Topic Modeling

Jonathan Neimann

2024-11-01

Note: I have set the seed on the lda but i am still getting different results in my clusters. the code should work fine but the results i write about might be different than what you are seeing on the pdf

```
movies = read.csv("movie_plots_with_genres.csv")
plots_by_word = movies %>% unnest_tokens(word,Plot)
plot_word_counts = plots_by_word %>%  anti_join(stop_words) %>%
count(Movie.Name, word, sort=TRUE)
```

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

```
data("freq_first_names")
first_names = tolower(freq_first_names$Name)
plot_word_counts = plot_word_counts %>% filter(!(word %in% first_names))

plots_dtm = plot_word_counts %>%  cast_dtm(Movie.Name, word, n)
```

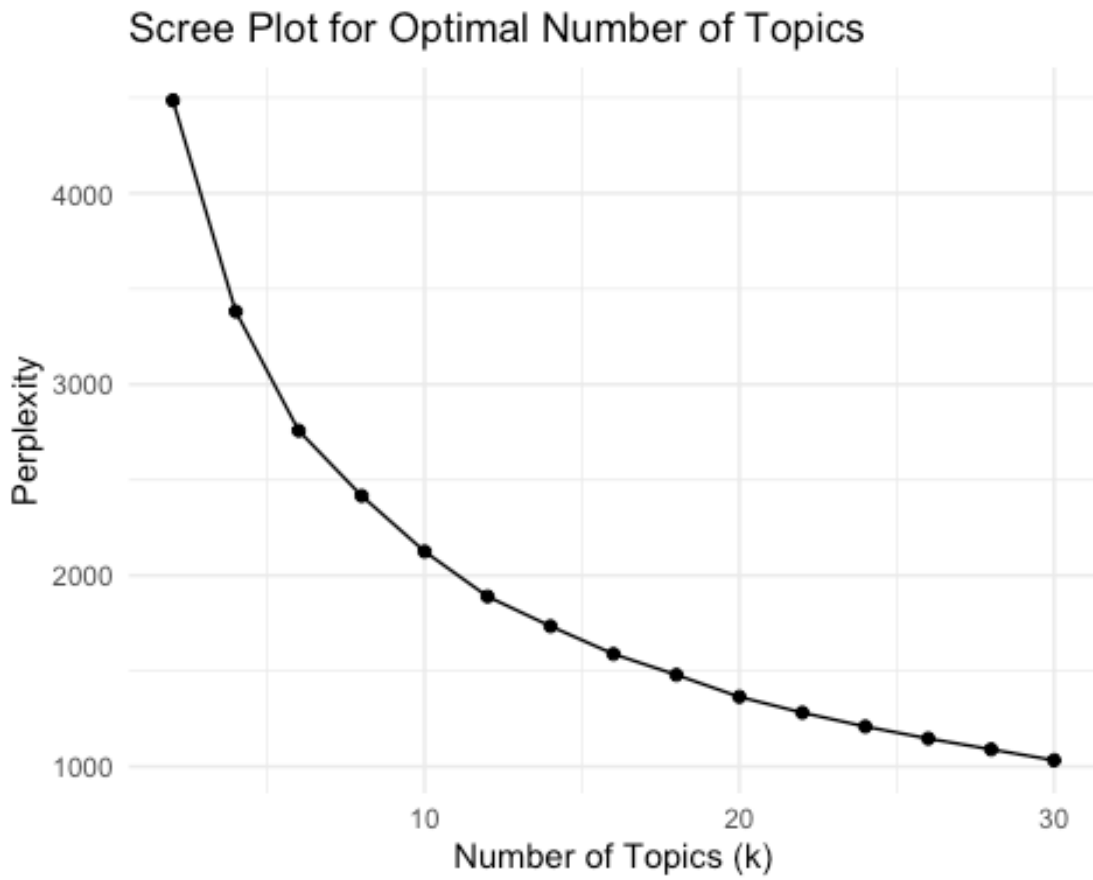
Create a screen plot to optimize the number of topics we should use

```
# Define a range of topic numbers (e.g., 2 to 30)
k_values <- seq(2, 30, by = 2)

# Initialize a data frame to store the results
perplexity_results <- data.frame(k = integer(), perplexity = numeric())

# Loop over each k and fit an LDA model, storing the perplexity
for (k in k_values) {
  lda_model <- LDA(plots_dtm, k = k, control = list(seed = 123))
  perplexity_val <- perplexity(lda_model, plots_dtm)
  perplexity_results <- rbind(perplexity_results, data.frame(k = k,
perplexity = perplexity_val))
}

# Plot the perplexity against the number of topics (k)
library(ggplot2)
ggplot(perplexity_results, aes(x = k, y = perplexity)) +
  geom_line() +
  geom_point() +
  labs(title = "Scree Plot for Optimal Number of Topics",
       x = "Number of Topics (k)",
       y = "Perplexity") +
  theme_minimal()
```



This scree plot does not seem to have an "elbow" that determines our optimal number of topics, so this can be up to the user's discretion. I am going to go with 20.

