Topic Modelling &
Latent Dirichlet Allocation
(LDA)

STQD 6114: Unstructured Data Analytics

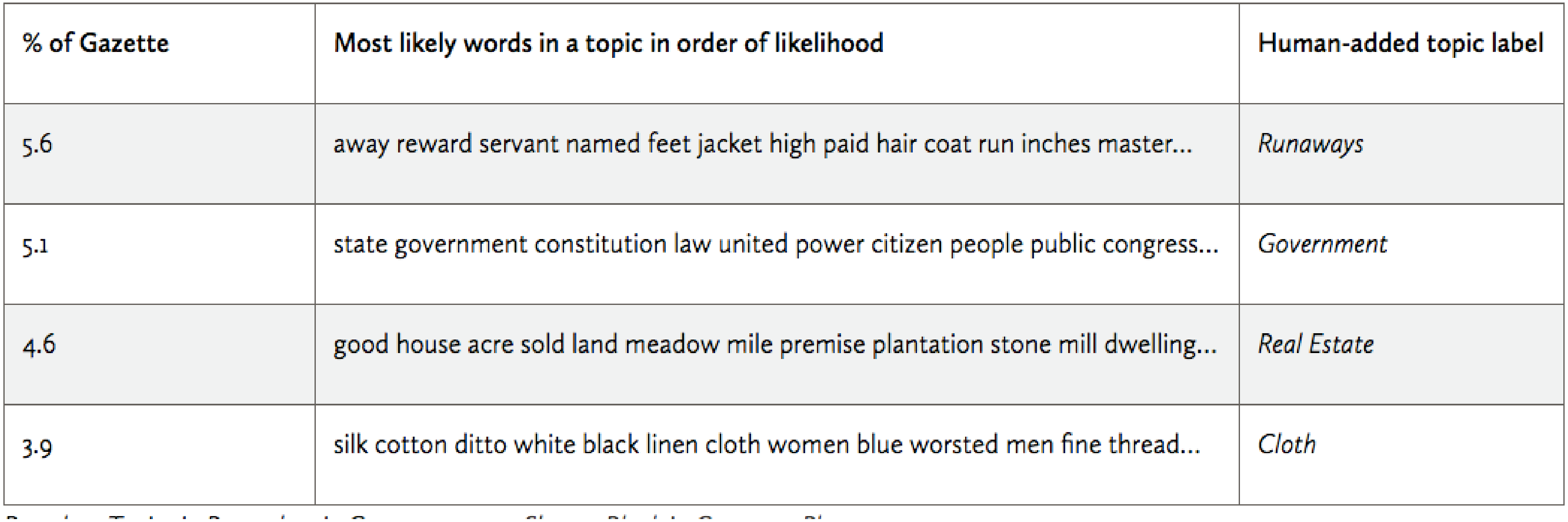
What is topic modelling?

- a statistical model to discover the topics that occur in a collection of documents
- method for finding a group of words (ie topics) from a collection of document that best represents the info in the collection
- it is a form of text mining

Why?

- Data exploration
- Rough idea on what is the structure/pattern/category of your text data
- Allow us to answer big picture questions quickly, & without human interventions

Example



% of Gazette	Most likely words in a topic in order of likelihood	Human-added topic label
5.6	away reward servant named feet jacket high paid hair coat run inches master…	*Runaways*
5.1	state government constitution law united power citizen people public congress…	*Government*
4.6	good house acre sold land meadow mile premise plantation stone mill dwelling…	*Real Estate*
3.9	silk cotton ditto white black linen cloth women blue worsted men fine thread…	*Cloth*

Example

- Topic modeling as
  - Qualitative Social Evidence
    - angry speech and patriotism over time
    - similar trend
  - Literary Theoretical Springboard
    - study on poem
    - the opaque features in a poem harden the machine’s task; in which human assessment is still needed
    - novel/book genre

More examples

Topic A : Family
Topic B: animals

topics are vaguely assigned by algorithm, not explicitly set by user

- Suppose you have these statements:
  - I love father
  - Mommy plays with sister
  - My hamster is cute
  - Sister likes rabbit
- Latent Dirichlet Allocation: a method to automatically discovering topics for a set of statements
- Referring to the example, and if you asked for 2 topics, LDA might produce the following results:
  - Sentences 1 and 2: 100% Topic A
  - Sentence 3: 100% Topic B
  - Sentence 4: 50% Topic A & 50% Topic B

The LDA model - How?

- LDA represents documents as mixtures of topics that contains words with certain probabilities
- Assume that the document is produced based on this process:
  - The number of words, N is decided; and distributed as Poisson (example)
  - Choose a topic mixture (according to Dirichlet distribution over a fixed set of K topics): for example; 2/3 on family and 1/3 on pet

- for each word, use the topic to generate the word; based on multinomial distribution. For example; with topic family, we might generate “mommy” with 30% probability, daddy 40% and so on
- then, LDA will try to backtrack this process to find a set of topics that are likely to have generated the collection
- hence, topic modeling is a way of extrapolating backward from a collection of documents to infer the topics that could have generated them

