A Group Recommendation System for Music Using Implicit User Feedback

By

Te Lan

This thesis is submitted in partial fulfilment of the requirements for the award of the degree:

MSc Data Science and AI for the Creative Industries

Creative Computing Institute, University of the Arts London

Supervisor: Dr. Rebecca Fiebrink

15 December 2022

Word Count: 7,311

Acknowledgements

I would like to deeply appreciate my thesis supervisor, Dr. Rebecca Fiebrink, for her invaluable guidance and feedback on my thesis project and for her generous patience and support in those emotionally intense moments to help me reach my goal. I am also thankful to my course leader, Dr. Louis McCallum, who generously provided knowledge and expertise at the very start of this journey. Thanks to my cohorts for all the inspiring discussions and mental support, especially my classmate and best friend, Aarav Pillai, who has consistently supported me in many ways since our acquaintance. I am also thankful to my classmates and friends, Xiaolin Deng and Simon Sheng, for their companionship throughout the process. Last but not least, I'd like to mention my mother for her support in my mental health and financial status, which allowed me to focus on my studies.

Abstract

Recommendation systems have been widely equipped in various platforms to provide users with suggestions. Similarly, a group recommendation system learns and provides suggestions to a group of people. In group events such as dinner parties and small gatherings, music is commonly used for entertainment or self-expression. In this work, the researcher developed a music recommendation system for a group of people using Matrix Factorisation(MF) based Collaborative Filtering. To train the MF model, the researcher utilised the implicit user feedback data by converting it to explicit user feedback such as rating. Three group recommendation methods based on the reproduced dataset were also tested and further evaluated for performance. Two explanation methods were also designed to inform why a song is recommended and how every group member is represented in the playlist. The research paper concludes with the researcher discussing the limitations, including selection bias in the reproduced dataset.

Table of Content

1. Introduction

Recommendation systems have become crucial nowadays as a method to filter massive information on the internet, provide suggestions based on user's preferences, and boost the decision making process. It has been applied in various platforms such as media streaming services, e-commerce business, online booking systems and social media platforms.

Similarly, a group recommendation system learns the preferences of a group of users using various strategies to align with every group member's needs as much as possible (Data et al., 2019). One good use case of a group recommender is to help plan a group event that involves a group decision-making process, which can be time-consuming and hard to conduct manually. A group recommendation system can help the process by utilising the learned preferences of individuals and generating recommendations that could potentially reach a consensus in a real group decision-making process.

In group events, such as various kinds of parties, music is commonly used for entertainment and self-expression(Rentfrow et al., 2003). Among other entertaining content such as movies, books and games, music requires a shorter time to experience, which makes it more handy and frequent for people to interact with, especially on digital platforms. Therefore, large amounts of data about the interaction between music and platform users can be easily obtained. To make use of these user data, Collaborative Filtering (CF) has become an efficient method to recommend items since it captures the similarity between users' preferences by modelling their explicit ratings on items or implicit behaviour patterns (Wang et al., 2006). When processing massive data with the CF method, Matrix Factorisation (MF) has been proven efficient on this issue as well as useful when dealing with the sparseness of the data (Koren et al., 2009). However, the MF method heavily relies on explicit user data, which is usually unavailable in actual applications (Hu et al., 2008). While there could be a structural modification on MF based CF method, an easier way to fill the gap between explicit data and implicit data is through transformation, such as estimating explicit data, e.g. ratings for music from implicit data, e.g. play counts of songs.

The frequent interaction between users and recommendation systems, such as sending queries and accepting the suggestions, is likely to let users shape a subjective understanding of how it works (Liao and Varshney., 2022). It's widely acknowledged that explanations of the recommendation results help to build a positive understanding and trust between the users and the systems (Preece et al., 2018). Thus, explainability comes as one of the essential attributes for recommendation systems apart from the quality of suggestions.

Explanations of the suggestions, in this music recommendation case, could tell the reason behind a recommendation as well as how a song is related to every group member that could potentially reach everyone's consensus. Moreover, the explanations can reflect how every member is represented in the playlist and how individuals are affecting the results. Those explanations as evidential materials are able to extend the resonating experience from a digital reflection.

In this paper, the researcher proposes a music recommendation system for a group of users and the explanation methods to present the results. The recommendation system was trained by utilising implicit user feedback, including user music play counts. The researcher used the Echo Nest Taste Profile dataset, trimmed it, and re-indexed it to fit it into the algorithm. The researcher also used the user play count data to estimate user ratings on songs to enable the matrix factorisation process. Using a post-hoc approach, the generated explanations aimed to imply the relationships between group members, the group, and recommended music. The relationship between individuals and recommended music explains the relevance between every member's taste and the songs, as well as how every member is represented in the playlist. Moreover, the relationship between individuals and the group tells the commonalities in every group member.

In this paper, the researcher will compare different music recommender methods and address the following challenges:

1. Implementing implicit user feedback data on a group recommendation system designed for explicit user feedback
2. Testing different group recommendation methods on a new dataset
3. Designing two explanation methods to help present the recommendation results

2. Literature Review

In this section, we analyse the related techniques of music recommendation systems for a group of people as well as the research areas to which they belong. Various music recommendation methods were discussed as well as different strategies to recommend

items to a group. Besides, we also looked into techniques to implement implicit user feedback on algorithms designed for explicit feedback, plus explanation methods to present the results.

2.1 Music Recommendation System

People frequently engage in listening to music because they believe it to be an essential part of their lives. Unlike books or movies, the length of a piece of music is much shorter, and most people listen to their favourite songs more than once. Previous studies have shown that participants engaged in music listening more frequently than any other activity, including watching television, reading books, and watching movies (Rentfrow et al., 2003). Because it is such a powerful way for both communication and self-expression, music has attracted a great deal of research. Methods from the field of Music Information Retrieval (MIR) have been developed to help with the organisation and management of the millions of music titles created by society, including genre classification(Lines et al., 2011), artist identification(Mandel, 2005), and instrument recognition(Marques and Moreno, 1999). (Pachet and Aucouturier, 2004).

The purpose of a music recommender is to aid users in filtering and discovering songs that suit their preferences (Song et al., 2012). A good music recommender system should be able to recognise preferences automatically and create playlists based on those preferences. Currently, the Collaborative Filtering (CF) algorithm has been found to work well based on users' listening behaviour and historical ratings (Burke, 2002). In conjunction with the usage of a content-based model, the user can obtain a list of songs that are similar based on high or low-level acoustic characteristics, such as genre, instrument, rhythm, and pitch (Bogdanov and Herrera, 2011).

As one of the most effective approaches in recommendation systems, CF makes the assumption that if users X and Y act or rate similarly on n items, they will act or rate similarly on other items (Resnick et al., 1997). Memory-based, model-based, and hybrid collaborative filtering are the three further divisions of collaborative filtering (Su and Khoshgoftaar, 2009). The goal of memory-based CF is to predict the item using the whole database of prior ratings. Every user is placed in a group with others who share their interests, and a new item is created by identifying the person's closest neighbour using a large number of explicit user votes (Burke, 2002). Model-based CF, as opposed to memory-based CF, makes use of machine learning and data mining technologies to build and train the users' preferences model. It creates a unique prediction model (Adomavicius and Tuzhilin, 2005) and uses a collection of rating scores to represent user preference. The system gives predictions for tests and real-world data based on the known model. A hybrid CF model combines multiple CF models to produce predictions. The hybrid CF model has been shown to outperform all other methods (Wang et al., 2006).

2.2 MF based CF Using Implicit User Feedback

Huge amounts of data have been produced over the past few decades as a result of the Internet's explosive growth, which presents a challenge for CF algorithms. CF implementations and algorithms for recommendation system applications face a number of difficulties. It starts with the size of the processed datasets. The second one is caused by the sparsity of the rating matrix, which means that for each user, only a small number of items they interacted with are rated. Matrix Factorisation (MF) effectively handles these difficulties (Koren, Bell and Volinsky, 2009). The latent factor model serves as the foundation for most MF models. A rating matrix is modelled as the result of a user factor matrix and an item factor matrix in a latent factor mode (Koren, Bell, and Volinsky, 2009). The high correlation between user factors and item factors leads to a recommendation. By fitting the previously recorded ratings, the system improves the model. The goal is to find a way to generalise these past ratings in a way that can predict ratings that have yet to be given.

