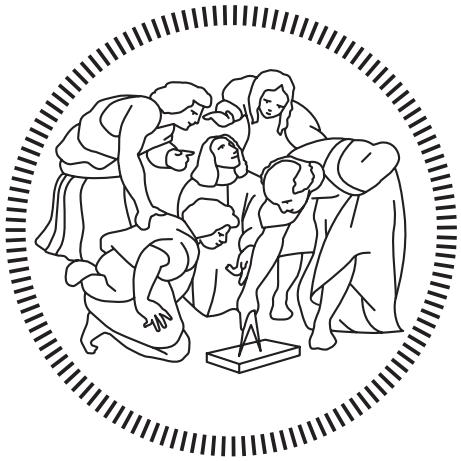
Politecnico di Milano

SCUOLA DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE
Laurea Magistrale – Computer Science Engineering



Which fairness measures for group recommendations?

Advisor
Dr. Letizia TANCA

Co-Advisor
Dr. Davide AZZALINI

Candidates
Andrea BOZZONI - 920319
Christian CAMMARERI - 920915

Academic Year 2019 – 2020

Ringraziamenti

Prima di procedere con la trattazione, vorremmo dedicare qualche riga a tutti coloro che ci sono stati vicini in questo percorso di crescita personale e professionale.
Il primissimo grazie va a noi stessi, che finalmente siamo arrivati all'ultimo giro di boa (vedi figura 1).

Figure 1. Il capitano Max "Acciuga" Allegri

Un ringraziamento particolare va al nostro relatore, prof.ssa Letizia Tanca, che ci ha seguito, con la sua infinita disponibilità, dandoci fiducia e la possibilità di sviluppare questa tesi.
Grazie anche al nostro correlatore Davide per i suoi preziosi consigli e per averci suggerito costantemente le giuste modifiche da apportare alla nostra tesi.

Bozzoni Andrea

E' sempre difficile mettersi davanti al computer con l'idea di dover ringraziare tutte le persone che hanno condiviso il loro tempo con me durante questo percorso che è durato più di cinque anni. Spero di non dimenticare nessuno.
Grazie, innanzitutto, ai professori che durante il mio percorso, non solo universitario, hanno contribuito a fare crescere la mia passione e curiosità rispetto alle loro materie di insegnamento. Il loro lavoro credo sia indispensabile più che mai oggi, e troppo spesso viene screditato.
Grazie a tutti i compagni di corso, che anche se le nostre strade si sono incrociate per poco.
Grazie specialmente a Fabio, Eva, Giacomo compagni frequenti di progetti e di pranzi disperati in L26 a base di pasta asciutta, ma in brodo.
Grazie a tutti i miei compagni di squadra della Polisportiva San Nicolò, con cui ho condiviso vittorie e sconfitte, ma soprattutto importanti mangiate post-partita e post-allenamento. Sono stati probabilmente gli anni in cui mi sono divertito di più a giocare. In special modo grazie a Fabio Bastiani che allena con passione e amore verso i suoi ragazzi e verso la pallavolo.
Grazie a Yuri e Guglie per le importantissime partite di calcetto o ping-pong e per i viaggi a recuperare la palla calciata con delicatezza nella troposfera.
Grazie Garla oltre che per tutti viaggi per andare ad allenamento, per la tua musica, per le partite di ping-pong, per le (troppo poche) partite di beach volley e per i grandi progetti sulle società sportive.
Grazie a tutto il gruppo scout Trebbia 1, a tutti i membri della comunità capi e a tutti i ragazzi. Ho dedicato un numero enorme di ore in questa associazione perche quello che mi lascia ha sicuramente valore inestimabile.
Grazie a Christian, Sergio e Giampietro per tutte le volte in cui mi hanno ospitato al Pestalozza center e per le partite a Tekken o Fifa dopo pranzo al posto di studiare FLC, Distributed Systems o Database 2.
Grazie a Christian per questo lavoro fatto insieme, per le foto del mare che mi mandavi quando da me pioveva.
Grazie a tutti i miei amici di sempre: Rose, Paolo, Vala, Ceci, Anto, Silvia, Chiara. Grazie per le vacanze, le uscite e tutti i bei momenti passati insieme che hanno reso tutto più facile. So che su di voi posso sempre contare.
Infine grazie alla mia famiglia: ai miei zii vicini e lontani che si ricordano sempre di me.
Grazie ai nonni che ci sono e ci sono stati, per i pranzi che mi hanno preparato e il tempo che hanno speso per me.
Grazie a mia mamma, mio papà e mio fratello, che mi hanno sempre supportato, incoraggiato e hanno sempre cercato di caricarsi parte della mia fatica.
Grazie, la mia felicità di oggi la voglio condividere e dedicare a tutti voi.

Cammareri Christian

Vorrei dedicare qualche riga a coloro che hanno contribuito alla realizzazione della mia tesi di laurea e, in generale, al mio percorso universitario iniziato oltre cinque anni fa al Politecnico di Milano. Spero di non dimenticare nessuno e, se per caso dovesse accadere, abbiate la bontà di perdonarmi.
Il primo grazie va a tutta la mia famiglia, che in tutti questi anni mi ha continuamente incoraggiato e motivato, ed in particolare ai miei genitori, che mi hanno permesso di intraprendere questo viaggio e che mi hanno sempre supportato sia emotivamente che economicamente.
Grazie anche al Politecnico per avermi fatto crescere in questi anni: è stata una formazione universitaria, quindi grazie a tutti i professori che mi hanno istruito, ma anche una formazione fisica non desiderata e/o aspettata (un no-grazie molto sentito va alle aule Natta e Spaccaculi in particolare).
Grazie alla mitica e unica via Pestalozza e a tutte le emozioni che mi ha regalato: mancherai sempre. Penso sia impossibile stimare il numero delle persone belle e brutte passate da lì in questi anni, ma se sono arrivato fino a qui in qualche modo è anche merito di quell'appartamento/bar/sala studio/sala giochi/ristorante/circolo ricreativo.
Un grazie necessario e doveroso va alla gang universitaria con cui ho condiviso gioie e dolori in questi anni, con cui ho passato giornate di cazzeggio alternate a giornate di studio pazzo, giornate che mai rimpiangerò: grazie a Sergio, Berna, Bura, Anna, Marco C., Giampi, Batta, Eva, Leo, Marco Bon. e tutti gli altri. E scusatemi se "qualche volta" mi sono addormentato in aula e poi vi è toccato spiegarmi le cose.
Un grazie molto sentito va a tutti quei colleghi che, volontariamente o costretti, almeno una volta sono venuti a giocare a calcetto al Giuriati: ogni volta sembrava impossibile trovare il decimo, ma ce l'abbiamo sempre fatta.
Ringrazio in particolar modo Bozzo, con cui ho scritto questa tesi in questi mesi: perdonami per le foto del mare, quando vuoi venirmi a trovare per vederlo dal vivo io sarò molto felice di ospitarti.
Un grazie anche a tutti i miei amici "di giù" con cui ho passato giornate pazze, in particolar modo nell'ultimo anno: Walter, Alberto, Ema, Mic, Dodo, Delia, Sofia, Marco, Alessio, Valerio. Visto che me lo chiedevate in continuazione adesso vi posso rispondere: forse finalmente mi laureo.
L' ultimo grazie, il più importante e davvero di cuore, l'ho voluto lasciare per Ale: dire che è stata una persona importante per me è riduttivo e se sto scrivendo questi ringraziamenti su una tesi per laurearmi al Politecnico di Milano è anche merito suo. Spero di non averti rotto troppo le scatole ultimamente, ma se così fosse mi farò perdonare il prima possibile con l'ennesima grigliatina.

Sommario

Questa tesi ha due obiettivi specifici: il primo è cercare di migliorare le prestazioni dei sistemi di raccomandazione da un punto di vista etico per gruppi di persone composte da utenti sconosciuti tra loro; il secondo consiste nel proporre nuove metriche per misurare l'equità delle raccomandazioni stesse.
Il punto di partenza della tesi è stato il progetto Fair, context-AwaRe Group RecOmmender (FARGO) (Quintarelli et al. [32]), un algoritmo che fa leva sui concetti di *influenza* dell'utente e *context awareness* per fornire raccomandazioni il più eque possibile.
Al fine di migliorare l'*equità*, vengono proposti due miglioramenti per FARGO: "*skyline filtering*" e "*stima degli zeri*". I risultati di questi miglioramenti vengono quindi presentati per valutare l'equità delle raccomandazioni utilizzando due data-sets specifici: un data-set molto grande contenente i dati di Auditel e un piccolo caso di studio: il data-set Musica.
Le performance vengono valutate utilizzando metriche comuni per i sistemi di raccomandazione di gruppo, come *Recall*, che quantifica la bontà delle raccomandazioni, e *Score disparity* e *Recommendation disparity* che quantificano quanto siano eque tali raccomandazioni. I nostri esperimenti mostrano alcuni miglioramenti nel Recall e nella Recommendation disparity; ma emerge anche che le sopracitate metriche che valutano l'equità presentano alcuni limiti.
Di conseguenza, questa tesi studia e propone nuove misure per valutare l'equità che superino tali limiti. Le misure tradizionali e quelle nuove vengono quindi messe a confronto. Ciò che emerge è che non esiste una misura di equità universale ottimale per i sistemi di raccomandazione di gruppo e che, soprattutto, è necessario prestare molta attenzione al dominio di applicazione specifico e al caso di studio per scegliere le metriche più appropriate.

Keywords: metriche per recommender system di gruppo, recommender system di gruppo, recommender systems context-aware, fairness, etica, equità

Abstract

This thesis has two specific objectives: the first one is trying to improve the performance of recommeder systems from an ethical point of view, for groups of people composed by users unknown to each other; the second one consists in proposing new metrics to measure the fairness of the recommendations themselves.
The starting point of the thesis was the Fair, context-AwaRe Group RecOmmender (FARGO) project (started by Quintarelli et al. [32]), an algorithm that leverages the concepts of user *influence* and context awareness to provide recommendations as fair as possible.
In order to ameliorate *fairness*, two improvements are proposed over FARGO: "*skyline filtering*" and "*zeros estimation*". The outcomes of these improvements are then presented to assess the equity of the recommendations by using of two specific data-sets: a very big data-set containing data from Auditel, and a little case study: the Music data-set.
The performances are evaluated using metrics common for group recommender systems, such as *Recall*, which quantifies the goodness of the recommendations, and *Score disparity* and *Recommendation disparity* which quantify how fair those recommendations are. Our experiments show some improvements in both Recall and Recommendation disparity; but it also emerges that the aforementioned metrics that assess fairness have some limitations.
As a consequence, this thesis studies and proposes new measures to evaluate fairness which overcome such limitations. Traditional measures and the new ones are then compared. What emerges is that there is no optimal universal fairness measure for group recommender systems and that, above all, it is necessary to pay close attention to the specific application domain and case study to choose the most appropriate ones.

Keywords: group recommender system metrics, group recommender systems, context-aware recommender systems, fairness, ethics, equity

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter we are going to introduce to the thesis topics.

1.1 Introduction to the problem

The problem of customizing users' web experience has exploded in the last years due to the constant growth of any kind of content available. Every big website such as e-commerce, streaming platforms, social networks have reached thousands or even millions of different items available. So, in the last 20 years it has emerged the necessity of building systems that can address the user to the content that he/she might like or he might be interested in. This kind of systems are called recommender system.
Recommender systems are special software where sources of data related to previous users' activities are combined, elaborated and analyzed in order to predict which items among the whole set of items could fit a specific user's preferences. Since the last decade of the century, recommender system gained huge popularity and became a very hot topic within universities as a research area. Many big companies started to invest in these systems too: Amazon was being one of the first. They developed a very powerful collaborative filtering system capable to suggest Amazon's users items they would have bought. Another important company that contributed to the development of recommender system is Netflix. Starting from 2006 this famous streaming company sponsored a competition with a prize of USD 1,000,000. This big prize would have been given to the team which would be able to improve the original Netflix recommendations by 10% over 100 million movie ratings.
Nowadays, recommender systems are employed in many real life activities such as watching a film, going to the restaurant, going to a pub. They aim of these systems is to help the user in filtering only the content he could like, and then, help him in choosing. For example to decide which film to watch or where to go for a dinner.
However, many of these activities are intrinsically group-based, they are performed in group of people. So, the problem of traditional recommendation to single users shifts to rather different problem, that is group recommendations. Performing group recommendations requires an higher effort with respect to the classical single-user recommendation, in fact we have to deal with heterogeneous tastes and preferences, and combine all them to reach a unique result. Again, users could behave differently when they are in group with respect to when they are alone or thy could be influenced by other group members. Eventually, finding an agreement among all the group members could be impossible sometimes. All these factors make group recommendations a real challenge.
In this thesis we focus on the group recommendation problem addressed to ephemeral groups. Ephemeral groups are composed by people that might never been together before. In this case we cannot rely on past information because the group is not a stable entity. We have to somehow combine at run-time each user's tastes and come out with some recommendation that can be suitable for all group members. However, some users show different behavior when are part of a group, and we have to keep into account this factor. In particular there could be several aspects that impact on the individual preference such as the context, the relationships among the members and the personality. In other words when we are aggregating the preferences we should give different weight to every user, so we can formalize some kind of group dynamics in a mathematical model. In our case, the previous work used the influence as a weight to determine which users' preferences should have greater importance in order

to model the group decision dynamics.
As a consequence of this type of modeling we spot a fairness and ethics problem. We think that different user preferences should not have the same importance. Applying this in a trivial way could lead to situations in which some users in the group are very satisfied by our recommendations and others that are not and eventually some users' preferences could not be taken in consideration at all. Considering the fact that traditional recommender systems aim to maximize the recall or other individual measures, the straightforward application of the same rules on a group recommender system can introduce situations where some users are discriminated in favor of other arrogant users. Thus, it is important to introduce some indicators and factors that help us provide ethical recommendations . We exploited our fairness element called "consensus" and some mathematical optimization techniques to provide both good (in terms of users' preferences) and fair (in terms of agreement and consensus) items. We then performed experiments on a large real-world dataset containing the personal and group choices regarding TV programs of 7,921 users for three months and on a case study dataset, gathering musical preferences from 280 real users. Our aim is to verify which is the best way to improve the ethics metrics without degrading the recall of the system. Of course we will have to deal with a trade-off among performance and fairness. We also analyze the current fairness measures and we highlight pros and cons of them and finally we propose a better measure that keeps in consideration the influence of each user.
In this thesis we:

- Consider fairness as a first-class criterion to be taken into account and introduce it into the computation of the recommendation;
- Propose a new method for selecting the optimal group of items to be recommended without scanning all the available items;
- Make a comparison against other fair aggregation methods.

1.2 Thesis structure

The following chapters are structured as follows:

- **Chapter 2**, that reports the current state of the art for what concerns our topic;
- **Chapter 3**, that contains a detailed explanation of FARGO method;
- **Chapter 4**, that contains the methods we introduced;
- **Chapter 5**, where we report all our experiment and summarize some results;
- **Chapter 6**, in this chapter we take some experiment and comparisons for evaluating the goodness and the characteristics of our new measures;
- **Chapter 7**, we take the conclusions of our work.

Chapter 2

Related Work

In this chapter we are introducing the main existing techniques for providing recommendations to groups of people. In particular, we focus on current trends in ethics for group recommender systems. We started by reviewing the metrics used to test these methods, and then, we describe also the main competitor algorithms that we used to compare our approach.

2.1 Recommender systems

After the advent of social networks, e-commerce and streaming platforms, the amount of content offered to the user has exploded. Nowadays there is a huge variety of contents among which a user can choose his/her favorites. Among the various possibilities, it is known that users usually exploit multi-medial platforms to watch movies or TV series, listen to music,connect with other people, shopping online but also look for job opportunities or book holidays. The aim of a recommender system is to personalize the content presented to a specific user, in order to help him in finding items in which he might be interested [17].These systems are very useful for example in order to show to the user items correlated to his previous research or in advertisement placing strategies. Recommender systems can be divided in families, based on the way they produce the recommendation. In Figure 2.1 the main different families are reported [33].

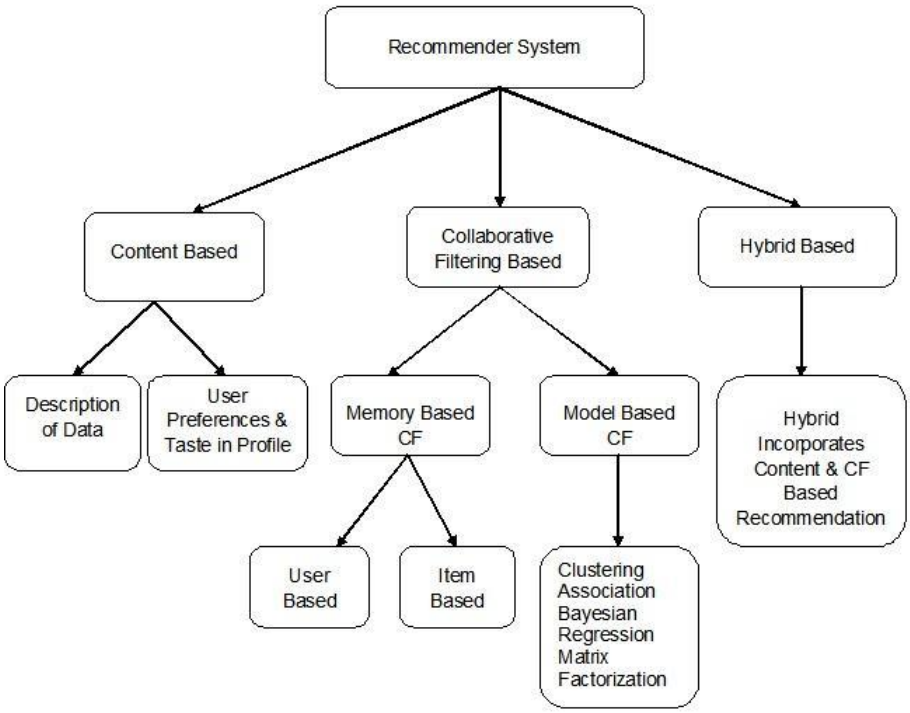


Figure 2.1. Recommender systems taxonomy

- Content based filtering: this group of methods tries to recommend to the users items that contains properties that the user likes. For example, if a user likes action movies these methods will suggest him/her some movies that belong to this category.
- Collaborative filtering: In this family of recommenders it is possible to distinguish memory based collaborative filtering and Model based collaborative filtering. Memory based collaborative filtering recommends to the user items that are

somehow similar to other items that the user has already interacted with or items that other users, similar to the one of interest, interacted with.
Model based collaborative filtering instead uses machine learning and statistical techniques to learn a model based on the status of interaction between users and items.

- Hybrid based filtering: this method incorporates content and collaborative filtering recommendation. In this case the benefits produced by all the previous types of recommenders are combined in order to boost performances.

2.2 Group recommendation

Group recommender systems are a specific category of recommender systems that compute recommendations targeted to groups instead of individuals. It is known that many activities are carried out by group of people instead of individuals by themselves, hence the group recommendation problem has become extremely relevant. As mentioned before, the main fields in which recommendation systems are applied (watching TV, listening to music, book a holiday and so on) often involve multiple users at the same time. Therefore, it is important to take into consideration many people interests and preferences in order to recommend the correct items to the group. The group recommender system problem is formalized in the following way:

$$Recommendation(G, I) = arg \max_{i \in I} prediction(G, i_k).$$

Many techniques have been discussed to deal with this problem.
Group recommendation mainly can be split in two types, based on how the group is composed [2]:

- Stable groups
- Ephemeral groups

Stable groups are composed of people who usually spend time together, as a consequence it is possible to consider the components of the group as an unique user. In the case of stable groups we have several information about the behavior of the group in the past: we have the possibility to check which items the group chose. In other words, in stable groups, since there had been other past occasions in which the group met, the previous choices of the group are known, hence it is possible to treat the group as a single user.
In this thesis we focus on the second type of groups: the ephemeral groups. These groups are composed by people who have never created a group together before. When we deal with ephemeral groups we are under the hypothesis that the past activities of the group are unknown. This implies that we are relying only on single users activities in order to give recommendations. Basing the recommendation on the single users preferences, the main aim of the group recommender systems, in the case of ephemeral groups, is to satisfy as more users as possible, even if they could not have the same interests.

Generally, group recommendation methods can be divided into two categories:

- Preference Aggregation
- Score Aggregation

The main techniques for recommending items to ephemeral groups are based on aggregation of personal preferences.
With preference aggregation methods we start from the individual preference list of each user and then we try to combine the lists together, in order to obtain a global list

of preferences. Instead, in the case of score aggregation methods we try to combine each user preference to get the group score of an item. Various aggregation functions have been proposed for preference aggregation, most of them come from the social choice theory.
Some aggregation function used are:

- Average (AVG): compute the arithmetic mean among the users' preferences [3, 8, 9, 13, 20, 26–28, 45]
- Maximum Satisfaction (MS): with which the chosen items are those for which the greatest value among the preferences of the group members is the highest [16, 20, 26–28]
- Least Misery (LM): tries to maximize the minimum preference of the group such as in [2, 9, 13, 16, 20, 24–28, 31]
- Expertise (EXP): in this case users' preferences are weighted with the expertise the individual user gets with the time [10, 11, 30]

The problem of the choice of aggregation function has been also treated in [13] and in [16], where it is reported the problem of grouping person and which strategy is the better to use based on how the group is composed. Other studies try to tackle the group recommendation problem with game theory and voting theory. Some of these uses non-cooperative game theory to find equilibrium items and some other uses voting strategies to elect the chosen items [12, 15, 22, 23, 29]. A survey on group recommender systems can be found in [40]

2.2.1 Context aware group recommendation systems

As stated in [6], the context is perceived as a set of variables that may be of interest for an agent and that influence its actions. Context can be identified as the set of information useful to characterize a situation. Context-aware computing was first defined by Schilit and Theimer [1] in 1994 as software that "adapts according to its location of use, the collection of nearby people and objects, as well as changes to those objects over time". Context in recommender systems can increase a lot the performance of the system [14, 21, 32, 41]. Context, thus, introduces a new dimension on which it's possible to optimize recommendations and play with to increase fairness and allows to be more precise in evaluating the users' preference. Context is a very important variable the drives the choice of the item to recommend can vary a lot depending on the time of the day (e.g. morning, afternoon, night), depending on the day of week (e.g. weekday or weekend) or even depending on the weather and on the location.
In this work, the dimensions *Time slot* and *Day of the week* have been used to characterize the context.
More details about context sub-dimensions are discussed in 5.1.1

2.3 Fairness in group recommendation

In traditional recommender systems unfairness is detected by verifying whether some category of users are discriminated. In a real-life scenario there are several categories under which users are subject to treatment disparity such as the gender, the social belonging or the ethnicity. However, in our case groups are composed by few people, that very often are friends or relatives, so our fairness concerns focus on different aspects. We consider the recommendations to be fair if each user of the group is satisfied with respect to his singular preferences. In other words, we aim to give recommendations trying to make everybody happy and avoiding to propose items that tend to enlarge the gap of happiness among users in the group.
Nowadays it is becoming of higher importance for a recommender system to be able to provide fair recommendations. The aim of a single person recommender is to maximize personalization. When we move to a group scenario the ability of the recommender system to be *fair* starts to gain importance. This problem seems to be a contradiction, because we must try to make each singular user the more satisfied as possible, but on the other hand we are trying to make everyone not too much unhappy with respect to the other people. The challenge here is to modify the standard methods of recommender systems to be able to be *fair*: that means that sometimes we must go deliberately against some users' preferences, in order to recommend items that overall satisfy the group. This is completely countercurrent with respect to the current trend of recommender system.
Different aspects of data equity are studied in [44], where authors highlight how inequity can came from different stages of data manipulation such as: data representation, feature availability, data access and outcome of data analysis. There are several way to assess fairness and ethics in group recommender systems. Some works introduce new metrics to specifically measure the degree of fairness of the system. For example measuring consensus among a group of people was studied in [37] where the pairwise difference among each users' rating is computed. Also a definition of fairness can be found in [37] computed as the ratio between the number of evaluation above a threshold over the same item and the cardinality of the group itself.
There are then also ethical recommendation strategies that come straight from group recommender systems since some algorithms are intrinsically fair. The following group recommender systems are built to be fair by their nature:

- Average AVG: compute the average of the scores to give a fair score to the item. AVG used in many works [5, 9];
- Least misery LM: aims to maximize the minimum score for the chosen items. It's used in [5, 16, 40].

The following other methods instead have been proposed specifically for adding ethics to group recommender systems:

- Envy-Freeness (ENVY-FREE): tries to optimize the position of the chosen item in each users' $TopK$ [34, 37];
- Fairness proportionality (FAIR-PROP): tries to optimize the number of items that each user has in common with the items recommended to the group [34];

- Fairness (FAIR-LIN): uses scalarization to solve Pareto optimization as concept of fairness [35];
- Disagreement (DIS): tries to optimize the disagreement among the users [5];

The authors in [5] introduce an aggregation function which tries to maximize the satisfaction of the group components, while, at the same time tries to minimize the disagreement among them. In [35] a solution based on the Pareto optimal solutions is proposed, where a multi-objective optimization is carried out to guarantee that all the users are satisfied. Another approach proposed in [39] is to consider the fairness based on the objects and interests liked by users, while the work proposed in [36] tries to take into account the unfairness coming from biased data, that lead to minor users' discrimination within the group. Recently a new approach was proposed in [43] in which a novel recommendation scheme combines a probabilistic model with coalition game strategy, to ensure the accuracy and fairness between groups of users. The latest work related to fairness in group recommender system proposed in [42] starts from the borda relevance definition in [43] in combination with some information retrieval probabilistic methods to diversify the results of the recommendations.

2.4 Evaluation metrics

The evaluation of classical recommender systems is based on some concepts that classify items in different sets, based on the actual choice of the user. In table 2.1 the used items represent the set of items the user interacted with, while the unused items are the set of items that the user did not interact with.

Table 2.1. Items classification

	Recommended	Not Recommended
Used	True Positive (TP)	False Negative (FN)
Not Used	False Positive (FP)	True Negative (TN)

In particular the elements present in table 2.1 are:

- TP, that is the cardinality of the set of items that we recommended to the user and he/she actually interacts with them;
- FP, that is the cardinality of the set of items the we recommended to the user, but he/she did not interact with them;
- FN, that is the cardinality of the set of items the we did not recommend to the user, but he/she interacted with them;
- TN, that is the cardinality of the set of items the we did not recommend to the user and he/she did not interact with them.

2.4.1 Recommendation metrics

Recall is defined as the number of relevant items recommended divided by the number of total items that the users interacted with

$$Recall = \frac{TP}{TP + FN} \tag{2.1}$$

In particular we use the $Recall@K$ defined as

$$Recall@K = \frac{|i \in TestSet : i_i \in TopK(i_G, i_c, i_t)|}{|i \in TestSet|} \tag{2.2}$$

Recall is a metric that tells us how many relevant items we suggested over the items that the user likes. In our case, we are trying to make *fair* recommendations, so we will keep in mind recall outcomes, but we are not considering them as the main metric to be optimized in our problem. This is due to the fact that the data-set we are dealing with is taken from a real-life scenario, where groups of users are not supposed to choose items based on some kind of "consensus" strategies, but they choose that item for several different reasons.

2.4.2 Fairness metrics

In this section we introduce some metrics that we used to verify the fairness of our method. These metrics are taken from [38].
User satisfaction $\mathcal{A}(u, c, t)$ is a formalization to verify how much an user u gains after the group recommender systems recommends the $TopK(G, c, t)$ items

$$\mathcal{A}(u, c, t) = \frac{\sum_{j \in TopK(G,c,t)} score(u, j, c)}{\sum_{j \in TopK(u,c,t)} score(u, j, c)} \tag{2.3}$$

Score disparity $D_S(G, c, t)$ is the Gini coefficient of user satisfaction (i.e. how much difference there is between users satisfactions)

$$D_S(G, c, t) = \frac{\sum_{u_1,u_2 \in G} |\mathcal{A}(u_1, c, t) - \mathcal{A}(u_2, c, t)|}{2n \sum_{u \in G} \mathcal{A}(u, c, t)} \tag{2.4}$$

User similarity tells us how similar is each user's $TopK$ compared to the group $TopK$ (i.e. how many items recommended to the group are also in user u $TopK$)

$$sim(u, c, t) = \frac{|TopK(G, c, t) \cap TopK(u, c, t)|}{K} \tag{2.5}$$

Recommendation disparity then is computed as the Gini coefficient of the users's similarity. This metric highlights how much disparity there is among the number of times an item is chosen by some users' lists.

$$D_R(G, c, t) = \frac{\sum_{u_1,u_2 \in G} |sim(u_1, c, t) - sim(u_2, c, t)|}{2n \sum_{u \in G} sim(u, c, t)} \tag{2.6}$$

Chapter 3

Background

In this chapter we introduce the reader to the main techniques used to deal with the problem of group recommendation and in particular to fair group recommendations to ephemeral groups.
We give a brief overview about the techniques and methods that we have exploited in our work. We introduce the already know methods from [32] from which we started to build our work.

3.1 Pareto efficiency

Pareto efficiency is a technique used in multi-variable optimization to select the best items in a set. Pareto optimality is a condition defined as follows:

- Given an initial situation, a Pareto improvement is a new situation where some agents will gain, and no users will lose;
- A situation is called Pareto dominated if there exists a possible Pareto improvement;
- A situation is called Pareto optimal or Pareto efficient if no change could lead to improved satisfaction for some user without some other user losing or if there's no scope for further Pareto improvement.

More formally, we have that given two points (n-dimensional arrays) $a = (a_1, ..., a_n)$, $b = (b_1, ..., b_n)$ such as we say that a strictly dominates b if and only if this relation holds:

$$a \succ b \iff \forall i \in [1, n] \ : a[i] > b[i] \tag{3.1}$$

Or we say that a simply dominates b if and only if this relation holds:

$$a \succeq b \iff \forall i \in [1, n] \ : a[i] \geq b[i] \wedge \exists j \in [1, n] \ : a[j] > b[j] \tag{3.2}$$

The Pareto frontier is the set of all Pareto efficient points, also commonly known as the Pareto front or Pareto set. Later on, in section 4.1 we will go in deeper and explain how we exploited this optimality model.

