Topic Modeling in R Studio

2nd Summer School in Computational Social Sciences

Ayse Deniz Lokmanoglu

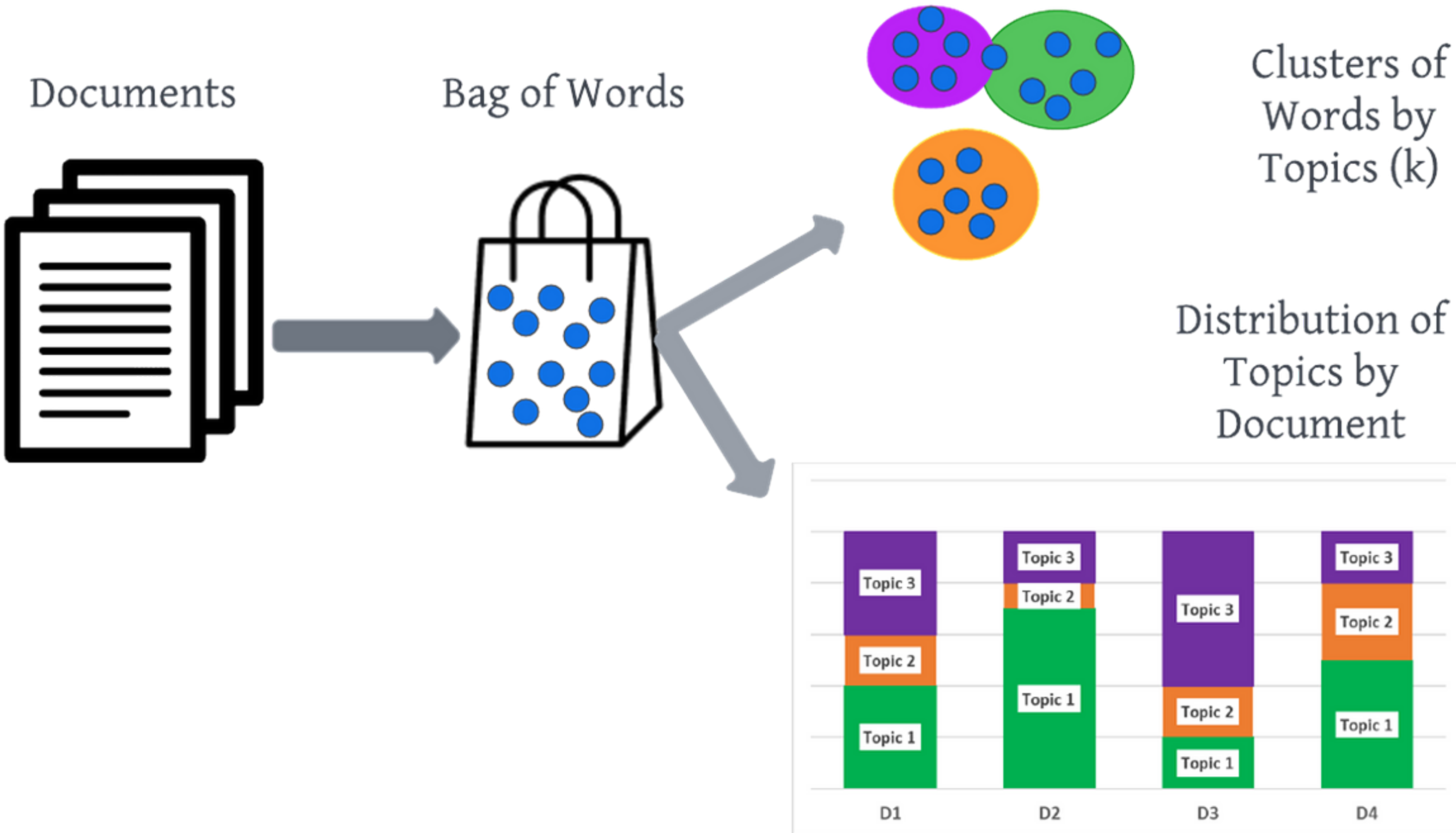
July 26, 2022

What is text analysis?

- Deriving information from a text
- When we have a lot of text, computational methods help us derive detailed information from the text faster, and with less researcher bias
- Today we will learn how to do topic modeling using R Studio

What is Topic Modeling?

- Topic modeling is a bag-of-words approach
- Topic modeling, or Latent Dirichlet Allocation (LDA), is a computational content analysis tool that surfaces the "hidden thematic structure of a collection of text" (Maier et al., 2018: 93)
- Through an inductive approach to quantitative measurements, it allows researchers to conduct semantic analysis on a large number of texts.
- LDA conducts measurements in three levels: corpus, documents, and terms. The corpus consists of a collection of documents, and each document consists of a collection of words (referred to as terms in the algorithm).
- The LDA algorithm models the representation of the words, with each other, within a document and within the corpus, through "topics" (Maier et al., 2018: 94).
- These facilitate researchers to label topics inductively, by using both the words within each topic, as well as the documents in each topic.
- Thus, LDA analysis allows a document to represent multiple topics, providing a deeper insight into the thematic structure of the corpus.


Documents
Bag of Words
Clusters of Words by Topics (k)
Distribution of Topics by Document
Topic 3
Topic 2
Topic 1
Topic 3
Topic 2
Topic 1
Topic 3
Topic 2
Topic 1
Topic 3
Topic 2
Topic 1
D1
D2
D3
D4

Let's start coding!