Among the methods tested, Matrix Factorisation was discovered to be the most effective at addressing the issue of excessive sparsity in the recommendation system database. Certain studies have employed dimensionality reduction techniques to aid with the problem. The issue with sparsity is that many values in the rating matrix are null since not all users rate all items. The fact that users have different numbers of items they have rated in common makes it challenging to calculate the distances between them. Singular Value Decomposition (SVD), which decreases the dimensionality of the rating matrix and discovers latent components in the data, is an alternative to including global means for null values or significance weighting (Vozalis and Margaritis, 2007).

There are various evaluation metrics used to evaluate the prediction accuracy and effective implementation of the MF models with CF algorithms in the recommendation system. One common metric is to calculate the precision and recall of recommended items (Bokde et al., 2015). Precision is defined as the

ratio of relevant items to recommended items. Precision can be calculated using the formula:

$$Precision = \frac{|Interesting\ Items \cap Recommended\ Items|}{|Recommended\ items|}$$

Recall is defined as the proportion of relevant items that have been recommended to the total number of relevant items. Recall is calculated by the formula:

$$Recall = \frac{|Interesting\ Items \cap Recommended\ Items|}{|Interesting\ Items|}$$

It is desirable for an algorithm to have high precision and recall values. However, both of the metrics are inversely related, such that when precision is increased, recall usually diminishes and vice versa (Dara et al., 2018).

By automatically inferring feature vectors from data, latent factor models aim to characterise users and products (Koren et al., 2009; Hu et al., 2008). Similar to explicit profiling in content-based systems, such feature vectors characterise persons and products across numerous dimensions; however, real interpretation of these aspects is typically unimportant and often unattainable (Koren et al., 2009).

Unfortunately, the majority of the work in matrix factorisation is centred on high-quality explicit feedback datasets, in which users make their preferences known by directly rating subsets of accessible items on a fixed scale (Hu et al., 2008). However, these explicit ratings are not available in many real-world situations. Instead, it is necessary to make use of implicit feedback, such as interaction counts and browsing histories. Pacula (2009) presented a straightforward tweak to an MF-based CF method that greatly enhances its performance on implicit data. They used data from last.fm, which tracked how often users played a song by a specific artist but did not include users' actual ratings for those songs. Ratings were then generated from play count data in order to employ the matrix factorisation techniques.

2.3 Group Recommendation System

Recommender Systems are useful tools as they help users make judgments and choose from a wide variety of options. Because of the rise in social activities and mobile devices, many activities are now done in groups, increasing the need for the development of group recommender systems. There has been a lot of research done to apply group recommendation systems to different areas, such as music (Kim and El Saddik, 2015), movies (Villavicencio et al., 2016), TV shows (Kim and Lee, 2014), and books (Kim et al., 2009). The group recommendation approaches can be roughly differentiated into two, the Pseudo-user approach and the Consensus approach.

The pseudo-user approach treats a group of people as a single virtual user by aggregating each group member's preferences into a single profile and recommending items based on the group profile (Kim and Lee, 2014). Ortega et al. (2016) used MF-based CF to perform group recommendations. They suggested three approaches to compute group factors representing group-item interactions in latent factor space and compared the proposed methods in three different scenarios: small, medium, and large group sizes. These approaches are categorised based on when the users' data is merged with the group's data: (a) before factorisation (BF and WBF); or (b) after factoring (AF). The BF technique models a group of users by creating a virtual user that represents the group's item preferences. To compute the group's factors, it uses the folding-in technique on the virtual user to add it to the factorised model. The WBF approach is an extension of the BF approach. It assigns a weight to each item that the virtual user has 'rated.' These weights will be determined by the number of ratings each item has gotten from the group's users, as well as the consensus of those ratings. When we compute the group latent factors, the items with the highest weights will contribute more than those with low weights. AF is the simplest way to compute recommendations to a group of users using an MF model. It computes the group's factors by combining the factors of the group's users. They used the group movie ratings generated from the MovieLens and Netflix datasets to test each approach for different group sizes. They demonstrated that the performance of group recommender systems varied with group size and dataset size.

The consensus approach is aimed at simulating the negotiation between group members as a group decision-making process and aggregating personalised recommendations into a final recommendation list that reach a consensus among group members (Villavicencio et al., 2016; Kim and El Saddik, 2015). Villavicencio et al. (2016) created a multi-agent system that utilises negotiation techniques to reach group consensus and improve group recommendations. Each user in the system is represented by a personal agent in order to carry out a cooperative negotiating process. Chen et al. (2008) employed a genetic algorithm to simulate group member interactions in order to estimate the rating that a group of members could give to an item. Kim et al. (2010) proposed a two-phased group recommendation approach. The first part involved using CF to build a candidate recommendation set, and the second

involved removing items from the candidate set to increase the satisfaction of individual members' preferences.

In order to help integrate all the preferences of the group members, different aggregation strategies must be chosen according to the requirements of the group. Both aforementioned approaches, the Pseudo-user approach and consensus approach, incorporate aggregation in them. The most commonly used aggregation strategies are (i) Aggregated Voting and (ii) Least Misery (Dara et al., 2018). The goal of Aggregated Voting is to identify the set of items that raise the total individual satisfaction scores of all users in the group. The objective is to maximise the overall satisfaction of the group. In the Least Misery method, the task is to recommend items such that they maximise the minimum individual satisfaction score among the users in a group. Moreover, Additive utilitarian is a strategy where individual ratings are summed for each item. It is also known as an average strategy because the resulting group ranking gives the same result as taking the average of individual ratings. In Multiplicative utilitarian, individual ratings are multiplied for each item. Multiplicative aggregation model generates the group recommendations by calculating the geometric mean of user preferences (Christensen and Schiaffino, 2011b). In Most Pleasure strategy, group rating is determined by the highest rating as opposed to Least Misery.

The challenges that arise when suggesting items to groups are explored in Jameson and Smyth (2007), and the issues are categorised in relation to the four subtasks of the group recommendation problem. The subtasks are: 1) Acquiring user preference information, 2) Generating recommendations, 3) Explaining recommendations and 4) Helping group members to settle on a final recommendation.

2.4 Explainable Recommendations

In a broad sense, the goal of most explainable artificial intelligence(XAI) work is to make AI understandable to people. It is acknowledged that explainability is essential in Human-AI interaction for the purpose of enhancing people's understanding of the system and building trust. Beyond the demand by data scientists or researchers within the AI field, the need for explainability creates an interdisciplinary environment that connects Human-computer interaction(HCI) research and user experience(UX) design. The human-centred approaches required to develop explainability mean it could be as much of a design challenge as an algorithmic challenge. While there is no one-fits-all solution for all systems, it depends on the users' needs that drive the choice of the explanation method. (Q. Vera et al., 2021) In Preece et al.(2018), one commonly recognised persona in the user community is decision-makers who utilise AI applications as a trusted source to make better decisions. Thus, explanations of AI predictions are key to improving users' confidence in accepting the information.

More specifically for recommendation systems, Zhang et al.(2020) concluded that personalised recommendation research can be sorted into 5W problems – when, where, who, what and why. Explainability refers to addressing the problem of Why; in other words, the system provides suggestions as well as explanations to clarify the reason behind it. The model that generates explanation can be classified into two types, model-intrinsic and model-agnostic. The model-intrinsic approach creates interpretable models with transparent decision mechanisms, which naturally provide explanations for the suggestion (Zhang et al., 2014). The model-agnostic approach (Wang et al., 2018), also known as the post-hoc explanation approach (Peake and Wang, 2018), creates an explanation model instead of generating explanations after an item is recommended.

2.5 The Research Gap

Plenty of research has been done on music recommendation and group recommendation. The Matrix Factorisation-based Collaborative Filtering approach has been popular in these fields since it efficiently handles large datasets and the sparseness of datasets. Among the group recommender research that utilises MF-based CF, few works used implicit user feedback, as the algorithm is mainly designed for explicit user feedback. However, high-quality explicit user feedback is not always available in actual applications. Considering practical limits in real-world applications, it is worth exploring how implicit feedback could fit into explicit feedback-oriented algorithms. Apart from the recommendation results, explanations should also be generated for users to better understand how the recommender system works and why a specific item has been recommended.

3. Methodology

In this paper, the researcher suggests a system for recommending music to a group of users and ways to explain the results. The recommendation system was trained using implicit feedback from users, such as the number of times users played music. The researcher used the Echo Nest Taste Profile dataset and trimmed and re-indexed it so that it would work with the algorithm. The researcher also used the number of

times a user played a song to estimate how the user rated the song. The modified dataset helped with the matrix factorisation process. Using a "post-hoc" method, the explanations were meant to show the connections between group members, the group, and the suggested music. The connection between people and recommended music explains how each member's tastes match the songs and how each member is represented in the playlist. Also, the relationship between each person and the group shows what each person has in common and what each person in the group has in common.

