ABSTRACT

Recommendations that are personalized help the users in getting the list of items that are of their interest in e-commerce sites. Majority of recommender systems use Collaborative Filtering techniques to generate recommendations to their users. This project implements an information filtering technique called as Collaborative Filtering for generating personalized recommendations in movies for user. Collaborative Filtering is of two types, namely, collaborative filtering based on users and collaborative filtering based on items. Collaborative Filtering based on users is more expensive computationally but it produces better results. Collaborative Filtering based on users is not preferred because it encounters the problems of Scalability when the number of users increases. Therefore, we use item-based Collaborative Filtering which is an alternative method. Collaborative Filtering, which is based on items uses two techniques- Pearson correlation technique and Adjusted cosine technique for calculating the similarity between items and to generate recommendations to users. In this Project both the above techniques are used and to measure the accuracy of the predictions generated by these techniques, Root Mean Square Error is computed.
## CONTENTS

<table>
<thead>
<tr>
<th>Contents</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1. INTRODUCTION</strong></td>
<td>1</td>
</tr>
<tr>
<td>1.1 Recommender System</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Collaborative filtering</td>
<td>1</td>
</tr>
<tr>
<td>1.3 Data Set</td>
<td>2</td>
</tr>
<tr>
<td>1.4 Scalability</td>
<td>4</td>
</tr>
<tr>
<td>1.5 Correlation Analysis</td>
<td>4</td>
</tr>
<tr>
<td><strong>2. LITERATURE REVIEW</strong></td>
<td>5</td>
</tr>
<tr>
<td>2.1 Existing Methods</td>
<td>8</td>
</tr>
<tr>
<td><strong>3. NARRATIVE</strong></td>
<td>9</td>
</tr>
<tr>
<td>3.1 System Overview</td>
<td>9</td>
</tr>
<tr>
<td>3.2 System Features</td>
<td>9</td>
</tr>
<tr>
<td>3.3 Problem Statement</td>
<td>11</td>
</tr>
<tr>
<td>3.4 Proposed System</td>
<td>11</td>
</tr>
<tr>
<td>3.5 Software Requirements Specification</td>
<td>11</td>
</tr>
<tr>
<td>3.6 Functional Requirements</td>
<td>12</td>
</tr>
<tr>
<td>3.7 Non Functional Requirements</td>
<td>13</td>
</tr>
<tr>
<td>3.8 Usecase Diagram</td>
<td>14</td>
</tr>
<tr>
<td><strong>4. DESIGN</strong></td>
<td>16</td>
</tr>
<tr>
<td>4.1 Detailed Design</td>
<td>16</td>
</tr>
<tr>
<td>4.2 Architecture of Proposed System</td>
<td>19</td>
</tr>
<tr>
<td>4.3 Algorithm</td>
<td>19</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table No.</th>
<th>Name</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>User-item rating</td>
<td>2</td>
</tr>
<tr>
<td>1.2</td>
<td>Different Data Sets</td>
<td>3</td>
</tr>
<tr>
<td>3.1</td>
<td>Use Case</td>
<td>15</td>
</tr>
<tr>
<td>4.1</td>
<td>User-item rating</td>
<td>20</td>
</tr>
<tr>
<td>6.1</td>
<td>Test Case 1</td>
<td>28</td>
</tr>
<tr>
<td>6.2</td>
<td>Test Case 2</td>
<td>30</td>
</tr>
<tr>
<td>6.3</td>
<td>Test Case 3</td>
<td>31</td>
</tr>
<tr>
<td>6.4</td>
<td>Test Case 4</td>
<td>33</td>
</tr>
<tr>
<td>6.5</td>
<td>Test Case 5</td>
<td>35</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure No.</th>
<th>Name of the Figure</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>MovieLens</td>
<td>3</td>
</tr>
<tr>
<td>3.1</td>
<td>Use case diagram</td>
<td>15</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Block diagram</td>
<td>17</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Flow chart</td>
<td>18</td>
</tr>
<tr>
<td>4.1.3</td>
<td>System Architecture</td>
<td>19</td>
</tr>
<tr>
<td>4.2</td>
<td>Pearson Similarity Formula</td>
<td>20</td>
</tr>
<tr>
<td>4.3</td>
<td>Adjusted Cosine Similarity Formula</td>
<td>20</td>
</tr>
<tr>
<td>4.4</td>
<td>Prediction Formula</td>
<td>21</td>
</tr>
<tr>
<td>4.5</td>
<td>Class Diagram</td>
<td>23</td>
</tr>
<tr>
<td>4.6</td>
<td>Sequence Diagram</td>
<td>24</td>
</tr>
<tr>
<td>6.1</td>
<td>Test Case1</td>
<td>29</td>
</tr>
<tr>
<td>6.2</td>
<td>Test Case2</td>
<td>31</td>
</tr>
<tr>
<td>6.3</td>
<td>Test Case3</td>
<td>32</td>
</tr>
<tr>
<td>6.4.1</td>
<td>Test Case4.1</td>
<td>34</td>
</tr>
<tr>
<td>6.4.2</td>
<td>Test Case4.2</td>
<td>34</td>
</tr>
<tr>
<td>6.5</td>
<td>Test Case5</td>
<td>36</td>
</tr>
<tr>
<td>6.6.1</td>
<td>Base Case</td>
<td>37</td>
</tr>
<tr>
<td>6.6.2</td>
<td>Test Case</td>
<td>38</td>
</tr>
</tbody>
</table>
6.6.3 Program to calculate rmse error 39
6.6.4 RMSE error 40
7.1 Generating Recommendations-1 41
7.2 Generating Recommendations-2 42
7.3 Viewing Search Results-1 43
7.4 Viewing Search Results-2 44
7.5 Calculating Error rates 45
8.1 Error rates 46
INTRODUCTION

