Topic Modeling

Author

Paul Moon

Introduction

In this project, we are assigned to look at data sets that have movie names, their genre, and their plots. Our goal is to organize them by topic and produce interesting graphs that can visually please the reader.

Movie Plots Cluster Plot

First, I will be doing this assignment by using the movie_plots csv in order to group movies by similarity in a cluster plot. This code might seen similar to you, Ajay, since this first part, I got from you. The first three chunks are what Jon Neimann sent me in order to help me jump start the project that was from you.

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

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)

plots_lda <- LDA(plots_dtm, k = 30, control = list(seed = 123))

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_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))
```

Movie Plots with Genre Cluster Plot

Now, that we have this cluster plot. we cannot just be done with this or there is nothing that I really did. Therefore, I will now try doing another cluster plot but this time by genre and how each genre plays a factor with how these movies are in relation to each other.

```
movie_genres <- read.csv("movie_plots_with_genres.csv")
```

After reading this csv, we will do very similar code to what was done above with some altercations.

```
#Same first part from the previous cluster plot
plots_by_word <- movie_genres %>%
  unnest_tokens(word, Plot) %>%
  anti_join(stop_words) %>%
  count(Movie.Name, Genre, word, sort = TRUE)

data("freq_first_names")
first_names <- tolower(freq_first_names$Name)
plots_by_word <- plots_by_word %>%
  filter(!(word %in% first_names))

plots_dtm <- plots_by_word %>%
  cast_dtm(Movie.Name, word, n)

lda_model <- LDA(plots_dtm, k = 10, control = list(seed = 123))

plots_gamma <- tidy(lda_model, matrix = "gamma") %>%
  pivot_wider(names_from = topic, values_from = gamma)

set.seed(123)
cluster_result <- kmeans(plots_gamma %>% select(-document), centers = 5)

plots_gamma$cluster <- factor(cluster_result$cluster)

fviz_cluster(cluster_result,
             data = plots_gamma %>% select(-document, -cluster),
             geom = "point",
             main = "Cluster Plot of Movies by Topic Distributions",
             labelsize = 4)
```

What we observed here is that the clustered looked a little different. However, we run into an issue. That it is indeed different. When we look at the two different data sets, the only thing that is different is the fact that there is an additional column “Genre”. Other than that, all the words are still the same and should result in similar topics. The conclusion is that the number of words in each “Genre” skewed the cluster plot into thinking that specific genres were more similar to each other, which in that case is what we wanted in the first place.

Movie Plots with Genre Beta Plot

Now, we will look at a Beta Plot. I think that this is a good way to look at the most common words visually.

```
#Restating this chunk to make it more organized
plots_by_word <- movie_genres %>%
  unnest_tokens(word, Plot) %>%
  anti_join(stop_words) %>%
  count(Movie.Name, Genre, word, sort = TRUE)

plots_dtm <- plots_by_word %>%
  cast_dtm(Movie.Name, word, n)

lda_model <- LDA(plots_dtm, k = 10, control = list(seed = 123))  # Adjust k as needed

beta_data <- tidy(lda_model, matrix = "beta")
#Creating a beta plot
beta_plot <- beta_data %>%
  group_by(topic) %>%
  #Using only 3 words due to space
  top_n(3, beta) %>%
  ungroup() %>%
  ggplot(aes(x = reorder_within(term, beta, topic), y = beta, fill = as.factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free_y", ncol = 2) +
  coord_flip() +
  labs(title = "Beta Plot: Top Terms in Each Topic",
       x = "Term",
       y = "Probability (Beta)") +
  scale_x_reordered() +
  theme_minimal() +
  theme(strip.text = element_text(size = 12),
        plot.margin = unit(c(1, 1, 1, 1), "cm"))

print(beta_plot)
```

I decided to only put in 3 words per topic. Honestly, the reason why I did this was because when I did more words, they overlapped each other and I had no idea how to make it smaller without basically making the words disappear. Therefore, the data is good but due to the lack of information, this graph MIGHT not be the best representation of the Beta Plot.

Movie Plots with Genre Scree Plot

Now, we look at the Scree Plot.

```
k_values <- seq(2, 20, by = 2)
log_likelihoods <- numeric(length(k_values))

for (i in seq_along(k_values)) {
  k <- k_values[i]
  lda_model <- LDA(plots_dtm, k = k, control = list(seed = 123))
  log_likelihoods[i] <- logLik(lda_model)
}

scree_data <- tibble(k = k_values, log_likelihood = log_likelihoods)
scree_plot <- scree_data %>%
  ggplot(aes(x = k, y = log_likelihood)) +
  geom_line() + geom_point() +
  labs(title = "Scree Plot for Optimal Number of Topics (k)",
       x = "Number of Topics (k)",
       y = "Log(Likelihood)") +
  theme_minimal()

print(scree_plot)
```

I don’t really know how this would help, but here is a good visualization on the logs of the topics and main themes.

Movie Plots with Genre Artsy-Looking Word Cloud

Now, I will create an artsy-looking word cloud using wordcloud2. Simple but interesting code.

```
library(wordcloud2)

#Creating the artsy-look thingy
top_terms <- betas %>%
  group_by(topic) %>%
  slice_max(beta, n =20) %>%
  ungroup() %>%
  count(term, wt = beta, sort = TRUE)

wordcloud2(top_terms, size = 1, color = "random-light")
```