3.1 Matrix Factorisation model

Matrix Factorisation models characterise both users and items by mapping them to a shared latent factor space that represents user-item interactions. A recommendation is made when there is a high correspondence between item and user factors (Koren et al., 2009). Fig. 1 shows a graphical representation of the factorisation process. The model used in this paper factorises the rating matrix into the following elements:

- $\overrightarrow{q}_i = (q_{i,1} \cdots q_{i,K})$ represents the factor vector of the item i.
- b_i represents the bias of the item i independent of any interaction.
- $\overrightarrow{p}_u = (p_{u,1} \cdots q_{u,K})$ represents the factor vector of the user u.
- b_u represents the bias of the user u independent of any interaction.

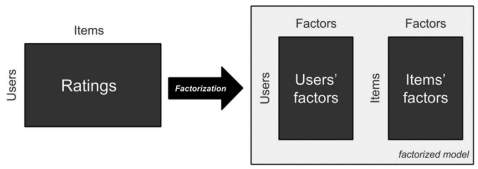

Fig. 1. Graphical example of Matrix Factorisation process. (Ortega et al., 2016, p.315)

To learn the factor vectors ($\overrightarrow{q}_i$ and $\overrightarrow{p}_u$)and the biases (b_i and b_u) the system minimises the following expression for a set of known ratings:

$$\min_{\overrightarrow{p}_u, \overrightarrow{q}_i, b_u, b_i} \sum_{r_{u,i} \neq \bullet} (r_{u,i} - \mu - b_u - b_i - \overrightarrow{p}_u{}^T \overrightarrow{q}_i)^2 + \lambda(||\overrightarrow{p}_u||^2 + ||\overrightarrow{q}_i||^2 + b_u^2 + q_i^2)$$

$$(1)$$

Where $r_{u,i}$ is the training rating of the user u to the item i, μ is the rating average of the dataset and λ is a parameter that controls the training process. Once the MF is learnt, prediction for the user u to the item i ($m_{u,i}$) can be computed using the following expression:

$$m_{u,i} = \mu + b_i + b_u + \overrightarrow{p}_u{}^T \overrightarrow{q}_i \quad (2)$$

In order to generate group predictions we need to compute the group's factor vector ($\overrightarrow{p}_G$) and the group's bias (b_G). We will describe how to compute these values in the following sections. Once the group is factorised, prediction for the group G to the item i can be computed as follows:

$$m_{G,i} = \mu + b_i + b_G + \overrightarrow{p}_G{}^T \overrightarrow{q}_i \quad (3)$$

Using the predicted values we compute the recommendations for the group G (R_G) as the set of N items (4) not rated for any user of the group (5) and with the prediction values (6).
The following expressions must be true:

$$\#R_G \leq N \quad (4)$$

$$\forall i \in R_G, \forall u \in G : r_{u,i} = \bullet \quad (5)$$

$$\forall i \in R_G, \forall j \notin R_G : m_{G,i} \geq m_{G,j} \quad (6)$$

We have used a stochastic gradient descent optimisation (expression (1)) to inference the parameters of the MF model. These parameters has been initialised randomly with values in the range [−1,1]. We repeat the following algorithm until the parameters converge: for each known rating of the training set, the error $e_{u,i}$ between the real rating and the predicted one is computed:

$$e_{u,i} = r_{u,i} - \mu - b_u - b_i - \overrightarrow{p}_u \top \overrightarrow{q}_i \quad (7)$$

Then the parameters are modified in the opposite direction of the gradient:

$$\overrightarrow{p}_u \leftarrow \overrightarrow{p}_u + \gamma \cdot (e_{u,i} \cdot \overrightarrow{q}_i - \lambda \cdot \overrightarrow{p}_u) \quad (8)$$

$$\overrightarrow{q}_i \leftarrow \overrightarrow{q}_i + \gamma \cdot (e_{u,i} \cdot \overrightarrow{p}_u - \lambda \cdot \overrightarrow{q}_i) \quad (9)$$

$$b_u \leftarrow b_u + \gamma \cdot (e_{u,i} - \lambda \cdot b_u) \quad (10)$$

$$b_i \leftarrow b_i + \gamma \cdot (e_{u,i} - \lambda \cdot b_i) \quad (11)$$

where γ is a parameter that controls the speed of the learning process.

3.2 Dataset processing

The group recommender model is adapted from the source code in Ortega et al.(2016) which employed MF based CF. The original model was tailored to

recommend movies using MovieLens dataset (Harper and Konstan, 2015).

Our work is based on the Taste Profile dataset (Bertin-Mahieux et al., 2011), a subset of the Million Song Dataset (McFee et al., 2012). The subset can be downloaded from Million Song Dataset website. It contains authentic user play counts data recorded in the format of "user - song - play count" triplets, which can be considered as implicit user feedback in contrast to explicit feedback such as ratings. The researcher will further discuss the impacts of implicit feedback in the following section. The full subset has triplets for one million users. Table 1 shows a brief dataset description of summary statistics.

Table 1. Summary statistics of The Taste profile dataset

Dataset	Taste Profile
#Users	1,019,318
#MSD–archived–songs	384,546
#User–song–play count triplets	48,373,586

For a further understanding of the dataset, distributions of every music's play count and user - song - play count triplets are analysed. According to Figure 1, the play counts of the Taste Profile dataset showed a long-tail distribution with obvious outliers at the end.

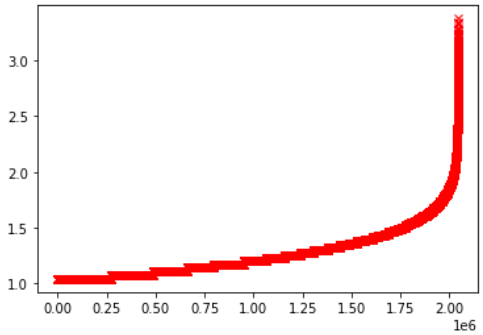

Fig 2. Play count (y-axis) distribution in Taste Profile Dataset.

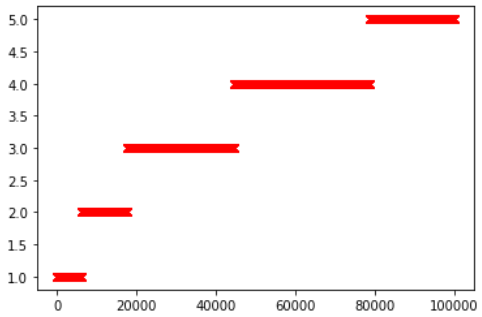

Fig 3. Rating (y-axis) distribution in MovieLens100K dataset

Considering the computational resources needed to support the sparse matrix operation in MF learning process, the researcher now introduces the Echo Nest Taste Profile Subset 41K (EN41K), a modified subset of the original Taste Profile dataset. The dataset contains contains the 41,062 user - song - play count triplets related to the first 2% of the most played songs and the first 0.5% of the most active users. The researcher defines the active level from the overall play count of a user's listening history in the dataset and retrieved 1,353 users' listening history profiles in the end. Within the listening history of those most active users, the researcher retrieved 4,362 music IDs that belong to the most played music. The percentage numbers of the most active users and the most played songs are determined by the requirements of recommendation, where the system only recommends songs that none of the group members had listened before. Therefore, the number of songs in the dataset has to be greater than the number of users so that a considerable amount of new songs can be recommended to a group. Table 2 shows a detailed statistical description of the sliced dataset.

Table 2. Summary statistics of the EN41K dataset

Dataset	EN41K dataset
#Users	1,353
#MSD–archived–songs	4362
#User–song–play count triplets	41,062

One of the problems with the original dataset is that the play count density in the input sparse matrix ($\vec{p}^{\,T}\vec{q}$) is not enough for the model to learn from. The density of a matrix is the number of non-zero elements divided by the total number of matrix elements. The original dataset used is MovieLens, which contains 1,000,209 anonymous ratings related to 3,706 movies and 6,040 users starting from the year of 2000. The ratings density for MovieLens dataset is around 4.4%. For the original Taste Profile dataset, the density number of user - song - play count triplets is around 0.01%. For the EN41K dataset, the density number is around 0.7%, which is nearly 70 times scaled compared to the Taste Profile dataset. The improvement of density helps to provide more information to the model and speed up the learning process.

