VE414 Lecture 20

Jing Liu

UM-SJTU Joint Institute

July 25, 2019

- Consider the following problem

Bayesian Network for Text Mining

Suppose we have been given a collection of m documents and we wish to determine how the documents are related, and whether they can be sorted into k categories. i.e. in terms of topic, author or era in which they were written.

- For example, consider a two-topic model of news, with one on "politics" and the other on "business." The most common words for politics might be

  "President", "Congress", and "government",

  while the business topic may be made up of words such as

  "profit", "budget", and "market".

- However, words can be equally important for both topics, i.e. a word like

  "expect".

Q: What is a sensible Bayesian Network model to use?

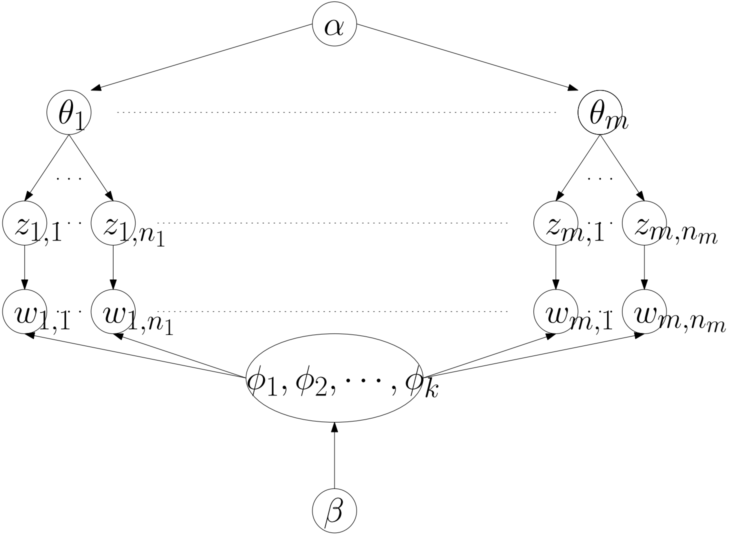
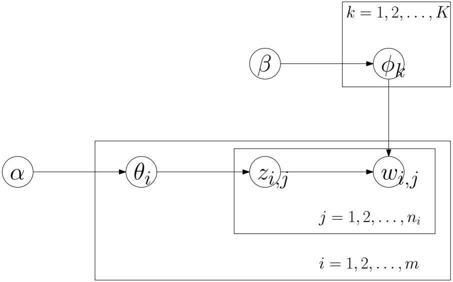


Plate Diagram for LDA



$$
\begin{aligned}
\phi_k &\sim \text{Dirichlet}\,(\beta) \\
\theta_i &\sim \text{Dirichlet}\,(\alpha) \\
z_{i,j} &\sim \text{Multinomial}\,(\theta_i) \\
w_{i,j} &\sim \text{Multinomial}\left(\phi_{z_{i,j}}\right)
\end{aligned}
$$

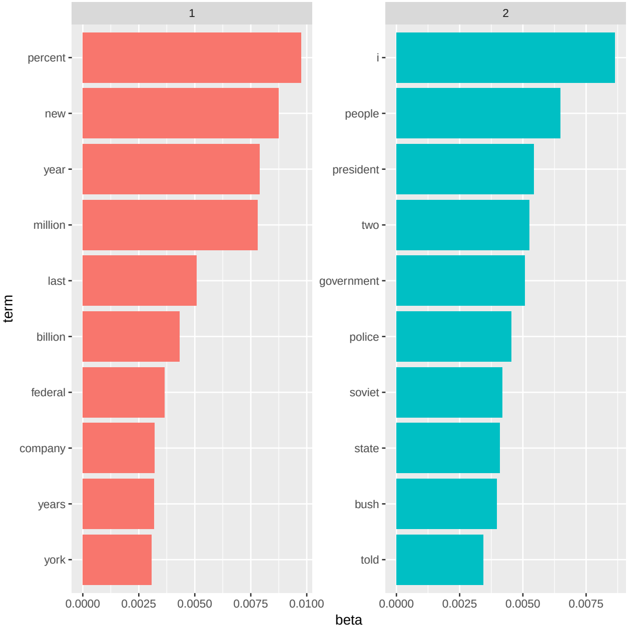
Q: What is the joint posterior distribution?

