

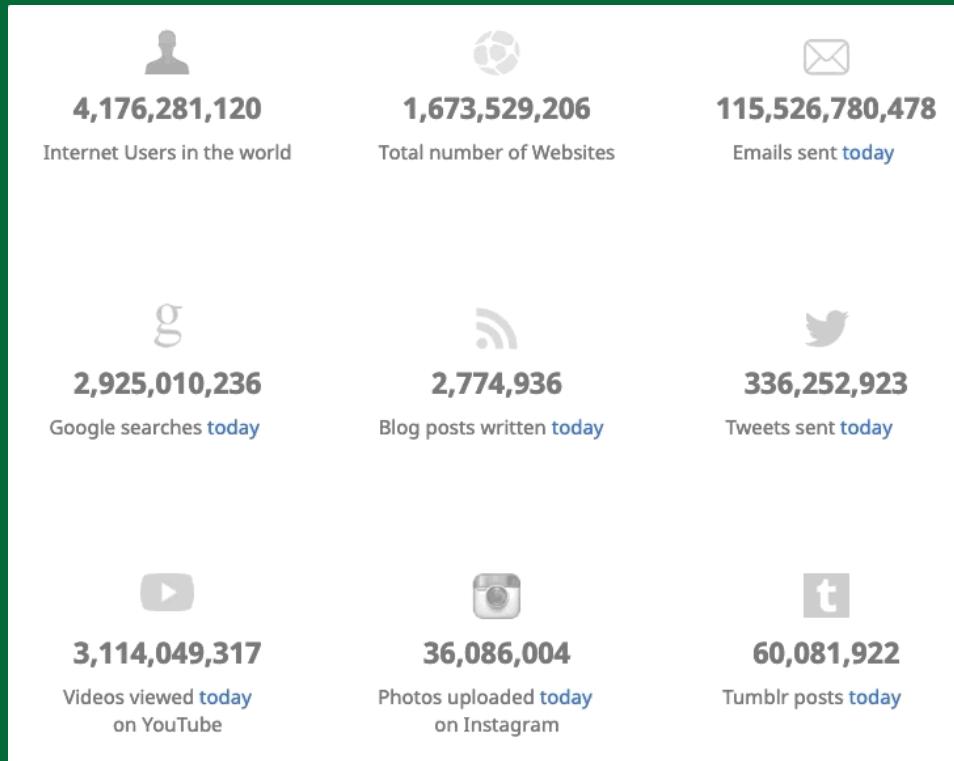
DSBA 5122: Visual Analytics

Class 8: Text-as-Data

Ryan Wesslen

March 18, 2019

Introduction to Text-as-Data



High dimensionality of text

 Thoughts of Dog 
@dog_feelings

Follow ▾

gooooob morning. i have a feeling. today will
be a good day. and if it's not. well that's
alright too. because there's always tomorrow.
and there's always peanut butter

11:55 AM - 11 Apr 2018

19,767 Retweets 82,188 Likes

302 20K 82K

High dimensionality of text

 **Thoughts of Dog** 
@dog_feelings

[Follow](#) 

11:55 AM - 11 Apr 2018

19,767 Retweets 82,188 Likes

 302  20K  82K 

High dimensionality of text

Thoughts of Dog

@dog_feelings

Follow

1,000

11:55 AM - 11 Apr 2018

19,767 Retweets 82,188 Likes

302 20K 82K

High dimensionality of text

Thoughts of Dog

@dog_feelings

Follow

1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 *
1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 *
1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 * 1,000 *

