sentimental_analysis

2023-04-03

```
library(gutenbergr)
```

```
## Warning: package 'gutenbergr' was built under R version 4.2.2
```

```
library(tidytext)
library(janeaustenr)
library(tidyr)
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.2.2
```

```
##
## 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(stringr)
library(ggplot2)
library(scales)
library(SnowballC)
library(here)
```

```
## here() starts at /home/hieu.tran1/penguins-hieutrn1205
```

```
library(textdata)
```

```
## Warning: package 'textdata' was built under R version 4.2.2
```

triebeard, urltools

```
gutenberg_works(title== "The Complete Works of William Shakespeare")
```

```
## # A tibble: 1 x 8
##   gutenberg_id title               author guten~1 langu~2 guten~3 rights has_t~4
##          <int> <chr>               <chr>    <int> <chr>   <chr>   <chr>  <lgl>
## 1          100 The Complete Works~ Shake~      65 en      Plays   Publi~ TRUE
## # ... with abbreviated variable names 1: gutenberg_author_id, 2: language,
## #   3: gutenberg_bookshelf, 4: has_text
```

```
#saveRDS(william, file = "william.rds")
william <- readRDS("william.rds")
```

```
cleaned_william <- william[58:nrow(william), ] |> mutate(reg_ex =
  sub("[A-Z ]*\\.", "", text))
cleaned_william <- cleaned_william |> mutate(reg_ex = sub("(\\[.*?\\])", "", reg_ex))
cleaned_william$reg_ex[grep("Enter", cleaned_william$reg_ex, fixed = TRUE)] <- ""
```

trimws() trim the whitespace between them line 33 rename the value of the chapter 10 to be matched with the actual text

```
chapter <- william[10:53,2]
chapter <- pull(chapter, text) |> trimws()
#chapter[10] <- "THE SECOND PART OF KING HENRY IV"
```

```
cleaned_chapter <- lapply(chapter, grep, cleaned_william$text, fixed = TRUE)
```

```
first_line <- integer()
for (i in 1:length(cleaned_chapter)){
     first_line[[i]] <- cleaned_chapter[[i]][1]
}
```

```
last_line <- Hmisc::Lag(first_line, shift =-1)
last_line <- last_line -1
last_line[44] <- nrow(cleaned_william)
```

```
chapter_n <- data.frame(first_line, last_line, chapter)
```

```
cleaned_william$play <- NA
for (i in 1:nrow(chapter_n)){
  cleaned_william[chapter_n$first_line[i]:chapter_n$last_line[i], 4] <- chapter_n$chapter[i]
}
# Subset Rows by column value
subset_df <- function(name) {
  play <- cleaned_william[cleaned_william$play == name,]
}
```

```
#loop for creating the play column in the text in order to tokenize them
cleaned_william <- cleaned_william |> group_by(play) |> mutate(linenumber = row_number()) |> ungroup()
```

```
#load stop_words and add more into stop_words such as common noun and name
data("stop_words")
new_words <- data.frame(word = c("king", "hamlet", "antony", "richard", "othello", "romeo", "caesar", "
stop_words <- stop_words |> select(-lexicon)
stop_words <- rbind(stop_words, new_words)
```

```
tokenized <- cleaned_william |> unnest_tokens(word, reg_ex) |> anti_join(stop_words) |> mutate(word = s
```

```
## Joining with `by = join_by(word)`
```

```
#Get total words in each play
total_words <- tokenized |>
  group_by(play) |> dplyr::summarize(total = sum(n))

countbyplay <- tokenized
combined_freq <- countbyplay |> left_join(total_words, by = "play")
```

```
#get ifr
getifr <- combined_freq |> mutate(relativefreq = n/total, ifr = log(relativefreq))

tf_idf <- combined_freq |> bind_tf_idf(stem, play, n)
```

```
#Graph
library(forcats)

p_out <- tf_idf |>
  filter(play %in% chapter_n$chapter[1:6]) |>
  group_by(play) |>
  slice_max(tf_idf, n =15) |>
  ungroup() |>
  ggplot(aes(tf_idf, fct_reorder(stem, tf_idf), fill = play)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~play, ncol = 2, scales = "free") +
  labs(x = "tf_idf", y = "NULL")
ggsave("tf.png", plot = p_out, width = 10, height= 15)
```

```
tf_idf_w <- tf_idf |>
  group_by(play) |>
  slice_max(tf_idf, n = 50) |>
  ungroup() |>
  pivot_wider(names_from = play, values_from = tf_idf)
```

```
#Analysis only for Tragedy play
#subset tragedy play

subset_words <- c("TRAGEDY")
tragedy <- subset(chapter, grepl(paste(subset_words, collapse = "|"), chapter))
```

```
#Graph
library(forcats)

p_out <- tf_idf |>
  filter(play %in% tragedy) |>
  group_by(play) |>
  slice_max(tf_idf, n =15) |>
  ungroup() |>
  ggplot(aes(tf_idf, fct_reorder(stem, tf_idf), fill = play)) +
```

```
  geom_col(show.legend = FALSE) +
  facet_wrap(~play, ncol = , scales = "free") +
  labs(x = "tf_idf", y = "NULL")
ggsave("tf.png", plot = p_out, width = 10, height= 15)
```

```
tf_idf_w <- tf_idf |>
  group_by(play) |>
  slice_max(tf_idf, n = 50) |>
  ungroup() |>
  pivot_wider(names_from = play, values_from = tf_idf)
```

```
library(topicmodels)
```

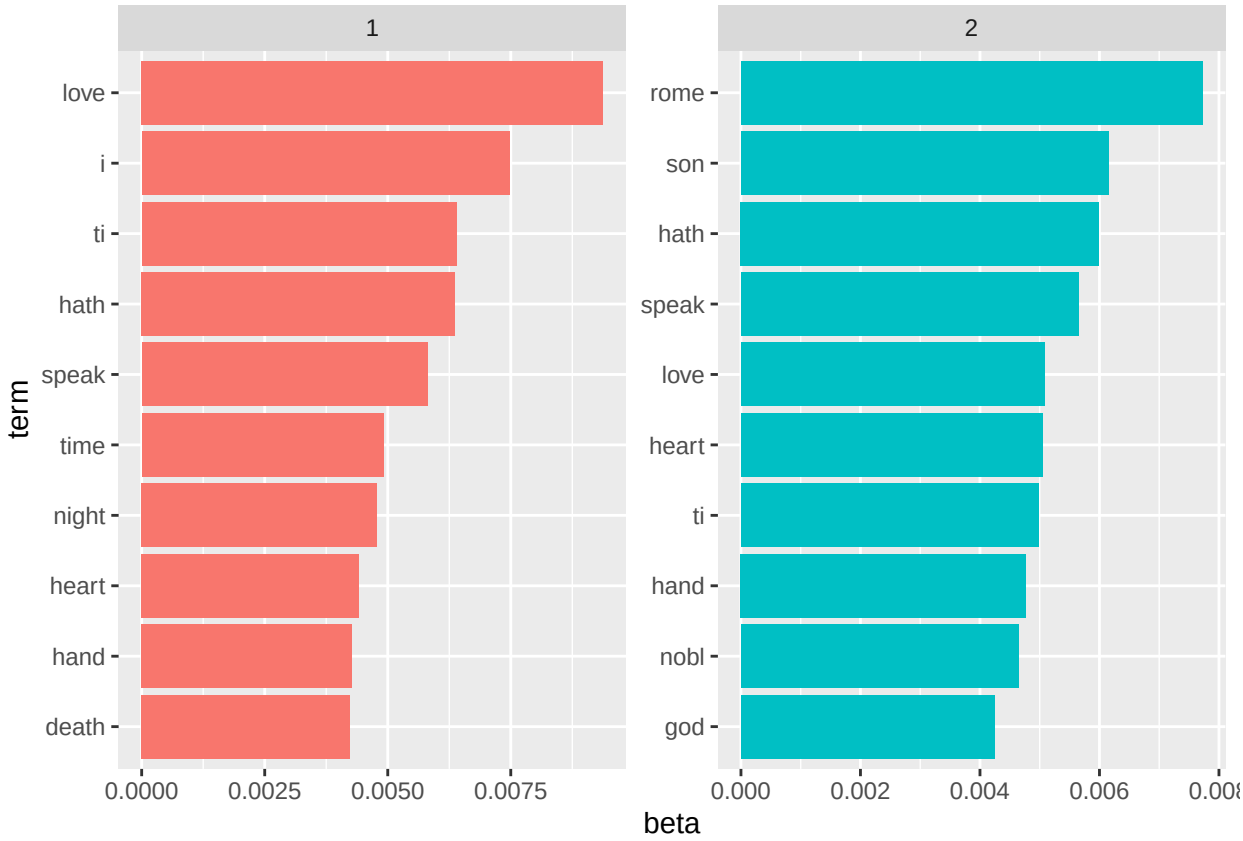
```
## Warning: package 'topicmodels' was built under R version 4.2.2
```

```
tragedy_play <- tf_idf |>
  filter(play %in% tragedy) |>
  cast_dtm(play, stem, n)
tragedy_lda <- LDA(tragedy_play, k = 2, control = list(seed = 1234))
```

```
tragedy_topics <- tidy(tragedy_lda, matrix = "beta")
```

