# Rock vs Hip-Hop & Pop

*Naga Santhosh Kartheek Karnati*

*8/23/2020*

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

#Loading libraries

```
library(ggplot2)
library(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
```

```
library(rvest)
```

```
## Loading required package: xml2
```

```
library(xml2)
library(selectr)
library(stringr)
library(jsonlite)
library(naniar)
library(tokenizers)
library(tidytext)
library(dbplyr)
```

```
## Warning: package 'dbplyr' was built under R version 3.6.3

##
## Attaching package: 'dbplyr'

## The following objects are masked from 'package:dplyr':
##
##     ident, sql
```

```r
library(tidyr)
library(tibble)
library(data.table)
```

```
##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

#Loading dataframes

```r
pewords <- fread("D:/NEU/Individual Projects/Bands project/Data/popsong_ele_words.csv")
rewords <- fread("D:/NEU/Individual Projects/Bands project/Data/song_lyrics_ele_words.csv")
pwords <- fread("D:/NEU/Individual Projects/Bands project/Data/popsong_words.csv")
rwords <- fread("D:/NEU/Individual Projects/Bands project/Data/song_lyrics_words.csv")

# View(pewords)
# View(rewords)
# View(pwords)
# View(rwords)
```

#Merging dataframes

```r
ewords <- rbind(rewords, pewords, use.names = FALSE)
words <- rbind(rwords, pwords, use.names = FALSE)

# View(ewords)
# View(words)
```

#Save dataframes

```r
#ewords df
write.csv(ewords,
        "D:/NEU/Individual Projects/Bands project/Data/ewords.csv",
          row.names = FALSE)

#words df
write.csv(words,
        "D:/NEU/Individual Projects/Bands project/Data/words.csv",
          row.names = FALSE)
```

#Comparing vocabulary

```r
n_words <- words %>% count(band, word, sort = TRUE)
n_words
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
```

```
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?
```

```
## # A tibble: 39,573 x 3
##    band               word      n
##    <chr>              <chr> <int>
##  1 The Rolling Stones baby    712
##  2 The Rolling Stones yeah    550
##  3 The Rolling Stones love    484
##  4 Queen              love    459
##  5 Chris Brown        girl    433
##  6 Aerosmith          yeah    377
##  7 Queen              yeah    365
##  8 Aerosmith          love    360
##  9 Rihanna            love    345
## 10 AC/DC              rock    330
## # ... with 39,563 more rows
```

```
total_words <- n_words %>% group_by(band) %>% summarise(words = sum(n))
total_words
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?
```

```
## # A tibble: 20 x 2
##    band               words
##    <chr>              <int>
##  1 AC/DC              11255
##  2 Adele               2393
##  3 Aerosmith          17226
##  4 Chris Brown        12000
##  5 Eagles              5224
##  6 Guns n Roses        7611
##  7 John Legend         3142
##  8 Led Zeppelin        5548
##  9 Metallica          11200
## 10 Miley Cyrus         3115
## 11 One Direction       7421
## 12 Pink               10364
## 13 Pink Floyd          8354
## 14 Queen              15877
## 15 Rihanna            11617
## 16 Selena Gomez        7588
## 17 Shawn Mendes         391
## 18 The Rolling Stones 24025
## 19 The Who            12425
## 20 Travis Scott         313
```

```
unique_words <- n_words %>% select(-n) %>% count(band, sort = TRUE)

vocabulary <- total_words %>% inner_join(unique_words) %>% mutate(ratio=n/words) %>% arrange(desc(ratio
```

```
## Joining, by = "band"
```

```
vocabulary
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?

## # A tibble: 20 x 4
##    band               words     n ratio
##    <chr>              <int> <int> <dbl>
##  1 Travis Scott         313   202 0.645
##  2 Shawn Mendes         391   151 0.386
##  3 John Legend         3142  1032 0.328
##  4 Pink Floyd          8354  2681 0.321
##  5 Eagles              5224  1612 0.309
##  6 Miley Cyrus         3115   945 0.303
##  7 Adele               2393   712 0.298
##  8 Led Zeppelin        5548  1519 0.274
##  9 Guns n Roses        7611  2055 0.270
## 10 The Who            12425  3160 0.254
## 11 Metallica          11200  2725 0.243
## 12 Pink               10364  2237 0.216
## 13 Aerosmith          17226  3632 0.211
## 14 AC/DC              11255  2330 0.207
## 15 One Direction       7421  1477 0.199
## 16 Queen              15877  3137 0.198
## 17 Chris Brown        12000  2340 0.195
## 18 Selena Gomez        7588  1426 0.188
## 19 The Rolling Stones 24025  4297 0.179
## 20 Rihanna            11617  1903 0.164
```

```
#Ignore Travis Scott and Shawn mendes since we have less data .
#John legend most
#Rihanna least

#Comparing rock bands vs hip-hop and pop singers
bands <- c("Pink Floyd", "Eagles","Led Zeppelin", "Guns n Roses", "The Who", "Metallica", "Aerosmith",
           "AC/DC", "Queen", "The Rolling Stones")

singers <- c("Travis Scott","Shawn Mendes","John Legend","Miley Cyrus","Adele","Pink","One Direction",
             "Chris Brown","Selena Gomez","Rihanna")

