Problem Set 11

NLP

[YOUR NAME]

Due Date: 2024-04-12

Open `RStudio` and create a new RMarkDown file (`.Rmd`) by going to `File -> New File -> R Markdown...` . Accept defaults and save this file as `[LAST NAME]_ps11.Rmd` to your `code` folder.

Copy and paste the contents of this `.Rmd` file into your `[LAST NAME]_ps11.Rmd` file. Then change the `author: [Your Name]` to your name.

We will be using the `Trump_tweet_words.Rds` file from the course github page (https://github.com/jbisbee1/DS1000_S2024/raw/main/data/Trump_tweet_words.Rds).

All of the following questions should be answered in this `.Rmd` file. There are code chunks with incomplete code that need to be filled in.

This problem set is worth 8 total points, plus two extra credit points. The point values for each question are indicated in brackets below. To receive full credit, you must have the correct code. In addition, some questions ask you to provide a written response in addition to the code.

You are free to rely on whatever resources you need to complete this problem set, including lecture notes, lecture presentations, Google, your classmates…you name it. However, the final submission must be complete by you. There are no group assignments. To submit, compiled the completed problem set and upload the PDF file to Brightspace on Friday by midnight. Also note that the TAs and professors will not respond to Campuswire posts after 5PM on Friday, so don't wait until the last minute to get started!

Good luck!

*Copy the link to ChatGPT you used here: _____

Question 0

Require `tidyverse` , `tidytext` and `tidymodels` , and then load the `Trump_tweet_words.Rds` (https://github.com/jbisbee1/DS1000_S2024/raw/main/data/Trump_tweet_words.Rds) data to an object called `tweet_words` .

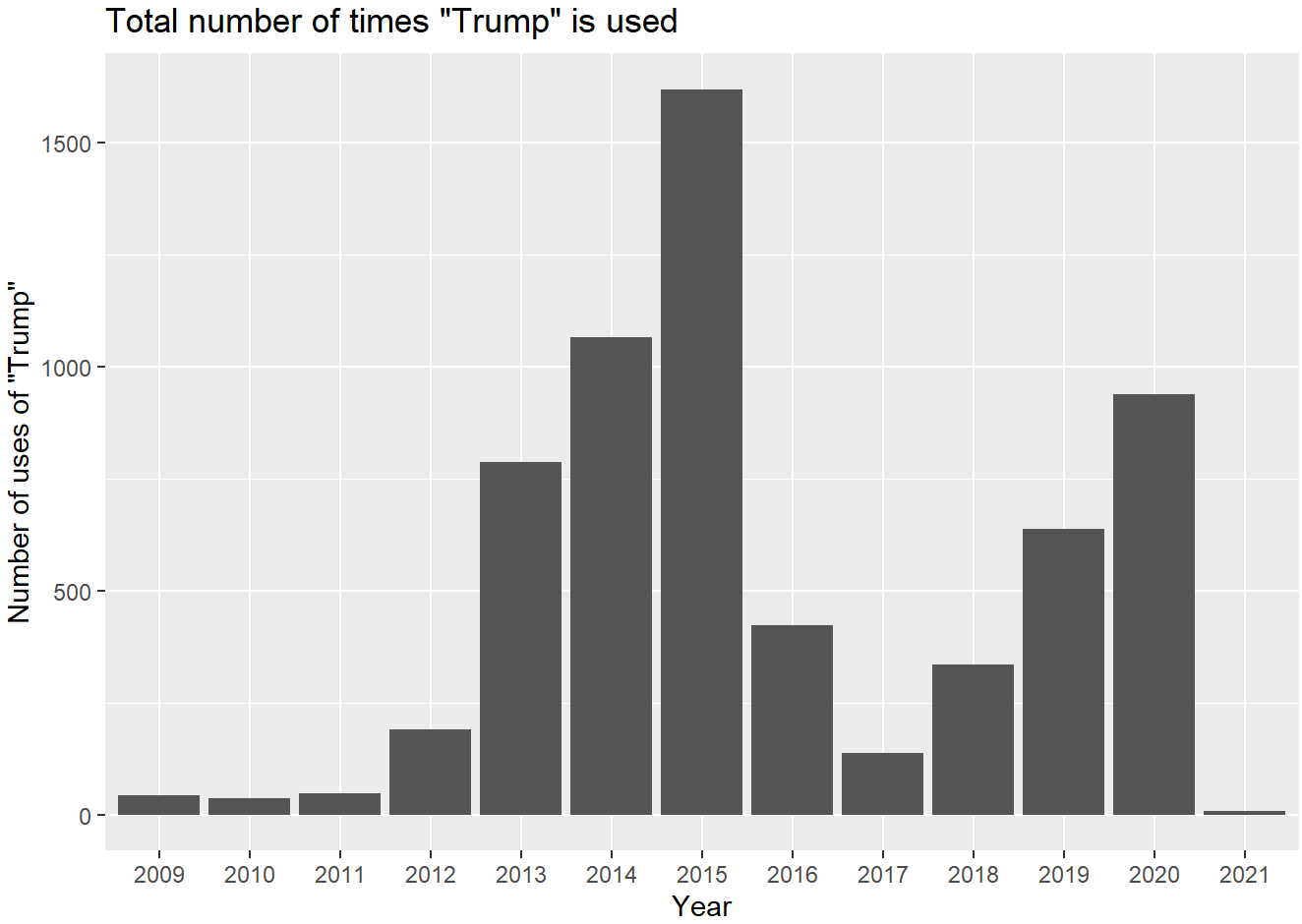
```
require(tidyverse)
require(tidytext)
require(tidymodels)
tweet_words <- read_rds(file="https://github.com/jbisbee1/DS1000_S2024/raw/main/data/Trump_tweet
_words.Rds")
```

Question 1 [2 points]

