The Recommenders: Recommendations based on Music Playlists

Stela Topalova**
stelatopalova@abv.bg
Maastricht University
Maastricht, Limburg, Netherlands

Joan Botzev*
jbotzev@gmail.com
Maastricht University
Maastricht, Limburg, Netherlands

Anna-Maria Pervan
a.pervan@student.maastrichtuniversity.nl
Maastricht University
Maastricht, Limburg, Netherlands

Utku Arslan*
utku.arslan@student.maastrichtuniversity.nl
Maastricht University
Maastricht, Limburg, The Netherlands

Abstract

Digital music streaming has brought a revolution in a way music is discovered and distributed. Recent studies show that playlists are on a track to become users' preferred way to discover new songs and expand their music taste. Therefore, providing accurate music recommendations regarding playlists both to an individual and a group of users is more important than ever.
This paper introduces several music recommendation approaches focused on an individual user including frequency sets, and nearest neighbor (NN) user- and item-based collaborative filtering. Furthermore, the paper presents an additive aggregation approach to creating music recommendations for a group of users.Finally, the paper explores how the generated recommendations can be explained to both an individual and a group of different users. To evaluate the effect different explanation styles have on users, a user survey has been distributed online. The findings were presented by performing a qualitative analysis of the survey results.

CCS Concepts

• Human-centered computing; **• Social and professional topics**; **• Information systems → Clustering and classification**;

ACM Reference Format:
Stela Topalova, Joan Botzev, Anna-Maria Pervan, and Utku Arslan. 2022. The Recommenders: Recommendations based on Music Playlists. In *Proceedings of Maastricht University (RECSYS'21).* ACM, New York, NY, USA, 8 pages.

1 Introduction

Spotify provided a public dataset for 'RecSys Challenge 2018. This paper uses a small subset of the provided data and aims to implement a recommendation algorithm capable of suggesting appropriate songs in order to continue the given playlist of a single user. It addresses the following research questions:
RQ1: How to choose a song to extend an individual user's playlist?
RQ2: How to choose a playlist to recommend to an individual user?
The first two research questions are handled by applying multiple individual recommendation strategies: Frequency Sets, User-based Collaborative Filtering (CF) with adjusted KNN and Nearest Neighbour Item-based CF. This paper introduces a novel approach by making adjustments to the User-based CF.
The paper also explores how to recommend a playlist a group of different users would enjoy. Therefore, it aims at answering the following research questions:
RQ3: How to choose a song to extend a group playlist?
RQ4: How to choose a playlist to recommend to a group of users?
To address the problem of group recommendations, the additive aggregation strategy is used.
The project implements explanations to help users gain understanding why they have been recommended a certain playlist or a song in an individual or a group setting. An evaluation of the explanations of both individual and group recommendations is performed to gauge which type of an explanation would perform best in terms of several different purposes, namely transparency, effectiveness, persuasivenss, interpretability and efficiency. This is done through a user survey which is distributed online. Users are asked to evaluate the different styles of explanations on how well they fulfilled the aforementioned purposes. The results are evaluated by performing a qualitative study of the data.

2 Related Work

Best performing approaches in the RecSys Challenge are published and presented during the 12th ACM Conference on Recommender Systems. In this section, neighbourhood- and collaborative filtering-based approaches which entered the list of top 10 best performing algorithms are presented.
Ludewig et. al uses a hybrid session-based music recommendation method which combines the NN techniques, item-based CF and a standard matrix factorization and a small set of heuristics based on the music playlist titles. [1]
Zhu et. al combine neighbour-based CF taking into the account both similarities between the playlists and the songs with discriminative reweighting and reranking techniques in order to improve recommendation quality. [2]
Kelen et. al based their approach on CF using the KNN method in order to compute the similarity of playlists and to recommend tracks from similar ones. [3]

*All authors contributed equally to this research.

RECSYS'21, December 2021, Maastricht, The Netherlands

Faggioli et. al created a hybrid algorithm based on content and collaborative filtering. They combine a kernel-based CF approach with the idea that playlists with similar names are more likely to have similar content. [4]
Approaches used by other high-ranking teams were two stage models, matrix factorization, neural networks and learning to rank algorithms. [5]
In contrast to the music recommendations for individual users, group music recommendations have not been studied well and therefore they're not covered in this section.

