SENTIMENT ANALYSIS OF THE FICTIONAL BOOKS AND THE MOVIES ADAPTED FROM THEM.

By

AJAY VERMA

Masters in Data Analytics

2023

A Thesis submitted in partial fulfillment of the requirements for the degree of

Masters in Data Analytics

Dundalk Institute of Technology

Dundalk, Ireland

SENTIMENT ANALYSIS OF THE FICTIONAL BOOKS AND THE MOVIES ADAPTED FROM THEM.

by

AJAY VERMA
Masters in Data Analytics

2023

A Thesis submitted in partial fulfillment of the requirements for the degree of
Masters in Data Analytics
Dundalk Institute of Technology
Dundalk, Ireland

Approved by	Signatures	Date
Dr. Natalia, Budarina Supervisor		

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisor, Dr. Natalia Budarina, for her unwavering support, guidance, and valuable feedback throughout the entire dissertation process. Her expertise and dedication played a pivotal role in shaping this research. I also extend my heartfelt thanks to the members of my dissertation committee, Dr. Abhishek Kaushil and Dr Jack Mc Donnell, for their insightful comments and constructive criticism, which greatly improved the quality of this work. I am deeply appreciative of Dundalk Institute of Technology, which allowed me to conduct the research on such subjective topic.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The success of the movies or the books totally depends upon what are the reviews/feedback of the readers or viewers who have already read the book or watched the movies. These reviews not only decide the success of the books or the movies but also helps the writers and the film crew to improve in their next one. As per the article by Hannah Rothwell Films which are adapted from books usually generate 53% of more revenue than the original titles which are produced every year. This phenomenon is observed because people want to witness the role that is depicted in the book in real life for what they have imagined that character would be. Our study solely focuses on the sentiment of the viewers and readers about the movies which are adapted from the books by considering the book's top 10 reviews and the viewer's top 10 reviews. Along with the analysis of the re-view our paper aims to display the visual analysis of the feedback's sentiments and visualization techniques.

CHAPTER 1

INTRODUCTION

As per the article published by Dean Talbot [1] There are around 4 million books published every year. The first question arises What are the chances of those books being adapted into a movie? This depends upon the feedback of the first 10 book enthusiasts. If the reviews are good a greater number of people will buy the book thus increasing the sale of the books and if the sales and the story catch the eye of the film producers, they eventually try to adapt the same into the movie. Both of the sections i.e., movies and books are classified into positive or negative categories by reviewing the feedback. This feedback is directly connected with the words occurring in the review text, whether these words have been used in the positive context or the negative context. Our study investigates the patterns and trains the ML model to analyze the viewer's and readers' reviews.

In the year 2022 goodreads.com which is the famous book review platform surpassed 65 million reviews whereas with the IMDB there is n number of reviews updated every hour. Humans are really subjective with their opinions they reflect their satisfaction with the product. And being able to get those details provides an advantage over multiple factors such as what aspect of that product requires an improvement whether being marketing or any quality of the product. Movie and book review reflects the writer's opinion positively or negatively helping others to understand the plot of the story that the characters are trying to display.

Customer Feedback visualization [2] is necessary as it affects the review analysis and helps to extract which field is more appeasable and helps in the decision-making process. Sentiment Visualization is one of the fields of visualization that demonstrates the analysis of the feedback extracted from the user's feedback.

Research Questions The main challenge in the Sentiment Analysis is subject detection and emotion detection as it is really difficult to understand the sarcasm or context through the test. SA can also be termed as deriving subjective data from concrete data on different levels.

CHAPTER 2

LITERATURE REVIEW

In the article [3] Author defines Sentiment Analysis as a process of the given text to analyze the emotions present in it which may be positive or negative feedback. There are multiple methods available that can be performed over these texts to extract the conclusion which requires knowledge of artificial intelligence, natural language processing, and Machine learning. The authors have performed the LSTM(Long Short Term Memory Networks) model for their research and compared it with other analysis models with the conclusion that LSTM yields the best results over the IMDB data sets.

Following the article [4] Authors state that Sentiment Analysis is one of the popular and simple tasks in Natural Language processing which predicts the polarity of the sentences or a review as positive negative or neutral. The authors concluded their research by combining three different models based on three different approaches which are generative, continuous, and clever reweighing of tf-idf(Term Frequency - Inverse Document Frequency) bag of words and found that the collective approach on the reviews proved to be a success on the overall system with more accurate results when compared to the individual results 2.1.

In the [5] authors used the standardized framework of tokenization, filtering, stemming, and classification for deriving the sentiments of the Movie reviews. For classification, the authors used 6 different methods Naive Bayes, Decision tree, random forest, Bayes Network, Support Vector Machine, and K nearest Neighbour from which the RF has more accurate results when compared to the other 5.

According to the authors of [6] Sentiment Analysis is usually used to predict the senti-

Table 2: Performance of Individual Models

Single Methods	Accuracy
N-gram	86.5%
RNN-LM	86.6%
Sentence Vectors	88.73%
NB-SVM Trigram	91.87%

Table 3: Performance of Different Model Combinations

Ensemble	Accuracy
RNN-LM + NB SVM Trigram	92.13%
RNN-LM + Sentence Vectors	90.4%
Sentence Vectors + NB-SVM Trigrams	92.39%
All	**92.57%**
State of the art	91.22%

Figure 2.1: Comparison of the results between individual models and collective models.

ments from a larger text content. So in order to classify the sentiments as positive, negative, or neutral the authors only used the Naive Bayes and Random Forest method. Based on the experimental setup and experiment conducted it was confirmed that the NB requires larger memory and is time-consuming while determining the sentiments when compared with the RF method.

In the [2] authors used different visualization techniques to classify the sentiments of the reviews provided on various books sold by Amazon. They used techniques such as Packed Bubbles for records of each book whereas a Line chart was used to represent the time of the published reviews, Stacked bars to represent the ratings of the books, and a word cloud to represent the set of words included in the books and also to represent which books has the highest number of reviews.

As per the [7] authors state that the SA is a text mining field of the research process which is growing rapidly. The process here authors follow are the collection of data -¿ Preprocessing Data (Removal of stop words, Extraction of text, POS tagging), Text Transformation, Feature Extraction, Classification and evaluation. The authors used the technique of Random

Forest to evaluate the polarity of positive and negative terms used in the reviews.

In the paper [8] author describes that Sentiment is usually either explicit or implicit and most of the time the analysis is performed on the first kind which is explicit where the opinion is directly highlighted by the person. 2.2 For example ("It was a nice play"). The author of this paper highlighted the Sentiment Analysis of the subjective side or thoughts. The paper concludes by sharing some light on the recent developments in the field of SA.

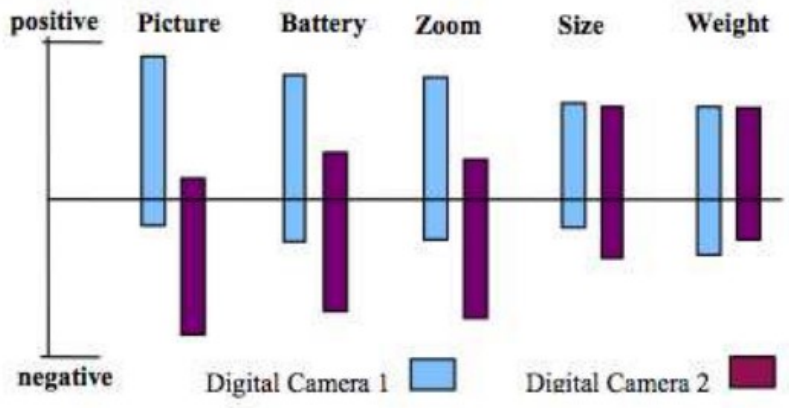


Figure 2.2: Visual comparison of consumer opinions on two products

If we look at article [9] Authors in this article define that Sentiment Analysis works with the user's point of view according to the subject with the polarity or interaction over a document or emotional review of the text. There are 3 approaches for Sentiment Analysis: Knowledge-based, Applied Math approach, and Hybrid Approach.

Knowledge-Based Technique: It distinguishes the text based on emotions i.e Happy, Sad, Afraid, or Bored. Applied Math Based Technique: This technique uses machine learning models like latent linguistics analysis, support vector machines, "Bags of words" and Linguistics. These methodologies involve the grammatical relationships of words square to extract the opinion. Hybrid Approach: It combines and maxes the advantages of each Machine learning model to extract the opinion of the users.

In the article Twitter Sentiment Analysis using NLTK and Vader sentiment [10] authors Shihab Elbagir and Jing Yang used NLTK and Vader analyzer to perform Sentiment Analysis

on the Twitter data which was related to the election tweets of the 2016 US presidential elections. At the end of their research, they were able to identify that the Vader Sentiment Analysis performed effectively for the tweets classification.

According to the article [11] authors C.J.Hutto and Eric Glibert executed the Vader classifier on the social media texts and found that the Vader algorithm outperformed human raters with a classification accuracy of 0.96 i.e. 96%. Figure 2 represents the performance details of Vaders when compared with other methods. As per the authors, Vader Classifier is not only self-contained but also domain agnostic which means it does not require an extensive set of training data yet it will perform well in diverse domains.

