International Journal of Mathematics and Computational Science
Vol. 1, No. 6, 2015, pp. 352-363
http://www.aiscience.org/journal/ijmcs
ISSN: 2381-7011 (Print); ISSN: 2381-702X (Online)

Music Recommendation Engine

Syam Murali*, Upma Vermani, Catherine Khaw

Enterprise Business Analytics, Institute of Systems Science, National University of Singapore, Singapore, Singapore

Abstract

Recommendation systems used to recommend music are different from those used to recommend products/services. A user is not always interested in a familiar and regular song recommendation which he is already aware of. Instead, users who like to explore new music are looking for diverse and new songs which they would not have heard otherwise. The engine should also ensure that the songs are not too far off from the user's taste that the user does not like it. Therefore, the key is a balanced recommendation. This paper addresses this problem by recommending novel and unfamiliar songs to the users based on their desire to explore music along with familiar recommendations. The recommendation engine was tested on certain parameters through a user survey and it was found to increase user satisfaction, particularly for the users who like exploring music.

Keywords

Recommendation Systems, Unfamiliar Songs, Collaborative Filtering, Recommender Systems, Music Exploration, Novel Songs

Received: November 27, 2015 / Accepted: December 17, 2015 / Published online: December 29, 2015

1. Introduction

Music recommendation and recommender engines have been around since the 1990s and have evolved from recommending songs in online stores to generating instant playlists on streaming sites over the years. Online music stores and streaming sites have gained significant popularity over the past decade.

Significant research has been done in music recommendation by industry players such as Spotify, Pandora, and Amazon. Current methods include recommending songs based on listening history of users and/or songs metadata. Websites typically use techniques such as collaborative filtering to give recommendations to the user. However, many of these methods do not always produce expected results. A lot of factors influence a person's music preferences such as his mood, time of the day, occasions, etc. Hence, it is very difficult to provide desirable recommendations unless these factors are accounted for. Thus, we can see that listening to music and a person's preferences to music are very subjective and this makes recommending music tricky.

Another area of concern when it comes to recommendation systems is the difficulty to define a good engine. Accuracy of predictions works for recommender systems in online stores, which are built to sell similar products; but it is a poor indicator of engine performance when it comes to music. Music by definition is very personal and it is impossible to state that one set of recommendations are more accurate than another.

Current methods of engine accuracy focus on checking recommender predictions against user listening history. This approach has its flaws. Unless a recommendation was presented to the user, it is impossible to know whether the user would have chosen the song.

Finally and most importantly, a recommender system should recommend songs that the user would not have come across otherwise. Recommending songs for bands or genres that the user typically listens to serves this purpose to an extent.

* Corresponding author
E-mail address: Syammurali.mail@gmail.com (S. Murali)

However, for an engine to be a true recommender system it is essential that it produces recommendations that surprise the listener. Unexpected recommendations which the user likes helps the listener discover new and novel music and this in turn increases the user's trust in the system. A user which trusts the recommender system of a site eventually will become a loyal subscriber.

The aim of the research is to create a recommendation engine which incorporates unexpected recommendations. The aim of such an engine would be to enable the user to discover unknown and new music which the user might like, thereby, expanding his/her musical horizons.

2. Research Objective

Current academic research on music recommendation engines focus on measuring engine accuracy.

However, as mentioned previously, listeners are also interested in unexpected song recommendations. The usefulness of an engine depends on how well the engine can generate unexpected recommendations which the user likes. An unexpected recommendation can be a song from an artist or a genre that is not familiar to the user. E.g. Recommending a Judas Priest song to a classical music fan. The success of the engine will lie in the user liking the song that was recommended. These recommendations help broaden the listener's music base and will also increase the trust that the listener has on the engine. In academic papers, such a recommendation would be termed a "serendipitous" recommendation.

The objective of this paper is to develop a recommendation engine which incorporates unexpectedness in its recommendations.

It is also of interest to us to understand if users would prefer such an engine over an engine which bases its recommendations on user history and item history having no special preference given to unexpectedness of music.

3. Existing Research

3.1. Related Work

Recommendation engines are tools to predict a user's preference for his next purchase such as downloading a song, watching a video, selecting a movie, purchasing an item, etc. Large industry players such as Amazon, Google, Spotify, Pandora, Lastfm, Netflix, TiVo, Yahoo, YouTube and Microsoft have been using these engines to provide customised experiences. Collaborative filtering, Content-based filtering and Hybrid models are some of the widely-used techniques for recommendation engines.

Collaborative filtering (CF) is a technique used for making recommendations about the preferences of a user by collecting related information from similar users.

One of the earliest implementation of recommendation engine which used collaborative filtering was Tapestry in 1992. The model used by Goldberg, D., Nichols, D., Oki, B. M., and Terry, D [1] relied on the user information in a closed-knit circle of office employees to make recommendations. The model, however, was not applicable for a large set of people as there would be no interaction among users.

This led to development of rating-based recommendation systems.

Two types of collaborative filtering techniques were user-based and item-based. Amazon.com uses item-to-item collaborative filtering as discussed in the industry report written by Greg Linden, Brent Smith, and Jeremy York [2]. The algorithm identifies items similar to user's purchase and rating history, aggregates it and then recommends similar items or highly co-related items to the user.

