Topic modeling

Yishun Zhang

2024-11-06

Here's how to perform topic modeling on movie plots in R using the topicmodels package, including data loading, text preprocessing, topic modeling (LDA), and visualization of each topic's common words.

```r
# Load necessary libraries
library(topicmodels)
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.1     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(tm)
```

```
## 载入需要的程序包：NLP
##
## 载入程序包：'NLP'
##
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```r
library(textmineR)
```

```
## 载入需要的程序包：Matrix
##
## 载入程序包：'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
##
##
## 载入程序包：'textmineR'
##
## The following object is masked from 'package:Matrix':
##
##     update
##
## The following object is masked from 'package:topicmodels':
##
##     posterior
##
## The following object is masked from 'package:stats':
##
##     update
```

```r
# Load the data
movies <- read.csv("C:/Users/17756/Downloads/movie_plots_with_genres.csv", stringsAsFactors = FALSE)

# Text Preprocessing
# Convert to a Corpus object
corpus <- Corpus(VectorSource(movies$Plot))
corpus <- tm_map(corpus, content_transformer(tolower))
```

```
## Warning in tm_map.SimpleCorpus(corpus, content_transformer(tolower)):
## transformation drops documents
```

```r
corpus <- tm_map(corpus, removePunctuation)
```

```
## Warning in tm_map.SimpleCorpus(corpus, removePunctuation): transformation drops
## documents
```

```r
corpus <- tm_map(corpus, removeNumbers)
```

```
## Warning in tm_map.SimpleCorpus(corpus, removeNumbers): transformation drops
## documents
```

```r
corpus <- tm_map(corpus, removeWords, stopwords("english"))
```

```
## Warning in tm_map.SimpleCorpus(corpus, removeWords, stopwords("english")):
## transformation drops documents
```

