# Music Recommendation System Using Association Rules

**Vinjamuri Swathi, M.Sunitha Reddy**

M.Tech, Dept of Computer Science, Vasavi College of Engineering, Hyderabad, India.
Assistant Professor, Dept of Computer Science, Vasavi College of Engineering, Hyderabad, India.
Email: vinjamuriswathi507@gmail.com, sashu2006@gmail.com

**ABSTRACT:** Recommender systems have been developed in variety of fields, including music recommender systems which are one of the most interesting ones. Because of the information overload and its varieties in music data, it is difficult to draw out the relevant music. Therefore, recommender systems play an important role in filtering and customizing the desired information. In the proposed system, association rules mining technique is applied to filter the similar users, and then the related songs are recommended to them. For finding the similar users Frequent Pattern Growth Algorithm is being used. Since, FP-Growth allows frequent itemset discovery without the generation of candidate itemset. After finding frequent patterns then association rules are generated. Finally recommendations are done to the users and the accuracy is checked.

**Keywords:** Data mining, Recommender system, frequent patterns, association rules, FP-Growth.

## 1. INTRODUCTION

Data mining can be defined as the practice of automatically searching large stores of data to discover patterns and trends that go beyond simple analysis. Data mining uses complicated mathematical algorithms to segment the data and to evaluate the probability of future events. For example, by applying data mining techniques automatic discovery of patterns, prediction of probable outcomes, formation of actionable information, focus on large data sets and databases are carried out. In this paper we explore the data mining technique for music recommendation system using association rule mining. In this paper we discuss and proposed solution for the users to listen to the similar songs based on the past experience. However, collaborative and content-based filtering are the mostly adopted techniques utilized in recommender systems. The collaborative filtering recommends items based on liked-mind users' opinions and users' preferences. Whereas, the content-based filtering technique is the identification of items which are similar to those a user has preferred in past. A recommendation system should be able to minimize user's effort required to provide feedback and simultaneously to maximize the user's satisfaction by playing appropriate song at the right time. Reducing the amount of feedback is an important point in designing recommendation systems. The goal of the system is to recommend songs that are preferred by the user, are fresh to the users' ear, and fit the user's listening pattern. The importance of music recommender systems is increasing because many online services that manage large music collections do not provide users with fully satisfactory access to their collections. Standard retrieval systems force users to discover their favourite musical pieces by using queries including the songs' titles or artist names. However, many users have difficulty in stating their musical preferences as queries. In fact, most users of music streaming services that let one freely listen to numerous songs for a flat fee, want to access their favourites one after another with no querying. Recommender systems should thus be able to select musical pieces that will likely be preferred by estimating user preferences.

## 2. LITERATURE

Recommender systems are tools for filtering and sorting things and data. Recommender systems fall into two categories:

- content-based recommendation
- collaborative recommendation

**2.1 Content based recommendation:** Content-based recommender systems work with profiles of users that are created at the beginning. A profile has information concerning a user and his taste which is based on how user rates the items. Generally, when creating a profile, recommender systems formulate a survey, to get initial information about a user in order to avoid the new-user problem. In the recommendation process, the system compares items that were already positive rated by user with items he did not rate and looks for similarities. Those items that are almost similar to the positive rated ones, will be recommended to the user.

**2.2 Collaborative Filtering:** To recommend items via the choice of other similar users, collaborative filtering technique has been proposed. As one of the most successful approaches in recommendation systems, it assumes that if user X and Y rate m items similarly or have similar behavior, they will rate or act on other items similarly. Instead of calculating the similarity between items, a set of 'nearest neighbor' users for each user whose past ratings have the strongest correlation are found. Therefore, scores for the unseen items are predicted based on a combination of the scores known from the nearest neighbors'. Collaborative filtering is divided into three subcategories: memory-based, model-based, and hybrid collaborative filtering.

2.2.1 Memory-based Collaborative Filtering Memory-based collaborative filtering is to predict the item based on the entire collections of previous ratings. Every user is grouped with people with similar interests, so that a new item is produced by finding the nearest neighbor using a massive number of explicit user votes.

2.2.2 Model-based Collaborative Filtering In contrast to memory-based CF, model-based CF uses machine learning and data mining algorithms which allow the system to train and model the users' preferences. It represents the user preference by a set of rating scores and constructs a special prediction model. Based on the known model, the system makes prediction for test and real-world data.

2.2.3 Hybrid Collaborative Filtering Hybrid CF model is to make prediction by combining different CF models. It is been proved that the hybrid CF model outperforms any individual method.