First we install all the packages! You only have to do this once, for all other times you just need to load the packages (see next slide)

```
# this code is to install all the packages, you only need to run this once, afterwards all you need is
install.packages(c(
  "tidyverse",    # foundation packages needed for text analysis
  "tidytext",     # foundation packages needed for text analysis
  "dplyr",        # foundation packages needed for text analysis
  "tm",           # text mining package
  "quanteda",     # Quantitative Analysis of Textual Data
  "ldatuning",    # Tuning of the Latent Dirichlet Allocation Models Parameters
  "topicmodels", # Topic Model package
  "scales",       # scale functions for visualizations
  "ggthemes",     # graph theme options
  "jtools",       # ggplot2 themes
  "ggplot2",      # visualization
  "lubridate",    # for dates
  "zoo"           # another package for dates
  ))
```

Load packages

```
library(tidyverse)
library(tidytext)
library(dplyr)
library(tm)
library(quanteda)
library(ldatuning)
library(topicmodels)
library(scales)
library(ggthemes)
library(lubridate)
library(jtools)
```

How to find details on packages? Type the package name preceded by ? and you will see the package details on the help window

```
?tidyverse
```

Our dataset

```
url<-c("https://dataverse.harvard.edu/api/access/datafile/6389385")
mydata <- read_csv(url)
glimpse(mydata)
```

```
## Rows: 1,500
## Columns: 5
## $ ...1          <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1~
## $ index         <dbl> 98474, 23319, 144569, 38059, 97919, 131938, 21368, 16530~
## $ date          <date> 2021-11-21, 2021-07-29, 2021-04-16, 2021-08-26, 2021-11~
## $ source.domain <chr> "foxnews.com", "foxnews.com", "dailywire.com", "abc13.co~
## $ originaltext  <chr> "chicago mayor needs to dump police boss if <U+0091>crim~
```

As you can see we have an index column from csv labelled *...1*, and date column. So first lets remove the extra column, make sure date is coded as date.

- Why did we write the command `select()` with its package name?

```
mydata <- mydata %>%
dplyr::select(-...1) %>% #from dplyr we drop the column with -, and this case
mutate(date=ymd(date)) %>% #using lubridate we change date column to date variable
mutate(text=originaltext) #Create a new column labelled text - to keep original text safe
glimpse(mydata) #lets see what our dataset is made of
```

```
## Rows: 1,500
## Columns: 5
## $ index         <dbl> 98474, 23319, 144569, 38059, 97919, 131938, 21368, 16530~
## $ date          <date> 2021-11-21, 2021-07-29, 2021-04-16, 2021-08-26, 2021-11~
## $ source.domain <chr> "foxnews.com", "foxnews.com", "dailywire.com", "abc13.co~
## $ originaltext  <chr> "chicago mayor needs to dump police boss if <U+0091>crim~
## $ text          <chr> "chicago mayor needs to dump police boss if <U+0091>crim~
```

Preprocessing, getting ready for LDA

- Tokenize it

```
toks <- tokens(mydata$text,
           remove_punct = TRUE,
           remove_symbols = TRUE,
           remove_numbers = TRUE,
           remove_url = TRUE,
           remove_separators = TRUE,
           split_hyphens = FALSE,
           include_docvars = TRUE,
           padding = FALSE) %>%
tokens_remove(stopwords(language = "en")) %>% #for this we used combined stopwords list from google with quanteda
tokens_select(min_nchar = 2)
head(toks)
```

```
## Tokens consisting of 6 documents.
## text1 :
##  [1] "chicago"   "mayor"     "needs"     "dump"      "police"    "boss"
##  [7] "crime"     "pandemic"  "isn"       "addressed" "critic"    "says"
## [ ... and 17 more ]
##
## text2 :
##  [1] "randi"      "weingarten" "ripped"     "telling"    "msnbc"
##  [6] "going"      "try"        "reopen"     "schools"    "cdc"
## [11] "mask"       "guidance"
## [ ... and 17 more ]
##
## text3 :
##  [1] "pfizer"  "ceo"     "third"   "covid"   "vaccine" "dose"    "likely"
##  [8] "needed"  "within"  "months"  "pfizer"  "ceo"
## [ ... and 22 more ]
##
## text4 :
```

Next Steps

- Change it into a document-feature matrix
- Match your dfm object with your original data frame through index

```
dfm_counts<- dfm(toks)
rm(toks)
docnames(dfm_counts)<-mydata$index#remove unused files to save space
```

LDA Object

- Convert dfm object to an LDA object

```
dtm_lda <- convert(dfm_counts, to = "topicmodels",docvars = dfm_counts@docvars) #convert
n <- nrow(dtm_lda) # number of rows for cross-validation method
rm(dfm_counts) # remove for space
dtm_lda
```

```
## <<DocumentTermMatrix (documents: 1500, terms: 9225)>>
## Non-/sparse entries: 38025/13799475
## Sparsity           : 100%
## Maximal term length: 22
## Weighting          : term frequency (tf)
```