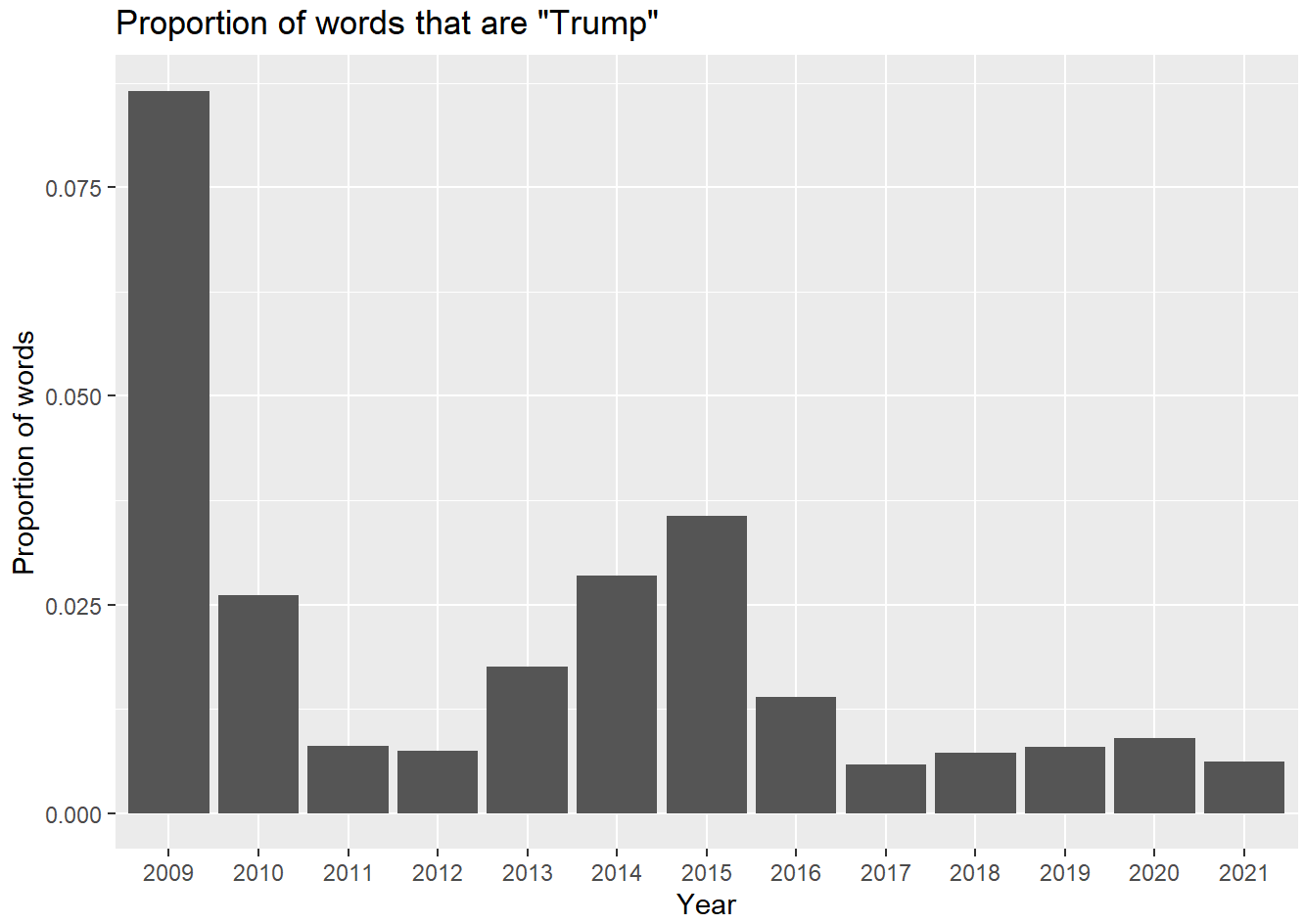
a. Plot the total number of times the word "trump" is used each year. Then, plot the proportion of times the word "trump" is used each year. Make sure to justify your choice of `geom_...()` ! [1 point]

b. Why are these plots so different? Which measure is better? Why? [1 point]

```
# Total number of times
tweet_words %>%
  count(Tweeting.year,word) %>%
  filter(word == 'trump') %>%
  ggplot(aes(x = Tweeting.year,y = n)) +
  geom_bar(stat = 'identity') +
  labs(x = 'Year',
       y = 'Number of uses of "Trump"',
       title = 'Total number of times "Trump" is used')
```




```
# Proportion of times
tweet_words %>%
  count(Tweeting.year,word) %>%
  group_by(Tweeting.year) %>%
  mutate(totWords = sum(n)) %>%
  ungroup() %>%
  mutate(prop = n / totWords) %>%
  filter(word == 'trump') %>%
  ggplot(aes(x = Tweeting.year,y = prop)) +
  geom_bar(stat = 'identity') +
  labs(x = 'Year',
       y = 'Proportion of words',
       title = 'Proportion of words that are "Trump"')
```



- These plots look different because Trump tweeted much more frequently in 2012 - 2016 than he did in 2009 - 2011, but he tweeted about himself as a fraction of total tweets much more frequently in 2009 than any other year. This comparison reveals that the proportion of total tweets is a better measure of Trump's behavior, since it accurately measures the quantity of interest. If we relied on the total tweets, we would conclude he was much more self-obsessed in 2015 than any other year, but this conclusion conflates how much he tweets in general with how much he tweets about himself.

Question 2 [2 points]

We want to only look at tweets written during Trump's first year as president (January 20th, 2017 through December 31st, 2017), and are interested if there are patterns in what he talks about.

We will use k-means clustering to learn about this data. To do so, follow these steps.

a. Create a document-term matrix (`dtm`), dropping any words that appear fewer than 20 times total, and using the `document` column as the document indicator. **NB: Drop the word `'amp'` .**

b. Calculate the TF-IDF using the appropriate function from the `tidytext` package.

c. Cast the DTM to wide format using the `cast_dtm()` function, also from the `tidytext` package.

d. Determine the optimal number of clusters / centers / topics / k by creating and manually inspecting an elbow plot. To save time, only examine the following sizes: `c(1,10,50,100,250,500,1000)` and set `nstart = 5` with `set.seed(123)`. (This will still take a little while to run so be patient!).

