Campaigns

Gov 1347: Election Analytics

Soubhik Barari, Sun Young Park

November 24, 2020

Harvard University

Today's agenda

1. Basics of **automated text analysis**
   - How to represent corpora of text as data? ⇝ *document-feature matrix*
   - How to visualize the quantitative representation of text? ⇝ *word cloud* and *keyness plot*
   - Example corpus: Presidential inaugural addresses 1789-2017
2. Application: **2020 campaign evaluation**
   - Data: General election campaign speeches 2020
3. **Breakout room exercise**: **Trump's Twitter campaign**
   - Data: Trump's tweets since 2011

Basics of automated text analysis

Tool: Text-as-data analysis with `quanteda`

Open `10-Campaigns.R` in `R Studio`. Let's run the code together!

```
library(quanteda) ## package for analyzing text-as-data
library(tidyverse)
library(ggplot2)
```

- Example: Presidential inaugural addresses 1789-2017
  - pre-formatted as `quanteda` corpus[1]

```
View(data_corpus_inaugural)
```

[1] corpus: a collection of textual documents.

How to represent text as data?

Text analysis is frequently qualitative.

Q: When is quantitative analysis useful?

How to represent text as data?

Reduce complexity: Language is extraordinarily complex, with subtlety and nuance. We need to represent documents as straightforward mathematical objects.

Pre-processing: What can be simplified? Which complexity can be removed?

1. Tokenize (using whitespace)
2. Remove grammatical structure: **"bag-of-words"** assumption
3. Remove punctuation
4. Remove capitalization
5. Remove stop words (ex: a, her, would...)
6. Stemming (ex: radicalize, radical $\rightsquigarrow$ radic)

```
toks_inaugural <- tokens(data_corpus_inaugural,
                         remove_punct = TRUE) %>%
    tokens_tolower() %>%
    tokens_remove(pattern = stopwords("en"))
```

How to represent text as data?

Document-feature matrix: Quantitative representation of corpus

```
dfm_inaugural <- dfm(toks_inaugural)
head(dfm_inaugural)
```

```
## Document-feature matrix of: 6 documents, 9,210 features (94.6% sparse) and 4
##                  features
## docs              fellow-citizens senate house representatives among
##   1789-Washington               1      1     2               2     1
##   1793-Washington               0      0     0               0     0
##   1797-Adams                    3      1     0               2     4
##   1801-Jefferson                2      0     0               0     1
##   1805-Jefferson                0      0     0               0     7
##   1809-Madison                  1      0     0               0     0
##                  features
## docs              vicissitudes incident life event filled
##   1789-Washington            1        1    1     2      1
##   1793-Washington            0        0    0     0      0
##   1797-Adams                 0        0    2     0      0
##   1801-Jefferson             0        0    1     0      0
##   1805-Jefferson             0        0    2     0      0
##   1809-Madison               0        0    1     0      1
## [ reached max_nfeat ... 9,200 more features ]
```

How to represent text as data?

Word-frequency matrix: Quantitative summarization of corpus

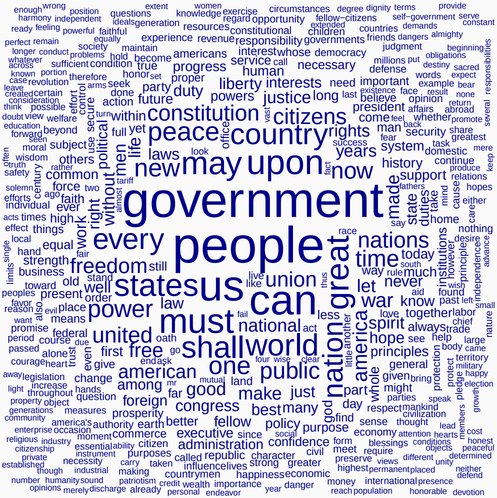
```
tstat_freq <- textstat_frequency(dfm_inaugural)
head(tstat_freq, 10)
```

```
##        feature frequency rank docfreq group
## 1       people       575    1      56   all
## 2   government       564    2      52   all
## 3           us       478    3      55   all
## 4          can       471    4      55   all
## 5         upon       371    5      47   all
## 6         must       366    6      51   all
## 7        great       340    7      55   all
## 8          may       338    8      53   all
## 9       states       333    9      46   all
## 10       shall       314   10      50   all
```

How to represent text as data?