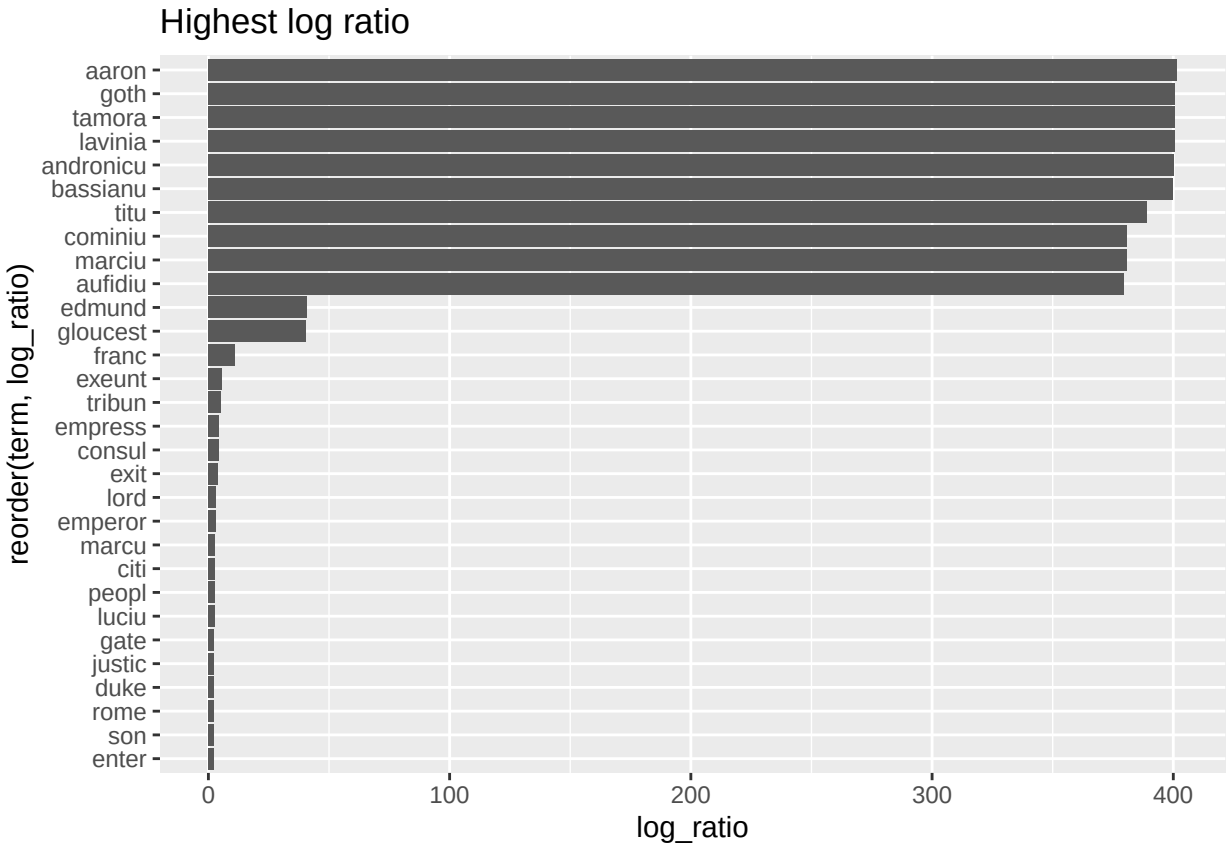
```
tragedy_top_term <- tragedy_topics |>
  group_by(topic) |>
  slice_max(beta, n = 10) |>
  ungroup() |>
  arrange(topic, -beta)
```

```
tragedy_top_term |> mutate(term = reorder_within(term, beta, topic)) |>
  ggplot(aes(beta, term, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  scale_y_reordered()
```