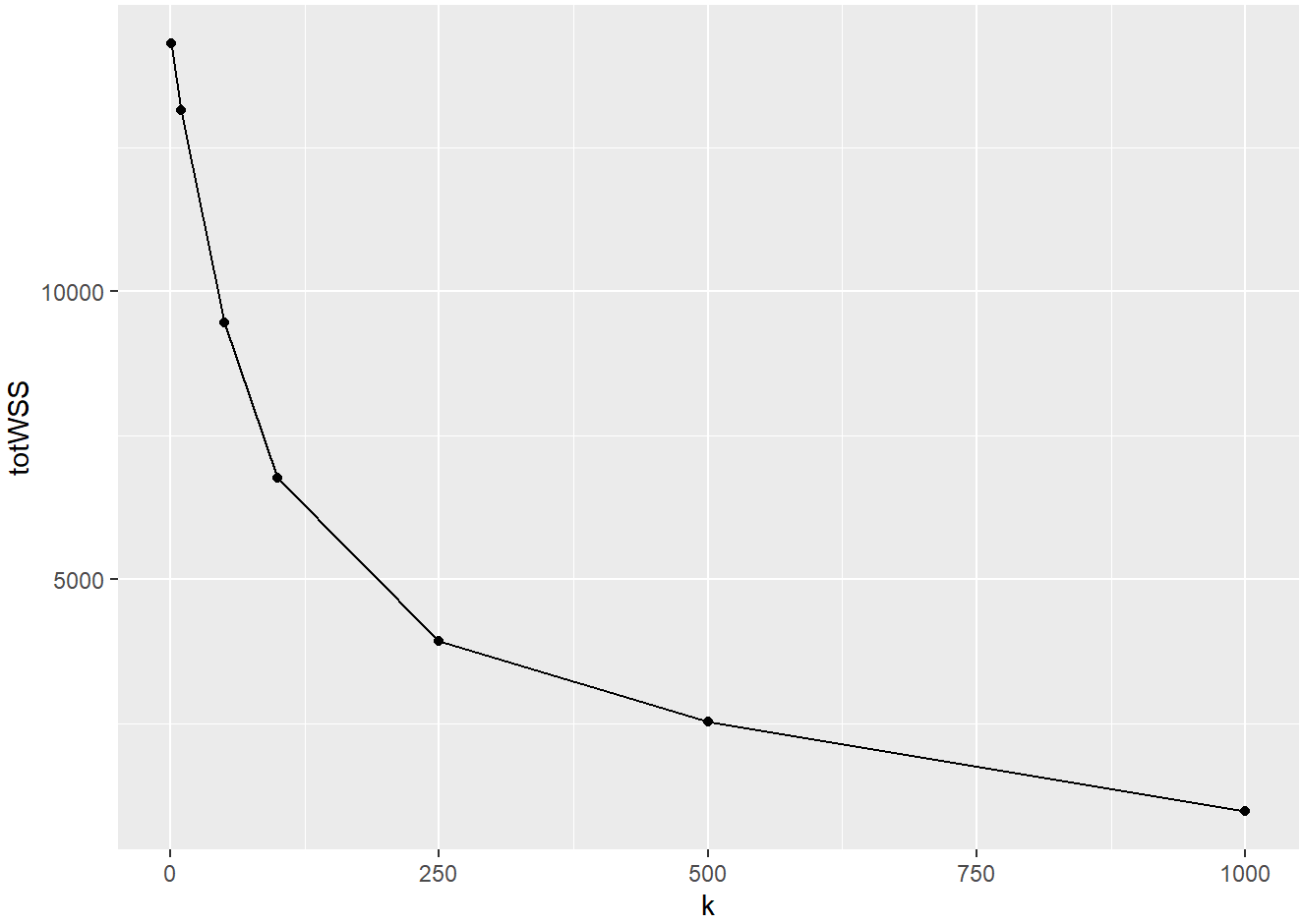
e. Using the optimal value from the elbow plot, run k-means on the data with `nstart` set to 5 and `set.seed(123)`.

f. Which are the top 3 most popular topics for Donald Trump in this period? Plot the top 10 highest scoring words for each of the top 3 most popular topics. What is each "about"?

```
# a.
dtm <- tweet_words %>%
  filter(Tweeting.date > as.Date('2017-01-20') & Tweeting.date < as.Date('2017-12-31'), # Filter
to the correct period
         word != 'amp') %>%  # Drop the word 'amp'
  count(document,word) %>%  # Count the number of times each word appears in each document
  group_by(word) %>%  # Count the total number of times a word appears overall
  mutate(tot_n = sum(n)) %>%
  ungroup() %>%
  filter(tot_n >20) # Filter to only words that appear more than 20 times in total

#b.
dtm.tfidf <- bind_tf_idf(tbl = dtm, term = word, document = document, n = n) # Calculate the TF-
IDF metric

#c.
castdtm <- cast_dtm(data = dtm.tfidf, document = document, term = word, value = tf_idf) # Cast t
o a DTM

#d.
set.seed(123) # Set common seed to ensure reproducability
totWSS <- NULL # Instantiate an empty object
for(k in c(1,10,50,100,250,500,1000)) {
  km_out <- kmeans(castdtm,
                centers = k, # Set the number of centers equal to k
                nstart = 5) # Set nstart = 5

  totWSS <- data.frame(totWSS = km_out$tot.withinss,
                       k = k) %>%
    bind_rows(totWSS)
  cat(k,'\n')
}
```

```
## 1
## 10
## 50
## 100
## 250
## 500
## 1000
```