2.1 Data

The dataset is taken from Spotify's Million Playlist Dataset [1] (MPD) re-released as a part of the RecSys Challenge 2018. It consists of 1 million user created playlists, including playlist titles and song titles. In order to build and test the aforementioned recommender system, statistical analysis is performed (Table 1) and the data are reformatted to fit the purpose of this paper.

2.1.1 Pre-processing Initially, the dataset is in sliced JSON file format, sorted by Playlist IDs (pid). The dataset is computationally too exhausting to prototype on therefore only a subset is taken.

To make use of the data, the JSON files are flattened and saved into CSV. The CSV file contains all the tracks with the playlists they are in and the playlists' name, the albums the tracks are in and the artists of the tracks.

2.1.2 User and Ratings Generation Since the data has no users or user listening history, synthetic data is generated by creating u number of users and each user is assigned p number of playlists. It is observed that a song appearing multiple times in playlists is rare, hence it is decided to count how many times an artist appears in a user's playlists and this is referred as "ratings" in the rest of this paper.

3 Methodology

This section starts by covering several implemented algorithms which make recommendations to individual users. It continues by explaining how group recommedations are generated using an additive aggregation strategy. Finally, it presents explanations implemented to explain these recommendations to the users.

3.1 Individual Recommendation

3.1.1 User-Based Collaborative Filtering using adjusted KNN Firstly, the target user that is to be recommended is decided. Adjusted KNN is used with a set amount of neighbours and the cosine similarity. The number of times an artist appears in all the neighbours' playlists is summed. To tackle RQ1, one song of the top rated artist (TRA) is picked from the playlists of the neighbours.
When it comes to RQ2, the h highest rated artists (HRA) are picked and a formula to decide how many songs per artist (SPA) to suggest is create:

$$SPA(A_i) = ceil\left(\frac{rating(A_i)}{sum(ratings)} \times N\right) \tag{1}$$

[1] aicrowd.com/challenges/spotify-million-playlist-dataset-challenge

where A_i is the i-th artist in the recommended list of artists, *rating(A_i)* is the rating of each artists, *sum(ratings)* is the sum of ratings of all artists and N is the number of songs that are to be recommended. The ratings are normalized to be between 0 and 1 and multiplied by the number of songs wished to be recommended. However, as our ratings are created based on the artists, there are popular ones which cause bias and have a higher probability of being picked. There will not be much diversity in the recommended playlist. To deal with this problem a threshold is set in which an artist is limited not to take more than 25% of the number of suggested songs. Adjusted KNN picks neighbours that have similar taste in artists like the target user, therefore the songs are taken from the neighbours' playlists as there is a high probability that the target user will like them.

3.1.2 Item-Based Collaborative Filtering The item-based collaborative filtering approach is based on the idea that the user is likely to enjoy songs that are similar to the songs that are already in their playlist. The similarity between the songs is based on their co-occurence across different playlists. Since the dataset doesn't contain any user information due to privacy, it was impossible to create a user-item ratings matrix. Instead, each song was represented as a binary vector over all playlists. The entry for a specific playlist was set to 1 if the song is included in it and to 0 otherwise. The songs which are already included in the playlist are called seed tracks. In order to generate recommendations for a single seed track in the playlist, k-nearest neighbours algorithm (KNN) was applied using Scikit-Learn framework with the cosine similarity as a measure. The seed tracks were excluded from consideration. The cosine similarity measures the similarity by the cosine of the angle between two vectors. Given two vectors s_1 and s_2, it is given by

$$S_C(s_1, s_2) = \frac{s_1 \cdot s_2}{||s_1||||s_2||} \tag{2}$$

. The recommendations for a single seed track are given by top k most similar songs ordered by similarity.
The downside of this approach is its tendency to focus on popular songs [6]. In order to make up for popularity bias, the inverse document frequency (IDF) was computed for each song to be recommended over all playlists using a formula

$$idf(s) = log_{10}\left(\frac{|P|}{|P_s|}\right) \tag{3}$$

where P is the set of all playlists and P_s is the set of all playlists containing song s.
The final sequence of recommendations was created based on a new score which weighs recommendations for individual seed tracks with the IDF of these recommended songs. The new score of song s is given by

$$NewScore(s) = S_C(s, s_t) * IDF(s) \tag{4}$$

where s_t represents the seed track that recommendation of s was based on, S_C is the cosine similarity between a given song and the seed track and IDF is an inverse document frequency of song s. Since the new scores are based on a cosine similarity measure between two songs with IDF acting as a weight to control the popularity bias, this implies that the higher the new score the more similar the two tracks are. Taking this into the account, individual song recommendations for every seed track have been gathered and ordered by their new score in a decreasing order in order to obtain the final sequence of recommendations.