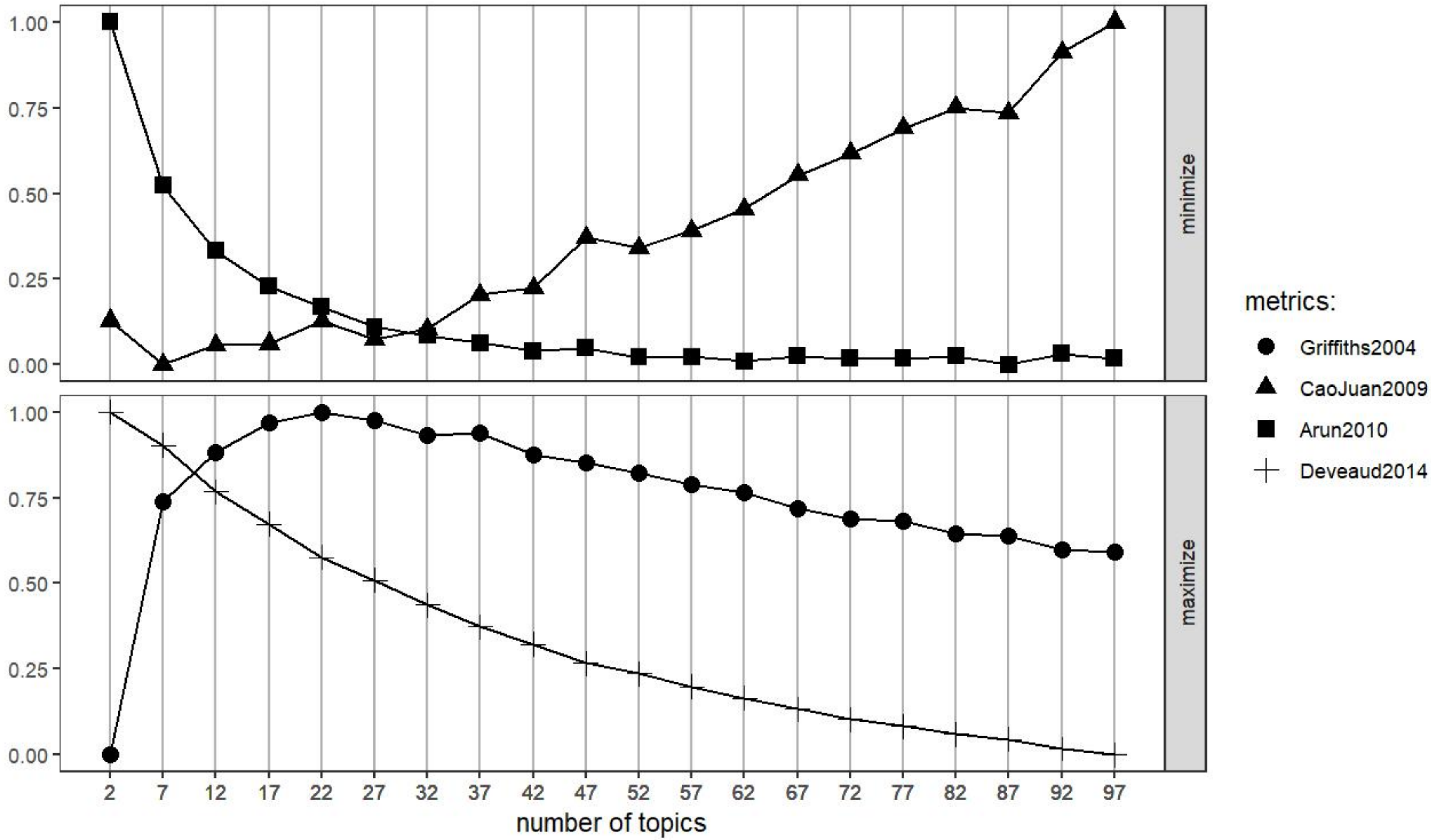
Let's run our topic model!

Find K

- This function is from ldatuning package **I ran the code already to save time** *You can run it on your own time by erasing the markdown option 'eval=FALSE'*

```
Sys.time()
result <- FindTopicsNumber(
dtm_lda,
topics = seq(2,50,by=10), # Specify how many topics you want to try.
metrics = c("Griffiths2004", "CaoJuan2009", "Arun2010", "Deveaud2014"),
method = "Gibbs",
control = list(seed = 9), # random seed number
mc.cores = 2L,
verbose = TRUE
)
Sys.time()
save(result, file="Class_FindK.Rda")
FindTopicsNumber_plot(result)
ggsave("Class_Find_K.jpg", width=8.5, height=5, dpi=150)
```

Plot Result



1.00
0.75
0.50
0.25
0.00
minimize
maximize
2 7 12 17 22 27 32 37 42 47 52 57 62 67 72 77 82 87 92 97
number of topics
metrics:
Griffiths2004
CaoJuan2009
Arun2010
Deveaud2014

Let's run our topic model!