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl [3] analysed different item-based recommendation systems and established experimentally how they have a higher accuracy as compared to user-based models. Yehuda Koren, Robert Bell and Chris Volinsky, [4] have discussed matrix factorization techniques used by recommender systems in their industry report. Another paper by Yehuda Koren [5] has discussed collaborative filtering with temporal dynamic by using matrix factorisation. Ding and Li [6] attempted to study the temporal effects by using collaborative filtering with time weighting scheme. Another approach used in collaboration filtering is the hybrid model or a unified model of both nearest neighbour and latent factor models. Yehuda Koren [7] has discussed this approach in his research paper highlights its advantages and limitations. In this model, the implicit feedback is integrated in neighbourhood model.

Michael J. Pazzani and Daniel Billsus [8] reviewed content-based recommendation engines. In a content-based recommendation system, the user history serves as training data for a machine learning algorithm. David Stern, Ralf Herbrich and Thore Graepel [9] talked about online Bayesian recommendations. Beth Logan [10] from HP Labs has discussed music recommendation based on similarity in acoustics from related songs by using K-means model of clustering. In a research paper by Cornell University, Yi Li, Rudhir Gupta, Yoshiyuki Nagasaki and Tianhe Zhang [11], use Million song dataset (MSD) to implement a song recommendation system. The approach used for recommendation was a combination of song-based CF, user-based CF, K-means clustering and a hybrid model combining user-based and item-based methods. It was observed that hybrid model gave the highest accuracy rate compared to KNN, matrix factorisation, K-means and other methods.

Paper by Fabio Aiolli [12] also uses Million Song Dataset. The data used for the recommender system is Taste Profile Subset which consists of users, song and play count. The method used was memory-based collaborative filtering. The research paper by Pablo Castells, Saúl Vargas, and Jun Wang [13] has discussed that novelty and diversity are important factors while giving a music recommendation. This study has covered the already existing novelty and diversity factors mentioned in literature. In addition, two new features have been mentioned in the study - ranking sensitivity and relevance-awareness. The research paper by Yuan Cao Zhang, Diarmuid Ó Séaghdha, Daniele Quercia, Tamas Jambor [14] talks about a new framework for recommender systems which focuses on producing highly personalised recommendations by taking into account accuracy, novelty, diversity and serendipity. The recommender system called Auralist recommender uses a variety of algorithms to define and use a range of metrics to measure the three non-accuracy factors at the same time.

One interesting finding from this model is that diversity, novelty and serendipity can be simultaneously increased without much trade-off between these three factors.

However, this model does not work well with limited user information or a cold-start.

3.2. Existing Engines

Several recommendation engines exist for music with each serving specific purposes. E.g. Amazon's recommendations are based on item based systems. The purpose of such an engine would be drive music sales and is therefore more focused on providing customers with music that they are more likely to buy.

Streaming services like Spotify and Pandora have propriety systems in place to provide listeners with the best listening experience possible. However, little research has been done in the academic field regarding development and testing of engines which are focussed on providing users with varied and unexpected songs and thus contributing to the overall listening experience.

As mentioned previously, research focus has been on improving prediction accuracy. In case of traditional engines in research that are available for evaluation, Fabio Aiolli's hybrid engine has been proved to perform with the highest accuracy (MAP).

Auralist engine is similar in concept to the recommendation engine being considered by this paper. The engine tries to address various issues like diversity, serendipity, etc.

This paper intends to consider the several aspects mentioned in the paper. However, the focus of the paper would be to develop an engine which can generate unexpected recommendations and to measure user satisfaction for the same.

3.3. Engine Gaps

As mentioned above, current research focuses on comparing and evaluating engines on accuracy. Only a few papers have looked into incorporating unexpected recommendations.

Also, existing papers have not considered the profile of users when it comes to choosing one engine over the other.

This papers looks into developing an engine which incorporates unexpected recommendations and tries to understand if listeners prefer such an engine over traditional engines. The paper also explores how other factors like ability to play instruments, affinity to new music, etc. of the user might affect such a preference. This is another aspect which has not been considered extensively by research papers.

4. Analytical Objectives

Based on our understanding of the problem, our analytical objectives are three fold:

1 Identify songs that are unexpected but are useful to the listener

2 Measure and compare usefulness of the new engine with existing engine

3 Measure the influence of factors such as demographics and user's musical interests on user's choice of a recommendation engine.

4.1. Testing

Comparison of usefulness, unexpectedness and likeability of new engine with existing engine was done through a user survey.

4.2. Metrics

4.2.1. Likeability

This is a measure of how much a user liked a song that was recommended. This metric is calculated as the average of the five point rating given to all the songs across all the recommendations for a particular engine.

4.2.2. Unexpectedness

This is a measure of unexpectedness of recommendations. This is an average of the five point rating for unexpectedness given to all the songs across all the recommendations for a particular engine. This metric helps us compare the user's perception of the recommendations by the engine.

4.2.3. Usefulness

This metric is indication of user's perception of the usefulness of a particular recommendation. A user may like a song that was recommended by the engine; however, he may or may not consider that song appropriate or useful as recommendation. This is an average of the two point rating for usefulness or appropriateness of the recommendation of all the songs across all the recommendations for a particular engine.

A higher rating in this regard would indicate a better engine.