3.1.3 Apriori Frequency Sets Initial form of recommendation refers to global correlations among artists and tracks. By using frequency sets, it is determined how likely for a set of artists or tracks to appear together by counting the co-occurrences of the items at each playlist. Therefore the Support and Confidence values are decided for a given set. Consider the example of The Beatles and Queen, support is defined at 5 which indicates the number of Beatles and Queen songs are in a playlist together divided by total number of songs of playlist summed over all playlists. Which captures the ratio of total co-occurrence of both artists over all playlists that has at least one of the artists.

$$s(TheBeatles\&Queen) = \frac{\sum_{p \in P} |p(TheBeatles \wedge Queen)|}{|P|} \quad (5)$$

The second measure is confidence which defined at 6 captures the relation of how likely Queen to be added to a playlist when there's a Beatles in the playlist.

$$c(TheBeatles \Rightarrow Queen) = \frac{\sum_{p \in P} |p(TheBeatles \wedge Queen)|}{|P(Beatles)|} \quad (6)$$

These measures are used to evaluate the likelihood of a collection of items to be in the same playlist together. Therefore it acts as a post-filtering approach to the recommendations of aforementioned collaborative filtering strategies to recommend only the most relevant items which evaluates based on.

$$Output := max(\frac{s(TheBeatles)}{\lambda + \epsilon}, s(TheBeatles\&Queen)) \quad (7)$$

For each item that has been recommended, where λ is the diversity parameter where it adjusts how strict the relevance should be among the items, greater it is more items will pass through. And ϵ is a regularization parameter with a small value $\approx 1e-4$. The hybridization is made in pipeline design defined by [7] where algorithm takes in the output artists/songs as a list format and follows the procedure at Appendix 8.3.

3.2 Group Recommendation using Additive Aggregation Strategy

For group recommendations a group of size s is taken and the ratings of all users in the group are summed up to generate a list with the HRA. Formula 1 is again used with the whole group being the target user to find out how many SPA to suggest. The recommended songs are taken from the whole database. In this case, there is no problem taking a song that appears in a user's individual playlists since the user is part of the group.

The additive strategy is a reasonable way to aggregate the user's preferences as it looks at what the majority of people would like to listen to. However, a limitation is that if an individual user has listened to a particular artist too many times their rating would be very high and will go higher in the list. Consequently, if many users have listened to the same artist but not that many times, that artist will not have high ratings and will be lower in the recommendation list hence get less songs even though a lot of people in the group like them.

Due to the sparsity of the matrix many aggregations strategies would not work. To prove this, the Multiplicative Strategy is implemented. Indeed, it only looks at common artists and there are cases when there is only one artist in common in which case all the recommended songs have to be from this artist. Hence, this strategy is not further explored in this paper.

3.3 Explanations

To provide the user with knowledge as to why a certain song, playlist or artist is suggested to them, multiple explanations are provided with respect to the different approaches this paper implements. When giving an explanation, it is aimed to provide four purposes defined by [8] with one additional purpose, namely:

(1) **Transparency**: Inner workings of the system are explained.
(2) **Effectiveness**: Helps users make "good" decisions.
(3) **Persuasiveness**: Explanation convinces user to follow the recommendation.
(4) **Interpretability**: Explanations are clear and users have no difficulty to understand.
(5) **Efficiency**: Improves decision making process and helps to make decisions faster.

3.3.1 Individual Explanations As individual explanations, the outline is as follows:

(1) Generic recommendation: ***<Song/Artist/Playlist>*** *has been recommended to you because we think you'll like it.*
(2) Song recommendation: ***<Song>*** *has been recommended to you because it is from the* ***<Artist>*** *that has been listened the most by people with similar taste in playlists.*
(3) Playlist recommendation: ***<Playlist>*** *has been recommended to you because it is from the artists that have been listened the most by people with similar taste in playlists.*
(4) Artist recommendation using Apriori post-filtering: ***<Artist which increase Support or Confidence>*** *has been recommended because* ***<Artist>*** *goes well with* ***<Favorite Artist>****, whom you like, have been played together in other playlists with* ***<high/medium/low>*** *support and* ***<high/medium/low>*** *confidence.*