3.2 Context model

The context is constituted by information that can be used to characterize the situation of a specific interaction. For the purpose of our work, it represents the particular time and day where the interaction between the user and the item took place. Context is defined as a tuple of n dimensions, each with some possible values (domain)

$$c = [dimension_1 = value_1 \wedge \cdots \wedge dimension_n = value_n] \tag{3.3}$$

In our specific case we use only two dimensions:

- Timeslot
- Day of week

The domain of each dimension varies according to the dataset. An example of a valid context may be as the following: $[timeslot = daytime \wedge day = weekend]$. In our case we consider context to be hierarchical and we apply this concept in two ways:

- Sub-dimensions: some domain values can generalize some more specific values. For example timeslot "night" is a generalization of "early_night" and "late_night". So, "early_night" and "late_night" are sub-dimensions of the "night" value;

- Undefined values: using undefined values can equally define a more general context. For example $c = [timeslot = daytime]$ is a more general context than $c = [timeslot = daytime \wedge day = "weekend"]$.

We use the symbol $\succ$ to define this hierarchical relationship and the relation $\succeq$ is also defined as:

$$\forall c_i, c_j \in \mathcal{C}, c_j \succeq c_i \iff c_j \succ c_i \vee c_j = c_i \tag{3.4}$$

Therefore, if we define a universal context $\tilde{c}$, i.e. a general context which applies in all the possible situations, we can obtain a hierarchical representation in which $\tilde{c}$ is the root of the tree and time slot and day of the week can be considered as sub-trees. The Universal context is a special context that is more general than any other context. Hence $\forall c_i \in \mathcal{C}\ \tilde{c} \succ c_i$. Fig. 3.1 below shows the hierarchical relationship between the universal context $\tilde{c}$ and its sub-dimensions.

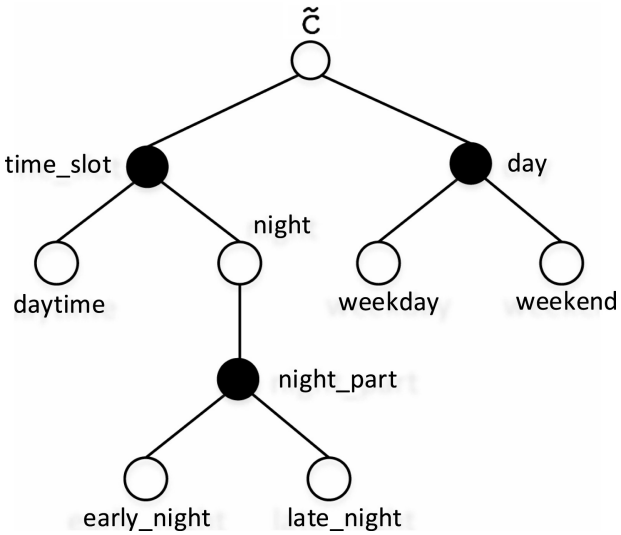


Figure 3.1. Context hierarchy [32]

3.3 Data model

We consider a set of items $\mathcal{I}$ and a set of users $\mathcal{U}$. Any group $G \in \wp(\mathcal{U})$, where $\wp(\mathcal{U})$ is the power set of $\mathcal{U}$, can be obtained from the set of users.
We have also $\mathcal{L}$ which is the log of the past groups activities. A single element of the logs $\mathcal{L}$ is a triple (G_j, c_j, i_j) where G_j is the group set and contains all the users that performed that activity together, $G_j \in \wp(\mathcal{U})$, c_j is the context in which the activity was performed, $c_j \in \mathcal{C}$, and finally $i_j \in \mathcal{I}$ is the actual item that the group decided to interact with.
$score(u, i, c)$, with $u \in \mathcal{U}$, $i \in \mathcal{I}$ and $c \in \mathcal{C}$, is the contextual scoring function that returns the preference of a user u for the channel i in a specific context c.
$TopK(u, c)$ is the list of the K items preferred by user u in context c, according to the values of $score(u, i, c)$

3.4 Problem statement

Let $\mathcal{U}$ be a set of users, $\mathcal{I}$ a set of items, $\mathcal{C}$ a set of contexts including also the universal context $\tilde{c}$. Supposing that the following elements are known:

- The context-aware scoring function $score(u, i, c), u \in \mathcal{U}, i \in \mathcal{I}, c \in \mathcal{C}$
- The log $\mathcal{L}$ recording the history of the items chosen by all past groups.

Then, given a test group $G \in \wp(\mathcal{U})$, at a context $c \in \mathcal{C}$, the problem of fair group recommendation is defined as recommending to the users in G a list (i.e. an ordered set) of K items considered to be interesting for them, in the given context c, from the available items in $\mathcal{I}$. Furthermore, in order to be ethical and fair, we have to make all the users in the group satisfied and, at the same time, prevent any form of selfishness or disparity, as far as possible.

3.5 FARGO

3.5.1 Contextual influence

An important element of FARGO is the user influence, based on previous group activities and studied in several works [7, 18, 19, 32]. Influence is a measure that reports for each user how many times it happened that his/her preferences were chosen by the group. For example, if a specific user joined several groups in the past and:

- in all of them the chosen item was in his preferences, he/she is a very influential person;
- in most of them the chosen item was in his preferences, he/she is a quite influential person;
- in very few of them the chosen item was in his preferences, he/she is not a influential person.

In real life scenarios every person has an individual personality. When we interact with other people we are subject to the influence of some charismatic people that tend to lead the group activity through their interests. Therefore, when trying to recommend items to a group we must be aware of this phenomenon and we must be able to balance it. The proposed way to have an estimation of how much an user is "influencer"in a given context is computed as a value between 0 and 1.

$$infl(u,c) = \frac{|l_j \in \mathcal{L} : c \succeq c_j \wedge u \in G_j \wedge i_j \in TopK(u,c,t_j)|}{|l_j \in \mathcal{L} : c \succeq c_j \wedge u \in G_j|} \tag{3.5}$$

Hence, the value of $infl(u,c)$ describes the ability of a user to direct the group decision in a specific context. Of course this metric does not lead to fair recommendations, since it rewards the user that influenced more often the group decision.

Example 3.5.1. Influence computation
Let's see a small example of how influence is computed. For simplicity we consider that our context has only one dimension (day) and the case in which $K = 1$. Let's suppose we want to compute the influence for users Alice (a) and Bob (b). The previous log of users' activities consists of tuples $< Group, context, item >$ (c,d,e,f,g are some other users we don't consider relevant)
Alice previous group activities:

- $< \{a, c, d\}, \text{saturday}, i_1 >$
- $< \{a, c\}, \text{saturday}, i_2 >$
- $< \{a, e, f, g\}, \text{saturday}, i_1 >$
- $< \{a, c, d\}, \text{saturday}, i_3 >$
- $< \{a, g\}, \text{tuesday}, i_1 >$
- $< \{a, c, d, e\}, \text{monday}, i_3 >$

- $< \{a, e\}, \text{monday}, i_2 >$

Supposing that Alice's $TopK$ contains i_1. Alice's influence in context $day = \text{saturday}$ is:

$$infl(a, \text{saturday}) = \frac{2}{4} = 0.5 \tag{3.6}$$

And we compute also the influence in the universal context $\tilde{c}$:

$$infl(a, \tilde{c}) = \frac{3}{7} = 0.428 \tag{3.7}$$

Bob's previous group activities are:

- $< \{b, c\}, \text{saturday}, i_1 >$
- $< \{b, c, d, f\}, \text{saturday}, i_2 >$
- $< \{b, g\}, \text{saturday}, i_3 >$
- $< \{b, d\}, \text{friday}, i_2 >$

Supposing that Bob's $TopK$ contains i_2. Bob's influence in context $day = \text{saturday}$ is:

$$infl(b, \text{saturday}) = \frac{1}{3} = 0.\bar{3} \tag{3.8}$$

And we compute also the influence in the universal context $\tilde{c}$:

$$infl(a, \tilde{c}) = \frac{2}{4} = 0.5 \tag{3.9}$$

3.5.2 Consensus as a fairness measure

One of the fastest way to compute a fairness metric among group preferences is the consensus. In a group decision it happens that every member of the group has a personal idea about items he/she would like the most and items he would preferably not use. The idea behind the calculation of consensus is that the more the variance among the individual preferences is high, the more that item leads to some kind of unfair treatment.

Definition 3.5.1 (Consensus)**.** Consensus measures how the group users preferences agrees on the evaluation of an item. It is defined as $1 - variance$, where $variance$ represents the variance of the users' scores for each item.

Considering the classical variance, where X is the set of all users' preferences:

$$\sigma_X^2 = \frac{\sum_i (x_i - \mu_X)^2}{n} \tag{3.10}$$

$$consensus = 1 - \sigma_X^2 \tag{3.11}$$

Traduced in our problem, the complete formula is:

$$consensus(G, i, c) = 1 - \frac{\sum_{u \in G}(score(u, i, c) - \overline{score}(u, i, c))^2}{|G|} \tag{3.12}$$

In equation 3.12 we have that $score(u, i, c)$ is the preference value of user u for item i in context c. Then, $\overline{score}(u, i, c)$ is the average preference among each users for item i and context c. Finally, G is the group set and $|G|$ is the cardinality of the group. Therefore *consensus* is a metric that becomes higher when the variance is relatively small and tends to decrease when the variance is high. In general consensus seeks its values into this range 3.13

$$consensus \in \left[\frac{1}{|G|}, 1\right] \tag{3.13}$$

Example 3.5.2. Consensus computation
Let's give a brief example about how consensus is computed.
Let's suppose that we have the users and items shown in Table 3.1

	u_1	u_2	u_3
i_1	0.0	0.3	0.9
i_2	0.3	0.4	0.8
i_3	0.8	0.0	0.3

Table 3.1. Users preferences example

Considering a group composed by u_1, u_2 and u_3 we have the following consensus scores for each item:

- $consensus(i_1) = 0.79$
- $consensus(i_2) = 0.93$
- $consensus(i_3) = 0.83\overline{6}$

We can see that the highest consensus is reached for item i_2 because this method tends to highlight items for which the score given by the users is the most similar.

3.5.3 Recommendation computation

The score aggregation function proposed is composed as follows:

$$groupscore(G, i, c) = \frac{\sum_{u \in G} infl(u, c) \cdot score(u, i, c)}{\sum_{u \in G} infl(u, c)} \cdot consensus(G, i, c)^{|G|} \tag{3.14}$$

The terms included in equation 3.14 are: $infl(u, c)$ that represents the influence factor described in 3.5. Function $score(u, i, c)$ gives us the score (preference) of user u for item i in context c. We use also the consensus definition in 3.12 as multiplicative factor to "weight" the final score. The use of contextual influence gives very good results in terms of recall [32], but does not keep in consideration the fairness. On the

contrary, it tends to put the emphasis on users who are "influencers". In order to achieve fairness we have to "discount" the score based on the consensus that the item gets. The exponent of consensus is the cardinality of the group.
If the number of contexts is high it may happen that some users didn't provide any feedback in some contexts. In this cases we rely on the universal context $\tilde{c}$ and on the non contextual preferences. The $TopK$ list of items preferred by a certain group G in context c is determined as follows:

- If all users in G have provided feedback in context c, we compute the context aware score with eq. 3.14
- Otherwise, we compute the score always with eq. 3.14, but using $\tilde{c}$ as context

Example 3.5.3. Score computation
Suppose that at a certain time a group composed by Alice (a), Bob (b) is watching TV. The users want a recommendation for a channel among the available i_1, i_2, i_3. This group has never watched TV together before, Alice and Bob only watched TV alone or with other groups. For simplicity we assume to have only one dimension in the context (day). The following are the singular user preferences:

Alice preferences	Bob preferences
$score(a, i_1, "saturday") = 0.3$	$score(b, i_1, "saturday") = 0.5$
$score(a, i_2, "saturday") = 0.15$	$score(b, i_2, "saturday") = 0.8$
$score(a, i_3, "saturday") = 0.7$	$score(b, i_3, "saturday") = 0.3$
	$score(b, i_1, "sunday") = 0.13$
	$score(b, i_2, "sunday") = 0.44$
	$score(b, i_3, "sunday") = 0.5$
$score(a, i_1, \tilde{c}) = 0.3$	$score(b, i_1, \tilde{c}) = 0.4$
$score(a, i_2, \tilde{c}) = 0.9$	$score(b, i_2, \tilde{c}) = 0.5$
$score(a, i_3, \tilde{c}) = 0.6$	$score(b, i_3, \tilde{c}) = 0.7$

Let's consider that we are in context day = saturday and we want to compute the score for the items (i_1, i_2, i_3). We use the influence values that we obtained from example 3.5.1. Both users provided feedbacks in that context so we can compute the score using the contextual scores of each user.
First we compute the consensus for each item according to the consensus formula 3.12.

- $consensus(\{a, b\}, i_1, \text{saturday}) = 1 - var(0.3, 0.5) = 0.98$
- $consensus(\{a, b\}, i_2, \text{saturday}) = 1 - var(0.15, 0.8) = 0.78875$
- $consensus(\{a, b\}, i_3, \text{saturday}) = 1 - var(0.7, 0.5) = 0.92$

Then we can compute the actual score:

- $groupscore(\{a, b\}, i_1, \text{saturday}) = \frac{0.3*0.5+0.5*0.\bar{3}}{0.5+0.3} \cdot 0.98^2 = 0.365$

- $groupscore(\{a,b\}, i_2, \text{saturday}) = \frac{0.15*0.5+0.8*0.\bar{3}}{0.5+0.\bar{3}} \cdot 0.78875^2 = 0.255$
- $groupscore(\{a,b\}, i_3, \text{saturday}) = \frac{0.7*0.5+0.3*0.\bar{3}}{0.5+0.\bar{3}} \cdot 0.92^2 = 0.457$

In this case the algorithm suggests to the group the item i_3.

We notice that the "unfair" method would recommend item i_3, contrarily to the FARGO method the recommends i_3
Now let's consider another context: $day = \text{sunday}$. We notice that not all the users provided feedbacks in this context. Alice did not provide any feedback in context $day = \text{sunday}$. This implies that when we have to take into consideration preferences and influences we have to use the universal context $\tilde{c}$.
First, we compute the consensus:

- $consensus(\{a,b\}, i_1, \text{sunday}) = 1 - var(0.3, 0.4) = 0.995$
- $consensus(\{a,b\}, i_2, \text{sunday}) = 1 - var(0.9, 0.5) = 0.92$
- $consensus(\{a,b\}, i_3, \text{sunday}) = 1 - var(0.6, 0.7) = 0.995$

Then, we compute the score:

- $groupscore(\{a,b\}, i_1, \text{sunday}) = \frac{0.3*0.428+0.4*0.5}{0.428+0.5} \cdot 0.995^2 = 0.350$
- $groupscore(\{a,b\}, i_2, \text{sunday}) = \frac{0.9*0.428+0.5*0.5}{0.428+0.5} \cdot 0.92^2 = 0.579$
- $groupscore(\{a,b\}, i_3, \text{sunday}) = \frac{0.6*0.428+0.7*0.5}{0.428+0.5} \cdot 0.995^2 = 0.647$

In this case the algorithm suggests to the group the item i_3.
Let's see what is the role of the consensus for fair recommendations. If we omitted the consensus in the score computation (we are calling this score "unfairgroupscore") we would reduce this method to a simple average method, and the results would be different:

- $unfairgroupscore(\{a,b\}, i_1, \text{saturday}) = \frac{0.3*0.428+0.4*0.5}{0.428+0.5} = 0.355$
- $unfairgroupscore(\{a,b\}, i_2, \text{saturday}) = \frac{0.9*0.428+0.5*0.5}{0.428+0.5} = 0.684$
- $unfairgroupscore(\{a,b\}, i_3, \text{saturday}) = \frac{0.6*0.428+0.7*0.5}{0.428+0.5} = 0.653$

How we can see the lack of consensus would lead to recommend i_2 instead of i_3.

3.6 Algorithm workflow

The current algorithm works as shown in figure 3.2.
The following are the preliminary steps we take before actually recommending items. Note that all this steps can be computed offline and then these computation does not affect the performance of the online recommendations.

- First we compute individual preferences. The outcome will be a sparse multidimensional matrix. The dimensions of this matrix includes: user, item, context. So the preference matrix will be a map that, given a user, an item and a context returns the corresponding preference $score(u, i, c) \rightarrow$ user item preference;
- Then we compute the influence. Influence is computed for each user according to equation 3.5. We are considering ephemeral groups, so for influence computation we rely on other groups' information. In other words, we use the information about the behavior of the user when he/she was in group with other people;
- For each user we compute his/her personal $TopK$. For completeness, we store the whole list of preferred items in a particular context for each user. The user $TopK$ maps the user and the context and returns a list of preferred items. $TopK : (u, c) \rightarrow List < item >$

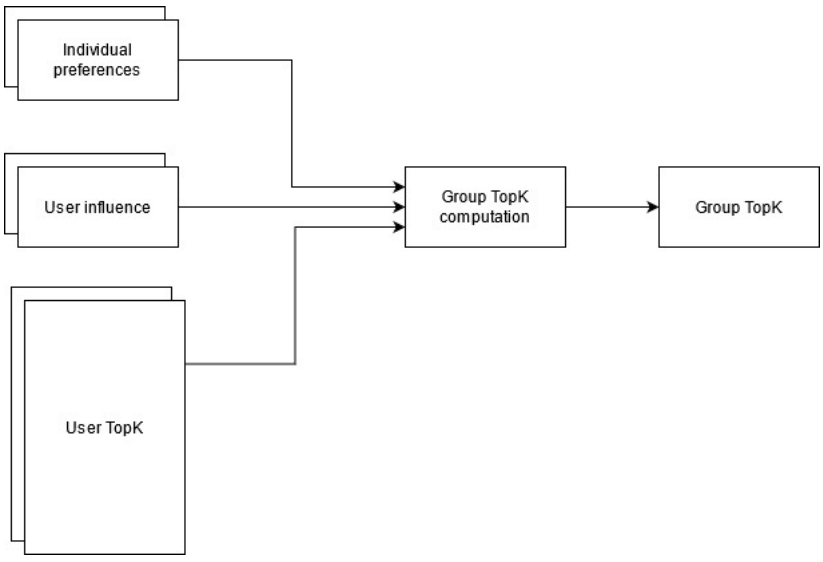


Figure 3.2. Group $TopK$ computation workflow

The algorithm carries out the following remaining steps online. These steps are the core of the algorithm and, in our thesis, we modify them in order to achieve better fairness and recall:

- we collect all the items coming from each individual user $TopK$ in a "global list";
- for each item of the global list we compute its score according to formula 3.14;
- we sort the items of the global list according to their fair group score;

- we return the $TopK$ of the global list that compose the actual recommendation set.

3.7 Analysis and possible improvements

In section 3.6 we presented how the current FARGO algorithm works. Our work aims to improve the quality of the recommender system, both in terms of recall and ethical guarantees. What we noticed from the analysis of the status quo is the following:

- the current score computation function 3.14 is based on the previous work [32] that includes the influence factor. As shown in the related paper this method boosts a lot the recall, but does not keep in consideration the fairness of recommendation. In fact the more a user led the group's decision in the past, the more he will lead the decision in the future (even though he will be part of another group). This behavior of course is contrary to any intuition of ethical recommendations;

- at the moment, influence is computed according to equation 3.5. Let's try to imagine a situation in which we have a user u that joined only 2 groups in the past, and in none of them the item chosen by the group was proposed by u. According to the formula, we know that u's influence is equals to 0. This implies that when he will join a new group in the future, the "weight" (influence) of his personal score on a particular item is also 0. From a pure recommender and statistical point of view it would be correct to give less importance to user's preference, but when trying to be fair, like in our case, we would like that all the group members' preference would weight the same;

- consensus is computed as is described in equation 3.12. It is clear that there is a problem in this case: in ephemeral groups the users can have very different preferences among the whole data-set of items. Therefore, it can happen that some items in the $TopK$ of some users are unknown to some other users. Let's suppose a simple case of a group of 2 users: u_1, u_2, and the score of item i must be computed. User u_1 has expressed a preference over i, but u_2 did not. When the score is computed, and in particular the consensus, we have to compute the variance among the two preference scores, specifically the variance among a score (e.g. 0.8) and a zero. Due to the missing score of u_2 over i, the consensus drops since the variance is high, and $1 - variance$ lets the consensus to fall. We tried to introduce the concept of "uncertain like" for the calculus of the consensus. This concept is based on the fact that in many cases people are somehow permissive when they have to give a score on an item they don't know, therefore we thought it could be possible to substitute the zeros, corresponding to the missing scores, with some positive values which could estimate the users' preferences relative to the item.

Chapter 4

Proposed Method

In this chapter we will present and discuss the work that we have done in order to improve FARGO's recommendation.
The FARGO project [32], on which we have carried out our experiments, has as its basic idea offering a recommendation method for ephemeral group, i.e. groups in which users, for various reasons, must share their preferences ($TopK$) with other unknown users. These preferences can be based on many factors or surrounding elements such as influences of the users themselves or their emotional state in the context. Like most of the competitors currently present in the market, the main objective for FARGO is to improve the recall as much as possible.
However, what guided the development of FARGO was mainly another aim, different from the aforementioned competitors, which is related to ethics, and in particular focuses on the fairness in group recommendation systems. In fact, the algorithm tries to propose ephemeral group recommendations that are as fair as possible for all users, avoiding disparity of contentment among the members of the group.
Regarding these aims, Skyline filtering and the "zeros estimation" are proposed below. Finally, after the analysis and evaluation of the problems of the measures of fairness currently used, in the last section of the chapter, new metrics of fairness will be proposed.

4.1 Skyline filtering

First, we started by implementing a skyline filtering. Skyline filtering is a method already used in recommender systems that exploit the Pareto efficiency technique (described in 3.1) to provide optimal items in multi variable optimization. In our thesis we refer to "point" as an array of users' preferences about a specific item. For example, if the group is composed by 5 people and they have to score an item, our point is represented by an array of 5 cells, where each cell is the preference of an user relative to the item. Therefore, before applying the algorithm, we build these "points" that are as numerous as the number of items that we want to choose from the recommendations. This process implies the implementation of the Pareto algorithm based on [35]. We exploited Pareto search to "automatically" select the best items in terms of score and to avoid the problems involved with the use of consensus as fairness metric to evaluate items. In particular, using consensus leads to evaluate items as less fair when the score is very different among users. Let's see an example:

	u_1	u_2	u_3
i_1	0.4	0.4	0.4
i_2	0.4	0.4	0.6

Table 4.1. Pareto example

According to the consensus calculation we have the following consensus values for each item:

- $i_1 = 1$
- $i_2 = 0.9867$

Therefore, in terms of consensus item i_1 is the preferred item. Nevertheless, if we look to item i_2 scores, we see that by choosing item i_2 we are not making u_1 and u_2 less satisfied but, instead, we are making u_3 happier.
In this case we could recommend to the group item i_2 since we are not violating any fairness principle.
This example shows that, by using pure variance computation, the recommendation is not optimal.
We could make some users happier by trying to prefer items that have a lower consensus but belong to Pareto frontier, while, at the same time, not making the other any more unhappy. Furthermore, by considering only Pareto optimal elements, we automatically let the system consider the best items with reference to their score.

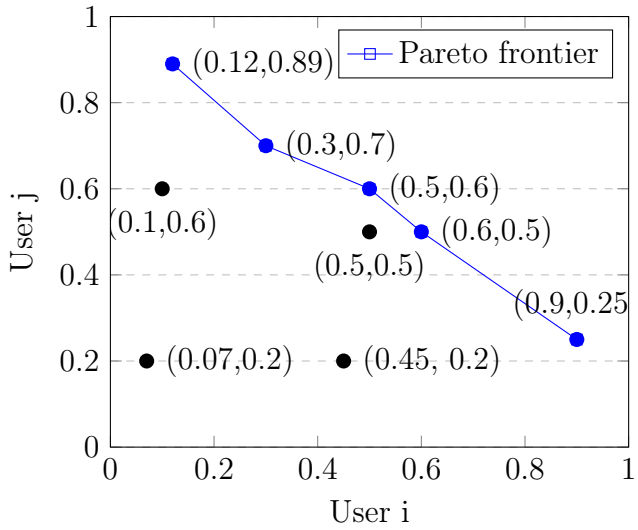


Figure 4.1. Pareto optimal example

Fig. 4.1 represents, in the case of a group of two users, their preferences for each item available. Each point in the graph is an item. The x coordinate represents user i preference, while the y coordinate represents user j preference. The blue-colored point are all Pareto optimal. For instance if we consider the item with coordinates $(0.5, 0.5)$ we can see that it is dominated (not strictly) by two other items with coordinates: $(0.5, 0.6)$ and $(0.6, 0.5)$. Moreover, if we consider the item with coordinates $(0.1, 0.6)$ it is strictly dominated by items $(0.12, 0.89)$ and $(0.3, 0.7)$ and it is simply dominated by item $(0.5, 0.6)$.
Thus, by applying the filter on this set of points, we can suggest items for which the preference for each user is maximized. We can see how choosing Pareto optimal items allows us to discard items where the score is dominated or strictly dominated.
In order to compute the skyline filtering in an efficient way we used the algorithm proposed in [4]. The algorithm receives as input the set of items and all the users' evaluations for each item. The items must be ordered using a monotone function. In [4] the monotone function can be one of the following:

- Sum: The list is sorted in decreasing order based on the sum of the scores
- Volume: The list is sorted in decreasing order based on the product of the scores
- Minimum Coordinate: first it sorts points considering for each of them their minimum coordinate value. Then, a sum of the skyline attributes is used in order to guarantee monotonicity in case of ties. In order to do that, other tie-breaking rules could in principle be used.

The algorithm pseudo-code is presented below:

Algorithm 1 Salsa Skyline Filtering

```
1: procedure SalsaSkylineFilter(r)                    ▷ Compute the skyline
2:     S ← ∅, stop ← false, p_stop ← undefined, u ← r
3:     while not stop ∧ u ≠ 0 do
4:         p ← next point from u, u ← u \ {p}
5:         if S ⊁ p then
6:             S ← S ∪ {p}, update p_stop
7:         end if
8:         if p_stop ≻ u then
9:             stop ← true
10:        end if
11:    end while
12:    return S
13: end procedure
```

We can observe that Algorithm 1 receives as input a sorted list (r) containing all the points with their relative scores. The order relation $\succ$ represents the Pareto domination defined in equation 3.1. In row 5 of the algorithm we have a comparison with $\not\succ$ among a set (S) and the point under analysis ($\mathbf{p}$). This implies that we must check if there is at least one point of the skyline set (S) that does not dominate the point $\mathbf{p}$. If that happens, $\mathbf{p}$ must be added to S, since it is part of the skyline. Furthermore, we can see that the early-stopping technique in row 8 of the algorithm allows us not to scan the whole set of points, given that they had been ordered before. The condition checks if the stopping point dominates all the points that are still to be considered. The instruction in row 8 is equivalent to checking if the current stop point ($\mathbf{p}_{stop}$) dominates the first element in the list of the points that have not been considered yet.

4.1.1 Iterative skyline filtering

We observed that, by simply applying the skyline filtering to propose items to the group, the recall decreased. In particular, the higher is the value of K (i.e. the number of items we are suggesting to the group) the lower the recall becomes. This problem is due to the fact that we are discarding many items considered not optimal, that are instead chosen by the group. If we try to figure out what happens in the moment of a real-life scenario decision, we realize that there are many other variables that we are not considering and of course the chosen items is not decided by applying mathematical rules. In particular, it is very hard to believe that whenever it exists a Pareto optimal element available the group will seek for it. In real life, it happens many times that the actual optimal items is chosen, but it's also very frequent the case in which a non optimal item is chosen due to the fact that we are humans, and not machines.
In our case the pure application of the filter leads us to loose many interesting items, that are then not suggested by the recommender system.
In order to pursue the idea of filtering, we proposed another similar method which allow us to cope with the problem we had with pure filtering. Pareto optimization

tries to optimize in some sense the items that are in the "top right" part of the graph 4.2 (in the two-user case).

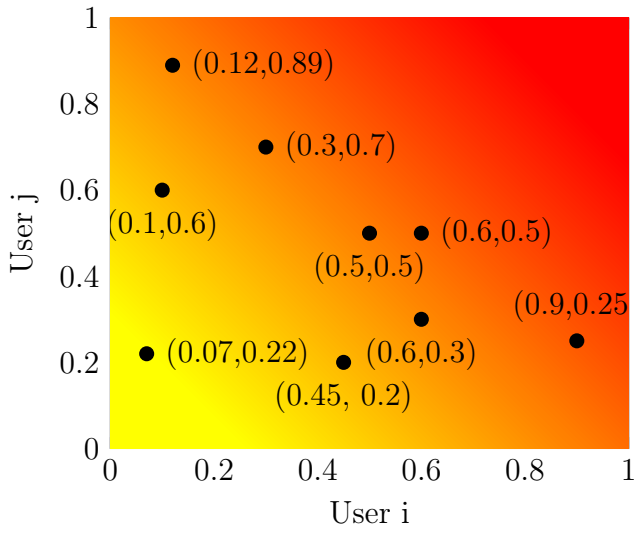


Figure 4.2. Pareto optimal heatmap

In other words, the Pareto filtering chooses points starting from the red area. The filter optimizes the score. For what concerns the fairness, we know that the points with highest consensus are placed as shown in graph 4.3. The most "fair" items are the ones for which the consensus is higher (i.e. on the bisector of the graph).