4.3. Data

The initial strategy was to utilize the data provided by Million song dataset. Million Song database is the largest music dataset that is available for public use. The dataset contains, in addition to song metadata, user listening history, acoustic features, etc. of a million songs.

The data set has been made available through the collaboration of EchoNest and LabRosa. Several studies have already been conducted on this dataset. The size of the dataset provides the best opportunity from an academic perspective to study music recommendations. It is important to point out that generally most of the data collected on song listening history and user preferences are not publically available, which is why this particular data set is important. The entire data set consists of 300 GB of data.

Statistics

The song set covers tracks from 1922, with most of the songs coming in the latter part of the 20^{th} century.

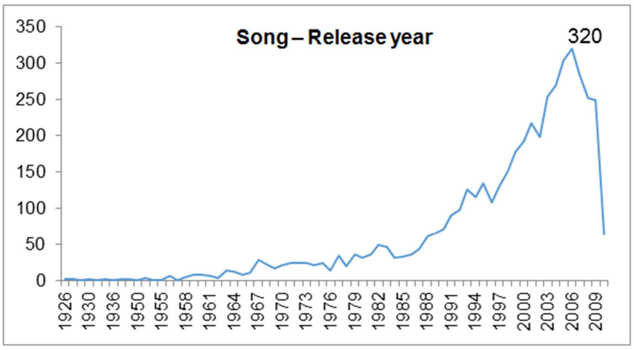


Figure 1. Overview of Million Song Data.

Table 1. Brief Snapshot of data.

No of songs	1,000,000
Dataset size	273 GB
Unique Artists	44,745
Tags (Echo Nest)	7,643
Unique Tags (Others)	2,321
Artists with at least one tag	43,943
Tracks which are dated	515,576

4.4. Data Gap and Challenges

In order to test the new engine, a user survey was conducted. In the survey, users filled more recent songs, most of which were not available in the database as the million song dataset is limited to 2010. Hence, in order to calculate user and song similarity, alternate data sources were required. Instead of using historic data to create user and song similarities, the songs collected from the user survey were used.

5. Modelling

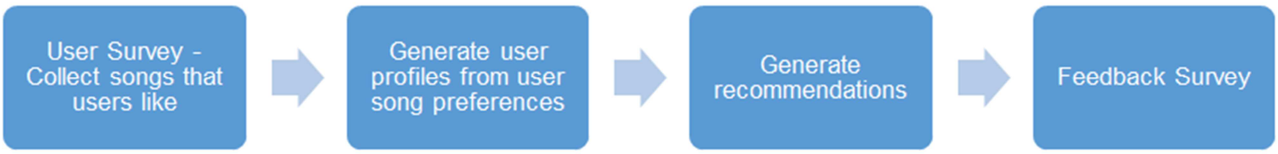


Figure 2. Process Overview.

The songs collected from the user survey 1 (please refer to the survey section, for details) formed the basis of generating a user profile and seeding the engine to generate recommendations based on user and song similarity.

The first step was to find similar users and to calculate user similarity – U.

For instance, consider *Ui to be the set of users* and *Si is the set of song.*

Three users, U_1, U_2 and U_3, listen to the following songs:

$$U_1 - [S_1, S_2, S_3]$$

$$U_2 - [S_2, S_4, S_5]$$

$$U_3 - [S_2, S_4, S_6]$$

Similarity between the users was calculated using the cosine function.

$$U_{1,2} = \frac{U_1 \cap U_2}{{U_1}^{1/2} * {U_2}^{1/2}}$$

The following user similarity matrix was obtained:

Table 2. User similarity matrix.

Users	U_1	U_2	U_3
U_1	1	0.33	0.33
U_2		1	0.67
U_3			1

User similarity scores were used to find similar users. In this case, U2 and U3 were found to have the highest similarity scores. Users were then ordered in descending order based on user similarity.

Similarly, song similarity scores were used for generating similar songs:

For example songs S_1, S_2 and S_3 were heard by the following users:

$$S_1 - [U_1, U_3, U_4]$$

$$S_2 - [U_2, U_3, U_7]$$

$$S_3 - [U_1, U_2, U_4]$$

Similarity between songs was computed using the cosine function:

$$S_{1,2} = \frac{S_1 \cap S_2}{{S_1}^{1/2} * {S_2}^{1/2}}$$

The following similarity matrix was obtained:

Table 3. Song Similarity Matrix.

Songs	S_1	S_2	S_3
S_1	1	0.33	0.67
S_2		1	0.33
S_3			1

The next step was to calculate the overall similarity rating which is a combination of user and song similarities.

Overall similarity rating = U + S

In order to add unexpected recommendations, a dissimilarity rating was created:

Overall dissimilarity rating = U + (1-S)

This was done with the aim to recommend songs which are dissimilar but similar users have listened to it, so that users do not end up getting a completely irrelevant recommendation.

Engines

5.1. Engine A

Engine A does not incorporate unexpected songs. The recommendations were purely based on user and song similarity.

Overall similarity rating = U + S

The top 5 recommendations based on overall similarity ratings were generated by the engine.

5.2. Engine B – Unexpected Songs

Engine B incorporates unexpected recommendations based on overall similarity ratings and overall dissimilarity ratings,

Overall dissimilarity rating = U + (1-S)

The number of dissimilar songs suggested to the user was based on the music exploration factor "N" decided by the user.