1.1 Recommender Systems

With the increase in the use of e-commerce sites, it has become very easy for the users to find the items of their interest without wasting a lot of time. Websites like Amazon and Ebay are examples for Recommender systems which provide recommendations to the users based on their search history and purchase history. Recommender systems provide recommendations of almost all the items ranging from books to movies to music. FaceBook and Twitter are also recommender sites which provide recommendations for friends. NetFlix.com is very famous as a movie recommender website. Yahoo News and Google news are very famous for news [3].

When a user tries to find an item using search engines, for example, Google Search engine and Yahoo Search engine, the user needs to type the exact name of the item. The data in the internet is huge which makes it very difficult for the user to find the items of his interest. Hence, there is a need for a system which learns the likes and dislikes of the user and generates recommendations based on his interest [1]. Many algorithms need to be used while designing a recommender system.

Recommender systems employ Information Filtering technique that focuses on providing the recommendations of the items to the users that are likely to be of the users interest [2]. A recommender system is defined as: “if \( U \) is the set of users and \( I \) is the set of all possible items that can be recommended, then there exists a function from \( U \times I \) to \( R \) where \( R \) is a totally ordered set of nonnegative integers or real numbers within a certain range”.

1.2 Collaborative Filtering

The main use of Collaborative Filtering methods is in the field of Business to consumer e-commerce where the recommendations are provided to the user by the owner. The owner provides recommendations to the customer based on his search history and past purchase history
Collaborative Filtering techniques are used in the e-commerce sites like Amazon and Ebay which deal with items on a very large scale.

The importance of Collaborative Filtering methods is: guessing the usefulness of a particular item for a particular user depending upon the previous history of other users. Consider Table1.1, where 'U' and 'S' represent user and item respectively. U1, U2,U3, U4, U5 represent users and S1, S2, S3, S4, S5 represent items. Assume that we want to suggest a particular item for 'U4'(user 4), S5(item 5) is recommended for the U4(user 4) who is “similar” to U3 (user 3) as both of them have given high rating for the item S3 User 4 is likely to like item S3. The recommendations are made by comparing the likes and dislikes of users who have taste that is similar to that of the current user.

Table 1.1 user-item rating table

<table>
<thead>
<tr>
<th></th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
</tr>
</thead>
<tbody>
<tr>
<td>U1</td>
<td>3</td>
<td>4</td>
<td></td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>U2</td>
<td>1</td>
<td></td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U3</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>U4</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td></td>
<td>?</td>
</tr>
</tbody>
</table>

1.3 Data Set

A data set is a collection of data, usually presented as in table 1.2. In this project the data is taken from the dataset provided by MovieLens site and is internally stored in the form of arrays. Table 1.2 shows the list of recommender websites- Movielens, Eachmovie and NetFlix are movie recommender sites, Jester joke recommends jokes, BookCrossing recommends books and Newsgroups recommends news. The table shows which information is provided by these websites like the demographic data, ratings, clean-up and non-rating.
Table 1.2 Different data sets

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Demographic</th>
<th>Rating</th>
<th>Cleaned</th>
<th>Non rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movielens</td>
<td>yes</td>
<td>Yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>JesterJoke</td>
<td>yes</td>
<td>Yes</td>
<td></td>
<td>yes</td>
</tr>
<tr>
<td>Each Movie</td>
<td>yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BookCrossing</td>
<td>yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Netflix</td>
<td>yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>News Groups</td>
<td>yes</td>
<td>Yes</td>
<td></td>
<td>yes</td>
</tr>
</tbody>
</table>