vocabulary %>% filter(band %in% bands) %>% summarise(no_rockwords = sum(words),
               unique_rockwords = sum(n)) %>% mutate(ratio=unique_rockwords/no_rockwords)
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?
```

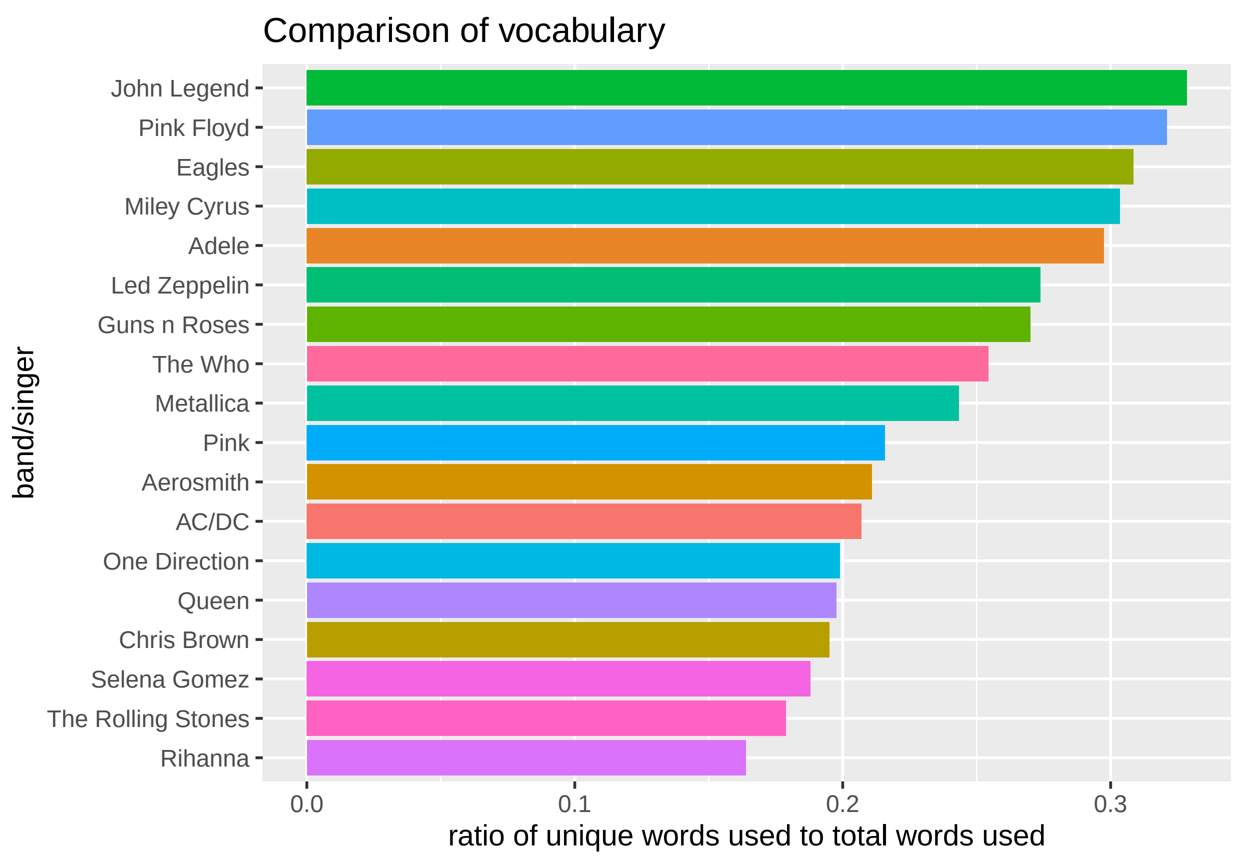
```
## # A tibble: 1 x 3
##   no_rockwords unique_rockwords ratio
##          <int>            <int> <dbl>
## 1       118745            27148 0.229
```

```
vocabulary %>% filter(band %in% singers) %>% summarise(no_hhpwords = sum(words),
               unique_hhpwords = sum(n)) %>% mutate(ratio=unique_hhpwords/no_hhpwords)
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?
```

```
## # A tibble: 1 x 3
##   no_hhpwords unique_hhpwords ratio
##         <int>           <int> <dbl>
## 1       58344           12425 0.213
```

```
#rock bands have better vocabulary than hip-hop and pop artists.

vocabulary %>% filter(!band %in% c("Travis Scott","Shawn Mendes")) %>%
  ggplot(aes(reorder(band, ratio), ratio, fill = band)) + geom_col(show.legend = FALSE) +
  labs(x = "band/singer", y="ratio of unique words used to total words used",
       title = "Comparison of vocabulary") + coord_flip()
```



#Topic modeling after eliminating words with low tf_idf scores

```
library(tokenizers)
library(tidytext)
library(tidyr)
library(tm)
```

```
## Warning: package 'tm' was built under R version 3.6.3

## Loading required package: NLP

##
## Attaching package: 'NLP'

## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library(topicmodels)
```

#Term frequencies

```r
##Calculating the term frequency per each band/singer:
tfs <- n_words %>%
  left_join(total_words)%>%
  mutate(tf = n/words) %>%
  arrange(desc(tf))
```

```
## Joining, by = "band"
```

```r
tfs
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?
```

```
## # A tibble: 39,573 x 5
##    band          word       n words     tf
##    <chr>         <chr> <int> <int>  <dbl>
##  1 Travis Scott  yeah     43   313 0.137
##  2 Adele         love    111  2393 0.0464
##  3 Shawn Mendes  mercy    17   391 0.0435
##  4 Selena Gomez  love    306  7588 0.0403
##  5 John Legend   love    126  3142 0.0401
##  6 Chris Brown   girl    433 12000 0.0361
##  7 Shawn Mendes  baby     13   391 0.0332
##  8 Travis Scott  light    10   313 0.0319
##  9 One Direction na      237  7421 0.0319
## 10 Led Zeppelin  baby    176  5548 0.0317
## # ... with 39,563 more rows
```

#tf_idfs

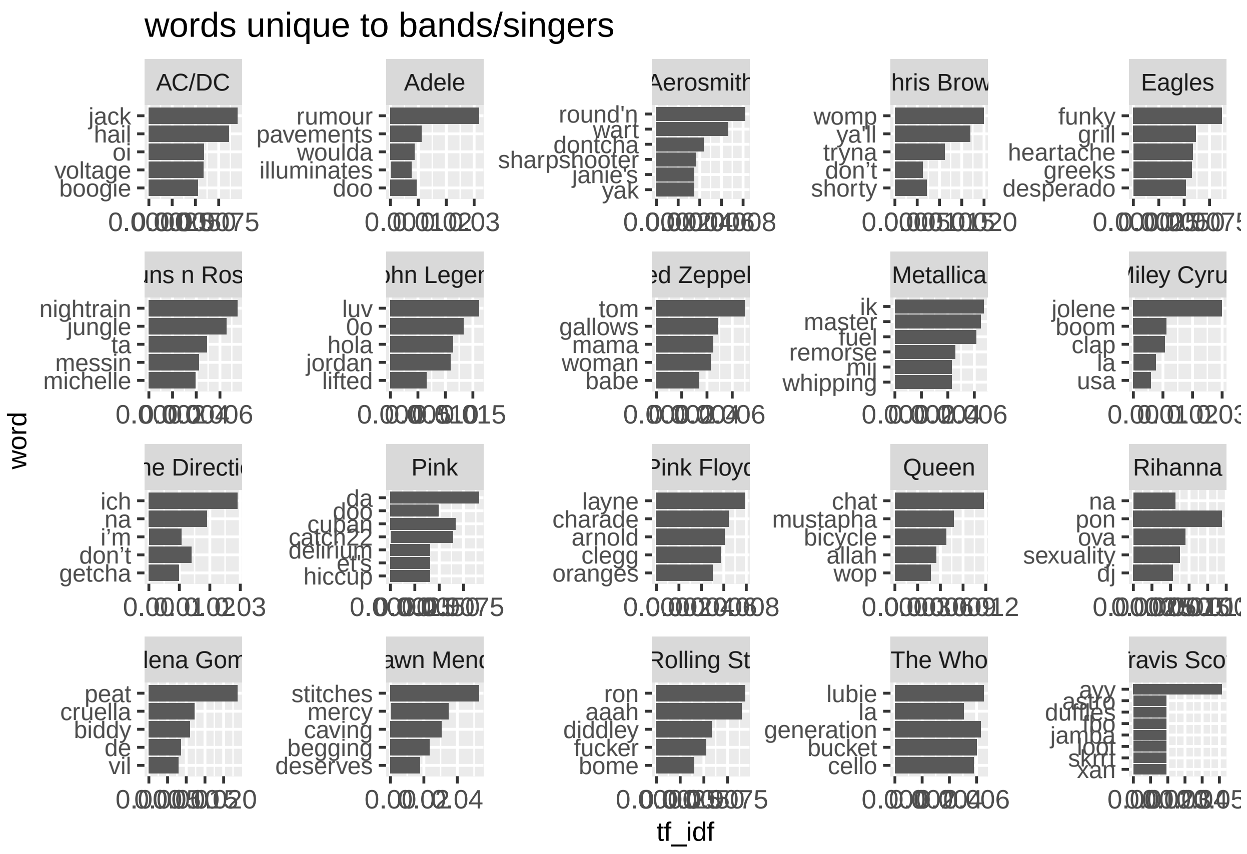
```r
unreq_words <- tibble(word = c("chorus","chris", "53rd", "miley", "pink", "rihanna"))