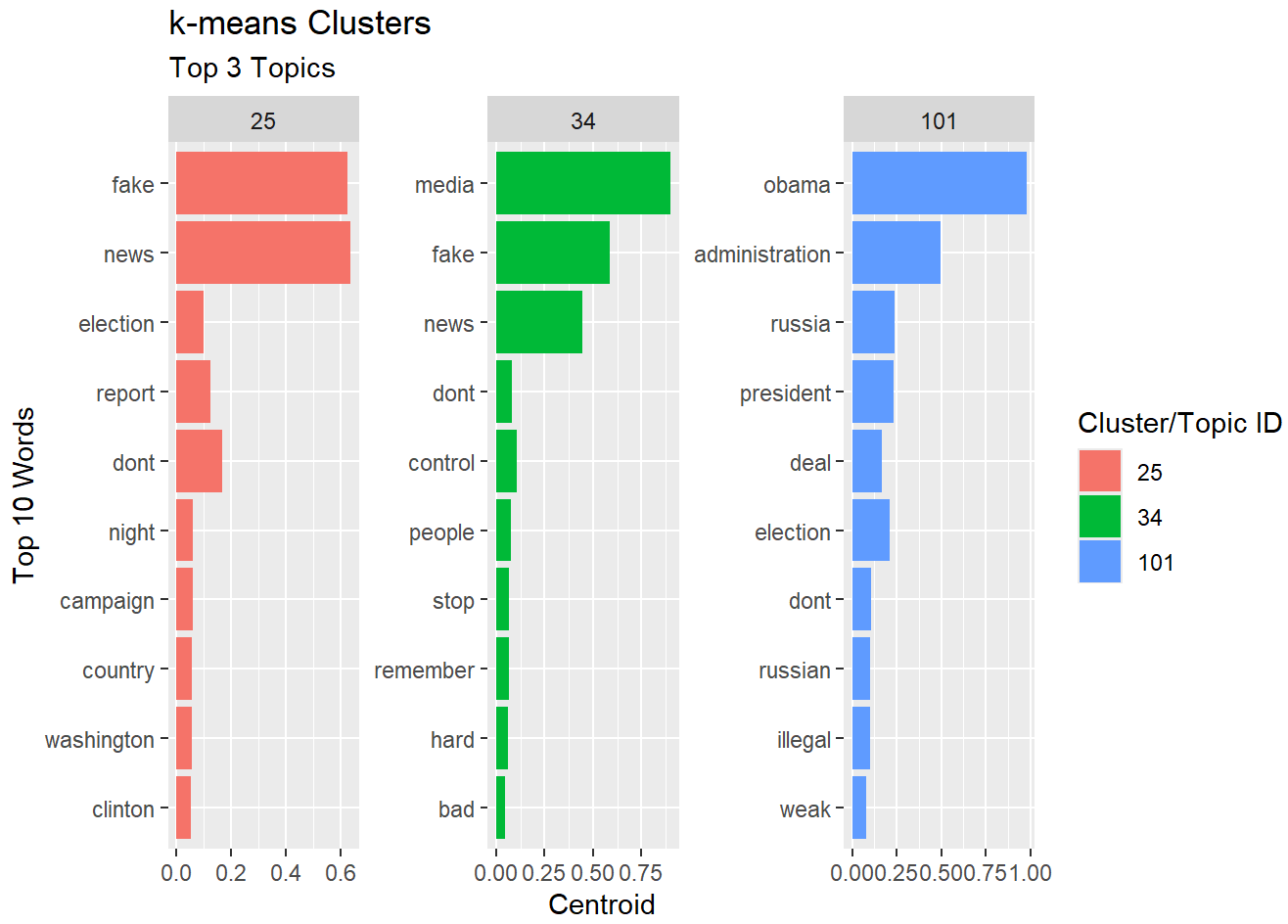
```
# Plot the elbow plot
totWSS %>%
  ggplot(aes(x = k,y = totWSS)) +
  geom_point() +
  geom_line()
```



```
#e.
km_out <- kmeans(castdtm,
                 centers = 250, # Set the number of centers to the value identified in the elbow
plot
                 nstart = 5) # Set nstart = 5

km_out_tidy <- tidy(km_out) %>% # Tidy the kmeans result
  gather(word,mean_tfidf,-size,-cluster,-withinss) %>% # Pivot to long format
  mutate(mean_tfidf = as.numeric(mean_tfidf)) # Convert the average TF-IDF value to numeric

# For students who can't load tidymodels
# km_out_tidy <- as_tibble(km_out$centers) %>%
#   mutate(size = km_out$size,
#          withinss = km_out$withinss,
#          cluster = factor(row_number())) %>%
#   gather(word,mean_tfidf,-size,-cluster,-withinss)

#f. Find the top 3 topics tweeted by Trump
(tops <- km_out_tidy %>%
  select(size,withinss,cluster) %>%
  distinct() %>%
  arrange(desc(size)) %>%
    slice(1:3))
```

```
## # A tibble: 3 × 3
##    size withinss cluster
##   <int>    <dbl> <fct>
## 1    34     38.1 34
## 2    33     56.9 25
## 3    29     63.4 101
```

```
#g. Visualize the top 10 words for these top 3 topics
km_out_tidy %>%
  filter(cluster %in% tops$cluster) %>%
  group_by(cluster) %>%
  arrange(-mean_tfidf) %>%
  slice(1:10) %>%
  ggplot(aes(x = mean_tfidf,y = reorder(word,mean_tfidf),
             fill = factor(cluster))) +
  geom_bar(stat = 'identity') +
  facet_wrap(~cluster,scales = 'free') +
  labs(title = 'k-means Clusters',
       subtitle = 'Top 3 Topics',
       x = 'Centroid',
       y = 'Top 10 Words',
       fill = 'Cluster/Topic ID')
```



- There are two topics that are clearly about Trump's favorite topics in 2017: his concern about fake news and his focus on Obama and claims about Russian interference.

Question 3 [2 points]

Now load the sentiment dictionary `nrc` from the `tidytext` package, and look at the clusters with sentiment scores by merging the `km_out_tidy` dataset with the `nrc` dataset using the `inner_join()` function. (If you can't open the `nrc` object from the `tidytext` package, you can just load it from GitHub with this link: `https://github.com/jbisbee1/DS1000_S2024/raw/main/data/nrc.Rds`)

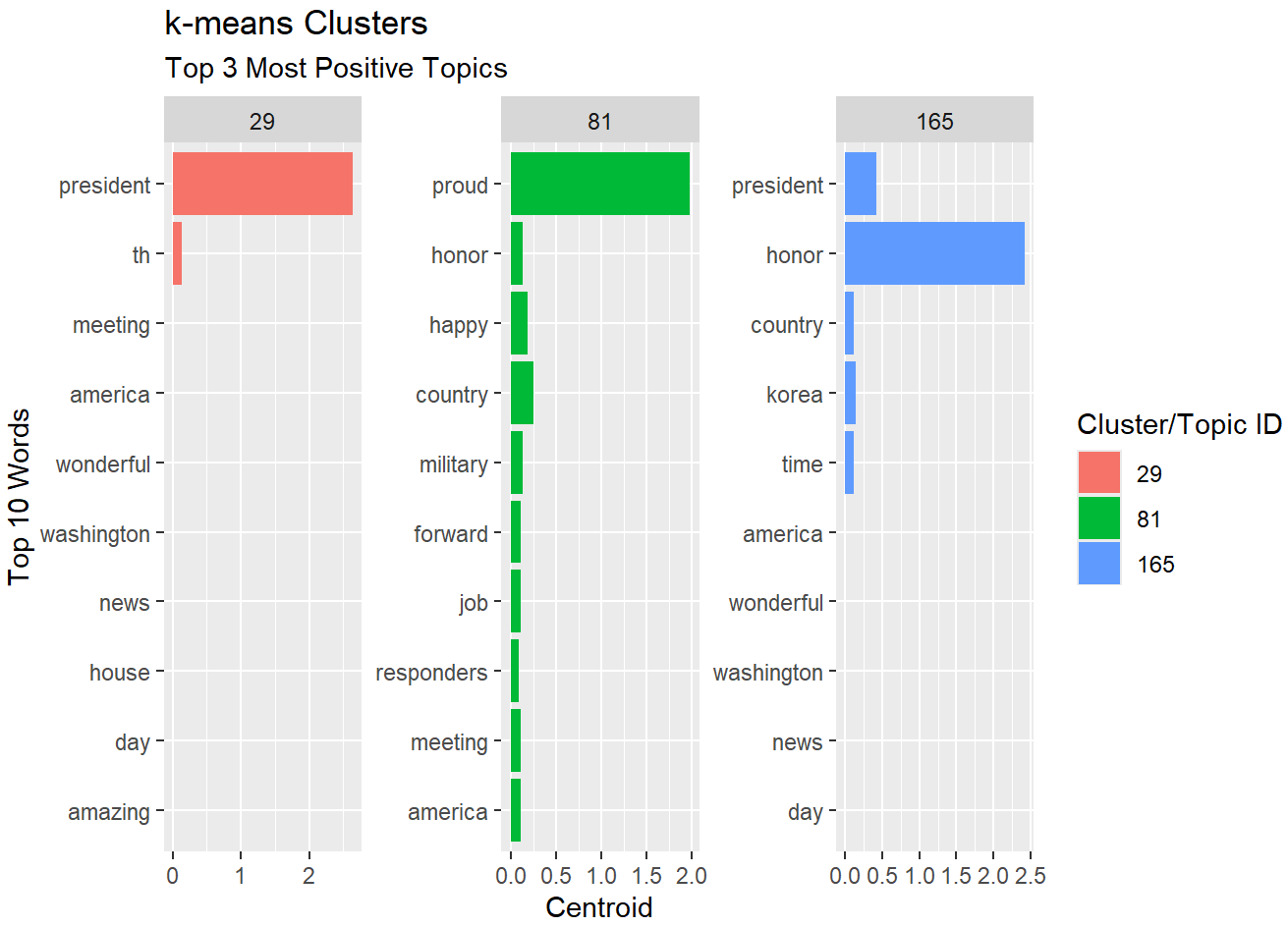
Filter to only look at `positive` and `negative` categories and then `select()` only the `size` , `cluster` , `word` , `mean_tfidf` , and `sentiment` columns. Then use either `spread()` or `pivot_wider()` to create two columns of `mean_tfidf` values: one for `positive` and one for `negative` . Replace `NA` values with 0! Finally, filter to only look at clusters with more than 10 tweets in them. Save this processed data to an object named `cluster_sentiment` .

