Text as Data

Marek Chadim

2024-10-14

Pre-processing

Read all speech-files into a corpus using the tm command VCorpus. Turn the data into a tibble with columns containing the name of file containing text, the word and row number.

```
senator_corpus = VCorpus(DirSource(indir))
senators_td = senator_corpus |>
tidy() |>
select(id, text) |>
mutate(id=str_match(id,"-(.*).txt")[,2]) |>
unnest_tokens(word, text) |>
group_by(id) |>
mutate(row=row_number()) |>
ungroup()
```

Remove non-alphabetic characters, stopwords and other words that you find to be uninformative.

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

Generate variables with bigrams and trigrams for each senator.

```
senator_bigram =senators_td2 |>
arrange(id,row) |>
group_by(id) |>
mutate(bigram = str_c(lag(word,1),word, sep=" ")) |>
filter(row==lag(row,1)+1) |>
select(-word) |>
ungroup()

senator_trigram =senators_td2 |>
arrange(id,row) |>
group_by(id) |>
mutate(trigram= str_c(lag(word,2),lag(word,1),word, sep=" ")) |>
filter(row==lag(row,1)+1 & lag(row,1)==lag(row,2)+1) |>
select(-word) |>
ungroup()
```

Simple analysis

Compute overall frequency list

```
wordlist=senators_td2 |>
count(word,sort=TRUE) |>
```

```
filter(row_number()<50) |>
mutate(word = reorder(word,n)) |>
ggplot(aes(word,n)) + geom_bar(stat="identity") + xlab(NULL) + coord_flip()
```

What are the most frequent bigrams and trigrams?

```
bigramlist=senator_bigram |>
count(bigram,sort=TRUE)
bigramlist
```

```
## # A tibble: 733,719 x 2
##    bigram                 n
##    <chr>              <int>
##  1 unanimous consent  13278
##  2 social security     6432
##  3 health care         5779
##  4 federal government  5510
##  5 american people     5175
##  6 balanced budget     4977
##  7 madam president     3845
##  8 majority leader     3081
##  9 appropriations bill 2721
## 10 president clinton   2688
## # i 733,709 more rows
```

```
trigramlist=senator_trigram |>
count(trigram,sort=TRUE)
trigramlist
```

```
## # A tibble: 454,186 x 2
##    trigram                        n
##    <chr>                      <int>
##  1 balanced budget amendment   1983
##  2 campaign finance reform     1527
##  3 federal debt stood          1115
##  4 armed services committee     937
##  5 world war ii                 930
##  6 internal revenue service     718
##  7 partial birth abortion       713
##  8 social security trust        697
##  9 line item veto               695
## 10 foreign relations committee  670
## # i 454,176 more rows
```

Compute frequency lists for bigrams and trigrams by party. Plot a wordcloud for the 50 words most frequently

```
#Merge in party information.
sen_party = sen_party |>
  mutate(id= paste0(tolower(lname), "-", tolower(stateab)))

bigramlist_p <- senator_bigram |>
inner_join(sen_party) |>
rename(word=bigram) |>
count(party,word,sort = TRUE) |>
group_by(party) |>
mutate(share=n()/sum(n()),rank=row_number()) |>
```

```
ungroup()
```

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

```
trigramlist_p <- senator_trigram |>
inner_join(sen_party) |>
rename(word=trigram) |>
count(party,word,sort = TRUE) |>
group_by(party) |>
mutate(share=n()/sum(n()),rank=row_number()) |>
ungroup()
```

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

```
#  Plot a wordcloud for the 50 words most frequently used by each party.
wordlist_p <- senators_td2 |>
inner_join(sen_party) |>
count(party,word,sort = TRUE) |>
group_by(party) |>
mutate(share=n()/sum(n()),rank=row_number()) |>
ungroup()
```

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

```
wordlist_p |>
select(word,party,n) |>
acast(word ~ party, value.var = "n", fill = 0) |>
comparison.cloud(max.words=50)
```

