Separation of Users' Loved Songs from Other Songs
CS134 Project Milestone Write-up

Kan Wu

May 10, 2011

1 Introduction

Yahoo! Music has amassed billions of user ratings for musical pieces. When properly analyzed, the raw ratings encode information on how songs are grouped, which hidden patterns link various albums, which artists complement each other, and above all, which songs users would like to listen to.

In this course project, I am going to participate in the competition of KDD Cup, which is sponsored by Yahoo! Music. The competition provides us both train data and test data (below). The train data includes the songs associated with the score rated by the users and the relationships between songs, albums, artists and genres. In the test data, users and songs are given in pairs, and we want to use learning algorithms to separate the songs, which are really favored by the specific user, from other songs.

At each dataset, user rating data is grouped by user. First line for a user is formatted as:

$$< UserID > | < \#UserRatings >$$

Each of the next $< \#UserRatings >$ lines describes a rating by $< UserID >$. Rating line format:

$$< ItemId >< Score >$$

The scores are integers lying between 0 and 100, and are *withheld* from the test set. All user id's and item id's are consecutive integers.

2 Collaborative Filtering

Collaborative Filtering (CF) is the most popular algorithm family in *recommendation system.* The basic idea of CF-based algorithm is to provide the rating prediction based on the opinions of other like-minded users.

In this project, I will implement the CF algorithm and predict the ratings of items given in the test data. And I will pick-up the three higher predicted score items as the favored songs of the user.

There are two main categories of CF algorithm - *User-based* and *Item-based* algorithms.

- **Memory-based Collaborative Filtering Algorithms.** These systems employ statistical techniques to find a set of users, which is also called *neighbors,* that have a history of similar interest with the target user. Once a neighborhood of users if formed, these systems use different algorithms to combine the rating history of neighbors to produce a prediction for the target user.

- **Item-based Collaborative Filtering Algorithms.** The item-based approach applies association rule discovery algorithm to find association between co-rated items. And compute the expected value of a user prediction based his/her ratings on other items and the strength of the association between them.

3 Implementation

Based on the experiment results of other researchers, it suggested that item-based algorithms provide better prediction quality than user-based algorithms.

The item-based approach looks into the set of items the target user has rated and computes how similar they are to the target item i and then selects k most similar items $\{i_1, \ldots, i_k\}$. So I want to compute their similarities $\{s_{i1}, \ldots, s_{ik}\}$ at first, the prediction is then computed by taking a weighted average of the target user's ratings on these similar items.

3.1 Item Similarity Computation

Once critical step in the item-based collaborative filtering algorithm is to compute the similarity between items and then to select the most similar items. The basic idea in similarity computation between two items i and j is to first isolate the users who have rated both of these items and then apply to similarity computation technique to determine the similarity s_{ij}.

In my project, I use *Adjusted Cosine Similarity* to calculate the similarity between items. This method is the modification of *Pearson-r* correlation. The main drawback of the basic measure is that different rating

scale between different users are not taken in to account. The adjusted cosine similarity offset this drawback by subtracting the corresponding user average from each co-rated pair.

Formally the similarity between item i and j using this method is given by

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \bar{R_u})(R_{u,j} - \bar{R_u})}{\sqrt{\sum_{u \in U}(R_{u,i} - \bar{R_u})^2}\sqrt{\sum_{u \in U}(R_{u,j} - \bar{R_u})^2}} \quad (1)$$

Here $R_{u,i}$ denotes the rating of user u on item i, $\bar{R_u}$ is the average of the u-th user's ratings.

In the practical implementation, I decide to pre-compute the similarities between all items i and j to avoid the repeat computation of repeat similarities. Then I found there needs $O(\#Items * \#Items)$ space to store the similarity of all item pairs, and there needs $O(\#Items * \#Items*)$ time to calculate the similarity of all item pairs.

My solution is to analyze the test data at first, and then write all the similarity-needed item pairs into the data file.Then I just need to calculate the similarities of the item pairs that appear in this list.

Now I am still calculating the needed similarities on the cluster.

3.2 Prediction Computation

Once we isolate the most similar item set based on the similarities calculated, the next step is to look into the target users ratings and obtain predictions.

Weighted Sum computes the prediction on item i for a user u by computing the sum of the ratings given by the user on the items similar to i. Each ratings is weighted by the corresponding similarity $s_{i,j}$ between items i and j. Formally we can denote the prediction $P_{u,i}$ as

$$P_{u,i} = \frac{\sum_{\text{all similar items},N}(s_{i,N} * R_{u,N})}{\sum_{\text{all similar items},N}(|s_{i,N}|)} \quad (2)$$

Regression is similar to weighted sum method but instead of directly using the ratings $R_{u,N}$ of similar items it uses an approximation of the ratings $R'_{u,N}$ based on regression model. The linear regression model can be expressed as

$$\bar{R'_N} = \alpha \bar{R_i} + \beta + \epsilon \quad (3)$$

In the regression model, R_i and R_N respectively denotes the vectors of the target item i and the similar item N.

4 Time-line

- Read paper and study basic learning algorithm, *04/04 - 04/24*
- Write python to process the data, *04/25 - present*
- Use regression to compute the prediction, *05/09 - 05/15*
- Try different methods separate the items in test data, *near future*
  - Current method: pick-up the three songs with the highest predicted rating
  - Possible method: pick-up the three songs with the most similar to the rated songs.

References

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17, June 2005.

[2] J. S. Breese, D. Heckerman, and C. M. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *UAI'98*, 1998.

[3] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 230–237, New York, NY, USA, 1999. ACM.

[4] Yahoo! Labs. Kdd cup 2011, 2011.

[5] Badrul Sarwar, George Karypis, Joseph Konstan, and John Reidl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, WWW '01, pages 285–295, New York, NY, USA, 2001. ACM.