- Further reading:

http://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf

Example

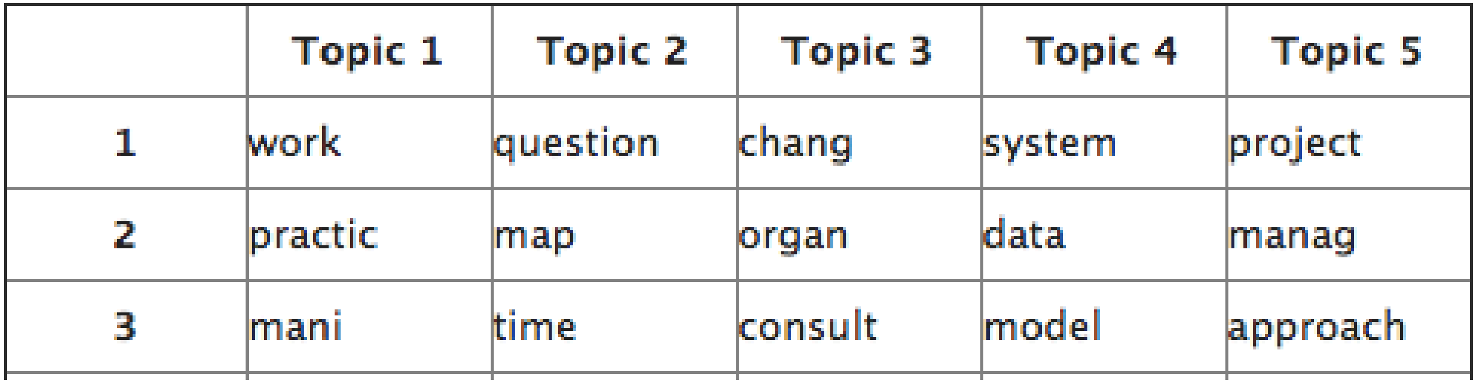
- Say that we pick 4 to be the number of words in our document
- Then, we decide that D will be ½ about family and ½ about pet
- Pick the first word from the family topic; → mommy
- Pick the second word from the pet topic; → hamster
- Pick the third word from the family topic; → sister
- Pick the fourth word from the pet topic; → rabbit
- Hence, the document generated will consists of these words: mommy-hamster-sister-rabbit

P(topic t | document d) --> the proportion of words in document d that are currently assigned to topic t
[gamma]

p(word w | topic t) --> the proportion of assignment to topic t
[beta]