#tf_idfs
tf_idfs <- words %>%
  anti_join(unreq_words) %>%
  count(band, word, sort=TRUE) %>%
  bind_tf_idf(word, band, n) %>%
  arrange(desc(tf_idf))
```

```
## Joining, by = "word"
```

```r
My_Theme = theme(
  axis.title.x = element_text(size = 10),
  axis.text.x = element_text(size = 10),
  axis.title.y = element_text(size = 10))
```

```
tf_idfs %>%
  arrange(desc(tf_idf)) %>%
  mutate(word = factor(word, levels = rev(unique(word)))) %>%
  group_by(band) %>%
  top_n(5) %>%
  ungroup() %>%
  ggplot(aes(reorder(word,tf_idf), tf_idf), fill = band) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "tf-idf") +
  facet_wrap(~band, ncol = 5,  scales = "free") +
  labs(x="word", y="tf_idf", title = "words unique to bands/singers")+
  coord_flip() + My_Theme
```

```
## Selecting by tf_idf
```



```
tf_idfs_top_5 <- tf_idfs %>%
  arrange(desc(tf_idf)) %>%
  mutate(word = factor(word, levels = rev(unique(word)))) %>%
  group_by(band) %>%
  top_n(5) %>%
  ungroup()
```

```
## Selecting by tf_idf
```

```r
summary(tf_idfs$tf_idf)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0001154 0.0001992 0.0003820 0.0003643 0.0530007
```

```r
#remove the bottom 25% of words from the tf_idf df
```

#Remove words in the bottom 25% of tf_idf scores

```r
tf_idfs_filtered <- tf_idfs %>% filter(tf_idf > 0.0001154) #%>% select(-n,-tf,-idf,-tf_idf)
#dim(tf_idfs)

#join words df and tf_idfs_filtered df on composite key (band and word)
words <- words %>% inner_join(tf_idfs_filtered, by = c("band"="band", "word"="word"))

words$genre[words$band %in% bands] <- "Rock"
words$genre[words$band %in% singers] <- "Hip-Hop & Pop"
```

#Words unique to rock bands and hip-hop and pop artists

```r
words %>% unique() %>% group_by(genre) %>% top_n(5, tf_idf) %>% ungroup()
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?
```

```
## # A tibble: 10 x 7
##    band               word         n      tf   idf  tf_idf genre
##    <chr>              <chr>    <int>   <dbl> <dbl>   <dbl> <chr>
##  1 The Rolling Stones ron         75 0.00312 3.00  0.00935 Rock
##  2 The Rolling Stones aaah        72 0.00300 3.00  0.00898 Rock
##  3 Eagles             funky       24 0.00460 1.90  0.00872 Rock
##  4 AC/DC              jack       179 0.0159  0.598 0.00951 Rock
##  5 Queen              chat        62 0.00391 3.00  0.0117  Rock
##  6 Adele              rumour      33 0.0138  2.30  0.0318  Hip-Hop & Pop
##  7 Travis Scott       ayy          7 0.0224  2.30  0.0515  Hip-Hop & Pop
##  8 Shawn Mendes       caving       4 0.0102  3.00  0.0306  Hip-Hop & Pop
##  9 Shawn Mendes       mercy       17 0.0435  0.799 0.0347  Hip-Hop & Pop
## 10 Shawn Mendes       stitches     9 0.0230  2.30  0.0530  Hip-Hop & Pop
```

#Topic modeling with each band/singer as a document

```r
words2 <- words %>% select(band, word)
```

#DTM with each band/singer as a document

```
bandsinger_dtm <- words2 %>%
  count(band, word) %>%
  cast_dtm(document = band, term = word, value = n)

bandsinger_dtm
```

```
## <<DocumentTermMatrix (documents: 20, terms: 14363)>>
## Non-/sparse entries: 29738/257522
## Sparsity           : 90%
## Maximal term length: 99
## Weighting          : term frequency (tf)
```

```
lda1 <- LDA(bandsinger_dtm, k = 20, control = list(seed = 5))
lda1
```

```
## A LDA_VEM topic model with 20 topics.
```

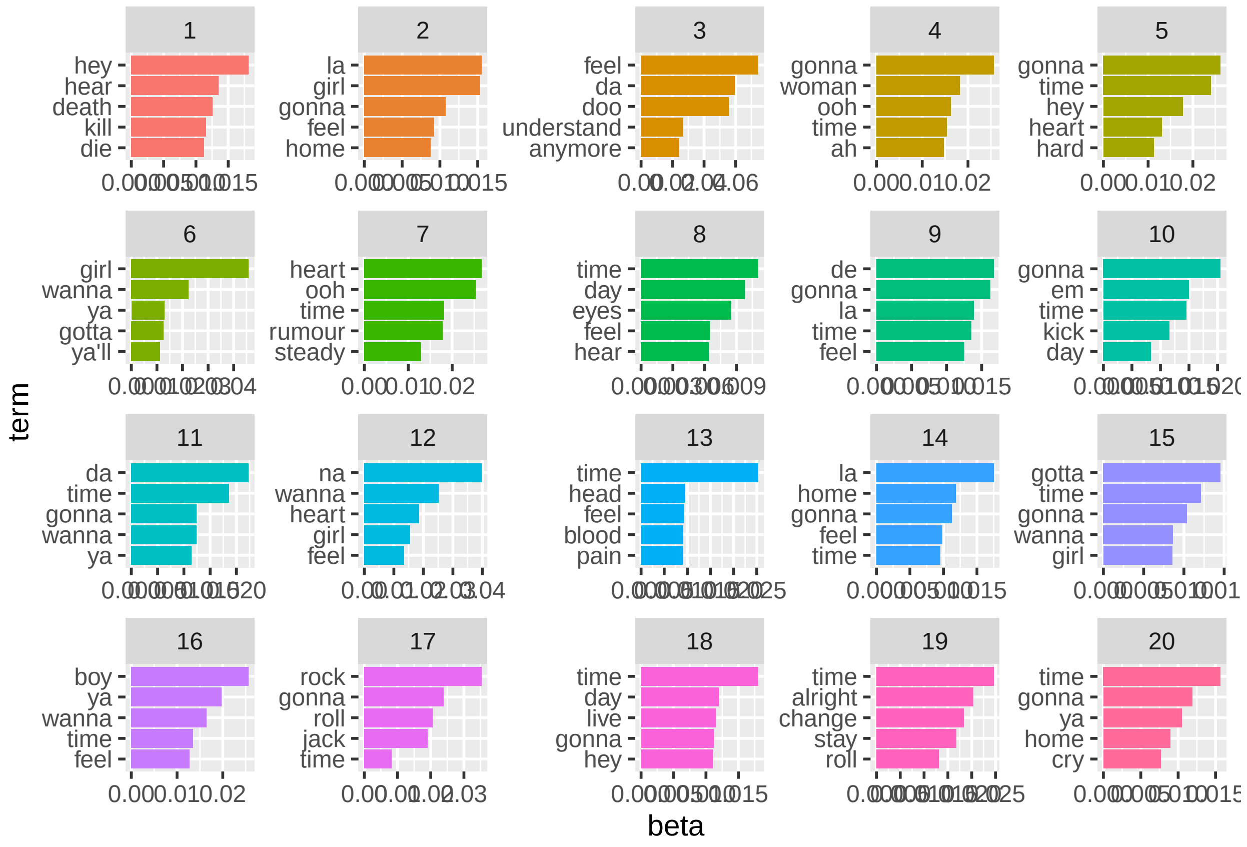
```
topics1 <- tidy(lda1, matrix = "beta")
topics1
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?
```