Word cloud: Visual representation of corpus

```
textplot_wordcloud(dfm_inaugural)
```



A lot of choices with pre-processing

Select $>$ 5 letter words:

```
toks_inaugural <- tokens(data_corpus_inaugural,
                         remove_punct = TRUE) %>%
    tokens_tolower() %>%
    tokens_remove(pattern = stopwords("en")) %>%
    tokens_remove(pattern = "president") %>%
    tokens_select(min_nchar=6)
```

A lot of choices with visualization

Group by president:

```
dfm_inaugural <- dfm(toks_inaugural, groups = "President")
ndoc(toks_inaugural); ndoc(dfm_inaugural)
```

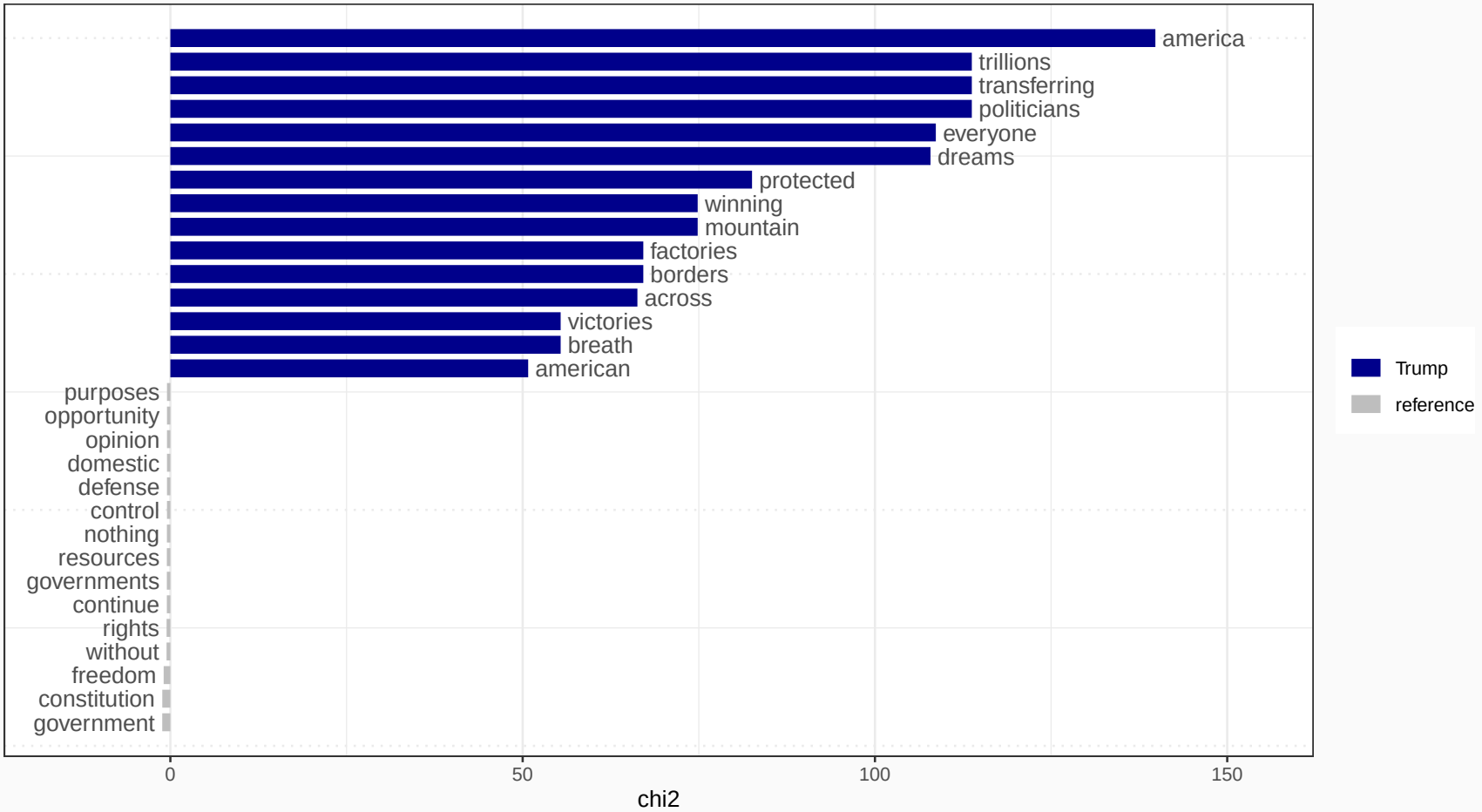
```
## [1] 58
```

```
## [1] 35
```

A lot of choices with visualization

Word "keyness" for a specific group of documents:

```
trump_keyness <- textstat_keyness(dfm_inaugural,
                                  target = "Trump")
textplot_keyness(trump_keyness, n = 15)
```



Application: 2020 campaign evaluation

Application: 2020 campaign evaluation

Data: General election campaign speeches 2020 (Trump vs. Biden)

```
speech_df <- read_csv("campaignspeech_2019-2020.csv")
speech_corpus <- corpus(speech_df, text_field = "text",
                        docid_field = "url")
```

Pre-processing

Additionally, now let's...

- select **custom stop words**
- choose certain **n-grams** (e.g. bigrams)