11:55 AM - 11 Apr 2018

19,767 Retweets 82,188 Likes

302 20K 82K

of Combinations = $1000^{30} \approx \# \text{ of Atoms in the Universe}$

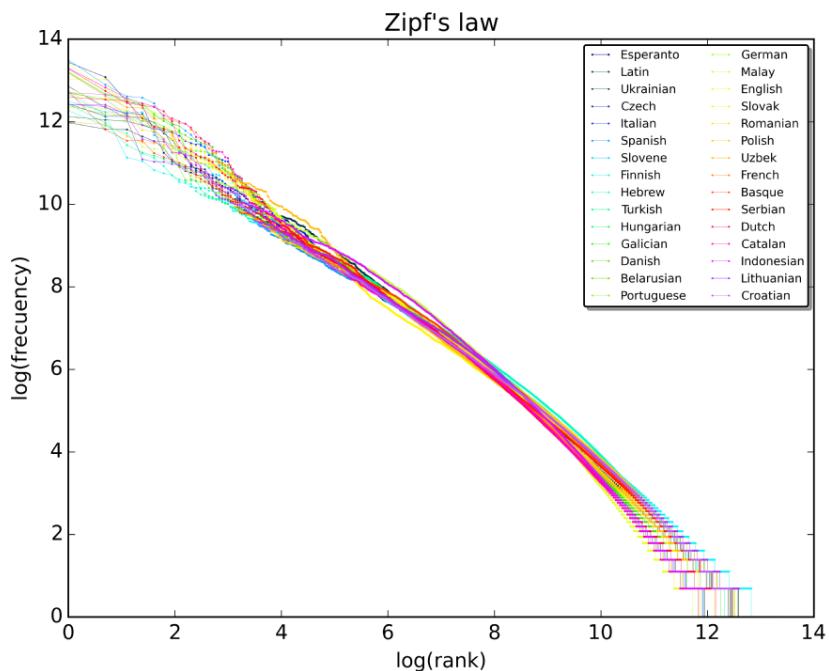
"A sample of 30-word Twitter messages that use only the 1,000 most common words in the English language ... has roughly as many dimensions as there are atoms in the universe." "Text as Data", 2017

Weird world of Text:

😂 2353998352	❤️ 1130101293	♻️ 946294559	😊 863581738	❤️ 706982856
🤣 684761013	😊 580256321	😴 480926559	💕 456126097	🥳 441438735
😢 399042774	😉 369017665	👌 342269528	😔 334036059	😡 325737726
☺️ 315443303	😉 273974507	👍 236570857	👉 228023789	😎 218485649
🙏 214844562	😉 214763612	😴 212806925	👀 209752718	❤️ 206977286
😎 205533735	♫ 203881512	💙 186956891	🎶 182196299	💜 181076438
😳 178890269	✨ 175721468	_DIP 168824899	💕 166759773	💯 164714238
🐰 158607215	🔥 158594713	😴 153211676	😊 152999718	😑 151850852
😊 140045964	😅 137955669	😴 135804137	😔 134732045	😢 132704679
💖 126375434	👋 124635518	😐 124298019	👉 121705956	💖 116298949
🌹 113376649	📸 111255484	😱 106959799	девушк 103674275	💛 101213111
💀 101113423	💋 100809913	🌸 100766261	👽 96714588	😡 95234214
😊 94049412	😈 92699191	😊 91813197	😊 91235208	🎉 90418407
😊 88990659	✋ 88879775	💪 88487122	😢 85461362	▶ 85374746
😊 81379853	🌿 80145246	🤔 79985376	💕 78995313	🌐 78819006

