Topic Modeling

Suheng Yao

2024-11-06

```
df <- read.csv("movie_plots.csv")
```

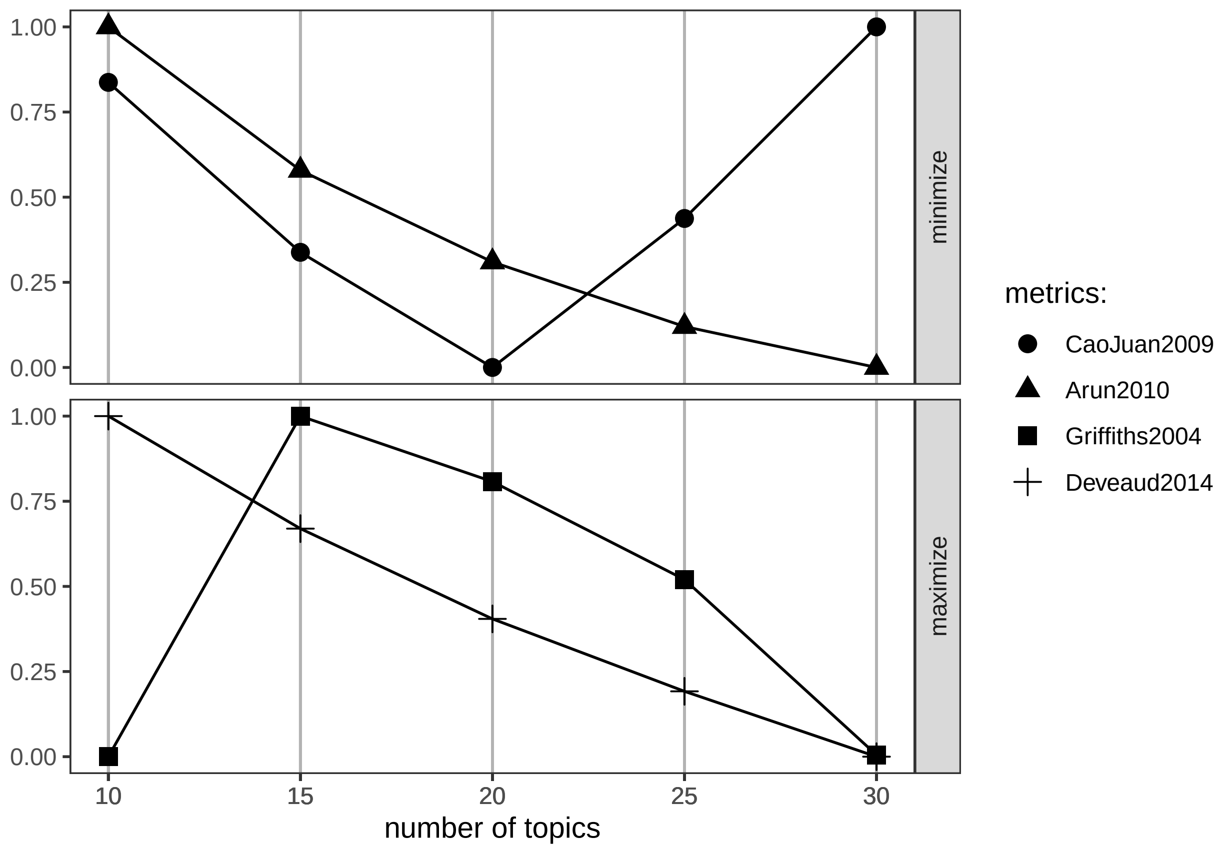
```
# Find document word counts
df_word <- df %>%
  unnest_tokens(word, Plot)

word_counts <- df_word %>%
  anti_join(stop_words, by="word") %>%
  count(Movie.Name, word, sort = TRUE)

# Use lexicon to remove common first names
data("freq_first_names")
given_name <- tolower(freq_first_names$Name)
word_counts <- word_counts %>%
  filter(!(word %in% given_name))

head(word_counts)
```

```
##                          Movie.Name      word  n
## 1                 King of the Pecos    stiles 17
## 2      French Baroque: Now and Then     dance 15
## 3            The Christmas Ornament christmas 14
## 4                   Hunted by Night     drugs 13
## 5 Islam in the Heart of the People     islam 13
## 6        Fighting Man of the Plains    dancer 12
```

