Topic modeling

LSI, LDA

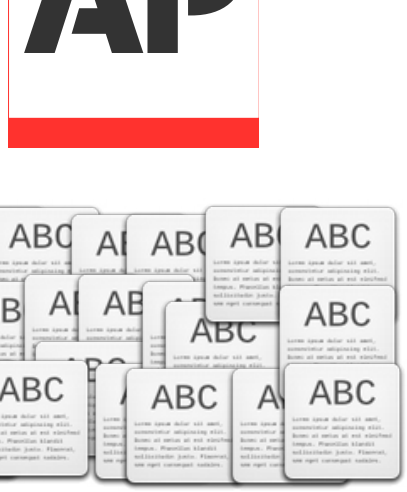
M Loecher

Intro I

In text mining, we often have collections of documents, such as blog posts or news articles, that we'd like to divide into natural groups so that we can understand them separately. Topic modeling is a method for unsupervised classification of such documents, similar to clustering on numeric data, which finds natural groups of items even when we're not sure what we're looking for.

Latent Dirichlet allocation (LDA) is a particularly popular method for fitting a topic model. It treats each document as a mixture of topics, and each topic as a mixture of words. This allows documents to "overlap" each other in terms of content, rather than being separated into discrete groups, in a way that mirrors typical use of natural language.

Intro II



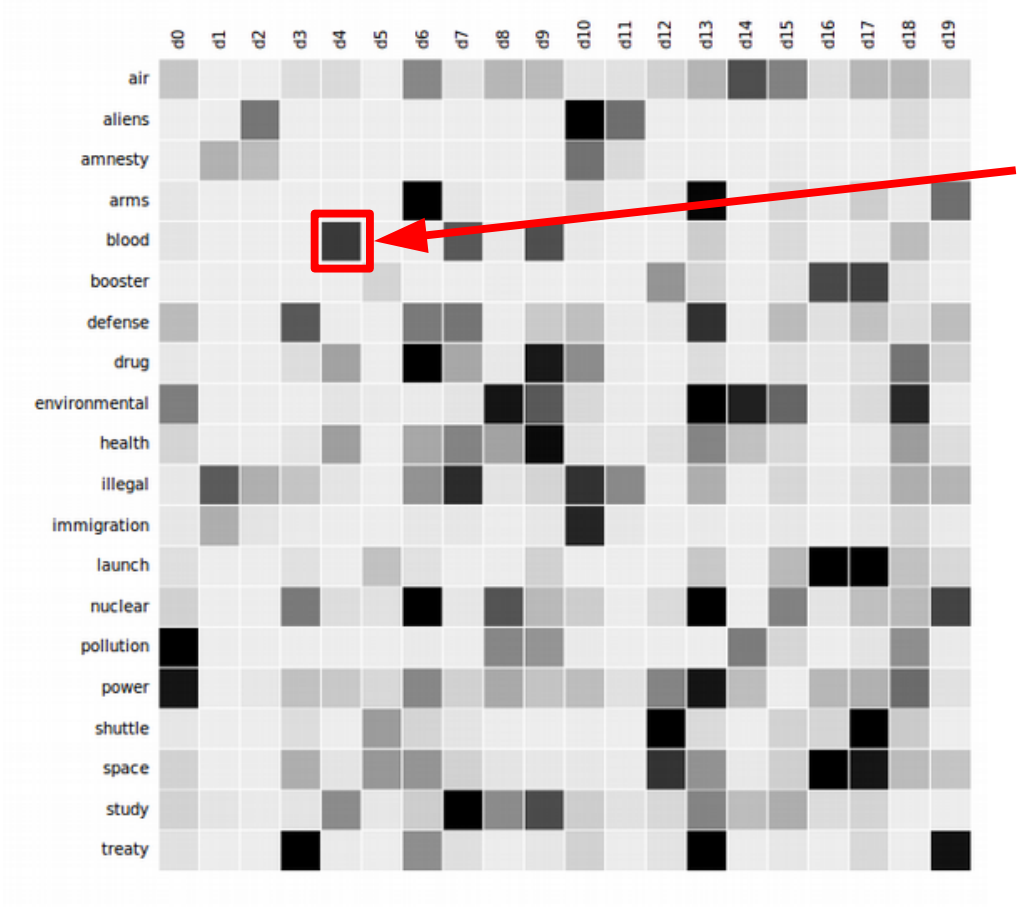
Articles are labelled with tags
(e.g. *politics, economy, sports, ...)*

Politics: *election, party, vote, candidate, ...*
Economy: *dollar, crisis, financial, market, …*
Sports: *soccer, basketball, match, score, ...*

Topics

Latent Semantic Analysis

Term-document matrix



how often does document 4 contain the word “blood”?

high occurrence
low occurrence

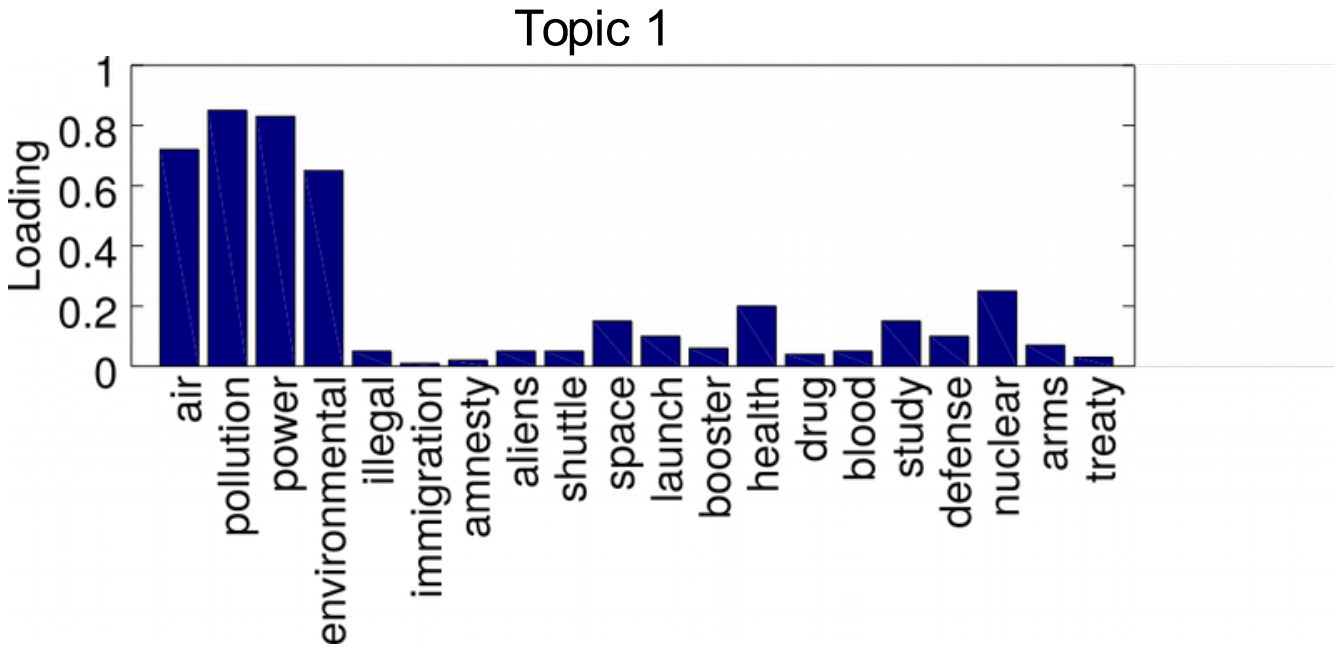
Latent Semantic Analysis (LSA)

- Topic model based on “matrix decomposition”

Latent Semantic Analysis (LSA)

- Topic model based on “matrix decomposition”

- Topics are described by “loadings” over the terms



The Test Dataset

Test dataset

document 0:	probabilistic topic model
document 1:	probabilistic topic model
document 2:	probabilistic topic model
document 3:	probabilistic topic model
document 4:	probabilistic topic model
document 5:	probabilistic topic model
document 6:	probabilistic topic model
document 7:	famous fashion model
document 8:	famous fashion model
document 9:	famous fashion model
document 10:	famous fashion model
document 11:	famous fashion model
document 12:	famous fashion model
document 13:	famous fashion model
document 14:	famous fashion model
document 15:	famous fashion model
document 16:	famous fashion model
document 17:	famous fashion model
document 18:	famous fashion model
document 19:	famous fashion model

Test dataset

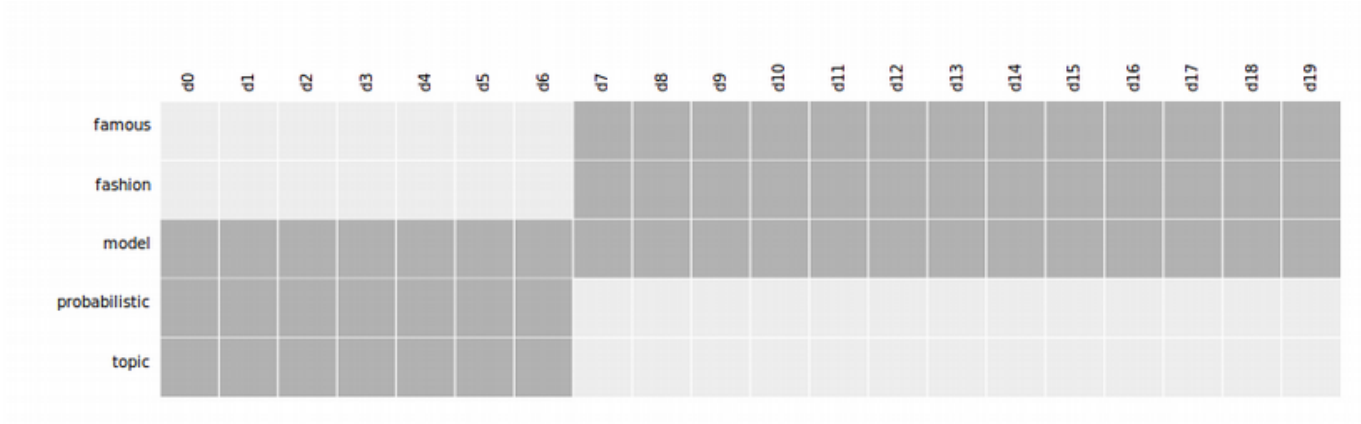
probabilistic topic model
probabilistic topic model
…
famous fashion model
famous fashion model
...

