Collaboration Filtering using K-Mean Algorithm  
Smrity Gupta  
Smrity_0501@yahoo.co.in  
Department of computer Science and Engineering  
University of RAJIV GANDHI PROUDYOGIKI SHWAVIDYALAYA, BHOPAL

Abstract:
Recommender systems apply data analysis techniques to the problem of helping users find the items they would like to purchase at E-Commerce sites by producing a predicted likeness score or a list of top-N recommended items for a given user. We apply Clustering algorithms for finding nearest similar item. To finding nearest item for this we use C++ language. We apply improved K-mean algorithms method on preprocessed data. Finally we proposed a method that can increase accuracy based on previous K-mean algorithms.

Introduction:
Recommender systems apply knowledge discovery techniques to the problem of making personalized recommendations for information, products or services during a live interaction. These systems, especially the k-nearest neighbor collaborative filtering based ones, are achieving widespread success on the Web. The tremendous growth in the amount of available information and the number of visitors to Web sites in recent years poses some key challenges for recommender systems. (Aggarwal et al., 1999). These are: producing high quality recommendations, performing many recommendations per second for millions of users and items and achieving high coverage in the face of data sparsity [Aggarwal et al., 1999]. In traditional collaborative filtering systems the amount of work increases with the number of participants in the system. New recommender system
technologies are needed that can quickly produce high quality recommendations, even for very large-scale problems. To address these issues we have explored item-based collaborative filtering techniques. Item-based techniques first analyze the user-item matrix to identify relationships between different items, and then use these relationships to indirectly compute recommendations for users. We analyze different item-based recommendation generation algorithms. We look into different techniques for computing item-item similarities (e.g., item-item correlation vs. cosine similarities between item vectors) and different techniques for obtaining recommendations from them (e.g., weighted sum vs. regression model). Finally, we experimentally evaluate our results and compare them to the basic k-nearest neighbor approach. Our experiments suggest that item-based algorithms provide dramatically better performance than user-based algorithms, while at the same time providing better quality than the best available user-based algorithms.

Related Work:

In this section we briefly explain the work related to my work;

Collaboration Filtering:

Collaborative Filtering systems can produce personal recommendations by computing the similarity between your preference and the one of other people. One of the most promising such technologies is collaborative filtering [shardanand et al., 1995]. Collaborative Filtering works by building a database of preferences for items by users. A new user, Neo, is matched against the database to discover neighbors, which are other users who have historically had similar taste to Neo. Items that the neighbors like are then
recommended to Neo, as he will probably also like them. Collaborative Filtering has been very successful in both research and practice, and in both information filtering applications and E-commerce applications. However, there remain important research questions in overcoming two fundamental challenges for Collaborative Filtering recommender systems.

The first challenge is to improve the scalability of the Collaborative Filtering algorithms. These algorithms are able to search tens of thousands of potential neighbors in real-time, but the demands of modern systems are to search tens of millions of potential neighbors. Further, existing algorithms have performance problems with individual users for whom the site has large amounts of information. For instance, if a site is using browsing patterns as indications of content preference, it may have thousands of data points for its most frequent visitors. These "long user rows" slow down the number of neighbors that can be searched per second, further reducing scalability.

- The second challenge is to improve the quality of the recommendations for the users. Users need recommendations they can trust to help them find items they will like. Users will "vote with their feet" by refusing to use recommender systems that are not consistently accurate for them. In some ways these two challenges are in conflict, since the less time an algorithm spends searching for neighbors, the more scalable it will be, and the worse its quality. For this reason, it is important to treat the two challenges simultaneously so the solutions discovered are both useful and practical. In this Thesis we address these issues of recommender systems by applying different approach Item-Based algorithms. The bottleneck in conventional Collaborative Filtering algorithms is the
search for neighbors among a large user population of potential neighbors [Herlocker et al., 1999]. Item-based algorithms avoid this bottleneck by exploring relationships between items first, rather than the relationships between users. Recommendations for users are computed by finding items that are similar to other items the user has liked. Because the relationships between items are relatively static, item-based algorithms may be able to provide the same quality as the user-based algorithms with less online computation.