```
## # A tibble: 287,260 x 3
##    topic term       beta
##    <int> <chr>     <dbl>
##  1     1 11x   1.03e-240
##  2     2 11x   5.19e-242
##  3     3 11x   2.34e-238
##  4     4 11x   1.21e-240
##  5     5 11x   2.20e-241
##  6     6 11x   6.60e-242
##  7     7 11x   4.75e-239
##  8     8 11x   2.13e-241
##  9     9 11x   8.32e-242
## 10    10 11x   1.10e-240
## # ... with 287,250 more rows
```

```
topterms1 <- topics1 %>%
  group_by(topic) %>%
  top_n(5, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

topterms1 %>%
  mutate(term = reorder_within(term, beta, topic)) %>%
```

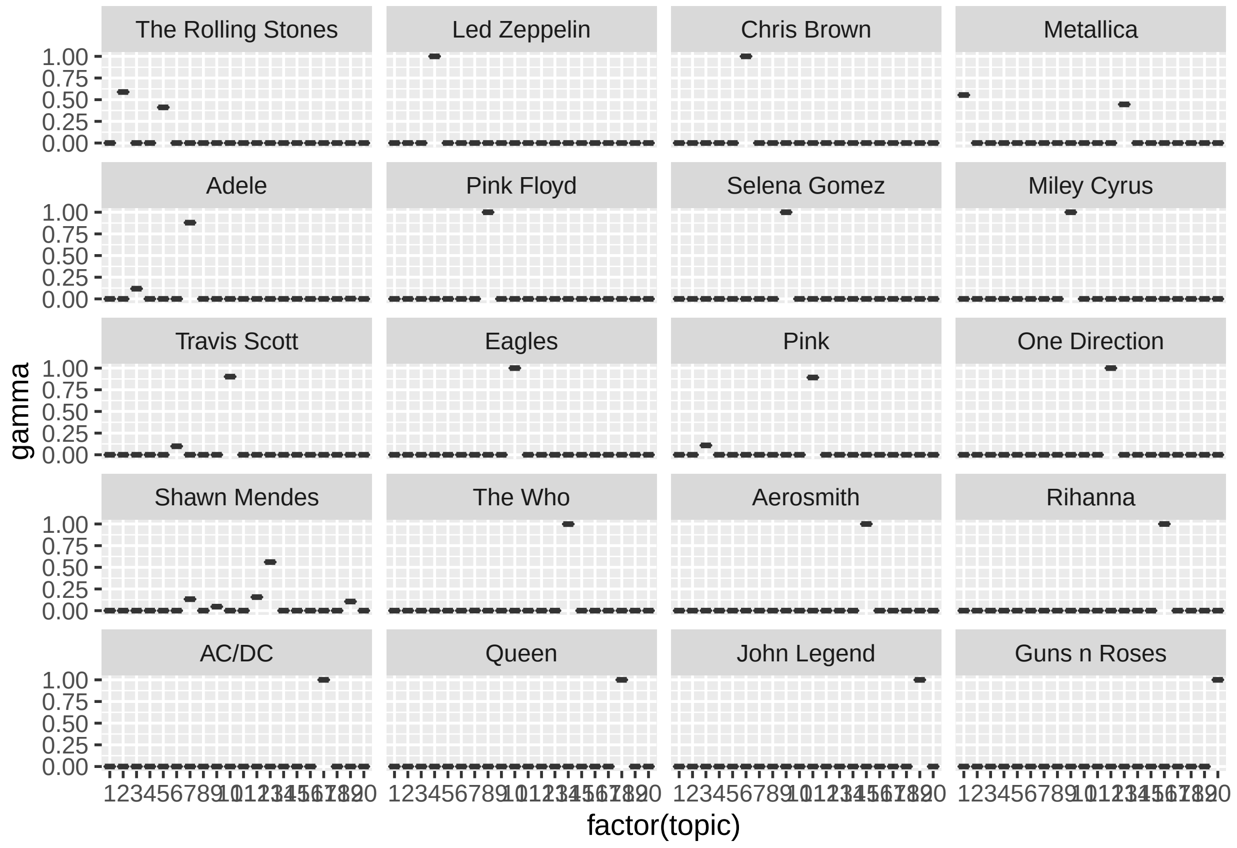
```r
ggplot(aes(term, beta, fill = factor(topic))) +
geom_col(show.legend = FALSE) +
facet_wrap(~ topic, scales = "free") +
coord_flip() +
scale_x_reordered()
```



```r
#which topics are associated with each singer (a document)?
gamma1 <- tidy(lda1, matrix = "gamma")
gamma1
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?