We identified our optimal k as 22 from the graph, but for our ease of analysis we will model on 5 topics

```
Sys.time()
```

```
## [1] "2022-07-21 20:47:35 CDT"
```

```
covid_lda <- LDA(dtm_lda, k = 5, control = list(seed = 1234))
save(covid_lda, file="Class_lda_K5.Rda") #always save your variables
Sys.time()
```

```
## [1] "2022-07-21 20:47:44 CDT"
```

```
covid_lda
```

```
## A LDA_VEM topic model with 5 topics.
```

Extract data from the lda model

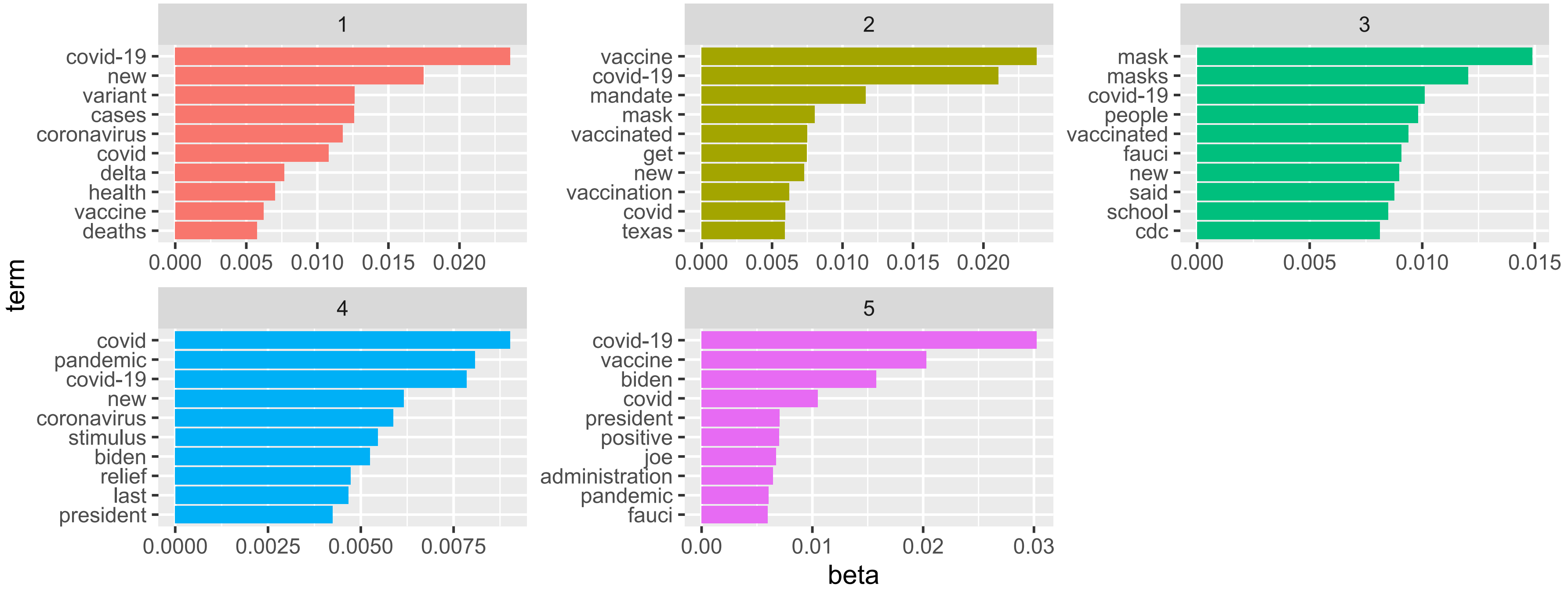
- We can extract top words and documents

```
covid_topics <- tidy(covid_lda, matrix = "beta")
head(covid_topics)
```

```
## # A tibble: 6 x 3
##   topic term        beta
##   <int> <chr>      <dbl>
## 1     1 chicago 6.16e- 4
## 2     2 chicago 5.42e- 4
## 3     3 chicago 1.10e- 4
## 4     4 chicago 6.74e-12
## 5     5 chicago 1.13e-26
## 6     1 mayor   1.04e- 3
```

Visualize top words

```
covid_top_terms <- covid_topics %>%
  group_by(topic) %>%
  slice_max(beta, n = 10) %>%
  ungroup() %>%
  arrange(topic, -beta)
```



We can label topics using top words

- In our case it looks like
    - Topic 1: Variants
    - Topic 2: Vaccine Mandates
    - Topic 3: Pandemic Regulations
    - Topic 4: Relief Stimulus
    - Topic 5: Covid19 and Government
- We should create a variable called topic_names and save it for future

```
topic_names<-c("Variants", "Vaccine_Mandates", "Pandemic_Regulations", "Relief_Stimulus", "Covid19
```

Why did I use underscore when creating the `topic_names` variable?

Document-topic probabilities

- We will extract the γ (“gamma”) value which is per-document-per-topic probabilities.This value estimates the proportion of words from each document that belong to that topic.

```
covid_documents <- tidy(covid_lda, matrix = "gamma")
glimpse(covid_documents)
```

```
## Rows: 7,500
## Columns: 3
## $ document <chr> "98474", "23319", "144569", "38059", "97919", "131938", "2136~
## $ topic    <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
## $ gamma    <dbl> 0.2794112292, 0.0016247089, 0.0013874387, 0.7576687978, 0.001~
```

Join with original document?

