Harika Yadav Gillela

Mohanaditya Karampudi

Shoalin Pu

**Movie Recommendation**

Table of Contents

1. **Executive Summary…………………………………………………………………………………………………….2**
2. **Problem Definition & Significance…………………………….…………………………….………………….2**
3. **Prior Literature…………………………………………………………………………………………………………..3**
4. **Data Source and Preparation………………………………………………………………………………………4**
5. **Text Analytics Workflow…………………………………………………………………………………………....5**
6. **Exploratory Data Analysis & Visualizations…………………………………………………………………6**
7. Ratings and Genre Distribution……………………………………………………………………………..…6
8. Movie Directors Analysis………………………………………………………………………………………….7
9. Comparing Audience score and Critic score……………………………………………………….......8
10. Ratings and Runtime trends over Release year brackets………………………………………....9
11. **Text Analytics and Results………………………………………………………………………………………..10**
12. Building Model……………………………………………………………………………………………………….10
13. Get Recommendations…………………………………………………………………………………………..11
14. Results……………………………………………………………………………………………………………………12
15. **Insights and Recommendations………………………………………………………………………………..13**
16. **References……………………………………………………………………………………………………………….13**
17. **Appendix………………………………………………………………………………………………………………….14**

# **Executive Summary**

Over the last decade, most of the world’s biggest entertainment and telecom have moved enormous attention to the new battlefield of streaming entertainment. From Statista’s report of *Frequency of streaming movies in the U.S. 2019-2020* (Watson, 2020), there are a total of 25 percent of adults who aged between 18 and 29 years old and a total of 9 percent of adults who aged above the age of 65 years old claimed that they watched movies every day. In response, there are more than 750 movies released in the United States and Canada in 2018 and 2019 (Watson, 2020). This growth of online movie consumption has brought a tremendous amount of information and choices to consumers and requires a better movie recommendation system to help consumers find the right movies they want. In general, content-based and collaborative Filtering recommendation system are the two most popular approaches.

Content-based and collaborative filtering recommendation both rely on the item-user interaction. Beside using users’ interactions and feedbacks, the content-based recommendation also requires rich information of features of the movie to find new movies that are similar to what the user has watched and liked in the past. On the other hand, collaborative filtering recommendation requires a good amount of users’ information that helps find similar users and create a recommendation list based on what a similar user enjoyed.

However, both recommendation systems have to face three problems that are: cannot recommend fresh items, cold start problem, and lack of side features for the query. In this project, we present a new approach to designing a new content-based recommender system to overcome these problems.

**Problem Definition & Significance**

Entertainment plays a very big role in ensuring that people live normal and happy lives. Movies are considered as an important art form and major source of entertainment. There are three problems we try to solve in this project.

1. **Cannot recommend fresh items**. This problem is caused by overspecialization that means recommendation system limits the list of movies based on user’s watched list and feedbacks and can’t recommend items that user and other similar users haven’t seen before.
2. **Cold start problem.**Content based and collaborative filtering recommendation rely on users’ information so that those two recommendation systems can’t generate suggestions if there are not enough user’s information.
3. **Lack of side features and cannot query without using movie’s name.** Commonly, recommendation system includes the features of film name, film ID and other official information, for example, the movie name, genre, PG rating, length of movie and release year. Side features are any features beyond film name and film ID.

In this project, our goal is to build a new content-based recommendation to solve these problems by bringing in more features from critics’ and audiences’ reviews and not rely on user historical performances. During the pandemic, the whole world is streaming more than ever, and daily consumption of movies increase marginally among U.S. (Alexander, 2020). We believe there will be a good amount demands for a better movie recommendation system.

**Prior Literature**

There are several similar research focuses on creating better movie recommendation systems that can generate personalized recommendation-content and overcome the problems of cannot recommend fresh items, cold start problem, lack of side features and cannot query without using film’s name. A popular method to achieve these goals is to use direct impact, which includes user’s rating, like, and share, and indirect impact, which includes user history, to create Apriori. Then using maximum entropy principle and LDA to find semantic relation and create a recommendation list of every given movie.

