CineMatch: A Content Based Recommendation System For Movies and T.V Shows

Avantika Prativadhi

May 8th, 2023

Abstract

Recommendation systems are very useful in the world today with every platform using them to enhance user experience. The aim of this research paper is to develop a content-based recommendation system for movies and TV shows called CineMatch. The system utilizes features such as actors, directors, plot, number of seasons, age certification, production country, and genre, along with the popularity and score data from TMDB and IMDb to accurately recommend a movie to a user and also predict the rating of a movie or a TV show. The motivation behind this project is to improve user experience and increase engagement on entertainment platforms. This paper discusses the importance of recommendation systems and the hope to learn more about applying various analysis methods and algorithms on multiple datasets. The project involves using different datasets from Netflix, Hulu, Amazon Prime, and HBO Max. The data was cleaned and combined into one major data frame. Statistical modeling and machine learning techniques such as feature engineering, similarity measures, clustering, and prediction using regression algorithms were used for the analysis. The software used for this project is Python and Jupyter Notebook. Overall, this project provides a learning experience and an opportunity to gain insight into recommendation systems while applying various techniques and algorithms.

1 Introduction

Recommendation systems have become very evident in our daily lives. From online shopping to social media and job searching to entertainment platforms, these systems have become an integral part of our online experience. In particular, streaming services such as Netflix, Hulu, HBO Max, and Amazon Prime heavily rely on recommendation systems to personalize content for their users and improve overall user experience.

Despite their prevalence, recommendation systems are still a complex and challenging field of research. One of the main goals of these systems is to accurately predict a user's preference for a given item based on their past interactions and behaviors. To achieve this, recommendation systems typically use content-based filtering, collaborative filtering, and/or hybrid filtering techniques. Collaborative filtering (CF) approach is where, "a model learns from a user's past behavior as well as similar decisions made by other users, and predict items (or ratings for items) that users may be interested in."[1]. While this method is widely used, there are some limitations that are present with it like the problem of sparsity and cold start. Sparsity occurs with a large number of users and items, it can be difficult to find similar users or items which can affect the accuracy of the recommendation. Cold Start problem occurs when there's a lot of new users or new movies that there's not enough information to make any recommendation.

On the other hand, Content based filtering," takes some information which can de-script the characteristic of item as its training data input." [2]. The limitation with this is the problem of overspecialization. Since this recommends items that are similar to what the user has already liked or interacted with, it can lead to the recommendations being too similar and prevents them from discovering new items. Finally, the hybrid approach is a combination of the two methods and is most commonly used. While this can deal with a lot of issues of the previous two, there is a problem with the scalability of this. With both methods combined, there's data on the individual movies as well as user data which can be a large amount of complex data that can affect the performance and accuracy of the system. Although the systems we have in place today are very accurate to a certain extent, they can still use more research and additional improvements to make user experience even more better.

Out of the three major ways of filtering, this paper proposes a content-based recommendation system for movies and TV shows called CineMatch. Rather than using personal and individual user data, this system utilizes a wide range of features of the movie/show itself such as actors, directors, plot, number of seasons, age certification, production country, and genre, along with popularity and score data from TMDB and IMDb. The goal is to mitigate that issue of overspecificity by adding many features and changing the way the recommendation is provided. The research question is to see if a content- based recommendation system utilizing features such as director, plot, number of seasons, age certification, production country, genre, along with popularity scores from IMDB and TMDB accurately predict a user's rating of a movie/ show and recommend the right shows. The project is split into two parts: Recommendation and Prediction.

The motivation for the research stems from my personal experiences with recommendation systems. While these systems have greatly improved the ability to discover new content, they are not always perfect. We have all encountered situations where the recommendations we receive are far away from what we are interested in, leaving us to waste a lot of time starting a show/movie that we definitely are not captivated by. Additionally,I'm also curious regarding the work put into establish something like this and would like to get a better understanding on the inner workings of recommendation systems and explore ways to improve their accuracy.

With this research, I hope to contribute to the broader field of recommendation systems and machine learning by exploring the effectiveness of different techniques in a real-world setting. The rest of this paper is organized as follows. In section 2, the literature review with some of the past work done will be provided. Then, part 3 discusses the data and describes the resources that were used for the project. Part 4 and 5 contains the analysis and the methods along with all the insight obtained through the models. Finally, the conclusion will go over the potential future works.

