EXPLORE - Explainable Song Recommendation

Abhinav Arun, Mehul Soni, Palash Choudhary, Saksham Arora

1 INTRODUCTION

We are conducting our project on building a music recommendation product with key focus on new song discovery and explainability. The system is an attempt to generate a playlist by suggesting songs heard by users having similar preferences. The product will take Spotify user-id of the user as an input and will generate a music recommendation playlist. The UI will also have toggles to select a mood to update the recommended playlist and will also have an explanation window for why a song was recommended. We plan to use a hybrid approach- a combination of collaborative filtering and content-based filtering- to develop our recommendation system. The user data will be procured from Spotify for Developers' Web API and the user-song data is taken from subset of MLHD[6] dataset. The Spotify API provides access to multi-level data such as user's playlist data, song's attribute data, artist data etc and the MLHD data has user and song interaction data.

2 PROBLEM DEFINITION

There are state of the art song recommendation systems available such as Spotify, Apple Music, Amazon Music etc. but they lack in 2 key areas: 1) They fail to provide any explainability on why a particular song was recommended. 2) These recommendation system do not provide users with tools to control the recommendations . Through our product, we aim to address these challenges.

3 LITERATURE REVIEW

Several research papers have been published over the years on recommendation systems, and visualization in the context of song recommendation. A few of them are discussed below.

3.1 Single User Recommendation

Recommendation systems are widely used applications of machine learning. They provide suggestions for items that may be useful to the customer and help users avoid selection paralysis. As suggested in [13] and [2], recommendation systems majorly use two approaches: i) Collaborative filtering i.e. assumption that people with similar preferences in past are likely to prefer same items in the future, and ii) content-based filtering i.e. recommendation based on the item's features.

Although novel techniques for recommendation have been used in the domain of Deep Learning [9], [7], they lack in explainability and in performance with respect to sparse matrices. Recommendations can be made explainable by using a variant of matrix factorization algorithm (AMCF -> Attentive Multitask Collaborative filtering)[1] that accounts for explainable recommendations [4]. The model will learn to map the difficult-to-interpret latent features onto the space of interpretable input features space using an attention mechanism as used in [15]. Sparse autoencoders [3] can also be used with a similar setup to provide better personalized recommendations. We plan to perform user studies for evaluation as conducted in [16].

3.2 Group Recommendation*

A key aspect is to identify how to make a recommendation for the group. [13] suggests two methodologies: i) developing recommendations for individuals and aggregating it for a group, and ii) aggregating individual preferences into a group and using this group to make recommendations. Combining individual ratings of users we can aggregate them using different metrics such as Plurality Voting, Average, Least Misery, Most Pleasure and feed into the single user recommendation model.

Designing explanations for group recommendations is another challenge[5] This paper, in the case of Collaborative Filtering talks about providing explanations based on based on the ratings of the nearest neighbors. Simplistic strategies have also been suggested for content-based recommendation systems.

Few considerations while designing group recommendation systems is handling fairness, this paper[5] talks about how including at least x items for each user can be one such way. Building Consensus is another nuance. It can be measured by checking the amount of disagreement between ratings of users.

3.3 Visualization

We surveyed papers to identify effective method of user engagement through interaction. [8] inspected the relationship between musicology and visualization. It's extensive task based analysis on previous existing visualizations provided insight towards our proposed visual interface. [14] discussed multiple algorithms to visualize network graphs such as forced directed network graph, network matrix etc.

Second, we look at the [10], which attempted to tackle the problem of music exploration. A subsequent work [11] implemented a comparative study on the visualization experience of the previous paper and their own; the visualizations with user control helped justify recommendations. Therefore, we plan to incorporate mood control similar to [11] Additionally, giving the users control over recommendations can be incorporated by providing mood based toggles[12].

