OMIS 670

SOCIAL MEDIA ANALYTICS PROJECT REPORT

TOPIC

AMAZON KINDLE REVIEWS

GROUP - 10

TEAM DETAILS

Z1889112 – VIJETA MISHRA

Z1801262- EUGENE CIESLA

Z1875854 -HETAL PANCHAL

TABLE OF CONTENTS

EXECUTIVE SUMMARY……..……....2
CONTEXT……..2
DATASET DESCRIPTION……3
Data Description……3
Data Pre-processing……4
Tools, Techniques, and Algorithms……4
MODELS……....5
Sentiment Analysis……..5
Topic Modeling……...5
Visualization……...6
ANALYSIS……6
Sentiment Analysis……6
Topic Modeling……12
Visualizations……....18
FINDINGS……19
APPENDIX……...21

EXECUTIVE SUMMARY

This project is intended to study the kindle book reviews and ratings for the timeline 2016 to 2019. Amazon Kindle Store is an e-book e-commerce store. Online reviews are a category of product information created by users based on personal handling experience. The online reviews have huge effect on consumer purchase behavior. As customer purchasing decision also depends on the opinion or reviews of others who have bought the products already, in this project we will show the quantity of positive and negative reviews which may impact the customers and business.

CONTEXT

Industry**:** eBook

Business Scenario**:** We are a startup book publishing company planning to get into market with a collection of physical copies of some of the most selling and high rated e-books as a part of which we are looking to learn more about the book industry by looking at kindle books review data. We will be looking at the dataset to find out the books with most positive reviews, books with low ratings and negative reviews. We will also try to find out the most liked genre/category/topic of books to help us figure out the categories in which we want to publish more books.

Challenges**:** Missing Product ID and product data table. Even if the information is missing, we assume that with the correct table we will be able to relate the product ID to the actual book.

Audience**:** Ourselves as business starters, stakeholders in the startup such as publishers and investors.

DATASET DESCRIPTION

Data source**:** https://www.kaggle.com/bharadwaj6/kindle-reviews

Rows and columns**:** The sample dataset has 2000 rows and 10 columns.

Data Description

The Kaggle dataset that has been used in this assignment is the Amazon Kindle Reviews. The dataset contains approx. 1million text reviews on books written by Amazon Kindle users along with an explicit rating between 1-5. Since this dataset has about 1 million records, we have taken a sample of 2000 reviews for the purpose of this analysis for faster processing. Below is the overview of the dataset and variables details that we are using for our analysis.

asin	product_id	helpful_rating	overall_rating	reviewText	reviewTime	reviewerID	reviewerName	summary	ReviewTime
87113	B005BREOCW	[0, 0]	5	Beast is the ma	07 14, 2012	A3IY6YW4I9K3I	Natalie	Awesome	1342224000
389900	B00AS0EU5I	[0, 0]	5	they feel or if	02 19, 2014	3E258BRKRVCO	ECH	BEAUTIFUL!!	1392768000
931972	B00KCCVB9S	[2, 2]	5	. Beautiful pict	05 17, 2014	UN5VQN0AD6O	delghani rhom	Wide Selection	1400284800
840391	B00ILX9OWS	[0, 0]	4	ers from her pr	04 2, 2014	23U2IKBE5NET	E. "SEWONN R	Good fast read.	1396396800
534790	B00D09W84Y	[1, 1]	5	mething about	11 29, 2013	W3E5VRRMQ8I	jackie	Fire HD User's	1385683200
604247	B00EA45DUO	[1, 1]	5	m there. I fell in	09 21, 2013	2FC3XQY0SWIS	es (My Not So	some New Adu	1379721600
2939	B001QIGZY0	[1, 1]	2	re to pull out o	03 31, 2009	1GARI2JT6EAW	eviews "DWD's	l then it just.....	1238457600
692548	B00G2571ZE	[2, 2]	1	are simplistic ar	04 25, 2014	ABW8ZL231G2T	Ellen	d I didn't even	1398384000

product_id - ID of the product helpful_rating - helpfulness

rating of the review - example: 2/3. overall_rating of the

product.

reviewText - text of the review (heading). reviewTime -

time of the review (raw). reviewerID - ID of the reviewer,

like A3SPTOKDG7WBLN reviewerName - name of the

reviewer summary - summary of the review (description).

ReviewTime - unix timestamp.

Data Pre-processing**:** Data available on Kaggle is highly unstructured therefore, for this project we have used R for sampling and tidying the data, i.e., renaming columns, deleting rows that had NA/Null values and python for preprocessing i.e.,

- Remove leading and ending white space.
- Remove all Punctuations.
- Remove all numbers and non-alphabetical words.
- Lowercasing all words
- Remove all English stop words.
- Lemmatization to reduce words to their base form.

(All the codes are posted in the appendix section of the project.

Tools, Techniques and Algorithms

- Data Extraction and Data Wrangling- R
- Sentiment Analysis: Python (nltk.sentiment.vader library)
- Topic Modelling: SAS
- Visualization: Tableau, R
- Word Cloud: Python (wordcloud library, matplotlib.pyplot library)

MODELS

Sentiment Analysis

Sentiment analysis also known as opinion mining is the interpretation and classification of emotions i.e., positive negative and neutral from a series of words or text data using text analysis techniques.

For our project we have performed "Sentiment Analysis" on review text column using python which resulted in three sentiments the positive sentiment, negative and the neutral sentiment based on the polarity of the text. This will be the most important factor in deciding which books have better reviews and which books can be recommended audience.

Topic Modeling

Topic modelling is a statistical model that specifically help you to understand abstract topic of the dataset which you have collected. Basically, there are two types of topic modelling one which refer to identifying major themes and replacing them to word cloud to help and understand the recurring themes. Second is which words are important to label for the topics. In order to use any method, we need to first count the number of words which can help us to determine the topics.

Topic modeling is a machine learning model that trains a system to classify records into specific categories based off the words used in a string. It brings together similar words and creates topics that act as bins for new records.

In this case, SAS Enterprise Miner (SAS EM) was used to train and analyze the data for topic modeling. This allowed for reviews that had similar words to be grouped together. For example, if reviews had words like “diet,” “exercise,” or “health” then it can be assumed that these reviews are for books that belong to a fitness topic.

SAS EM allows the data to be manipulated using several different nodes. After loading in the data, the first node used was text parsing. This splits the text into readable bits of information. Following the parsing was the text filter node. This node identified and took out stopping words and unnecessary words. The final node was text topic which took the filtered data and place the records into bins as topics based on what words were used. No topic was defined, but similar words are grouped together that can then be identified as a specific topic.

Visualization

We used tableau to visualize the review dataset and displayed all the information in Figure. In the dashboard, we covered major aspects of how book reviews play an important role in understanding customer behavior and supporting our startup company's growth. We have counted reviewers in the dashboard based on the book rating provided and calculated the sentiment score of each review given by the reviewer. We also created a bar chart and pie chart to emphasize the different types of reviews, such as negative, positive, and neutral.

ANALYSIS

Sentiment Analysis

The visual (Figure 1) shows the overall percentage of reviews per rating. Most of the reviews have five-star rating which is 59% of the total ratings followed by 4-star rating which is 25.6 % of the total ratings. This shows that we have a good distribution of ratings across the books in our dataset. We have used R studio to create this figure, R code for it is in the appendix section.