```
speech_toks <- tokens(speech_corpus,
    remove_punct = TRUE,
    remove_symbols = TRUE,
    remove_numbers = TRUE,
    remove_url = TRUE) %>%
    tokens_tolower() %>%
    tokens_remove(pattern=c("joe","biden","donald","trump",
                            "president","kamala","harris")) %>%
    tokens_remove(pattern=stopwords("en")) %>%
    tokens_select(min_nchar=3) %>%
    tokens_ngrams(n=2)
```

Summarisation

```
speech_dfm <- dfm(speech_toks, groups = "candidate")
tstat_freq <- textstat_frequency(speech_dfm,
                                 groups = "candidate")
head(subset(tstat_freq, group == "Donald Trump"), 10)
```

```
##            feature frequency rank docfreq          group
## 1       thank_much       906    1       1 Donald Trump
## 2       four_years       768    2       1 Donald Trump
## 3    united_states       539    3       1 Donald Trump
## 4        great_job       435    4       1 Donald Trump
## 5        years_ago       374    5       1 Donald Trump
## 6        right_now       335    6       1 Donald Trump
## 7        going_win       334    7       1 Donald Trump
## 8          usa_usa       319    8       1 Donald Trump
## 9       want_thank       304    9       1 Donald Trump
## 10    thank_thank       297   10       1 Donald Trump
```