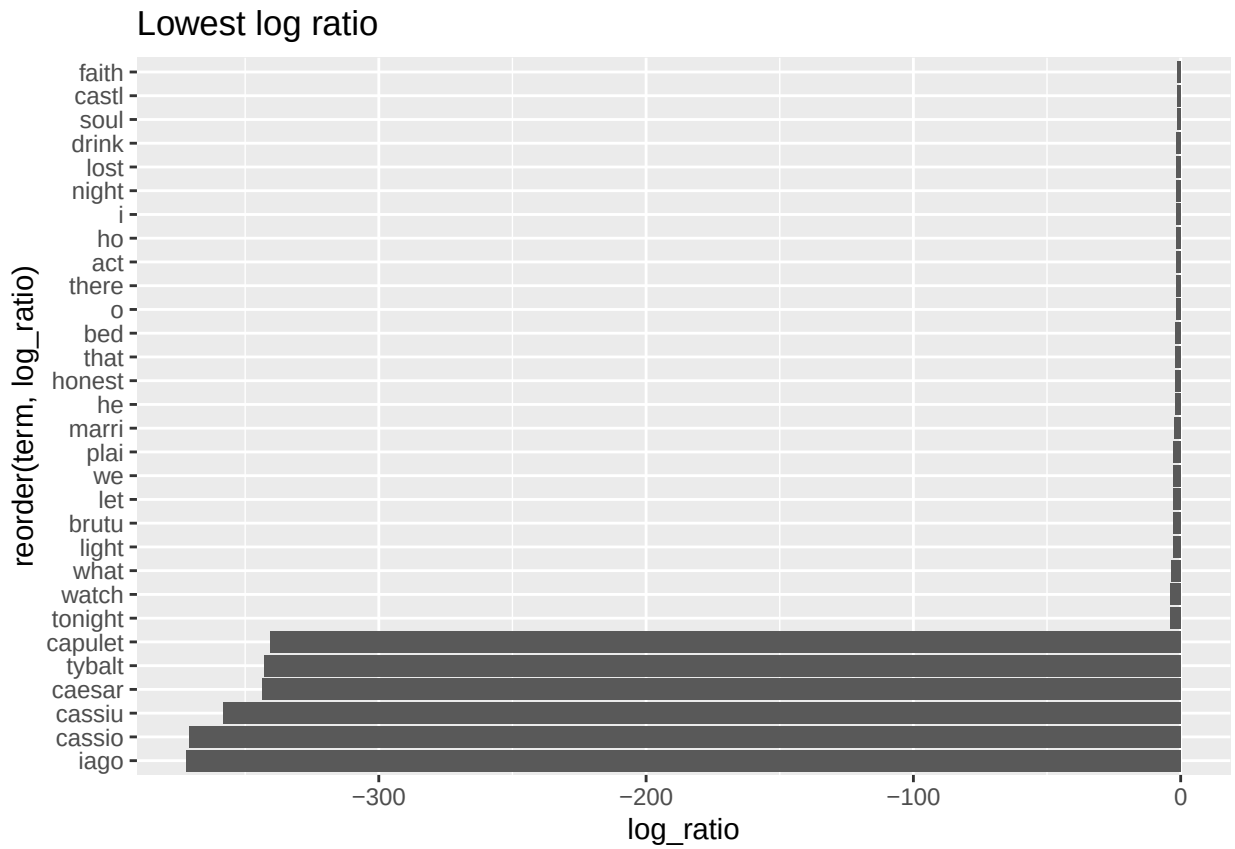



```
beta_wide <- tragedy_topics |>
  mutate(topic = paste0("topic", topic)) |>
  pivot_wider(names_from = topic, values_from = beta) |>
  filter(topic1 > .001 | topic2 > .001) |>
  mutate(log_ratio =log2(topic2 / topic1))
```

```
beta_wide |> arrange(-1*log_ratio) |>
  slice_head(n = 30) |>
  ggplot(aes(x = reorder(term, log_ratio), y= log_ratio)) +
  geom_col(show.legend = FALSE, orientation = "x") +
  coord_flip() +
  ggtitle(("Highest log ratio"))
```

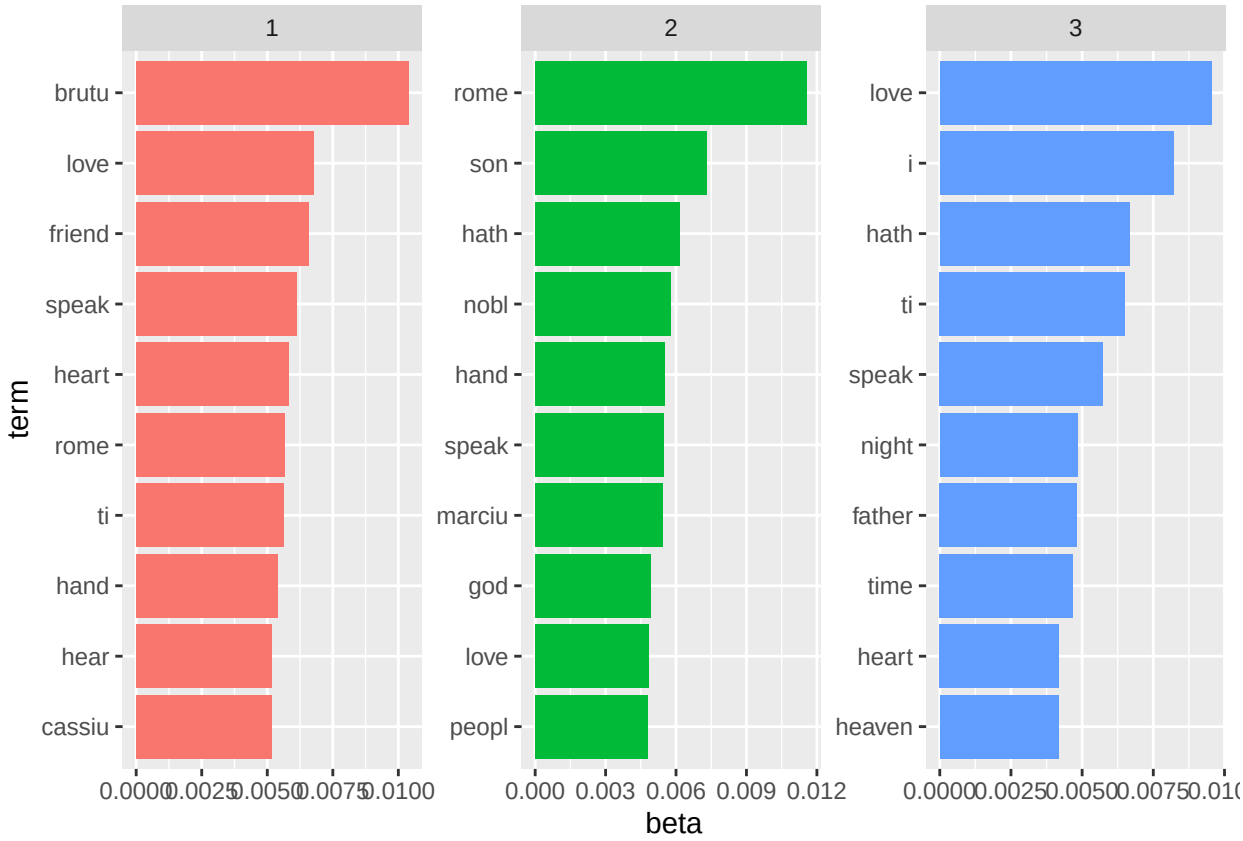



```
beta_wide |> arrange(log_ratio) |>
  slice_head(n = 30) |>
  ggplot(aes(x = reorder(term, log_ratio), y = log_ratio)) +
  geom_col(show.legend = FALSE, orientation = "x") +
  coord_flip() +
  ggtitle("Lowest log ratio")
```




```
tragedy_documents <- tidy(tragedy_lda, matrix = "gamma")
```

```
tragedy_lda3 <- LDA(tragedy_play, k =3, control= list(seed = 1234))
tragedy_documents <- tidy(tragedy_lda3, matrix = "beta")

tragedy_top_terms <- tragedy_documents |>
  group_by(topic) |>
  slice_max(beta, n = 10) |>
  ungroup() |>
  arrange(topic, -beta)

tragedy_top_terms |>
  mutate(term = reorder_within(term, beta, topic)) |>
  ggplot(aes(beta, term, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~topic, scales = "free") +
  scale_y_reordered()
```