The data is taken from Movielens site, shown in figure 1.1. The users can download the information for free from this site and use it for testing. “The data set in the site consist of 943 users and 1682 movies where each user has given ratings for at least 20 movies”. This data given to the program that I have designed which stores this raw data in the form of arrays. The data from the arrays is used for further computations.
1.4 Scalability

The data available in the websites is very huge. The data comprises of both data of the users and the data related to the items available on that website. It is very important for the websites to function properly with the increasing number of items and users day by day. Many items are added to the websites for instance, e-commerce sites daily and many new users are added which makes it very important for a site to scale well. Scalability is an important factor for the websites.

1.5 Correlation Analysis

Correlation analysis focuses on analyzing the relation between two random variables or observed variables. Correlations are important for predicting the relationship between entities. For instance, consider prepaid CPL electricity billing process. This site provides information of the available balance along with the details like in how many days the balance is going to fall to 0$. They estimate the average usage cost based on the previous week's usage details.

Two random variables are said to be dependent if they do not satisfy probabilistic independence condition. “The correlation coefficients, denoted by ’ρ’ or ’r’, for calculating the degree of correlation”. The method that is commonly used is 'Pearson correlation coefficient' method. “Pearson Correlation technique is used to calculate the linear relationship between two variables ” [9]. In this project Pearson correlation technique is used to calculate the relation among two entities and provide suggestions to the user.
LITERATURE REVIEW

The concept of recommender systems emerged in mid-1990s. In past 10 years there has been a tremendous growth in the development of recommender sites. The people using the recommender systems is increasing exponentially which makes it very important for these systems to generate recommendations that are close to the items of users interest.

Jia Zhou and Tiejan Luo [4], has published a paper on Collaborative Filtering applications. The paper describes about the collaborative filtering techniques which were currently in used in that generation. It is stated that the Collaborative Filtering techniques used in that generation could be divided into heuristic-based method and model-based method. The paper discusses about the limitations of the Collaborative Filtering techniques in that generation and suggests some improvements to increase the recommendation capabilities of the systems.

SongJie Gong and Zhejiang [10] proposes that 'personalized recommendation systems' are widely utilized in e-commerce websites to provide recommendations to its users. The paper states that the recommendation systems use Collaborative Filtering technique which has been successful in providing recommendations. A technique to solve the common problems that are encountered in recommender systems namely, sparsity and scalability is suggested in this paper. This paper suggests the recommender system which combines both user clustering and item clustering can be used to provide recommendations. This approach is employed to provide recommendations in this project which makes the prediction smoother. In this approach, item clustering is done using the two techniques Pearson correlation technique and Adjusted cosine similarity technique to find the similarity between the items. Then, users are clustered depending on alikeness between the user targeted and cluster center. Users are grouped into clusters based on their likes and dislikes for an item and every cluster has a center. The authors state that the proposed method is more accurate than the traditional method in generating recommendations.

Robert M Bell and Yehuda Koren [11], state that recommender systems provide recommendations to the users based on past user-item relationship. Based on past user-item relationship the neighbors are computed which makes the prediction easy. The weights of all the neighbors are calculated separately and are interpolated concurrently for many interactions to
provide optimized solution to the problem. The proposed method is stated to provide recommendation in 0.2 milliseconds. The training also takes less time unlike very lengthy time in large scale applications. The proposed method was tested on Netflix data which consisted of 2.8 million queries which was processed in 10 minutes.

Micheal Pazzani [12] discusses about recommending data sources for news articles or web sites after learning the taste of the user by learning his profile. This paper mentions various types of information that can be considered to learn the profile of a user. Based on ratings given by a user to different sites, ratings that other users have given to those sites and demographic information about users the recommendations can be made. This paper describes how the above information can be combined to provide recommendations to the users.

Lee W. S [13], proposed a method in which he assumes that each user is likely to belong to any one of the 'm' clusters and the rating of each user depends upon one of the items that belong to the n cluster of items. Bayesian sequential probability is used to calculate the performance of this method. Heuristic approximations are proposed to Bayesian sequential probability for making experiments on the data set comprising of the ratings of movies. The method suggested is believed to have good performance and tested results are observed to be near to the actual values.

'Liu et al.' [14] suggested a 'hybrid recommendation system' to eliminate the issues of scalability and sparsity. Based on the similarity between rated and non-rated items, weighted average ratings are computed. The Collaborative Filtering techniques are applied on the matrix consisting of user and item ratings. The additional method proposed to solve the scalability problem is dividing the users into different clusters based on the common features. All the users having common taste are present in a group and the target user would belong to any one of the groups.

