Untitled

TA: Leslie Huang

Course: Text as Data

Date: 4/12/18

Recitation 11: Unsupervised Learning IIa

```
rm(list = ls())

setwd("/Users/Lingyi/TAD/lab/Text-as-Data-Lab-Spr2018/W11_04_12_18/")

set.seed(1234)

# Check for these packages, install them if you don't have them
# install.packages("tidytext")
#install.packages("topicmodels")
#install.packages("ldatuning")
# install.packages("stringi")
#install.packages("rjson")

libraries <- c("ldatuning", "topicmodels", "ggplot2", "dplyr", "rjson", "quanteda", "lubridate", "paral
lapply(libraries, require, character.only = TRUE)
```

```
## Loading required package: ldatuning
## Loading required package: topicmodels
## Loading required package: ggplot2
## Loading required package: dplyr
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
## Loading required package: rjson
## Loading required package: quanteda
## quanteda version 1.0.0
## Using 3 of 4 threads for parallel computing
```

```
##
## Attaching package: 'quanteda'

## The following object is masked from 'package:utils':
##
##     View

## Loading required package: lubridate

##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date

## Loading required package: parallel

## Loading required package: doParallel

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'doParallel'

## Loading required package: tidytext

## Warning: package 'tidytext' was built under R version 3.4.4

## Loading required package: stringi

## Warning: package 'stringi' was built under R version 3.4.4

## [[1]]
## [1] TRUE
##
## [[2]]
## [1] TRUE
##
## [[3]]
## [1] TRUE
##
## [[4]]
## [1] TRUE
##
## [[5]]
## [1] TRUE
##
## [[6]]
## [1] TRUE
##
## [[7]]
## [1] TRUE
##
## [[8]]
## [1] TRUE
##
## [[9]]
## [1] FALSE
##
## [[10]]
## [1] TRUE
```

```
##
## [[11]]
## [1] TRUE
```

1 Preprocessing

```
# Load data
blm_tweets <- read.csv("blm_samp.csv", stringsAsFactors = F)

# Create date vectors
blm_tweets$datetime <- as.POSIXct(strptime(blm_tweets$created_at, "%a %b %d %T %z %Y",tz = "GMT")) # fu
blm_tweets$date <- mdy(paste(month(blm_tweets$datetime), day(blm_tweets$datetime), year(blm_tweets$date

# Collapse tweets so we are looking at the total tweets at the day level
blm_tweets_sum <- blm_tweets %>% group_by(date) %>% summarise(text = paste(text, collapse = " "))

# Remove non ASCII characters
blm_tweets_sum$text <- stringi::stri_trans_general(blm_tweets_sum$text, "latin-ascii")

# Removes solitary letters
blm_tweets_sum$text <- gsub(" [A-z] ", " ", blm_tweets_sum$text)

# Create DFM
blm_dfm <-dfm(blm_tweets_sum$text, stem = F, remove_punct = T, tolower = T, remove_twitter = T, remove_
```

2 Selecting K

```
# Identify an appropriate number of topics (FYI, this function takes a while)
#k_optimize_blm <- FindTopicsNumber(
#  blm_dfm,
#  topics = seq(from = 2, to = 30, by = 1),
#  metrics = c("Griffiths2004", "CaoJuan2009", "Arun2010", "Deveaud2014"),
#  method = "Gibbs",
#  control = list(seed = 2017),
#  mc.cores = 6L,
#  verbose = TRUE
#)
```

```
#FindTopicsNumber_plot(k_optimize_blm)

# Where do these metrics come from?

# Go here for the citations (and another tutorial)
# https://cran.r-project.org/web/packages/ldatuning/vignettes/topics.html

# What should you consider when choosing the number of topics you use in a topic model?

# What does robustness mean here?
```

