Analysis and Prediction of Movie Review Ratings on Rotten Tomatoes

Megan Chiang, Michelle Kim, Jasmine Wong
CSE 163 AD/Group 055 | Spring 2023

Summary

1. **What are the 20 best and worst movies based on critics' ratings on Rotten Tomatoes? Are there certain words that are more commonly associated with positive or negative reviews? Are there any patterns in language used in positive and negative reviews?**

   Based on reviews from 2015-2020 by top critics, the 20 best movies are:

   1. *Roma*
   2. *Parasite*
   3. *Carol*
   4. *45 Years*
   5. *Moonlight*
   6. *Portrait of a Lady on Fire*
   7. *Manchester by the Sea*
   8. *Spotlight*
   9. *The Irishman*
   10. *Son of Saul*
   11. *Leave No Trace*
   12. *Eighth Grade*
   13. *Shoplifters*
   14. *Lady Bird*
   15. *Anomalisa*
   16. *Call Me by Your Name*
   17. *Marriage Story*
   18. *La La Land*
   19. *The Florida Project*
   20. *Three Billboards Outside Ebbing, Missouri*

   The highest-rated movie (*Roma*) had a rating of 4.821/5, and the 20th best movie (*Three Billboards Outside Ebbing, Missouri*) had a rating of 4.611/5.

   The 20 worst movies are:

   1. *Transformers: The Last Night*
   2. *Rambo: Last Blood*
   3. *Hellboy*

4. *Dolittle*
5. *Hot Pursuit*
6. *The Happytime Murders*
7. *Pixels*
8. *The Mummy*
9. *London Has Fallen*
10. *Venom*
11. *Fantastic Four*
12. *The Dark Tower*
13. *Rock the Kasbah*
14. *Self/Less*
15. *Cats*
16. *The Divergent Series: Allegiant*
17. *Alice Through the Looking Glass*
18. *The Kitchen*
19. *Baywatch*
20. *Serenity*

The lowest-rated movie (*Transformers: The Last Night*) had a rating of 1.815/5, and the 20th worst movie (*Serenity*) had a rating of 2.348/5.

While the most frequent words for positive reviews of top 20 movies are “story”, “best”, “performance”, the most frequent words for negative reviews of bottom 20 movies are “less”, “never”, and “character”. There are few words that appear in both of the word clouds such as “movie”, “film” and “make”.

2. **How does average word count and average review score per movie relate?**
   We found that there is no significant relationship between word count and review score. This is because there is almost a normal distribution along both axes.

3. **How accurately can we predict the review rating based on the review content using a logistic regression model?**
   Our accuracy score was around 0.48177802319160684, indicating that our model is slightly better than random chance (where chance would be 1 divided by the number of categories (5), or 0.2). Additionally, our **F-1 scores**, or the harmonic mean of precision and recall, for this multi-class classification are (1): 0.32895, (2): 0.61322, (3): 0.40695, (4): 0.32143, (5): 0.01307.

Motivation

Our motivation for this project began with an interest in entertainment and linguistics. Entertainment is a vital part of popular culture, so we decided to focus on movies. Movies can captivate audiences and influence societal discussions. Our group will use data from Rotten Tomatoes, a popular online resource for reviewing and rating movies, as well as viewing other critics' reviews.

We will explore the highest and lowest rated movies, while examining the positive and negative words associated with them. Our project will determine if there is a correlation between word counts of review texts and review scores, and explore patterns between entertainment and the linguistics of critics' reviews. To predict review scores, we will use machine learning and data science techniques.

The overarching goal of our project is to investigate the complex relationship between entertainment and linguistics. The application of our research can aid the entertainment industry and popular culture. Our findings can assist filmmakers, producers, content creators, and many others in understanding the factors that contribute to a movie's critical success or failure. This will enable them to make more informed decisions regarding storytelling, production, and marketing strategies. Ultimately, our investigation will provide a broader understanding of societal impact and influence of movies, which can shape the role of media and public opinions.

Data Setting

We used a Rotten Tomatoes movies and critic review dataset. The dataset consists of two files: rotten_tomatoes_critic_reviews.csv and rotten_tomatoes_movies.csv.

In the critics dataset, each row represents a critic review published on Rotten Tomatoes with its critic name, review publication, review type, review score, review date and content. In the movies dataset, each row represents a movie available on Rotten Tomatoes with its movie title, movie info, critics consensus, content rating, genres, directors, authors and original release date. Given its size and complexity of the dataset, we will focus exclusively on reviews from top critics written in 2015-2020.