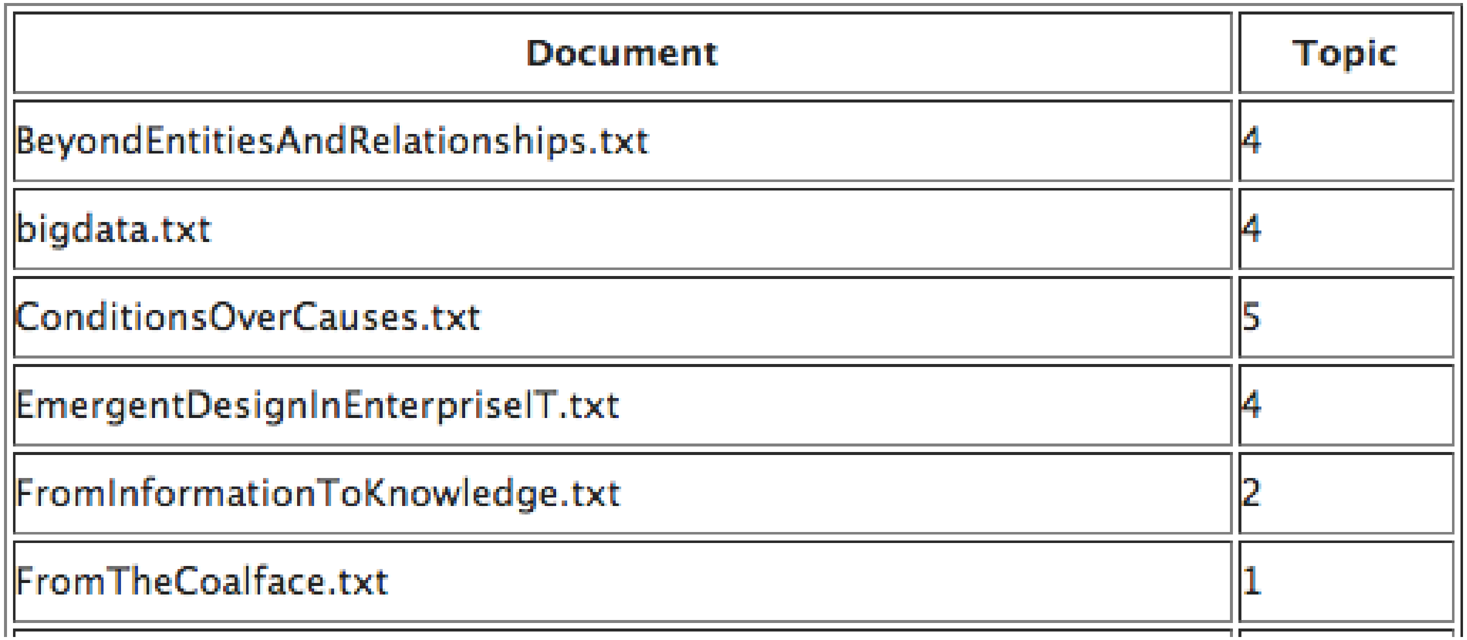
Expected output from LDA

- List of terms in each topic



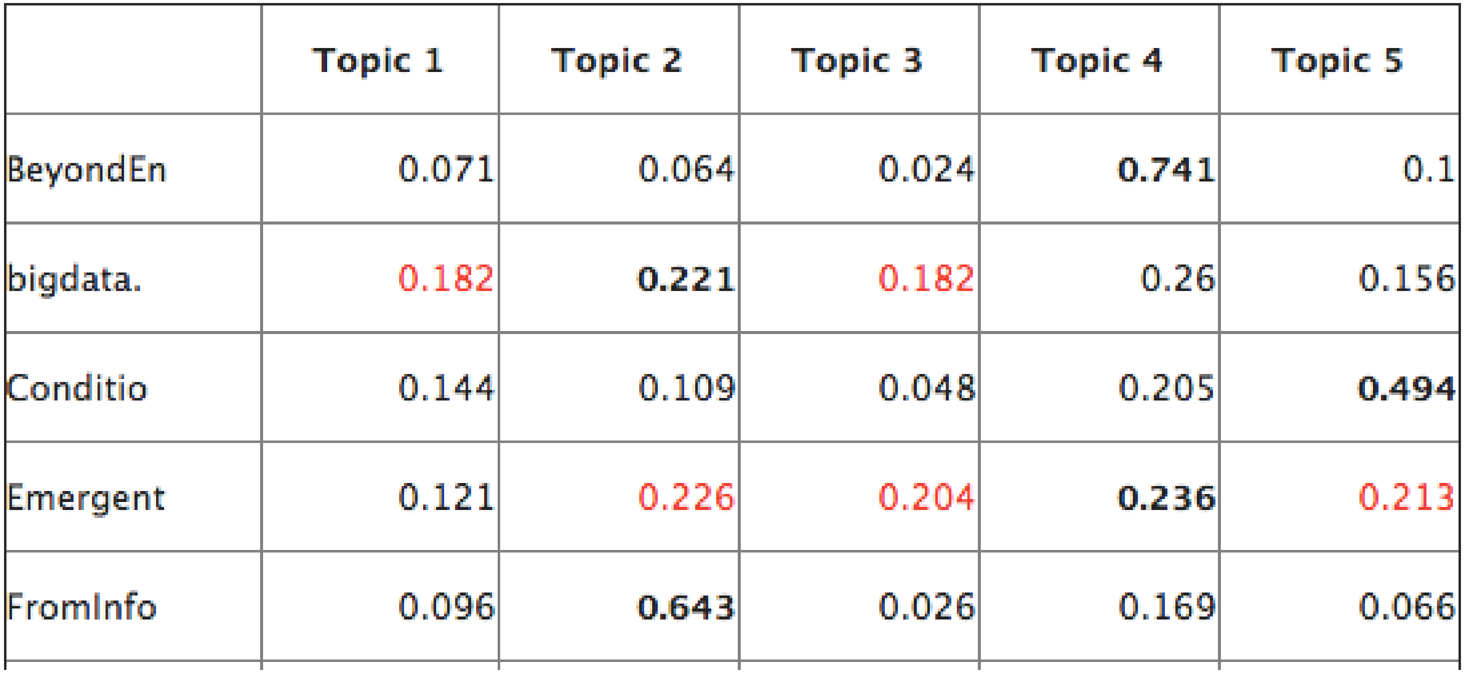
	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
1	work	question	chang	system	project
2	practic	map	organ	data	manag
3	mani	time	consult	model	approach

List of the document to the (primary) topic



Document	Topic
BeyondEntitiesAndRelationships.txt	4
bigdata.txt	4
ConditionsOverCauses.txt	5
EmergentDesignInEnterpriseIT.txt	4
FromInformationToKnowledge.txt	2
FromTheCoalface.txt	1

- Topic probabilities by documents



	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
BeyondEn	0.071	0.064	0.024	**0.741**	0.1
bigdata.	0.182	**0.221**	0.182	0.26	0.156