4 METHODOLOGY

The design of the product is discussed in below sections:

4.1 Data Preparation

We are using MLHD data set for building our recommendation system. The dataset has user-song interaction information of 992 users for 1083471 songs. Although the data set has records from 2006-2013, it is sparsely populated from 2009-2013 and thus we will only use 2006-2008 data for training our model. This data contains time based user-song interaction information. We will utilize this information to create a user-song interaction rating matrix. Rating is a score in range 1-5 that suggests how much a user likes the song. We developed this rating by calculating monthly song frequency and inverse song frequency. Monthly song frequency(TF) is defined as the number of times a user has listened to a song in a month and inverse song frequency(IDF) is defined as a function of the number of users who have listened to that song in that particular month. So overall, the rating for that month is given as:

$$\mathbf{Rating} = \mathbf{TF_{Mt}} * \mathbf{IDF_{Mt}}$$

where,

$$\mathbf{IDF_{Mt}} = \log(\frac{\mathbf{N}}{\mathbf{1 + df}})$$

N : Total Number of users
df : Number of users who have listened to that song in that particular month
This is done so that we capture niche user tastes in our song recommendation and songs that are listened by relatively less users is given higher weightage. Further, to capture temporal relationship, we give relatively higher weightage to recent songs i.e. most recent month is given weight as 1, second recent month is given weight as (23/24) and so on.

Further, we extracted following data-sets from Spotify's Web API: 1) New user's song data, 2) MLHD Song's attributes, and 3) Top 500 songs of 2022 and Top 500 songs of All time playlist's songs and their attributes.

4.2 Recommendation Algorithms

We will be using the following set of algorithms for forking out recommendations to the end users :

Collaborative Filtering
In this technique, we can base our algorithm either on user-user or on song-song similarity. We identify a set of closest neighbors for a given user i (identified through the Pearson correlation coefficient) based on their ratings for common songs. Then, we take the weighted average of the ratings that these neighboring users give to a song j in order to come up with a predicted rating r(i,j) for a given user i for the song j. Also , to account for user bias, we compute the same on deviations of ratings around the mean for a given user.

Matrix Factorization Algorithm
This algorithm uses the concept of singular value decomposition technique to express users and songs in terms of latent factors (k-dimensional latent feature space). We can then use cosine distance to identify the songs with the highest similarity to a given user's liking.

Explainable Matrix Factorization Algorithm
This algorithm is a variant of the above-mentioned matrix factorization algorithm wherein we also provide as input , external metadata in the form of interpretable song features which could be used to help explain recommendations. For each item latent factor j, we train a

predictor

$$f_j \approx i_{ij}$$

in terms of the known interpretable attributes for a song i. Thus, we would get a mapping from the interpretable feature to latent feature set .

This mapping would help us understand which of the interpretable features are driving the individual latent factors.

Now, given that a song (s) was recommended to a user (u) , it means that the feature vectors of the song and user in latent space were similar. Using this, we can identify the latent dimension in which the feature vectors had the highest similarity. Given this latent dimension, we can use the above-computed mapper to get the corresponding interpretable song feature.

Thus, we would be able to back our recommendations in terms of simple interpretable song features. Out of the above algorithms, we went through with Collaborative Filtering. Our tests with matrix factorization were not promising due to the size of our dataset. We were unable to improve on the large computation time needed.

4.3 Data Workflow

The process involved taking the spotify user id as input and getting data output for developing tableau dashboard. It can be divided into the following steps: 1) Taking user Spotify user-id and pulling all their playlists and extracting all songs 2) Mapping user genre affinity. 3) Identify genre representative songs. 4) Creating user song matrix for a new user that is used as an input for collaborative filtering algorithm. 5) Generating personalised playlist using recommendation engine. 6) Generating recommendations from "Top 500 songs of 2022" and "Top 500 songs of All Time" playlists by mapping ranked output to the mentioned playlist using cosine similarity. As user has intrinsic preferences towards a type of song, we exploited this assumption to map recommended songs to out of corpus songs using content based similarity. The data pipeline produces 3 output files that are used as an input for Tableau dashboard.

