Similarity Metric Learning for Item-based Collaborative Filtering

CS 134 Project Final Write-up

Kan Wu[1]

Dartmouth College, Hanover NH 03755, USA
wukan@cs.dartmouth.edu

Abstract. In this project, I participated in the KDD Cup 2011. The goal of the competition is to separated each user's favored songs. I implement an item-based collaborative filtering system that provide the prediction based on the rating score of each songs for the same user. The project is totally implemented by *Python* and the experiment result (19% error percentage for best result) shows the correctness of my methods.

1 KDD Cup 2011

The human tastes in music are quite diverse, for our difference in personalities and cultures. Yahoo! Music has collected billions of user ratings for musical pieces. After analyzing the hidden pattern links between various albums, artists and songs, we should be able to find which songs users would like to listen to. Such an exciting problem introduces new data mining challenges. The KDD Cup 2011 releases over 300 million ratings performed by over 1 million anonymous users.

In track2 the competitors are required to separate track scored highly by specific users from tracks not scored by the. The test set includes six items per user, three of which were favored (score 80 or higher) by the user and three were not rated by the user. Our goal is to classify each item as either rated or not rated by the user (1 or 0 respectively).

The dataset is split into two subsets: **Train data** and **Test data**. At each subset, user rating data is grouped by user. First line for a user is formatted as: `<UsedId>|<#UserRatings>\n`. Each of the next `<#UserRatings>` lines describes a single rating by `<UsedId>`. Rating line format: `<ItemId>\t<Score>\n` The scores are integers lying between 0 and 100, and are withheld from the test set.

Train data	**Test data**
UserId|#UserRatings	UserId|#UserRatings
ItemId Score	ItemId
ItemId Score	ItemId
ItemId Score	ItemId
ItemId Score	ItemId
ItemId Score	ItemId
ItemId Score	ItemId
...	...

The evaluation criterion is the error rate, which is the fraction of wrong predictions.

2 Recommendation System

Recommendation systems appears as an independent research area in the mid-1990s. In its common formulation, the recommendation problem is reduced to the problem of estimating ratings for the items that have not been rated by a user. Intuitively, this estimation is usually based on the previous ratings given by this user to other items and on some other information. Once we can predict ratings for the yet unrated items, we can recommend to the user the item(s) with the highest estimated rating(s).

The new ratings of the not-yet-rated items can be estimated in many different ways using methods from machine learning, approximation theory, and various heuristics. Moreover, recommender systems are usually classified into the following categories, based on how recommendations are made:

- **Content-based methods**: The user will be recommended items similar to the ones the user preferred in the past;
- **Collaborative Filtering methods**: The user will be recommended items that people with similar tastes and preferences liked in the past;

2.1 Content-based Recommendation System

In content-based recommendation methods, the prediction function $r(u, i)$ of item i for user u is estimated based on the utilities $r(u, i')$ assigned by user u to items i' that are "similar" to item i. For example, in a movie recommendation application, in order to recommend movies to user u, the content-based recommender system tries to understand the properties among the movies user u has rated highly in the past (specific actors, directors, genres, subject matter, etc.). Then, only the movies that have a high degree of similarity to whatever the user's preferences are would be recommended.

However, content-based techniques are limited by the features that are explicitly associated with the objects that these systems recommend. Therefore, in order to have a sufficient set of features, the content must either be in a form that

can be parsed automatically by a computer (e.g., text) or the features should be assigned to items manually. The problem is these features are often not practical to assign attributes by hand due to limitations of resources.

2.2 Collaborative Filtering

Unlike content-based recommendation methods, collaborative filtering recommendation systems try to predict the rating of items for a particular user based on the items previously rated by other users. More formally, the prediction function $r(u, i)$ of item i for user u is estimated based on the utilities $r(u', i)$ assigned to item i by those users u' who are "similar" to user u. For example, in a movie recommendation application, in order to recommend movies to user u, the collaborative recommender system tries to find the "neighbors" of user u', i.e., other users that have similar tastes in movies (rate the same movies similarly). Then, only the movies that are most liked by the "neighbors" of user u' would be recommended.

3 Item-based Collaborative Filtering

In this project I implement the item-based collaborative filtering recommendation system to predict the given pairs of users and items in the test dataset. The item-based approach looks into the set of items the target user u has rated and computes how similar they are to target item i and then selected k most similar items $\{i_1, i_2, \ldots, i_k\}$. At the same time their corresponding similarities $\{s_{i1}, s_{i2}, \ldots, s_{ik}\}$ are also computed. Once the most similar items are found, the prediction is then computed by taking a weighted average of the target user's ratings on these similar items. I will describe the two steps, the **similarity computation** and the **prediction generation** in details later.

3.1 Similarity Computation

The first 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 firstly find the users who have rated both item i and j and then to apply a similarity computation technique to determine the similarity s_{ij}. **Figure 1** shows this process; here the matrix rows represents users and the columns represents items.

There are a number of different methods to compute the similarity between items. Here we present two such methods. They are correlation-based similarity and adjusted-cosine similarity.