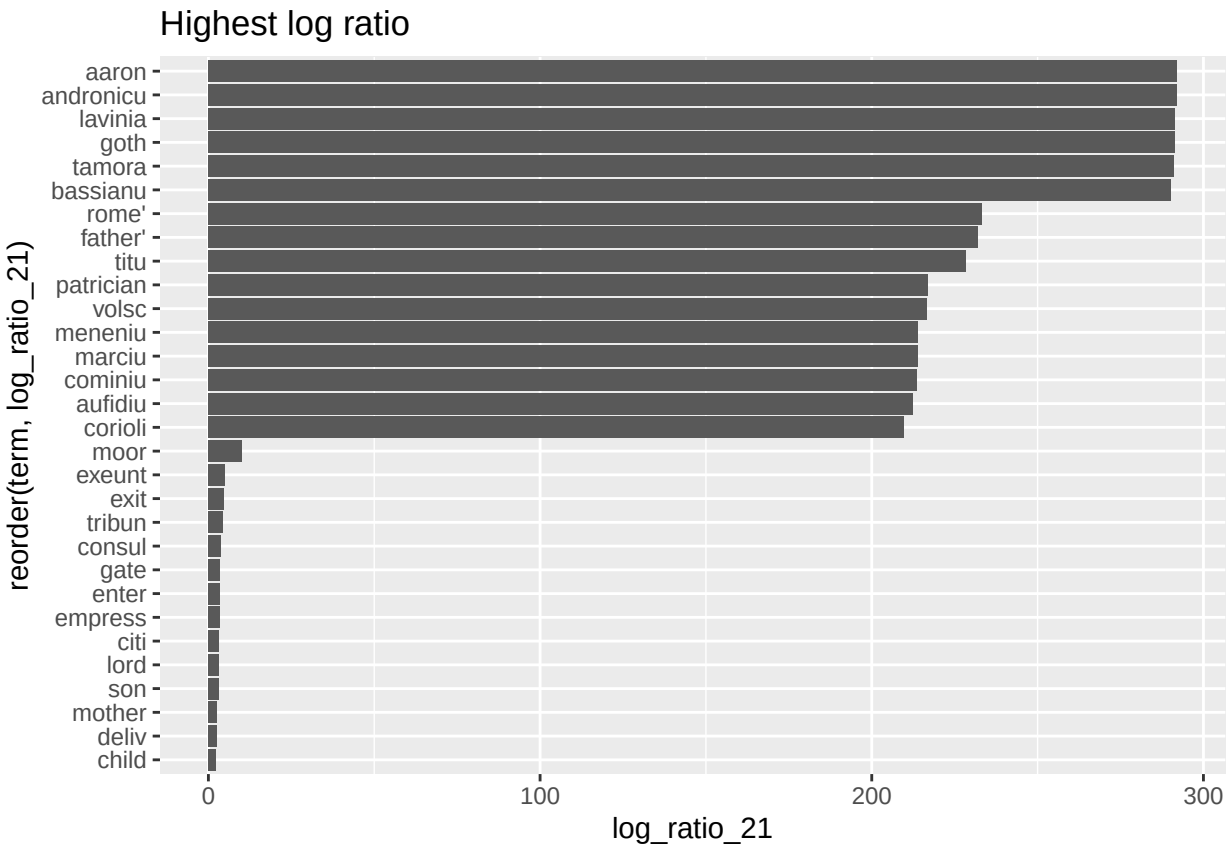



```
tragedy_documents_gamma <- tidy(tragedy_lda3, matrix = "gamma")
tragedy_documents_gamma
```

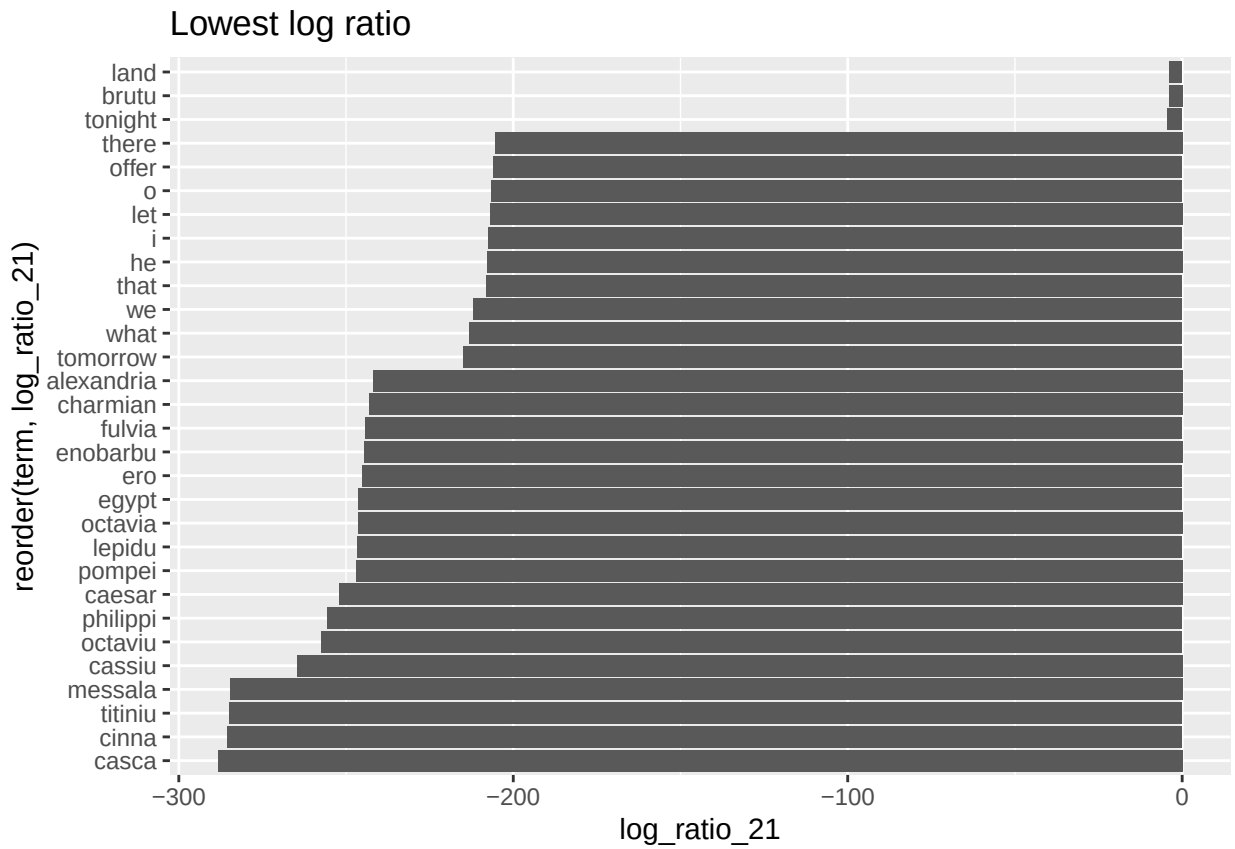
```
## # A tibble: 27 x 3
##    document                                    topic      gamma
##    <chr>                                       <int>      <dbl>
##  1 THE TRAGEDY OF ROMEO AND JULIET                 1 0.00000280
##  2 THE TRAGEDY OF JULIUS CAESAR                    1 1.00
##  3 THE TRAGEDY OF OTHELLO, THE MOOR OF VENICE      1 0.0290
##  4 THE TRAGEDY OF CORIOLANUS                       1 0.00000282
##  5 THE TRAGEDY OF TITUS ANDRONICUS                 1 0.00000319
##  6 THE TRAGEDY OF KING LEAR                        1 0.00000272
##  7 THE TRAGEDY OF HAMLET, PRINCE OF DENMARK        1 0.00216
##  8 THE TRAGEDY OF ANTONY AND CLEOPATRA             1 1.00
##  9 THE TRAGEDY OF MACBETH                          1 0.00000408
## 10 THE TRAGEDY OF ROMEO AND JULIET                 2 0.00000280
## # ... with 17 more rows
```

```
beta_wide3 <- tragedy_documents |>
  mutate(topic = paste0("topic", topic)) |>
  pivot_wider(names_from = topic, values_from = beta) |>
  filter(topic1 >.001 | topic2 > .001) |>
  mutate(log_ratio_21 = log2(topic2/topic1),
         log_ratio_23 = log2(topic2/topic3),
         log_ratio_13 = log2(topic1/topic3))
```