LDA with 20 topics

```
plots_lda = LDA(plots_dtm, k = 20, control = list(seed=123))
```

Now we need to retrieve the gammas from this lda, which represent the topics.

```
#retrieving gammas

betas = tidy(plots_lda, matrix = "beta")
betas_wider = betas %>%  pivot_wider(names_from = topic, values_from = beta)

plots_gamma = tidy(plots_lda, matrix = "gamma")
plots_gamma_wider = plots_gamma %>% pivot_wider(names_from = topic,
                                                values_from = gamma
                                                )
```

```
plots_gamma_wide_No_na = plots_gamma_wider %>% drop_na()
cluster = kmeans(plots_gamma_wider %>%  select(-document),10)
```

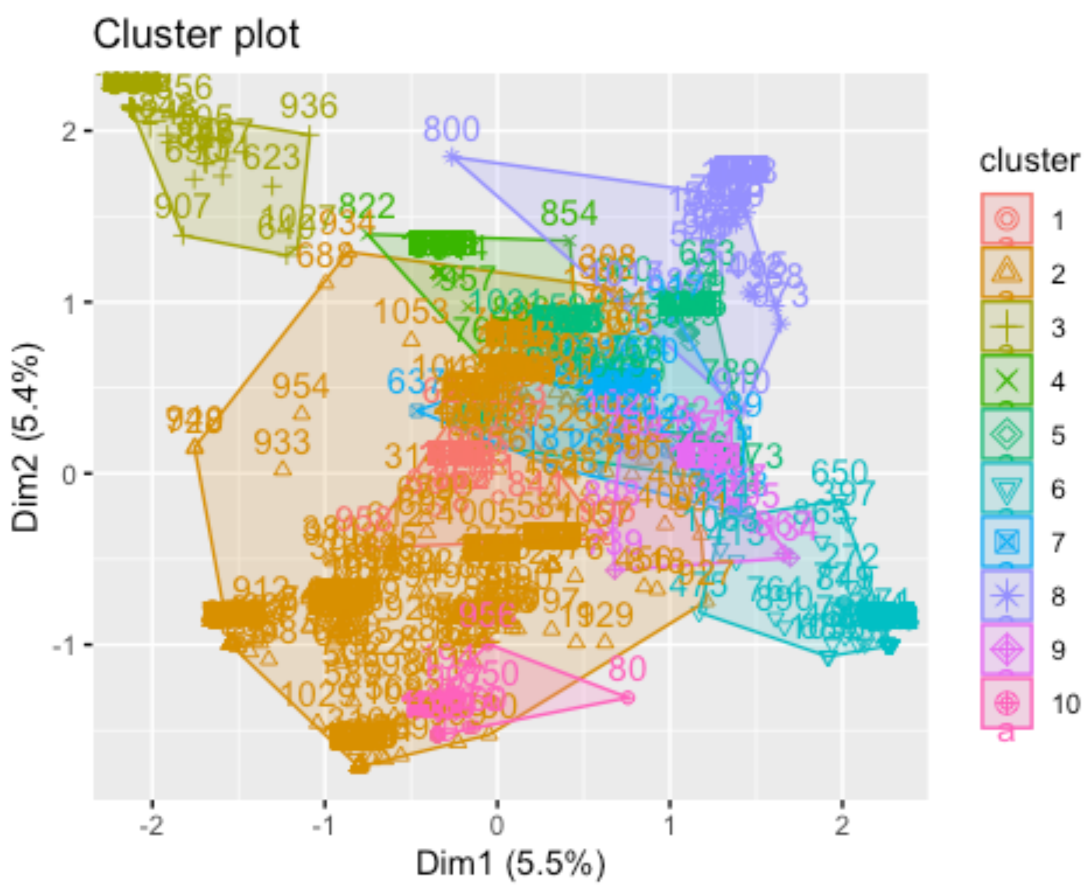
After finding these gammas, we can take the highest gamma for each movie to get a quick look at what general topic each movie belongs to

```
top_movies_by_topic <- plots_gamma_wider %>%
  pivot_longer(cols = `1`:`20`, names_to = "topic", values_to = "gamma") %>%
  group_by(document) %>%  # Group by movie/document
  slice_max(gamma, n = 1) %>%  # Get the row with the highest gamma for each
movie
  ungroup() %>%
  select(document, topic, gamma)  # Keep relevant columns
```

