Amazon_Book_Review_30074741

2024-01-23

Load necessary packages

- Firstly Install Libraries

```
libraries <- c("tm", "tidytext", "ggplot2", "wordcloud", "syuzhet", "dplyr", "magrittr", "tib
ble", "textstem", "textdata", "tidyr","stringr","reshape2", "LDAvis", "jsonlite","RColorBrewe
r", "sentimentr","Matrix", "topicmodels", "stm")

 #install.packages(libraries) # Comment out after first execution


for (lib in libraries) {
  library(lib, character.only=TRUE) #Library takes function names without quotes, character o
nly must be used in a loop of this kind.
}
```

Package Information

- `tm` :
  - **Purpose:** Provides a framework for text mining applications within R.
  - **Key Features:** Offers tools for importing, managing, and transforming text data.
- `tidytext` :
  - **Purpose:** Integrates text mining with the `tidyverse` approach in R.
  - **Key Features:** Simplifies the process of text analysis and manipulation using tidy data principles.
- `ggplot2` :
  - **Purpose:** A system for declaratively creating graphics, based on the Grammar of Graphics.
  - **Key Features:** Enables the creation of complex and aesthetically pleasing data visualizations.
- `wordcloud` :
  - **Purpose:** For generating word cloud visualizations.
  - **Key Features:** Provides a visual representation of text data, highlighting the most frequent or important words.
- `syuzhet` :
  - **Purpose:** Designed for sentiment analysis and extracting narrative arcs from textual data.
  - **Key Features:** Offers tools for sentiment extraction using various established lexicons.
- `dplyr` :
  - **Purpose:** A grammar of data manipulation, providing a consistent set of verbs for data manipulation tasks.
  - **Key Features:** Includes functions for filtering, selecting, mutating, summarizing, and arranging data, optimized for performance and usability.
- `tibble` :
  - **Purpose:** A modern re imagining of the data frame in R, enhancing usability and integration with `tidyverse` packages.
  - **Key Features:** Offers enhanced printing, non-altering of string variables and variable names, consistent data type maintenance in subsetting, and omission of row names for simplicity.
- `textstem` :

  - **Purpose:** Provides comprehensive tools for text preprocessing, including lemmatization and stemming.
  - **Key Features:** Offers advanced text normalization capabilities, suitable for preparing text data for various natural language processing tasks.
- `textdata`：
  - **Purpose:** Streamlines the process of downloading, parsing, and loading various text datasets commonly used in text analysis and natural language processing.
  - **Key Features:** Provides an easy and standardized way to access a variety of text datasets, including sentiment lexicons, word embeddings, and other language resources.
- `tidyr`：
  - **Purpose:** Designed to help tidy data, which means making it suitable for analysis by restructuring it into a consistent format.
  - **Key Features:** Includes functions like `gather()`, `spread()`, `pivot_longer()`, and `pivot_wider()` that transform data frames to and from long and wide formats, simplifying many common data reshaping operations.

```
filepath <- 'C:\\Users\\DELL\\OneDrive - University of South Wales\\Desktop\\Data_Mining_Asse
ssment\\MS4S09_CW_Book_Reviews.csv' # Define file path. Windows requires \ to be replaced by
\\. / Works on Mac (apparently).
bookdata <- as_tibble(read.csv(filepath, stringsAsFactors = FALSE)) # Since we have text data
we do not want this read as a factor

# Inspect summary and first few rows of data
print(summary(bookdata))
```

```
##     Title             Book_Price     Reviewer_id            Rating
##  Length:59296       Min.   :  1.00   Length:59296       Min.   :1.000
##  Class :character   1st Qu.: 10.36   Class :character   1st Qu.:4.000
##  Mode  :character   Median : 14.15   Mode  :character   Median :5.000
##                     Mean   : 20.81                      Mean   :4.231
##                     3rd Qu.: 22.99                      3rd Qu.:5.000
##                     Max.   :995.00                      Max.   :5.000
##       Time           Review_title       Review_text        Found_helpful_ratio
##  Min.   :8.688e+08   Length:59296       Length:59296       Min.   :0.0000
##  1st Qu.:1.087e+09   Class :character   Class :character   1st Qu.:0.0000
##  Median :1.169e+09   Mode  :character   Mode  :character   Median :0.6667
##  Mean   :1.173e+09                                         Mean   :0.5491
##  3rd Qu.:1.279e+09                                         3rd Qu.:1.0000
##  Max.   :1.362e+09                                         Max.   :1.0000
##   Publisher         First_author          Genre
##  Length:59296       Length:59296       Length:59296
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
```

```
print(head(bookdata))
```

```
## # A tibble: 6 × 11
##   Title            Book_Price Reviewer_id Rating   Time Review_title Review_text
##   <chr>                 <dbl> <chr>        <int>  <int> <chr>        <chr>
## 1 In Six Days: Wh…      10.2  APD7XINUVG…      4 9.99e8 Solid testi… "Working f…
## 2 Lord Jim              15.6  AITANZIKX8…      5 9.16e8 &quot;You d… "A terrifi…
## 3 White Socks Only       5.16 AYB19RB36G…      4 1.34e9 White Socks… "As I open…
## 4 The Secret of t…      15.0  A1BOLCK0Q5…      5 1.28e9 great!       "Excellent…
## 5 Left to Tell: D…      17.5  A3WKJ88K78…      5 1.36e9 Great Book   "This book…
## 6 Don't Make Me T…      20.9  A1E6I4IPWW…      5 1.20e9 Don't think… "Excellent…
## # ℹ 4 more variables: Found_helpful_ratio <dbl>, Publisher <chr>,
## #   First_author <chr>, Genre <chr>
```

Data Selection and Sampling

Text mining is often very computationally expensive, and the process of tokenizations greatly increases the number of rows of data we need to process. To combat this we will take a small sample of the data.

```
# Check the number of columns in your dataframe
ncol_bookdata <- ncol(bookdata)
print(ncol_bookdata)
```

```
## [1] 11
```

```
# Selecting columns for analysis
if (ncol(bookdata) >= 11) {
  # If the dataframe has at least 11 columns, select specific columns
  bookdata_selected <- bookdata[, c(1, 2, 4, 6, 7, 11)] # Selected columns: "Title", "Book_Pr
ice", "Rating", "Review_title", "Review_text", "Genre"
} else {
  # If the dataframe doesn't have enough columns, notify the user
  print("Columns required for analysis do not exist.")
}

# Remove rows with missing values
bookdata <- na.omit(bookdata_selected)

# Add a unique identifier column to reviews
bookdata$Reviewer_no <- 1:nrow(bookdata)

# Print the cleaned dataframe
print(bookdata)
```

```
## # A tibble: 59,296 × 7
##    Title            Book_Price Rating Review_title Review_text Genre Reviewer_no
##    <chr>                 <dbl>  <int> <chr>        <chr>       <chr>       <int>
##  1 In Six Days: Wh…      10.2       4 Solid testi… "Working f… Reli…           1
##  2 Lord Jim              15.6       5 &quot;You d… "A terrifi… Fict…           2
##  3 White Socks Only       5.16      4 White Socks… "As I open… Juve…           3
##  4 The Secret of t…      15.0       5 great!       "Excellent… Reli…           4
##  5 Left to Tell: D…      17.5       5 Great Book   "This book… Biog…           5
##  6 Don't Make Me T…      20.9       5 Don't think… "Excellent… Comp…           6
##  7 Eldest (Inherit…      34.0       5 A good book… "I've neve… Juve…           7
##  8 Search Engine V…      32.8       5 Finally! An… "Many peop… Comp…           8
##  9 Getting to Know…      32.2       5 Excellent f… "This is t… Comp…           9
## 10 Life is tough a…      11.3       5 It's sad, i… "I love th… Educ…          10
## # ℹ 59,286 more rows
```

```
# Count the number of reviews by title and sort in descending order
title_reading <- table(bookdata$Title)

# Sort the titles by review counts
title_reading <- sort(title_reading, decreasing = TRUE)

# Print the top 5 most reviewed titles
head(title_reading)
```

```
##
##                                                Eldest (Inheritance, Book 2)
##                                                                         276
##                                                          Great Expectations
##                                                                         263
##                                                                    Hannibal
##                                                                         258
##                                                               Good to Great
##                                                                         203
##                      The Five Love Languages: The Secret to Love that Lasts
##                                                                         201
## Love & Respect: The Love She Most Desires; The Respect He Desperately Needs
##                                                                         144
```

The titles were reviewed multiple times, so will select all titles with 144 reviews and above for analysis.

```
# Filter titles with 144 reviews and above
filtered_title_counts <- count(bookdata, Title) %>%
  filter(n >= 144)

# Select titles with 144 reviews and above
selected_titles <- filtered_title_counts$Title

# Filter dataframe to include only rows with selected titles
bookdata <- bookdata[bookdata$Title %in% selected_titles, ]

# Print summary statistics of the filtered dataframe
cat("Summary Statistics of Filtered Dataframe:\n")
```

```
## Summary Statistics of Filtered Dataframe:
```

```
print(summary(bookdata))
```

```
##     Title             Book_Price        Rating      Review_title
##  Length:1345        Min.   : 7.67   Min.   :1.000   Length:1345
##  Class :character   1st Qu.:13.64   1st Qu.:3.000   Class :character
##  Mode  :character   Median :19.25   Median :5.000   Mode  :character
##                     Mean   :20.52   Mean   :3.993
##                     3rd Qu.:26.95   3rd Qu.:5.000
##                     Max.   :33.97   Max.   :5.000
##  Review_text           Genre            Reviewer_no
##  Length:1345        Length:1345        Min.   :    7
##  Class :character   Class :character   1st Qu.:15299
##  Mode  :character   Mode  :character   Median :29582
##                                        Mean   :29559
##                                        3rd Qu.:44009
##                                        Max.   :59226
```

```
# Display the first few rows of the filtered dataframe
cat("First Few Rows of Filtered Dataframe:\n")
```

```
## First Few Rows of Filtered Dataframe:
```

```
print(head(bookdata))
```

```
## # A tibble: 6 × 7
##   Title             Book_Price Rating Review_title Review_text Genre Reviewer_no
##   <chr>                  <dbl>  <int> <chr>        <chr>       <chr>       <int>
## 1 Eldest (Inherita…      34.0       5 A good book… "I've neve… Juve…           7
## 2 Hannibal                7.67      1 So sorry yo… "What a mo… Fict…          11
## 3 Good to Great          27.0       5 Good to Gre… "An excell… Busi…          35
## 4 Eldest (Inherita…      34.0       4 Love it for… "I'm 37 an… Juve…         372
## 5 Love & Respect: …      13.6       5 Love this b… "Love & Re… Psyc…         426
## 6 Hannibal                7.67      5 breath-taki… "This book… Fict…         440
```

Tokenize Reviews

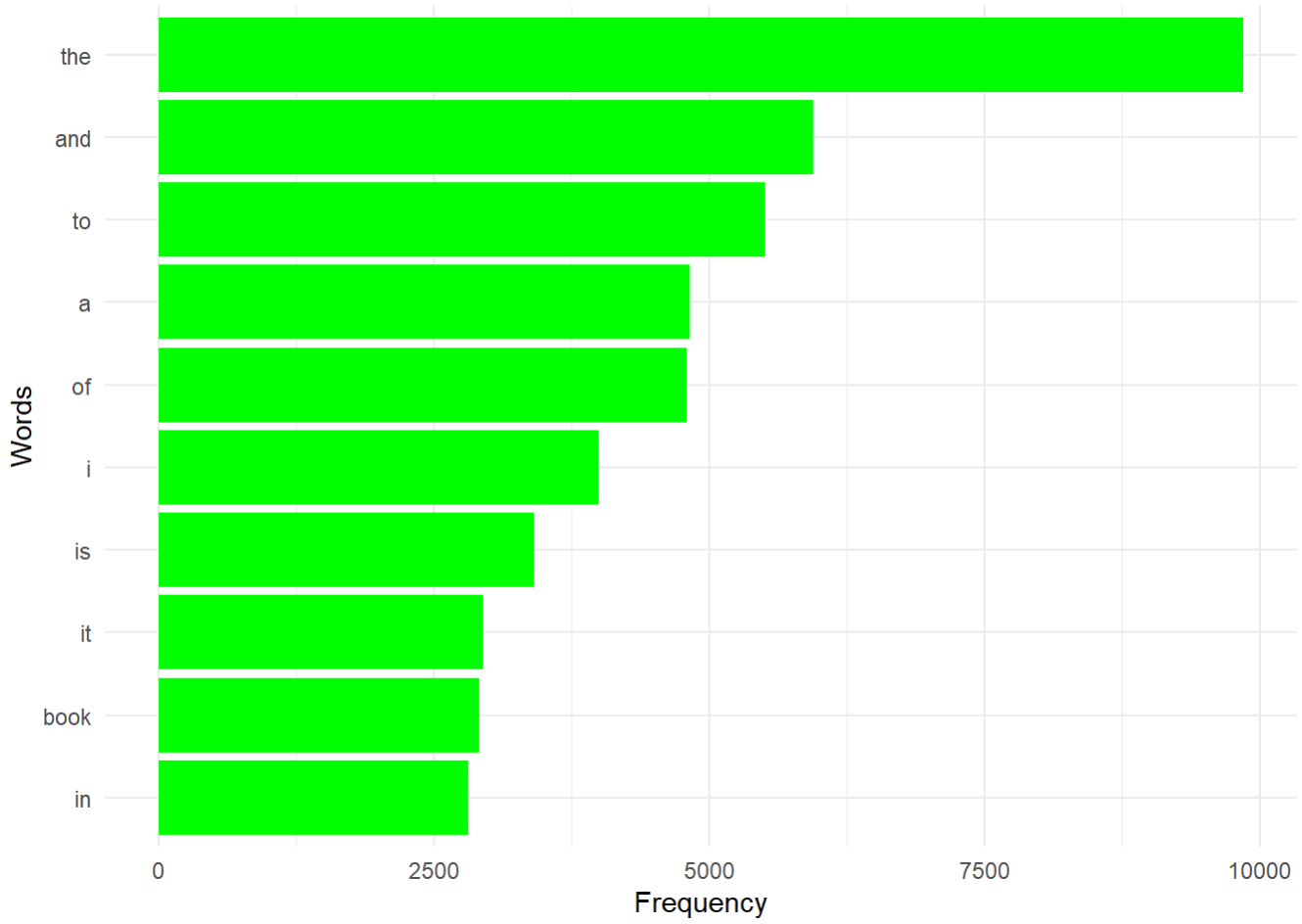
Now that our data is sampled, we can begin to perform some analysis on the reviews. We will begin by tokenizing our data.

```
# Tokenize the review text column into individual words
word_tokenized_data <- bookdata %>%
  unnest_tokens(output = word, input = Review_text, token = "words", to_lower = TRUE)

# Tokenize the review text column into bigrams (pairs of consecutive words)
bigram_tokenized_data <- bookdata %>%
  unnest_tokens(output = bigram, input = Review_text, token = "ngrams", n = 2, to_lower = TRU
E)
```

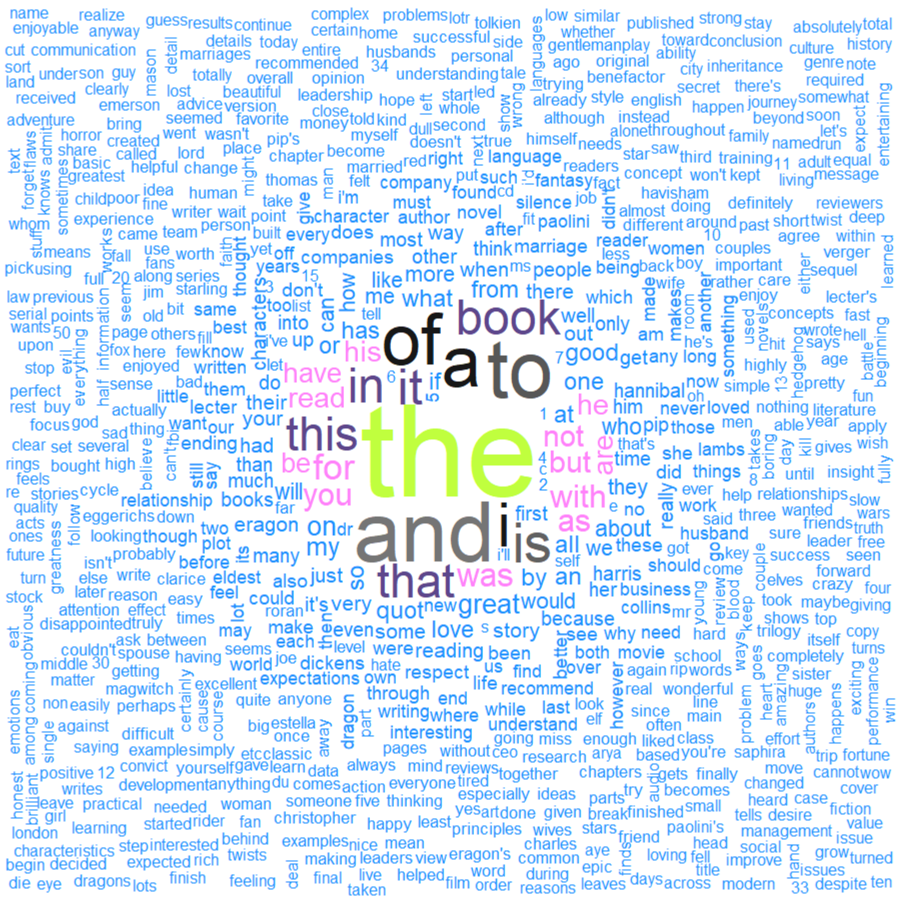
Initial Exploratory Analysis

We can perform some initial exploratory analysis to see the most common words and bigrams in our reviews.

```
# Count the occurrences of each word and sort in descending order
word_counts <- word_tokenized_data %>%
  count(word, sort = TRUE)

# Plot the frequency of the top 10 most common words
ggplot(word_counts[1:10, ], aes(x = reorder(word, n), y = n)) +
  geom_col(fill = "Green") +
  labs(x = "Words", y = "Frequency") +
  coord_flip() +
  theme_minimal()
```