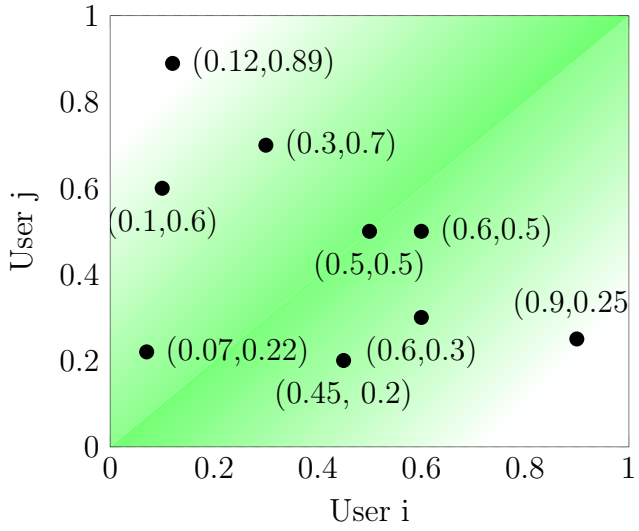


Figure 4.3. Consensus heatmap

What we are searching for is a sort of trade off between the optimization of the score and our fairness measure.
The idea is to use a fair measure to check which items obtain the higher consensus value among the optimal (score based) items. We want to try to see if we can find the best items, from the point of view of a certain scoring function among the already

filtered items (which are the optimal ones, from the point of view of the score).
Let's give a visual explanation of what is happening (using the same values as before) in fig. 4.4, fig. 4.5 and fig. 4.6.
At the first iteration of the filter (fig. 4.4) we get the same items that we would get by using the standard filter technique.

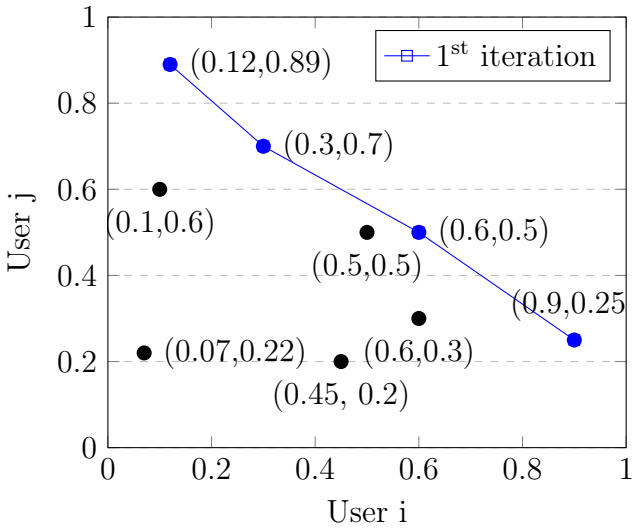


Figure 4.4. Pareto optimal 1st iteration

When we move to second iteration, i.e. we decide to apply the Pareto filtering on the remaining items (fig. 4.5), we get other items:

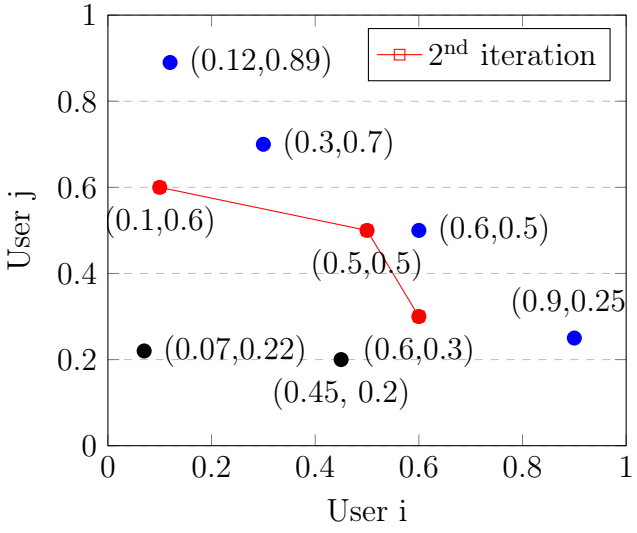


Figure 4.5. Pareto optimal 2nd iteration

If we proceed to the 3rd iteration we get the last two items (fig. 4.6).
We observe that as the number of iterations increases, we start to include sub-optimal items with reference to the score. But, at the same time, we may find other items for

which we have an higher value of consensus. In our example, we notice that the item with the highest consensus is the item with scores $(0.5, 0.5)$, that is found at iteration number 2.

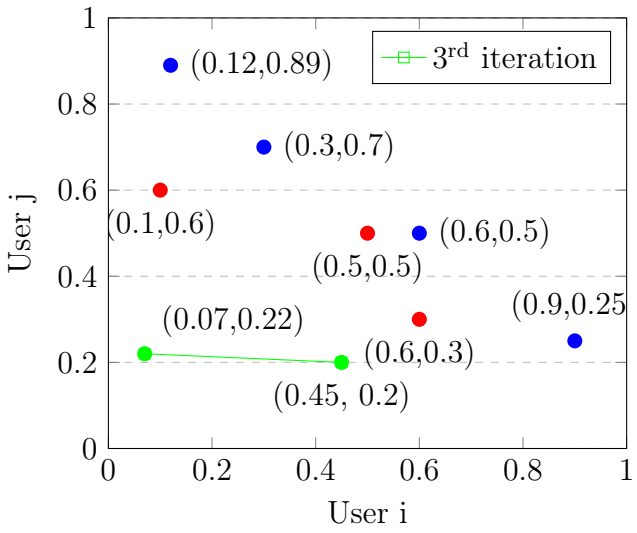


Figure 4.6. Pareto optimal 3rd iteration

The algorithm works as follows:

Algorithm 2 Iterative Skyline Filtering

```
1: procedure ITERATIVESKYLINE(G, c, usersTopKs) ▷ Compute the topK of the group
2:     topK = []
3:     items = ∅
4:     for topK ∈ usersTopKs do
5:         items ← items + topK.items
6:     end for
7:     while step ≤ ITERATIONS do
8:         skylineItems ← calculateSkyline(items)
9:         if skylineItems = ∅ then
10:            return topK
11:        end if
12:        items ← items \ skylineItems
13:        for item ∈ items do
14:            score = computeScore(item, G)
15:            topK ← orderedInsert(topK, item, score)
16:        end for
17:        step ← steps + 1
18:    end while
19:    return topK
20: end procedure
```

The algorithm contains some functions which are not defined in the above pseudo-code, let's present them:

- computeSkyline($items$): this function is the application of the Salsa algorithm proposed by [4]. It computes the Pareto optimal values based on the contextual scores of each user.
- computeScore($item, G$): this function calculates the group score for the item. We tried to use different scoring functions.

4.1.2 Score computation

Assigning a group score to a given item is of course a crucial step in our algorithm. Since we assign a group score to items that have been skyline filtered, we can assume that we are already dealing with the best items, from the point of view of the score. Therefore, at this point, we must keep in consideration the fairness metrics.
We have taken in consideration the following scoring functions:

- Consensus scoring:

$$groupscore(G, i, c) = consensus(G, i, c)^{|G|} \tag{4.1}$$

This is a simplification of the previous scoring function: it does not keep in consideration the score of each user since we are already optimizing the score using the skyline filter. We try to sort the items in the group $TopK$ using only the consensus since we should already be dealing with the optimal items in terms of score. In this way we expect to take the best items thanks to the filter, and then sort them based on the consensus.

- Consensus Average scoring:

$$groupscore(G, i, c) = \frac{\sum_{u \in G} score(u, i, c)}{|G|} \cdot consensus(G, i, c)^{|G|} \tag{4.2}$$

The previous scoring function uses contextual influence to determine which user is the most "influencer" in the group, and then gives more importance to his score in the group decision. To be fair we try to apply a simpler scoring function AVG, also weighted by the consensus.

- Fargo scoring: (i.e. the scoring function 3.14)

$$groupscore(G, i, c) = \frac{\sum_{u \in G} infl(u, c) \cdot score(u, i, c)}{\sum_{u \in G} infl(u, c)} \cdot consensus(G, i, c)^{|G|} \tag{4.3}$$

4.2 Zero as negative feedback vs zero as doubt

In order to improve the quality of the recommendations, we tried to take into account the case in which some users have 0 as score for some items. The interpretation of a $pref(u, i, c) = 0$ that was considered in previous works is a bad evaluation, in particular the worst possible one since $0 \leq pref(u, i, c) \leq 1$. Thus, both in the score and in the consensus, a $pref(u, i, c) = 0$ has the effect of lowering them, and can lead the recommender system to discard the item.
We decided to verify how many times does this problem affect the actual result in our data-set. We checked how many times the item chosen by the group was discarded and it had one or more 0 evaluation by the users. In other words, we counted the percentage of times where the test item was not in the group $TopK$ and the evaluation of the item was 0 for some users. 4.2 shows the results:

K	% lost	Recall	D_S	D_R
$K = 1$	6.07%	37,899%	7,694	18,263
$K = 2$	16.21%	54,154%	1,900	12,392
$K = 3$	27.45%	64,261%	0,889	10,066

Table 4.2. Percentage of lost items with zero evaluations

The percentage of lost items with zero evaluations was surprisingly high. Therefore, we introduced a new function in order to assign to items an hypothetical score during the calculus of consensus, which replaces the 0. This method is based on the idea that, for example, sometimes there can be an event on a particular TV channel which is not usually watched by the users, but that captures the interest of the users in the specific case, therefore it will be chosen by the users even if they haven't watched it before. It can also happen that users have a zero score simply because they were not even aware that the channel itself existed. In these examples, it is clear that the score of these hypothetical channels is bad, because it contains zeros, but it is chosen anyway for some reasons.
The estimation of the value which replaces the zero can be done by using several methods. In the next sections they will be discussed in detail.

4.2.1 Similar users for unknown items

The first method implemented in our estimation function exploits users' similarities in order to predict the missing scores. This method is based on the scores that other users, similar to the one whose score is lacking, gave to the same item. We decided to use some Collaborative Filtering (CF) techniques. In particular we used cosine similarity to compute similarity among users. It is computed as follows:

$$sim(u, w) = \frac{\vec{r_u} \cdot \vec{r_w}}{\|\vec{r_u}\|_2 \|\vec{r_w}\|_2} = \frac{\sum_{i \in I} r_{ui} r_{wi}}{\sqrt{\sum_{i \in I} r_{ui}} \sqrt{\sum_{i \in I} r_{wi}}} \tag{4.4}$$

Where $\vec{r_u}$ and $\vec{r_w}$ represent the arrays containing all the evaluations of user u and user w. In the same way, r_{ui} and r_{wi} represent the evaluations of users u and w for the item i. The set I represents the set of all the items available. By using this formula,

then we calculated all the similarities among all the users.
Of course we took into consideration only the pair of users who reached a similarity score greater than a threshold (a percentage of similarity that we have chosen empirically during our experiments), because we didn't want to consider two users to be similar if their cosine similarity is low. Hence, we exploited the similarity information to estimate a preference of a user for an unknown item. If some of the similar users did evaluate the missing items, we take the average among their evaluation. In other words, at the moment of collecting a user's preferences, when we notice that the score about an item is absent, we try to see if other users similar to him have evaluated that item in particular and we substitute the zero with the average of their evaluations. The complete algorithm is explained in the following pseudo-code.

Algorithm 3 Score estimation from similar users

```
1: procedure ESTIMATESCORE(u, i, S, score, c)
2:     similarScoresList ← List<>
3:     est_score ← 0
4:     for s ∈ S do
5:         if score.contains(s, i, c) then
6:             similarScoresList.add(score(s, i, c))
7:         end if
8:     end for
9:     est_score ← similarScoresList.sum()/similarScoresList.size()
10:    return est_score
11: end procedure
```

Let's analyze what is in Algorithm 3. The algorithm receives as input 5 parameters:

- u is the user for which we would try to estimate the score for item i;
- i is the considered item;
- S is a set containing all the users whose are similar to u;
- *score* is the matrix contains all the preference for each user and item, for a specific context;
- c is the context in which we want to perform the estimation.

The algorithm firstly checks among all the similar users which of them have rated the item under analysis (i). If there are some similar users who rated the item we add their preference value to a list. At the end of the cycle we have a list containing the opinions of some similar users and we take the average. Finally, this average is returned as score estimation for user u.
In table 4.3 the result of the estimation are shown:

K	% lost	Recall	D_S	D_R	Consensus
$K = 1$	5,5%	37,9%	7,695	18,264	94,404
$K = 2$	13.3%	54,152%	1,899	12,397	93,180
$K = 3$	22,7%	64,265%	0,889	10,071	93,519

Table 4.3. Percentage of lost items after estimation with similar users

4.2.2 Popular items

Another factor that affects the consensus is the popularity of the items. For example, let's consider the situation in which a user u has never seen a channel that is very popular though. In this case when user u joins a group of people that have in their preferences this channel, u's 0 rating weights a lot in the consensus. However, even if u has never seen this channel he may want to "try" to see it, after all, there has to be a reason why it is so popular!
Based on this idea we introduced a new function to estimate the missing value of a popular channel that works as follows: we compute at the beginning a list of the top popular channels both in a specific context and in any context. When we find out that a user has a missing rating we search in the popular channel list and if the missing rating regards a channel in the list, we substitute the missing score with this new score:

$$score(u, i, c) = \frac{1}{position} \tag{4.5}$$

where *position* is the position in the top popular item list.
We also tried to consider the idea that users with zero scores for certain items can instead appreciate them (perhaps in some contexts or times of the day). So another our proposal is to replace these zero scores with a default value, regardless of the popularity of items and depending on the scale of scores that the various items have in the data-set considered. In our case we worked with scores bounded between 0 and 1 and we empirically chose 0.9 as the default score; summarized in formula:

$$score(u, i, c) = 0.9 \tag{4.6}$$

Let's give an example: let's consider an user u and an item i and let's suppose that we want to predict the score of this user for this item. Let's hypothesize that item i occupies the 10^{th} position in the popular items' list. By applying the first formula, $score(u, i, c) = 0.1$, instead if we consider the second formula we get $score(u, i, c) = 0.9$. At the contrary to what we expected, the algorithm works better when we assign an higher score to items that are at the bottom of the top popular list. This is due to the fact that if we consider an item that was not evaluated by a user, even though it is a popular channel, it means that that particular user doesn't like it.

K	% lost	Recall	D_S	D_R	Consensus
$K = 1$	0%	37,928%	7,685	18,221	94,336
$K = 2$	0%	54,367%	1,927	12,150	93,114
$K = 3$	0%	64,511%	0,898	9,803	93,479

Table 4.4. Percentage of lost items after estimation with popular items

Of course with the popular item criteria we always assign a score (sometimes very low) to any item for each user. Therefore, we have no zero-values while computing the consensus.

4.2.3 Zero preference estimation

In order to combine all these aspects we average over all the score obtained. We combined both the estimated score from the similarities and from the popular channels taking into account the contextual information and the context free information. We came out with four possible estimations:

- Contextual similar items
- Similar items
- Contextual popular items
- Popular items

We evaluated the outcomes of all these methods and they all are reported in section 5.4. Then, we combined all these estimations in a single possible score taking the average of this four possible values, when not equal to 0, in order to obtain a unique value that includes all the previous information. We also had some empirical attempt in order to give to some of these 4 factors different weight, but with poor results. After our method is applied, these are the results:

K	% lost	Recall	D_S	D_R	Consensus
$K = 1$	1%	37,916%	7,686	18,220	94,340
$K = 2$	2%	54,345%	1,925	12,180	93,120
$K = 3$	4%	64,506%	0,894	9,825	93,482

Table 4.5. Percentage of lost items with zero evaluations

As we can see, both Recall and Recommendation disparity improved!
Score disparity seems to perform worse, due to the fact that it is computed using original scores, which are the sames score we are altering with this method.

4.3 New fairness measures

The fairness metrics that have been used so far in FARGO to evaluate ethics present, in our opinion, some problems:

- The items' score can be subject to many kind of bias: for example in the Auditel data-set the preference is given by the percentage of time spent in watching the channel, while in some other data-sets the preference is given by directly asking to the users a rating between 0 and a maximum value, or other data-sets instead use implicit ratings;

- Score disparity and Recommendation disparity, the metrics shown in 2.4, calculate the fairness based on the comparison among each user of the group;
- None of the current measures keep in consideration the influence that users have in each group;
- Score disparity and Recommendation disparity don't take into account the length of $TopK$, both users and recommendations ones.

In order to solve these issues we propose some new fairness measures that try to take into account the satisfaction of single users instead of the difference of the happiness among the users of the group.
A key element of our proposal is the influence $infl(u, c)$ that a specific user u has in a specific context c. As we already presented in section 3.5.1, influence can be easily calculated offline, starting from the current data stored in the data-set.
The first measure we propose is composed as follows: ee start considering all the items in the group $TopK$, and for each of them (i) and for each users (u) we take the difference between the position of i in the $TopK$ of u and the position of i in the group $TopK$ to obtain a sort of distance. This comes from the idea that when the position of the item i in the group $TopK$ corresponds to the position of i in the $TopK$ of u the user is highly satisfied. At the opposite, when the positions of the items in the group $TopK$ are distant to the users' $TopK$ position, these users are mainly unhappy. We multiply then this distance value by the influence to obtain an increase of the fairness measure when the user is unsatisfied by the distance of the positions and at the same time was even discriminated in the past. Finally, we divide all by the group size. Considering these premises, we propose here the following equation to measure fairness:

$$fair(c, G) = \frac{\sum_{u \in G} \sum_{i \in TopK(G,c)} [1 - infl(u, c)] \cdot |p_{ui} - p_{gi}|}{|G|} \tag{4.7}$$

In this equation we keep into consideration the ordering of the items: p_{ui} is the position of item i in user u TopK list. If the item i is absent in some users' $TopK$, we assign to p_{ui} the default value $K + 1$. Similarly, p_{gi} is the position of item i in the group $TopK$ list. Hence, this measure (4.7) gives better results when the recommender system is good at ordering the items according to the user preference, so we suggest to use measure 4.7 when we are interest also in the item ordering in the recommendation.
The next two measures follow the intuition that a user is satisfied when his/her preferences are actually recommended to the group he/her belongs to. We use the cardinality of how many items in the user $TopK$ are also present in the group recommendation $TopK$ performing the intersection. Also here, we use influence as multiplicative factor, in order to increase the measure in the case that the user was already discriminated before. Finally, we perform this multiplication for each user in the group, and the we divide by the group size. These two measures are quite similar, with the exception that the second one (4.9) is bounded from 0 to 1, while the first one (4.8) is bounded from 0 to K:

$$fair(c, G) = \frac{\sum_{u \in G} [1 - infl(u, c)] \cdot [K - TopK(u, c) \cap TopK(G, c)]}{|G|} \tag{4.8}$$

$$fair(c, G) = \frac{\sum_{u \in G}[1 - infl(u, c)] \cdot [1 - \frac{[TopK(u,c) \cap TopK(G,c)]}{K}]}{|G|} \tag{4.9}$$

The role of the influence is to highlight the cases in which there are users who have already been discriminated in terms of previous group choices and so, we take more into account that users with low influence.
In section 6.1.1 we report the results of the new fairness measures comparing FARGO with the other competitor algorithms. Then, in section 6.2 we have a comparison among the already existing metrics and our new ones.

Chapter 5

Experimental Results

In this chapter we will present experimental results to prove the goodness of our approach.
We will first introduce the two data-sets (Auditel and Music) on which we tested our improvements for FARGO.
Subsequently, the performances of FARGO and its competitors will be compared, both in terms of efficacy and ethics.
Finally, the numerical results will be listed and discussed. The performance metrics that have been considered relevant were examined: Recall, Score disparity and Recommendation disparity. The experimental results will be presented both for the skyline filtering and for the estimation of zeros.

5.1 Data model and problem statement

We have tested our work on two data-sets: Auditel data-set and Music data-set.
The first consists of a large real data-set formed by the preferences of the individual users about TV programs in multiple contexts, derived from the absolute viewing time of the various channels, and from the choices of TV programs made by the groups, built by the users themselves.
This was certainly the reference data-set, between the two, since it has characteristics that are quite important and useful for our objectives, such as reality, simplicity and the hugeness of the data contained.
The second one is a small data-set built thanks to the reviews, collected through a survey, of users about singers. The preferences were collected in just two generic contexts (car trip and dinner time as background music) and the groups were formed randomly.
We also decided to include this data-set in our work because, despite being small, it contains preferences derived from user reviews and this represents something more truthful than the parameters of the other data-set.

5.1.1 Auditel Data-set

Auditel data-set contains TV viewing information related to users and groups. It contains 4,968,231 viewings, in which we consider only those that have a duration greater than 3 minutes. 3,519,167 viewing are performed by singular users, while the remaining 1449064 viewings have been done by groups (i.e. multiple people together). Each row of the data-set contains information about the group id that performed the viewing, the user id, the channel id and the context in which the viewing took place. The context is given by the combination of two variables, *day of week* and *time slot*, that have the following domains :

- Day of week: $\mathcal{D} = \{week\ day, weekend\}$
- Time slot: $\mathcal{D} = \{graveyard\ slot, early\ morning, morning, daytime, early\ fringe, prime\ access, prime\ time, late\ fringe\}$

Start Time	End Time	Time slot
02:00	07:00	Graveyard slot
07:00	09:00	Early morning
09:00	12:00	Morning
12:00	15:00	Daytime
15:00	17:00	Early fringe
18:00	20:30	Prime access
20:30	22:30	Prime time
22:30	02:00	Late fringe

Table 5.1. Time slot table

In this data-set the preferences are not really user's feedback, but they are actually computed as the percentage of time watching a channel divided by the whole time

spent watching all the channels. The data-set contains the duration of each viewing. In order to derive the explicit preference of the users, the following formula was applied:

$$pref(u, i, c) = \frac{watchtime(u, i, c)}{\sum_j watchtime(u, j, c)} \tag{5.1}$$

We can see that the preference of a user u for an item i in a context c is a comparative value. This is conceptually correct, since if user u watches channel i in context c 90% of the time, he is supposed to prefer channel i in context c at 90%. But this is not actually a "preference". Consider the case in which, for example, a user (u_1) has seen 10 channels in the same context, all of these for the same time. The preference (always referred to the same context) of each channel, according to equation 5.1 is 0.1. Now suppose that this user decides to perform a group view with another user (u_2), so we need to suggest some items to this group. Suppose that u_2 watched for 80% of the time one of the 10 channels that u_1 watched too. When we compute the consensus we have to take into consideration the variance among 0.1 and 0.8. This value is quite high since the result is 0.755.
Therefore, the consensus is not that high, but we are focusing on an item that is equally preferred among 10 items by u_1 and that is u_2's preferred channel.

5.1.2 Music data-set

Music data-set was created by Andrea Fretti while he was working on his thesis. It contains information about music listening performed by groups of users and every singular user preference among some singers, retrieved thought a survey.
280 users were asked to fill two different forms:

- an individual form collecting demographic data (i.e., age and gender) and contextual individual preferences (0-4 rating) about 30 music artists;
- a group form to be filled in groups depending on a collective choice of a music artist available in a particular context.

The following two listening contexts have been selected, taking into account two particular situations that everyone often experiences: during a car trip and at dinner as background music.

5.2 Previous Results

The performances of FARGO have been compared with others that are based on aggregation functions presented in chapter 2 using both data-sets. The outputs that we have taken as reference points are presented below in subsections 5.2.1 and 5.2.2. The similarities and differences in performances between the algorithms are highlighted for each data-set, for both the performances and the ethic concerns.

5.2.1 Auditel data-set analysis

Method	Recall	D_S	D_R	AVG Consensus
FARGO	**37,894**	7,693	18,257	94,408
AVG	33,914	7,074	18,148	93,535
LM	31,187	6,352	14,358	95,388
MS	29,558	9,362	**12,59**	**95,852**
DIS	34,497	**6,81**	17,374	94,612
FAIR-LIN	32,433	8,846	18,56	94,518
FAIR-PROP	32,225	8,825	13,757	95,228
EXP	34,613	8,111	18,227	94,685
ENVY-FREE	33,915	7,074	18,127	93,535

Table 5.2. Initial performance Auditel data-set $K = 1$

Method	Recall	D_S	D_R	AVG Consensus
FARGO	**54,149**	**1,9**	12,393	93,182
AVG	51,564	2,933	**8,781**	92,317
LM	47,538	2,672	10,206	93,696
MS	46,06	4,082	11,93	**93,871**
DIS	51,831	2,73	9,77	92,871
FAIR-LIN	50,038	3,596	7,442	92,233
FAIR-PROP	49,93	4,253	8,907	93,136
EXP	52,325	2,367	11,475	93,151
ENVY-FREE	51,541	2,933	8,79	92,317

Table 5.3. Initial performance Auditel data-set $K = 2$

Method	Recall	D_S	D_R	AVG Consensus
FARGO	**64,26**	**0,889**	10,075	93,519
AVG	62,911	1,361	7,526	92,888
LM	58,485	1,263	8,228	93,721
MS	57,765	1,917	9,755	**93,912**
DIS	62,989	1,264	7,982	93,135
FAIR-LIN	60,634	1,625	**6,992**	92,634
FAIR-PROP	61,559	1,789	7,714	93,401
EXP	63,135	1,152	9,465	93,39
ENVY-FREE	62,909	1,361	7,522	92,888

Table 5.4. Initial performance Auditel data-set $K = 3$

Tables 5.2, 5.3 and 5.4 show that FARGO has the following behavior:

- is currently the best in terms of **Recall**, for any value of K;
- the **Score disparity** remains the best for K equal to 2 and 3 and it is also quite performing for K equal to 1;
- **Recommendation disparity** unfortunately turns out to be among the worst;
- the **AVG consensus** remains broadly competitive with respect to the outputs of the other algorithms.

5.2.2 Music data-set analysis

Method	Recall	D_S	D_R	AVG Consensus
FARGO	**25**	2,196	**1,62**	**99,957**
AVG	12,5	0,805	3,241	99,904
LM	13,889	1,136	3,762	99,976
MS	11,806	1,102	3,877	99,898
DIS	22,917	0,747	3,414	99,907
FAIR-LIN	11,111	2,185	4,167	99,952
FAIR-PROP	13,19	**0,657**	2,546	99,918
EXP	12,5	0,747	3,241	99,916
ENVY-FREE	12,5	0,805	3,241	99,904

Table 5.5. Initial performance Music data-set $K = 1$

Method	Recall	D_S	D_R	AVG Consensus
FARGO	**40,278**	0,87	**2,025**	99,903
AVG	25	0,392	2,951	99,9
LM	25	0,351	3,125	**99,931**
MS	22,917	0,353	3,819	99,889
DIS	34,722	0,434	2,836	99,856
FAIR-LIN	23,611	0,809	2,141	99,899
FAIR-PROP	20,833	0,377	3,241	99,901
EXP	24,306	**0,329**	2,836	99,908
ENVY-FREE	25	0,392	2,951	99,9

Table 5.6. Initial performance Music data-set $K = 2$

Method	Recall	D_S	D_R	AVG Consensus
FARGO	**49,306**	0,53	2,398	99,895
AVG	34,722	0,252	2,712	99,888
LM	34,722	0,278	1,989	**99,919**
MS	33,333	0,258	3,619	99,868
DIS	41,667	0,319	2,49	99,855
FAIR-LIN	31,944	0,476	**1,806**	99,891
FAIR-PROP	29,861	0,374	2,996	99,893
EXP	34,722	**0,211**	2,481	99,897
ENVY-FREE	34,722	0,252	2,712	99,888

Table 5.7. Initial performance Music data-set $K = 3$

Tables 5.5, 5.6 and 5.7 show that FARGO, also with the Music data-set, has different characteristics. In particular, the parameters that evaluate fairness, present both similarities and differences with the Auditel data-set:

- FARGO is again the best in terms of **Recall**, for any value of K;
- the **Score disparity** totally changes and becomes the worst;
- the **Recommendation disparity** improves and is even the best for K equal to 1 and 2;
- the **AVG consensus** follows the same trend as the Auditel data-set.

5.3 Skyline filtering

The outputs obtained after the implementation of the skyline filtering described in the 4.1 section are illustrated below.
It is good to remember that the filter was tested using three different scoring functions. In fact, the ordering of the elements obtained by the scoring function (line 14 of Algorithm 2) can be based:

- on the consensus, using 4.1 equation;
- on the score corrected by the consensus (average), using 4.2 equation;
- directly on the score that the group gives to the item in question, using 4.3 equation.

Furthermore, only the first 4 iterations of the filter will be reported below, for each variant, as the subsequent ones did not significantly modify the outputs.

5.3.1 Optimizing based on consensus

The following tables have been filled using the 4.1 equation as scoring function, i.e. taking into account only the consensus of the item within the group.

Auditel data-set

	Recall	D_S	D_R	AVG Consensus
Before	37,894	7,693	18,257	94,408
$iteration_1$	29,099	8,526	10,750	97.179
$iteration_2$	16,054	11,524	3,027	98,739
$iteration_3$	11,990	13,007	2,377	98,878
$iteration_4$	10,384	13,843	2,147	98,899

Table 5.8. Optimizing based on consensus Auditel data-set $K = 1$

	Recall	D_S	D_R	AVG Consensus
Before	54,149	1,9	12,393	93,182
$iteration_1$	46,282	4,099	9,236	93,595
$iteration_2$	33,909	4,307	10,428	96,397
$iteration_3$	24,148	4,793	8,085	97,219
$iteration_4$	20,667	4,957	6,008	97,378

Table 5.9. Optimizing based on consensus Auditel data-set $K = 2$

	Recall	D_S	D_R	AVG Consensus
Before	64,26	0,889	10,075	93,519
$iteration_1$	57,528	2,093	7,415	93,090
$iteration_2$	48,576	1,612	9,792	95,274
$iteration_3$	37,274	1,778	10,700	96,399
$iteration_4$	31,159	1,853	9,835	96,791

Table 5.10. Optimizing based on consensus Auditel data-set $K = 3$

Music data-set

	Recall	D_S	D_R	AVG Consensus
Before	25	2,196	1,62	99,957
$iteration_1$	15.972	0.727	1.042	99.996
$iteration_2$	15.278	0.819	0.868	99.997
$iteration_3$	14.583	0.691	0.694	99.999
$iteration_4$	14.583	0.508	0.694	99.999

Table 5.11. Optimizing based on consensus Music data-set $K = 1$

	Recall	D_S	D_R	AVG Consensus
Before	40,278	0,87	2,025	99,903
$iteration_1$	27.083	0.208	2.604	99.897
$iteration_2$	24.306	0.166	2.488	99.980
$iteration_3$	23.611	0.247	2.662	99.980
$iteration_4$	23.611	0.210	2.488	99.980

Table 5.12. Optimizing based on consensus Music data-set $K = 2$

	Recall	D_S	D_R	AVG Consensus
Before	49,306	0,53	2,398	99,895
$iteration_1$	38.889	0.122	2.126	99.879
$iteration_2$	30.556	0.078	2.324	99.964
$iteration_3$	29.167	0.105	2.411	99.970
$iteration_4$	29.167	0.094	2.064	99.970

Table 5.13. Optimizing based on consensus Music data-set $K = 3$

Comments and analysis

We noticed that we have a rather consistent loss for what concerns the Recall as the value of iteration increases. This is due to the fact that as we add more and more items to be "scored", we encounter items that have similar score, but that are not optimal. Therefore, many items with low score are put in a higher position in the $TopK$ causing the recommendation of items with high consensus, but in many cases

not optimal.
For what concerns the Score disparity, despite showing good performances compared with other fairness methods, we noticed a worsening trend with respect to the standard FARGO method. Of course this result is correlated to the low Recall we get. For the same reason, sometimes we choose sub-optimal items (with respect to the score) that lead to increasing the gap between a user personal evaluation and the item chosen.
Looking at the Recommendation disparity we noticed that in general this method allows to achieve better results. The Recommendation disparity becomes lower as we increase the number of iterations.
Also the consensus of course gets better as the number of iterations increases. This is due naturally to the way in which we are optimizing the $TopK$.
Overall, we found that the results are good from the point of view of the ethical indicators. The Recall outcomes, instead, perform actually bad.