3.3 Implicit music feedback

Given that the Taste Profile dataset does not contain any rating data, it is necessary to derive ratings from

play count information in order to make use of the matrix factorisation techniques. To achieve this, we define play frequency $freq$ for a given user i and song j to be the user's play count for that song normalised by user's total plays:

$$freq_{i,j} = \frac{count(i,j)}{\sum_{j'} count(i,j')} \quad (12)$$

We also adopt the notation $freq_k(i)$ to denote the frequency of the k-th most listened-to song for user i. (As Figure 2 shows, play frequencies have a clear long-tail distribution.) A rating for a song with rank k is computed as a linear function of the frequency percentile:

$$r_{i,j} = 4 \cdot \left(1 - \sum_{k'=1}^{k-1} freq_{k'}(i)\right) \quad (13)$$

As a result of this, we give the artists that fall within the top 25% frequency percentile a rating in the range of 3-4 stars, a rating of 2-3 stars to the artists who fall within the next 25% frequency percentile, and so on. Artists that are not listened to very often have ratings that are very close to 0.

Fig 4. Shows the distribution of the ratings from MovieLens dataset and EN41K dataset. The long tail feature of the distribution is no longer seen in the processed data of EN41K dataset, while the hypothetical curve of which is similar to the one in MovieLens dataset.

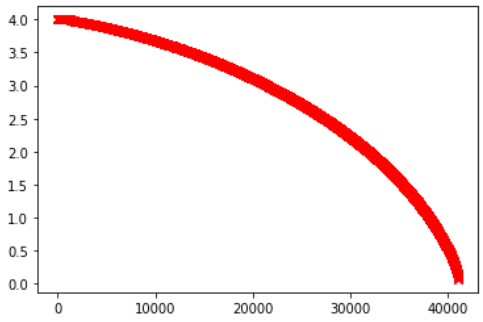

Fig 4. Rating (y-axis) distribution in MovieLens100K dataset

3.4 Group recommendation method

The researcher used an open-source code from Ortega et al.(2016) to conduct group recommendation process. The original source code from Ortega et al.(2016) developed three different methods to compute group factors in latent factor space, named as After Factorisation (AF), Before Factorisation (BF) and Weighted BF (WBF). The factors were then used to generate recommendations. As explained in 2.3, this recommendation approach can be regarded as pseudo-user approach since the group factors can be viewed as factors for a virtual user. Fig. 5 summarises the proposed methods.

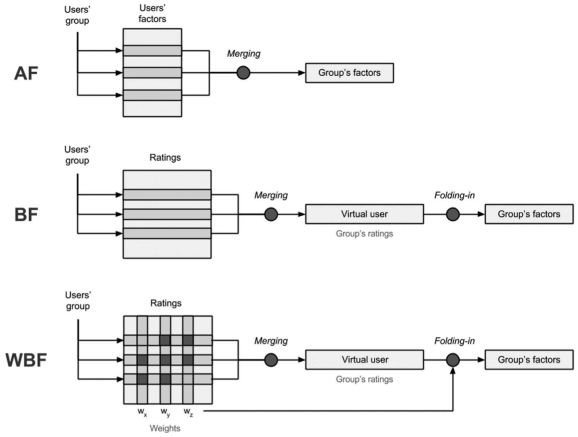

Fig 5. Matrix Factorisation based CF group recommendation approaches. (Ortega et al., 2016, p.315)

After factorisation approach (AF)

The After Factorisation (AF) method factorises a group of users by combining the factors of each user in the group. This strategy just employs the information created after factorisation, not the rating information, so that when the MF model is built, the users are unified into a group.

Let $G = \{u_1 \dots u_n\}$ be the set of users that belongs to the group G, let $\vec{p}_u = (p_{u,1} \dots p_{u,K})$ be the factor vector of the user u and let b_u be the bias of the user u.

We define $\vec{p}_G$ as the factor vector of the group G:

$$\vec{p}_G = \begin{pmatrix} h(p_{u_1,1}, \cdots, p_{u_n,1}) \\ \vdots \\ h(p_{u_1,K}, \cdots, p_{u_n,K}) \end{pmatrix} (14)$$

We define b_G as the bias of the group G:

$$b_G = h(b_{u_1}, \cdots, b_{u_n}) \, (15)$$

Where $h(x_1, \dots, x_n)$ is an aggregation function that combines the $x_1 \dots x_n$ values into one representative value (i.e.: average, maximum, etc).

Before factorisation approach (BF)

The Before Factorisation (BF) approach is based on the idea of representing the preferences of the group of users, $G = \{u_1, \cdots, u_n\}$, by means of a virtual user, u_G, aggregating the ratings of all the users in the group G. The aggregation is performed with the information we have before the MF process (the ratings). This approach is based on two steps:

- Simulating the ratings that the virtual user uG makes on the items, $r_{G,i}$ (Fig. 5, BF "merging"). This step is performed by means of a specific aggregation function h in the following way:

$$r_{G,i} = h(r_{u_1,i}, r_{u_2,i}, \cdots, r_{u_n,i}) \quad (16)$$

where $r_{u_1,i}, \cdots, r_{u_n,i}$ are the observed ratings of the users of the group G to the item i.

- Calculating the virtual user factors vector $(\overrightarrow{p}_G = (p_{G,1}, \cdots, p_{G,K}))$ and the virtual user bias (b_G) once the ratings $(r_{G,i})$ are defined (Fig. 5, BF "folding-in"). This step is achieved by solving the expression (17) where $\overrightarrow{q_i}$, b_i and μ are set to the values previously calculated in the learning phase:

$$\min_{\overrightarrow{p}_G, b_G} \sum_{r_{u_G,i} \neq \bullet} (r_{G,i} - \mu - b_u - b_i - \overrightarrow{p}_G{}^T \overrightarrow{q}_i)^2 + \lambda(||\overrightarrow{p}_{u_G}||^2 + ||\overrightarrow{q}_i||^2 + b_{u_G}^2 + q_i^2)$$

$$(17)$$

The previous expression can be simplified:

$$\min_{\overrightarrow{p}_G, b_G} \sum_{r_{u_G,i} \neq \bullet} \left((r_{G,i} - \mu - b_i) - (\overrightarrow{p}_G{}^T \overrightarrow{q}_i + b_G)\right)^2 + \lambda(||\overrightarrow{p}_G||^2 + b_G^2)$$

$$(18)$$

We define:

- $s_{G,i} = r_{G,i} - \mu - b_i$.
- $\overrightarrow{p}_G{}^* = (\overrightarrow{p}_G, b_G) = (p_{G,1}, \cdots, p_{G,K}, b_G)$ extends the vector $\overrightarrow{p}_G$.
- $\overrightarrow{q}_i{}^* = (\overrightarrow{q}_i, 1) = (q_{i,1}, \cdots, q_{i,K}, 1)$ extends the vector $\overrightarrow{q}_i$.

According to the definition, the previous expression is derived as follows:

$$\min_{\overrightarrow{p}_G{}^*} \sum_{s_{G,i} \neq \bullet} (s_{G,i} - \overrightarrow{p}_G{}^{*T} \overrightarrow{q}_i{}^*)^2 + \lambda ||\overrightarrow{p}_G{}^*||^2$$

$$(19)$$

As we can see, this minimisation corresponds to the definition of ridge regression. In order to simplify the notation, we will consider that the virtual user u_G has rated the items $\{1, \cdots, n_G\}$, and we define the matrix A:

$$A = \begin{pmatrix} q_{1,1} & \cdots & q_{1,k} & 1 \\ q_{2,1} & \cdots & q_{2,k} & 1 \\ \vdots & \ddots & \vdots & \vdots \\ q_{n_G,1} & \cdots & q_{n_G,k} & 1 \end{pmatrix} \quad (20)$$

We have that:

$$\begin{pmatrix} \overrightarrow{p}_G \\ b_G \end{pmatrix} = \begin{pmatrix} p_{G,1} \\ p_{G,2} \\ \vdots \\ p_{G,K} \\ b_G \end{pmatrix} = (A^T A + \lambda I)^{-1} A^T \begin{pmatrix} s_{G,1} \\ s_{G,2} \\ \vdots \\ s_{G,n_G} \end{pmatrix} \quad (21)$$

Weighted before factorisation approach (WBF)

Unlike the previous BF approach, the Weighted Before Factorisation (WBF) approach will weight each item according to the ratings of the users of the group. Here we will give more importance to those items that: (a) have been most rated by the users and (b) have similar ratings to the users of the group. We define $w_{G,i}$ as the weight of the item i for the group G as :

$$w_{G,i} = \frac{\#u \in G \,|\, r_{u,i} \neq \bullet}{\#G} \cdot \frac{1}{1 + \sigma_{G,i}} \quad (22)$$

where $\sigma_{G,i}$ is the standard deviation of the group G members' ratings for the item i and #denotes the cardinality of a set.