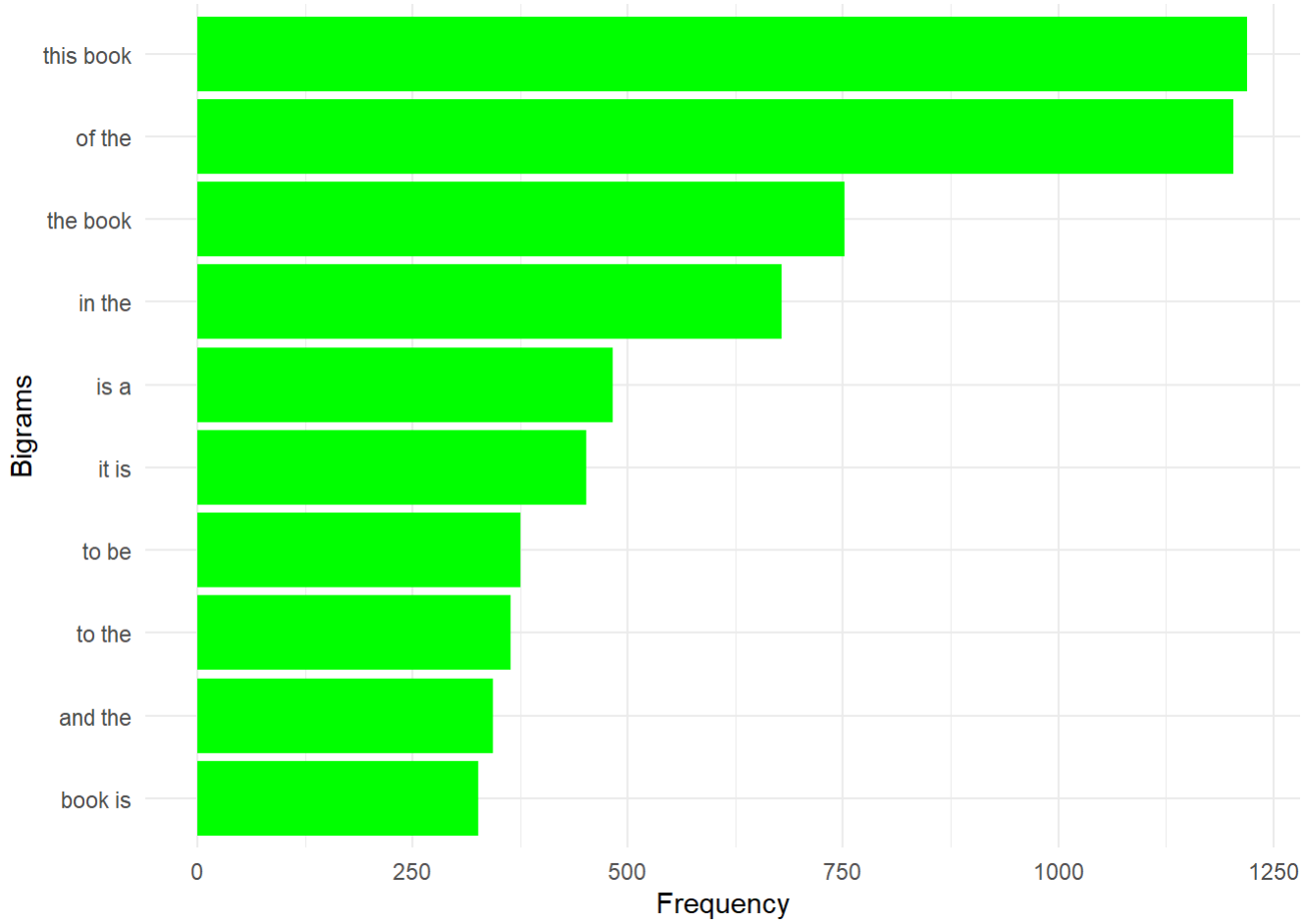


```
# Set a random seed for reproducibility
set.seed(1)

# Suppress warnings during word cloud generation
suppressWarnings({
  # Generate a word cloud with specified parameters
  wordcloud(
    words = word_counts$word,
    freq = word_counts$n,
    min.freq = 10,
    random.order = FALSE,
    random.color = FALSE,
    colors = sample(colors(), size = 10)
  )
})
```




```
# words = vector of words, freq = vector of frequencies, min.freq = minimum frequency to plo
t, random.order=FALSE means words are plotted in order of n, random.color=FALSE colors accord
ing to frequency and colors key word specifies colors to use.
```

```
# Count the occurrences of each bigram and sort in descending order
bigram_counts <- bigram_tokenized_data %>%
  count(bigram, sort = TRUE)

# Plot the frequency of the top 10 most common bigrams
ggplot(bigram_counts[1:10, ], aes(x = reorder(bigram, n), y = n)) +
  geom_col(fill = "Green") +
  labs(x = "Bigrams", y = "Frequency") +
  coord_flip() +
  theme_minimal()
```




```
# Set a random seed for reproducibility
set.seed(1)

# Suppress warnings during word cloud generation
suppressWarnings({
  # Generate a word cloud with specified parameters
  wordcloud(
    words = bigram_counts$bigram,
    freq = bigram_counts$n,
    min.freq = 10,
    random.order = FALSE,
    random.color = FALSE,
    colors = sample(colors(), size = 10)
  )
})
```

Cleaning Data

Initial analysis revealed that a lot of filler or “stop” words are present and dominating the visualisations we will perform cleaning that does the following: - Removes stop words - Removes special characters and numbers - lemmatizes words - Removes empty tokens

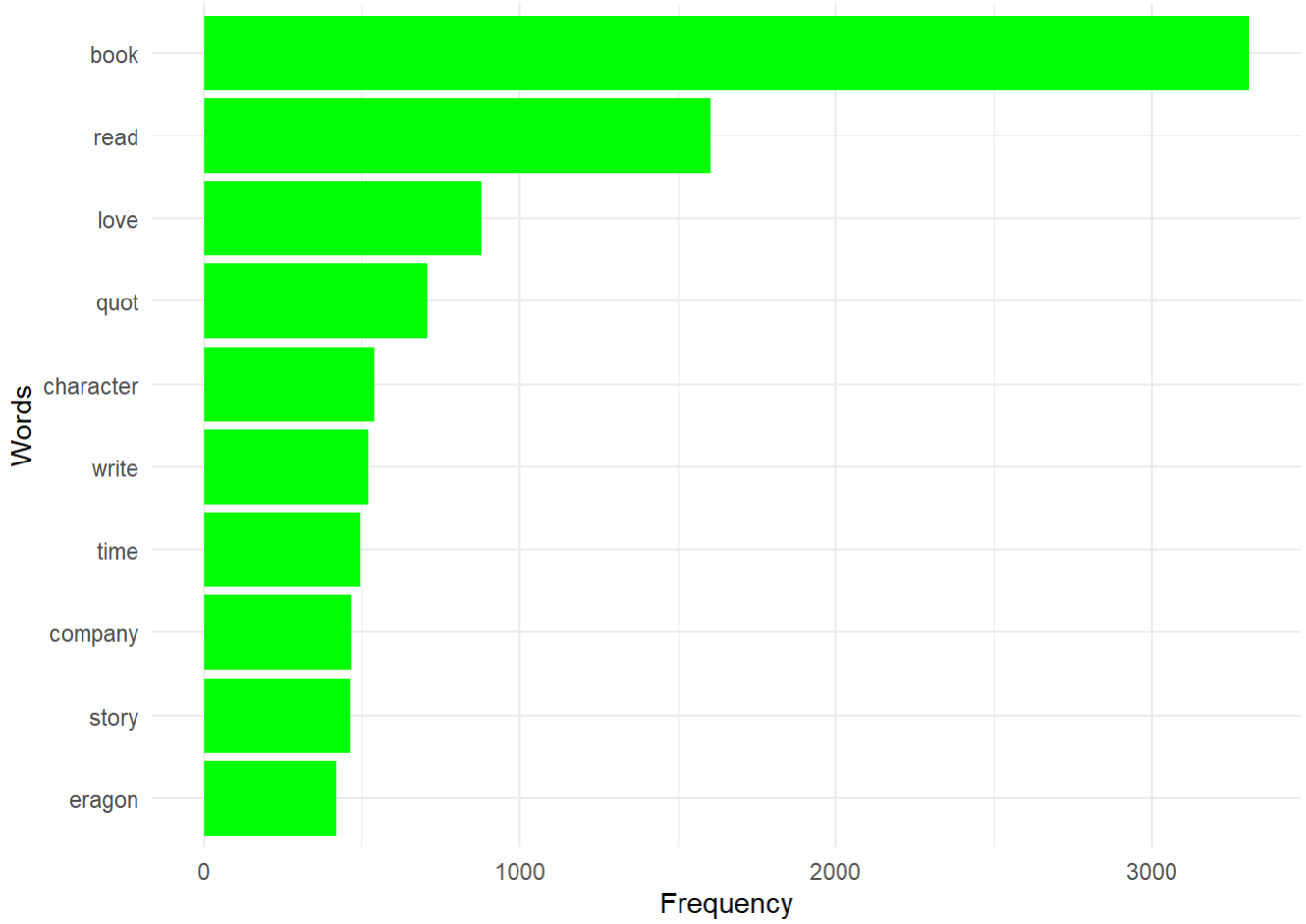
```
# Remove stop words from the word tokenized data
clean_tokens <- word_tokenized_data %>%
  anti_join(stop_words, by = "word")

# Remove special characters and numbers from the word column
clean_tokens$word <- gsub("[^a-zA-Z ]", "", clean_tokens$word) %>%
  na_if("") %>%
  lemmatize_words()

# Remove rows with NA values
clean_tokens <- na.omit(clean_tokens)

# Group by Reviewer_no, concatenate cleaned words into sentences, and join with original data
frame
untokenized_data <- clean_tokens %>%
  group_by(Reviewer_no) %>%
  summarize(clean_review = paste(word, collapse = " ")) %>%
  inner_join(bookdata[, c(1, 2, 3, 4, 6, 7)], by = "Reviewer_no")

# Tokenize the cleaned review text into bigrams
clean_bigrams <- untokenized_data %>%
  unnest_tokens(output = bigram, input = clean_review, token = "ngrams", n = 2, to_lower = TR
UE)

# Print cleaned word tokens and bigrams
print(clean_tokens)
```

```
## # A tibble: 70,025 × 7
##    Title                  Book_Price Rating Review_title Genre Reviewer_no word
##    <chr>                       <dbl>  <int> <chr>        <chr>       <int> <chr>
##  1 Eldest (Inheritance, …       34.0      5 A good book… Juve…           7 watch
##  2 Eldest (Inheritance, …       34.0      5 A good book… Juve…           7 star
##  3 Eldest (Inheritance, …       34.0      5 A good book… Juve…           7 war
##  4 Eldest (Inheritance, …       34.0      5 A good book… Juve…           7 read
##  5 Eldest (Inheritance, …       34.0      5 A good book… Juve…           7 lord
##  6 Eldest (Inheritance, …       34.0      5 A good book… Juve…           7 ring
##  7 Eldest (Inheritance, …       34.0      5 A good book… Juve…           7 idea
##  8 Eldest (Inheritance, …       34.0      5 A good book… Juve…           7 seri…
##  9 Eldest (Inheritance, …       34.0      5 A good book… Juve…           7 rela…
## 10 Eldest (Inheritance, …       34.0      5 A good book… Juve…           7 enjoy
## # ℹ 70,015 more rows
```

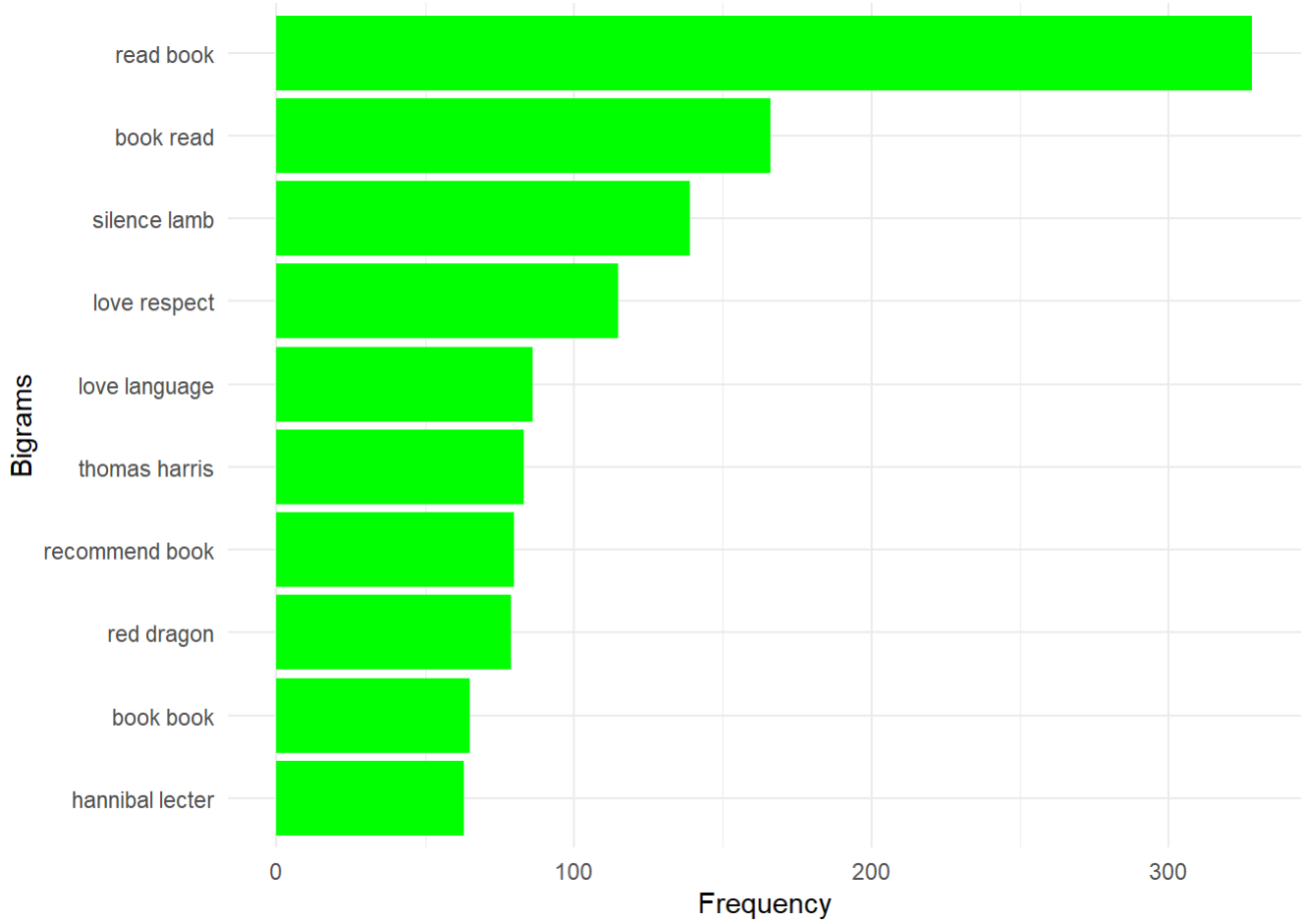
```
print(clean_bigrams)
```

```
## # A tibble: 68,694 × 7
##    Reviewer_no Title                 Book_Price Rating Review_title Genre bigram
##          <int> <chr>                      <dbl>  <int> <chr>        <chr> <chr>
##  1           7 Eldest (Inheritance,…       34.0      5 A good book… Juve… watch…
##  2           7 Eldest (Inheritance,…       34.0      5 A good book… Juve… star …
##  3           7 Eldest (Inheritance,…       34.0      5 A good book… Juve… war r…
##  4           7 Eldest (Inheritance,…       34.0      5 A good book… Juve… read …
##  5           7 Eldest (Inheritance,…       34.0      5 A good book… Juve… lord …
##  6           7 Eldest (Inheritance,…       34.0      5 A good book… Juve… ring …
##  7           7 Eldest (Inheritance,…       34.0      5 A good book… Juve… idea …
##  8           7 Eldest (Inheritance,…       34.0      5 A good book… Juve… serie…
##  9           7 Eldest (Inheritance,…       34.0      5 A good book… Juve… relat…
## 10           7 Eldest (Inheritance,…       34.0      5 A good book… Juve… enjoy…
## # ℹ 68,684 more rows
```

```
# Count the occurrences of each cleaned word and sort in descending order
word_counts <- clean_tokens %>%
  count(word, sort = TRUE)

# Select the top 10 most frequent words
top_words <- top_n(word_counts, 10, n)$word

# Filter word counts to include only the top words and reorder them for plotting
filtered_word_counts <- filter(word_counts, word %in% top_words)
filtered_word_counts$word <- factor(filtered_word_counts$word, levels = top_words[length(top_
words):1])

# Plot the frequency of the top 10 most common cleaned words
ggplot(filtered_word_counts, aes(x = reorder(word, n), y = n)) +
  geom_col(fill = "Green") +
  labs(x = "Words", y = "Frequency") +
  coord_flip() +
  theme_minimal()
```




```
# Count the occurrences of each cleaned bigram and sort in descending order
bigram_counts <- clean_bigrams %>%
  count(bigram, sort = TRUE)

# Select the top 10 most frequent bigrams
top_bigrams <- top_n(bigram_counts, 10, n)$bigram

# Filter bigram counts to include only the top bigrams and reorder them for plotting
filtered_bigram_counts <- filter(bigram_counts, bigram %in% top_bigrams)
filtered_bigram_counts$bigram <- factor(filtered_bigram_counts$bigram, levels = top_bigrams[l
ength(top_bigrams):1])

# Plot the frequency of the top 10 most common cleaned bigrams
ggplot(filtered_bigram_counts, aes(x = reorder(bigram, n), y = n)) +
  geom_col(fill = "Green") +
  labs(x = "Bigrams", y = "Frequency") +
  coord_flip() +
  theme_minimal()
```