Topic 1: *famous, fashion, model*
Topic 2: *model, probabilistic, topic*

Expected topics

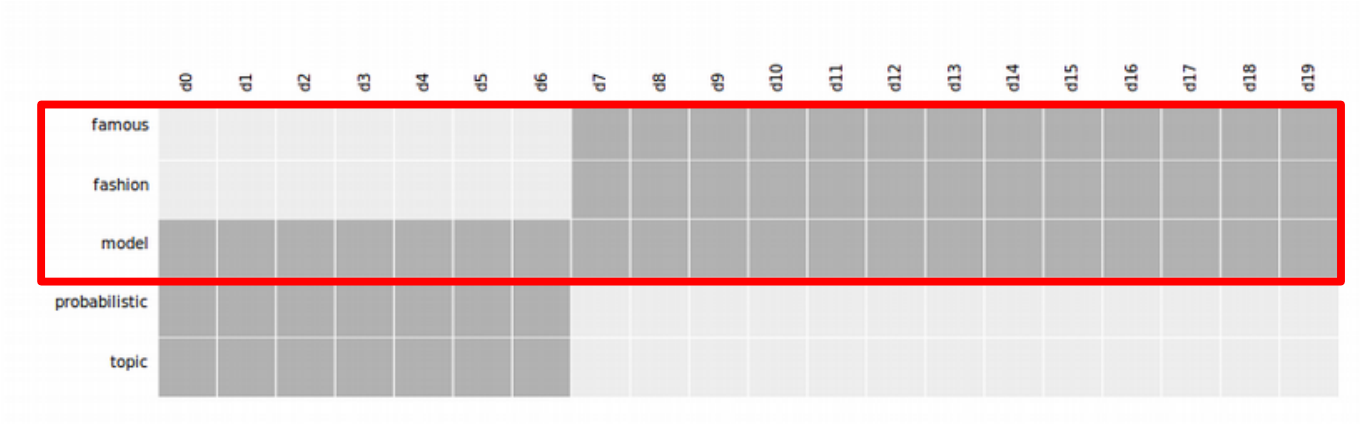
Test dataset

Term-document matrix



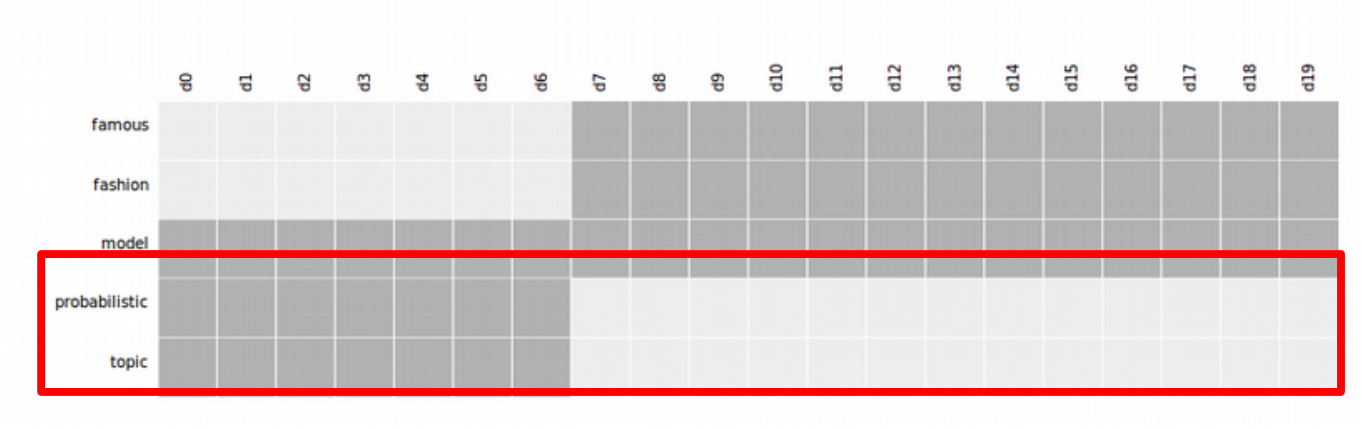
Test dataset

Term-document matrix

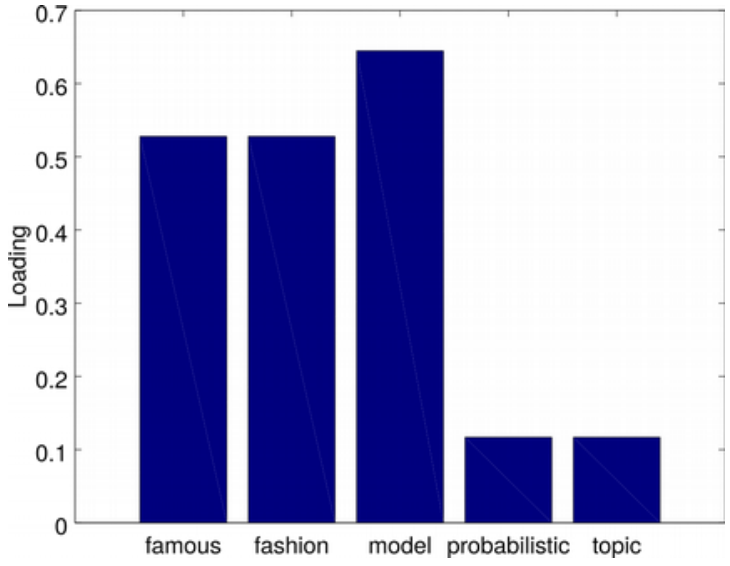


Test dataset

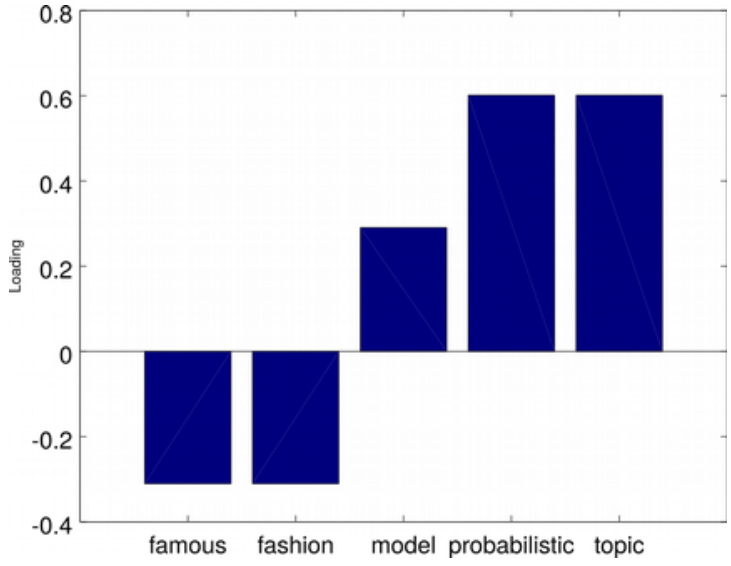
Term-document matrix



LSA



Topic 1



Topic 2

LSA – Weaknesses

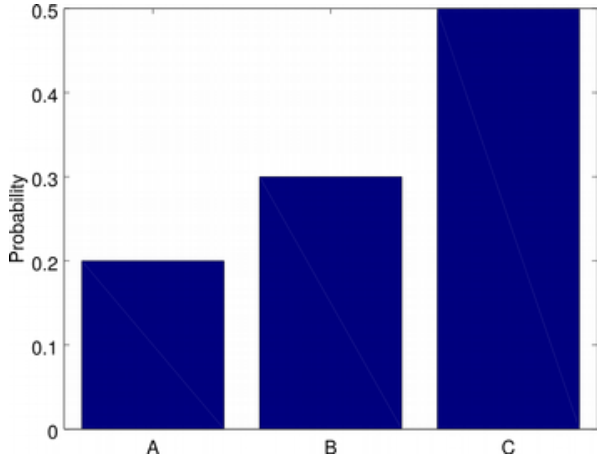
- Topic loadings can be negative → hard to interpret!

- LSA has problems to cope with word ambiguities

Probabilistic LSA

Probabilistic LSA (PLSA)

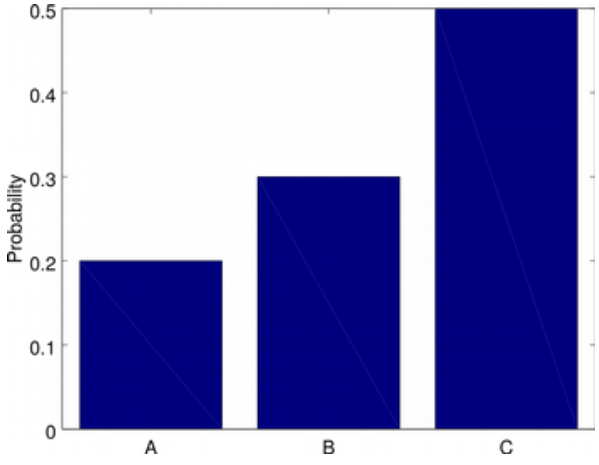
- Based on *categorical* distributions



Probabilistic LSA (PLSA)

- Based on *categorical* distributions

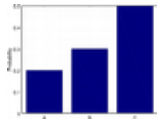
- *Probabilistic model* that explains the creation of documents



Probabilistic LSA (PLSA)

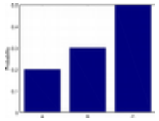
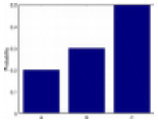
The PLSA model for the creation of words in documents:

1) Documents have each a categorical distribution t over the topics



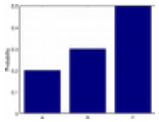
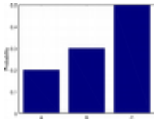
Probabilistic LSA (PLSA)

The PLSA model for the creation of words in documents:

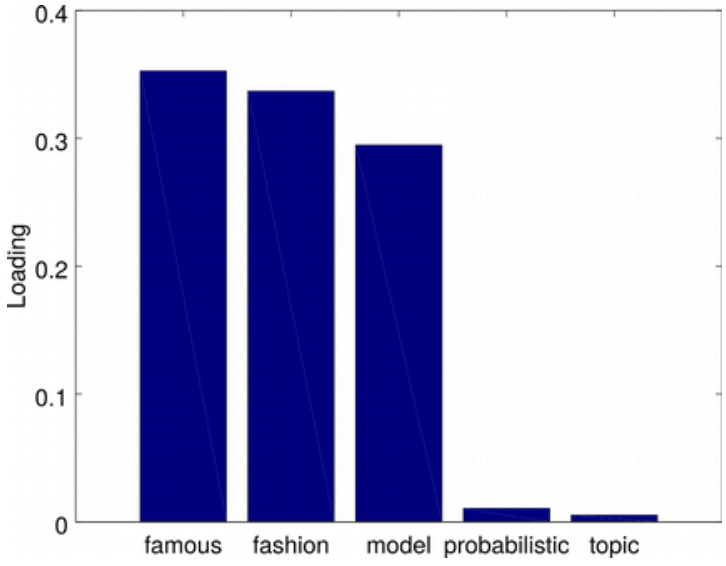
1) Documents have each a categorical distribution t over the topics 
2) Topics have each a categorical distribution f over all words 

