Group 5 - Philipp Markopulos, Nikolaus Czernin, Luka Corsovic

Final Report

Abstract:

In today's digital and interconnected world, where billions of users visit media-sharing and streaming platforms each day to consume an endless variety of media content, making accurate predictions on users' future behavior has become an utmost important task for media companies. By offering more relevant content to its users, platforms can increase their user engagement and thus revenue significantly.
Basic recommendation systems mostly use content-based filtering as well as item-based collaborative filtering, which have long dominated the recommender system market. However, both approaches are not perfect and suffer from several trade-offs as well as their own individual limitations. The most significant drawbacks include sparse matrices and the cold-start problem, which can severely impact the quality of recommendations and accuracy.

Therefore, we propose a new Hybrid-Recommender system, thus combining the two approaches in a sophisticated manner to overcome their individual drawbacks and enable us to make better recommendations, thereby reducing the MAE, increasing predictive accuracy as well as precision.

Introduction:

In this paper, we will be comparing the accuracy of collaborative and content-based filtering techniques to a new hybrid Recommender system, which will incorporate features of both item-based collaborative filtering and content-based filtering algorithm. We will hereby focus on one of the most popular/known use-cases of recommender systems: streaming services, particularly movie-recommendation systems for file-sharing platforms.

Given a very large set of objects/digital media in such platforms, movie-recommendation systems increase user comfort by suggesting to them a small subset of items derived from their personal interests and use history on the platform.

Content-based filtering uses item features to recommend other items like what the user likes, based on their previous actions or explicit feedback. In our case, the recommendations are based on direct feedback from active-user ratings. We would be using keywords in our data to compare similarities between the items/movies. A significant issue concerning the exclusive usage of content-based filtering for our model is the cold start problem. Thus, the system can hardly handle fresh/new items.

Since the feature representation of the items is hand-engineered to some extent, this technique requires a lot of domain knowledge. Therefore, the model can only be as good as the hand-engineered features.
The model can only make recommendations based on the existing interests of the user. In other words, the model has limited ability to expand on the users' existing interests.

In contrast, item-based collaborative filtering is a form of collaborative filtering for recommender systems based on the similarity between items calculated using people's ratings of those items. In this case, we would be using user ratings to establish similarities between items.

In item-based collaborative filtering, the prediction of the model for a given (user, item) pair is thus the dot product of the corresponding embeddings. So, if an item is not seen during training, the system can't create an embedding for it and can't query the model with this item. This significant issue is often called the cold-start problem.

The hybrid recommendation system works by using the content-based filtering initially to fill in the empty reviews that are left by users. Through this, it aims to minimize the effect of the issue of missing ratings, as users, in general, do not rate a lot of movies, but through the content-based filtering approach, we can predict how they would rate the "missing movies" based on the keyword similarity of those movies, to movies which the user has already rated. In the next step, the enhanced user ratings would be run through an item-based collaborative filtering algorithm to come up with recommendations for the user. Of course, an important step in this process is to make sure that the user does not get a recommendation to a movie that has already been seen, hence the suggested movie must come from the movies where content-based filtering was used to predict the rating.

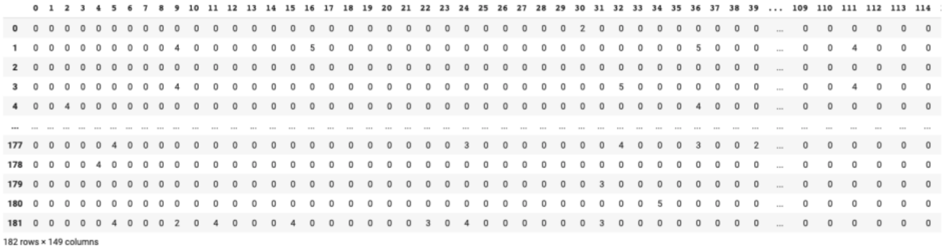
Implementation

Dataset

We are testing both the traditional content-based and item-based collaborative filtering algorithms as well as our proposed Hybrid Recommendation system on a movies-data set from the data science online platform kaggle.com. The dataset is a static subset from the full MovieLens Dataset offered by the research lab group sense from the University of Minnesota, offering over 27 million ratings by 280.000 users for a total of 58.000 movies. The dataset has last been updated in September 2018 by GroupLens and the subset in July 2017. Furthermore, the movie details, credits, and keywords have been derived from the TMDB Open API and were added to the Kaggle dataset.