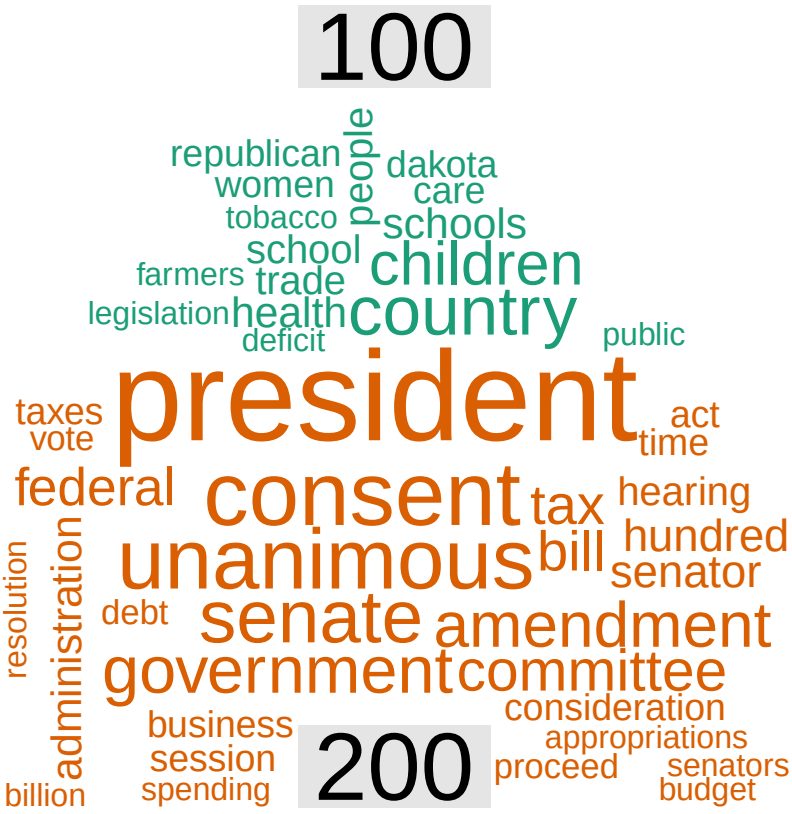
```
## Warning in comparison.cloud(acast(select(wordlist_p, word, party, n), word ~ :
## amendments could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(acast(select(wordlist_p, word, party, n), word ~ :
## subcommittee could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(acast(select(wordlist_p, word, party, n), word ~ :
## quorum could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(acast(select(wordlist_p, word, party, n), word ~ :
## defense could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(acast(select(wordlist_p, word, party, n), word ~ :
## trillion could not be fit on page. It will not be plotted.
```



Analysis

Lasso logit

Estimate a Lasso logit model predicting the party of the senator based on bigrams.

```
wordlist_s2 <- senator_bigram |>
rename(word=bigram) |>
inner_join(sen_party) |>
count(id,party,word,sort=TRUE) |>
ungroup()
```