2 Literature Review

Over the years, there has been extensive research done on the topic of recommendation systems. One of the major issues that may arise with collaborative filtering is the cold start problem where if there is not much information on the user due to them being new, it will be difficult to find movies suited to their taste. This is where content based filtering can be helpful. Given that the movie has a clear and detailed description and metadata, this cold start problem can be solved. Salton Gerard [3] discusses the uses the textual information from the descriptions to find item features which are extracted and represented as keywords with respective weighting measures calculated by term frequency and/or inverse document frequency(TF/IDF) measure. M. K. Delimayanti and

other researchers [4] implemented a content-based filtering based on genre connection using cosine similarity and KNN algorithm to improve accuracy. H.Khatter [5] concluded that cosine similarity provides better and efficient results for a recommendation system. The research that this paper will focus on is the "A New Approach for Movie Recommender System using K-means clustering and PCA" where author Yadav Vikas uses methods like cluster analysis and PCA for a content based filtering system[11]. While that research focuses on the MovieLens dataset, this will focus on the various specific platforms that are popular today and will also add in the prediction section to further elaborate the work.

3 Data Description

The datasets used in this project were obtained from Kaggle and were scraped from a streaming guide called "JustWatch". The collaborator for this project is Victor Soeiro. The project includes four different datasets: Netflix, Hulu, HBO Max, and Amazon Prime. The dataset contains two CSV files each , one for credits and one for titles. The titles data frame contains information about the movie like the title, description, release year, runtime, genre, etc. The credits file contains information about the name of the actor/director and their role in the show/movie. Both these CSV files were combined into one with a new column called "service" added, which specifies which of the four platforms (Prime, Hulu, Netflix, HBO Max). The type of variables in the datasets can be seen in figure 1. These four were combined into one major data frame called "df", which serves as the metadata for this project.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 157366 entries, 0 to 1216
Data columns (total 20 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   id                    157366 non-null  object
 1   title                 157366 non-null  object
 2   type                  157366 non-null  object
 3   description           157270 non-null  object
 4   release_year          157366 non-null  int64
 5   age_certification     81836 non-null   object
 6   runtime               157366 non-null  int64
 7   genres                157366 non-null  object
 8   production_countries  157366 non-null  object
 9   seasons               18012 non-null   float64
 10  imdb_id               150945 non-null  object
 11  imdb_score            150126 non-null  float64
 12  imdb_votes            150098 non-null  float64
 13  tmdb_popularity       157351 non-null  float64
 14  tmdb_score            146844 non-null  float64
 15  service               157366 non-null  object
 16  person_id             157366 non-null  int64
 17  name                  157366 non-null  object
 18  character             138802 non-null  object
 19  role                  157366 non-null  object
dtypes: float64(5), int64(3), object(12)
memory usage: 25.2+ MB
```

Figure 1: The combined dataset and the attributes

Before starting the process of creating the model, the data was cleaned and explored to understand the initial shape of the data and to see if any patterns could be discovered as this would help in the variable selection process. To treat the missing values, they were either completely removed depending on their importance in the research or were replaced using a process call mean imputation where the null values were replaced by the mean of the column. The figure below shows the visualizations obtained through the data exploration process.

The plots show the breakdown of what the data looks like. It appears that the number of

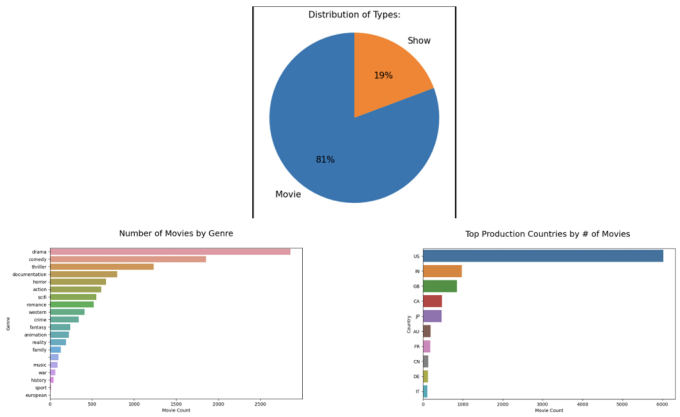


Figure 2: Visualizations of the movies/tv shows data

movies in the dataset are a lot more than the TV Shows. The movies also seem to be majorly produced in the US. and are in the genre of drama. This does show a skewness of the data and there's no uniform distribution of how the moves/tv are distributed within the categories. Here, the age ratings of the movies/ TV shows between the four platforms in shown. This seems to have a normal distribution for all the platforms with the rating R having the most count throughout.

Based on these plots, it's evident that the movies that are going to be recommended will be focused on the given information. R rated drama movies produced in the United States will more likely be recommended.

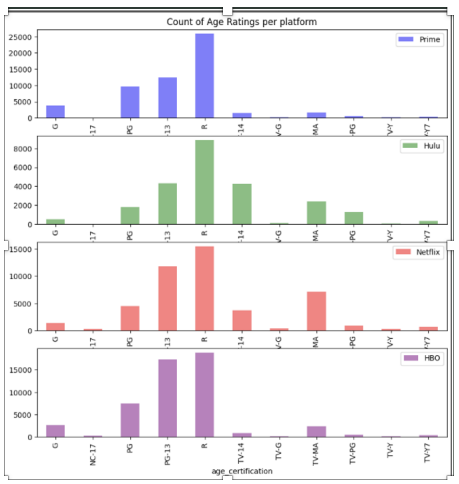


Figure 3: Distribution of Age Ratings by Platform

4 Model Building

This paper focuses on the following methods to conduct the analysis : Feature Engineering, Similarity Measures, Principal Component Analysis, Cluster Analysis, and Support Vector Machines. The sci-kit learn package in Python was used to conduct the analysis.

4.1 Weighted Ratings

The dataset has four different columns pertaining to the ratings of the movies/shows: tmdbpopularity, tmdbscore, imdbscore, and imdbvotes. To see how these various categories can be combined to get one solid score, IMDB has a formula for weighted ratings which can be found in figure 4. Using that formula, each movie or show was given a weighted rating. This new rating is fed into the models for the recommendation systems.

$$W = \frac{Rv + Cm}{v + m}$$

where:

W = Weighted Rating
R = average for the movie as a number from 0 to 10 (mean) = (Rating)
v = number of votes for the movie = (votes)
m = minimum votes required to be listed in the Top 250 (currently 3000)
C = the mean vote across the whole report (currently 6.9)

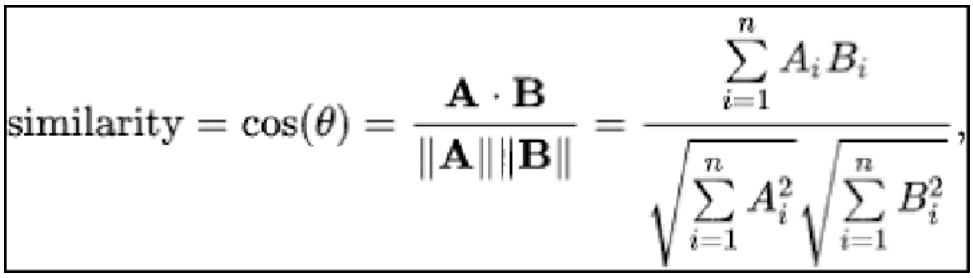
Figure 4: Weighted Ratings

The ratings derived by this formula show an accurate representation of how well the movie is doing as it takes the popularity into consideration. Since all the movies don't have the same number of votes, this formula allows that to be averaged out so no one movie gets a higher rating solely due to the large number of votes. This helps standardize the scores and gives all the movies a fair chance to be recommended.

4.2 Cosine Similarity

Cosine similarity among two objects measures the angle of cosine between the two objects. It compares two documents on a normalized scale. It can be done by finding the dot product between the two identities." [6]. The features used were the title, type, genre, description, actor, director, weighed score, and production country. Using these features, one movie is compared to the features of another movie. For the analysis, all the features are first converted to a matrix and all the categorical and text variables are converted to numerical features using the Count Vectorization function provided by the Sci-kit learn library. Then, the linear kernel function was used to calculate the cosine similarity between the movies in the matrix. The formula for the cosine similarity is given in figure 5.

To evaluate the performance of the model, the dataset was split into training and testing data with a 80:20 ratio where 20 percent of the data was used for testing and 80 percent for training. Precision, recall, and F1 score were calculated to evaluate the performance of the model. The results showed that the model achieved a precision of 0.8790, recall of 0.8836, and F1 score of 0.8766.



$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}},$$

Figure 5: Formula for Cosine Similarity

Precision, recall, and F1 score are measures of a model's performance in terms of its ability to correctly identify positive instances (True Positives) and negative instances (True Negatives) in a dataset. The scores achieved by this model indicate the good performance of the model. Although, there is still room for improvement.

4.3 Principal Component Analysis

The next approach used to see if this model could be further improved was cluster analysis. But Since the data had a lot of categorical variables, they had to be converted into dummy variables using one-hot coding method, the total variables went up to 29,447. With this large number of features to go through, the computation of the model was very slow.To help reduce the dimensionality of the data with minimum data loss, Principal Component Analysis (PCA) was applied. PCA is a technique that identifies patterns in data and summarizes them by creating a set of new variables called principal components [11]. Since PCA does not do well with missing values, they were imputed using sklearn's SimpleImputer. Then, the data was standardized and fit into the model to retail 95 percent of the variation in the original data. This was then transformed to be used for the cluster analysis.

4.4 Cluster analysis

Subsequently, cluster analysis was carried out using k-means clustering. This clustering algorithm gets recommendations for a given movie based on the cluster label and distance between the other movies in the same cluster. An elbow curve was first used to find the optimal number of clusters and then Euclidean distance was used to find the distance between the movies. The silhouette score, a measure of how similar an object is to its own cluster compared to other clusters, was 0.7923. Showing that the movies were put in a fairly similar cluster.

Lastly, based on the clusters generated, a function was developed to recommend a movie. The function accepts a movie as input and recommends a similar movie from the same cluster. Figure 6 gives a look at what the output is.

The scores at the end of the bars are the similarity scores calculated using the cosine similarity function. The total appears to be out of 0.40 and all the movies close to that and similar to the movie that was input is provided. The scores shows how similar the recommendations are to the given movie.

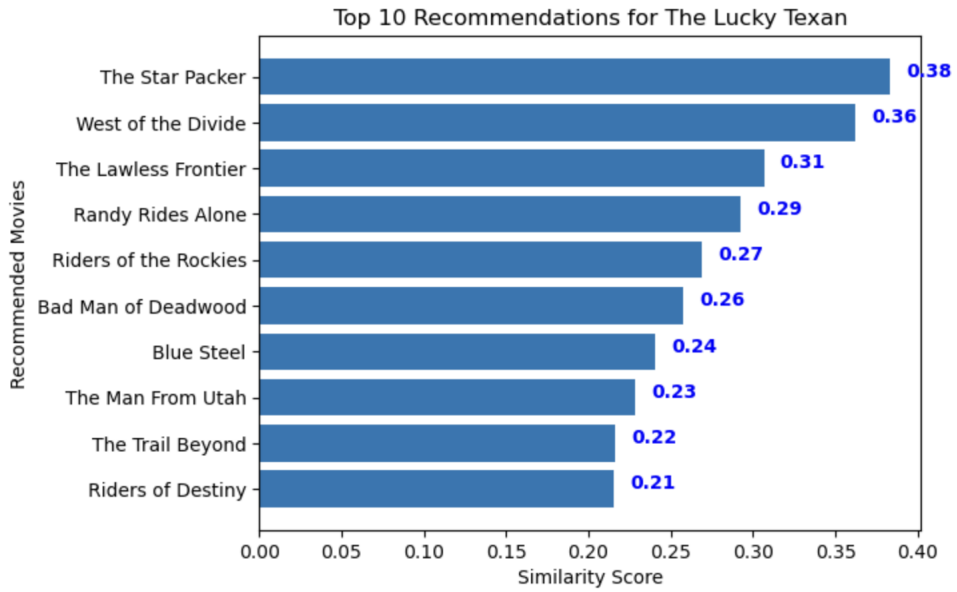


Figure 6: The recommendation system presents a list of 10 similar movies along with their similarity score

5 Prediction

After the recommendation section, we move to the prediction to see if the features we have been using throughout this paper will also help accurately predict the score that the movie/show would get. To find the best model to train the dataset on, the r-squared valued and mean squared error for the four regression models: Linear Regression, Decision Tree, SVR, and Random Forest. R-squared value is," a statistical measure that indicates how much of the variation of a dependent variable is explained by an independent variable in a regression model."[10].So this value indicates how well the predictors are to the response variable, the closer the variation is to 1, the better it usually is.The linear regression model had an r-squared value of 0.34, the decision tree had an r-squared value of -0.1069, the random forest had the value of 0.2116, and the SVR had the value of 0.524. While none of the scores were really high, the SVR model seems to be doing the best of out of them. Support Vector Regression (SVR) finds a hyperplane in a high-dimensional space that maximally separates the target variable into different classes of regression values. To conduct this regression, the data was split into test and train sets. The model was then fit on them to get a prediction score for the movie that is given. Based on the points, the regressor is able to output the predicted scores. The movies seem to have a consistent set of scores were were par to what the weighted ratings showed. The movie that returned the highest predicted score was the movie Dexter with a score of 8.47905. So, no movie was able to get that full 10 but still got pretty close.

6 Conclusion

To conclude, this study leveraged cosine similarity and cluster analysis to generate movie recommendations based on a range of features. The precision, recall, and F1 score metrics demonstrated the model's effectiveness in identifying similar movies. Nevertheless, it should be noted that the recommendations are confined to the movies included in the given dataset.

Future work for this study involves integrating the system with an open-source database to extend the recommendations to a larger collection of movies. Furthermore, I intend to focus on enhancing the accuracy of the prediction model by modifying the data, variables, or model type to determine if any adjustments could more precisely forecast the scores.

Despite the limitations, this study lays a solid groundwork for constructing a robust movie recommendation system. With further enhancements and expansion, this system could offer valuable recommendations to a wider audience of movie enthusiasts.

References

[1] Wang, D., Liang, Y., Xu, D., Feng, X., andamp; Guan, R. (2018, May 17). A content-based recommender system for Computer Science Publications. Science Digest. Retrieved April 10, 2023, from `https://www.sciencedirect.com/science/article/pii/S0950705118302107`

[2] H. -W. Chen, Y. -L. Wu, M. -K. Hor and C. -Y. Tang, "Fully content-based movie recommender system with feature extraction using neural network," 2017 International Conference on Machine Learning and Cybernetics (ICMLC), Ningbo, China, 2017, pp. 504-509, doi: 10.1109/ICMLC.2017.8108968.

[3] Salton, G. (1989). Automatic text processing: The transformation, analysis, and retrieval of. Reading: Addison-Wesley, 169.

[4] M. K. Delimayanti et al."Web-Based Movie Recommendation System using Content-Based Filtering and KNN Algorithm," 2022 9th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE), Semarang, Indonesia, 2022, pp. 314-318, doi: 10.1109/ICITACEE55701.2022.9923974.

[5] H. Khatter, N. Goel, N. Gupta and M. Gulati, "Movie Recommendation System using Cosine Similarity with Sentiment Analysis," 2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 2021, pp. 597-603, doi: 10.1109/ICIRCA51532.2021.9544794.

[6] Singh, R. H., Maurya, S., Tripathi, T., Narula, T., and Srivastav, G. (2020). Movie recommendation system using cosine similarity and KNN. International Journal of Engineering and Advanced Technology, 9(5), 556-559

[7] Choi, S.-M., Han, Y.-S.: A content recommendation system based on category correlations. In: The Fifth International Multi-Conference on Computing in the Global Information Technology, pp. 1257–1260 (2010)

[8] D. Das, H. T. Chidananda and L. Sahoo, "Personalized Movie Recommendation System Using Twitter Data" in Progress in Computing Analytics and Networking, Singapore:Springer, vol. 710, 2018.

[9] N. Mishra, S. Chaturvedi, V. Mishra, R. Srivastava and P. Bargah, "Solving Sparsity Problem in Rating-Based Movie Recommendation System" in Computational Intelligence in Data Mining, Singapore:Springer, vol. 556, 2017.

[10] Fernando, J. (2023, April 15). R-squared: Definition, calculation formula, uses, and limitations. Investopedia. Retrieved April 26, 2023, from https://www.investopedia.com/terms/r/r-squared

[11] Yadav Vikash, et al. "A New Approach for Movie Recommender System Using K-Means Clustering and PCA." Journal of Scientific and Industrial Research (JSIR), Nov. 2021.