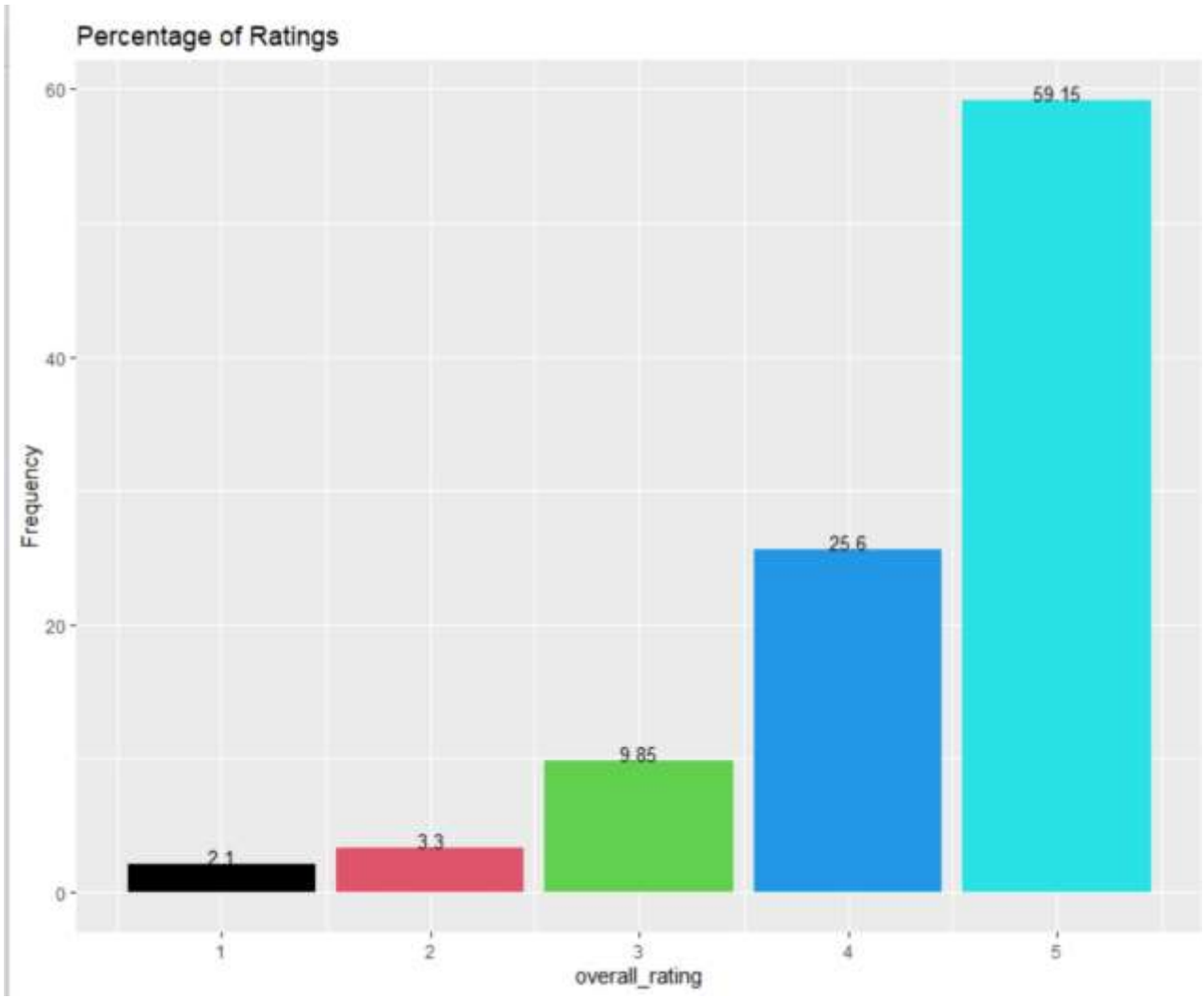


Figure 1: Percentage of reviews per rating

The below figure1.1 shows the percentage of positive, negative, and neutral reviews based on calculated sentiment t score. Sentiment score less than - 0.05 is marked as negative, score more than 0.05 is marked as positive and scores between –0.05 and 0.05 is marked as neutral. Here we see that 89.6% of the total reviews have positive sentiments, 8.5% has negative sentiment and 1.8% of the total has neutral sentiment. R studio was used to create the figure, R code for the below figure is in the appendix section.

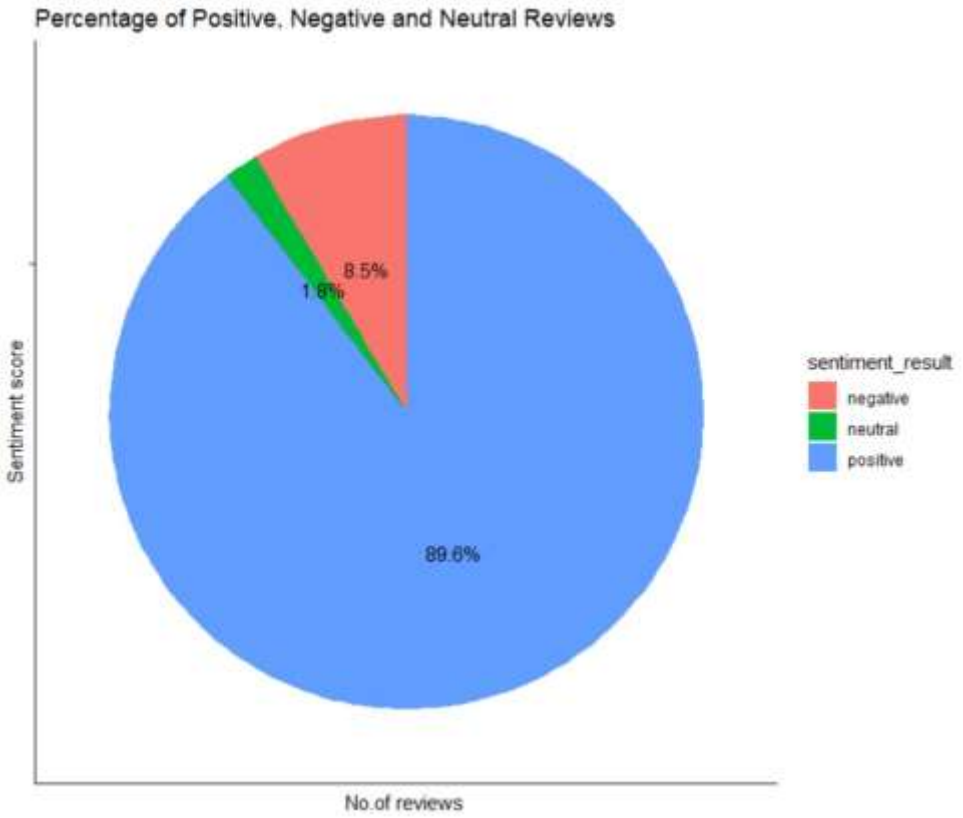


Figure 1.1: Review Sentiment Percentage

The below Figure 1.2 shows the top 5 books/ asin number with highest sentiment score. The dataset has asin number, we used the website. To get the book name from amazon. This will help us

to know the bestselling books in the market. R studio was used to create the figure, R code for the below figure is in the appendix section.

