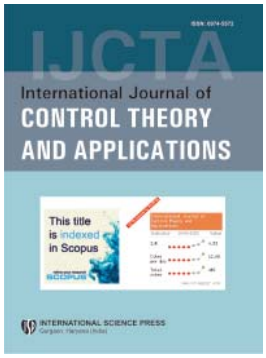

International Journal of Control Theory and Applications

ISSN : 0974–5572

Volume 9 • Number 40 • 2016

Music Recommender System Based on user Profile

M. Ferni Ukrit[a] J.S. Femilda Josephin[a] S. Shivani Reddy[b] and K. Susmitha[b]

[a]*Assistant Professor, Department of Software Engineering, SRM University*

E-mail: fernijegan@gmail.com, Corresponding Author

[b]*Student, Department of Software Engineering, SRM University*

Abstract: With the increasing number of music services such as google Play Music, iTunes, Spotify etc. there is a demand to engage the customers to their sites. Recommending songs to the user is one way of meeting the demand. In our recommender system, we would be using the Last.fm dataset which contains around a thousand users and their listening history. We are using the user profiles of the users and we show that recommendations based on this are better than recommendations without considering their profile. The Last.fm dataset, however, doesn't contain any ratings given by users and it makes it difficult to use a collaborative filtering algorithm over this. To overcome this problem, we are predicting the ratings which might have been given by users by assuming a Hidden Markov Model which uses ratings as the hidden variables and we find these ratings by calculating the values of the ratings which would have given rise to a maximum probability of the assumed model. We show that considering user profiles and the Hidden Markov Model is a good way to overcome the problem that collaborative filtering has when there is no user rating available.

Keywords: *Music Dataset, Hidden Markov Model, User profiles.*

1. INTRODUCTION

For a music service, recommendations are a way to catch the customer's attention and their goal is to keep them occupied. It can be achieved when a music recommender successfully suggests songs that the customer is interested in. It can result in, customer purchasing a subscription, or the customer paying for the song. In both cases, these music services can derive financial gain by improving their recommendations to potential customers of the songs they like. Thus, there is strong financial incentive to implement a good Music recommendation system.

Collaborative filtering methods as discussed in papers [1]- [4] are popular recommendation algorithm whose predictions depend on the rating or behavior of other users in the system. A drawback of this method is that it cannot work without ratings. To overcome this drawback, many recommender system consider a pseudo rating. Some recommender systems give a rating of 0 (if the user did not listen to the song) and 1 (if the user the listened to the song). Other recommender systems consider the count [8] of a song as the rating given by the user. The dataset that we are using to train and test our system is Last.fm dataset. This dataset does not contain the ratings for the songs. In our paper, we propose a method to rate the songs in such a way that they result in recommendations better than traditional method.

Papers on corresponding work in the field talk about how a distinct method can affect the recommendations [3]. For example, the winner of the Kaggle MSDS competition3 provided a Memory-based Collaborative Filtering method [7]. Also, many other papers talked about different song similarity approaches [6]. In our paper, we are going to implement a user similarity approach where we calculate the user similarity using various measurable quantities like age, sex, Location etc. We use this similarity to recommend the songs to the users. These publications [5]- [6] focused on deriving methods that provide a better precision. We surveyed some of these algorithms in our work and used them as baselines and in designing our final solution.

In this paper, we propose a general music recommender from user profiling, and compare its efficiency against other traditional Recommender System. Section II gives a detailed description of the method employed and in section III, results and efficiency of our findings is done. To the end of this paper, we conclude and give an insight into our future work.

2. METHODOLOGY USED

2.1. Traditional Method

One of the common approaches employed by recommender systems to deal with the non-existence of the ratings is to consider the count as the rating for the song. The count here is the number of times the user has listened to that particular song. The traditional method in our paper uses such an approach to rate the songs.