Hence, final recommendations of 5 songs were decided by:

(5-N/2) * Songs from set of similar songs + N/2 * Songs from set of dissimilar songs

Where N = how much user likes to explore music (1-5)

The factor N was obtained from the user survey.

6. Survey

The idea behind a user survey was two pronged. Survey provides a real life scenario to measure and compare both the engines. Also, the survey would provide qualitative insights that may be useful to improve the engine in the future. The purpose of the user survey is to compare the new engine with an existing engine which does not incorporate factors of unexpectedness. The survey was sent out in two parts. In the first part, users were asked to fill basic information and 5 songs of their choice in order to generate a basic user profile. Recommendations were generated for each user based on user and song similarity. The users were given recommendations from one of the two engines – A or B, and were then asked to rate the recommendations in a separate survey to compare the two engines.

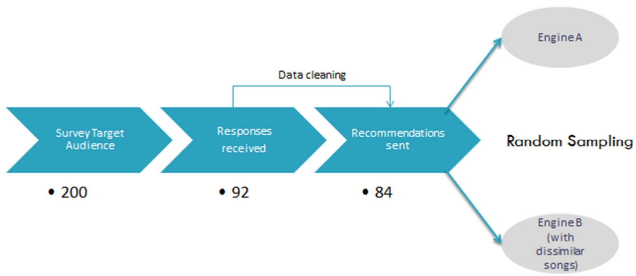


Figure 3. Survey Process.

6.1. Survey Part 1

The survey was sent out to 200 listeners who were primarily college students. Out of these, 92 responses were received. However, some of the responses were incomplete. After data cleaning, recommendations were generated for 84 users and sent out via emails. Random sampling was done to generate recommendations from either engine A or engine B (with dissimilar songs).

Survey questions

The survey questions were categorized into three main groups.

1 Basic listener information

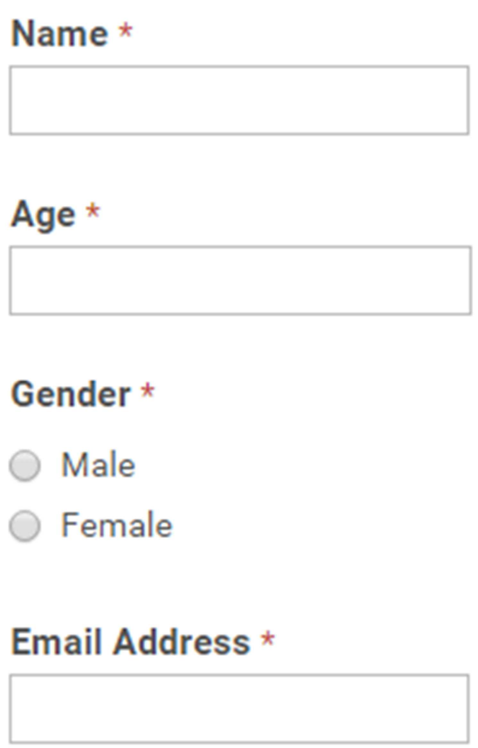
2 Music tastes

3 Questions pertaining user's affinity to new music and musical ability

6.1.1. Basic Listener Information

The survey started with capturing basic user information. These questions assisted in building user profiles and also provide information for further profile analysis.

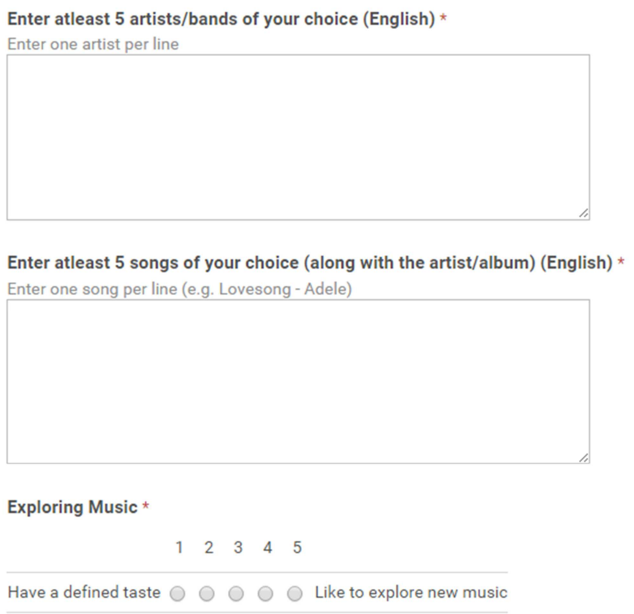
Questions:



6.1.2 Music Tastes

In order to generate good recommendations, it is imperative that the system understands the listener. The listener's user profile was generated with the help of questions pertaining to their music tastes. While the profile may not be entirely accurate due to the lack of extensive listening history, it still provided the engine with a general idea of the user.

Questions:

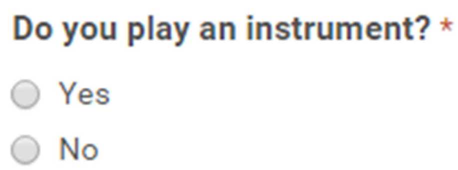


The third question in the section, regarding new types and kinds of music, helped understand the user's acceptance of unexpected music suggestions. This question helped us explore if user's preference engines were influenced by their natural receptiveness to new music.