Yang et al. [15] proposed a novel Collaborative Filtering technique to solve the disadvantages that are found in the current Collaborative Filtering technique. The disadvantages are, the conventional technique is over assertive, removes some important information from the user's profile and often makes conclusions that are not trust worthy. 'Yang et al.' divided the jointly
rated items into three classes based on the difference in the given ratings to the items that are of equal weight in the same class.

Koren [16] stated that the user tastes change with the changing time and developed a recommendation system implementing temporary data into 'factor modeling' and 'item-item neighbor' modeling. [21] The author also suggested a novel 'neighbor model' in which he considers both implicit data and the explicit data.

'Salakhutdinov and Srebro' [17] presented a weighted form to achieve uniformity. To follow standard rules is a prominent attribute of a system for finishing the 'user–item' rating lattice in 'Collaborative Filtering'. Be that as it may, the performance of the system is bad when entrances of the 'user–item' rating framework are inspected non-consistently. Keeping in mind the end goal to take care of the issue, they proposed a follow standard weighted by the recurrence of clients. Takács et al. [20] proposed "a few grid factorization (MF) – based routines (i.e., a regularized MF, a quick semi-positive MF, an exact energy based MF, an incremental variation of MF)". What's more, they plot a remedy technique for MF.

'Shambour and Lu' [18] investigated the 'recommender frameworks' in the setting of the e-government space, and stated a reliable business accomplice proposal e-administrations for little to-medium organizations. They created (1) an understood trust sifting proposal methodology fusing trust engendering and 'Jaccard metric' and (2) a client based 'CF' suggestion methodology upgraded via J'accard metric'. Furthermore, to make the preferences of the two methodologies, they created a mixture trust-upgraded CF suggestion approach 'Tecf' which incorporates both methodologies.

'Yin and Peng' [19] given a reasonable schema to the execution correlation among another suggestion calculation and the delegate/generally acknowledged calculations. “They introduced a relative assessment of eight CF calculations in point of interest (i.e., k-closest neighbor (KNN), solitary worth deterioration (SVD), non-negative MF , weighted non-negative MF, central part examination (PCA) – KNN, Svd–knn, Nmf–knn, and Eigentaste) on two datasets (i.e., Jester and Movielens) by utilizing three quality measurements (i.e., root mean square blunder (RMSE), review, and standardized separation based execution measure (NDPM) ”.
The ultimate goal of all the methods proposed above is to provide accurate recommendations to the users. The main problems encountered by the recommender systems is scalability, sparsity and cold start. The most common methods used in recommender systems is 'Pearson correlation' and 'Adjusted cosine similarity' methods. I have utilized both these methods to measure the alikeness among the items using item clustering and computed the Root Mean Square Error for each of these methods to show the accuracy of the predictions.

2.1 Existing Methods

Amazon and YouTube uses Item Clustering Collaborative Filtering technique. Last.fm and Reddit use Collaborative Filtering technique [21]. Pandora uses Content based approach. Facebook, MySpace, LinkedIn use 'Collaborative Filtering technique' to make friend suggestions, groups and other social connections by observing the network of connections between a user and people present in their connections [22].

Twitter [22] makes use of several signals and in-memory calculations for suggesting whom to follow. Netflix is a hybrid system. “They make recommendations by comparing the watching and searching habits of similar users (Collaborative Filtering) as well as by offering movies that share characteristics with films that a user has rated highly (Content based Filtering)”.

Pandora [22] utilizes the features of a song or artist for tuning into a station which plays music with alike features. Feedback given by the users is used to tune the station, not considering some features like dislikes and gives more importance to the other features like liking a particular song. This is a 'Content Based approach'. With little information Pandora can get started and has limited scope. If the song is alike to the original seed, only then it is suggested.

In Last.fm, [22] a station is created in which songs are suggested to the users based on history of the user and compares the taste of current user against other users. Last.fm plays songs which the users with similar taste listen frequently. This is an example of 'User based Collaborative Filtering' because suggestions are made by considering other users choice.
Last.fm needs huge data related to a particular user to make reliable suggestions. This experiences cold start problem.

**NARRATIVE**

3.1 System Overview

With the increase in the use of e-commerce sites, it has become very easy for the users to find the items of their interest without wasting a lot of time. Websites like Amazon and Ebay examples for Recommender systems which provide recommendations to the users based on their search history and purchase history. Recommender systems provide recommendations of almost all the items ranging from books to movies to music. FaceBook and Twitter are also recommender sites which provide recommendations for friends. NetFlix.com is very famous as a movie recommender website. Yahoo News and Google news are very famous for news [3].