2.2. Proposed Method

Our method uses a different approach towards rating the songs and finding the similar users. Figure.1 shows the architecture of our system where various modules of the system are explained as follows:

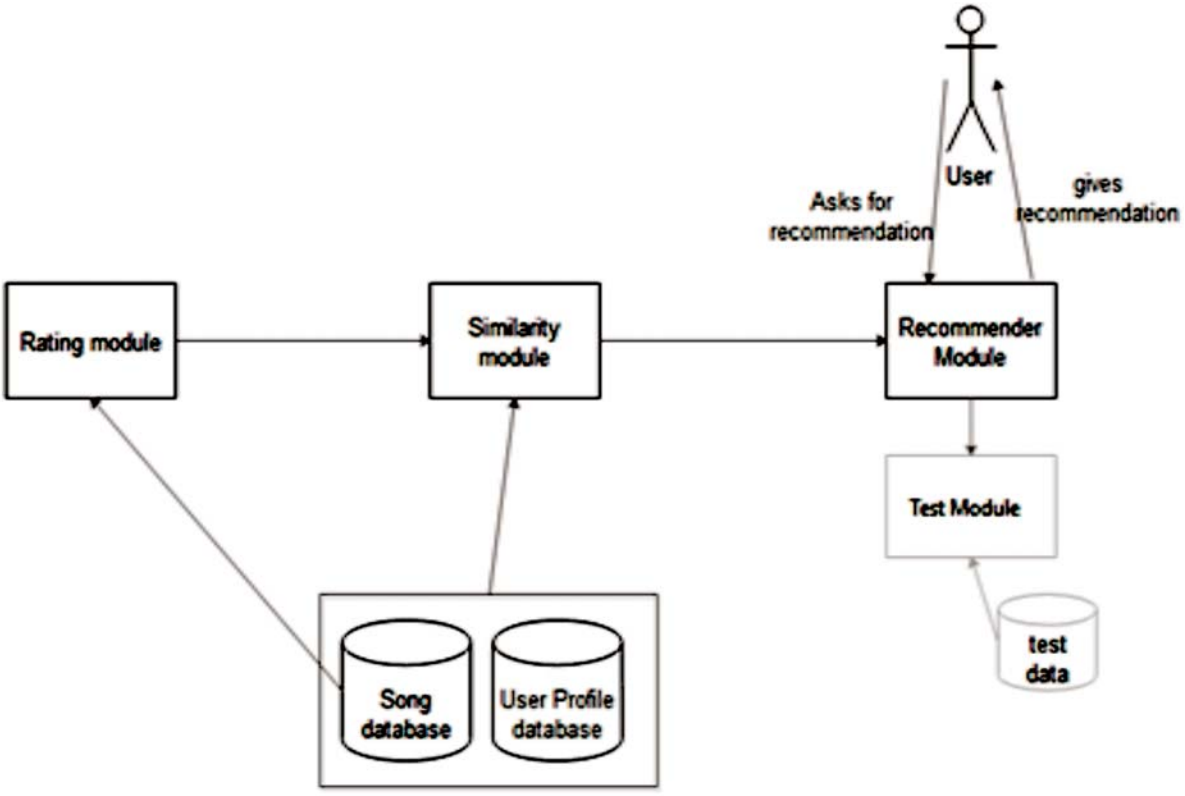

Figure 1: System Architecture

2.2.1. Rating module

To overcome the drawbacks of the existing methods, we propose a new method which is based on a Hidden Markov Model (HMM). The outcomes are given in the set Y = $\{Y_{us}\}$ where Y_{us} is the rating we say the user u gives to song s based on our original method. The actual ratings of users are in the set R = $\{R_{us}\}$ where R_{us} is the rating the user may have given which gives rise to the behavior of Y_{us} . These are the hidden variables of the HMM. We want to have our model in such a way that each song has an inherent rating and therefore the prior probability of a song 's' to have rating R_{us} is given by a Gaussian with mean μ_s, standard deviation σ_s. Our model will try to consider user variability and therefore the prior probability for a user '*u*' to give rating R_{us} is given by a Gaussian with mean μ_{us}, standard deviation σ_u. The likelihood for a user '*u*' to give a song '*s*' a rating R_{us} is exp $(-K*(Y_{us}-R_{us})^2)$ where a higher K means that we want to give a higher preference to the observed data. The above described HMM can be depicted as in Figure 2