	Correlation to ground truth (mean of 20 human raters)	3-class (positive, negative, neutral) Classification Accuracy Metrics			Ordinal Rank (by F1)		Correlation to ground truth (mean of 20 human raters)	3-class (positive, negative, neutral) Classification Accuracy Metrics		
		Overall Precision	Overall Recall	Overall F1 score				Overall Precision	Overall Recall	Overall F1 score
Social Media Text (4,200 Tweets)							**Movie Reviews (10,605 review snippets)**			
Ind. Humans	**0.888**	0.95	0.76	0.84	2	1	0.899	0.95	0.90	0.92
VADER	0.881	**0.99**	**0.94**	**0.96**	1*	2	0.451	0.70	0.55	0.61
Hu-Liu04	0.756	0.94	0.66	0.77	3	3	0.416	0.66	0.56	0.59
SCN	0.568	0.81	0.75	0.75	4	7	0.210	0.60	0.53	0.44
GI	0.580	0.84	0.58	0.69	5	5	0.343	0.66	0.50	0.55
SWN	0.488	0.75	0.62	0.67	6	4	0.251	0.60	0.55	0.57
LIWC	0.622	0.94	0.48	0.63	7	9	0.152	0.61	0.22	0.31
ANEW	0.492	0.83	0.48	0.60	8	8	0.156	0.57	0.36	0.40
WSD	0.438	0.70	0.49	0.56	9	6	0.349	0.58	0.50	0.52
Amazon.com Product Reviews (3,708 review snippets)							**NY Times Editorials (5,190 article snippets)**			
Ind. Humans	0.911	0.94	0.80	0.85	1	1	0.745	0.87	0.55	0.65
VADER	0.565	0.78	0.55	0.63	2	2	0.492	0.69	0.49	0.55
Hu-Liu04	0.571	0.74	0.56	0.62	3	3	0.487	0.70	0.45	0.52
SCN	0.316	0.64	0.60	0.51	7	7	0.252	0.62	0.47	0.38
GI	0.385	0.67	0.49	0.55	5	5	0.362	0.65	0.44	0.49
SWN	0.325	0.61	0.54	0.57	4	4	0.262	0.57	0.49	0.52
LIWC	0.313	0.73	0.29	0.36	9	9	0.220	0.66	0.17	0.21
ANEW	0.257	0.69	0.33	0.39	8	8	0.202	0.59	0.32	0.35
WSD	0.324	0.60	0.51	0.55	6	6	0.218	0.55	0.45	0.47

Figure 2.3: Vader classification as compared to individual Human Raters and 7 established lexicon classifiers.

Purposed Scheme in [12] by authors Purtata Bhoir and Shilpa Kolte is Naive Bayes and SentiWordNet. The data set they used consisted of 5000 subjective and 5000 objective sentences. The result generated in their research concluded that the Naive Bayes Classifier produced more accurate results when compared with the SentiWordNet classifier.

In the research [13] authors Praveen Gujjar and Prasanna Kumar used another widely used package called TextBlob for the sentiment classification. They executed textblob for the market survey which proved to be successful for business intelligence in identifying weak and strong points of the business.

CHAPTER 3

DATA

To begin our research we had to extract or scrape the data from public domains. The process of extracting or scraping the data from websites into sheets or files is called data scraping. There are multiple libraries in Python to scrape the data such as Beautiful Soup, requests, Scrapy, selenium, urllib, etc. In order to scrape one has to learn the HTML document structure which can be a difficult task. We have scraped the raw data for this research purpose from two of the leading review platforms IMDB(Internet Movie Database) and the story graph. IMDB is one of the major data source and platform for most of the users who wants to know the movie information before viewing the movie. The domain contains data such as runtime, release time, trailers, reviews, and ratings. On the other hand, the domain story graph contains data about the novels that are published up to the current date. It contains data like author name, number of pages, release year, ratings, and reviews.

3.1 Scraping the data

There are multiple scrapes available as a library in Python for example Beautiful Soup, Scrapy, Selenium, Playwright, lmxl, urllib3, Mechanical Soup etc. For our research process, I have used Selenium, Beautiful Soup, and lmxl.

3.1.1 What is Scraping?

Web Scraping means the extraction of useful data from publicly available domains and websites and web scrapers like Scrapy, Selenium, lmxl, etc. are the tools used to scrape those

Table 3.1: Fictions with the respective movies

Fiction - Release Year	Movie - Release Year
Angles and Demons - 2000	Angels and Demons - 2009
Anna Karenina - 2004	Anna Karenina - 2012
BeoWulf - 2001	BeoWulf - 2007
The BGF - 1982	The BGF - 2016
The boy in the striped Pajamas - 2006	The boy in the striped Pajamas - 2008
Bridge to Terabithia - 1977	Bridge to Terabithia -2007

required data for research. Top companies like Google and Amazon provide these tools for free to the end users.

In order to scrape the data from both of these domains I had to verify the fiction titles which were adapted to the movies. For the verification, I have scraped the data from Wikipedia which is the database of all the events across the globe. The following table 3.1 shows some of the fiction which were adapted into the movies:

We have used a mixture of element selectors for extraction purposes for example we have used a regular class selector for the scraping of the Wikipedia domain whereas we have used xpath as a selector for the thestorygraph and the imdb domain. While extracting the data for Fictions from the storygraph domain at the first instance the domain only loaded the 7 reviews and the loading of the rest reviews were controlled through the pagination method(Javascript method for resource utilization). Hence to load more reviews one has to scroll to the bottom of the page. In order to tackle that situation we added a logic in the selenium as

```
driver.execute_script("window.scrollTo(0, document.body.scrollHeight) ")
```

Whereas while extracting the review from IMDB portal we had the same issue of pagination in order to reduce the infinite hit on the API the portal has added a button at the bottom clicking which the rest of the reviews load. In order to tackle the situation we implemented the click method of selenium as below:

```
WebDriverWait(driver, 60).until( EC.element_to_be_clickable((By.ID, 'load-more-trigger'))
```

sample class selector for Wikipedia using beautiful Soup:

`soup.find_all('h1', class_='wikitable td+ td a')`

sample class selector for IMDb and storygraph using xpath:

`(By.XPATH, '//div[@class="content"]')`

XPATH-XML Path Language:

It is based on the tree representational of XML structured documents which provides the ability to navigate across the structure by selecting nodes using different criteria. XPATH is the language used for the transformation or query of XML documents. It was defined in 1999 by W3C (World Wide Web Consortium) which can be used to extract values from the XML documents.

Sample Code for the extraction of data from the portal:

```
reviews_to_collect = 195
reviews_collected = 0
data = []
url = "https://www.imdb.com/title/tt0808151/reviews/?ref_=
   tt_ql_2"
try:
    driver.get(url)
    while reviews_collected < reviews_to_collect:
        try:
            loadMoreBtn = WebDriverWait(driver, 60).until(
                EC.element_to_be_clickable((By.ID, 'load-more-
                   trigger'))
            )
            if loadMoreBtn:
                loadMoreBtn.click()
                WebDriverWait(driver, 30).until(
```

```
                    EC.presence_of_all_elements_located((By.
                        XPATH, '//div[@class="content"]'))
                )
                reviews = driver.find_elements(By.XPATH, '//div
                    [@class="content"]')
                ratings = driver.find_elements(By.XPATH, '//
                    span[@class="rating-other-user-rating"]/span
                    [1]')
                for review, rating in zip(reviews, ratings):
                    review_text = review.text
                    rating_text = rating.text
                    if rating_text and review_text:
                        url_rating_review = (url, rating_text,
                            review_text)
                        data.append(url_rating_review)
                        reviews_collected += 1
                        if reviews_collected >=
                            reviews_to_collect:
                            break
        except Exception as e:
            print(f"Error␣while␣scraping␣{url}:␣{e}")
            break
except Exception as e:
    print(f"Error:␣{e}")
finally:
    driver.quit()
AngelsAndDemons_Movie = pd.DataFrame(data, columns=["mURL", "
```

```
mRating", "mReview"])
```

Listing 3.1: Extraction of data

A sample of the scraped data is as per the table below:

Table 3.2: sample of the scraped data for book:

bURL	bRating	gbReview
https....	3	I don't know ....
https....	3	Loved it!...
https....	4	"3.75âï¸ Robert Langdon is a...."
https....	4	Very good book!!....
https....	4	"Angels and Demons ...."
https....	4.5	"I read this way back ...."

The data Description is as below:

The data of the first book Angels and Demons is stored in the variable with 3 columns with the relevant fields which are URL, bRatings, and bReviews. URL (String Format): represents the URL from which the data was scraped. bRatings(String Format): represents how many stars out of 5 users have provided after reading the book. bReviews(String Format): represents the feedback that the users have provided over the portal.

In the case of the Movie reviews we still have the same 3 columns as of the fictions which are URL, mRatings, and MReviews. mRatings(String Format): represents the number of stars out of 10 viewers have provided after watching the movie. mReviews: represents the feedback that the users have provided over the portal.

Table 3.3: sample of the scraped data for movie:

mURL	mRating	mbReview
https....	7.0	Tom Hanks returns as Dan Brown's symbologist R...
https....	8.0	This is an OK adaptation of the breath taking ...
https....	5.0	I am sorry for all the readers, but I don't kn...
https....	7.0	As is the case with many films of this ilk, my...
https....	8.0	Symbologist Robert Langdon (Hanks) is called t...
https....	7.0	Personally, I enjoyed ANGELS AND DEMONS much m...

CHAPTER 4

METHODOLOGY

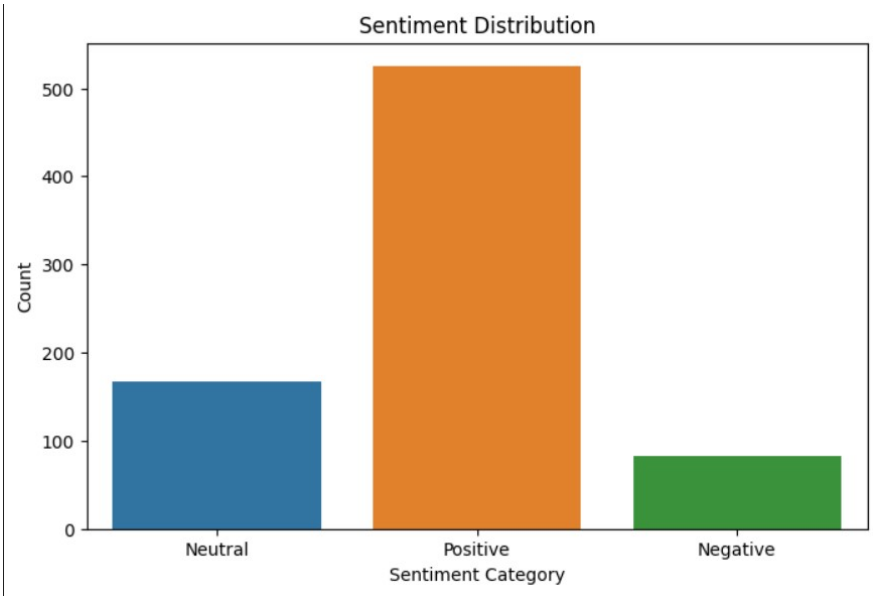
In order to proceed with the research we had to convert our scraped data into a valid format. Following Fig.1 Shows the tree structure of the SA process:

Data Cleaning and Structuring Before data is analyzed we had to check the data for any ambiguity like null values, blank spaces or any missing values. In order to prepare the data we have used a widely used library called pandas. The scraped data was in object format hence before analysis we had to convert the data in respective formats like rating has to be in float type, whereas we had to make sure the reviews were in string type since the analysis had to be performed on the reviews. sample codes for pre-processing data:

```
AngelsAndDemons_Book.info()
```

Using the above code we can confirm the data types and the structure of the data(number of rows and total entries) RangeIndex: 1000 entries, 0 to 999 Data columns (total 3 columns):
0 bURL 1000 non-null object
1 bRating 1000 non-null object
2 bReview 1000 non-null object dtypes: object(3)
If we consider the above sample we have 1000 rows and 3 columns in the data frame and all the data types is object. In order to check the null values we have used the following code:

```
missing_counts = AngelsAndDemons_Book.isnull().sum()
```



Result: bURL 0 bRating 0 bReview 0 dtype: int64

Converting the rating values to float:

```
AngelsAndDemons_Book['bRating'] = AngelsAndDemons_Book['bRating'].astype('float')
```

We create a line graph to check the what is the most rating the book has received. Fi

4.0.1 Data Preprocessing

Data Preprocessing is one of the crucial steps in the Sentiment Analysis process. It includes the following steps cleaning which we have already performed above, transformation, and organization.