Probabilistic LSA (PLSA)

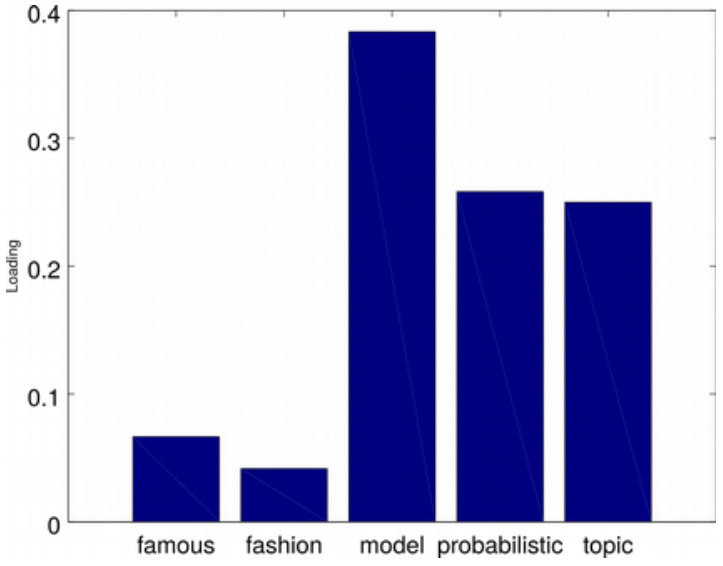
The PLSA model for the creation of words in documents:

1) Documents have each a categorical distribution t over the topics 
2) Topics have each a categorical distribution f over all words 
3) Creation of a word in document i:
    1) Draw a topic z from t_i
    2) Draw a word from f_z

Probabilistic LSA (PLSA)

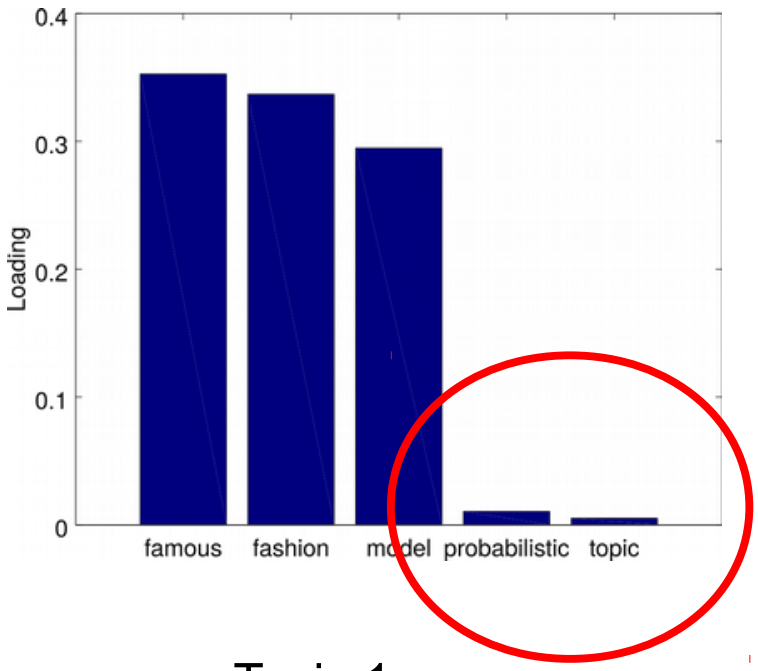


Topic 1

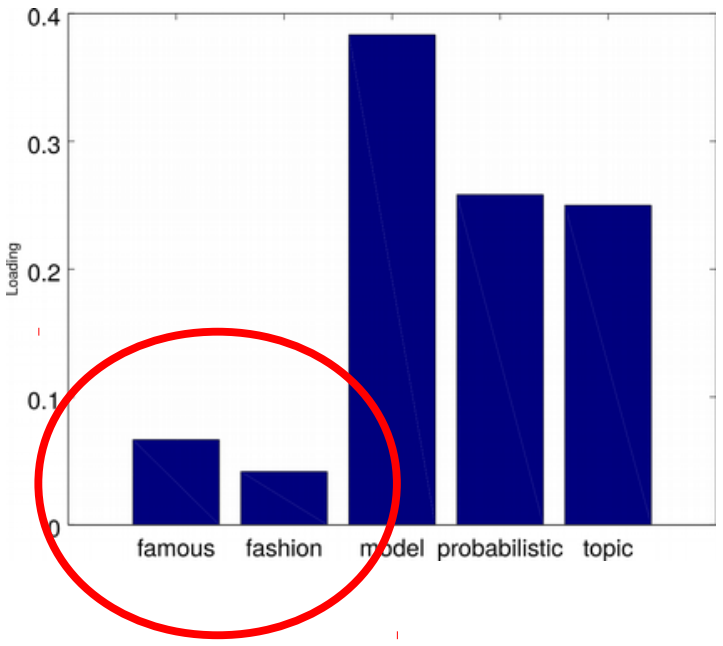


Topic 2

Probabilistic LSA (PLSA)

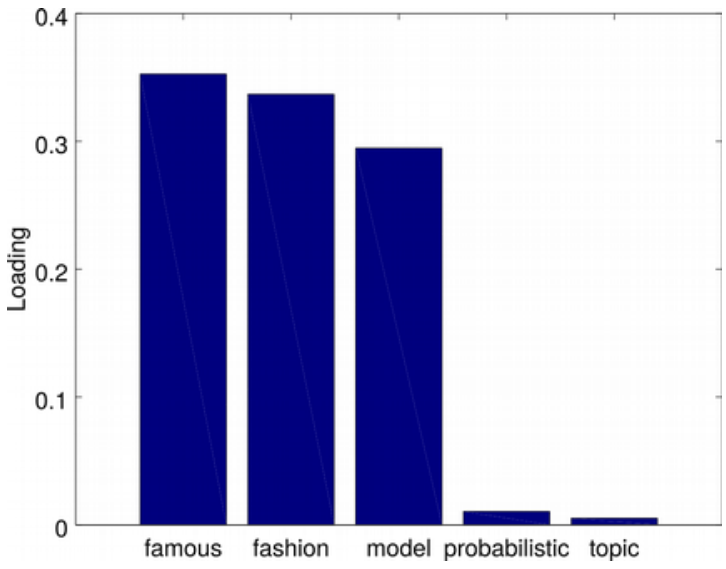


Topic 1

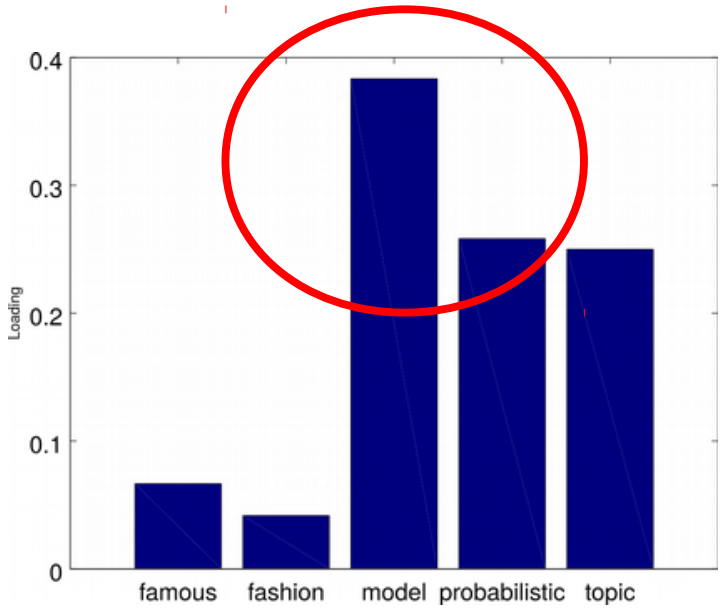


Topic 2

Probabilistic LSA (PLSA)

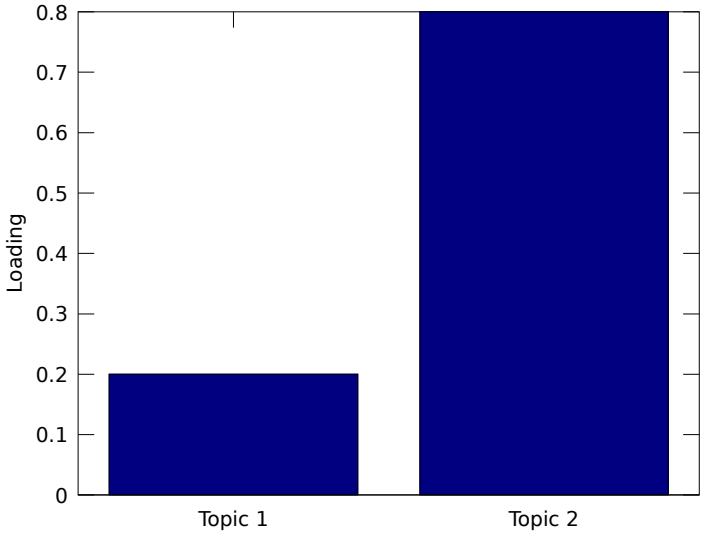


Topic 1



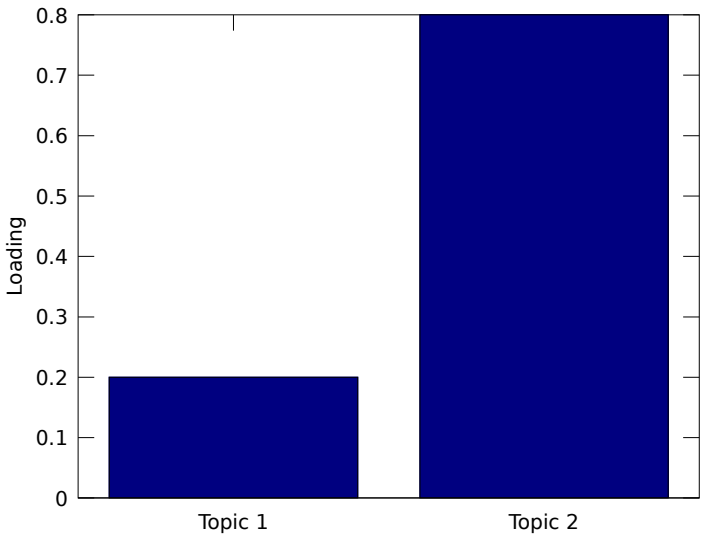
Topic 2

Probabilistic LSA (PLSA)