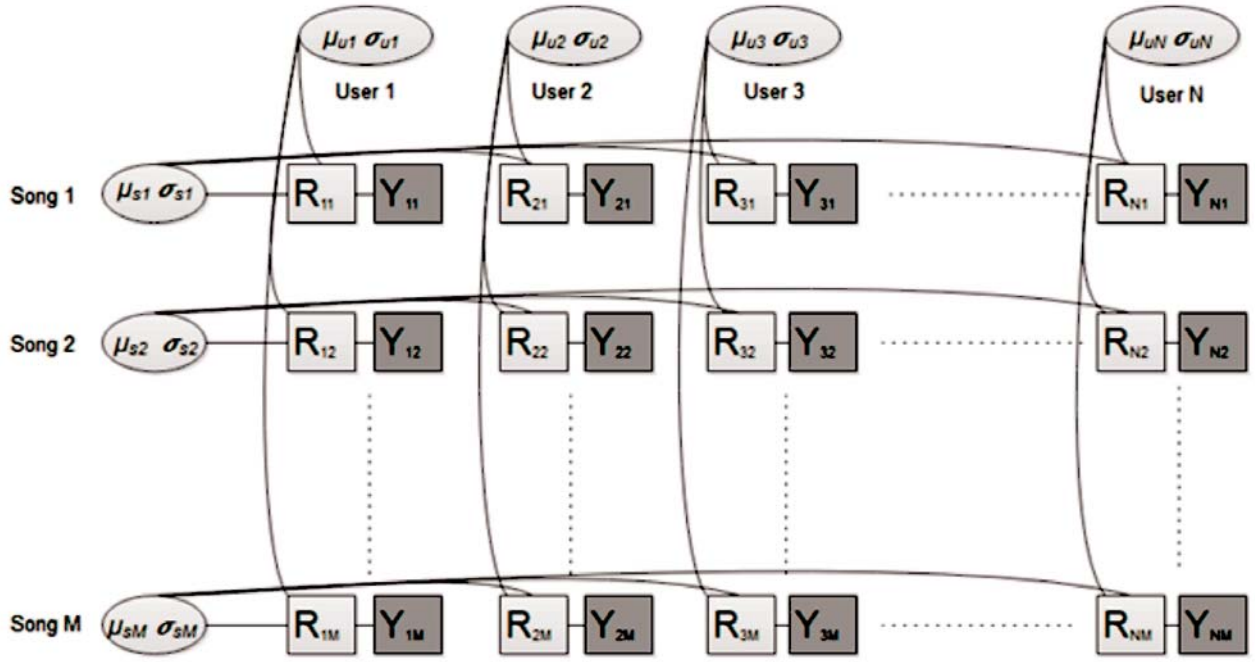

Figure 2: HMM for the proposed method

The probability of R = $\{R_{us}\}$ can be written in mathematical notation as:

$$P(R \mid Y) = P(Y \mid R) * P(R) / P(Y) \propto P(Y \mid R) * P(R)$$

We want to find R so that P(R | Y) is maximized. In mathematical notation, it can be written as:

$$R^* = \arg\max_R P(R \mid Y)$$

$$= \arg\max_R P(Y \mid R)\, P(R)$$

$$= \arg\max_R \prod_u \prod_s P(Y_{us} \mid R_{us}) * P(R_{us})$$

$$= \arg\max_R \prod_u \prod_s \exp(-K(Y_{us} - R_{us})^2) * \exp\left(-\frac{(R_{us} - \mu_u)^2}{2\sigma_\mu^2}\right)$$

$$* \exp\left(-\frac{(R_{us} - \mu_u)^2}{2\sigma_\mu^2}\right) / (2\pi\sigma_s\sigma_u)$$

To find R^* we propose to use the Expectation Maximization algorithm so that in the E-step we find the expectation of μ_s, σ_s, μ_u, σ_u which are the mean and standard deviations of rating for user 'u' and song 's'. After calculation, the M-step update was found to be the update to each R_{us} such that:

$$R_{us} = \frac{\sigma_s^2 \mu_u + \sigma_u^2 \mu_s + \sigma_s^2 \mu_u^2 Y_{us}}{\sigma_s^2 + \sigma_u^2 + \sigma_s^2 \mu_u^2}$$

We run the EM algorithm until convergence and give the calculated rating R_{us} to the similarity module which will calculate the similarities between users.

