Movie Recommender Report

Peter Waysocher, Max Jakobitsch, Alexander Widmann

June 2020

1 Introduction

The goal is to create a system that recommends movies based on different strategies. We use Django and Bootstrap for the web interface, where the user can choose a movie and receives recommendations based on the chosen movie algorithm. The methods use Pandas to efficiently process data and Sklearn for text based approaches. The TMDb API provides us with poster-URLs for the movies. The project is designed to be easily adaptable, so we can add or adjust strategies without problems.

2 Methods

This is an overview of the recommendation strategies that made it into our evaluation phase. For implementation details consult the project's *README.md* file.

2.1 User-Based

We evaluated two user-based strategies. They first collect ratings by users that have rated the base movie, then find the top-rated movies in this reduced set of ratings.

2.1.1 Filter out below-average ratings

This method only includes users that have rated the base movie favourably (above their own average rating), using only positive propagation to generate the recommendations.

2.1.2 Popularity-based

Like the previous method, but also applies a popularity bias, thereby preventing lesser known and niche movies to appear too often.

2.2 Contend-Based and Hybrids

2.2.1 Genre-based

The genre-based method iterates through all movies and calculates the number of genres that both have in common. Afterwards the system recommends the five highest ones. Furthermore, this method has the option to includes the ratings and popularity of the movies. Therefore, the average rating, popularity and the similar genres are used to calculate the score.

2.2.2 Keywords only

A simple method that only compares keywords. This (in combination with genres) seems to be the approach used in creating the recommendations shown on TMDb.

2.2.3 Meta-mix

This method works similar to the genre-based method. Additionally to the matching genres, the numbers of matching actors, directors, keywords and production countries are calculated. These numbers are divided through the highest occurring number and multiplied with a multiplier based on importance. Furthermore, the year difference is calculated and inversed.

Component	Multiplier
Genres	35%
Actors	20%
Directors	8%
Keywords	10%
Production countries	7%

After the calculations each movie has a score between 0 and 100 and the five highest ones will be recommended. This method has an option for popularity too and works similar to the genre-based one.

2.2.4 First direct sequel + co

This method reserves the first recommendation "slot" for direct sequels. If there is no direct sequel, the first slot will be reassigned to the second part. The other slots follow a similar strategy as the *Meta-mix*, combining user-based similarity, common actors, common directors and genre similarity with weights 2, 1, 1 and 3 respectively.

2.2.5 Mixed Algorithms

The principle is the same as with the Meta-mix, but this approach is focused on creating an extendable, adaptable version of it.There is an adapter method, that can use multiple different recommender systems, assigning them a weight

and then calculating the score accordingly. The "recommender systems" used for this approach are all column based. That means there is an algorithm for every column of data used. The implementation of those algorithms is fit to the specific columns. For example the keywords similiarity is based on machine learning (2.2.2) and the year is based on a simple arithmethic distance.

For the evaluation we used every column (Keywords, summary, genre, actor, director, year). Keywords were assigned the highest weight, to create a different algorithm than the Meta-mix.

2.2.6 Mixed Algorithms with popularity bias

This step basically only adds a consideration for the popularity to the mixed algorithm.

The default formula for the final score is: score * popularity

In the implementation the formula is: score ^{x} * popularity, where x is changeable, to keep the algorithm adaptable.

2.3 Evaluation

For the evaluation, we used 22 different movies such as Star Wars, Toy Story or Pirates of the Caribbean. We selected these movies with the goal of being as diverse as possible with regards to genres, popularity and overall setting.

We score the recommendations between 1 and 10 (where 10 is the best score) based on title, cover, summary, genres, setting and a general feeling that the movie is a fitting recommendation (Would I recommend this movie to friends, knowing that they like the base movie?). We had to choose many factors because no one of us is an expert on this topic, so the scoring is amateurish and should be taken with a grain of salt. Furthermore, we gave prequels and sequels less score if there are many. Each member of the team separately scored each recommendation for each movie for each method, then we calculated the average for each method.

The full scoring and calculations are included as Excel files in /docs/results.

2.3.1 User-based

We tried to evaluate and score both user-based methods like the others but after some iterations, we gave up on scoring them. The problem is the recommended movies from the "filtered-below-average" are very random. They don't really share similarities with the base movie and are solely chosen based on their good ratings. Furthermore, the results were mostly unpopular movies that none of us had ever heard of. On the other hand, the popularity based method too often recommends the same few movies (Shawshank Redemption, The Godfather, Fargo, The Usual Suspects, Pulp Fiction, Star Wars, ...) which are popular and have high ratings.

2.3.2 Content-based and Hybrids

Rank	Method	Score
1	Meta-mix	6.74
2	Genre-based	6.17
3	Mixed Algorithms	6.16
4	First direct sequel + co	6.14
5	Mixed Algorithm with popularity bias	6.06
6	Keywords only	5.41

Meta-mix is the clear winner in our evaluation, but the second place is interesting too. Similar genres play a more important role for good recommendations than keywords. This is probably because we are not experts and decides the score more based on matching genres than the rest. But only looking at the genres don't give the best results. It is best to choose a tactic that combines many factors, but genres play a huge role. "Keywords only" scores the lowest. This is due to our knowledge of this subject. Keywords will probably play a higher role if you know the movies.

2.3.3 Additional observations

Anne Frank is a biography of her life. It has the genres "Drama, Foreign, Documentary". Interesting is that the methods using genres present more documentaries than the methods which are based on keywords.

A problem was the movies which have a big universe for example "Marvel universe" or "Lord of the Rings". The recommendations often were movies from the universe which are no prequel or sequel and matched pretty well with the base movie. These movies scores are higher than other movies.

Some methods have trouble making recommendations for less popular movies. The reason is that methods that uses popularity bias struggle to provide good recommendations and that the results are often unpopular movies which we could not score probably.

A funny observation is that the methods based on keywords sometimes recommend movies which do not match well. For example, the method "Mixed Algorithm" recommends the horror movie "Deathship" for "Titanic". This result is probably archived by the keywords "ship" or "shipwreck".

Animations or Children movies have the highest score. It was easy to evaluate and it got a higher score because of it.