## # A tibble: 400 x 3
##    document     topic       gamma
##    <chr>        <int>       <dbl>
##  1 AC/DC            1 0.000000906
##  2 Adele            1 0.00000412
##  3 Aerosmith        1 0.000000642
##   4 Chris Brown      1 0.000000898
##   5 Eagles           1 0.00000191
##   6 Guns n Roses     1 0.00000133
##   7 John Legend      1 0.00000333
##   8 Led Zeppelin     1 0.00000185
##   9 Metallica        1 0.554
##  10 Miley Cyrus      1 0.00000312
## # ... with 390 more rows
```

```
gamma1 %>%
  mutate(document = reorder(document, gamma * topic)) %>%
  ggplot(aes(factor(topic), gamma)) +
  geom_boxplot() +
  facet_wrap(~ document, ncol = 4)
```



```
#The rolling stones associated with topics 2 and 5
#Metallica associated with topcis 1 and 13
#Adele associated with topcis 7 and 3
#Travis scott associated with topics 10 and 6
#Pink associated with topics 11 and 3
#Shawn mendes associated with topics 13, 12, 19, 12, 7 and 9
```

#band/singer classifications

```r
bandsinger_classifications <- gamma1 %>%
  group_by(document) %>%
  top_n(1, gamma) %>%
  ungroup()

bandsinger_classifications
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?
```

```
## # A tibble: 20 x 3
##    document           topic gamma
##    <chr>              <int> <dbl>
##  1 Metallica              1 0.554
##  2 The Rolling Stones     2 0.589
##  3 Led Zeppelin           4 1.00
##  4 Chris Brown            6 1.00
##  5 Adele                  7 0.880
##  6 Pink Floyd             8 1.00
##  7 Miley Cyrus            9 1.00
##  8 Selena Gomez           9 1.00
##  9 Eagles                10 1.00
## 10 Travis Scott          10 0.901
## 11 Pink                  11 0.892
## 12 One Direction         12 1.00
## 13 Shawn Mendes          13 0.560
## 14 The Who               14 0.999
## 15 Aerosmith             15 1.00
## 16 Rihanna               16 1.00
## 17 AC/DC                 17 1.00
## 18 Queen                 18 1.00
## 19 John Legend           19 1.00
## 20 Guns n Roses          20 1.00
```

#assigning words in each document to a topic

```r
bandsinger_assignments <- augment(lda1, data = bandsinger_dtm)
bandsinger_assignments
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?
## # A tibble: 29,738 x 4
##    document           term  count .topic
##    <chr>              <chr> <dbl>  <dbl>
##  1 AC/DC              11x       1     17
##  2 AC/DC              15        1     17
##  3 Adele              15        1      7
##  4 Chris Brown        15        1      6
##  5 Guns n Roses       15        1     20
##  6 AC/DC              24        1     17
##  7 Miley Cyrus        24        1      9
##  8 Selena Gomez       24        2      9
##  9 The Rolling Stones 24        2      2
## 10 AC/DC              3.00      1     17
## # ... with 29,728 more rows
```

```
bandsinger_assignments <- bandsinger_assignments %>%
  inner_join(bandsinger_classifications, by = c(".topic" = "topic"))
bandsinger_assignments
```

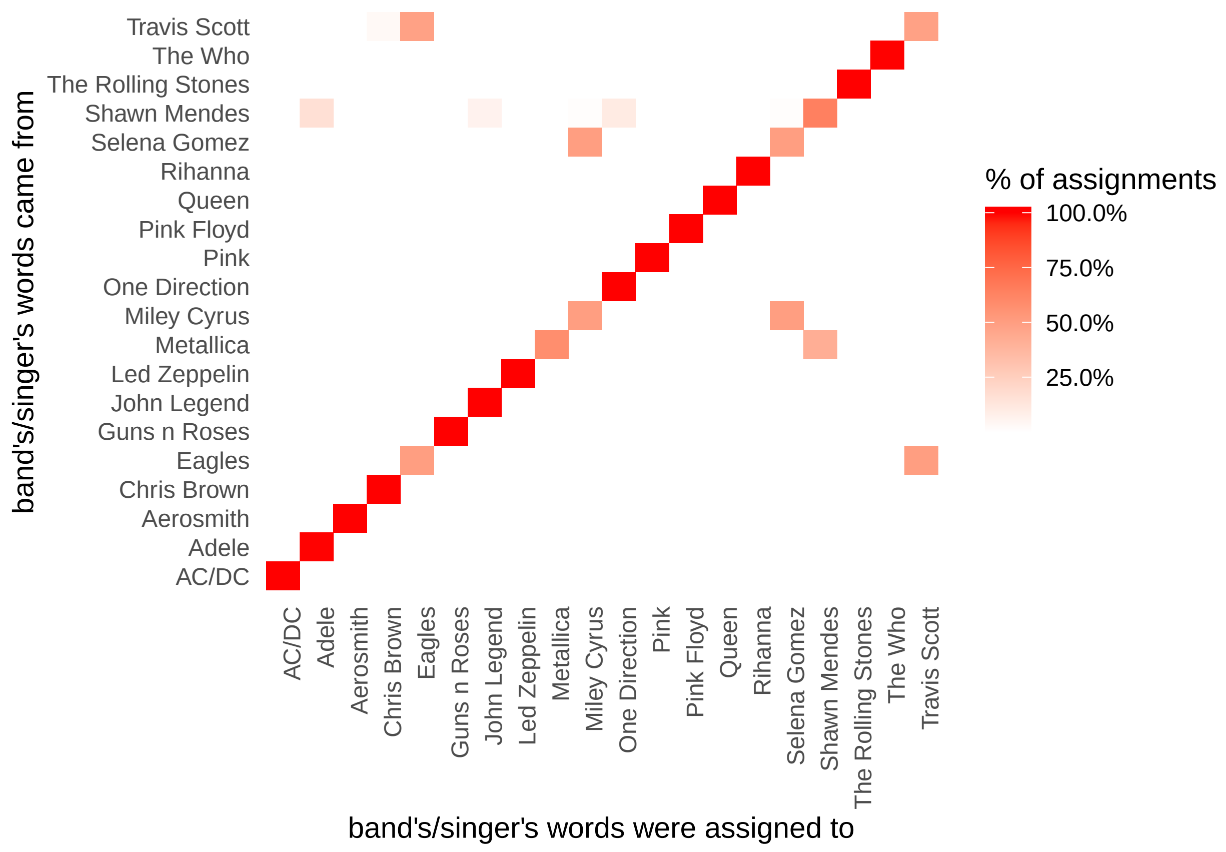
```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?
```

```
## # A tibble: 32,317 x 6
##    document.x   term  count .topic document.y   gamma
##    <chr>        <chr> <dbl>  <dbl> <chr>        <dbl>
##  1 AC/DC        11x       1     17 AC/DC        1.00
##  2 AC/DC        15        1     17 AC/DC        1.00
##  3 Adele        15        1      7 Adele        0.880
##  4 Chris Brown  15        1      6 Chris Brown  1.00
##  5 Guns n Roses 15        1     20 Guns n Roses 1.00
##  6 AC/DC        24        1     17 AC/DC        1.00
##  7 Miley Cyrus  24        1      9 Miley Cyrus  1.00
##  8 Miley Cyrus  24        1      9 Selena Gomez 1.00
##  9 Selena Gomez 24        2      9 Miley Cyrus  1.00
## 10 Selena Gomez 24        2      9 Selena Gomez 1.00
## # ... with 32,307 more rows
```