This approach only differs from the BF approach in the second step where we calculate the virtual user vector ($\overrightarrow{p}_G$) and the virtual user bias (b_G). In this case, our objective function is:

$$\min_{\overrightarrow{p}_G, b_G} \sum_{r_{u_G,i} \neq \bullet} w_{G,i}(r_{G,i} - \mu - b_u - b_i - \overrightarrow{p}_G{}^T \overrightarrow{q}_i)^2 + \lambda(||\overrightarrow{p}_{u_G}||^2 + ||\overrightarrow{q}_i||^2 + b_{u_G}^2 + q_i^2)$$

$$(23)$$

Following the same reasoning used in the previous section, we conclude that this expression is equivalent to a weighted ridge regression. Consequently, we have that:

$$\begin{pmatrix} \overrightarrow{p}_G \\ b_G \end{pmatrix} = \begin{pmatrix} p_{G,1} \\ p_{G,2} \\ \vdots \\ p_{G,K} \\ b_G \end{pmatrix} = (A^T W A + \lambda I)^{-1} A^T W \begin{pmatrix} s_{G,1} \\ s_{G,2} \\ \vdots \\ s_{G,n_G} \end{pmatrix} \quad (24)$$

where W is the diagonal matrix:

$$W = \begin{pmatrix} w_{G,1} & 0 & \dots & 0 \\ 0 & w_{G,2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & w_{G,n_G} \end{pmatrix} \quad (25)$$

3.5 Explaining Recommendations

The method proposed for explainability is a post-hoc approach, which means the explanation is generated after the recommendation process is performed. Several sources of data in the recommendation model that are useful in a way to reveal the relationships between group members, recommended songs and the group itself, including user factors, group factors and item factors. The sequence of the factors to a subject represents the vector of the subject in a latent vector space.

By computing the Euclidean distance between every group member's vector and the group's vector in the latent space, it tells the similarity between group members and the group itself. Group member vectors with lower Euclidean distance to the group vector mean they have a higher similarity. The more similar a group member's vector is to the group vector, the less different the group members' rating would be to the group rating. Because part of the recommendation process is to predict the group's overall rating by calculating the dot product of the group vector and the item vector. Consequently, the similarity between a group vector and a group member vector can be regarded as how much the overall recommended music as a playlist will represent the group member's taste. This representation can be seen as the influence on the recommendation result, particularly the influence on the playlist. To explain this simply and intuitively on the interface, the researcher made up the description: "Top influencers for the playlist", along with the ranking of the user name following the similarities computed using every group member vector.

$$similarity = \frac{1}{1 + d(\overrightarrow{p}_G, \overrightarrow{p}_u)} \quad (26)$$

The vector of recommended items and the group member vectors can also be useful to generate explanation. By computing the dot product and adding the bias of the user vector and item vector, the rating a user may give to an item can be predicted. When applying this to the group member vectors and recommended item vectors and ranking the results, it can be assumed that group members in the first few places are more likely to give higher ratings to the item; in another word, they enjoy more with the item. Therefore, the group member rankings for every recommended music can be explained in the playlist by using the phrase "Likely enjoyed by" along with the group members ranking on the basis of predicted ratings. This phrase and the group member information followed also explain why a song is recommended as part of the playlist.

4. Evaluation

4.1 Experiment Setup

The researcher reports results on a modified version of the Echo Nest Taste Profile dataset, named EN41K in short. The dataset contains 1,353 users with play counts for 4,362 songs they have ever listened to. The user-song-play count triplets reached 41,062 in total and have a user-song density of approximately 0.7%. In the Matrix Factorisation stage, the dataset was split into 80% for training and 20% for testing. The root mean square error (RMSE) was used as the performance evaluation criteria for recommending items to a single user. The researcher tested different training parameter sets to produce the best results as much as possible. Since there were certain restrictions with the algorithm, some training parameters could only be set within a limited range.

Figure 1 shows that increasing λ value can mitigate the over-fitting problem for the model since the error was initially really low in the training dataset and high in the test dataset. The gap between training error and test error was becoming closer until the λ value reached to 0.7 maximum. Figure 2 also shows a similar tendency of decreasing difference between training error and test error as the learning rate goes up until 0.15 maximum, a sweet point to have both lower test error and reasonable rise with training error. Figure 3 shows the effect on RMSE with increasing matrix factorisation iteration time; the rate of 5 iteration time is the most efficient. The researcher also tested the influence of different feature numbers for both the user vectors and

item vectors learned in a shared latent vector space. As seen in Figure 4, the preferred feature number is five because the difference between training error and test error gets bigger as the feature number goes up, which indicates a potential over-fitting problem. The increasing feature number may also require more data for training. The final training parameter set applied is λ = 0.7, Learning Rate = 0.15, Iteration time = 5 and Feature Number = 5, and the errors were 0.91447380953 training and 1.0147757779 test. Table 3 shows the parameters used to factorise the dataset.

Table 3.
Parameters used to factorise EN41K dataset.

Parameter	Value
λ	0.7
Learning rate (μ)	0.15
Feature numbers	5
Max. iterations	5

For group recommendation experiments, The researcher ran the experiment for all approaches along with three group sizes for 20 times. Neither the Echo Nest Taste Profile dataset nor EN41K dataset contains information about how the users are grouped. 10 groups of users were randomly generated, fixing their sizes from 3 users for small groups, 5 users for medium groups and 10 users for large groups. The proposed recommendation approaches are capable of working with different h function implementations. Authors in Ortega et al., 2016 tried each approach with six classical aggregation functions: average, weighted average, mode, median, least misery and most pleasure. The researcher continued to use their conclusion as follows: for the AF approach the best result is achieved using weighted average (the number of ratings of each user has been used as a weight) for MovieLens and median for Netflix. For the BF and WBF approaches, the best results are achieved using least misery in all cases.

4.2 Evaluation Metrics

In order to check the quality of recommendations computed for the group of users, we define precision and recall for the group G as:

$$precision_G = \frac{\#TP_G}{\#(TP_G \bigcup FP_G)} \quad (27)$$

$$recall_G = \frac{\#TP_G}{\#T_G} \quad (28)$$

Where TP_G, FP_G and T_G denote the true positive, false positive and expected recommendations sets respectively:

$$FP_G = \left\{ i \in R_G \,|\, \exists g \in G \text{ such that } \hat{r}_{g,i} < \theta \right\} \quad (29)$$

$$TP_G = \left\{ i \in R_G \,|\, \exists g \in G \text{ such that } r_{g,i} \neq \bullet \text{ and } \forall u \in G\, \hat{r}_{u,i} \neq \bullet \rightarrow \hat{r}_{u,i} \geq \theta \right\} \quad (30)$$

$$T_G = \left\{ i \in I \,|\, \exists g \in G \text{ such that } r_{g,i} \neq \bullet \text{ and } \forall u \in G\, \hat{r}_{u,i} \neq \bullet \rightarrow \hat{r}_{u,i} \geq \theta \right\} \quad (31)$$

where R_G represents the set of items recommended to the group G, $\hat{r}_{u,i}$ denotes the test-rating of the user u to the item i (the test ratings are not used during the recommendations computation phase), and θ is a threshold to measure whether a user likes or dislikes an item (the test dataset have a rating scale from 1 star to 4 stars, with median rating being around 3, therefore threshold $\theta = 3$ is used).

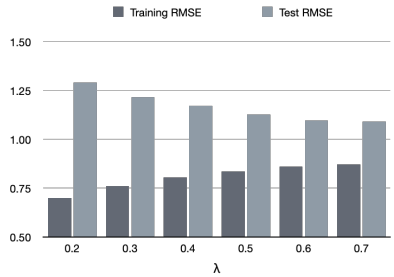

Figure 1. RMSE in Training dataset and Test dataset versus λ value. The finial errors are approximately 0.87 training and 1.09 test. Other parameters are Learning Rate = 0.05, Iteration Time = 5 and Feature Numbers = 15.

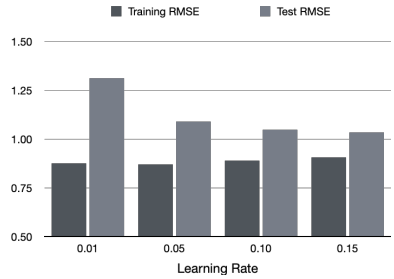

Figure 2. RMSE in Training dataset and Test dataset versus Learning Rate parameter. The final errors are approximately 0.91 training and 1.03 test. Other parameters are λ = 0.7, Iteration Time = 5 and Feature Numbers = 15.