Conditio	0.144	0.109	0.048	0.205	**0.494**
Emergent	0.121	0.226	0.204	**0.236**	0.213
FromInfo	0.096	**0.643**	0.026	0.169	0.066

LDA in R

- Data source & R code:

https://cran.r-project.org/web/packages/topicmodels/vignettes/topicmodels.pdf

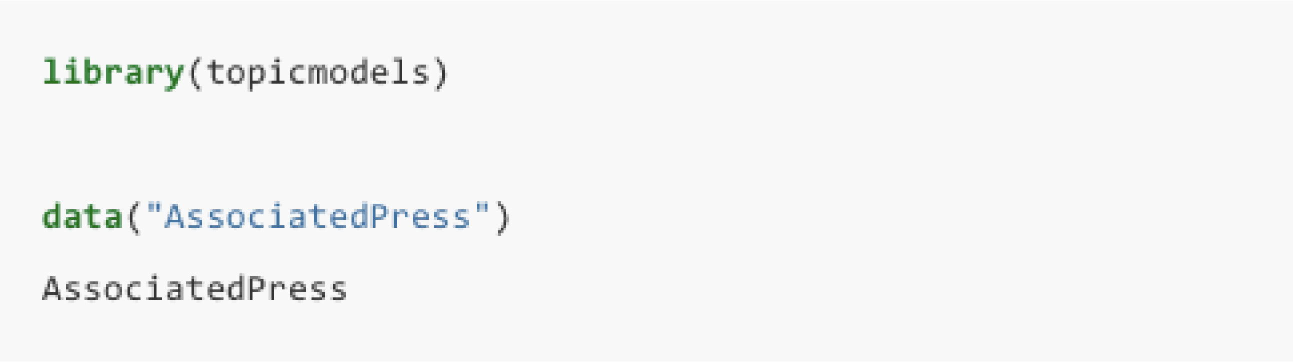
https://eight2late.wordpress.com/2015/09/29/a-gentle-introduction-to-topic-modeling-using-r/

https://www.tidytextmining.com/topicmodeling.html

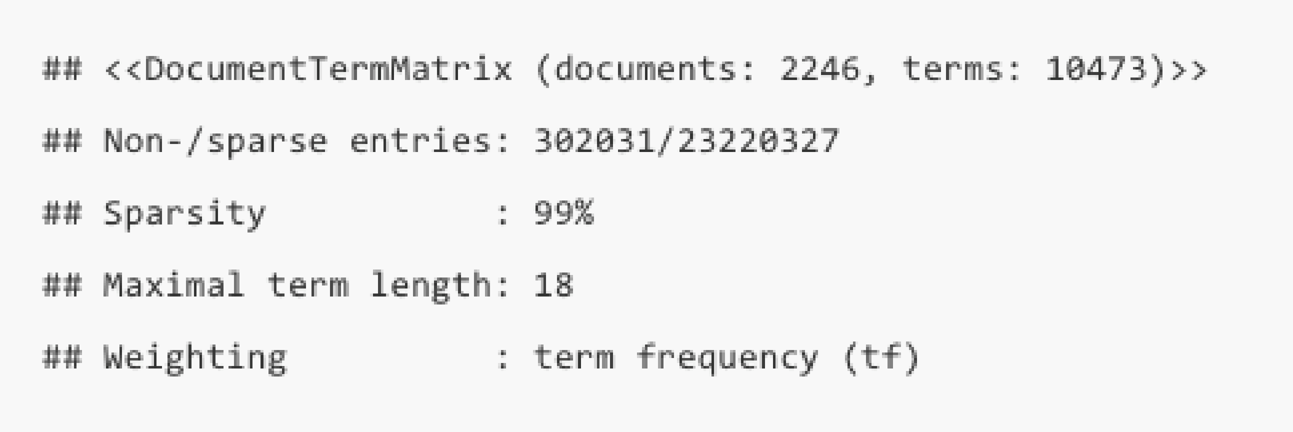
Example in R – topicmodels package

- prepping the data: transform to lower case, remove symbols, punctuations, general errors (different versions of English, stopwords
- create a document term matrix
- frequency of each word