5.3.2 Optimizing based on average

The following tables have been filled using the 4.2 equation as scoring function, i.e. taking into account not only the consensus, but the score weighted with consensus of the item within the group.

Auditel data-set

	Recall	D_S	D_R	AVG Consensus
Before	37,894	7,693	18,257	94,408
$iteration_1$	33,992	6,928	17,489	94,710
$iteration_2$	34,061	7,006	17,312	93,578
$iteration_3$	34,050	7,010	17,290	93,763
$iteration_4$	34,046	7,010	17,287	93,764

Table 5.14. Optimizing based on average Auditel data-set $K = 1$

	Recall	D_S	D_R	AVG Consensus
Before	54,149	1,9	12,393	93,182
$iteration_1$	49,487	4,029	7,052	92,081
$iteration_2$	51,456	2,982	9,089	92,557
$iteration_3$	51,419	2.916	9,127	92,561
$iteration_4$	51,416	2.917	9,121	92,561

Table 5.15. Optimizing based on average Auditel data-set $K = 2$

	Recall	D_S	D_R	AVG Consensus
Before	64,26	0,889	10,075	93,519
$iteration_1$	59,224	2,205	6,640	92,550
$iteration_2$	62,450	1,369	7,469	92,983
$iteration_3$	62,705	1,359	7,912	93,050
$iteration_4$	62,685	1,360	7,914	93,049

Table 5.16. Optimizing based on average Auditel data-set $K = 3$

Music data-set

	Recall	D_S	D_R	AVG Consensus
Before	25	2,196	1,62	99,957
$iteration_1$	14.582	0.463	1.042	99.923
$iteration_2$	13.194	0.488	1.215	99.923
$iteration_3$	13.194	0.488	1.215	99.923
$iteration_4$	13.194	0.488	1.215	99.923

Table 5.17. Optimizing based on average Music data-set $K = 1$

	Recall	D_S	D_R	AVG Consensus
Before	40,278	0,87	2,025	99,903
$iteration_1$	25.694	0.299	2.488	99.874
$iteration_2$	22.222	0.216	2.836	99.914
$iteration_3$	22.222	0.202	2.951	99.912
$iteration_4$	22.222	0.202	2.951	99.912

Table 5.18. Optimizing based on average Music data-set $K = 2$

	Recall	D_S	D_R	AVG Consensus
Before	49,306	0,53	2,398	99,895
$iteration_1$	37.500	0.162	1.798	99.874
$iteration_2$	28.472	0.114	2.816	99.900
$iteration_3$	27.778	0.117	3.038	99.913
$iteration_4$	27.778	0.117	3.038	99.913

Table 5.19. Optimizing based on average Music data-set $K = 3$

Comments and analysis

If we analyze the tables above we can see that Recall tends to loose some points as the number of iteration increases. Contrarily to the case in which we optimized the consensus, the Recall is much higher because we kept into account also the actual preferences of the users.
For what concerns the Score disparity we see that we have higher values except for

$K = 1$.
In Recommendation disparity our method gives excellent results with respect to FARGO. The consensus seems to perform slightly worse, but the difference is in the order of decimal factors.

5.3.3 Optimizing based on score

The following tables have been filled using the 4.3 equation as scoring function, i.e. taking into account directly the score that the group gives to an item due to influences and singular preferences.

Auditel data-set

	Recall	D_S	D_R	AVG Consensus
Before	37,894	7,693	18,257	94,408
$iteration_1$	37,408	7,594	18,187	94,327
$iteration_2$	37,632	7,627	18,101	94,396
$iteration_3$	37,629	7,630	18,094	94,411
$iteration_4$	37,625	7,628	18,098	94,413

Table 5.20. Optimizing based on score Auditel data-set $K = 1$

	Recall	D_S	D_R	AVG Consensus
Before	54,149	1,9	12,393	93,182
$iteration_1$	50,795	3,647	7,706	92,072
$iteration_2$	53,996	2,012	11,472	93,052
$iteration_3$	54,057	1,928	11,801	93,075
$iteration_4$	54,084	1,928	11,838	93,077

Table 5.21. Optimizing based on score Auditel data-set $K = 2$

	Recall	D_S	D_R	AVG Consensus
Before	64,26	0,889	10,075	93,519
$iteration_1$	59,408	1,997	6,147	92,262
$iteration_2$	63,577	0,946	8,183	93,142
$iteration_3$	64,142	0,903	9,210	93,341
$iteration_4$	64,179	0,903	9,425	93,372

Table 5.22. Optimizing based on score Auditel data-set $K = 3$

Music data-set

	Recall	D_S	D_R	AVG Consensus
Before	25	2,196	1,62	99,957
$iteration_1$	15.278	0.908	1.794	99.940
$iteration_2$	15.278	0.908	1.794	99.940
$iteration_3$	15.278	0.908	1.794	99.940
$iteration_4$	15.278	0.908	1.794	99.940

Table 5.23. Optimizing based on score Music data-set $K = 1$

	Recall	D_S	D_R	AVG Consensus
Before	40,278	0,87	2,025	99,903
$iteration_1$	27.083	0.308	2.257	99.861
$iteration_2$	24.306	0.308	2.257	99.906
$iteration_3$	24.306	0.308	2.431	99.906
$iteration_4$	24.306	0.308	2.431	99.906

Table 5.24. Optimizing based on score Music data-set $K = 2$

	Recall	D_S	D_R	AVG Consensus
Before	49,306	0,53	2,398	99,895
$iteration_1$	38.889	0.184	1.952	99.868
$iteration_2$	30.556	0.184	1.784	99.900
$iteration_3$	29.861	0.184	1.958	99.905
$iteration_4$	29.861	0.184	1.958	99.905

Table 5.25. Optimizing based on score Music data-set $K = 3$

Comments and analysis

Starting from the Recall results we see that as the number of iteration increases, the Recall approaches a limit value, which is very similar to the previous one.
Also the Score disparity seems to work quite well: as the iteration number gets higher we approach the original value, except for $K = 1$ where we are able to do even better than the original. The Recommendation disparity performs better for any value of K. For what regards the consensus, we observe that for $K = 1$ we can reach better values as the number of iteration raises. For values of K greater than 1 we reach very few lower results.

5.3.4 Comparison of outputs

In this section, by showing some plots, the results presented so far will be analyzed and the three scoring functions will be compared with the initial outputs of FARGO.

In order, Recall, Score disparity, Recommendation disparity and the average of consensus are analyzed for the values of K equal to 1, 2, 3.

Recall outputs - Auditel data-set

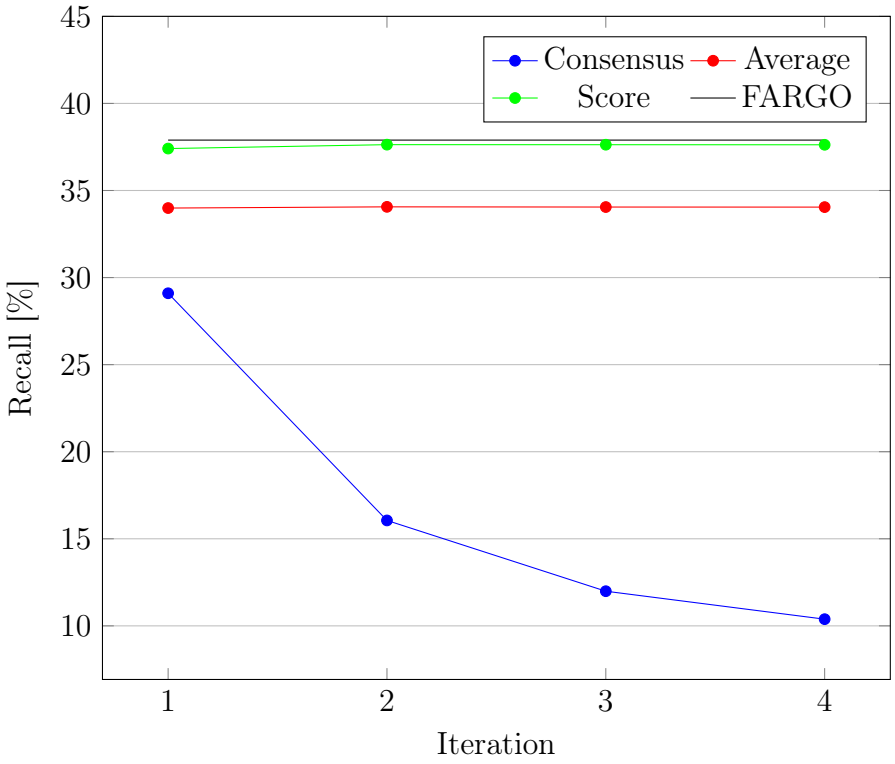


Figure 5.1. Recall varying iterations Auditel data-set $K = 1$

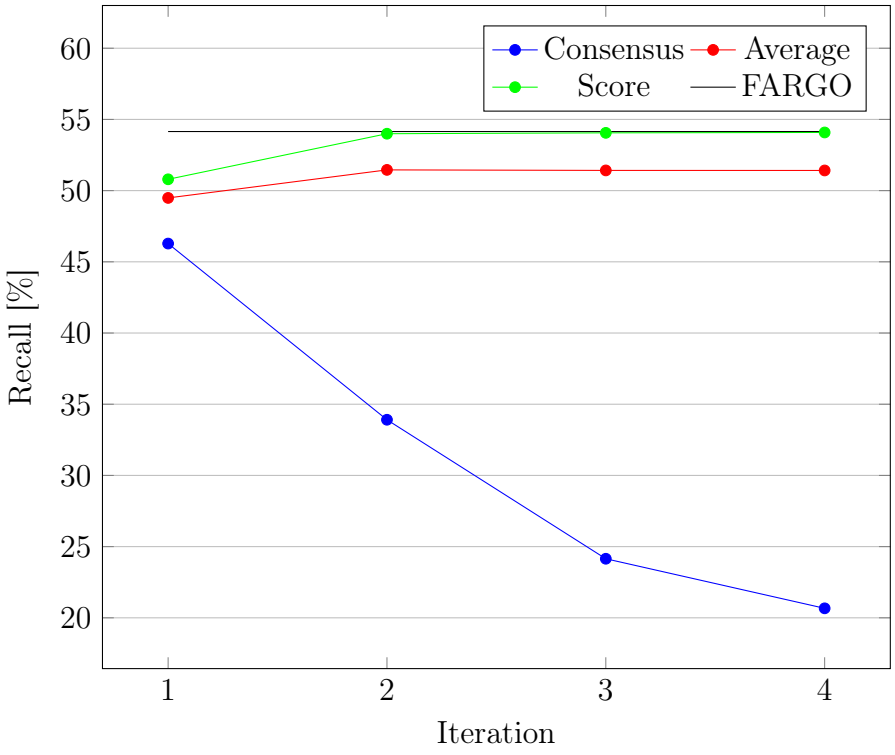


Figure 5.2. Recall varying iterations Auditel data-set $K = 2$

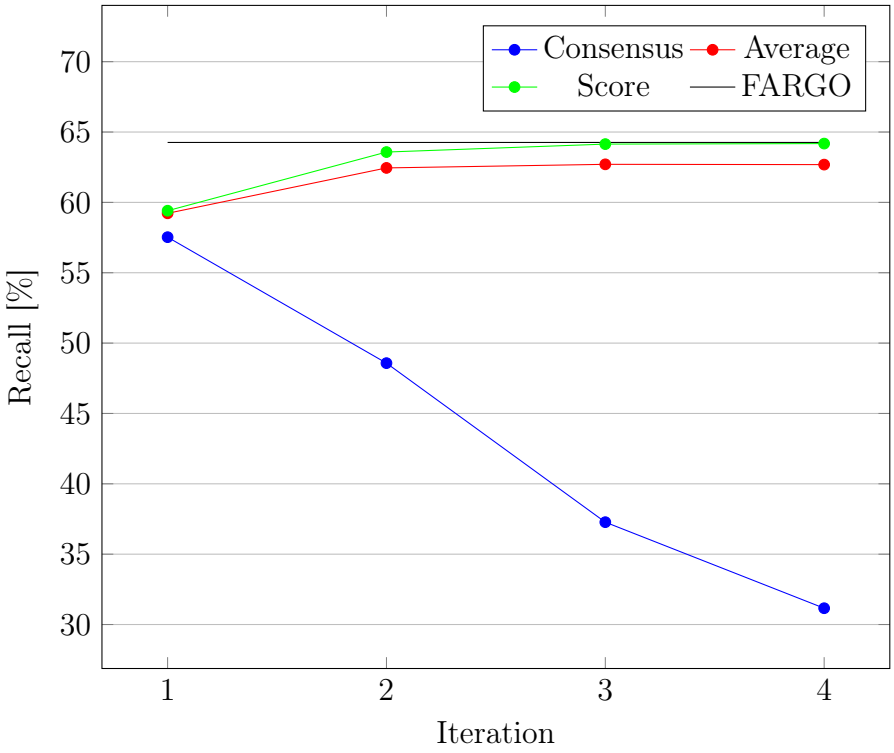


Figure 5.3. Recall varying iterations Auditel data-set $K = 3$

Plots 5.1, 5.2 and 5.3 above show that with the Auditel data-set, by increasing the iterations, Recall has more or less the same behavior for any value of K:

- we obtain a much lower value of performance by using the formula based on **consensus**, especially if the number of iteration increases;
- we obtain a little lower performances by using the formula based on **average**;
- Recall doesn't change by using the formula based on **score**, especially for value of K greater than 1. Recall remains more or less similar to the original FARGO with at least two iterations.

Recall outputs - Music data-set

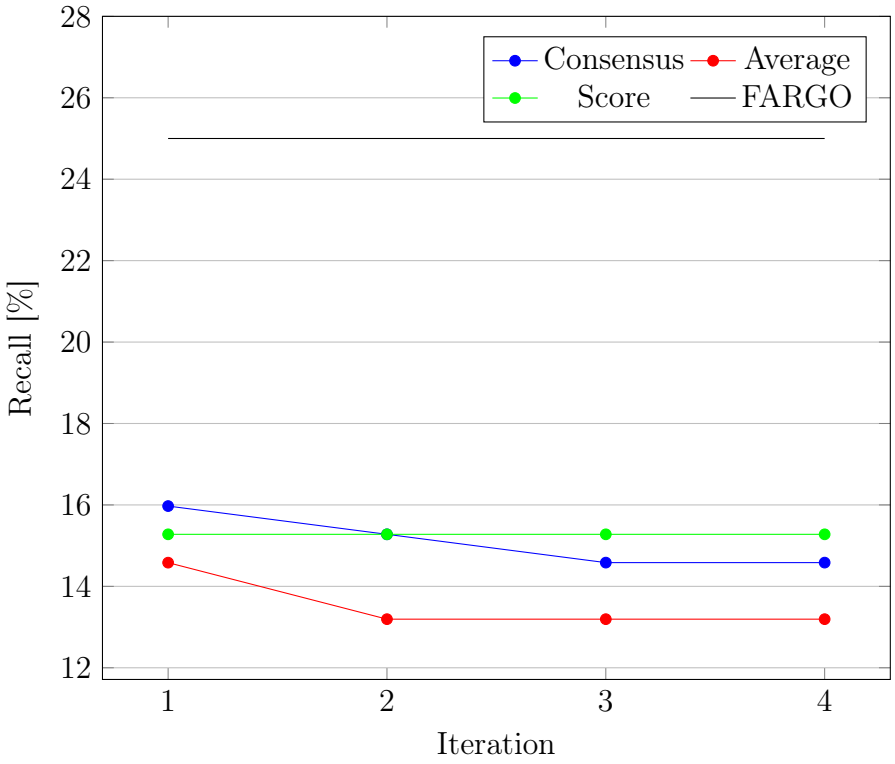


Figure 5.4. Recall varying iterations Music data-set $K = 1$

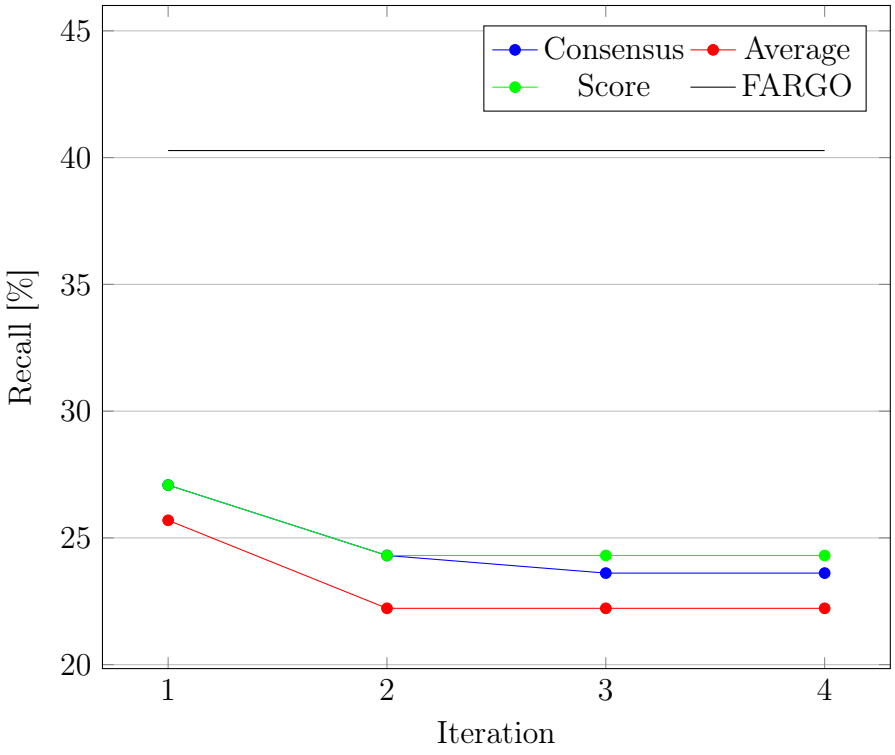


Figure 5.5. Recall varying iterations Music data-set $K = 2$

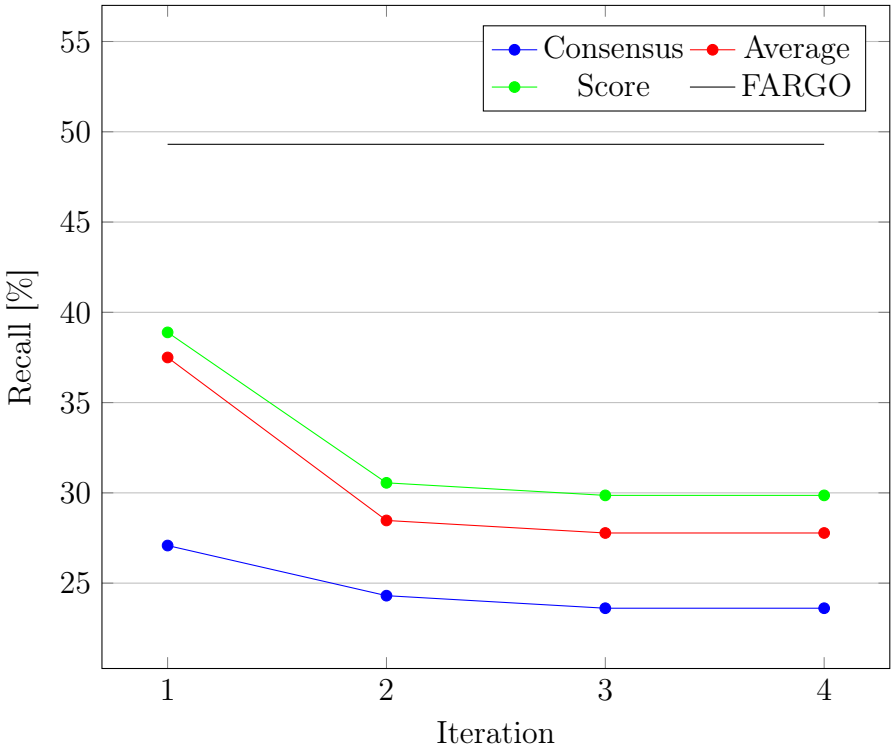


Figure 5.6. Recall varying iterations Music data-set $K = 3$

Plots 5.4, 5.5 and 5.6 above show that with the Music data-set Recall gets much worse compared to the starting one for any value of K and, differently from before, it becomes worse by increasing iteration:

- Recall this time doesn't change a lot by changing the methods, as we seen instead in the previous cases with the Auditel data set, but it is the worst with $K = 1$;
- in the case of values of K less than 3, Recall is the worst using the formula based on **average**;
- Recall is better using the formula based on **score**.

Score disparity outputs - Auditel data-set

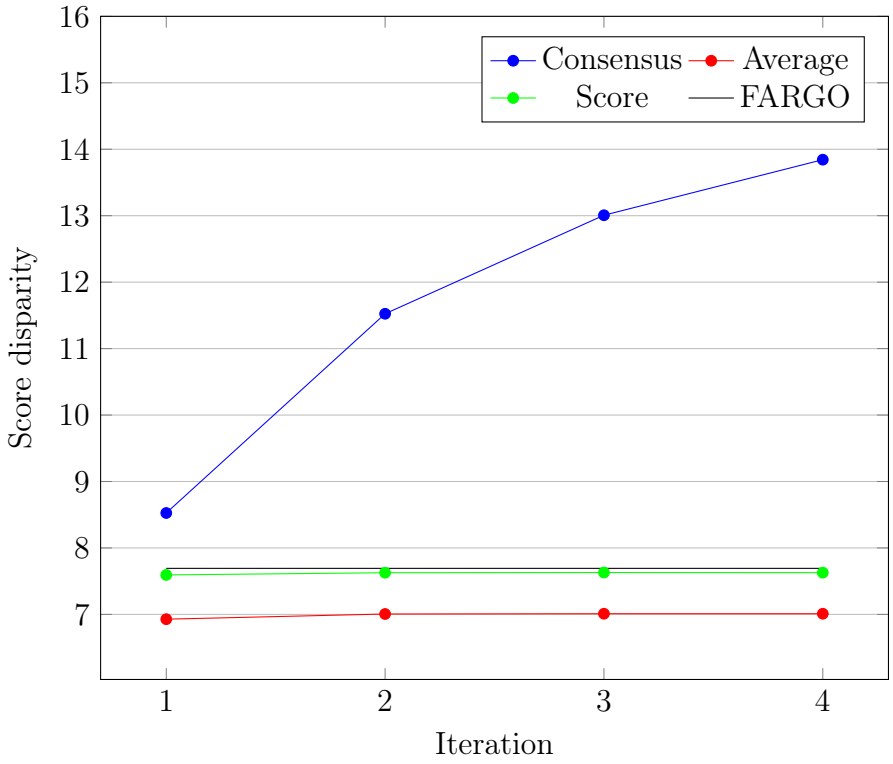


Figure 5.7. Score disparity varying iterations Auditel data-set $K = 1$

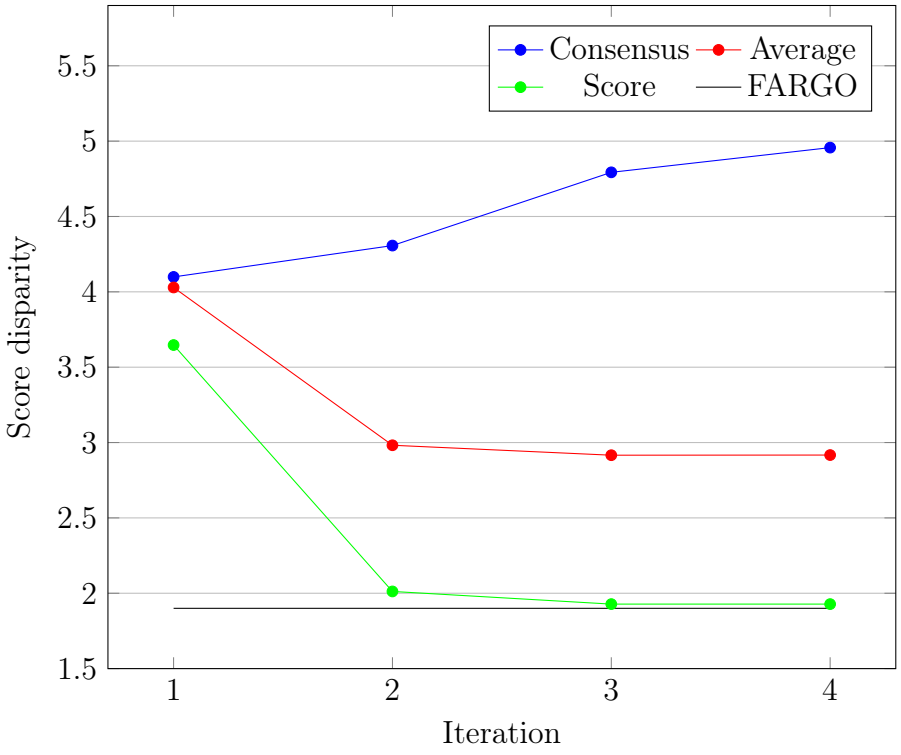


Figure 5.8. Score disparity varying iterations Auditel data-set $K = 2$

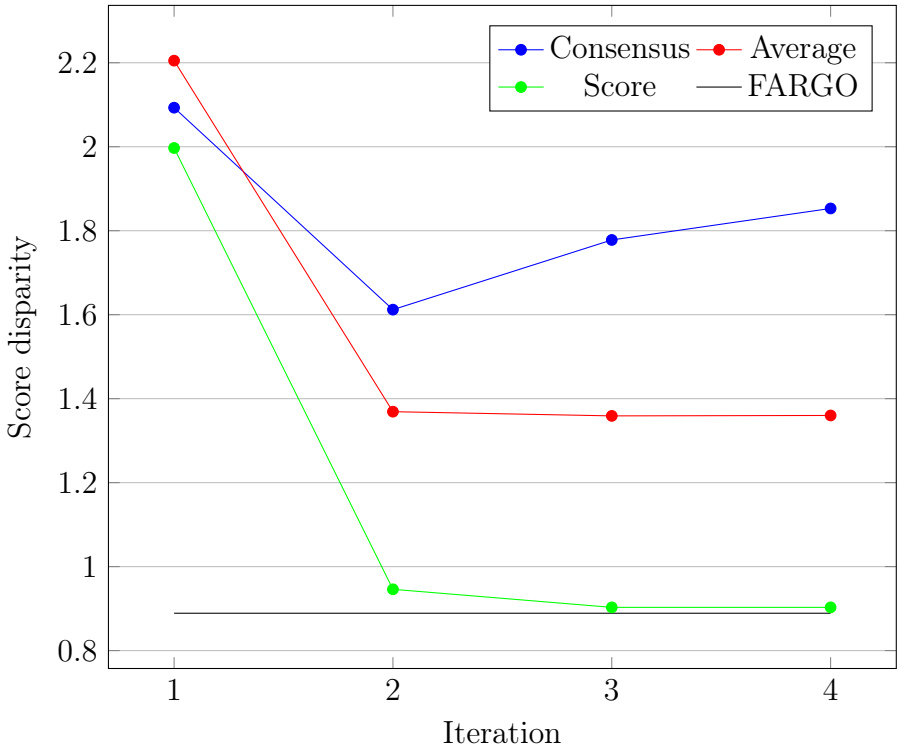


Figure 5.9. Score disparity varying iterations Auditel data-set $K = 3$

Plots 5.7, 5.8 and 5.9 above show that with the Auditel data-set the score disparity behavior presents different trends varying the value of K. It gets better with $K = 1$, but not for the other values of K. In the case of K equal to 2 and 3,the plots show the same trend, which is different from the case in which K is equal to 1. In particular, while computing Score disparity:

- the formula based on **consensus** is the worst among all and it gets worse increasing the number of iterations;
- the formula based on **average** has a good trend, especially with $K = 1$ and it becomes lower with more than one iteration;
- Score disparity is really good using the formula based on **score** for any value of K and it even improves for $K = 1$ with respect to the FARGO one, with at least two iterations.