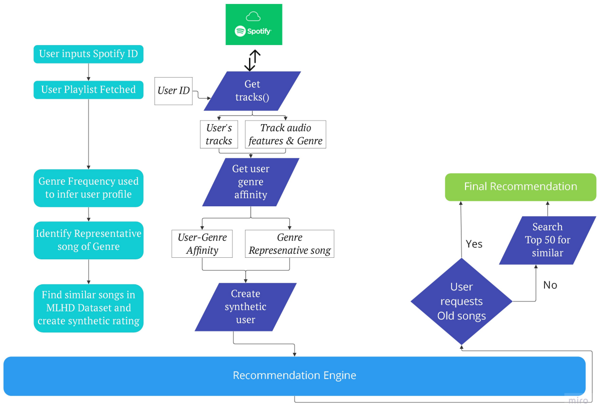
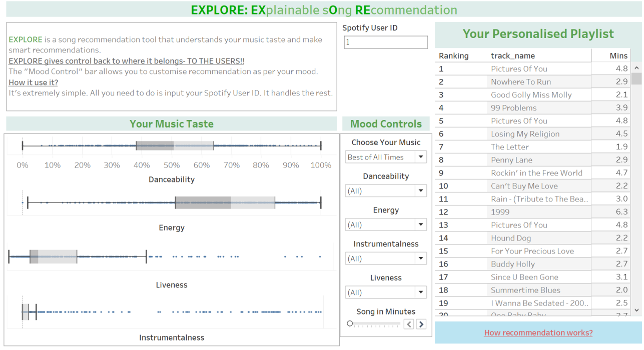


Figure 1: Data Integration Flow Chart

4.4 Visualization and Product Features

The final product is a Tableau dashboard that uses Data workflow output. It shows the personalised recommended playlist and graphical representation of music attributes. Further, it provide tools to control the recommendations and dynamically update the playlist as per user's mood.



EXPLORE: EXplainable sOng REcommendation

EXPLORE is a song recommendation tool that understands your music taste and make smart recommendations.
EXPLORE gives control back to where it belongs- TO THE USERS!!
The "Mood Control" bar allows you to customise recommendation as per your mood.
How it use it?
It's extremely simple. All you need to do is input your Spotify User ID. It handles the rest.

Spotify User ID: 1

Your Personalised Playlist

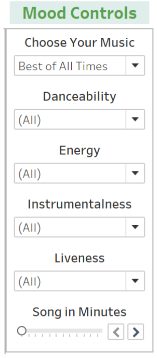
Ranking	track_name	Mins
1	Pictures Of You	4.8
2	Nowhere To Run	2.9
3	Good Golly Miss Molly	2.1
4	99 Problems	3.9
5	Pictures Of You	4.8
6	Losing My Religion	4.5
7	The Letter	1.9
8	Penny Lane	2.9
9	Rockin' in the Free World	4.7
10	Can't Buy Me Love	2.2
11	Rain - (Tribute to The Bea..	3.0
12	1999	6.3
13	Pictures Of You	4.8
14	Hound Dog	2.2
15	For Your Precious Love	2.7
16	Buddy Holly	2.7
17	Since U Been Gone	3.1
18	Summertime Blues	2.0
19	I Wanna Be Sedated - 200..	2.5
20	Ooo Baby Baby	2.7

Mood Controls
Choose Your Music: Best of All Times
Danceability: (All)
Energy: (All)
Instrumentalness: (All)
Liveness: (All)
Song in Minutes

Figure 2: Product User Interface

The user interface has the following sections: 1) Input field to take user's spotify id. 2) Playlist panel that has recommended songs along with their respective ranks 3) Graphical representation of recommended playlist's song attributes. 4) Mood Control bar to control and change the recommendations as per user's mood. 5) "How recommendation works?" button to describe and visualise how recommendations work.

The mood control section has the following toggles: 1) Choose Your Music: Selecting recommendation type



Mood Controls

Choose Your Music: Best of All Times

Danceability: (All)

Energy: (All)

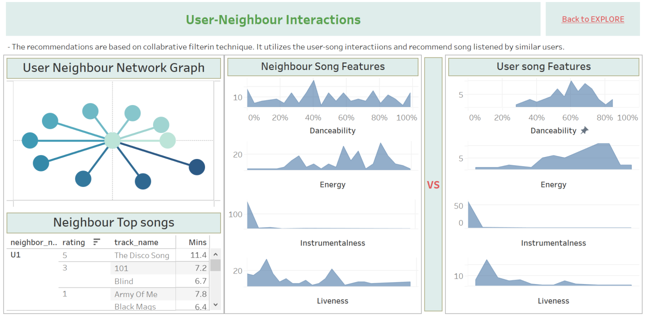
Instrumentalness: (All)

Liveness: (All)

Song in Minutes

Figure 3: Mood Control Bar

from 3 options- Feeling Nostalgic, Best of 2022, and Best of All Times. 2)Danceability 3) Energy 4) Instrumentalness 5) Liveness 6) Song in Minutes: Length of a song in minutes.