6.1.3. Ability to Play Instruments

This question was added to address the secondary research objectives of how a person's affinity to music and ability to play instruments affect a person's preference for new music.

Questions:



After calculating user and song similarity for each user, the recommendations were generated by both the new engine B (which incorporates factors of unexpectedness) as well as the existing engine A for 84 users. Users were randomly selected to receive recommendations generated from either engine A or engine B.

6.2. Survey Part 2

The second part of the survey was sent out along with the recommendations generated from either engine A or engine B. The purpose of this survey was to obtain user feedback on the recommendations. One set of 5 recommendations were sent to the user. The user was unaware of whether the recommendations came from the new engine A or the existing engine B.

The users were then encouraged to listen to any new song in the recommendations. Users were asked to rate the recommendations on a variety of parameters.

Questions based on the recommendations

These questions were posed for each song in the list of songs recommended.

- Have you heard this song before? (Yes/ No)

Have you heard these songs before?*

	Yes	No
Twist and Shout	○	○
Comfortably Numb	○	○
Bombay Rain	○	○
Anuva's Sky	○	○
Sweet Child of Mine	○	○

The user was requested to listen to songs which he/she marked as No.

- Please rate the song (Scale of 1-5)

Please rate the song (1 / 5)*

	1	2	3	4	5
Twist and Shout	○	○	○	○	○
Comfortably Numb	○	○	○	○	○
Bombay Rain	○	○	○	○	○
Anuva's Sky	○	○	○	○	○
Sweet Child of Mine	○	○	○	○	○

The user was asked to give a higher rating for songs that he liked and a lower rating for songs that he disliked. This rating was used to calculate likeability metric.

- How unexpected was the recommendation? (Scale of 1-5)

How unexpected was this recommendation? (1 / 5)*

	1	2	3	4	5
Twist and Shout	○	○	○	○	○
Comfortably Numb	○	○	○	○	○
Bombay Rain	○	○	○	○	○
Anuva's Sky	○	○	○	○	○
Sweet Child of Mine	○	○	○	○	○

This question enquired if the user expected a particular song or artist to be recommended to him/her. A rating of 5 stands for highly unexpected and a rating of 1 being highly expected. Unexpectedness metric was computed using this rating.

- Do you think the recommendation was useful? (Yes/No)

These are recommendations that the user considered appropriate given the context. The user may like a song but may still consider the recommendation as not useful. A useful recommendation increases the user's trust in the engine. Usefulness metric was computed using this rating.

7. Analysis

Out of 84 users who received recommendations, 44 users provided their feedback. Out of these 44 users, 50% got recommendations from the new engine.

Overview:

Table 4. Summary of survey users.

Number of users for whom recommendations were generated	**84**
Total number of songs received	442
Number of unique songs	342

The graphs below shows the demographic and music exploration split for the final 44 users.

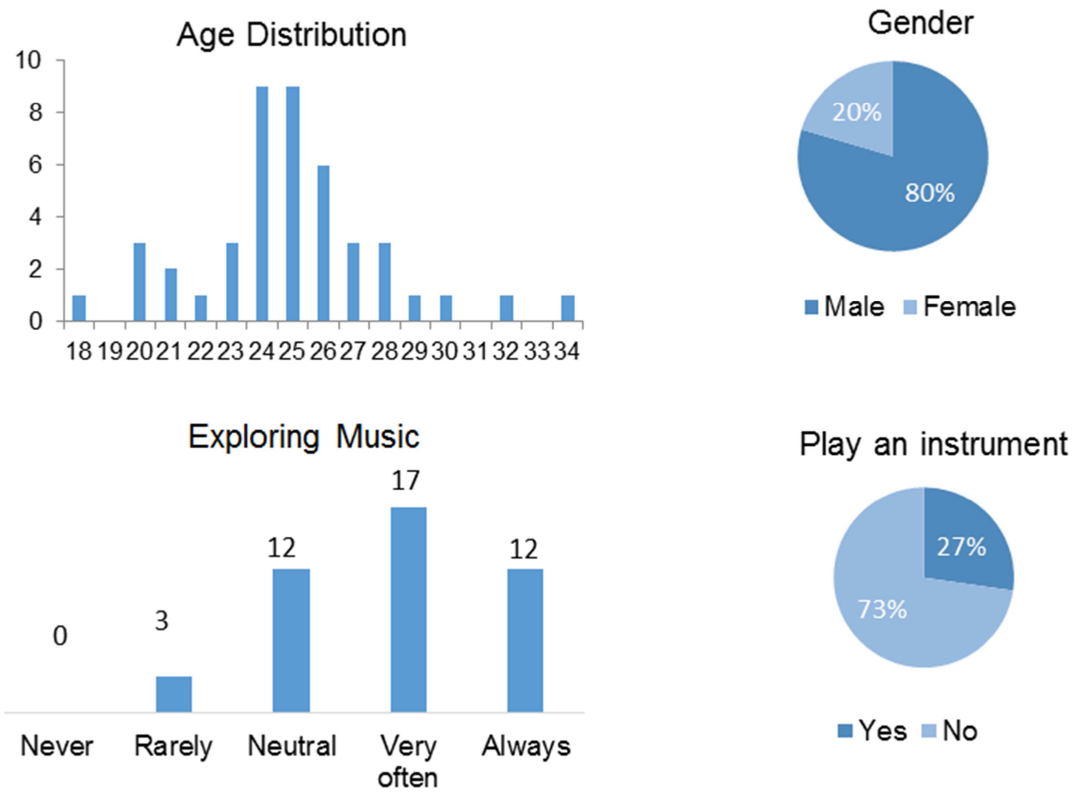


Figure 4. Demographic and music exploration split.