Key Properties about Text

Zipf's Law



VSauce video on Zipf's Law

Bag of Words

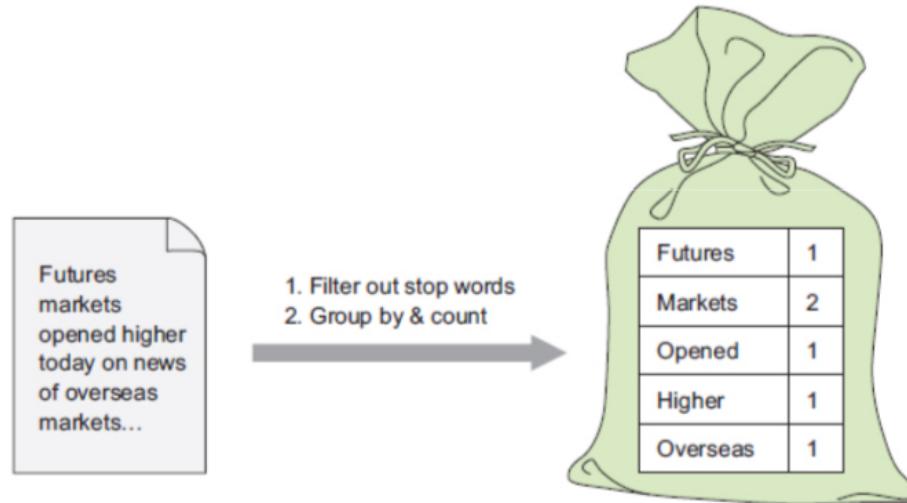
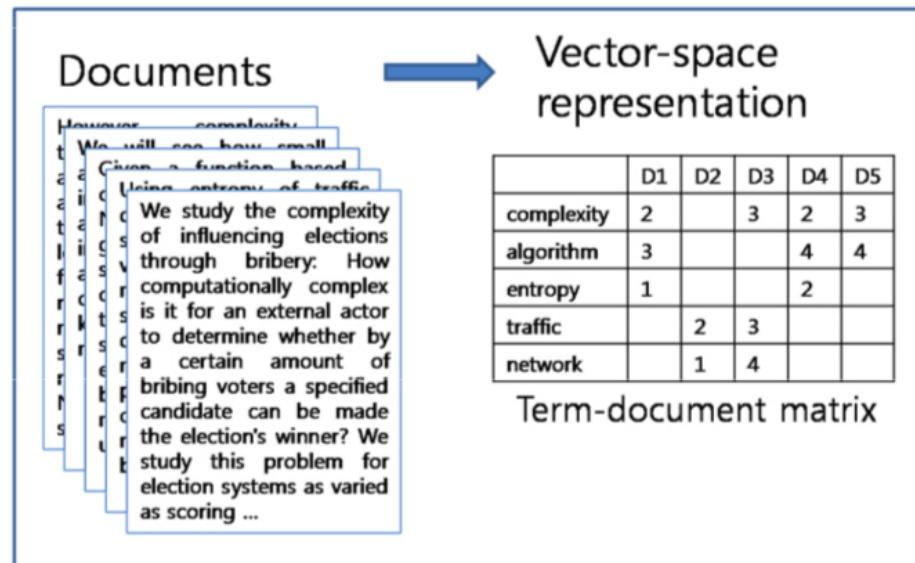


Figure 7.6 Bag of words representation of a document.

Document Term Matrix



Corpus

```
library(quanteda)
tweets <- read_csv("../data/CharlotteTweets20Sample.csv")

# create corpus
twcorpus <- corpus(tweets$body)

summary(twcorpus, n = 10)

## Corpus consisting of 47274 documents, showing 10 documents:
##
##   Text Types Tokens Sentences
##   text1    22     23      2
##   text2    14     16      1
##   text3     2      7      1
##   text4    20     34      1
##   text5     9      9      1
##   text6    25     27      2
##   text7     8      8      1
##   text8     3      3      1
##   text9    18     18      1
##   text10    6      6      1
##
## Source: /Users/rhymenoceros/Dropbox (UNC Charlotte)/dsba5122-spring2019/static/slides/* on x86_64 by rhymenoceros
## Created: Sun Mar 17 21:26:40 2019
## Notes:
```

Document-Feature Matrix

```
# additional stop words to remove, unique to Twitter
stopWords <- c("t.co", "https", "rt", "amp", "http", "gt", "f", "u", "w")
# add profile ID
docvars(twcorpus, "actor.id") <- as.character(tweets$actor.id)

twdfm <- dfm(twcorpus,
               remove = c(stopwords("english"), stopWords),
               remove_punct = TRUE,
               remove_numbers = TRUE,
               remove_symbols = TRUE,
               remove_url = TRUE,
               ngrams= 1L)
# remove sparse terms
twdfm <- dfm_trim(twdfm, min_docfreq = 3)

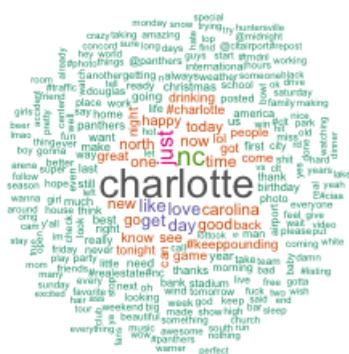
head(twdfm)

## Document-feature matrix of: 6 documents, 11,767 features (100.0% sparse).
```

Word Clouds

Raw Weighting

```
library(RColorBrewer)  
  
textplot_wordcloud(twdfm,  
  min_size = .75,  
  max_size = 3.5,  
  color = brewer.pal(8, "Dark2"),  
  random_order = F,  
  rotation = 0.1,  
  max_words = 250)
```



TF-IDF Weighting

```
textplot_wordcloud(dfm_tfidf(twdfm),  
  min_size = .75,  
  max_size = 3.5,  
  color = brewer.pal(8, "Dark2"),  
  random_order = F,  
  rotation = 0.1,  
  max_words = 250)
```