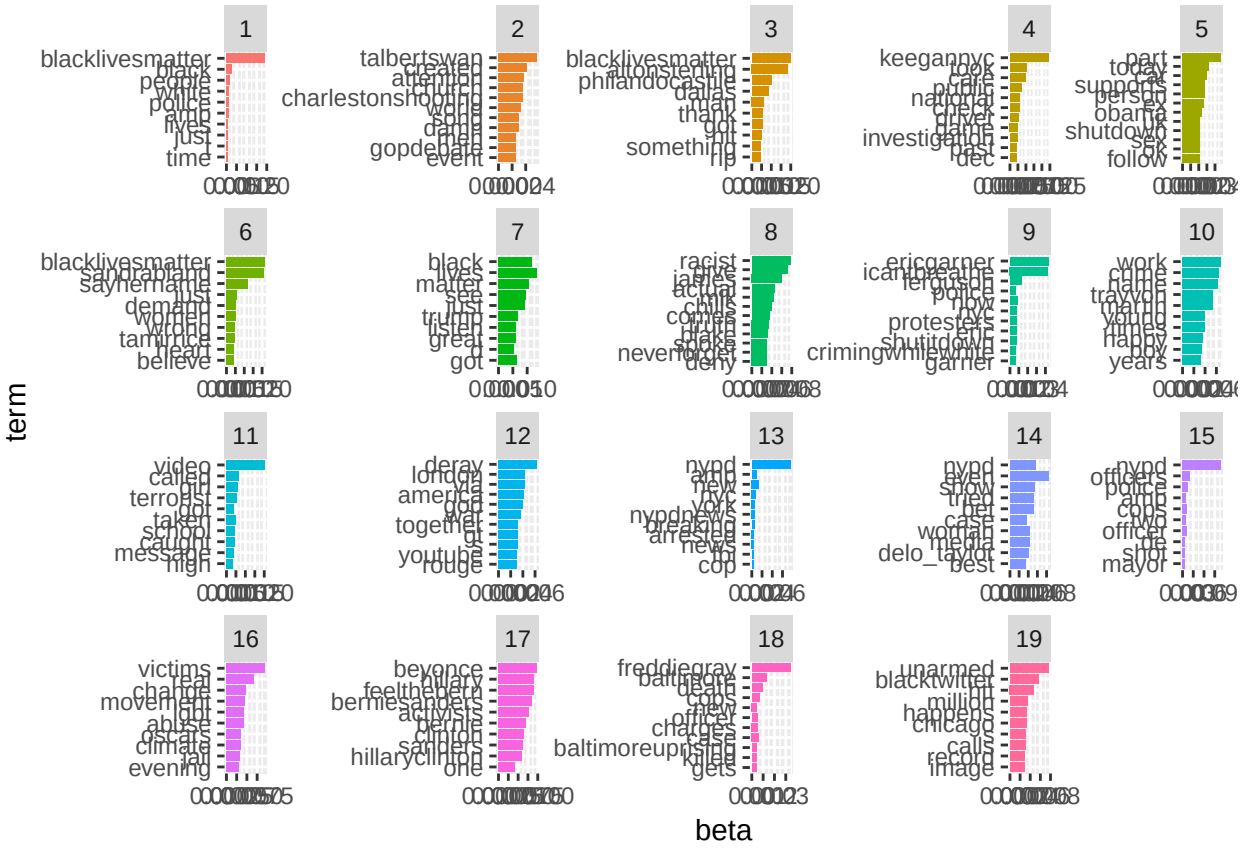
3 Visualizing Word weights

```
# Set number of topics
k <- 19

# Fit the topic model with the chosen k
blm_tm <- LDA(blm_dfm, k = k, method = "Gibbs",  control = list(seed = 1234))

# Other parameters that we do not use here (because they increase the time the model takes) can be pass
#?`LDAcontrol-class`
# iter : num iterations
# thin : every thin iteration is returned for iter iterations
# burnin : number of initial iterations discarded

## Letter soup

# gamma = posterior topic distribution over documents
#blm_tm@gamma
# Docs x topic_proportions array
rowSums(blm_tm@gamma) # Each row sums to 1
```

```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [36] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [71] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [106] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [141] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [176] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [211] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [246] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [281] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [316] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [351] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [386] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [421] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [456] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [491] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [526] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [561] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [596] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [631] 1 1 1 1 1 1 1 1 1 1 1 1
```

```
# beta = log word distributions over topics
#blm_tm@beta
# Topics x log word proportion

# alpha = scaling parameter for symmetric Dirichlet distribution over topic distributions
# in this implementation, alpha is estimated as 50/k but can be set in parameters of LDA()
#blm_tm@alpha

# z = topic assignments of specific words
#blm_tm@z

# Quickly extracts the word weights and transforms them into a data frame
blm_topics <- tidy(blm_tm, matrix = "beta")
```