- We saw in our gamma values we have a document number **equal to our index** (from previous slides). We can join with our document and see for example which topics come from what domains?
- But first we can see that the dimensions of `covid_documents` and `mydata` are different, why?

```
dim(covid_documents)
```

```
## [1] 7500    3
```

```
dim(mydata)
```

```
## [1] 1500    5
```

Wide documents

- What we have is a long document, What we need to do is change the document to wider, having each document with topics as columns
- We can reshape the data frame using`dpylr`'s `pivot_wider()` and `pivot_longer()`

```
?pivot_wider()
```

```
covid_documents_wide<- covid_documents %>%
pivot_wider(names_from = topic,
          values_from = gamma)
dim(covid_documents_wide)
```

```
## [1] 1500    6
```

```
dim(mydata)
```

```
## [1] 1500    5
```

Column Names - Good Practice

- It is good practice to not have numbers as column names, so let's add a prefix of X

```
colnames(covid_documents_wide)[2:6] <- paste("X", colnames(covid_documents_wide[,c(2:6)]), sep = "_")
colnames(covid_documents_wide)
```

```
## [1] "document" "X_1"      "X_2"      "X_3"      "X_4"      "X_5"
```

- We can also add our topic_names the same way

```
covid_documents_wide_test<-covid_documents_wide # to save a backup copy
colnames(covid_documents_wide_test)[2:6] <- topic_names
colnames(covid_documents_wide_test)
```

```
## [1] "document"             "Variants"                "Vaccine_Mandates"
## [4] "Pandemic_Regulations"   "Relief_Stimulus"         "Covid19_and_Government"
```

Now let's join!

```
meta_theta_df<-left_join(mydata, covid_documents_wide, by=c("index" = "document"))
```

```
## Error in `left_join()`:
## ! Can't join on `x$index` x `y$index` because of incompatible types.
## i `x$index` is of type <double>>.
## i `y$index` is of type <character>>.
```

We need to change document in covid_documents_wide to number

```
covid_documents_wide <- covid_documents_wide %>%
  mutate(document=as.numeric(document))
typeof(covid_documents_wide$document) # this is a way to check the type
```

```
## [1] "double"
```

Let's try again!

```
meta_theta_df<-left_join(mydata, covid_documents_wide, by=c("index" = "document"))
meta_theta_df
```

```
## # A tibble: 1,500 x 10
##     index date       source.domain    originaltext text       X_1     X_2     X_3
##     <dbl> <date>     <chr>            <chr>        <chr>    <dbl>   <dbl>   <dbl>
##  1  98474 2021-11-21 foxnews.com      chicago may~ chic~ 0.279   0.716   0.00162
##  2  23319 2021-07-29 foxnews.com      randi weing~ rand~ 0.00162 0.00162 0.894
##  3 144569 2021-04-16 dailywire.com    pfizer ceo:~ pfiz~ 0.00139 0.00139 0.00139
##  4  38059 2021-08-26 abc13.com        texas a&m r~ texa~ 0.758   0.174   0.0638
##  5  97919 2021-11-28 silive.com       nyc civil s~ nyc ~ 0.00188 0.992   0.00188
##  6 131938 2021-03-12 fox13news.com    american, u~ amer~ 0.00196 0.00196 0.00196
##  7  21368 2021-07-23 huffpost.com     ted cruz<U+~ ted ~ 0.00143 0.00143 0.00143
```

Let's look at the domains

```
domains <- meta_theta_df %>%
  dplyr::select(source.domain, X_1:X_5) %>% # selected domains and topic gammas for each document
  group_by(source.domain) %>% # grouped by domains
  summarise(across(everything(), sum)) # summed all the topic gammas
dim(domains)
```

```
## [1] 472   6
```

Now you can see we have a new data-set with 472 domains, and the topic probabilities for each domain

Which domain has the highest topic probabilities?

- let's do topic 4 and the top 10 domains

```
topic4 <- domains %>%
slice_max(X_4, n=10)
head(topic4)
```

```
## # A tibble: 6 x 6
##   source.domain        X_1   X_2   X_3   X_4   X_5
##   <chr>              <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 nbcnews.com         9.06  4.84  9.04 11.8  10.3
## 2 nytimes.com         9.95 10.5   6.08  7.92  7.55
## 3 cnn.com            11.1   6.81  7.93  7.75 14.4
## 4 washingtonpost.com  5.56  6.33 12.7   6.79  2.66
## 5 cnbc.com            5.15  1.61  2.99  6.55  6.71
## 6 dailywire.com       8.04  4.10  6.15  5.69  6.02
```

- play with different topics and `slice_max()` & `slice_min()` from `dplyr` package

Visualize comparison of domains

- Let's compare cnn.com and nbcnews.com
  - We want to make a graph bar graph that has both domains and topic probabilities.
  - First let's create a smaller dataframe with the two domains

```
domain_comp<- domains %>%
  filter(source.domain=="cnn.com" | source.domain=="nbcnews.com")
domain_comp
```

```
## # A tibble: 2 x 6
##   source.domain   X_1   X_2   X_3   X_4   X_5
##   <chr>         <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 cnn.com       11.1   6.81  7.93  7.75  14.4
## 2 nbcnews.com    9.06  4.84  9.04 11.8   10.3
```