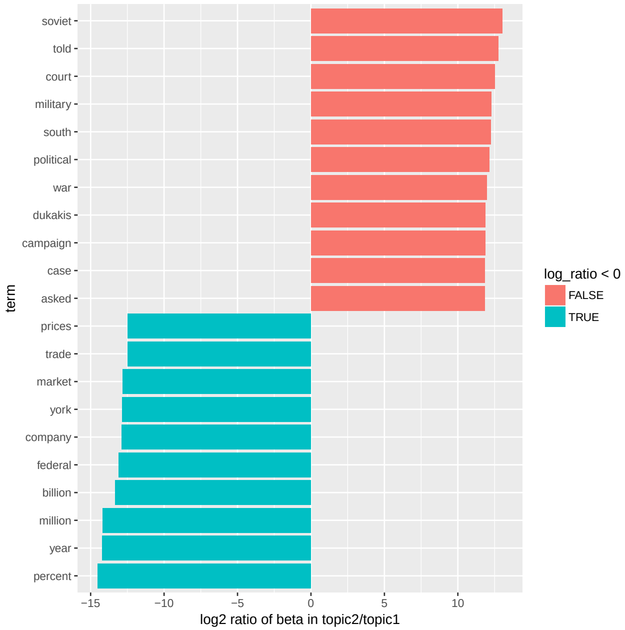
- beta's here are components of ϕ_k

```
> ap_lda = LDA(AssociatedPress, k = 2,
+              method = "Gibbs",
+              control = list(seed = 1234))
>
> ap_topics = tidy(ap_lda, matrix = "beta")
> ap_topics
```

```
# A tibble: 20,946 x 3
   topic term              beta
   <int> <chr>            <dbl>
 1     1 aaron      0.000000501
 2     2 aaron      0.0000424
 3     1 abandon    0.000000501
 4     2 abandon    0.0000634
 5     1 abandoned  0.0000255
 6     2 abandoned  0.000147
 7     1 abandoning 0.000000501
 8     2 abandoning 0.0000256
 9     1 abbott     0.0000406
10     2 abbott     0.000000420
# with 20,936 more rows
```

- It is often interesting to see terms that are most common within each topic.


1
2
percent
new
year
million
last
billion
federal
company
years
york
i
people
president
two
government
police
soviet
state
bush
told
term
beta
0.0000
0.0025
0.0050
0.0075
0.0100


soviet
told
court
military
south
political
war
dukakis
campaign
case
asked
prices
trade
market
york
company
federal
billion
million
year
percent
term
−15
−10
−5
0
5
10
log2 ratio of beta in topic2/topic1
log_ratio < 0
FALSE
TRUE

- gamma's here are components of θ_i

```
> ap_documents = tidy(ap_lda, matrix = "gamma")
> ap_documents
```

```
# A tibble: 4,492 x 3
   document topic gamma
      <int> <int> <dbl>
 1        1     1 0.192
 2        2     1 0.411
 3        3     1 0.272
 4        4     1 0.405
 5        5     1 0.457
 6        6     1 0.240
 7        7     1 0.641
 8        8     1 0.362
 9        9     1 0.526
10       10     1 0.301
# with 4,482 more rows
```

```
> ap_documents[ap_documents$document==1,]
```

```
# A tibble: 2 x 3
  document topic gamma
     <int> <int> <dbl>
1        1     1 0.192
2        1     2 0.808
```

```
> tidy(AssociatedPress) %>%
+    filter(document == 1) %>%
+    arrange(desc(count))
```

```
# A tibble: 186 x 3
   document term        count
      <int> <chr>       <dbl>
 1        1 police          7
 2        1 school          7
 3        1 teacher         7
 4        1 shot            5
 5        1 students        5
 6        1 boy             4
 7        1 boys            4
 8        1 classroom       4
 9        1 gun             3
10        1 guns            3
11        1 jammed          3
12        1 marino          3
13        1 yearold         3
14        1 adult           2
15        1 apparently      2
# with 171 more rows
```

- This looks like a political news instead of business news, which means LDA has correctly sorted the document in this case.

Q: Anyone uses twitter? Anyone follow Donald Trump's twitter account?

- There was a claim about his tweets circulating in 2016:

Todd Vaziri

> *Every non-hyperbolic tweet is from iPhone.*
>
> *i.e. his staff handled*
>
> *Every hyperbolic tweet is from Android.*
>
> *i.e. really from him*

```
> library(twitteR)
> # access to the twitter API.
> consumer_key = "your_consumer_key"
> consumer_secret = "your_consumer_secret"
> access_token = "your_access_token"
> access_secret = "your_access_secret"
> setup_twitter_oauth(consumer_key, consumer_secret,
+                     access_token, access_secret)
```

```r
> library(dplyr)
>
> trump_tweets =
+   userTimeline("realDonaldTrump", n = 5)
> class(trump_tweets)
```

```
[1] "list"
```

```r
> (trump_tweets_tb =
+   as_tibble(
+     purrr::map_dfr(trump_tweets, as.data.frame)))
```

```
# A tibble: 5 x 16
  text          favorited favoriteCount replyToSN created             truncated
  <chr>         <lgl>             <dbl> <lgl>     <dttm>              <lgl>
1 Secretary Po? FALSE            53953. NA        2018-05-09 12:35:51 TRUE
2 I am pleased? FALSE            77505. NA        2018-05-09 12:30:56 TRUE
3 Congratulati? FALSE            32838. NA        2018-05-09 12:00:30 TRUE
4 Candace Owen? FALSE            62859. NA        2018-05-09 11:48:17 TRUE
5 The Fake New? FALSE            56652. NA        2018-05-09 11:38:45 TRUE
# ... with 10 more variables: replyToSID <lgl>, id <chr>, replyToUID <lgl>,
#   statusSource <chr>, screenName <chr>, retweetCount <dbl>, isRetweet <lgl>,
#   retweeted <lgl>, longitude <lgl>, latitude <lgl>
```

```r
> rm(list = ls())
> # Tweets in 2016
> load("~/Desktop/trump_tweets.rda")
> # load R object that was saved
> trump_tweets_tb
```

```
# A tibble: 1,512 x 16
  text           favorited favoriteCount replyToSN created             truncated
  <chr>          <lgl>             <dbl> <chr>     <dttm>              <lgl>
1 My economic ? FALSE              9214. NA        2016-08-08 15:20:44 FALSE
2 Join me in F? FALSE              6981. NA        2016-08-08 13:28:20 FALSE
# ... with 1,510 more rows, and 10 more variables: replyToSID <lgl>, id <chr>,
#   replyToUID <chr>, statusSource <chr>, screenName <chr>, retweetCount <dbl>,
#   isRetweet <lgl>, retweeted <lgl>, longitude <chr>, latitude <chr>
```

```r
> (sel_tb =
+      select(trump_tweets_tb,
+             id, statusSource, text, created))
```

```
# A tibble: 1,512 x 4
  id                 statusSource          text              created
  <chr>              <chr>                 <chr>             <dttm>
1 762669882571980801 "<a href=\"http://tw? My economic pol? 2016-08-08 15:20:44
2 762641595439190016 "<a href=\"http://tw? Join me in Faye? 2016-08-08 13:28:20
# ... with 1,510 more rows
```

```
> head(sel_tb$statusSource)
```

```
[1] "<a href=\".../android\" rel=\"nofollow\">Twitter for Android</a>"
[2] "<a href=\".../iphone\" rel=\"nofollow\">Twitter for iPhone</a>"
[3] "<a href=\".../iphone\" rel=\"nofollow\">Twitter for iPhone</a>"
[4] "<a href=\".../android\" rel=\"nofollow\">Twitter for Android</a>"
[5] "<a href=\".../android\" rel=\"nofollow\">Twitter for Android</a>"
[6] "<a href=\".../android\" rel=\"nofollow\">Twitter for Android</a>"
```

```
> (ext_tb =
+       extract(sel_tb,
+               col = statusSource,
+               into = "source",
+               regex = "Twitter for (.*?)<"))
```

```
# A tibble: 1,512 x 4
  id                 source  text                         created
  <chr>              <chr>   <chr>                        <dttm>
1 762669882571980801 Android My economic policy speech wil? 2016-08-08 15:20:44
2 762641595439190016 iPhone  Join me in Fayetteville, Nort? 2016-08-08 13:28:20
# ... with 1,510 more rows
```

```
> unique(ext_tb$source)
```

```
[1] "Android" "iPhone"  NA          "iPad"
```

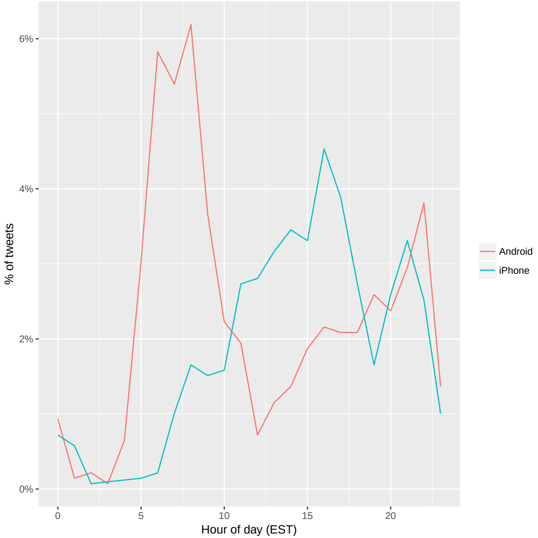
```
> trump_tidy_tb =
+   filter(ext_tb,
+            source %in% c("iPhone", "Android"))
>
> by_source = group_by(trump_tidy_tb, source)
> summarise(by_source, freq = n())
```

```
# A tibble: 2 x 2
  source   freq
  <chr>   <int>
1 Android   762
2 iPhone    628
```

- In practice, you would do those steps in one chunk using a compact syntax

```
> trump_tidy_tb = trump_tweets_tb %>%
+   select(id, statusSource, text, created) %>%
+   extract(statusSource, "source",
+           "Twitter for (.*?)<") %>%
+   filter(source %in% c("iPhone", "Android"))
```

- Investigating the relation between % of tweets by source and time, we have



```r
> trump_tidy_tb %>%
+   count(source, hour =
+           lubridate::hour(
+             lubridate::with_tz(
+               created, "EST"))) %>%
+   mutate(percent = n / sum(n)) %>%
+   ggplot(
+     aes(hour,
+         percent, color = source)
+     ) +
+   geom_line() +
+   scale_y_continuous(
+     labels = scales::percent_format()
+     ) +
+   labs(x = "Hour of day (EST)",
+        y = "% of tweets",
+        color = "")
```

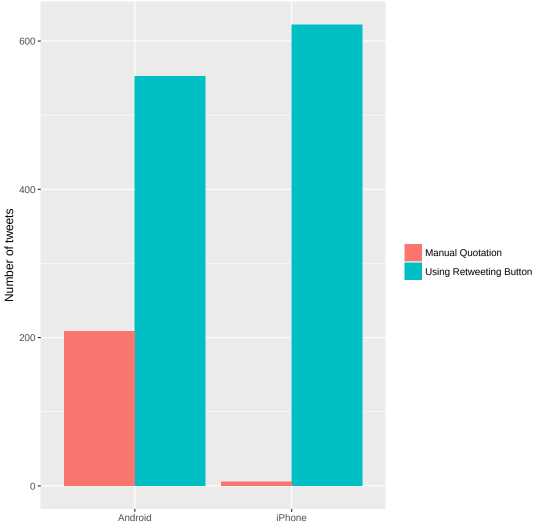
- Another place we can spot a difference is in Trump's tendency of