Using this data, plot the top 10 words for the three most positive clusters and the top 10 words for the three most negative clusters. Describe what you see. What are Trump's most positive and negative topics about?

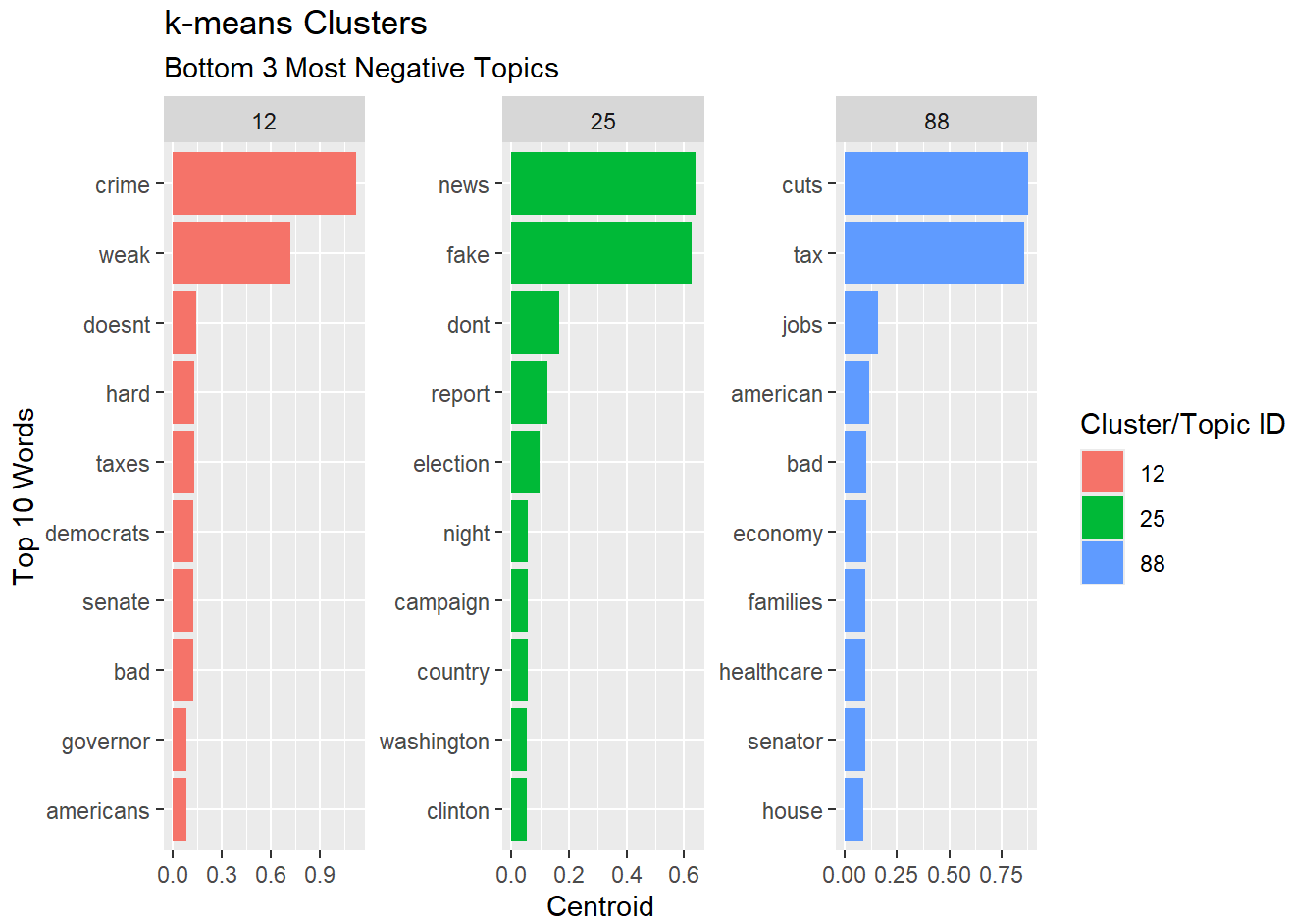
```
nrc <- read_rds('https://github.com/jbisbee1/DS1000_S2024/raw/main/data/nrc.Rds') # Load the NRC dictionary

cluster_sentiment <- km_out_tidy %>%
  inner_join(nrc %>% # Join on the words that appear in both (ignore the warning)
               filter(sentiment %in% c('positive','negative'))) %>% # Filter the nrc to only positive and negative labels
  select(size,cluster,word,mean_tfidf,sentiment) %>% # Select the columns size, cluster, word, mean_tfidf, and sentiment
  spread(sentiment,mean_tfidf,fill = 0) %>% # Spread the data into two columns, one for
  mutate(net_sentiment = positive - negative) %>% # Calculate the net_sentiment as positive - negative
  group_by(cluster,size) %>% # Calculate the average sentiment by cluster and size
  summarise(negative = mean(negative),
            positive = mean(positive),
            net_sentiment = mean(net_sentiment),.groups = 'drop') %>%
  ungroup() %>%
  filter(size > 10) # Drop clusters with fewer than 10 tweets
```

```
## Joining with `by = join_by(word)`
```

```
# Calculate the top 3 most positive clusters
top_sentiment <- cluster_sentiment %>%
  arrange(desc(net_sentiment)) %>%
  slice(1:3)

# Visualize the top 10 words in the top 3 most positive clusters
km_out_tidy %>%
  filter(cluster %in% top_sentiment$cluster) %>%
  group_by(cluster) %>%
  arrange(-mean_tfidf) %>%
  slice(1:10) %>%
  ggplot(aes(x = mean_tfidf,y = reorder(word,mean_tfidf),
             fill = factor(cluster))) +
  geom_bar(stat = 'identity') +
  facet_wrap(~cluster,scales = 'free') +
  labs(title = 'k-means Clusters',
       subtitle = 'Top 3 Most Positive Topics',
       x = 'Centroid',
       y = 'Top 10 Words',
       fill = 'Cluster/Topic ID')
```




```
# Calculate the bottom 3 most negative clusters
bottom_sentiment <- cluster_sentiment %>%
  arrange(net_sentiment) %>%
  slice(1:3)


# Visualize the top 10 words in the bottom 3 most negative clusters
km_out_tidy %>%
  filter(cluster %in% bottom_sentiment$cluster) %>%
  group_by(cluster) %>%
  arrange(-mean_tfidf) %>%
  slice(1:10) %>%
  ggplot(aes(x = mean_tfidf,y = reorder(word,mean_tfidf),
             fill = factor(cluster))) +
  geom_bar(stat = 'identity') +
  facet_wrap(~cluster,scales = 'free') +
  labs(title = 'k-means Clusters',
       subtitle = 'Bottom 3 Most Negative Topics',
       x = 'Centroid',
       y = 'Top 10 Words',
       fill = 'Cluster/Topic ID')
```