Text Transformation

Removal of Emojis

While providing feedback we as a human have the tendency to add emojis to express our emotions about a product, service, book, or movie. As the sentiment analyzer may consider emojis as a noise to avoid such a situation we removed the emojis from the review.

Removal of Stop words:

Stop words are words that are considered to have little or no meaning in the Sentiment Analysis process. These words are removed to reduce the noise and improve the efficiency of the text analysis. Some common stop words are "a","at","the", "on", "it" etc.

In order to remove the stop words we used the library in Python called NLTK(Natural Language Processing Kit). NLTK has a stopwords collection which is in 16 languages. The library is imported as follows:

```
from nltk.corpus import stopwords
```

Tokenization

The next step in data preprocessing is Tokenization. Tokenization is a process of extracting individual words from a sequence of text. In other words, it means breaking long sequences of text into small individual meaningful elements. For this process, we again used the NLTK library. NLTK has a package called tokenize to convert the sequence into tokens. The library is imported as follows:

```
from nltk.tokenize import word_tokenize
```

The code snippet for this process is as follows:

```
1 def preprocess_text(text):
2     tokens = word_tokenize(text)
3     tokens = [word.lower() for word in tokens if word.isalpha()
        ]
4     stop_words = set(stopwords.words('english'))
5     tokens = [word for word in tokens if word not in stop_words
        ]
6     cleaned_text = ' '.join(tokens)
```

```
7       return cleaned_text
8   AngelsAndDemons_Book['PreprocessedReviews'] =
        AngelsAndDemons_Book['bReview'].apply(preprocess_text)
9   print(AngelsAndDemons_Book)
```

Listing 4.1: Preprocessing Data

4.0.2 Stemming:

This one of the process in data pre-processing which convert the tokenized word into shortened root words such as "Walked", "Walking" is transformed into the root word "Walk". We performed the stemming process on the PreprocessedReviews column and table 4.1 represents the output:

PreprocessedReviews	**StemmedReviews**
watched movie recently decided book surprised ...	watch movi recent decid book surpris much see ...
still shocked review come	still shock review come
angels demons thought decent read enjoyed char...	angel demon thought decent read enjoy charact ...
really smart written	realli smart written
stars remember anything previous read things m	star rememb anyth previou read thing may consi...

Table 4.1: Preprocessed Stemmed Reviews

If we look at the result the words are trimmed to a shorter version but some of the words will not make any sense in the analysis. Hence we will perform the classification based on the PreprocessedReviews column.

We have one more step in the data pre-processing before proceeding with the classification of the reviews is checking the language of the reviews. Since we are performing the data analysis on English reviews we wanted to remove any reviews which is not in English language. In order to do so we have used the lang detect library which is available in Python. To import such a library we use the following code:

```
from langdetect import detect
```

Now for the classification of the data we used the ratings of the users. We have classified the reviews as Positive, Neutral and Negative based on the ratings given by the individual users. For example, In case of books if the rating is 2.5(Since the users have rated the books out of 5) or lower it is sorted into the category of Negative category whereas if the rating is between 3.5 and 2.5 it is sorted as Neutral category and if the ratings are between 3.5 and 5 the reviews are sorted in Positive category. In the case of Movies where users have rated them out of 10 the classification is done as follows: ratings below 4.5 are classified as Negative whereas if the ratings are between 4.5 and 7.5 the reviews are classified as Neutral and in case the ratings are above 7.5 the reviews are classified as Positive. Sample for Positive review: The ending was so unexpected but also came together so perfectly. So much action and no minute to rest - Rated 4.75. Sample for Neutral review: It wasn't really that boring and I liked the setting but goodness, I couldn't wait to finally finish it! Not the best sign, usually I'm sad to finish a good book but this time turning the last page just filled me with joy - Rated 3 Sample for Negative review: It's such crap. It was painful to read the whole thing and yet I wanted to finish it because everyone I knew had given it 5 stars. Worse than a long boring episode of a police procedural, it contains unnecessary jargon, first scientific then architectural, then just Italian words. A lot of it, even plot-wise could've been edited out. Not thrilling. It's a boring fantasy. I was intrigued by the concept of religion vs science but this book doesn't do it enough justice. Rated 1
Figure 4.0.2 represents the bar graph based on the category distribution for the book dataset.

Figure 4.0.2 represents the bar graph based on the category distribution for the movie dataset.

We also created a word cloud for both of the scenarios that is fiction and movie in order to highlight the top words used to express the feedback.

Figure 4.0.2 represents the word cloud for the fiction Angels and Demons. Whereas the