As the table above shown, the most used word in all those plots is "stile" with 17 counts in the movie King of Pecos.

```
# LDA on films

# First create a DocumentTermMatrix for further topic modeling
film_dtm <- word_counts %>%
  cast_dtm(Movie.Name, word, n)

# Evaluate the number of topics (k) from 10 to 30
result <- FindTopicsNumber(
  film_dtm,
  topics = seq(10, 30, by = 5),
  metrics = c("CaoJuan2009", "Arun2010", "Griffiths2004", "Deveaud2014"),
  method = "Gibbs",
```

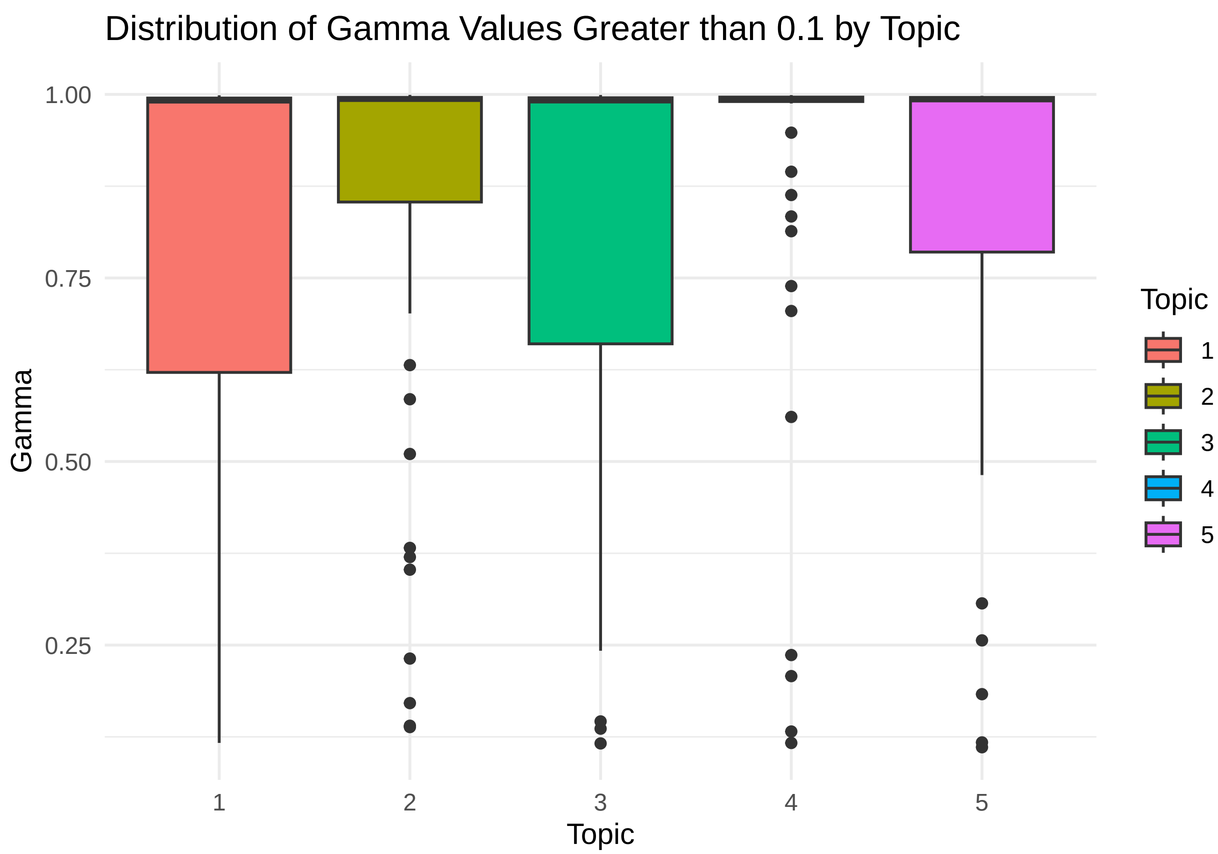
```r
  control = list(seed = 724)
)

FindTopicsNumber_plot(result)
```