- Trump's top 3 most positive topics appear to be about the position of the president (topic 29), and honoring the military or America in general (topics 81 and 165). His bottom 3 most negative topics are about Democrats and criminals (topic 12), fake news and the election (topic 25), and the US economy (topic 88).

Question 4 [2 points]

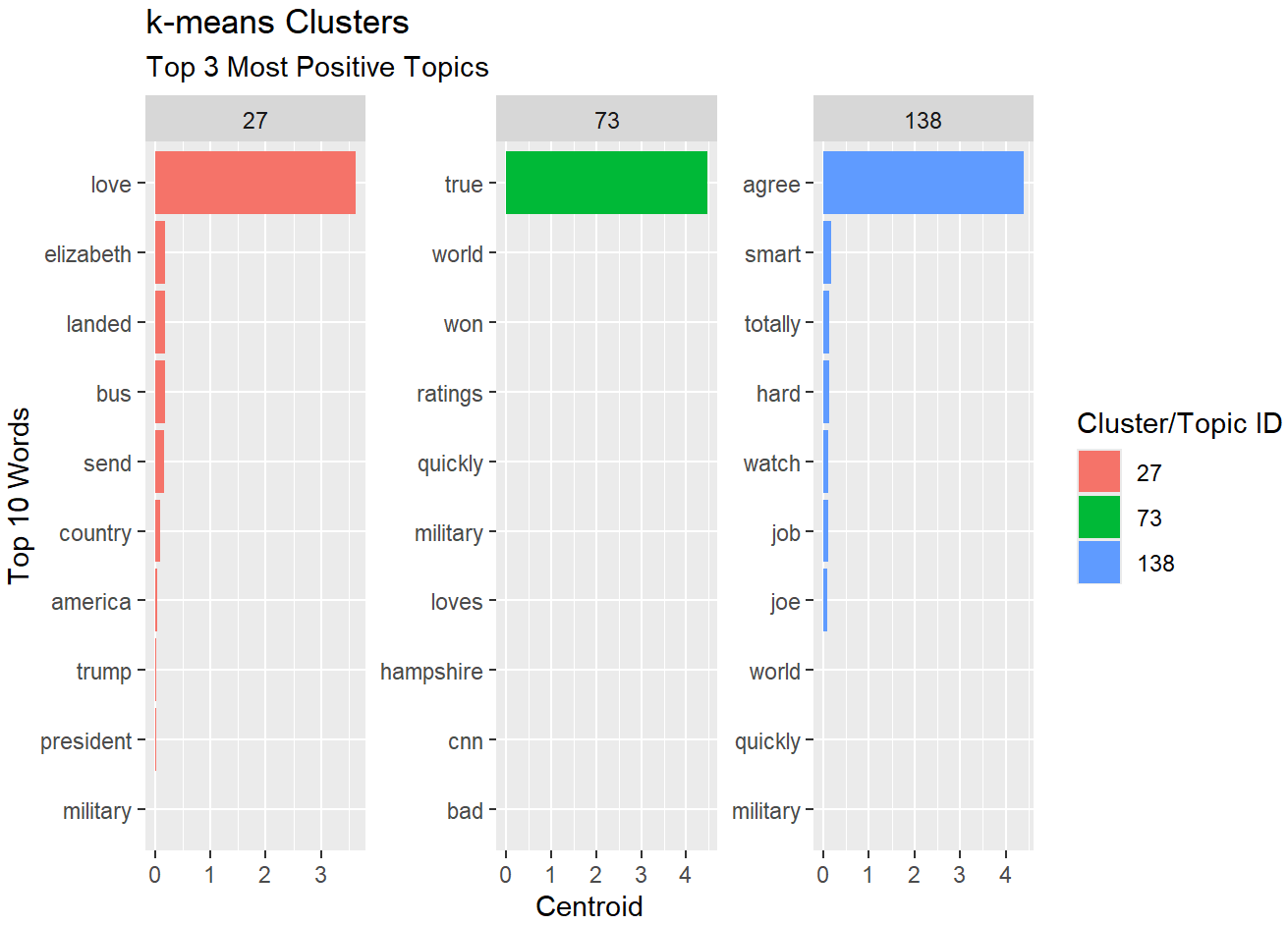
Re-run the previous analysis, except now look at Trump's last year in office, running from January 1st 2020 to January 8th 2021 (when he was kicked off Twitter). What are the top 3 positive topics and the top 3 negative topics? (Just use the same k value you identified in your first elbow plot for this analysis.)

```
# a.
dtm <- tweet_words %>%
  filter(Tweeting.date > as.Date('2020-01-01') & Tweeting.date < as.Date('2021-01-08'),
         word != 'amp') %>%
  count(document,word) %>%
  group_by(word) %>%
  mutate(tot_n = sum(n)) %>%
  ungroup() %>%
  filter(tot_n >20)

#b.
dtm.tfidf <- bind_tf_idf(tbl = dtm, term = word, document = document, n = n)

#c.
castdtm <- cast_dtm(data = dtm.tfidf, document = document, term = word, value = tf_idf)

#e. (Skip d. here...no need to take forever re-running the elbow plot)
km_out <- kmeans(castdtm,
                 centers = 250,
                 nstart = 5)

km_out_tidy <- tidy(km_out) %>%
  gather(word,mean_tfidf,-size,-cluster,-withinss) %>%
  mutate(mean_tfidf = as.numeric(mean_tfidf))


cluster_sentiment <- km_out_tidy %>%
  inner_join(nrc %>%
               filter(sentiment %in% c('positive','negative'))) %>%
  select(size,cluster,word,mean_tfidf,sentiment) %>%
  spread(sentiment,mean_tfidf,fill = 0) %>%
  mutate(net_sentiment = positive - negative) %>%
  group_by(cluster,size) %>%
  summarise(negative = mean(negative),
            positive = mean(positive),
            net_sentiment = mean(net_sentiment),.groups = 'drop') %>%
  ungroup() %>%
  filter(size > 10)
```