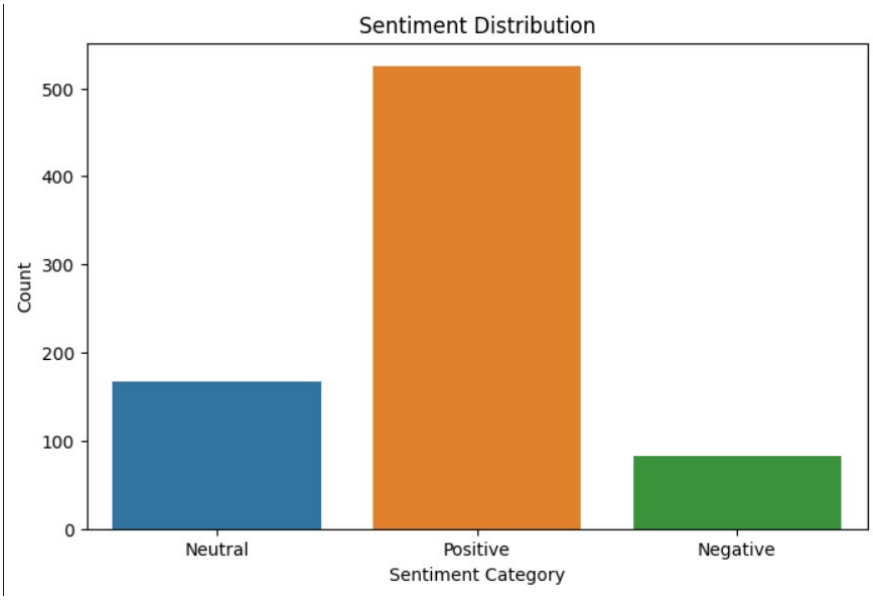


Figure 4.1: Category Count for Angels and Demons Fiction

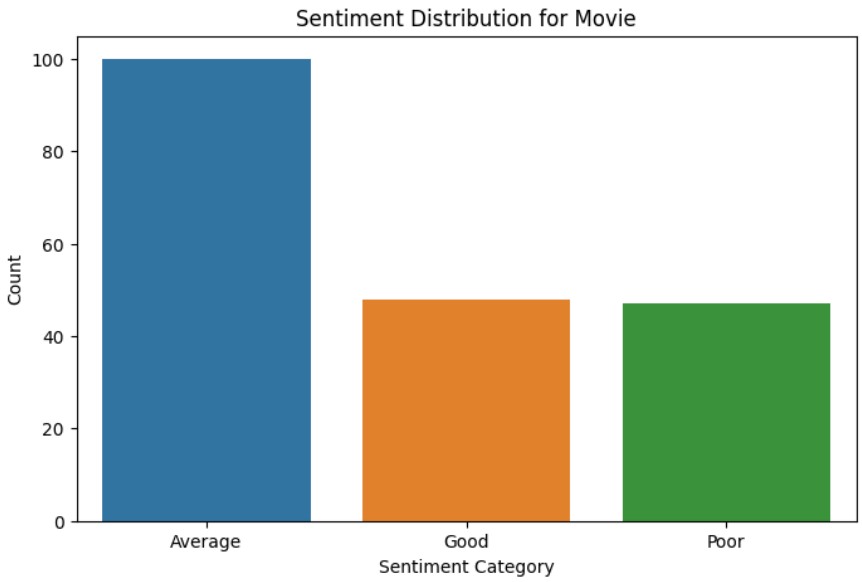


Figure 4.2: Category Count for Angels and Demons Movie

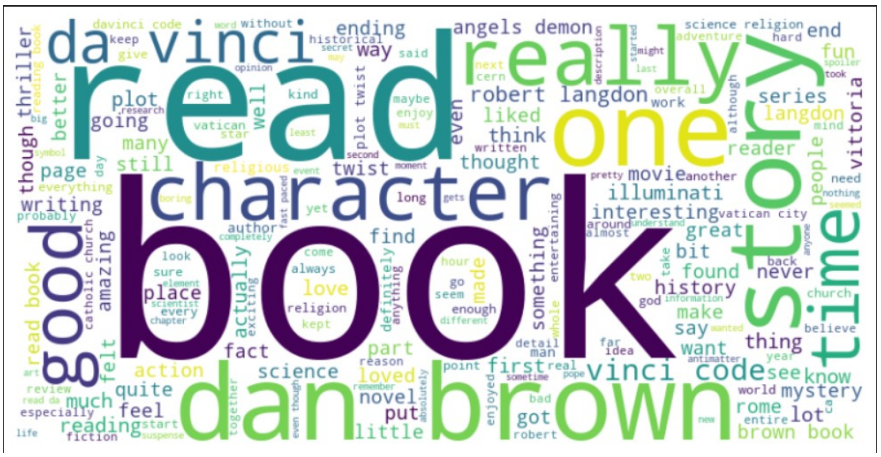


Figure 4.3: Word Cloud for the Fiction Angels and Demons

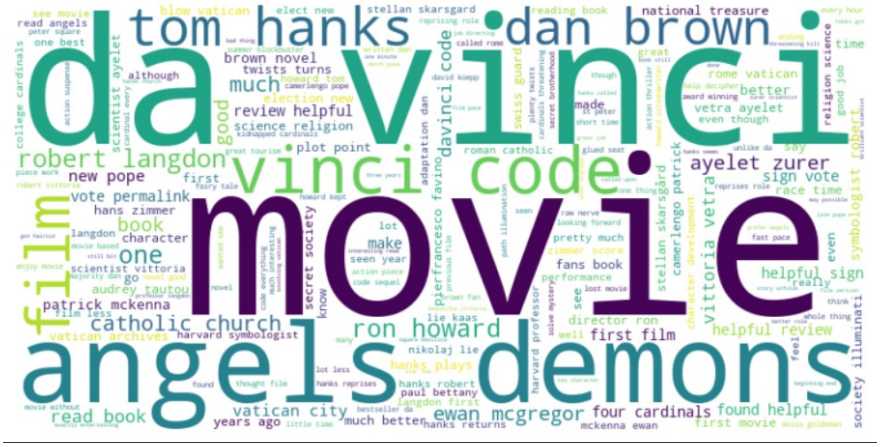


Figure 4.4: Word Cloud for the movie Angels and Demons

figure 4.0.2 represents the word cloud for the movie adapted from the fiction Angels and Demons.

4.1 Classification

As per the computer scientist Wolpert, David H. has mentioned in his article [14], The Lack of A Priori Distinctions Between Learning Algorithms. Neural Computation, that:

"For each problem, you must select the right algorithm. Your question is how to do this. If you have plenty of computational resources, you can test multiple algorithms and parameter settings. In this approach, the main question is how to estimate and compare the performance of the algorithms in a reliable way."

There are 3 main approaches for the Sentiment Analysis a. Lexical Based Approach b. Machine Learning Approach c. Hybrid Approach
But we will be using only Lexicon-based approaches live Vader and Text Blob for our research.

4.2 Lexicon Based Approach

In the Lexicon-based approach, the text is categorized into three polarities i.e. positive, negative, and neutral. The two main packages that are widely used for this approach are Vader and TextBlob.

4.2.1 Vader:

Vader uses a Lexicon-based approach for text analysis which is built for social media content. It is a mixture of Lexical heights which are highlighted by the semantic directions into the polarity of Positive, Negative and Neutral.

Working of Vader Classifier:

Word Level Analysis:

When a text is processed by Vader, it breaks the text into individual words and checks in its lexicon to retrieve its polarity score. Few of the words have already defined sentiment scores whereas for some of the words, the sentiment scores are calculated based on the grammar

and the context in which that word is used.

Sentiment Aggregation:

Sentiment polarity score is calculated based on the polarity score of each sentence of the text by aggregating the polarity scores of each word.

Emotional Intensity:

Once the polarity is calculated Vader classifies the text into one of the four categories i.e. Positive, Negative, Neutral, and Compound.

bReview	VASentiment
I've watched the movie recently so I've decide...	Positive
I am still shocked. Review to come! :)	Positive
Angels & Demons thought this was a dece...	Positive
Really smart written.	Positive
4.75 stars, this is a reread for me BUT I...	Positive
This book was lack luster. I'm here for action...	Negative

Table 4.2: Vader classification

```
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
```

4.2.2 TextBlob:

TextBlob is a library in Python that offers a simple API to perform normal basic NLP tasks. It performs different operations on the textual data such as nouns and phrases etc.

Working of the Text blob classifier:

Tokenization:

The first step of textblob is to tokenize the text into words and sentences.

Part of Speech Tagging:

TextBlob assigns a grammatical category to the text in the process of Part-of-speech tagging.

Sentiment Analysis:

One of the top features of TextBlob is that it has and utilizes a pre-trained model to categorize the sentiments which in return a polarity of the sentences.

```
from textblob import TextBlob
```

bReview	TBSentiment
I've watched the movie recently so I've decide...	Positive
I am still shocked. Review to come! :)	Negative
Angels & Demons I thought this was a dece...	Positive
Really smart written.	Positive
4.75 stars So, this is a re-read for me BUT I...	Positive

Table 4.3: TextBlob Sentiment

CHAPTER 5

RESULTS

5.1 CASE-1 - Angels and Demons

5.1.1 Vader Classification :

Using the Vader classifier we got the following results for the fiction Angles and Demons: Figure 5.2 represents how the categories are divided.

As per our research questions we wanted to calculate the success rate of the movies that were adapted from fictional books hence we created a wordcloud to highlight the positive words, negative words, and neutral words. Figure 5.3 represents all the positive words that were provided by the readers in their review. As we can see in the graph the top words are love, good, great, well, loved, enjoyed, etc. As we can already see the polarity of the reviews are mostly positive. Also, the mean of the collected ratings is 3.8605 for the raw dataset of Angels and Demons Fiction.

Whereas if we consider the figure 5.4 Vader was able to extract the negative words such as wrong, boring, doubt, ridiculous, stupid, dislike, terrible, etc.

Following the negative word cloud we have the graph of neutral wordcloud 5.5. Now the neutral words can be considered as positive ones as they do not truly depict the statement given by the readers are truly negative. The following table **??** represents the top 10 words of each category which the reviews depict in data set.

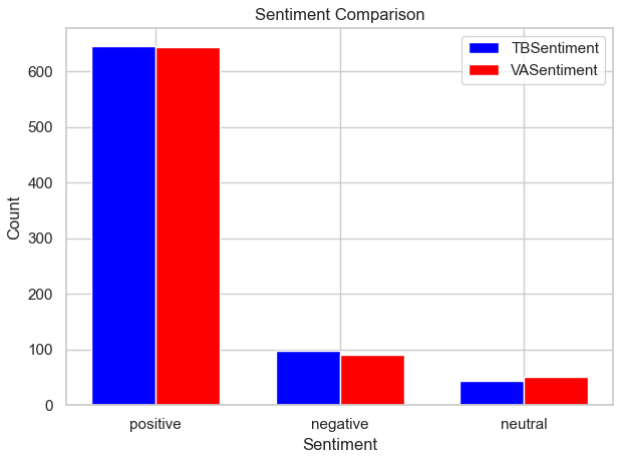


Figure 5.1: Bar chart representation for the sentiment classification for Angels and Demons Fiction using Vader and Text Blob

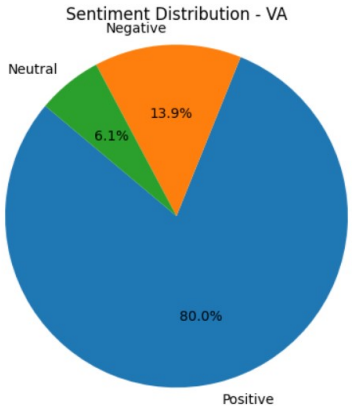


Figure 5.2: Pie chart representation for the sentiment classification for Angels and Demons Fiction

Figure 5.3: Word cloud for the positive words

Figure 5.4: Word cloud for the negative words

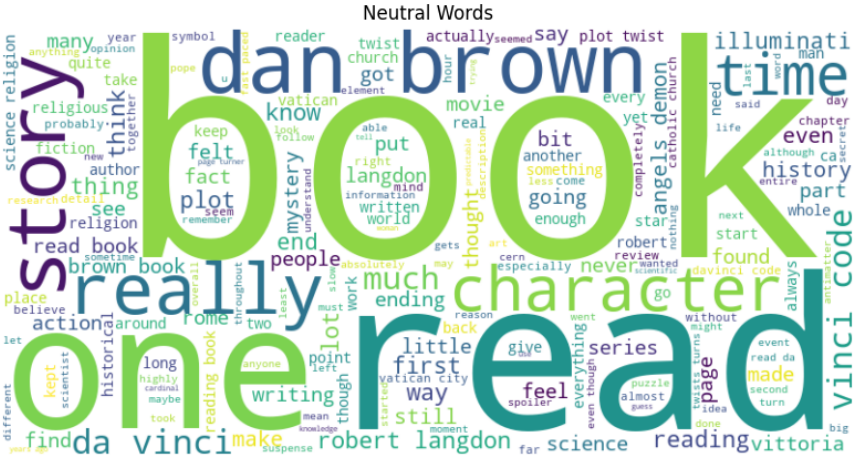


Figure 5.5: Word cloud for the neutral words

Table 5.1: Top 10 Words and Frequencies

Word	Frequency	Word	Frequency
Top 10 Positive Words		**Top 10 Negative Words**	
like	326 times	bad	52 times
good	207 times	hard	38 times
love	129 times	boring	31 times
well	127 times	wrong	26 times
great	119 times	stop	25 times
loved	113 times	lost	25 times
better	106 times	conspiracy	24 times
enjoyed	105 times	murder	24 times
novel	103 times	bomb	19 times
interesting	96 times	murdered	18 times
Top 10 Neutral Words			
book	1160 times		
read	615 times		
brown	454 times		
one	337 times		
dan	324 times		
langdon	309 times		
story	299 times		
really	289 times		
reading	242 times		
code	231 times		

5.1.2 Text Blob:

Using the TextBlob classifier we got the following result:

Figure 5.6 represents the percentage of the positive, negative, and neutral classifications using the TextBlob classification library.