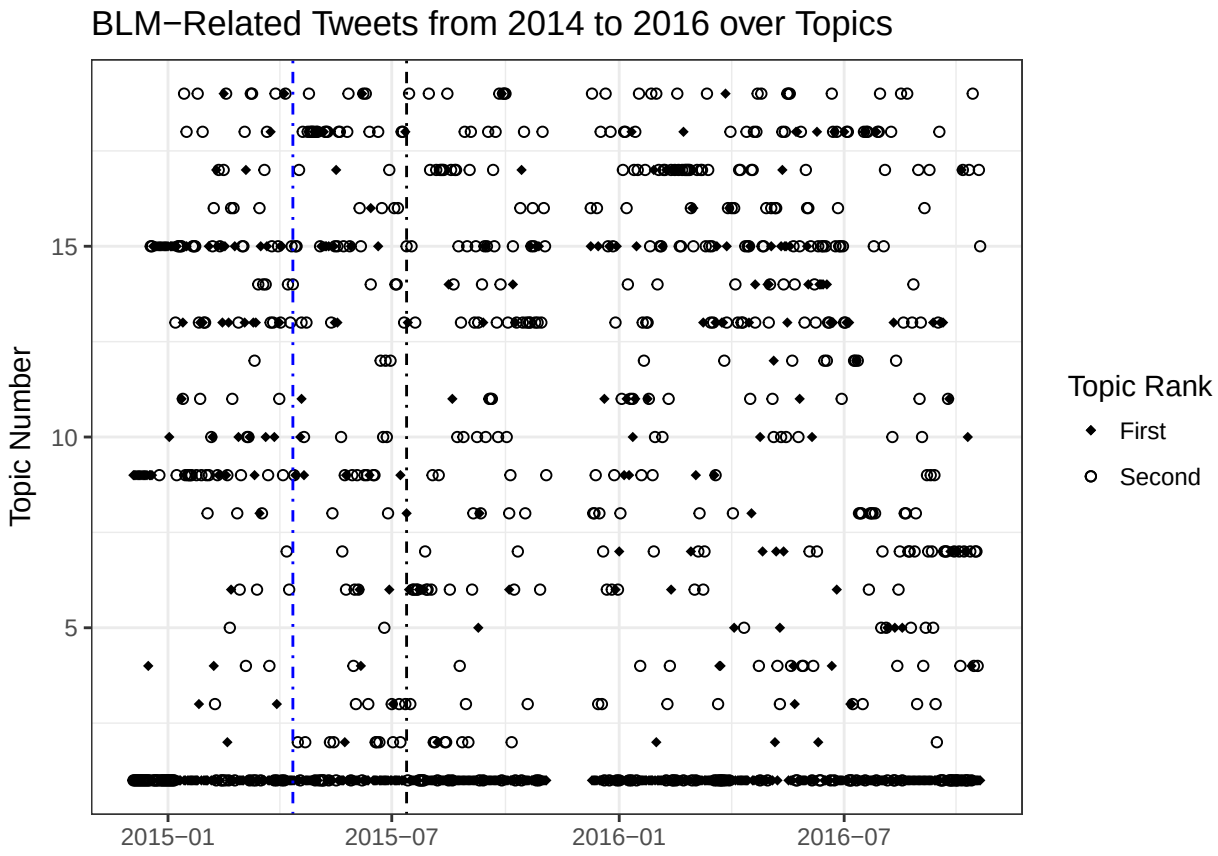
```
# Side note: You can pass objects between tidytext() and topicmodels() functions because tidytext() imp

# Generates a df of top terms
blm_top_terms <- blm_topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

# Creates a plot of the weights and terms by topic
blm_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()
```



4 Visualizing topic trends over time

```
# Store the results of the distribution of topics over documents
doc_topics <- blm_tm@gamma

# Store the results of words over topics
words_topics <- blm_tm@beta
```

```r
# Transpose the data so that the days are columns
doc_topics <- t(doc_topics)

# Arrange topics
# Find the top topic per column (day)
max<-apply(doc_topics, 2, which.max)

# Write a function that finds the second max
which.max2 <- function(x){
  which(x == sort(x,partial=(k-1))[k-1])
}

max2 <- apply(doc_topics, 2, which.max2)
max2 <- sapply(max2, max)

# Coding police shooting events
victim <- c("Freddie Gray", "Sandra Bland")
shootings <- mdy(c("04/12/2015","7/13/2015"))

# Combine data
top2 <- data.frame(top_topic = max, second_topic = max2, date = ymd(blm_tweets_sum$date))

# Plot
blm_plot <- ggplot(top2, aes(x=date, y=top_topic, pch="First"))

blm_plot + geom_point(aes(x=date, y=second_topic, pch="Second") ) +theme_bw() +
  ylab("Topic Number") + ggtitle("BLM-Related Tweets from 2014 to 2016 over Topics") + geom_point() + x
  geom_vline(xintercept=as.numeric(shootings[1]), color = "blue", linetype=4) + # Freddie Gray (Topic)
  geom_vline(xintercept=as.numeric(shootings[2]), color = "black", linetype=4)  + # Sandra Bland
  scale_shape_manual(values=c(18, 1), name = "Topic Rank")
```


BLM−Related Tweets from 2014 to 2016 over Topics
Topic Number
15
10
5
2015−01
2015−07
2016−01
2016−07
Topic Rank
First
Second