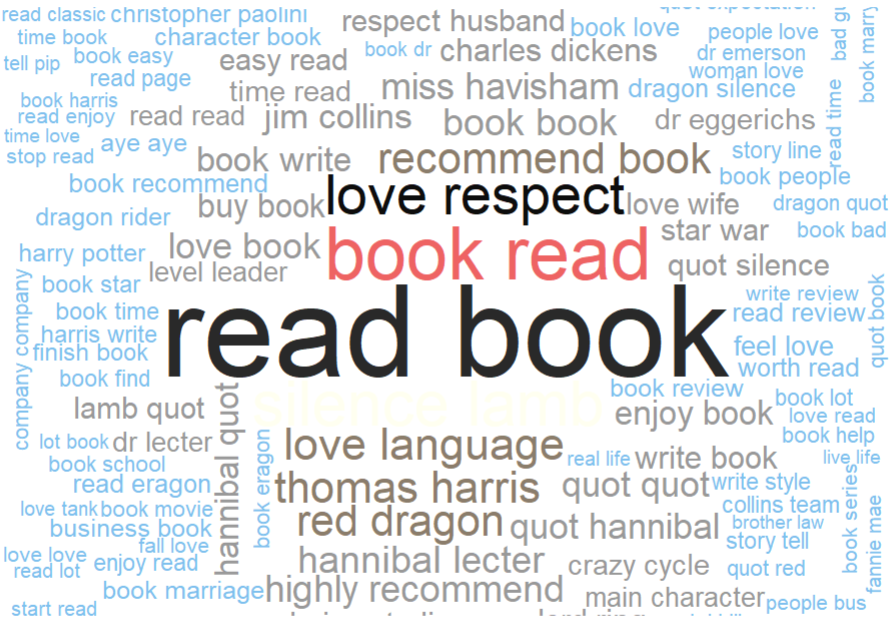
Top 10 Words & Bigrams grouped by title

```
# Select the top 10 words
top_words <- top_n(word_counts, 10, n)$word

# Group clean_tokens by title and count the occurrences of each word, then filter to include
only the top 10 words
grouped_count <- clean_tokens %>%
  group_by(Title) %>%
  count(word) %>%
  filter(word %in% top_words)

# Order the top words according to overall frequency
grouped_count$word <- factor(grouped_count$word, levels = top_words[length(top_words):1])

# Plot grouped bar chart showing the frequency of top words for each title
ggplot(data = grouped_count, aes(x = word, y = n, fill = Title)) +
  geom_col(position = "dodge") +
  labs(x = "Words", y = "Frequency", fill = "Title") +
  coord_flip() +
  theme_minimal()
```


```
# Select the top 10 bigrams
top_bigrams <- top_n(bigram_counts, 10, n)$bigram

# Group clean_bigrams by title and count the occurrences of each bigram, then filter to inclu
de only the top 10 bigrams
grouped_count <- clean_bigrams %>%
  group_by(Title) %>%
  count(bigram) %>%
  filter(bigram %in% top_bigrams)

# Order the top bigrams according to overall frequency
grouped_count$bigram <- factor(grouped_count$bigram, levels = top_bigrams[length(top_bigram
s):1])

# Plot grouped bar chart showing the frequency of top bigrams for each title
ggplot(data = grouped_count, aes(x = bigram, y = n, fill = Title)) +
  geom_col(position = "dodge") +
  labs(x = "Bigrams", y = "Frequency", fill = "Title") +
  coord_flip() +
  theme_minimal()
```


```
suppressWarnings({
  # Set a random seed for reproducibility
  set.seed(1)

  # Generate a word cloud for individual words
  wordcloud(words = word_counts$word,
            freq = word_counts$n,
            min.freq = 10,
            random.order = FALSE,
            random.color = FALSE,
            colors = sample(colors(), size = 10))
})
```


```
suppressWarnings({
  # Generate a word cloud for bigrams
  wordcloud(words = bigram_counts$bigram,
            freq = bigram_counts$n,
            min.freq = 10,
            random.order = FALSE,
            random.color = FALSE,
            colors = sample(colors(), size = 10))
})
```



Sentiment Analysis

Following initial exploration we can perform sentiment analysis on the data to get some deeper insights. We will apply bing lexicon: - bing

The bing lexicon assigns words either positive or negative sentiment

```
# Load the Bing sentiment lexicon
bing_sentiments <- get_sentiments("bing")

# Display summary statistics of the Bing sentiment lexicon
summary(bing_sentiments)
```

```
##      word            sentiment
##  Length:6786        Length:6786
##  Class :character   Class :character
##  Mode  :character   Mode  :character
```

```
# Print unique sentiment categories in the Bing lexicon
print(unique(bing_sentiments$sentiment))
```

```
## [1] "negative" "positive"
```

```
# Set a random seed for reproducibility
set.seed(2)

# Return a sample of 5 rows from the Bing sentiment lexicon
bing_sentiments[sample(nrow(bing_sentiments), 5),]
```

```
## # A tibble: 5 × 2
##   word         sentiment
##   <chr>        <chr>
## 1 maladjusted  negative
## 2 retaliatory  negative
## 3 punitive     negative
## 4 happiness    positive
## 5 peacekeepers positive
```

Words having sentiment values between -5 and +5 are given a score in the AFINN lexicon. Positive and negative numbers represent positive and negative sentiment, respectively, while the magnitude of the numbers indicates how strongly the attitude is held.

```
# Load AFINN sentiment lexicon
load_afinn_sentiments <- function() {
  afinn_sentiments <- get_sentiments("afinn")

  # Print summary of AFINN sentiment lexicon
  cat("Summary of AFINN Sentiment Lexicon:\n")
  print(summary(afinn_sentiments))

  # Print unique sentiment values
  cat("Unique Sentiment Values:\n")
  print(sort(unique(afinn_sentiments$value)))

  # Set seed for reproducibility
  set.seed(1)

  # Sample 5 rows from the AFINN sentiment lexicon
  cat("Sampled Rows from AFINN Sentiment Lexicon:\n")
  print(afinn_sentiments[sample(nrow(afinn_sentiments), 5), ])
}

# Call function to load AFINN sentiment lexicon
load_afinn_sentiments()
```

```
## Summary of AFINN Sentiment Lexicon:
##      word               value
##  Length:2477        Min.   :-5.0000
##  Class :character   1st Qu.:-2.0000
##  Mode  :character   Median :-2.0000
##                     Mean   :-0.5894
##                     3rd Qu.: 2.0000
##                     Max.   : 5.0000
## Unique Sentiment Values:
##  [1] -5 -4 -3 -2 -1  0  1  2  3  4  5
## Sampled Rows from AFINN Sentiment Lexicon:
## # A tibble: 5 × 2
##   word           value
##   <chr>          <dbl>
## 1 frantic           -1
## 2 disappointing     -2
## 3 sullen            -2
## 4 fabulous           4
## 5 misinformation    -2
```

```
# Creating dataset containing only words with associated sentiment and add a sentiment column
sentiment_data <- clean_tokens %>%
  inner_join(get_sentiments("bing"), by = "word")

# Calculate sentiment scores for each review
sentiment_scores <- sentiment_data %>%
  group_by(Reviewer_no) %>%
  summarize(bing_sentiment = sum(sentiment == "positive") - sum(sentiment == "negative"))

# Merge sentiment scores with original dataframe
bookdata_with_sentiment <- bookdata %>%
  inner_join(sentiment_scores, by = "Reviewer_no")

# Print the dataframe with sentiment scores
print(bookdata_with_sentiment)
```

```
## # A tibble: 1,277 × 8
##    Title            Book_Price Rating Review_title Review_text Genre Reviewer_no
##    <chr>                 <dbl>  <int> <chr>        <chr>       <chr>       <int>
##  1 Eldest (Inherit…      34.0       5 A good book… "I've neve… Juve…           7
##  2 Hannibal               7.67      1 So sorry yo… "What a mo… Fict…          11
##  3 Good to Great         27.0       5 Good to Gre… "An excell… Busi…          35
##  4 Eldest (Inherit…      34.0       4 Love it for… "I'm 37 an… Juve…         372
##  5 Love & Respect:…      13.6       5 Love this b… "Love & Re… Psyc…         426
##  6 Hannibal               7.67      5 breath-taki… "This book… Fict…         440
##  7 Eldest (Inherit…      34.0       5 Long read, … "I believe… Juve…         446
##  8 Eldest (Inherit…      34.0       3 Disappointi… "I LOVED E… Juve…         510
##  9 Good to Great         27.0       4 An analytic… "After hav… Busi…         524
## 10 Great Expectati…      15.0       5 Amazing aud… "I love cl… Fict…         626
## # ℹ 1,267 more rows
## # ℹ 1 more variable: bing_sentiment <int>
```

Let’s inspect the reviews with highest and lowest sentiment

```
# Find the worst review text based on Bing sentiment score
worst_review_texts <- bookdata_with_sentiment[order(bookdata_with_sentiment$bing_sentiment)
[1], "Review_text"]

# Print the worst review text
for (review_text in worst_review_texts) {
  print(review_text)
}
```

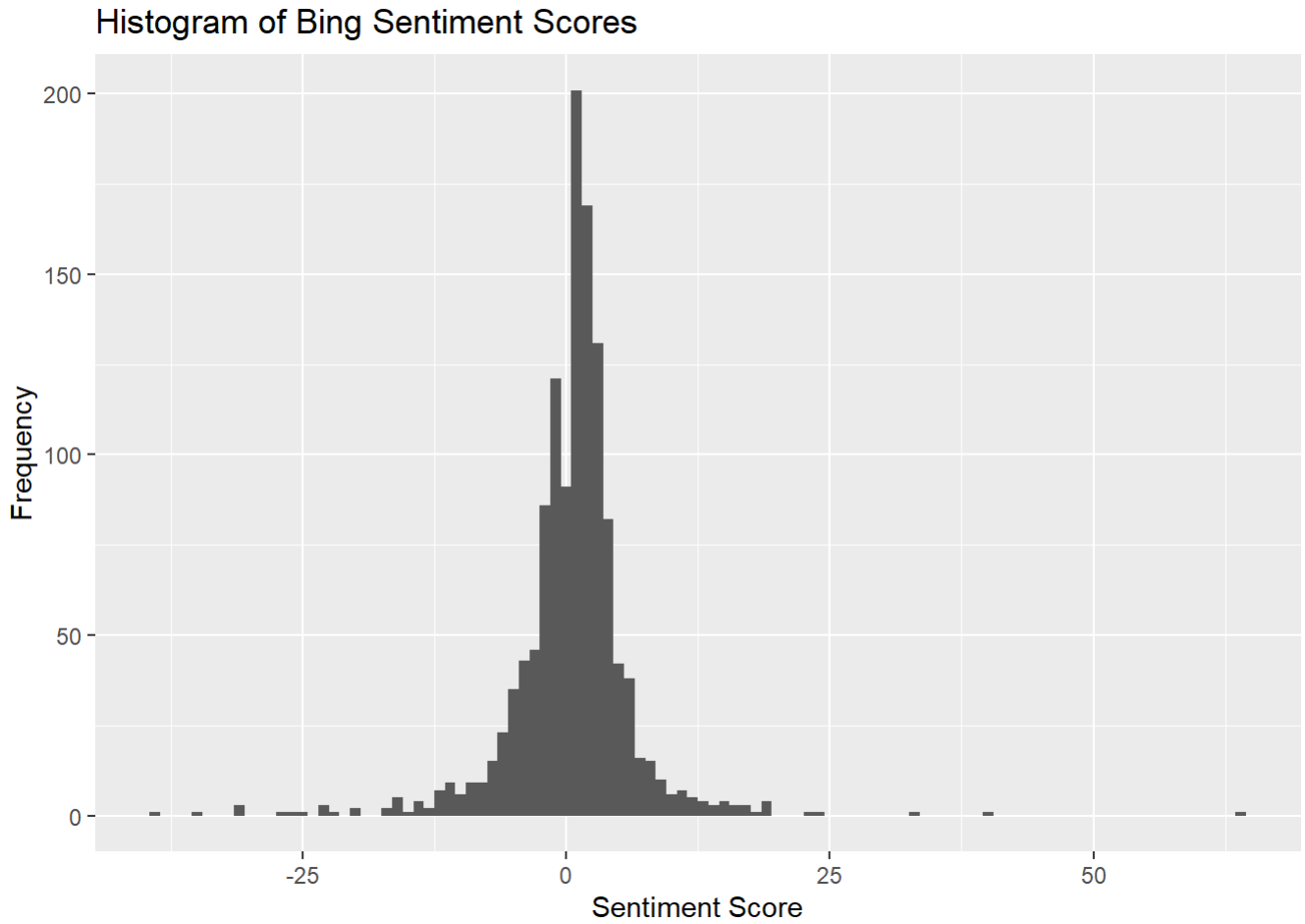
```
## [1] "Having just rewatched SILENCE OF THE LAMBS, and desperately needing some low-brow ent
ertainment, I bought HANNIBAL. Although Harris can be compelling in parts, he allows minor ch
aracters to take up loads of space with their banal babbling, while Starling's major nemeses
are two-dimensional caricatures. I guess the goal is to make us feel 'alright' with their gru
esome ends -we become participatory in their deaths- but they are so ludicrous that they are
not particularly interesting (Krendler, Verger, etc). Basically, everyone in the the book is
crazy as hell, which sort of takes away from Lecter's monstrosity, making him less interestin
g. Starling's boring Elektra complex compounds the dullness. Mind-numbing lists of Lecter's e
ffete purchases give the novel a feel of bland inertia. Then some icky guy dies an icky deat
h, and it hardly seems notable.Grammatical errors and typos abound. Apparently Dell editors a
re either cowards or clueless. Either way they should be fired. Harris' style at the beginnin
g of each chapter is particularly annoying: each introductory being short, hapless fragments
in the style of a screenplay. Incomplete sentences simply denote bad writing. He tries to eng
age the reader as the observer (&amp;quot;Now we are walking up the steps where blah blah blah&qu
ot;) but it all falls flat and the tense gets all screwed up. The narrative is disrupted with
such attempts at flair, and the scenes are subsequently poorly constructed. I got the feeling
like he was writing this novel in order to make a bundle of the movie, and the fluidity of th
e prose suffered as a consequence. This makes the flow of the narrative very tedious.The conc
lusion of the book elevates Starling and Lecter to a mythic inhumanity. It is so melodramatic
that you feel as if you have lost all touch with the characters. Harris adores his creations
too much, and it kills the story in the end. I give this book two stars because I'm not one o
f the die-hard fans who has had his hopes and dreams crushed by this silly book; plus the mov
ie managed to be even dumber."
```

```
# Find the best review text based on Bing sentiment score
best_review_texts <- bookdata_with_sentiment[order(bookdata_with_sentiment$bing_sentiment, de
creasing = TRUE)[1], "Review_text"]

# Print the best review text
for (review_text in best_review_texts) {
  print(review_text)
}
```