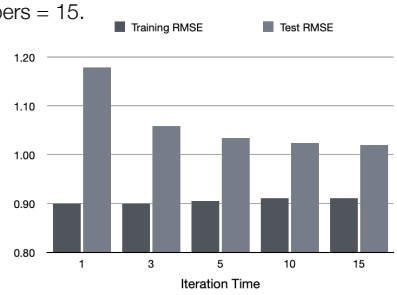

Figure 3. RMSE in Training dataset and Test dataset versus Iteration Time parameter. The errors for 5 iteration time are approximately 0.91 training and 1.02 test. Other parameters are λ = 0.7, Learning Rate = 0.15 and Feature Numbers = 15.

Figure 4. RMSE in Training dataset and Test dataset versus Feature Number parameter. The errors for 5 feature number are approximately 0.91 training and 1.03 test. Other parameters are λ = 0.7, Learning Rate = 0.15 and Iteration Time = 5.

Table 4.
Averaged recall values after 20 experiments of AF, BF and WBF approaches in three group sizes

	Small group (n=3)	Mid size group (n=5)	Large group (n=10)
AF	0.004194	0.003133	0.004312
BF	0.001704	0.002062	0.002363
WBF	0.002349	0.001849	0.002459

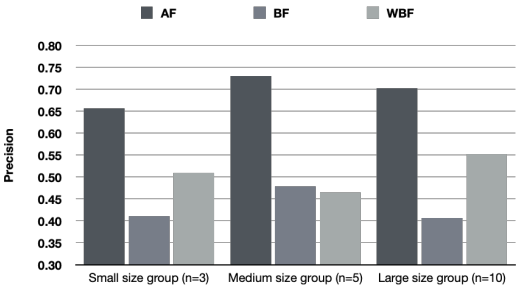

Figure 5. Averaged precision values after 20 experiment of AF, BF and WBF approaches in three group sizes: (a) small size group (n=3) (b) medium size group (n=5) (c) large size group (n=10)

Table 5.
Precision and recall value of AF, BF and WBF in 20 experiments, classified in three group sizes.

	Small group (n=3)			Medium size group (n=5)			Large group (n=10)		
	AF	**BF**	**WBF**	**AF**	**BF**	**WBF**	**AF**	**BF**	**WBF**
1	[1.0, 0.004223]	[0.5, 0.000925]	[1.0, 0.000925]	[1.0, 0.003270]	[0.3333333, 0.001162]	[0.5, 0.002055]	[0.75, 0.001776]	[0.25, 0.001162]	[0.6, 0.001777]
2	[0.666667, 0.005449]	[nan, 0.0]	[0.0, 0.0]	[0.75, 0.003732]	[1.0, 0.002778]	[nan, 0.0]	[0.75, 0.001772]	[0.6, 0.004135]	[0.75, 0.002813]
3	[0.0, 0.0]	[0.0, 0.0]	[0.333333, 0.001667]	[0.75, 0.004768]	[1.0, 0.003465]	[0.666667, 0.003414]	[1.0, 0.008757]	[0.5, 0.003722]	[1.0, 0.004658]
4	[0.8, 0.008631]	[nan, 0.0]	[1.0, 0.003472]	[0.714286, 0.008427]	[0.0, 0.0]	[1.0, 0.001808]	[0.571429, 0.004077]	[0.0, 0.0]	[0.833333, 0.005537]
5	[1.0, 0.006439]	[0.4, 0.002488]	[0.0, 0.0]	[1.0, 0.003337]	[0.0, 0.0]	[0.5, 0.001064]	[0.75, 0.003239]	[0.416667, 0.002633]	[0.25, 0.000893]
6	[0.666667, 0.003042]	[0.0, 0.0]	[0.333333, 0.0007692]	[0.5, 0.00078125]	[0.0, 0.0]	[0.0, 0.0]	[0.333333, 0.0025]	[0.5, 0.002381]	[0.5, 0.002381]
7	[1.0, 0.005442]	[0.0, 0.0]	[0.0, 0.0]	[0.666667, 0.004257]	[0.5, 0.005132]	[0.5, 0.003631]	[0.7, 0.003694]	[0.583333, 0.003764]	[0.5, 0.001163]
8	[1.0, 0.002]	[0.5, 0.002083]	[0.333333, 0.002273]	[1.0, 0.001724]	[0.166667, 0.000877]	[0.5, 0.002944]	[0.666667, 0.003636]	[0.666667, 0.003102]	[0.0, 0.0]
9	[0.0, 0.0]	[1.0, 0.001064]	[0.0, 0.0]	[1.0, 0.003329]	[0.5, 0.002941]	[0.333333, 0.000725]	[0.833333, 0.007052]	[0.0, 0.0]	[1.0, 0.002947]
10	[0.625, 0.007446]	[0.5, 0.00125]	[0.5, 0.005779]	[0.333333, 0.000893]	[0.75, 0.004361]	[1.0, 0.000893]	[0.6, 0.005705]	[0.25, 0.000877]	[0.5, 0.000877]
11	[0.714286, 0.011369]	[0.0, 0.0]	[0.333333, 0.002632]	[1.0, 0.001852]	[0.25, 0.000747]	[0.0, 0.0]	[0.75, 0.004027]	[1.0, 0.003286]	[1.0, 0.002414]
12	[0.0, 0.0]	[0.5, 0.003333]	[1.0, 0.003333]	[0.5, 0.002]	[0.666667, 0.005573]	[0.5, 0.002]	[0.6, 0.004027]	[0.0, 0.0]	[0.0, 0.0]
13	[1.0, 0.003448]	[0.0, 0.0]	[1.0, 0.003448]	[1.0, 0.002322]	[0.25, 0.001613]	[0.5, 0.002419]	[0.7, 0.003637]	[0.2, 0.000877]	[0.5, 0.000862]
14	[0.5, 0.000725]	[0.666667, 0.003215	[1.0, 0.000725]	[0.5, 0.001724]	[0.333333, 0.002632]	[0.75, 0.005282]	[0.416667, 0.004832]	[0.5, 0.006161]	[0.25, 0.002273]
15	[1.0, 0.005392]	[1.0, 0.003333]	[1.0, 0.004314]	[0.333333, 0.002273]	[0.0, 0.0]	[0.0, 0.0]	[0.785714, 0.008667]	[0.2, 0.000962]	[0.5, 0.003709]
16	[1.0, 0.006275]	[1.0, 0.005719]	[1.0, 0.004248]	[0.25, 0.001852]	[0.333333, 0.00125]	[0.25, 0.00125]	[0.625, 0.004579]	[0.5, 0.002677]	[0.5, 0.003476]
17	[0.666667, 0.00625]	[0.5, 0.0012120]	[0.5, 0.002083]	[0.5, 0.004167]	[1.0, 0.004630]	[1.0, 0.005903]	[0.75, 0.005087]	[0.4, 0.002178]	[0.333333, 0.002477]
18	[1.0, 0.003589]	[0.0, 0.0]	[0.0, 0.0]	[1.0, 0.002428]	[0.5, 0.001136]	[0.0, 0.0]	[0.8, 0.003478]	[0.666667, 0.004265]	[0.666667, 0.007384]
19	[0.5, 0.004167]	[0.333333, 0.004688]	[0.2, 0.004167]	[1.0, 0.001282]	[1.0, 0.0012820]	[nan, 0.0]	[0.666667, 0.002369]	[0.416667, 0.0040018]	[0.375, 0.003002]
20	[0.0, 0.0]	[0.5, 0.004762]	[0.666667, 0.007143]	[0.8, 0.008252]	[1.0, 0.001667]	[0.375, 0.003587]	[1.0, 0.003338]	[0.5, 0.001064]	[1.0, 0.000532]

Table 6.
Factors and bias of group profile and group members, an example from a medium size group using AF method

User Index	Similarity	F_1	F_2	F_3	F_4	F_5	b_u
(Group profile)	-	0.02101781	0.04496065	-0.07088136	0.00342788	-0.02857114	-0.04823289
734	93%	0.07451212	0.08319631	-0.05291501	-0.00177048	-0.02293285	-0.16143093
808	92%	-0.01362764	0.01744778	-0.11239957	0.00766848	-0.07983841	-0.13342482
512	92%	-0.02173481	0.01814226	-0.04460008	0.00639390	0.03168001	0.25780084
1134	48%	-0.64848781	0.18999331	0.06307645	0.73891721	-0.28622930	0.00000000
1270	45%	0.09736188	0.26146698	0.41860049	0.70812905	0.73932672	0.00000000

4.3 Experimental results

Figure 5 shows the averaged precision values from the results of 20 times experiment in AF, BF and WBF approaches along with three group sizes: (a) small group (n=3) (b) mid size group (n=5) (c) large group (n=10). Table 4 shows the averaged recall values for three proposed approaches along with three group sizes. Table 5 shows the complete precision and recall results in 20 experiments, divided into three group recommendation methods and three group sizes.