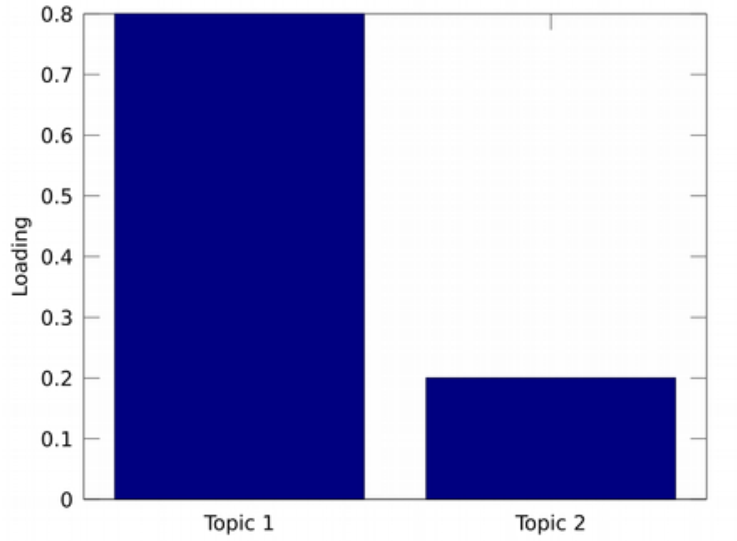


Document 0 (probabilistic topic model)

Probabilistic LSA (PLSA)



Document 0 (probabilistic topic model)



Document 7 (famous fashion model)

PLSA – Strengths & Weaknesses

- Topics are probability distributions and easy to interpret!

- PLSA still has problems to cope with ambiguous words

Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA)

- A word in a document is likely to belong to the same topic as the other words of that document

Latent Dirichlet Allocation (LDA)

- A word in a document is likely to belong to the same topic as the other words of that document

document 7: **famous fashion model**

Latent Dirichlet Allocation (LDA)

- A word in a document is likely to belong to the same topic as the other words of that document

document 7: **famous fashion model**

Topic 1 Topic 1 ?

Latent Dirichlet Allocation (LDA)

- A word in a document is likely to belong to the same topic as the other words of that document

document 7: **famous fashion model**

Topic 1 Topic 1 → **Topic 1**

Latent Dirichlet Allocation (LDA)

- A word in a document is likely to belong to the same topic as the other words of that document

document 0: **probabilistic topic model**

Latent Dirichlet Allocation (LDA)

- A word in a document is likely to belong to the same topic as the other words of that document

document 0: **probabilistic topic model**

Topic 2 Topic 2 ?

Latent Dirichlet Allocation (LDA)

- A word in a document is likely to belong to the same topic as the other words of that document

document 0: **probabilistic topic model**

Topic 2 Topic 2 → **Topic 2**

Latent Dirichlet Allocation (LDA)

- A word in a document is likely to belong to the same topic as the other words of that document

- We would need some preference for already assigned topics in a document

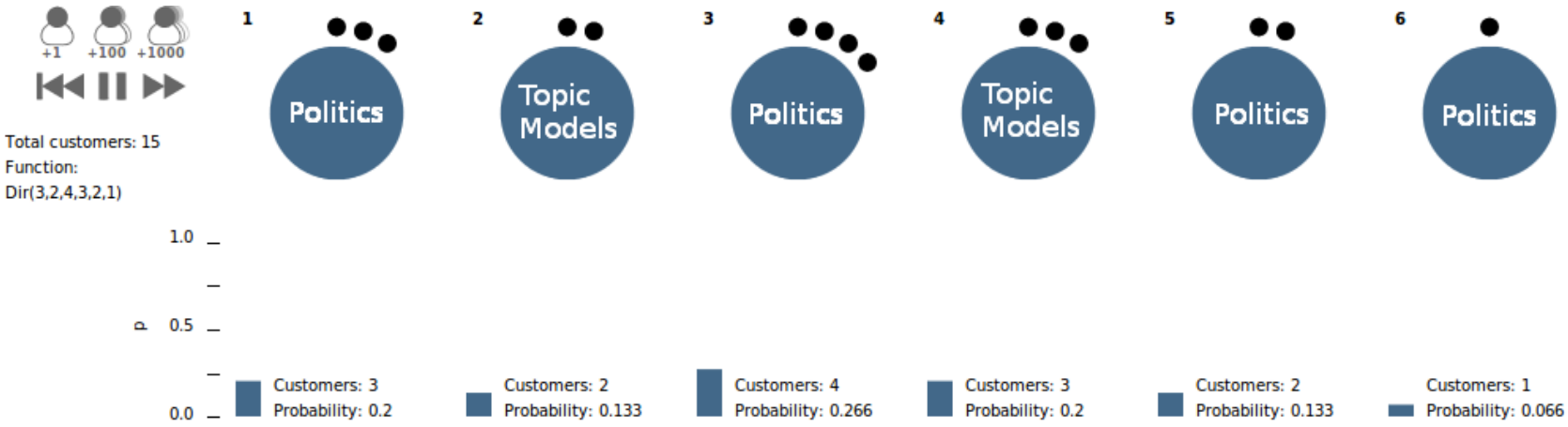
Latent Dirichlet Allocation (LDA)

- A word in a document is likely to belong to the same topic as the other words of that document

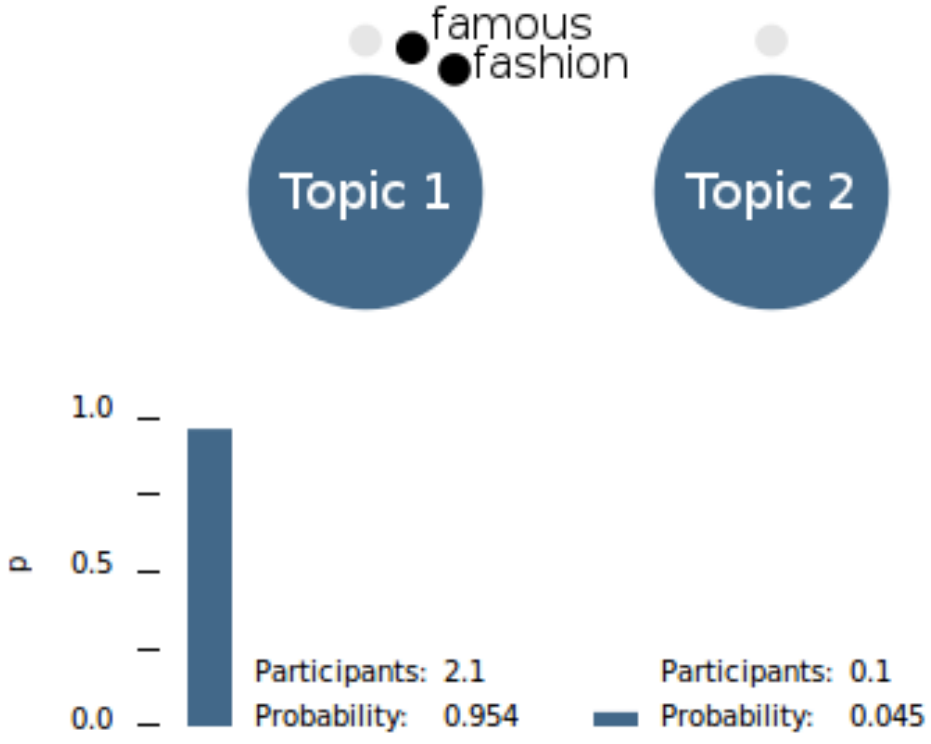
- We would need some preference for already assigned topics in a document

→ Dirichlet distribution!

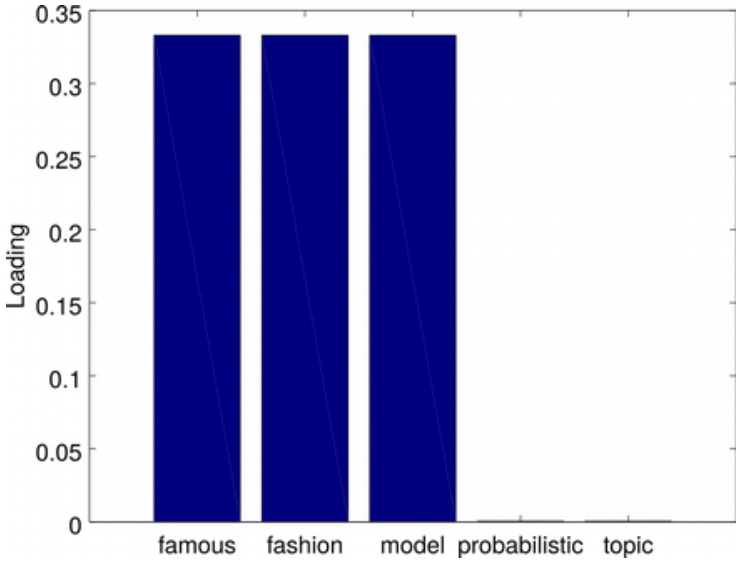
Dirichlet distribution



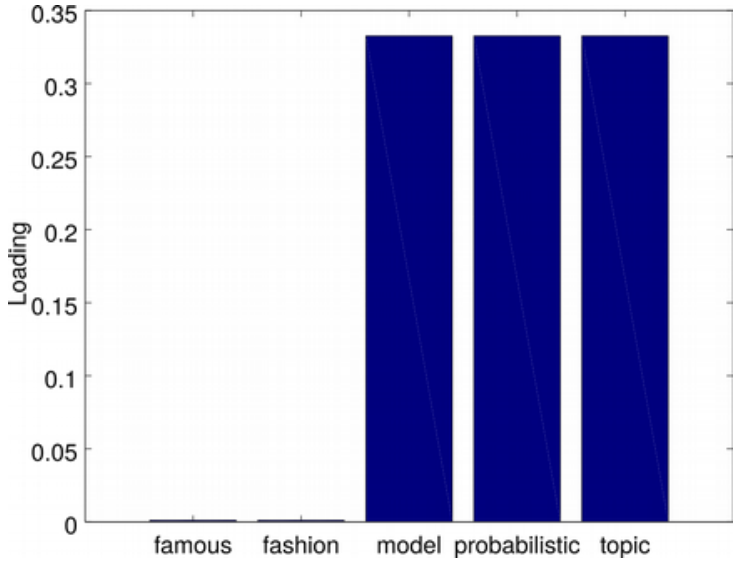
Dirichlet distribution



Latent Dirichlet Allocation (LDA)