The dataset is structured into seven different-separate csv. files, which include credits, identifiable keywords, links, cased-crew data (e.g., budget), and ratings of individual movies. While all this information can be integrated into a streaming platform, we will mainly user ratings as well as identifiable keywords for our analysis and algorithms. Keywords are "strings," which are descriptions of either the content or background of a movie (e.g., based on a novel, jealousy...). These will help us to establish a comparison and find similarities between items concerning item-based filtering. The ratings, which are of the type "integer," were derived from a diverse group of individual users. We will be using those for the content-based-filtering approach and to establish similarities between movies by considering the ratings of individual users.

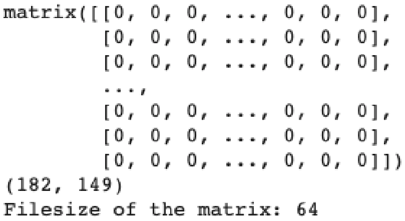
The user-item-ratings-matrix and the movie-keyword-matrix included in the dataset were very large and full of missing values, they would have taken up a long time and RAM to work with. Instead of using those dense matrices full of missing values, we used sparse *NumPy* matrices as represented in Figure 2, which used a lot less space and drastically decreased the computation time.



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	...	109	110	111	112	113	114
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	...	0	0	4	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	...	0	0	4	0	0	0
4	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	...	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
177	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	4	0	0	0	3	0	0	2	...	0	0	0	0	0	0
178	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
179	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
180	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	...	0	0	0	0	0	0
181	0	0	0	0	0	4	0	0	0	2	0	4	0	0	0	4	0	0	0	0	0	0	3	0	4	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0

182 rows × 149 columns

Figure 1 Dense User Item Ratings Matrix

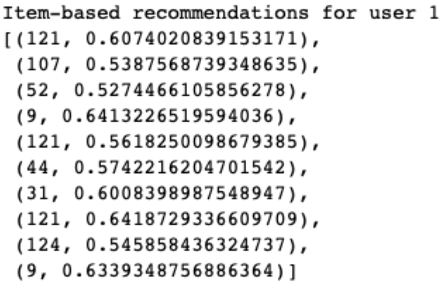


```
matrix([[0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        ...,
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0]])
(182, 149)
Filesize of the matrix: 64
```

Figure 2 Sparse Ratings Matrix

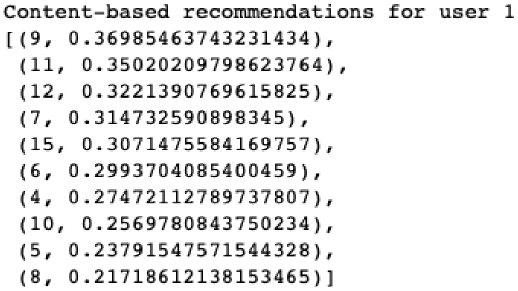
Results and Discussion

Figure 3 shows exemplary results of the content-based recommendation algorithm, and Figure 4 shows the results of the item-based recommendation algorithm. For the item-based filtering approach, the order is dependent on the ratings of the already liked movies, as well as the similarity of the other movies to the highly rated ones. Thus, the recommendations are not ordered by similarity. The content-based recommendations are ordered by their keyword-profiles' similarity to the user-keyword-profile, which is accumulated from the keyword-profiles of the movies the user liked.



```
Item-based recommendations for user 1
[(121, 0.6074020839153171),
 (107, 0.5387568739348635),
 (52, 0.5274466105856278),
 (9, 0.6413226519594036),
 (121, 0.5618250098679385),
 (44, 0.5742216204701542),
 (31, 0.6008398987548947),
 (121, 0.6418729336609709),
 (124, 0.545858436324737),
 (9, 0.6339348756886364)]
```

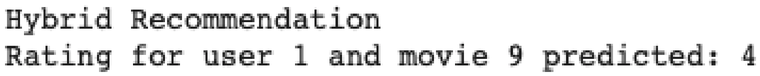
Figure 3 Item Based Recommendation Results



```
Content-based recommendations for user 1
[(9, 0.36985463743231434),
 (11, 0.35020209798623764),
 (12, 0.3221390769615825),
 (7, 0.314732590898345),
 (15, 0.3071475584169757),
 (6, 0.2993704085400459),
 (4, 0.27472112789737807),
 (10, 0.2569780843750234),
 (5, 0.23791547571544328),
 (8, 0.21718612138153465)]
```

