Movie Recommender System

Project Report

1. K. Pravallika

17wh1a0525@bvrithyderabad.edu.in

2. B. Praneetha

17wh1a0551@bvrithyderabad.edu.in

3. S. Samiksha

17wh1a0516@bvrithyderabad.edu.in

ABSTRACT:

Recommender systems have become ubiquitous in our lives. Yet,currently, they are far from optimal. In this project, we attempt to understand recommendation systems using **MovieLens** dataset.

1.INTRODUCTION

A recommendation system is a type of information filtering system which attempts to predict the preferences of a user, and make suggests based on these preferences. There are a wide variety of applications for recommendation systems. These have become increasingly popular over the last few years and are now utilized in most online platforms that we use.

The content of such platforms varies from movies, music, books and videos, to friends and stories on social media platforms, to products on e-commerce websites, to people on professional and dating websites, to search results returned on Google.

Often, these systems are able to collect information about a users choices, and can use this information to improve their suggestions in the future.

Due to the advances in recommender systems, users constantly expect good recommendations. They have a low threshold for services that are not able to make appropriate suggestions. If a music streaming app is not able to predict and play music that the user likes, then the user will simply stop using it. This has led to a high emphasis by tech companies on improving their recommendation systems. However, the problem is more complex than it seems. Every user has different preferences and likes. In addition, even the taste of a single user can vary depending on a large number of factors, such as mood, season, or type of activity the user is

doing. For example, the type of music one would like to hear while exercising differs greatly from the type of music he'd listen to when cooking dinner. Another issue that recommendation systems have to solve is the exploration vs exploitation problem. They must explore new domains to discover more about the user, while still making the most of what is already known about of the user. Two main approaches are widely used for recommender systems. One is content-based filtering, where we try to profile the users interests using information collected, and recommend items based on that profile. The other is collaborative filtering, where we try to group similar users together and use information about the group to make recommendations to the user.

2. PROBLEM STATEMENT

This Project on Movie Data Analysis and Recommendation Systems is an interesting one.

In this project, you have to attempt at implementing a few recommendation algorithms (content based and collaborative filtering) and try to build an ensemble of these models to come up with our final recommendation system. With us, we have two MovieLens datasets.

--> **The Full Dataset:** Consists of 26,000,000 ratings and 750,000 tag applications applied to 45,000 movies by 270,000 users. Includes tag genome data with 12 million relevance scores across 1,100 tags.

--> **The Small Dataset:** Comprises of 100,000 ratings and 1,300 tag applications applied to 9,000 movies by 700 users.

3. ALGORITHMS

For our project, we focused on two main algorithms for recommendations: Collaborative filtering & Content-based filtering.

3.1 Collaborative Filtering

Collaborative Filtering techniques make recommendations for a user based on ratings and preferences data of many users. The main underlying idea is that if two users have both liked certain common items, then the items that one user has liked that the other user has not yet tried can be recommended to him. We see collaborative filtering techniques in action on various Internet platforms such as Amazon.com, Netflix, Facebook. We are recommended items based on the ratings and purchase data that these platforms collect from their user base.

3.2 Content Based Recommendations

Content Based Recommendation algorithm takes into account the likes and dislikes of the user and generates a User Profile. For generating a user profile, we take into account the item profiles(vector describing an item) and their corresponding user rating. The user profile is the weighted sum of the item profiles with weights being the ratings user rated. Once the user profile is generated, we calculate the similarity of the user profile with all the items in the dataset, which is calculated using cosine similarity between the user profile and item profile.

Advantages of Content Based approach is that data of other users is not required and the recommender engine can recommend new items which are not rated currently, but the recommender algorithm doesn't recommend the items outside the category of items the user has rated.

Single Value Decomposition

One way to handle the scalability and sparsity issue created by CF is to leverage a latent factor model to capture the similarity between users and items. Essentially, we want to turn the recommendation problem into an optimization problem. We can view it as how good we are in predicting the rating for items given a user. One common metric is Root Mean Square Error (RMSE). The lower the RMSE, the better the performance.

Now talking about latent factor you might be wondering what is it ?It is a broad idea which describes a property or concept that a user or an item have. For instance, for music, latent factor can refer to the genre that the music belongs to. SVD decreases the dimension of the utility matrix by extracting its latent factors. Essentially, we map each user and each item into a latent space with dimension r. Therefore, it helps us better understand the relationship between users and items as they become directly comparable. The below figure illustrates this idea.

4. DATASET

Context: These files contain metadata for all 45,000 movies listed in the full MovieLens Dataset. The dataset consists of movies released on or brfore July 2017. Data points include cast, crew, plot keywords, budget, revenue, posters, release dates, languages, production companies, countries, TMDB vote counts and vote averages.

This dataset also has files containing 26million ratings from 270,000 users for all 45,000 movies. Ratings are on a scale of 1-5 and have been obtained from the official GroupLens website.

Content: This dataset consists of the following files:

movies_metadata.csv: The main Movies Metadata file. Containd information on 45,000 movies featured in the Full MovieLens dataset. Features include posters, backdrops, budget, revenue, release dates, languages, production countries and companies.

keywords.csv: Contains the movie plot keywords for our MovieLens movies. Availble in the form of a stringified JSON Object.

credits.csv: Consists of Cast and Crew information for all our movies. Available in the form of a stringified JSON Object.

links.csv: The file that contains the TMDB and IMDB IDs of all the movies featured in the Full MovieLens dataset.

links_small.csv: The file that contains the TMDB and IMDB IDs of 9,000 movies featured in the Full MovieLens dataset.

ratings_small.csv: The subset of 100,000 ratings from 700 users on 9,000 movies.

5. ISSUES FACED

5.1 Scalability Issues

One of the major challenges in working with the 26 million dataset is memory constraints.

6. POSSIBLE FUTURE WORK

There are plenty of way to expand on the work done in this project. Firstly, the content based method can be expanded to include more criteria to help categorize the movies. The most obvious ideas is to add features to suggest movies with common actors, directors or writers. In addition, movies released within the same time period could also receive a boost in likelihood for recommendation. Similarly, the movies total gross could be used to identify a users taste in terms of whether he/she prefers large release blockbusters, or smaller indie films. However, the above ideas may lead to overfitting, given that a users taste can be highly varied, and we only have a guarantee that 20 movies (less than 0.2%) have been reviewed by the user.

7.CONCLUSION

A hybrid approach is taken between context based filtering and collaborative filtering to implement the system. This approach overcomes drawbacks of each individual algorithm and improves the performance of the system. we can work on hybrid recommender using clustering and similarity for better performance. Our approach can be further extended to other domains to recommend songs, video, venue, news, books, tourism and e-commerce sites, etc.

Git link to our project:

https://github.com/praneetha-bhupathiraju/Movie-recommendation-system