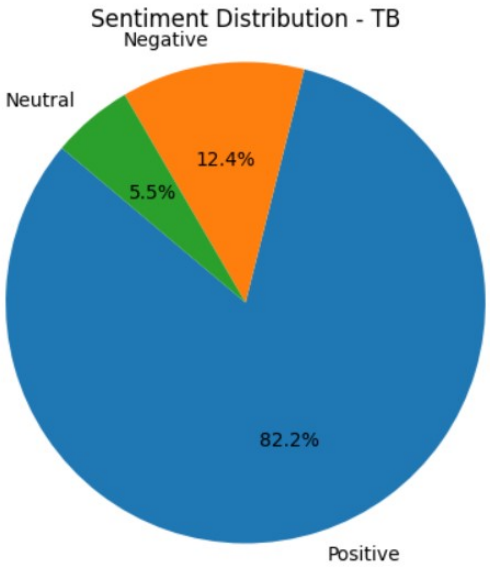


Figure 5.6: Sentiment classification using textblob - Pie Chart

If we look at the classification charts produced by the Text Blob classifier we can see that Text blob and Vader have different pre-defined words to predict the sentiment score which as a result has some minor predictions between the positive, neutral, and negative scores. The Text Blob positive sentiment score is 2.2% higher when compared with the Vader score. Whereas if we consider negative and neutral scores we can see that for negative Text blob score was 1.3% less than of Vader and for neutral Text Blob was 0.6% less than the Vader score.

Now we also created a word cloud to represent the top-10 positive, negative, and neutral words using TextBlob: If we look at the wordcloud for the positive words 5.7 generated by Text Blob we are able to see that for positive sentiments readers have used words such as really, great, loved, really, interesting, fast, excellent, much, enjoyed, entertaining, etc.

If we look at the wordcloud for the negative words 5.8 generated by Text Blob we are able to see that for negative sentiments readers have used words such as least, boring, bad, terrible, wrong, mean, unnecessary, unrealistic, etc.

We also filtered out the top word which readers used to represent the sentiment either in positive, neutral or negative way: For the Movie section, we have the following results:

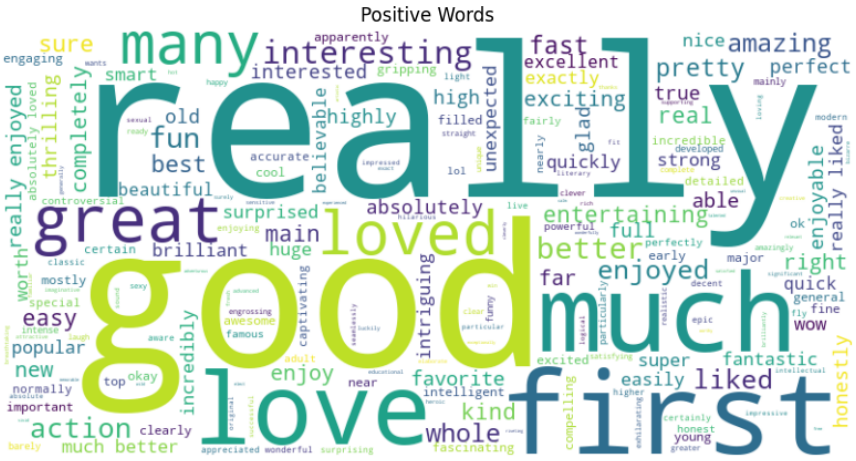


Figure 5.7: Positive words identified by Text Blob

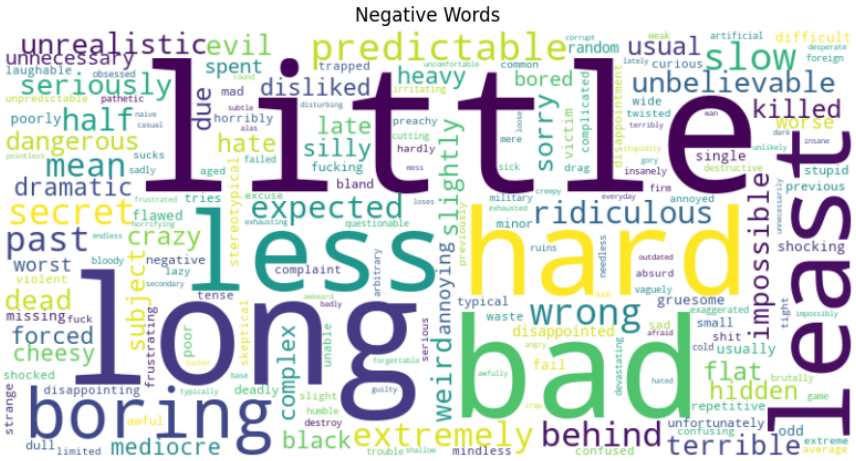


Figure 5.8: Negative words identified by Text Blob

Figure 5.9: Neutral words identified by Text Blob

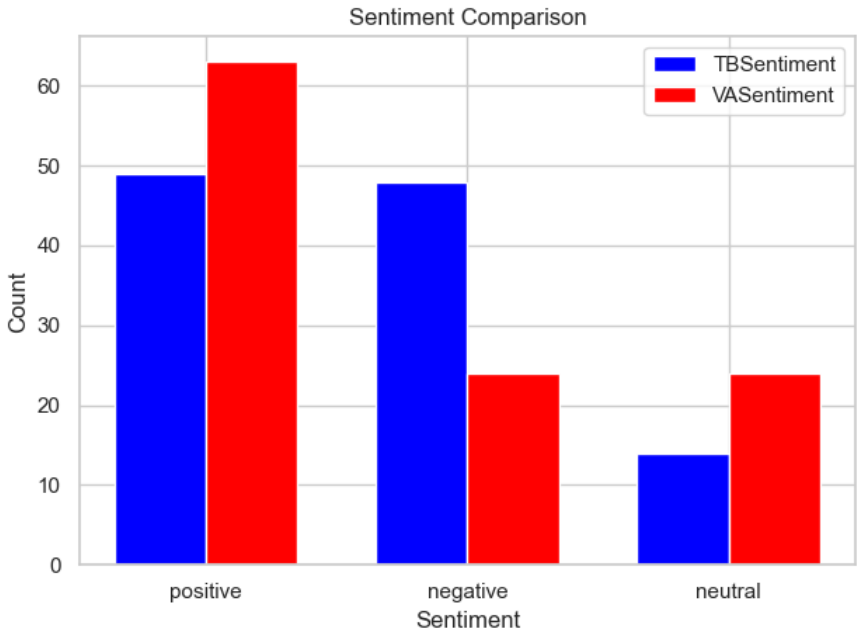


Figure 5.10: Bar chart representing sentiment classification results using Text Blob and Vader on Angels and Demons - Movie Reviews

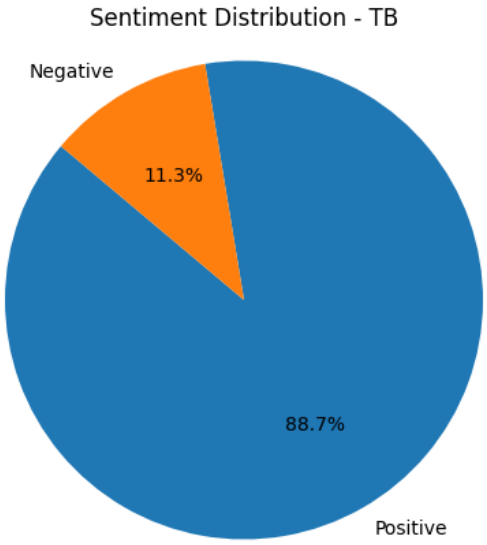


Figure 5.11: Pie chart representing sentiment classification results using Text Blob on Movie Reviews

Table 5.2: Top 10 Words and Frequencies

Word	Frequency	Word	Frequency
Top 10 Positive Words		**Top 10 Negative Words**	
really	289 times	little	87 times
good	207 times	long	63 times
much	169 times	bad	52 times
first	169 times	hard	38 times
love	129 times	less	32 times
great	119 times	least	31 times
loved	113 times	boring	31 times
better	106 times	predictable	27 times
enjoyed	105 times	slow	27 times
many	101 times	wrong	26 times
Top 10 Neutral Words			
book	1160 times		
read	615 times		
brown	454 times		
one	337 times		
like	326 times		
dan	324 times		
langdon	309 times		
story	299 times		
reading	242 times		
code	231 times		

Figure 5.10 represents the counts of the positive and negative sentiments calculated by Vader and TextBlob classifier.

Considering both the pie charts 5.11 and 5.12 we found that Text Blob was able to identify more positive words hence increasing the score of positive sentiments whereas when compared with the negative words recognition Text Blob lacked in identifying the negative words hence reducing the sentiment score against the Vader classifier.

Post classification we created a word cloud to analyze the words viewers have used to review the movie.

The figure 5.13 represents the positive words viewers/watchers have used to depict their review on IMDB platform. The most common terms used are love, good, great, enjoyed, interesting, love, amazing, sure etc. The details of these words as how many times it has

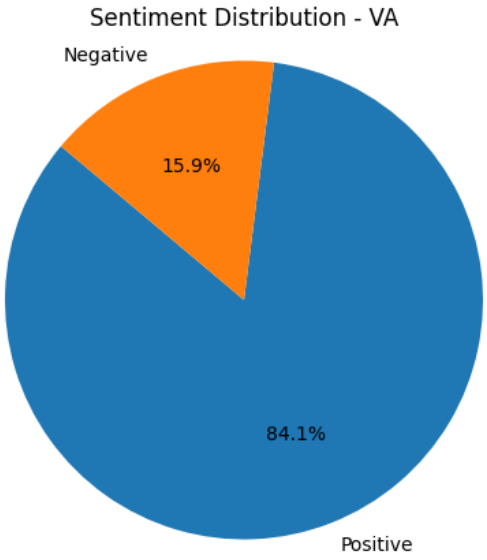


Figure 5.12: Pie chart representing sentiment classification results using Vader on Movie Reviews

been repeated in the movie data set is reflected in the table 5.3:

Table 5.3: Top 10 Positive Words

Word	**Frequency**
like	326 times
good	207 times
love	129 times
well	127 times
great	119 times
loved	113 times
better	106 times
enjoyed	105 times
novel	103 times
interesting	96 times

Figure 5.13: Positive word cloud generated using Vader

The figure 5.14 represents the negative words viewers/watchers have used to depict their feelings on IMDB platform. The most common terms used are least, bad long, fake, wrong, little, missing weak, bad, sorry subject, disappointed, seriously, etc. The details of these words as how many times it has been repeated in the movie data set are reflected in the table 5.4:

Table 5.4: Top 10 Negative Words

Word	**Frequency**
bad	52 times
hard	38 times
boring	31 times
wrong	26 times
stop	25 times
lost	25 times
conspiracy	24 times
murder	24 times
bomb	19 times
murdered	18 times