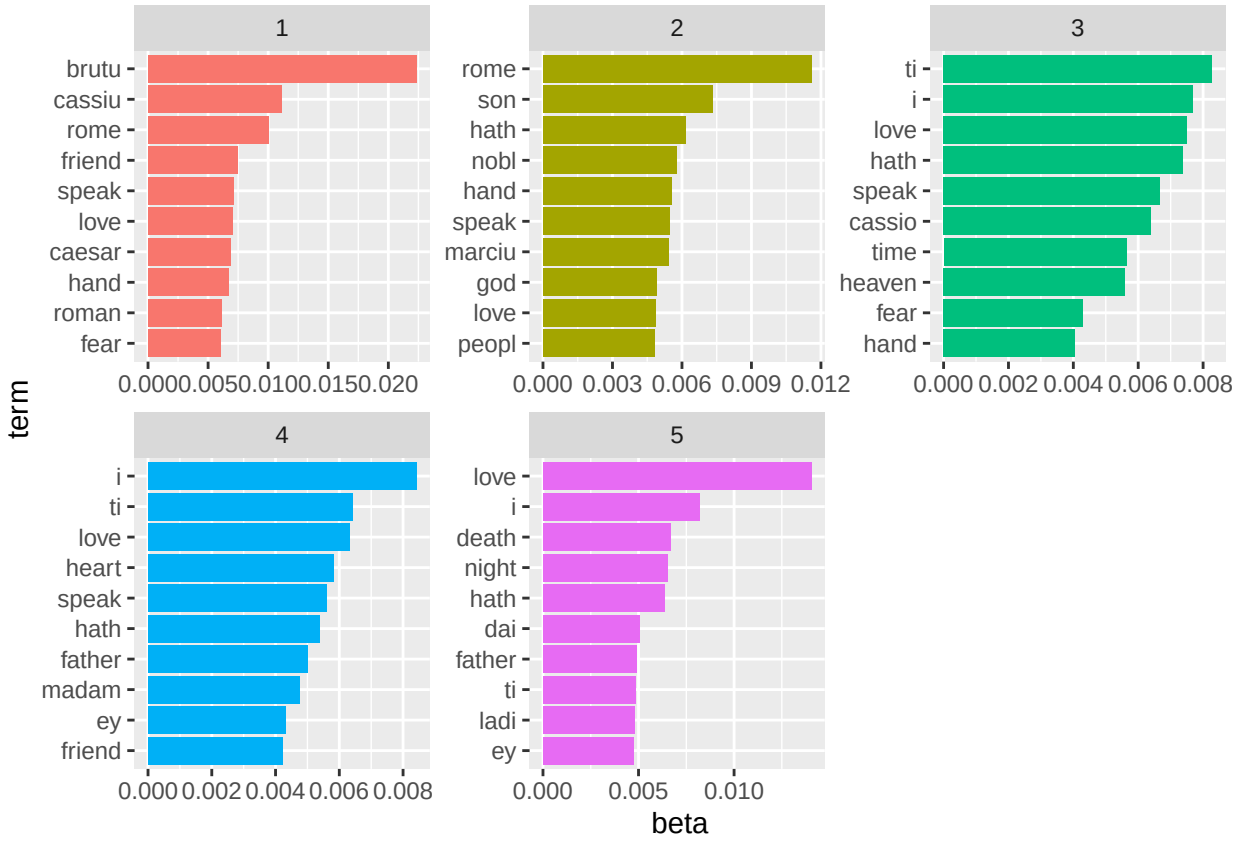
```
beta_wide3 |> arrange(-log_ratio_21) |>
  slice_head(n =30) |>
  ggplot(aes(x = reorder(term, log_ratio_21), y = log_ratio_21)) +
  geom_col(show.legend = FALSE, orientation = "x") +
  coord_flip() +
  ggtitle("Highest log ratio")
```




```
beta_wide3 |> arrange(log_ratio_21) |>
  slice_head(n =30) |>
  ggplot(aes(x = reorder(term, log_ratio_21), y = log_ratio_21)) +
  geom_col(show.legend = FALSE, orientation = "x") +
  coord_flip() +
  ggtitle("Lowest log ratio")
```




```
tragedy_lda5 <- LDA(tragedy_play, k = 5, control= list(seed = 1234))
tragedy_documents <- tidy(tragedy_lda5, matrix = "beta")

tragedy_top_terms <- tragedy_documents |>
  group_by(topic) |>
  slice_max(beta, n = 10) |>
  ungroup() |>
  arrange(topic, -beta)

tragedy_top_terms |>
  mutate(term = reorder_within(term, beta, topic)) |>
  ggplot(aes(beta, term, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~topic, scales = "free") +
  scale_y_reordered()
```




```
tragedy_documents_gamma <- tidy(tragedy_lda, matrix = "gamma")
tragedy_documents_gamma
```

```
## # A tibble: 18 x 3
##    document                                topic      gamma
##    <chr>                                   <int>      <dbl>
##  1 THE TRAGEDY OF ROMEO AND JULIET             1 1.00
##  2 THE TRAGEDY OF JULIUS CAESAR                1 1.00
##  3 THE TRAGEDY OF OTHELLO, THE MOOR OF VENICE  1 1.00
##  4 THE TRAGEDY OF CORIOLANUS                   1 0.00000347
##  5 THE TRAGEDY OF TITUS ANDRONICUS             1 0.00000393
##  6 THE TRAGEDY OF KING LEAR                    1 0.119
##  7 THE TRAGEDY OF HAMLET, PRINCE OF DENMARK    1 0.979
##  8 THE TRAGEDY OF ANTONY AND CLEOPATRA         1 1.00
##  9 THE TRAGEDY OF MACBETH                      1 1.00
## 10 THE TRAGEDY OF ROMEO AND JULIET             2 0.00000344
## 11 THE TRAGEDY OF JULIUS CAESAR                2 0.00000483
## 12 THE TRAGEDY OF OTHELLO, THE MOOR OF VENICE  2 0.00000354
## 13 THE TRAGEDY OF CORIOLANUS                   2 1.00
## 14 THE TRAGEDY OF TITUS ANDRONICUS             2 1.00
## 15 THE TRAGEDY OF KING LEAR                    2 0.881
## 16 THE TRAGEDY OF HAMLET, PRINCE OF DENMARK    2 0.0210
## 17 THE TRAGEDY OF ANTONY AND CLEOPATRA         2 0.00000368
## 18 THE TRAGEDY OF MACBETH                      2 0.00000502
```

```
play_topics <- tragedy_documents_gamma |>
  count(document, topic) |>
  group_by(document) |>
  slice_max(n, n = 1)|>
  ungroup() |>
  mutate(consensus = document, topic)
```