Score disparity outputs - Music data-set

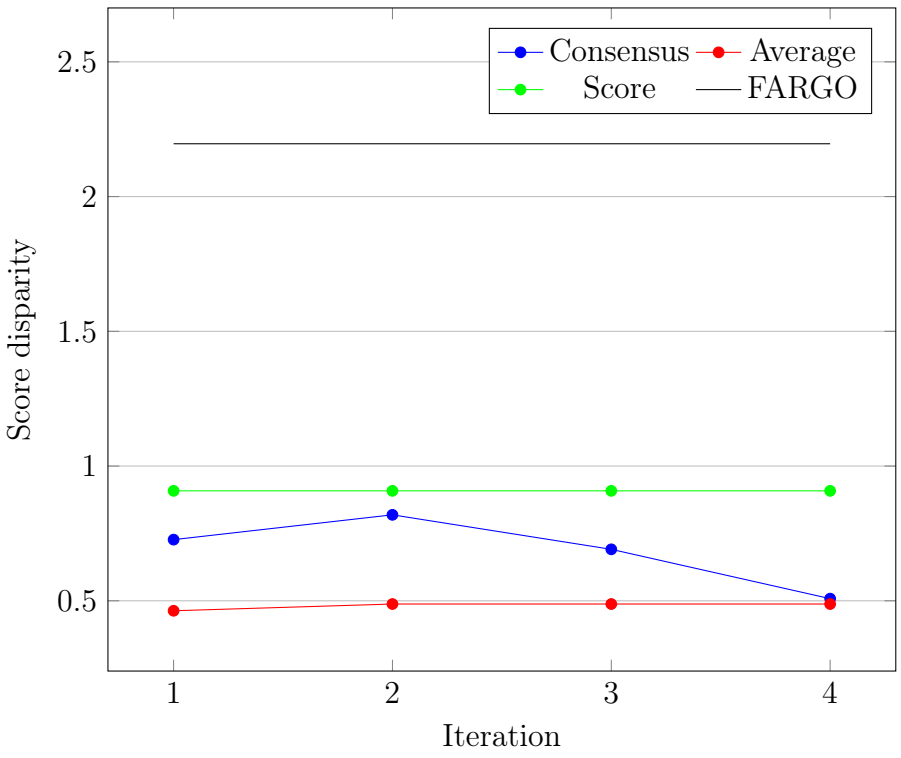


Figure 5.10. Score disparity varying iterations Music data-set $K = 1$

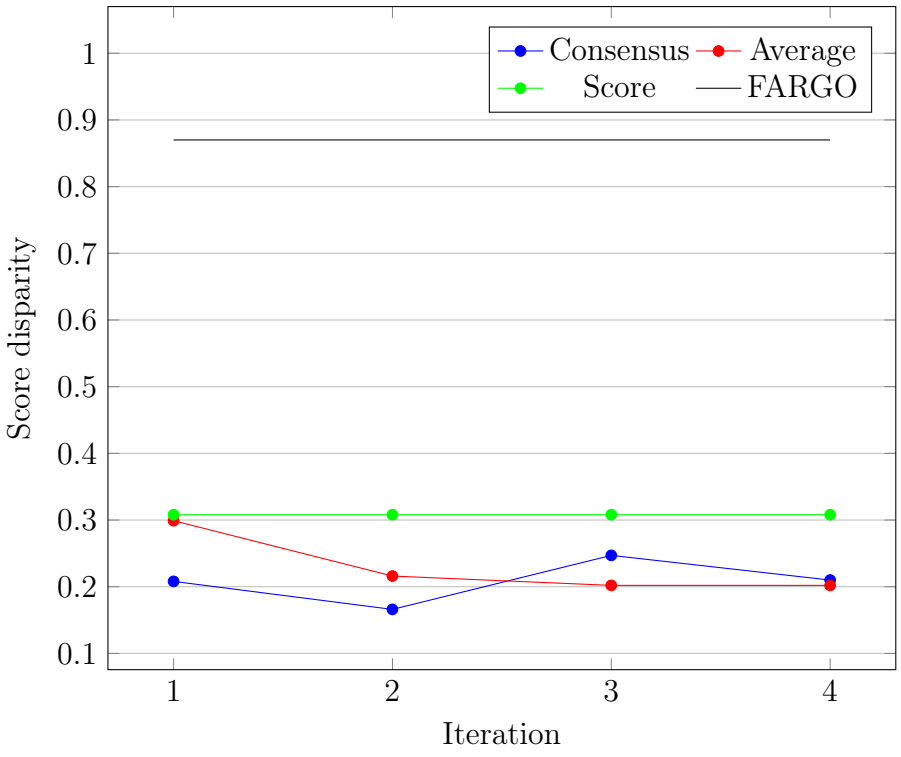


Figure 5.11. Score disparity varying iterations Music data-set $K = 2$

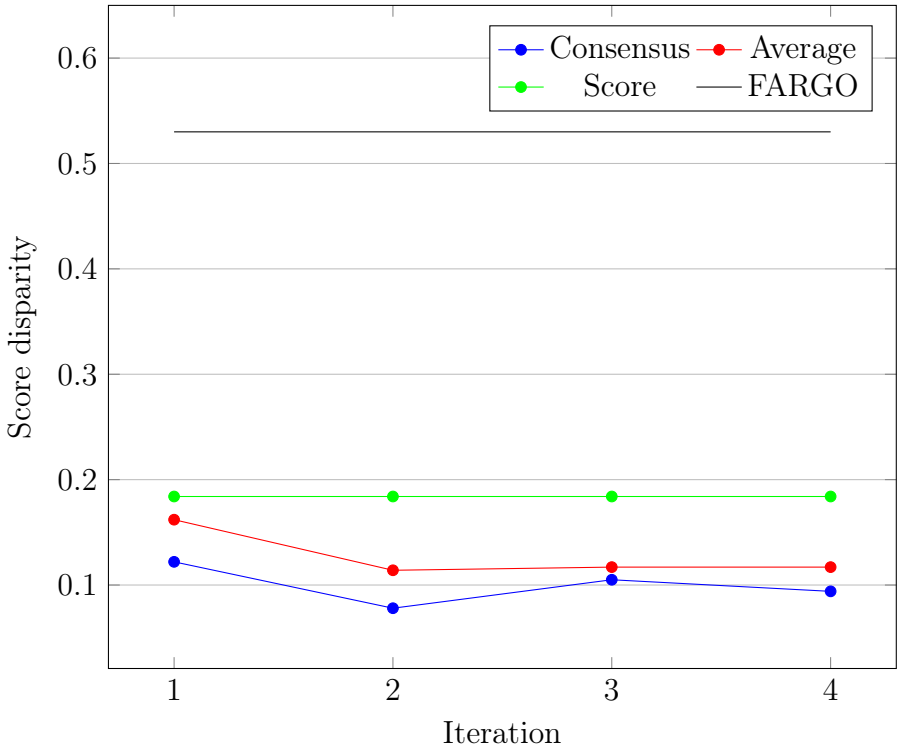


Figure 5.12. Score disparity varying iterations Music data-set $K = 3$

Plots 5.10, 5.11 and 5.12 above show that with the Music data-set Score disparity improves a lot for all values of K. The number of iterations doesn't significantly affect the outputs. In particular:

- Score disparity by using the formula based on **consensus** is pretty good, and it is the best with $K = 1$, contrarily to what we observed in the case of Auditel data-set;
- Score disparity improves also by using the formula based on **average** and it shows a good trend, especially with $K = 1$;
- unlike the previous data-set, Score disparity is the worst using the formula based on **score** for any value of K.

Recommendation disparity outputs - Auditel-dataset

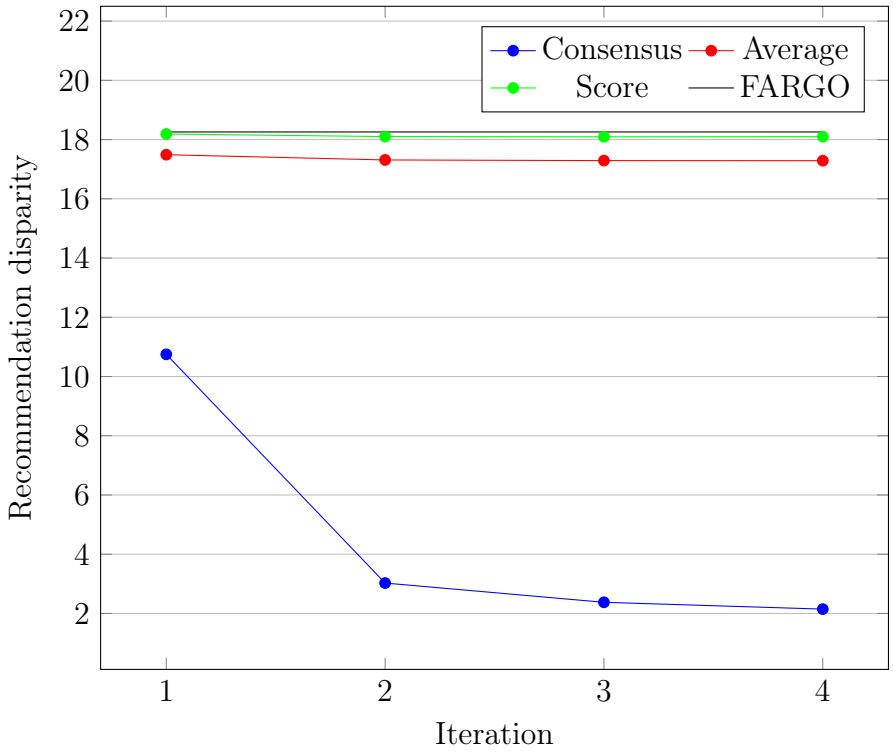


Figure 5.13. Recommendation disparity varying iterations Auditel data-set $K = 1$

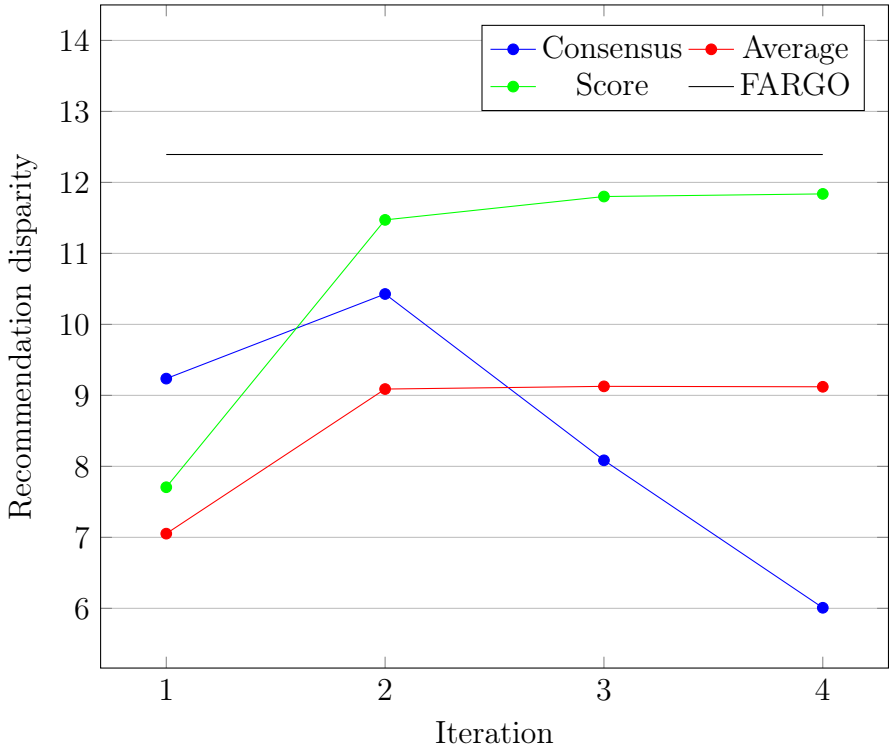


Figure 5.14. Recommendation disparity varying iterations Auditel data-set $K = 2$

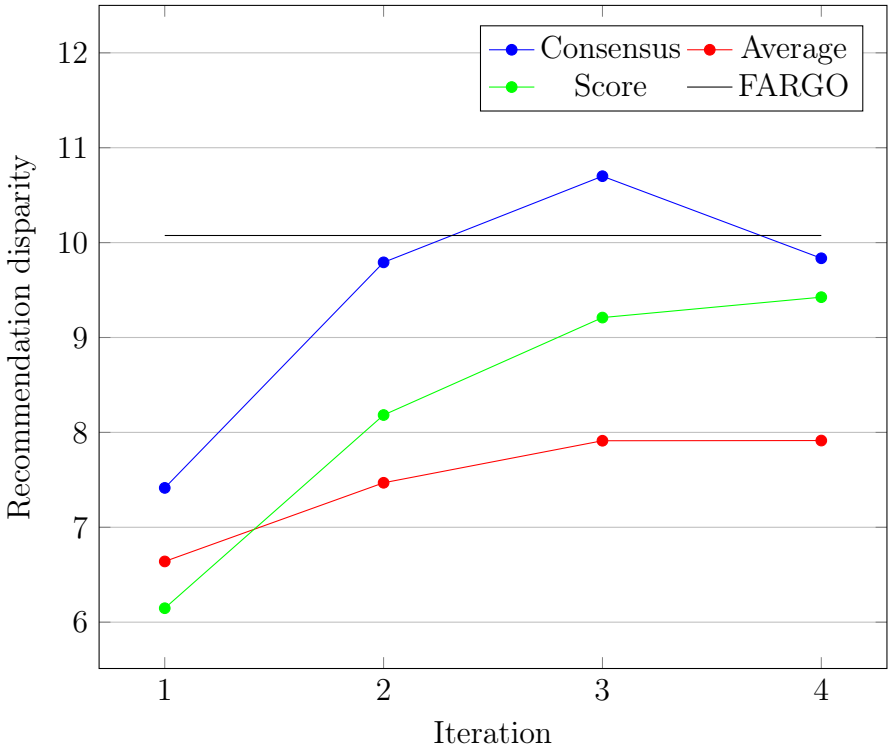


Figure 5.15. Recommendation disparity varying iterations Auditel data-set $K = 3$

Plots 5.13, 5.14 and 5.15 above show that in the case of the Auditel data-set Recommendation disparity improves for any values of K and for any number of iterations. The three plots do not show the same behavior, i.e. the best value of recommendation score changes depending on the value of K and the number of iteration. In particular:

- Recommendation disparity by using the formula based on **consensus** is better with $K = 1$ and $K = 2$ with at least two iterations;
- Recommendation disparity improves also by using the formula based on **average** and shows a good trend, especially $K = 3$ and at least two iterations;
- Recommendation disparity is the worst using the formula based on **score** with respect to the others equations, but the outputs are still better than FARGO.

Recommendation disparity outputs - Music data-set

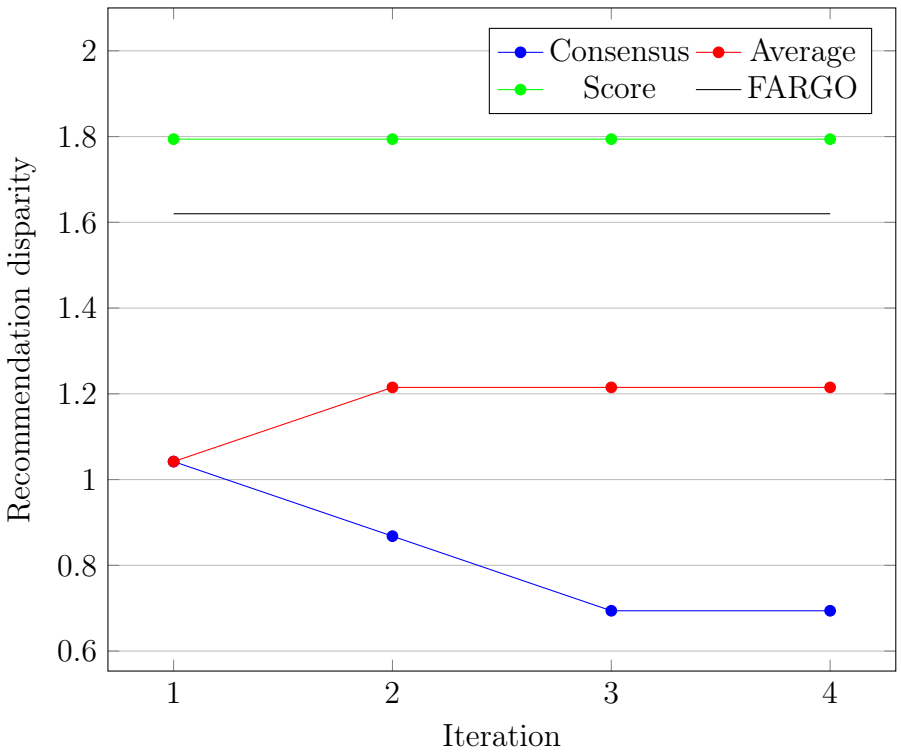


Figure 5.16. Recommendation disparity varying iterations Music data-set $K = 1$

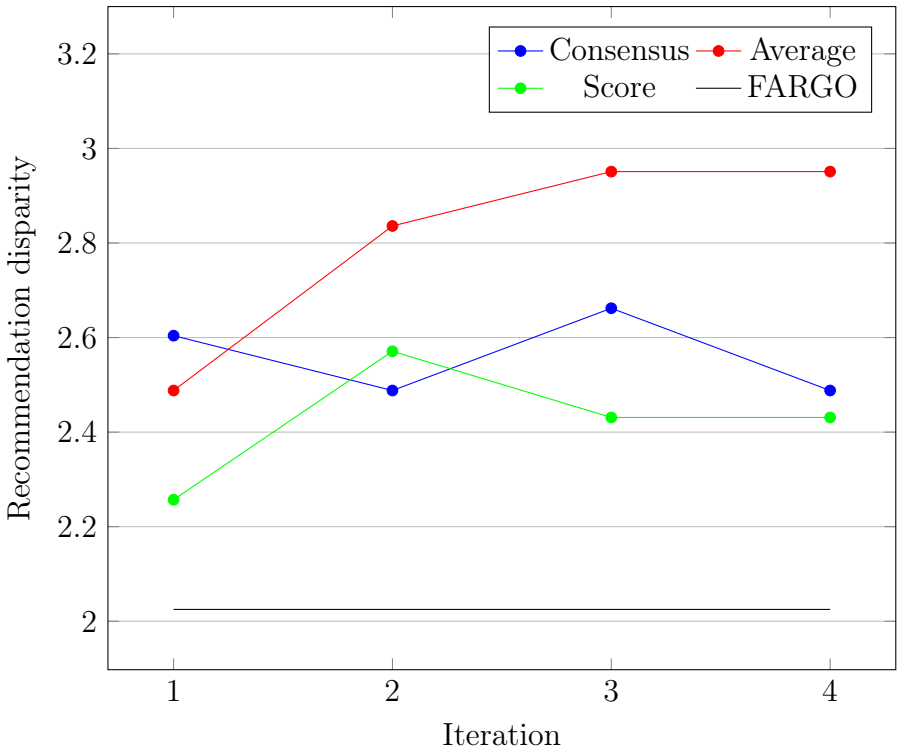


Figure 5.17. Recommendation disparity varying iterations Music data-set $K = 2$

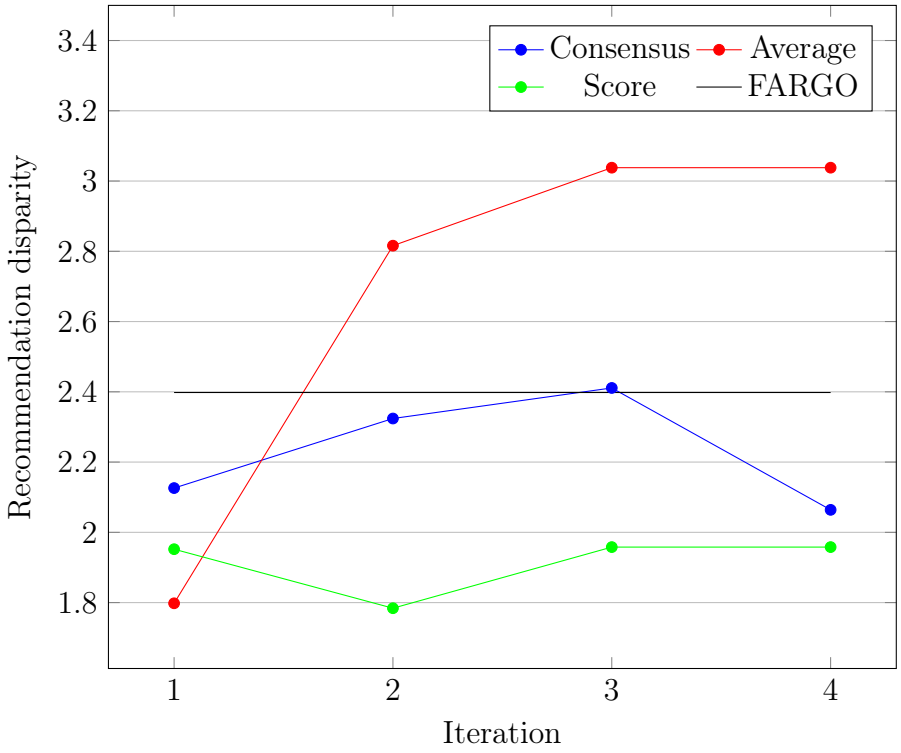


Figure 5.18. Recommendation disparity varying iterations Music data-set $K = 3$

Plots 5.16, 5.17 and 5.18 above show that in the case of the Music data-set Recommendation disparity has contrasting outcome, depending on the value of K and on the number of iterations: it improves for $K = 1$ and $K = 3$, but gets worse with $K = 2$. In particular:

- Recommendation disparity using the formula based on **consensus** is best with $K = 1$;
- Recommendation disparity is the worst than FARGO one using the formula based on **average** for values of K greater than 1;
- Recommendation disparity improves also using the formula based on **score** with $K = 3$, but in other cases is worst than FARGO one.

AVG consensus outputs - Auditel data-set

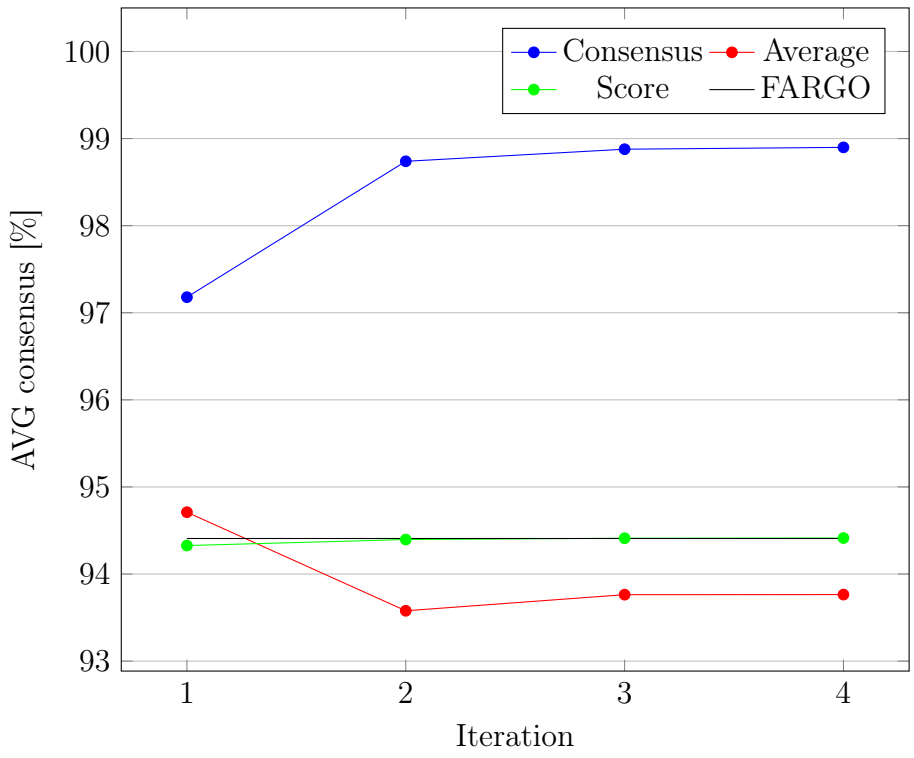


Figure 5.19. AVG consensus varying iterations Auditel data-set $K = 1$

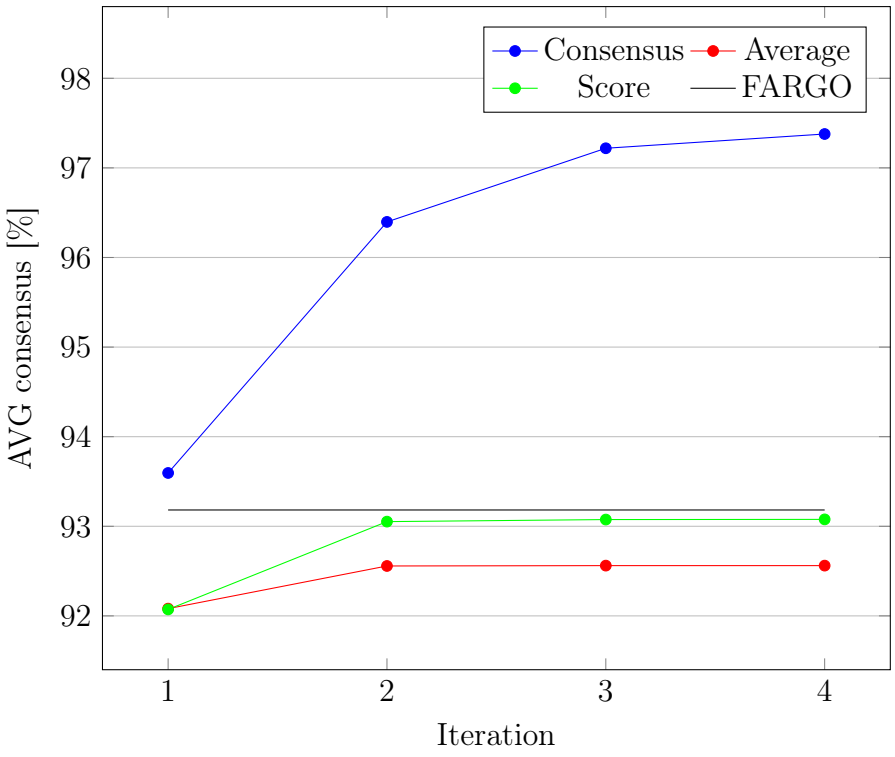


Figure 5.20. AVG consensus varying iterations Auditel data-set $K = 2$

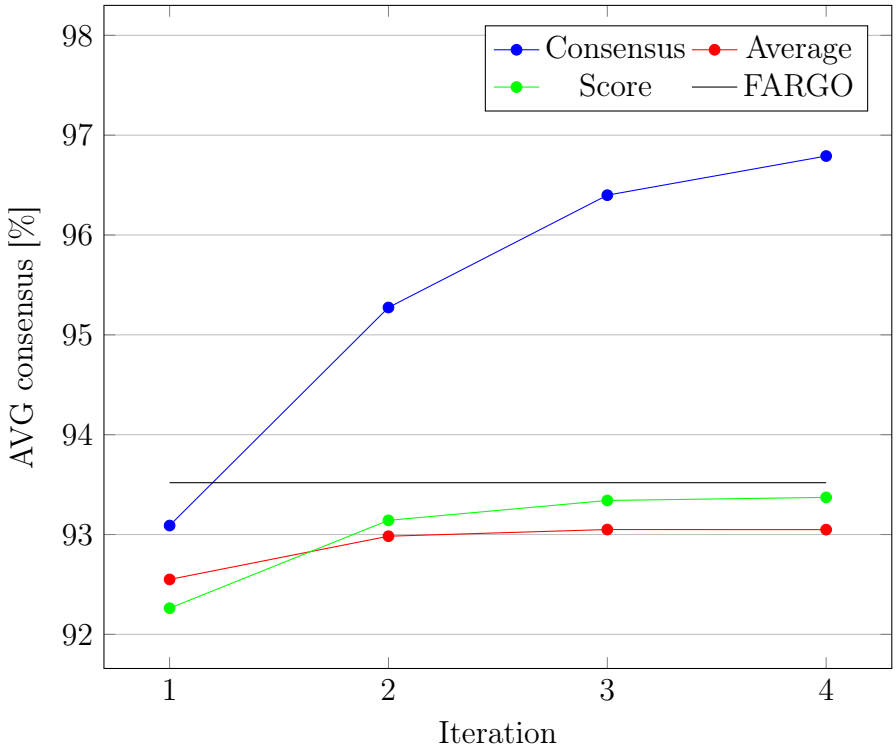


Figure 5.21. AVG consensus varying iterations Auditel data-set $K = 3$

Plots 5.19, 5.20 and 5.21 above show that in the case of the Auditel data-set AVG consensus improves for any value of K. In particular:

- AVG consensus using the formula based on **consensus** is so far the best among all, especially with more iterations and also better than the FARGO one;
- AVG consensus is the worst using the formula based on **average**, mostly with more iterations;
- AVG consensus is pretty the same of FARGO using the formula based on **score** with $K = 1$, and in other cases is however acceptable.