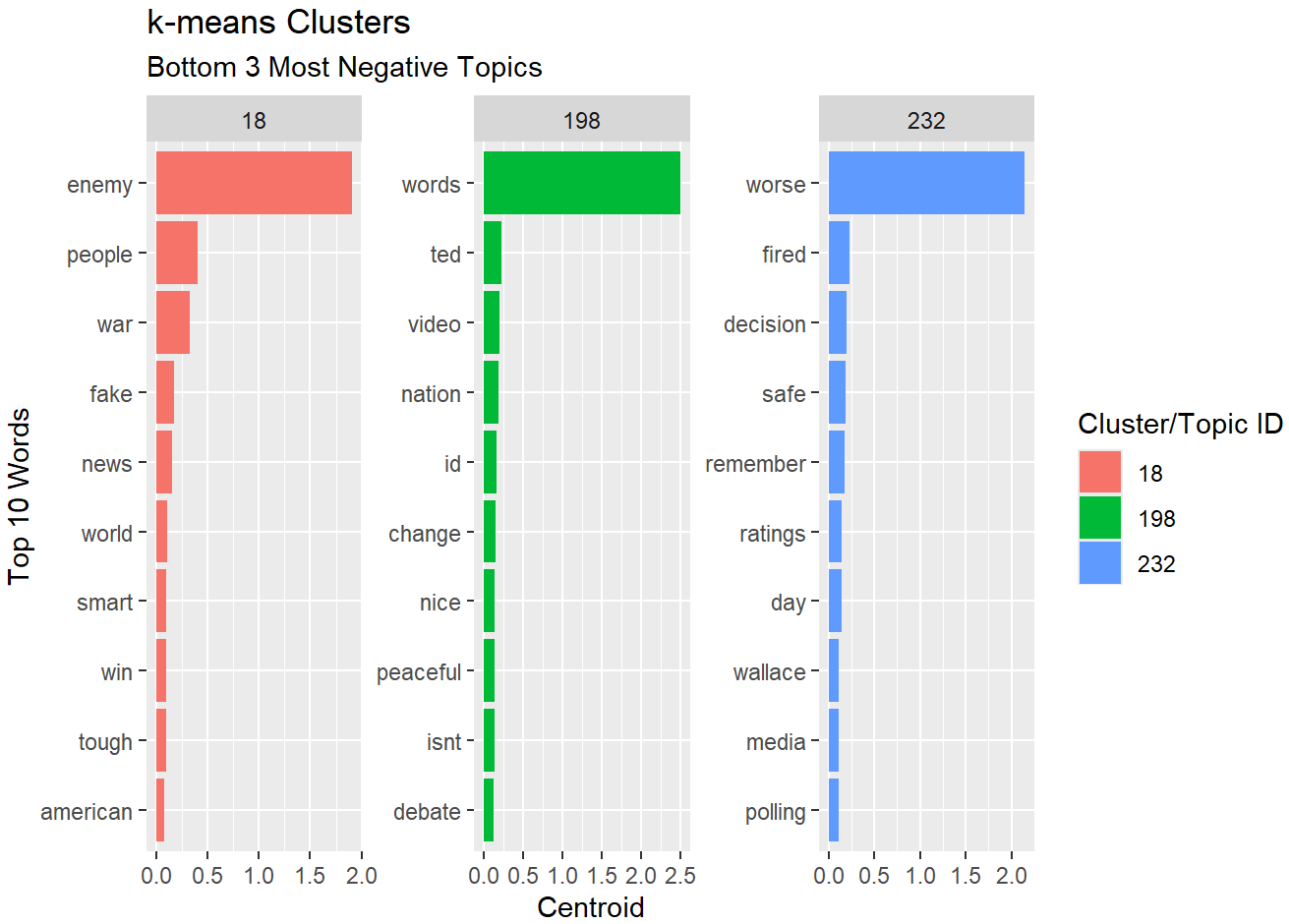
```
## Joining with `by = join_by(word)`
```

```
## Warning in inner_join(., nrc %>% filter(sentiment %in% c("positive", "negative"))): Detected
an unexpected many-to-many relationship between `x` and `y`.
## ℹ Row 11251 of `x` matches multiple rows in `y`.
## ℹ Row 415 of `y` matches multiple rows in `x`.
## ℹ If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```
#f.
top_sentiment <- cluster_sentiment %>%
  arrange(desc(net_sentiment)) %>%
  slice(1:3)

km_out_tidy %>%
  filter(cluster %in% top_sentiment$cluster) %>%
  group_by(cluster) %>%
  arrange(-mean_tfidf) %>%
  slice(1:10) %>%
  ggplot(aes(x = mean_tfidf,y = reorder(word,mean_tfidf),
             fill = factor(cluster))) +
  geom_bar(stat = 'identity') +
  facet_wrap(~cluster,scales = 'free') +
  labs(title = 'k-means Clusters',
       subtitle = 'Top 3 Most Positive Topics',
       x = 'Centroid',
       y = 'Top 10 Words',
       fill = 'Cluster/Topic ID')
```



```
bottom_sentiment <- cluster_sentiment %>%
  arrange(net_sentiment) %>%
  slice(1:3)

km_out_tidy %>%
  filter(cluster %in% bottom_sentiment$cluster) %>%
  group_by(cluster) %>%
  arrange(-mean_tfidf) %>%
  slice(1:10) %>%
  ggplot(aes(x = mean_tfidf,y = reorder(word,mean_tfidf),
             fill = factor(cluster))) +
  geom_bar(stat = 'identity') +
  facet_wrap(~cluster,scales = 'free') +
  labs(title = 'k-means Clusters',
       subtitle = 'Bottom 3 Most Negative Topics',
       x = 'Centroid',
       y = 'Top 10 Words',
       fill = 'Cluster/Topic ID')
```



- The top 3 most positive topics in 2020 are hard to determine based on the highest scoring words. They are dominated by a single highest scoring word, suggesting that there aren't many tweets assigned to each of them. The bottom 3 most negative topics are also harder to determine, but seemingly about the media once again.