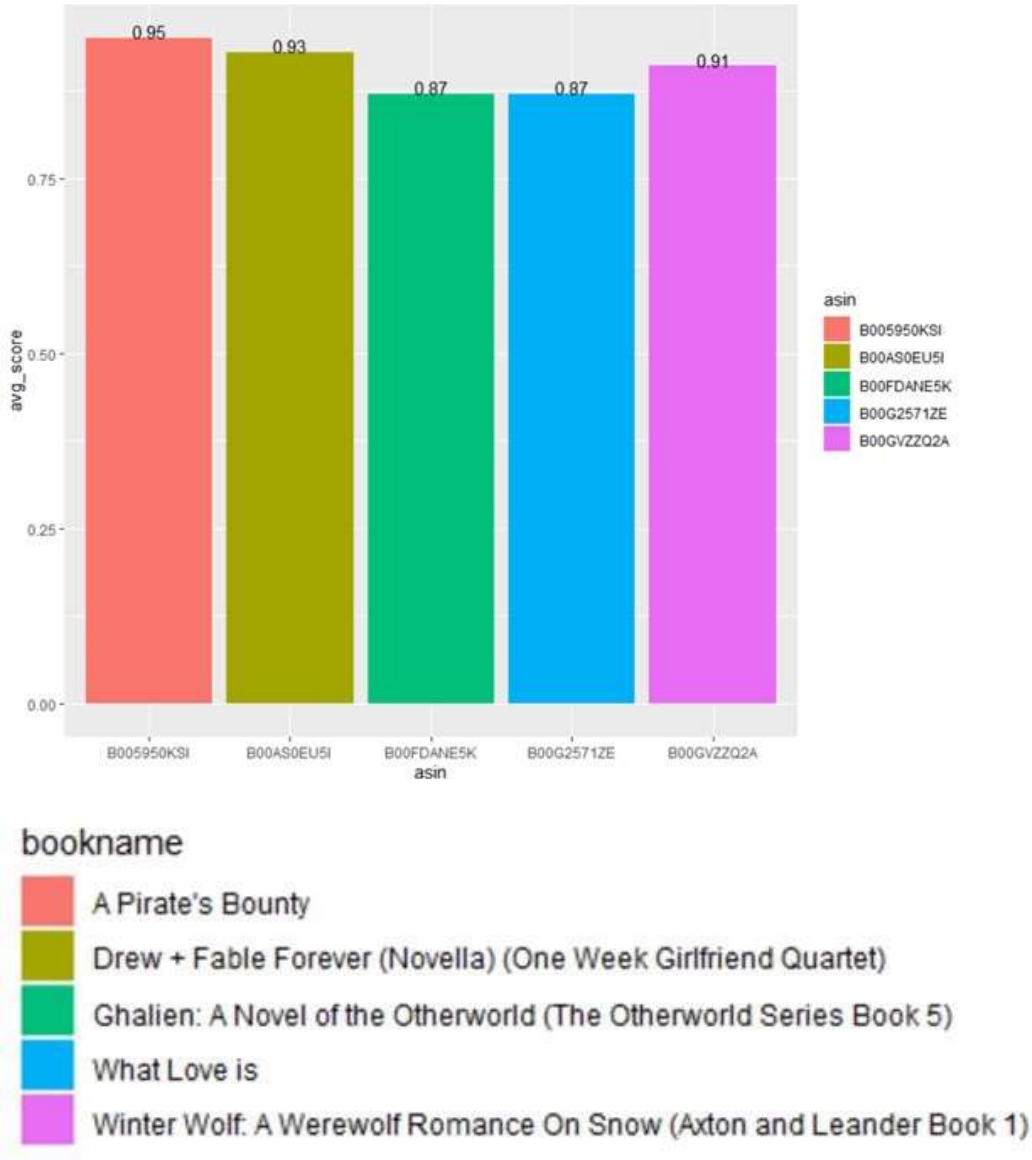


Figure 1.2: Books with Highest Sentiment Score

The Figure# 1.4 below gives us the data on the bottom 5 books that has the least average ratings. This will help us to figure out which books are not liked by the customer or books that needs

more marketing. R studio was used to create the figure, R code for the below figure is in the appendix section.

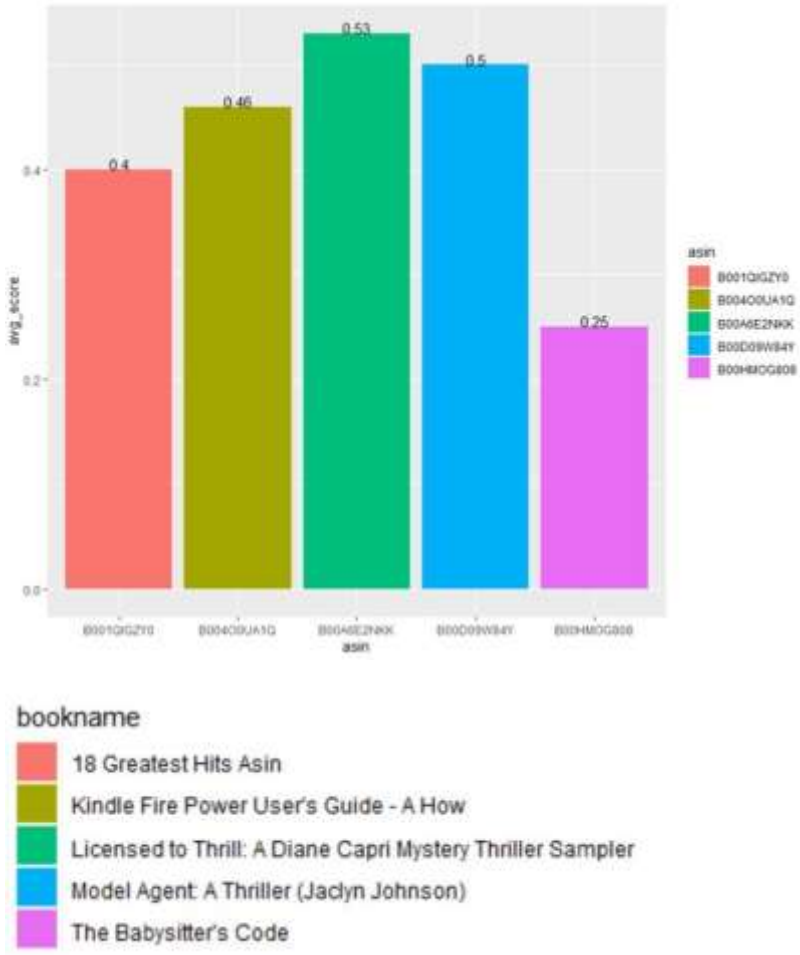


Figure 1.3: Books with Highest Sentiment Score

The below figures are the word cloud, we have used python to create all the below figures with sentiment analysis score. The first figure 1.4 shows the most frequently used words in the text review of a kindle book. As it is a word cloud of all the reviews, it provides us a general idea of the review sentiment of the user/reviewer. The word frequency in the figure gives greater prominence to

words that appear more frequently in a review text. The larger the word in the visual the more common the word was in the review from the user.

Figure 1.4: Most frequent word in reviews

The below figure 1.5 shows most frequent words used in reviews with positive sentiment scores. This gives us an idea of which books or books of which genre are more liked by the readers. Words that can be seen in the below figures and can be related to genre are romance, love, life, loved, happy, hope etc. So, when it comes to recommending books or publishing books to catch the market, this would be best option to start with.

Figure1.5: Frequent positive word in reviews

The below figure 1.6 shows most frequent words used in reviews with negative sentiment scores. This gives us an idea of which books or books of which genre are not liked by the readers. Words that can be seen in the below figures and can be related to genre are romance, fire, fear, horror, broken etc. Books with such title or genre should be avoided.