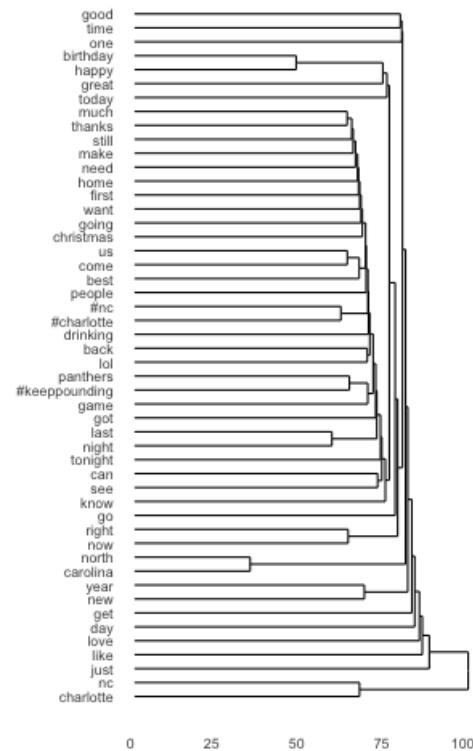
<https://quanteda.io/articles/pkgdown/examples/plotting.html>

Clustering

```
numWords <- 50

wordDfm <- twdfm %>%
  dfm_tfidf() %>%
  dfm_sort()

wordDfm <- t(wordDfm)[1:numWords,] # keep the top numWords words
wordDistMat <- dist(wordDfm)
wordCluster <- hclust(wordDistMat)
# create dendrogram
ggdendro::ggdendrogram(wordCluster, rotate = TRUE)
```



```

tag_dfm <- dfm_select(twdfm, pattern = ("#*"))
toptag <- names(topfeatures(tag_dfm, 50))
head(toptag)

```

```

## [1] "#keepbounding" "#charlotte"      "#nc"          "#realestate"
## [5] "#clt"           "#traffic"

```

```

tag_fcm <- fcm(tag_dfm)
head(tag_fcm)

```

```

## Feature co-occurrence matrix of: 6 by 6 features.
## 6 x 6 sparse Matrix of class "fcm"
##             features
## features   #vscoco #vscocam #primus #charlotte #livemusic #lunch
##   #vscoco     0      3      1      1      1      0
##   #vscocam    0      0      1      1      1      0
##   #primus     0      0      0      1      1      0
##   #charlotte   0      0      0      0      5      0
##   #livemusic   0      0      0      0      0      0
##   #lunch       0      0      0      0      0      0

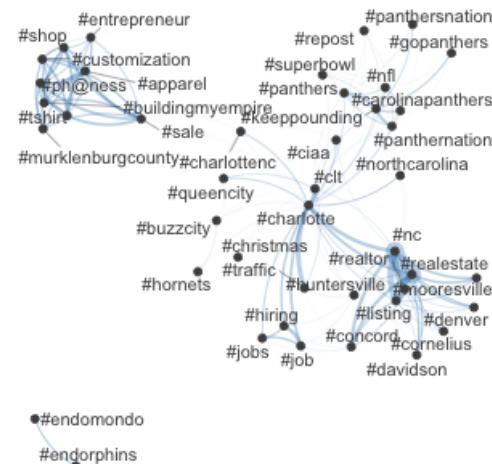
```

```

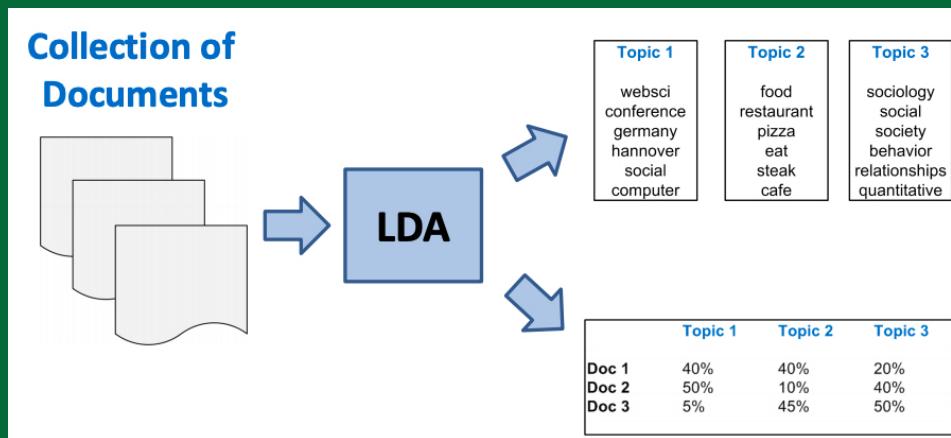
topgat_fcm <- fcm_select(tag_fcm,
                           pattern = toptag)

textplot_network(topgat_fcm,
                 edge_alpha = 0.3,
                 edge_size = 2)