Methodology

- **Data cleaning**
    - Filter critics.csv by selecting reviews written between 2015-01-01 and 2020-12-31 by top critics
    - Merge filtered critics.csv with movies.csv by the rotten_tomatoes_link column
    - Create a new dataset by selecting critic_name, review_score, review_content and review_date, movie_title and genre.
    - Remove rows where the rating is null, a letter grade (e.g., "A+), or a score without a denominator

    - Convert review scores (e.g., a string written as “4/5") to float values, then create a new column of review labels (1 to 5) that correspond to these float values.

- **What are the 20 best and worst movies based on critics' ratings on Rotten Tomatoes?**
    - To have a large enough sample size, consider only movies with at least 25 reviews. Filter dataset to include only these movies.
    - Calculate the mean review score for each of these movies using groupby().
    - Find the top 20 highest- and lowest-rated movies (based on average review score) using nlargest() and nsmallest(), respectively.
    - Create two bar charts using Seaborn to display this data.

- **Are there certain words that are more commonly associated with positive or negative reviews?**
    - Create two subsets of the dataset based on a review score, one for positive reviews (score greater than 3) and the other for negative reviews (score less than 3)
    - Iterates over each review and tokenizes the review using the word_tokenize function from the NLTK library
    - Cleans the words by converting them to lowercase, excluding stopwords and punctuations.
    - Uses the FreqDist function to calculate the frequency distribution of words.
    - Find the top 100 most common words in reviews
    - Extract the words and their frequencies
    - Convert the word frequency to a string
    - Creates a word cloud which visually shows the most frequent words in each subset based on their frequency.

- **How does average word count and average review score per movie relate?**
    - Calculate the average review scores per movie by grouping the `movie_title` by `review_score` and calculating the mean.
    - Calculate the average word counts for each movie review by grouping the `movie_reviews` by the `movie_title` column and apply a lambda function to the `review_content` column. The lambda function will calculate an average word count per movie.
    - Create a new data frame which stores the average review score and average review word count
    - Then plot a scatter plot using Seaborn and the new data frame, that shows the relationship between average word count and average review score per movie.

- **How accurately can we predict the review rating based on the review content using a logistic regression model?**
    - Extract the `review_content` and `score_category` columns from the `movie_reviews` data frame.
    - Split the data into training and testing sets with a test size and random state.
    - Create an instance of a CountVectorizer, this is used to create a "bag of words" vectorization so that the model can read the input.
    - Create an instance of a TfidfTransformer, which creates a Term Frequency-Inverse Document frequency.
    - Fit and transform the training data and the testing data using the instances of the CountVectorizer and the TfidfTransformer
    - Create an instance of the logistic regression model, set the multi class parameter to multinomial so that the model can classify more than two categories, with the solver 'lbfgs'.
    - Train the logistic regression model using the fit method and passing in the transformed training data and the labels.
    - Predict the score categories for the testing data
    - Calculate the accuracy and F-1 scores for multi-class classification
    - Create a Seaborn heatmap, with the corresponding labels and title.
    - Return the F-1 scores and accuracy score in string format

Results

1. **Which movies have the highest and lowest review scores on Rotten Tomatoes? Are there certain words that are more commonly associated with positive or negative reviews? Are there any patterns in language used in positive and negative reviews?**

a. **Which movies have the highest and lowest review scores on Rotten Tomatoes?**
Here are the top 20 highest- and lowest-rated movies on Rotten Tomatoes based on reviews by top critics from 2015-2020:

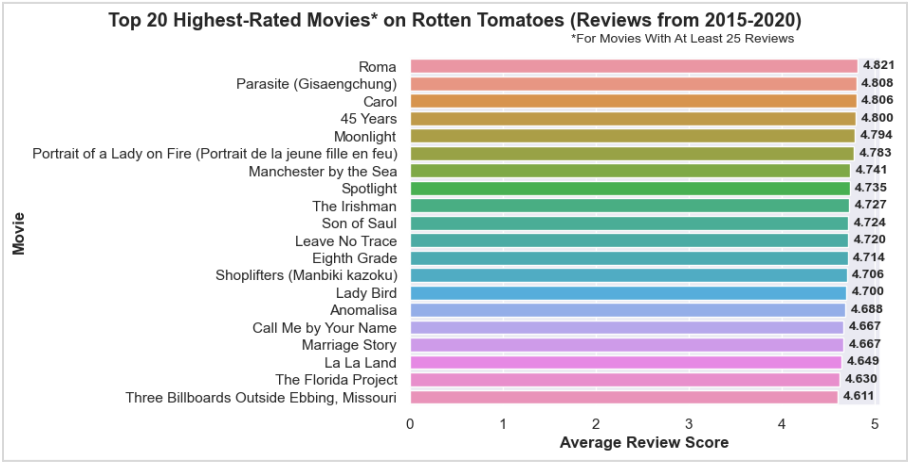

Fig. 1. Bar chart of the 20 highest-rated movies

We were not surprised by these results, as these movies are widely-acclaimed and highly-rated on other critic review sites like Metacritic. For instance, *Roma* (the highest-rated movie) has a Metascore of 96/100 on Metacritic, which is almost equivalent to 4.821/5. It was interesting to see that all 20 of these movies had average ratings greater than 4.5/5, ranging from 4.611 to 4.821.

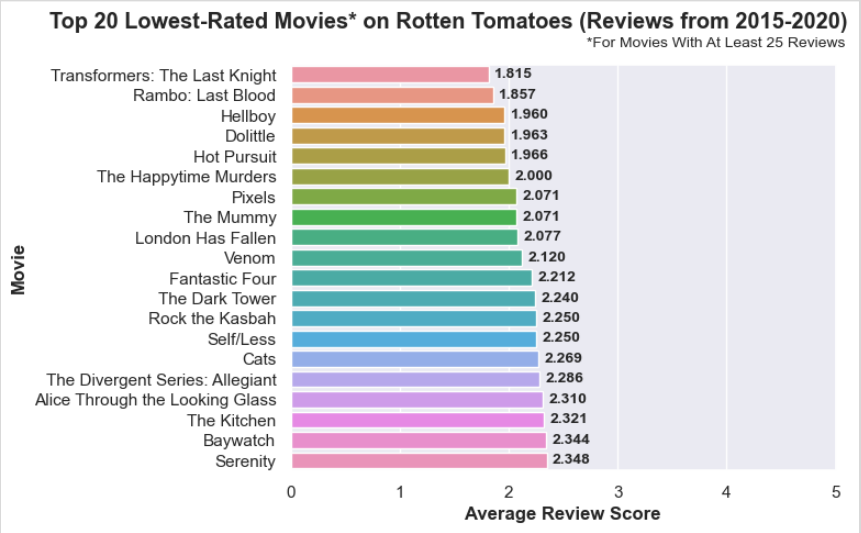

Fig. 2. Bar chart of the 20 lowest-rated movies

The 20 lowest-rated movies had average ratings ranging from 1.815 to 2.348. We were not surprised about some of these results because we had anecdotal evidence that they were poorly-received. When *Cats* was released, there were many articles and social media posts that shared people's distaste of the movie. For other movies, we did not have much background knowledge about them (compared to our knowledge on the top 20 highest-rated movies), but after comparing their ratings with Metacritic, we found that the average ratings were very similar. For instance, *Transformers: The Last Night* (the lowest-rated movie) has a Metascore of 27/100 on Metacritic, which is very similar to an average score of 1.815/5 on Rotten Tomatoes.

b. Are there certain words that are more commonly associated with positive or negative reviews? Are there any patterns in language used in positive and negative reviews?

Fig. 3. Positive word cloud of top 20 movies

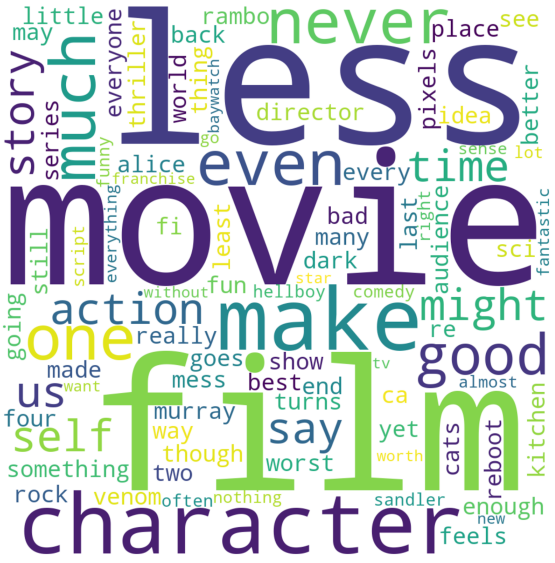

Fig. 4. Negative word cloud of bottom 20 movies

The result of the positive and negative word clouds reveal interesting patterns of the most frequently used words in top and bottom 20 movie reviews. In the positive word cloud (Fig. 3), the prominent words are “performance”, “year”, “feel”, “story” and “drama”. As expected, there were positive words such as “best”, “masterpiece” and “beautiful” in the positive word cloud.