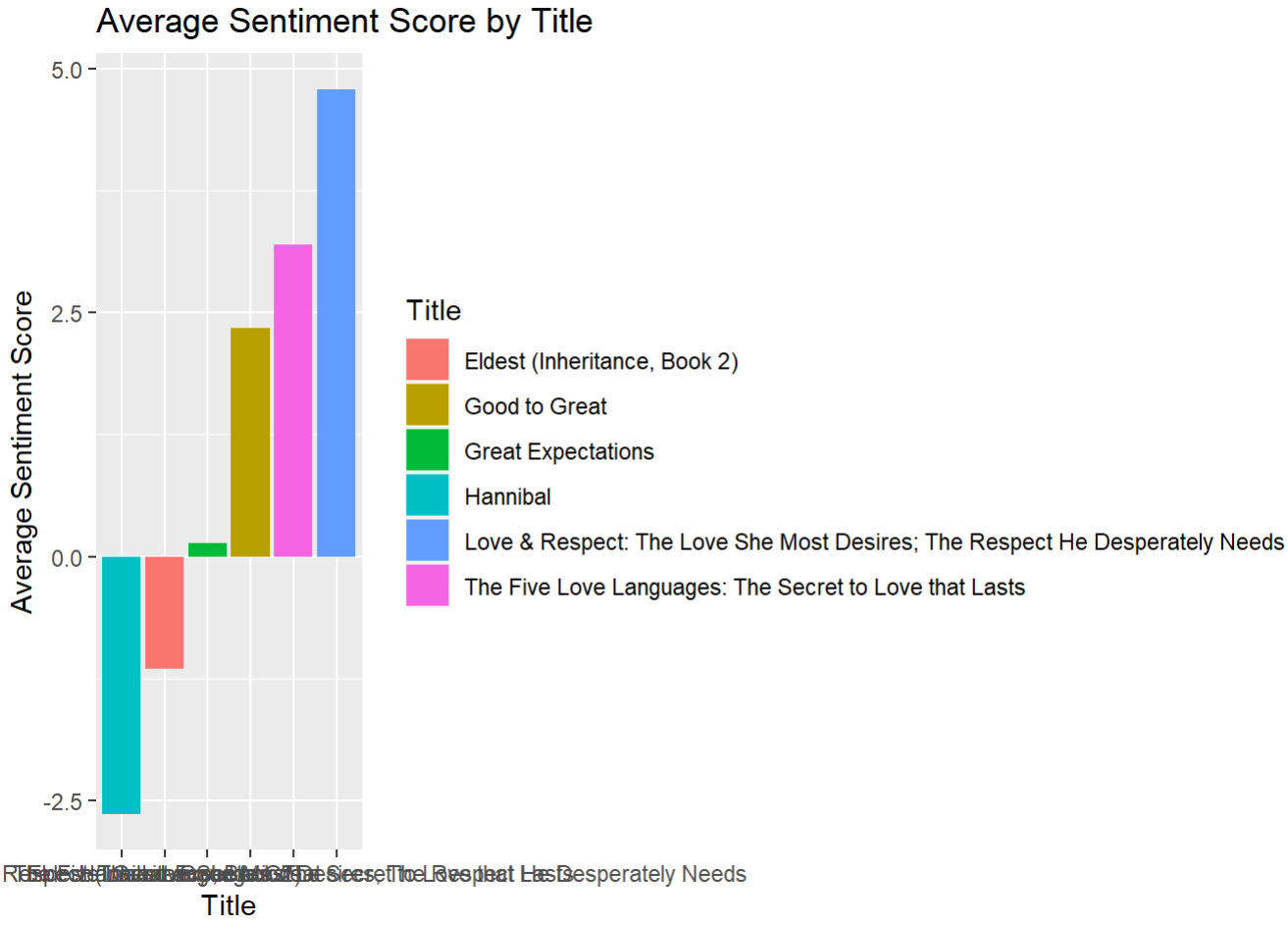
[1] "As of the time I am writing this review 368 out of 398 reviewers gave this book a 4 o r 5 star rating - that's 92% \"I liked it\" and \"I loved it\" ratings. With these many posit ive reviews there are some critical reviews as well that are worth reading to get a balanced overall review - there may actually be more (and likely are more) than 5 love languages or ca tegories. The author has a significant amount of knowledge and experience regarding married c ouples and it is certainly worth considering his input. What will make the information in thi s book the most beneficial is incorporating it with personal experience, and this subject wil l likely be a \"work in progress\" project with a focus on getting better everyday to result in a lasting, happy, and fulfilling marital arrangement.My favorite review is \"Learning to S peak, December 23, 2010\" where the reviewer's review could have been a superb foreword for t his book. May I suggest reading it as in my opinion it is brief, clear, and simple. If you ha ve time consider reading the other reviews and comments too.Of course, some may not agree or totally agree with this book's author; however, the subject of marriage is simple, yet comple xed - and even compounding at times.In my opinion this is one of the better books on this sub ject. There is some good material here making it worth considering reading it.This book did s timulate my thinking on the different viewpoints in marriage and if you'd like to read my com ments on this marriage subject contine, if not please feel free to move on. I am just hoping that some of these thoughts may help some considering marriage or who are already married.Som e believe that men and women basically use different parts of their brains. Often heard are: \"The left brain thinks, the right brain feels.\" \"The left brain analyzes, the right brain intuits.\" \"The left brain is logical, the right brain is emotional.\" Likely, our thinking, feeling, and loving are more complex than these simple statements; yet, at least on occasion (likely more often) men and women think and feel differently and express themselves different ly - the author of this book identifies, categorizes, and classifies love into five language s. I would add one additional language, which is the ability to sincerely and promptly say \"I'm sorry\" from one's heart. From my 45+ years of marriage and from what I have learned fr om many others, a successful, lasting, and happy marriage involves two great forgivers and ap ologizers. In my three and a half decades of managing people I have found that those who neve r or almost never say \"I'm sorry\" have difficulties with their working and personal relatio nships.A husband and a wife differ to varying degrees about how they both think and feel abou t things, and this is in harmony with how the Creator said regarding Adam that He was going t o make a helper for him, as a complement of him (not an identical twin of him - she was made different in a good way). A complement completes, perhaps making something just right. A husb and and wife will benefit from loving each other, especially as the other person wants and ne eds to be loved.Couple this with deep respect and you hold the two keys to a successful, last ing, and happy marriage and family life - Love and Respect.Hopefully adding this thought will help your loving and respectful marriage grow more each and every day: \"I love you more toda y than yesterday, but only half as much as tomorrow.\"And one additional thought: \"It is mor e beneficial for me to be respectful and loving in all that I do, than for me to be loved (so mething I very much want).\"Every marriage has the potential to be successful, lasting, and h appy, especially using the two keys of \"Love\" and \"Respect.\" Your marriage can be a most precious, valuable, and wonderful gift by using these two keys with sincerity and heartfelt c aring; and, never let pride, the childish silent treatment, or other unloving disrespectful t raits mar your treasured marriage!A good \"PRIDE\" antidote expressed before the end of the d ay: \"I'm sorry - I was mistaken - How can I make it up to you? - I'll do my best to be bette r - Will you please forgive me?\"A good \"CHILDISH SILENT TREATMENT\" antidote as soon as pos sible: Rescue the loving, caring, and respect adult within you. \"Whining\" and \"I won't tal k to you\" are childish - they rarely worked in childhood and have no place among true adult s. \"Scolding\" and \"Lecturing\" is easily blocked out. The best communications are loving, caring, and respectful adult expressions coupled with a big dose of attentive listening and u nderstanding.In ballroom dancing it has been said that \"it takes two to tango,\" and \"it ta kes one to lead.\" Many have found a successful, permanent, and happy marriage includes three - the loving husband, the respectful wife, and the Creator and Author of marriage (who perfec tly knows what's best).A good question to ask yourself at the beginning of each day: \"What w ill I do today that shows I both love and respect my spouse?\" TIP: While certainly one posit

```
ive act or action daily is a good start, many are even better and will bring more benefits.AD
DITIONAL BENEFICIAL READING: \"One Minute for Myself [Yourself]: How to Manage Your Most Valu
able Asset\" by Spencer Johnson, MD - while it is good to have a great relationship with your
spouse; it is essential to have a good relationship with yourself, especially if your goal is
to love your neighbor as yourself. Keep in mind if this is one of your goals that your closes
t neighbor is your spouse. Good relationships with ourselves and others I believe is what our
true success in life is all about. My thought is that one needs a good relationship with ones
elf first in order to have good relationships with others - and it is wise to pursue \"self-r
espect\" by being respectful of yourself and all others. I like the thought of \"self-respect
\" rather than \"self-esteem\" because it is easily possible to think too much of oneself; be
tter to just focus on being respectful, caring, loving, and having proper self-respect.ADDEND
UM: One of best ways to tell your spouse \"I Love You\" is to say \"I love you just the way y
ou are.\" The principle here is if you want to be accepted in any relationship you should giv
e your acceptance first. How many of us really want someone to relentlessly badger us to chan
ge this or change that about ourselves. Change in itself can be difficult, but that is anothe
r subject to consider."
```

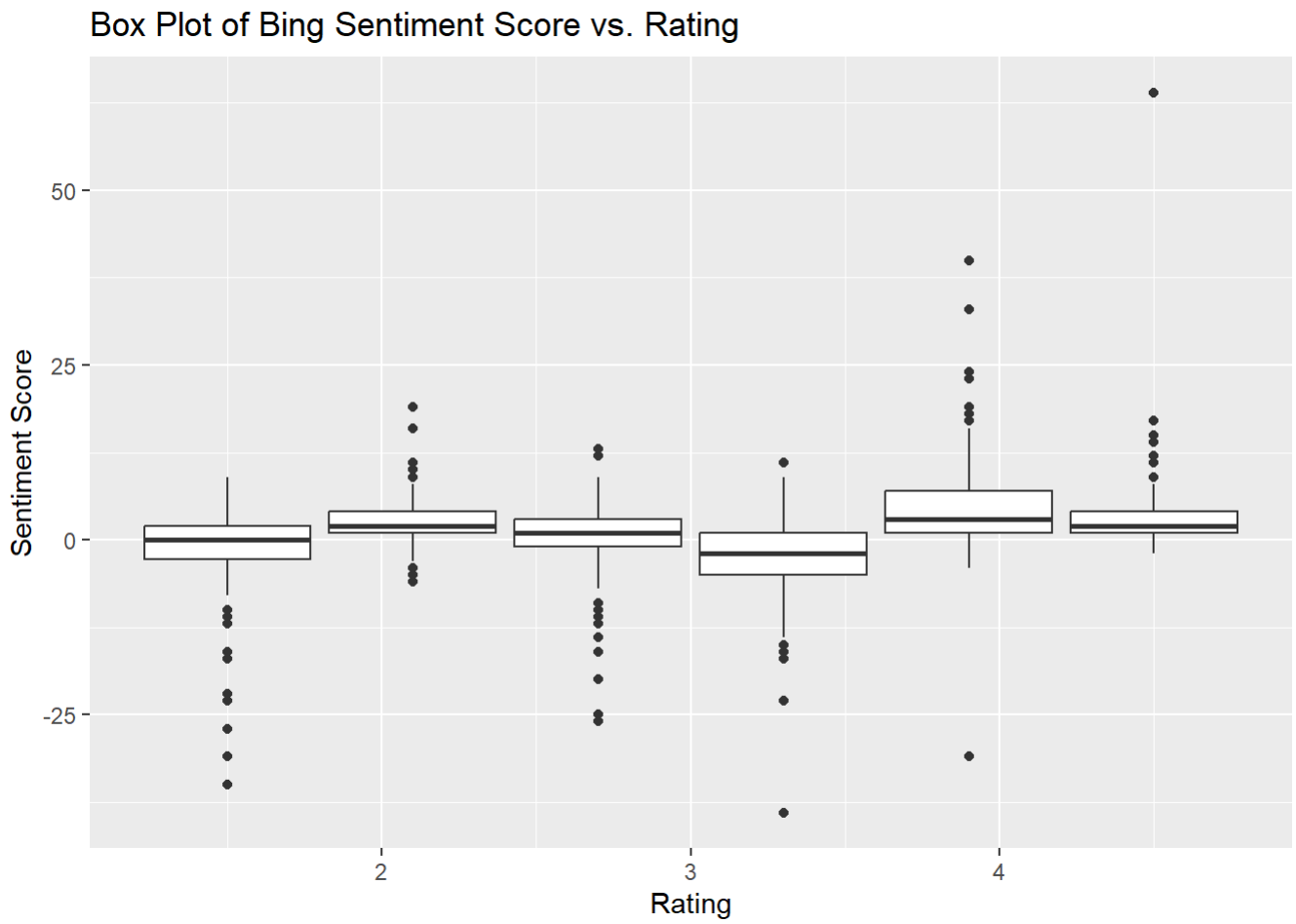
```
# Histogram of sentiment scores
ggplot(bookdata_with_sentiment, aes(x = bing_sentiment)) +
  geom_histogram(binwidth = 1) +
  labs(title = "Histogram of Bing Sentiment Scores", x = "Sentiment Score", y = "Frequency")
```



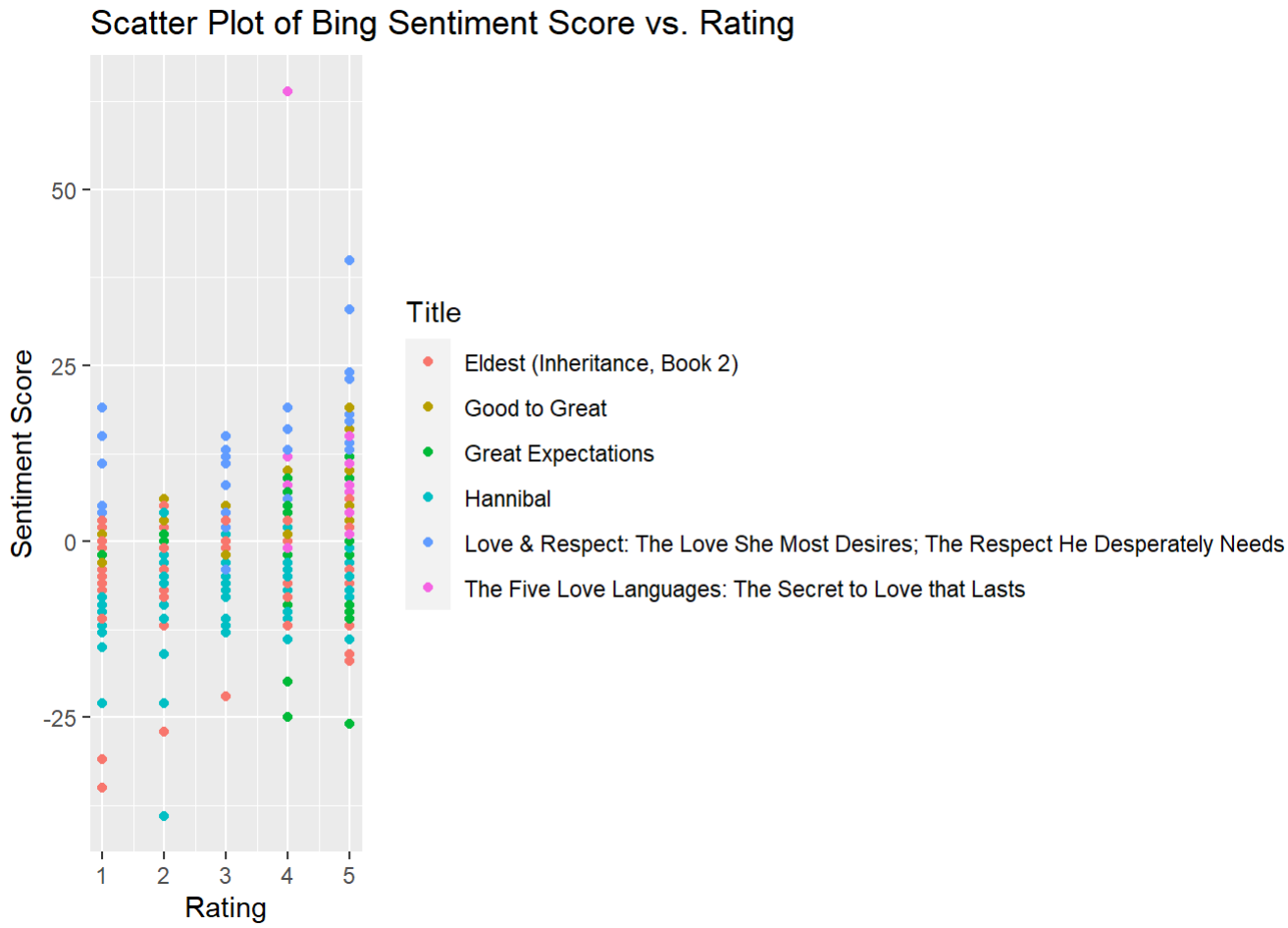
```
# Average Sentiment by Title
title_sentiment <- bookdata_with_sentiment %>%
  group_by(Title) %>%
  summarize(Average_Bing_Sentiment = mean(bing_sentiment))

ggplot(title_sentiment, aes(y = reorder(Title, Average_Bing_Sentiment), x = Average_Bing_Sent
iment, fill = Title)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labs(title = "Average Sentiment Score by Title", y = "Title", x = "Average Sentiment Scor
e")
```




```
# Box Plot of Sentiment against rating
ggplot(bookdata_with_sentiment, aes(x = Rating, y = bing_sentiment, group = Title)) +
  geom_boxplot() +
  labs(title = "Box Plot of Bing Sentiment Score vs. Rating",
       x = "Rating",
       y = "Sentiment Score")
```




```
# Scatter Plot of Bing Sentiment Score vs. Rating
ggplot(bookdata_with_sentiment, aes(x = Rating, y = bing_sentiment, color = Title)) +
  geom_point() +
  labs(title = "Scatter Plot of Bing Sentiment Score vs. Rating",
       x = "Rating",
       y = "Sentiment Score")
```