```



Topic Modeling



Latent Dirichlet Allocation

Dimensionality Reduction on DTM

LSA

$$\begin{matrix} \text{documents} \\ \text{words} \\ C \end{matrix} = \begin{matrix} \text{dims} \\ \text{words} \\ U \end{matrix} \begin{matrix} \text{dims} \\ \text{dims} \\ D \end{matrix} \begin{matrix} \text{documents} \\ \text{dims} \\ V^T \end{matrix}$$

TOPIC MODEL

$$\begin{matrix} \text{documents} \\ \text{words} \\ C \end{matrix} = \begin{matrix} \text{topics} \\ \text{words} \\ \Phi \end{matrix} \begin{matrix} \text{topics} \\ \text{topics} \\ \Theta \end{matrix}$$

http://blog.csdn.net/u010693617

normalized co-occurrence matrix mixture components mixture weights

Interpreting Topics

```
# install.packages("topicmodels")
library(topicmodels)

# we now export to a format that we can run the topic model with
dtm <- convert(twdfm, to="topicmodels")

# estimate LDA with K topics
K <- 20
lda <- LDA(dtm, k = K, method = "Gibbs",
            control = list(verbose=25L, seed = 123, burnin = 100, iter = 500))

## K = 20; V = 11767; M = 44955
## Sampling 600 iterations!
## Iteration 25 ...
## Iteration 50 ...
## Iteration 75 ...
## Iteration 100 ...
## Iteration 125 ...
## Iteration 150 ...
## Iteration 175 ...
## Iteration 200 ...
## Iteration 225 ...
## Iteration 250 ...
## Iteration 275 ...
## Iteration 300 ...
## Iteration 325 ...
## Iteration 350 ...
## Iteration 375 ...
## Iteration 400 ...
## Iteration 425 ...
## Iteration 450 ...
## Iteration 475 ...
## Iteration 500 ...
## Iteration 525 ...
## Iteration 550 ...
## Iteration 575 ...
## Iteration 600 ...
## Gibbs sampling completed!
```

Interpreting Topics

```
# see Ch 6 of tidytext mining
library(tidytext)
beta <- tidy(lda, matrix = "beta") %>%
  arrange(desc(topic), desc(beta))

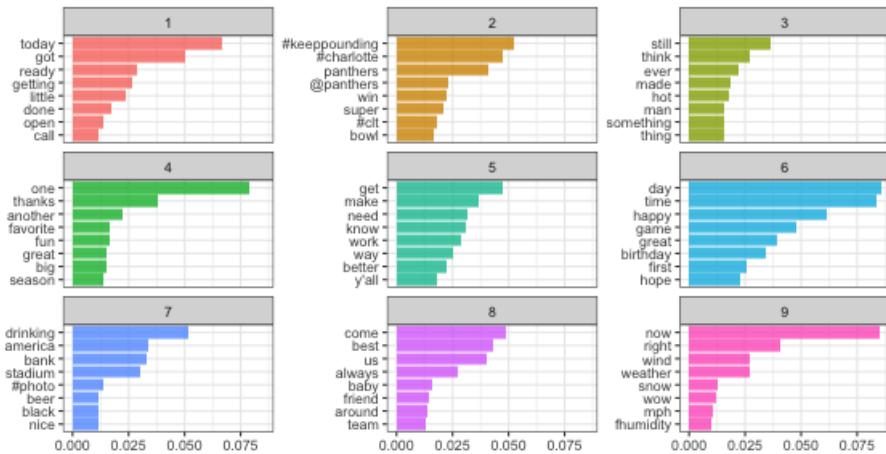
head(beta, n = 10)
```

```
## # A tibble: 10 x 3
##   topic term      beta
##   <int> <chr>    <dbl>
## 1 20 carolina 0.0898
## 2 20 north    0.0801
## 3 20 good     0.0787
## 4 20 charlotte 0.0673
## 5 20 morning   0.0272
## 6 20 pretty    0.0114
## 7 20 news      0.00869
## 8 20 top       0.00817
## 9 20 boys      0.00683
## 10 20 today    0.00654
```

```
top_terms <- beta %>%
  group_by(topic) %>%
  top_n(8, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)
```

```
top_terms
```

```
## # A tibble: 161 x 3
##   topic term      beta
##   <int> <chr>    <dbl>
## 1 1 today    0.0668
## 2 1 got      0.0500
## 3 1 ready    0.0290
## 4 1 getting   0.0267
## 5 1 little   0.0239
## 6 1 done     0.0170
## 7 1 open     0.0134
## 8 1 call     0.0119
## 9 2 #keeppounding 0.0527
## 10 2 #charlotte 0.0472
## # ... with 151 more rows
```



```
# code for topic interpret plot
library(ggthemes)

scale_x_reordered <- function(..., sep = "___") {
  reg <- paste0(sep, ".+\\$")
  ggplot2::scale_x_discrete(labels = function(x) gsub(reg, "", x), ...)
}

reorder_within <- function(x, by, within, fun = mean, sep = "___", ...) {
  new_x <- paste(x, within, sep = sep)
  stats::reorder(new_x, by, FUN = fun)
}

top_terms %>%
  filter(topic %in% 1:9) %>%
  mutate(term = reorder_within(term, beta, topic),
         topic = factor(topic)) %>%
  ggplot(aes(term, beta, fill = topic)) +
  geom_col(alpha = 0.8, show.legend = FALSE) +
  facet_wrap(~ topic, ncol = 3, scales = "free_y") +
  theme_tufte(base_family = "IBMPlexSans", ticks = FALSE) +
  scale_x_reordered() +
  theme_bw() +
  coord_flip() +
  labs(x = "", y = "")
```

