Fairness in group recommendations

Allegra Strippoli*
all.strippoli@stud.uniroma3.it
Universita degli studi Roma Tre
Rome

ACM Reference Format:
Allegra Strippoli. 2024. Fairness in group recommendations.

Source code available on *GitHub*[1]

1 ABSTRACT

In this report, the approach for the fairness issue in group recommendation is described. Three scenarios are considered:

(1) Ten movies are recommended to an individual user.
(2) Ten movies are recommended to a group.
(3) A group receives sequential recommendations.

The goal is to design a recommendation mechanism that is fair, efficient, and has a high accuracy rate. All tests are repeatable, and random elements in the code have been removed.

2 INTRODUCTION

A recommender system is a specialized type of information filtering system designed to offer suggestions for items that are most relevant to a specific user. Recommender systems are particularly useful when an individual needs to choose an item from a potentially overwhelming number of items that a service may offer. These systems find application in various fields. Some well-known examples are the recommendation of movies, songs, and users on social media. Typically, recommender systems employ either collaborative filtering, content-based filtering, or a combination of both. In this report, a collaborative filtering approach is adopted, wherein a model is built based on a user's past interactions (previously rated items) and similar choices made by other users. This model forecasts items that the user might be interested in.

The fairness issue in group recommendation poses several challenges:

(1) When a user rates only a few items, it's challenging to provide accurate suggestions.
(2) Similarly, when an item has been rated by only a few users, it's difficult to determine when to recommend it.
(3) Moreover, it's desirable that, as the number of users or items increases, the query execution time remains relatively low.
(4) As for group recommendations, when a user's tastes do not align well with the rest of the group members, it's important to provide suggestions that do not exclude that user.
(5) Lastly, in sequential recommendations, if a user is not satisfied in the previous round, they should have the opportunity to be satisfied in the next round.

The solution presented in this paper addresses all the problems mentioned above. Although it may not be an optimal solution, the results are considered satisfactory for educational purposes.

*This report is for the Advanced Topics in Computer Science course at Roma Tre.
[1] https://github.com/allegrastrippoli/atcs.git

- When a user has not rated any items or, in a more general scenario, the system is unable to suggest movies that a user might like, the most popular items in the whole system are recommended.
- Movies that have not been rated by many users are taken into consideration and might be suggested to a user. This is because the analyzed movies (for which a predicted score is calculated) are chosen randomly. However, for the testing phase, to make the code deterministic, the movies are selected sequentially rather than randomly.
- To keep execution times quite low, thresholds are used, so as soon as a good candidate solution is found, it is immediately returned.
- To not exclude any user, an approach that minimizes disagreement among users is used. It aims to avoid situations where a user behaves differently from the rest of the group: when a movie is rated, it is preferable that all users, as uniformly as possible, either like or dislike it.
- Lastly, in sequential recommendations, the suggested movies are updated each round, according to the level of user satisfaction. Therefore, a 'dissatisfied' user has more influence than other users in the next round.

3 APPROACH

Ten movies are recommended to an individual user.

The first part of the project concerns the recommendation of ten movies to a user. To better visualize the flow of operations, the sequence diagram is shown.

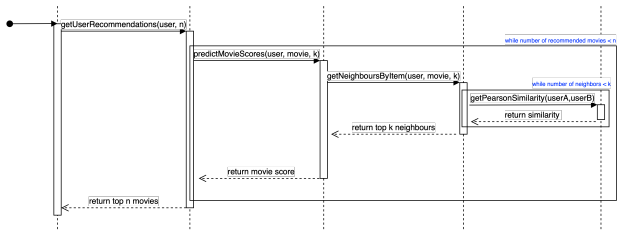

Figure 1: SSD

(1) Once a sufficient number of neighbors with a similarity score greater than a certain threshold have been found, the remaining users are not considered. Specifically, for each movie, 30 neighbors with a similarity score > 0.3 are selected. Usually, fewer than 30 neighbors are found, allowing for a relatively low threshold to be maintained;
(2) When predicting the rating for a movie, only neighbors who have rated that movie are taken into account. Unfortunately,

this constraint does not always yield accurate predictions, and a potential future development could involve implementing an algorithm to overcome this limitation;
(3) If the predicted score for a movie is higher than 4/5, then the movie may be among those recommended. (In exceptional cases, if the running time is too long for a user, movies with a score of 3/5 are also accepted).

Ten movies are recommended to a group.

For the second part, the recommendations do not concern a single user, but a group. The approach is:

(1) Select N users to form a group;
(2) Get the 10 best recommendations for each of them ($N * 10$ movies in total are collected);
(3) Obtain a score on all movies from all group members (through other predictions);
(4) Aggregate the results minimising disagreement among users; The disagreement related to a movie by a group g is: $dis(movie) = max((r_{u,movie} - \overline{r_{g-\{u\},movie}}) \; \forall \; u \; \in \; g)$

The top-i movie (for i ranging from 1 to 10) is the one that has the minimum $min(dis(movie))$ value among all movies. To avoid falling into the same problem as the least misery method (where movies that users don't like are suggested), all movies with an average group rating $\overline{r_{g,movie}} < 3$ are discarded by default. This constraint does not allow the system to operate if there is no movie with an average group rating greater than the threshold. If that happens, the most popular movies in the whole system are shown. The system behaves as shown in Table 1.

avg group rating	$dis(movie)$	system behaviour
> 3	$\simeq 0$	ideal candidate
> 3	$\simeq 5$	may be recommended
< 3	$[0, 5]$	discarded
< 3	$[0, 5]$	discarded

Table 1

If $dis(movie)$ is equal to 0, it means that all users gave the same score. When it's equal to 5, it means that one user rated 0/5 and the mean of the other users, except for him, is 5/5.