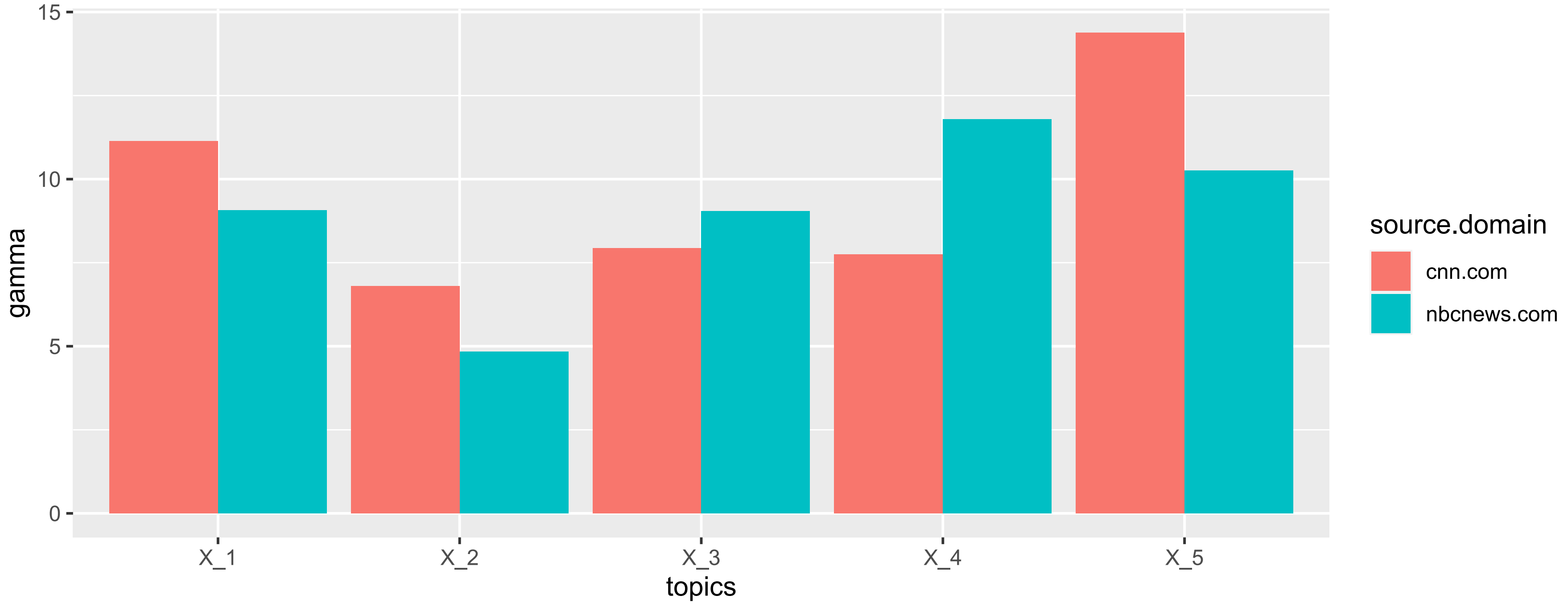
Set the data for bar graph

- We need to make domains a group, topics as x axis and gamma values as y.
- So we need to make the document long, by `dplyr` packages `pivot_longer()`

```
domain_long <- domain_comp %>%
pivot_longer(!source.domain,
           names_to = "topics", # names as topic
           values_to = "gamma")
head(domain_long)
```

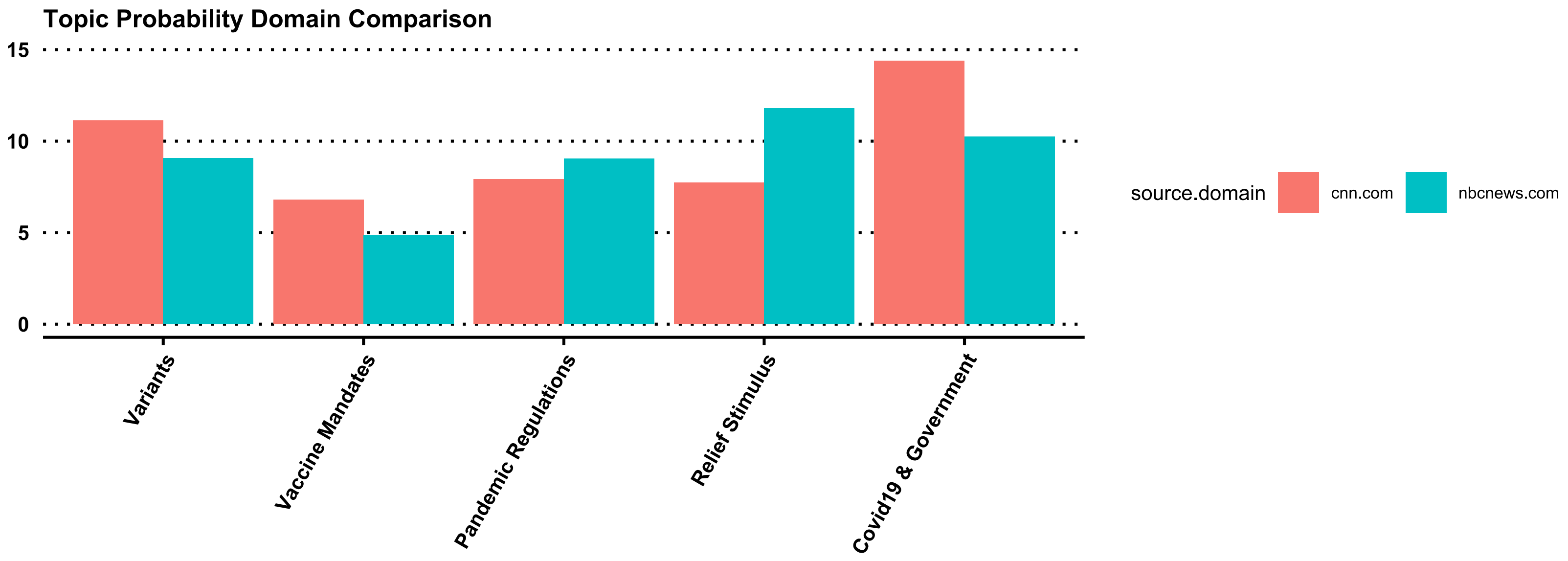
```
## # A tibble: 6 x 3
##   source.domain topics gamma
##   <chr>         <chr>  <dbl>
## 1 cnn.com       X_1    11.1 
## 2 cnn.com       X_2     6.81
## 3 cnn.com       X_3     7.93
## 4 cnn.com       X_4     7.75
## 5 cnn.com       X_5    14.4 
## 6 nbcnews.com   X_1     9.06
```