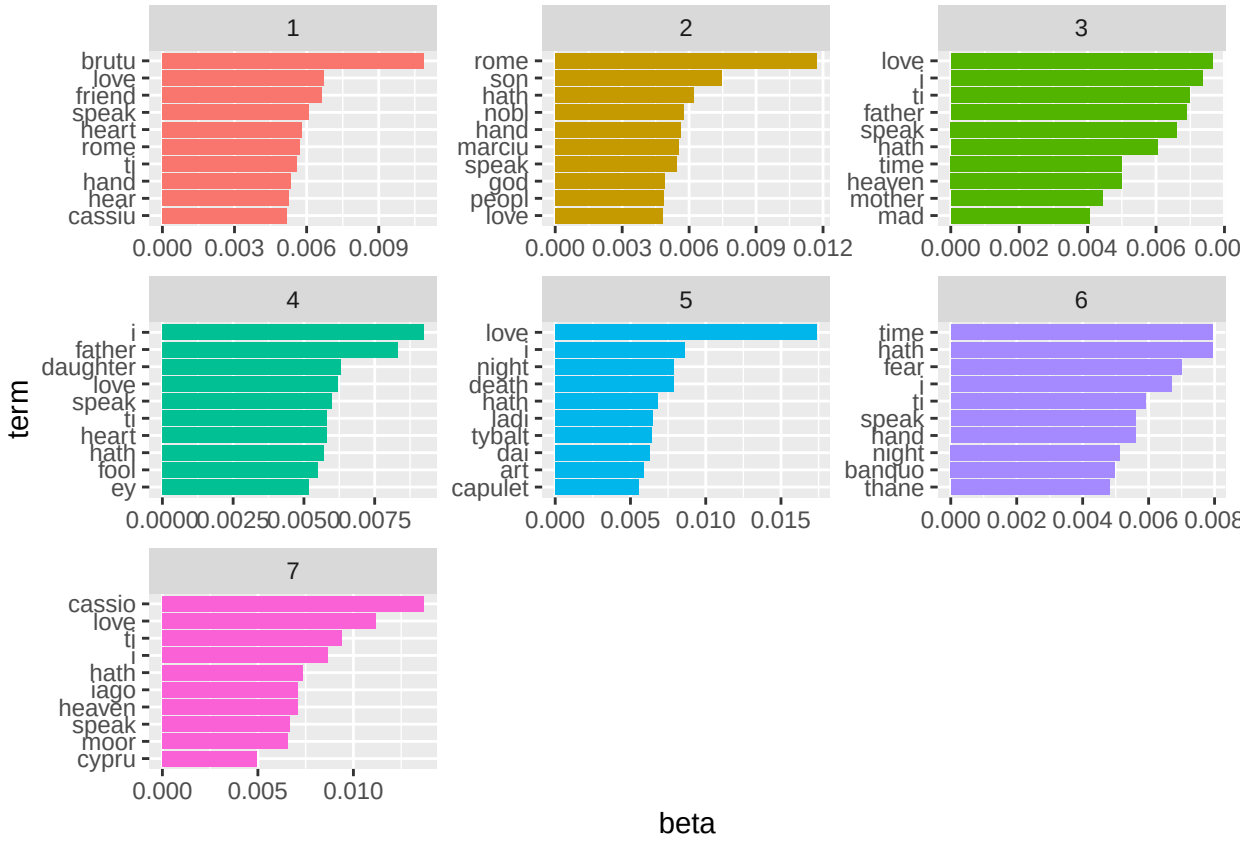
```
tragedy_documents_gamma |>
  inner_join(play_topics, by = "topic")
```

```
## Warning in inner_join(tragedy_documents_gamma, play_topics, by = "topic"): Each row in ‘x‘ is expecte
## i Row 1 of ‘x‘ matches multiple rows.
## i If multiple matches are expected, set ‘multiple = "all"‘ to silence this
##   warning.
```

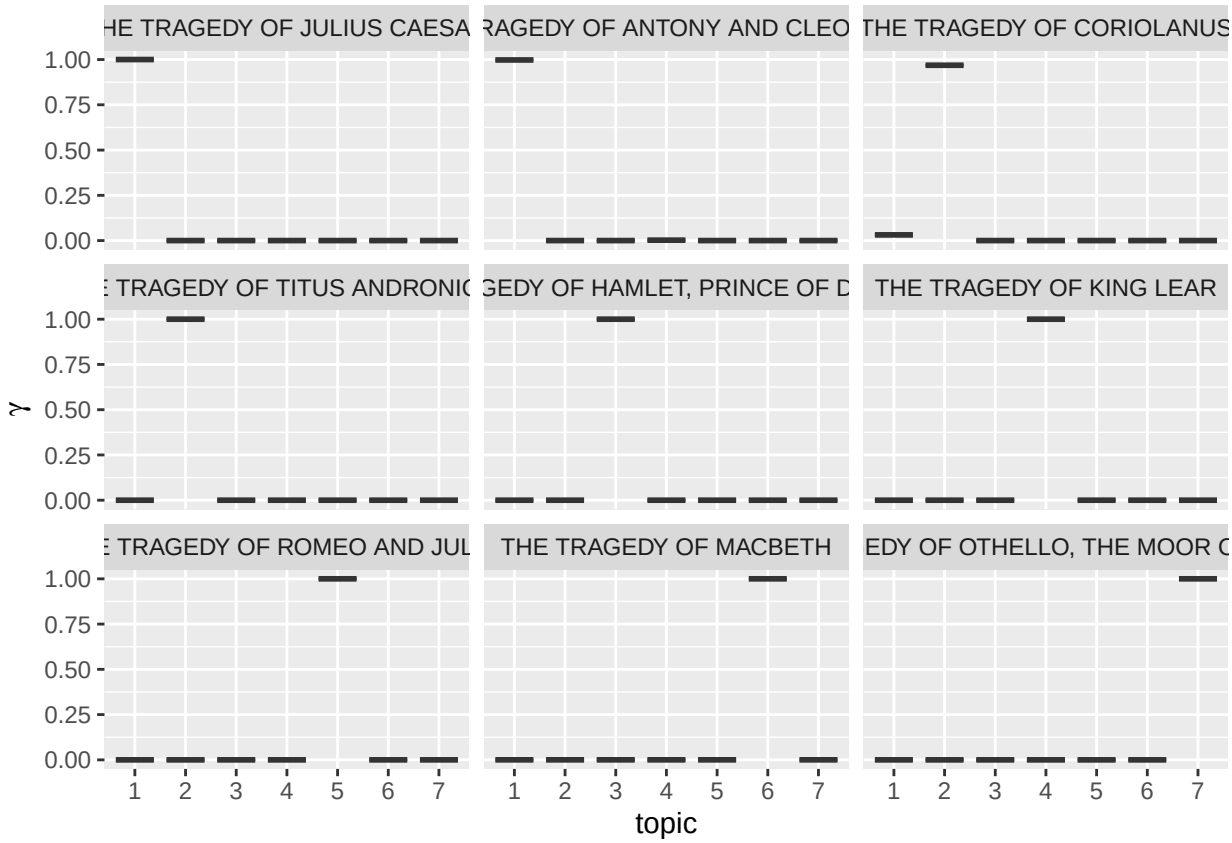
```
## # A tibble: 162 x 6
##    document.x                      topic gamma document.y              n conse~1
##    <chr>                           <int> <dbl> <chr>                <int> <chr>
##  1 THE TRAGEDY OF ROMEO AND JULIET     1  1.00 THE TRAGEDY OF ANT~     1 THE TR~
##  2 THE TRAGEDY OF ROMEO AND JULIET     1  1.00 THE TRAGEDY OF COR~     1 THE TR~
##  3 THE TRAGEDY OF ROMEO AND JULIET     1  1.00 THE TRAGEDY OF HAM~     1 THE TR~
##  4 THE TRAGEDY OF ROMEO AND JULIET     1  1.00 THE TRAGEDY OF JUL~     1 THE TR~
##  5 THE TRAGEDY OF ROMEO AND JULIET     1  1.00 THE TRAGEDY OF KIN~     1 THE TR~
##  6 THE TRAGEDY OF ROMEO AND JULIET     1  1.00 THE TRAGEDY OF MAC~     1 THE TR~
##  7 THE TRAGEDY OF ROMEO AND JULIET     1  1.00 THE TRAGEDY OF OTH~     1 THE TR~
##  8 THE TRAGEDY OF ROMEO AND JULIET     1  1.00 THE TRAGEDY OF ROM~     1 THE TR~
##  9 THE TRAGEDY OF ROMEO AND JULIET     1  1.00 THE TRAGEDY OF TIT~     1 THE TR~
## 10 THE TRAGEDY OF JULIUS CAESAR        1  1.00 THE TRAGEDY OF ANT~     1 THE TR~
## # ... with 152 more rows, and abbreviated variable name 1: consensus
```