Example in R



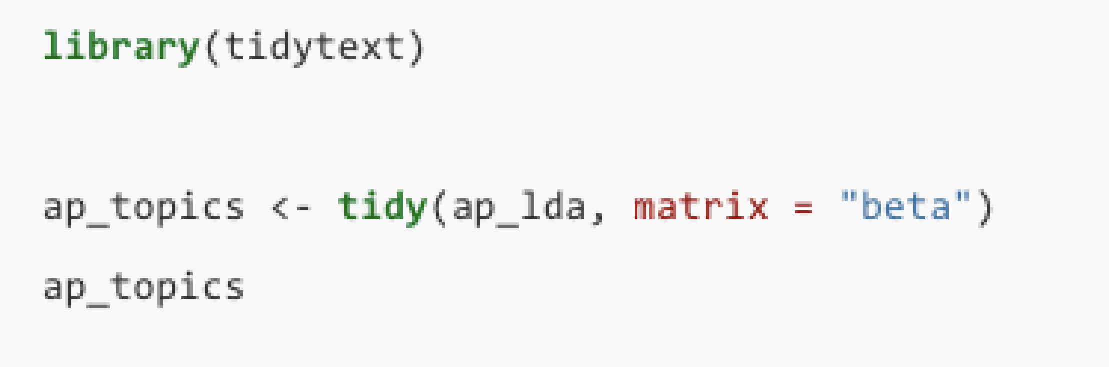
```r
library(topicmodels)


data("AssociatedPress")

AssociatedPress
```



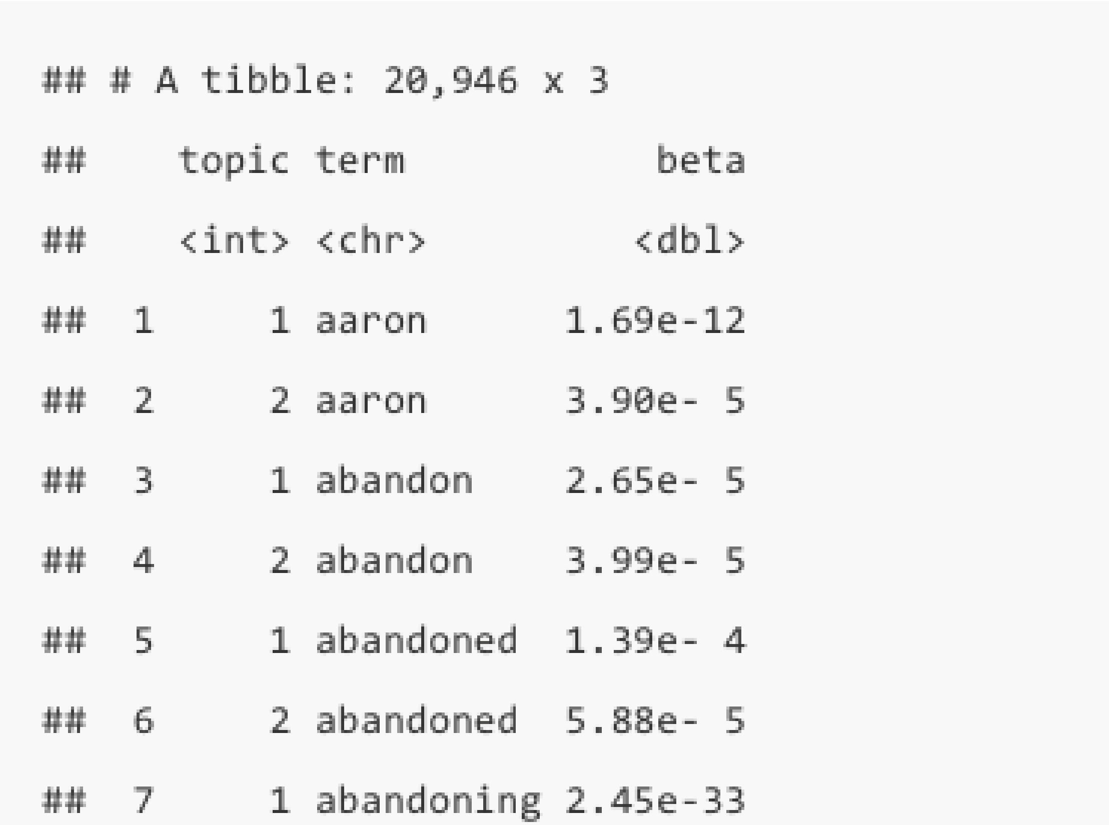
```
## <<DocumentTermMatrix (documents: 2246, terms: 10473)>>

## Non-/sparse entries: 302031/23220327

## Sparsity           : 99%

## Maximal term length: 18

## Weighting          : term frequency (tf)
```

```r
# set a seed so that the output of the model is predictable
ap_lda <- LDA(AssociatedPress, k = 2, control = list(seed = 1234))
ap_lda
```

```
## A LDA_VEM topic model with 2 topics.
```