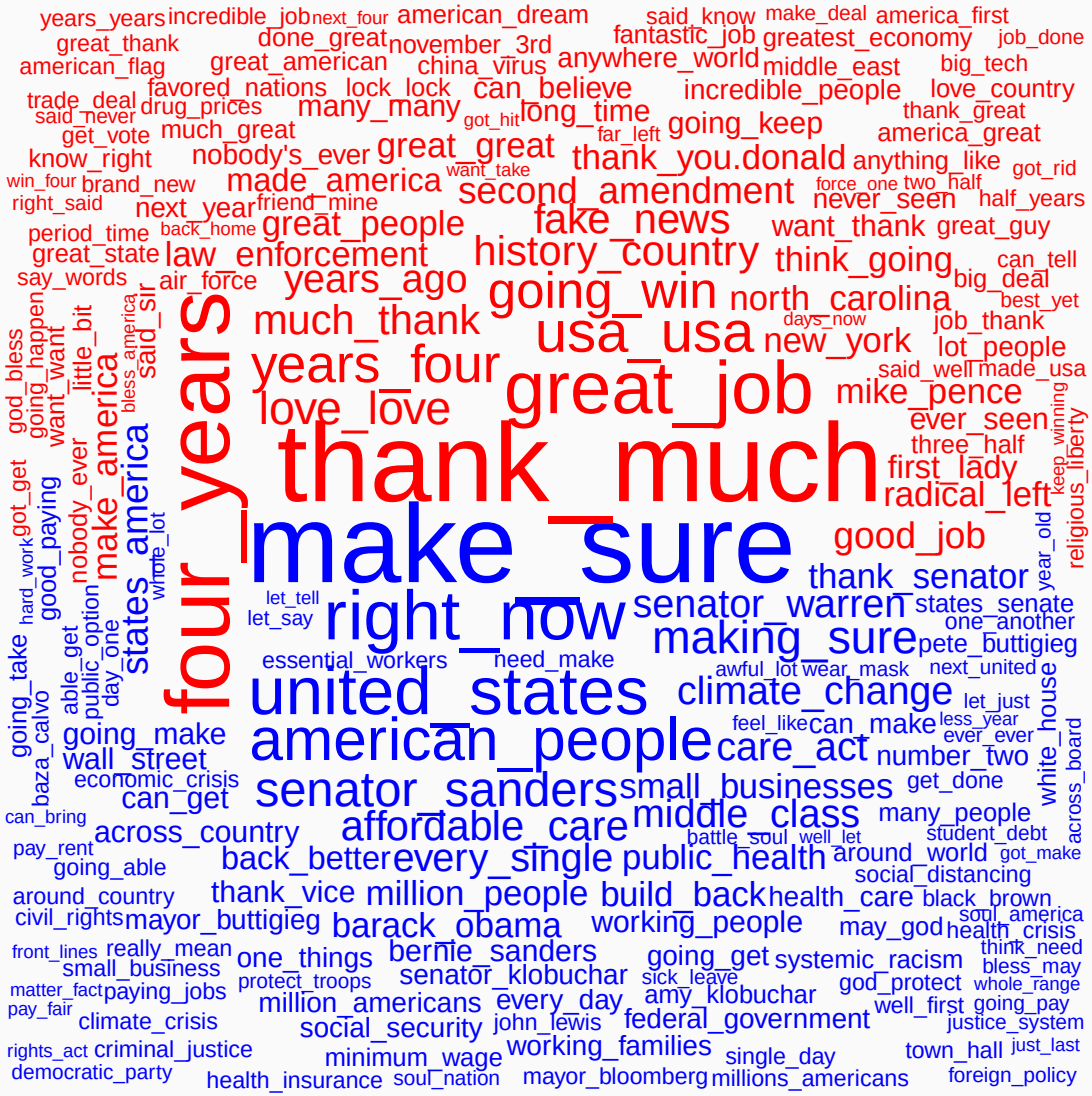
Summarisation

```
head(subset(tstat_freq, group == "Joe Biden"), 10)
```

```
##                feature frequency rank docfreq     group
## 202717   united_states       908    1       1 Joe Biden
## 202718       make_sure       810    2       1 Joe Biden
## 202719       right_now       713    3       1 Joe Biden
## 202720 american_people       500    4       1 Joe Biden
## 202721    every_single       292    5       1 Joe Biden
## 202722     white_house       284    6       1 Joe Biden
## 202723 senator_sanders       247    7       1 Joe Biden
## 202724     making_sure       238    8       1 Joe Biden
## 202725     thank_thank       236    9       1 Joe Biden
## 202726       going_get       222   10       1 Joe Biden
```