3.3.2 Group Explanations The group explanations are more focused on why a particular song or playlist is recommended with respect to the whole group. An example of the structure is:

(1)(a) Song recommendation: ***<Song>*** *has been recommended to the group because it is from the* **<Artist>** *that has been listened the most by the group.*
(b) Song recommendation: ***<Song>*** *has been recommended to the group because* ***<Group Member 1>****and* ***<Group Member 2>*** *really likes* ***<Artist>****.*
(2)(a) Playlist recommendation: ***<Playlist>*** *has been recommended to you because it is from the artists that have been listened the most by the group.*
(b) Playlist recommendation: ***<Playlist>*** *has been recommended to the group because* ***<Group Member 1>****and* ***<Group Member 2>*** *has their favorite artist* ***<Artist>*** *in it.*

4 Experiments

In this section, experiments for user-based CF with adjusted KNN are introduced. Additionally, experiments with different group sizes are implemented. The experiments that are performed for explanations are also stated. The section also covers the experiments that were attempted using the item-based CF and the difficulties that were faced during this process due to data sparsity.

In this paper, there are two types of parameters: variables and constants, which can be seen in Table 2. All parameters are constrained due to the time and speed concerns and page limit.

The variables are important because they produce the most crucial changes. For adjusted KNN 5 and 10 neighbours are taken and how that changes the results is shown. Additionally, different group sizes (2, 4, 6, 10 - 5 groups for every group size) are set to see how the number of users in a group produces various results. An important note is that groups of the same size do not overlap while groups of different sizes overlap. For instance, if the group is 2.1 that means this is the first group of two people. Groups 2.1, 2.2, 2.3, 2.4, 2.5 do not overlap while the group 4.1 is made up of groups 2.1 and 2.2. This is done in order to see how combining different users in the same group changes the recommendations.

4.1 User-based CF with adjusted KNN

To see how the recommendations change, two different variants of User-based CF with adjusted KNN are tested. The first variant is using 5 neighbours, while the second variant is using 10 neighbours. That way, it could be seen how the diversity of recommendations would change when examining a bigger group of similar artists to the tager user.

4.2 Item-based CF

The planned experiment for the item-based CF was to observe how changing the number of most similar songs to a seed track k and the number of seed tracks *num_seed* affected the obtained recommendations. However, due to playlists not having many songs in common, the resulting playlist-song matrix turned out to be a large binary sparse matrix. This has caused great issues with the application of kNN algorithm. Increasing k and *num_seed*, to relatively small values, still caused the algorithm to either run very slowly or killed the program.

To deal with the sparsity, the following approach was taken. The playlists were filtered out based on their length. By keeping only long playlists, the probability of two playlists overlapping would be increased. In order to optimize the memory storage and reduce the running time, the data was stored in Scipy's a compressed sparse row matrix format. While this approach helped, it didn't manage to solve the underlying issue. Therefore, the planned experiments weren't conducted for the item-based CF algorithm.

4.3 Group Recommendations

For each Aggregation Strategy, we choose to show the 15 highest recommended artists and the corresponding 15 recommended songs. The recommendations of artists and song distribution among the suggested artists for all groups of 2 users are compared to those of groups with 4, 6 and 10 users to see how the suggested artists change, hence how the distribution of songs change.

4.4 Explanation Evaluation

To investigate the effects of different explanations favoring various aspects, a user survey has been conducted which could be accessed through the link in appendix. Users have been presented with hypothetical scenarios which incorporate several contextual dimensions such as location, time of the day, activity, group composition (where applicable) and these dimensions are shifted throughout the narrated scenarios in order to evaluate contextual correlations to user experience of explanations. Thereafter two scenarios for individual setting and two scenarios for group setting have been prepared. For each scenario a mock recommendation is generated and two explanations for each recommendation. Users are asked to evaluate the proposed explanations for aforementioned five purposes on a Likert scale, after evaluating each explanation users filled out a multiple choice question on which explanation they prefer given the proposed scenario and a section for textual commentary is reserved.

5 Results

In this section the results for user-based CF adjusted KNN are shown with 5 and 10 neighbours. Additionally, results for group recommendations are also presented. The results are summed up in Tables 3,4 and 5. The results from the evaluation of explanations are also mentioned.