  "manually retweeting"

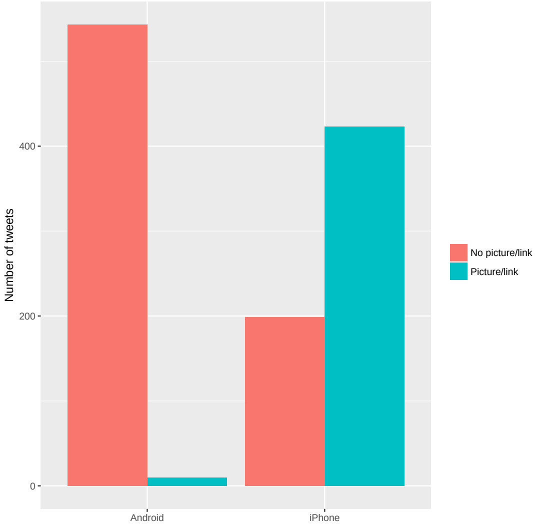
  others by copying-pasting, then putting them under quotation marks.

```
> library(stringr) # Better handling on strings
> tweet_quotation_counts = trump_tidy_tb %>%
+   count(source, quotation =
+           ifelse(str_detect(text, '^"'),
+             "Manual Quotation",
+             "Using Retweeting Button"))
>
> ggplot(tweet_quotation_counts,
+        aes(source, n, fill = quotation)) +
+   geom_bar(stat = "identity",
+            position = "dodge") +
+   labs(x = "", y = "Number of tweets", fill = "")
```

- Almost all of those quoted tweets are posted from the android device.



- Another difference involves sharing links or pictures in tweets.



```r
> tweet_picture_counts = trump_tidy_tb %>%
+   filter(!str_detect(text, '^"')) %>%
+ # we have to remove retweeting cases
+ # that were done manually
+   count(source, picture =
+           ifelse(str_detect(text, "t.co"),
+             "Picture/link", "No picture/link")
+        )
> # twitter uses the domain https://t.co/
> # for all pictures and links, e.g.
> trump_tidy_tb$text[2]
```

```
[1] "Join me in Fayetteville, North Carolina tomorrow evening at 6pm. Tickets now
available at: https://t.co/Z80d4MYIg8"
```

```r
> ggplot(tweet_picture_counts,
+        aes(source, n, fill = picture)) +
+   geom_bar(stat = "identity",
+            position = "dodge") +
+   labs(x = "", y = "Number of tweets", fill = "")
```

Q: What were the most common words in Trump's tweets overall?

```
> library(tidytext) # Good for tidy up strings
> reg = "([^A-Za-z\\d#@']|'(?![A-Za-z\\d#@]))"
> # Separator between words in his tweets

> trump_words = trump_tidy_tb %>%
+   filter(!str_detect(text, '^"')) %>%
+   mutate(text = str_replace_all(
+     text, "https://t.co/[A-Za-z\\d]+|&amp;",
+     "")) %>%
+   # remove all pictures and links
+   unnest_tokens(word, text,
+                 token = "regex",
+                 pattern = reg) %>%
+   # split sentences into words
+   filter(!word %in% stop_words$word,
+          str_detect(word, "[a-z]"))
>   # keep only relevant words
```

```
> trump_words
```

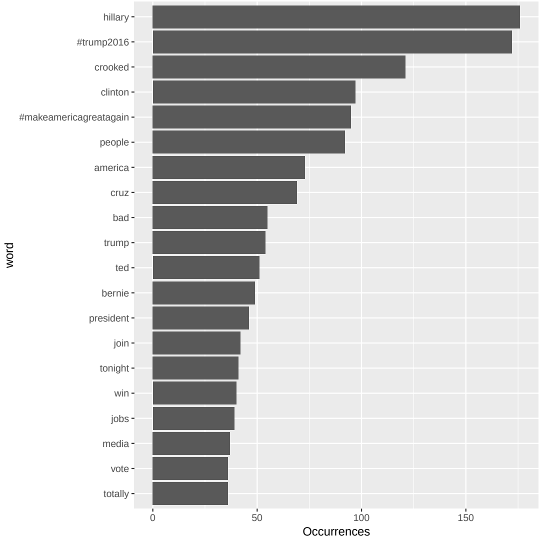
```
# A tibble: 8,753 x 4
  id                 source created             word
  <chr>              <chr>  <dttm>              <chr>
1 676494179216805888 iPhone 2015-12-14 20:09:15 record
2 676494179216805888 iPhone 2015-12-14 20:09:15 health
# ... with 8,751 more rows
```

```
> trump_tidy_tb
```

```
# A tibble: 1,390 x 4
  id                 source  text                           created
  <chr>              <chr>   <chr>                          <dttm>
1 762669882571980801 Android My economic policy speech wil? 2016-08-08 15:20:44
2 762641595439190016 iPhone  Join me in Fayetteville, Nort? 2016-08-08 13:28:20
# ... with 1,388 more rows
```

```
> trump_words %>%
+   count(word, sort = TRUE) %>%
+   head(20) %>%
+   mutate(word = reorder(word, n)) %>%
+   ggplot(aes(word, n)) +
+   geom_bar(stat = "identity") +
+   ylab("Occurrences") +
+   coord_flip()
```

- Recall the data is for 2016, so no surprise there!