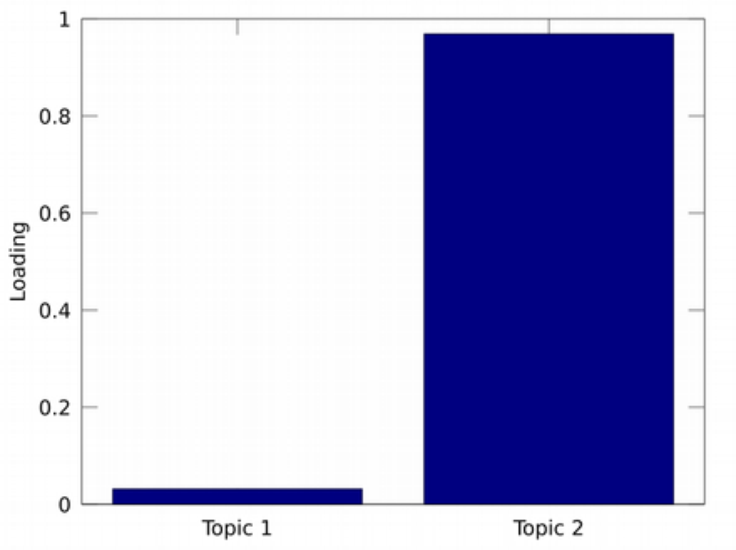


Topic 1

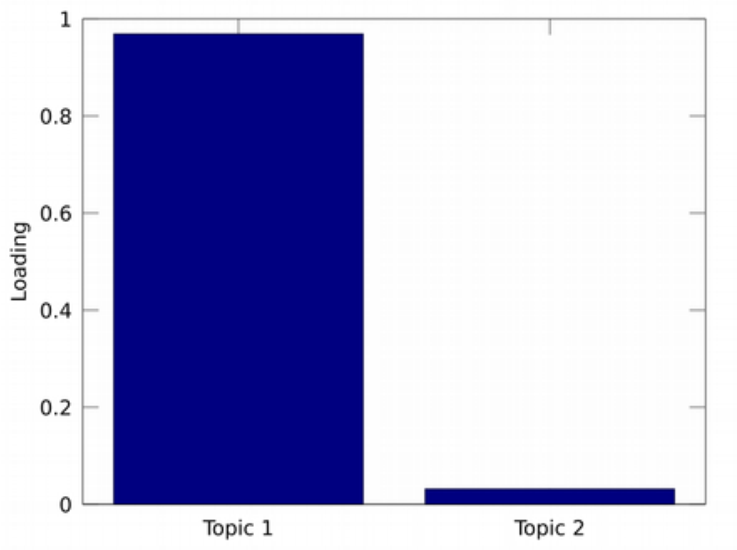


Topic 2

Probabilistic topic model (with sparse Dirichlet)



Document 0 (probabilistic topic model)



Document 7 (famous fashion model)

Intro III

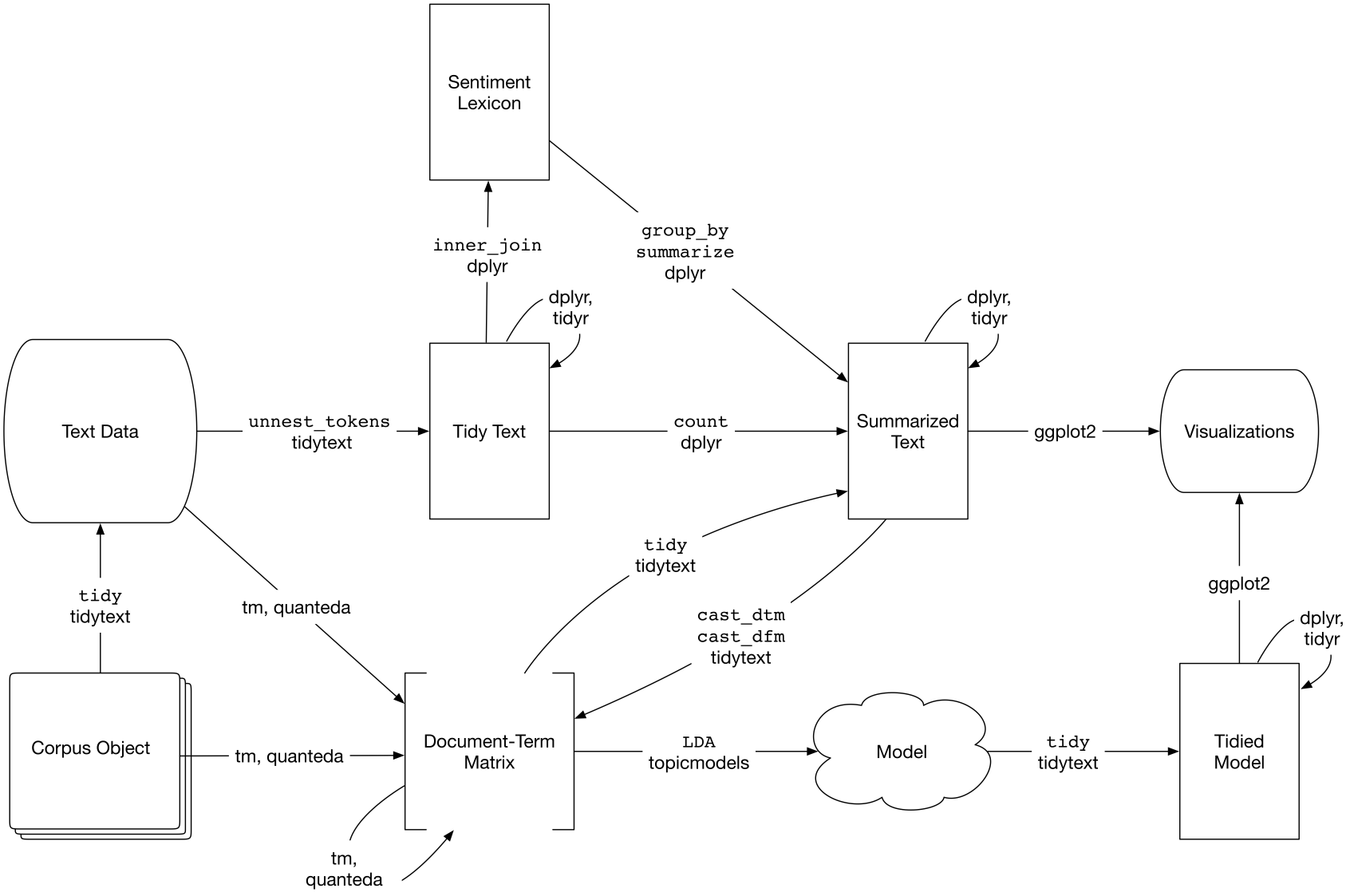


Figure 1: Document-Term Matrix -> topic modeling.

Latent Dirichlet allocation

Latent Dirichlet allocation is one of the most common algorithms for topic modeling. Without diving into the math behind the model, we can understand it as being guided by two principles.

- **Every document is a mixture of topics.** We imagine that each document may contain words from several topics in particular proportions. For example, in a two-topic model we could say "Document 1 is 90% topic A and 10% topic B, while Document 2 is 30% topic A and 70% topic B."
- **Every topic is a mixture of words.** For example, we could imagine a two-topic model of American news, with one topic for "politics" and one for "entertainment." The most common words in the politics topic might be "President", "Congress", and "government", while the entertainment topic may be made up of words such as "movies", "television", and "actor". Importantly, words can be shared between topics; a word like "budget" might appear in both equally.

LDA

LDA is a mathematical method for estimating both of these at the same time: finding the mixture of words that is associated with each topic, while also determining the mixture of topics that describes each document. There are a number of existing implementations of this algorithm, and we'll explore one of them in depth.

The `AssociatedPress` dataset is a collection of 2246 news articles from an American news agency, mostly published around 1988.

We create a two-topic LDA model.

```
# set a seed so that the output of the model is predictabl
ap_lda <- LDA(AssociatedPress, k = 2, control = list(seed =
ap_lda_2 <- LDA(AssociatedPress, k = 2, control = list(alph
ap_lda_3 <- LDA(AssociatedPress, k = 2, control = list(alph
save(ap_lda,ap_lda_2,ap_lda_3, file = "data/ap_lda.rda")
```

Word-topic probabilities

Step 1: extract the per-topic-per-word probabilities, called β ("beta"), from the model.

```
##   topic      term         beta
## 1     1     aaron 1.686917e-12
## 2     2     aaron 3.895941e-05
## 3     1   abandon 2.654910e-05
## 4     2   abandon 3.990786e-05
## 5     1 abandoned 1.390663e-04
## 6     2 abandoned 5.876946e-05
```

For each combination topic-term, the model computes the probability of that term being generated from that topic. For example, the term "aaron" has a 1.686917×10^{-12} probability of being generated from topic 1, but a 3.8959408×10^{-5} probability of being generated from topic 2.