#Word Assignments

```
play_dtm <- tokenized |> filter(play %in% tragedy) |> cast_dtm(play, stem, n)
play_lda <- LDA(play_dtm, k = 7, control = list(seed = 1234))
play_topics <- tidy(play_lda, matrix = "beta")

top_terms <- play_topics |> group_by(topic) |>
  slice_max(beta, n =10) |>
  ungroup() |>
  arrange(topic, -beta)

top_terms |>
  mutate(term = reorder_within(term, beta, topic)) |>
  ggplot(aes(beta, term, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  scale_y_reordered()
```




```
play_gamma <- tidy(play_lda, matrix = "gamma")

play_gamma |> mutate(play = reorder(document, gamma*topic)) |>
  ggplot(aes(factor(topic), gamma)) +
  geom_boxplot() +
  facet_wrap(~ play) +
  labs(x = "topic", y = expression(gamma))
```




```
play_classification <- play_gamma |>
  group_by(document) |>
  slice_max(gamma) |>
  ungroup()

play_topic <- play_classification |>
  count(document, topic) |>
  group_by(document) |>
  slice_max(n, n = 1) |>
  ungroup() |>
  transmute(consensus = document, topic)

play_classification |> inner_join(play_topic, by = "topic") |>
  filter(document != consensus)
```

```
## Warning in inner_join(play_classification, play_topic, by = "topic"): Each row in ‘x‘ is expected to
## i Row 1 of ‘x‘ matches multiple rows.
## i If multiple matches are expected, set ‘multiple = "all"‘ to silence this
##   warning.
```

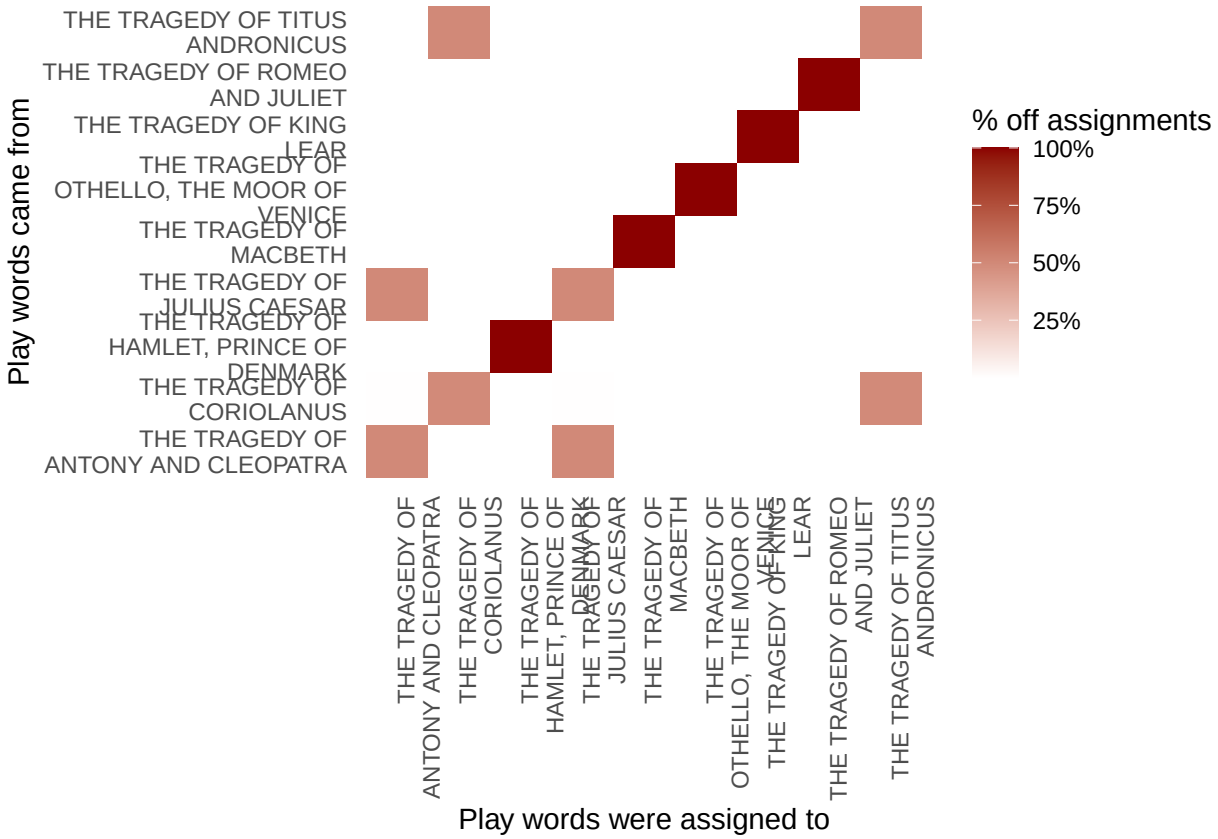
```
## # A tibble: 4 x 4
##   document                              topic gamma consensus
##   <chr>                                 <int> <dbl> <chr>
## 1 THE TRAGEDY OF ANTONY AND CLEOPATRA       1 0.998 THE TRAGEDY OF JULIUS CAESAR
## 2 THE TRAGEDY OF CORIOLANUS                 2 0.968 THE TRAGEDY OF TITUS ANDRONIC~
```

```
## 3 THE TRAGEDY OF JULIUS CAESAR            1 1.00  THE TRAGEDY OF ANTONY AND CLE~
## 4 THE TRAGEDY OF TITUS ANDRONICUS         2 1.00  THE TRAGEDY OF CORIOLANUS
```

```
assignments <- augment(play_lda, data= play_dtm)
assignments <- assignments |> inner_join(play_topic, by = c(".topic" = "topic"))
```

```
## Warning in inner_join(assignments, play_topic, by = c(.topic = "topic")): Each row in `x` is expecte
## i Row 2 of `x` matches multiple rows.
## i If multiple matches are expected, set `multiple = "all"` to silence this
##   warning.
```

```
library(scales)

assignments |>
  count(document, consensus, wt = count) |>
  mutate(across(c(document, consensus), ~str_wrap(., 20))) |>
  group_by(document) |>
  mutate(percent = n/sum(n)) |>
  ggplot(aes(consensus, document, fill = percent)) +
  geom_tile() +
  scale_fill_gradient2(high = "darkred", label = percent_format()) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        panel.grid = element_blank()) +
  labs(x = "Play words were assigned to",
       y = "Play words came from",
       fill = "% off assignments")
```




```
#Sentimental Analysis
get_sentiments(lexicon = c("bing", "afinn", "loughran", "nrc"))
```

```
## # A tibble: 6,786 x 2
##    word        sentiment
##    <chr>       <chr>
##  1 2-faces     negative
##  2 abnormal    negative
##  3 abolish     negative
##  4 abominable  negative
##  5 abominably  negative
##  6 abominate   negative
##  7 abomination negative
##  8 abort       negative
##  9 aborted     negative
## 10 aborts      negative
## # ... with 6,776 more rows
```

```
#Every play's sentiments

tokenized_without_stem <- cleaned_william |>
  filter(play %in% tragedy) |>
  group_by(play) |>
  mutate(linenumber = row_number()) |>
  ungroup() |>
```

```
  unnest_tokens(word, reg_ex) |>
  mutate(word = str_extract(word, "[a-z']+"))

tokenized_without_stem |>
  count(play, word, sort = TRUE)
```

```
## # A tibble: 32,759 x 3
##    play                                    word      n
##    <chr>                                   <chr> <int>
##  1 THE TRAGEDY OF HAMLET, PRINCE OF DENMARK   the    1094
##  2 THE TRAGEDY OF CORIOLANUS                  the     937
##  3 THE TRAGEDY OF HAMLET, PRINCE OF DENMARK   and     930
##  4 THE TRAGEDY OF KING LEAR                   the     907
##  5 THE TRAGEDY OF OTHELLO, THE MOOR OF VENICE i       891
##  6 THE TRAGEDY OF TITUS ANDRONICUS            and     814
##  7 THE TRAGEDY OF ANTONY AND CLEOPATRA        the     795
##  8 THE TRAGEDY OF OTHELLO, THE MOOR OF VENICE the     763
##  9 THE TRAGEDY OF OTHELLO, THE MOOR OF VENICE and     750
## 10 THE TRAGEDY OF HAMLET, PRINCE OF DENMARK   to      732
## # ... with 32,749 more rows
```

```
#Words in plays
bing_word_counts <- tokenized_without_stem |>
  inner_join(get_sentiments("bing")) |>
  count(word, sentiment, sort = TRUE) |>
  ungroup()
```

```
## Joining with `by = join_by(word)`

## Warning in inner_join(tokenized_without_stem, get_sentiments("bing")): Each row in `x` is expected to
## i Row 73189 of `x` matches multiple rows.
## i If multiple matches are expected, set `multiple = "all"` to silence this
##   warning.
```

```
william_sentiment <- tokenized_without_stem |>
  inner_join(get_sentiments("bing")) |>
  count(play, index = linenumber%/% 20, sentiment) |>
  pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) |>
  mutate(sentiment = positive - negative)
```

```
## Joining with `by = join_by(word)`

## Warning in inner_join(tokenized_without_stem, get_sentiments("bing")): Each row in `x` is expected to
## i Row 73189 of `x` matches multiple rows.
## i If multiple matches are expected, set `multiple = "all"` to silence this
##   warning.
```

```
p_out <- ggplot(william_sentiment, aes(index, sentiment, fill = play)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~play, ncol = 3, scales = "free_x")
ggsave("sentiments.png", plot = p_out, width = 15, height = 10)
```

```
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
library(reshape2)
```

