UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

Mitigate Bias Due To The Popularity Effect In Recommendation Algorithms

Master in Innovation and Research in Informatics

Specialization: Data Science

Thesis supervisor: Ludovico Boratto, EURECAT
Tutor: Oscar Romero, Department of ESSI, UPC
Author: Joan Prat Sicart

Barcelona, January 28th, 2021

Acknowledgments

First of all, I would like to express my gratitude to my research supervisors: Ludovico Boratto, Mirko Marras, Guilherme Ramos and Oscar Romero from the UPC. I want to thank them for their patient guidance, supervision and advice throughout the development of this master's thesis. It was my privilege to work with them.

Second, to all the professors and the Stack Overflow community that provided me the necessary knowledge to overcome the difficulties encountered during this Data Science master's degree. Also, to my classmates that made this experience much more enriching from a personal point of view. I wish them all the best in life.

Third, to Roger, Núria and my life-long friends for always supporting me and making my life much more cheerful and funny.

And finally, to my parents, Emma and Joan for supporting and providing me the best education they could afford.

Abstract

The objective of group recommendations is to provide a ranked set of items, a Top-N$_G$, that contains relevant items for every member of the group. A common approach to achieve this consists on selecting items the group can agree to, by means of so called "group modeling strategies". However, most of these group modeling strategies select items independently of the previous selected items. Therefore, they can not guarantee properties such as fairness. Understanding fair recommendations means providing an equitable number of relevant items and in similar ranks for each member. Apart from that, there exist also group modeling strategies that try to provide fair recommendations by penalizing or simply dismissing the opinion of those members that obtained a significant item in the previous recommendation, and consequently considering just the happiness of a subset of users instead of the overall group happiness.

In this thesis, we introduce a novel definition of fairness that balances the preferences of the members without penalizing or dismissing their opinion, just simply making it more tolerant or more radical according to "how favorable were the previous recommendations for him in comparison with the others". In this thesis we also mathematically formalize this notion and we provide an algorithm, ARM (Adaptive Relevance-based Modeling), for finding the Top-N$_G$ set of group recommendations that satisfy our definition.

We benchmark the performance of ARM against 5 of the most widely known group modeling strategies in the literature. Also, we evaluate the performance of all of them for 12 configurations (one dataset, three different group types, and four different metrics to evaluate the group modeling performance). At the end, we find out that ARM performs better in 10 out of 12 scenarios, and in only two of them it generates a recommendations slightly worse than other strategies, providing in this way a significant improvement to the literature.
We attribute ARM's success to its novel criteria and its new approach on balancing the user preferences. Most of the current methods do not select items in accordance with the precedent selected items in Top-N$_G$ when generating a new recommendation, and the ones that do, provide recommendations that do not consider always all the user preferences.

Contents

1. Introduction

Recommender systems are usually employed to suggest items to individual users based in their preferences. However, in some occasions, group recommendations are required, e.g, friends in a car deciding which song to listen to, a family deciding a movie to watch, etc. Thus, nowadays the recommender system should satisfy not just one individual but groups of individuals by providing relevant items for each one of them. This might include diverse or even conflicting choices. For this reason, balancing the preferences of each user is essential, since biasing the results in favor of the majority of the members may consequently make minority users lose the trust and leave the platform.

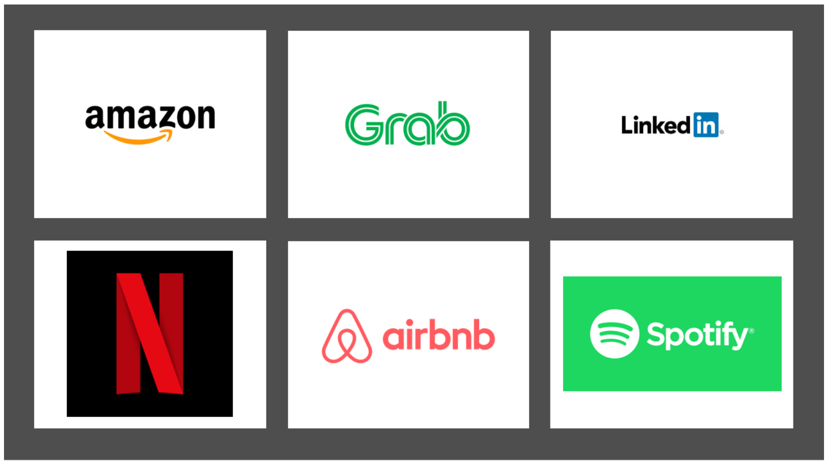

Figure 1.1: successful companies highly influenced by their recommenders systems

The motivation driving this thesis is that even though the recommendation technologies, such as collaborative filtering (CF) and its modern implementation like matrix factorization (MF), have been widely studied, while in group recommendations there is still room for improvement. Specifically, the societal impact of group recommendation pipelines is an under-explored problem. It is widely known that algorithms might affect users by generating systematically worse results for a subset of users (*unfairness*)[8]. Unfairness can affect the group recommendation process, since a large number of group modeling techniques (which aim to combine the individual preferences into a unique model) try to optimize the overall happiness of the group; consequently, in some scenarios, the modeling strategy tends to incentive the favorite items of the users with similar preferences while penalizing the favourite items of the minority. This results in a bias, in function of the popularity of the items.

In the literature, there are developed group modeling strategies that mitigate the unfairness between members of the same group, such as:

- FAI[5] strategy, which adds each user favorite item in the group recommendations.
- Greedy Least Misery [20] that apart from selecting those items that have a higher mean relevance across users, also penalizes them in function of their min score, to ensure that a user does not find "unwanted" items in the earlier ranks of the recommendation.
- XPO [14] assigns those items that are not dominated by the others or dominated by the lower number of items possible, and defines an item as dominant if there is no item with higher relevance.
- Single Proportionality Greedy [16] ensures that the proportion of relevant items is equal among members by simply considering the preferences of those users that have obtained less significant items in the group recommendations.
- And GFAR [11] simply penalizes the user's right to vote according to how relevant was the previous recommendation.

We will explain these approaches and their main drawbacks in detail in section 2.3.

However, we consider their strategies can still be improved mainly due to these reasons:

- Some of them are based on principles that do not impose fairness in all the cases, e.g. If seven out of eight users love an item, Least Misery Greedy will most probably remove this item from the group recommendations.
- In other cases, the strategies are too general and can not distinguish distinct scenarios, e.g. Single Proportionality Greedy and FAI will not identify the difference between an item that is relevant for a member and detested by the others or an item that is significant for a user and mediocre for the others.
- And finally, strategies as GFAR, penalize the opinion of the user in function of how favorable were the previous recommendations. We think that on a real scenario, the user's opinion counts equally regardless of the previous recommendations. In other words, GFAR is using a wrong balancing criteria by tending to penalize the opinion of those users that in the previous recommendations obtained more relevant items, however we consider that to be fair is to give the users the right to vote always, as in reality. 4.5.2.2.

Consequently, we consider it necessary to provide a group modeling strategy which is capable of balancing the user relevance without penalizing their opinion, simply making it more tolerant or radical in function of how favorable were the previous recommendations in comparison with the others.

Moreover, in the literature have been used metrics designed to evaluate individual recommendations to appraise the overall group recommendations by simply transforming the group score as the mean of the individual scores. This leads to metrics that are incapable to assess fairness in the group, e.g. the metric will not be able to distinguish between a group in which one user got a score of 1 and the other a score of 0 and a group with a score of 0,5 for each user. In section 4.5 we explain in detail this metrics together with its advantages and disadvantages.

In addition to the solution of this problem, we consider as necessary the specification of an aggregation strategy for metrics capable to transform the group score as the minimum of the individual scores when the variance across the scores in the group is high, and to transform the group score as the mean of the individual scores when the variance is low.

1.1 Context of the Study

The present thesis was carried out during the first semester of course 2020/21, in collaboration with EURECAT, the international technology center of Catalonia settled in Barcelona. The need of the company consists on developing a group modelling strategy that significantly improves the group recommendations, by ensuring societal and beyond-accuracy properties of algorithms (fairness, in this case).

It is relevant to mention that EURECAT is a data-driven company, and the output of this thesis is intended to be considered by the *Big Data* department. So, we are going to focus on the optimization of the novel technique instead of making the model usable for people with no programming background.

1.2 Objectives

The main objectives to be achieved on this project are the following:

1. Provide a novel group modelling strategy capable of solving the drawbacks of the existent group modeling techniques and consequently provide more fair and more accurate group recommendations.

2. Provide an aggregation strategy for metrics capable of transforming the group score as the minimum of the individual scores when the variance across the scores in the group is high, and oppositely, transform the group score as the mean of the individual scores when the variance is low.

1.3 Overview of the Thesis

This thesis is organized on six chapters. More specifically, in Chapter 2, the state of the art, we have explained different architectures a group recommender system can adopt. Traditional group modeling and fair group modeling techniques are also presented. In Chapter 3, we start describing the basis of our new approach, then, we provide the pseudo-code of the algorithm and at the end, we have included an example to make everything more concrete. Chapter 4 contains the explanation of the whole pipeline of our experimental set up, from the data sources to the evaluation metrics. At the end of this chapter, we propose a slight modification of these evaluation metrics to assess also the divergence of the scores between the users of the same group. To conclude, in Chapter 5 we present the results obtained from the experiments and in Chapter 6 we reveal all the conclusions obtained from this thesis and the futures lines of work.

2. State of the Art

In this chapter, we firstly introduce different existing architectural solutions and we discuss which one is more suitable for our thesis. Secondly, we describe the classical approaches that have been considered to recommend items to a group given each member preferences, and finally, we explain the latest fair group recommender strategies that have shown better results.

2.1 Architectural Solutions

Since until now, there is no public dataset with groups and their structure, it is necessary to create synthetic groups. Note that, in real life applications the groups are given naturally, and there is no need to create them.
Apart from that, in order to obtain group recommendations from the individual user ratings, it is also necessary to predict the individual preferences and then mix them and obtain the group recommendations through a group modeling strategy. Another approach is to simply build a model that directly outputs the group recommendations given the structure of the groups and the user's ratings[10, 5].
We will explain each one of them in the following section:

2.1.1 Constructing Group Preference Models

In this section, we present the group preference model architecture. Firstly, as it can be seen in Figure 2.1, the system creates groups based on the user preferences, characteristics and constraints imposed by the system (such as how many groups, or how many users per group). Secondly, a group model with the users preferences and groups structure is generated in order to predict the whole group relevances, and finally, the system will use these model to compute the group relevance for each item and use it to obtain the group recommendations.

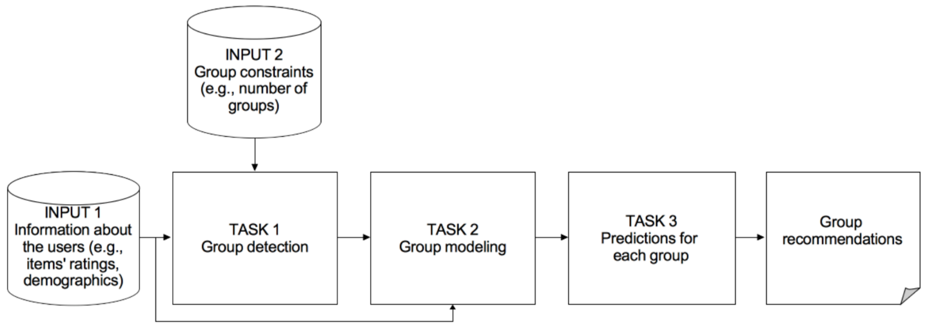
INPUT 2
Group constraints (e.g., number of groups)
INPUT 1
Information about the users (e.g., items' ratings, demographics)
TASK 1
Group detection
TASK 2
Group modeling
TASK 3
Predictions for each group
Group recommendations

Figure 2.1: Group preference models architecture, from: [5]

2.1.2 Aggregating Individual Predictions

With this approach, it is still necessary to generate groups from the data and the imposed constraints, but, unlike the previous architecture, now the groups can be created in parallel with task 2, since this will compute just the individual predictions regardless of the groups structure. At the end, the group modeling will obtain the group recommendations by merging the individual predictions of the users in each group (Figure 2.2).

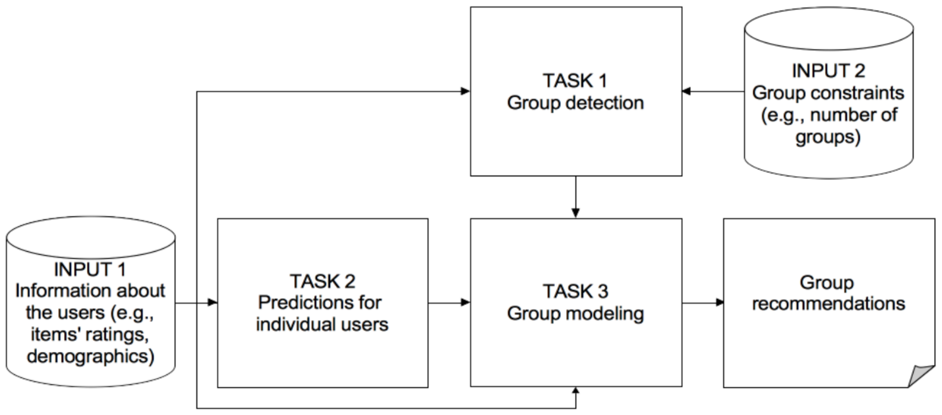
TASK 1
Group detection
INPUT 2
Group constraints (e.g., number of groups)
INPUT 1
Information about the users (e.g., items' ratings, demographics)
TASK 2
Predictions for individual users
TASK 3
Group modeling
Group recommendations

Figure 2.2: Aggregating individual predictions architecture, from: [5]

Note that this process simply consists on: on one hand building a traditional recommender that predicts each user preferences, and on the other hand, by means of a group building strategy which we willll define in section 4, generate synthetic groups, and finally use an ensemble model (Group model in the Figure 2.2) and combine them.