Now lets graph it



Play with graphs

- We can also add our topic labels, play styles using `ggthemes()` package



Lastly let's do topics over time

- For this we well again use our meta_theta_df document, this time we will summarize by dates

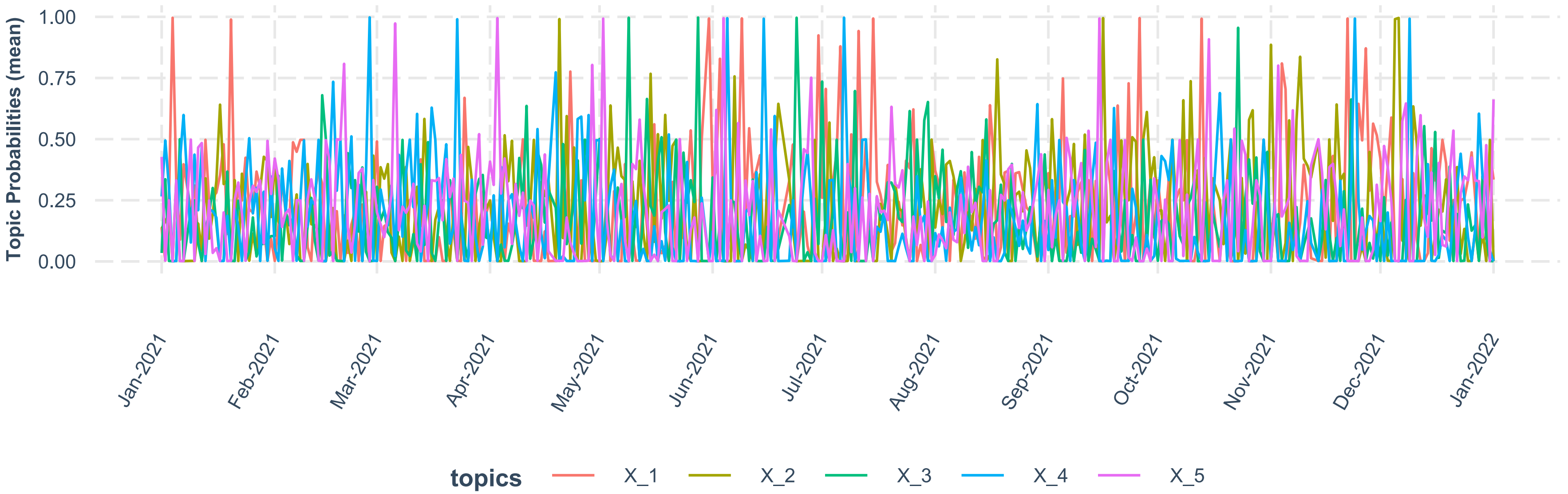
```
topics_time <- meta_theta_df %>%
dplyr::select(date, X_1:X_5) %>% # selected dates and topic gammas for each document
group_by(date) %>% # grouped by dates
summarise(across(everything(), mean)) # summed all the topic gammas
```

- We have to make this document long to plot it, using `pivot_longer()`

```
topic_time_long <- topics_time %>%
pivot_longer(!date, # long from date
           names_to = "topics", # names as topic
           values_to = "gamma") # values as gamma
topic_time_long
```

```
## # A tibble: 1,785 x 3
##    date       topics   gamma
##    <date>     <chr>    <dbl>
##  1 2021-01-01 X_1    0.138
##  2 2021-01-01 X_2    0.135
##  3 2021-01-01 X_3    0.0345
##  4 2021-01-01 X_4    0.266
##  5 2021-01-01 X_5    0.427
##  6 2021-01-02 X_1    0.00309
##  7 2021-01-02 X_2    0.163
##  8 2021-01-02 X_3    0.336
##  9 2021-01-02 X_4    0.495
## 10 2021-01-02 X_5    0.00309
## # ... with 1,775 more rows
```