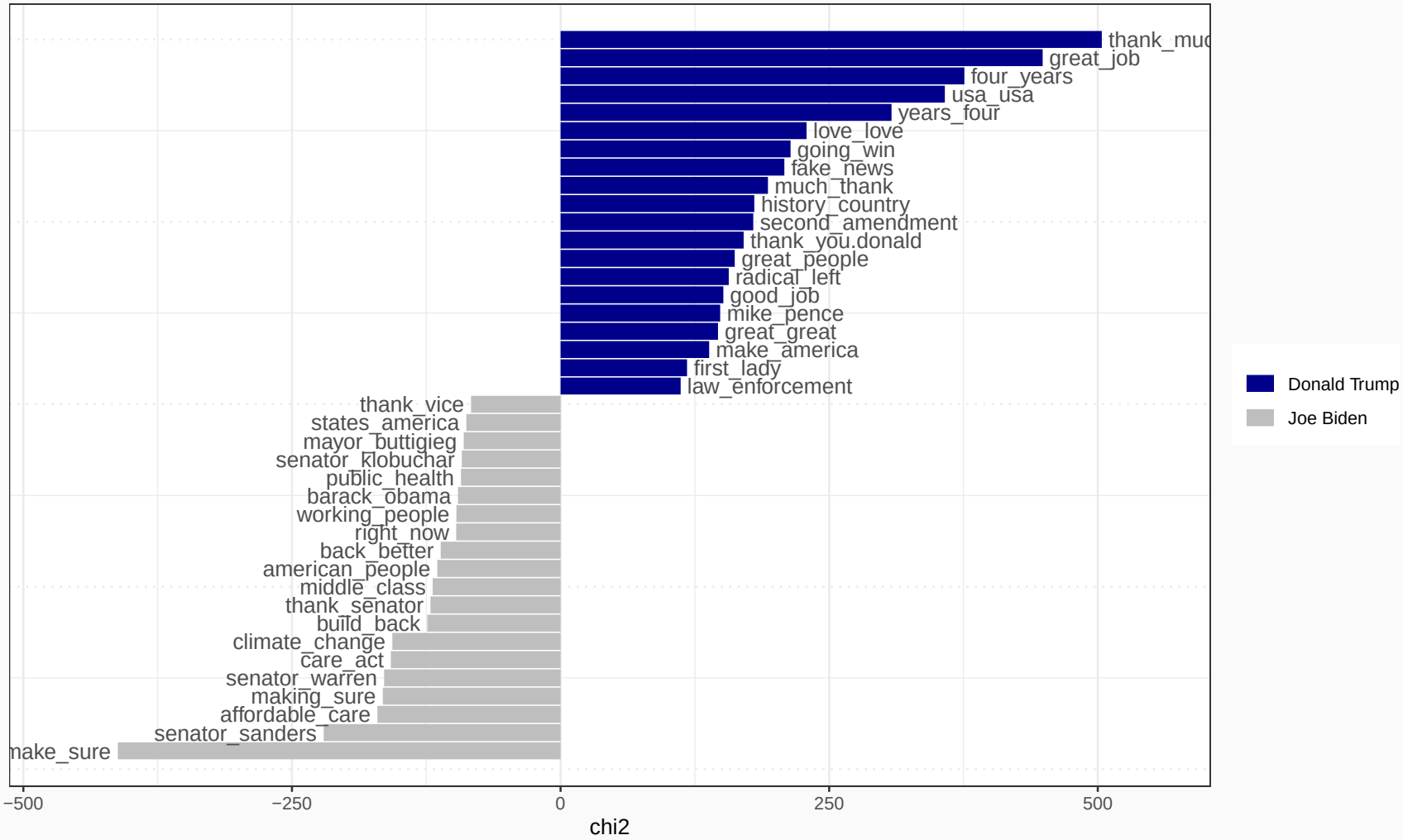
Visualization

```
textplot_wordcloud(speech_dfm, color = c("red", "blue"),
                   comparison = T, labelsize = 0)
```



Visualization

```
trump_keyness <- textstat_keyness(speech_dfm,
                                  target = "Donald Trump")
textplot_keyness(trump_keyness)
```



Breakout room exercise: Trump's Twitter Campaign

Data: Tweets by Trump since 2011

1. **Load** data: `trumptweets_2016-2020.csv`
2. **Pre-process** as you wish.
3. Make a **visual representation** of corpus.
4. You can try different pre-processing methods and see how the visualization changes. Or you can compare visualizations of different subsets of the corpus (e.g., early in the campaign; only tweets with many retweets.)