Retrieving Documents by Topic

```
td_gamma <- tidy(lda, matrix = "gamma")
head(td_gamma)
```

```
## # A tibble: 6 x 3
##   document topic  gamma
##   <chr>     <int> <dbl>
## 1 text1      1 0.0417
## 2 text2      1 0.0455
## 3 text3      1 0.0490
## 4 text4      1 0.0439
## 5 text5      1 0.0472
## 6 text6      1 0.0424
```

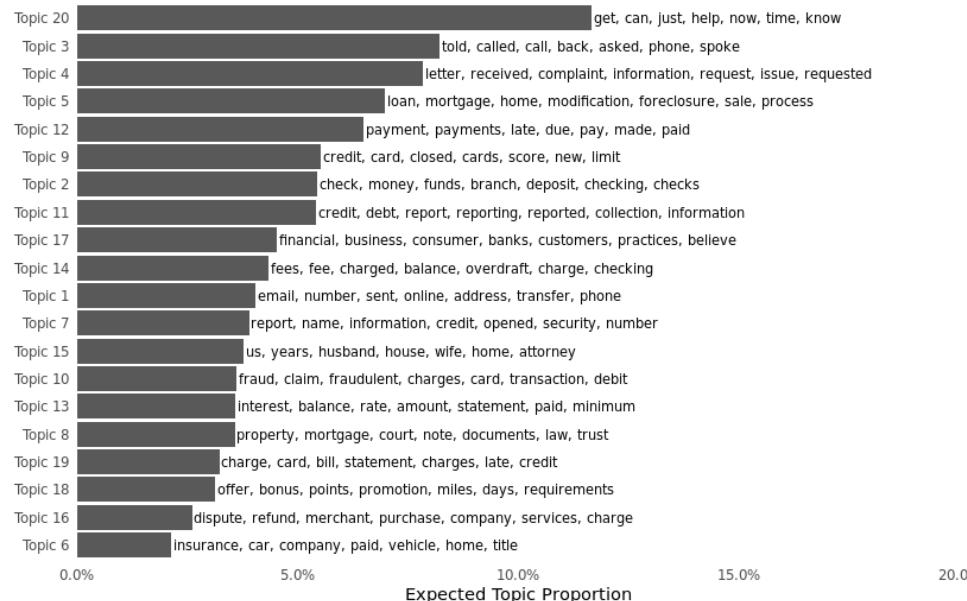
```
td_gamma %>%
  filter(topic == 2) %>%
  arrange(desc(gamma)) %>%
  head(n=6)
```

```
## # A tibble: 6 x 3
##   document topic  gamma
##   <chr>     <int> <dbl>
## 1 text5172   2 0.25
## 2 text24680  2 0.246
## 3 text1337   2 0.242
## 4 text10525  2 0.227
## 5 text23298  2 0.223
## 6 text31622  2 0.214
```