```
#here document.x is the "true" band whereas document.y is the "consensus" band.
```

#confusion matrix for LDA where each band/singer is a document

```
library(scales)

bandsinger_assignments %>%
  count(document.x, document.y, wt = count) %>%
  group_by(document.x) %>%
  mutate(percent = n / sum(n)) %>%
  ggplot(aes(document.y, document.x, fill = percent)) +
  geom_tile() +
  scale_fill_gradient2(high = "red", label = percent_format()) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        panel.grid = element_blank()) +
  labs(x = "band's/singer's words were assigned to",
       y = "band's/singer's words came from",
       fill = "% of assignments")
```



```
#The few songs that shawn mendes has can be mistaken as Adele, John Legend, One direction songs
#Travis scott's song can be mistaken as an Eagles song.
#Selena Gomez's songs can be mistaken as Miley cyrus songs and vice versa
```

#topic modeling with a different approach

```
female_artists <- c("Adele", "Pink", "Selena Gomez", "Miley Cyrus", "Rihanna")
male_artists <- c("John Legend", "Travis Scott", "Shawn Mendes", "One Direction", "Chris Brown")

words$genre[words$band %in% male_artists] <- "Hip-Hop/Pop Male"
words$genre[words$band %in% female_artists] <- "Hip-Hop/Pop Female"

words3 <- words %>% select(genre, word)
```

#DTM with each genre as a document

```
genre_dtm <- words3 %>%
  count(genre, word) %>%
  cast_dtm(document = genre, term = word, value = n)

genre_dtm
```

```
## <<DocumentTermMatrix (documents: 3, terms: 14363)>>
## Non-/sparse entries: 19108/23981
## Sparsity           : 56%
## Maximal term length: 99
## Weighting          : term frequency (tf)
```

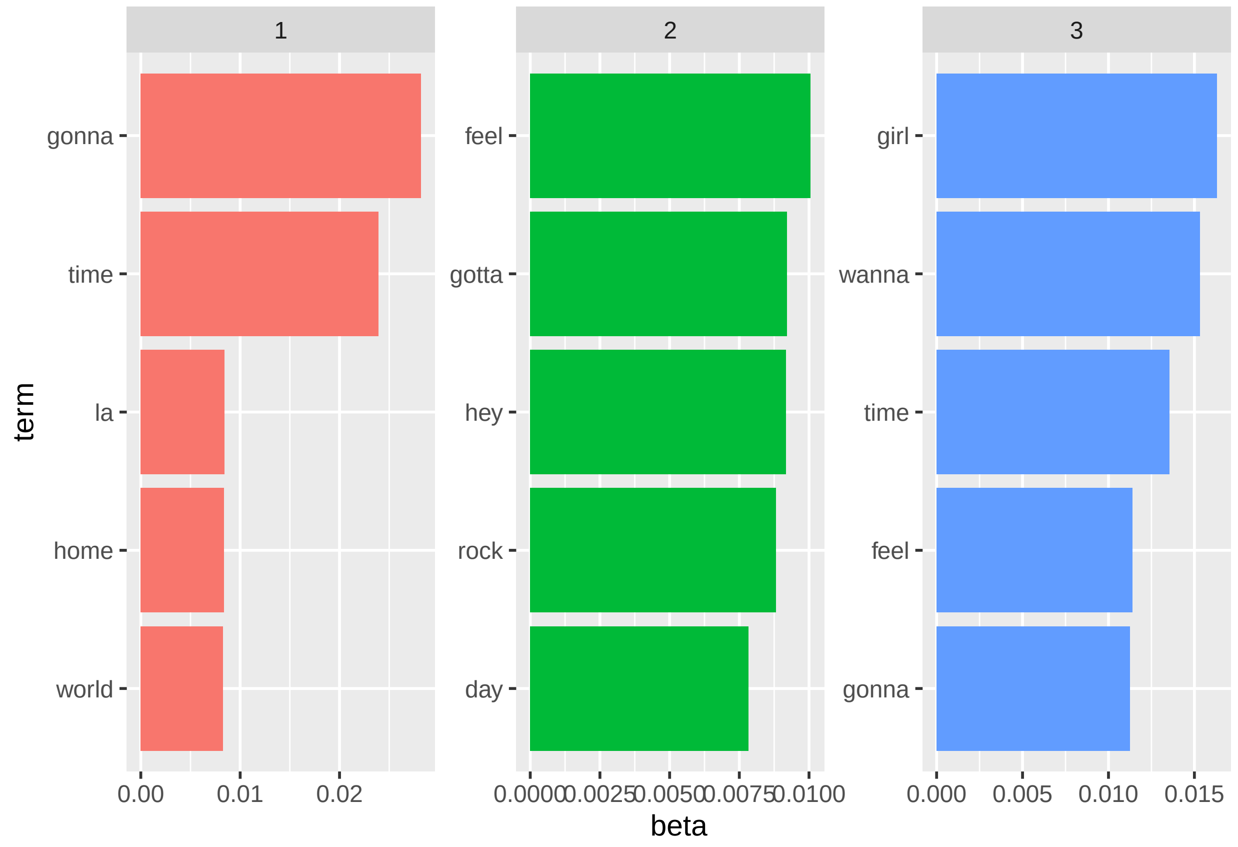
```
lda2 <- LDA(genre_dtm, k = 3, control = list(seed = 10))
lda2
```

```
## A LDA_VEM topic model with 3 topics.
```

```
topics2 <- tidy(lda2, matrix = "beta")
topics2
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?