Correlation-based Similarity In this case, the similarity between item i and j is measured by computing the *Pearson-r* correlation. To make the correlation accurate we must find the co-rated users, the users rated both i and j as shown

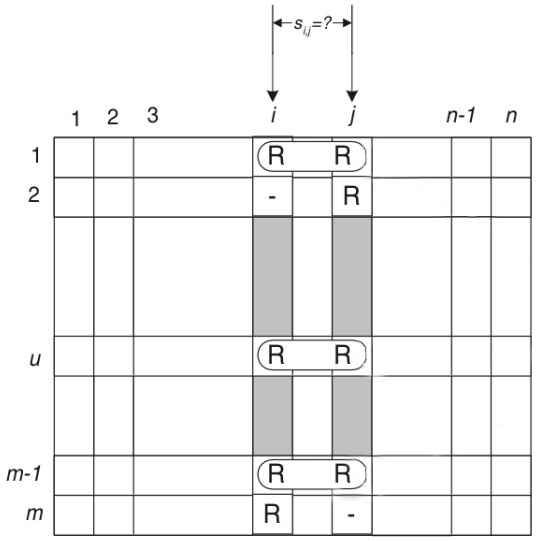

Fig. 1. Similarity Computation

in **Figure 1**. Let the set of users who rated both i and j are denoted by U then the correlation similarity is give by

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \bar{R}_i)(R_{u,j} - \bar{R}_j)}{\sqrt{\sum_{u \in U}(R_{u,i} - \bar{R}_i)^2}\sqrt{\sum_{u \in U}(R_{u,j} - \bar{R}_j)^2}}$$

Here $R_{u,i}$ denotes the rating of user u on item i, $\bar{R}_i$ is the average rating of the i-th item.

Adjusted Cosine Similarity The fundamental difference between the similarity computation in user-based collaborative filtering and item-based collaborative filtering is that in case of user-based CF the similarity is computed along the rows of the matrix but in case of the item-based CF the similarity is computed along the columns. Computing similarity using basic correlation measure in item-based CF has one important drawback - the differences in rating scale between users are not taken into account. The adjusted cosine similarity offsets this drawback by subtracting the corresponding user average from each co-rated pair. Finally the formula similarity between item i and j is given by

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

Here $\bar{R}_u$ denotes the average of the u-th user's ratings.

3.2 Prediction Computation

The second step in collaborative filtering system is to generate the output in terms of prediction. Once we find the set of most similar items based on the similarity measure, the next step is to look into the target users' ratings and use a technique to provide predictions. Here I implement the **weighted sum** method.

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

$$P_{u,i} = \frac{\sum_{\text{most similar items},N}(s_{i,N} \times R_{u,N})}{\sum_{\text{most similar items},N}(|s_{i,N}|)}$$

4 Experiments

4.1 Experimental Parameters

- **Neighborhood Size**, the size of the neighborhood has significant impact on the prediction quality. To determine the sensitivity of this parameter, I performed an experiment where we varied the number of neighbors to be used and computed the error percentage.
- **User Activity**, when we computer the similarity between two items the original measure is just summing up every user's contribution equally. I argue that user's contribution should be negatively to the number of the user's total ratings. Instead of counting each user's contribution equally, I introduce the factor below:

  $$f_u = \frac{1}{\log(\alpha + N_u)}$$

  Here N_u denotes the number of ratings given by the user u.

4.2 Experimental Results

In this project, I have implemented both the original correlation based similarity measure and the correlation with the user activity factor. In the experiments I run the experiments on both two similarity computation on several different neighborhood size, and the result is presented in the **Figure 2**.

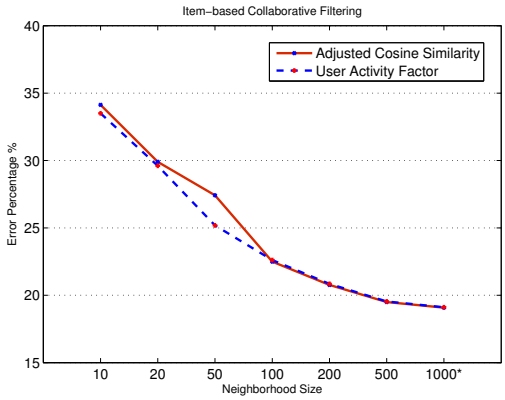

Fig. 2. Experimental Results

From the result we can see that when the neighborhood size is less than 100, the performance of correlation with user activity factor is over the original adjusted cosine similarity. We can conclude that when the size of neighborhoods is limited, the measure with user activity provides us a better prediction accuracy. When the size of neighborhood size increases, the impact of the measure method decreases and the results is likely to aggregate from more similar items.

References

1. J.L. Herlocker, J.A. Konstan, A. Borchers, and J. Riedl. An Algorithmic Framework for Performing Collaborative Filtering In *SIGIR 1999*
2. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-Based Collaborative Filtering Recommendation Algorithms In *WWW 2001*
3. G. Adomavicius and A. Tuzhilin. Towards the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions in *IEEE Transactions on Knowledge and Data Engineering 2005*
4. Yahoo! Labs. KDD Cup from Yahoo! Labs http://kddcup.yahoo.com/