2.2.2. Similarity module

The dataset we used was provided by Last.fm and it contains a collection of 1K users, more than 40K artists and the numbers of songs played are above 1 Million. We are using collaborative filtering to find the similarity between users to be able to recommend songs. Similarity between two users can be modelled accurately if we consider their profile and their song listening history. A song history vector is a vector of size |S| where S is the set of songs such that S(i) = rating given to song i by the user.

To find the cosine similarity between song history vectors S_1 and S_2 we do,

$$\text{Cosine Similarity}(S_1, S_2) = \cos(\theta) = S_1 \cdot S_2 / (\|S_1\| * \|S_2\|) \text{ where,}$$

$$S_1 \cdot S_2 = \sum_i S_1(i) * S_2(i)$$

and

$$\|S\| = \text{sqrt}(S \cdot S)$$

The cosine similarity is a measure of how close two vectors are. The greater this measure, in our case it would mean that the similarity for users with similar song listening history would be greater. We also have user profile in the form of 4 measurable quantities, Sex, Age, Location, Date of Joining. Putting all these together, we created a feature similarity index between two users by using their feature vectors. The similarity between such feature vectors F_1 and F_2 is given by,

$$\text{Feature Similarity}(F_1, F_2) = G(F_1, F_2), \text{ where G is our measure}$$

Using these two similarities, we compute a total similarity between users U_1 and U_2 given by,

$$\text{User Similarity}(U_i, U_2) = \alpha * \text{Cosine Similarity}(S_1, S_2) + (1-\alpha) * \text{Feature Similarity}(F_1, F_2)$$

2.2.3. Recommender Module

Once we get similarities between all users, to recommend songs to a user, we take the top similar users to that user and recommend the top-rated songs of this user that the current user has not heard before. To be able to recommend songs well, we keep a pool of 5 most similar users. From this pool of 5 users we add songs to the recommendation list by giving more preference to more similar users and then if we are not able to add a song to the list because it already exists in the list or if the user we are recommending has heard it before, we move to the next song to recommend which can be from the same user or from the next user. Once we get several recommendations we stop and return this list to the user.

3. RESULTS AND DISCUSSION

To be able to test how good our method compares to the traditional method which uses just counts for the ratings, we employ a method discussed in [9]. According to that method we remove a few songs from the list of songs of a user and using the new list, we are calculating the similarities and give recommendations based on that. We say a recommendation is a good recommendation if the recommendation list consists of a song from the list of songs we removed. Using this we define the recommendation accuracy to be the percentage of users who received a good recommendation.