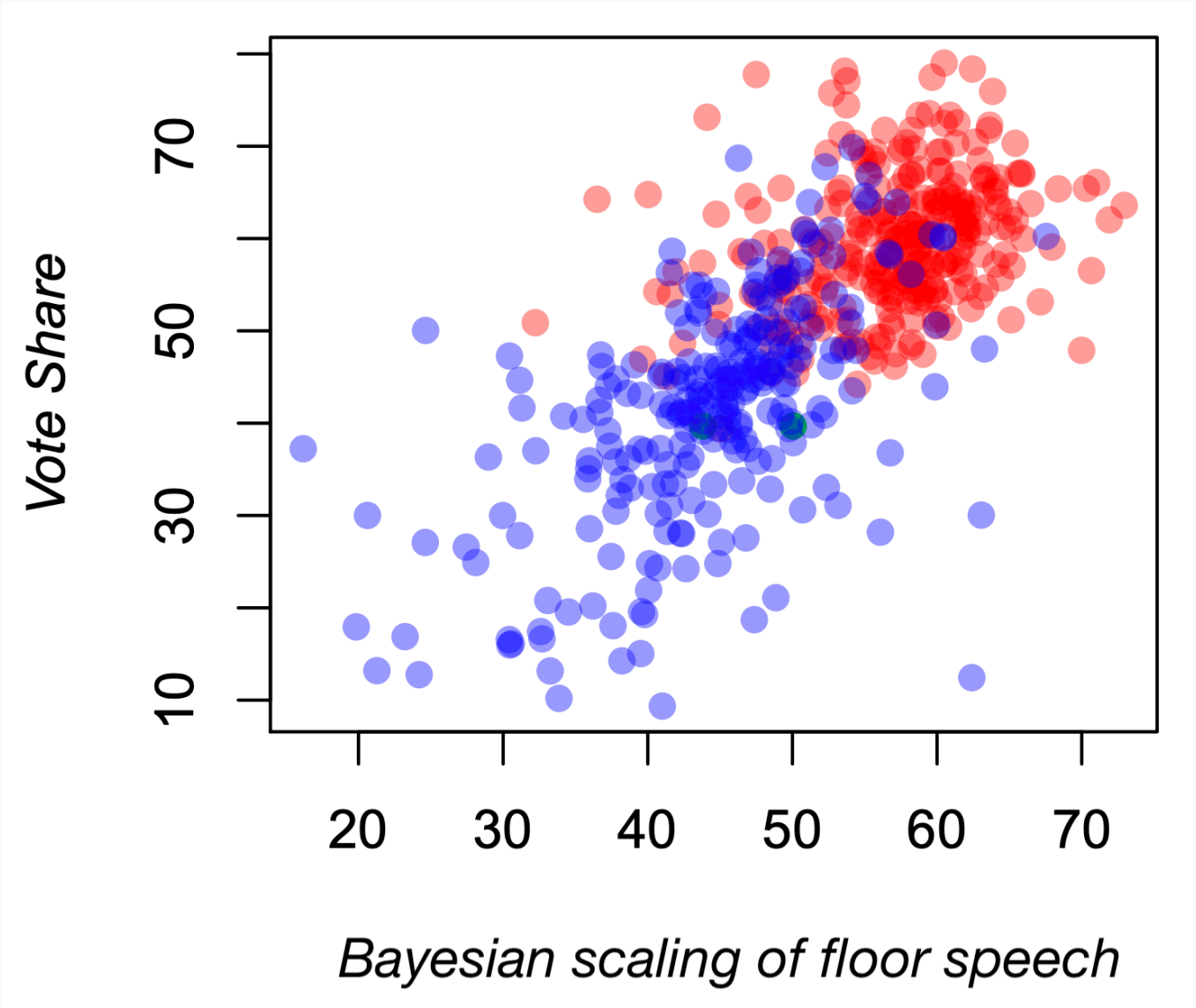
You may want to subset the data (`filter` in `dplyr` or `corpus_subset` in `quanteda`) if the data is too large for your computer to deal with. Refer to `https://quanteda.io/reference`.

Text-as-data analysis can do a lot more!

- Document comparison (ex: how similar are Trump's tweets to other Republicans?)
- Topic models (ex: automatically measure the proportion of topics across campaign speeches)
  - ⇝ `stm` package in `R`
- Sentiment analysis (ex: when/about what topics is Trump angriest?)
  - ⇝ `syuzhet` package in `R`