**Limitations of Collaborative Filtering techniques:**
New User Problem: Collaborative filtering has the same problem as the content-based approach which is new users entering the system. In order to make recommendations to a user, the systems need to have knowledge of the user's preferences from the ratings that the user makes. Thus, system will not be able to provide accurate recommendations. New Item Problem: The systems should contain rated items in order to recommend some items to the users. Whenever a new item enters the systems, the item has not rated by users yet. As a result, the systems will not be able to recommend it to the users. Sparsity : The total number of ratings is important in the recommendation system. In order to provide accurate recommendations by the recommendation systems, sufficient number of ratings should exist in the systems. Scalability: In many practical collaborative filtering recommendation systems, the number of users and items increase rapidly in the system. Therefore, the system needs to provide more and complicated computational process, and this leads the computational resources going beyond the acceptable levels.

**Recommendation using association rules**
Recommendation using association rules is to predict preference for item k when the user preferred item i and j, by adding up confidence of the association rules that have k in the result part, i or j in the condition part used the rule with the maximum confidence, but we used the sum of confidence of all rules in order to give more weight to the item that is associated with more rules. Recommendation using association rules describes as follows. Let P be the preference matrix of n users on m items. In this matrix, $p_{ij}$ is 1 if the user i has preference for the item j, and 0 otherwise. Let A be an association matrix containing confidence of association rules of m items to each other. The matrix A is computed from P. In this matrix, $a_{ij}$ is confidence of association rule $i \Rightarrow j$. Then the recommendation vector r for the target user can be computed from the association matrix A and the preference vector u of the target user as equation. The top-N items are recommended to the target user based on the values.

## 3. PROBLEM DESCRIPTION

Recommendation systems can be classified based on what characteristics of the data they are considering. Typically, they can be categorized into three groups:

1. Pure Static Recommendation System: Here only the rating triplet (user, item, rating) or count data (user, item, count) or just binary transaction data (user, item) is considered. No additional information is made use of.
2. Hybrid Static Recommendation System: Here apart from the mentioned data, the recommendation system also utilized additional features related to the user or the item itself.
3. Dynamic Recommendation System: Here the recommendation system assumes that the user's preferences change over time and hence processes the data.

Our problem description corresponds to the pure static recommendation problem, which is most common in literature. Our approach is as follows:

1. Dataset (Taste Profile) is considered, which gives the information related to user ratings and songs and is preprocessed using the constrains like(user listened >3 and song >30 times).
2. Construct User-item matrix from the given dataset, summarizes the unique users and the songs they have listened.
3. Split the dataset into 3:1 proportion as training data and test data.
4. FP-Growth algorithm is used to generate the frequent itemsets on the database generated.
5. The threshold minimum support is used to check the frequency of the pattern.
6. If the pattern satisfies the threshold the rules are formed on the frequent items.
7. The rules satisfying the threshold are strong association rules and prune out the remaining rules.
8. Then calculate the true positives and false positives for the calculation of precision, recommend the top-N songs to the target user which are in that pattern.

## 4. DATA

We intended to try our approach on very large datasets. Taste Profile Dataset a subset of Million Song Dataset, which is a freely available set of audio features and metadata for a million contemporary popular music tracks. It consists of:

- 280 GB of data
- 1, 000, 000 songs/files
- 44, 745 unique artists
- 43, 943 artists with at least one term

The million song dataset is also cluster of complementary dataset contributed by:

1. Second Hand Songs dataset
2. MusicXmatch dataset
3. Last.fm dataset
4. Taste profile subset

**Taste Profile Subset:** Taste Profile subset, the official user dataset of the Million Song Dataset. The dataset contains real user – play-counts from private partners, all songs already matched to the MSD.

- 1,019,318 unique users

- 48,373,586 user - song - play count triplets

The data looks like this:

User ID Song ID count
b80344d063b5ccb3... SOYHEPA12A8C13097F 8
b80344d063b5ccb3... SOYYWMD12A58A7BCC9 1
85c1f87fea955d09... SOACWYB12AF729E581 2

The data set is split into 75%/25% disjoint splits into training and test data. This is used for fold cross validation.

## 5. ASSUMPTIONS

The algorithm will depend on the subsequent assumptions:

- Rating is ordinal. However, it is ok to have ordinal or continuous rating prediction.
- A user may listen to song any number of times.
- A user may not listen all the songs and listened only a small subset of the entire list of songs.
- Songs not listened by a user are given an ordinal rating of 0.
- Data preprocessing is done by taking the constraint as the user has listened at least three songs and the songs are preprocessed using song listened by at least 30 users.

## 6. ALGORITHM

The algorithm used is FP-Growth. This approach is based on divide and conquers strategy for producing the frequent item sets. FP-growth is mainly used for mining frequent item sets without candidate generation. Major steps in FP-growth are

**Step:** It firstly compresses the database showing frequent item set in to FP-Tree. FP-Tree is built by means of 2 passes over the dataset MSD described above.