## # A tibble: 43,089 x 3
##    topic term      beta
##    <int> <chr>    <dbl>
##  1     1 0     1.70e-22
##  2     2 0     1.38e-43
##  3     3 0     8.53e- 5
##  4     1 1     2.25e- 4
##  5     2 1     9.49e- 5
##  6     3 1     2.03e- 3
##  7     1 1'2   6.77e-23
##  8     2 1'2   2.62e-44
##  9     3 1'2   4.26e- 5
## 10     1 10    8.08e-25
## # ... with 43,079 more rows
```

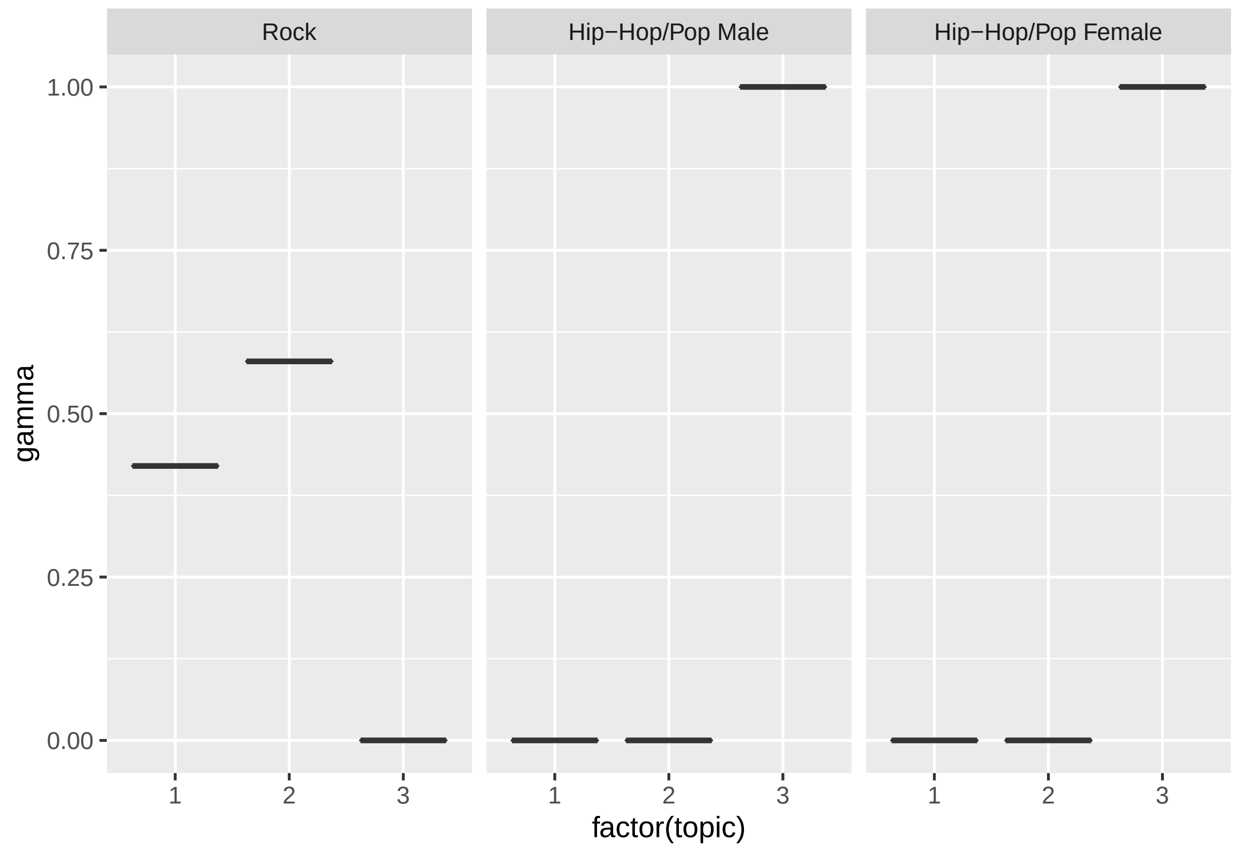
```
topterms2 <- topics2 %>%
  group_by(topic) %>%
  top_n(5, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

topterms2 %>%
  mutate(term = reorder_within(term, beta, topic)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip() +
  scale_x_reordered()
```



```
#which topics are associated with each genre (a document)?
gamma2 <- tidy(lda2, matrix = "gamma")
gamma2
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?

## # A tibble: 9 x 3
##   document           topic      gamma
##   <chr>              <int>      <dbl>
## 1 Hip-Hop/Pop Female     1 0.00000194
## 2 Hip-Hop/Pop Male       1 0.00000297
## 3 Rock                   1 0.420
## 4 Hip-Hop/Pop Female     2 0.00000194
## 5 Hip-Hop/Pop Male       2 0.00000297
## 6 Rock                   2 0.580
## 7 Hip-Hop/Pop Female     3 1.00
## 8 Hip-Hop/Pop Male       3 1.00
## 9 Rock                   3 0.0000325
```

```
gamma2 %>%
  mutate(document = reorder(document, gamma * topic)) %>%
  ggplot(aes(factor(topic), gamma)) +
  geom_boxplot() +
  facet_wrap(~ document, ncol = 4)
```



```
#rock bands are associated with topics 2 and 1
#hip-hop and pop artists (male and female) are associated with topic 3.
```

#genre classifications

```
genre_classifications <- gamma2 %>%
  group_by(document) %>%
  top_n(1, gamma) %>%
  ungroup()

genre_classifications
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?

## # A tibble: 3 x 3
##   document           topic gamma
##   <chr>              <int> <dbl>
## 1 Rock                   2 0.580
## 2 Hip-Hop/Pop Female     3 1.00
## 3 Hip-Hop/Pop Male       3 1.00
```

#assigning words in each document to a topic

```
genre_assignments <- augment(lda2, data = genre_dtm)
genre_assignments
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?

## # A tibble: 19,108 x 4
##    document           term     count .topic
##    <chr>              <chr>    <dbl>  <dbl>
##  1 Hip-Hop/Pop Female 0            4      3
##  2 Hip-Hop/Pop Female 1           66      3
##  3 Hip-Hop/Pop Male   1           29      3
##  4 Rock               1           14      1
##  5 Hip-Hop/Pop Female 1'2          2      3
##  6 Hip-Hop/Pop Female 10           2      3
##  7 Hip-Hop/Pop Male   10           3      3
##  8 Hip-Hop/Pop Female 10,000ft     2      3
##  9 Hip-Hop/Pop Female 10s          1      3
## 10 Hip-Hop/Pop Female 11           2      3
## # ... with 19,098 more rows
```

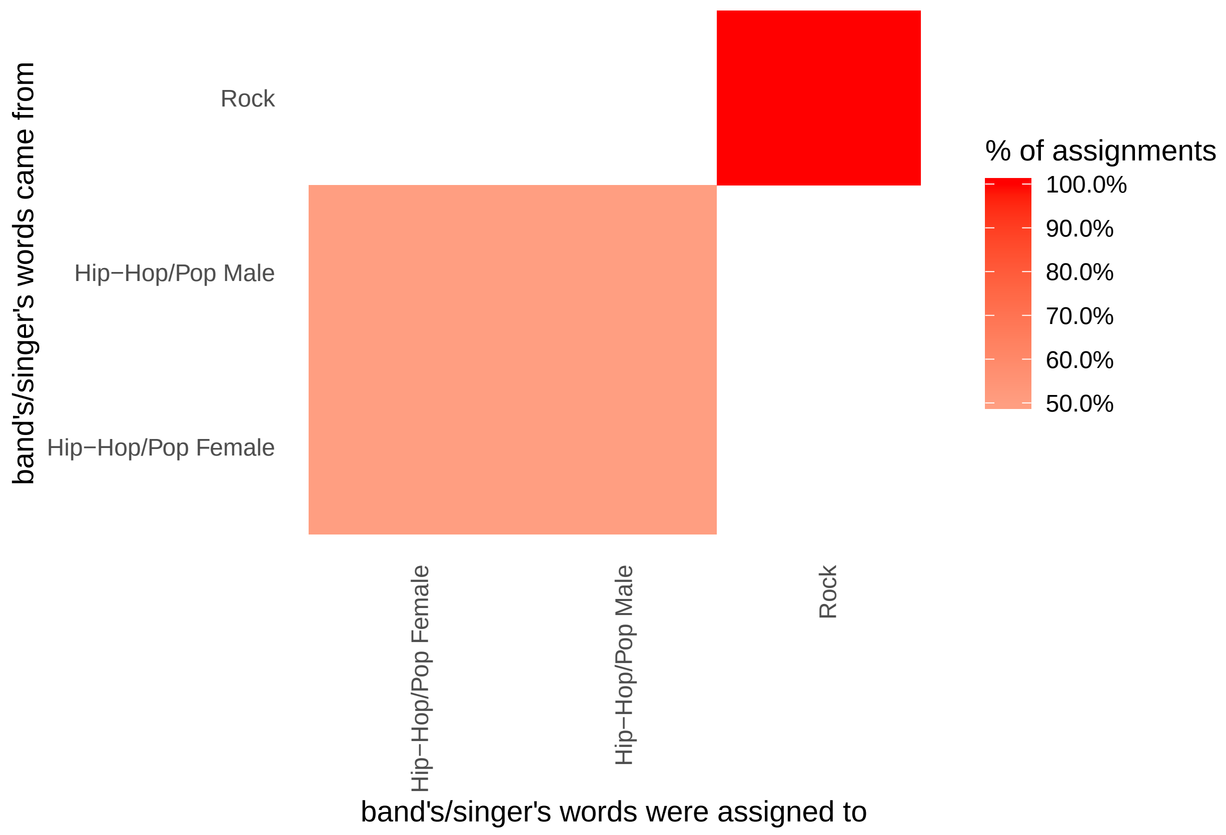
```
genre_assignments <- genre_assignments %>%
  inner_join(genre_classifications, by = c(".topic" = "topic"))
genre_assignments
```