In the negative word cloud (Fig. 4), the most frequently used words are “character”, “less”, “never”, “much”. The term “never” could indicate dissatisfaction with certain elements lacking in the movie.

One takeaway from these word clouds is that they lack granularity in differentiating between positive and negative sentiments within the same word, which can lead to inaccurate representations of sentiments in the reviews. For example, terms such as “movie” and “film” appear in both word clouds. Besides these terms, there are also other terms that indicate neutrality and fail to represent the tone of the words used frequently in each review. Therefore, it is essential to explore the underlying context of the reviews and be aware of these limitations in capturing full nuances of sentiment.

2. **How does average word count and average review score per movie relate?**

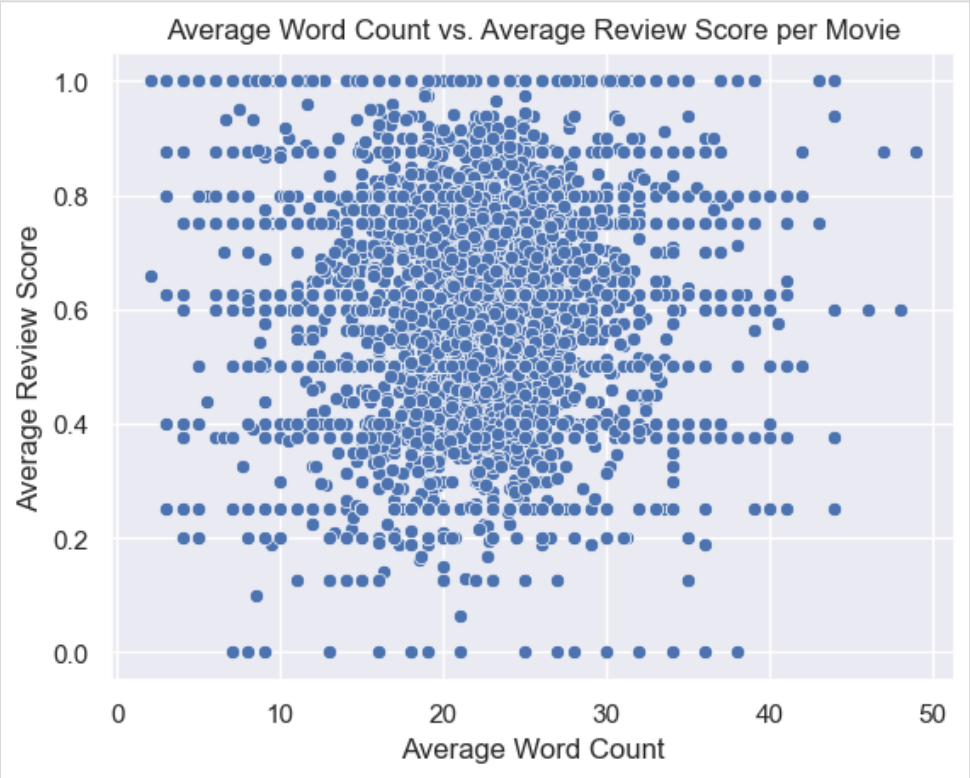

Fig. 5. Scatter plot of the average word count vs. average review score per movie

Fig. 6. Distributions of average word count and average review score

When finding a relationship between average word count and average review score, we were surprised to find that there was no significant relationship between these two

variables. While examining further we realized that there is a normal distribution for average word count, while the average review score has a left skew. Something to note that we found interesting is that the scatter plot displays most reviews per movie have an average word count between 15-25 words.

While further investigating the data set we found that most of the review content was short blurbs or sentences, and viewing the Rotten Tomatoes site we realized that this is because Rotten Tomatoes has a curated review display by their team. The about page states, “Rotten Tomatoes has assembled a team of curators whose job it is to read thousands of movie and TV reviews weekly. The team collects movie and TV reviews from Tomatometer-approved critics and publications every day, generating Tomatometer scores. Our curators carefully read these reviews, noting if the reviews are Fresh or Rotten, and **choose a representative pull-quote.**” The pull-quote is what is represented in the `review_content` column in our data set.