From the plot created above, the best number of topic could be 20, which means that k=20.

```r
# Use LDA to create a 10 topic model
film_lda <- LDA(film_dtm, k = 20, control = list(seed = 724))
film_topics <- tidy(film_lda, matrix="gamma")
film_beta <- tidy(film_lda, matrix="beta")
```

```r
top_movie1 <- film_topics %>%
  group_by(topic) %>%
  filter(gamma > 0.1) %>%
  filter(topic %in% 1:5)

ggplot(top_movie1, aes(x = as.factor(topic),
                       y = gamma, fill = as.factor(topic))) +
  geom_boxplot() +
  labs(title = "Distribution of Gamma Values Greater than 0.1 by Topic",
       x = "Topic",
       y = "Gamma",
       fill = "Topic") +
  theme_minimal()
```



From the box plot above, it is clear that a lot of the gamma values are centered at 1 for each topic, but the distribution is kinda similar, so we can further explore what is the difference in some of the topics, for example topic 1 and 3:

```
# For topic 1
topic1_all <- film_topics %>%
  group_by(topic) %>%
  filter(topic == 1) %>%
  summarise(count=n())
print(topic1_all)
```

```
## # A tibble: 1 x 2
##   topic count
##   <int> <int>
## 1     1  1063
```

```
topic1_over0.9 <- film_topics %>%
  group_by(topic) %>%
  filter(topic == 1 & gamma > 0.9) %>%
  summarise(count=n())
print(topic1_over0.9)
```

```
## # A tibble: 1 x 2
##   topic count
##   <int> <int>
## 1     1    51
```

```r
topic1_less0.1 <- film_topics %>%
  group_by(topic) %>%
  filter(topic == 1 & gamma < 0.1) %>%
  summarise(count=n())
print(topic1_less0.1)
```

```
## # A tibble: 1 x 2
##   topic count
##   <int> <int>
## 1     1   980
```

From the summary above, for topic 1, it is clear that most of the movies have gamma values less than 0.1.

```r
# For topic 1
topic3_all <- film_topics %>%
  group_by(topic) %>%
  filter(topic == 3) %>%
  summarise(count=n())
print(topic3_all)
```

```
## # A tibble: 1 x 2
##   topic count
##   <int> <int>
## 1     3  1063
```

```r
topic3_over0.9 <- film_topics %>%
  group_by(topic) %>%
  filter(topic == 3 & gamma > 0.9) %>%
  summarise(count=n())
print(topic3_over0.9)
```

```
## # A tibble: 1 x 2
##   topic count
##   <int> <int>
## 1     3    39
```

```r
topic3_less0.1 <- film_topics %>%
  group_by(topic) %>%
  filter(topic == 3 & gamma < 0.1) %>%
  summarise(count=n())
print(topic3_less0.1)
```

```
## # A tibble: 1 x 2
##   topic count
##   <int> <int>
## 1     3  1001
```

From the summary above, for topic 3, it is also clear that most of the movies have gamma values less than 0.1. Although the box plot for topic 1 and 3 in the plot above looks similar, topic 3 actually have more movies with gamma values less than 0.1, and less movies with gamma values greater than 0.9, compared to topic 1.

```r
top_movie <- film_topics %>%
  group_by(topic) %>%
  slice_max(gamma) %>%
  ungroup()

print(top_movie)
```

```
## # A tibble: 20 x 3
##    document                                                      topic gamma
##    <chr>                                                         <int> <dbl>
##  1 "Futbaal: The Price of Dreams "                                   1 0.999
##  2 "Riders of the Purple Sage "                                      2 0.999
##  3 "Feud of the Trail "                                              3 0.999
##  4 "The Phantom Creeps "                                             4 0.999
##  5 "The Song the Zombie Sang "                                       5 0.998
##  6 "The Dancer's Peril "                                             6 0.999
##  7 "Pioneers of the West "                                           7 0.999
##  8 "Belly of the Beast "                                             8 0.999
##  9 "Streets of Ghost Town "                                          9 0.999
## 10 "The Purifiers "                                                 10 0.999
## 11 "The Prototypes "                                                11 0.999
## 12 "You Are the One "                                               12 0.999
## 13 "King of the Pecos "                                             13 0.999
## 14 "The Goose Girl "                                                14 0.999
## 15 "Summer in the Vineyard "                                        15 0.999
## 16 "The Forty-Niners "                                              16 0.999
## 17 "A Loving Gentleman "                                            17 0.999
## 18 "Last Man in Dhaka Central: The Young Man Was, Part III "         18 0.999
## 19 "Party Like the Queen of France "                                19 0.999
## 20 "The Christmas Ornament "                                        20 0.999
```