Clusters

But do get a more detailed look, we need to cluster the movies into 10 clusters by topic

```
plots_gamma_wider_No_na = plots_gamma_wider %>% drop_na()
cluster = kmeans(plots_gamma_wider %>%  select(-document),10)
fviz_cluster(cluster, data = plots_gamma_wider %>% select(-document))
```



We can now take these clusters and assign them to each movie in the dataframe

```
#drop duplicate movie names to match the rows
movies <- movies %>%
  distinct(Movie.Name, .keep_all = TRUE)

clusters <- cluster[["cluster"]]
cluster$cluster <- clusters
movies$cluster <- clusters
```

now we can see which movies belong to each cluster and take a deeper look.

```
#combine clusters with plots_gamma_wider for topic probabilies
plots_gamma_wider <- plots_gamma_wider %>%
  left_join(movies %>% select(Movie.Name, cluster), by = c("document" =
"Movie.Name"))

#take all movies and split them by clusters

cluster_1 <- plots_gamma_wider %>%
  filter(cluster == 1)

cluster_2 <- plots_gamma_wider %>%
  filter(cluster == 2)

cluster_3 <- plots_gamma_wider %>%
  filter(cluster == 3)

cluster_4 <- plots_gamma_wider %>%
  filter(cluster == 4)

cluster_5 <- plots_gamma_wider %>%
  filter(cluster == 5)

cluster_6 <- plots_gamma_wider %>%
  filter(cluster == 6)

cluster_7 <- plots_gamma_wider %>%
  filter(cluster == 7)

cluster_8 <- plots_gamma_wider %>%
  filter(cluster == 8)

cluster_9 <- plots_gamma_wider %>%
  filter(cluster == 9)

cluster_10 <- plots_gamma_wider %>%
  filter(cluster == 10)
```

We can now take the averages of these dataframes to see which topic is associated most with each cluster

```
#create a function that takes averages of the columns

average_columns <- function(df) {
  # Select only columns named 1 to 20
  selected_columns <- df %>%
    select(`1`:`20`)

  # Calculate the column averages
  column_averages <- colMeans(selected_columns, na.rm = TRUE)

  return(column_averages)
}

# use this function for a cluster (1)
averages_cluster_1 <- average_columns(cluster_1)
print(averages_cluster_1)
```

```
##           1           2           3           4           5           6
## 0.074588920 0.076211741 0.054285215 0.072251828 0.086216236 0.001397285
##           7           8           9          10          11          12
## 0.039313256 0.052618655 0.058959002 0.039351455 0.119292812 0.037636649
##          13          14          15          16          17          18
## 0.039321689 0.006324736 0.019852370 0.056355101 0.119063649 0.006169812
##          19          20
## 0.020930394 0.019859193
```

We can see that these probailites are pretty small, however a few of them stick out, particuarily topics 4 and 14. This indicates that cluster 1 is most assiciates with topics 4 and 14.

We can now create a word cloud from these topics

```
library(wordcloud)
```

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

```
# Define a function to create word clouds for a given topic
create_wordcloud <- function(topic_number) {
  # Filter betas for the chosen topic and select the top words by beta
  top_words <- betas %>%
    filter(topic == topic_number) %>%
    top_n(30, beta) %>%   # Adjust 30 to however many top words you want
    arrange(desc(beta))

  # Generate the word cloud
  wordcloud(words = top_words$term,
            freq = top_words$beta,
```

```
             min.freq = 0,
             scale = c(3, 0.5),    # Adjust word size range
             random.order = FALSE,
             colors = brewer.pal(8, "Dark2"))
}

# Create a word cloud for Topic 4
create_wordcloud(4)
```


```
#topic 14
create_wordcloud(14)
```

based on these word clouds the genre is not that clear. However we can kind of say it is between western and sci fi