For the Text blob method, we got the following results: The figure **??** represents the positive words viewers/watchers have used to depict their review on IMDB platform. The most common terms used are love, good, great, enjoyed, interesting,love, amazing, sure etc. The details of these words as how many times it has been repeated in the movie data set is

Figure 5.14: Negative word cloud generated using Vader

reflected in the table 5.3:

Table 5.5: Top 10 Positive Words

Word	**Frequency**
much	253 times
good	173 times
first	147 times
better	126 times
great	112 times
new	96 times
action	89 times
many	86 times
interesting	71 times
really	70 times

Figure 5.15: Neutral word cloud generated using TextBlob

Figure 5.16 represents the negative words viewers/watchers have used to depict their feelings on IMDB platform. The most common terms used are least, bad long, fake, wrong, little, missing weak, bad, sorry subject, disappointed, seriously, etc. The details of these words as how many times it has been repeated in the movie data set are reflected in the table 5.4:

Table 5.6: Top 10 Negative Words

Word	**Frequency**
bad	75 times
less	72 times
long	47 times
secret	47 times
little	45 times
previous	45 times
least	42 times
behind	26 times
boring	25 times
slightly	24 times

Further, we also ran the hybrid model(Vader + TextBlob classifier) over our book and movie dataset and below are the results:

The hybrid model produced a total of 648 positive sentiment scores, 111 negative sentiment scores, and 26 neutral sentiment scores. Vader sentiment classification: VADER Sentiment

Figure 5.16: Negative word cloud generated using TextBlob

Counts: 628-Positive, 109-Negative, 48-Neutral. In the case of TextBlob: TB Sentiment Counts: 645-Positive, 97-Negative, 43-Neutral. The hybrid model score nearly matches the score of the textBlob classifier. Hence with this result, it can be clear that the TextBlob calculates the sentiments of the text more accurately when compared with the TextBlob.

5.2 CASE-2 - Anna Karenina

In the case of Anna Karenina the fiction/Book was really appreciated by the readers. Book was rated as 4 approximately overall with strong emotion, character development, diverse characters etc. For the fictional part we have 500 reviews available over thestorygraph portal and the stats for it is as below: Average rating for the 500 reviews is: 4.08 Post running Vader and Text Blob analyser the result we got are similar for both algorithm. The figure 5.17 represents the polarity categorized using the TextBlob and Vader analyzer. Now if we check the results both of the algorithm has the similar performance.

Top positive, negative and neutral words used to describe the fiction are as below:

Now since the book was appreciated by the readers we can see the positive word repetition is high when compared with the Negative words which justifies the appreciation of the book. Some of the positive negative and neutral reviews are represented below:

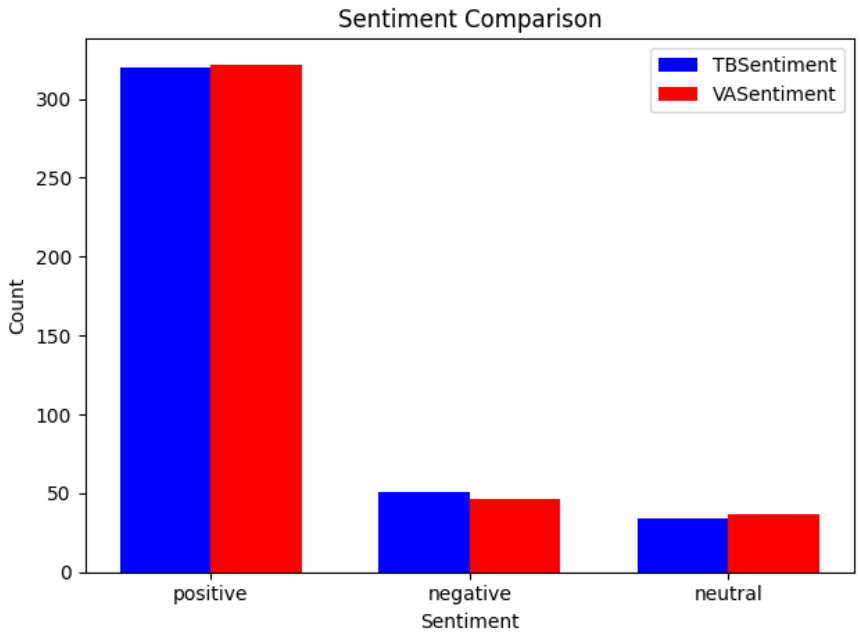


Figure 5.17: Bar Chart representing the polarity classification using Vader and TextBlob - Anna Karenina Fiction

For the Anna Karenina Movie, we have the following stats:

Table 5.7: Top 10 Words and Frequencies - Anna Karenina - Fiction

Word	Frequency	Word	Frequency
Top 10 Positive Words		**Top 10 Negative Words**	
like	230 times	hard	39 times
love	195 times	bad	34 times
novel	172 times	death	28 times
good	95 times	war	28 times
well	76 times	tragic	26 times
enjoyed	68 times	boring	24 times
great	66 times	difficult	18 times
loved	65 times	tragedy	17 times
interesting	61 times	hate	16 times
liked	51 times	hated	16 times
Top 10 Neutral Words			
anna	555 times		
book	555 times		
tolstoy	332 times		
levin	320 times		
read	292 times		
characters	260 times		
story	222 times		
one	212 times		
time	192 times		
really	186 times		

If we concentrate on the word cloud representing the positive, negative, and neutral words we can see that the viewers say that the actors were great, story was beautiful, Brilliant, creative effort etc. Fig 5.19 represents the positive words cloud of the reviews.

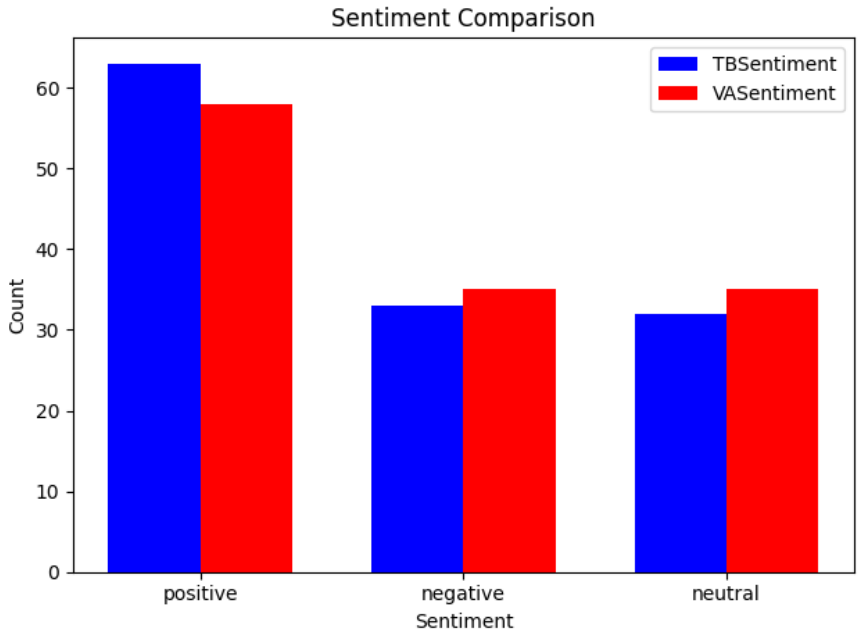


Figure 5.18: Bar Chart representing the polarity classification using Vader and TextBlob - Anna Karenina Movie

Figure 5.19: Word Cloud representing positive words viewers have used to review- Anna Karenina Movie

Table 5.8: Top Positive reviews for Anna Karenina - Fiction

Review Number	Review Text
1	It's almost a 3.5 because honestly I really di...
2	Great book, nearly perfect, but 5 stars must b...
3	I cannot put half stars so imagine a 4.5.
4	I enjoyed some parts, and it says something th...
5	I WILL finish this. I wanted to munch this so...

Table 5.9: Top Negative Reviews for Anna Karenina - Fiction

Review Number	Review Text
1	This is a story of politics, farming, indifel...
2	Longest story... Ever... Following the lives...
3	Oh Lawd, this was the longest, and at times mo...
4	Holy shit. This is one of my favorite books no...
5	This is the worst book that has ever been...

In figure 5.20 we can see that most of the negative words used are Weak, Experiment gone wrong, Serious miscasting, irritation format etc.

The top positive, negative, and neutral reviews for the Anna Karenina movie is represented in the table below:

5.3 Case 3- Eragon

Post researching the fiction titled Eragon we found that the fiction was originally published in 2002 which is written by author Christopher Paolini. The fiction has an average rating of 3.9 stars out of 5 on Google Reads, 5/5 on Scholastic Shop, and 4.5 on Amazon reviews. Whereas when we compare the reviews with the movie on IMDB it is seen that the movie didn't do well on the screen with an average rating of 5.1 out of 10.

Table 5.10: Top Neutral Reviews for Anna Karenina - Fiction

Review Number	Review Text
1	So many thoughts. Will write them down in a bit.
2	Started reading it only because of Aaron Taylor...
3	Will read later when I have more time.
4	A classic for a reason.
5	I wanted to give it three stars with the drag-throug...

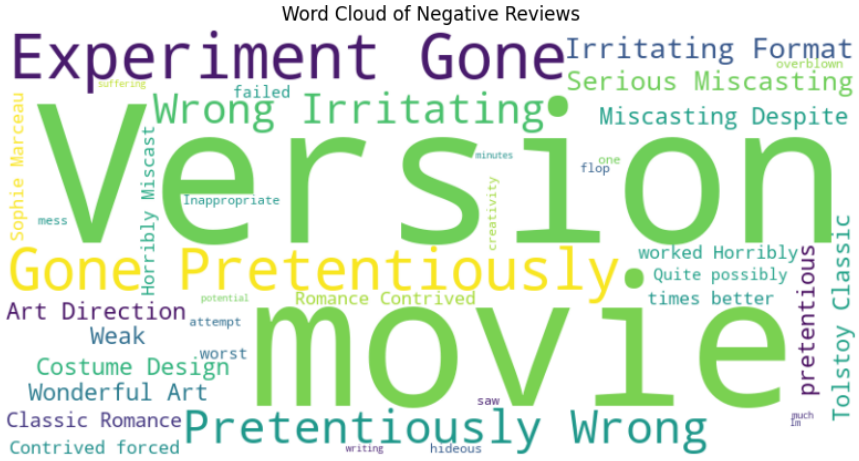


Figure 5.20: Word Cloud representing negative words viewers have used to review- Anna Karenina Movie

Looking at the graph it is quite clear that the sentiment related to the fiction is quite positive when compared with the negative and neutral. Classification with the Vader and TextBlob performs approximately with the same accuracy.