3.2 System Features

The main focus of recommender systems is the user-rating data of the operating system prior to introducing the algorithms of recommender systems, therefore, we define the terms related to this

Information that will be used throughout:

- User: A user is referred to that person who uses the system and the system suggests items to that person. The total set of users belongs to one community.

- Item: An item is an entity in a 'Recommender system' which can be a song or a movie or any product.

- Rating is a number given by a user to a particular item on a particular scale based on liking of that person.

Profile: the ratings given to all items by a particular user is regarded as the user's profile.
• User-Item Matrix: The matrix which contains ratings given by a user to an item is defined as 'User-Item matrix'.

'Recommender systems' consider a set of user profiles; by having a collection of ratings of the available content, this set provides a source of data that is not valid and that can be used to provide recommendations for each user. Ratings can be collected either \textit{explicitly} or \textit{implicitly};

In general, recommendation process has the following phases:

i. Learn likes and dislikes of a user.
ii. Compare him to other users.
iii. Recommend items based on the rating patterns of his and the ones similar to him.

There are many variations possible in the three phases but the purpose remains the same. Learning likes and dislikes is the process of feedback. It is important to learn likes and dislikes in the best possible manner so that results can be personalized as well as possible. There are broadly two forms of feedback, namely, implicit feedback and explicit feedback.

As the name implies, implicit feedback is taken implicitly without asking the user to fill up data. It is done by using web logs, by analyzing user activity in terms of downloads, ratings given etc. Analyzing implicit data is relatively difficult because the behavior of users is not deterministic. Not rating a movie can either mean that he did not like the movie or it can also mean he liked it but was not interested in rating it. Predicting the likes and dislikes this way is error prone and also, data is very sparse. But, in some situations, implicit feedback is the only way possible. For example, in music recommendations, implicit feedback is the way to go as a song has a lot of possible dimensions/traits to it and it is not possible for a user to rate them on all of them or even a subset. Thus, in such circumstances, listening habits of users serve as the best option. On the other hand, movies and books are great examples of explicit feedback as they can be rated on a scale of a number. As each one has its pros and cons, most recommender systems learn about a new user by using explicit feedback and then keep improving the user profile by using implicit feedback.
3.3 Problem Statement

The existing recommender systems only used one similarity measure to chalk out neighbors for users as stated in "Towards an Introduction to Collaborative Filtering" [4]. But, each similarity measure has its own drawbacks and thus, all those drawbacks cause faulty and bad recommendations. Pearson correlation coefficient is the most widely cited in the literature but it is not so suitable when the users being compared are active and inactive [4]. Thus, in such situations, adjusted cosine works better because it takes into consideration the average of the users. Also, when the number of co rated movies is less, which means the movies that are rated by users under consideration, then, there has to be a way to bring normalcy to the comparison [4].

3.4 Proposed System:

The proposed system implements an information filtering technique called as Collaborative Filtering. Broadly, collaborative filtering can be of two types, namely, 'User-based collaborative filtering' and 'item-based collaborative filtering' [4]. Although, 'user based collaborative filtering' has been proved to have produced better results, it is highly computationally expensive and does not scale well with increase in the number of users. Thus, we favored item-based collaborative filtering. We used two different similarity techniques, namely, 'Pearson correlation coefficient' and 'Adjusted cosine similarity' and we also implemented global effects for the dataset. For measuring the error rate, we used Root Mean Square Error (RMSE) [4].

3.5 Software Requirements Specification

Following are the software requirements specifications:

3.5.1 Purpose:
Recommender system is developed using the algorithms proposed. This makes the user recommendations more reliable. This project aims at minimizing the flaws present in the traditional systems.

3.5.2 Specific Requirements

This section describes the hardware software requirements of the project.

3.5.2.1 System Requirements

Below are the specifications of software and hardware needed for the execution of the project.

3.5.2.2 Software Requirements

The software needed for the demonstration of the project are:

- Language : JAVA for implementing the Algorithms & JSP.
- User Interface : HTML

3.5.2.3 Hardware Requirements

The hardware requirements for the project are:

- Processor : Pentium IV
- RAM : 500 MB
- HARD_DISK : 40 GB

3.6 Functional Requirements
The functions that have to be performed by the proposed system are listed below:

3.6.1 Tabulating the user-Item rating data

The main use of Collaborative Filtering recommendation algorithm is to predict the rating of selected item which is not rated by the user depending upon the item-ratings of items that the user has already rated.