```
## Warning: `...` is not empty.
##
## We detected these problematic arguments:
## * `needs_dots`
##
## These dots only exist to allow future extensions and should be empty.
## Did you misspecify an argument?
## # A tibble: 23,150 x 6
##    document.x         term  count .topic document.y         gamma
##    <chr>              <chr> <dbl>  <dbl> <chr>              <dbl>
##  1 Hip-Hop/Pop Female 0         4      3 Hip-Hop/Pop Female  1.00
##  2 Hip-Hop/Pop Female 0         4      3 Hip-Hop/Pop Male    1.00
##  3 Hip-Hop/Pop Female 1        66      3 Hip-Hop/Pop Female  1.00
##  4 Hip-Hop/Pop Female 1        66      3 Hip-Hop/Pop Male    1.00
##  5 Hip-Hop/Pop Male   1        29      3 Hip-Hop/Pop Female  1.00
##  6 Hip-Hop/Pop Male   1        29      3 Hip-Hop/Pop Male    1.00
##  7 Hip-Hop/Pop Female 1'2       2      3 Hip-Hop/Pop Female  1.00
##  8 Hip-Hop/Pop Female 1'2       2      3 Hip-Hop/Pop Male    1.00
##  9 Hip-Hop/Pop Female 10        2      3 Hip-Hop/Pop Female  1.00
## 10 Hip-Hop/Pop Female 10        2      3 Hip-Hop/Pop Male    1.00
## # ... with 23,140 more rows
```

```
#here document.x is the "true" band whereas document.y is the "consensus" band.
```

#confusion matrix for LDA where each genre is a document

```
library(scales)

genre_assignments %>%
  count(document.x, document.y, wt = count) %>%
  group_by(document.x) %>%
  mutate(percent = n / sum(n)) %>%
  ggplot(aes(document.y, document.x, fill = percent)) +
  geom_tile() +
  scale_fill_gradient2(high = "red", label = percent_format()) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        panel.grid = element_blank()) +
  labs(x = "band's/singer's words were assigned to",
       y = "band's/singer's words came from",
       fill = "% of assignments")
```



```
#male and female artist's lyrics are mistaken as each other's.
#Rock lyrics aren't mistaken for another genre.
```

#Wordclouds

```
library(wordcloud)
```

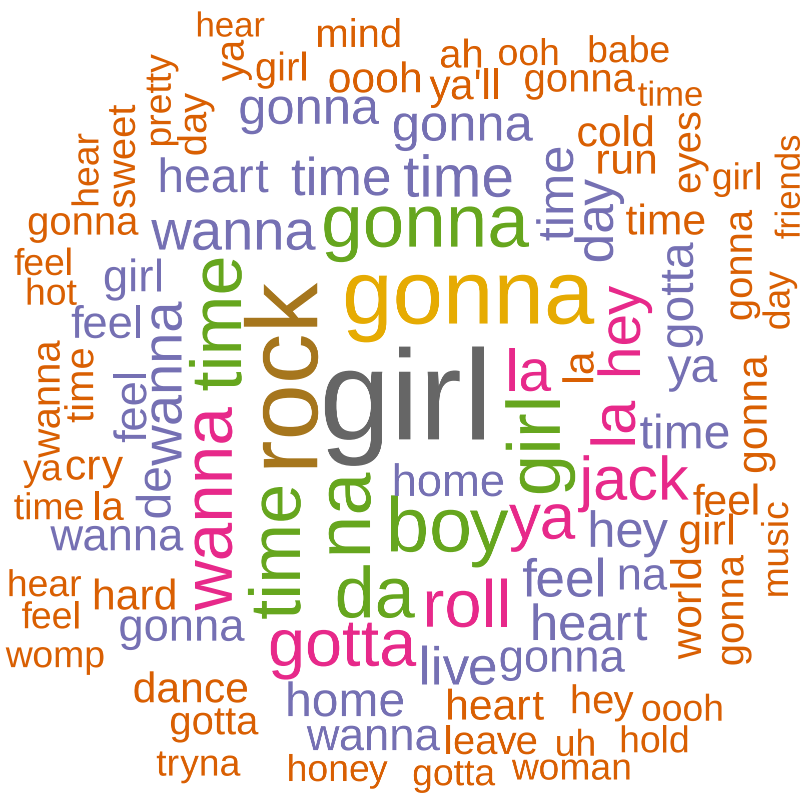
```
## Warning: package 'wordcloud' was built under R version 3.6.3

## Loading required package: RColorBrewer
```

```
library(wordcloud2)
```

```
## Warning: package 'wordcloud2' was built under R version 3.6.3
```

```
library(RColorBrewer)
```

```
wordcloud_df <- words %>% unique() %>% select(word, n)


set.seed(1234)

wordcloud(words = wordcloud_df$word, freq = wordcloud_df$n, min.freq = 1,
          max.words = 100, random.order = FALSE, rot.per = 0.35, colors = brewer.pal(8, "Dark2"))
```



```r
#wordcloud2(data=wordcloud_df, size=1.6, color='random-dark')
```