2.1.3 Merging Individual Recommendations

While in the previous architecture all the individual preferences are contemplated and aggregated together, with this approach are just going to be merged the individual top n-items through an union Figure 2.3.

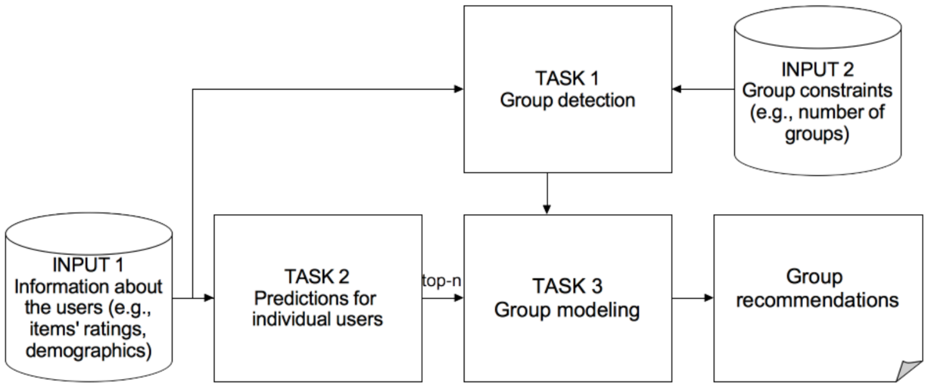
TASK 1
Group detection
INPUT 2
Group constraints (e.g., number of groups)
INPUT 1
Information about the users (e.g., items' ratings, demographics)
TASK 2
Predictions for individual users
top-n
TASK 3
Group modeling
Group recommendations

Figure 2.3: Merging individual recommendations architecture, from: [5]

The advantages of this system, compared to the aggregating individual predictions architecture, are basically in terms of computational performance, since the set of items the group modeling strategy will be working on will decrease considerably. On the other hand, working just with the union of the individual top N-recommendations, instead of all the items, is an inevitably loss of information and consequently a decrease in the potential accuracy on recommending items.

2.1.4 Discussion

In this thesis, we focus on merging individual recommendations and aggregating individual predictions architectures for two reasons: in practice both aggregating and merging individual preferences have shown slightly better performance than the group preference model, and apart from this, both aggregating and merging individual preferences make the overall group recommender system more versatile, since it needs to provide individual recommendations instead of group recommendations just needs to omit the group modeling step.

Consequently, the group modeling strategy proposed will have to provide group recommendations based on the detected groups and on the individual relevance obtained from a recommender system.

2.2 Classic Group Modelling Strategies

To ease the explanation of these strategies, we will consider the set of all items I and the set of all users U, list of recommendations Top-N$_G$ the group recommender system has to deliver, the length N of Top-N$_G$, the relevance of item i for the user u $rel(u, i)$ obtained from the the underlying recommender system (in our case tensorflow recommenders 4), a group $G = \{u_1, ..., u_m\}$ consisting of m

users drawn from U.

In this section, we present some of the most widely know techniques that have settled the principles of more complex strategies such as the ones we will cite in the following one 2.3.

2.2.1 Additive Utilitarian

As depicted in Table 2.1, what the additive utilitarian does is to simply transform the relevance of the item for the group as the sum of all the individuals relevance $rel(i) \sum_{u \epsilon G} rel(u,i)$, and then select the item with higher group relevance. Even though the strategy is basic, it works very good specially when the users in the same group have similar preferences [3]. Oppositely, when the users preferences are diverse, this approach tends to be unfair because the strategy itself biases the recommendations on the most popular items. E.g., in the example in the Table 2.1, while user 1 and user 3 get their favourite item recommended in the first position, user 2 finds his favourite item on the fourth position.

item	User 1	User 2	User 3	Score
Item 1	8	2	9	19
Item 2	7	5	6	18
Item 3	5	6	4	15
Item 4	3	10	4	17
Item 5	9	1	10	20

Table 2.1: Example of the additive utilitarian strategy

2.2.2 Multiplicative Utilitarian

Now, for each item, their different user relevance will be multiplied in Table 2.2, so with this approach the items of similar relevance and higher mean value will tend to have a higher rank compared to the previous example. Christensen and Schiaffino [4] adopted this strategy in order to produce music recommendations.

item	User 1	User 2	User 3	Score
Item 1	8	2	9	144
Item 2	7	5	6	210
Item 3	5	6	4	120
Item 4	3	10	4	120
Item 5	9	1	10	90

Table 2.2: Example of the multiplicative utilitarian strategy

Although this approach selects those items with higher mean and lower variance, in some cases, specially when the member's preferences are diverse, it may end up recommending mediocre items which any user can like or dislike.

2.2.3 Borda Count

The Borda Count approach [1] assigns scores in function of the position of each item in the user's list. Therefore, the least favorite item of a user gets a score of 0, the 2nd least favorite item gets one point more and so on. In cases when two items have the same relevance for a user, the points are divided between them, e.g. as it is demonstrated in the Table 2.3 both, Item 3 and Item 4, share the position 0 and 1 for the User 3, thus, the points will be split, so each item will get a score of 0,5.

item	User 1	User 2	User 3	Score
Item 1	3	1	3	7
Item 2	2	2	2	6
Item 3	1	3	0,5	4,5
Item 4	0	4	0,5	4,5
Item 5	4	0	4	8

Table 2.3: Example of the Borda count strategy

Although with this strategy we ensure that all the sums of the preferences of an user are equally, there's still a bias to recommend the majority's preferences. Moreover, if the individual preferences follow an non-linear tendency, the new relevant scores are not correctly assign to the users.

2.2.4 Copeland Rule

Copeland Rule strategy [6] compares each item with the rest. If an item beats the other, which means it got higher relevance than another one for at least one user, then 1 point it's added to the item score. Oppositely, if the item is beaten by another one, then 1 point it's subtracted from the score, and finally if they are equally relevant, nothing is added nor subtracted, Table 2.4.

item	Item 1	Item 2	Item 3	Item 4	Item 5	Score
Item 1	0	+1	+1	+1	-1	+2
Item 2	-1	0	+1	+1	-1	+1
Item 3	-1	-1	0	0	-1	-3
Item 4	-1	-1	0	0	-1	-3
Item 5	+1	+1	+1	+1	0	+4

Table 2.4: Example of the Copeland rule strategy

The drawbacks mentioned previously are still present in this method, which means: if more than the half of the members in a group agree on a set of items, no matter the opinion of the others, the recommender will suggest the items of the majority .

2.2.5 Plurality Voting

Plurality voting [15] aims to simulate a real scenario where each user votes for their favorite item and the alternative (or alternatives) that gets more votes wins.

Users	Iteration 1	Iteration 2	Iteration 3	Iteration 4
User 1	I5	I1	I2	I3
User 2	I4	I4	I4	I4
User 3	I5	I1	I2	I4,I3
Selected	I5	I1	I2	I4,I3

Table 2.5: Example of the plurality voting strategy

Even though the recommendations are democratically chosen, the strategy is far from being fair. As it can be seen in the example depicted in Table 2.5 the recommendations still discriminate the minorities.

2.2.6 Least Misery

Least Misery [13] simply assigns the minimum relevance as score for each item. So, the strategy minimizes the number of disliked items in the early ranks, but it may also be too radical. For instance, in the case introduced in the Table 2.6, Item 5 is the favorite item for both User 1 and User 3; However, it will be retrieved as the last one, because is the least favorite for User 2.

item	User 1	User 2	User 3	Score
Item 1	8	2	9	2
Item 2	7	5	6	5
Item 3	5	6	4	4
Item 4	3	10	4	3
Item 5	9	1	10	1

Table 2.6: Example of the least misery strategy

Besides its drawbacks, multiple strategies adopt this approach and include this principle when calculating each item score, such as the Greedy Least Misery Group Modeling strategy [20], which we will present in detail in Section 2.3

2.2.7 Most Pleasure

While Least Misery assigns as item score the minimum relevance, most pleasure does the opposite and assigns the maximum relevance as item score, check Table 2.7, forcing each user favourite item to appear in top ranks, even though this might mean putting on top ranks one item that another dislikes.

item	User 1	User 2	User 3	Score
Item 1	8	2	9	9
Item 2	7	5	6	7
Item 3	5	6	4	6
Item 4	3	10	4	10
Item 5	9	1	10	10

Table 2.7: Example of the most pleasure strategy

2.3 Fair Group Modeling Strategies

The famous No Free Lunch (NFL) theorem states that: *"There is no model that is a priori guaranteed to work better. The only way to know for sure which model is best is to evaluate them all. Since this is not possible, in practice we make some reasonable assumptions about the data and evaluate only a few reasonable models"* [7].

Keeping this in mind, we have selected here some of the most recent fair group modeling techniques according to different criteria, such as the novelty introduced by these approaches, the relevance and the importance in the literature, and especially the accuracy of these strategies that is measured by metrics such as Z-Recall, Discounted First Hit and Normalized Discounted Cumulative Gain. Z-Recall calculates the percentage of users in the group that did not obtain any significant item in Top-N_G, Discounted First Hit measures how soon a member receives a relevant item, Normalized Discounted Cumulative Gain assigns a score to a user in function of how many relevant items obtained in each rank of Top-N_G. However, we will explain them in detail in section 4.5.
Considering everything, the selected fair group modeling strategies conform these characteristics are:

2.3.1 FAI aggregation strategy

Felfering et al. [5] proposed an algorithm under the principle that each member in the group should find his favorite items in the first positions of the group recommendations list. They called it FAI, which is an acronym of "Fairness" and what basically does is: it inserts each user's favorite item to the top-N of the group (denoted as top-N_g), and once each member has introduced their main recommendation to top-N_g, the process is repeated in the reverse order. So, the last member who added his/her favorite item now will be the first one to introduce his/her second main preference. This process is repeated until top-N_g is completed. Although it is a simple strategy, it already ensures that all the members find an equitable number of relevant items in top-N_g and more or less in similar positions as the others. These are some properties that fair group recommendations should have.

Even though the FAI aggregation strategy ensures that all the members find their favorite items in top-N_g, nothing prevents it from recommending also the least favorite items of other users in the first positions. Moreover, in the worst case, where all the preferences of the users are distinct, a user will need to wait 2 times the number of users -1 to obtain a relevant item, which definitely is not what an user expects from a group recommender system.

2.3.2 Greedy Least Misery

Greedy Least Misery was built by Xiao et al.[20], in order to increase the happiness of the overall group and to minimize the dissatisfaction of the least happy member of the group given an item.
In one hand, Greedy Least Misery tries to optimize the overall satisfaction by introducing the concept of member utility. This concept is described as the proportion of relevance obtained from the recommendations compared to the maximum that would have gotten in the best scenario. In mathematical terms, Greedy Least Misery maximizes Social Welfare, which is the mean utility of all the users in the group:

$$SW(g,I) = \frac{1}{|G|}\sum_{u \epsilon G} U(u,I)$$

where G is the set of users in a group, I is the set of the recommended items and $U(u,I)$ is the utility of a user for a set of Items which is defined as:

$$U(u,I) = \frac{1}{K \cdot rel_{max}}\sum_{i \epsilon I} rel(u,i)$$

Where $rel(u,i)$ is the relevance of an item for a user, and rel_{max} is then the maximum relevance that has an item for a user.

On the other hand, to minimize the dissatisfaction of the least happy member, the authors introduce the term $F(g,I)$, to penalize those items that have a smaller minimum utility. Consequently, the objective function is defined as:

$$\lambda \cdot SW(g,I) + (1-\lambda) \cdot F(g,I)$$

λ is then a parameter between 0 and 1, defined by the user in order to give more importance to the overall satisfaction or to the fairness of the recommendations. And finally, $F(g,I)$ is the minimum utility of an item for a user:

$$F(g,I) = min\{U(u,I),\, u\,\epsilon\, G\}$$

Intuitively it seems correct to maximize both, the overall group happiness and the least misery. However, there is still room for improvement, since there is no mechanism to balance the utility across the members, and consequently a user might end up with a higher proportion of relevant items than an other.

2.3.3 XPO

The X-Pareto Optimal [14] approach recommends those items that are not dominated by the others. An item dominates another if, for each member, an item i ranks at least as good as another item i' , and there exists at least one member for whom i ranks better:

$$r_u(i) \leq r_u(i'),\; and\, \exists\, u'\,\epsilon\, G : r'_u(i) < r'_u(i')$$

However, since the comparison of the rank of each user for each item is infeasible, Sacharidis proposed to compare just the union of the N individual recommendations, where N is the number of recommendations the group recommender system has to give.
Moreover, to obtain the N-Pareto Optimal recommendations more efficiently, they opted for using a

simple Monte Carlo method. Specifically, they generate a large number of random weight vectors, each representing a different linear aggregation strategy, and count how many times each N-PO item ranks within the top-N . Then, items are ranked decreasingly by their counts. The logic behind this strategy is that items with a higher number of counts will tend to have a higher probably of not being dominated by other items Figure 2.4.

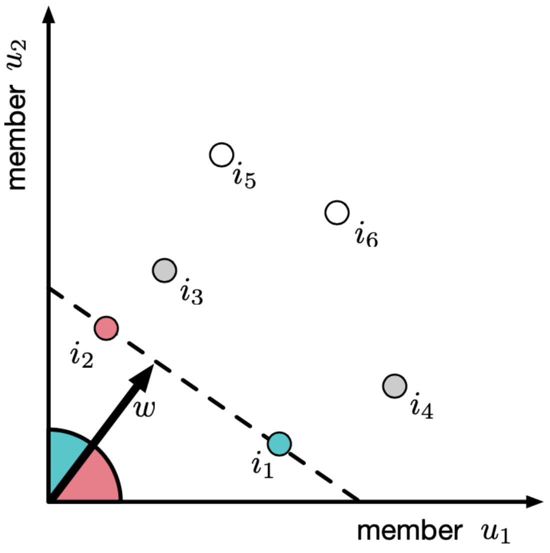

Figure 2.4: Example of items positioned according to their rank for user 1 and user 2 and multiplied by a random vector W , source: [14]

Intuitively, it seems fair to recommend first those items that are not dominated by the others, but in practice it is very complicated to find items with such characteristics, specially when the groups are bigger and the preferences across the members are different. If the algorithm can not find any element that is superior or equal to all the others for all the users, it will try to find items dominated by the least number of items possible. Consequently, it will tend to recommend those items that the majority of the members agree and again biasing the results in function with the popularity of the items.