Top 10 Word-topic probabilities

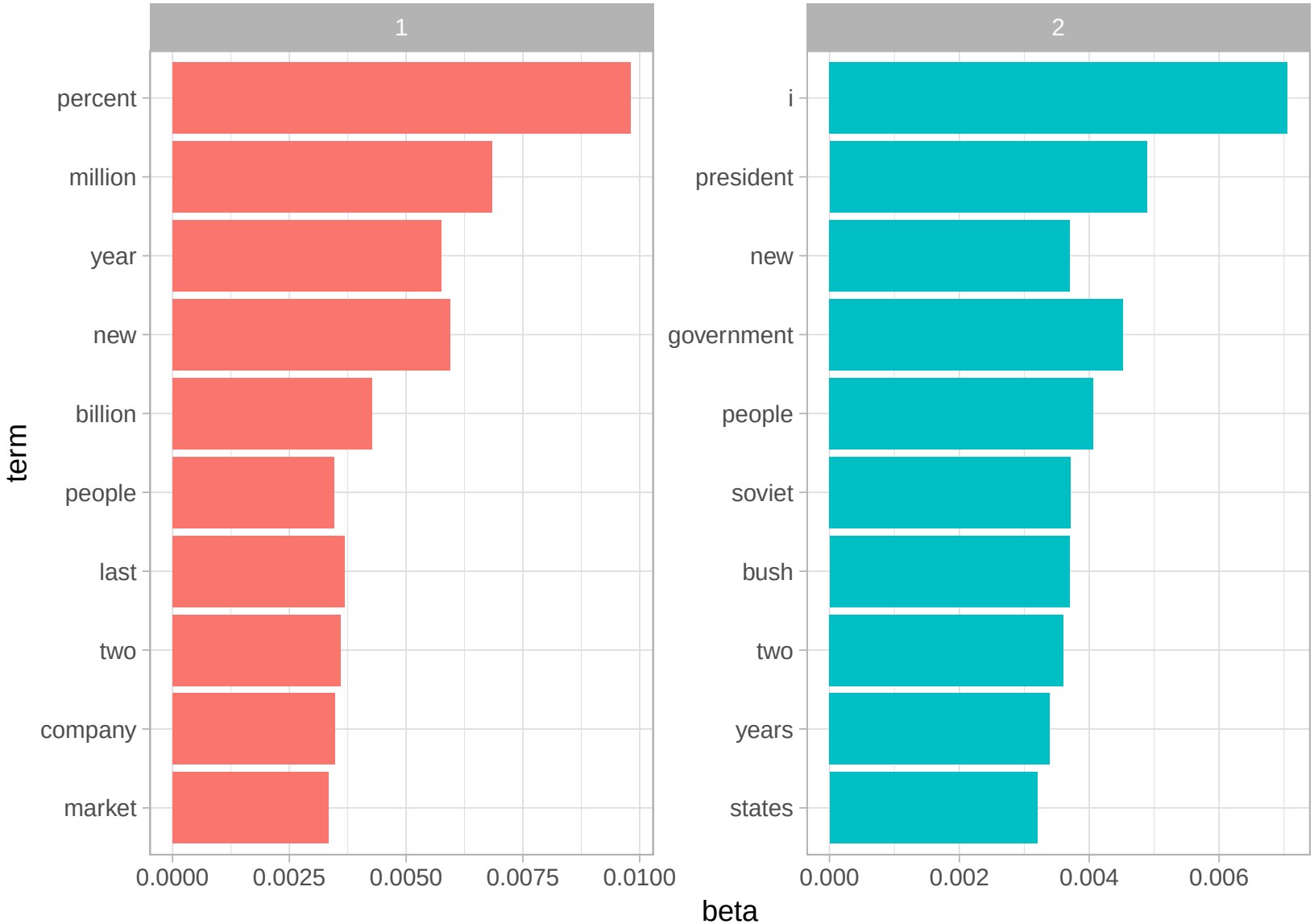


Figure 2: The terms that are most common within each topic

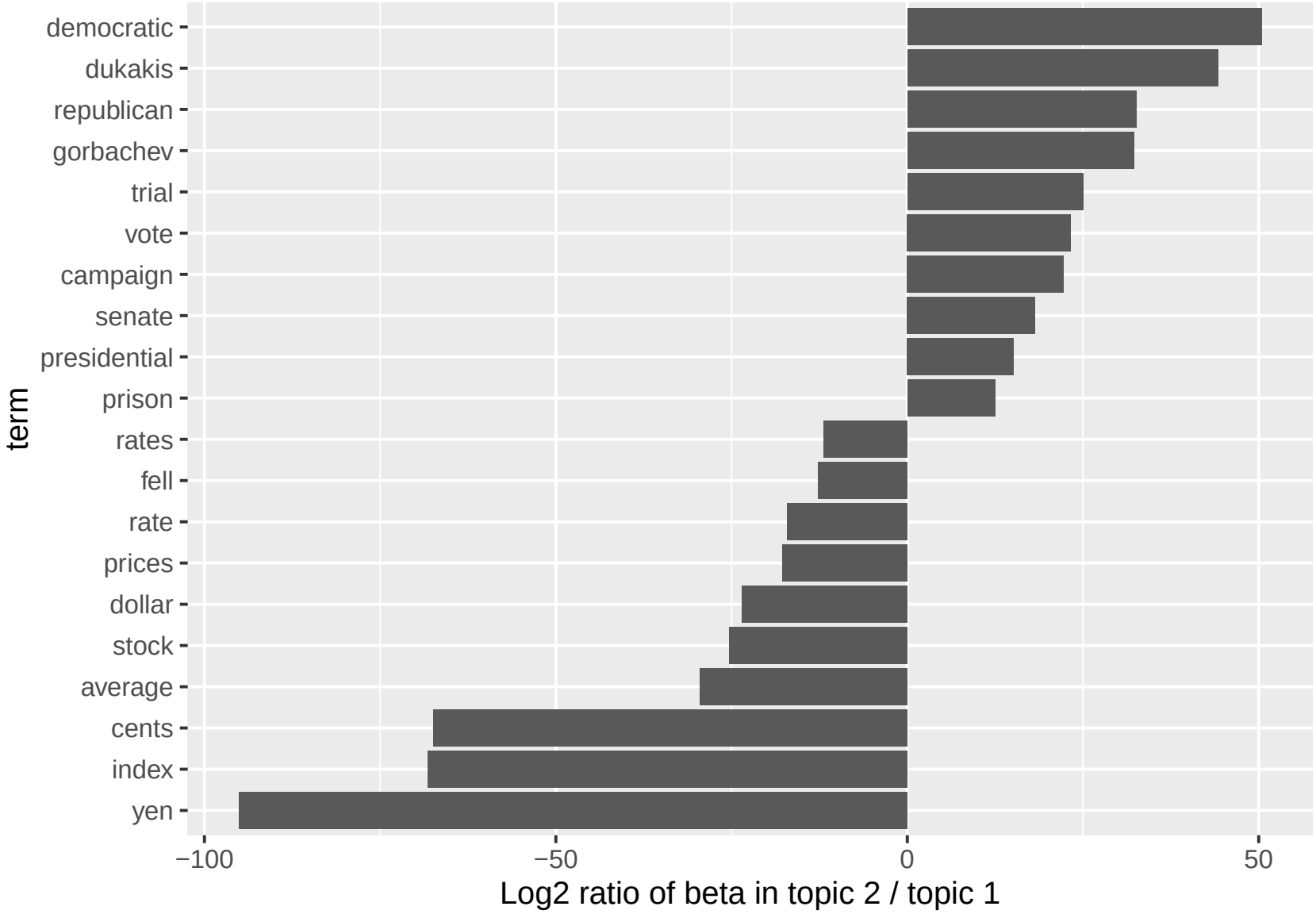
Top 10 Word-topic probabilities

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”, suggeting 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.

Greatest differences

As an alternative, we could consider 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: $\log_2(\frac{\beta_2}{\beta_1})$ (a log ratio is useful because it makes the difference symmetrical: β_2 being twice as large leads to a log ratio of 1, while β_1 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.

Words with the greatest difference in β



Difference

We can see that the words more common in topic 2 include political parties such as “democratic” and “republican”, as well as politician’s names such as “dukakis” and “gorbachev”. Topic 1 was more characterized by currencies like “yen” and “dollar”, as well as financial terms such as “index”, “prices” and “rates”. This helps confirm that the two topics the algorithm identified were political and financial news.

Document-topic probabilities

Besides estimating each topic as a mixture of words, LDA also models each document as a mixture of topics. We can examine the per-document-per-topic probabilities, called γ ("gamma").

```
##   document topic        gamma
## 1        1     1 0.2480616686
## 2        2     1 0.3615485445
## 3        3     1 0.5265844180
## 4        4     1 0.3566530023
## 5        5     1 0.1812766762
## 6        6     1 0.0005883388
```

Each of these values is an estimated proportion of words from that document that are generated from that topic. For example, the model estimates that only about 24.8% of the words in document 1 were generated from topic 1.

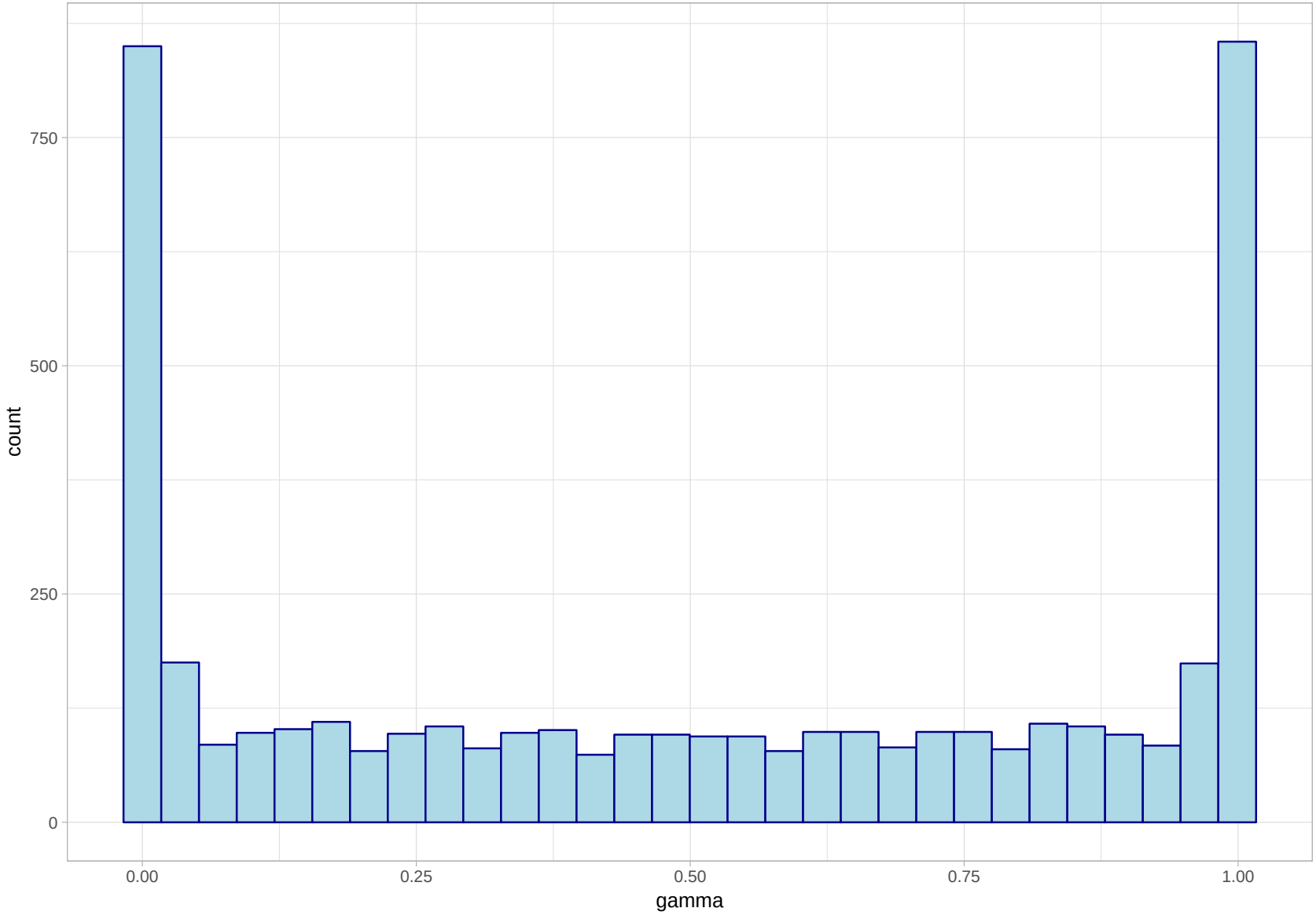
Document-topic probabilities

We can see that many of these documents were drawn from a mix of the two topics, but that document 6 was drawn almost entirely from topic 2, having a γ from topic 1 close to zero. Here are the most common words in the document-term matrix:

```
##   document           term count
## 1        6        noriega    16
## 2        6         panama    12
## 3        6        jackson     6
## 4        6         powell     6
## 5        6 administration     5
## 6        6       economic     5
```

Based on the most common words, this appears to be an article about the relationship between the American government and Panamanian dictator Manuel Noriega, which means the algorithm was right to place it in topic 2 (as political/national news).