Similarity between Items:

Similarity computation is the bottleneck in a collaborative filtering system. There are several ways of doing it. One of the most popular ways of computing similarity is by using Pearson Correlation co-efficient (PCC) & adjusted cosine similarity.

Selection of Neighbors

Neighbors are selected based on the similarity with the current user. Threshold based selection is a technique in which a user is considered to be the neighbor of the current user if the similarity exceeds a threshold value.

Prediction using Similarities in Item-Item:

Once comparisons between the user and the rest of the community of recommenders (regardless of the method applied) are complete, predicted ratings of unrated content can be computed. As above, there are a number of means of computing these predictions.

3.7 Non-functional Requirements

Following are Non-functional requirements:

Performance
The software should use very less amount of memory. The processor must be used efficiently by the process. User must complete the operations in a short interval of time.

**Reliability**

When user calls a software over a specific period of time, the software must deliver the expected services. If the product provides wrong services the product is not reliable.

**Availability**

The software must provide proper service to the user when it is run. The requested services should be delivered in time.

**Security**

Only admin user should know the id and password. All data is protected and not accessible by others.

**Maintainability**

Our project was highly flexibility and maintainability if we want we can add any new features to this project at any time. It is more understandable to others who want to maintain this project.

**Portability**

It is portable it can be use any system where the specified requirements are satisfied

### 3.8 Use Case Diagram

The Use diagram is represented in figure 3.1 and table 3.1
Table 3.1 Use Case

<table>
<thead>
<tr>
<th>Usecase</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rate Items</td>
<td>Rate the movies</td>
</tr>
<tr>
<td>Search movies</td>
<td>To search movies</td>
</tr>
<tr>
<td>Get recommendations</td>
<td>Generate recommendations</td>
</tr>
</tbody>
</table>
DESIGN

4.1 Detailed Design

Design phase involves planning different stages for implementing the Software. A good design helps in developing a good software with less effort.

The system is totally menu driven. It is a very powerful tool for interactive control the menu structure is designed in such a way that the user can navigate easily by selecting the links.

In a system, result of a process can be seen in the output. An efficient output design improves the system’s association with the user and helps in making a decision. The design of output specification carried out with much user friendliness. The system being developed provides the output in the form of screens. The system also provides messages for user friendliness. The screen is provided with help menus and messages that they can help users at difficult situations. Also provides error message, wherever necessary.

In the user login screen, the user enters the login id. After login into the system the user home page opens & it contains four links. The links are home, get recommendations, rate movies & logout. There is search option also. Figure 4.1.1 represents the block diagram for the system.
When clicked on the **home** link, the movies rated most number of times are displayed along with these the movies rated most highly are also present in home page.

When clicked on the **get recommendations** link the movies are recommended for the user using both the methods pearson & adjusted methods.

When clicked on **rate movies**, the movies which were unrated by the user are displayed along with the options of the rating from 1-5. At a time only one movie can rated. Figure 4.1.2 represents the flow chart.
When clicked on the **search** option by entering the data in the search space, the details of the movie along with prediction are displayed on the screen using both Pearson & Adjusted Cosine similarity.

The error rates are found on the command prompt by running the class, the mean error rates are displayed for five test cases for all the methods. For measuring the accuracy of the techniques (Pearson and Adjusted cosine), the data is divided into two parts base case and test case, where base case is used for machine learning to predict the ratings for a movie that is not listed in that base case list. The test case has the list of actual ratings which is used for comparing with the calculated values. I have designed a program which calculates the prediction for that unlisted movie in base case using both the techniques and compares its values to the actual rating present in the test case. These values are displayed in the command prompt. Refer to Figure 6.6.1, Figure 6.6.2, Figure 6.6.3 and Figure 8.1.
4.2 Architecture of the proposed system

This structure in figure 4.1 shows the system architecture and how the system is implemented and achieves the functionalities proposed. The data is read from the input files and is tabulated as a item-user rating table. Then the similarity analysis is done on the data by applying the suitable methods like Pearson and adjusted cosine similarity. The similarity analysis helps in identify the neighbors and generate the prediction. The effectiveness of the methods are observed by their error rates calculated from the test cases data which is actual data split into 80-20, 80 for learning and 20 for testing. Then the differences are squared followed by mean of them is calculated and square root is calculated for each user item having rating greater than zero. All these error rates mean is presented.

![System Architecture Diagram]

4.3 Algorithm

The algorithms used in the project are Pearson correlation technique and Adjusted cosine similarity technique for calculating the similarity between the items.