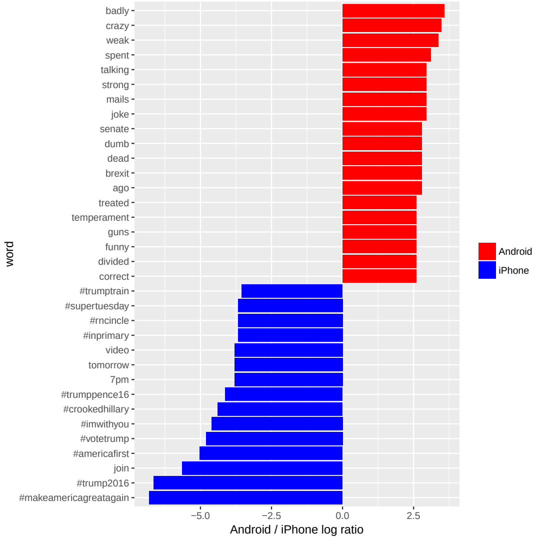
- We sort words according whether it is more likely to coming from an android

$$\log\left(\frac{\frac{\#\text{ Android}+1}{\text{Total Android}+1}}{\frac{\#\text{ iPhone}+1}{\text{Total iPhone}+1}}\right)$$

```
> android_iphone_ratios = trump_words %>%
+   count(word, source) %>%
+   # count the occurence of a word by source
+   spread(source, n, fill = 0) %>%
+   # convert source into two columns
+   mutate_at(c("Android", "iPhone"),
+             funs((. + 1) / sum(. + 1))) %>%
+   # apply a function two both columns
+   mutate(logratio = log2(Android / iPhone)) %>%
+   # create a new column
+   arrange(desc(logratio))
```

```
> android_iphone_ratios %>%
+   group_by(logratio > 0) %>%
+   top_n(15, abs(logratio)) %>%
+   # 15 posistive and 15 negative
+   ungroup() %>%
+   mutate(word = reorder(word, logratio)) %>%
+   # sort word according to logratio
+   ggplot(aes(word,
+              logratio,
+              fill = logratio < 0)) +
+   geom_bar(stat = "identity") +
+   coord_flip() +
+   ylab("Android / iPhone log ratio") +
+   scale_fill_manual(
+     name = "", labels = c("Android", "iPhone"),
+     values = c("red", "blue"))
```

Q: What can we say based on the following plot?



- The NRC emotion lexicon, which comes with `library(tidytext)`,

```
> (nrc = sentiments %>%
+    filter(lexicon == "nrc") %>%
+    select(word, sentiment))
```

```
# A tibble: 13,901 x 2
  word      sentiment
  <chr>     <chr>
1 abacus    trust
2 abandon   fear
3 abandon   negative
4 abandon   sadness
# ... with 1.39e+04 more rows
```

is a way to associate common English words to 2 sentiments:

negative or positive

and 8 emotions:

anger, fear, anticipation, trust, surprise, sadness, joy, and disgust

- To measure the sentiment of the Android and iPhone tweets,

```
> trump_words
```

```
# A tibble: 8,753 x 4
  id                 source created             word
  <chr>              <chr>  <dttm>              <chr>
1 676494179216805888 iPhone 2015-12-14 20:09:15 record
2 676494179216805888 iPhone 2015-12-14 20:09:15 health
3 676494179216805888 iPhone 2015-12-14 20:09:15 #makeamericagreatagain
# ... with 8,750 more rows
```

we divide the number of Trump's words into each of the following categories

```
> unique(nrc$sentiment)
```

```
 [1] "trust"        "fear"         "negative"     "sadness"      "anger"
 [6] "surprise"     "positive"     "disgust"      "joy"          "anticipation"
```

```
> (join_tb = inner_join( # Join the two data sets
+    trump_words, nrc, by = "word"))
```

```
# A tibble: 5,109 x 5
  id                 source created             word     sentiment
  <chr>              <chr>  <dttm>              <chr>    <chr>
1 676509769562251264 iPhone 2015-12-14 21:11:12 accolade anticipation
2 676509769562251264 iPhone 2015-12-14 21:11:12 accolade joy
3 676509769562251264 iPhone 2015-12-14 21:11:12 accolade positive
# ... with 5,106 more rows
```

```r
> # Count the number of sentiment grouped by tweet
> (count_tb = count(join_tb, sentiment, id))
```

```
# A tibble: 3,722 x 3
  sentiment id                     n
  <chr>     <chr>              <int>
1 anger     680503951440121856     1
2 anger     680734915718176768     1
3 anger     685490467329425408     1
# ... with 3,719 more rows
```

```r
> # Add all possible combination of id and sentiment
> (complete_tb =
+      complete(count_tb,
+               sentiment, id, fill = list(n = 0)))
```

```
# A tibble: 8,790 x 3
  sentiment id                     n
  <chr>     <chr>              <dbl>
1 anger     676509769562251264    0.
2 anger     680496083072593920    0.
3 anger     680503951440121856    1.
# ... with 8,787 more rows
```