As shown in the table above, we can find out the most related movie in each topic. After that, I will try to name the first five topic and see the content within each topic by looking at the beta values related to each word in the plot:

```r
# All movies in topic 1
topic1_word <- film_beta %>%
  filter(topic == 1) %>%
  arrange(-beta)
head(topic1_word)
```

```
## # A tibble: 6 x 3
##   topic term     beta
##   <int> <chr>   <dbl>
## 1     1 life  0.0182
## 2     1 team  0.00691
## 3     1 world 0.00614
## 4     1 time  0.00548
## 5     1 movie 0.00477
## 6     1 crime 0.00384
```

The topic 1 could be related to action movie.

```r
# All movies in topic 2
topic2_word <- film_beta %>%
  filter(topic == 2) %>%
  arrange(-beta)
head(topic2_word)
```

```
## # A tibble: 6 x 3
##   topic term        beta
##   <int> <chr>      <dbl>
## 1     2 ranch    0.0122
## 2     2 money    0.00626
## 3     2 riders   0.00600
## 4     2 daughter 0.00575
## 5     2 girl     0.00561
## 6     2 venters  0.00480
```

The topic 2 could be related to western movie.

```r
# All movies in topic 3
topic3_word <- film_beta %>%
  filter(topic == 3) %>%
  arrange(-beta)
head(topic3_word)
```

```
## # A tibble: 6 x 3
##   topic term        beta
##   <int> <chr>      <dbl>
## 1     3 people   0.00802
## 2     3 film     0.00770
## 3     3 world    0.00606
## 4     3 islam    0.00533
## 5     3 religion 0.00533
## 6     3 takes    0.00531
```

The topic 3 could be related to history movies.

```r
# All movies in topic 4
topic4_word <- film_beta %>%
  filter(topic == 4) %>%
  arrange(-beta)
head(topic4_word)
```

```
## # A tibble: 6 x 3
##   topic term      beta
##   <int> <chr>    <dbl>
## 1     4 match  0.00477
## 2     4 war    0.00475
## 3     4 set    0.00438
## 4     4 world  0.00429
## 5     4 team   0.00406
## 6     4 serial 0.00398
```

The topic 4 could be related to war movie.

```
# All movies in topic 5
topic5_word <- film_beta %>%
  filter(topic == 5) %>%
  arrange(-beta)
head(topic5_word)
```

```
## # A tibble: 6 x 3
##   topic term           beta
##   <int> <chr>         <dbl>
## 1     5 world       0.00651
## 2     5 fight       0.00614
## 3     5 gold        0.00530
## 4     5 time        0.00488
## 5     5 black       0.00458
## 6     5 documentary 0.00453
```

The topic 5 could be also be related to history movies.

After finding out what is in each of those topics, we can use k-means clustering to cluster those data together:

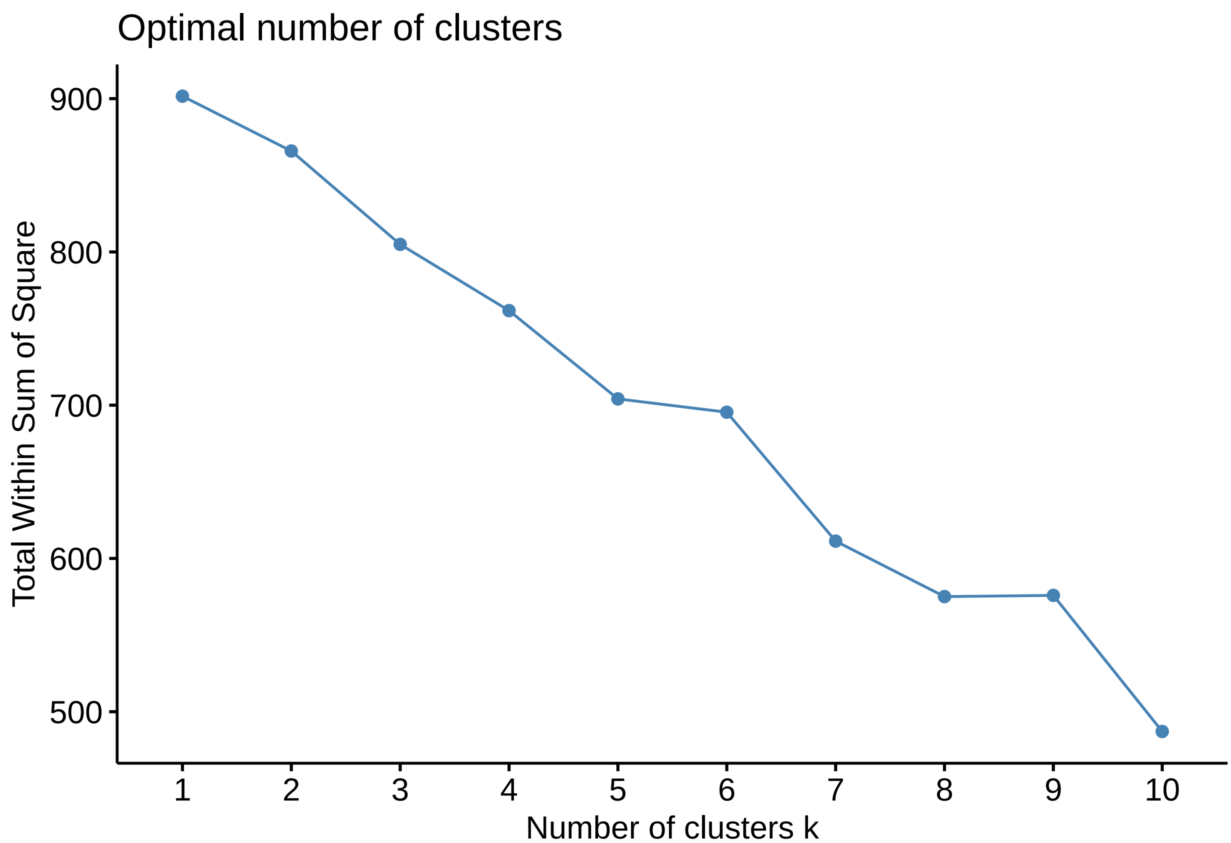
```
# Read in the movie plots with genres data
df1 <- read.csv("movie_plots_with_genres.csv")
num_genre <- df1 %>%
  group_by(Genre) %>%
  summarise(count = n()) %>%
  arrange(desc(count))
print(num_genre)
```

```
## # A tibble: 8 x 2
##   Genre   count
##   <chr>   <int>
## 1 western   323
## 2 action    246
## 3 sci-fi    132
## 4 sport     103
## 5 romance    91
## 6 history    85
## 7 fantasy    76
## 8 war        21
```

As shown in the table here, there are 8 genres, and the genre with most number of movies is “western”.