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

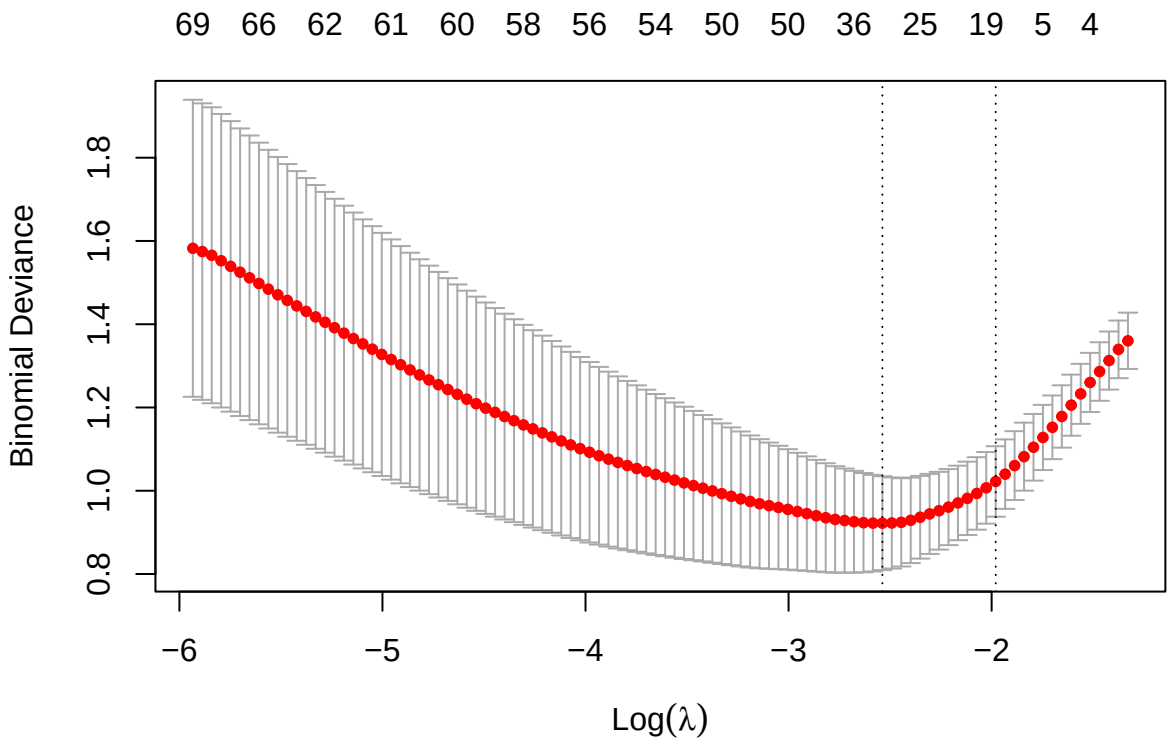
```
s <- wordlist_s2 |>
cast_sparse(id,word,n)
class(s)
```

```
## [1] "dgCMatrix"
## attr(,"package")
## [1] "Matrix"
```

```
s=s[order(rownames(s)),]
s2=wordlist_s2[order(rownames(wordlist_s2)),]

y=sen_party[order(sen_party$id),]
y<-as.matrix(y$party)
y<-as.factor(y)
```

```r
set.seed(1)
train <- sample(1:nrow(s), nrow(s) * 0.8)
cv_lasso <- cv.glmnet(s[train,], y[train], alpha = 1, family = "binomial")
plot(cv_lasso)
```




```r
lasso_pred <-predict(cv_lasso, s[-train,], s="lambda.min")
lasso_pred <- ifelse(lasso_pred<0,0,1)

table(predict=lasso_pred, truth=y[-train])
```

```
##         truth
## predict 100 200
##       0  10   0
##       1   0  10
```

```r
cv_lasso <- cv.glmnet(s, y, alpha = 1, family = "binomial")
lasso_best <- predict(cv_lasso, s = "lambda.min", type = "coefficients")
lasso_coef <- as.matrix(coef(cv_lasso, s = "lambda.min"))
coef_lasso <- data.frame(names = lasso_best@Dimnames[[1]][lasso_best@i + 1], coefficients = lasso_best@
```

Bigrams most predictive of party

```r
coef_lasso <- coef_lasso[coef_lasso$names != "(Intercept)", ]
coef_lasso |>
  arrange(desc(abs(coefficients))) |>
  head(10)
```

```
##                   names coefficients
## 1  taxpayers investment    1.1545188
## 2       america expects    0.9657745
## 3  subcommittee hearing    0.9282121
## 4      organized effort    0.7486120
## 5         freedom house    0.7038293
## 6   anticipate rollcall    0.6348434
## 7   community education   -0.6172622
## 8   encouraging private    0.6023084
## 9  thursday immediately    0.4815475
## 10         sec chairman   -0.4717381
```