2.3.4 Single Proportionality Greedy

Single Proportionality Greedy was proposed by Serbos et al. [16] with the objective of ensuring that each user has the same proportion of relevant items as any other, so mathematically they maximize:

$$F_{prop}(G,P) = \frac{|G_P|}{|G|}$$

Where $|G_P|$ is the number of users with at least one relevant item in the recommendations package P, and an item will be considered relevant for an user if is in his individual Top-N recommendations. The procedure the algorithm will be then:

Algorithm 1 Single Proportionality Greedy

```
1: Input: Group of users $G$, Items $I$, Num of recommendations $N$
2: Output: Group recommendations $Top - N_G$
3: $Top - N_G \leftarrow \emptyset$
4: for $j = 1$ to $N$ do
5:     $i \leftarrow argmax_{i \,\epsilon\, I}\, F_{prop}(G, P)$
6:     $Top - N_G \leftarrow Top - N_G \cup : \{i\}$
7:     $I \leftarrow I \,/\, \{i\}$
8: Return $Top - N_G$
```

Even though the objective of the algorithm is fair, it still has some drawbacks. For instance, it does not make any distinction between the favorite item of a user and the N_{th} item and also treats as equal the least favorite item for an user and his $(N + 1)_{th}$ favorite item.

2.3.5 Group Fairness-Aware Recommendation

Group Fairness-Aware Recommendation (GFAR) [11], as SPGreedy, XPO and FAI, also uses the "Merging Individual recommendations" architecture. Hence, the algorithm just takes the individual recommendations and assigns a score in function of the position in the individual Top-N$_u$. They follow Xiao et al. [] approach and define this score as the probability that item i is relevant to u; in mathematical terms:

$$p(rel|u,i) = \frac{\text{Borda-rel(u,i)}}{\sum_{j \,\epsilon\, Top-N_u} \text{Borda-rel(u,j)}}$$

And according to Xiao et al. [], $Borda_rel(u,i)$ is defined as:

$$\text{Borda-rel(u,i)} = |\{j \;:\; rank(j, \text{Top-N}_u) \;>\; rank(i, \text{Top-N}_u),\; \forall\, j \,\epsilon\, \text{Top-N}_u\}|$$

So, for instance, if the Top-N$_u$ has 3 recommendations, the first item would have a $p(rel|u,i) = 2/3$, the second $p(rel|u,i) = 1/3$ and the third $p(rel|u,i) = 0$.
Once the scores have been assigned to the items, GFAR proceeds by assigning the candidates in the $Top - N_g$ according to:

$$f(i, S) = \sum_{u \epsilon G} [p(rel|u,i) \prod_{j \epsilon S} (1 - p(rel|u,j))]$$

Consequently, if an item previously recommended has a high probability to be relevant to a user (e.g. $p(rel|u,i) = 2/3$) the GFAR will penalize now his opinion by multiplying his scores by $1 - 2/3$, and if for another user the previous recommendation was not relevant (e.g. $p(rel|u,i) = 0$) now their relevances will keep intact (would have been multiplied by $1 - 0$). So, the users that did not obtain a relevant item in the previous recommendations, will now have higher probabilities to obtain a relevant item compared to those that have already obtained relevant items.
GFAR has shown the best results in comparison with the other techniques, but still has deficiencies. First of all, it assigns scores linearly according to the ranks of the items but in reality the relevances might follow another distribution, e.g. imagine that the relevances for User1 are: $rel(User1, Item1) = 0.9$, $rel(User1, Item2) = 0.35$, $rel(User1, Item1) = 0.32$ GFAR will convert them to $p(User1, Item1) =$

0.66 $p(User1, Item2) = 3$,$p(User1, Item1) = 0.0$ which is far from representing the real relevance of the items. Secondly, we consider that penalizing the opinion of some users at expenses of the others is not the correct way to balance the proportion of relevant items in Top-N$_g$ for each user; And finally, the scores tend to 0 when the number of Top-N$_g$ is large, which obviously does not affect the accuracy of the method, but may present computation problems.

2.3.6 Discussion

In order to assess the efficiency of our new approach it is necessary to compare it with the actual group modeling strategies. Hence, we have selected Additive utilitarian, FAI, Aggregation strategy, Greedy Least Misery, SPGreedy and GFAR. The other techniques presented will be dismissed with the reason that there is not a significant difference with one of the already selected strategies or simply because the previous studies have shown worse accuracy and fairness in widely know metrics such are Normalized Discounted Cumulative Gain, Discounted First Hit, Zero Recall, which we will explain in detail in section 4.5.

3. Our group Modelling approach

In this section, we introduce ARM: Adaptive Relevance-based Modeling, our novel definition of fairness for group recommendations. We will give an algorithm for finding group recommendations that satisfy this definition. As we previously mentioned, we will define I as the set of all items and U the set of all users, Top-N$_G = \{s_1, ..., s_n\}$ list of recommendations the group recommender system has to deliver, N the length of Top-N$_G$, $rel(u, i)$ the relevance of item i for the user u obtained from the the underlying recommender system (in our case tensorflow recommenders 4.2), $G = \{u_1, ..., u_m\}$ a group consisting of m users drawn from U.

term	meaning
$i \epsilon I$	item of the set of candidate items
$u \epsilon U$	user of the set of Users
$G = \{u_1, ..., u_m\}$	a group composed of m users
$rel(u, i)$	relevance of an item for a user
s_k	the previous recommendation
Top-N$_G$	The top group recommendations
$rel_{k+1}(u, i, s_k)$	The relevance of item I for user U given the previous recommendation s_k for iteration $k+1$
$f(s, i)$	The ARM's score for the i item given the previous recommendation s

Table 3.1: Notation used to describe the ARM strategy

According to this notation we define below the Adaptive Relevance-based Model.

3.1 Definition

Our definition of fairness adopts some of the concepts proposed in the previous group modeling strategies such as:

- The group modeling strategy should try to maximize the overall happiness [3, 4, 1, 6, 15, 20, 14], understanding as maximizing the overall happiness by taking into account all the user preferences equally and always, regardless of the previous recommendations. If the approach penalizes the opinion of a user or simply does not consider it, such as SPGreedy and GFAR does, then they are not considering always the overall happiness, since just the happiness of a subset of the members in the group is considered.
- The Adaptative Relevance-based model should provide a mechanism to ensure that all members find a equitable number of relevant items in Top-N$_G$, considering as relevant items the N items with higher relevance for a user [11, 16, 5].

- A user should identify its relevant items in similar ranks as the others users [11, 16, 5].
- A user can accept "hated" items in Top-N$_G$ if the other recommendations are more favorable to him, considering as hated items those that have lower relevance for the user [11].

Besides, we also consider that:

- A good recommender system should try to minimize the distance between relevant items for a user in Top-N$_G$. Note that this also implies point 2 and 3.
- And finally, user's opinion should not be penalized or dismissed in function with the previous recommendations, understanding as penalized the reduced total sum of the member relevances. Instead, we consider that the user opinion should always count the same, and therefore the sum of each member relevances will be equally. We interpret that the previous recommendations will just tend to make more tolerant or more radical the member. A user will tend to be more tolerant if the differences across their relevances reduce, and consequently accepting reducing the desire to obtain a relevant item on Top-N$_G$ and increase the acceptance of the disliked items, and inversely, we suppose that the user radicalizes when the differences of the relevances expand.

We assume that if the ARM strategy follows all these principles, it will mitigate the mistakes of the other approaches to obtain the group recommendations, previously described in the precedent section 2.3.

In order to follow these principles, ARM first guarantees that all the relevances are positive and different from 0, and then equalizes the power of decision of all the users by ensuring that $\sum_{i\,\epsilon\,I} rel(u,i)$ is the same for all the users:

$$rel(u,i) = rel(u,i) - min(rel(u)) + |min(rel(u))|$$

$$rel_0(u,i) = \frac{rel(u,i)}{\sum_{i \epsilon I} rel(u,i)}$$

Given the new scores, now ARM will opt for selecting those items that have a higher mean relevance, and consequently act as the Additive Utilitarian strategy 2.2. However as mentioned previously, additive utilitarian is far from being fair, therefore apart from recommending the items with higher average relevances, ARM will also radicalize or soften the users preferences by modifying its relevances in function of how much did the previous recommendations satisfy him instead of the others. In mathematical terms we define ARM as:

$$f(s_k, i) = \sum_{u \epsilon G} rel_{k+1}(u, i, s_k)$$

Where s_k is the k_{th} (last) recommendation of the previous recommendations list S, and $rel_{k+1}(u,i,s)$ is then:

$$rel_{k+1}(u, i, s_k) = \frac{rel_k(u,i)^{\left(\frac{\sum_{u \epsilon G} rel_k(u,s_k)}{|G| * rel_k(u,s_k)}\right)^p}}{\sum_{i \epsilon I} rel_{k+1}(u,i)} \tag{3.1}$$

Thus, ARM basically assesses how inappropriate was the previous recommendation s_k for a member compared to the others by dividing the mean relevance $\frac{\sum_{u\epsilon G} rel_k(u,s_k)}{|G|}$ by the user relevance $rel_k(u,s_k)$, and then in the next iteration simply raises the $rel_{k+1}(u,i)$ to the power of this division, so for instance, a user with higher relevance than the mean for the last recommendation, e.g. $\frac{\sum_{u\epsilon G} rel_k(u,recommendation1)}{|G|} = 1,5$ and $rel_k(u,s_k) = 1,8$ will observe how in the next iteration his relevances shrink because they have been raised to the power of $1,5/1,8 = 0,83$ and oppositely, a user with lower relevance than the mean, e.g. $\frac{\sum_{u\epsilon G} rel_k(u,recommendation1)}{|G|} = 1,5$ and $rel_k(u,s_k) = 1$ will see how the differences across the items increases because their relevances have been raised to the power of $1,5/1 = 1,5$. Finally, ARM will simply divide the new relevances by their sums, to ensure that all members have the same influence to request a recommendation independently of the previous recommendations.

Moreover, to make ARM more versatile and adaptable according to the data and the purposes of the recommender system, we also included the parameter P to soft ($p < 1$) or accentuate ($p > 1$) the ARM effect. Note that with $p << 1$ the ARM strategy will tend to work as the Additive utilitarian strategy and with $p >> 1$ will tend to recommend to each user his/her favorite item. So, intuitively larger values of p will improve fairness metrics such as z-recall while smaller values of p will enhance the overall accuracy. Thus, a good configuration of p is essential to optimize the trade off fairness / accuracy.

3.2 Algorithm

We can say that our approach is computationally simpler in comparison with other approaches such as spgreedy (see Section 2.3) in which is necessary to compare each item with the individual preferences in order to check whether the user is satisfied or not, or in the case of XPO (see Section 2.3) which requires multiple multiplications by random vectors. However, if the candidate set of items is huge, ARM may end up being computationally intensive. So, as we previously mention in section 2.1, this strategy is designed to work with both "aggregating" and "merging individual preferences architectures". So, the user can define whether to work with all the items relevance previously obtained by the individual recommender or just operate with the union of the Top-N individual recommendations. In section 5 we will depict the differences between these two approaches in both accuracy and performance time, which for our case in particular is minimal.
In any case, the architecture of the algorithm is the following:

Algorithm 2 Adaptaive Relevance based Modeling

```
1: **Input:** Group of users $G$, Items $I$, Num of recommendations $N$
2: **Output:** Group recommendations Top-N$_G$, $rel$relevance matrix with size $I \times U$
3: Top-N$_G \leftarrow \emptyset$
4: for $u$ IN $G$ do
5:     $rel[u] \leftarrow rel[u] - min(rel[u]) + |rel[u]|$
6:     $rel[u] \leftarrow rel[u]/sum(rel[u])$
7: for $j = 1$ to $N$ do
8:     $i \leftarrow argmax_{i \,\epsilon\, I}\, F_{prop}(G, P)$
9:     Top-N$_G \leftarrow$ Top-N$_G \cup : \{i\}$
10:    $mean \leftarrow sum(rel[:, i])/|rel[:, i]|$
11:    for $u$ IN $G$ do
12:        $rel[u] \leftarrow rel[u]^{(mean/rel[u,i])^p}$
13:        $rel[u] \leftarrow rel[u]/sum(rel[u])$
14:    $I \leftarrow I \,/\, \{i\}$
15: Return $Top - N_G$
```

3.3 Use Case

Here we illustrate an example of the process. ARM will provide 4 recommendations from a set of 16 items (in this case movies) for a group of 4 members, and considering a p = 2 (to accentuate the changes for the example):

term	User 39	User 934	User 380	User 305
$\frac{\sum_{u \epsilon G} rel_k(u,s_k)}{\lvert G\rvert * rel_k(u,s_k)})^p$	1	1	1	1