The consequence of this pull-quote data is that it is a limited representation because they do not fully represent the complete review content. This means that the word count derived from these pull-quotes might not capture the full depth and breadth of the reviewers’ opinions.

3. **How accurately can we predict the review rating based on the review content using a logistic regression model?**

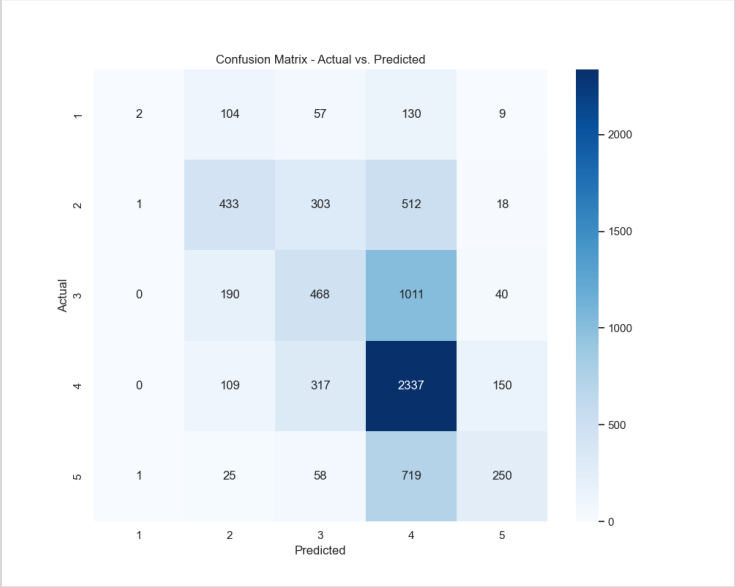

Fig. 7. Heat map of confusion matrix comparing the actual values with the predicted values

We trained a multi-class logistic regression model and got an **accuracy score** of 0.48177802319160684. This means that we cannot accurately predict the review rating based on the review but our model was able to predict 48.18% accuracy which is better than random chance guessing (20%). Additionally, our **F-1 scores**, or the harmonic mean of precision and recall, for this multi-class classification are (1): 0.32895, (2): 0.61322, (3): 0.40695, (4): 0.32143, (5): 0.01307. The score category with the best F-1 score is 2, which suggests a relatively higher performance compared to other categories, as this score is closer to 1. This model achieved a good balance between precision and recall for this score category.

We also generated a heatmap of our confusion matrix, which shows the highest accuracy in predicting the score category of 4 with 2,337 counts of actual and predicted being the same, it indicates that the model performs well in correctly classifying instances with a score of 4. The second highest accuracy in predicting the score category, was for a score of 3 with 468 counts.

The F-1 scores and confusion matrix heatmap do not reflect each other because they capture different aspects of the predictions. The F-1 score is a metric that summarizes the performance of the model across all classes. It considers both precision and recall, providing a balanced measure of the model's accuracy. On the other hand, the confusion matrix provides a detailed breakdown of the model's predictions for each class.

The result of our machine learning model was not expected at first because we did not know that the `review_content` column of our data set were just "representative pull-quotes" from the actual review, which is not fully representative of the actual review given by top critics. However, now that we know that the data set did not include the entire original written review from critics it is surprising that our machine learning model does have an accuracy score that is slightly higher than random.

The implications presented by this machine learning model and the data used to train and fit are that it is limited, the fact that the dataset only contains representative pull-quotes from the actual reviews, rather than the complete reviews themselves, suggests that the model's performance is based on a limited amount of information. This limitation does not allow for the nuances present in the original reviews. Our machine learning also has moderate performance, which suggests that there might be some patterns or features in the provided pull-quotes that can be indicative of the review rating to some extent. Ultimately, the performance of the model should be interpreted with caution when applying it to unseen data because of the fact that the model was trained on pull-quotes rather than full reviews. It is possible that the model's accuracy could differ when applied

to data that has complete review content, since the model uses a bag of words approach for its training data.

Impact and Limitations

Benefits: The word clouds can be used for the audience with an overview of what to expect from the movies and decide whether to watch the movie or not. While these word clouds can give valuable insight to audiences, filmmakers and studios can use this analysis to receive feedback on the strengths and weaknesses of movies, helping them to make improvements for future movies. Lastly, the word clouds can also assist film critics and researchers in identifying recurring patterns and sentiments in movie reviews to support their research and analysis. Yet, people who rely solely on word clouds may miss the detailed analysis of individual movie reviews.