Collaborative-Filtering-enabled Web sites that recommend books, CDs, movies, and so on, have become very popular on the Internet. Such sites recommend items to a user on the basis of the opinions of other users with similar taste. Recommender systems apply data analysis techniques to the problem of helping users find the items they would like to purchase at E-Commerce sites by producing a predicted likeliness score or a list of top-$N$ recommended items for a given user. Item recommendations can be made using different methods. Recommendations can be based on demographics of the users, overall top selling items, or past buying habit of users as a predictor of future items. Collaborative Filtering (CF) [Shardanand et al., 1995] is the most successful recommendation technique to date. The basic idea of CF-based algorithms is to provide item recommendations or predictions based on the opinions of other like-minded users. The opinions of users can be obtained explicitly from the users or by using some implicit measures. The growth of the Internet has resulted in the availability of a tremendous amount of information and a vast array of choices for consumers. Recommenders systems are designed to help a user cope with this situation by selecting a small number of options to present to the user. They filter and recommend items on the basis of a user
preference model. Among the various types of recommender systems that have been proposed, their filtering techniques fall into two categories: content-based filtering and Collaborative Filtering or social filtering.

**Proposed Work**

In this section we briefly present my work related to Collaborative Filtering, recommender system. Other technologies have also been applied to recommender system, including clustering. Clustering is a division of data into groups of similar objects. Representing the data by fewer clusters necessarily loses certain fine details, but achieves simplification. It models data by its clusters. Data modeling puts clustering in a historical perspective rooted in mathematics, statistics, and numerical analysis.

**Experimental Procedure**

Experimental steps: We started our experiment by dividing the data set into training set and test set. First we have to find the sensitivity of data by different algorithm. to find the sensitivity of data we work with only training data then we further divide the training data into two parts for finding sensitivity of data we randomly choosing training and test data each time and talking MAE (Karypis G., 2000)

K-means is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori. The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result. So, the
better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid.

The algorithm is composed of the following steps:

- Place K points into the space represented by the objects that are being clustered. These points represent initial group centroids.
- Assign each object to the group that has the closest centroids.
- When all objects have been assigned, recalculate the positions of the K centroids.
- Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects into groups from which the metric to be minimized can be calculated.

Fig 5.1: To predict
means \( m_1 \) and \( m_2 \) move into the centers of two clusters.

**Proposed Algorithm**

**Start:** Clustering \_algo(\text{Iteration,Counting,Destination,Center,Check\_result})

**Step1:** Initialize: \text{Iteration}=0;\text{Counter}=0; a by Two dimension Array; Center of the object, new center of the new object; Final Center of the resultant Object;

**Step2:** X position and Y position is defined;

**Step3:** Initialize Graphics Mode.

**Step4:** Now insert Threshold Value and insert no of objects

**Step5:** loop1: until value not equal to 0;

\begin{align*}
\text{Draw the Circle according to the Object value: show the cluster;}
\end{align*}

**Step6:** loop2 If Circle is \(<=3\) then don’t Change the iteration;

**Step 7:** loop3 An Iteration =3|| iteration=2;

\begin{align*}
\text{Else if circle}\geq3 \text{ Then iteration++;Or } \text{Iteration}=\text{iteration}+1;
\end{align*}

**Step8:** Calculate the Cluster and center points of the object

\begin{align*}
\text{X direction} &= (a[\text{loop3}][0]-\text{Center}[\text{loop}][0])*(a[\text{loop3}][0]-\text{center}[0]); \\
\text{Y direction} &= (a[\text{loop3}][1]-\text{center})*(a[\text{loop}][1]-\text{center});
\end{align*}

\text{Compute the square root of x and Y result;End loop.}

**Step9:** Rescanning the object and results

\begin{align*}
\text{Loop1: Compare the threshold value with Cluster destination Increment the Cluster}
\end{align*}

\text{Loop2: Set the Cluster position:}

\text{Now again Calculate X and Y direction Values Go to Step 5.}
Step 10: Again compute the square root of x and Y result. If Square Root < threshold value the
Max value = new Square root Circle radius = Max value; Print :all Cluster:

Step 11: Merge the object with all cluster

Loop 1 start Initialize value 1.
Loop 2 start Initialize value 2;
Calculate X and Y direction Value Go to step 5;
New Square root of x and y Value New center = new square root;

Step 12: If object < previous result and Cluster < previous Cluster

Then PRINT New result:
PRINT destination Matrix
PRINT evolution Matrix;
PRINT new comparing Result and final clusters
Again PRINT “Iteration and Final Comparing Values (Errors)

Result Analysis and Comparison

In this chapter we propose a prediction which shows that improved clustering algorithm shows better result than other basic approach.