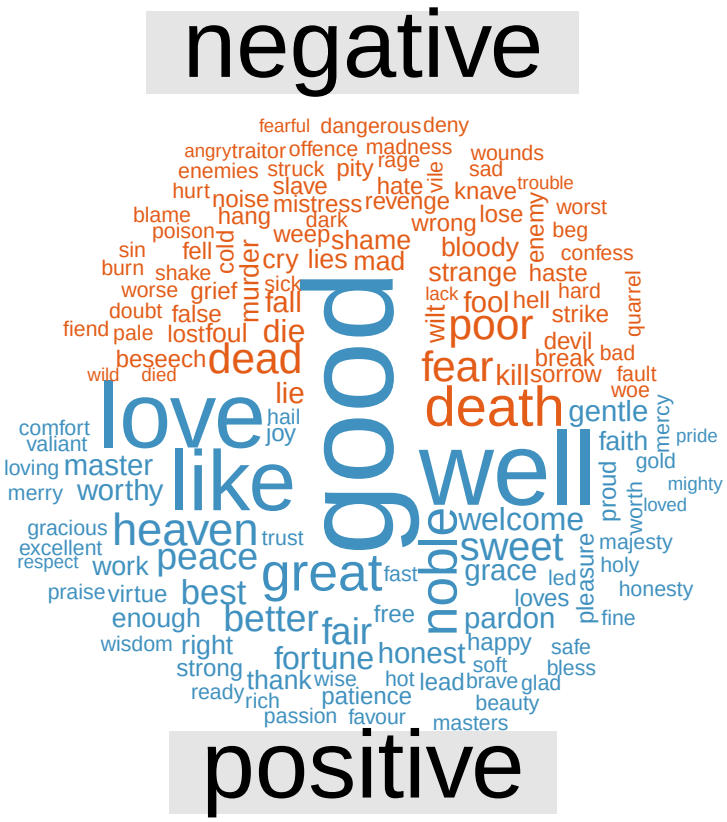
```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
tokenized_without_stem |>
  inner_join(get_sentiments("bing")) |>
  count(word, sentiment, sort = TRUE) |>
  acast(word ~ sentiment, value.var = "n", fill = 0) |>
  comparison.cloud(colors = c("#E35C18", "#4091BF"),
                   max.words = 150)
```

```
## Joining with `by = join_by(word)`
```

```
## Warning in inner_join(tokenized_without_stem, get_sentiments("bing")): Each row in `x` is expected to
## i Row 73189 of `x` matches multiple rows.
## i If multiple matches are expected, set `multiple = "all"` to silence this
##   warning.
```


negative
fearful
dangerous
deny
angry
traitor
offence
madness
wounds
enemies
struck
pity
rage
vile
sad
hurt
slave
hate
trouble
noise
mistress
revenge
knave
blame
hang
dark
wrong
lose
enemy
worst
poison
weep
shame
beg
sin
fell
cold
murder
cry
lies
mad
bloody
confess
burn
shake
strange
haste
worse
grief
sick
lack
fool
hell
hard
quarrel
doubt
false
fall
wilt
poor
strike
fiend
pale
lost
foul
die
devil
beseech
dead
break
bad
wild
died
fear
kill
sorrow
fault
lie
death
woe
love
good
hail
gentle
comfort
joy
mercy
valiant
faith
pride
loving
master
like
well
gold
merry
worthy
proud
mighty
worth
loved
gracious
heaven
trust
welcome
noble
sweet
majesty
excellent
respect
work
peace
great
fast
grace
led
holy
pleasure
praise
virtue
best
loves
honesty
enough
better
free
pardon
fair
fine
wisdom
right
happy
safe
fortune
honest
soft
bless
strong
thank
wise
hot
lead
brave
glad
ready
rich
patience
beauty
passion
favour
masters
positive