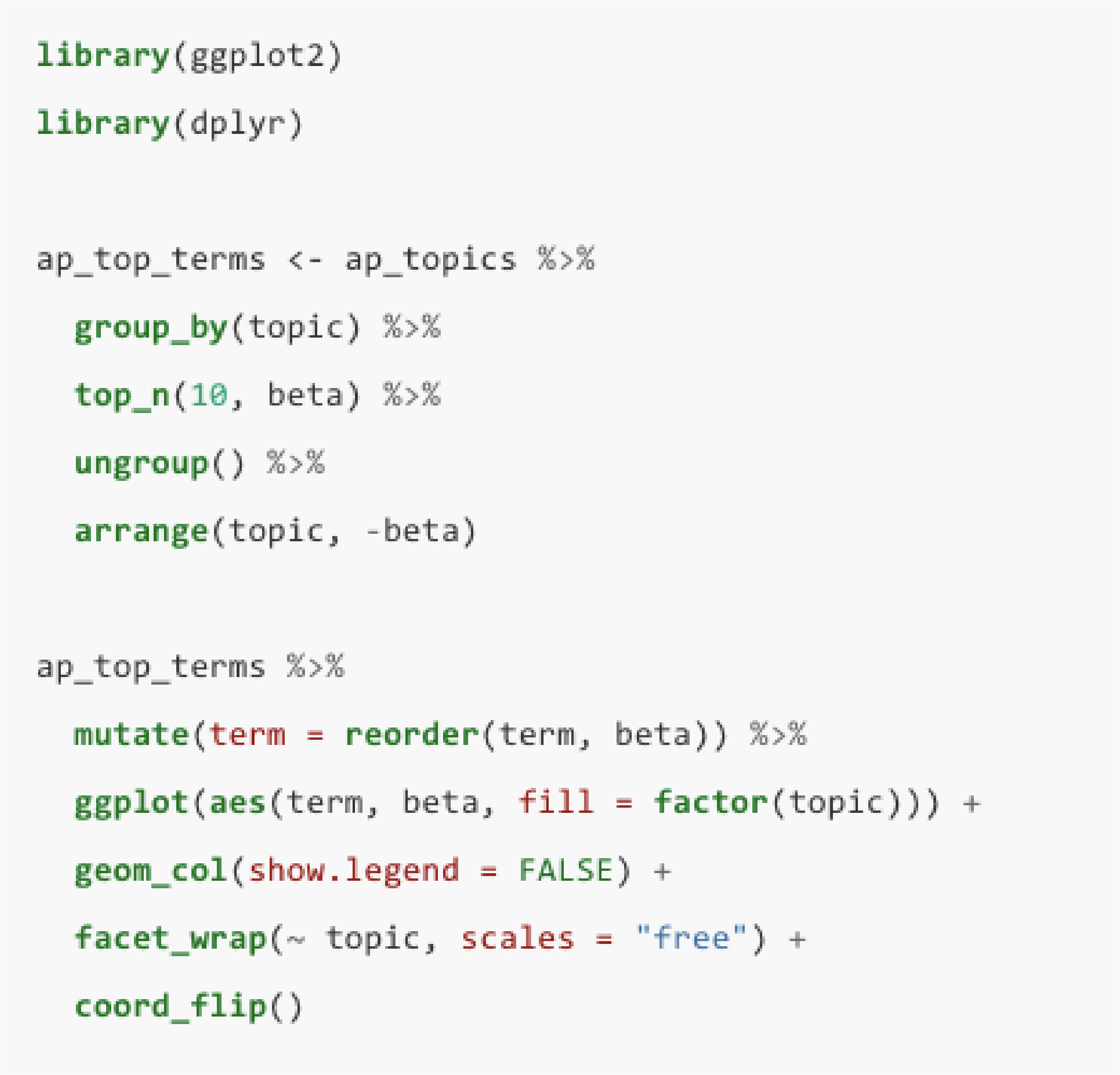


```
library(tidytext)

ap_topics <- tidy(ap_lda, matrix = "beta")
ap_topics
```



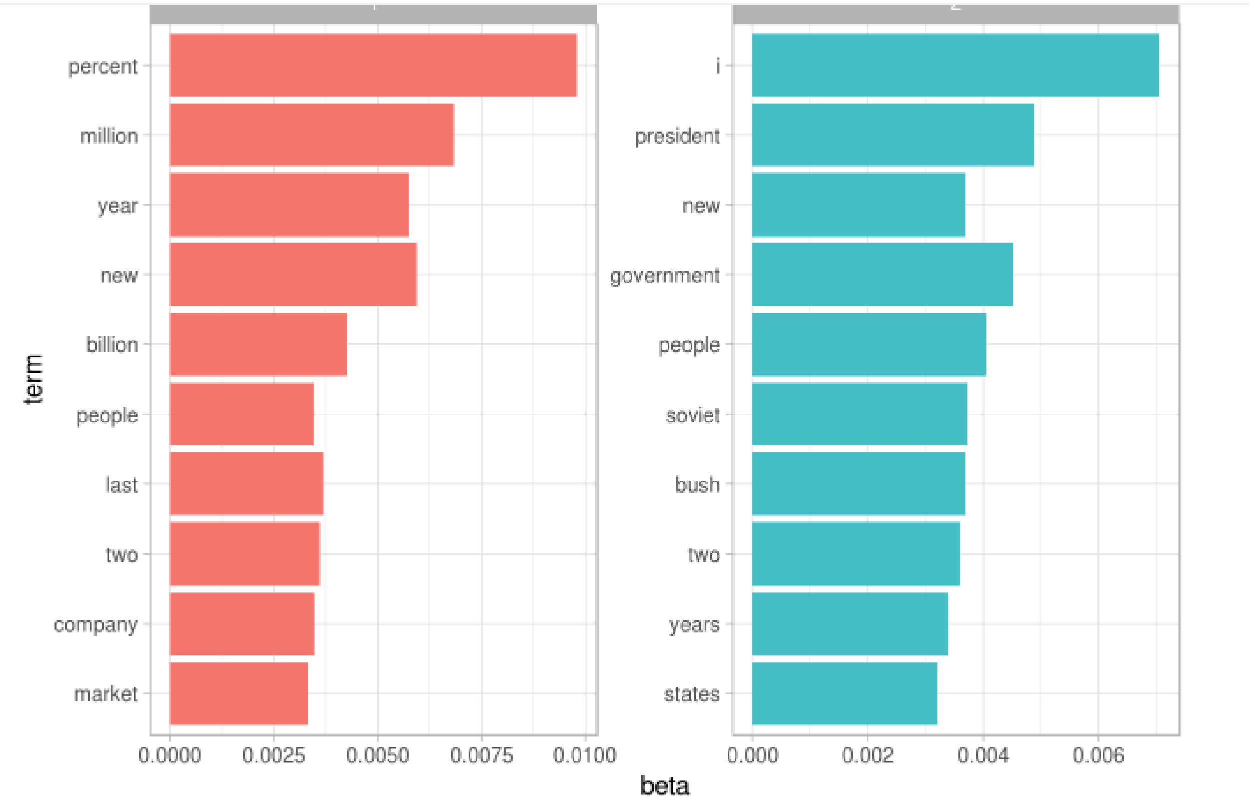
```
## # A tibble: 20,946 x 3
##    topic term           beta
##    <int> <chr>         <dbl>
##  1     1 aaron      1.69e-12
##  2     2 aaron      3.90e- 5
##  3     1 abandon    2.65e- 5
##  4     2 abandon    3.99e- 5
##  5     1 abandoned  1.39e- 4
##  6     2 abandoned  5.88e- 5
##  7     1 abandoning 2.45e-33
```