Top positive words found by the classifier is: Top 10 Positive Words: like: 418 times good: 281 times love: 216 times great: 149 times loved: 148 times well: 135 times enjoyed: 126 times novel: 106 times better: 97 times liked: 96 times. Whereas the top 10 negative words are: Top 10 Negative Words: bad: 81 times wars: 80 times hard: 58 times boring: 50 times battle: 45 times hate: 33 times evil: 27 times wrong: 22 times poor: 21 times problem: 20 times.

When we compare the movie reviews with the fiction we found most of the reviews were mixed and not positive. The Bar chart 5.22 we can see that most of the reviews were mixed. The number of positive reviews is comparatively the same as the number of negative reviews.

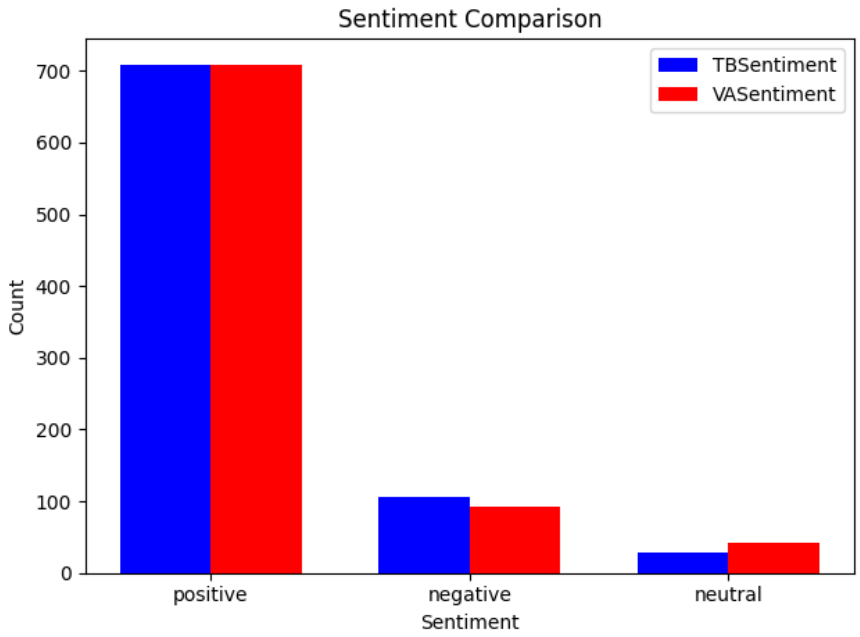


Figure 5.21: Bar Chart representing the review classification by Vader and Text blob for the fiction - Eragon

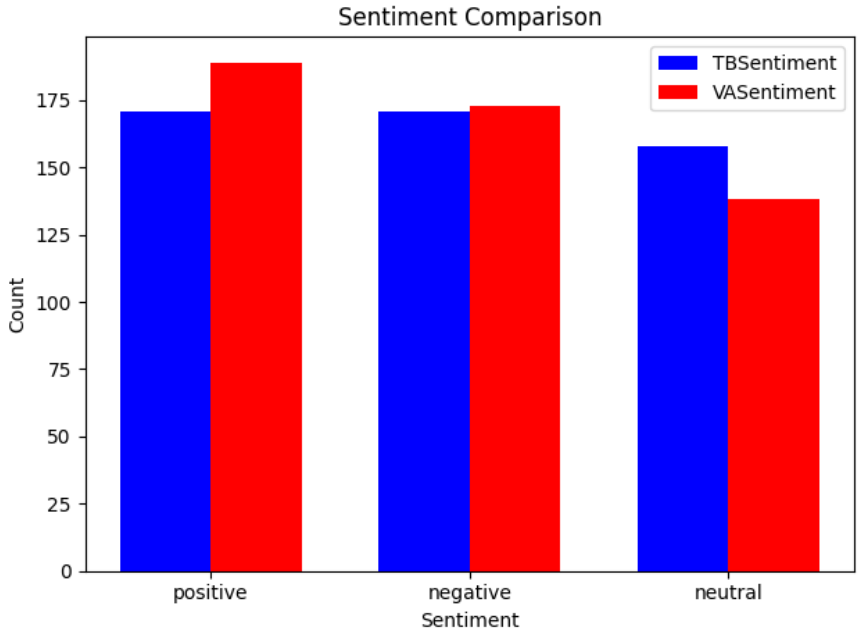


Figure 5.22: Bar Chart representing the review classification by Vader and Text blob for the Movie - Eragon

Table 5.11: Top 5 Positive reviews for Anna Karenina - Movie

Review Number	Review Text
1	A beautiful bore
2	A Sumptuous Feast for the Eye and Ear and Mind...
3	Beautiful, Enchanting and Bold
4	A Very Creative Effort
5	Brilliant adaptation

Table 5.12: Top 5 Negative Reviews for Anna Karenina - Movie

Review Number	Review Text
1	An Experiment Gone Pretentiously Wrong
2	Irritating Format and Serious Miscasting
3	Despite the Wonderful Art Direction and Costum...
4	Contrived, forced and pretentious, this movie ...
5	Horribly Miscast - The Sophie Marceau version ...

Top positive words found by the classifier is: Top 10 Positive Words: good: 18 times great: 10 times like: 8 times better: 8 times special: 7 times laughable: 5 times enjoyed: 5 times adventure: 5 times entertaining: 3 times loved: 3 times

Top 10 Negative Words: bad: 23 times worst: 11 times horrible: 7 times sad: 5 times disappointing: 5 times hated: 5 times murder: 5 times poor: 5 times battles: 5 times terrible: 4 times Also if we look at the Kernel density plot 5.23 of the Eragon movie we can clearly see that most of the sentiment of the reviews are negative and neutral rather than positive.

Table 5.13: Top 5 Positive reviews for Eragon - Fiction

Review Number	Review Text
1	Mystery, adventure, fantasy all rolled into a ...
2	Rereading this with Murtaugh coming out. This ...
3	pretty bad but has a special place in my heart...
4	Loved this book, only wish I read it as a kid ...
5	i didn't think rereading a book could be just ...

Table 5.14: Top 5 Negative reviews for Eragon - Fiction

Review Number	Review Text
1	Frankly epic choice for my commuting and falling asleep audiobook
2	I had a hard time getting into this book, I never read any of the other ones.
3	I couldnt even finish it.So boring.
4	This series is hard to sink my teeth into, although I keep reading out of pure curiosity.
5	could not get into it

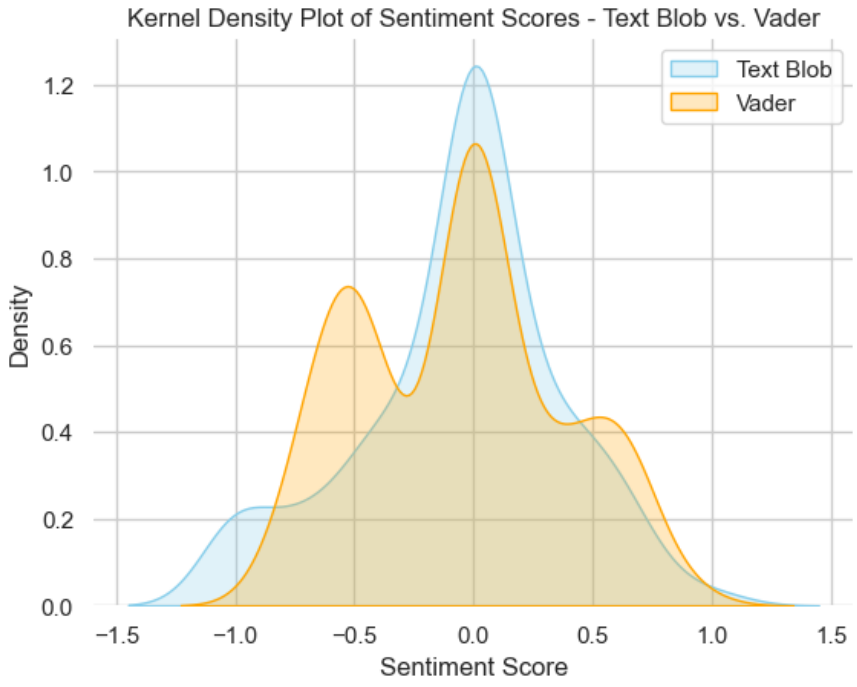


Figure 5.23: Kernel Density plot for the Eragon Movie

CHAPTER 6

DISCUSSIONS

This section explains the reason behind conducting this research and analysis. Whenever there is an adaptation from fiction into a Movie most off the filmmakers try to redraw characters and the plot of the movie trying some experiments. Which result may or may not be successful for the movie. In our research, the goal was to emphasize those components that viewers wanted in the movie that were adapted from fiction. Now performing both of the classification techniques that is TextBlob and Vader it was quite clear that both of the classifiers can perform well in identifying the sentiments behind the reviews provided by the readers and viewers.

There are different aspects one can look at while concluding the results for example not all directors are liked by the viewers which can be one reason for the movie not doing well as in the case 3 - Eragon, movie plot not follow the story as of the Fiction, lack of acting skills etc. Since there are not much reviews available on both of the portals like Storygraph there were more than 17k reviews whereas for the movie on the IMDB portal, there were around only 2k reviews. Even in these reviews most of them had only ratings and not the context, since there is a huge difference between these scraped reviews, there is a possibility that might prove to be one of the major reasons behind the lacking co-relation between the fiction sentiments and the Movie Sentiments.

Case 1 - Angels and Demons: With the analysis such as making a word cloud and bar chart along it can be concluded that the filmmakers had stick to the plot and the development of the characters according to the fiction which resulted in making the film successful and

same goes with Case 2 - Anna Karenina whereas when we check the Case 3 - Angels and Demons the filmmakers did not stick to the characters and the plot of the fiction which had an adverse effect like viewers did not like the movie.