Summarizing Topic Models

Top 20 topics by prevalence in the CFPB Complaints

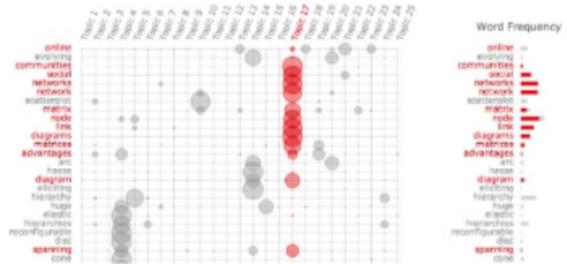
With the top words that contribute to each topic



For code, see <https://github.com/wesslen/nlp-ryan/blob/master/complaints.R>

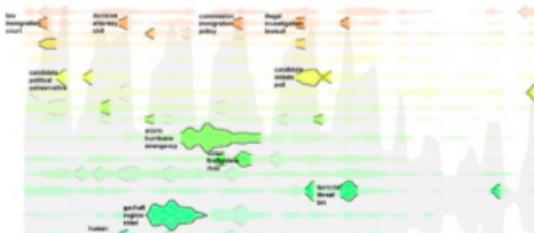
Visualizing Topic Models

Topic-Oriented



Termite (Chuang et al 2012)

Time-Oriented



LeadLine (Dou et al 2012)

Hierarchical



HierarchicalTopics (Dou et al 2013)

Graph (Relational) Based



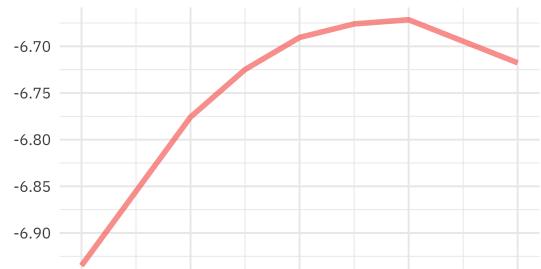
TopicPanorama (Wang et al 2016)

