Volume 5, Issue 3, March 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Increasing Retrieval Quality in Critiquing based Recommender System

Sachin Pawar*, Milind Munde, Nikhil Kshirsagar, Rajdeep Sharma, Sunita Barve
Department of Computer Engineering, University of Pune,
Pune, Maharashtra, India

Abstract— Recommender systems are a hot topic in this age, our group has developed a decision tools based on example critiquing to help users find their preferred products in e-commerce environments. Recommender systems are systems that recommend new items to the user based on information about a user's past patterns.In this Paper, we study different filtering methods and algorithm for music recommender system. Critiquing is a common and powerful form of feedback, we critiquing and summarize the major results by using different entity or attributes used to store the items information .System recommend the items to the user based on user preferences for that purpose we take the rating from the user. But the problem with this is what item will recommend when new user come this is called as cold start problem to avoid this problem we use the user profile data for the recommendation.

Keywords— Critiquing, preferences, Filtering, Cold Start Problem.

I. INTRODUCTION

Recommender systems are a hot topic in this age of extremely large data and web marketing. Recommender systems are systems that, based on information about a user's past patterns recommend new items to the user. There are different types of recommender system, some systems uses information about the items, some are based only on user preferences. Other system is known as a collaborative filtering system. Instead of asking the user to explicitly pick filters for a search, collaborative method uses information about the user's past preferences and similar users for recommendation.

Applications of recommender systems can be found outside the online retail trade, although that is one of the most popular places to find them. Our group was to build a recommendation system project to recommend music to the user.

II. PROJECT DESCRIPTION

Recommender systems can be present in different types of system and situations, and thus can be implemented in the different ways. The overview of the methods of implementation that will help to understanding what we done in our project.

A] Data Collection:

In order to make any recommendations, the system has to collect data. We collect the data is to get the find user preferences which my use in future for future user preferences. There are two ways to collect the data. The first method is to ask for explicit ratings from a user, such as rating a song S from one to five stars. The second is to collect data implicitly as the user is in the domain of the system that is, to log of a user on the site. There is a small difficulties related data gathering Data collection can only record the actions of a user, and knows nothing about the person behind the computer it is near impossible to properly a user who actually happens to be two people using the same computer.

B] Data Filtering Method

Once you've gathered your data, there data filtering methods used for speed up the processing of the data retrieved from dataset. There are different method of data filtering we used in our project as follows.

1) Passive and Active Filtering:

In the method is **passive filtering**, which simply uses data aggregates of rating to make Recommendation. The advanced method is to use **active filtering**, which uses user history to recommend. An example of this would be finding similar users to the current user and using their history to predict a rating.

2) User-centric and Item-centric Filtering:

A **user-centric system** will find similarities between users then use the similar users' preferences to predict ratings. An alternative to user centric method is an **item-centric system** which will attempt to find the relationships between items, and make Recommendations then only based on a user's preferences and these relationships.
It is not necessary to focus only on users or items, but most typically only find similarities between users or similarities between items and not both.

III. DATA STRUCTURE

One of the early challenges we faced when developing our recommender system was choosing data structures that would allow easy and efficient access to large data sets. This problem became even more important when we started using the Million Songs data. With over 100 million of songs, it was important to optimize for both speed and memory usage. We tried a relational database techniques for storing our data.

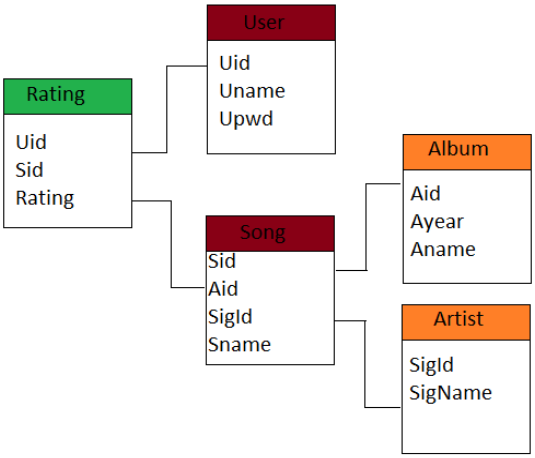

Fig. 1 Data structure.