In order to bring in new features that represent the real feeling of audiences when they watch the movie, some research propose an approach of movie review mining and summarization that identifies the words of feature and opinion from the reviews and pairs feature with opinion to produce a summary of review. Then add those feature-opinion pairs into movie’s feature list. This approach can help solve issue of cannot recommend fresh items that could enable recommendation system find new related fresh movie for users.

To improve performance, content-based recommendation with weights is also commonly used. Some research proposes a multi-knowledge-based approach to link the movie to different feature and predict possible link to other movies. Meanwhile, user’s viewing history of each movie are converted to an implicit rating. In order to filter low quality user and generate quality recommendation, some research compute a feature weight based on user’s watching habit and user’s past rating for each movie. The recommended movie list is sorted according to the weighted ratings of each movie.

**Data Source & Preparation**

There are multiple online movie review websites available. We get numerous results for a movie review from google search engine, but it’s hard to find a website with steady results. We have chosen the website Rotten Tomatoes to scrape the movie data as it is the most popular website. Rotten Tomatoes [https://www.rottentomatoes.com](https://www.rottentomatoes.com/) is an American review-aggregation website for film and television. The company was launched in August 1998 by three undergraduate students at the University of California, Berkeley: Senh Duong, Patrick Y. Lee, and Stephen Wang. In rotten tomatoes, reviews are taken from a large number of regional and national film critics.

The data was scraped using beautiful soup. We scraped data of around 10,000 movies with the following attributes: Movie Name, Director, Rating, Rating in words, Top 5 celebrities, Critic Score, Audience Score, Critic Consensus, Runtime, Studio, Genre, Release Date. Most of the columns were directly scraped from the data but two columns Rating in words and release year were derived from columns ‘Rating’ and ‘In theatre’ date respectively. The columns’ Critic Score, Audience Score, Runtime, and Release date had a lot of information which in turn will create many features and increase sparsity. To avoid this, we have created bins and categorized the data into a few categories.

The text data was then cleaned and preprocessed for analysis. The process involved removing punctuation and stop words, replacing null values, deleting duplicates based on the movie name, lemmatizing, and stemming. Figure1 shows the generated corpus of the data.

A picture containing table

Description automatically generated

Figure1: cleaned corpus of data

# **Text Analytics Workflow**

Once the data is scraped from Rotten Tomatoes, we made sure that we don’t have duplicate movies data and performed preprocessing and data cleaning. Later, the extracted features are explored and visualized. To build a content-based recommendation system we chose to combine features and calculate the cosine similarity. Later, Similarity score analysis was performed to identify the unique top 10 movies.

Extract top 10 recommendations

Load scarped data from Rotten tomatoes

Rank movies based on similarity

Data Cleaning Preprocessing

Get movie Recommendations for given input

Create categories for Release year, runtime, Audience score,

Critic Score

Find Cosine Similarity

Find Cosine Similarity

Exploratory Data Analysis & Visualization

Text Vectorization: Create TFIDF matrix

Combine features

Fig 2: Flow chart of the process

**Exploratory Data Analysis & Visualizations**

## **Ratings and Genre Distribution**

To get better insights of the data we performed exploratory data analysis before building the model. We worked on finding answers to the following questions:

* Number of Ratings and Genres
* Number of movies per Director
* Comparing audience score with critic score
* Ratings for Release year brackets
* Runtime trend over Release years

Chart, pie chart

Description automatically generated

Movies are given ratings by audiences and

critics to rate the film’s suitability for certain audiences based on its content. Figure 3 explains the distribution of ratings in our dataset. Our dataset had more than 50% of movies with rating R followed by PG-13 (28%). We had the least number of movies with rating NC-17.

Figure 3 Movie Ratings Distribution

**Description of the 5 categories of Ratings:**

|  |  |
| --- | --- |
| G – General Audience | All ages admitted. |
| PG – Parental Guidance Suggested | Some material may be inappropriate for children |
| PG-13 – Parents Strongly Cautioned | Some material may be inappropriate for children under 13 |
| R-Restricted | Under 17 requires accompanying parent or adult guardian |
| NC-17 – Adults Only | No One 17 and Under Admitted. |

In rotten tomatoes few movies had multiple genres, for example, the movie Knives out included Genres “Comedy, Drama, Mystery and Thriller, Crime “.  With the movies having many Genres our dataset had different combinations of unique Genres. In total, we had 154 unique Genres. Figure 4 below shows the distribution of the top 10 genres in our dataset. In our data, maximum number of movies had either had 1 or 2 genres, with action and adventure movies being maximum, followed by drama, art, and comedy.

Chart, pie chart

Description automatically generated

Figure 4: Top 10 Movie Genres

## **Movie Directors Analysis**

Our dataset of 10000 movies had 6453 unique movie directors. We wanted to check the distribution of the number of movies directed by each director. Figure 5 below shows the top 20 directors based on the total number of movies directed by each director. We see that the director with the maximum number of movies is Tyler Perry who has directed 16 movies in total followed by Steven Soderbergh with 15 movies.

Chart, bar chart

Description automatically generated

Figure 5: Top 20 movie directors

## **Comparing audience score with critic score**

There are many instances where audience reviews can be very different from critic reviews. There are several blockbuster movies that have been appreciated by audiences but not by critics. On the other hand, there are quite a few art films that critics adored, but not audience. This was clearly seen in rotten tomatoes.

The columns audience and critic score brackets are categorized based on the scores. We have scores between 0 -100 with brackets as shown below:

* < 20 -- worst
* 20 - 40 -- below average
* 40 - 60 -- average
* 60 - 80 – best
* 80 – 100 -- good

Chart, bar chart

Description automatically generated

Figure 6: Audience Score and Critic Score

Looking at figure 5 we can conclude that:

* Movies rated with average score by audience have all 5 categories of critic score brackets with more than 50% of movies rated as good and best.
* About 80% of the movies that were appreciated by audience as good movies are rated mostly as good or best by critics. The movies rated as best by audience are also rated as best or good by critics. We see that there is similarity of opinion here between audience and critics. These movies had similar audience score and critic score.
* The movies that had worst audience score “less than 20” have critic score more than 60%, rated as good and best.

## **Runtime trends over Release year brackets**

Similar to bins created for audience score and critic score we have created bins for the runtime and release years. We have 6 categories of release years and 4 categories of runtimes.

Release Year

* 1920-1950 – Black and White
* 1950-1980 – Pretty old
* 1980-1990 – year\_80\_90
* 1990-2000 – year\_90\_2000
* 2000-2010 – year 2000\_2010
* 2010-2020 – year 2010\_2020

Runtime

* 10-90 – Short\_till90
* 90-120 – conventional\_90\_120
* 120-180 – Lengthy\_120\_180
* 180-800 – Superlengthy\_180\_800

Figure 7 below shows the distribution of the runtime of the movies over the release years. Our dataset had maximum number of movies with release years between the years 2010-2020 with most of them lasting for 120 minutes. The movies with runtime between 90-120 are most common in all the release year brackets from 1920-2020

Chart, bar chart

Description automatically generated

Fig 7: Runtimes over release years

**Text Analytics & Results**

1. **Building Model**

After cleaning the text and performing exploratory analysis, our approach was to build a content-based recommender system, that recommends items based on the **similarity**between the content. To find the similarity between the items we can use either Euclidean or Cosine Similarity.

Diagram

Description automatically generated

While cosine looks at the angle between vectors (thus not taking into regard their weight or magnitude), Euclidean distance is like using a ruler to measure the distance.

Figure 8: Euclidean and Cosine Similarity

In this project, we built a machine learning model based on cosine similarity, which was done in the following steps:

**Combining Features**

We combined all the relevant features to a single feature. We combined all the columns of final dataset and created a new column “combined” in a new dataset. The column “combined” will contain useful features from the respective rows in a combined single string.

**Vectorization**

The sklearn.feature\_extraction.text was used to extract features from the data consisting text. We used TF-IDF Vectorizer and used the result matrix to find the cosine similarity.

**Finding Similarity**

The linear\_kernel from sklearn.metrics was used to calculate the similarity between the movies. The matrix obtained from the linear\_kernel is an array with calculated similarity between the movies. Figure 10 shows the cosine similarity matrix; the cosine similarity of movie 0 with movie 0 is 1; they are 100% similar. This suggests that every movie is similar to itself. The similarity between movie 0 and movie 1 is 0.0103, similarity score = 0.0103

Table

Description automatically generated

Figure 10: Cosine Similarity matrix

1. **Get Recommendations**

The next step in our process is to recommend the similar movies to the user, the input to this is the users’ current selected movie. In order to extract the recommendations, we add a count to the list of movies and similarity scores sorted in descending.

Table

Description automatically generated

Figure 11 shows sorted list of the movies in the descending order based on the similarity score. The user input here is the movie ‘mulan’ which is at index 2. The index 0 shows the similarity of the movie with itself, followed by movies 3709, 411 etc. Later, these indexes are replaced with the titles.

Figure 11: Similarity scores

To display the recommendations to the user we map the index number to the movie name which are already sorted as in figure 11. Here we are considering only the top 10 movies from the sorted list.

1. **Results**

To test the model, we did few manual explorations by giving various movies and inputs. Figure 12 shows the recommendations of the movies “the lego movie 2: the second part”,” get out” and “Knives out”.

Text

Description automatically generated

Figure 12(a): Results 1

The Lego movie second part has movies suggested that very similar to the input. All the movies suggested are animated movies, that have genres as comedy, fantasy and kids adventure.

A picture containing text

Description automatically generatedText

Description automatically generated

Figure 12(b & c): Results 2

The movie get out, which is a kind of horror and mystery film has suggestions involving similar actors (movies of Daniel Kaluyya (Queen & slim, chatroom) , Catherine Keener (Hamlet 2, the oranges, little pink house) and few other horror and mystery movies like drag me to hell, ready or not etc. Similarly, the movie Knives Out which is a drama, mystery and thriller has recommendations of similar genre and cast (Casino Royale of Daniel Craig, The night clerk of Anna De Armas).

Since we didn’t have a definite metric to evaluate the recommendations, we have manually explored the recommendations and ensured that the recommendations were reliable which can be improved further by adding metrics to evaluate accuracy of recommendations and increasing the data.

**Insights & Recommendations**

With our approach, we find the movie recommendation system can be easily trained on new dataset and produce reasonable results thus making it easily scalable. However, lack of evaluation method and humongous data are the two biggest challenges of this project. Even the model can give reasonable recommendation within dataset, it may not work on any other random movie that comes from the outsides of our dataset. For feature research, multiple models with different approach should be built. By comparing our model with other models, we can have a sense of how good this model is and how good the recommendations are.

To overcome data problems, we need to break the predictor features that has large size into clusters and generate new categories to reduce the size and duplicate of feature. Then applying TF-IDF approaches after better data cleaning and feature engineering. We also need to think about give a weight to every feature and reduce the impact of less important feature on recommendation.

**References**

Zhuang, L., Jing, F., & Zhu, X. Y. (2006, November). Movie review mining and summarization. In Proceedings of the 15th ACM international conference on Information and knowledge management (pp. 43-50).

Understanding various attributes and measures of the movies <https://www.rottentomatoes.com/about>

Frequency of watching or streaming movies among adults in the United States in 2019 - 2020

https://www.statista.com/statistics/935492/movies-watching-streaming-frequency-us/

https://www.statista.com/statistics/187122/movie-releases-in-north-america-since-2001/

Movie Streaming more than ever – importance of recommendation systems

<https://www.theverge.com/2020/3/27/21195358/streaming-netflix-disney-hbo-now-youtube-twitch-amazon-prime-video-coronavirus-broadband-network>

Content Based Recommendation systems and approaches

<https://towardsdatascience.com/introduction-to-two-approaches-of-content-based-recommendation-system-fc797460c18c>

Cosine Similarity – Understanding how it works

<https://www.machinelearningplus.com/nlp/cosine-similarity/>

**Appendix**

GitHub repository link: <https://github.com/aditya-karampudi/rotten_recommendation>

This link has the web scraping and model building python notebooks with raw corpus, cleaned corpus data.

Python Model building notebook is attached in submission along with the document.