Figure 1.6: Frequent negative word in reviews

Topic Modeling Analysis

In SAS EM, the topic modeling node separated the reviews into 25 bins. It does this by looking at the terms in the reviews and putting together like words. These bins contain anywhere from 16-60 reviews. When displaying the results, SAS EM chose to name the topics with the grouping of terms. For example, the first topic is “short, +short story, +story, +good, +want” which can be understood under a more comprehensive name “short stories.” Each record in the data set is a single review which is labeled as a document in SAS EM.

Once the dataset has been thoroughly analyzed with SAS EM, we will dive deeper into visualizing and taking into consideration all other factors for further analysis on topic modeling with R programming. Considering Figure 2, this what are the most frequent word which are repeated in

the review are tracked using below R script and they are displayed using bar charts. Above are the top 10 words which are frequently found in the book review. Based on that we can perform topic modelling to understand what topic people more interested into.

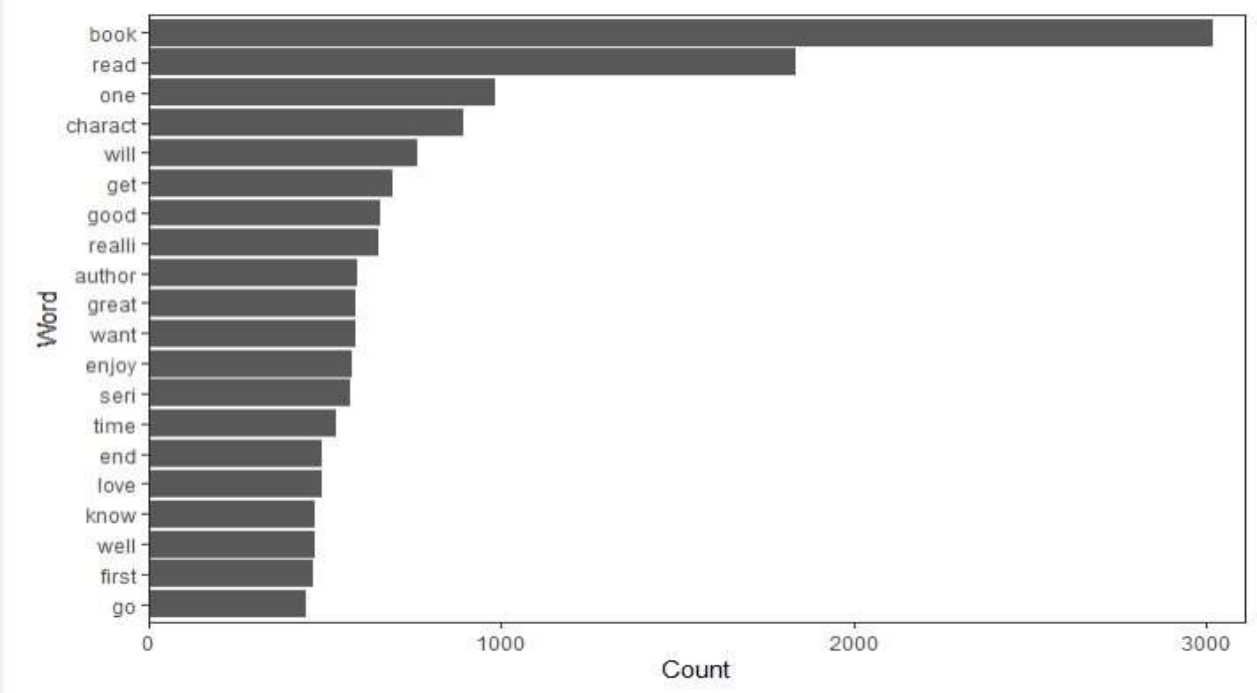


Figure 2: Term count using R script

Once the preprocessing is performed, we have transfer term frequencies into a documentterm matrix (DTM). The values of the matrix are respective frequency of each term. Based on that we need to define topic for this review. For that we have performed Latent Dirichlet Allocation (LDA) for finding number of topics related to this. Now that we've built a dtm object, we can move on to implementing the topic modeling driving force. As topic mentioned in each of the document, we can determine how they are associated with other topics. Considering this we can determine number of topics as per the need to start with we have selected 6 topics. Each topic has several terms associated with it, each with its own beta showing how much it refers to that topic.

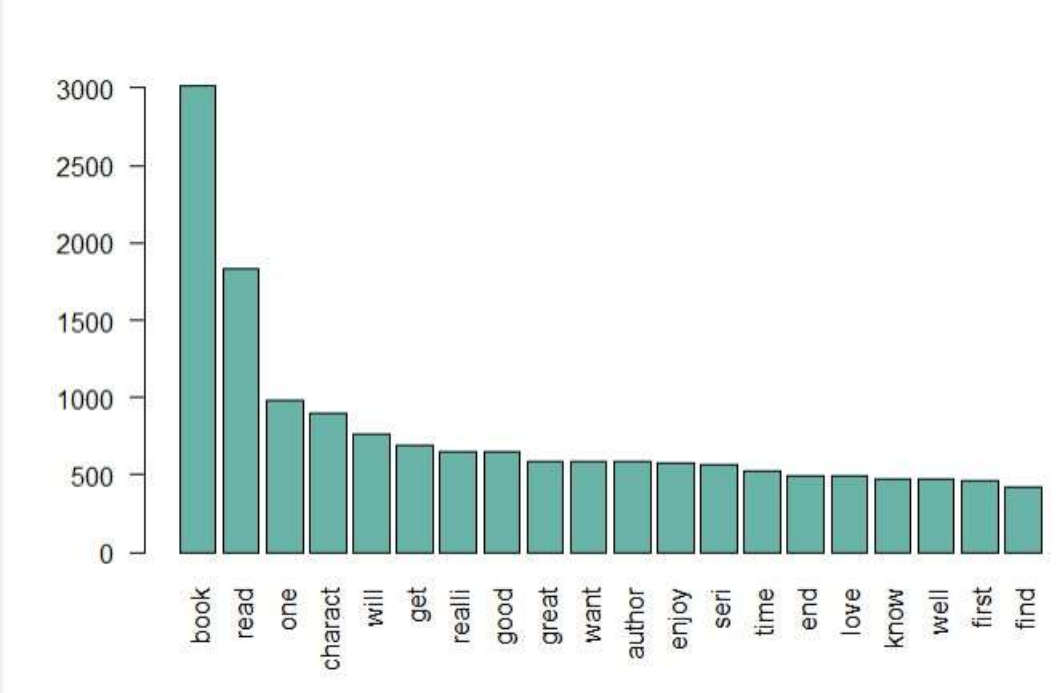


Figure 2.1: Frequently viewed words in reader's review

In Figure 2.1, there are 10 most frequent words which are used to determine the topic modelling. It is the word count which is creating using word cloud package for the R program.