LDA

Estimate al LDA topic model with 5 topic based on the speeches by the senators.

```
senators_td3 = senators_td2[!is.na(senators_td2$word),]
senators_td3 = senators_td3 |>
mutate(d=cumsum(word=="docno"))

droplist=c("text","doc","docno","") #
senators_td3 = senators_td3[!(senators_td3$word %in% droplist),]
senators_td3 = senators_td3 |>
mutate(x=ifelse(d!=lag(d,1)|id!=lag(id,1),1,0) )|>
mutate(speech= cumsum(ifelse(is.na(lag(d,1)),0,x)))

wordlist_s <- senators_td3 |>
count(speech,word,sort=TRUE) |>
 ungroup()

wordlist=senators_td3 |>
count(word,sort=TRUE)
wordlist
```

```
## # A tibble: 61,048 x 2
##    word            n
##    <chr>       <int>
##  1 president   92621
##  2 senator     60334
##  3 bill        56392
##  4 amendment   45313
##  5 senate      39877
##  6 people      39310
##  7 time        38801
##  8 federal     27341
##  9 legislation 27299
## 10 committee   25657
## # i 61,038 more rows
```

```
wordlist_m50=wordlist |>
filter(n>50) |>
select(word)

wordlist_s <- wordlist_s |>
inner_join(wordlist_m50)
```

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

```
s <- wordlist_s |>
cast_sparse(speech,word,n)
class(s)
```

```
## [1] "dgCMatrix"
## attr(,"package")
## [1] "Matrix"
```

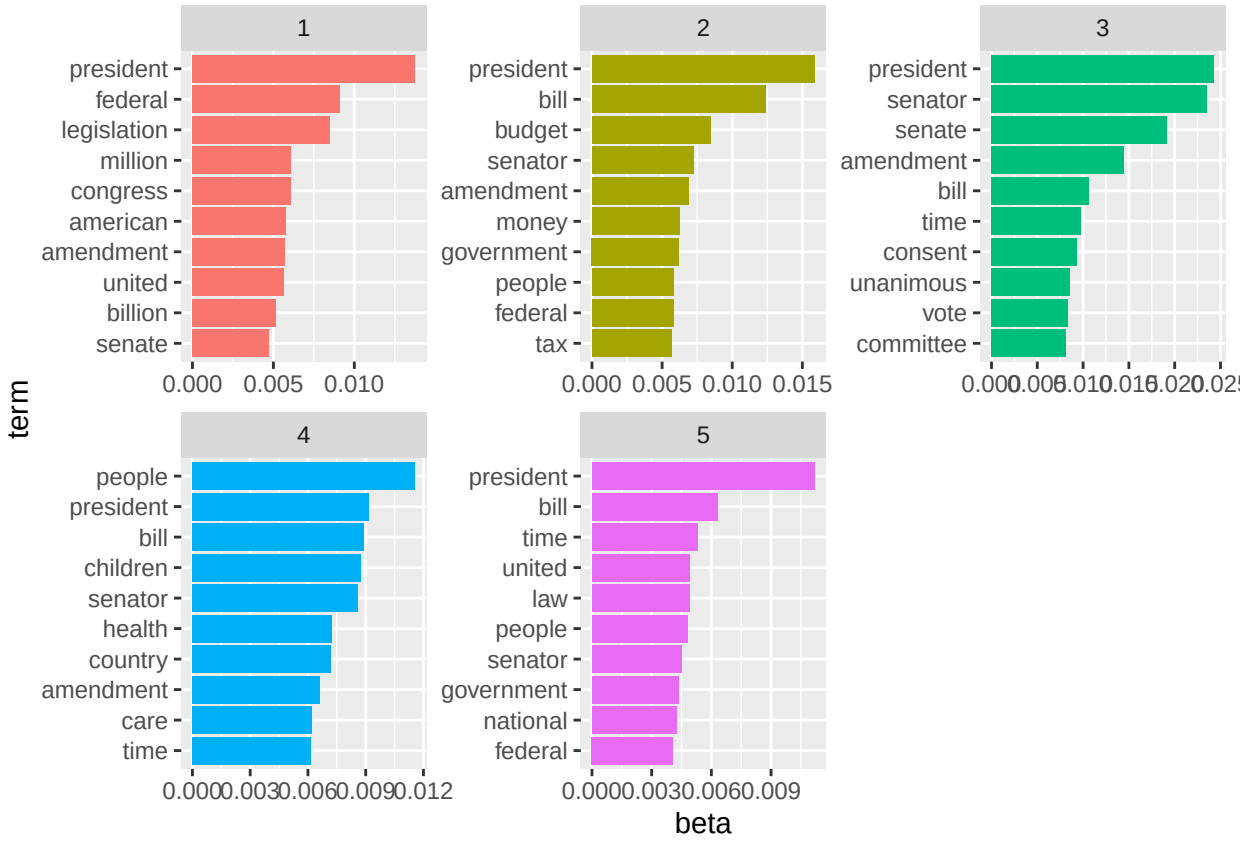
```
ap_lda10 <- LDA(s, k = 5, control = list(seed = 1234))
ap_topics <- tidy(ap_lda10, matrix = "beta")
ap_topics
```

```
## # A tibble: 48,740 x 3
##    topic term         beta
##    <int> <chr>       <dbl>
##  1     1 president 0.0138
##  2     2 president 0.0159
##  3     3 president 0.0243
##  4     4 president 0.00915
##  5     5 president 0.0112
##  6     1 senate    0.00470
##  7     2 senate    0.00111
##  8     3 senate    0.0192
##  9     4 senate    0.00440
## 10     5 senate    0.00393
## # i 48,730 more rows
```