The averaged values in both precision and recall results showed a similar pattern. Group recommendation using the AF approach has the highest precision and recall values for all sized groups. The BF approach used in group recommendation has slightly higher precision and recall values than the WBF approach in medium size group. Finally, the WBF approach has significantly higher precision and recall values than the BF approach in the small and large groups.

Table 6 presents an example group vector and the group member vectors sorted according to their similarities to the group vector. Table 6 contains the recommended music index along with the factors and bias. Table 7 shows the group member rate rankings for each recommended song. The similarities presented in Table 6 can be used to support the explanation ”Top influencers of the playlist”. In Table 7, user indexes with a high rate ranking in each song can be used to support the explanation “Likely enjoyed by”.

Table 7.
Indexes of recommended song to a group and user indexes of every group member ranked by their predicted rating for corresponding song, an example from a medium size group using AF method

	Music Index	User Index				
		1st	2nd	3rd	4th	5th
1	1540	512	1134	808	734	1270
2	3471	512	1270	808	734	1134
3	4114	1134	512	808	734	
4	4125	1270	512	1134	734	808
5	3997	1134	1270	512	808	734
6	2732	512	1134	1270	808	734
7	3826	512	1134	808	734	
8	3731	512	1134	808	734	1270
9	3714	512	1134	1270	808	734
10	3610	1134	512	1270	808	734
11	1694	512	808	734		
12	3870	512	1134	808	734	1270
13	4241	1134	512	1270	808	734
14	3339	1134	512	1270	808	734
15	3696	1134	512	808	734	
16	4218	512	1270	1134	808	734
17	55	512	1134	1270	808	734
18	1669	1134	512	808	1270	734
19	1992	512	1270	808	734	1134
20	1475	1134	512	1270	808	734
21	2887	1270	512	1134	734	808
22	287	512	1134	1270	808	734
23	220	512	1134	1270	808	734
24	4002	512	808	734	1270	1134
25	3341	512	808	734	1270	1134

5. Discussion

The results from the training process showed the feasibility of implementing implicit user feedback on the algorithm designed for explicit user feedback. Since the root mean square errors from matrix factorisation and process are in a reasonable range, it is practical to convert play count data to rating data as a method similar to feature engineering. Not only was the play count value controlled in a shorter range for factorisation, but the long tail effect of the distribution was also regulated well to remove any outliers that may interfere with the training process and the final model. This method was initially applied to a Last.fm dataset in the work of Pacula, 2010. And the skewed distribution of the unprocessed dataset impacted parameter estimation because the optimisation is susceptible to getting stuck in a local minimum, where for each user, all available items have a low rating. They addressed the local minimum issue by applying a new Stochastic Gradient Descent algorithm. This issue was not investigated and addressed in this paper as it may potentially affect both the factorisation and the group recommendation process with certain complexity, which would be better discussed in another paper.

The evaluation results from the group recommendation process showed the performance differences between the three proposed group recommendation methods. Among the three group recommendation methods, The AF method had a significantly higher precision value in all three group sizes, with the best performance (precision = 0.729881) for medium-size groups (n=5). This also applies to the performance of recall value, where the AF method outperforms the BF and the WBF methods. However, it can be observed that all recall values are extremely low. The recall value tells the ratio of recommended and liked songs to all songs that meet the same criteria. Based on the meaning of recall value, the low recall value is because the system only recommends 25 songs to each group, while the whole dataset has 41,062 songs. The recall value itself may not reflect enough information to assess the model's ability to detect positive samples in this case, but it's still informative when compared with other recall values from different group recommendations.

It can also be observed in Table 5 that the precision value can go up to one and down to zero in certain situations. The precision value means the percentage of songs recommended to a group that every group member would like. When a precision value is one, it means all the recommended songs are over the satisfaction threshold ($\theta = 3$) that was previously set. Similarly, a zero precision value means all songs in the result are under the satisfaction threshold. This could result from the trimming process conducted with the original dataset, where the researcher extracted users with the highest listening times and songs with the highest play count. The issue can come in two ways; one is that some most-played music is popular enough for every group member to like and give a high rate, and another is that the listening history of all the members in a group covers a large part of the most-played songs. This could be addressed by adjusting the trimming process or using a larger dataset. It may also relate to the satisfaction threshold.

Another aspect of the influence from the trimming process is the selection bias embedded in the dataset-trimming process. Although the extracted dataset has a high density for a better learning process, it lacks the variety of music and users, in a way, made the model have a strong tendency to predict higher ratings of popular music, as well as providing fewer options in the candidate filtering stage of group recommendation. Furthermore, the trimmed dataset only reflects a tiny part of the music streaming platform since it was highly concentrated. The limited reflection of the real-world listening activities may bring strong bias to the recommendation results.

The explanation methods were proven to be able to generate after the recommendation had been performed. "Top influencers for the playlist" informs the representation of every group member in the playlist, while "Likely enjoyed by…" tells why a song is recommended in detail. The explanations provide clarity to the results so that users can have a clearer understanding of the playlist. On the other hand, those two explanations may not help to improve the user's satisfaction with the results and can even reveal any bias they have when the "top influencers" outweigh other group members, though it may still help to build user trust. It would be interesting to conduct a user test to look into how the explanations affect the mental model in users. The user test may also draw on whether the explanation actually brings trust and clarity to the recommendation process.

6. Conclusion and future work

In this paper, the researcher implemented an implicit user feedback dataset on a group recommendation system designed for explicit user feedback, given the limits of obtaining explicit user feedback in real-world applications. The researcher also tested three group recommendation methods and designed two explanation methods for the recommendations.

The experiment result shows it is practical to utilise the implicit user feedback by estimating ratings from the user - song - play count data. This conversion process restricted the play count value in a shorter range and mitigated the long tail effect from the original dataset. The researcher also trimmed the original dataset before conversion by extracting the most listened songs and users with the highest listening times. After these procedures, the final dataset was successfully applied to the matrix factorisation process, and the group recommendation process performed well.

Among the three group recommendation methods, AF is proven to be the best for all group sizes, with higher precision and recall value. The explanation methods derived from the group recommendation process were executed after the group recommendation was made. The explanations worked by revealing the relationships between the recommendation result, group members and the group itself, thus informing the reason for every suggested music and how much every group member is represented in the playlist. The explanation brought clarity and may potentially help to build user trust, but it may also reveal any bias from the system. The researcher is considering doing future studies around user tests that draw on how the explanation affects the mental model of users.

However, the dataset trimming process before the conversion also created selection bias, which affects performance assessment and dataset validity in representing the real world. The researcher looks forward to future studies to help address these issues, including but not limited to: a) modifying the evaluation method, b) applying a larger dataset, and c) optimising the sampling process.