One of the most natural ways to store a large amount of data is a relational database. Programs can access the database using the structured query language, which supports a wide variety of queries.

IV. ALGORITHMS

A] Incremental critiquing:

It assume conversational recommender system in Which Each recommendation session starts with an initial user query resulting in the retrieval of a song S. with the highest rating. The user has the opportunity to accept the current result, thereby ending the recommendation session, or to critique it as a way to generate the next cycle. The incremental critiquing process consists of following steps.

1. The new song S is recommended to the user based on The current query qi and previous critiques.
2. The user reviews the recommendation and take action Based on critiquing.
3. The query qi is checked for the next cycle Incremental critiquing.
4. The user model U = {U1; . . . ; Un } is updated by Adding the last critique and removing all the critiques

Finally, the recommendation process terminates either when the user retrieves a suitable case or when it explicitly finishes the process.

B] K-Mean Similarity Algorithm

The basic step of k-means clustering is simple. In the beginning we determine number of cluster K and we assume the centroid or center of these clusters. We can take any random objects as the initial centroids or the first K objects in sequence can also serve as the initial centroids

Then the K means algorithm will do the three steps below until convergence Iterate until *stable* (= no object move group):

1. Determine the centroid coordinate.
2. Determine the distance of each object to the centroids.
3. Group the object based on minimum distance.

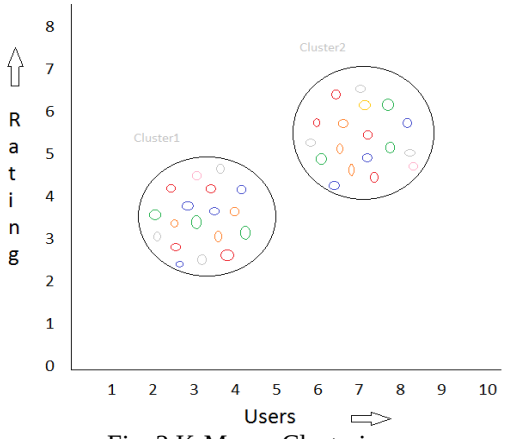

Fig. 2 K-Means Clustering

Suppose we have several objects of rating and each object we are considering two attributes 1)rating value 2)User Id. Our goal is to group these objects into K=3 group of Rating based on the two features (UserId and Rating Value).

V. FUNCTIONAL MODEL AND DESCRIPTION

1) Add New Items: This function inserts new items by indexing them like the previously saved items. This functionality may be achieved by a SQL. After adding this item to database, this item will also ranked according to its album id for users who listens that album.

2) Get Customer Profile: This function provides customer profile information and statistics to supplier of the whole system. Actually, this function provides the whole information needed to make a recommendation.

3) Monitor User Actions/Feedback: The system should monitor user actions as the system receives implicit feedback through it.

4) View Recommendations: It is to make user tend to make use of an opportunity of a recommendation, monitoring is an important point. Here Server uses this functionality to get attention of the user on a recommended item by displaying it in an attractive way.

5) Provide Feedback: the users must be able to read reviews and provide feedback to the system. This feedback is crucial to provide more accurate recommendations in time.

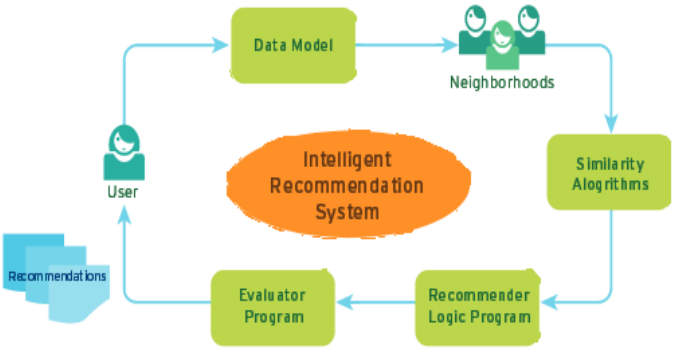

FIG. 3 FUNCTIONAL DATA FLOW MODEL

VI. TESTING METHODOLOGY

We tested our algorithms by removing portions of the dataset then trying to predict what the users would have rated the songs. This is a simple test which allows us to see how the user actually recommended music. In order to determine the amount of success, we used a algorithm. The algorithm that is, the basic prediction method is simply using the global average of rating to calculate predicted value. This is a simple algorithm, so it is used for the implementation.