Figure 2.2: Word Cloud for most frequently used words

As mentioned, we have used Latent Dirichlet allocation (LDA) which is the most common algorithm for topic modelling. Each review includes feedback that is shared across all topics When a customer shares a review, there are many terms that are listed that help to explain the topic. For e.g., if there are two topics on which we have a lot of word count, consider 30 percent for Topic A and 70 percent for Topic B, this can be set aside. Based on that LDA is a statistical method for estimating each of these at the same time: calculating the mixture of terms associated with each subject as well as the mixture of topics that characterize each text. We have used LDA () function along with that k= 6 to create six topic model of the reviews present. This form, named ("beta"), is provided by the tidy text package for extracting the per-topic-per-word probabilities from the model.

topic <int>	term <chr>	beta <dbl>
1	book	0.029328210
2	book	0.012038395
3	book	0.028766730
4	book	0.057605099
5	book	0.011081987
6	book	0.025668161
4	enjoy	0.010337162
4	seri	0.013456847
3	love	0.009775879
1	read	0.028346317

Figure 2.3: Performed LDA for topic modelling

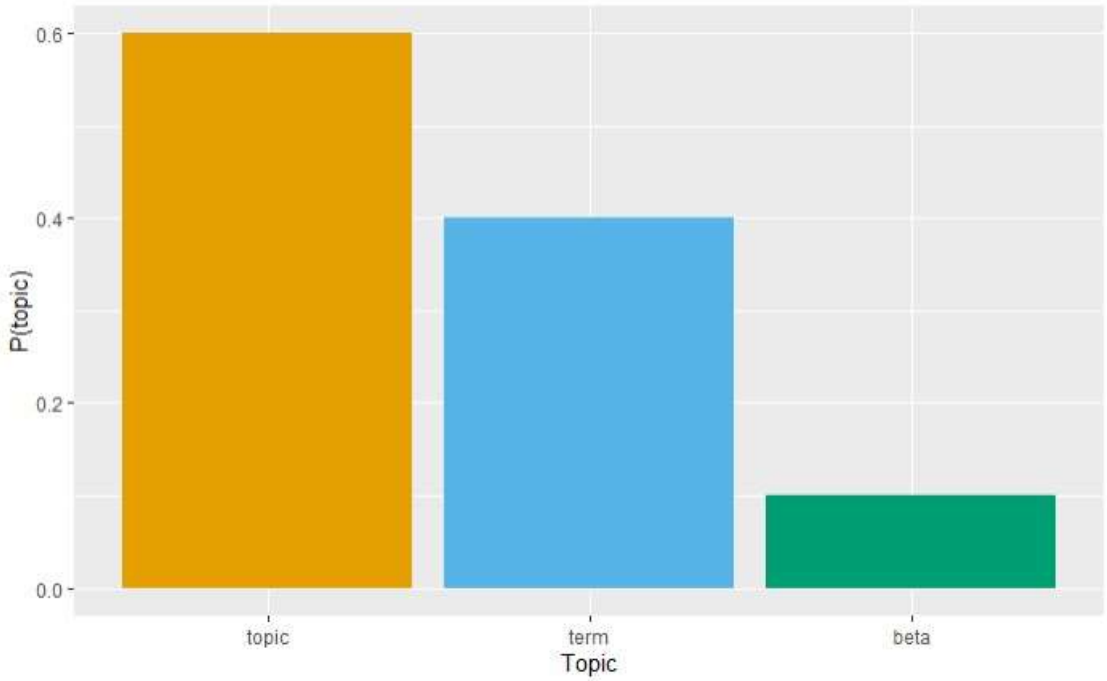


Figure 2.4: Comparing topic, term and beta for distribution

Let's look at the topic six modeling visualization that was extracted from the analysis dataset. The words "good," "great," and "enjoy" are often used in topic 1, implying that the book review is positive and that many people enjoyed all the books in this category. Consider topic 2 “character”, “vampire”, “life” it gives idea about which character is being love most by the people. One of the most important observation is “book”, “read”, “character” is the common word in Topic 1 and Topic 2. Topic modeling has an advantage over “hard clustering” approaches in that natural language topics can have some overlap in terms of words.

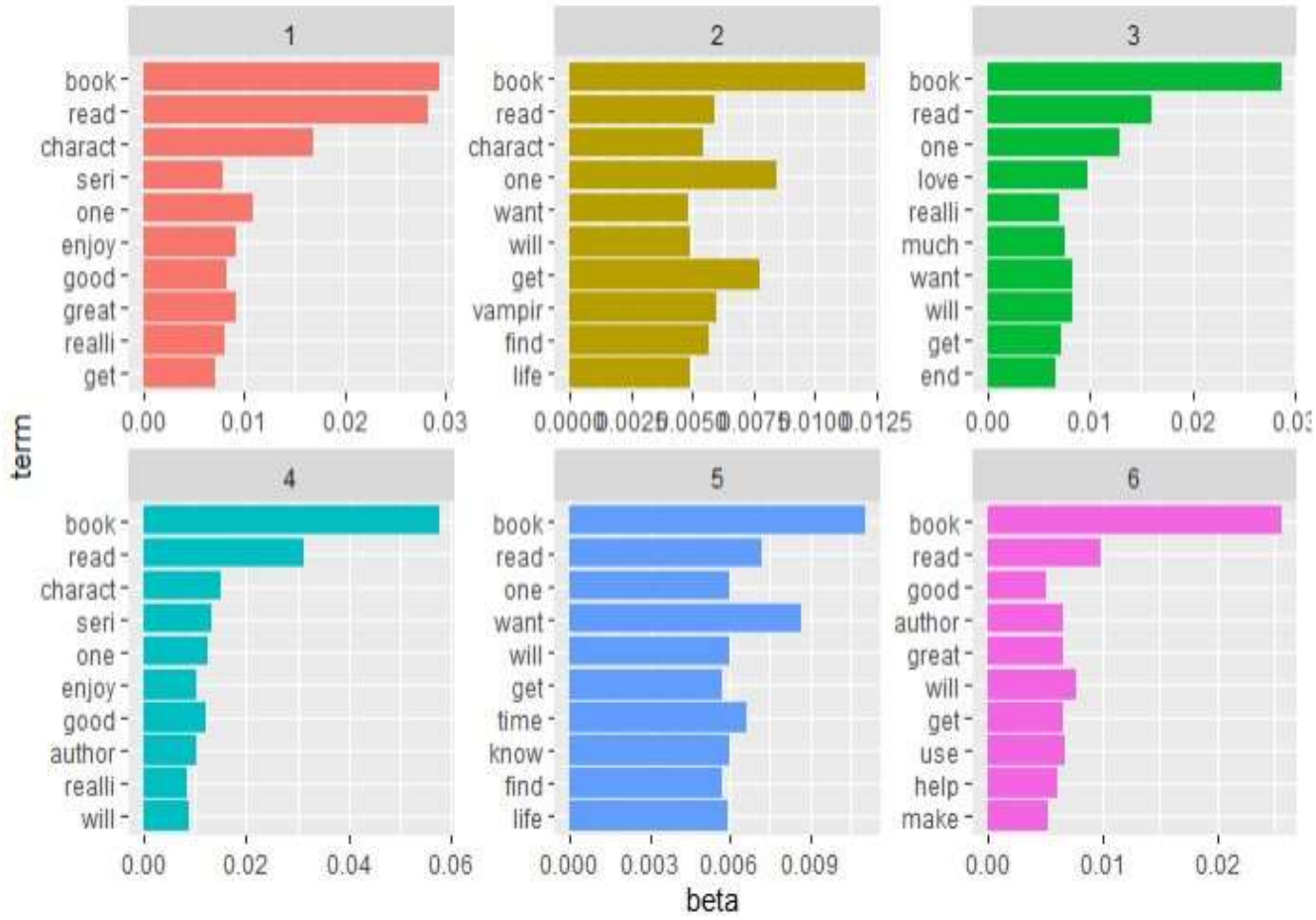


Figure 2.5: Topic modelling on review using LDA

In the Figure 2.5 graph we can see there are 6 topic which are used. Higher the ranking will be more will be the probable the word will belong to many topics. There are couple of depending on topic which we can see from this review process. In order to combine those topics, we need to create Dendrogram as an analyst to show association between those related topics. A Dendrogram determines whether two topics are related by using Hellinger distance (the distance between two probability vectors).

```
review_tdm2 <- removeSparseTerms(review_tdm, sparse = 0.9)
hc <- hclust(d = dist(review_tdm2, method = "euclidean"), method = "complete")
# Plot a dendrogram
plot(hc)
```