CHAPTER 7

CONCLUSION

The final goal of this dissertation was to identify the requirements of a movie which were adapted from fiction. The classification of the reviews was carried out using the Lexicon-based approaches which are Vader and TextBlob and both of these classifiers have performed with similar accuracy producing approximately similar results as well.

If we look at the wordcloud chart, the top positive, negative, and neutral words along with the positive, negative, and neutral reviews it is quite clear that most of the viewers prefer to stick to the story of the fiction rather than having any modification into the plot. The second important part for a movie to be successful is to avoid adding any extra minutes to the movie like the introduction of a character or unnecessary background scenes or adding any new characters.

CHAPTER 8

FUTURE WORK

For future prospects, we are looking to build a user interface so that the data can be scraped directly using the URL and the number of reviews required for the analysis. Need of the analysis portal? While performing our research we found that there were multiple attempts to remake of same fiction into movies and in most such cases the remake has failed drastically. In the future, we would like to improvise the sentiment analysis task so that filmmakers can decrease the failure rate before starting the movie. This portal will not only allow the filmmakers to film the movie based on the plot but will also allow them to highlight all the best and love able characters of the fiction resulting in better ratings and reviews along with the collection.

REFERENCES

[1] D. Talbot. (2022) Number of books published per year. Accessed: Feb 2022. [Online]. Available: https://wordsrated.com/number-of-books-published-per-year-2021/#:~:text=The%20total%20number%20of%20new,are%20considered%20to%20be%20produced.

[2] A. Almjawel, S. Bayoumi, D. Alshehri, S. Alzahrani, and M. Alotaibi, "Sentiment analysis and visualization of amazon books' reviews," in *2019 2nd International Conference on Computer Applications Information Security (ICCAIS)*, May 2019, pp. 1–6.

[3] J. D. Bodapati, N. Veeranjaneyulu, and S. N. Shareef, "Sentiment analysis from movie reviews using lstms." *Ingénierie des Systèmes d Inf.*, vol. 24, no. 1, pp. 125–129, 2019.

[4] G. Mesnil, T. Mikolov, M. Ranzato, and Y. Bengio, "Ensemble of generative and discriminative techniques for sentiment analysis of movie reviews," *arXiv preprint arXiv:1412.5335*, 2014.

[5] M. Yasen and S. Tedmori, "Movies reviews sentiment analysis and classification," in *2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT)*, 2019, pp. 860–865.

[6] T. M. Untawale and G. Choudhari, "Implementation of sentiment classification of movie reviews by supervised machine learning approaches," in *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, 2019, pp. 1197–1200.

[7] R. Wankhede and A. Thakare, "Design approach for accuracy in movies reviews using sentiment analysis," in *2017 International conference of Electronics, Communication and Aerospace Technology (ICECA)*, vol. 1, 2017, pp. 6–11.

[8] Y. Mejova, "Sentiment analysis: An overview," *University of Iowa, Computer Science Department*, 2009.

[9] V. U. Ramya and K. T. Rao, "Sentiment analysis of movie review using machine learning techniques," *International Journal of Engineering & Technology*, vol. 7, no. 2.7, pp. 676–681, 2018.

[10] S. Elbagir and J. Yang, "Twitter sentiment analysis using natural language toolkit and vader sentiment," in *Proceedings of the international multiconference of engineers and computer scientists*, vol. 122, 2019, p. 16.

[11] C. Hutto and E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text," in *Proceedings of the international AAAI conference on web and social media*, vol. 8, no. 1, 2014, pp. 216–225.

[12] P. Bhoir and S. Kolte, "Sentiment analysis of movie reviews using lexicon approach," in *2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, 2015, pp. 1–6.

[13] J. P. Gujjar and H. P. Kumar, "Sentiment analysis: Textblob for decision making," *Int. J. Sci. Res. Eng. Trends*, vol. 7, no. 2, pp. 1097–1099, 2021.

[14] D. H. Wolpert, "The Lack of A Priori Distinctions Between Learning Algorithms," *Neural Computation*, vol. 8, no. 7, pp. 1341–1390, 10 1996. [Online]. Available: https://doi.org/10.1162/neco.1996.8.7.1341

APPENDIX A

CODES

Scraping process:

```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.firefox.options import Options
from selenium.webdriver.firefox.service import Service
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as
   EC
import csv

firefox_options = Options()
firefox_options.add_argument('-headless')
firefox_binary_path = r"C:\Program␣Files\Mozilla␣Firefox\
   firefox.exe"
firefox_options.binary_location = firefox_binary_path
service = Service(executable_path=r"C:\Users\Ajay\Downloads\
   geckodriver-v0.33.0-win32\geckodriver.exe")
```

```
driver = webdriver.Firefox(service=service, options=
    firefox_options)

ratings_to_collect = 1000

data = []
url = "https://app.thestorygraph.com/book_reviews/d593cc1a
    -9322-4a94-8706-30a819885f6f"
try:
    driver.get(url)
    collected_ratings = 0  # Reset collected_ratings for each
        URL
    collected_reviews = set()  # Keep track of collected
        reviews
    while collected_ratings < ratings_to_collect:
        try:
            ratings = WebDriverWait(driver, 20).until(
                EC.presence_of_all_elements_located((By.XPATH,
                    '//p[@class="mb-2"]'))
            )
            reviews = driver.find_elements(By.XPATH, '//div[
                contains(@class, "review-explanation")]')
            for rating, review in zip(ratings, reviews):
                review_text = review.text
                if review_text not in collected_reviews:
                    url_rating_review = (url, rating.text,
                        review_text)
```

```
                        data.append(url_rating_review)
                        collected_reviews.add(review_text)
                        collected_ratings += 1
                        print(rating.text)
                        print(review_text)
                        if collected_ratings >= ratings_to_collect
                           or len(reviews) < len(ratings):
                            break
                driver.execute_script("window.scrollTo(0, document.
                   body.scrollHeight);")
            except Exception as e:
                print(f"Error while scraping {url}: {e}")
                break

except Exception as e:
    print(f"Error: {e}")

finally:
    driver.quit()

Eragon_Book = pd.DataFrame(data, columns=["URL", "Rating", "
   Review"])
```

Listing A.1: Extraction of data

Classification Process: TextBlob:

```
import pandas as pd
from textblob import TextBlob
```

```
from sklearn.metrics import precision_score, f1_score
text_samples = AngelsAndDemons_Book['bReview']
ground_truth_labels = AngelsAndDemons_Book['bReviewCategory']
textblob_predictions = []

for text_sample in text_samples:
    analysis = TextBlob(text_sample)
    if analysis.sentiment.polarity > 0:
        textblob_predictions.append("positive")
    elif analysis.sentiment.polarity < 0:
        textblob_predictions.append("negative")
    else:
        textblob_predictions.append("neutral")
precision = precision_score(ground_truth_labels,
   textblob_predictions, average='weighted', labels=['positive'
   , 'negative', 'neutral'])
f1 = f1_score(ground_truth_labels, textblob_predictions,
   average='weighted', labels=['positive', 'negative', 'neutral
   '])

print(f"Precision: {precision:.2f}")
print(f"F1 Score: {f1:.2f}")
```

Listing A.2: Classification of data -Text Blob

Classification process using Vader:

```
import pandas as pd
from nltk.sentiment.vader import SentimentIntensityAnalyzer
```

```
from sklearn.metrics import precision_score, f1_score
sia = SentimentIntensityAnalyzer()
text_samples = AngelsAndDemons_Book['bReview']
ground_truth_labels = AngelsAndDemons_Book['bReviewCategory']
vader_predictions = []
for text_sample in text_samples:
    sentiment_scores = sia.polarity_scores(text_sample)
    if sentiment_scores['compound'] >= 0.05:
        vader_predictions.append("positive")
    elif sentiment_scores['compound'] <= -0.05:
        vader_predictions.append("negative")
    else:
        vader_predictions.append("neutral")
precision = precision_score(ground_truth_labels,
   vader_predictions, average='weighted', labels=['positive', '
   negative', 'neutral'])
f1 = f1_score(ground_truth_labels, vader_predictions, average='
   weighted', labels=['positive', 'negative', 'neutral'])
print(f"Precision: {precision:.2f}")
print(f"F1 Score: {f1:.2f}")
```

Listing A.3: Classification of data -Text Blob

Bar Chart:

```
#creating a bar plot based on the categories
plt.figure(figsize=(8, 5))
sns.countplot(data=Eragon_Movie, x='TBSentiment')
plt.title('Sentiment Distribution - VA')
```

```
plt.xlabel('Sentiment Category')
plt.ylabel('Count')
plt.show()
```

Listing A.4: Bar Chart

Word Cloud:

```
def create_word_cloud(words, title):
    wordcloud = WordCloud(width=800, height=400,
       background_color='white').generate(" ".join(words))
    plt.figure(figsize=(10, 5))
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis("off")
    plt.title(title)
    plt.show()

create_word_cloud(positive_words, "Positive Words")
create_word_cloud(negative_words, "Negative Words")
create_word_cloud(neutral_words, "Neutral Words")
```

Listing A.5: WordCloud

Kernel Density plot:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from textblob import TextBlob
from vaderSentiment.vaderSentiment import
   SentimentIntensityAnalyzer
```

```
6
7
8  textblob_sentiment_scores = []
9  vader_sentiment_scores = []
10
11 analyzer = SentimentIntensityAnalyzer()
12 for index, row in Eragon_Movie.iterrows():
13     review_text = row['PreprocessedReviews']
14     textblob_sentiment = TextBlob(review_text).sentiment.
          polarity
15     textblob_sentiment_scores.append(textblob_sentiment)
16     vader_sentiment = analyzer.polarity_scores(review_text)['
          compound']
17     vader_sentiment_scores.append(vader_sentiment)
18 sns.set(style="whitegrid")
19 sns.kdeplot(textblob_sentiment_scores, shade=True, color='
      skyblue', label='Text␣Blob')
20 sns.kdeplot(vader_sentiment_scores, shade=True, color='orange',
       label='Vader')
21 sns.despine(left=True, bottom=True)
22 plt.xlabel('Sentiment␣Score')
23 plt.ylabel('Density')
24 plt.title('Kernel␣Density␣Plot␣of␣Sentiment␣Scores␣-␣Text␣Blob␣
      vs.␣Vader')
25 plt.legend()
26
27 plt.show()
```

Listing A.6: Kernel Density Plot