- The recommendations are based on collabrative filterin technique. It utilizes the user-song interactiions and recommend song listened by similar users.

neighbor_n..	rating	track_name	Mins
U1	5	The Disco Song	11.4
	3	101	7.2
		Blind	6.7
	1	Army Of Me	7.8
		Black Mags	6.4

Figure 4: User Neighbour Interaction View

The "How recommendation works?" button showcases the explainability aspect of the product. It visually explains the concept of collaborative filtering and visualizes user-neighbour interactions.

4.5 Innovations

Summarizing the above sections, we introduced following innovation through our product : 1) Defining rating matrix using song listening history of users. 2) Explainability aspect to recommendation i.e. explaining the attributes of the recommended playlist. 3) Interactive Mood controls giving users control over their recommendations. 4) Toggle to recommend songs from "Top 500 songs of 2022" and "Top 500 songs of All Time" playlists i.e. out of corpus recommendations. 5) Functionality to recommend songs to new users i.e. Spotify users not part of MLHD dataset 5) Visualizing working of recommendation algorithm using user-neighbour interactions.

5 EVALUATION

We conduct following evaluation process for our recommendation system.

Train-Test Validation: In this step, we evaluate the recommendation system algorithm using train-test validation. We use the train data to develop our model using the above mentioned methodology and use it to predict the user rating for songs in test data. We use RMSE (root mean square error) metric to evaluate the performance.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum\nolimits_{n=1}^{N}(\epsilon_i - \hat{\epsilon})^2}$$

User Validation: We conducted a user study where we asked a few users to share their spotify user-id and our product generated a playlist recommendation. We then asked the user to rate the recommendation as good or bad. We defined the performance on the basis of the accuracy of recommendations.

$$\textbf{Accuracy} = \frac{\textbf{Total Good Recommendations} * 100}{\textbf{Total Recommendation}}$$

The results were as follows:

Ranking Based Evaluations

Although we can evaluate our recommendation systems through the above 2 methods, it is important thing when making out recommendations is the order in which recommendations are made . Thus, we will use the 2 below mentioned metrics for evaluating the output of our recommendation system .

Ranking Evaluation Metrics:

1. Mean Average Precision@K:

This metric measures how many of the recommended results are relevant and are showing at the top. Precision@K is the fraction of relevant items in the top K recommended results. The mean average precision@K measures the average precision@K averaged over all queries in the dataset.

$$AP@k = \frac{1}{r_k}\sum_{k=1}^{k} s_k * rel(k)/k$$

,
where s_k is the number of relevant songs in top k results, rel(k) = 1 if kth songs are relevant and 0 otherwise

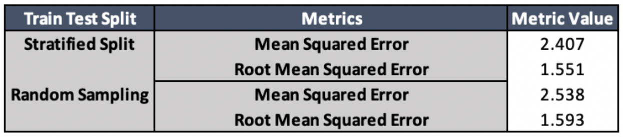
$$\boldsymbol{mAP@k = \frac{1}{N}\sum_{i=1}^{N} AP@k}$$

2. Normalized Discounted Cumulative Gain:
Gain refers to the relevance score for each item (song in this case) recommended. Cumulative gain at K is the sum of gains of the first K items recommended. Discounted cumulative gain (DCG) weighs each relevance score based on its position, with a higher weight to the recommendations at the top.

$$NDCG@K = \frac{DCG@K}{IDCG@K}$$

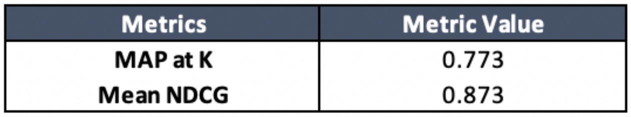
, where
$DCG@K = \sum_{i=1}^{K} \frac{G_i}{log_2(i+1)}$ and
$IDCG@K = \sum_{i=1}^{K_i} \frac{G_i}{log_2(i+1)}$, where K_i is K(ideal) and G_i is G (relevance score of each recommended item).