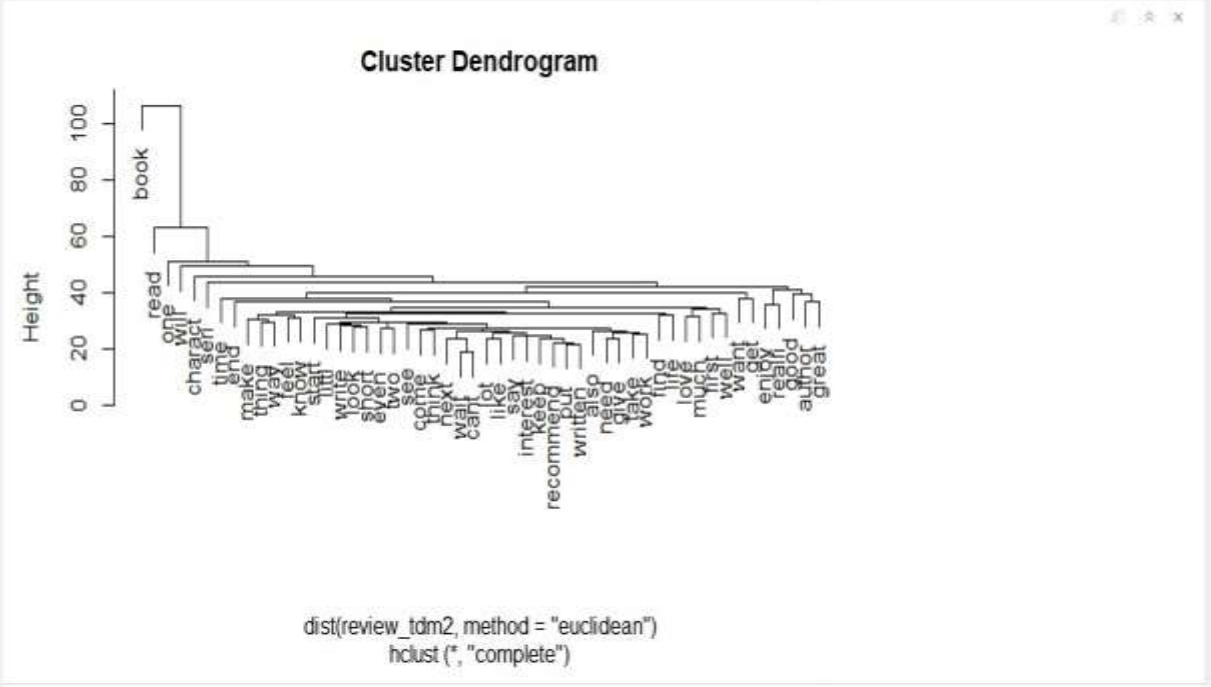


Figure 2.6: Dendrogram for most frequent words

Visualization:

Figure 2.7: Visualization for Amazon kindle review

Figure 2.7 shows a list of books available for purchase on Amazon Kindle. Essentially, this visualization helps us understand four charts. When I first uploaded the data, I cleaned it by changing the data type and splitting it into valid formats so that we could analyze it.

- The line graph on the left side of the dashboard represents the count of reviewers based on rating, which helps in determining the sentiment score of the customer's overall review.
- The next chart, toward the top right side, is a pie chart that displays all the products rated from highest to lowest.
- The bar charts show the yearly sentimental analysis distribution based on the number of products provided to customers.
- The entire dashboard assists in the analysis of reviewer data and product analysis based on rating.

FINDINGS

Question1- What are the most liked book titles that we should consider publishing as a hard copy, and which one should we avoid?

Inference: To evaluate the most liked books we can refer to Figure 1.6, which shows the top 5 high rated books with highest sentiment score and figure 1.7 shows the book titles with least sentiment score. This is a way to know the books that most liked by people. This visualization also gives an idea of which books should we consider printing to get hold of market.

Recommendation- We should consider the books shown in figure 1.6 and similar books of same genre and should avoid books with negative sentiment. Books that are already have a good market and are liked by people are best genre to start with the publications. We can also look at figure 1.6 which is a word cloud figure that shows us the most used words in review with positive sentiment analysis score. Words like romance, love, hope indicates that these are most like genre, publishing book of these genre can be very profitable. On the other hand, figure 1.7 which is a word cloud of negative sentiment review contains words like bad, fear, wrong, sad, fire etc. Books of such genre should be avoided.

Questions 2: How can we personalized recommendations of the books to reader based on their pervious reading habits?

Inference: To evaluate the number of reviewers who rated between 1 and 5, including each reviewer's sentiment score, so that we can provide personalized recommendations on books to the reader. In the figure above, we performed preliminary analysis to count the number of reviewers who have read a specific book, as well as the ratings provided by those reviewers based on their reading. Once we have analyzed this, we need to know which books the reader is interested in so that we can recommend similar books in the future. Also, once the reader's rating is obtained, we can calculate the total count of sentiment analysis for it. Based on this analysis, we can see that the total number of reviewers who have given a rating of 5 is 1,178, and we can derive that whoever has rated this book 5 has a total sentiment analysis count of 899.07. This encourages us to personalize the user experience by recommending books of a particular genre.

Recommendation: Based on the review, we can also provide them with a comparison for other similar books. This will assist us in better understanding customer reading patterns, which will benefit the company's sales. It will also assist us in keeping track of books that require physical copies. It provides stability in managing space management and maintaining good relationships with readers and vendors.

Questions 3: If we are considering expanding a premium subscription service where readers can receive a free book every few months, how can we decide which book would be the best option?

Inference: The approach used would be to give the same book to all of the subscribers. This book selected based on good reviews and relevance. It would be similar to finding a personalized book recommendation, but it would be for all readers as a collective. The first step in choosing this people’s choice book would be to filter down to only the reviews with positive sentiment analysis. Next would be to set a filter based on a minimum number of those positive reviews. A book could have a perfect score, but if it is only reviewed by 10 people, that is not as valuable. The minimum number can be set at 50 reviews. Next would be to filter those reviews down to the last 3 months

to make sure that the selection is relevant. Once the data is filtered, the last pieces is to select top rated books that have been trending with positive reviews. The selection may not be the book with the most positive reviews, but it should be the one that has had the greatest increase in positive reviews in the past 3 months.

Recommendation: Installing this subscription service is beneficial to the customer because they get a deal on their purchases and a free book every few months. It is beneficial to the company because it opens up a new revenue stream with minimal cannibalization of their other avenues. Additionally, by picking the trending book based on the readers' opinions, the company can expand sales on a product that might not otherwise be as popular.