Harms: Some of our visualizations may cause confusion for viewers. Our bar charts don't show the years of the movies. There are multiple movies with the same name, so viewers may be confused without that additional information. Even though the title says the *reviews* are from 2015-2020, it does not say anything about the years of the *movies*. Viewers should not have to assume that just because the reviews are from 2015-2020, the movies are too. One example is *Hellboy* in Fig. 2. The visualization is referring to the film made in 2019, but there is also a film with the same name made in 2004. In our visualization, *Hellboy* (2019) has an average rating of 1.96/5, but *Hellboy* (2004) actually has a score of 81% on Rotten Tomatoes (this is a huge difference!). A viewer might not know which movie the visualization is referring to and could get confused.

Biases: The analysis of word clouds is limited by potential biases in the data, such as the reliance of Rotten Tomatoes as a single review website, which may have biases based on critics' preferences. In addition, the language processing used in generating word clouds does not capture the tone of the reviews but rather represents the frequency of words. This may result in the overrepresentation of common words that are not as significant such as the word "movie" and "self". Lastly, word clouds lack granularity in differentiating between positive and negative sentiments within the same word, which can lead to inaccurate representations of sentiments in the reviews. Therefore, users should be aware of these limitations and seek additional sources for a more comprehensive understanding of movie reviews.

Limitations: The data set contains a column called `review_content`, this column holds string values of the reviews given by the critic. At first, we thought that this encompassed the entire review, however, with further examination of the data set, we realized that the values in the `review_content` column were one or two sentences. The reason for this is because Rotten Tomatoes only pulls a quote from the original review and displays it on their site, thus the `review_content` does not fully represent the entire review content by a critic and is a limitation

to the data and the answers presented in this project. This limitation affects any analysis that utilizes the `review_content` column from the data.

Also, our original dataset was *very* large, so we made some decisions to reduce its size. Firstly, we only considered reviews written by top critics. While this ensured that all reviews and ratings were credible, it omitted reviews by non-top critics who may have offered valuable insight as well. Additionally, we wanted to only consider relatively recent reviews, so we filtered the dataset even more by keeping only reviews written between January 1, 2015 to December 31, 2020. Because we omitted reviews made before this range, our findings are not comprehensive of the entire original dataset.

Our bar charts of the 20 highest and lowest rated movies (Fig. 1 and Fig. 2) only considers movies with at least 25 reviews. We originally did not take this into account, but soon found that some movies in the dataset only had one review. If that review is 5/5 or 1/5, it would appear as the "best" or "worst" movie in the visualizations, respectively. We felt that one person's review might not be representative of the overall sentiment of the movie, so we decided to filter for just the movies with at least 25 reviews. The maximum number of reviews for a movie was 78, so we thought that 25 reviews would be a good sample size. However, this meant that we did not consider movies with fewer than 25 reviews; even if a movie had a high average rating but only had 24 reviews, it would not appear in the bar chart at all.

Challenge Goals

1. **Machine learning**: We created a machine learning model that will predict a critic's rating based on the text of their review. We originally planned to create two models (a classification-based model and a regression-based model) and compare their performance, but we ultimately decided to just create one classification-based model using logistic regression and a “bag of words” technique so that our reviews can be interpreted by the model. We planned to use multi-label classification with the categories "positive," "negative," and "neutral," but decided to predict the critics' numeric ratings (from 1 to 5) instead.
2. **External library**: We used Natural Language Toolkit (NLTK), a language processing library that supports text classification, tokenization, stemming, tagging, parsing, and semantic reasoning. We were initially going to use the SentimentIntensityAnalyzer package to determine the general sentiment of a given text but incorporating the feedback we got from the TA, we decided to use the current review score and use NLTK in filtering the review comments. NLTK facilitated the process of tokenizing the review comments using the word_tokenize function and provided us with sets of stopwords and punctuations. Moreover, the use of Freqdist allowed us to determine the frequency of each word in the reviews. Additionally, we utilized word cloud, a visual representation of

textual data, to generate data visualizations showcasing the most commonly used words in each review.

Plan Evaluation

- **Cleaning data:** All group members will work collaboratively and synchronously to ensure that the data is clean and works well to achieve our project's goals. If we run into challenges, we can work together to debug and make decisions together as a team. Each group member will be there to be in support of each other.
    - Estimated time: *2-3 hours*
    - Actual time: 4 hours. Cleaning data took us a little longer than we expected, as we had struggled to convert string review scores to float values. Also, it took some time to decide how we wanted to clean the data, what values to keep, and what to take out.
- **Creating visualizations:** Each group member will create one portion of the data visualizations by themselves, and the other group members can check for quality when they are finished. If a group member is having trouble, all other group members will be there to help and support each other.
    - Estimated time: *2 hours*
    - Actual time: around 2 hours. We each worked on our own data visualizations. Although we had struggled downloading external libraries into VSCode, we helped and supported each other.
- **Creating ML model:** All group members will work collaboratively and synchronously to ensure that the ML model is built well.
    - Estimated time: *3 hours*
    - Actual time: 2 hours. The actual time spent was shorter than we expected because of the resources and guidance provided by our mentor, however we encountered challenges when trying to generate a visualization that effectively shows the result of the ML model. Despite these challenges, we collaborated and managed to create a heatmap to represent the result.
- **Testing code**: All group members will work together to ensure that the code is clean, works, and meets the quality guidelines.
    - Estimated time: *2 hours*
    - Actual time: 1 hour 15 mins. The actual time spent was shorter than we expected. We achieved this by utilizing a smaller dataset, specifically each selecting 10 movies from both the top and bottom 50 movies. By importing the functions in main.py, we were able to save time.
- **Write report:** Group members will collaborate on a Google Doc to discuss our project goals, methodology, and results.
    - Estimated time: *2 hours*

- Actual time: 2-3 hours. Some parts took longer than we expected due to length of the document.

Testing

Since our research questions were centered around creating data visualizations and a machine learning model, we felt it was best to test our code by creating a subset of our dataset and making sure our visualizations and model still made sense.

For our first research question, we used the movies with at least 25 reviews (471 in total) to create a smaller dataset consisting of 40 of these movies and their reviews. 20 movies had high average ratings and the other 20 had low average ratings. For our two bar charts, we first made sure the title, axis labels, and overall plot looked correct. We then calculated the average rating for each movie and made sure the charts displayed the movies in the correct order with the correct average ratings. We also noticed that if multiple movies had the same average rating, they are displayed in the same order as they appear in the dataset (e.g., *Pixels* and *The Mummy* in Fig. 2). We made sure that this was also the case for our test dataset (e.g., *Spider-Man: Into the Spider-Verse* and *The Farewell* in Fig. 8).

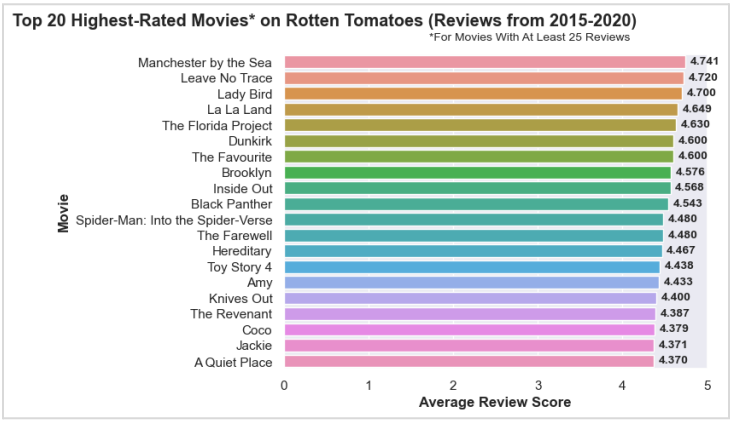

Fig. 8. Bar chart of 20 highest-rated movies from testing subset

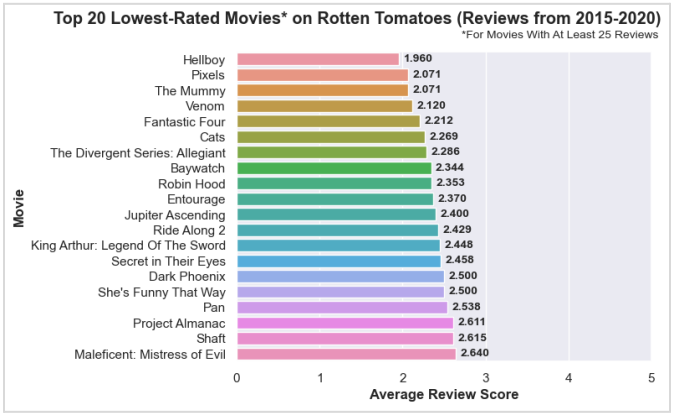

Fig. 9. Bar chart of 20 lowest-rated movies from testing subset

For our two world clouds, we used the same dataset consisting of 40 to create world clouds and analyze the most frequently used words. Before proceeding with the code, we conducted a manual review of the movie reviews and identified several common words that appeared in each review. Subsequently, we calculated the frequency distribution of words in each review and generated the word clouds. Notably, we observed that the words we initially discovered were present in the world clouds. In addition, the presence of words such as “movie” and “film” in both word clouds also indicates that it produces similar outcomes to the word clouds depicted in fig. 3 and 4.