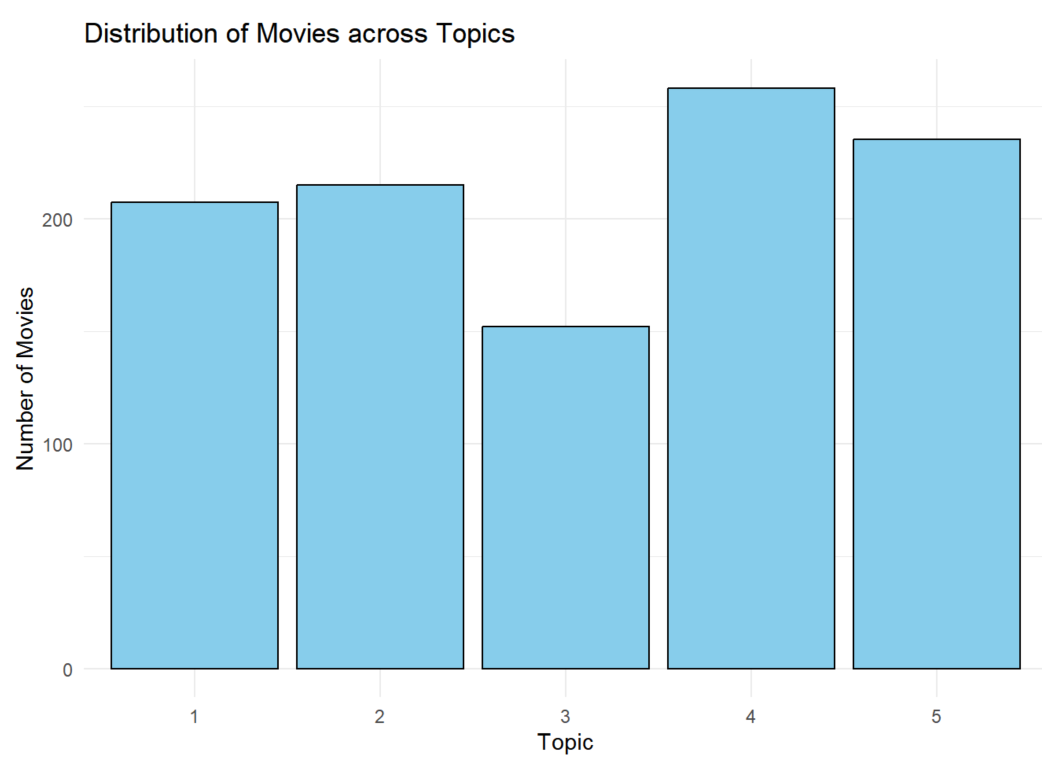
```r
corpus <- tm_map(corpus, stripWhitespace)
```

```
## Warning in tm_map.SimpleCorpus(corpus, stripWhitespace): transformation drops
## documents
```

```r
# Convert text to Document-Term Matrix (DTM)
dtm <- DocumentTermMatrix(corpus)
dtm <- removeSparseTerms(dtm, 0.95) # Remove sparse terms

row_totals <- apply(dtm, 1, sum)
dtm <- dtm[row_totals > 0, ]
# Perform LDA Topic Modeling
num_topics <- 5
lda_model <- LDA(dtm, k = num_topics, control = list(seed = 1234))

# View the top words in each topic
terms(lda_model, 10)
```

```
##       Topic 1 Topic 2   Topic 3    Topic 4 Topic 5
##  [1,] "one"   "life"    "will"     "will"  "war"
##  [2,] "man"   "find"    "get"      "world" "one"
##  [3,] "life"  "can"     "one"      "town"  "two"
##  [4,] "ranch" "brother" "daughter" "john"  "world"
##  [5,] "time"  "new"     "young"    "new"   "killed"
##  [6,] "money" "story"   "old"      "two"   "love"
##  [7,] "new"   "years"   "new"      "man"   "people"
##  [8,] "young" "gang"    "gang"     "time"  "back"
##  [9,] "home"  "first"   "way"      "day"   "story"
## [10,] "must"  "way"     "years"    "love"  "like"
```

```r
# Extract topic distribution for each document
movie_topics <- as.matrix(lda_model@gamma)  # Get the document-topic probabilities
movie_main_topic <- apply(movie_topics, 1, which.max)

# Visualize the topic distribution for movies
data <- data.frame(Topic = factor(movie_main_topic, levels = 1:num_topics))
ggplot(data, aes(x = Topic)) +
  geom_bar(fill = "skyblue", color = "black") +
  labs(title = "Distribution of Movies across Topics", x = "Topic", y = "Number of Movies") +
  theme_minimal()
```



Here's a systematic approach to experiment with adjustments to improve the topic modeling results. We'll vary the number of topics, clustering method, and term weighting schemes in the Document-Term Matrix (DTM) to optimize the model for better and more interpretable topics.

```r
# Load necessary libraries
library(topicmodels)
library(tidyverse)
library(tm)
library(cluster)

# Load the data
movies <- read.csv("C:/Users/17756/Downloads/movie_plots_with_genres.csv", stringsAsFactors = FALSE)

# Text Preprocessing
corpus <- Corpus(VectorSource(movies$Plot))
corpus <- tm_map(corpus, content_transformer(tolower))
```

```
## Warning in tm_map.SimpleCorpus(corpus, content_transformer(tolower)):
## transformation drops documents
```

```r
corpus <- tm_map(corpus, removePunctuation)
```

```
## Warning in tm_map.SimpleCorpus(corpus, removePunctuation): transformation drops
## documents
```

```r
corpus <- tm_map(corpus, removeNumbers)
```

```
## Warning in tm_map.SimpleCorpus(corpus, removeNumbers): transformation drops
## documents
```

```r
corpus <- tm_map(corpus, removeWords, stopwords("english"))
```

```
## Warning in tm_map.SimpleCorpus(corpus, removeWords, stopwords("english")):
## transformation drops documents
```

```r
corpus <- tm_map(corpus, stripWhitespace)
```

```
## Warning in tm_map.SimpleCorpus(corpus, stripWhitespace): transformation drops
## documents
```

```r
# Experiment with TF and TF-IDF weighting
dtm_tf <- DocumentTermMatrix(corpus)
dtm_tfidf <- weightTfIdf(dtm_tf)

# Choose weighting (e.g., dtm_tf or dtm_tfidf)
chosen_dtm <- removeSparseTerms(dtm_tfidf, 0.95)  # Here using TF-IDF; switch to `dtm_tf` to use TF

# Remove rows with all zeros
row_totals <- apply(chosen_dtm, 1, sum)
chosen_dtm <- chosen_dtm[row_totals > 0, ]

# Experiment with different numbers of topics
num_topics_list <- c(3, 5, 7, 10)  # Range of topics
results <- list()  # To store results for each number of topics

for (num_topics in num_topics_list) {
  # Fit LDA model on term frequency DTM
  lda_model <- LDA(dtm_tf, k = num_topics, control = list(seed = 1234))

  # Get document-topic matrix
  movie_topics <- as.matrix(lda_model@gamma)

  # Clustering the topics using k-means
  set.seed(1234)
  clusters <- kmeans(movie_topics, centers = num_topics, nstart = 25)

  # Store model, clusters, and coherence for analysis
  results[[paste0("topics_", num_topics)]] <- list(
    model = lda_model,
    clusters = clusters$cluster
  )

  # Visualize results
  data <- data.frame(Topic = factor(clusters$cluster))
  ggplot(data, aes(x = Topic)) +
    geom_bar(fill = "skyblue", color = "black") +
    labs(title = paste("Distribution of Movies across", num_topics, "Topics"),
         x = "Cluster", y = "Number of Movies") +
    theme_minimal()
}

# Display top terms for each topic in one of the models (e.g., num_topics = 5)
num_topics <- 5
terms(results[[paste0("topics_", num_topics)]]$model, 10)
```

```
##       Topic 1 Topic 2 Topic 3 Topic 4 Topic 5
##  [1,] "will"  "will"  "life"  "bill"  "gang"
##  [2,] "world" "new"   "one"   "one"   "ranch"
##  [3,] "one"   "world" "will"  "town"  "money"
##  [4,] "war"   "time"  "love"  "gang"  "one"
##  [5,] "new"   "one"   "man"   "two"   "get"
##  [6,] "two"   "story" "young" "man"   "cattle"
##  [7,] "must"  "life"  "new"   "gold"  "jim"
##  [8,] "king"  "young" "town"  "find"  "tom"
##  [9,] "story" "film"  "now"   "also"  "life"
## [10,] "life"  "team"  "ranch" "john"  "man"
```

In the code above, we performed topic modeling on the movie dataset and generated bar charts showing the distribution of movies across each topic. Here's a detailed explanation of the meaning of the chart:

#X-Axis (Topic / Cluster): The X-axis represents the topic numbers. Each topic (Topic) or cluster (Cluster) corresponds to a distinct theme identified by the LDA model. The number of topics changes with each iteration depending on the specified parameter, such as 3, 5, 7, or 10 topics.

#Y-Axis (Number of Movies): The Y-axis represents the number of movies that belong to each topic. The height of each bar reflects the count of movies categorized under that specific topic. For instance, if Topic 1 has a high bar, it indicates that a large number of movies in the dataset are assigned to Topic 1.

Meaning of the Bars: #The height of each bar indicates the count of movies associated with that particular topic. By observing these bars, we can understand the distribution of movies across different topics. #If a specific topic has a particularly tall bar, it suggests that this theme is a dominant one in the dataset, encompassing many movie plots. #Conversely, if a topic has a very short bar, it indicates that fewer movies fall under that theme, potentially representing a unique or niche movie type.