AVG consensus outputs - Music data-set

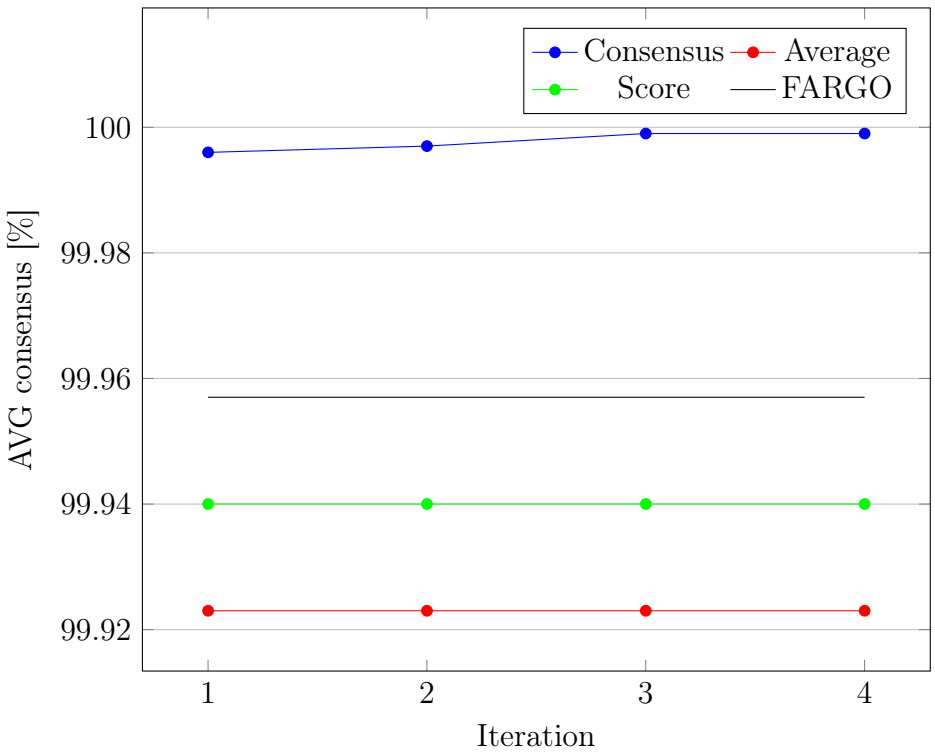


Figure 5.22. AVG consensus varying iterations Music data-set $K = 1$

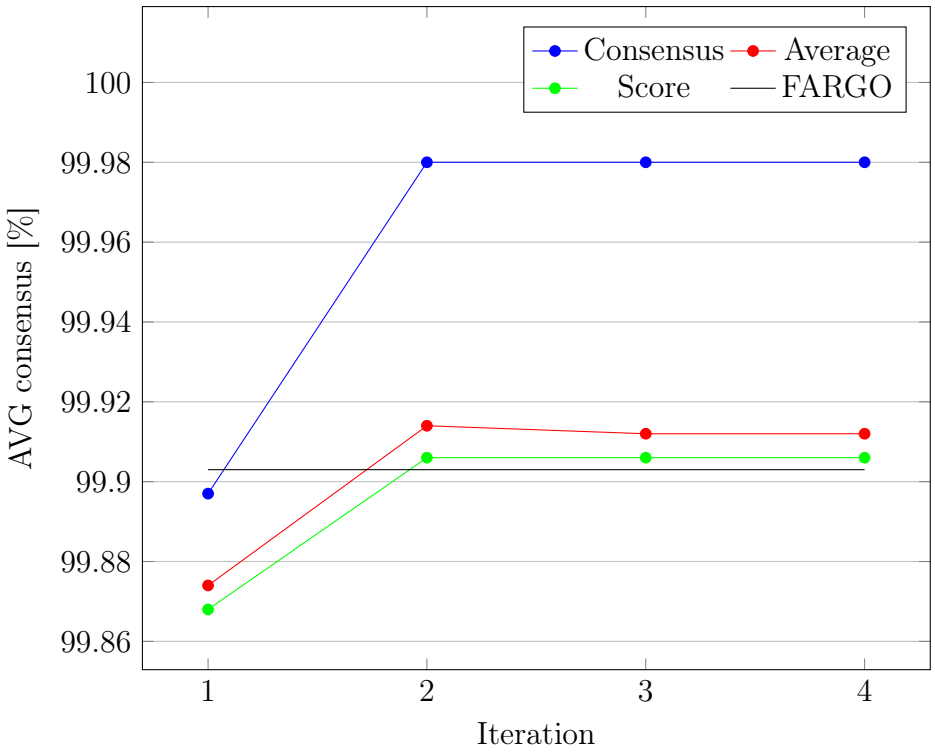


Figure 5.23. AVG consensus varying iterations Music data-set $K = 2$

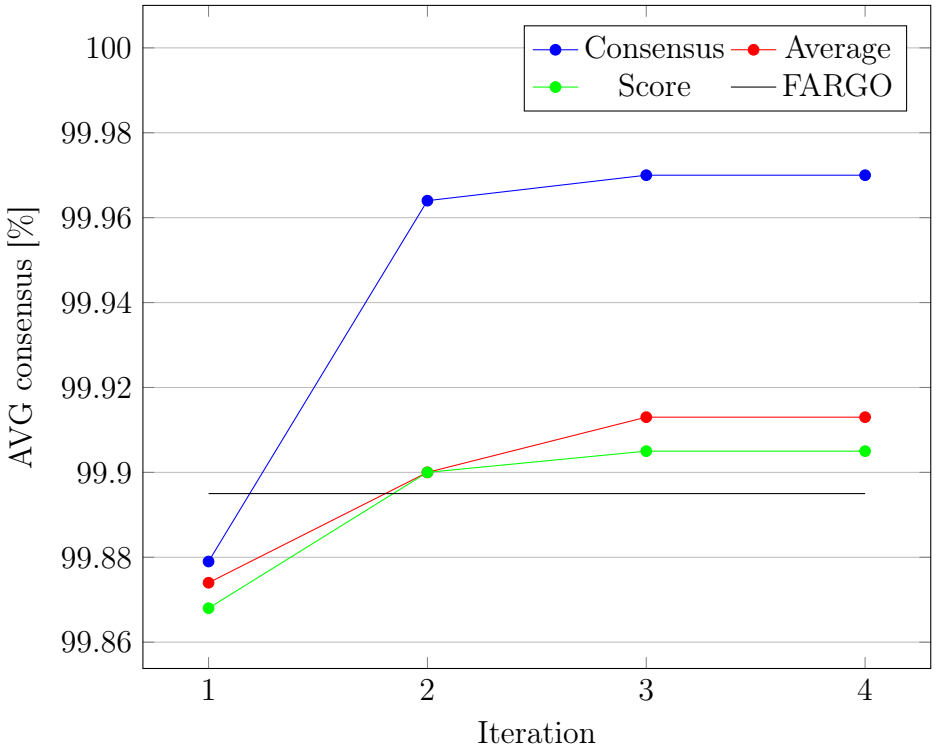


Figure 5.24. AVG consensus varying iterations Music data-set $K = 3$

Plots 5.22, 5.23 and 5.24 above show that in the case of the Music data-set AVG consensus follows the behavior that has with the Auditel one, but with higher values. In particular:

- AVG consensus using the formula based on **consensus** is again so far the best among all, especially with at least iterations;
- AVG consensus is the worst using the formula based on **average** with $K = 1$, but with at least two iterations is good with other values of K;
- using the formula based on **score** AVG consensus is pretty the same of the average one.

5.3.5 Considerations

As we can see from the results just presented the Recall is affected by a drastic decrease as the number of iteration increases using the formula based on consensus. This is due to the fact that iterating the filter means adding new items to the $TopK$ since we may find sub-optimal items (in terms of score) that have a very similar evaluation by the users. This produces a very high consensus value, even though there could be items already taken in consideration in previous iterations that of course have higher scores. For example if we have a score situation like this:

	u_1	u_2
i_1	0.7	0.8
i_2	0.5	0.5

Table 5.26. Example: consensus based iteration

Based on the table 5.26, at the first iteration of the filter we get only item i_1 and its consensus is: 0.93. When we take the second iteration (so we are moving to non optimal preference items) we must evaluate also i_2 and we get that the consensus of i_2 is 1. Obviously i_2 will be placed first and i_1 will come after. This of course explains why the more iterations we take, the more Recall decreases. Because of the way consensus is defined, i.e., not taking into account how much "high" or "low" the ratings are, it only takes into account the diversity among them. So, ordering purely by consensus it is not a good strategy, especially with $K = 1$, despite the fact that the values that measure fairness perform fairly well compared to the other two methods. Moreover, it is really difficult to choose between the other two proposals, as neither is clearly better than the other:

- in terms of Recall they assume quite similar values, especially considering multiple iterations and they remain competitive with the FARGO starting Recall;
- for the Score disparity the formula based on the average allows to get better results if we consider the experiments with Music data-set (significantly improving the starting ones of FARGO), while if we consider the Auditel data-set the formula based on the score prevails (even if the first formula actually performs better for $K = 1$ and improves the starting one);
- for the Recommendation disparity the formula based on the average clearly prevails if we consider the Auditel data-set or the Music data-set with $K = 1$, but also the formula based on the score performs well when compared to the Recommendation disparity by FARGO. Note that both, however, on both data-sets worsen relatively if we increase the number of iterations;
- AVG consensus is certainly better with the score-based formula and multiple iterations using the Auditel data-set, while with the Music data-set the values are more or less equivalent.

As consequence of this analysis, we consider the two proposals (formula based on average and formula based on consensus) both valid and a choice between them can be done depending on what we want to improve the most:

- if we consider the objective of improving the Score disparity, i.e. minimize the difference in users' satisfaction, then the formula based on the score is better;
- if, on the other hand, the objective is to improve the Recommendation disparity, i.e. to make sure to choose the same number of items from the $TopK$ of all users, then the formula based on average is preferable.

5.4 Zero as negative feedback vs zero as doubt

In this section, the outputs of the algorithm after the improvements that we have proposed in section 4.2 regarding the estimation of zeros will be analyzed, but only using the Auditel data-set.
The reason why we decided to take into consideration only this data-set and to exclude the music one can be found in their structures: in the first case, i.e. Auditel data-set, a user's preference for a item equal to 0 means that the user has never watched the channel, and that therefore he could still like it even if he has never had the opportunity to watch it, while in the second case, i.e. Music data-set, the user scores range on a scale from 0 to 4, so zero means non liking a certain artist.

5.4.1 Similar users for unknown items

In this experiment we have took into consideration the scores equal to 0 that are in the Auditel data-set and which are used for the calculation of the consensus.
We have replaced them with an estimation of the user's preference for the particular item, based on the evaluations of users considered similar to him by using the Algorithm 3.

Algorithm 3

	Recall	D_S	D_R	AVG Consensus
Before	37,894	7,693	18,257	94,408
After	37,925	7,685	18,224	94,336

Table 5.27. Optimizing based on Algorithm 3 Auditel data-set $K = 1$

	Recall	D_S	D_R	AVG Consensus
Before	54,149	1,9	12,393	93,182
After	54,369	1,926	12,164	93,114

Table 5.28. Optimizing based on Algorithm 3 Auditel data-set $K = 2$

	Recall	D_S	D_R	AVG Consensus
Before	64,26	0,889	10,075	93,519
After	64,517	0,898	9,803	93,479

Table 5.29. Optimizing based on Algorithm 3 Auditel data-set $K = 3$

5.4.2 Popular items

Similarly to the previous experiment, also in this one we substituted the scores equal to 0 used for the calculation of the consensus.

This time we decided to replace them with an estimation of the user's preference for the particular item based on general evaluations that the item has(i.e. its popularity). In the 4.2.2 section Equations 4.5 and 4.6 have been proposed and below they will be analyzed one by one.

Equation 4.5

	Recall	D_S	D_R	AVG Consensus
Before	37,894	7,693	18,257	94,408
After	37,907	7,697	18,269	94,374

Table 5.30. Optimizing based on equation 4.5 Auditel data-set $K = 1$

	Recall	D_S	D_R	AVG Consensus
Before	54,149	1,9	12,393	93,182
After	54,086	1,899	12,344	93,158

Table 5.31. Optimizing based on equation 4.5 Auditel data-set $K = 2$

	Recall	D_S	D_R	AVG Consensus
Before	64,26	0,889	10,075	93,519
After	64,193	0,885	10,032	93,502

Table 5.32. Optimizing based on equation 4.5 Auditel data-set $K = 3$

Equation 4.6

	Recall	D_S	D_R	AVG Consensus
Before	37,894	7,693	18,257	94,408
After	37,901	7,693	18,256	94,406

Table 5.33. Optimizing based on equation 4.6 Auditel data-set $K = 1$

	Recall	D_S	D_R	AVG Consensus
Before	54,149	1,9	12,393	93,182
After	54,153	1,899	12,405	93,182

Table 5.34. Optimizing based on equation 4.6 Auditel data-set $K = 2$

	Recall	D_S	D_R	AVG Consensus
Before	64,26	0,889	10,075	93,519
After	64,265	0,89	10,076	93,519

Table 5.35. Optimizing based on equation 4.6 Auditel data-set $K = 3$

5.4.3 Both similar users and popular item

Finally we also tried to combine the two ideas: we proposed a new score that is the arrhythmic mean of the two methods.

Algorithm 3 and Equation 4.5

	Recall	D_S	D_R	AVG Consensus
Before	37,894	7,693	18,257	94,408
After	37,905	7,697	18,256	94,374

Table 5.36. Optimizing based on algorithm 3 and Equation 4.5 Auditel data-set $K = 1$

	Recall	D_S	D_R	AVG Consensus
Before	54,149	1,9	12,393	93,182
After	54,086	1,9	12,341	93,157

Table 5.37. Optimizing based on Algorithm 3 and Equation 4.5 Auditel data-set $K = 2$

	Recall	D_S	D_R	AVG Consensus
Before	64,26	0,889	10,075	93,519
After	64,212	0,887	10,035	93,503

Table 5.38. Optimizing based on Algorithm 3 and Equation 4.5 Auditel data-set $K = 3$

Algorithm 3 and Equation 4.6

	Recall	D_S	D_R	AVG Consensus
Before	37,894	7,693	18,257	94,408
After	37,917	7,685	18,225	94,34

Table 5.39. Optimizing based on Algorithm 3 and Equation 4.6 Auditel data-set $K = 1$

	Recall	D_S	D_R	AVG Consensus
Before	54,149	1,9	12,393	93,182
After	54,339	1,926	12,185	93,119

Table 5.40. Optimizing based on Algorithm 3 and Equation 4.6 Auditel data-set $K = 2$

	Recall	D_S	D_R	AVG Consensus
Before	64,26	0,889	10,075	93,519
After	64,514	0,894	9,826	93,481

Table 5.41. Optimizing based on Algorithm 3 and Equation 4.6 Auditel data-set $K = 3$

5.4.4 Comparison of outputs

In this subsection, by showing plots, the results presented so far will be analyzed: all the improvements will be compared among them and with the initial outputs of FARGO.
In order, Recall, Score disparity, Recommendation disparity and the Average of consensus are analyzed for the values of K equal to 1, 2, 3.
In the following plots on the x-axis the improvements are listed in this way:

- improvement with Algorithm 3 (Sim);
- improvement with Equation 4.5 ($Pop1$);
- improvement with Equation 4.6 ($Pop2$);
- improvement with Algorithm 3 and Equation 4.5 ($Combo1$);
- improvement with Algorithm 3 and Equation 4.6 ($Combo2$).

Recall outputs

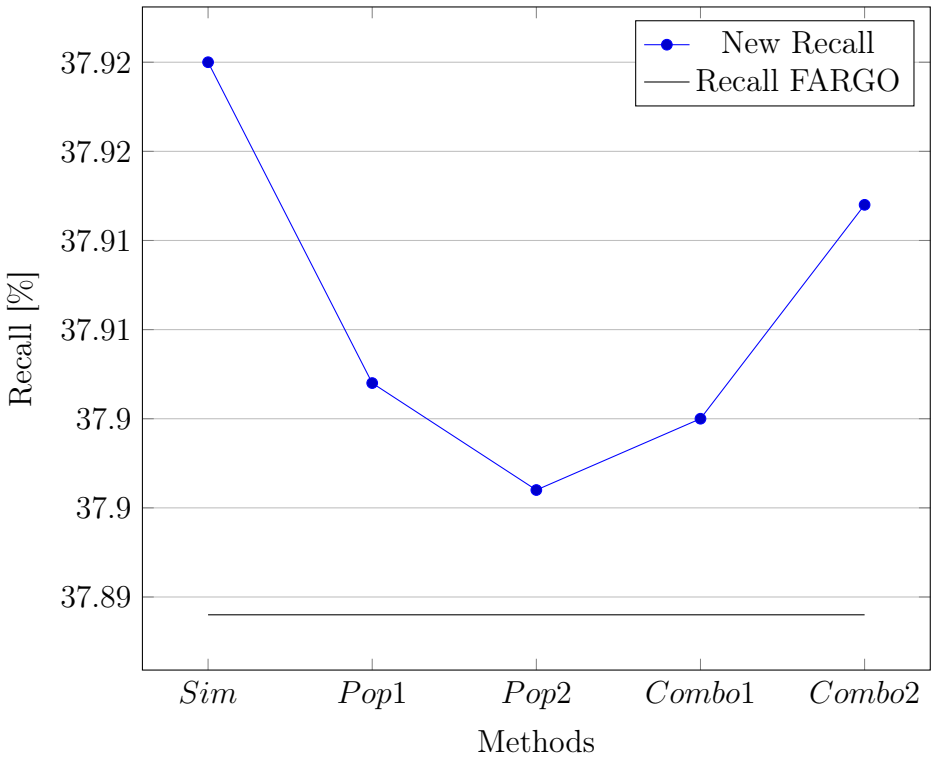


Figure 5.25. Recall Auditel data-set $K = 1$

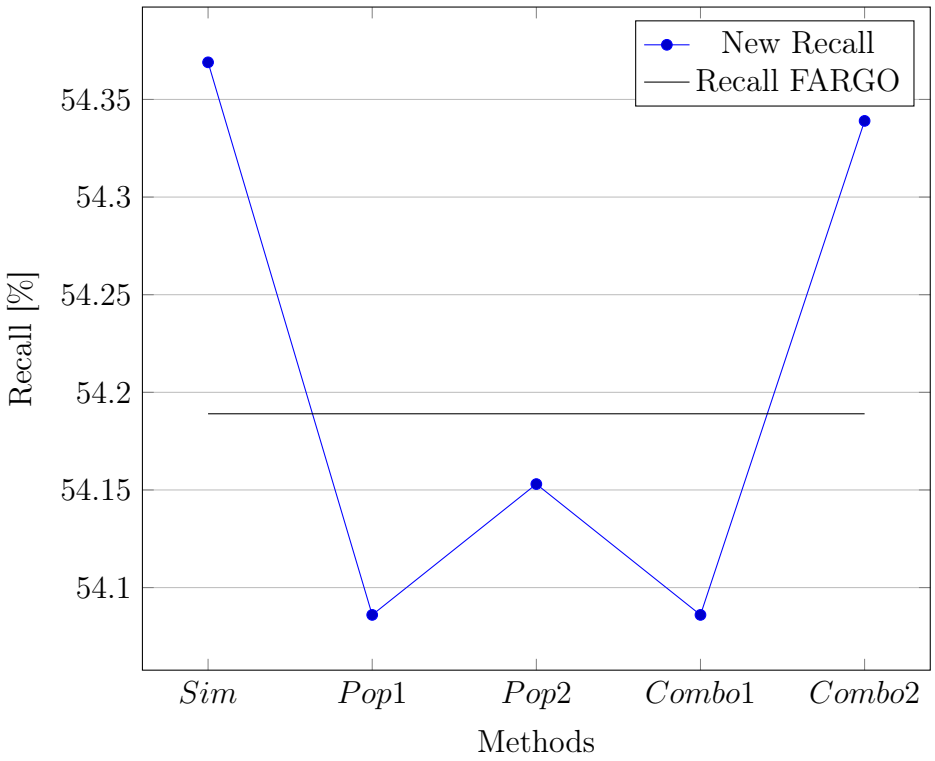


Figure 5.26. Recall Auditel data-set $K = 2$

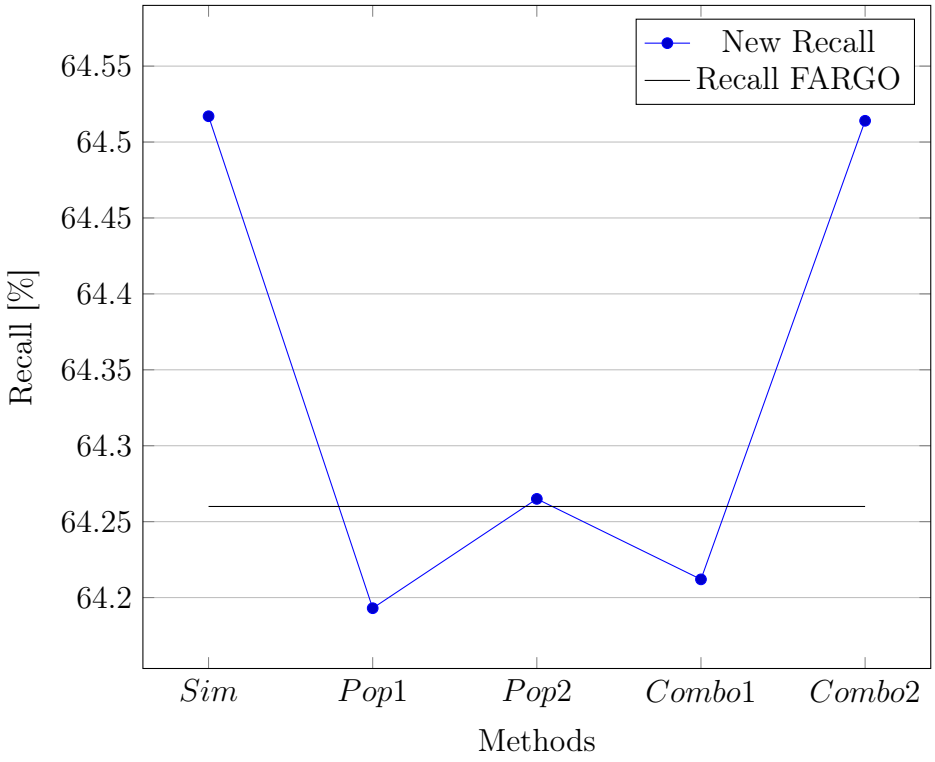


Figure 5.27. Recall Auditel data-set $K = 3$

Plots 5.25, 5.26 and 5.27 above show that Recall has a good behavior with the Auditel data-set when we use Algorithm 3 (*Sim*). In particular:

- with $K = 1$ Recall increases with all the improvements;
- with $K = 2$ only with *Sim* and *Combo*2 Recall gets better, but in the other cases it doesn't go down so much;
- the Recall's trend with $K = 3$ is more or less the same with $K = 2$, with the difference that worsening is really minimal.

Score disparity outputs

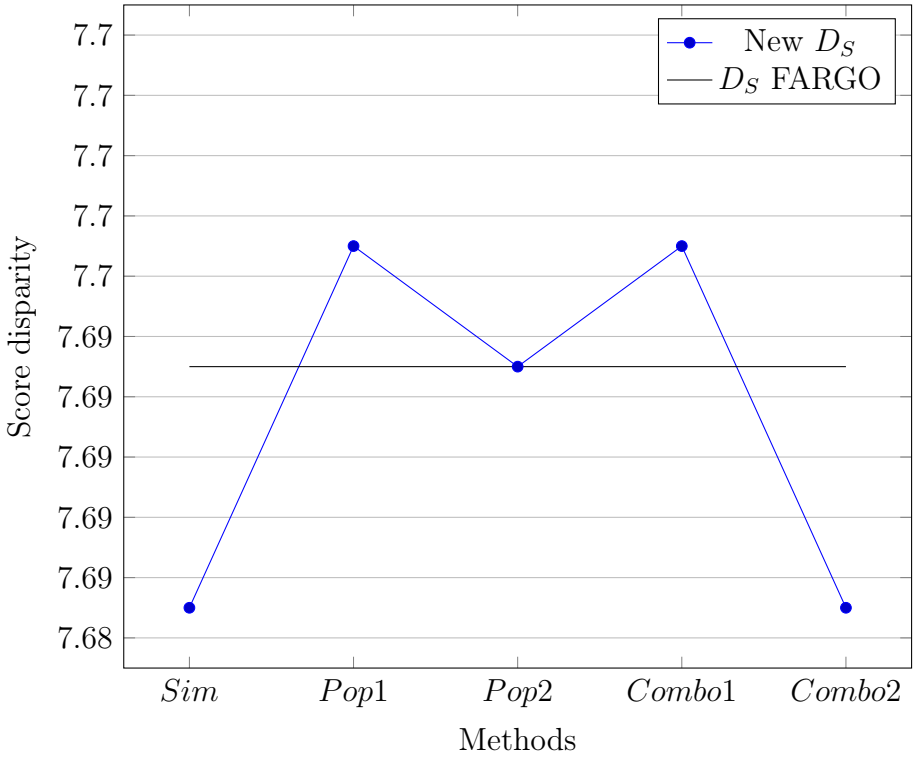


Figure 5.28. Score disparity Auditel data-set $K = 1$

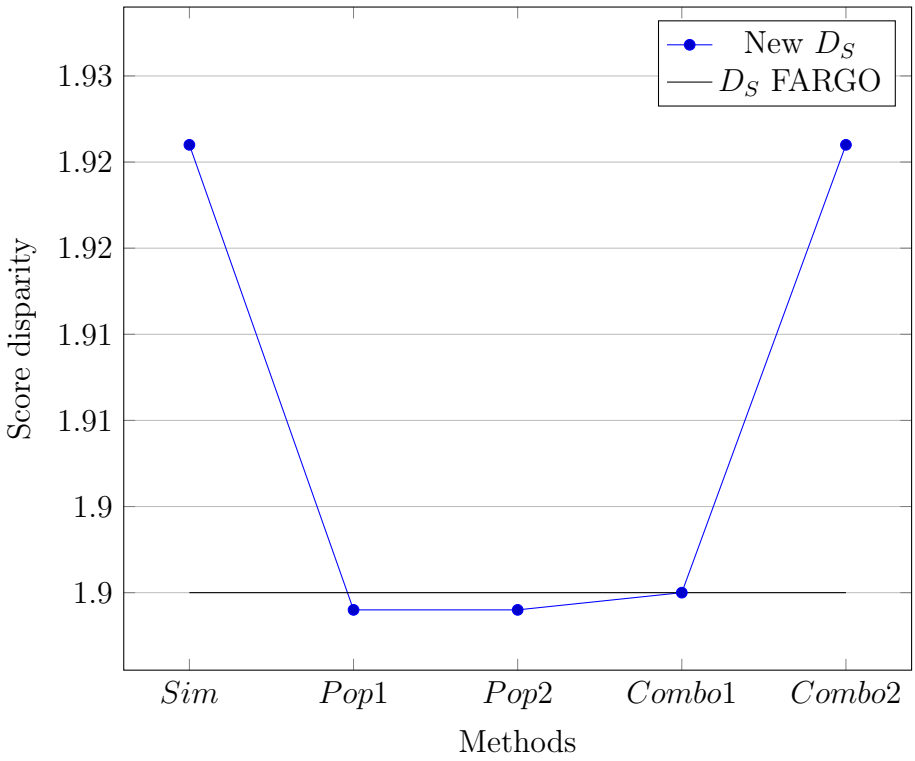


Figure 5.29. Score disparity Auditel data-set $K = 2$

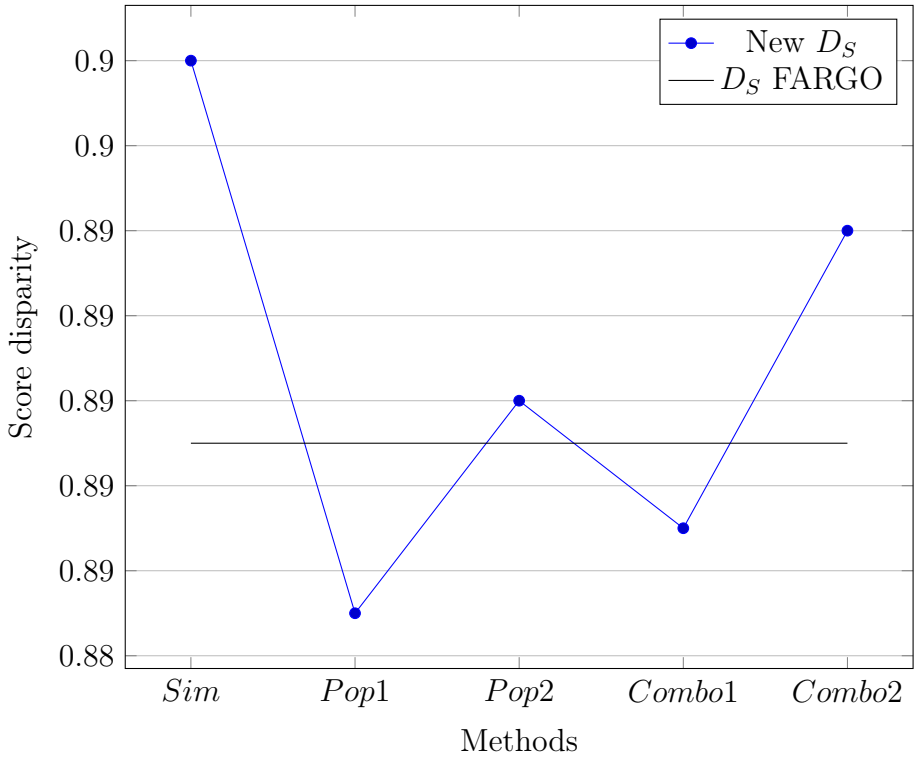


Figure 5.30. Score disparity Auditel data-set $K = 3$

Plots 5.28, 5.29 and 5.30 above show that Score disparity has the following characteristics with the Auditel data-set:

- with $K = 1$ Score disparity has the same behavior of Recall with values of K greater than 1, i. e. it improves with Sim and $Combo2$;
- Score disparity with $K = 2$ grows up with Sim and $Combo2$, but it remains practically the same as the old value;
- Score disparity with $K = 3$ gets better with $Pop1$ and $Combo1$ and gets worse in the other cases, but also in this case all the values are practically the same as the old ones.