5.1 User-based CF with adjusted KNN

All the results can be seen in Table 3. It can be seen that Luke Bryan has a rating of 32 with 5 neighbours and is in the first place of the list. On the other hand, in the list with 10 neighbours he has a rating of 56 but he reaches a second position since Kenny Chesney achieves a higher rating. In both variants the ratings of the first artist are not a lot higher than the ratings of the rest of the recommended artists. On the other hand, 5-NN ranks Three Days Grace a lot higher than 10-NN.

From further observations of Table 3, difference is seen between the target user ratings (TUR) and the ratings produced by the adjusted KNN algorithm. In the case of 5 neighbours, the results show a relatively good matching between TUR and the rating of the recommended artists that do exist in the target user's list. Additionally, 50% of artists do not exist in the target user's playlist.

The results with 10 neighbours are similar. The amount of unseen artists is a bit less, but still notable. There is a variation in the order of some of the artists that appear in the 5 neighbours case as well as presence of others.

5.2 Group Recommendations

The recommended artists for all 5 groups of two people are examined together with how the songs are distributed between those artists. These suggestions are compared to the ones given for a group with 4, 6 and 10 users. All results can be seen at Appendix **??** Tables 4 and 5.

The Additive Strategy for group 2.1 shows that Mobb Deep is listened by the group 109 times and is the highest rated artist and therefore has the most songs in the created playlist, in this case 4 songs out of 15. He should have more songs but because Formula 1 is applied with not letting an artist have more than 25% of the ratings the song recommendation is cut to only 4 out of 15 songs. Drake is lower in the list of recommended artists but still takes 1 song while In group 2.2 he is the highest rated artist but since he has a rating of only 20 he takes 2 songs of the playlist. When looking at group 4.1 (which is groups 2.1 and 2.2 combined) it can be seen that Mobb Deep is still the most popular among all but now Drake is in a second place with an increased rating of 33 so instead of taking only 1 song as in group 2.1 he gets 2 songs. In group 6.1 Pitbull goes higher in the ratings than he is in group 2.1 because there are more people who listened to him. In group 10.1 Pitbull gets an even higher spot hence a lot of people in the group listened to him.

5.3 Explanation Evaluation

In total 30 users responded to the survey. The questionnaire provides 4 different scenarios with 2 variants of explanations. The first variant is a mock template while the second variant is the recommendation provided by the system. The idea is to compare both and see if explanations proposed in this paper implement the five key measures for explanations - transparency, effectiveness, persuasiveness, interpretability and efficiency. The questionnaire implements a Likert scale and the subjects have 5 options - Strongly Agree, Agree, Neutral, Strongly Disagree and Disagree. The subjects are asked to determine which of the two variants of explanations they prefer.

Individual Explanations with adjusted KNN - the first scenario represents the individual user-based recommendations. When observing the mock, it's noted that the users have different opinions for all the key points. However, they believe simpler explanation saves them time. On the other hand, the suggested recommendation from the system receives a significantly higher agreement based on a Likert scale from the users overall compared to baseline. The outcome of the two recommendations shows that people prefer the second explanation with 53.3% while for the first one is only 20%.

Individual Explanations with Apriori - in the case of Apriori-based recommendation explanations the mock example shows a higher disagreement rate across people. In contrast, the suggested explanation from the system receives higher agreement score which dominates across all proposed purpose metrics. To show the consistency, users have preferred the provided recommendation from system with 53.3% whereas the mock one receives only 20%. The intention here aligns with findings of [9] where users prefer more descriptive and detailed explanation when they are presented with a situation that they were not expecting.

Group Explanations for a playlist recommendation - the template explanation of the playlist justifies the decision, saves time and is clear. The explanation convinces about half the users are to listen to the provided playlist and helped them make a better decision. The template seems to be more preferred by the users which is confirmed by the pie chart with 13.4% more than system's explanation while in the Likert scale both explanations appear to be favoured by the subjects. The point that stands out is in the scenario users are presented with a song that they did not want in a loosely-coupled homogeneous group and overall more precise justification is preferred.

Group Explanations for a song recommendation - both variants of explanations are agreed upon. In the case of Tchaikovsky it's observed that the explanations have a quite similar rating from the users and the results show that when they need to pick which variant they prefer the difference is only 3.3% favoring general explanation rather than explicit thightly-coupled group members preferences (e.g. most by the group over your spouse and your family relative) which is slightly off the findings of [10] where specificity of information did not make a significant difference in explanation preference with slight inclination towards less informative explanation.