Tabulating the user-Item rating data

The task of the 'collaborative filtering recommendation algorithm' is to make suggestions to the user about an item that the user has not rated based on the past history of the user. The 'user-item rating' database is used for making observations [4].
Each user has item-rating pairs, and can be represented in a 'user-item table', which has ratings 'Rij' that are given by ith user for the jth item, as shown in table 4.1

<table>
<thead>
<tr>
<th></th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
</tr>
</thead>
<tbody>
<tr>
<td>U1</td>
<td>3</td>
<td>4</td>
<td></td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>U2</td>
<td>1</td>
<td></td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U3</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>U4</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td></td>
<td>?</td>
</tr>
</tbody>
</table>

**Similarity between Items:**

Similarity computation is the bottleneck in a collaborative filtering system. There are several ways of doing it. One of the most popular ways of computing similarity is by using Pearson Correlation co-efficient (PCC) as mentioned in [4]. The formula is in figure 4.2

\[
pearson(x, y) = \frac{\sum(xy) - \left(\frac{\sum x \sum y}{n}\right)}{\sqrt{\left(\sum(x^2) - \frac{\sum x^2}{n}\right)\left(\sum(y^2) - \frac{\sum y^2}{n}\right)}}
\]

Figure 4.2 Pearson Similarity Formula

PCC shows how “proportional” are changes between two given variables. The units of the variables does not matter in the case of pcc. The similarity is between -1 and +1. -1 shows least similarity and +1 shows highest similarity.

Another way of doing it is Adjusted Cosine Similarity as is shown in figure 4.3

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

Figure 4.3 Adjusted cosine similarity formula

In cosine similarity, the similarity is never negative and so, it is always between 0 and 1. Adjusted cosine similarity is better than Cosine similarity because it takes into consideration the
user average. This implies that if a user gives always 5 to each movie and another gives 1 or 2 to all, then, that pattern of ratings is taken into consideration when recommending movies [4].

**Prediction using Similarities in Item-Item:**

Once comparisons between the user and the rest of the community of recommenders (regardless of the method applied) are complete, predicted ratings of unrated content can be computed [4]. As above, there are a number of means of computing these predictions. Here we present one method, guessing the rating $P_i$ of an item $I$ for user $a$ is computed as a "weighted average of neighbor ratings $R_{bi}$". The first is a weighted average of neighbor ratings as shown in figure 4.4

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

Figure 4.4 Prediction formula

### 4.4 CLASS DIAGRAM

Class diagrams are the diagrams used in object-oriented systems for modeling. Various objects and how each object is related to other object are depicted in a class diagram.

#### 4.4.1 Collaborative class

The major purpose of this class is reading the data from the files and calculates the basic operations on the data like averages and identifying top items in the data. The functions present in the class are **getTopmovies()** which identifies the movies which are rated most number of times, **minmax()** function returns either minimum or maximum based on the arguments, **getusermovielist()** returns all the movies rated by the use, **Search()** returns the movie id of the movie searched by the user, **UserAverage()** function calculates and returns the average of all the ratings given by him, **getRate()** function takes the rating given by the user and stores in the files.
4.4.2 CFAlgorithms class

The major purpose of the this class is implementing the similarity analysis like pearson and adjusted cosine similarity. The functions present in the class are `calcPear()` which calculate the pearson similarity between the items, `calcAdjustedcosine()` which calculate the adjusted cosine similarity between the items, `getTopsimilar()` returns the neighbors of the itemid passed, `getRecommendation()` returns the recommendations to the user passed.

4.4.3 RMSE Class

The main purpose of the RMSE class is to calculate the error rates of the methods used in the recommendations generation. The main functions in this class is `computeRmse()`, it is used for calculating the error rates for all the methods.

Figure 4.5 Class diagram
4.5 SEQUENCE DIAGRAM

A sequence diagram is a diagram which shows the flow of the project. It shows various modules and also the messages passed in between them. In the Figure 4.6 different modules of the project are shown. Rate items, Search items, Recommendation and Predict rate are the modules and the function of each module is shown in Figure 4.6.

![Sequence Diagram](image)

Figure 4.6 Sequence diagram
IMPLEMENTATION

5.1 Java Technology

'Applets and applications' are different programs in Java Programming language. An Applet is a small application written in java which runs in a java enabled browser [23]. Applets are very useful in developing interactive websites.

An application is defined as a 'standalone program' which runs on Java platform. An application server is used to support clients on the network. Proxy servers, mail servers, and print servers are different types of servers. The main advantages of application servers are high security, good performance, centralized configuration data integrity and code integrity.