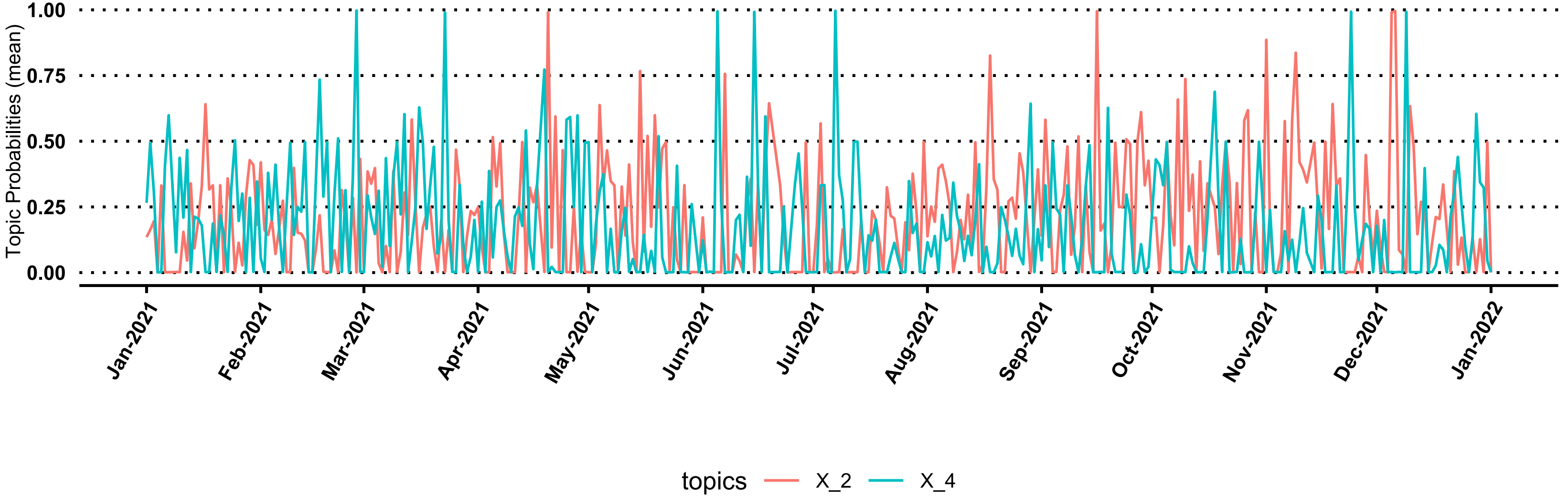
Visualize it



- This is very crowded

Simpler graph

- let's pick topics 2 and 4



Even simpler graph

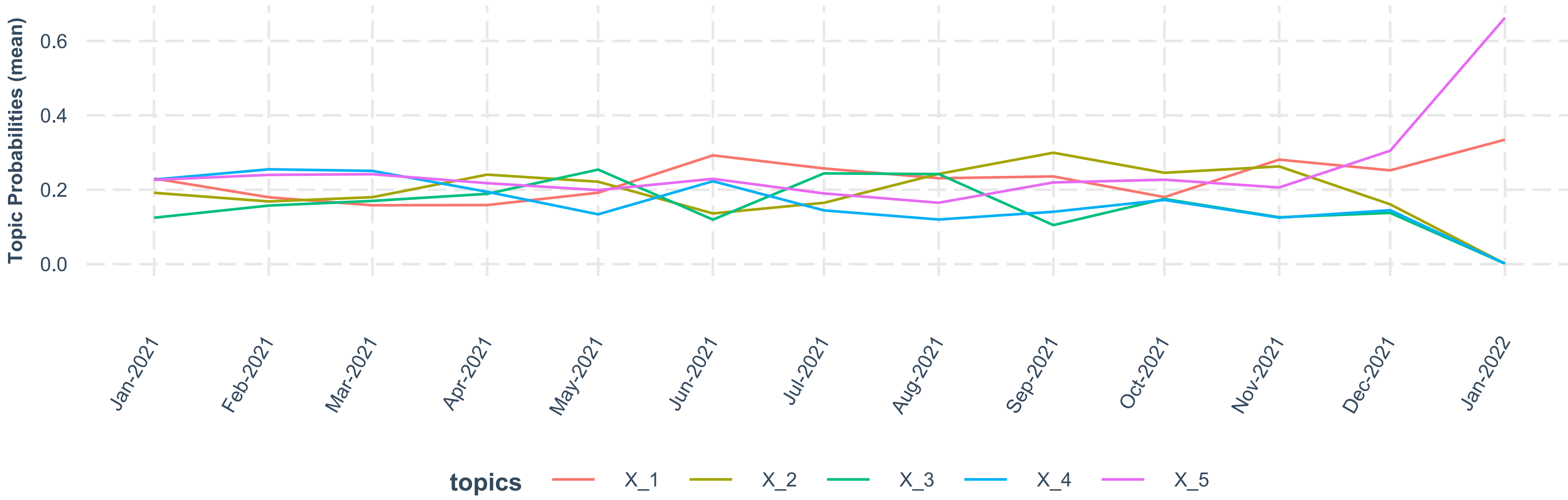
- Make it monthly

```
meta_theta_df$yearmonth<-my(zoo::as.yearmon(meta_theta_df$date))
topics_year_mon <- meta_theta_df %>%
dplyr::select(yearmonth, X_1:X_5) %>% # selected dates and topic gammas for each document
group_by(yearmonth) %>% # grouped by dates
summarise(across(everything(), mean))
# pivot longer like before
topics_year_mon_long <- topics_year_mon %>%
pivot_longer(!yearmonth, # long from date
           names_to = "topics", # names as topic
           values_to = "gamma") # values as gamma
head(topics_year_mon_long)
```

```
## # A tibble: 6 x 3
##   yearmonth  topics gamma
##   <date>     <chr>  <dbl>
## 1 2021-01-01 X_1    0.230
## 2 2021-01-01 X_2    0.192
## 3 2021-01-01 X_3    0.125
## 4 2021-01-01 X_4    0.227
## 5 2021-01-01 X_5    0.226
## 6 2021-02-01 X_1    0.180
```

Lets graph it again

- Mean monthly topic probabilities



Questions?

Thank you!

My email is ayse.lokmanoglu@northwestern.edu and my github page where I have more challenging topic model codes to play with!