- This dataset gives the counts of the 10 categories for each tweet.

```r
> # Create a data set on tweets by source
> # One row for each of his tweets
> (sources = trump_words %>%
+     group_by(source) %>%
+     mutate(total = n()) %>%
+     # create a new variable
+     # total number of iPhone/Android
+     ungroup() %>%
+     distinct(id, source, total))
```

```
# A tibble: 1,172 x 3
  id                 source  total
  <chr>              <chr>   <int>
1 676494179216805888 iPhone   3852
2 676509769562251264 iPhone   3852
3 680496083072593920 Android  4901
# ... with 1,169 more rows
```

```r
> length(unique(trump_words$id))
```

```
[1] 1172
```

```r
> # Put source back into the dataset
> (complete_with_source_tb =
+   inner_join(complete_tb, sources))
```

```
Joining, by = "id"
# A tibble: 8,790 x 5
  sentiment id                      n source  total
  <chr>     <chr>               <dbl> <chr>   <int>
1 anger     676509769562251264     0. iPhone   3852
2 anger     680496083072593920     0. Android  4901
3 anger     680503951440121856     1. Android  4901
# ... with 8,787 more rows
```

```r
> # counts by source and sentiment
> words_by_source_sentiment =
+   complete_with_source_tb %>%
+   group_by(source, sentiment, total) %>%
+   summarize(counts = sum(n)) %>%
+   ungroup()
>
> words_tidy_source_sentiment
```

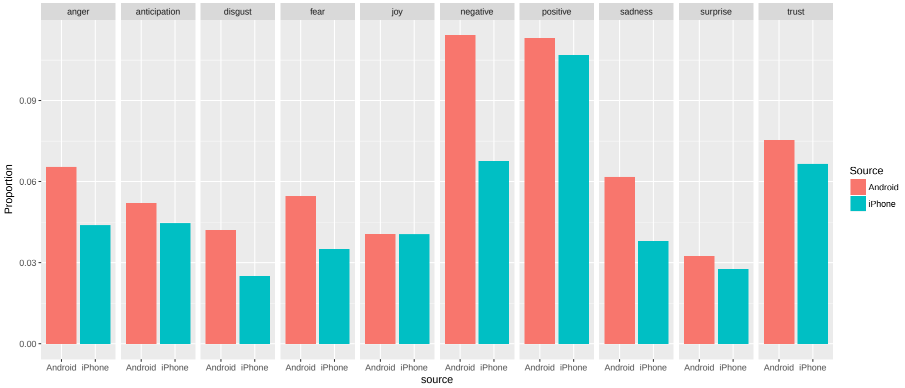
- Everything together in a single chunk

```
> sources = trump_words %>%
+   group_by(source) %>%
+   mutate(total = n()) %>%
+   ungroup() %>%
+   distinct(id, source, total)
>
> words_by_source_sentiment = trump_words %>%
+   inner_join(nrc, by = "word") %>%
+   count(sentiment, id) %>%
+   ungroup() %>%
+   complete(sentiment, id, fill = list(n = 0)) %>%
+   inner_join(sources) %>%
+   group_by(source, sentiment, total) %>%
+   summarize(counts = sum(n)) %>%
+   ungroup()
>
> words_tidy_source_sentiment
```

```
# A tibble: 20 x 4
   source  sentiment    total  counts
   <chr>   <chr>        <int>   <dbl>
 1 Android anger         4901    321.
 2 Android anticipation  4901    256.
 3 Android disgust       4901    207.
 4 Android fear          4901    268.
 5 Android joy           4901    199.
 6 Android negative      4901    560.
 7 Android positive      4901    555.
 8 Android sadness       4901    303.
 9 Android surprise      4901    159.
10 Android trust         4901    369.
11 iPhone  anger         3852    169.
12 iPhone  anticipation  3852    172.
13 iPhone  disgust       3852     97.
14 iPhone  fear          3852    135.
15 iPhone  joy           3852    156.
16 iPhone  negative      3852    260.
17 iPhone  positive      3852    412.
18 iPhone  sadness       3852    147.
19 iPhone  surprise      3852    107.
20 iPhone  trust         3852    257.
```

```
> ggplot(words_by_source_sentiment, aes(
+   source, counts/total, fill = source)) +
+   geom_bar(stat = "identity", position = "dodge") +
+   labs(y = "Proportion", fill = "Source") +
+   facet_grid(~sentiment)
```

- It seems there is a clear difference in the category “negative.

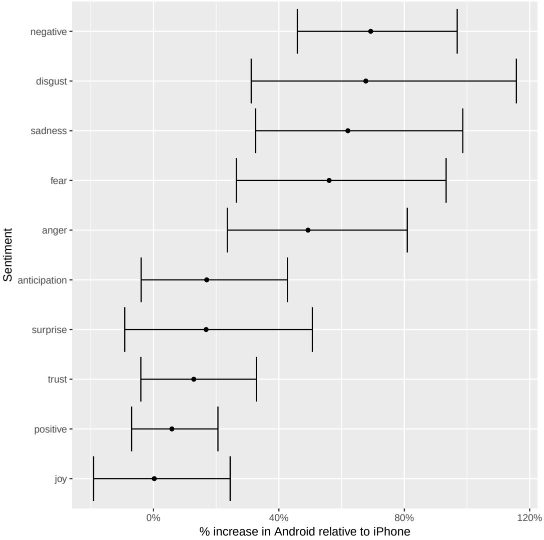


- However, graphical analysis alone is not enough for other categories.
- This is a count data, so let us test by assuming Poisson assumptions.

```
> sentiment_differences
+   words_by_source_sentiment %>%
+   group_by(sentiment) %>%
+   do(broom::tidy(poisson.test(.$counts, .$total)))
>
> sentiment_differences
```