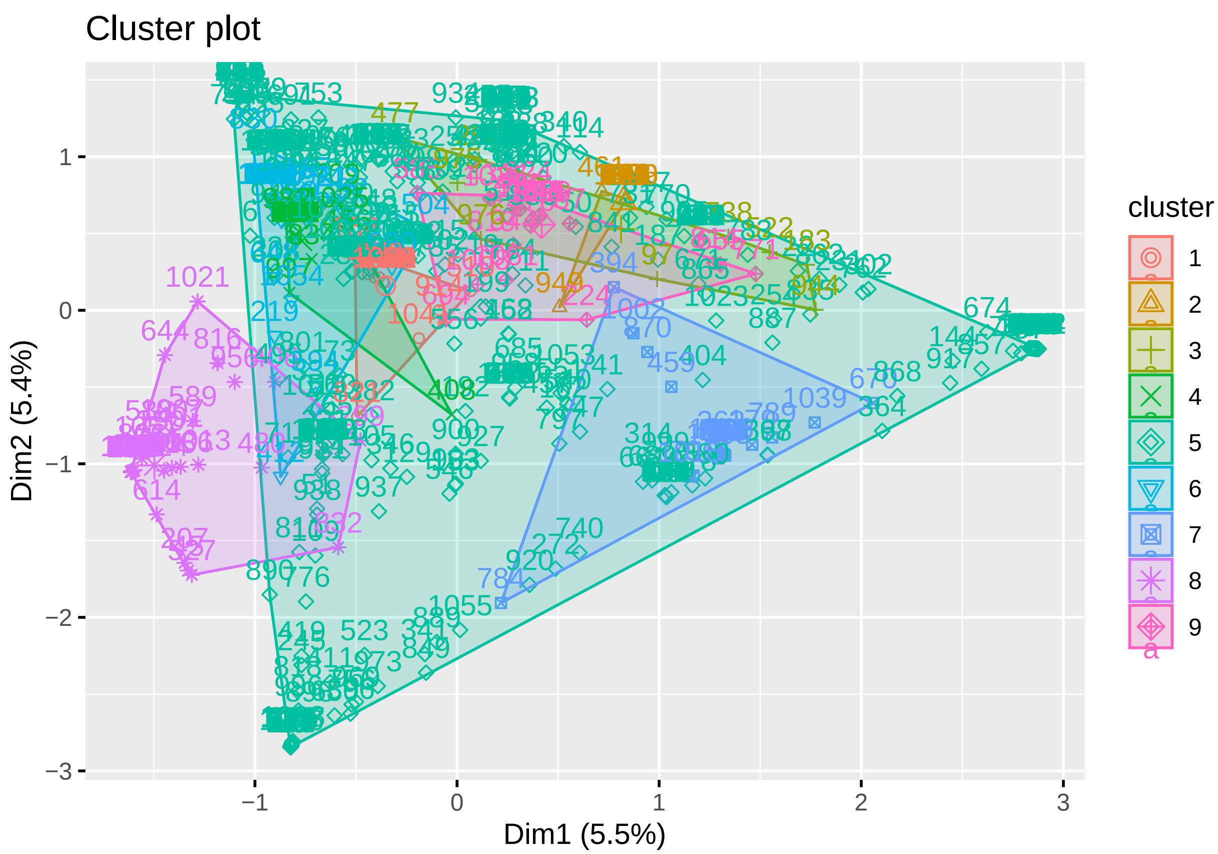
```
wider_data <- film_topics %>%
  pivot_wider(
    names_from = "topic",
    values_from = "gamma"
  )

wider_data <- wider_data %>% drop_na()
```

```r
# Use k-means to create the clusters
# First determine the optimal number of k
fviz_nbclust(wider_data %>% select(-document), kmeans, method = "wss")
```



From the scree plot above, the elbow point is at 9, so the optimized number of cluster k should be 9.

```r
set.seed(724)
cluster <- kmeans(wider_data %>% select(-document), 9)
fviz_cluster(cluster, data = wider_data %>% select(-document))
```



The graph may not be very clear, we can find out what actually in first six clusters:

```
wider_data$cluster <- cluster$cluster

df <- rename(df, document=Movie.Name)
df1 <- rename(df1, document=Movie.Name)
```

```
# Define a function to create a word cloud
generate_wordcloud <- function(cluster_num, data1=wider_data, data2=df) {
  cluster_data <- wider_data %>%
    filter(cluster == cluster_num) %>%
    select(document, cluster)

  # Join with original dataset to get the 'Plot' column
  cluster_data <- left_join(cluster_data, df, by = "document")

  cluster_data <- cluster_data %>%
    unnest_tokens(word, Plot)

  set.seed(123)

  # Create the word cloud
  cluster_data %>%
    anti_join(stop_words) %>%
    count(word) %>%
    with(wordcloud(
      words = word,
```

```
      freq = n,
      max.words = 50,
      scale = c(3, 0.5),
      min.freq = 2,
      colors = brewer.pal(8, "Dark2"),
      random.order = FALSE,
      rot.per = 0.2
    ))
}
```