Results:
We followed two different approaches to sample our dataset into 80%-20% train-test split. The first approach takes a stratified sampling route where for each user the split is done randomly into 80-20 split. The second approach does an overall random shuffling and we obtain the train (80%) and test (20%).



Train Test Split	Metrics	Metric Value
Stratified Split	Mean Squared Error	2.407
	Root Mean Squared Error	1.551
Random Sampling	Mean Squared Error	2.538
	Root Mean Squared Error	1.593

Figure 5: RMSE-MSE Evaluation



Metrics	Metric Value
MAP at K	0.773
Mean NDCG	0.873

Figure 6: Ranking Evaluation

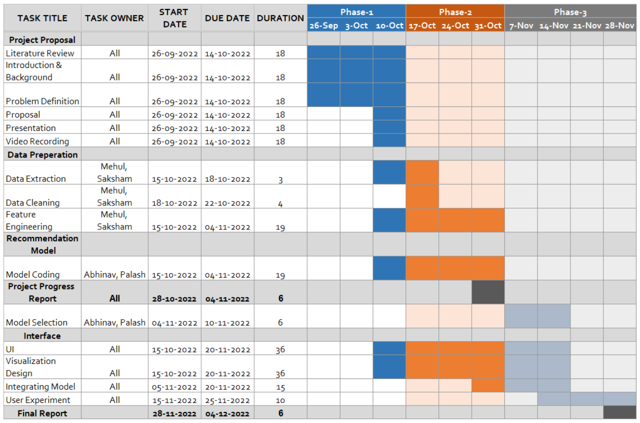
The ranking metrics (MAPK for k = 3 : 0.773, Mean NDCG: 0.873) and the RMSE values show that our collaborative filtering algorithm does a good job in prediction of ratings as well as generating relevant song recommendations.

6 PROJECT TIMELINES

The team members took ownership of the following aspects:

Table 1: Contribution Table

Member Name	Contribution
Abhinav	Recommendation algorithm, Data Integration, Poster, Git Repository Setup, Data Preparation
Mehul	Recommendation Algorithm, Data Preparation, Matrix Factorization, Evaluation, Poster
Palash	Tableau Dashboard and UI, Poster, Report, User Study and Experiments, Data pull, Data mappers
Saksham	Recommendation Algorithm, TabPy Integration, Data Integration, Poster, Report, Data pull, UI