**Step2:** It divides the FP-Tree into a set of conditional database and mines each database separately, thus extract frequent item sets from FP-Tree directly.

It consist of one root node labeled as null, a set of items prefix sub trees as the children of the root, and frequent item header table. Each node in the item prefix sub tree contains of three fields: item-name, count and node link. Each item in the header table consists of two fields' item name and head of node link, which points to the initial node in the FP-Tree carrying the item name. The pseudo code of mining on FP-tree is depicted here.

Input: generated FP-tree
Output: Set of frequent patterns
Method: Call FP-growth (FP-tree, null).
procedure FP-growth (Tree, α)
{
if Tree contains a single path p then
for each combination do generate pattern β ∪ α with support = minimum support of nodes in β.
Else For each header $x_i$ in the header of Tree do {
Generate pattern β = $x_i$ ∪ α with support = $x_i$.support;
Construct β.s conditional pattern base and β.s conditional FP-tree Tree β
If Tree β = null
Then call FP-growth (Tree β, β)}
}

## 7. EXPERIMENTAL RESULTS

Using divide and conquer strategy, we converted taste profile dataset into a format that can be used by the association rule mining algorithm. Due to limitations in JVM we have considered a dataset of 50,000 records for which we got about 474 users and 2264 songs with the constraint mentioned above. Next splitting the database using holdout method into 75% as training data i.e 355 records & 25% as test data as 119 records.
Generate the frequent itemsets and association rules for the splitted datasets with the thresholds support as 10% and confidence as 60%.

$$\text{support}(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N}$$

$$\text{confidence}(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

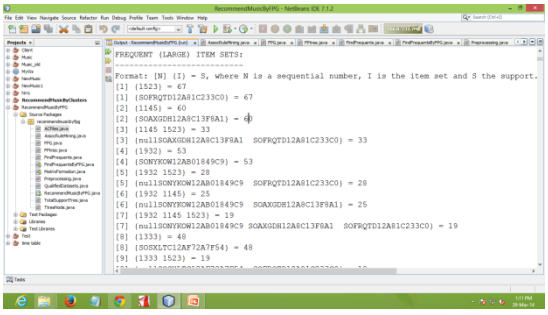


**Figure1** showing frequent itemsets.

After the association rules are formed then the antecedent of rules in training set are compared with the antecedent part of rules in test set and precision is checked then if both the parts are same then the consequent part of the train rule i.e, the songs are recommended to the user in the test set.

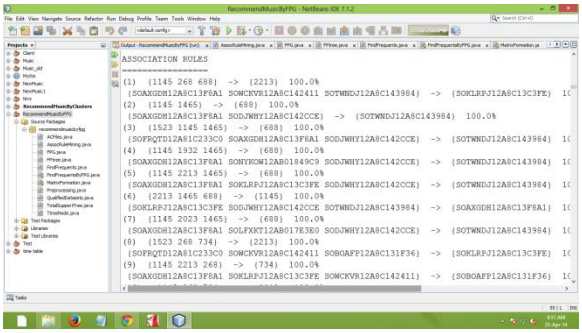


**Figure 2** showing association rules.

## CONCLUSION

In this paper we have presented a scalable framework for music recommender system using association rule mining from MSD data. Our framework includes an efficient data structure for storing frequent itemsets combined with a recommendation algorithm which allows for the generation of recommendations without first generating all association rules from itemsets. Our results show that the proposed framework can provide an effective alternative to standard collaborative filtering mechanism. In particular, we have shown that the association based recommendation framework can, improve on the kNN based collaborative filtering both in terms of the precision and coverage of recommendations. In future, the association rules can be combined with clustering techniques for improved results as per the current knowledge.

## FUTURE WORK

As of now the recommendations are done only using the association rules but for more improved results the association rules along with the clustering can be used for more accurate results.

### REFERENCES

[1]. G Linden, B Smith, J York, “Amazon. Com recommendations: item-to-item collaborative filtering”, Internet Computing, IEEE, 200

[2]. Agrawal, R., Imielinski, T., Swami, A. Data Mining: A performance Perspective. IEEE Trans. Knowledge and Data Engineering, vol, 5,6, 1993a, pp. 914-925.

[3]. A Survey of Music Recommendation Systems and Future Perspectives Yading Song, Simon Dixon, and Marcus Pearce.

[4]. Rakesh Agrawal, T. Imielinski, and A. Swami. Mining Association Rules between Sets of Items in Large Databases.

[5]. R. Agrawal, R. Srikant. “Fast Algorithms for Mining Association Rules”.

[6]. Bayardo Jr, Roberto J. "Efficiently mining long patterns from databases.“

[7]. Pang-Ning Tan, Michael Steinbach, Vipin Kumar: Introduction to Data Mining.

[8]. http://labrosa.ee.columbia.edu/millionsong/tasteprofile