Applying AFINN lexicon

```
# Function to calculate sentiment scores and merge with original dataframe
calculate_and_merge_sentiment <- function(clean_tokens, bookdata_with_sentiment) {
  # Load AFINN lexicon
  afinn_lexicon <- get_sentiments("afinn")

  # Join clean_tokens with AFINN lexicon to get words with associated sentiment
  sentiment_data <- inner_join(clean_tokens, afinn_lexicon, by = "word")

  # Calculate sentiment scores for each review
  sentiment_score <- sentiment_data %>%
    group_by(Reviewer_no) %>%
    summarize(afinn_sentiment = sum(value))

  # Merge sentiment scores with original dataframe
  bookdata_with_sentiment <- inner_join(bookdata_with_sentiment, sentiment_score, by = "Revie
wer_no")

  return(bookdata_with_sentiment)
}

# Call function to calculate sentiment scores and merge with original dataframe
bookdata_with_sentiment <- calculate_and_merge_sentiment(clean_tokens, bookdata_with_sentimen
t)
```

```
worst_review_texts = bookdata_with_sentiment[order(bookdata_with_sentiment$afinn_sentiment)
[1],"Review_text"]

for (review_text in worst_review_texts){
  print(review_text)
}
```

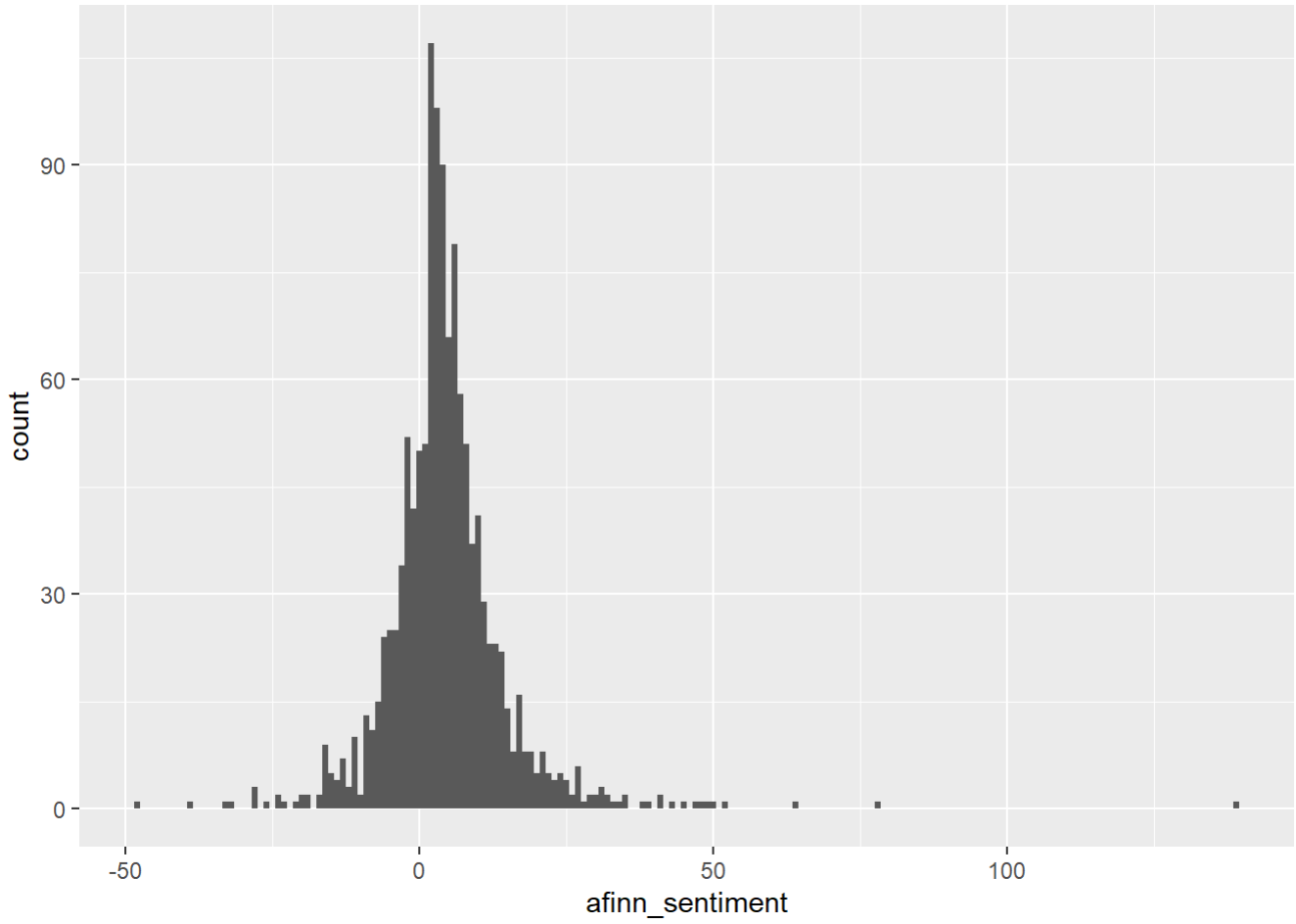
```
## [1] "Having just rewatched SILENCE OF THE LAMBS, and desperately needing some low-brow ent
ertainment, I bought HANNIBAL. Although Harris can be compelling in parts, he allows minor ch
aracters to take up loads of space with their banal babbling, while Starling's major nemeses
are two-dimensional caricatures. I guess the goal is to make us feel 'alright' with their gru
esome ends -we become participatory in their deaths- but they are so ludicrous that they are
not particularly interesting (Krendler, Verger, etc). Basically, everyone in the the book is
crazy as hell, which sort of takes away from Lecter's monstrosity, making him less interestin
g. Starling's boring Elektra complex compounds the dullness. Mind-numbing lists of Lecter's e
ffete purchases give the novel a feel of bland inertia. Then some icky guy dies an icky deat
h, and it hardly seems notable.Grammatical errors and typos abound. Apparently Dell editors a
re either cowards or clueless. Either way they should be fired. Harris' style at the beginnin
g of each chapter is particularly annoying: each introductory being short, hapless fragments
in the style of a screenplay. Incomplete sentences simply denote bad writing. He tries to eng
age the reader as the observer (&amp;quot;Now we are walking up the steps where blah blah blah&qu
ot;) but it all falls flat and the tense gets all screwed up. The narrative is disrupted with
such attempts at flair, and the scenes are subsequently poorly constructed. I got the feeling
like he was writing this novel in order to make a bundle of the movie, and the fluidity of th
e prose suffered as a consequence. This makes the flow of the narrative very tedious.The conc
lusion of the book elevates Starling and Lecter to a mythic inhumanity. It is so melodramatic
that you feel as if you have lost all touch with the characters. Harris adores his creations
too much, and it kills the story in the end. I give this book two stars because I'm not one o
f the die-hard fans who has had his hopes and dreams crushed by this silly book; plus the mov
ie managed to be even dumber."
```

```
best_review_texts = bookdata_with_sentiment[order(bookdata_with_sentiment$afinn_sentiment, de
creasing = TRUE)[1],"Review_text"]

for (review_text in best_review_texts){
  print(review_text)
}
```

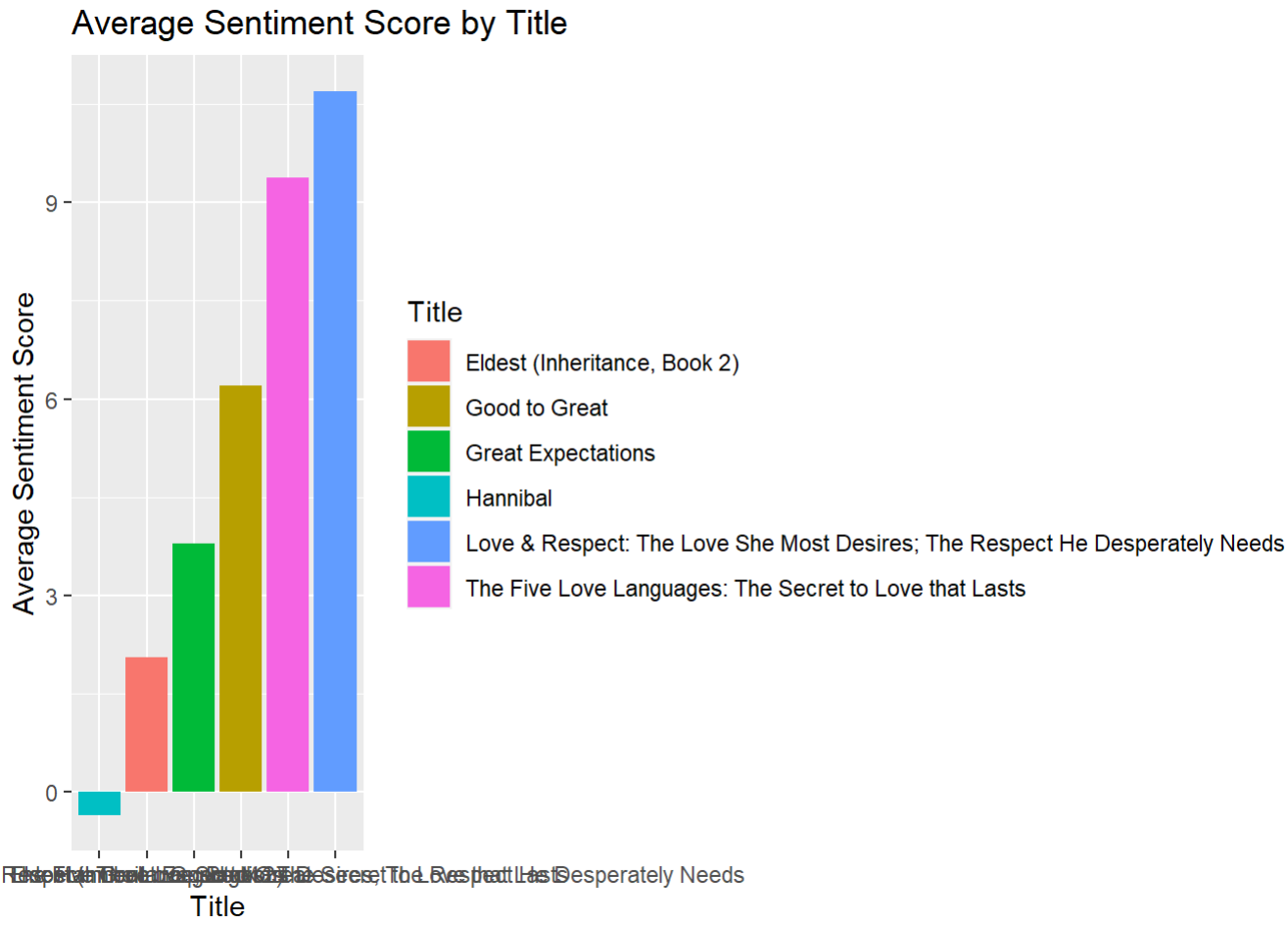
[1] "As of the time I am writing this review 368 out of 398 reviewers gave this book a 4 o r 5 star rating - that's 92% \"I liked it\" and \"I loved it\" ratings. With these many posit ive reviews there are some critical reviews as well that are worth reading to get a balanced overall review - there may actually be more (and likely are more) than 5 love languages or ca tegories. The author has a significant amount of knowledge and experience regarding married c ouples and it is certainly worth considering his input. What will make the information in thi s book the most beneficial is incorporating it with personal experience, and this subject wil l likely be a \"work in progress\" project with a focus on getting better everyday to result in a lasting, happy, and fulfilling marital arrangement.My favorite review is \"Learning to S peak, December 23, 2010\" where the reviewer's review could have been a superb foreword for t his book. May I suggest reading it as in my opinion it is brief, clear, and simple. If you ha ve time consider reading the other reviews and comments too.Of course, some may not agree or totally agree with this book's author; however, the subject of marriage is simple, yet comple xed - and even compounding at times.In my opinion this is one of the better books on this sub ject. There is some good material here making it worth considering reading it.This book did s timulate my thinking on the different viewpoints in marriage and if you'd like to read my com ments on this marriage subject contine, if not please feel free to move on. I am just hoping that some of these thoughts may help some considering marriage or who are already married.Som e believe that men and women basically use different parts of their brains. Often heard are: \"The left brain thinks, the right brain feels.\" \"The left brain analyzes, the right brain intuits.\" \"The left brain is logical, the right brain is emotional.\" Likely, our thinking, feeling, and loving are more complex than these simple statements; yet, at least on occasion (likely more often) men and women think and feel differently and express themselves different ly - the author of this book identifies, categorizes, and classifies love into five language s. I would add one additional language, which is the ability to sincerely and promptly say \"I'm sorry\" from one's heart. From my 45+ years of marriage and from what I have learned fr om many others, a successful, lasting, and happy marriage involves two great forgivers and ap ologizers. In my three and a half decades of managing people I have found that those who neve r or almost never say \"I'm sorry\" have difficulties with their working and personal relatio nships.A husband and a wife differ to varying degrees about how they both think and feel abou t things, and this is in harmony with how the Creator said regarding Adam that He was going t o make a helper for him, as a complement of him (not an identical twin of him - she was made different in a good way). A complement completes, perhaps making something just right. A husb and and wife will benefit from loving each other, especially as the other person wants and ne eds to be loved.Couple this with deep respect and you hold the two keys to a successful, last ing, and happy marriage and family life - Love and Respect.Hopefully adding this thought will help your loving and respectful marriage grow more each and every day: \"I love you more toda y than yesterday, but only half as much as tomorrow.\"And one additional thought: \"It is mor e beneficial for me to be respectful and loving in all that I do, than for me to be loved (so mething I very much want).\"Every marriage has the potential to be successful, lasting, and h appy, especially using the two keys of \"Love\" and \"Respect.\" Your marriage can be a most precious, valuable, and wonderful gift by using these two keys with sincerity and heartfelt c aring; and, never let pride, the childish silent treatment, or other unloving disrespectful t raits mar your treasured marriage!A good \"PRIDE\" antidote expressed before the end of the d ay: \"I'm sorry - I was mistaken - How can I make it up to you? - I'll do my best to be bette r - Will you please forgive me?\"A good \"CHILDISH SILENT TREATMENT\" antidote as soon as pos sible: Rescue the loving, caring, and respect adult within you. \"Whining\" and \"I won't tal k to you\" are childish - they rarely worked in childhood and have no place among true adult s. \"Scolding\" and \"Lecturing\" is easily blocked out. The best communications are loving, caring, and respectful adult expressions coupled with a big dose of attentive listening and u nderstanding.In ballroom dancing it has been said that \"it takes two to tango,\" and \"it ta kes one to lead.\" Many have found a successful, permanent, and happy marriage includes three - the loving husband, the respectful wife, and the Creator and Author of marriage (who perfec tly knows what's best).A good question to ask yourself at the beginning of each day: \"What w ill I do today that shows I both love and respect my spouse?\" TIP: While certainly one posit

```
ive act or action daily is a good start, many are even better and will bring more benefits.AD
DITIONAL BENEFICIAL READING: \"One Minute for Myself [Yourself]: How to Manage Your Most Valu
able Asset\" by Spencer Johnson, MD - while it is good to have a great relationship with your
spouse; it is essential to have a good relationship with yourself, especially if your goal is
to love your neighbor as yourself. Keep in mind if this is one of your goals that your closes
t neighbor is your spouse. Good relationships with ourselves and others I believe is what our
true success in life is all about. My thought is that one needs a good relationship with ones
elf first in order to have good relationships with others - and it is wise to pursue \"self-r
espect\" by being respectful of yourself and all others. I like the thought of \"self-respect
\" rather than \"self-esteem\" because it is easily possible to think too much of oneself; be
tter to just focus on being respectful, caring, loving, and having proper self-respect.ADDEND
UM: One of best ways to tell your spouse \"I Love You\" is to say \"I love you just the way y
ou are.\" The principle here is if you want to be accepted in any relationship you should giv
e your acceptance first. How many of us really want someone to relentlessly badger us to chan
ge this or change that about ourselves. Change in itself can be difficult, but that is anothe
r subject to consider."
```

