) %>%
  unnest_tokens(output = word, input = text
                , token = "ngrams", n = 2) %>%
  count(word, sort=TRUE) %>%
  select(word, n)
```

```
## Adding missing grouping variables: `source`
```

```
#Used https://www.tidytextmining.com/ngrams.html for help on filtering out stop words

bigrams_separated <- tweets2_bigrams_freq %>%
  separate(word, c("word1", "word2"), sep = " ")

bigrams_filtered <- bigrams_separated %>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word)

tweet_bigrams_filtered <- bigrams_filtered %>%
  unite(word, word1, word2, sep = " ")

tweet_bigrams_filtered1 <- tweet_bigrams_filtered %>%
  group_by(source) %>%
  slice(1:10) %>%
  ungroup()

plot <- ggplot(data= tweet_bigrams_filtered1, aes(x = reorder(word,n)
                            , y = n
                            , fill = source)) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "Number of Uses") +
  facet_wrap(~ source, ncol = 2, scales = "free") +
  coord_flip()
```

2c. Consider the sentiment. Compute the proportion of words among the tweets within each source classified as "angry" and the proportion of words classified as "joy" based on the NRC lexicon. How does the proportion of "angry" and "joy" words compare between the two sources? What about "positive" and "negative" words?

> ANSWER: The iPhone tweets have higher proportions of joy and anger than the android. However, Android tweets had a higher proportin of positive and negative tweets.

```
nrc_lexicon <- get_sentiments("nrc")

trumptweetswords_sentiments <- tweets2 %>%
  unnest_tokens(output = word, input = text, token = "words") %>%
  group_by(source) %>%
  anti_join(stop_words, by = "word") %>%
  select(word) %>%
  count(word, sort = TRUE) %>%
  select(word, n) %>%
  inner_join(nrc_lexicon, by="word")
```

```
## Adding missing grouping variables: `source`
## Adding missing grouping variables: `source`
```

```
trumpsentimentsandroid <- trumptweetswords_sentiments %>%
  filter(source=="Android")

trumpsentimentsiphone <- trumptweetswords_sentiments %>%
  filter(source=="iPhone")

#Android Sentiments
tally(~sentiment, data=trumpsentimentsandroid)
```

```
## sentiment
##        anger anticipation      disgust         fear          joy     negative
##          106           94           78          115           77          227
##     positive      sadness     surprise        trust
##          238          108           54          140
```

```
prop_joy_android <- 77/nrow(trumpsentimentsandroid)
prop_anger_android <- 106/nrow(trumpsentimentsandroid)
prop_positive_android <- 238/nrow(trumpsentimentsandroid)
prop_negative_android <- 227/nrow(trumpsentimentsandroid)

#Apple Sentiments
tally(~sentiment, data=trumpsentimentsiphone)
```

```
## sentiment
##        anger anticipation      disgust         fear          joy     negative
##           82           67           47           80           59          147
##     positive      sadness     surprise        trust
##          166           77           40           99
```

```r
prop_joy_iphone <- 59/nrow(trumpsentimentsiphone)
prop_anger_iphone <- 82/nrow(trumpsentimentsiphone)
prop_positive_iphone <- 166/nrow(trumpsentimentsiphone)
prop_negative_iphone <- 147/nrow(trumpsentimentsiphone)

ifelse(prop_joy_android > prop_joy_iphone, "android more joyful", "iphone more joyful")
```

```
## [1] "iphone more joyful"
```

```r
ifelse(prop_anger_android > prop_anger_iphone, "android more angry", "iphone more angry")
```

```
## [1] "iphone more angry"
```

```r
ifelse(prop_positive_android > prop_positive_iphone, "android more positive", "iphone more positive")
```

```
## [1] "android more positive"
```

```r
ifelse(prop_negative_android > prop_negative_iphone, "android more negative", "iphone more negative")
```

```
## [1] "android more negative"
```

2d. Lastly, based on your responses above, do you think there is evidence to support Robinson's claim that Trump only writes the (angrier) Android half of the tweets from realDonaldTrump? In 2-4 sentences, please explain.

ANSWER: I think the results of this sentiment test are inconclusive. If the android displays more negative and positive proportions than the iPhone, yet the iPhone contains more joyful and angry proportions than the Android, then the results seem to contradict themselves. I'd like to look into the stop words to see if any important words were omitted, as well as look through the lexicon to find any potentially problematic sentiment definitions.