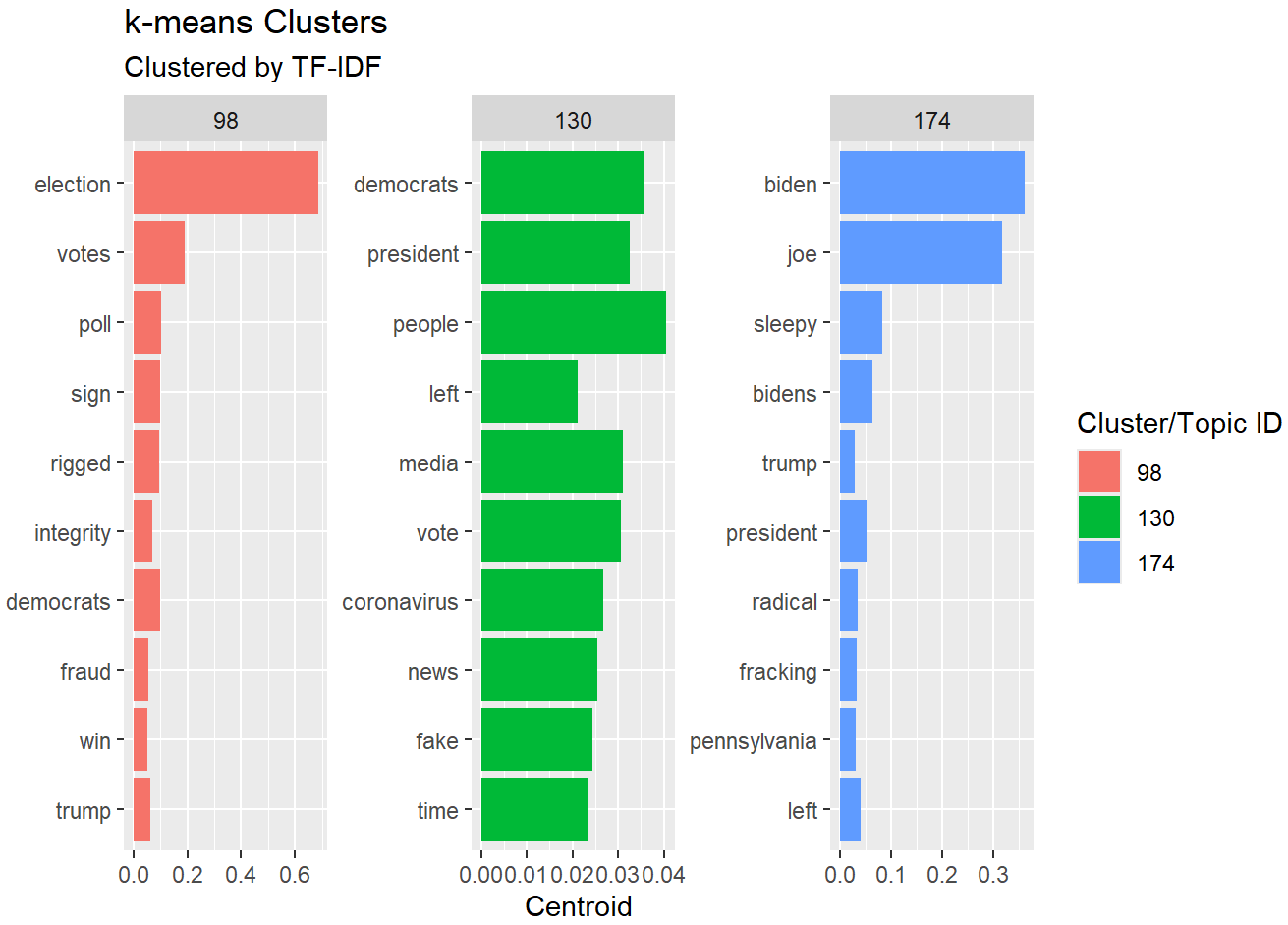
Extra Credit [2 points]

Which of Trump's topics are the most "popular", measured by total retweets? You will need to get creative with this final extra credit question. Broadly, you will need to link each tweet to the topic it was assigned as well as the number of retweets it received. This will require you to exploit the fact that the `km_out$cluster` data includes both the cluster to which each observation was assigned, as well as the tweet ID associated with that observation. Once you have created this lookup object, you can then link the original `tweet_words` dataset with the clusters. (NOTE: not every tweet will be assigned to a cluster, since we are dropping many of them.) This will require you to pay attention to object types (the names of the `km_out$cluster` are character, but the document IDs are stored as numeric in the `tweet_words` object), think creatively about how to merge the datasets, be aware of `NA`'s and think about how to deal with them, and then finally analyze the data once you have built it. Your end result should be, as above, the top 10 words associated with the top 3 most popular topics. Good luck!

```
lookup <- data.frame(document = as.numeric(names(km_out$cluster)),
                     topic = km_out$cluster) %>%
  as_tibble()

top_topics <- lookup %>%
  left_join(tweet_words) %>%
  select(document,topic,retweets) %>%
  group_by(topic) %>%
  summarise(tot_retweets = sum(retweets,na.rm=T),
         mean_retweets = mean(retweets,na.rm=T)) %>%
  ungroup() %>%
  arrange(desc(tot_retweets)) %>%
  slice(1:3)
```

```
## Joining with `by = join_by(document)`
```

```
km_out_tidy %>%
  filter(cluster %in% top_topics$topic) %>%
  group_by(cluster) %>%
  arrange(-mean_tfidf) %>%
  slice(1:10) %>%
  ggplot(aes(x = mean_tfidf,y = reorder(word,mean_tfidf),
             fill = factor(cluster))) +
  geom_bar(stat = 'identity') +
  facet_wrap(~cluster,scales = 'free') +
  labs(title = 'k-means Clusters',
       subtitle = 'Clustered by TF-IDF',
       x = 'Centroid',
       y = NULL,
       fill = 'Cluster/Topic ID')
```



- Trump's most popular topics are tweets about election fraud (topic 98), tweets about the Democrats / fake news / Covid-19 (topic 130), and tweets about Joe Biden (topic 174).