extracting the per-topic-per-word probabilities

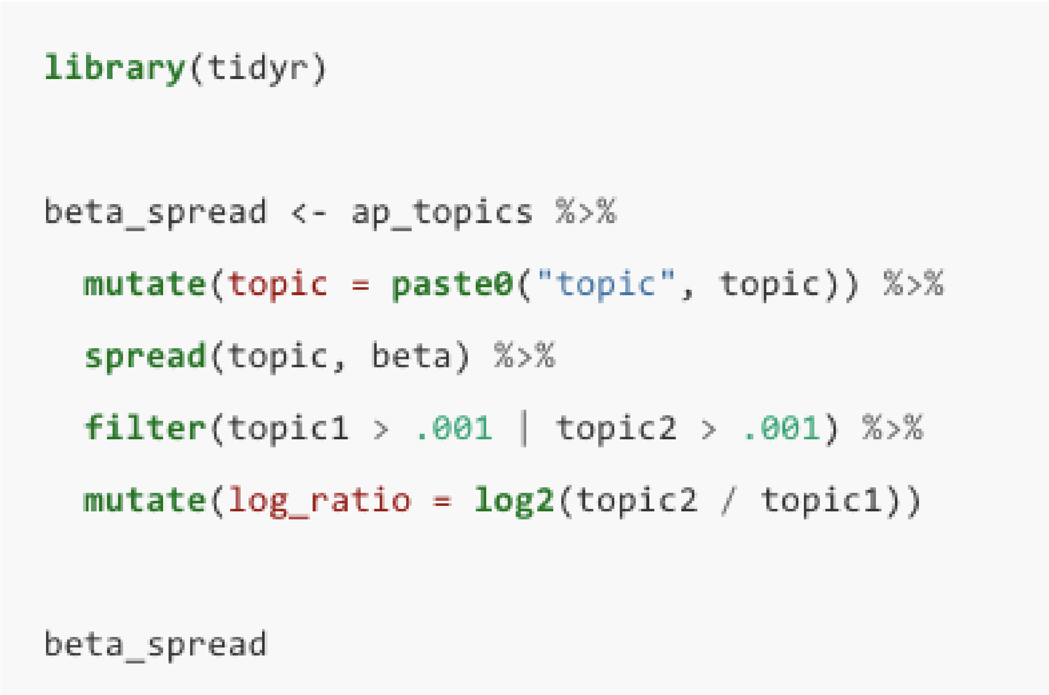


```
library(ggplot2)
library(dplyr)

ap_top_terms <- ap_topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

ap_top_terms %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```

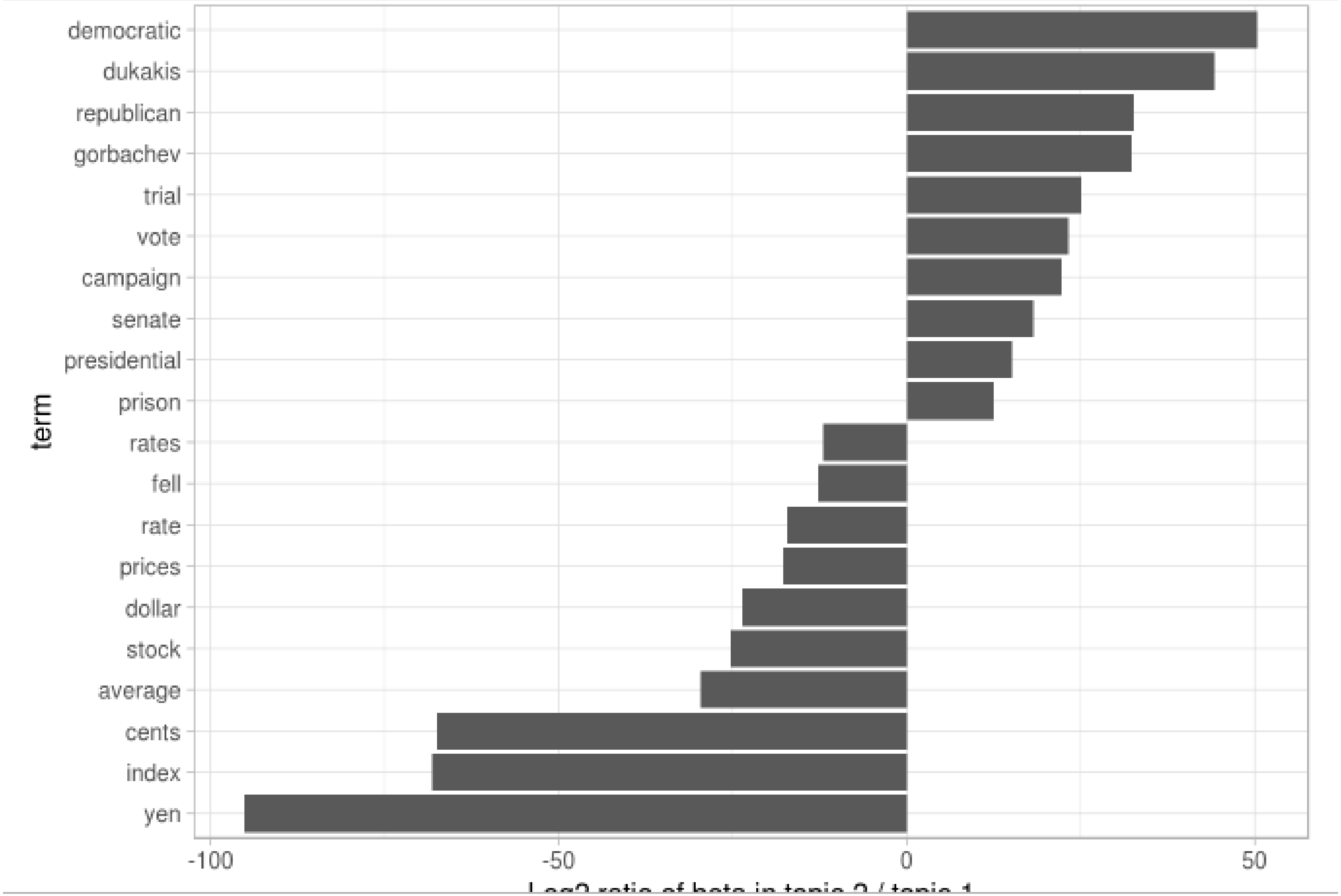
find the 10 terms that are most common within each topic

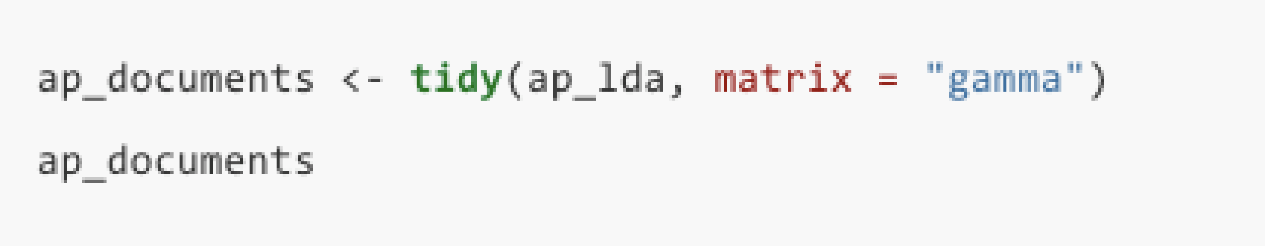


This visualization lets us understand the two topics that were extracted from the articles. The most common words in topic 1 include “percent”, “million”, “billion”, and “company”, which suggests it may represent business or financial news. Those most common in topic 2 include “president”, “government”, and “soviet”, suggesting that this topic represents political news. One important observation about the words in each topic is that some words, such as “new” and “people”, are common within both topics. This is an advantage of topic modeling as opposed to “hard clustering” methods: topics used in natural language could have some overlap in terms of words the terms that had the greatest difference in β between topic 1 and topic 2. This can be estimated based on the log ratio of the two: log (β2/β1). A log ratio is useful because it makes the difference symmetrical: β2 being twice as large leads to a log ratio of 1, while β1 being twice as large results in -1). To constrain it to a set of especially relevant words, we can filter for relatively common words, such as those that have a β greater than 1/1000 in at least one topic.

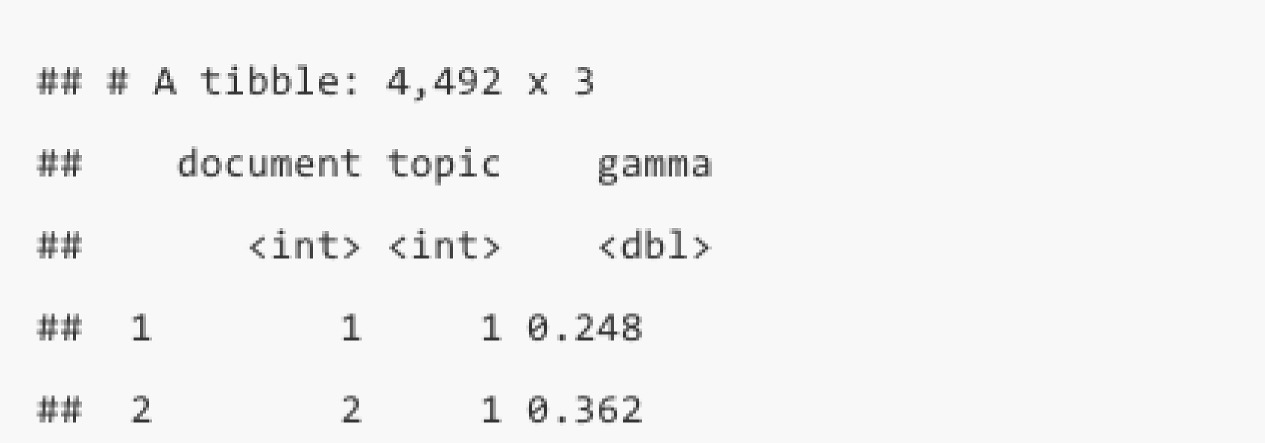