What ten words are most characteristic of each topic?

```
#most common
ap_top_terms <- ap_topics |>
group_by(topic) |>
slice_max(beta, n = 10) |>
ungroup() |>
arrange(topic, -beta)

ap_top_terms |>
mutate(term = reorder_within(term, beta, topic)) |>
ggplot(aes(beta,term,fill=factor(topic))) + geom_col(show.legend = FALSE) + facet_wrap(~topic, scales =
```




```
#relative use measure
sumlogbeta <- ap_topics |>
mutate(logbeta = log(beta)) |>
group_by(term) |>
summarize(s_logbeta = sum(logbeta))

ap_top_terms2 <- ap_topics |>
inner_join(sumlogbeta) |>
mutate(logbeta = log(beta)) |>
mutate(term_score=beta*(log(beta)-s_logbeta/10)) |>
group_by(topic) |>
slice_max(term_score, n = 10) |>
ungroup() |>
arrange(topic, -term_score)
```

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

```
ap_top_terms2 |>
mutate(term = reorder_within(term, beta, topic)) |>
ggplot(aes(beta,term,fill=factor(topic))) + geom_col(show.legend = FALSE) + facet_wrap(~topic, scales =
```


1
hundred
eighty
ninety
sixty
barrels
salting
hawaiian
whitehurst
abate
ssig
0.000 0.001 0.002 0.003
2
yucca
cove
blm
yellowstone
glacier
anchorage
casks
alaskans
lwcf
nevadans
0e+00 1e−04 2e−04
3
xxii
jenkins
ntsb
usoc
atr
rnc
confirmations
alaskans
anchorage
biopsy
0e+00 5e−05 1e−04
4
forks
hmos
bushel
blanca
blizzard
biopsy
hudspeth
fargo
exploitative
durum
0e+00 1e−04 2e−04 3e−04
5
morrow
antipersonnel
fpi
atr
rnc
cda
sotomayor
salton
braude
idfa
0e+00 1e−04 2e−04
term
beta