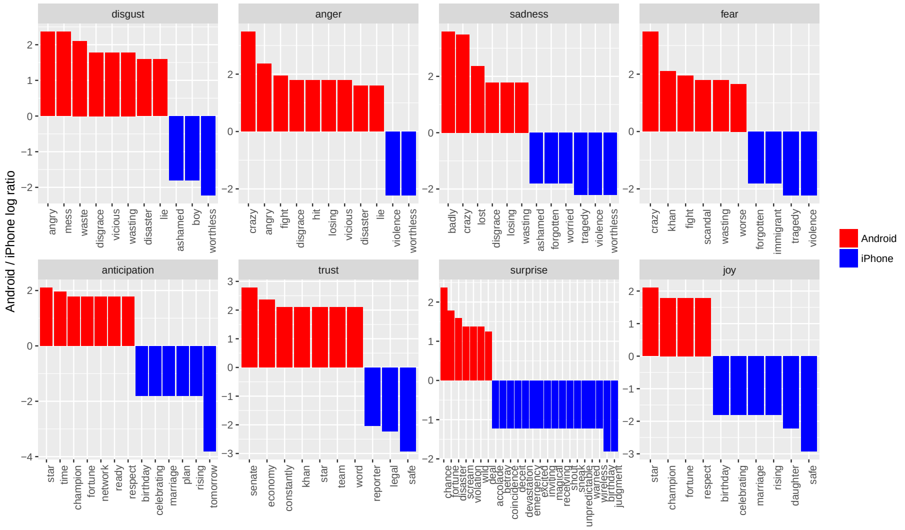
```
# A tibble: 10 x 9
# Groups:   sentiment [10]
   sentiment  estimate statistic  p.value parameter conf.low conf.high method
   <chr>         <dbl>     <dbl>    <dbl>     <dbl>    <dbl>     <dbl> <fct>
 1 anger          1.49      321. 2.19e- 5      274.    1.24       1.81 Compari?
 2 anticipat?     1.17      256. 1.19e- 1      240.    0.960      1.43 Compari?
 3 disgust        1.68      207. 1.78e- 5      170.    1.31       2.16 Compari?
 4 fear           1.56      268. 1.89e- 5      226.    1.26       1.93 Compari?
 5 joy            1.00      199. 1.00e+ 0      199.    0.809      1.24 Compari?
 6 negative       1.69      560. 7.09e-13      459.    1.46       1.97 Compari?
 7 positive       1.06      555. 3.82e- 1      541.    0.930      1.21 Compari?
 8 sadness        1.62      303. 1.15e- 6      252.    1.33       1.99 Compari?
 9 surprise       1.17      159. 2.17e- 1      149.    0.908      1.51 Compari?
10 trust          1.13      369. 1.47e- 1      351.    0.960      1.33 Compari?
# ... with 1 more variable: alternative <fct>
```

- And we can visualise the difference with a 95% confidence interval:



```
> sentiment_differences %>%
+   ungroup() %>%
+   mutate(sentiment =
+            reorder(sentiment, estimate)) %>%
+   mutate_at(c("estimate",
+               "conf.low",
+               "conf.high"),
+             funs(. - 1)) %>%
+   ggplot(aes(estimate, sentiment)) +
+   geom_point() +
+   geom_errorbarh(aes(
+     xmin = conf.low, xmax = conf.high)) +
+   scale_x_continuous(
+     labels = scales::percent_format()) +
+   labs(
+     x = "% increase in Android relative to iPhone",
+     y = "Sentiment")
```

Q: Which words in each category are driving those differences?



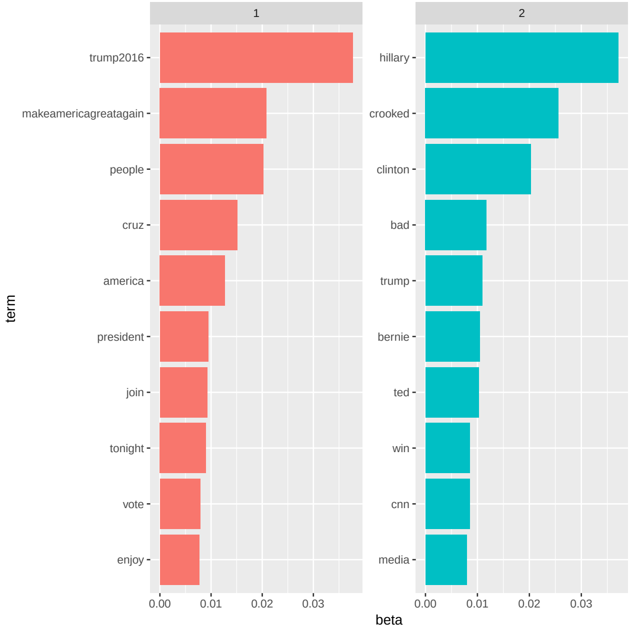
```
> android_iphone_ratios %>%
+   inner_join(nrc, by = "word") %>%
+   filter(!sentiment %in%
+              c("positive", "negative")) %>%
+   mutate(sentiment = reorder(sentiment, -logratio),
+            word = reorder(word, -logratio)) %>%
+   group_by(sentiment) %>%
+   top_n(10, abs(logratio)) %>% ungroup() %>%
+   ggplot(aes(
+     word, logratio, fill = logratio < 0)) +
+   facet_wrap(~ sentiment,
+                scales = "free", nrow = 2) +
+   geom_bar(stat = "identity") +
+   theme(axis.text.x =
+              element_text(angle = 90, hjust = 1)) +
+   labs( x = "",
+          y = "Android / iPhone log ratio") +
+   scale_fill_manual(
+     name = "", values = c("red", "blue"),
+     labels = c("Android", "iPhone"))
```