```r
library(tidyr)


beta_spread <- ap_topics %>%
  mutate(topic = paste0("topic", topic)) %>%
  spread(topic, beta) %>%
  filter(topic1 > .001 | topic2 > .001) %>%
  mutate(log_ratio = log2(topic2 / topic1))


beta_spread
```


term
democratic
dukakis
republican
gorbachev
trial
vote
campaign
senate
presidential
prison
rates
fell
rate
prices
dollar
stock
average
cents
index
yen
-100
-50
0
50
Log2 ratio of beta in topic 2 / topic 1

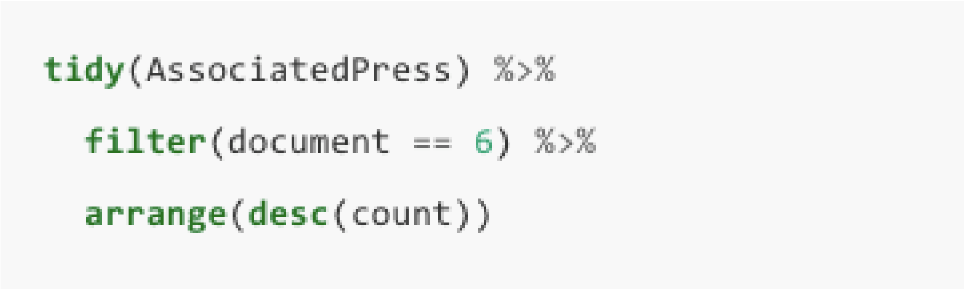


```r
ap_documents <- tidy(ap_lda, matrix = "gamma")
ap_documents
```

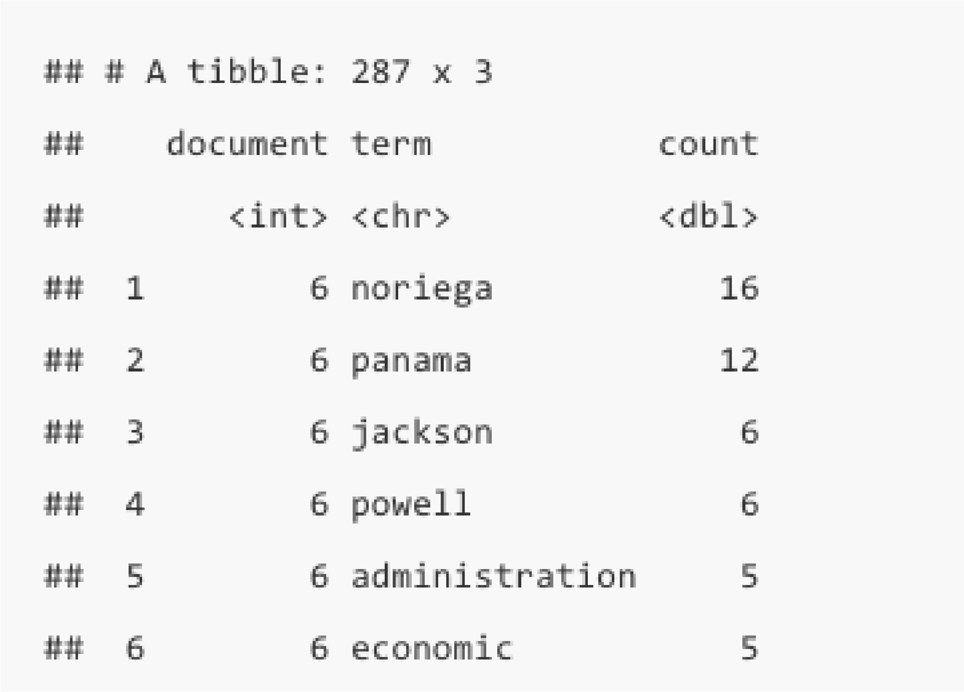


```
## # A tibble: 4,492 x 3
##    document topic     gamma
##       <int> <int>     <dbl>
##  1        1     1 0.248
##  2        2     1 0.362
```

document-topic
probabilities



```
tidy(AssociatedPress) %>%
  filter(document == 6) %>%
  arrange(desc(count))
```



```
## # A tibble: 287 x 3
##    document term           count
##       <int> <chr>          <dbl>
##   1        6 noriega           16
##   2        6 panama            12
##   3        6 jackson            6
##   4        6 powell             6
##   5        6 administration     5
##   6        6 economic           5
```

list of words for a specific document

```
### Text Analysis I: LDA
library(tidytext)
library(topicmodels)
library(tidyr)
library(ggplot2)
library(dplyr)

data("AssociatedPress")

ap_lda<-LDA(AssociatedPress,k=2,control=list(seed=1234)) #create two-topic LDA model
ap_topics<-tidy(ap_lda,matrix="beta") #Extract the per-topic-per-word-probabilities

#Find terms that are most common within each topics
ap_top_terms <- ap_topics %>% group_by(topic) %>% top_n(10,beta) %>% ungroup () %>% arrange (topic, -beta)
ap_top_terms%>% mutate(term=reorder(term,beta))%>%
ggplot(aes(term,beta,fill=factor(topic)))+geom_col(show.legend=FALSE)+
  facet_wrap(~topic,scales="free")+coord_flip() #visualize the above

beta_spread <- ap_topics %>% mutate (topic=paste0("topic",topic)) %>% spread(topic,beta) %>%
  filter (topic1>0.001 | topic2 > 0.001) %>% mutate(log_ratio = log2(topic2/topic1))
beta_spread%>% mutate(term=reorder(term,log_ratio))%>%
ggplot(aes(term,log_ratio))+geom_col(show.legend=FALSE)+coord_flip()

ap_documents<-tidy(ap_lda,matrix="gamma") #Extract the per-document-per-topic-probabilities
ap_documents
tidy(AssociatedPress)%>%filter(document==6)%>%arrange(desc(count)) #Check the most common words in the document, eg
document 6
```

Exercise

- Perform the LDA analysis to your own choice of data. Interpret the results.