A] Million Songs Dataset Performance

The Million Song dataset was about 1.5GB, with 100 billion entries. Also, their testing set had near about 10000 of predictions they wanted. Thus, we only try out two algorithms on the dataset. Of the two we tried, both performed better than global average. And other is the incremental critiquing algorithm which performed better for the variety of items.

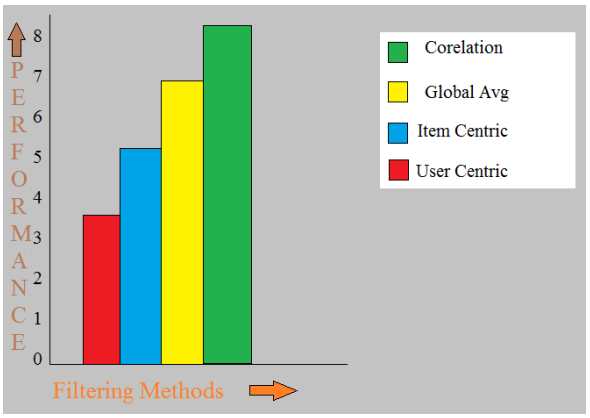

Fig. 3 Performance Analysis.

We tried with a number of different types of algorithms for recommender systems. The Evaluation of algorithm is based on the following criteria. In above graph greater the value better the performance.

1) Accuracy

This is obvious one of Music song we want our recommender to make good recommendations. So we want it to perform better than any "dumb" recommendation algorithm which just uses global data, such as an average rating for music.

2) Speed/Scalability

Most of the recommender systems work in online setting, and so that they can start recommending items for a user. This means that the algorithm cannot take long time to make any predictions it has to work fast. Directly related to speed is the scalability of the algorithm. Again, systems in online setting can have a large dataset. The algorithm must maintain its speed even if there are many millions of ratings.

3) Easy to Update

The datasets behind recommender systems are constantly updating with new ratings from users. As such, the algorithm must handle this updated information quickly. If the algorithm required some hours to run completely, it may miss its chance to make recommendations based on new information quickly.

VII. CONCLUSIONS

In this paper, we proposed an Incremental critiquing technology that increases retrieval quality in a recommender system which is a powerful technique that help the users to find what they want and ecommerce companies to improve their sales. Each recommender algorithm gives the different result with respect to different datasets. This paper we analyse the behaviour of different recommender algorithms. On evaluation of obtained results, we can declare that a RS should have different models to make recommendations to different groups of users.

REFERENCES

[1] A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In Proceedings of the Symposium on Principles of Database Systems, pages 223-228, 2004.

[2] D. Aha, L. Breslow, and H. Munoz-Avila, "Conversational Case-Based Reasoning," Applied Intelligence, vol. 14, pp. 9-32, 2000.

[3] B. Smyth, "Case-Based Recommendation," The Adaptive Web, P. Brusilovsky, A. Kobsa, and W. Nejdl, eds., pp. 342-376, Springer-Verlag, 2007.

[4] J.B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative Filtering Recommender Systems," The Adaptive Web, P. Brusilovsky A. Kobsa, and W. Nejdl, eds., pp. 291-324, Springer-Verlag, 2007.

[5] L. Sweeney. k-anonymity: A model for protecting privacy. International Journal on Uncertainty, Fuziness, and Knowledge-based Systems, 10(5):557-570, 2002.

[6] R. Burke, "Interactive Critiquing for Catalog Navigation in ECommerce," Artificial Intelligence Rev., vol. 18, nos. 3/4, pp. 245-267, 2002.

[7] F. Ricci and N. Nguyen, "Mobyrek: A Conversational Recommender System for on-the move Travelers," Destination Recommendation Systems.

[8] R. Burke, "Interactive Critiquing for Catalog Navigation in ECommerce," Artificial Intelligence Rev., vol. 18, nos. 3/4, pp. 245-267, 2002.