Recommendation disparity outputs

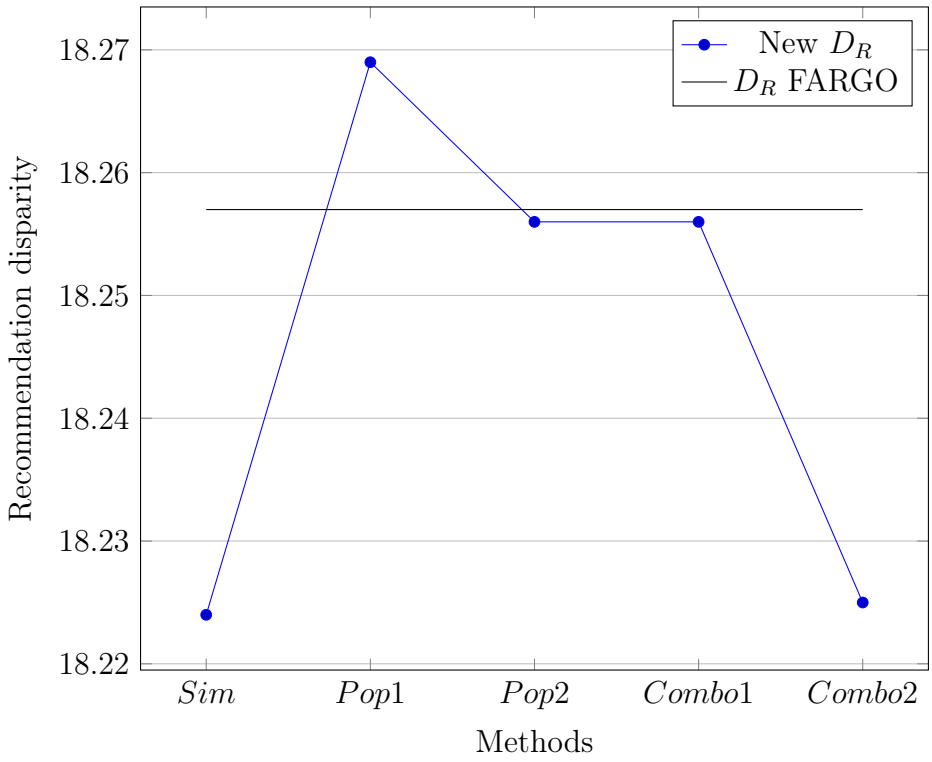


Figure 5.31. Recommendation disparity Auditel data-set $K = 1$

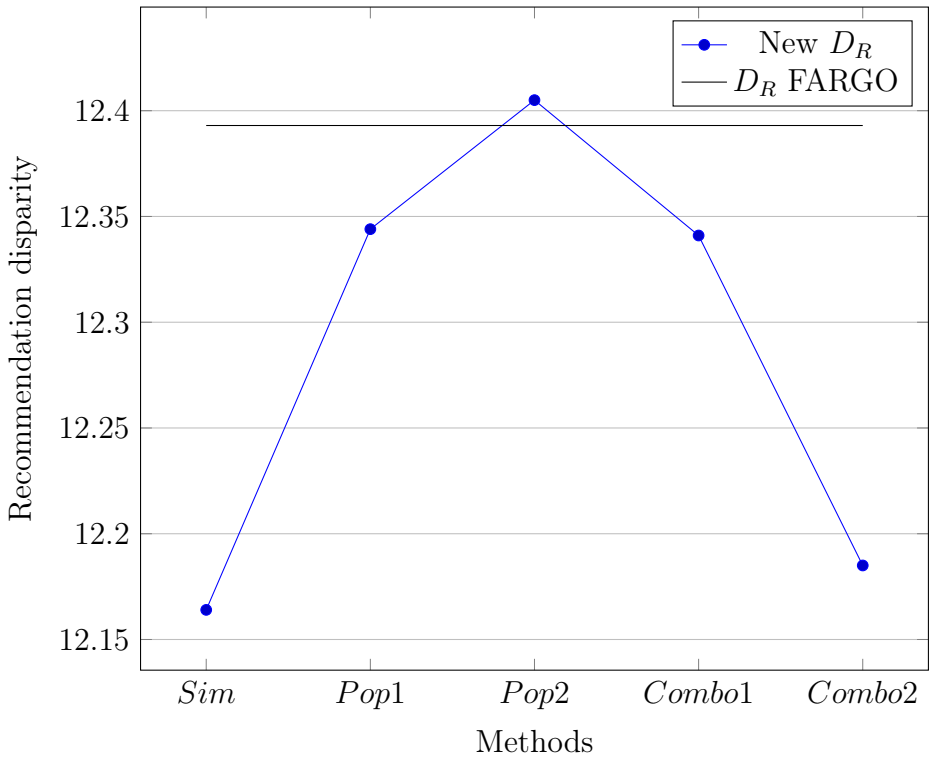


Figure 5.32. Recommendation disparity Auditel data-set $K = 2$

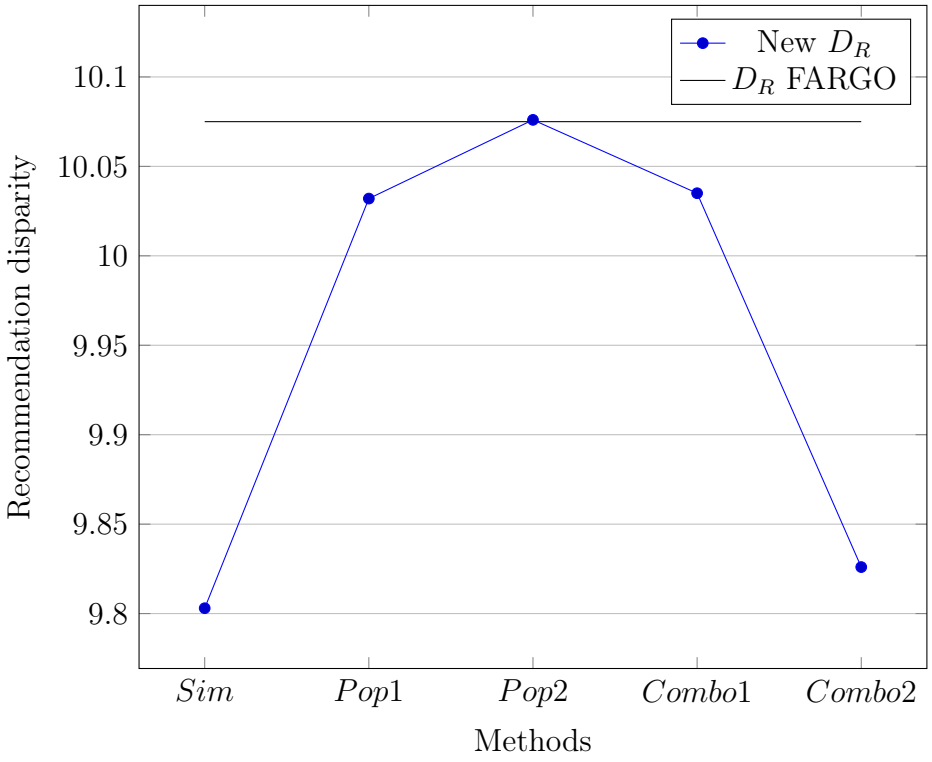


Figure 5.33. Recommendation disparity Auditel data-set $K = 3$

The plots 5.31, 5.32 and 5.33 above show as Recommendation disparity has a non-homogeneous behavior. in detail:

- with $K = 1$ Recommendation disparity improves more or less with every improvement, except for $Pop1$;
- Recommendation disparity with $K = 2$ gets better in particular with Sim and $Combo2$, but it is good also with $Pop1$ and $Combo1$;
- the Recommendation disparity with $K = 3$ has the same trend as it does with $K = 2$.

AVG consensus outputs

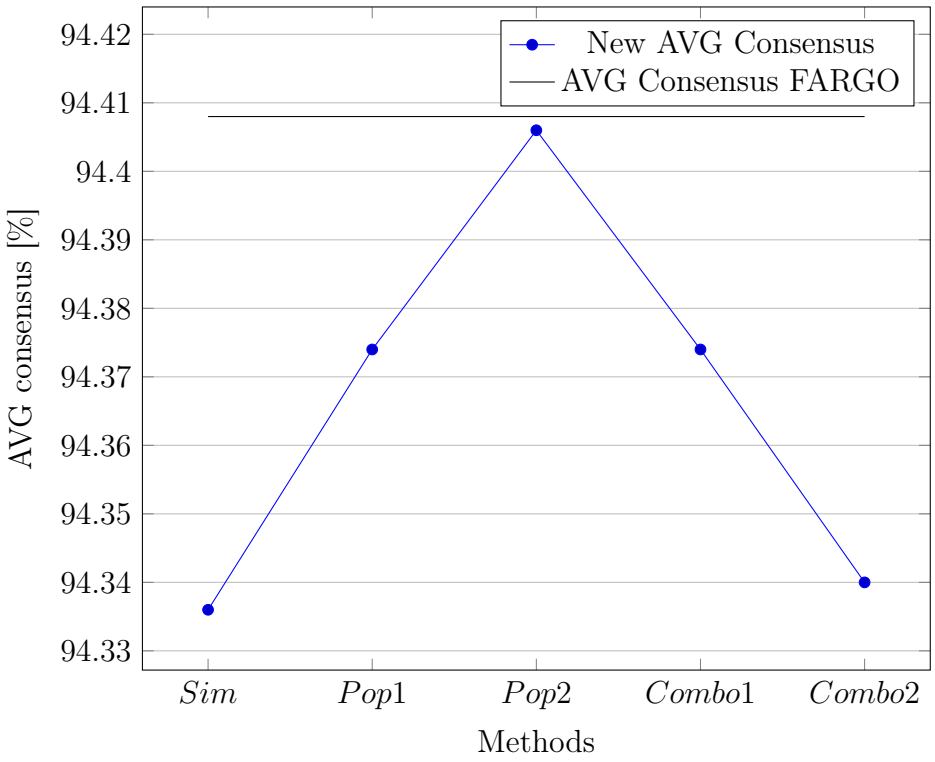


Figure 5.34. AVG consensus Auditel data-set $K = 1$

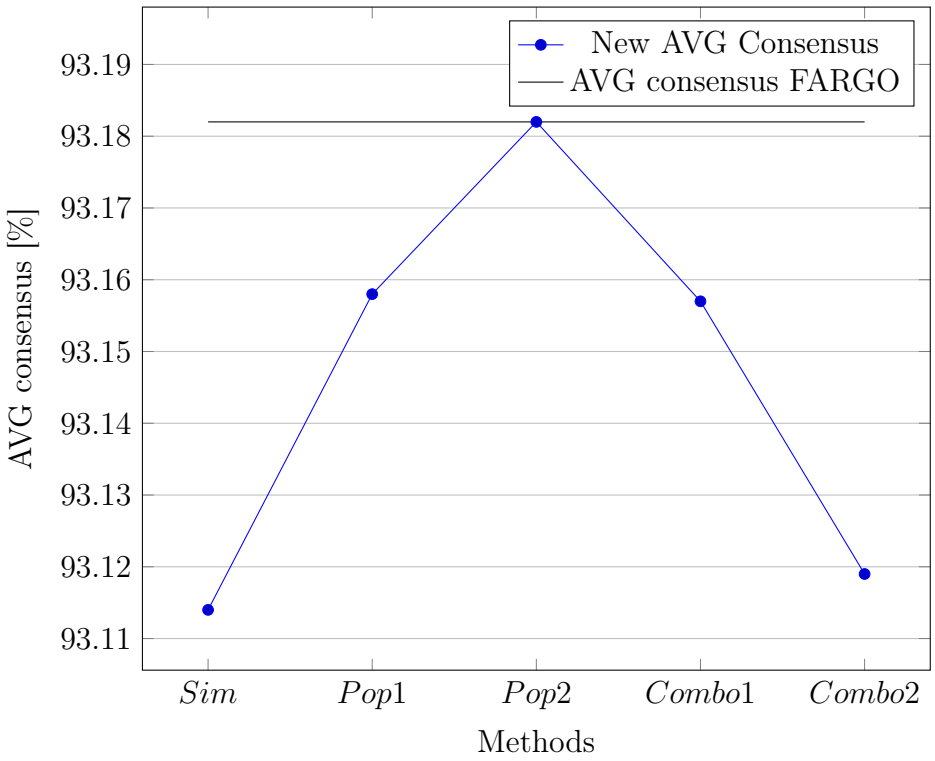


Figure 5.35. AVG consensus Auditel data-set $K = 2$

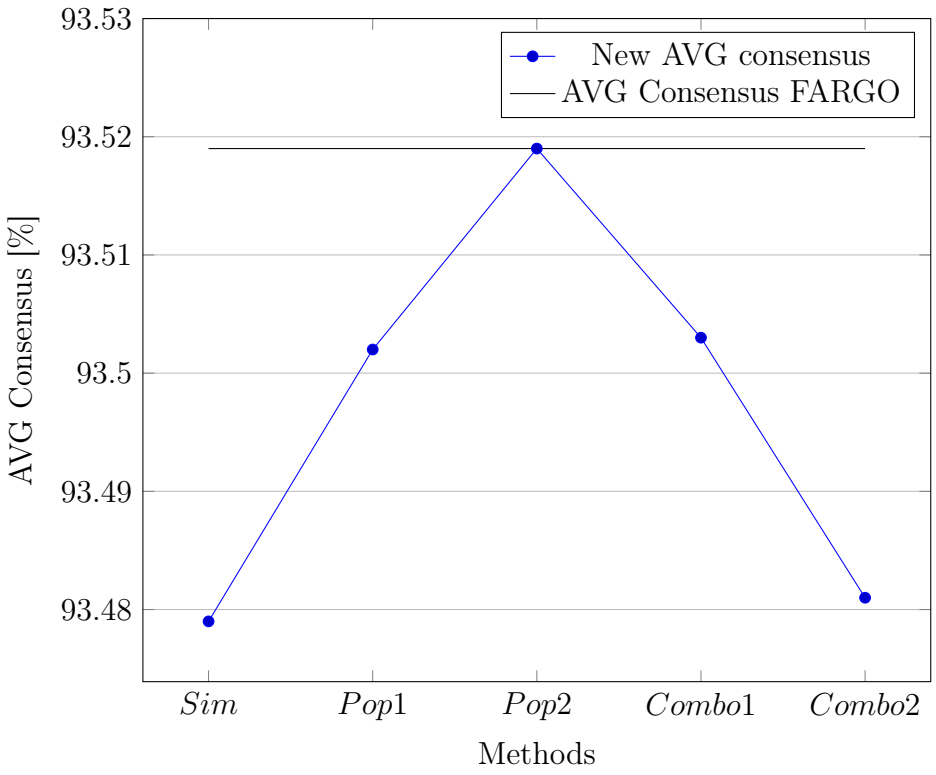


Figure 5.36. AVG consensus Auditel data-set $K = 3$

Plots 5.34, 5.35 and 5.36 above show that AVG consensus has more or less the same trend with the Auditel data-set for all value of K, with the exception that it gets slightly worse with all improvements, but not with $Pop2$.

5.4.5 Considerations

By considering subsection 5.4.4, in which the various improvements are compared with each other, the following analysis can be done:

- in terms of Recall the best improvements are the Algorithm 3 and its mix with equation 4.6;
- the Score disparity doesn't change so much and there isn't a specific method that improves it for any value of K;
- the Recommendation disparity improves with all method, and in particular, like the Recall, with the Algorithm 3 and its mix with equation 4.6;
- AVG consensus gets worse practically with all our improvements, but the differences with the old one are not so relevant.

Therefore, the conclusions we can draw can be summarized by saying that Algorithm 3 is the best improvement as it increases the performance of FARGO in terms of Recall and Recommendation disparity without worsening the other measures. Furthermore, the mix with Equation 4.6 is also quite good and behaves similarly to the Algorithm 3 alone. By supporting this thesis, we do not want to belittle the other proposals which still remain valid alternatives to those mentioned above.

Chapter 6

Fairness Measures Experimental

In this chapter we analyze the results of the new proposed fairness measures. The competitor algorithms will first be compared with FARGO using the new fairness metrics. Then, the new fairness metrics will be compared with the existing ones (i.e. Score disparity and Recommendation disparity) applying all of them on FARGO in order to understand similarities and differences.

6.1 New Fairness Measures with Competitors

In this section we report some experimental results involving our new fairness measures. We implemented all the new measures (equations 4.7, 4.8 and 4.9), and then we ran FARGO and all the other competitor algorithms. In the following subsections we will compare and analyze the results.

Raw Results for Auditel Data-Set

	$K = 1$	$K = 2$	$K = 3$
FARGO	0,262	0,369	0,472
AVG	0,26	0,378	0,468
LM	**0,24**	**0,35**	**0,44**
MS	0,257	0,38	0,472
DIS	0,268	0,382	0,474
FAIR-LIN	0,258	0,394	0,479
FAIR-PROP	0,287	0,402	0,531
EXP	0,272	0,373	0,463
ENVY-FREE	0,26	0,377	0,468

Table 6.1. Output equation 4.7 Auditel data-set

	$K = 1$	$K = 2$	$K = 3$
FARGO	0,272	0,436	**0,569**
AVG	0,268	0,42	0,58
LM	**0,248**	**0,412**	0,588
MS	0,265	0,435	0,614
DIS	0,276	0,428	0,584
FAIR-LIN	0,265	0,428	0,63
FAIR-PROP	0,295	0,463	0,642
EXP	0,279	0,445	0,607
ENVY-FREE	0,267	0,42	0,58

Table 6.2. Output equation 4.8 Auditel data-set

	$K = 1$	$K = 2$	$K = 3$
FARGO	0,272	0,218	0,198
AVG	0,268	0,21	**0,193**
LM	**0,248**	**0,206**	0,196
MS	0,265	0,217	0,204
DIS	0,276	0,214	0,194
FAIR-LIN	0,265	0,214	0,21
FAIR-PROP	0,295	0,231	0,214
EXP	0,279	0,222	0,202
ENVY-FREE	0,267	0,21	**0,193**

Table 6.3. Output equation 4.9 Auditel data-set

Raw results for Music data-set

	$K = 1$	$K = 2$	$K = 3$
FARGO	0,531	0,752	0,974
AVG	0,451	0,649	0,85
LM	0,43	0,639	0,871
MS	0,502	0,701	0,909
DIS	0,446	0,643	0,856
FAIR-LIN	**0,377**	**0,587**	**0,786**
FAIR-PROP	0,479	0,691	0,92
EXP	0,465	0,673	0,88
ENVY-FREE	0,451	0,649	0,85

Table 6.4. Output equation 4.7 Music data-set

	$K = 1$	$K = 2$	$K = 3$
FARGO	0,825	1,492	2,01
AVG	0,7	1,131	1,515
LM	0,675	1,092	1,508
MS	0,766	1,25	1,688
DIS	0,692	**1,123**	1,502
FAIR-LIN	**0,661**	1,124	**1,461**
FAIR-PROP	0,751	1,247	1,68
EXP	0,722	1,161	1,511
ENVY-FREE	0,7	1,131	1,515

Table 6.5. Output equation 4.8 Music data-set

	$K = 1$	$K = 2$	$K = 3$
FARGO	0,825	0,746	0,673
AVG	0,7	0,565	0,505
LM	0,675	**0,546**	0,502
MS	0,766	0,625	0,562
DIS	0,692	0,561	0,5
FAIR-LIN	**0,661**	0,562	**0,487**
FAIR-PROP	0,751	0,623	0,56
EXP	0,722	0,58	0,503
ENVY-FREE	0,7	0,565	0,505

Table 6.6. Output equation 4.9 Music data-set

6.1.1 Comparison and analysis

After some experimental tests here we aggregate the results to better understand the properties of our new measures. We have compared the fairness outcomes for each new measure and for each competitor method against FARGO. We gathered the results coming from both Auditel data-set and Music case study. The plots 6.1, 6.2 and 6.3 represent the outcomes using the Auditel data-set; the plots 6.4, 6.5 and 6.6 represent the outcomes using the Music data-set.

Auditel data-set

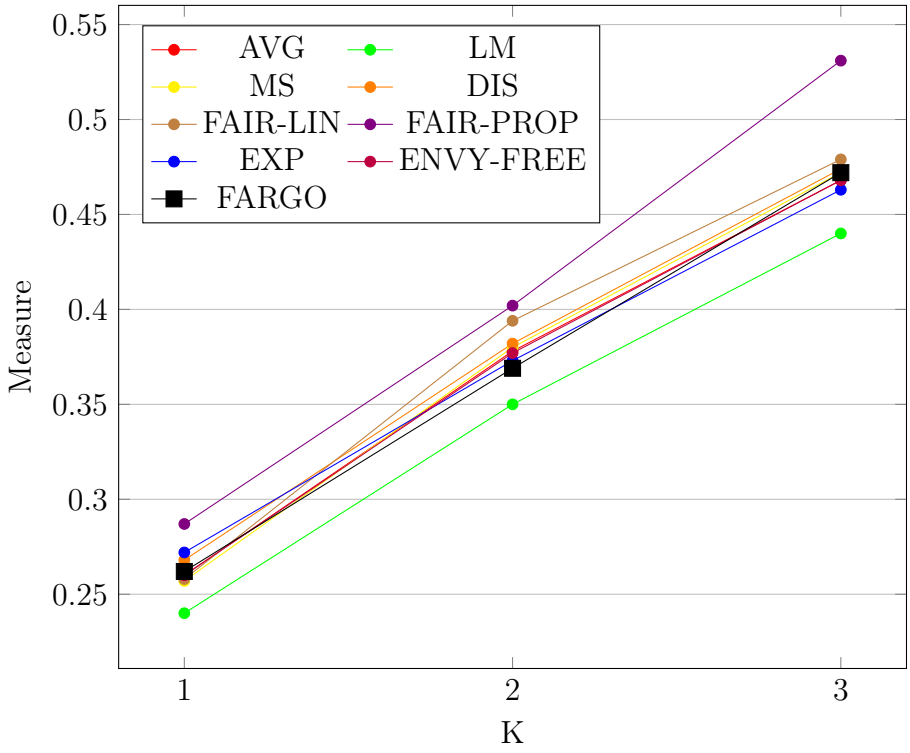


Figure 6.1. Equation 4.7 Auditel data-set

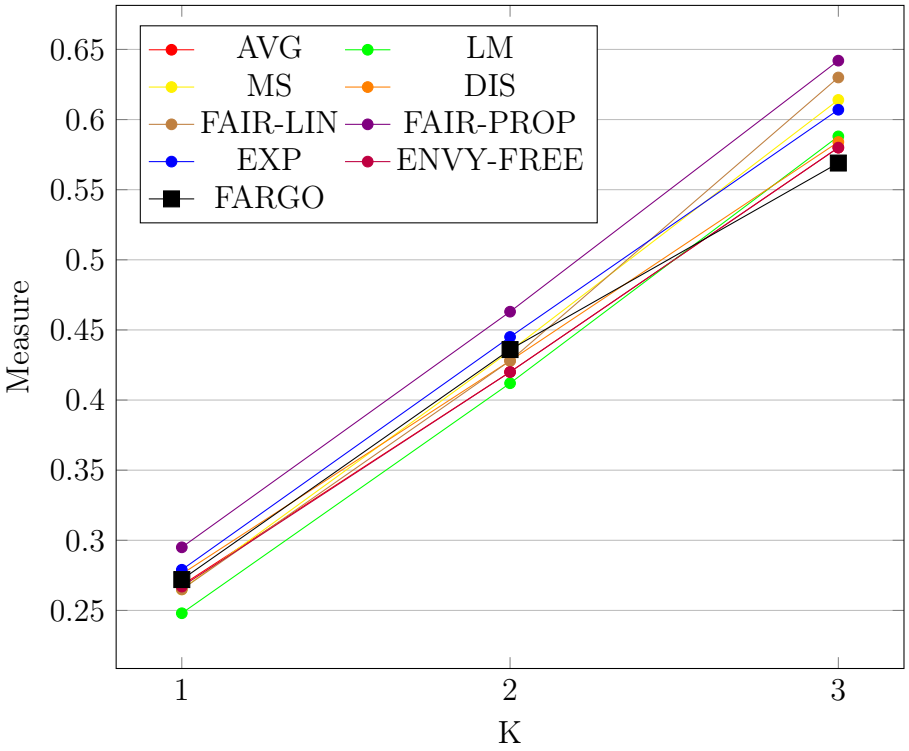


Figure 6.2. Equation 4.8 Auditel data-set

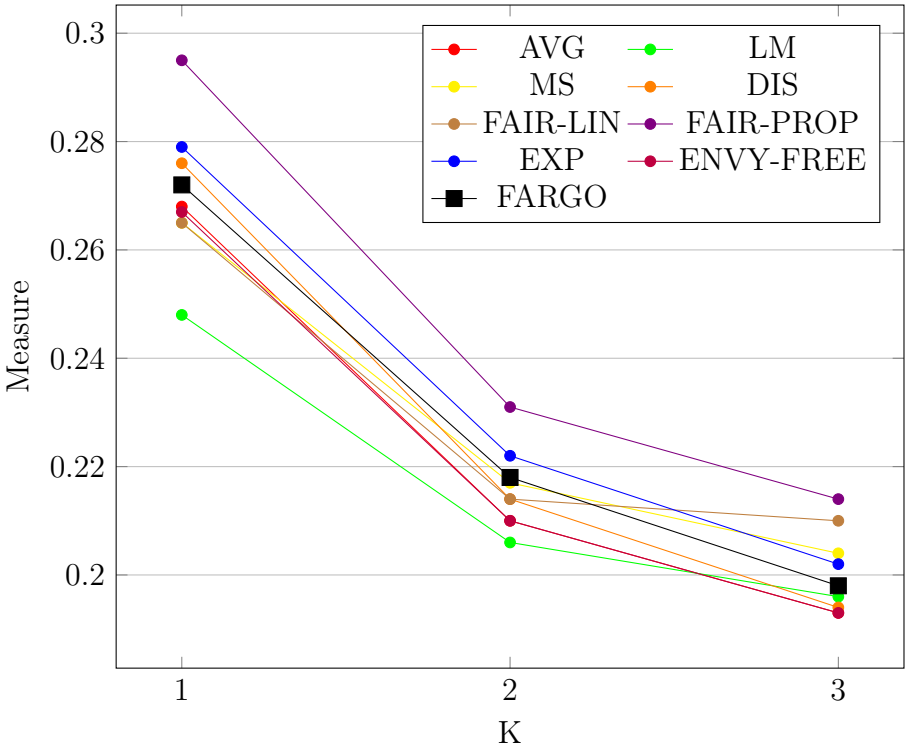


Figure 6.3. Equation 4.9 Auditel data-set

Music data-set

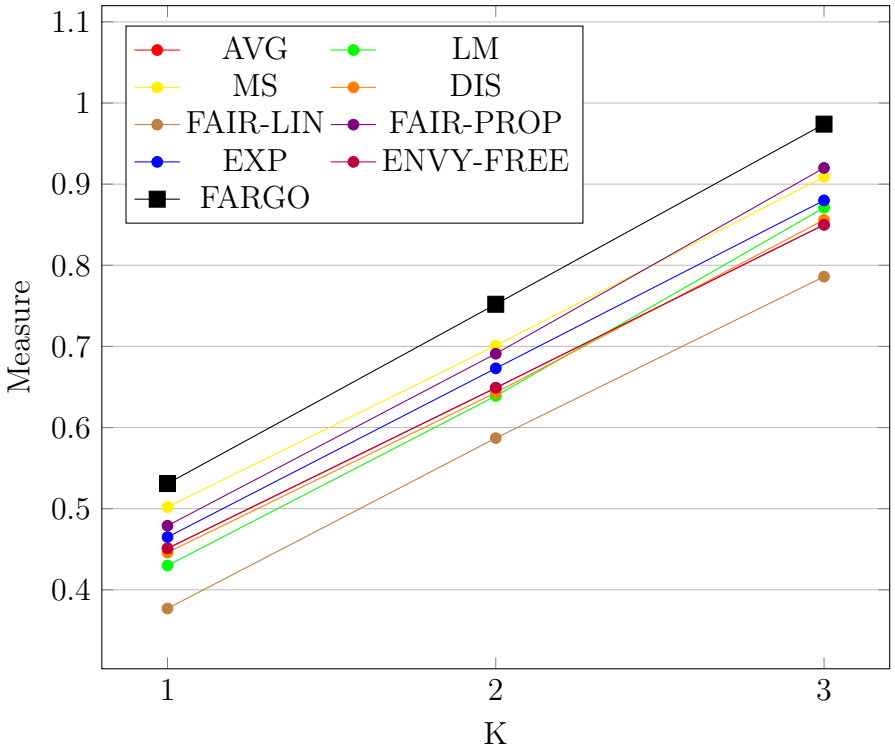


Figure 6.4. Equation 4.7 Music data-set

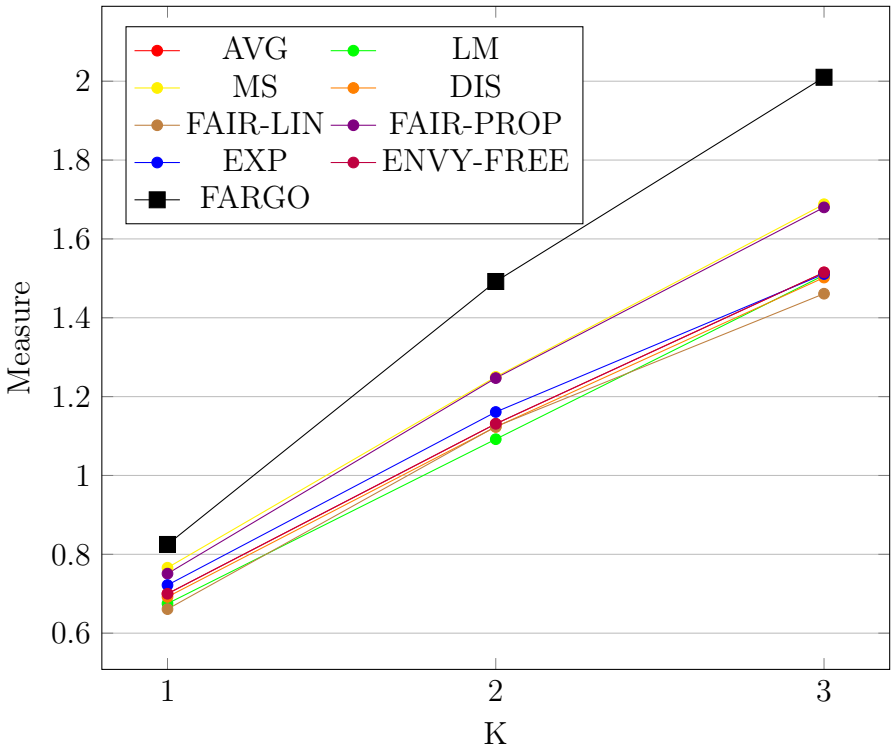


Figure 6.5. Equation 4.8 Music data-set

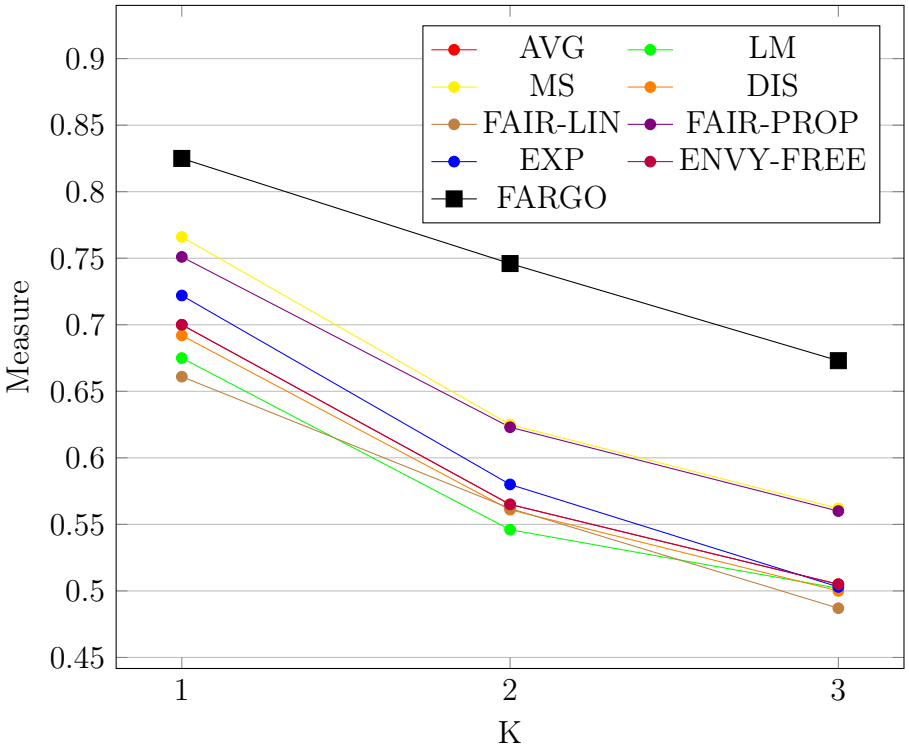


Figure 6.6. Equation 4.9 Music data-set

6.1.2 Considerations

For what concerns the Auditel data-set we can observe that LM is the best algorithm for Eq. 4.7 for any K. It is also the best algorithm for Eq. 4.8 and Eq. 4.9 except for $K = 3$. With $K = 3$ using Eq. 4.8 FARGO is the best and using Eq. 4.9 ENVY-FREE behaves better. Overall, FARGO performs very similarly to the other methods and becomes better with higher values of K. For what concerns Music data-set, generally FARGO performs slightly worse than the other methods. The best methods for this data-set are LM and FAIR-LIN.