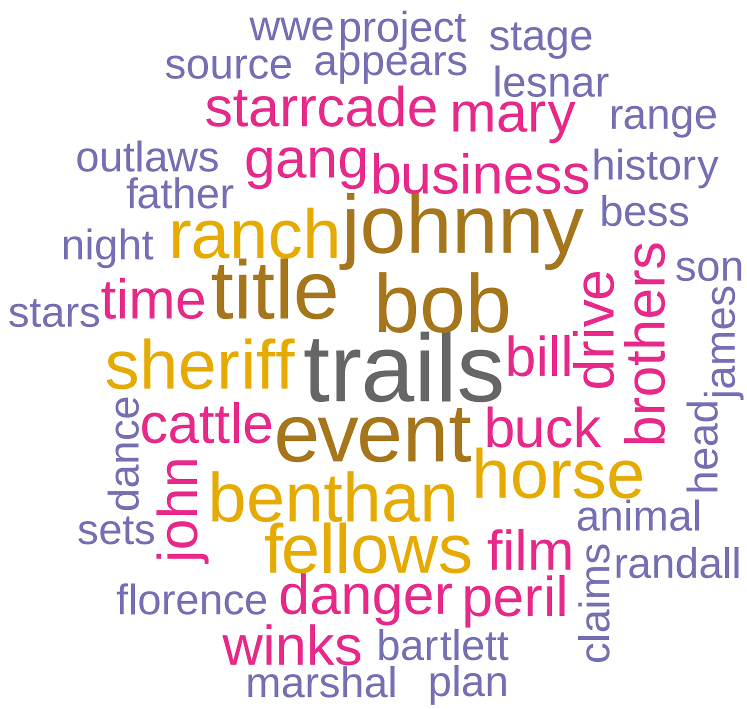
```
generate_wordcloud(1)
```

From the word cloud, the first cluster could be related to western movie.

```
generate_wordcloud(2)
```

The second cluster could also be related to romantic movie.

```
generate_wordcloud(3)
```



Cluster 3 could also be related to history movie.

```
generate_wordcloud(4)
```

Cluster 4 could be related to action movie.

```
generate_wordcloud(5)
```

Cluster 5 could be related to war movie.

```
generate_wordcloud(6)
```

Cluster 6 could be related to sci-fi movie.

```r
# Find genre in each cluster
find_genre <- function(cluster_num) {
  cluster_data <- wider_data %>%
    filter(cluster == cluster_num) %>%
    select(document, cluster)

  # Join with original dataset to get the 'Plot' column
  cluster_data <- left_join(cluster_data, df, by = "document")
  cluster_data <- left_join(cluster_data, df1, by = "document")

  top_genre <- cluster_data %>%
    group_by(Genre) %>%
    summarise(count = n()) %>%
    arrange(desc(count))

  print(top_genre)
}
```

```r
find_genre(1)
```

```
## # A tibble: 8 x 2
##   Genre   count
##   <chr>   <int>
## 1 action     19
```

```
## 2 western    11
## 3 history     6
## 4 sci-fi      6
## 5 sport       4
## 6 fantasy     3
## 7 war         3
## 8 romance     2
```

The first cluster contains a lot of western movie, which is consistent with the word cloud created before.

```r
find_genre(2)
```

```
## # A tibble: 7 x 2
##   Genre   count
##   <chr>   <int>
## 1 sport      10
## 2 action      8
## 3 western     8
## 4 fantasy     4
## 5 history     4
## 6 romance     4
## 7 sci-fi      1
```

The top genre in the second cluster is action movies, which is also consistent with the word cloud created before.

```r
find_genre(3)
```

```
## # A tibble: 4 x 2
##   Genre   count
##   <chr>   <int>
## 1 western     5
## 2 action      2
## 3 romance     1
## 4 sci-fi      1
```

Cluster 3 contains a lot of movies in action, sci-fi and western genres, this is probably why there are words like “star”, “night” and “planet”.

```r
find_genre(4)
```

```
## # A tibble: 7 x 2
##   Genre   count
##   <chr>   <int>
## 1 action     20
## 2 western     9
## 3 sci-fi      6
## 4 fantasy     4
## 5 sport       2
## 6 war         2
## 7 history     1
```

Cluster 4 again contains a lot of action movies, and that is probably why we see words like "drug", "party", "action" and "war" in the word cloud.

```r
find_genre(5)
```

```
## # A tibble: 8 x 2
##   Genre   count
##   <chr>   <int>
## 1 western   228
## 2 action    161
## 3 sci-fi     90
## 4 sport      70
## 5 romance    68
## 6 history    48
## 7 fantasy    43
## 8 war        12
```

Most of the movies in cluster 5 belong to western genre, which is why we again see a lot of words like "gang", "ranch" and "horse" in the word cloud.

```r
find_genre(6)
```

```
## # A tibble: 8 x 2
##   Genre   count
##   <chr>   <int>
## 1 western    14
## 2 action     11
## 3 history    11
## 4 fantasy     8
## 5 romance     6
## 6 sport       5
## 7 sci-fi      3
## 8 war         1
```

Most of the movies in cluster 6 belong to action genre, which is why we see a lot of words like "fight", "war" and "expedition" in the word cloud.

In summary, most of the clusters created by k-means correctly reflected the genre of movies and correspond to the word in the word clouds. However, some clusters still overlap with each other, and some genres in the movie_plots_with_genre data are missing, indicating that the model may not accurately classify some of the movies, and this is the problem I can work on in the next step.