Table 3.2: Term $(\frac{\sum_{u \epsilon G} rel_k(u,s_k)}{|G| * rel_k(u,s_k)})^p$ in the 1st iteration of the example

movie	User 39	User 934	User 380	User 305	$f(i,s)$
Reluctant Debutante, The (1958)	2.086365	0.838982	0.964249	1.014411	4.904007
African Queen, The (1951)	2.159747	0.921519	0.500333	1.150051	4.731650
Braindead (1992)	1.016288	1.180571	0.880726	1.600319	4.677904
American in Paris, An (1951)	1.965113	0.850871	1.133078	0.725281	4.674343
Innocent Sleep, The (1995)	1.194233	0.926129	1.435117	1.085064	4.640543
Stag (1997)	0.647014	1.247327	0.799865	1.672224	4.366429
French Kiss (1995)	0.674788	0.955387	1.150879	1.580114	4.361168
Aparajito (1956)	1.793500	0.967189	0.793317	0.718968	4.272974
To Catch a Thief (1955)	0.676886	1.142296	0.815845	1.594623	4.229650
Band Wagon, The (1953)	0.597143	1.326415	0.865279	1.191376	3.980214
Strictly Ballroom (1992)	0.574054	1.295725	0.500333	1.241739	3.611851
Assignment, The (1997)	0.679706	1.301784	0.839660	0.711805	3.532954
Deer Hunter, The (1978)	0.662167	1.328671	0.877184	0.436183	3.304206
Dante's Peak (1997)	0.551624	0.589548	1.526747	0.541107	3.209026
Denise Calls Up (1995)	0.455943	0.451834	1.436919	0.412457	2.757153
Smile Like Yours, A (1997)	0.265430	0.675752	1.480469	0.324278	2.745929

Table 3.3: $rel(u,i)$ and $f(i,s)$ values for the 1st iteration of the example

term	User 39	User 934	User 380	User 305
$(\frac{\sum_{u \epsilon G} rel_k(u,s_k)}{\lvert G\rvert * rel_k(u,s_k)})^p$	0.34530	2.13538	1.61660	1.46067

Table 3.4: Term $(\frac{\sum\limits_{u \epsilon G} rel_k(u,s_k)}{\lvert G\rvert * rel_k(u,s_k)})^p$ in the 2nd iteration of the example

movie	User 39	User 934	User 380	User 305	$f(i,s)$
Stag (1997)	0.850172	1.456214	0.660647	1.975471	4.942504
Braindead (1992)	0.993623	1.294836	0.771938	1.852634	4.913032
French Kiss (1995)	0.862601	0.824034	1.189636	1.818567	4.694838
To Catch a Thief (1955)	0.863526	1.206840	0.682116	1.843010	4.595491
Innocent Sleep, The (1995)	1.050553	0.771083	1.699723	1.050264	4.571623
Band Wagon, The (1953)	0.826948	1.660499	0.750170	1.203906	4.441523
Strictly Ballroom (1992)	0.815764	1.579534	0.309438	1.278963	3.983698
Assignment, The (1997)	0.864766	1.595346	0.714593	0.567358	3.742062
American in Paris, An (1951)	1.247690	0.643430	1.160032	0.583115	3.634268
Deer Hunter, The (1978)	0.856995	1.666534	0.766927	0.277448	3.567905
African Queen, The (1951)	1.289050	0.762910	0.309438	1.143399	3.504796
Dante's Peak (1997)	0.804614	0.293927	1.878589	0.380122	3.357252
Aparajito (1956)	1.208935	0.845923	0.651926	0.575716	3.282501
Smile Like Yours, A (1997)	0.625010	0.393368	1.787400	0.179936	2.985714
Denise Calls Up (1995)	0.753389	0.166539	1.703175	0.255683	2.878786

Table 3.5: $rel(u,i)$ and $f(i,s)$ values for the 2nd iteration of the example

term	User 39	User 934	User 380	User 305
$(\frac{\sum_{u\epsilon G} rel_k(u,s_k)}{\lvert G\rvert * rel_k(u,s_k)})^p$	2.11232	0.71998	3.49812	0.39123

Table 3.6: Term $(\frac{\sum_{u\epsilon G} rel_k(u,s_k)}{\lvert G\rvert * rel_k(u,s_k)})^p$ in the 3rd iteration of the example

movie	User 39	User 934	User 380	User 305	$f(i,s)$
Innocent Sleep, The (1995)	1.160235	0.863623	2.597759	1.055225	5.676842
Dante's Peak (1997)	0.660501	0.431271	3.686423	0.709032	5.487227
Smile Like Yours, A (1997)	0.387392	0.531951	3.097496	0.529166	4.546004
Denise Calls Up (1995)	0.574816	0.286493	2.616262	0.607138	4.084708
American in Paris, An (1951)	1.668449	0.758113	0.682703	0.838240	3.947505
Braindead (1992)	1.031420	1.254305	0.164232	1.317589	3.767546
African Queen, The (1951)	1.787431	0.857023	0.006709	1.090891	3.742054
French Kiss (1995)	0.765092	0.905924	0.745617	1.308057	3.724689
Band Wagon, The (1953)	0.699827	1.500302	0.148594	1.113122	3.461845
Aparajito (1956)	1.560868	0.923186	0.090939	0.834063	3.409056
To Catch a Thief (1955)	0.766826	1.192330	0.106543	1.314907	3.380606
Strictly Ballroom (1992)	0.679985	1.447264	0.006709	1.139774	3.273732
Assignment, The (1997)	0.769154	1.457681	0.125368	0.829304	3.181508
Deer Hunter, The (1978)	0.754627	1.504226	0.160533	0.626857	3.046242

Table 3.7: $rel(u,i)$ and $f(i,s)$ values for the 3rd iteration of the example

term	User 39	User 934	User 380	User 305
$(\frac{\sum_{u \epsilon G} rel_k(u,s_k)}{\lvert G\rvert * rel_k(u,s_k)})^p$	1.49624	2.70050	0.29846	1.80885

Table 3.8: Term $\frac{\sum_{u \epsilon G} rel_k(u,s_k)}{\lvert G\rvert * rel_k(u,s_k)})^p$ in the 4th iteration of the example

movie	User 39	User 934	User 380	User 305	$f(i,s)$
Braindead (1992)	1.006134	1.333505	0.766854	1.622453	4.728945
Band Wagon, The (1953)	0.563146	2.162856	0.744290	1.195905	4.666198
American in Paris, An (1951)	2.066267	0.342356	1.173265	0.715966	4.297854
African Queen, The (1951)	2.290597	0.476763	0.295261	1.153051	4.215672
To Catch a Thief (1955)	0.645700	1.162962	0.673939	1.616484	4.099085
Assignment, The (1997)	0.648637	2.000912	0.707475	0.702219	4.059243
Strictly Ballroom (1992)	0.539425	1.962533	0.295261	1.248200	4.045419
French Kiss (1995)	0.643517	0.553838	1.204544	1.601282	4.003181
Deer Hunter, The (1978)	0.630392	2.178167	0.761657	0.423267	3.993484
Aparajito (1956)	1.870144	0.582801	0.642827	0.709525	3.805297
Dante's Peak (1997)	0.516464	0.074627	1.940820	0.528912	3.060824
Denise Calls Up (1995)	0.419517	0.024728	1.752009	0.399490	2.595744
Smile Like Yours, A (1997)	0.232448	0.131514	1.842565	0.311548	2.518075

Table 3.9: $rel(u,i)$ and $f(i,s)$ values for the 4th iteration of the example

Note that in the example the sum(rel[u]) is different from 1 because we multiplied all the scores by the number of candidate items to obtain values around 1 and to ease the interpretation of the process ARM follows.

As it can be seen on the example, on the first iteration all the weights $(\frac{\sum_{u \epsilon G} rel_k(u,s_k)}{|G| * rel_k(u,s_k)})^p)$ are equal to 1 Table 3.2 because there are no previous recommendations and consequently there are not modifications in user relevances Table 3.3.
In the next iteration, since there is already a previous recommendation with different relevances between the members in the group, ARM will balance each user preferences, and consequently those users that have previously obtained a more favorable recommendation will perceive how their relevances shrink. In particular, for the user 39, who in the previous recommendation has clearly benefited in comparison with the others, the probability to get a relevant movie in the new iteration will clearly decrease Table 3.5, denoting a more tolerant behaviour of this user. (note that while the power of decision of the user 39 (sum(rel[u])) does not decrease, the probabilities of obtaining a relevant movie yes).

4. Experimental Setup

The goals of the experiments are three-fold. Firstly, to understand the trade off between fairness and quality, and thus, to ensure that our method, apart from giving fair recommendations, does not penalize the overall performance. Secondly, we want to assess the performance of our new approach by comparing it with some of the most relevant fair group modeling techniques; And finally, we are interested in understanding how ARM will perform in function of the size of the groups and the similarity between the members in a group.
To do so, we will construct a pipeline with the following characteristics:

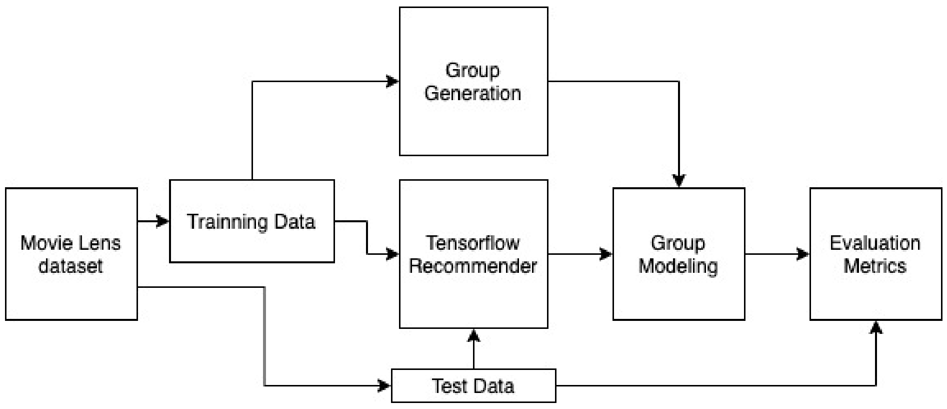

Figure 4.1: our recommender system

4.1 Datasets

There are multiple scenarios in which group recommender systems can be applied, hence, a large number of datasets can be useful. However, in this thesis, as it can be observed in Figure 4.1, we will work with just the movie lens 100 thousand and 1 million datasets. Nevertheless, the pipeline built will be as versatile as possible to allow us work with different data as it could be KGRecmusic, if necessary.

The movie-lens dataset contains the following information:

- *movieId*: a unique identifier of the rated movie
- *movieTitle*: the title of the rated movie with the release year in parentheses
- *movieGenres*: a sequence of genres to which the rated movie belongs

- *userId*: a unique identifier of the user who made the rating
- *userRating*: the score of the rating on a five-star scale
- *timestamp*: the timestamp of the ratings, represented in seconds since midnight Coordinated Universal Time (UTC) of January 1, 1970
- *userGender*: gender of the user who made the rating; a true value corresponds to male
- *bucketizedUserAge*: bucketized age values of the user who made the rating, the values and the corresponding ranges are
- *userOccupationLabel*: the occupation of the user who made the rating represented by an integer-encoded label; labels are preprocessed to be consistent across different versions
- *userOccupationText*: the occupation of the user who made the rating in the original string; different versions can have different set of raw text labels
- *userZipCode*: the zip code of the user who made the rating

Although movie lens contain ratings contextualized with lots of characteristics such as timestamps, movie categories, user age... to make the whole pipeline more flexible, our recommender system will just take as input those data that can be found in any dataset, which is the triple(userId, movieTitle, ratings).

Again as depicted in Figure 4.1, we will split the dataset in two parts where 70 % of it will be used to train the tensorflow recommender which will provide the individual user preferences and the remaining 30 % will be used to compute the accuracy of the tensorflow recommender, and to assess the group recommendations obtained with the different evaluation. The individual user preferences will be used by our group modeling strategy metrics (Section 4.5).
Since movie lens dataset has been widely used and minutely cleaned by the community, there is no need to perform sanity-check tasks and look for "NAN" values, outliers, duplicates or errors.

4.2 Individual Recommender System

As mentioned previously in Section 2.1 we will be working on both "merging individual recommendations" and "aggregating individual predictions" architectures. Before applying our group modeling strategy, it is necessary to obtain the individual preferences and to do so, we will use TensorFlow Recommenders (TFRS) [12], an open-source TensorFlow package focused on recommender systems.

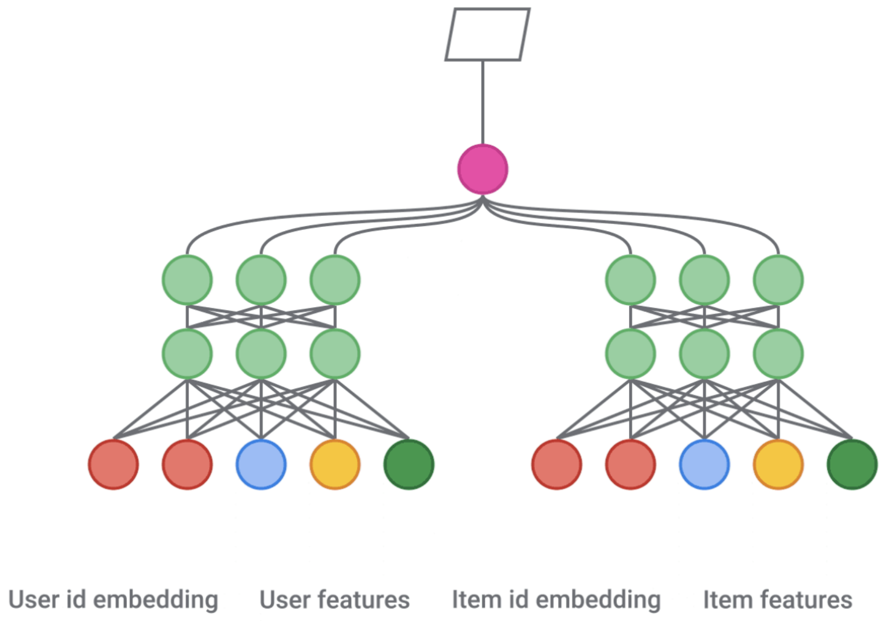

Figure 4.2: architecture of our recommender system , source: [12]

As we previously mentioned, in consideration of our recommender system should be the most flexible possible, we will work with the triple(userId, movieTitle, ratings). As seen in the Figure 4.2, we will first compute the user id embedding from the user id and the item embedding id from the movie title, both through the tensorflow keras embedding layer. Then, again from the movie title, we will calculate first the item id and then the item features, such as the similarity across movie titles by means of the text vectorizer keras layer (and thus ensure for instance that if a user have seen star wars 1 the recommender will suggest also star wars 2 because have similar movie titles).