6 Discussion

In this section the reasons for the results are explained for user-based CF with adjusted KNN, for group recommendations and for explanations.

6.1 Individual Recommendation

6.1.1 User-based CF with adjusted KNN The amount of neighbours with adjusted KNN definitely influences the list of recommended artists as it adds more users and they have other artists. However, since the neighbours listen to similar artists it is possible that artists get even higher ratings and go higher in the list. The opposite is also possible, artists could go lower in the rankings as other artists appear. In both cases that does not change much the song distribution for each artist since Formula 1 ensures getting at most 25% of the playlist in order to enhance the variety of the suggested playlist.

There is around 50% of unseen artists from the target person. It is important to note that although those artists have not been listened by the user, his neighbours have very high ratings for the same, meaning that there is chance he will also like them. Thus, the different suggestions of the user's lower and higher rated artists together with the unseen ones provide a good diversity to his recommendations. Even though some bias is observed in the recommendations towards more popular artists, Formula 1 compensates for that and makes the suggested playlist more balanced.

6.2 Group Recommendations

Combining the groups significantly changes the order of the recommended artists. However, the artists do not necessarily take a lot more songs (sometimes they even take the same amount) but that is due to having set a threshold when generating suggested songs. This ensures some diversity in the playlists. Otherwise, having high differences in ratings between the first recommended artist and the rest would make the playlist full of songs from the highest recommended artists and only 1 or 2 songs from the rest of the artists. Hence, the threshold indeed ensures less dominance of the first artist.

6.3 Explanation Evaluation

From the observations in Section 5.4 it can be observed that the majority of people prefer to have a longer but more descriptive explanation as to why a certain song or a playlist is recommended rather than a short explanation with no reasons why a recommendation is picked. Nevertheless, this is not true when too much technical details are provided. Subjects tend to be very confused what this terminology means. To further illustrate this, a comment from a user is given: "Beyoncé does not go well with Logic lmao Also the phrase "playlists with medium support and high confidence" is unclear. I don't know what that means." When it comes to groups, people generally prefer to have more information about the group such as who really likes the recommended artist. A comment that allows to gain more insight into this is: "Personal explanation, with actual names, is more persuasive than 'the group' somehow.".

7 Conclusion & Future Work

In this paper, User-Based CF with adjusted KNN was implemented for individual recommendations of a song or a playlist using Formula 1 with a threshold. Experiments with a different number of neighbours are performed to see how they affect the recommendation lists. Furthermore, Frequency Sets using Apriori approach are used as a post-filter to produce more diverse suggestions.
Another implemented approach NN Item-based CF method is able to recommend a sequence of songs to be added to an individual user's playlist with IDF weights controlling the popularity bias. However, data sparsity caused significant running time issues and prevented the planned experiments.
Group recommendation algorithm with an additive aggregation strategy is also implemented. While increasing the size of the group tends to alter the order of the suggested artists, the imposed threshold ensures a well-balanced list of recommendations.
Both individual and group explanations are successfully implemented. Several different types of explanations are evaluated on how well they fulfilled the different main purposes through an online user survey and by performing an analysis of the results. The study concluded that the users preferred a longer explanation which provides rationale behind the recommendation as long as the technical jargon is kept to the minimum. In a group setting, the users appreciated the additional details preferring to see whose preferences the recommendation is based on.

With the aim of improving the results, the future planned work includes:
Hybrid Algorithms and Contextual Extensions - Merging the Frequency Sets (Apriori) with both CF approaches. This would enable switching between the outputs depending on the context and enable a more controlled song extraction.
Data and parameters - Exploration of possibilities with additional data like the number of followers or albums.
Group Strategies - Enhance the quality of recommendations by using plurality or approval voting strategy. These methods consider how many people in the group like a certain artist.
Algorithm for optimal amount of neighbours - Extension of an adjusted KNN by application of an algorithm for automatic detection of the optimal number of neighbours.
Online Testing - Experiments are currently conducted on the synthetic data assumed playlist relationships due to the lack of user history and user-item information. Possible improvement woult be to mine association rules among playlists to generate *higher-quality* synthetic user data or better test with actual users in real-world settings over the whole dataset.