References

1. Adomavicius, G. and Tuzhilin, A. (2005). Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6), pp.734–749. doi:10.1109/tkde.2005.99.
2. Aucouturier, J. and Pachet, F. (2004). Improving Timbre Similarity: How high's the sky? *Journal of negative results in speech and audio sciences*, 1(1).
3. Bertin-Mahieux, T., P. W. Ellis, D., Whitman, B. and Lamere, P. (2011). The Million Song Dataset. Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011). doi:https://doi.org/10.7916/D8NZ8J07.
4. Bogdanov, D. and Herrera, P. (2011). How much metadata do we need in music recommendation? A subjective evaluation using preference sets. *12th International Society for Music Information Retrieval Conference*, ISMIR 2011.
5. Bokde, D., Girase, S. and Mukhopadhyay, D. (2015). Matrix Factorisation Model in Collaborative Filtering Algorithms: A Survey. *Procedia Computer Science*, 49, pp.136–146. doi:10.1016/j.procs.2015.04.237.
6. Boratto, L. and Carta, S. (2010). Information Retrieval and Mining in Distributed Environments pp 1–20Cite as State-of-the-Art in Group Recommendation and New Approaches for Automatic Identification of Groups. *Information Retrieval and Mining in Distributed Environments*. doi:DOI: 10.1007/978-3-642-16089-9_1.
7. Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modelling and User-Adapted Interaction*, [online] 12(4), pp.331–370. doi:10.1023/a:1021240730564.
8. Chen, Y.-L., Cheng, L.-C. and Chuang, C.-N. (2008). A group recommendation system with consideration of interactions among group members. *Expert Systems with Applications*, 34(3), pp.2082–2090. doi:10.1016/j.eswa.2007.02.008.
9. Christensen, I.A. and Schiaffino, S. (2011). Entertainment recommender systems for group of users. *Expert Systems with Applications*, 38(11). doi:10.1016/j.eswa.2011.04.221.
10. Colomer, J.M. (2011). Ramon Llull: from 'Ars electionis' to social choice theory. *Social Choice and Welfare*, 40. doi:https://doi.org/10.1007/s00355-011-0598-2.
11. Crossen, A., Budzik, J. and Hammond, K.J. (2002). Flytrap: intelligent group music recommendation. *Proceedings of the 7th international conference on Intelligent user interfaces*. doi:https://doi.org/10.1145/502716.502748.
12. Dara, S., Chowdary, C.R. and Kumar, C. (2019). A survey on group recommender systems. *Journal of Intelligent Information Systems*. doi:10.1007/s10844-018-0542-3.
13. Garcia, I., Sebastia, L. and Onaindia, E. (2011). On the design of individual and group recommender systems for tourism. *Expert Systems with Applications*, [online] 38(6), pp.7683–7692. doi:10.1016/j.eswa.2010.12.143.
14. Hu, Y., Koren, Y. and Volinsky, C. (2008). Collaborative Filtering for Implicit Feedback Datasets. *2008 Eighth IEEE International Conference on Data Mining*. doi:10.1109/icdm.2008.22.
15. Kaššák, O., Kompan, M. and Bieliková, M. (2016). Personalized hybrid recommendation for group of users: Top-N multimedia recommender. *Information Processing & Management*, 52(3), pp.459–477. doi:10.1016/j.ipm.2015.10.001.
16. Kim, H.-N. and Saddik, A.E. (2015). A stochastic approach to group recommendations in social media systems. *Information Systems*, 50, pp.76–93. doi:10.1016/j.is.2014.10.002.
17. Kim, J.K., Kim, H.K., Oh, H.Y. and Ryu, Y.U. (2010). A group recommendation system for online communities. *International Journal of Information Management*, 30(3), pp.212–219. doi:10.1016/j.ijinfomgt.2009.09.006.
18. Kim, N. and Lee, J.-H. (2014). Group recommendation system: Focusing on home group user in TV domain. 2014 Joint 7th International Conference on Soft Computing and Intelligent Systems (SCIS) and 15th International Symposium on Advanced Intelligent Systems (ISIS). doi:10.1109/scis-isis.2014.7044866.
19. Kompan, M. and Bielikova, M. (2014). Group Recommendations: Survey and Perspectives. *COMPUTING AND INFORMATICS*, 33(2).
20. Koren, Y., Bell, R. and Volinsky, C. (2009). Matrix Factorisation Techniques for Recommender Systems. *Computer*, 42(8), pp.30–37. doi:10.1109/mc.2009.263.
21. Liao, Q.V. and Varshney, K.R. (2022). Human-Centered Explainable AI (XAI): From Algorithms to User Experiences. [online] doi:https://doi.org/10.48550/arXiv.2110.10790.
22. Lines, B., Tsunoo, E., Tzanetakis, G. and Ono, N. (2011). Beyond Timbral Statistics : Improving Music Classification Using Percussive. *IEEE Transactions on Audio, Speech and Language Processing*. doi:10.1109/TASL.2010.2073706.
23. Marques, J. and Moreno, P.J. (1999). A Study of Musical Instrument Classification Using Gaussian Mixture Models and Support Vector Machines.
24. Mccarthy, J.F. and Anagnost, T.D. (2000). Musicfx: An arbiter of group preferences for computer

supported collaborative workouts. doi:10.1145/289444.289511.
25. McCarthy, K., Salamó, M., Coyle, L. and Mcginty, L. (2006). CATS: A Synchronous Approach to Collaborative Group Recommendation. *Proceedings of the Nineteenth International Florida Artificial Intelligence Research Society Conference*.
26. McFee, B., Bertin-Mahieux, T., Ellis, D.P.W. and Lanckriet, G.R.G. (2012). The million song dataset challenge. *Proceedings of the 21st international conference companion on World Wide Web - WWW '12 Companion*. doi:10.1145/2187980.2188222.
27. Michael I, M. and Daniel P. W., E. (2005). Song-Level Features and Support Vector Machines for Music Classification. *ISMIR 2005: 6th International Conference on Music Information Retrieval: Proceedings*. doi:https://doi.org/10.7916/D8QV3WWQ.
28. Ortega, F., Hernando, A., Bobadilla, J. and Kang, J.H. (2016). Recommending items to group of users using Matrix Factorization based Collaborative Filtering. *Information Sciences*, 345, pp.313–324. doi:10.1016/j.ins.2016.01.083.
29. Pacula, M. (2010). *A Matrix Factorization Algorithm for Music Recommendation using Implicit User Feedback*. [online] Available at: https://mpacula.com/~mpaculac/publications/lastfm.pdf [Accessed 6 Oct. 2022].
30. Peake, G. and Wang, J. (2018). Explanation Mining. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. doi:10.1145/3219819.3220072.
31. Preece, A., Harborne, D., Braines, D., Tomsett, R. and Chakraborty, S. (2018). Stakeholders in Explainable AI. doi:https://doi.org/10.48550/arXiv.1810.00184.
32. Quijano-Sánchez, L., Recio-García, J.A. and Díaz-Agudo, B. (2010). Personality and Social Trust in Group Recommendations. *2010 22nd IEEE International Conference on Tools with Artificial Intelligence*. doi:10.1109/ICTAI.2010.92.
33. Ravi, L. and Vairavasundaram, S. (2016). A Collaborative Location Based Travel Recommendation System through Enhanced Rating Prediction for the Group of Users. *Computational Intelligence and Neuroscience*, 2016, pp.1–28. doi:10.1155/2016/1291358.
34. Rentfrow, P.J. and Gosling, S.D. (2003). The do re mi's of everyday life: The structure and personality correlates of music preferences. *Journal of Personality and Social Psychology*, [online] 84(6), pp.1236–1256. doi:10.1037/0022-3514.84.6.1236.
35. Resnick, P. and Varian, H.R. (1997). Recommender systems. *Communications of the ACM*, 40(3), pp.56–58. doi:10.1145/245108.245121.
36. Salehi-Abari, A. and Boutilier, C. (2015). Preference-oriented Social Networks: Group Recommendation and Inference. *Proceedings of the 9th ACM Conference on Recommender Systems*. doi:https://doi.org/10.1145/2792838.2800190.
37. Song, Y., Dixon, S. and Pearce, M. (2012). A Survey of Music Recommendation Systems and Future Perspectives. *9th International Symposium on Computer Music Modeling and Retrieval*, 4.
38. Sotelo, R., Blanco-Fernández, Y., López-Nores, M. and Gil, A. (2009). TV program recommendiation for groups based on multidimensional TV-Anytime classifications. *Digest of Technical Papers - IEEE International Conference on Consumer Electronics*. doi:10.1109/ICCE.2009.5012309.
39. Su, X. and Khoshgoftaar, T.M. (2009). A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence*, 2009, pp.1–19. doi:10.1155/2009/421425.
40. Villavicencio, C.P., Schiaffino, S., Díaz-Pace, J.A. and Monteserin, A. (2016). A Group Recommendation System for Movies based on MAS. *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal*, 5(3), pp.1–12. doi:10.14201/adcaij201653112.
41. VOZALIS, M. and MARGARITIS, K. (2007). Using SVD and demographic data for the enhancement of generalized Collaborative Filtering. *Information Sciences*, 177(15), pp.3017–3037. doi:10.1016/j.ins.2007.02.036.
42. Wang, J., de Vries, A.P. and Reinders, M.J.T. (2006). Unifying user-based and item-based collaborative filtering approaches by similarity fusion. *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '06*. [online] doi:10.1145/1148170.1148257.
43. Wang, X., Chen, Y., Yang, J., Wu, L., Wu, Z. and Xie, X. (2018). A Reinforcement Learning Framework for Explainable Recommendation. *2018 IEEE International Conference on Data Mining (ICDM)*. doi:10.1109/icdm.2018.00074.
44. Zhang, Y. and Chen, X. (2020). Explainable Recommendation: A Survey and New Perspectives. *Foundations and Trends® in Information Retrieval*, 14(1), pp.1–101. doi:10.1561/1500000066.
45. Zhang, Y., Lai, G., Zhang, M., Zhang, Y., Liu, Y. and Ma, S. (2014). Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, 1. doi:10.1145/2600428.2609579.