Once the ids and features have been generated for both user and items, we will pass them through a set of sequential dense layers, which have shown the optimal configuration with one layer of 64 neurons with relu activation function and one last layer of 32 neurons without any activation function.
With the obtained output, tensorflow recommender model will provide the preferences of each user that minimize the metrics we have imposed. In our case, this is a mix between the root mean squared error and binary cross entropy to make sure the recommender does both: ranks items according to the probabilities to be seen (binary cross entropy), and also does not incentive those that have obtained low ratings (root mean squared error). And finally, the metric used to assess the individual recommender system is the factorized Top K

Having an accurate individual recommender is essential to obtain valid group recommendations, since the relevances predicted fron it will serve as baseline for the posterior group modeling strategies. However, we will not enter more in details because we do not consider it as the purpose of this thesis.

4.3 Group detection

Since movie lens dataset does not contain information regarding the structure of the groups, it is necessary to construct groups from the data. In the literature, there are multiple group detection approaches, such as creating groups considering the users features, e.g. join users considering the age, the gender, the interests, their demographic data... or in the case of social networks data usually those users that have strong social ties (follower / followee) are merged in the same group. However, the most common procedure and the one we are going to consider for our experiments consists on creating groups based on the similarities between their ratings.

In consideration of our experiments, we want to evaluate how ARM works in function of the size of the groups and the similarity across members in a group. This is why we will create groups from 2 users up to 8 users following three different approaches:

- *Random*: Members of Random groups are selected with replacement from the set of users U with an uniform probability. Random groups loosely correspond to the real-life equivalent of groups that have unrelated members.
- *Similar*: Pretends to select those members that have similar preferences, following the approach based on [2, 9]. We compute the similarities between pairs of users as the Pearson Correlation Coefficient (PCC) between their ratings. Previous works suggested that pearson correlation coeficient greater than 0.3 already provides significant differences with respect to the randomly selected method. So, to form a synthetic group, we randomly select a user from U and then greedily select at random further users but only drawing them from those who have a PCC greater than 0.3.
- *Divergent*: Oppositely to the similar group building approach, now divergent strategy will try to create groups with divergent preferences by following the same strategy similar groups use but selecting now the users who have a PCC less than 0.1, which according to [9] will already provide a small significant difference respect to randomly selecting users.

4.4 Group modeling strategies

In order to assess the efficiency of ARM it is necessary to compare it with different group modeling strategies, hence, as we previously discussed in Section 2.3.6, we have selected Additive utilitarian, FAI aggregation strategy, Greedy Least Misery, SPGreedy and GFAR.

4.5 Evaluation Metrics

Here, we will present the metrics that according to us can evaluate more accurately our model. A brief explanation of the reasons why we considered these metric as the most accurate will be at the end of the section.

Even though we believe that these metrics are excellent in terms of interpretability, we consider that all of them may assess the overall group recommendations too generally, and consequently in some cases they are unable to distinguish between fair and unfair scenarios. Thus, at the end of the section we introduce our proposal metric which we consider can evaluate more accurately unfair recommendations, and we are going to compare it with the others.

4.5.1 Effectiveness metrics

We describe as effectiveness metrics those metrics that grade individually the recommendations provided by the recommender and does not aim to assess whether the recommendations are fair or not.

4.5.1.1 Mean Discounted First Hit

Discounted First Hit measures how soon a user finds a relevant item in the Top-N$_G$. Relevant item are those items in the Top-N$_G$ that have been rated by the user in the test dataset. Therefore, Discounted First Hit is defined as:

$$MDFH@N(G) = \sum_{u \epsilon G} \frac{DFH@N(u)}{|G|}$$

$$DFH@N(u) = \frac{1}{\log_2(fhr(u) + 1)}$$

Where $fhr(u)$ is the rank "first hit", the rank in Top-N$_G$ where user u finds the first relevant item. In case a user does not find any relevant item in Top-N$_G$ then their individual DFH(u) is equal to 0 . Although it is rank sensitive, it is very general and may not represent correctly the real situation, e.g. a user may have much more relevant items in Top-N$_G$ than another one, but if the second user has the first recommendation sooner, he will obtain a higher score than the first one.

4.5.1.2 Mean Normalized Discounted Cumulative Gain

For each rank, Normalized Discounted Cumulative Gain calculates how many relevant items are present and weights them in function of the rank position they appear, so now apart from being rank sensitive as MDFH, also takes into account the whole list of recommendations.

$$MNDCG@N(G) = \sum_{u \epsilon G} \frac{NDCG@N(u)}{|G|}$$

$$NDCG@N(u) = \frac{DCG@N(u)}{IDCG}$$

$$DCG@N(u) = \sum_{k=1}^{N} \frac{\text{Top-N}_G \cap ir_u}{\log_2(k+1)}$$

Where ir_u is the set of the user real relevant items, which we will consider all those that have been rated by the user. Besides, $IDCG$ will be the maximum possible value of $DCG@N(u)$

Adversely to MDFH, now MNDCG will consider always all the items in Top-N$_G$ and thus, evaluate more precisely the recommendations, however, as MDFH, MNDCG can not distinguish between a group that both users obtained the same "x" score and a group with a member with a score of $2 \cdot$"x" and a member with a 0 score.

4.5.2 Fairness metrics

Fairness metrics are those that try to measure how fair are the group recommendations. However, as it will be described in this section, they usually tend to poorly asses the accuracy of the recommendations. Hence, apart from introducing the most widely known fair metric (Zero Recall) we also include our metric proposal that will try to mitigate the problems arising from the previous metrics.

4.5.2.1 Zero Recall

Zero Recall [19] measures the proportion of users in a group that did not get any relevant item in Top-N$_G$:

$$zRecall@N(G) = \frac{|\{u \epsilon G : recall@N(u) = 0\}|}{|G|}$$

$$recall@N(u) = \frac{|\text{Top-N}_G \cap ir_u|}{|ir_u|}$$

Although Zero Recall informs us which proportion of candidates did not obtain any relevant item, the metric itself is poor because it is not rank sensitive and because it does not differentiate between a user with just one relevant item and a user with multiple relevant items.

4.5.2.2 Our Proposal

First of all, we want to emphasize that the objective of this metric is not the interpretability. We are conscious that merging accuracy and fairness into one metric implicitly makes it more ambiguous and harder to interpret it. The purpose of this metric is just to represent more accurately the overall satisfaction of the group with the modeling strategy.

The motivation driving this metric arises from the fact that the actual metrics fail in some scenarios to express correctly the fairness or/and the overall accuracy. This might happen because in some cases they tend to be too general (z-recall) and in other cases because they simply transform the overall group score as the mean of the member's individual scores, without assessing the differences across the members scores.
We have depicted an example to demonstrate these metrics deficiencies:

relevant for user?	User 1	User 2	Group Score
item 1	true	true	-
item 2	false	true	-
item 3	false	true	-
item 4	false	true	-
item 5	false	true	-
item6	false	false	-
zRecall	0	0	0
DFH	1	1	1
NDCG	0.34	0.95	0.63

Table 4.1: zRecall, DFH, NDCG scores for an unfair example

relevant for user?	User 1	User 2	Group Score
item 1	true	true	-
item 2	true	true	-
item 3	false	true	-
item 4	false	false	-
item 5	false	false	-
item6	false	false	-
zRecall	0	0	0
DFH	1	1	1
NDCG	0.58	0.74	0.64

Table 4.2: zRecall, DFH, NDCG scores for a fair example

As it can be seen in Table 5.1 and 4.1 both examples present very distinct scenarios. In both of them, we can clearly observe that zRecall and DFH can not quantify the differences in the recommendations for user1 and user2 in table 4.1, and although ndcg metric correctly distinguished both users differences, at group level it also fails to determine the discrepancy between the members in the group.
Having in mind the drawbacks arising from the actual metrics, we will define a strategy capable of aggregating the individual scores and evaluating both efficiency and fairness, which will have as a pillar the following objectives:

- The metric should appreciate the proportion of relevant items for each user
- the metric should take into account whether the relevant items have been retrieved in earlier ranks or in the latest positions.

Note that such characteristics could be measured by the average ndcg score of a group. Nevertheless, as depicted in the previous example, considering simply the average may fail to assess fairness in the group recommendations, so apart from the these objectives, we will also consider that our metric should:

- Evaluate whether the proportion of relevant items for each user is balanced across members

- Assess whether all users receive relevant items in similar ranks as the others.

In order to obtain a metric capable of assessing all these characteristics, we will first compute the individual ndcg scores, and then assess how balanced the scores are across members by giving higher importance to the lower ndcg scores in the group, and consequently if the scores are regular between users, the metric will tend to output the average ndcg of the group. On the other hand, if the scores are very unfair (distinct) this approach will tend to the minimum ndcg. To do so, we are going to assign weights to each score in function of their rank in the group. Considering that S is the list with the ndcg scores of the groups members, we will define the weights as:

$$weights(s) = rank(s, OS)$$

$$nweights(s) = \frac{weights(s)}{\sum\limits_{s \epsilon S} weights(s)}$$

where $rank(s, OS)$ is the position the score s will have in the OS ordered list of scores. And finally, the Balanced Normalized Dicounted Cumulative Gain (BNDCG) will have as value the ndcg score multiplied by the assigned weight:

$$BNDCG = \sum_{s \epsilon S} s \cdot nweight(s)$$

So, for instance, if the S has 3 recommendations, the first score in OS would have a $weight(s) = 1/6$, the second $weight(s) = 2/6$ and the third $weight(s) = 3/6$.
With this aggregation strategy, now the ndcg scores apart from evaluating the accuracy, will also be able to appraise the balance across metrics in the same group. Thus, if the S are very irregular and tend to favor more certain users, it will tend to the minimum ndcg score of the group and oppositely if the s are very balanced will tend to the mean value (note that if all the scores are regular the max score is almost equal to the mean score).

If we compare now the previous examples with the aggregation strategy we proposed we can clearly appreciate the difference of group punctuations with the ndcg scores:

relevant for user?	User 1	User 2	fair aggregation strategy
item 1	true	true	-
item 2	false	true	-
item 3	false	true	-
item 4	false	true	-
item 5	false	true	-
item6	false	false	-
zRecall	0	0	0
DFH	1	1	1
NDCG	0.34	0.95	0,54

Table 4.3: zRecall, DFH, NDCG scores for an unfair example

In this case, user 1 NDCG score has been multiplied by $1/3 = 0.33$, while the user 2 has been multiplied by $2/3 = 0.66$, and consequently the final score obtained is then: $0.33 \cdot 0.95 + 0.66 \cdot 0.34 = 0.54$

relevant for user?	User 1	User 2	Group Score
item 1	true	true	-
item 2	true	true	-
item 3	false	true	-
item 4	false	false	-
item 5	false	false	-
item6	false	false	-
zRecall	0	0	0
DFH	1	1	1
NDCG	0.58	0.74	0.63

Table 4.4: zRecall, DFH, NDCG scores for a fair example

Following the same procedure as in the previous example, we obtain now a group score of: $0.33 \cdot 0.74 + 0.66 \cdot 0.58 = 0.63$, and if we compare both examples, we can observe that spite of the mean is almost the same for the both recommendations. In this example the overall group score is considerably higher due to the balance across the scores.

Even though this aggregation strategy is simple, in our experiments it has shown good results evaluating both accuracy and fairness (Tables 4.4, 4.3). Moreover, we consider the simplicity as strength, since on one hand it makes it easier to calculate it, and on the other hand it makes it clearly to interpret it. Although, as we mentioned at the beginning the objective of this metric is not seeking interpretability, it seeks to provide a metric capable to detect the best modeling strategy considering both accuracy and fairness.

Moreover, we want to add that we also considered using the variance of S and we come up with the following definitions:

$$min(s) + (max(s) - min(s)) \cdot \frac{var_{max}(|S|) - \sum_{s \epsilon S}(s - \mu)^2}{var_{max}(|S|)}$$

Where the worst variance given a mean is then:

$$var_{max}(\mu, |S|) = \frac{\sum_{N=1}^{\lfloor \frac{\mu \cdot |S|}{s_{max}} \rfloor} (s_{max} - \mu)^2 + ((\mu \cdot |S| \, mod \, s_{max}) - \mu)^2 + \sum_{\lceil \frac{\mu \cdot |G|}{s_{max}} \rceil}^{|S|} (0 - \mu)^2}{|S|}$$

However, apart from adding unnecessarily complexity, the aggregation strategy has shown that in some cases it penalized very much the overall score with lower fairness.

Finally, we could have had introduced more metrics such as Recall which measures the mean proportion of relevant items for each user, or Mean Reciprocal Rank, or Kendall's tau [17, 18], but we have considered that they would not contribute with any additional significant information that the selected metrics already provide. e.g. If we obtain a high DFH score and low NDCG score, implicitly means that there are few interesting items for each user in Top-N$_G$ but they are found in earlier ranks, oppositely, if we get a low DFH and high NDCG, we can extrapolate that there is a large number of relevant items in Top-N$_G$, but they are found in the last ranks. Moreover, if we have a high NDCG and a high Zero Recall (so there is a large proportion of users with any relevant item) we will know that a small set of users obtained highly accurate recommendations. So with this metrics complementing each other, we could get an idea of how balanced are across the users and in which positions they are usually found.
Besides, since in the literature have been widely used, to make a fair comparison between our strategy and the state of art, their implementation is almost a mandatory.
To conclude, note that some group modeling techniques could obtain high values of NDCG group scores by optimizing some members recommendations at expenses of penalizing the other members recommendations. For this reason, we will use NDCG mainly to evaluate the precision (ranking quality) of the Top-N$_G$ recommendations, while we will use the Zero Recall to assess the Fairness of the Top-N$_G$.