APPENDIX

```
def data_preprocess_and_tokenize(df):
    result_list = []

    for r in df.iterrows():
        sentence = r[1].reviewText

        # Remove leading and ending whitespace
        sentence = sentence.strip()

        # All lowercase
        sentence = sentence.lower()

        # Remove punctuations
        sentence = sentence.translate(str.maketrans('', '', string.punctuation))

        # Remove numbers
        sentence = sentence.translate(str.maketrans('', '', '0123456789'))

        # Create tokens for the text
        tokens = nltk.word_tokenize(sentence)
        result_list.append(tokens)
    return result_list
```

Figure 4: Python Code for reading the data- preprocessing.

```
def remove_stopwords(sentence_list):
    stops = set(stopwords.words("english"))
    for tokens in sentence_list:
        for token in tokens:
            if token in stops:
                tokens.remove(token)

def lemmatize_sentence(sentence_list):
    lemmatizer = nltk.wordnet.WordNetLemmatizer()
    for tokens in sentence_list:
        for i, token in enumerate(tokens):
            tokens[i] = lemmatizer.lemmatize(token)
```

Figure 4.1: Code to remove stop words.

```
def generate_sentiment_score(df, sentence_list):
    siaObject = SentimentIntensityAnalyzer()
    sentiment_score = []
    sentiment_result = []
    for tokens in sentence_list:
            score = siaObject.polarity_scores(" ".join(tokens))
            compound_score = score['compound']
            sentiment_score.append(compound_score)
            if(compound_score < -0.05):
                sentiment_result.append('negative')
            elif (compound_score > 0.05):
                sentiment_result.append('positive')
            else:
                sentiment_result.append('neutral')

    df.insert(10, "sentiment_score", sentiment_score)
    df.insert(11,"sentiment_result", sentiment_result)
```

Figure 4.2: Code to calculate sentiment score.

```
def plot_word_cloud(sentence_list):
    all_sentences = []
    for tokens in sentence_list:
        sentence = " ".join(tokens)
        all_sentences.append(sentence)

    wordcloud = WordCloud().generate(" ".join(all_sentences))
    plt.imshow(wordcloud)
    plt.axis("off")
    plt.show()
```

Figure 4.3: Code for word cloud (All reviews)

```
def wordcloud_positive_negative_words(sentence_list):
    siaObject = SentimentIntensityAnalyzer()
    positive_freq = defaultdict(lambda: 0)
    negative_freq = defaultdict(lambda: 0)
    for tokens in sentence_list:
        for token in tokens:
            score = siaObject.polarity_scores(token)
            compound_score = score['compound']
            if (compound_score < -0.05):
                negative_freq[token] +=1
            elif (compound_score > 0.05):
                positive_freq[token] +=1

    wordcloud = WordCloud(background_color="white", max_words=1000)
    wordcloud.generate_from_frequencies(positive_freq)
    plt.imshow(wordcloud)
    plt.axis("off")
    plt.show()

    wordcloud.generate_from_frequencies(negative_freq)
    plt.imshow(wordcloud)
    plt.axis("off")
    plt.show()
```

Figure 4.4: Code for word cloud (Positive and Negative reviews)

```
##Visualization 1

by_rating <- kindle_reviews_with_sentimentscore %>%
  group_by(overall_rating) %>%
  summarise(count= n()) %>%
  mutate(percentage = count/sum(count)*100)

ggplot(data = by_rating, mapping = aes (x= overall_rating, y=
percentage))+
  geom_col(stat="identity", fill= by_rating$overall_rating)+
  ggtitle("Percentage of Ratings")+ geom_text(aes(label= percentage),
vjust=.02, color="black", size=3.5)+
  labs(x = "overall_rating", y = "Frequency", fill = "percentage")
```

Figure 5: Code for visualization in R – Percentage of review per rating

```
##Visualization 2

by_review <- kindle_reviews_with_sentimentscore %>%
  group_by(sentiment_result) %>%
  summarise(count= n()) %>%
  mutate(percentage = count/sum(count)*100)

ggplot(data=by_review, mapping=aes(x="",y=percentage,fill =
sentiment_result)) +
geom_bar(stat="identity")+
ggtitle("Percentage of Positive, Negative and Neutral Reviews")+
geom_text(aes(label=paste0(round(percentage,1),"%")), position =
position_stack(vjust= 0.5)) +
coord_polar("y",start=0)+
theme_classic()+
theme(axis.text = element_blank()) + xlab("Sentiment score") +
ylab("No.of reviews ")
```

Figure 5.1: Pie chart percentage of sentiment scores.

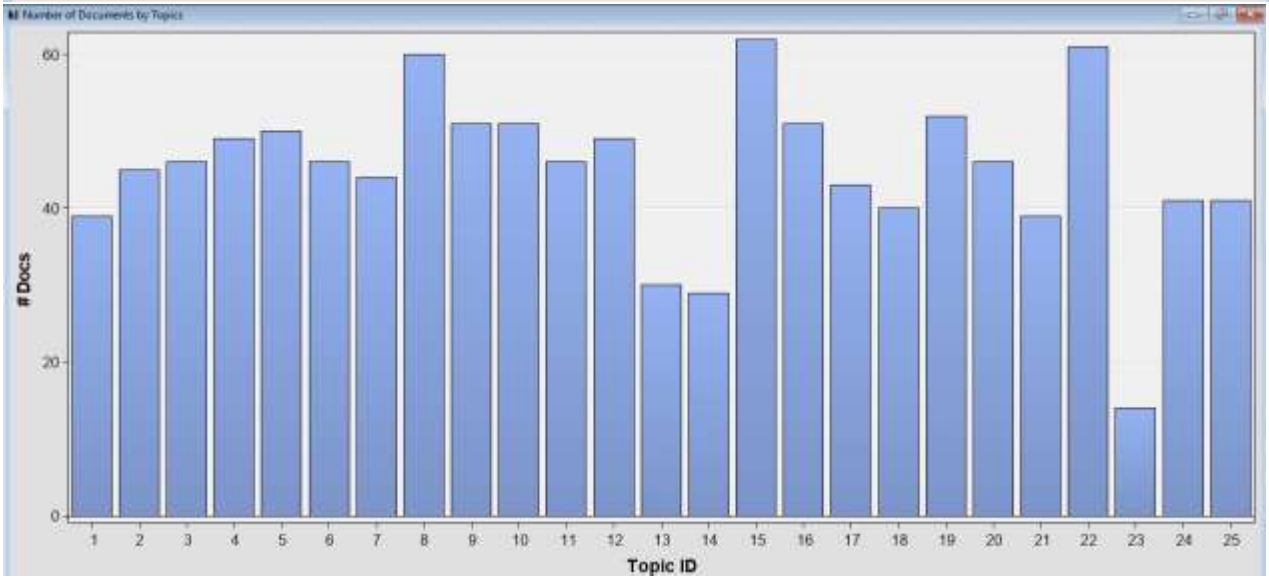
```
kf <-kindle_review_final %>% group_by(asin) %>% summarise(avg_score=
round(mean(sentiment_score),2)) %>% arrange(desc(avg_score))

kf_5 <- top_n(kf, 5)

ggplot(data = kf_5)+
  geom_col(mapping = aes(x= asin, y= avg_score, fill=asin))+
  geom_text(aes(x = asin, y = avg_score, label= (avg_score),
vjust=0.02))

kf1 <-kindle_review_final %>% group_by(bookname) %>%
summarise(avg_score= round(mean(sentiment_score),2)) %>%
arrange(avg_score)
kf1_5 <- top_n(kf1, -5)

ggplot(data = kf1_5)+
  geom_col(mapping = aes(x= bookname, y= avg_score, fill=bookname))+
  geom_text(aes(x = bookname, y = avg_score, label= (avg_score),
vjust=0.02))
```