Distribution of γ



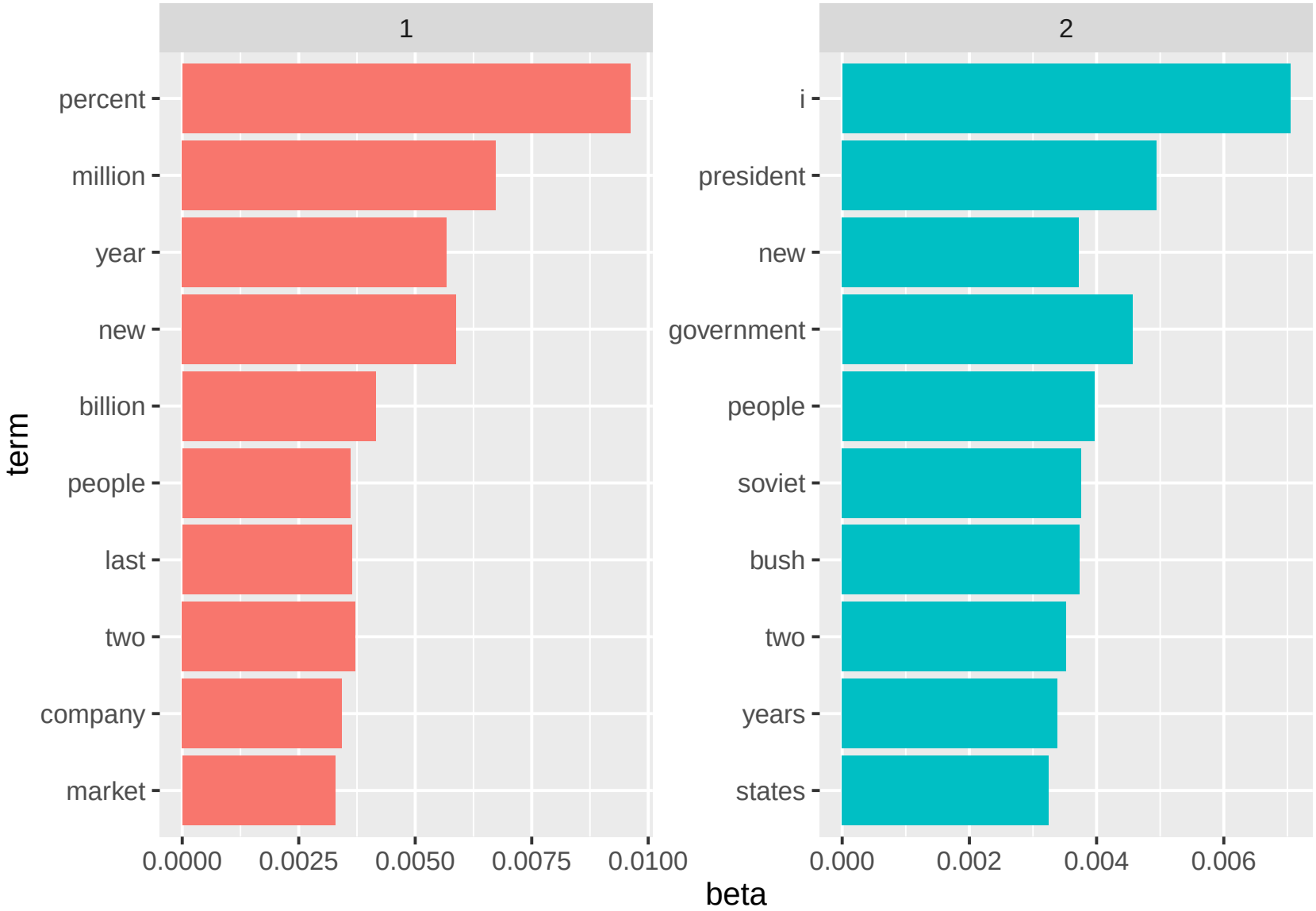
Priors

alpha and beta hyperparameters

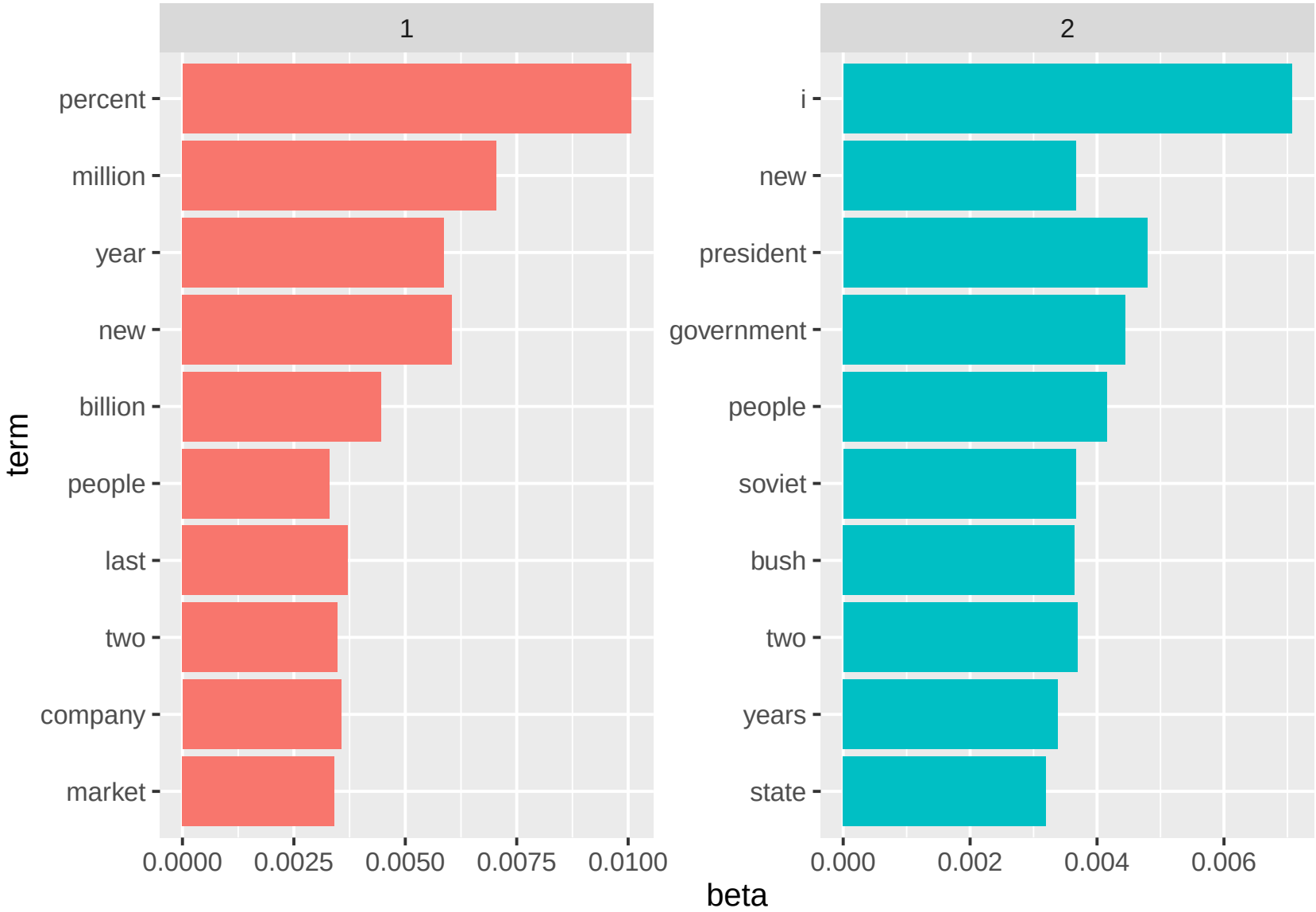
Assuming symmetric Dirichlet distributions (for simplicity), a low alpha value places more weight on having each document composed of only a few dominant topics (whereas a high value will return many more relatively dominant topics). Similarly, a low beta value places more weight on having each topic composed of only a few dominant words.