5. Results

Before explaining the results we would like to make a brief recap of the metrics and how they are interpreted:

- *NDCG*: After BNDCG, it's the most complete metric since it takes always into account all recommendations in Top-N_G and it's rank sensitive. We will use it to assess the ranking quality (accuracy) of the recommendations.
- *BNDCG*: Is the most complete metric, besides what NDCG evalutes, BNDCG also appraise how similar are the scores across members in a group.
- *Zero Recall*: Calculates the proportion of users that DID NOT get any relevant item in Top-N_G, since can not assess the ranking quality we will use it mainly to observe how fair is an approach.
- *DFH*: Shows us how soon the members find a relevant item in the Top-N_G. We will use it as complementary of NDCG, to understand if the recommendations are found in earlier ranks or in the lasts positions.

5.1 How the size of the groups impact on the group modeling performance

As mentioned in experimental setup section, to assess how the technique behaves, first of all, we are going to observe the performance of the approach according to the group size. To do so, we are going to compute the Top-N_G considering synthetic groups from 2 users up to 8 picked randomly.
The results obtained for each group size are the following:

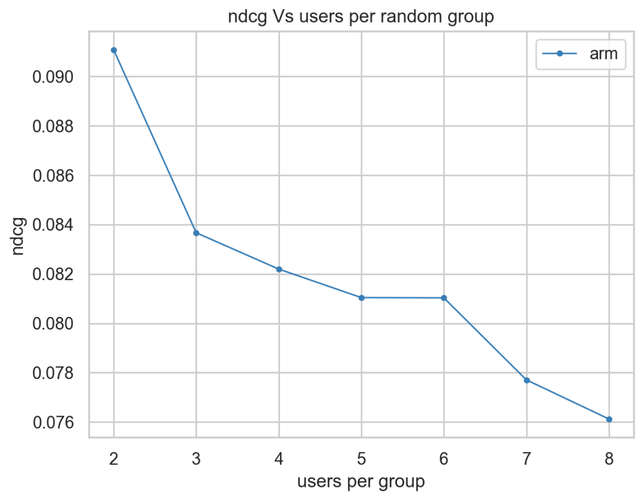

Figure 5.1: ARM’s NDCG scores for the different group sizes

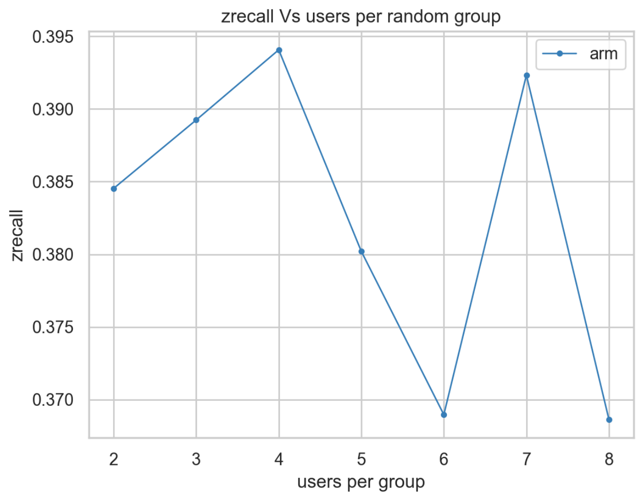

Figure 5.2: ARM’s Zero Recall scores for the different group sizes

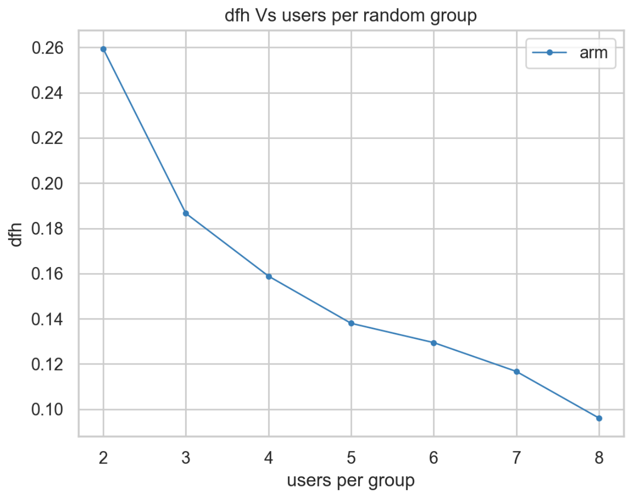

Figure 5.3: ARM's DFH scores for the different group sizes

Intuitively, if we increase the number of members in the groups but instead we keep the number of recommendations constant, we are implicitly reducing the proportion of member's favorite items in Top-N$_g$, and consequently reducing the probabilities of each user to get a relevant recommendation. Moreover, as the proportion of user's favorite items decreases in Top-N$_g$, we can also expect that each user finds a relevant item in higher ranks.

Regarding Figures 5.1 and 5.4 we can see that the X axis refer to the different sizes of groups (2 means that groups are built with two members), and the Y axis show the score of the metric (NDCG, DFH and Z-Recall).
Regarding the behaviour we can say is the expected one, except for the Z-Recall metric ; the group scores tend to decrease as group size increases due to less user's favorite items in the Top-N$_g$.
However, as it can be observed in figure 5.2, Zero Recall and therefore the proportion of users that did not get a relevant item in Top-N$_g$ keep constant regardless of the size of the groups.

5.2 How the similarity between members affect the group modeling performance

In order to appreciate how the similarity across members influence the ARM accuracy, we basically plot the scores obtained with the different group building approaches 4.3. In particular, blue line are the scores obtained with those groups with members with similar interests, which we described as those users that considering their ratings have a Pearson Correlation Coefficient greater or equal than 0.3. Orange line represents the scores obtained for those groups generated randomly, and finally, green line show the scores obtained with groups of a Pearson Correlation Coefficient across members smaller than -0.1:

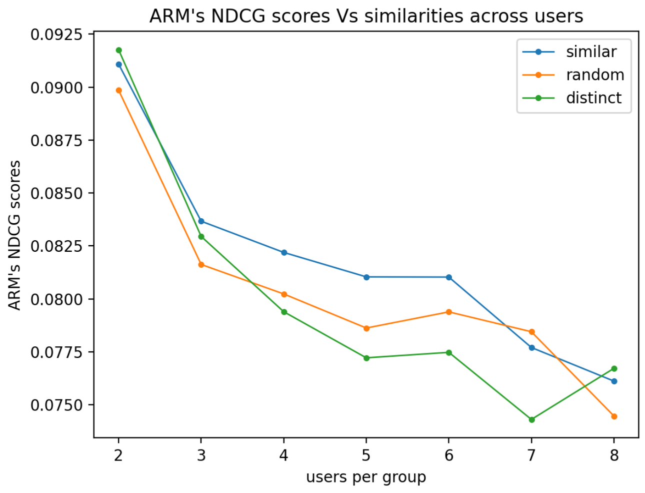

Figure 5.4: ARM's NDCG scores for the different group building strategies

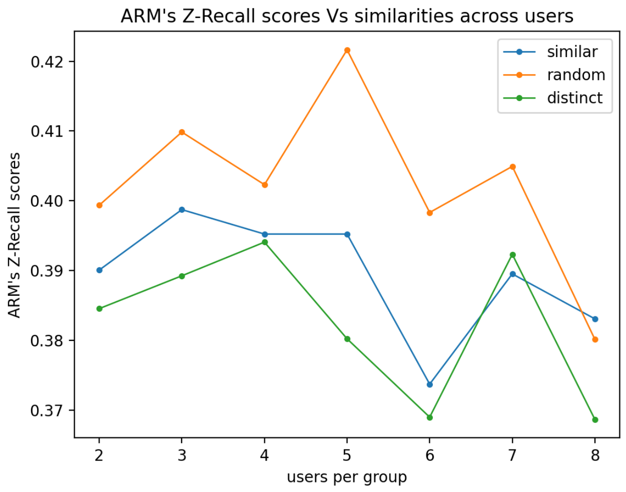

Figure 5.5: ARM's Zero Recall scores for the different group building strategies

And the Pearson Correlation Coefficients across members for each group building strategy are:

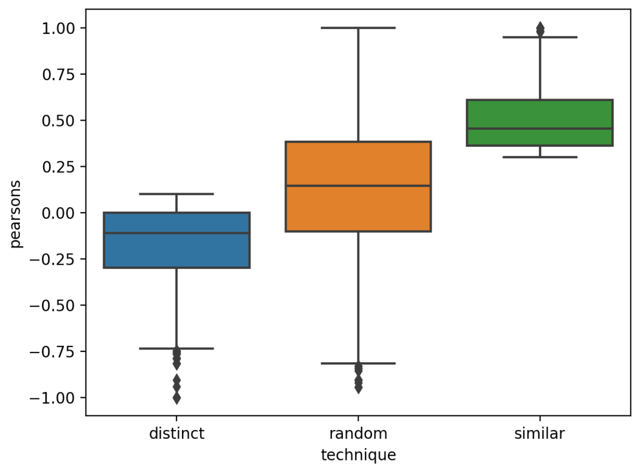

Figure 5.6: Boxplot with the Pearsons correlation coefficients for each group building strategy

As expected, the best scores are obtained for those groups with more similarities across members, and oppositely, the worst scores are generated from those groups with distinct preferences. These scores differences are stimulated by the fact that if two members in group have similar taste, considering the preferences of one implicitly means considering the preferences of the other, and consequently the proportion of relevant items for each user increases.
Besides, generating groups with random preferences, without any criteria that prevents from forming minorities and majorities in groups may aggravate the fairness in a group. Since the group modeling strategy does not provide recommendations based on fair objectives, it may end up biasing the Top-N$_g$ recommendations in favour of the majorities preferences. However, as it can be observed in figure 5.6, the proportion of users with any relevant recommendation in Top-N$_g$ (Zero Recall) remains almost constant across the three different building techniques, which definitely is a good indicator.

5.3 Aggregating individual preferences vs merging individual recommendations

As we previously mentioned, both approaches have their advantages and disadvantages. While just merging individual recommendations can boost the performance of the group recommender system, by considering just the union of the Top-N individual recommendations, the loss of information due to considering just a small set of the total candidates may lead to a decrease of the accuracy.
So, in order to evaluate the trade off between accuracy and elapsed time we are going to compare both of them:

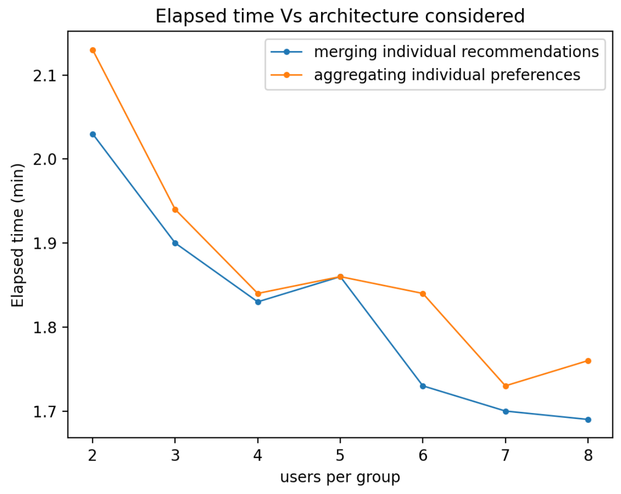

Figure 5.7: Elapsed time of ARM strategy merging individual recommendations vs aggregating individual preferences

As expected, the elapsed time of considering all the candidates relevances rather than just the union of the individual recommendations is more computationally intensive. However, this difference can be increased or decreased in function with the number of candidates the dataset contains. In this experiment, have been considered 1664 movies as candidates, and the merging individual recommendations architecture has obtained the results with a mean of 4 seconds less than the other, which is insignificant.

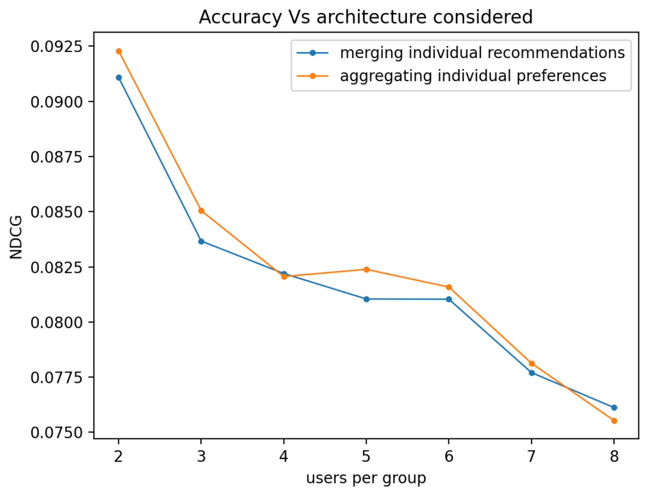

Figure 5.8: NDCG of ARM strategy merging individual recommendations vs aggregating individual preferences

Oppositely, aggregating all the items relevances, we obtain higher accuracy. However, for our experiments the differences are again insignificant.

In view of the results in Figures 5.8 and 5.7, we recommend using an architecture or another in function of the number of candidates the recommender system is working on: if there is a huge number of candidates probably considering just the union of the individual recommendations is enough, oppositely, if there is a small number of candidates, probably the smartest option is to work with all the candidates.