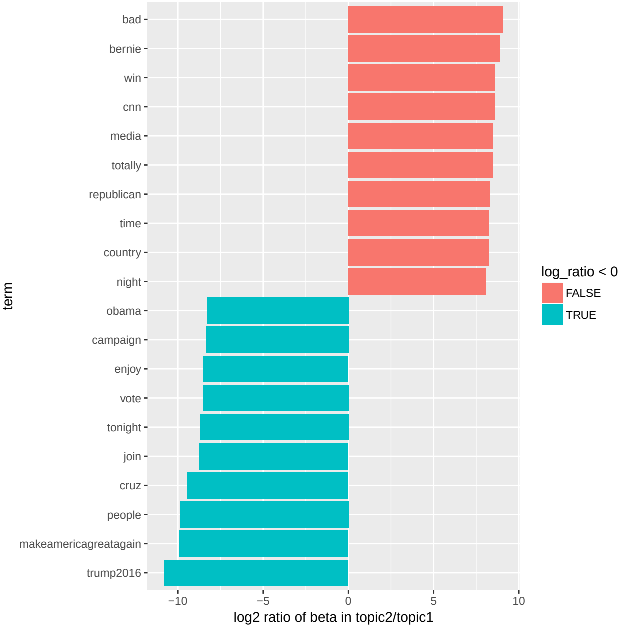
- beta's here are components of ϕ_k

```
> ap_lda = LDA(Trump, k = 2,
+                 method = "Gibbs",
+                 control = list(seed = 1234))
>
> ap_topics = tidy(ap_lda, matrix = "beta")
> ap_topics
```

```
# A tibble: 5,068 x 3
   topic term          beta
   <int> <chr>        <dbl>
 1     1 american 0.0000219
 2     2 american 0.00343
 3     1 donald   0.00177
 4     2 donald   0.0000213
 5     1 great    0.0000219
 6     2 great    0.000235
 7     1 mass     0.0000219
 8     2 mass     0.000235
 9     1 s        0.0000219
10     2 s        0.000874
# with 5,058 more rows
```

- It is often interesting to see terms that are most common within each topic.


1
2
trump2016
makeamericagreatagain
people
cruz
america
president
join
tonight
vote
enjoy
hillary
crooked
clinton
bad
trump
bernie
ted
win
cnn
media
term
beta
0.00
0.01
0.02
0.03



bad
bernie
win
cnn
media
totally
republican
time
country
night
obama
campaign
enjoy
vote
tonight
join
cruz
people
makeamericagreatagain
trump2016
term
−10
−5
0
5
10
log2 ratio of beta in topic2/topic1
log_ratio < 0
FALSE
TRUE

- gamma's here are components of θ_i

```
> ap_documents = tidy(ap_lda, matrix = "gamma")
> ap_documents
```

```
# A tibble: 2,344 x 3
   document            topic gamma
   <chr>               <int> <dbl>
 1 759381869267980288      1 0.441
 2 762106904436961280      1 0.517
 3 758492727583576064      1 0.483
 4 746895602798178304      1 0.491
 5 697182075045179392      1 0.474
 6 701779181986680832      1 0.426
 7 747105049352888320      1 0.483
 8 750865499660091392      1 0.483
 9 761757988516401152      1 0.464
10 754747397700485120      1 0.459
# with 2,334 more rows
```

```
> ap_documents[
+   ap_documents$document==759381869267980288,]
```

```
# A tibble: 2 x 3
  document            topic gamma
  <chr>               <int> <dbl>
1 759381869267980288      1 0.441
2 759381869267980288      2 0.559
```

```
> trump_source_tb[
+   trump_source_tb$id == 759381869267980288, 2]
```

```
# A tibble: 1 x 1
  source
  <chr>
1 iPhone
```

```
> trump_source_tb[
+   trump_source_tb$id == 759381869267980288, 3]
```

```
# A tibble: 1 x 1
  text
  <chr>
1 American homeownership rate in Q2 2016 was 62.9% - lowest rate in 51yrs. WE will br?
```

- It is not clear whether LDA has correctly sorted the document in this case.
- We need more content, or words in each document in order to

```
# A tibble: 2 x 3
  document           topic gamma
  <chr>              <int> <dbl>
1 759381869267980288     1 0.441
2 759381869267980288     2 0.559
```

have a decisive cut like the one we have seen in the news articles case.