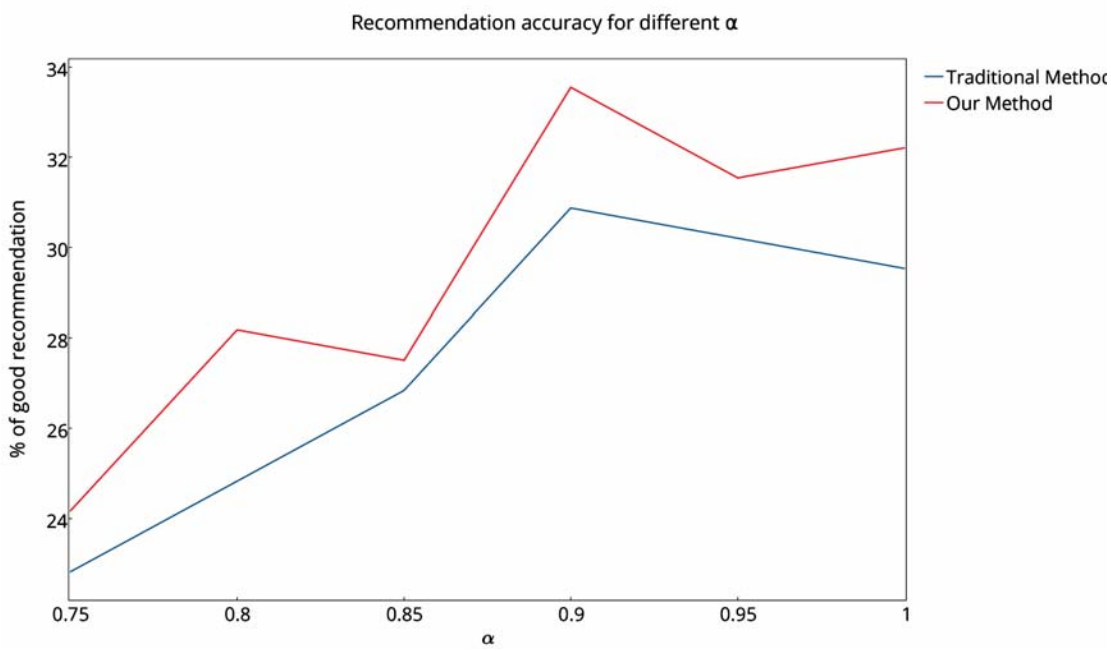

Figure 3: Songs recommended and top 4 songs removed

In Figure 3, we show how varying the free parameter α changes the recommendation accuracy. Recall that α is the weightage given to the similarity from the user's ratings. 1-α is the weightage given to the similarity from the users' profiles. With this is mind, we see that when α = 0.9 the recommendation accuracy is the highest. So, when we have a mixture of user profile and user ratings, the recommendations seem to be the most accurate. If we look at the case where α = 1, *i.e*, we are considering only the user ratings for recommendation, it's worse than the case where we are using the profile. This shows that using profile data increases the recommendation accuracy. The figure also shows how our method outperforms the traditional method at every α showing that our method is superior to the traditional method of recommendation.

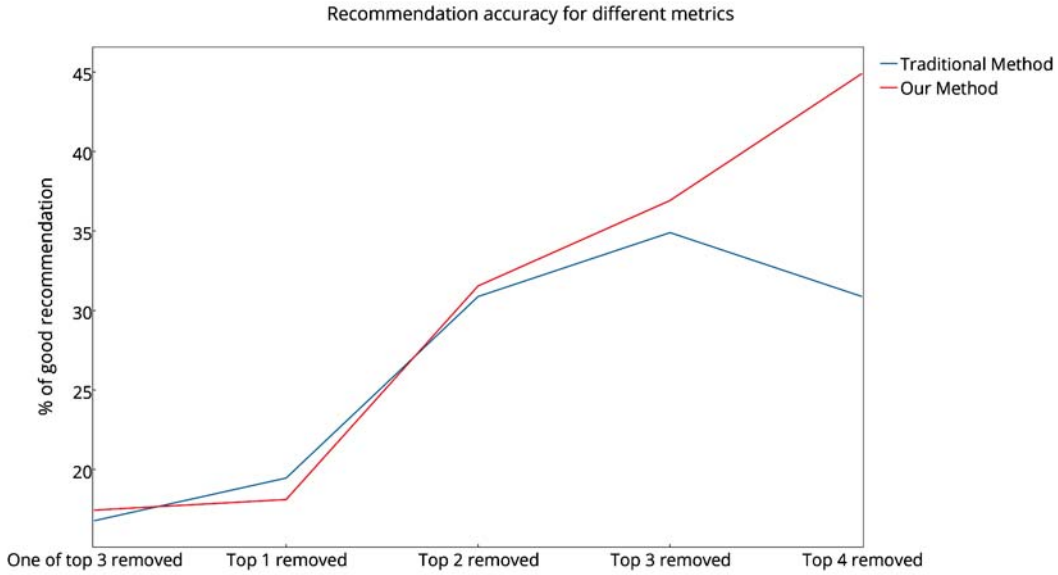

Figure 4: Songs recommended with α = 0.9

We perform another experiment where we remove a different number of songs from the users history. In particular, we show results after removing the top 1,2,3 and 4 songs and removing one of the top 3 songs at random. The results as shown in Figure 4, indicate that our method performs much better than the traditional method as we remove more and more songs from the users listening history. The case when 4 songs are removed is interesting in particular because the traditional method performs poorly compared to our method.

In Figure 5, we show how changing the number of songs recommended affects the accuracy. We see that our method outperforms the traditional method when a few songs are recommended. This is the way most recommender systems behave and the fact that our method outperforms the traditional method at low songs recommended shows that our method is much better.