References

[1] M. Ludewig, I. Kamehkhosh, N. Landia, and D. Jannach, "Effective nearest-neighbor music recommendations," *Proceedings of the ACM Recommender Systems Challenge 2018 on - RecSys Challenge 18*, 2018.
[2] L. Zhu, B. He, M. Ji, C. Ju, and Y. Chen, "Automatic music playlist continuation via neighbor-based collaborative filtering and discriminative reweighting/reranking," *Proceedings of the ACM Recommender Systems Challenge 2018 on - RecSys Challenge 18*, 2018.
[3] D. M. Kelen, D. Berecz, F. Beres, and A. A. Benczur, "Efficient k-nn for playlist continuation," *Proceedings of the ACM Recommender Systems Challenge 2018 on - RecSys Challenge 18*, 2018.
[4] G. Faggioli, M. Polato, and F. Aiolli, "Efficient similarity based methods for the playlist continuation task," *Proceedings of the ACM Recommender Systems Challenge 2018 on - RecSys Challenge 18*, 2018.
[5] H. Zamani, M. Schedl, P. Lamere, and C. Chen, "An analysis of approaches taken in the acm recsys challenge 2018 for automatic music playlist continuation," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 5, p. 1–21, 2019.
[6] H. Abdollahpouri, "Popularity bias in ranking and recommendation," *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, 2019.
[7] R. Burke, "Hybrid recommender systems: Survey and experiments," *User Modeling and User-Adapted Interaction*, vol. 12, 11 2002.
[8] N. Tintarev and J. Masthoff, "A survey of explanations in recommender systems," *2007 IEEE 23rd International Conference on Data Engineering Workshop*, pp. 801–810, 2007.
[9] S. Najafian and N. Tintarev, "Generating consensus explanations for group recommendations: An exploratory study," in *UMAP '18 Adjunct Publication of the 26th Conference on User Modeling, Adaptation and Personalization*, pp. 245–250, Association for Computer Machinery, 2018. UMAP Late breaking Results Accepted author manuscript; UMAP 2018 : The 26th Conference on User Modeling, Adaptation and Personalization , UMAP '18 ; Conference date: 08-07-2018 Through 11-07-2018.
[10] S. Najafian, "Generating natural language explanations for group recommendations in high divergence scenarios," pp. 31–32, 03 2020.

8 Appendix

8.1 Evaluating Explanations

Links to Evaluation:
User Survey: https://forms.gle/XdLpjv2ffPA4jHnQ7
Results of Survey: https://tinyurl.com/therecommenders

8.2 Data

Property	Dataset	Subset
Number of Playlists	1,000,000	6000
Number of Unique Tracks	2,262,292	121,855
Track Uniqueness	0.34	0.31
Number of Albums	734,684	54,868
Number of Artists	295,860	25,127
Number of Unique Titles	92,944	3380
Avg. Playlist Length	66.35	66.89

Table 1: Statistical Properties of Data

8.3 Post-Filtering

```
features [recommended artists];
output := features[1];
for each artist
   check if s(output and artist) > s(output)/lambda
    check if c(output to artist) > s(output)/lambda
     if yes, add artist to output
     if no, move on to next artist
end
```

Group	Artist	Rating	SPA
2.1	Mobb Deep	109	4
	Logic	26	2
	Luke Bryan	20	1
	Wiz Khalifa	18	1
	U2	17	1
	Coldplay	16	1
	SOJA	16	1
	Zac Brown Band	16	1
	Ariana Grande	15	1
	The Green	14	1
	Johnnyswim	14	1
	Drake	13	1
	Death Cab for Cutie	12	1
	Pitbull	11	1
	Kanye West	10	1
2.2	Drake	20	2
	Gloria Trevi	17	2
	Nicki Minaj	14	2
	Kane Brown	12	2
	Big Sean	11	2
	JAY Z	11	2
	Fetty Wap	10	1
	J. Cole	10	1
	Paulina Rubio	9	1
	Florida Georgia Line	9	1
	Luke Bryan	9	1
	David Guetta	9	1
	Lil Wayne	8	1
	Social Club Mistfits	8	1
	Jhene Aiko	7	1
2.3	Electric Light Orchestra	14	2
	Billy Joel	13	2
	Flume	12	2
	Skrillex	10	2
	Queen	9	2
	Michael Jackson	9	2
	Marshmello	8	1
	Alan Walker	8	1
	Fall Out Boy	7	1
	Asap Rocky	7	1
	Bob Marley & The Wailers	6	1
	JJ Lin	6	1
	The Weekend	6	1
	Daryl Hall & John Oates	6	1
	Common Kings	6	1
2.4	Kid Cudi	20	2
	Zumba Fitness	17	2
	Zomboy	15	2
	Daft Punk	13	2
	Chris Brown	13	2
	Pitbull	10	1
	Eminem	9	1
	Kanye West	9	1
	The Black Eyed Peas	8	1
	Drake	8	1
	P!nk & The Wailers	7	1
	Beyonce	7	1
	Big Sean	7	1
	Young Money & John Oates	7	1
	Justin Timberlake	7	1
2.5	Drake	14	2
	Dex Osama	14	2
	Bonobo	11	2
	Michael Jackson	9	2
	Matchbox Twenty	9	2
	Nicky Jam	9	2
	Ty Dolla Sign	8	1
	Counting Crows	8	1
	BandGang	8	1
	Future	7	1
	Don Omar	7	1
	Nelly	7	1
	Rae Sremmurd	7	1
	J Balvin	7	1
	Fleetwood Mac	7	1