5.4 How parameter P influence the ARM behaviour

As depicted in Equation 5.1 we have included parameter P in the Adaptative Relevance-based Modeling technique with the objective of letting the user define the behaviour of the ARM group modeling technique. So, on one side, if the user considers the group recommender system should not balance too much the user relevances and therefore tend to favor more the majority preferences can do it by setting the parameter p to a smaller value, and on the other side, if the user wants to accentuate the balance of the user relevances, and consequently to ensure that each user preferences are considered, simply just needs to increase the parameter P.

$$rel_{k+1}(u,i,s_k) = \frac{rel_k(u,i)^{(\frac{\sum_{u \epsilon G} rel_k(u,s_k)}{|G|*rel_k(u,s_k)})^p}}{\sum_{i \epsilon I} rel_{k+1}(u,i)} \tag{5.1}$$

In practice, to try to observe this behaviour we are going to compute the Zero Recall and NDCG for different values of p:

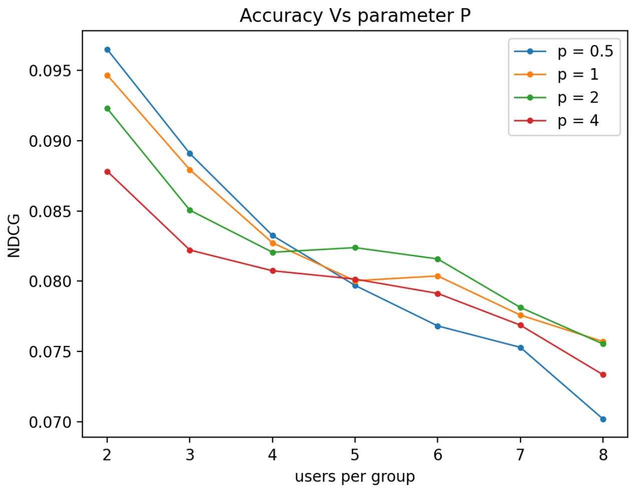

Figure 5.9: NDCG of ARM in function of parameter p

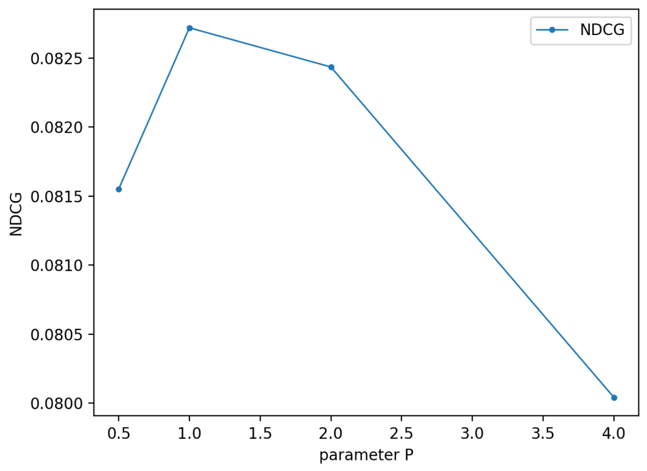

Figure 5.10: mean NDCG values in function of parameter p

In figure 5.9 we can observe how parameter P impacts the ARM modeling. Intuitively, reducing the balance of users relevances would lead into an approach more similar to the Additive Utilitarian and consequently recommending those items that the majority prefer, so mean ndcg should increase until a certain point for smaller values of P, whereas the proportion of users with relevant items will tend decrease as the P increases and vice versa.

However as it can be appreciated in figure 5.10 the maximum accuracy is obtained with p values close to 1, and from the results in figure 5.9, we can observe that the reduction of the mean ndcg scores for smaller P is due to the worsening of the ndcg scores for groups with larger sizes, since opting to recommend those items just considering the mean, may lead to select mediocre items, that are not disliked and neither significant for any member.
So, to increase the accuracy for larger groups, it is not enough to consider just the majorities preferences, it is necessary a good balance across members.

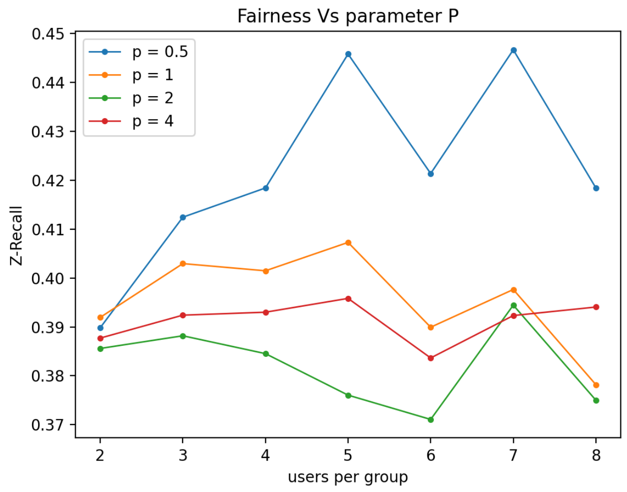

Figure 5.11: NDCG of ARM in function of parameter p

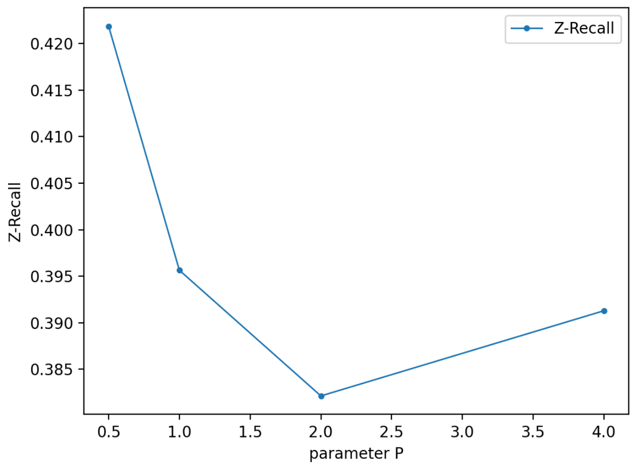

Figure 5.12: mean Zero Recall values in function of parameter p

Inversely, we expect to obtain higher fairness (Lower z-recall values) with higher P values, since the balance across members relevances will be accentuated, and consequently the group modelling strategy will consider more each member preferences.
However, in figure 5.12 we observe that the best fairness is not achieved by increasing the p infinitely, instead is achieved by P values closer to 2.

It is evident that to obtain a good accuracy is necessary a good fairness, e.g. it is impossible obtain a high ndcg if we have a Z-Recall equal to 1, however, as depicted in Figures 5.10 and 5.12 the trade off between fairness and accuracy exists, since it is possible to gain accuracy (increase in NDCG scores) at expenses of sacrificing fairness (decrease in NDCG scores). In particular, in our results we can observe how we can negotiate between these two by setting the parameter P closer to 1 or closer to

2. Therefore, besides observing the accuracy / fairness trade off, we can observe how the ARM group modeling strategy can be slightly modified in function of parameter P and better adapt to the group recommender system needs.

Moreover, we can already predict the ARM will perform better than techniques such as "Additive Utilitarian" which is similar to ARM's approach with $P = 1/\inf$ and "FAI" which is close to the ARM's strategy when $P = \inf$.

5.5 Our metric

Thanks to the scores obtained in the preceding section 5.4 display the fair-accuracy trade off, we can show if our metric proposal reflects correctly the accuracy and the fairness of the group modeling strategies. Therefore, since in the previous experiments the best accuracy was achieved in P values closer to 1 and the fairness with a P closer to 2, if the metric works well, it should output the best value for P values closer to 1 and 2.

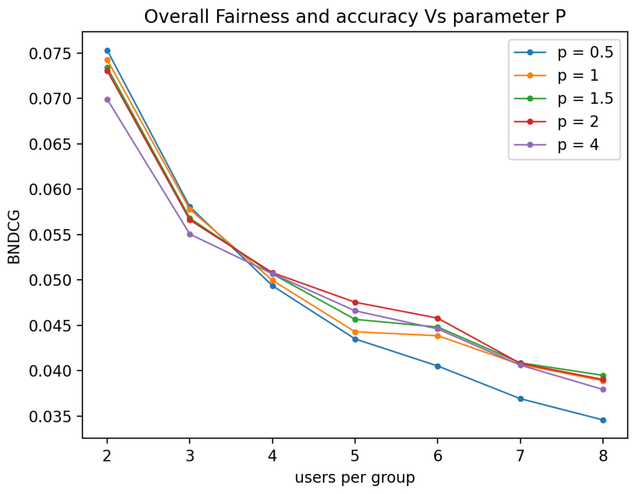

Figure 5.13: BNDCG of ARM in function of parameter p

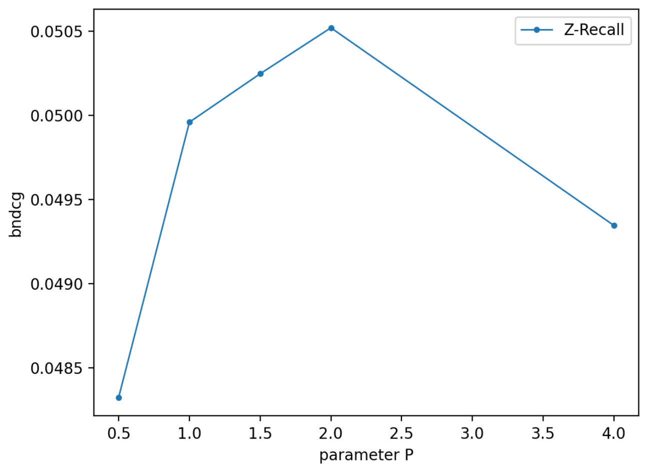

Figure 5.14: mean BNDCG values in function of parameter p

As depicted in Figures 5.14 and 5.13 we can observe multiple indicators that demonstrate the metric correctly evaluates both accuracy and fairness. First of all, as we advanced, a good metric should get the best value with a P within 1 and 2, which is the case, BNDCG obtained the best score for a P equal to 2. Secondly, if we now observe the mean Zero Recall achieved by ARM with parameter P = 2 and P = 4, we can appreciate how with P = 4 the ARM provided a little bit more fair recommendations than P = 2 5.12, and inversely, with parameter P = 2 the ARM obtained a significant higher ndcg score than with P = 4 5.12. Thus, a good metric should also output a higher value for P = 2 because was much better than with a P = 4 in terms of accuracy, but reduce the difference between both, because P = 4 provided a slightly better fair recommendations, which again is what BNDCG has done.
Observing the results, we consider that we can use the BNDCG to evaluate which technique is better in all aspects, however, as we have previously emphasized the objective of this metric is not the interpretability, but to understand whether the problem is in terms of fairness or in terms of accuracy it is better to use the other metrics.

5.6 ARM vs previous fair group modeling techniques

As we previously indicated the group modeling techniques, we are going to compare ARM with are: GFAR, FAI, Average (Additive Utilitarian 2.2), Greedylm (Greedy Least Miseray) and SPGreedy (Single Proportionality Greedy). We have cautiously selected these strategies because they have introduced a new novel concept or because they have simply shown good results in comparison with the others.

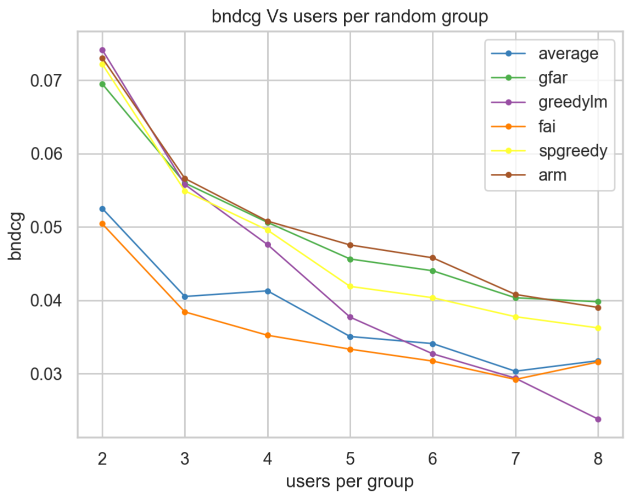

Figure 5.15: BNDCG scores for the different group modeling techniques

According to the BNDCG score, ARM is the best strategy in general, followed closely by GFAR and SPGreedy, and then by GreedyLM, Average and FAI

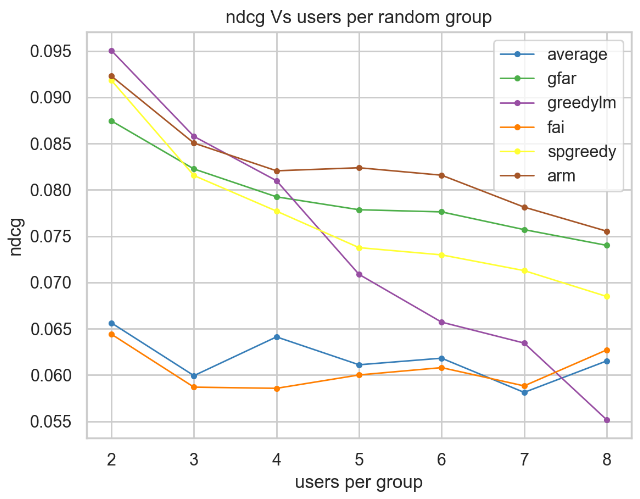

Figure 5.16: NDCG scores for the different group modeling techniques

Considering the NDCG punctuations, the ranking of strategies is the same, nevertheless, NDCG scores have shown a higher gap between ARM and the others

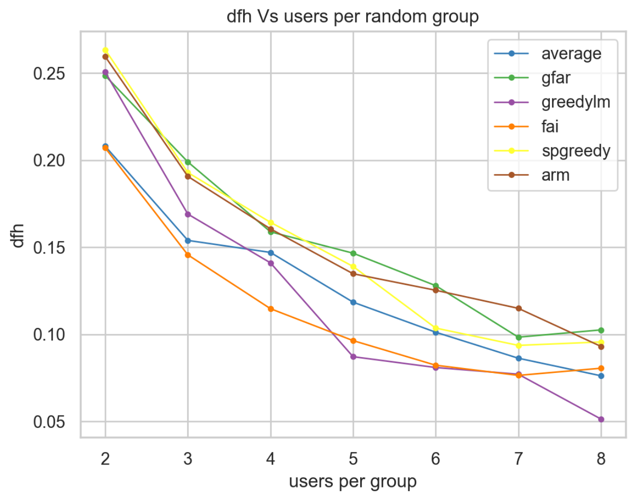

Figure 5.17: DFH scores for the different group modeling techniques

Since the DFH is less precise than NDCG, we can observe how the differences across strategies shrink, and now both ARM and GFAR are the best with almost the same scores. Besides, observing NDCG and DFH differences, we can consider that GFAR tends to add a relevant item to each user in the Top-N$_G$ as soon as ARM, however, from then on, ARM provide more fair and more accurate recommendations.