6.2 Fairness measures comparison

In this section we try to highlight which are the differences and the similarities among the previously existing fairness measures and our proposed measures. We ran FARGO for all the different values of K and we collected the values of all the fairness indicators. The previous measures are not bounded between two values, therefore it isn't easy to compare them with the new ones. We plotted the fairness measures to have at a glance an overview of their trend, varying K.
The plot 6.7 is built using Auditel data-set; the plot 6.8 is built using Music data-set. In order to better understand the similarities and the differences between original fairness metrics and our new ones, we decided to plot them on different two scales on y-axis for Auditel data-set.

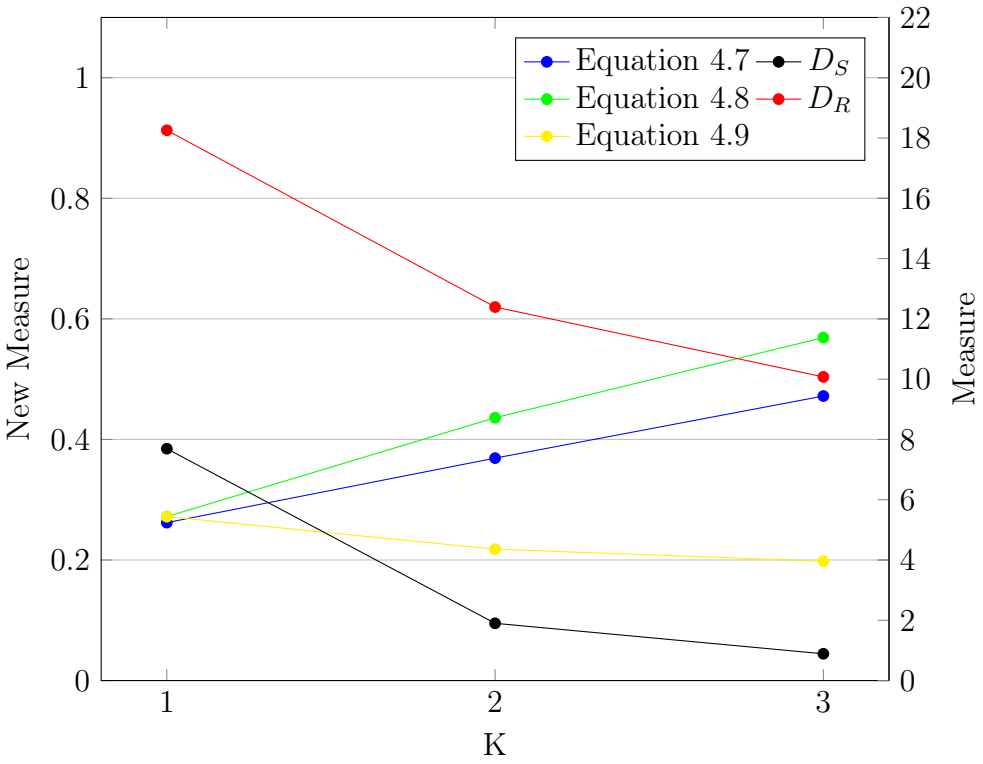


Figure 6.7. Comparison among the measures with Auditel data-set

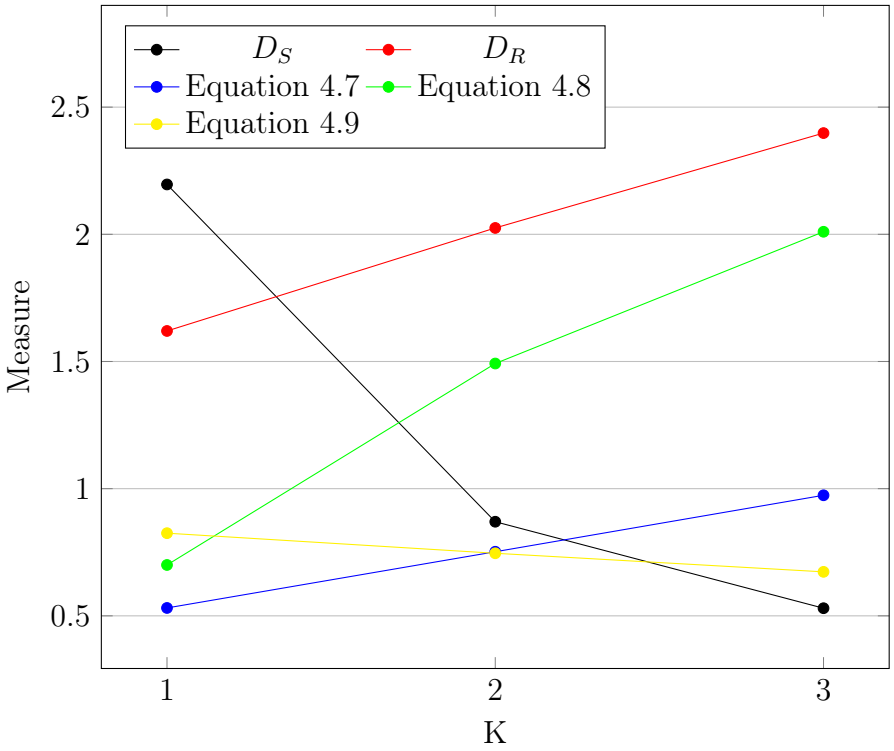


Figure 6.8. Comparison among the measures with Music data-set

Considerations

After analyzing the plots we can observe that:

- for what concerns the Score disparity we can see that the general trend is decreasing for increasing values of K. Our new measures instead, show an opposite behavior: they all increase with higher K. In fact, we want to keep in consideration the size of the recommendations (K) since, when there are many items to choose from, of course the unfairness tends to increase: this is due to the fact that increasing the size of $TopK$ it becomes difficult to find items that are appreciated by all users of the group;

- contrarily to the previous existing indicators, our measures are all bounded between some values: the Eq. 4.7 and Eq. 4.8 are bounded in the range $[0, K]$, while the Eq. 4.9 is bounded in the range $[0, 1]$. The bounding allows to make direct comparisons among different methods and to have a direct feedback on how fair the recommendations are;

- the Recommendation disparity seems to have very different behavior depending on the data-set. We can note that, depending on the data-set, it changes radically both at the trend level and above all at the numerical level: this factor makes it impossible to use Recommendation disparity to compare the behavior of FARGO with different data-sets. Moreover, this incompatibility also emerges by looking at the Score disparity, which assumes little comparable values depending on the data-set used.

Chapter 7

Conclusions

In this final part of the thesis we report all the final considerations about the experimental results. Lastly, we make a brief discussion about which could be the next steps to deepen the exploration of the topic and the possible further improvements.

7.1 Conclusions

In this thesis we analyzed some new methods to provide ethical recommendations to groups of people that decide to do some activity together for the first time. We studied the ethics/performance trade-off. These two factors are somehow in contradiction since, as we try to maximize one, we end up with degrading the other. Furthermore, we must consider that our data-sets includes logs of group activities that were not made with any particular ethics strategy, since in real life group decisions are prone to several human factors. This means that increasing standard recommender system performance, such as the Recall, does not lead to better ethical improvement, but rather they are somehow inversely correlated. We have tried to improve both recall and fairness in section 4.1, exploiting the skyline filtering, i.e.trying to exclude items that are dominated by other ones, to optimize the recall and using, at the same time, the consensus to keep fair our recommendations. Contrarily to what we expected, the experimental results in 5.3 have highlighted that recall remains on the same previous level with some slight improvements on the Recommendation disparity. Furthermore, experimental results have shown that if we optimize on the consensus the items filtered by the skyline filter, the Recall drops. We have also tried to increase both measures by estimating zeros' scores in sections 4.2, and got different results: our work improves a little bit Recall and Recommendation disparity, but on the other hand it doesn't improve Score disparity. So, even with these changes we have not fully achieved the desired results.
Analyzing these numbers, we therefore asked ourselves if the metrics to measure fairness used so far were the right ones: this is why we introduced three new measures in order to assess the fairness of group recommendations. Our new measures include mainly the novel idea of influence to evaluate how many times an user influenced the decision in previous groups. We took into consideration that influence indicates which users are the most discriminated and we came out with measures that are higher if the ethics of the recommendations is low. In fact, one of the core ideas of our new fairness measures was that if a user is discriminated (information obtained from the influence) and in the current recommendation we make the same user still unhappy, our fairness measures have to take this factor into account and therefore the fairness has to decrease. The introduction of the influence in fairness evaluation brings several advantages since it is a simple but pragmatic indicator to assess fairness and it can be computed easily starting from the previous group logs. Another core idea of our new fairness measures is their independence from the data-sets: in fact, with our experiments, we saw that Score disparity and Recommendation disparity are strictly correlated with the data-sets used, because these measures are calculated using item's scores. However, the scores present some issues, such as being subject to many biases, being calculated/derived in different ways depending on the data-set, assuming different minimum or maximum values, and so on.
What we have learned is that the ethics/performance trade-off is an element that could never be cancelled from group recommender systems and it will always force the developer to make a choice: the developer has to decide how much he/she wants to be ethical and how much he/she is willing to lose Recall. This choice will never have a universal numerical compromise.
Another aspect that emerged from our work is related to fairness metrics: we high-

lighted that some of the previous existing measures were inadequate for different reasons that we reported above in section 4.3, and then we proposed some valid alternatives trying to overcome these problems. We do not mean to imply that the existing measures are wrong or that ours are better than those that already exist, but we want to underline that the fairness measures used to establish how much ethical a group recommender system is, must also be reasoned and contextualized, taking into considerations several elements, for example the data-set used. When dealing with ephemeral groups recommendations, our new fairness measures are an optimal choice, since we introduced the influence as a parameter that allows to have an objective indicator of how much the users in a group are satisfied.
In conclusion, the answer to our question *"Which fairness measures for group recommendations?"* is basically that a perfect fairness measure may never exist, and there are many aspects to take into consideration when someone approaches this topic. Of course, the topic of group recommender systems is very difficult to face, because we must never forget that users are still human beings, with feelings, sensations, impressions, which certainly influence the choices that they make and they are not robots/automata that always behave in the same way.

7.2 Future works

This section illustrates and discusses some of our ideas that could lead to an improvement in the fairness of FARGO. These are proposals that take into consideration different elements of the algorithm itself: in fact, they range from influence to scores, passing through context.

7.2.1 Influence update

The use of influence introduces a new issue: in fact, after a group has chosen an item to interact with, the influence (in that context) of the user in the group changes. This is obvious and it's related to the way we compute the contextual influence for each user. For example if user Alice took part to three group decisions before and in none of them her preferences were approved by the group, her influence in that context is zero. Then, imagine that Alice joins another group (her fourth) and her preferred item is chosen by the group. Now Alice contextual influence changes, it becomes $\frac{1}{4} = 0.25$. Hence we have to deal with these periodical influence updates that involve users who took part to new groups. What could be the strategy? We could update the user influence after every participation to a new group decision or maybe we could perform the influence update after a certain number of participation.

7.2.2 Context in consensus

The concept of context in FARGO is certainly a fundamental element of the algorithm itself, but it presents some limitations in our opinion: if it is true that on the one hand users' scores of the various items depend directly on the context (i.e. the users presented different scores for the same item depending on the context), it is also true that on the other hand it never intervenes in the estimation of consensus, or at least not enough. In fact, the consensus is derived from the scores, which as mentioned above depend on the context, but in estimating the consensus there is no difference between the various contexts. This factor in everyday life is not negligible, as it could happen that our contentment or our willingness to accept content proposed by other users are not always the same: a trivial example may be the weekend in which perhaps we want to let off steam or try new things. Therefore, the challenge we propose here is to estimate and distinguish "numerically" the various contexts in which the groups find themselves and to consider this numerical estimation into the calculation of the consensus, for example as a multiplying factor of the current consensus.

7.2.3 The problem of zeros

In this thesis project the outputs that have been presented so far regarding all the metrics, i.e. recall and the various fairness metrics, both old and new, have been calculated by using the algorithm mainly with the Auditel data-set. As explained better in the 5.1.1 subsection, the scores are estimated from the viewing time that individual users of the various channels and these scores can assume values between 0 and 1. This derivation step presents a fairly important issue: if a user has score for a channel equal to 0 it is therefore certainly because he has never seen the specific

channel. But who tells us that the fact that the user has never seen the channel means that its relative score must be zero, which is the worst of all? It would be like saying that users are prejudiced about contents they have never seen. And what if these are channels that they might like, for example because they are similar to channels that have seen a lot instead? Hence, the proposal is to try to replace these scores equal to 0 with other numerical values, following the wake of our work on consensus zeros in the 4.2 section, through similarities among users or popularity of channels.

Acronyms

FARGO	Fair, context-AwaRe Group RecOmmender
LM	Least Misery
AVG	Average
DIS	Disagreement
ENVY-FREE	Envy-Freeness
FAIR-PROP	Fairness proportionality
FAIR-LIN	Fairness
MS	Maximum Satisfaction
EXP	Expertise

Bibliography

[1] B. Schilit and M. Theimer, "Theimer, m.m.: Disseminating active map information to mobile hosts. ieee netw. 8(5), 22-32," *Network, IEEE*, vol. 8, pp. 22–32, Oct. 1994. DOI: 10.1109/65.313011.

[2] M. O'Connor, D. Cosley, J. A. Konstan, and J. Riedl, "Polylens: A recommender system for groups of users," in *ECSCW 2001: Proceedings of the Seventh European Conference on Computer Supported Cooperative Work 16–20 September 2001, Bonn, Germany*, W. Prinz, M. Jarke, Y. Rogers, K. Schmidt, and V. Wulf, Eds. Dordrecht: Springer Netherlands, 2001, pp. 199–218, ISBN: 978-0-306-48019-5. DOI: 10.1007/0-306-48019-0_11. [Online]. Available: https://doi.org/10.1007/0-306-48019-0_11.

[3] J. Masthoff, "Group modeling: Selecting a sequence of television items to suit a group of viewers," 2004. [Online]. Available: https://link.springer.com/article/10.1023/B:USER.0000010138.79319.fd.

[4] I. Bartolini, P. Ciaccia, and M. Patella, "Salsa: Computing the skyline without scanning the whole sky," 2006. [Online]. Available: https://www.researchgate.net/publication/221613248_SaLSa_Computing_the_skyline_without_scanning_the_whole_sky.

[5] S. Amer-Yahia, S. Basu Roy, A. Chawla, G. Das, and C. Yu, "Group recommendation: Semantics and efficiency.," *Proceedings of the VLDB Endowment*, vol. 2, pp. 754–765, Aug. 2009. DOI: 10.14778/1687627.1687713.

[6] C. Bolchini, C. A. Curino, G. Orsi, E. Quintarelli, R. Rossato, F. A. Schreiber, and L. Tanca, "And what can context do for data?" *Commun. ACM*, vol. 52, no. 11, pp. 136–140, Nov. 2009, ISSN: 0001-0782. DOI: 10.1145/1592761.1592793. [Online]. Available: https://doi.org/10.1145/1592761.1592793.

[7] J. A. Recio-Garcia, G. Jimenez-Diaz, A. A. Sanchez-Ruiz, and B. Diaz-Agudo, "Personality aware recommendations to groups," in *Proceedings of the Third ACM Conference on Recommender Systems*, ser. RecSys '09, New York, New York, USA: Association for Computing Machinery, 2009, pp. 325–328, ISBN: 9781605584355. DOI: 10.1145/1639714.1639779. [Online]. Available: https://doi.org/10.1145/1639714.1639779.

[8] C. Shin and W. Woo, "Socially aware tv program recommender for multiple viewers," *IEEE Transactions on Consumer Electronics*, vol. 55, no. 2, pp. 927–932, 2009, cited By 37. DOI: 10.1109/TCE.2009.5174476. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-

68949172528&doi=10.1109%5C%2fTCE.2009.5174476&partnerID=40&md5=7fef48f7d1717bc2a7979d085a0d7612.

[9] L. Baltrunas, T. Makcinskas, and F. Ricci, "Group recommendations with rank aggregation andcollaborative filtering," 2010. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/1864708.1864733.

[10] S. Berkovsky and J. Freyne, "Group-based recipe recommendations: Analysis of data aggregation strategies," in *Proceedings of the Fourth ACM Conference on Recommender Systems*, ser. RecSys '10, Barcelona, Spain: Association for Computing Machinery, 2010, pp. 111–118, ISBN: 9781605589060. DOI: 10.1145/1864708.1864732. [Online]. Available: https://doi.org/10.1145/1864708.1864732.

[11] M. Gartrell, X. Xing, Q. Lv, A. Beach, R. Han, S. Mishra, and K. Seada, "Enhancing group recommendation by incorporating social relationship interactions," in *Proceedings of the 16th ACM International Conference on Supporting Group Work*, ser. GROUP '10, Sanibel Island, Florida, USA: Association for Computing Machinery, 2010, pp. 97–106, ISBN: 9781450303873. DOI: 10.1145/1880071.1880087. [Online]. Available: https://doi.org/10.1145/1880071.1880087.

[12] L. Naamani Dery, M. Kalech, L. Rokach, and B. Shapira, "Iterative voting under uncertainty for group recommender systems," in *Proceedings of the Fourth ACM Conference on Recommender Systems*, ser. RecSys '10, Barcelona, Spain: Association for Computing Machinery, 2010, pp. 265–268, ISBN: 9781605589060. DOI: 10.1145/1864708.1864763. [Online]. Available: https://doi.org/10.1145/1864708.1864763.

[13] C. Senot, D. Kostadinov, M. Bouzid, J. Picault, A. Aghasaryan, and C. Bernier, "Analysis of strategies for building group profiles," in *User Modeling, Adaptation, and Personalization*, P. De Bra, A. Kobsa, and D. Chin, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 40–51, ISBN: 978-3-642-13470-8.

[14] G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Boston, MA: Springer US, 2011, pp. 217–253, ISBN: 978-0-387-85820-3. DOI: 10.1007/978-0-387-85820-3_7. [Online]. Available: https://doi.org/10.1007/978-0-387-85820-3_7.

[15] F. Guzzi, F. Ricci, and R. Burke, "Interactive multi-party critiquing for group recommendation," in *Proceedings of the Fifth ACM Conference on Recommender Systems*, ser. RecSys '11, Chicago, Illinois, USA: Association for Computing Machinery, 2011, pp. 265–268, ISBN: 9781450306836. DOI: 10.1145/2043932.2043980. [Online]. Available: https://doi.org/10.1145/2043932.2043980.

[16] J. Masthoff, "Group recommender systems: Combining individual models," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Boston, MA: Springer US, 2011, pp. 677–702, ISBN: 978-0-387-85820-3. DOI: 10.1007/978-0-387-85820-3_21. [Online]. Available: https://doi.org/10.1007/978-0-387-85820-3_21.

[17] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, *Recommender Systems Handbook*. 2011. [Online]. Available: https://www.springer.com/gp/book/9780387858203.

[18] S. Seko, T. Yagi, M. Motegi, and S. Muto, "Group recommendation using feature space representing behavioral tendency and power balance among members," in *Proceedings of the Fifth ACM Conference on Recommender Systems*, ser. RecSys '11, Chicago, Illinois, USA: Association for Computing Machinery, 2011, pp. 101–108, ISBN: 9781450306836. DOI: 10.1145/2043932.2043953. [Online]. Available: https://doi.org/10.1145/2043932.2043953.

[19] X. Liu, Y. Tian, M. Ye, and W.-C. Lee, "Exploring personal impact for group recommendation," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, ser. CIKM '12, Maui, Hawaii, USA: Association for Computing Machinery, 2012, pp. 674–683, ISBN: 9781450311564. DOI: 10.1145/2396761.2396848. [Online]. Available: https://doi.org/10.1145/2396761.2396848.

[20] E. Ntoutsi, K. Stefanidis, K. Nørvåg, and H.-P. Kriegel, "Fast group recommendations by applying user clustering," in *Conceptual Modeling*, P. Atzeni, D. Cheung, and S. Ram, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 126–140, ISBN: 978-3-642-34002-4.

[21] K. Verbert, N. Manouselis, X. Ochoa, M. Wolpers, H. Drachsler, I. Bosnic, and E. Duval, "Context-aware recommender systems for learning: A survey and future challenges," *IEEE Transactions on Learning Technologies*, vol. 5, no. 4, pp. 318–335, 2012. DOI: 10.1109/TLT.2012.11.

[22] L. A. M. Carvalho and H. T. Macedo, "Generation of coalition structures to provide proper groups' formation in group recommender systems," in *Proceedings of the 22nd International Conference on World Wide Web*, ser. WWW '13 Companion, Rio de Janeiro, Brazil: Association for Computing Machinery, 2013, pp. 945–950, ISBN: 9781450320382. DOI: 10.1145/2487788.2488089. [Online]. Available: https://doi.org/10.1145/2487788.2488089.

[23] L. A. M. C. Carvalho and H. T. Macedo, "Users' satisfaction in recommendation systems for groups: An approach based on noncooperative games," in *Proceedings of the 22nd International Conference on World Wide Web*, ser. WWW '13 Companion, Rio de Janeiro, Brazil: Association for Computing Machinery, 2013, pp. 951–958, ISBN: 9781450320382. DOI: 10.1145/2487788.2488090. [Online]. Available: https://doi.org/10.1145/2487788.2488090.

[24] J. Gorla, N. Lathia, S. Robertson, and J. Wang, "Probabilistic group recommendation via information matching," in *Proceedings of the 22nd International Conference on World Wide Web*, ser. WWW '13, Rio de Janeiro, Brazil: Association for Computing Machinery, 2013, pp. 495–504, ISBN: 9781450320351. DOI: 10.1145/2488388.2488432. [Online]. Available: https://doi.org/10.1145/2488388.2488432.

[25] O. Van Deventer, J. De Wit, J. Vanattenhoven, and M. Gualbahar, "Group recommendation in a hybrid broadcast broadband television context," eng, vol. 997, Springer, 2013, pp. 12–18. [Online]. Available: $$Uhttps://lirias.kuleuven.be/retrieve/229855$$Dgrouprs2013_paper_3.pdf%20[freely%20available].

[26] A. J. Chaney, M. Gartrell, J. M. Hofman, J. Guiver, N. Koenigstein, P. Kohli, and U. Paquet, "A large-scale exploration of group viewing patterns," in *Proceedings of the ACM International Conference on Interactive Experiences for TV and Online Video*, ser. TVX '14, Newcastle Upon Tyne, United Kingdom: Association for Computing Machinery, 2014, pp. 31–38, ISBN: 9781450328388. DOI: 10.1145/2602299.2602309. [Online]. Available: https://doi.org/10.1145/2602299.2602309.

[27] T. De Pessemier, S. Dooms, and L. Martens, "Comparison of group recommendation algorithms," *Multimedia Tools and Applications*, vol. 72, Oct. 2014. DOI: 10.1007/s11042-013-1563-0.

[28] N. Kim and J. Lee, "Group recommendation system: Focusing on home group user in tv domain," in *2014 Joint 7th International Conference on Soft Computing and Intelligent Systems (SCIS) and 15th International Symposium on Advanced Intelligent Systems (ISIS)*, 2014, pp. 985–988. DOI: 10.1109/SCIS-ISIS.2014.7044866.

[29] M. Kompan and M. Bielikova, "Voting based group recommendation: How users vote," vol. 1210, Sep. 2014.

[30] I. Ali and S.-W. Kim, "Group recommendations: Approaches and evaluation," in *Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication*, ser. IMCOM '15, Bali, Indonesia: Association for Computing Machinery, 2015, ISBN: 9781450333771. DOI: 10.1145/2701126.2701208. [Online]. Available: https://doi.org/10.1145/2701126.2701208.

[31] S. Ghazarian and M. A. Nematbakhsh, "Enhancing memory-based collaborative filtering for group recommender systems," *Expert Systems with Applications*, vol. 42, no. 7, pp. 3801–3812, 2015, ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2014.11.042. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417414007386.

[32] E. Quintarelli, E. Rabosio, and L. Tanca, "Recommending new items to ephemeral groups using contextual user influence," 2016. [Online]. Available: https://dl.acm.org/doi/10.1145/2959100.2959137.

[33] H. Mohana and D. Suriakala, "A study on ontology based collaborative filtering recommendation algorithms in e-commerce applications," *IOSR Journal of Computer Engineering*, vol. 19, pp. 2278–661, Jul. 2017. DOI: 10.9790/0661-1904011419.

[34] D. Serbos, S. Qi, N. Mamoulis, E. Pitoura, and P. Tsaparas, "Fairness in package-to-group recommendations," in *Proceedings of the 26th International Conference on World Wide Web*, ser. WWW '17, Perth, Australia: International World Wide Web Conferences Steering Committee, 2017, pp. 371–379, ISBN:

9781450349130. DOI: 10.1145/3038912.3052612. [Online]. Available: https://doi.org/10.1145/3038912.3052612.

[35] L. Xiao, Z. Min, Z. Yongfeng, G. Zhaoquan, L. Yiqun, and M. Shaoping, "Fairness-aware group recommendation with pareto-efficiency," 2017. [Online]. Available: https://dl.acm.org/doi/10.1145/3109859.3109887.

[36] S. Yao and B. Huang, "Beyond parity: Fairness objectives for collaborative filtering," May 2017.

[37] A. Felfernig, L. Boratto, M. Stettinger, and M. Tkalčič, "Evaluating group recommender systems," in *Group Recommender Systems : An Introduction*. Cham: Springer International Publishing, 2018, pp. 59–71, ISBN: 978-3-319-75067-5. DOI: 10.1007/978-3-319-75067-5_3. [Online]. Available: https://doi.org/10.1007/978-3-319-75067-5_3.

[38] J. Leonhardt, A. Anand, and M. Khosla, "User fairness in recommender systems," 2018. [Online]. Available: https://www.researchgate.net/publication/324640535_User_Fairness_in_Recommender_Systems.

[39] H. Steck, "Calibrated recommendations," in *Proceedings of the 12th ACM Conference on Recommender Systems*, ser. RecSys '18, Vancouver, British Columbia, Canada: Association for Computing Machinery, 2018, pp. 154–162, ISBN: 9781450359016. DOI: 10.1145/3240323.3240372. [Online]. Available: https://doi.org/10.1145/3240323.3240372.

[40] S. Dara, C. R. Chowdary, and C. Kumar, "A survey on group recommender systems," 2019. [Online]. Available: https://link.springer.com/article/10.1007/s10844-018-0542-3.

[41] E. Quintarelli, E. Rabosio, and L. Tanca, "Efficiently using contextual influence to recommend new items to ephemeral groups," *Information Systems*, vol. 84, pp. 197–213, 2019, ISSN: 0306-4379. DOI: https://doi.org/10.1016/j.is.2019.05.003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306437918304484.

[42] M. Kaya, D. Bridge, and N. Tintarev, "Ensuring fairness in group recommendations by rank-sensitive balancing of relevance," in *Fourteenth ACM Conference on Recommender Systems*, ser. RecSys '20, Virtual Event, Brazil: Association for Computing Machinery, 2020, pp. 101–110, ISBN: 9781450375832. DOI: 10.1145/3383313.3412232. [Online]. Available: https://doi.org/10.1145/3383313.3412232.

[43] Y. Xiao, Q. Pei, L. Yao, S. Yuc, L. Bai, and X. Wang, "An enhanced probabilistic fairness-aware group recommendation by incorporating social activeness," 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804520300539.

[44] H. Jagadish, J. Stoyanovich, and B. Howe, "The many facets of data equity." [Online]. Available: http://ceur-ws.org/Vol-2841/PIE+Q_6.pdf.

[45] Y. Zhiwen, Z. Xingshe, H. Yanbin, and G. Jianhua, "Tv program recommendation for multiple viewers based on user profile merging,"