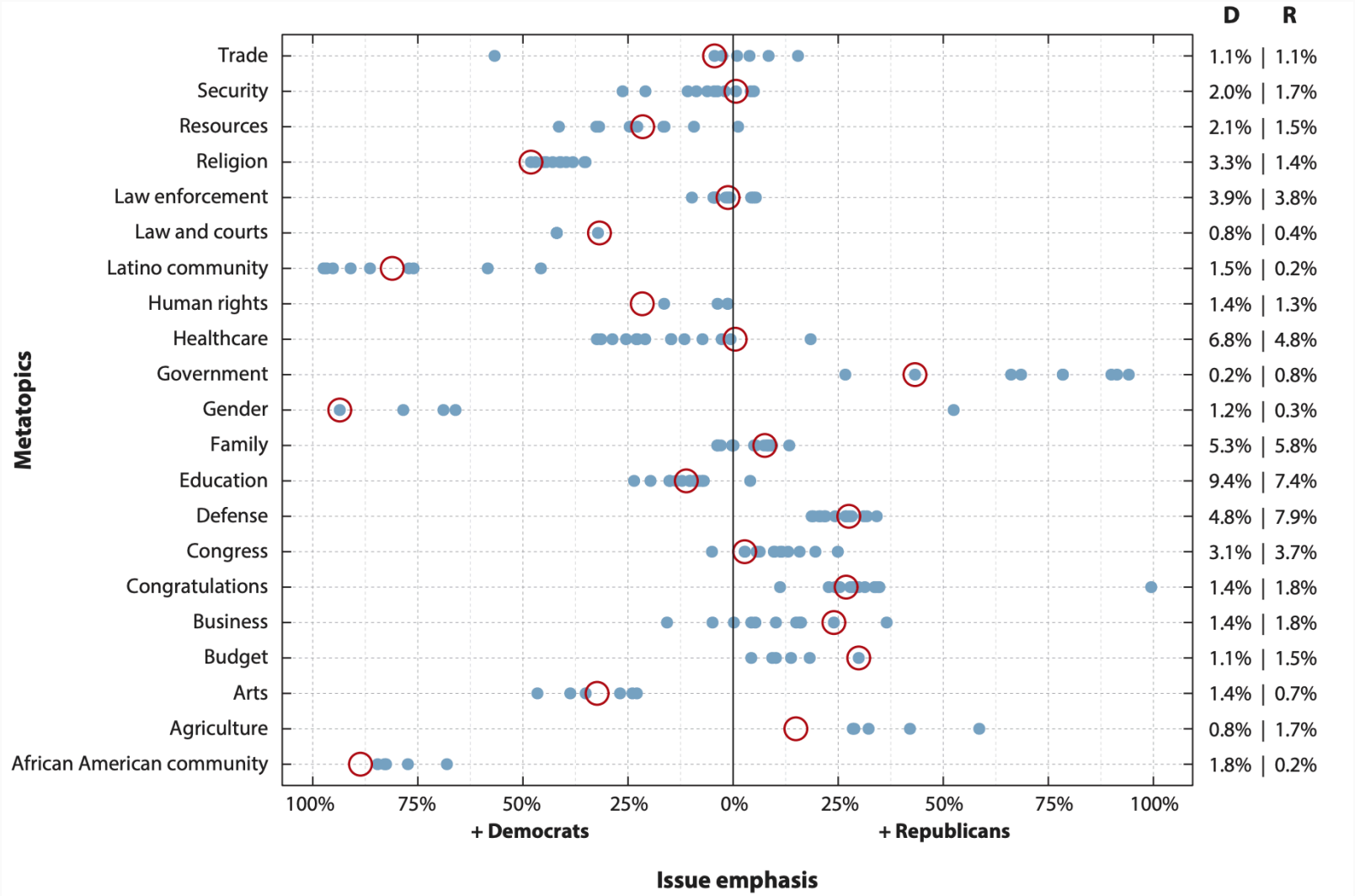
Text-as-data in political science

Measuring House members' ideology using floor speech:



Text-as-data in political science

Measuring partisan issue attention using floor speech:



Blog Extension / Presenters' Assignment

In *The Message Matters* (2009), Lynn Vavreck says that nearly all successful presidential campaigns talk about the economy in a certain way.

1. What exactly is that "way"?
2. Did Trump and/or Biden talk about the economy in "that way" during the campaign?

Hint: Use the two corpora from today's section to visualize how often Trump and/or Biden used a set of keywords that you think are substantively important. Refer to relevant documentation at `https://quanteda.io`.