Table 4: Results of group recommendations using the Additive Strategy of group with size 2

Group	Artist	Rating	SPA
4.1	Mobb Deep	109	4
	Drake	33	2
	Luke Bryan	29	2
	Logic	26	1
	Wiz Khalifa	24	1
	U2	19	1
	Pitbull	18	1
	David Guetta	17	1
	JAY Z	17	1
	Coldplay	17	1
	Zac Brown Band	17	1
	Gloria Trevi	17	1
	Ariana Grande	16	1
	SOJA	16	1
	Johnnyswim	15	1
6.1	Mobb Deep	109	4
	Drake	33	2
	Luke Bryan	29	2
	Wiz Khalifa	26	2
	Logic	24	1
	Pitbull	19	1
	Coldplay	18	1
	U2	17	1
	David Guetta	17	1
	Kanye West	17	1
	Nicki Minaj	17	1
	Zac Brown Band	17	1
	Gloria Trevi	16	1
	Ariana Grande	16	1
	The Weeknd	15	1
10.1	Mobb Deep	109	4
	Drake	59	2
	Wiz Khalifa	34	1
	Pitbull	33	1
	Luke Bryan	32	1
	Logic	30	1
	Kanye West	30	1
	Coldplay	28	1
	Big Sean	28	1
	Chris Brown	27	1
	JAY Z	25	1
	Ariana Grande	24	1
	U2	24	1
	Rihanna	23	1
	Kid Cudi	23	1

Table 5: Results of group recommendations using the Additive Strategy for the combined groups of size 2

8.4 Experiment Results

	Abbrv[2]	Meaning	Size
Constants	h	number of highest-rated artists	10
	u	number of users	50
	g	number of groups per size	5
	ppu	number of playlists per user	10
Variables	s	group size	2, 4, 6, 10
	n	number of neighbours	5, 10

Table 2: The parameters used in the paper

Neighbours	Artist	Rating	TUR [3]	SPA [4]
5	Luke Bryan	32	19	2
	Kenny Chesney	27	4	2
	Papa Roach	23	NA [5]	2
	Three Days Grace	22	NA	2
	Bon Iver	20	NA	2
	Jesus Adrian Romero	18	NA	1
	Beyonce	17	NA	1
	Halsey	16	NA	1
	Slipknot	16	NA	1
	Keith Urban	15	4	1
	Rascal Flatts	15	5	1
	Sam Hunt	15	6	1
	Florida Georgia Line	15	5	1
	Jessie James Decker	14	NA	1
	blink-182	14	1	1
10	Kenny Chesney	63	4	2
	Luke Bryan	56	19	2
	Drake	53	NA	2
	Keith Urban	43	4	2
	Shawn Mendes	41	1	2
	Rihanna	33	2	1
	The Weekend	31	NA	1
	Wiz Khalifa	31	1	1
	Ed Sheeran	30	NA	1
	Nicki Minaj	30	NA	1
	Eminem	29	4	1
	Florida Georgia Line	28	5	1
	Kanye West	28	6	1
	J. Cole	28	NA	1
	Three Days Grace	26	NA	1

Table 3: Results of User-based CF with adjusted KNN

[2] Abbreviation
[3] Target User Ratings
[4] Songs per Artist
[5] The target user has not listened to this artist before, hence the rating is 0