Experimental Setup:

We experimented with the MovieLens1, EachMovie2, and ook-crossing3 data sets. While we report only the Movie Lens results (out of space considerations), the model behaves consistently across the three data sets. The Movie Lens data set contains
100,000 ratings (1-5 scales) from 943 users on 1682 movies (items), where each user has rated at least 20 items. To test on different number of training users, we selected the users in the data set at random into a training user set (100, 200, 300, training users, respectively) and the remaining users into a test user set.

Table 5.1: Calculated Values

<table>
<thead>
<tr>
<th>Neighborhood</th>
<th>MAE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.5</td>
<td>0.8</td>
</tr>
<tr>
<td>0.1</td>
<td>0.6</td>
<td>0.9</td>
</tr>
<tr>
<td>0.1</td>
<td>0.8</td>
<td>1.2</td>
</tr>
<tr>
<td>0.2</td>
<td>0.9</td>
<td>1.5</td>
</tr>
<tr>
<td>0.3</td>
<td>1.0</td>
<td>1.9</td>
</tr>
<tr>
<td>0.5</td>
<td>2.0</td>
<td>2.1</td>
</tr>
<tr>
<td>0.6</td>
<td>2.5</td>
<td>2.4</td>
</tr>
<tr>
<td>0.8</td>
<td>2.9</td>
<td>2.6</td>
</tr>
<tr>
<td>0.9</td>
<td>3.5</td>
<td>2.7</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>2.8</td>
</tr>
<tr>
<td>1.5</td>
<td>5</td>
<td>2.899</td>
</tr>
</tbody>
</table>
Comparison graph 5.1  Sensitivity of parameter x versus mean absolute error

Series 1: shows improved clustering line chart
Series 2: shows previous clustering line chart

We have drawn two conclusions from the result. First proposed item based algorithm provide better quality then previous one. Second the old clustering based algorithm perform better with very sparse data set, but as we add more data the quality goes down.
Table 5.2  Proposed Algorithms

<table>
<thead>
<tr>
<th>Neighborhood</th>
<th>MAE</th>
<th>MAE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.8</td>
<td>1</td>
<td>0.5</td>
</tr>
<tr>
<td>0.1</td>
<td>0.9</td>
<td>1.23</td>
<td>0.6</td>
</tr>
<tr>
<td>0.2</td>
<td>1.2</td>
<td>1.4</td>
<td>0.8</td>
</tr>
<tr>
<td>0.3</td>
<td>1.5</td>
<td>1.8</td>
<td>0.9</td>
</tr>
<tr>
<td>0.5</td>
<td>1.9</td>
<td>2.1</td>
<td>1.0</td>
</tr>
<tr>
<td>0.6</td>
<td>2.1</td>
<td>2.4</td>
<td>2.0</td>
</tr>
<tr>
<td>0.8</td>
<td>2.4</td>
<td>2.7</td>
<td>2.5</td>
</tr>
<tr>
<td>0.9</td>
<td>2.6</td>
<td>2.8</td>
<td>2.9</td>
</tr>
<tr>
<td>1</td>
<td>2.7</td>
<td>2.8</td>
<td>3.5</td>
</tr>
<tr>
<td>1.5</td>
<td>2.8</td>
<td>2.9</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>2.899</td>
<td>2.8</td>
<td>5</td>
</tr>
</tbody>
</table>

Comparison graph 5.2  Sensitivity of mean absolute error versus parameter

Series 1: shows improved clustering line chart

Series 2: shows previous clustering line chart

Series 3: shows user based line chart
In this section we present our experimental results of applying item-based collaborative filtering techniques for generating predictions. Our results are mainly divided into two parts-quality results and performance results. In assessing the quality of recommendations, we first determined the sensitivity of some parameters before running the main experiment.

**Performance Results**

Having clearly established the superior quality of item-based algorithms over the user-based ones, we focus on the scalability challenges item-based similarity is more static and allows us to pre compute the nearest items. These methods although save the time require an $O(n^2)$ space for $n$ times.

**Conclusion**

In this work we have proposed a recommender systems based on “Item Based Collaboration Filtering”. This system helps the users to find items they want to buy. For this we developed efficient Clustering algorithms which are able to generate most similar items efficiently and with high accuracy. In development of our new algorithm we have applied iterative approach on k-means algorithm to improve accuracy of the Clusters. In each iteration algorithm generate the clusters and test the accuracy of the Clusters. This improved algorithm shows how much error is reduced, by previous one. This is benefit of my algorithms which shows Error and calculates it. After calculating again testing Filtering process is started. This result is computed on Few Data which are predefined in the form of array. The result can be vary if Change the data from the Database.
References:


