Creation of Movie Recommender

Danielė Ačaitė, Evita Petkevičiūtė, Rūta Okulič-Kazarinaitė

Abstract

The World Wide Web continues to grow at a massive rate and all the sites with all the information and their complexity continues to grow along with it. Thus, it becomes time consuming and burdensome for people to find similar information themselves. To enhance the search for relevant things and make it easier to use it, websites can be personalized. This is where recommender systems become valuable because they can dynamically add hyperlinks which results in making it easier for users to find interesting things and in the meantime making the interaction between the system and the user better. These personalized recommender systems are used in various movie sites.

Movie recommendation systems are increasingly popular and widely used in the entertainment industry. They aim to predict the ratings of preferences a user would give to a movie, based on the user's past ratings and the ratings of other users. By recommending movies that are likely to be highly rated by a user, the system can help users discover new films to watch and improve their experience. However, it requires expensive computations in order to find the essential properties as the number of users as well as movies increase exponentially.

According to M.Sridevi and Dr .R.Rajeswara Rao [1], there are several approaches to building recommender systems, including Collaborative filtering, Content based filtering, Demographic filtering and Hybrid Recommenders system. Collaborative filtering is based on the idea of predicting users' ratings for a movie based on the ratings given by other users with similar tastes. Content-based filtering uses the characteristics of the movies themselves (for example: genre, director, actors) to recommend similar movies to users. Hybrid methods combine both collaborative filtering and content-based filtering to make recommendations.

In this research paper, we will explore the use of machine learning techniques to build a movie recommendation system based on I. Ahmed method [2], . We will discuss the challenges and limitations of building movie recommendation systems.

1 Project goals and Outcomes

Our objective is to create a movie recommendation system based on 4 types of recommender systems which is acknowledged by literature review.

The principal steps of our study are: 1)Reviewing the types of movie recommender systems in literature 2)Finding the dataset required for the project 3)Performing the modeling 4)Report results and explain all the steps of the modeling process thoroughly

Completing this project will require familiarising with various types of movie recommenders: Collaborative filtering, Content based filtering, Demographic filtering and Hybrid Recommenders system. Gaining practice and expertise in distinguishing these types and learn reproducible experimental design.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

2 Literature Research

2.1 COLLABORATIVE FILTERING

Collaborative filtering is a method of making recommendations for a user by collecting preferences or taste information from many other users. This information is then used to identify patterns and suggest items that are likely to be of interest to the user. In the case of a movie recommender system, collaborative filtering involves collecting ratings or reviews from many users and using those ratings to suggest movies to a particular user. The assumption behind collaborative filtering is that if a user A has similar preferences to a user B, and B likes a particular movie, then A is more likely to like that movie as well. M.Sridevi and Dr .R.Rajeswara Rao address 3 main issues of this type of movie recommender system [3]:

- Most users do not rate the movies.
- The recommender fails to recommend a movie when new users come to the environment since they do not rate movies and the system lacks data.
- As the number of users increases exponentially it becomes expensive and computationally hard to compute an accurate system.

2.2 CONTENT-BASED RECOMMENDER SYSTEM

Content-based filtering systems choose items based on the correlation between items' content and the user's preferences as opposed to the collaborative filtering system meaning that they suggest items based on a particular topic [4]. Recommendations are made by comparing a profile with the matter of each file in the correlation where there are a lot of wordings. This system has item metadata where there is information about the director, genre, description, actors, etc for films to to create these recommendations. Some words in the plot description like tags and stop words are removed because they occur pretty often and cannot be used as discriminators. The remaining usually are minimized by removing prefixes and suffixes. In the end, content-based recommender systems look for items that a person has liked and then search for similar ones. Indicating that the movies who were the most liked will likely be similar and recommended.

Steps that we taken:

- Overview columns indicate the plot's description, therefore, a conversion into matrix by computing TF-IDF vectors was needed.
- The cosine similarity was used to denote the similarity between two different movies.
- Define recommendation system.

2.3 DEMOGRAPHIC FILTERING

Demographic filtering is a method of making recommendations for a user based on demographic information about the user. The demographic properties may be things like the user's age, gender, location, or occupation. This information can be used to suggest movies that are likely to be of interest to the user based on the preferences or characteristics of people with similar demographics. For example, if a recommender system knows that a user is a 20-year-old woman, it might recommend romantic comedies or dramas, which are known to be popular among young women. It is more advantageous than collaborative filtering since it addresses the issue of new users with limited data and is easier to implement. If new users have not rated any movies the system can recommend movies based on one's demographic information. On the other hand, it lacks uniqueness. For instance, let's say there is a new young male user and the system recommends action movies as it is the most popular among males. However, it does not capture that this male user might not enjoy action movies since he is more of a romantic comedies fan.

2.4 HYBRID FILTERING

Hybrid filtering is a technique used in movie recommendation systems that combines the properties of collaborative filtering and content-based filtering. Collaborative filtering relies on the preferences of similar users to make recommendations, while content-based filtering uses the characteristics of the movie itself to make recommendations. By combining these two approaches, hybrid filtering is able to make more accurate and personalized recommendations. For example, a hybrid system might consider a user's past ratings of comedies, as well as the genre and actors of the movie being recommended, to provide a more tailored suggestion. Hybrid filtering can help to improve the effectiveness of movie recommendation systems by combining the two methods.

3 Methodology

3.1 Datasets

All provided datasets were extracted from kaggle [5]. The features of the datasets were displayed in the coding environment.

4 Demographic filtering modification

This paper examines the demographic filtering methodology based on the TMDb (The Movie Database) website. This is one of the major databases about movies and television shows. TMDb allows users to access metadata for various tv shows, movies. There users are able to find the status of the movies, get a list of trending films and get the features. One of the simplest metrics that could be used is movie rating. However, this method is not as advantageous since, firstly, it does not take into account the popularity of the movie. If there is a movie with rating of 9 by 50 000 users will have a lower place than a movie which was rated 9.5 by only three users, hence, the list would not be reliable. Therefore, one of the main metrics that are used when building a movie recommender is the weighted rate used in the TMDb [6].

WR = (v/(v+m))*R + (m/(v+m))*C)

- v - number of votes for the movie -> already given
- m - minimum number of votes that keep movie in the list (we take 80
- R - mean rating of the movie –> already given
- C - mean rating of all the movies in the list (calculated in the code, mean is 6.092)

Steps:

- filtering of movies that qualify for the list $\rightarrow$ 96
- define function of tmdb weighted rating and create new feature score and calculate the value by applying a function to our DataFrame of qualified movies
- sorting movies based on score:

These are the results of basic Demographic Filtering Movie Recommender. The table of 10 movies displays the list of the movies which can be treated as "Trending now" movies. That proves that demographic filtering does not take into account the individual preferences as tastes of a particular user.

5 CONTENT-BASED FILTERING

This recommendation system is based on the pairwise similarity index. The index is used according to the plot descriptions and movies are recommended based on the similarity score. Each word vector needs to be converted by computing TF-IDF vectors. This provides a matrix whose each column is a word in the overview vocabulary and each row shows the original movie name. Multiplied TF-IDF is used to minimize the importance of repeating words that are found in overview. After using the scikit-learn function it showed that there are more than 20 000 different words that describe more than 4800 films in the given dataset. Cosine similarity was used to calculate a numeric quantity that denotes the likeness between two different movies. This index is more independent of magnitude and is in comparison not difficult and faster to calculate. In order the system would recommend, the

	title	vote_count	score
1881	The Shawshank Redemption	8205	8.248353
662	Fight Club	9413	8.096134
3337	The Godfather	5893	8.077404
3232	Pulp Fiction	8428	8.074738
65	The Dark Knight	12002	8.044250
809	Forrest Gump	7927	7.972814
96	Inception	13752	7.969290
95	Interstellar	10867	7.937399
1990	The Empire Strikes Back	5879	7.904757
1818	Schindler's List	4329	7.900080

Figure 1: List representing movies based on weighted rating

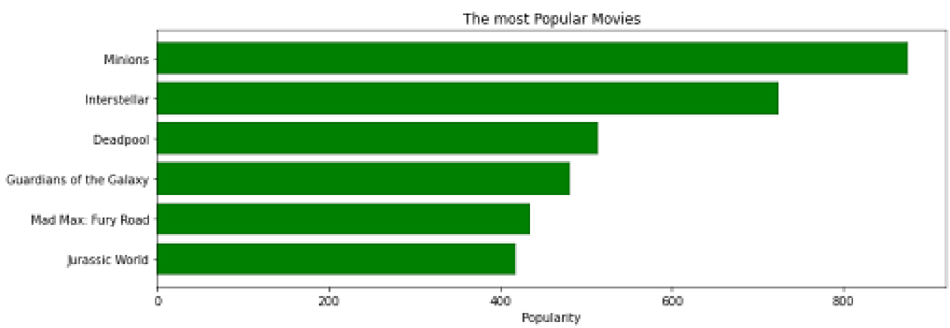

Figure 2: List representing movies based on popularity

output is set to a movies name and the input is set to the movies names and their description of other liked 10 similar movies. Later for this, a reverse napping of movie titles and DataFrame indices are needed.

```
get_recommendations("Grown Ups")

4420                   Black Rock
1816               The Best of Me
1057                 Coach Carter
3382              The Mighty Macs
3478                      College
4541           The Slaughter Rule
2562    When the Game Stands Tall
507             Independence Day
2228            An American Carol
3900                      Air Bud
Name: title, dtype: object
```

Figure 3: Recommendations made using TF-IDF

However, the given output quality can be improved because the recommendation system does not take into account the cast, crew, keywords and genres. After converting the list into a safe and usable structure, the functions are written to help to extract the needed information from each feature.

Later all the words are put into lowercases and the spaces are moved so that the system would calculate the name and surnames as one word. The string “metadata soup” is created where all the metadata is because it is needed to give to the vectorizer. Now the countvectorizer() is used instead of TF-IDF because there is no need to down-weight the presence of actors or directors. Finally, the recommendation system outputs similar movies.

```
] get_recommendations('Grown Ups', cosine_sim2)

436                          Grown Ups 2
499                        Jack and Jill
2989                       Happy Gilmore
361      You Don't Mess with the Zohan
445                      Just Go with It
1539                           Big Daddy
796                  The Ridiculous 6
802                        That's My Boy
1126                  Here Comes the Boom
1392                  The Benchwarmers
Name: title, dtype: object
```

Figure 4: Recommendations made using countvectorizer()

6 COLLABORATIVE FILTERING

To provide an even more personalized recommendation system, it is worthy to implement collaborative filtering. In this case, we are looking at specific persons (userID) ratings for certain movies, and try to predict another person’s rating for the same movie. That can be used to recommend movies to people in the same interest groups.

To implement this model, we used Surprise Python Scikit due to its specific recommendation functions. We use a 5 fold cross-validation, to predict new data, together with RMSE and MAE to predict the quality of the predictions - we aim for a low RMSE for more precise results. We get the RSME equal to 0.89684716, which is decent. Then we train the model:

```
trainset = data.build_full_trainset()
svd.fit(trainset)
```

Figure 5: Training the data set

Now to check the results we pick a random user based on their id, and predict the rating for the movie (also based on the id), in this case: Here, we predict what user (10) would rate the movie (20). SVD -

```
[108] # predicting users rating for a movie
      svd.predict(10, 20, 3)

      Prediction(uid=10, iid=20, r_ui=3, est=3.0636579409030085, details={'was_impossible': False})
```

Figure 6: Predicting the ratings

Singular Value Decomposition algorithm is used to model the user and item biases from users and items, which uses stochastic gradient descent to optimize the parameters.

7 Summary

Current trends of movie recommender systems show that there is huge potential for them to improve to even more precise level. Our project summarizes and explains how the recommend systems work with certain data and models. Digging deeper into recommender system issues, there are few main ones to be named: one of them is a slow start for new users - their experience will not be pleasant and they will not receive personalized recommendations until they have watched enough movies. Another issue is that some recommender systems are vulnerable to feedback loops, when recommendations made by the system influence the data that is used to generate future recommendations. Another problem is that if the data that the models were trained on were biased, so will be the recommendations, and the model will not understand that. This and many other issues do not have a quick or easy fix - it requires attention and ethical decisions. To start with, the data set could be including metadata, diverse (including different genders, age groups, races etc.), and updated often [7]. Besides these issues, movie recommender systems can benefit not only the business owner by increasing profits and user spent time on the platform, but it can benefit the user as well by saving them time and helping to find good fitting movies faster.

References

[1] M. Sridevi. Table 1 from decors : A simple and efficient demographic collaborative recommender system for movie recommendation: Semantic scholar, Jan 1970.

[2] Ibtesama. Getting started with a movie recommendation system, May 2020.

[3] Iateilang Ryngksai and L Chameikho. Recommender systems: types of filtering techniques. *International Journal of Engineering Researck & Technology, Gujarat*, 3(2278-0181):251–254, 2014.

[4]

[5] Rounak Banik. The movies dataset, Nov 2017.

[6]

[7] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. Bias and debias in recommender system: A survey and future directions. *arXiv preprint arXiv:2010.03240*, 2020.