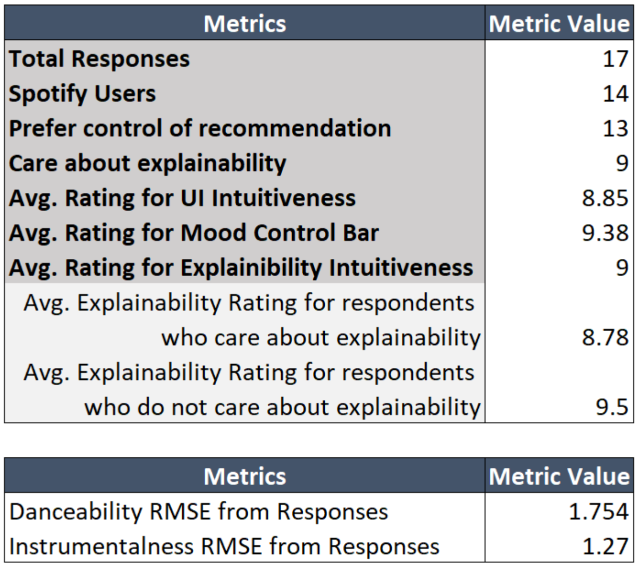


TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION	Phase-1			Phase-2			Phase-3			
					26-Sep	3-Oct	10-Oct	17-Oct	24-Oct	31-Oct	7-Nov	14-Nov	21-Nov	28-Nov
Project Proposal														
Literature Review	All	26-09-2022	14-10-2022	18	🟦	🟦	🟦							
Introduction & Background	All	26-09-2022	14-10-2022	18	🟦	🟦	🟦							
Problem Definition	All	26-09-2022	14-10-2022	18	🟦	🟦	🟦							
Proposal	All	26-09-2022	14-10-2022	18			🟦							
Presentation	All	26-09-2022	14-10-2022	18			🟦							
Video Recording	All	26-09-2022	14-10-2022	18			🟦							
Data Preperation														
Data Extraction	Mehul, Saksham	15-10-2022	18-10-2022	3			🟦	🟧						
Data Cleaning	Mehul, Saksham	18-10-2022	22-10-2022	4				🟧						
Feature Engineering	Mehul, Saksham	15-10-2022	04-11-2022	19			🟦	🟧	🟧	🟧				
Recommendation Model														
Model Coding	Abhinav, Palash	15-10-2022	04-11-2022	19			🟦	🟧	🟧	🟧				
Project Progress Report	**All**	**28-10-2022**	**04-11-2022**	**6**						⬛				
Model Selection	Abhinav, Palash	04-11-2022	10-11-2022	6							▨	▨		
Interface														
UI	All	15-10-2022	20-11-2022	36			🟦	🟧	🟧	🟧	▨	▨		
Visualization Design	All	15-10-2022	20-11-2022	36			🟦	🟧	🟧	🟧	▨	▨		
Integrating Model	All	05-11-2022	20-11-2022	15						🟧	▨	▨		
User Experiment	All	15-11-2022	25-11-2022	10								▨	▨	▨
Final Report		**28-11-2022**	**04-12-2022**	**6**										⬛

Figure 7: Gantt Chart of project timeline

7 EXPERIMENTS

We conducted user survey to seek feedback on our product. We got 17 responses and the overall response was positive. One promising observation has been a strong positive response for "Explainability Visualization" from respondents who did not care about explainability in the first place.



Metrics	Metric Value
Total Responses	17
Spotify Users	14
Prefer control of recommendation	13
Care about explainability	9
Avg. Rating for UI Intuitiveness	8.85
Avg. Rating for Mood Control Bar	9.38
Avg. Rating for Explainibility Intuitiveness	9
Avg. Explainability Rating for respondents who care about explainability	8.78
Avg. Explainability Rating for respondents who do not care about explainability	9.5

Metrics	Metric Value
Danceability RMSE from Responses	1.754
Instrumentalness RMSE from Responses	1.27

Figure 8: User Survey Evaluation

8 CONCLUSION AND FUTURE SCOPE

Through our final product, the team was able to address all the key challenges mentioned in the "Problem Definition" section. Following milestones were achieved: 1) Product to generate personalised playlists. 2) Incorporate explainability aspect to recommendations. 3) Adding tools for users to control and dynamically update recommendations.
However, there are shortcomings which we would like to work in future. 1) The product might face challenges with scalability due to a) dependency on Spotify API to fetch songs and b) recommendation algorithm latency 2) Recommendation is based on old data and on the assumption that music tastes are purely based on song characteristics.

Although we are answering a few existing challenges in state-of-the-art recommendation systems, there is potential to improve on the following aspects in future: 1) Making recommendations for a group. 2) Incorporating user feedback for recommended songs in UI(Like/Dislike button) and using their responses to improve the recommendation.

REFERENCES

[1] Behnoush Abdollahi and Olfa Nasraoui. 2017. Using Explainability for Constrained Matrix Factorization. In *Proceedings of the Eleventh ACM Conference on Recommender Systems* (Como, Italy) *(RecSys '17)*. Association for Computing Machinery, New York, NY, USA, 79–83. https://doi.org/10.1145/3109859.3109913

[2] Adiyansjah, Alexander A S Gunawan, and Derwin Suhartono. 2019. Music Recommender System Based on Genre using Convolutional Recurrent Neural Networks. *Procedia Computer Science* 157 (2019), 99–109. https://doi.org/10.1016/j.procs.2019.08.146 The 4th International Conference on Computer Science and Computational Intelligence (ICCSCI 2019) : Enabling Collaboration to Escalate Impact of Research Results for Society.