Most of the respondents belonged to the age group of 20-30 and were interested in exploring new music. Out of the 44 users, 35 were male and rest were females. Only 12 people knew how to play an instrument.

The music exploration factor was found to be similar in the two groups of users receiving recommendations from different engines.

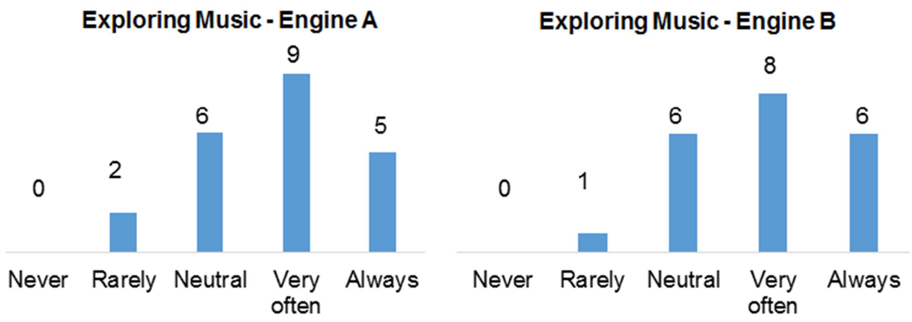


Figure 5. Music Exploration Factor.

The user similarity among respondents is as below:

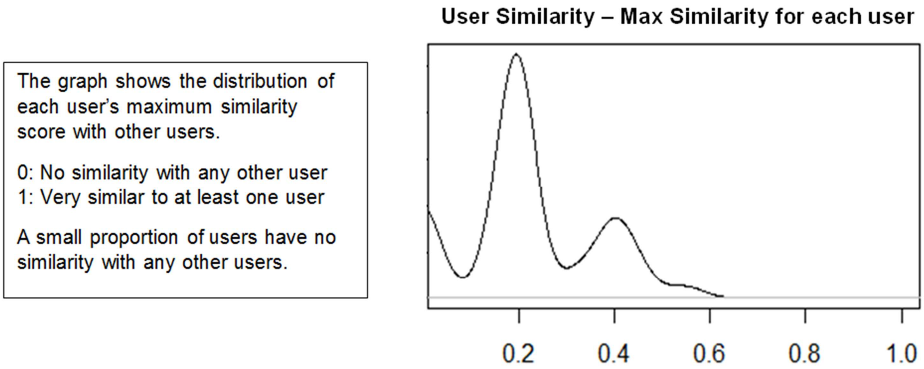


Figure 6. Distribution of User similarity score.

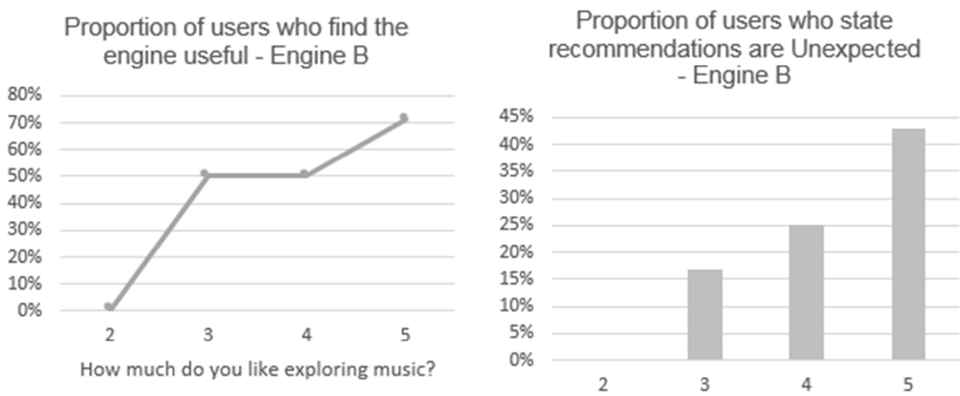


Figure 7. Proportion of users who find Engine B useful and unexpected.

This graph indicates that a fair amount of similarity exists between the users in the sample. Thus, justifying the use of the survey data to generate user profiles and recommendations.

Users were asked to rate each song as useful or not useful. If the users found more than 3 songs useful, the engine recommendations are classified as useful. Engine A was found to be useful by 63% of the respondents and Engine B by 55%.

Table 5. Comparison of Engine Usefulness.

Engine	**Not Useful**	**Useful**
Engine A	8	14
Engine B	10	12

The proportionality tests show a p-value of 0.7591. Thus, there was no statistical difference between the two engines. Thus, the engine is able to add unexpected recommendations without adversely affecting the usability of the engine.

Since the unexpected recommendations will appeal to users who like discovering new music, we look at how users' preference for music exploration correlate with their engine ratings.

The proportion of users who find engine B useful increases, with increase in music exploration factor. Since Engine B is primarily concerned about bringing new and unexpected recommendations to the user, it will be more useful to users who prefer exploring new kinds of music. (Explore ratings more than 3)

Similarly, users were asked to rate each song as unexpected or as per their expectation.

For the new engine with unexpected recommendations, the proportion of users who state that the recommendations are unexpected increase with increase in the explore music value. Higher value for music exploration would mean the user had received more unexpected recommendations, which explains the higher proportion. (Note: a recommendation is termed unexpected if more than 3 (60%) of the songs were rated unexpected by the user)

Finally, we look at whether users like the recommendations.

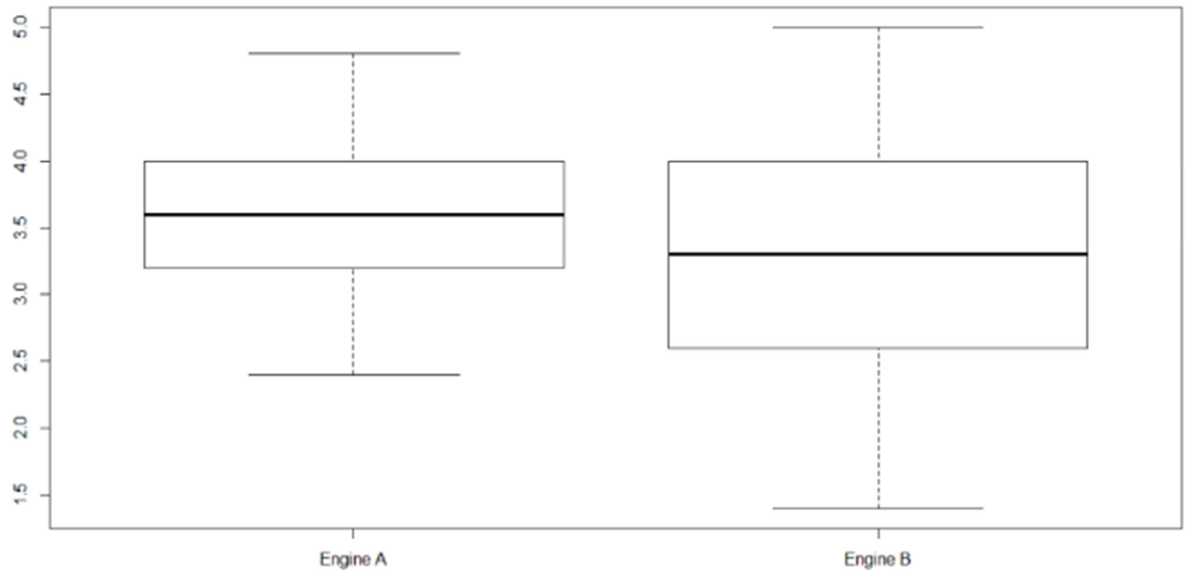


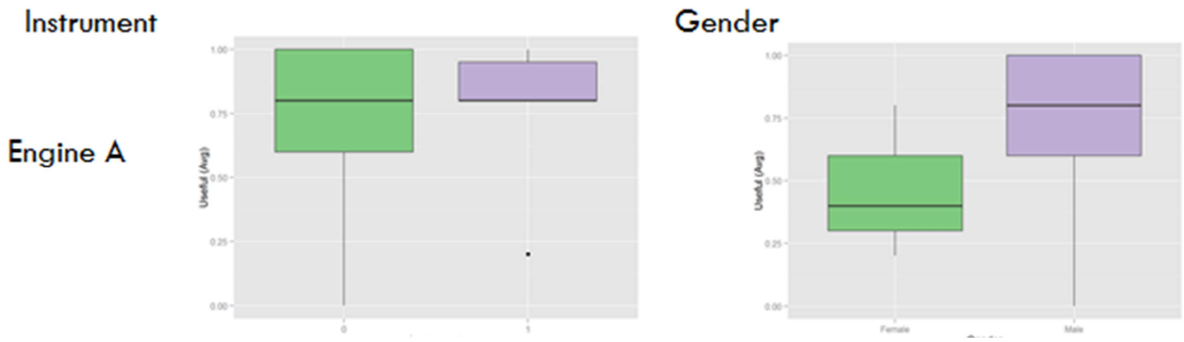
Figure 8. Graph depicting user likeability.

Based on the user's rating for songs (out of 5), we notice that there is no statistical difference between the two engines. (P-value = 0.3005)

Thus, likeability is also not affected by the unexpected recommendations.

The other factors such as age, gender and ability to play a musical instrument on user's preference of the engine were also analysed. The aim to consider these factors was to understand whether significant differences exist between the user's preference of the engine A and engine B based on these factors.

The following observations were made from the survey results.



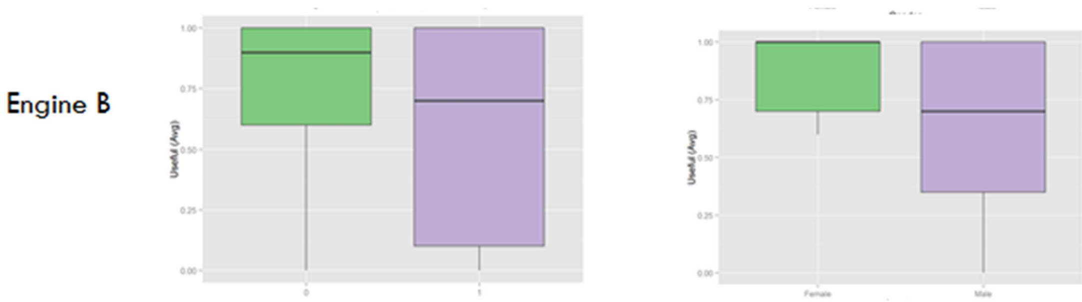


Figure 9. User preference based on - ability to play instrument and gender.

As the survey sample was not very large, no significant differences were found between the different groups of users based on demographics such as age and gender. However, it was observed that users who play an instrument prefer familiar songs rather than new or unexpected recommendations. This maybe because these users have a defined taste in music and would prefer listening to familiar songs (related to the instrument they play).

In conclusion, we have managed to develop an engine which incorporates unexpected recommendations without adversely affecting the engine's usability. In addition, we notice that users who prefer exploring music tend to find the recommendations from the new engine more useful. Due to the small sample size, significant differences could not be observed in terms of user demographics. A bigger sample size with extensive user listening history would yield better results and could be considered for future work.

Since it has been noticed that users who like to explore music like unexpected recommendations, an exploration scale for users to decide the nature of their recommendations can be incorporated. If a user would like to listen to familiar music, he/she can choose a lower rating on the "exploration scale" and vice versa.

8. Conclusion

Listeners expect recommendation engines to show them new music. Most engines discussed in academic fields stay true to this by recommending songs based on user's tastes and listening history. However, users sometimes like to be surprised by unexpected recommendations which turn out to be good. Ultimately, the usefulness of any recommendation engine depends on user satisfaction and trust. When listeners get recommendations which they like, their trust in the engine increases.

In the age of music streaming, it is important for streaming sites to have a loyal subscriber base. It is, therefore, important to have good recommendation engines which the customer trusts.

Unexpectedness of recommendation is an important factor which should be considered while evaluating engines.

We have successfully generated a music recommendation engine which incorporates dissimilar recommendations along with familiar ones, without adversely affecting the usability of the engine. We have also concluded that engine has been extremely useful for those users who are inclined to listen to unexpected or new recommendations.

References

[1] Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. Using Collaborative Filtering to Weave an Information Tapestry. Communications of the ACM 35, 1992

[2] G. Linden, B. Smith and J. York, "Amazon.com Recommendations: Item-to-item Collaborative Filtering", IEEE Internet Computing 7, 2003

[3] B.M. Sarwar et al., "Item-Based Collaborative Filtering Recommendation Algorithms," 10th Int'l World Wide Web Conference, ACM Press, 2001.

[4] Koren, Yehuda, Bell Robert and Volinsky Chris. Matrix factorization techniques for recommender systems, Journal Computer, Volume 42 Issue 8, August 2009

[5] Koren, Yehuda. Collaborative Filtering with Temporal Dynamics. Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, 2009.

[6] Y. Ding and X. Li. Time weight collaborative filtering. Proc. 14th ACM international conference on Information and knowledge management (CIKM'04), 2004.

[7] Koren, Yehuda. Factorization meets the neighborhood: a multifaceted collaborative filtering model. Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, 2008

[8] Pazzani, Michael J, Billsus, Daniel. Content-Based Recommendation Systems. The adaptive web, Springer-Verlag Berlin, Heidelberg, 2007

[9] Stern, David, Herbrich, Ralf, Graepel, Thore. Matchbox: Large Scale Online Bayesian, In WWW, 2009

[10] Logan, Beth. Music Recommendation from Song Sets, In Proc ISMIR, 2004

[11] Yi Li, Rudhir Gupta, Yoshiyuki Nagasaki, Tianhe Zhang, Million Song Dataset Recommendation Project Report, personal.umich.edu, 2012

[12] Fabio Aiolli.: Preliminary Study on a recommender system for the Million Songs Dataset challenge, In Proceedings of ECAI Workshop on Preference Learning: Problems and Application in AI, 2012

[13] Pablo Castells, Saúl Vargas, and Jun Wang. Novelty Diversity Metrics for Recommender Systems: Choice, Discovery and Relevance, DDR-2011: International Workshop on Diversity in Document Retrieval at the ECIR 2011: the 33rd European Conference on Information Retrieval. Dublin, April 18, 2011

[14] Yuan Cao Zhang, Diarmuid Ó Séaghdha, Daniele Quercia, Tamas Jambor. Auralist: Introducing Serendipity into Music Recommendation, WSDM '12 Proceedings of the fifth ACM international conference on Web search and data mining, 2012

[15] R. M. Bell and Y. Koren. Lessons from the Netflix prize challenge. ACM SIGKDD Explorations Newsletter, 2007.

[16] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the 14th ACM Conference on Uncertainty in Artificial Intelligence, 1998.

[17] J. L. Herlocker, J. A. Konstan, A. Borchers and John Riedl, “An Algorithmic Framework for Performing Collaborative Filtering”, Proc. 22nd ACM SIGIR Conference on Information Retrieval, 1999

[18] R. Salakhutdinov and A. Mnih, “Probabilistic Matrix Factorization”, Advances in Neural Information Processing Systems 20 (NIPS’07), 2008.

[19] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, “Application of Dimensionality Reduction in Recommender System – A Case Study”, WEBKDD’2000

[20] Beth Logan and Ariel Salomon. A music similarity function based on signal analysis. In ICME 2001, Tokyo, Japan, 2001.