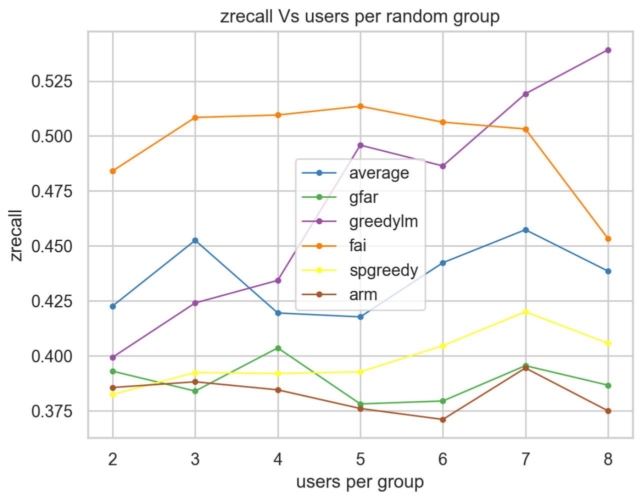

Figure 5.18: Zero Recall scores for the different group modeling techniques

And According to Z-Recall the best metric is again the ARM.
Summing up, the most unfair metrics and also the less accurate are FAI and Average. Then Greedylm, although it obtained the best scores for small groups, its performance rapidly decrease as the groups become larger. Finally, the best in all the aspects have been SPGreedy, GFAR and specially ARM. We attribute ARM's success to these aspects:

- First of all, ARM is one of the few novel strategies to use the relevances obtained by the recommender instead of assigning scores in function of the position they are found. Assigning scores to an item according to the position of it in the user preferences, may not respect the real importance of the item for each user. e.g. imagine that the relevances for User1 are: $rel(User1, Item1) = 0.9$, $rel(User1, Item2) = 0.35$, $rel(User1, Item1) = 0.32$ GFAR will convert them to $p(User1, Item1) = 0.66$ $p(User1, Item2) = 3$,$p(User1, Item1) = 0.0$ which is far from representing the real relevance of the items.
- Our strategy always maximize the overall happiness (since always count equally the opinion of an user). Oppositely, in the state of art have been commonly penalized or dismissed the user opinion according how significant was to previous recommendation to him.
- ARM is the first strategy to balance the probabilities of an user to obtain a significant recommendation to ensure fairness across members in a group, by shrinking or expanding the user relevances.
- Finally, ARM is also the first approach that does not balance according to how relevant were the previous recommendation, instead it balances based on how relevant were the precedent recommendations in comparison with the other users, and thus, ensure that the preferences across users have been taken into account equally.

To make sure the strategies hierarchy of the approaches keeps the same regardless of the nature of the groups (whether the members preferences are similar or not) we are going to observe the behaviour of these techniques for all the group building strategies defined in the experimental setup.

Metric	Group Building	Average	Gfar	Greedylm	Fai	Spgreedy	Arm
GNDCG	Distinct	0.037	0.0481	0.043	0.035	0.046	**0.0483**
Z-Recall	Distinct	0.429	0.391	0.471	492	0.404	**0.389**
NDCG	Distinct	0.060	0.077	0.073	0.059	0.074	**0.079**
DFH	Distinct	0.132	**0.158**	0.121	0.116	0.148	0.152
GNDCG	Random	0.037	0.049	0.043	0.035	0.047	**0.050**
Z-Recall	Random	0.435	0.388	0.471	0.496	0.398	**0.382**
NDCG	Random	0.061	0.079	0.073	0.060	0.076	**0.082**
DFH	Random	0.127	0.154	0.122	0.114	0.150	**0.154**
GNDCG	Similar	0.037	0.047	0.042	0.035	0.045	. **0.048**
Z-Recall	Similar	0.432	0.392	0.474	0.492	0.402	**0.382**
NDCG	Similar	0.061	0.076	0.073	0.059	0.074	**0.080**
DFH	Similar	0.132	**0.156**	0.124	0.123	0.155	0.154

Table 5.1: zRecall, DFH, NDCG scores for a fair example

In consideration of Adaptative Relevance-based Model have achieved the best mean score in the 0.83%(10/12) of the scenarios, we can conclude that the strategy is significantly better than the others to provide recommendations, that ensure fairness across the members in the group and yet improves the accuracy.

6. Conclusions

6.1 Summary of the thesis

The main steps conducted throughout this thesis are the followings:

1. A recommender system capable of providing individual recommendations with high accuracy given a set of users, ratings and movies that will serve as an effective baseline to the group recommendation process.
2. A group building strategy capable of generating groups of users with similar, random and distinct interests, in function of their preferences (expressed as ratings).
3. Replicas of the best fair group modeling strategies in order to benchmark our new approach.
4. A novel group modelling strategy that minimizes some of the most widely know metrics to assess recommendations, such as Zero Recall, Discounted First Hit, Normalized Discounted Cumulative Gain.
5. And finally, a new evaluation metric that assess in detail both, the accuracy of the recommendations and the fairness, between members of the same group.

6.2 Conclusions of the research

After analyzing in detail the results obtained and the observations made along the present study, it is possible to draw the following conclusions:

While developing this thesis, it has been possible to appreciate, evaluate and understand the advantages but also the deficiencies of the existent group modeling techniques, which lead to the construction of new objectives that served as a baseline for the "Adaptive Relevance Model" novel strategy. Some of these objectives, such as ensuring equivalent proportion and ranks of relevant items for each user, were adopted from these techniques because they guarantee that an user will not find unwanted items in top ranks if the previous recommendations have not been favorable to him. There are also innovative objectives that hasn't been taken into account previously, such as balancing the user preferences according to how favorable was the previous recommendation to a member in comparison to the others. Another important consideration that has been removed is the balance of the user preferences by penalizing or boosting the user right to vote for the new recommendation. We kept the user right to vote always constant, and balanced it simply by radicalizing or softening the user opinion. As a result, the proposed strategy based on these grounds has shown a better accuracy and fairness in comparison to the other group modeling approaches for the most widely used metrics, such as Normalized

Discounted Cumulative Gain, Discounted First Hit and Zero Recall.

Moreover, during the project the metrics that aim to asses the behaviour of a recommender have been also replicated and studied. Even though they can provide useful insights about the performance of a model, they still have inconveniences when evaluating fairness or/and accuracy. In some cases, this is due to the fact that the metric transformed the group score as the mean of the individual score of the members without assessing the balance across the scores, or simply because the metric itself is poor by nature, which is the case of Zero Recall that can not appraise the difference of a group where all users obtained 1 relevant item with a group where each user got 20 relevant items. To solve these drawbacks it has been proposed a metric that tends to the mean / maximize NDCG score when the variance across scores is low and tends to the minimize NDCG when the variance is high. Thus, it makes it capable to assess both fairness and accuracy of a group modeling strategy and determine which is better without the necessity of using a supplementary metric.

6.3 Future lines of work

In this project we have been focused on balancing the user relevance as a function of "How good the prior recommendation was for a member compared to the others" because we considered that that was the best way to do it, but there are still some interesting approaches that could provide fair recommendations to a group. One of them can be balancing the user relevances according to the number of unrelated items in Top-N$_G$ since the last relevant item.

Another interesting aspect to study could be modifying the parameter P in function of the size of the groups. In our study , as it has been presented in 5.14, the overall accuracy and fairness for small groups was higher for lower values of P, while for large groups the best results were obtained for higher values of P.

Also, when the recommender system has to retrieve a large number of recommendations, ARM first tries to recommend convergent items, that most of the users likes, but in a certain point have to begin recommending divergent items that just a small set of users like, when this occurs, the differences across members relevances increase exponentially which leads to accentuate the behaviour of the ARM too much, so this effect could be also somehow an interesting bound.

Finally, a strong suggestion for the following studies is to consider storing as much data as possible regarding the structure of the groups and its preferences, since discrepancies may appear on the results obtained with the real data, and the simulations with synthetic groups.

Bibliography

[1] Linas Baltrunas, Tadas Makcinskas, and Francesco Ricci. Group recommendations with rank aggregation and collaborative filtering. In *RecSys'10 - Proceedings of the 4th ACM Conference on Recommender Systems*, pages 119–126, New York, New York, USA, 2010. ACM Press.

[2] Linas Baltrunas, Tadas Makcinskas, and Francesco Ricci. Group recommendations with rank aggregation and collaborative filtering. In Xavier Amatriain, Marc Torrens, Paul Resnick, and Markus Zanker, editors, *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26-30, 2010*, pages 119–126. ACM, 2010.

[3] Ludovico Boratto and Salvatore Carta. The rating prediction task in a group recommender system that automatically detects groups: architectures, algorithms, and performance evaluation. *J. Intell. Inf. Syst.*, 45(2):221–245, 2015.

[4] Ingrid A. Christensen and Silvia Schiaffino. Entertainment recommender systems for group of users. *Expert Systems with Applications*, 38(11):14127–14135, oct 2011.

[5] Alexander Felfernig, Ludovico Boratto, Martin Stettinger, and Marko Tkalčič. *Group Recommender Systems*. SpringerBriefs in Electrical and Computer Engineering. Springer International Publishing, Cham, 2018.

[6] Alexander Felfernig, Christoph Zehentner, Gerald Ninaus, Harald Grabner, Walid Maalej, Dennis Pagano, Leopold Weninger, and Florian Reinfrank. Group decision support for requirements negotiation. In Liliana Ardissono and Tsvi Kuflik, editors, *Advances in User Modeling - UMAP 2011 Workshops, Girona, Spain, July 11-15, 2011, Revised Selected Papers*, volume 7138 of *Lecture Notes in Computer Science*, pages 105–116. Springer, 2011.

[7] Aurélien Géron. *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition*. 2019.

[8] Sara Hajian, Francesco Bonchi, and Carlos Castillo. Algorithmic bias: From discrimination discovery to fairness-aware data mining. In Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 2125–2126. ACM, 2016.

[9] Daniel Herzog and Wolfgang Wörndl. A user study on groups interacting with tourist trip recommender systems in public spaces. In George Angelos Papadopoulos, George Samaras, Stephan Weibelzahl, Dietmar Jannach, and Olga C. Santos, editors, *Proceedings of the 27th ACM Conference on User Modeling, Adaptation and Personalization, UMAP 2019, Larnaca, Cyprus, June 9-12, 2019*, pages 130–138. ACM, 2019.

[10] Anthony Jameson and Barry Smyth. Recommendation to groups. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 4321 LNCS, pages 596–627. Springer Verlag, 2007.

[11] Mesut Kaya, Derek G. Bridge, and Nava Tintarev. Ensuring fairness in group recommendations by rank-sensitive balancing of relevance. In Rodrygo L. T. Santos, Leandro Balby Marinho, Elizabeth M. Daly, Li Chen, Kim Falk, Noam Koenigstein, and Edleno Silva de Moura, editors, *RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26, 2020*, pages 101–110. ACM, 2020.

[12] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H. Chi. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1930–1939. Association for Computing Machinery, jul 2018.

[13] Mark O'Connor, Dan Cosley, Joseph A. Konstan, and John Riedl. Polylens: A recommender system for groups of user. In Wolfgang Prinz, Matthias Jarke, Yvonne Rogers, Kjeld Schmidt, and Volker Wulf, editors, *Proceedings of the Seventh European Conference on Computer Supported Cooperative Work, 16-20 September 2001, Bonn, Germany*, pages 199–218. Kluwer, 2001.

[14] Dimitris Sacharidis. Top-n group recommendations with fairness. In *Proceedings of the ACM Symposium on Applied Computing*, volume Part F1477, pages 1663–1670, New York, NY, USA, apr 2019. Association for Computing Machinery.

[15] Christophe Senot, Dimitre Kostadinov, Makram Bouzid, Jérôme Picault, Armen Aghasaryan, and Cédric Bernier. Analysis of strategies for building group profiles. In Paul De Bra, Alfred Kobsa, and David N. Chin, editors, *User Modeling, Adaptation, and Personalization, 18th International Conference, UMAP 2010, Big Island, HI, USA, June 20-24, 2010. Proceedings*, volume 6075 of *Lecture Notes in Computer Science*, pages 40–51. Springer, 2010.

[16] Dimitris Serbos, Shuyao Qi, Nikos Mamoulis, Evaggelia Pitoura, and Panayiotis Tsaparas. Fairness in package-to-group recommendations. In *26th International World Wide Web Conference, WWW 2017*, pages 371–379. International World Wide Web Conferences Steering Committee, 2017.

[17] Thiago Silveira, Min Zhang, Xiao Lin, Yiqun Liu, and Shaoping Ma. How good your recommender system is? A survey on evaluations in recommendation. *International Journal of Machine Learning and Cybernetics*, 10(5):813–831, may 2019.

[18] Maik Thiele, Wolfgang Lehner, and Gunnar Schröder. Setting Goals and Choosing Metrics for Recommender System Evaluations Self-tuning BI Systems View project Setting Goals and Choosing Metrics for Recommender System Evaluations. Technical report, 2011.

[19] Koen Verstrepen and Bart Goethals. Top-n recommendation for shared accounts. In Hannes Werthner, Markus Zanker, Jennifer Golbeck, and Giovanni Semeraro, editors, *Proceedings of the 9th ACM Conference on Recommender Systems, RecSys 2015, Vienna, Austria, September 16-20, 2015*, pages 59–66. ACM, 2015.

[20] Lin Xiao, Zhang Min, Zhang Yongfeng, Gu Zhaoquan, Liu Yiqun, and Ma Shaoping. Fairness-Aware Group Recommendation with Pareto-Efficiency. 2017.