[3] Milad Ahmadian, Mahmood Ahmadi, and Sajad Ahmadian. 2022. A reliable deep representation learning to improve trust-aware recommendation systems. *Expert Systems with Applications* 197 (2022), 116697. https://doi.org/10.1016/j.eswa.2022.116697

[4] Anupam Datta, Sophia Kovaleva, Piotr Mardziel, and Shayak Sen. 2017. Latent Factor Interpretations for Collaborative Filtering. *CoRR* abs/1711.10816 (2017). arXiv:1711.10816 http://arxiv.org/abs/1711.10816

[5] Alexander Felfernig, Nava Tintarev, Thi Ngoc Trang Tran, and Martin Stettinger. 2021. Designing Explanations for Group Recommender Systems. *CoRR* abs/2102.12413 (2021). arXiv:2102.12413 https://arxiv.org/abs/2102.12413

[6] ISMIR 2017. *The Music Listening Histories Dataset (MLHD)*. Retrieved Nov 06, 2022 from https://archives.ismir.net/ismir2017/paper/000180.pdf

[7] Mohamed Khoali, Abdelhak Tali, and Yassin Laaziz. 2020. Advanced Recommendation Systems Through Deep Learning. In *Proceedings of the 3rd International Conference on Networking, Information Systems amp; Security* (Marrakech, Morocco) *(NISS2020)*. Association for Computing Machinery, New York, NY, USA, Article 51, 8 pages. https://doi.org/10.1145/3386723.3387870

[8] R. Khulusi, J. Kusnick, C. Meinecke, C. Gillmann, J. Focht, and S. Jänicke. 2020. A Survey on Visualizations for Musical Data. *Computer Graphics Forum* 39, 6 (2020), 82–110. https://doi.org/10.1111/cgf.13905 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13905

[9] Seungyeon Lee and Dohyun Kim. 2022. Deep learning based recommender system using cross convolutional filters. *Information Sciences* 592 (2022), 112–122.

[10] Yu Liang and Martijn C. Willemsen. 2019. Personalized Recommendations for Music Genre Exploration. In *Proceedings of the 27th ACM Conference on User Modeling, Adaptation and Personalization* (Larnaca, Cyprus) *(UMAP '19)*. Association for Computing Machinery, New York, NY, USA, 276–284. https://doi.org/10.1145/3320435.3320455

[11] Yu Liang and Martijn C. Willemsen. 2021. Interactive Music Genre Exploration with Visualization and Mood Control. In *26th International Conference on Intelligent User Interfaces* (College Station, TX, USA) *(IUI '21)*. Association for Computing Machinery, New York, NY, USA, 175–185. https://doi.org/10.1145/3397481.3450700

[12] Millecamp Martijn, Cristina Conati, and Katrien Verbert. 2022. "Knowing Me, Knowing You": Personalized Explanations for a Music Recommender System. *User Modeling and User-Adapted Interaction* 32, 1–2 (apr 2022), 215–252. https://doi.org/10.1007/s11257-021-09304-9

[13] Judith Masthoff. 2011. *Group Recommender Systems: Combining Individual Models*. Springer US, Boston, MA, 677–702. https://doi.org/10.1007/978-0-387-85820-3_21

[14] Michael J. McGuffin. 2012. Simple algorithms for network visualization: A tutorial. *Tsinghua Science and Technology* 17, 4 (2012), 383–398. https://doi.org/10.1109/TST.2012.6297585

[15] Deng Pan, Xiangrui Li, Xin Li, and Dongxiao Zhu. 2021. Explainable Recommendation via Interpretable Feature Mapping and Evaluation of Explainability. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence* (Yokohama, Yokohama, Japan) *(IJCAI'20)*. Article 373, 7 pages.

[16] Yongfeng Zhang, Xu Chen, et al. 2020. Explainable recommendation: A survey and new perspectives. *Foundations and Trends® in Information Retrieval* 14, 1 (2020), 1–101.

APPENDIX

There was an update in the team structure in middle of the semester. One of our team members dropped out of the course on 28th Oct'2022. This change forced us to redefine the scope of the project and hence you might see deviation from initial proposal.

The scope was redefined in the following areas. 1) Considering the drop of a team member and difficulty in evaluation of group recommendation(especially given the change in data set) we limited our scope to single user recommendation. 2) Getting access to users listening history was a challenge as the API does not expose this without explicit user authorization. We only get access to public playlists of the user. Thus we used the Last.fm Data set to base our recommendations. This opened the challenge of recommending songs to a new user which we have addressed in the Data Workflow section.