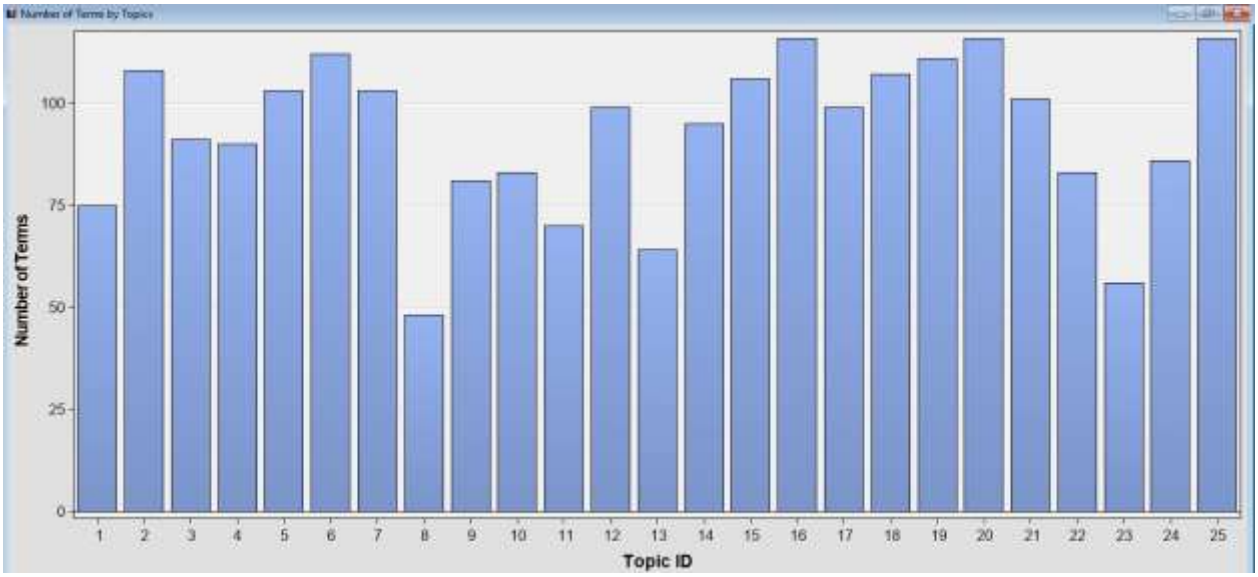
Figure 5.2- Top 5 Books with highest and top 5 books with lowest sentiment score

SAS EM Diagrams

Topics

Category	Topic ID	Document Cutoff	Term Cutoff ▼	Topic	Number of Terms	# Docs
Multiple	1	0.113	0.046	short,+short story,+story,+good,+want	75	39
Multiple	2	0.090	0.046	+woman,+guy,+brother,+break,+man	108	45
Multiple	3	0.102	0.046	+write,+style,little,+author,writing	91	46
Multiple	4	0.089	0.046	+reader,+vampire,+look,+feel,+pull	90	49
Multiple	5	0.105	0.046	first,+page,interesting,series,+novel	103	50
Multiple	6	0.077	0.046	+good,+dragon,+action,+line,+good	112	46
Multiple	7	0.092	0.046	+end,+disappoint,+happen,especially,+start	103	44
Multiple	9	0.100	0.046	+keep,edge,excellent,+page,+seat	81	51
Multiple	10	0.096	0.046	+character,+main character,main,+plot,alex	83	51
Multiple	11	0.092	0.046	+hot,+scene,sex,sex,+heroine	70	46
Multiple	12	0.103	0.046	+holiday,christmas,fun,+heart,first	99	49
Multiple	14	0.078	0.046	+review,honest,+dragon,+honest review,+copy	95	29
Multiple	15	0.083	0.046	+good,+know,+end,+man,+start	106	62
Multiple	16	0.087	0.046	+know,always,+people,+thing,+scene	116	51
Multiple	17	0.088	0.046	+child,+year,+young,school,+old	99	43
Multiple	18	0.084	0.046	+life,+happen,+book,+help,find out	107	40
Multiple	19	0.095	0.046	series,+great,+love,first,first	111	52
Multiple	20	0.093	0.046	+feel,+great,+time,faith,+author	116	46
Multiple	21	0.094	0.046	highly,+recommend,+great,second,+life	101	39
Multiple	22	0.102	0.046	love,love story,+fall,great,+love	83	61
Multiple	24	0.095	0.046	ending,+happy,+great,happy ending,reading	86	41
Multiple	25	0.093	0.046	+friend,+relationship,+move,+continue,jake	116	41
Multiple	8	0.134	0.045	+wait,next,series,+enjoy,+book	48	60
Multiple	13	0.109	0.045	+recipe,+look,wonderful,+idea,+book	64	30
Multiple	23	0.093	0.044	kindle,hd,fire,+include,price	56	14





Topic modelling and visualization using R:

```r
# Count of Word
corpus_review <- Corpus(VectorSource(data_review$reviewText))
corpus_review=tm_map(corpus_review, tolower)
corpus_review=tm_map(corpus_review, removePunctuation)
corpus_review=tm_map(corpus_review, removeWords, stopwords("english"))
corpus_review=tm_map(corpus_review, removeWords,c("love", "story","like", "excit", "made", "can", "satisfy", "dress",
"just", "i"))
corpus_review=tm_map(corpus_review, stemDocument)
corpus_review[["8"]][1]
term_count <- freq_terms(corpus_review,Top=100)
plot(term_count)
```

Figure 2.8: Word count

```r
review_dtm <- DocumentTermMatrix(corpus_review)
review_tdm <- TermDocumentMatrix(corpus_review)
# Convert TDM to matrix
review_m <- as.matrix(review_tdm)
# Sum rows and frequency data frame
review_term_freq <- rowSums(review_m)
# Sort term_frequency in descending order
review_term_freq <- sort(review_term_freq, decreasing = T)
# View the top 10 most common words
review_term_freq[1:10]
```

Figure 2.9: Filtered most frequent word

```r
review_dtm <- DocumentTermMatrix(corpus_review)
review_tdm <- TermDocumentMatrix(corpus_review)
# Convert TDM to matrix
review_m <- as.matrix(review_tdm)
# Sum rows and frequency data frame
review_term_freq <- rowSums(review_m)
# Sort term_frequency in descending order
review_term_freq <- sort(review_term_freq, decreasing = T)
# View the top 10 most common words
review_term_freq[1:10]
```

```r
# Print the word cloud with the specified colors
wordcloud(review_word_freq$term, review_word_freq$num,
  max.words = 150, colors = c("aquamarine","darkgoldenrod","tomato"))
```

Figure 2.10: Word Cloud

```r
install.packages("lda")

sample_lda <- LDA(review_dtm, k = 6, control = list(seed = 1234))

chapter_topics <- tidy(sample_lda, matrix = "beta")

top_n(chapter_topics, 20)
```

Figure 2.11: Latent Dirichlet allocation (LDA)

```r
top_terms <- chapter_topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)


top_terms %>%
  mutate(term= reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```

Figure 2.12: Topic Modelling to analyze their association or relationship with one another