Lets do this again for cluster 6

```
averages_cluster_6 <- average_columns(cluster_6)
print(averages_cluster_6)

##          1          2          3          4          5          6
7
## 0.01356578 0.01664848 0.01663862 0.11156594 0.03579947 0.05687211
0.01423304
##          8          9         10         11         12         13
14
## 0.03471028 0.06257654 0.03771703 0.08897297 0.06540915 0.03172539
0.10792952
##         15         16         17         18         19         20
## 0.05843596 0.05231028 0.04244371 0.06560673 0.07168099 0.01515801
```

this cluster seems to be mostly associated with topics 1 and 12 so let’s see their word clouds.

```
create_wordcloud(1)
```


```
create_wordcloud(12)
```

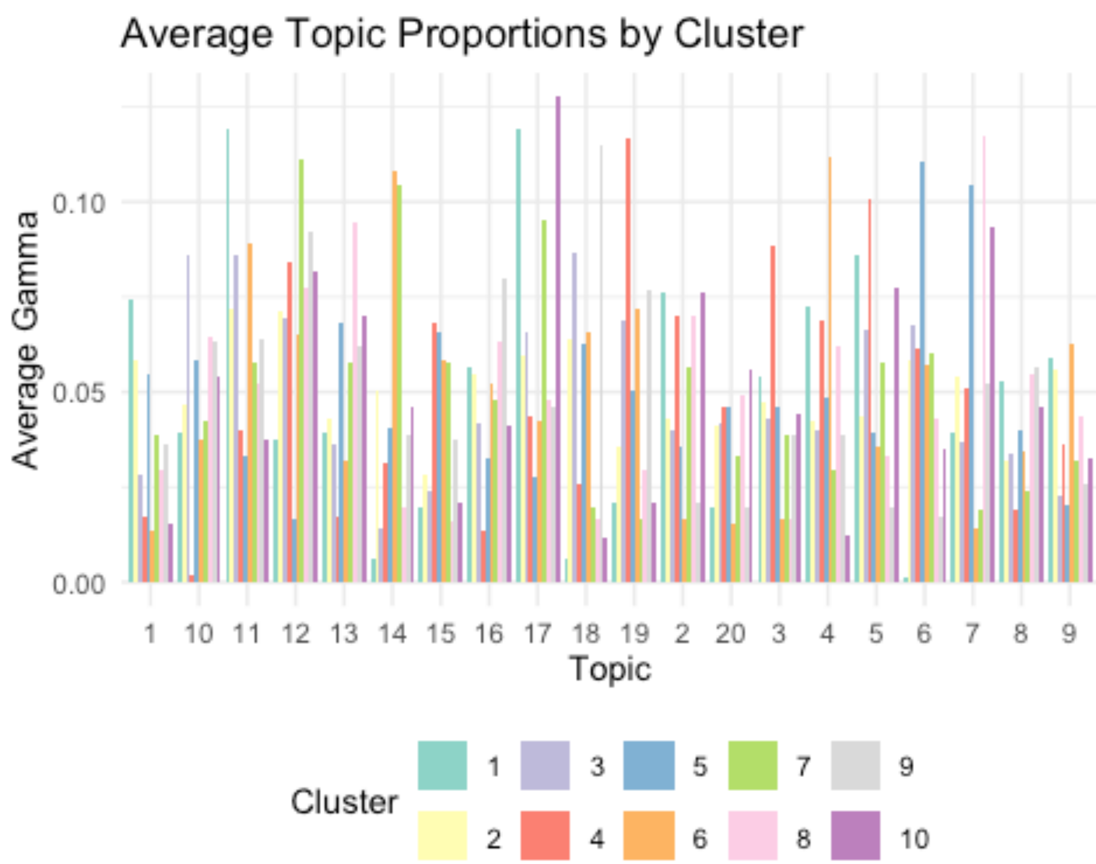
Joining these two word clouds it seems that cluster 6 can be classified as crime thrillr/mystery movies

Additionaly we can look at the highest proportional topics across clusters

```
cluster_topic_averages <- plots_gamma_wider %>%
  group_by(cluster) %>%
  summarize(across(`1`:`20`, mean, na.rm = TRUE)) %>%
  pivot_longer(cols = `1`:`20`, names_to = "topic", values_to =
"average_gamma")
```

```
## Warning: There was 1 warning in `summarize()`.
## ℹ In argument: `across(`1`:`20`, mean, na.rm = TRUE)`.
## ℹ In group 1: `cluster = 1`.
## Caused by warning:
## ! The `...` argument of `across()` is deprecated as of dplyr 1.1.0.
## Supply arguments directly to `.fns` through an anonymous function instead.
##
##   # Previously
##   across(a:b, mean, na.rm = TRUE)
##
##   # Now
##   across(a:b, \(x) mean(x, na.rm = TRUE))
```

```
# Plot topic proportions for each cluster
ggplot(cluster_topic_averages, aes(x = factor(topic), y = average_gamma, fill
= factor(cluster))) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Average Topic Proportions by Cluster", x = "Topic", y =
"Average Gamma") +
  theme_minimal() +
  theme(legend.position = "bottom") +
  scale_fill_brewer(palette = "Set3", name = "Cluster")
```