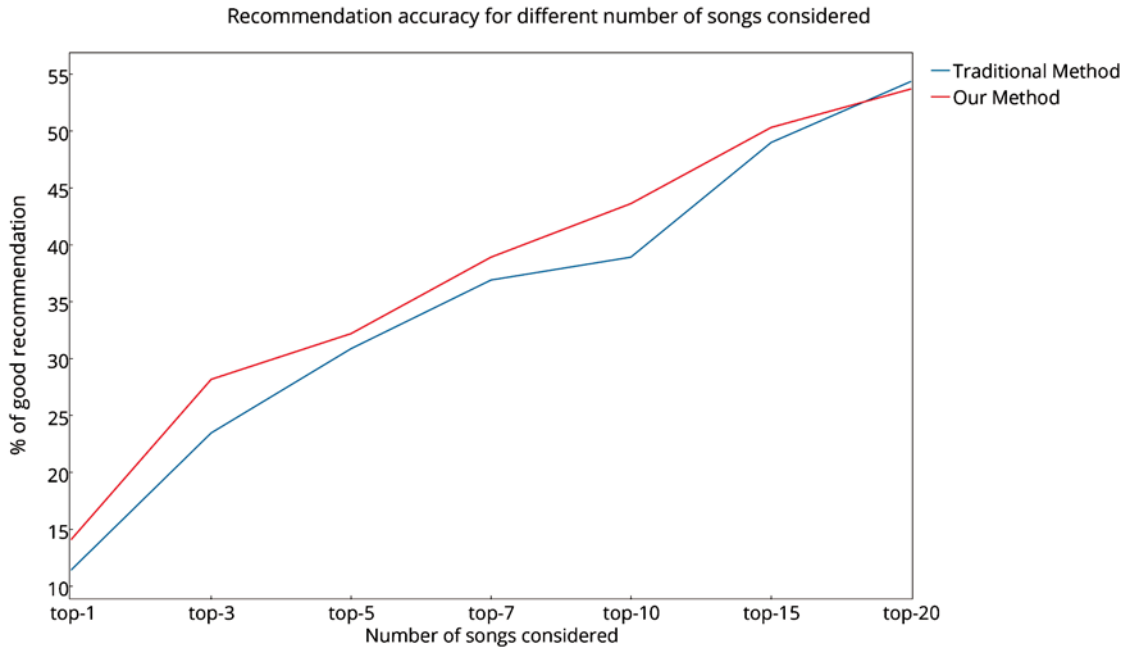

Figure 5: Top 4 removed with α = 0.9

4. CONCLUSION

The results as discussed in the above section show that the proposed method overcomes the problems of the traditional method, when there are no ratings to work with. The probabilistic framework of our method is a better way to model the ratings than consider ratings just by the count. We also show how using the user profile can affect recommendations and have shown that considering it a much better alternative than not using the information we have about the users.

REFERENCES

[1] Yading Song, Simon Dixon, and Marcus Pearce. A Survey of Music Recommendation Systems and Future Perspectives. In 9th International Symposium on Computer Music Modelling and Retrieval (CMMR 2012)

[2] F.O.Isinkaye, Y.O.Folajimi and B.A.Ojokoh. Recommendation systems: Principles, methods and evaluations. Egyptian Informatics Journal,pp 261-273, 2015.

[3] Kazuyoshi Yoshii, Masataka Goto, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. An Efficient Hybrid Music Recommender System Using an Incrementally Trainable Probabilistic Generative Model. IEEE transactions on audio, speech, and language processing,vol 16(2), February 2008.

[4] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-Based Collaborative Filtering Recommendation Algorithms. In proceedings of 10th international conference on World Wide Web (WWW 2001)

[5] YajieHu and MitsunoriOgihara. NEXTONEPLAYER: A MUSIC RECOMMENDATION SYSTEM BASED ON USER BEHAVIOR In the proceeding of 12th International Society for Music Information Retrieval Conference (ISMIR 2011)

[6] Al Mamunur Rashid, George Karypis, and John Riedl. Learning Preferences of New Users in Recommender Systems: An Information Theoretic Approach. Newsletter 10(2):90-100,2008

[7] Fabio Aiolli.: Preliminary Study on a recommender system for the Million Songs Dataset challenge. (2011)

[8] Hung-chen Chen, Arbee L.P.Chen. A Music Recommendation System Based on Music and User Grouping. Journal of Intelligent Informatics Systems,24:2/3, 113-132, 2005.

[9] B. Logan, "Music recommendation from song sets," in Proc. Int. Conf. Music Inf. Retrieval (ISMIR), 2004, pp. 425–428.