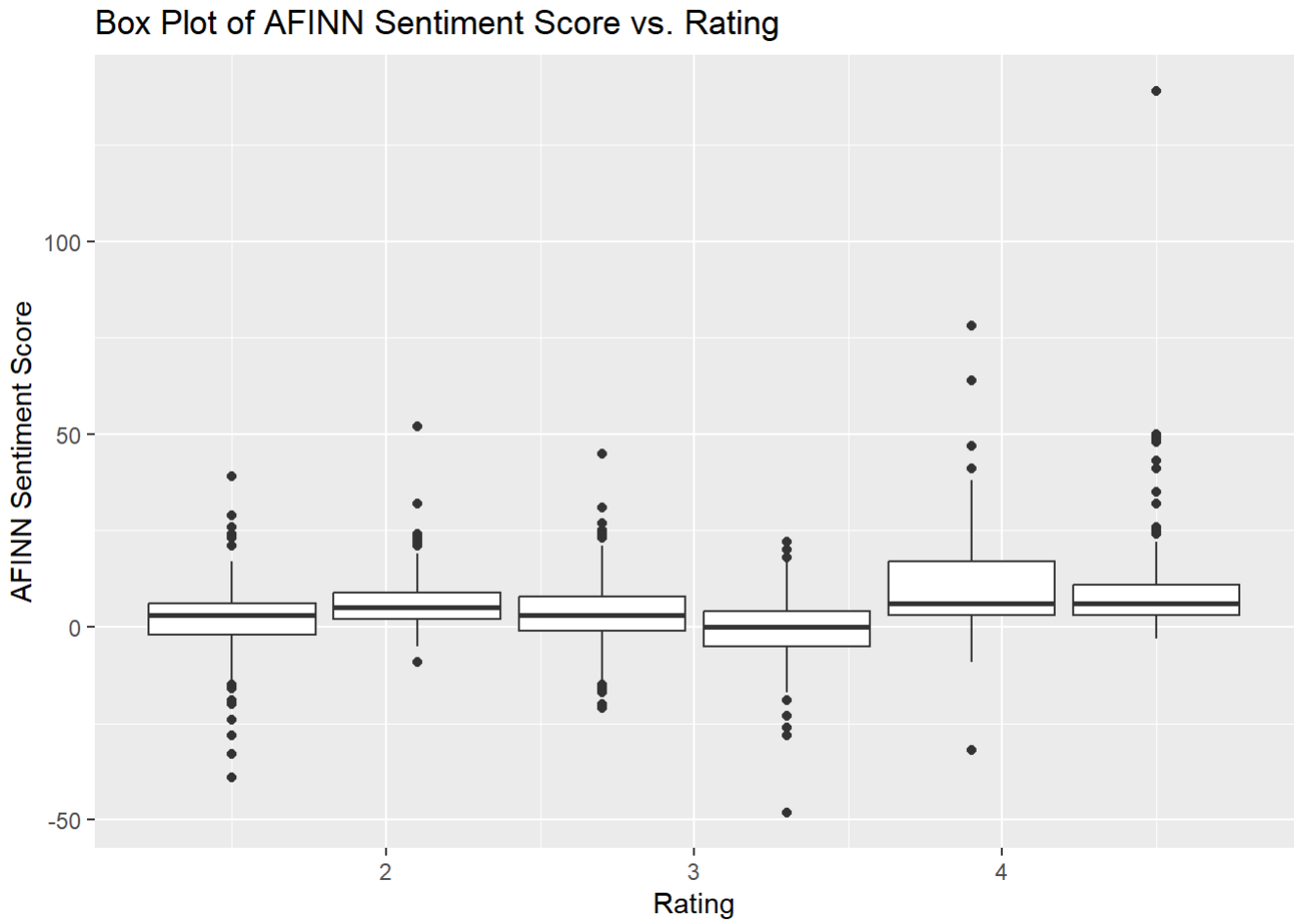
```
# Histogram of sentiment scores
ggplot(bookdata_with_sentiment, aes(x = afinn_sentiment)) +
  geom_histogram(binwidth = 1)
```



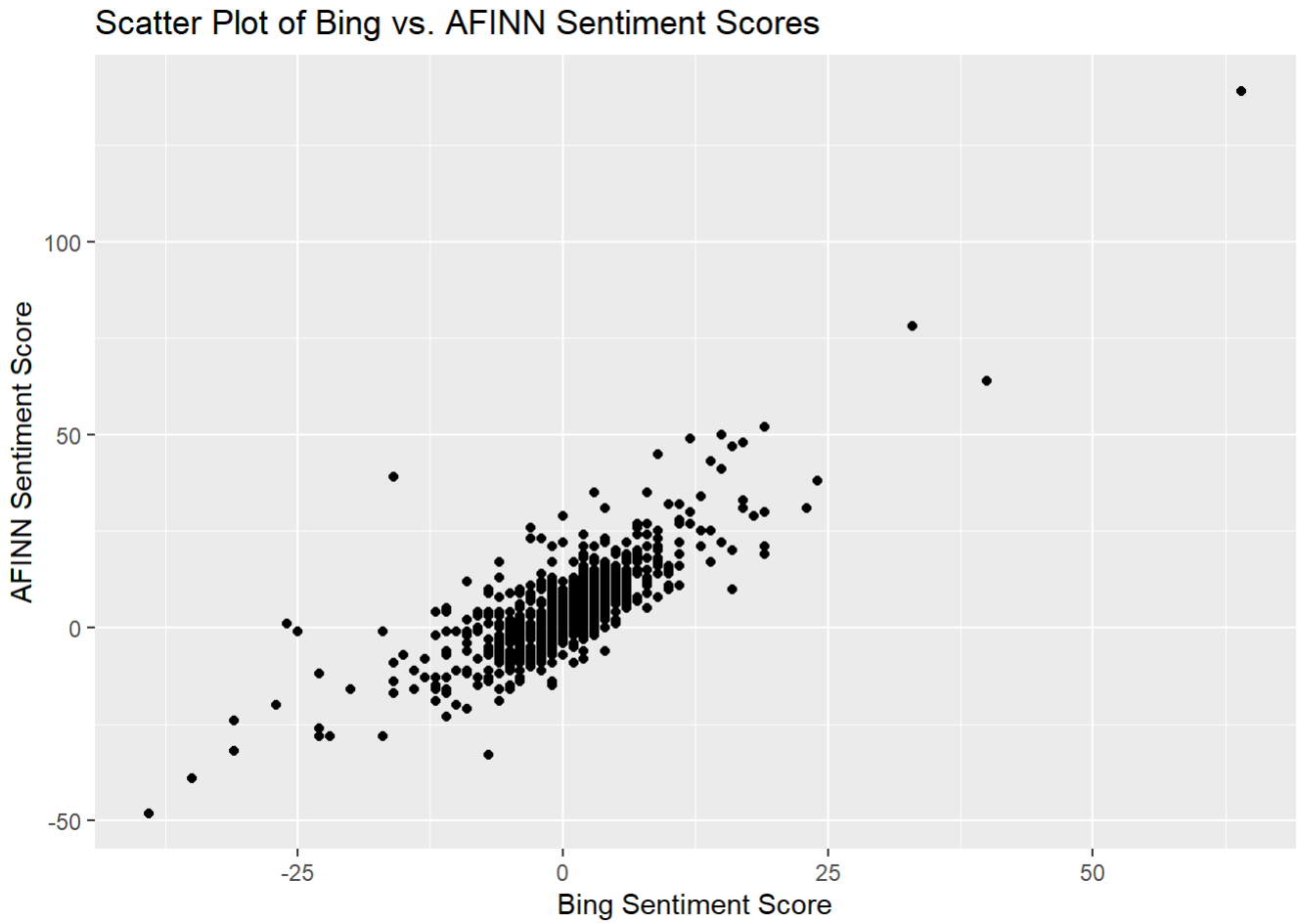
```
# Average Sentiment by Title
title_sentiment <- bookdata_with_sentiment %>%
  group_by(Title) %>%
  summarize(Average_Afinn_Sentimet = mean(afinn_sentiment))

ggplot(title_sentiment, aes(y = reorder(Title, Average_Afinn_Sentimet), x = Average_Afinn_Sen
timet, fill = Title)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labels(title = "Average Sentiment Score by Title", y = "Title", x = "Average Sentiment Scor
e")
```




```
# Box Plot of Sentiment against rating
ggplot(bookdata_with_sentiment, aes(x = Rating, y = afinn_sentiment, group = Title)) +
  geom_boxplot() +
  labels(title = "Box Plot of AFINN Sentiment Score vs. Rating",
       x = "Rating",
       y = "AFINN Sentiment Score")
```




```
# Scatter Plot of Bing vs. AFINN Sentiment
ggplot(bookdata_with_sentiment, aes(x = bing_sentiment, y = afinn_sentiment)) +
  geom_point() +
  labs(title = "Scatter Plot of Bing vs. AFINN Sentiment Scores",
       x = "Bing Sentiment Score",
       y = "AFINN Sentiment Score")
```



Sentiment Distribution: The study revealed that the sentiment expressed in customer assessments was not uniform, with certain titles predominantly displaying positive emotion and others displaying a combination of both positive and negative sentiment.

Relationship Sentiment and Rating: There was a definite correlation between the two, with higher-rated titles often denoting more positive sentiment and vice versa. This suggests that customers' evaluations align with their overall satisfaction level with the books.

Frequency of Different Emotions: A variety of emotions were prevalent across titles, indicating a nuanced feeling among buyers. Though they were less common, negative emotions like despair and fury were still present. Positive emotions like contentment, trust, and expectation were expressed frequently.

TOPIC MODELING

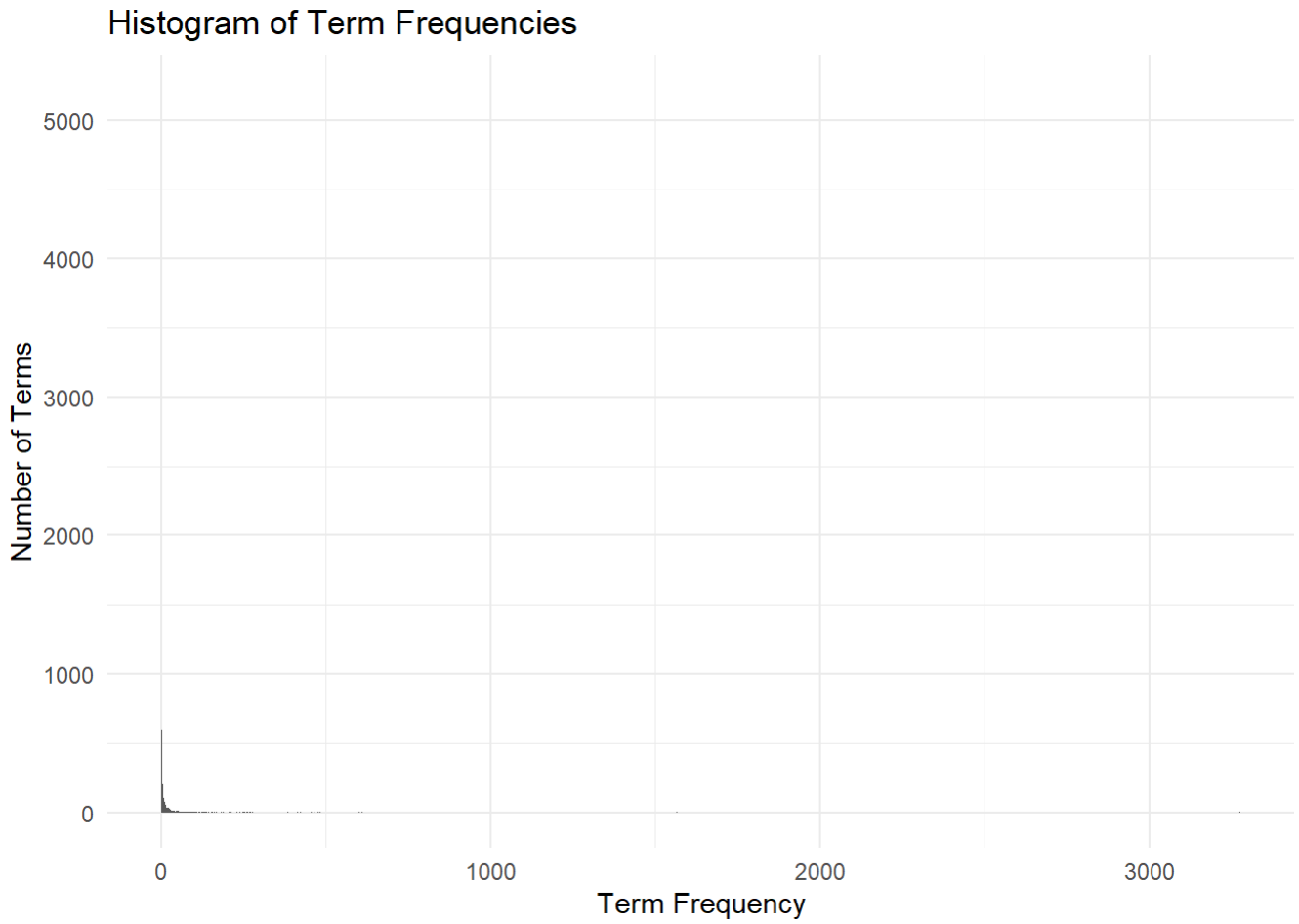
Convert data to a clean Term Document Matrix (TDM)

```
# Convert the text column to a corpus
corpus <- VCorpus(VectorSource(bookdata$Review_text))

# Remove missing values from the corpus
corpus <- corpus[!is.na(corpus)]

# Apply text cleaning to non-missing values
corpus <- tm_map(corpus, content_transformer(tolower)) %>%
  tm_map(content_transformer(function(x) gsub("[^a-zA-Z ]", "", x))) %>%
  tm_map(removeWords, stopwords("en")) %>%
  tm_map(stemDocument)

# Convert the corpus to a term document matrix
tdm <- TermDocumentMatrix(corpus, control = list(wordLengths = c(3, 15)))

# Convert the term document matrix to a matrix
tdm_matrix <- as.matrix(tdm)
```

```
# Create a data frame for plotting
term_frequency_bookdata <- data.frame(term = rownames(tdm_matrix), frequency = rowSums(tdm_ma
trix))

# Sort the data frame by frequency in descending order and select the top 10 terms
top_terms <- term_frequency_bookdata %>%
  arrange(desc(frequency)) %>%
  head(10)

# Display the top 10 terms
print(top_terms)
```

```
##            term frequency
## book       book      3276
## read       read      1567
## great     great       957
## love       love       877
## one         one       773
## good       good       677
## like       like       611
## will       will       602
## charact charact       532
## time       time       490
```

```
# Create the histogram
ggplot(term_frequency_bookdata, aes(x = frequency)) +
  geom_histogram(binwidth = 1) +
  labs(title = "Histogram of Term Frequencies",
       x = "Term Frequency",
       y = "Number of Terms") +
  theme_minimal()
```




```
# Find terms that appear in more than 10% of documents
frequent_terms <- findFreqTerms(tdm, lowfreq = 0.1 * ncol(tdm_matrix))
# Find terms that appear in less than 1% of documents
rare_terms <- findFreqTerms(tdm, highfreq = 0.01 * ncol(tdm_matrix))

print("Frequent Terms")
```

```
## [1] "Frequent Terms"
```

```
print(frequent_terms)
```

```
##   [1] "also"         "anoth"        "author"       "becom"        "believ"
##   [6] "best"         "better"       "book"         "busi"         "can"
##  [11] "chang"        "chapter"      "charact"      "collin"       "come"
##  [16] "compani"      "concept"      "coupl"        "dicken"       "didnt"
##  [21] "differ"       "dont"         "dragon"       "eldest"       "end"
##  [26] "enjoy"        "eragon"       "even"         "everi"        "expect"
##  [31] "fantasi"      "feel"         "find"         "first"        "found"
##  [36] "get"          "give"         "good"         "great"        "hannib"
##  [41] "harri"        "help"         "high"         "husband"      "interest"
##  [46] "just"         "know"         "languag"      "last"         "learn"
##  [51] "lecter"       "life"         "like"         "littl"        "long"
##  [56] "look"         "lot"          "love"         "made"         "make"
##  [61] "mani"         "marri"        "marriag"      "may"          "movi"
##  [66] "much"         "must"         "need"         "never"        "new"
##  [71] "novel"        "now"          "one"          "page"         "paolini"
##  [76] "part"         "peopl"        "person"       "pip"          "plot"
##  [81] "point"        "put"          "read"         "reader"       "realli"
##  [86] "recommend"    "relationship" "respect"      "review"       "right"
##  [91] "say"          "see"          "seem"         "show"         "silenc"
##  [96] "starl"        "still"        "stori"        "take"         "thing"
## [101] "think"        "though"       "thought"      "time"         "tri"
## [106] "two"          "understand"   "use"          "want"         "way"
## [111] "well"         "will"         "women"        "wonder"       "work"
## [116] "world"        "write"        "written"      "year"
```

```
print("First 20 Infrequent Terms")
```

```
## [1] "First 20 Infrequent Terms"
```

```
print(head(rare_terms, 20))
```

```
##  [1] "abandon"   "abash"     "abbott"    "abel"      "abet"      "abhor"
##  [7] "abhorr"    "abhorsen"  "abnorm"    "abod"      "abolish"   "abort"
## [13] "abosoleut" "abound"    "aboutfac"  "aboutit"   "aboutth"   "aboutwif"
## [19] "abovement" "abovemi"
```

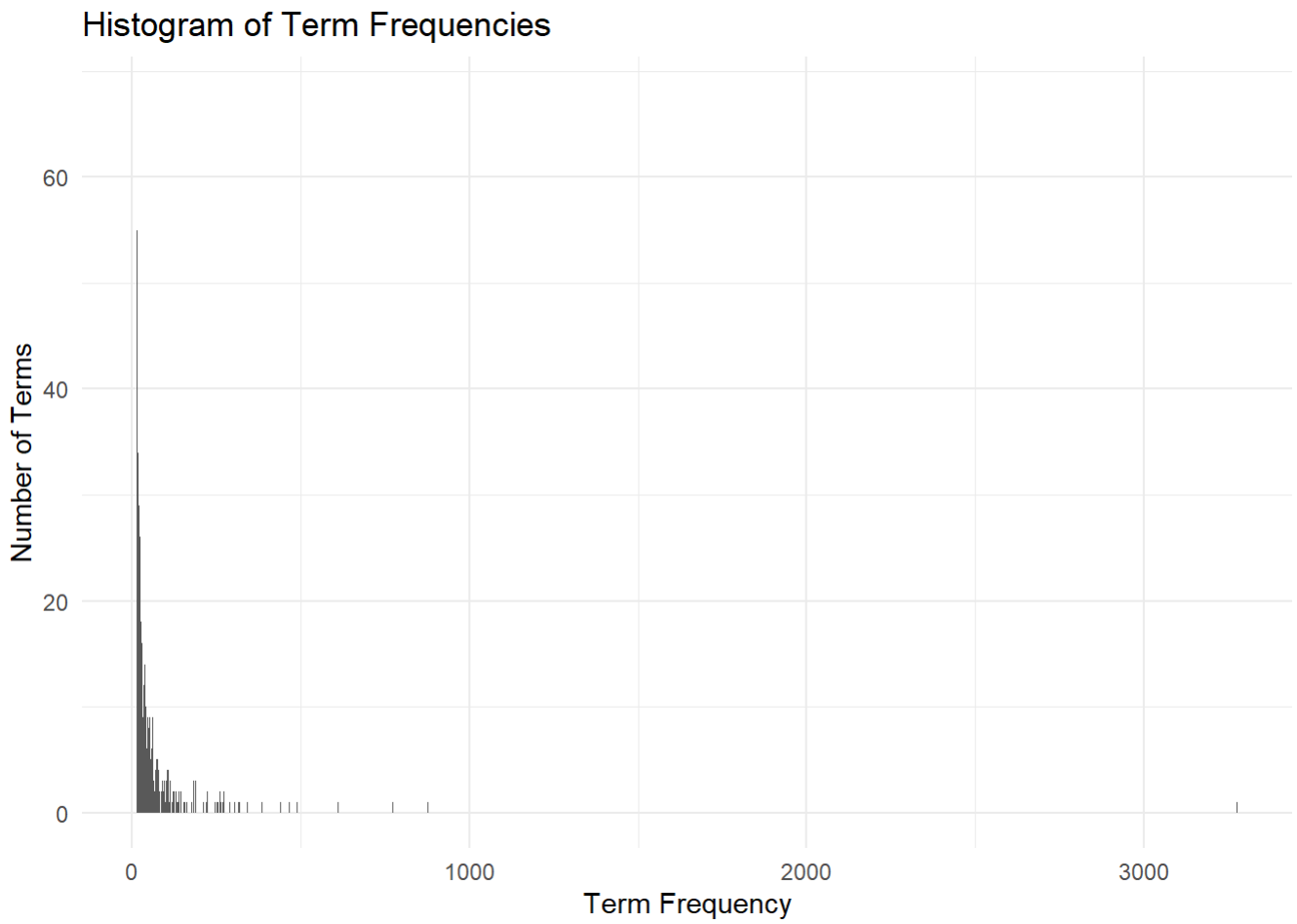
```
# Edit list of words to remove rare terms
to_remove <- rare_terms

# Filter TDM Matrix
filtered_tdm_matrix <- tdm_matrix[!rownames(tdm_matrix) %in% to_remove, ]

# Remove columns with zero sum from the term document matrix
column_sums <- colSums(filtered_tdm_matrix)
zero_columns <- which(column_sums == 0)
if(length(zero_columns) > 0) {
  filtered_tdm_matrix <- filtered_tdm_matrix[, -zero_columns]
} else {
  print("No zero columns in the TDM matrix")
}
```

```
## [1] "No zero columns in the TDM matrix"
```

```
term_frequencies <- rowSums(filtered_tdm_matrix)

# Create a data frame for plotting
term_frequency_df <- data.frame(term = names(term_frequencies), frequency = term_frequencies)

# Sort the data frame by frequency in descending order and select the top 10 terms
top_terms <- term_frequency_df %>%
  arrange(desc(frequency)) %>%
  head(10)

# Display the top 10 terms
print(top_terms)
```

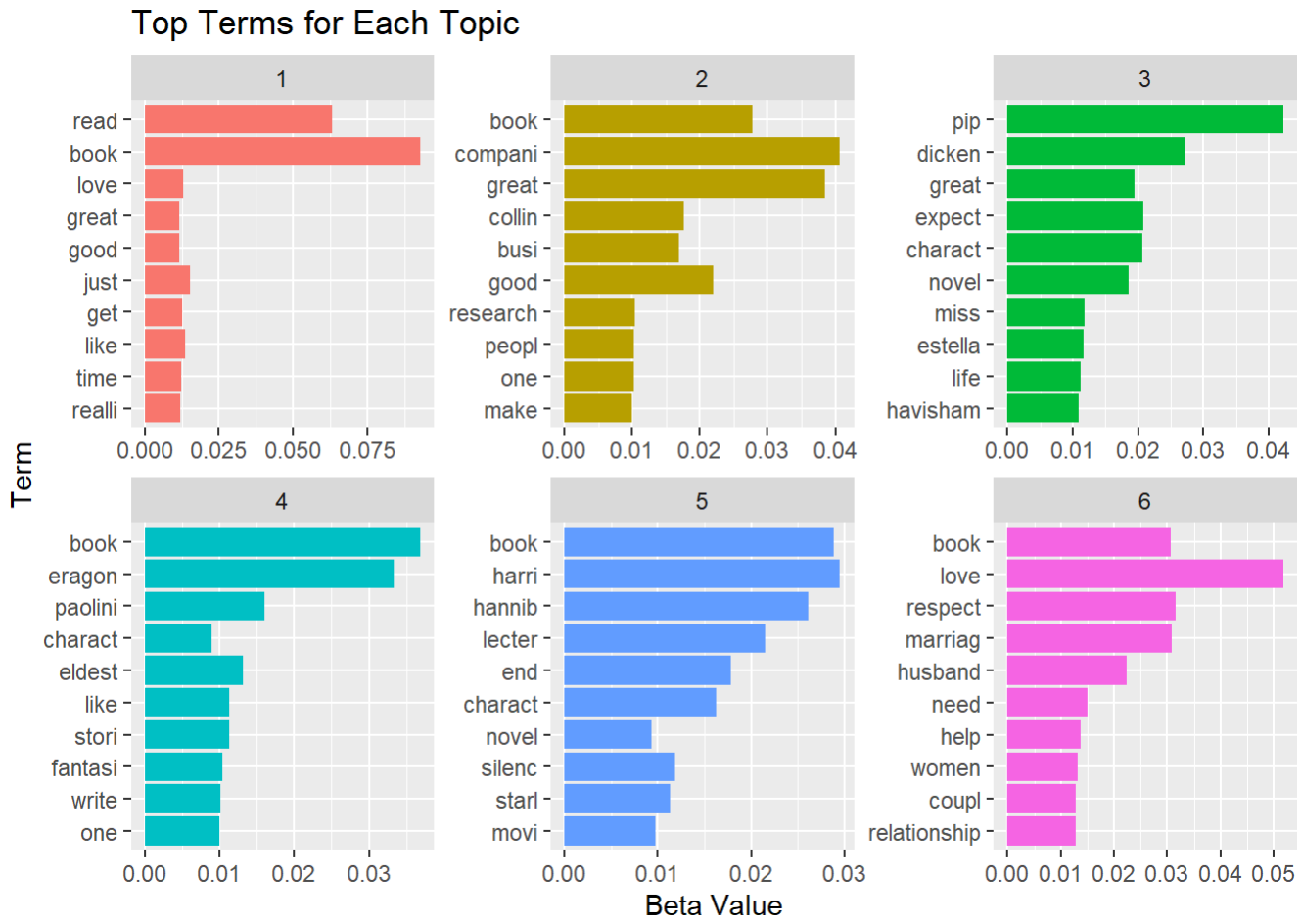
```
##            term frequency
## book       book      3276
## read       read      1567
## great     great       957
## love       love       877
## one         one       773
## good       good       677
## like       like       611
## will       will       602
## charact charact       532
## time       time       490
```

```
# Create the histogram
ggplot(term_frequency_df, aes(x = frequency)) +
  geom_histogram(binwidth = 1) +
  labs(title = "Histogram of Term Frequencies",
       x = "Term Frequency",
       y = "Number of Terms") +
  theme_minimal()
```




```
# Convert the filtered term document matrix to a document term matrix
dtm <- t(filtered_tdm_matrix)

# Apply Latent Dirichlet Allocation (LDA) model with 6 topics
lda_model <- LDA(dtm, k = 6)
```

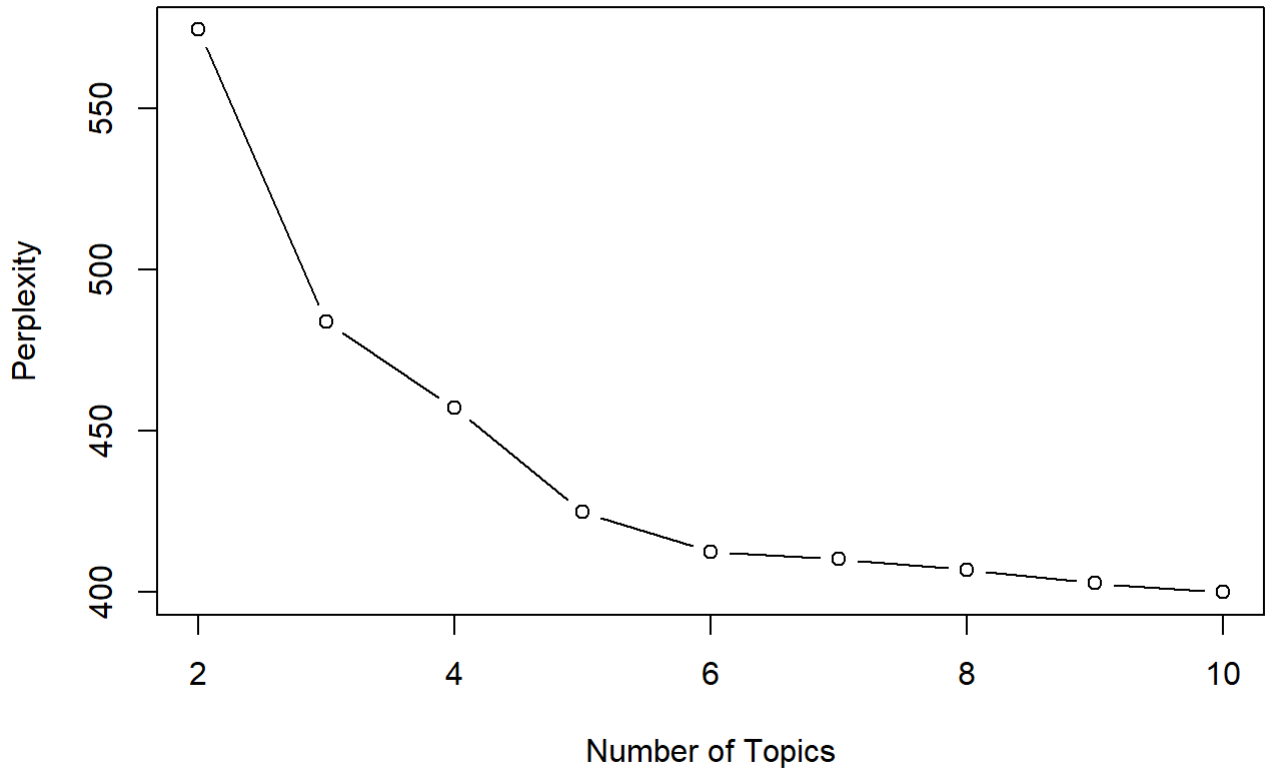
```
# Extract the topics and their associated terms from the LDA model
topics <- tidy(lda_model, matrix = "beta")

# Select the top 10 terms for each topic
top_terms <- topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

# Plot the top terms for each topic
top_terms %>%
  ggplot(aes(x = reorder(term, beta), y = beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip() +
  labs(title = "Top Terms for Each Topic") +
  xlab("Term") +
  ylab("Beta Value")
```



Selecting k

Finding an appropriate value of k can be aided by metrics such as perplexity. **Perplexity** quantifies the degree to which a probability model or probability distribution can forecast a sample. A better model is indicated by less confusion.A lower number indicates higher comprehension of new texts by the model, similar to a score for comprehension.

```
# Define the range of k values
range_k <- seq(2, 10, by = 1)

# Calculate perplexity for each value of k
perplexities <- sapply(range_k, function(k) {
  model <- LDA(dtm, k = k, control = list(seed = 1))
  perplexity(model)
})

# Plot perplexities
plot(range_k, perplexities, type = "b", xlab = "Number of Topics", ylab = "Perplexity")
```




```
print(perplexities)
```

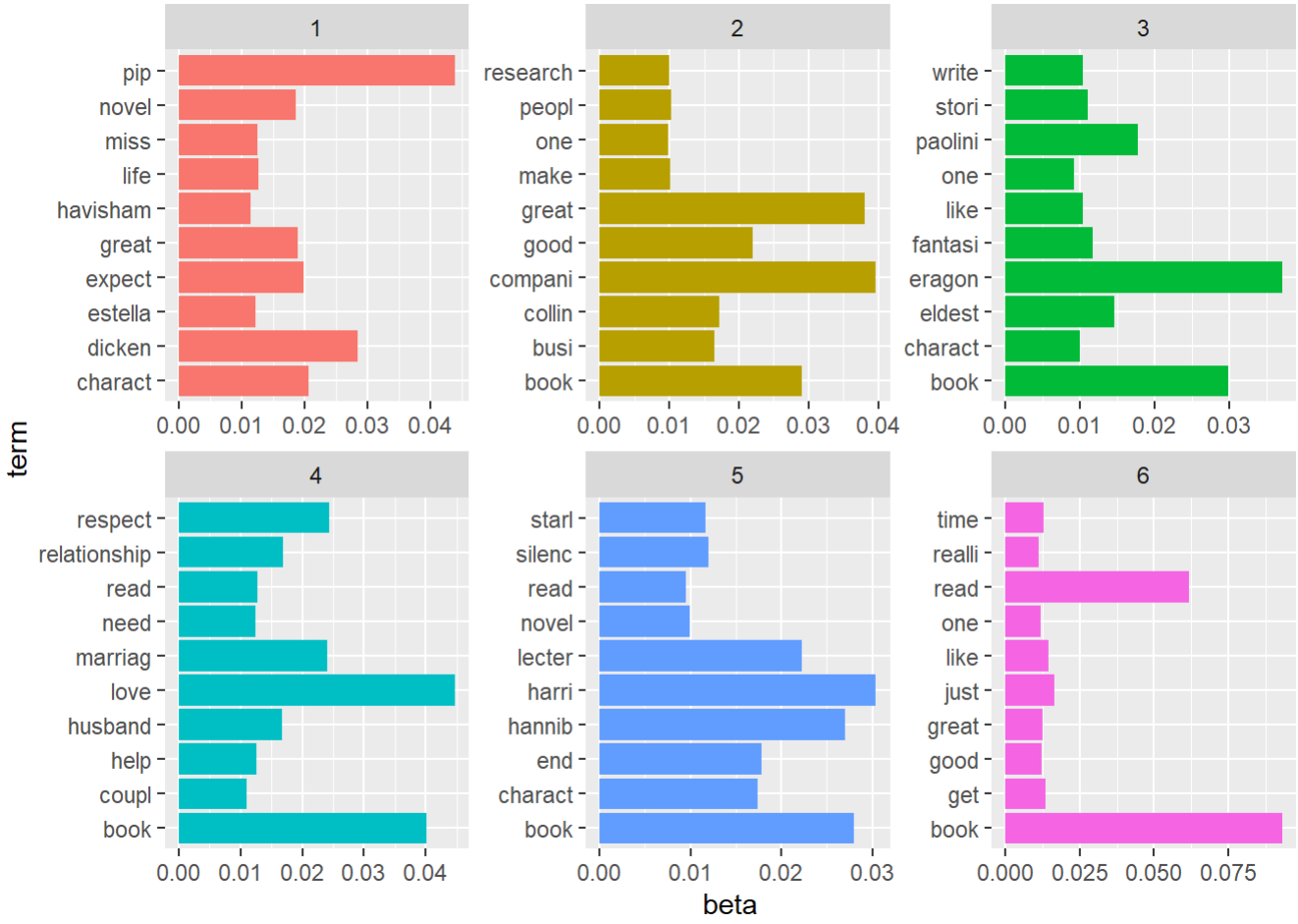
```
## [1] 574.3990 483.7188 457.1047 424.7669 412.6192 410.1711 406.8506 402.7719
## [9] 399.9912
```

Interactive Principal Componenet Space Visualisation

```
# Set seed for reproducibility
# Remove missing values from the document-term matrix
dtm <- dtm[complete.cases(dtm), ]

# Fit LDA model with 6 topics
lda_model <- LDA(dtm, k = 6)

# Create JSON data for visualization
lda_vis <- createJSON(phi = posterior(lda_model)$terms,
                      theta = posterior(lda_model)$topics,
                      doc.length = rowSums(as.matrix(dtm)),
                      vocab = colnames(as.matrix(dtm)),
                      term.frequency = colSums(as.matrix(dtm)))

# Display interactive visualization in RStudio Viewer
serVis(lda_vis, output_format = "viewer")
```

```
## Loading required namespace: servr
```

```
# Extract the topics and their associated terms from the LDA model
topics <- tidy(lda_model, matrix = "beta")

# Save the plot as an image file
ggsave("plot.png", width = 10, height = 8)

# Select the top 10 terms for each topic
top_terms <- topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

# Plot the top terms for each topic
top_terms %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```




```
# Extract document-topic distributions
documents <- tidy(lda_model, matrix = "gamma")
```

Based on the analysis having six title with six different genre negleting the perplexity justification, we identified six distinct topics from the book reviews dataset. These topics include:

Topic 1: A good topic name for the book based on these terms ‘starl’, ‘silenc’, ‘novel’, ‘lecter’, ‘lamb’, ‘harri’, ‘hannib’, ‘end’, ‘charact’, ‘book’ would be:

“Psychological Thrillers and Character Studies”

Character-driven storylines, denoted by phrases like “character,” and psychological components like Hannibal Lecter are reflected in this label. It covers the themes of psychology, suspense, and nuanced character development that are implied by the phrases used in the topic. . Topic 2: Based on these terms ‘research, peopl, one, make, great, good, compani, collin, busi, book’, a suitable topic name for the book could be:

“Innovation and Entrepreneurship in Business” Research, creativity (‘make’), excellence (‘great’, ‘good’), and business (‘company’, ‘busi’) are all represented in this name. It implies that the literature will have a thematic emphasis on innovation, entrepreneurship, and corporate success. Terms like ‘book’ and ‘collin’ suggest an authorial or personal take on these subjects, either from the perspective of a particular person called Collin or from the literary setting as a whole.

Topic 3: Based on these terms ‘pip’, ‘novel’, ‘miss’, ‘life’, ‘havisham’, ‘great’, ‘expect’, ‘estella’, ‘dicken’, ‘charact’, a suitable topic name for this book could be:

“Exploring Characters and Themes in Great Expectations” This name reflects terms found in Charles Dickens’ novel “Great Expectations” that are associated with characters (‘pip’,‘miss’, ‘havisham’, ‘estella’, ‘charact’) and important themes (‘life’, ‘great’, ‘expect’). It implies a thematic concentration on life events, character growth, and the novel’s recurrent themes of relationships and expectations.

Topic 4: Based on these terms ‘time’, ‘really’, ‘read’, ‘one’, ‘like’, ‘just’, ‘great’, ‘good’, ‘get’, ‘book’, a suitable topic name for this book could be:

“Exploring Reading Experiences and Literary Appreciation” This name reflects the presence of terms related to reading (‘read’, ‘like’, ‘get’, ‘book’), expressions of enjoyment or praise (‘great’, ‘good’), and descriptors of personal engagement (‘really’, ‘just’) within the topic. It suggests a thematic focus on readers’ experiences, perceptions, and appreciation of literature, encompassing aspects of enjoyment, understanding, and engagement with texts.

Topic 5: Based on these terms ‘respect’, ‘relationship’, ‘read’, ‘need’, ‘marriage’, ‘love’, ‘husband’, ‘help’, ‘couple’, ‘book’, a suitable topic name for this book could be:

“Exploring Relationships and Love in Literature” This name reflects the presence of terms related to relationships (‘relationship’, ‘marriage’, ‘couple’), expressions of affection (‘love’, ‘husband’), and themes of support or assistance (‘help’) within the topic. It suggests a thematic focus on interpersonal relationships, romantic connections, and the dynamics of love within literary narratives, encompassing aspects of emotional connection, partnership, and mutual support.

Topic 6: Based on these terms ‘write’, ‘will’, ‘story’, ‘paolini’, ‘one’, ‘like’, ‘fantasy’, ‘eragon’, ‘eldest’, ‘book’, a suitable topic name for this book could be:

“Exploring Fantasy Worlds in Paolini’s Novels” This name reflects the presence of terms related to fantasy literature (‘fantasy’, ‘Eragon’, ‘Eldest’), storytelling (‘story’, ‘write’), and specific references to Christopher Paolini’s works (‘Paolini’, ‘Eragon’, ‘Eldest’). It suggests a thematic focus on fantasy fiction, narrative construction, and exploration of the fantastical worlds and characters within the novels of Christopher Paolini, particularly his “The Inheritance Cycle” series.

These topics can serve as valuable insights for customer segmentation, allowing marketers and publishers to better understand readers’ preferences, interests, and sentiments towards different aspects of books. In conclusion, the topic modeling analysis provides a deeper understanding of the underlying themes and sentiments within the book reviews, offering actionable insights for customer segmentation and strategic decision-making in the publishing industry.

K-Means Clustering

```
library <- c("kableExtra", "NbClust", "Rtsne", "factoextra")

# install.packages(library)
```

```
# Install the Rtsne package if not already installed
if (!requireNamespace("Rtsne", quietly = TRUE)) {
  install.packages("Rtsne")
}

# Load the Rtsne package
library(Rtsne)
```

```
# Select only the features 'Rating' and 'Book_Price'
features <- bookdata %>%
  select(Rating, Book_Price)

# Perform k-means clustering with k = 5 (K value was determined using the books Rating range
0 to 5)
set.seed(123) # Set seed for reproducibility
k <- 5
kmeans_result <- kmeans(features, centers = k)

# Add cluster assignments to the dataset
bookdata$cluster <- as.factor(kmeans_result$cluster)

# Interpretation of the clustering result
# 1. Visualize the clusters
library(ggplot2)
ggplot(bookdata, aes(x = Rating, y = Book_Price, color = cluster)) +
  geom_point() +
  labs(title = "K-Means Clustering of Book Reviews",
       x = "Rating",
       y = "Book Price") +
  theme_minimal()
```


```
# 2. Assess the cluster centers
cluster_centers <- as.data.frame(kmeans_result$centers)
cluster_centers$cluster <- factor(1:k)
names(cluster_centers) <- c("Rating", "Book_Price", "Cluster")
print("Cluster Centers:")
```

```
## [1] "Cluster Centers:"
```

```
print(cluster_centers)
```

```
##     Rating Book_Price Cluster
## 1 3.929019   30.99493       1
## 2 4.520325    7.67000       2
## 3 4.501529   20.67569       3
## 4 1.962963    7.67000       4
## 5 4.252669   14.27868       5
```

```
# 3. Evaluate cluster sizes
cluster_sizes <- table(kmeans_result$cluster)
print("Cluster Sizes:")
```

```
## [1] "Cluster Sizes:"
```

```
print(cluster_sizes)
```

```
##
##   1   2   3   4   5
## 479 123 327 135 281
```

```
# 4. Assess cluster characteristics
library(dplyr)
cluster_characteristics <- bookdata %>%
  group_by(cluster) %>%
  summarize(mean_rating = mean(Rating),
            mean_price = mean(Book_Price),
            num_reviews = n())
print("Cluster Characteristics:")
```

```
## [1] "Cluster Characteristics:"
```

```
print(cluster_characteristics)
```

```
## # A tibble: 5 × 4
##   cluster mean_rating mean_price num_reviews
##   <fct>         <dbl>      <dbl>       <int>
## 1 1              3.93      31.0          479
## 2 2              4.52       7.67         123
## 3 3              4.50      20.7          327
## 4 4              1.96       7.67         135
## 5 5              4.25      14.3          281
```

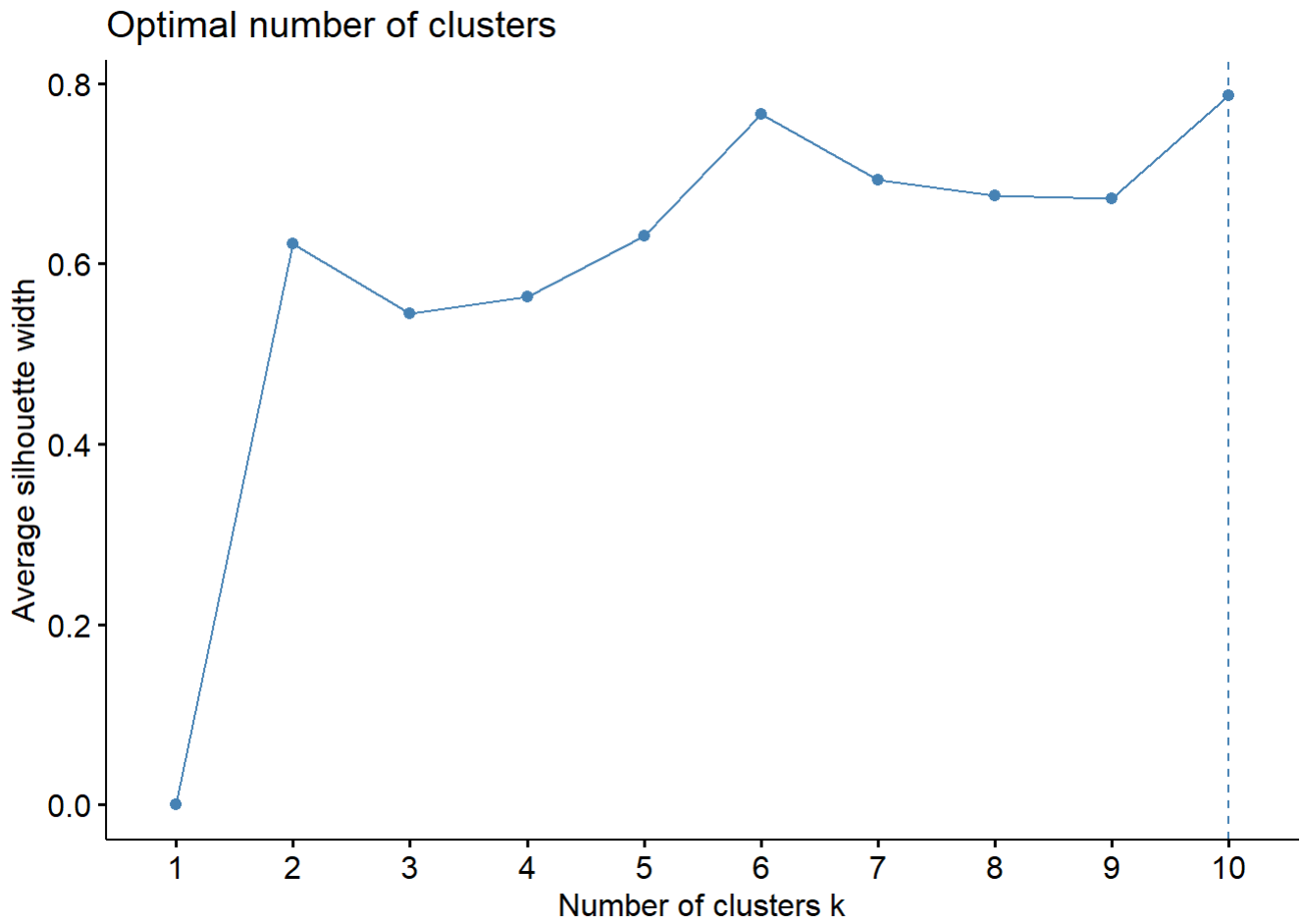
```
# 5. Assess cluster separability
# Silhouette analysis
library(cluster)
silhouette_score <- silhouette(kmeans_result$cluster, dist(features))
print(paste("Silhouette Score:", mean(silhouette_score)))
```

```
## [1] "Silhouette Score: 2.12711402415794"
```

```
# Cluster validation metrics
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
fviz_nbclust(features, kmeans, method = "silhouette")
```



Cluster Sizes:

Cluster 1: 479 observations Cluster 2: 123 observations Cluster 3: 327 observations Cluster 4: 135 observations Cluster 5: 281 observations

Silhouette Score: The silhouette score is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). A higher silhouette score indicates better-defined clusters. In this case, the silhouette score is 2.12711402415794, which is relatively high and suggests that the clusters are well-separated.

The aim of perform K-Clustering is to investigate trends in book reviews by combining ratings and prices, and to recognise unique groups of books based on these characteristics.

Insights:

Price vs. Rating Relationship: We may examine whether there is a correlation between a book's rating and price by grouping books according to both. For instance, are books that cost more generally rated higher, or is there no discernible relationship between the two?

Book Segmentation: K-means clustering divides books into groups according to how similar their ratings and costs are. Understanding the various book market segments—such as high-priced books with poor ratings, mid-priced books with moderate ratings, and low-priced books with good ratings—can be gained from this segmentation.

• Discussion of future work.

Predictive Modelling: Using the features in the dataset, create predictive models to estimate book sales or anticipate client preferences. Techniques like time series forecasting, classification algorithms, and regression analysis may be used in this.

Recommendation Systems: Create customised systems for book recommendations depending on user behaviour and preferences. Users can be given personalised book recommendations by experimenting with collaborative filtering, content-based filtering, or hybrid recommendation techniques.

Data Integration and Collection: To enhance the quality of the current dataset and the resilience of data mining models and analytics, it is imperative to consistently gather fresh data from diverse sources, including social media, online forums, and e-commerce platforms.

Deployment and Integration: To support strategic initiatives, maximise marketing efforts, boost customer satisfaction, and ultimately improve corporate performance, integrate data mining models and insights into decision-making procedures and business operations.

You can further use the insights from data mining analyses to drive innovation, extract important knowledge, and make well-informed decisions across a range of domains and businesses by pursuing these lines of future work.

References:

https://www.youtube.com/watch?v=ELct2RRENQM&list=PLjXODJ_lGN_WtxhPsQ_t0aHtFAcsIh1-8 (https://www.youtube.com/watch?v=ELct2RRENQM&list=PLjXODJ_lGN_WtxhPsQ_t0aHtFAcsIh1-8) - LDA Topic Modeling

https://www.youtube.com/watch?v=mQLXR_LaGes (https://www.youtube.com/watch?v=mQLXR_LaGes) - Sentiment analysis on text data

Aggarwal, C. C., & Zhai, C. X. (Eds.). (2018). Mining text data. Springer.

Kamps, J., Marx, M., & de Rijke, M. (Eds.). (2021). Information retrieval and data mining. Springer.

Leskovec, J., Rajaraman, A., & Ullman, J. D. (2020). Mining of massive datasets. Cambridge University Press.

Tan, P. N., Steinbach, M., & Karpatne, A. (2019). Introduction to data mining. Pearson.

Weiss, S. M., Indurkhya, N., & Zhang, T. (2015). Fundamentals of predictive text mining. Springer.

Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., … & Steinbach, M. (Eds.). (2017). Top 10 algorithms in data mining. CRC Press.

Zaki, M. J., & Meira Jr, W. (2020). Data mining and analysis: Fundamental concepts and algorithms. Cambridge University Press.