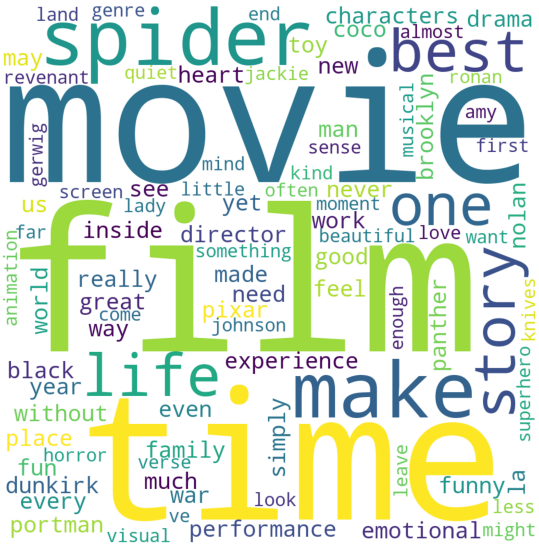

Fig. 10. Word cloud of top 20 highest-rated movies from testing subset

Fig. 11. Word cloud of top 20 lowest-rated movies from testing subset

Additionally, for the second research question we used the above subset of data to be used with the function that produces a scatter plot that compares the average word count and average review score per movie to test that the function operates correctly or outputs similar results on both datasets. Fig. 12 (test scatter plot), presents similar results to Fig. 6, in which they both do not have a significant relationship. The only difference is that the test scatter plot displays most reviews per movie with an average word count between 22-25 words and there is no average low ratings. However, this may be because of the reviews selected for the testing dataset (we selected only movies from the top and bottom average review scores). We know that the code computes correctly while visually comparing Fig. 12 to Fig. 6.

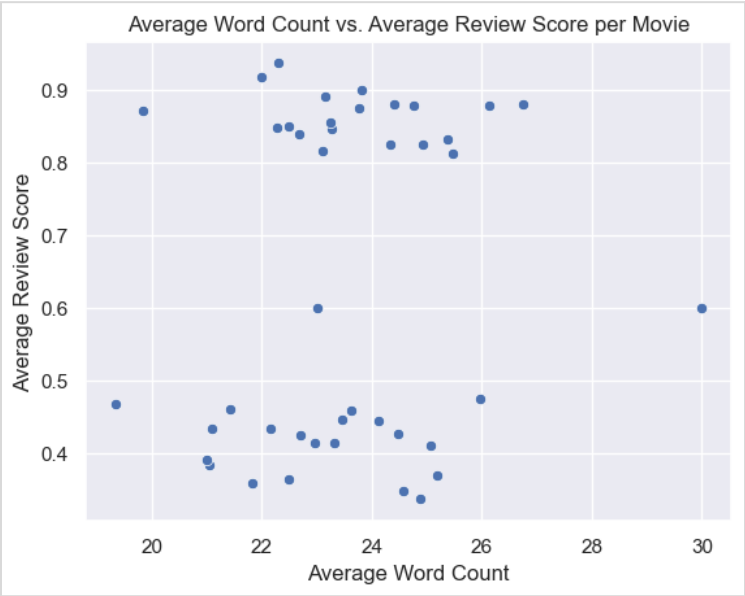

Fig. 12. Scatter plot created with the testing data set

Lastly, when testing machine learning with the smaller subset of data the accuracy score improved, however, we know that this is because the smaller subset of the data contains bias (on purpose, for testing the above research questions). On the other hand, the accuracy score of the machine learning model with the main data set was ~48.18% accurate, with 80% of the data going to training, and 20% going into the testing set.

Collaboration

Aside from the course staff, we went over our course readings, assignments, and lectures to review Python functions and proper code quality/documentation guidelines. We also referred to the documentations of various libraries such as Seaborn, Matplotlib, NLTK, and wordcloud to use and customize functions. Any resources we used are commented as links in the code files. Finally, we used StackOverflow posts to guide us if we had any specific questions.