Figure 4 Conent Based Results

We combined these two approaches to create the desired Hybrid Recommendation System, whose output we can see in Figure 5. This approach performs the content-based filtering method to predict a rating for every unrated movie in the user-movie-rating-matrix. This is done using the similarity of each rated movie to each unrated movie as weights, which are multiplied with the ratings for the rated movies and summed up (Formula 1). This way, the sparse user-item-rating-matrix is filled with pseudo-ratings. Then, for every pseudo-rating, the item-based collaborative filtering algorithm is applied to make a more informed prediction of the rating, while taking into account all other movies' pseudo-ratings. Again, this is done by using the similarities of rated and unrated movies are weights to multiply with the rated movies' ratings (Formula 2).



```
Hybrid Recommendation
Rating for user 1 and movie 9 predicted: 4
```

Figure 5 Hybrid Recommendation Result

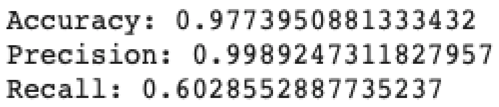
$$predictedRatingCB(u_x, i_y) = \Sigma_0^i sim(movie\_keyword[y,], movie\_keyword[, i]) * ratings[x, y]$$

Formula 1: Content-based filtering rating prediction

$$predictedRatingIBCF(u_x, i_y) = \Sigma_0^i sim(ratings[, y], ratings[, i]) * ratings[x, y]$$

Formula 2: Item-based collaborative filtering rating prediction

As we can see, the top recommendation of the content-based model, movie 9, seems to be the preferred choice of the hybrid recommendation. Movie 9 also appears in 4th place when looking at the item-based recommendations for the user.



```
Accuracy: 0.9773950881333432
Precision: 0.9989247311827957
Recall: 0.6028552887735237
```

Figure 6 Model Evaluation Metrics

Figure 6 shows us the accuracy, precision and recall of our hybrid recommendation system, based on a test data set that we created of about 500 users. The accuracy metric is quite high, being around 98 percent. The precision metric is even higher, lying around 99 percent. This tells that 99 percent of the ratings predicted by the model were relevant, i.e., that the values were present in the test dataset. However, the recall metric is only around 60 percent. This metric tells us that the model predicted about 60 percent of the relevant test ratings.

The implications of the model performance in the context of a movie recommender system are that the model's predictions are very accurate and that the predictions almost always hold true based on our test data. This would mean that the model can predict a user's affinity towards a certain movie and make very accurate recommendations to the users due to the precision being so high. However, the recall metric proves that there is room for improvement. It implies that there are around 40 percent more movies that the user will potentially enjoy that are not being recommended to him. This is a feature that should be considered in further implementations of the model.

When considering the practical implications of a movie dataset, we must consider that people usually do not like to waste their time watching movies they will not enjoy. Therefore, we believe that precision in the context of this model is more relevant than recall, as precision ensures that the predictions of the model will be in line with the user's preference.

Limitations of the approach

One of our greatest limitations of this model was computational power. Due to the size of our dataset and the processing power available to us, we could not perform the testing of the model on a larger dataset than that of 500 without our interface crashing. This could have potentially harmed our results, so we would test this model again with a bigger dataset on a machine with more computational power.

The approach does tackle most of the limitations of the content-based and the user-based recommender systems, however the first rater problem persists. Without any reviews left by the user, the system cannot perform any recommendations without some initial reviews.

Apart from that, perhaps the dataset we choose could have been more modelling friendly with less preprocessing being required, and perhaps another keyword being made available for us to do the content-based filtering analysis on.

Conclusion

In conclusion, we believe that this model shows promising results, however it should be put to the test more before its final use can begin. As mentioned above we were limited by computational power, and we would like to see how our model would perform on a bigger test dataset.

Based on the papers we analyzed that outline this method, the model does seem to perform better than a standard collaborative filtering approach. However, the difference is not very large (around 4%). The papers also outline that there are potentially also other ways to include the content and user-based approaches to arrive at the hybrid model, however that these should be further explored and compared to the model outlined in the paper. Clustering and similarity are methods that should be reviewed and potentially included in the model to increase performance. The model could also potentially be expanded to be applicable across a range of different products such as books, songs, tourism, e-commerce, and many others, however for the model to be compatible it would need to be adjusted to work with ratings in different industries.

Bibliography:

- Sharma, Poonam and Yadav, Lokesh Yadav, Movie Recommendation System Using Item Based Collaborative Filtering (2020). International Journal of Innovative Research in Computer Science & Technology (IJIRCST), ISSN: 2347-5552, Volume-8, Issue-4, July 2020
- Prem Melville, Raymod J. Mooney, and Ramadass Nagarajan. 2002. Content-boosted collaborative filtering for improved recommendations. In Eighteenth national conference on Artificial intelligence. American Association for Artificial Intelligence, USA, 187–192