But the effects are pretty small in practice: "the data overwhelm the prior":

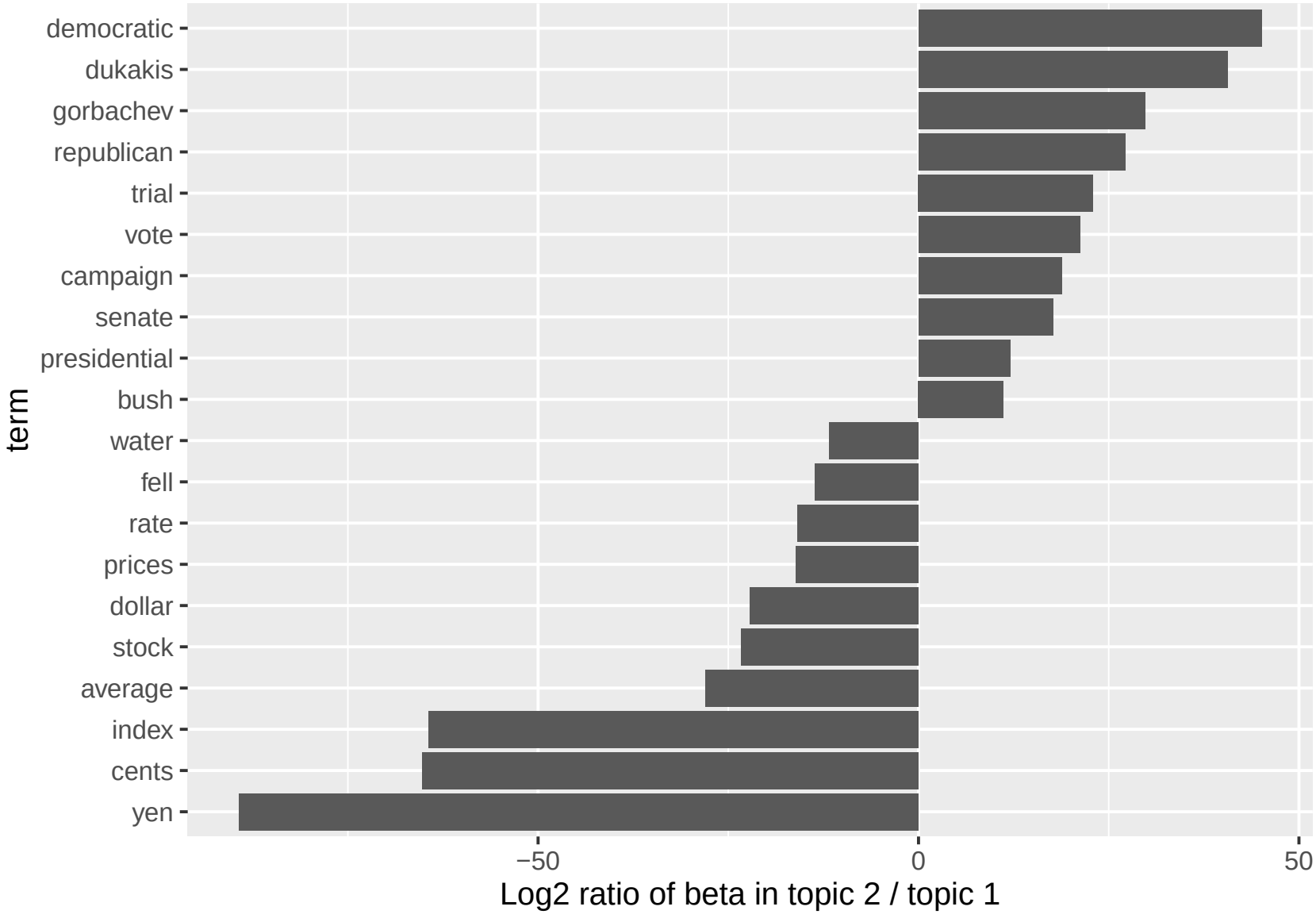
$\alpha = 0.05$


1
2
percent
million
year
new
billion
people
last
two
company
market
i
president
new
government
people
soviet
bush
two
years
states
term
0.0000
0.0025
0.0050
0.0075
0.0100
0.000
0.002
0.004
0.006
beta

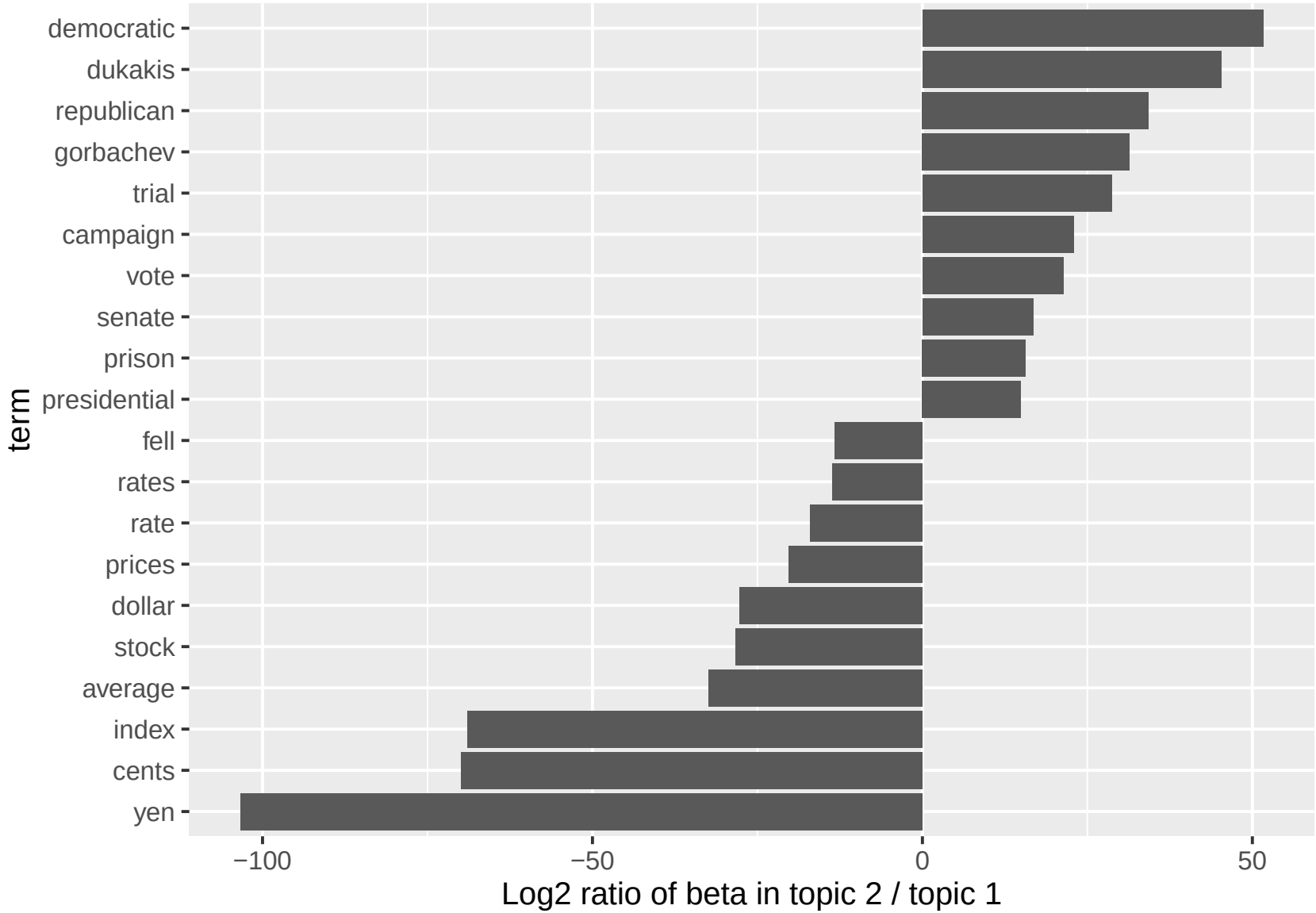
$\alpha = 5$


1
2
percent
million
year
new
billion
people
last
two
company
market
i
new
president
government
people
soviet
bush
two
years
state
term
0.0000
0.0025
0.0050
0.0075
0.0100
0.000
0.002
0.004
0.006
beta

$\alpha = 0.05$

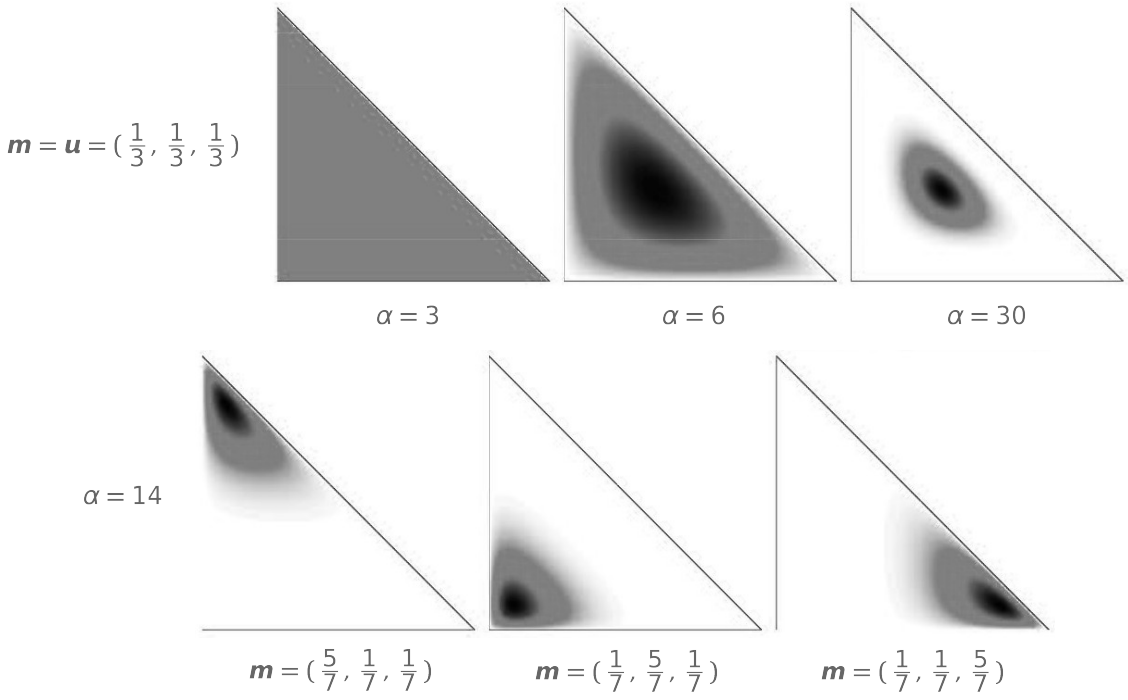

democratic
dukakis
gorbachev
republican
trial
vote
campaign
senate
presidential
bush
water
fell
rate
prices
dollar
stock
average
index
cents
yen
term
−50
0
50
Log2 ratio of beta in topic 2 / topic 1

$\alpha = 5$


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

Dirichlet

Dirichlet Parameters



Validation

Perplexity

Topic Coherence

Summary

This chapter introduces topic modeling for finding clusters of words that characterize a set of documents, and shows how the `tidy()` verb lets us explore and understand these models using dplyr and ggplot2. This is one of the advantages of the tidy approach to model exploration: the challenges of different output formats are handled by the tidying functions, and we can explore model results using a standard set of tools. In particular, we saw that topic modeling is able to separate and distinguish chapters from four separate books, and explored the limitations of the model by finding words and chapters that it assigned incorrectly.

Great tutorial and toolbox: http://topicmodels.west.uni-koblenz.de/