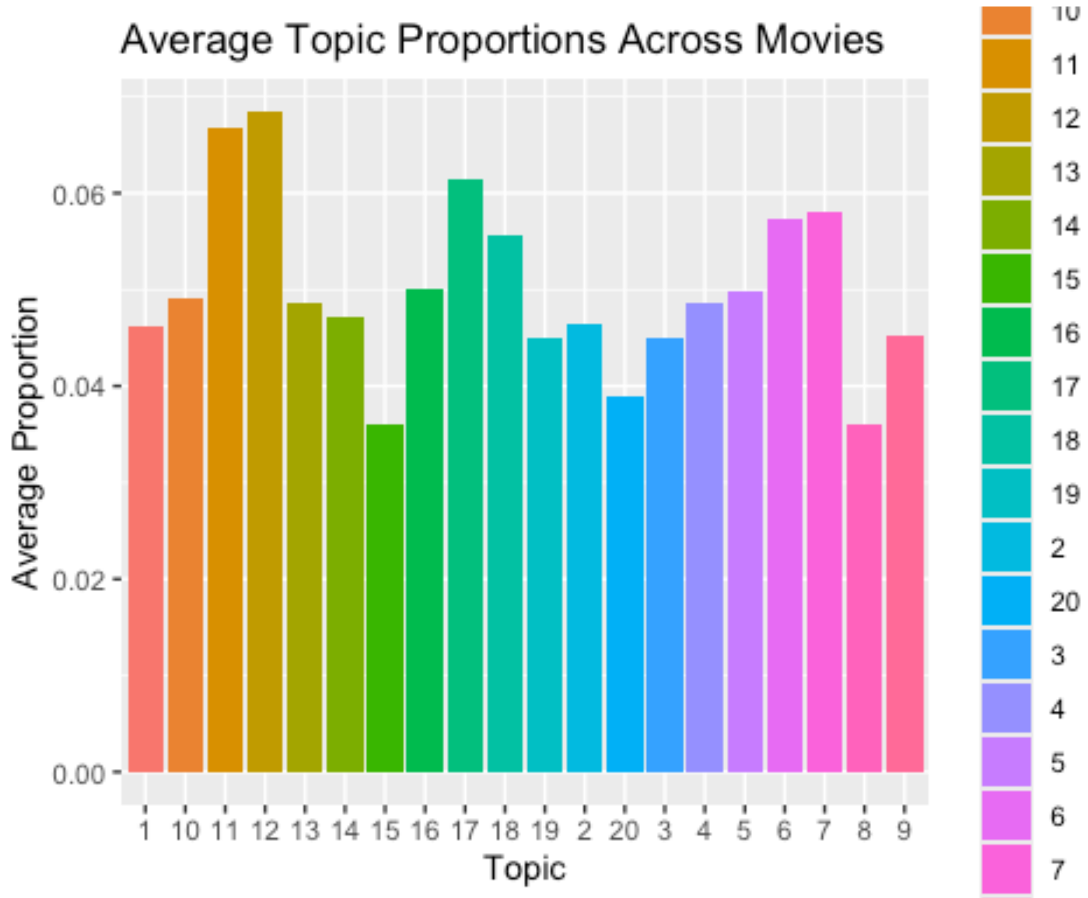


This shows us which clusters pertain the most to each individual topic on a broader level. (NoteL: I am not sure why topics are unordered)

We can also see the proportion of topics across movies

```
plots_gamma_wider %>%
  pivot_longer(cols = `1`:`20`, names_to = "topic", values_to = "gamma") %>%
# Pivot all at once
  group_by(topic) %>%
  summarise(avg_proportion = mean(gamma, na.rm = TRUE)) %>%
  ggplot(aes(x = topic, y = avg_proportion, fill = topic)) +
  geom_bar(stat = "identity", position = "dodge") +
```

```
  labs(title = "Average Topic Proportions Across Movies", x = "Topic", y =
"Average Proportion")
```



This shows the average proportion of each topic across all movies. Which helps to identify which topics are more or less prominent in the movie dataset