A group receives sequential recommendations.

For the third and final part, the recommendations are sequential. During a round:

(1) For each user in the group, calculate the ratio between ideal and current satisfaction to find the sat value ($sat = \frac{actualSat}{idealSat}$);
(2) Users are sorted by their sat value, and their ratings are weighted according to their position (pos) in the ranking (the user in position 0 is the most dissatisfied). The new score for the round is $r_{u,movie} = \frac{1}{1+pos_u} * r_{u,movie}$

4 EXPERIMENTS

Accuracy of the predicted score for a movie

To calculate the accuracy of the predicted score for a movie, the system was asked to predict values that were already known. A prediction is considered correct if it belongs to this interval: actual value - 1 ≤ predicted value ≤ actual value + 1. The predicted value is rounded to one decimal place.

user id	*accuracy*	*number of rated movies*
1	95%	232
19	88%	703
50	79%	309
122	99%	292
610	80%	1302
Overall	85%	2838

Table 2: accuracy score

All those cases in which the denominator of the prediction or the similarity functions is zero are not handled. This has a negative impact on the accuracy score.

Fairness in Group Recommendations

A group was created with 3 similar users and one dissimilar user. The three similar users have a Pearson similarity score of about 65% among them, and each pair has approximately 20 movies in common (the movies are not necessarily the same for all pairs). The dissimilar user has a Pearson similarity score with all other group members of about 3%, and he gave a score < 3 to 16 out of 30 movies proposed by the three similar users.

Three methods are tested: average, least misery and the one presented in the paper, which minimises disagreement. User satisfaction is again measured as $sat = \frac{actualSat}{idealSat}$.

Recommended movies for the user (without considering the group) have an average score shown in the table below:

user id	*avg ideal score*
62	4.4
1	4.4
494	4.3
3	3.6

Table 3

This information is important to understand the data not only in percentage terms, but also in absolute terms.

user id = 62	*sat*
MinDisagreement	98%
LeastMisery	100%
Average	100%

Table 4

user id = 3	*sat*
MinDisagreement	92%
LeastMisery	95%
Average	92%

Table 5

The three methods give similar results in terms of user satisfaction. This is probably because there is a sufficient number of movies that have a score > 3 for user 3. By removing some of these, the difference between user 3 and the other group members emerges.

user id = 62	*sat*
MinDisagreement	99%
LeastMisery	92%
Average	93%

Table 6

user id = 3	*sat*
MinDisagreement	57%
LeastMisery	55%
Average	57%

Table 7

In this scenario, where it is more difficult for the system to provide acceptable recommendations, the *MinDisagreement* method makes users more satisfied. (Although there remains a gap between similar and dissimilar user satisfaction).

Satisfaction in Sequential Recommendations

For this part of testing, ten movies are chosen that are "fairly" liked by the similar users (however, they are not the optimal choices), and that the dissimilar user dislikes. The IDs of the movies are shown:

- Round 1: 21, 2, 11, 26, 70, 3, 15, 7, 99, 13

The sequence recommendation algorithm improves the satisfaction of all users up to convergence. Moreover, the most important result is that fairness is also guaranteed for users, as user 3 reduces its satisfaction gap in round 2 and 3.

- Round 2: 28, 85, 50, 69, 47, 16, 32, 29, 25, 6
- Round 3: 28, 85, 50, 69, 47, 32, 16, 6, 70, 25

user id = 62	*sat*
round 1	82%
round 2	100%
round 3	100%

Table 8

user id = 3	*sat*
round 1	39%
round 2	92%
round 3	94%

Table 9

5 CONCLUSION

In this study, the fairness issue in group recommendations through a collaborative filtering approach is addressed. Three scenarios are considered: individual recommendations, group recommendations, and sequential recommendations. The approach used tackled several challenges including providing accurate suggestions for users with limited ratings, determining the timing for recommending items with few ratings, maintaining low query execution times as the number of users or items increases and ensuring fairness to all users.

The experiments demonstrated promising results in terms of prediction accuracy and user satisfaction. The average accuracy score for predicting movie ratings was approximately 85%, with certain users achieving high accuracy rates. In group recommendations, the method presented in this paper, which minimizes disagreement among users, showed comparable results to other methods such as average and least misery. Indeed, although the algorithm performs well, it does not completely solve the fairness problem. In sequential recommendation, the most disadvantaged user improved his satisfaction from 39% to 94%.

There are limitations and potential areas for future work. For instance, improving the prediction accuracy by considering more advanced algorithms could enhance the overall performance of the system.

Movie predictions require different latencies; however, it would be interesting to maintain a similar time for each user.

Lastly, it would be interesting to expand the tests on multiple groups and more sequential recommendations to have a complete overview of the entire dataset.

6 BIBLIOGRAPHY

All references come from material provided by the professor during lectures.