Evaluating Topic Models

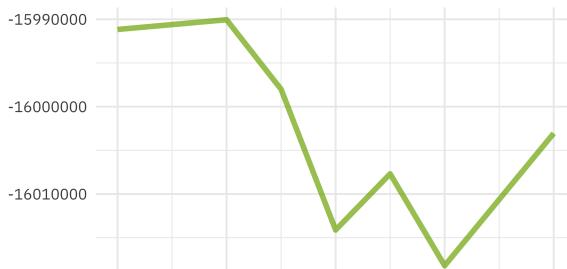
Model diagnostics by number of topics

These diagnostics indicate that a good number of topics would be around 60

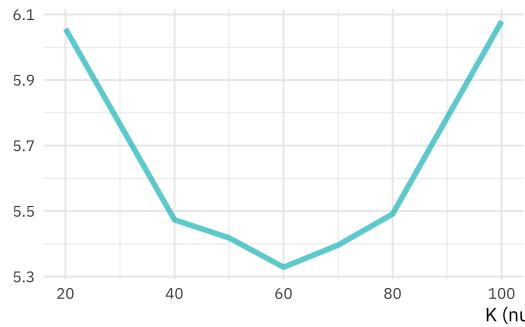
Held-out likelihood



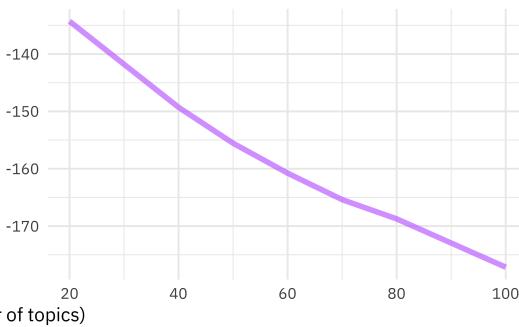
Lower bound



Residuals

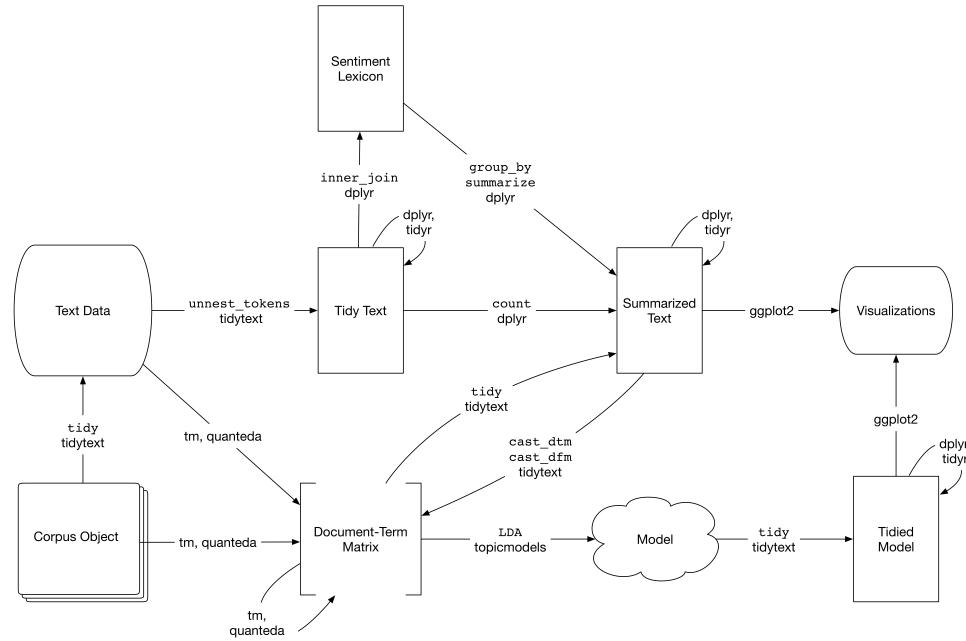


Semantic coherence

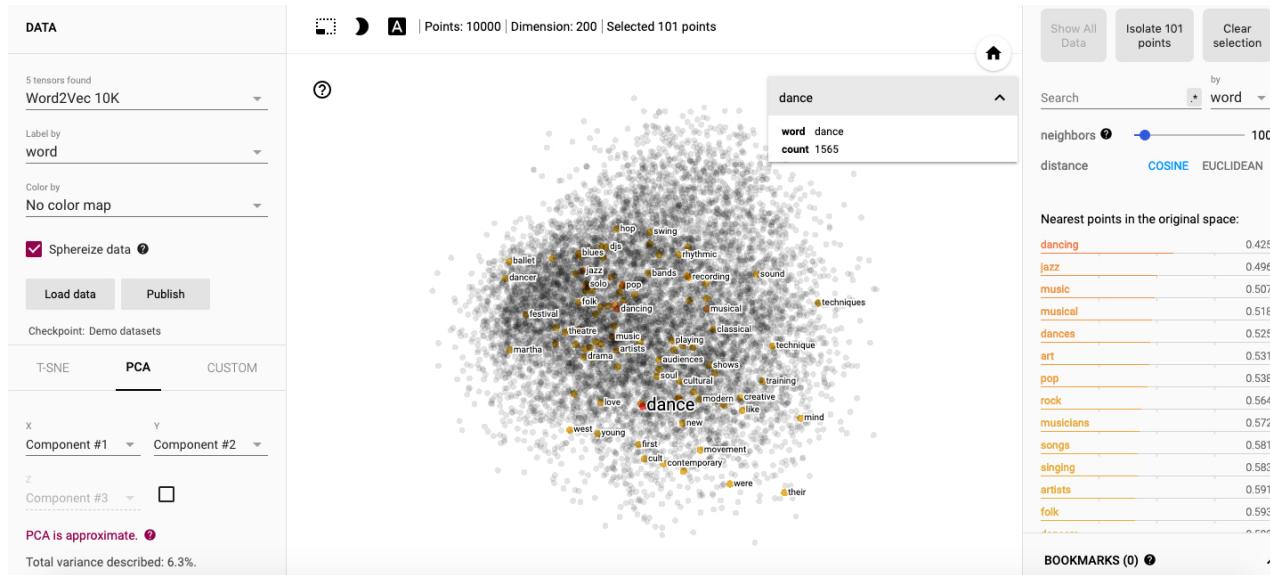


<https://juliasilge.com/blog/evaluating-stm/>

Text Workflows



Word Embedding Models



<https://projector.tensorflow.org/>