Another specialized program is a servlet. "A servlet can almost be thought of as an applet that runs on the server side". "Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts". "Servlets are similar to applets in that they are runtime extensions of applications". "Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server".[23]

5.2 Steps of Implementation

Step 1: Tabulating the 'user-Item' rating data

The ratings given by a user to an item are tabulated in a 'User-Item' rating table. This data is used to make predictions.
**Step 2: Similarity between Items:**

Computing similarity is by using Pearson Correlation co-efficient (PCC) and Adjusted Cosine Similarity.

**Step 3: Selecting Neighbors**

Neighbors are selected based on a threshold value which is calculated by using Threshold-based selection technique. If the similarity value exceeds a threshold value that user is considered to be a neighbor.
TESTING

6.1 Testing Objectives

Testing process constitutes checking errors in the code. The main objective of testing is to find the undiscovered errors in the code. Testing is a very important module in software life cycle.

6.2 Various methods of testing

There are various methods of testing the code

White Box Testing:

To check the control structure of the program White box Testing can be used. Test cases ensure that all the functionalities of the software have been tested at least once.

Black Box Testing:

Black box testing is designed to check if the software validations are properly done without considering the internal working of the software.

6.3 Levels of Testing

The various levels of testing are listed below:

Unit testing:

In Unit testing all the modules of the software are tested to check if they are in accordance with the modules provided during the design phase. Testing the internal logic of the code is the main motive of Unit testing. White box testing techniques are utilized to check if the control structure is good and does maximum error detection.

System Testing:
In System testing the software is checked to see if the software meets the requirements. In this project, lowest level of unit testing is done to test the data, reports and stored procedures that are developed.

**Testing Process:**

Software testing is done for finding the errors in the software. This is the last phase before software is delivered

### 6.4 UNIT TESTING

#### 6.4.1 Test Case 1

This test case checks if ratings are read properly using the dataset taken from movieLens website.

Test case 1 is represented in table 6.1 it describes the objective of the test which is to check if the values are read properly, the input given is the data set taken from MovieLens and the expected output and the output obtained are the user ratings.

<table>
<thead>
<tr>
<th>Test Name</th>
<th>Unit test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective</td>
<td>Check if ratings are read properly using data in files</td>
</tr>
<tr>
<td>Input</td>
<td>Data in files</td>
</tr>
<tr>
<td>Expected output</td>
<td>Users ratings</td>
</tr>
<tr>
<td>Original output</td>
<td>Users ratings</td>
</tr>
<tr>
<td>Error info</td>
<td>-----</td>
</tr>
<tr>
<td>Solution for error</td>
<td>-----</td>
</tr>
</tbody>
</table>

Table 6.1 Test case 1
Test performed by  Shivani  
Dated on  11-24-2014  

The Figure 6.1 shows the page which loads proper data from the data set. The **home** page displays the list of movies which the user has given ratings for.

![Collaborative Filtering Algorithm Analysis](image)

Figure 6.1 Test case 1

### 6.4.2 Test Case 2

This test case is to check if the recommendations are generated properly. The table 6.2 describes the objective which is to check if the recommendations are generated properly. The recommendations are given based on the values obtained from the algorithms.
Table 6.2 Test case 2

<table>
<thead>
<tr>
<th>Test Name</th>
<th>Unit test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective</td>
<td>Check if recommendations are generated</td>
</tr>
<tr>
<td>Input</td>
<td>Ratings of user</td>
</tr>
<tr>
<td>Expected output</td>
<td>Recommendations for user</td>
</tr>
<tr>
<td>Original output</td>
<td>Recommendations for user</td>
</tr>
<tr>
<td>Error info</td>
<td>-----</td>
</tr>
<tr>
<td>Solution for error</td>
<td>-----</td>
</tr>
<tr>
<td>Test performed by</td>
<td>Shivani</td>
</tr>
<tr>
<td>Dated on</td>
<td>11-24-2014</td>
</tr>
</tbody>
</table>

Figure 6.2 shows the page which displays the recommendations which are obtained from Pearson Correlation method and Adjusted Cosine similarity methods. When the user clicks on **GET RECOMMENDATIONS** option he gets recommendations computed by using both the techniques.
6.4.3 Test Case 3

This test case checks if the search option is working properly. Search option is present at the home page which can be used to search details of a particular movie.

<table>
<thead>
<tr>
<th>Test Name</th>
<th>Unit test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective</td>
<td>Check if Proper function of search option</td>
</tr>
<tr>
<td>Input</td>
<td>Movies data in files</td>
</tr>
<tr>
<td>Expected output</td>
<td>Similar movies of searched movie</td>
</tr>
<tr>
<td>Original output</td>
<td>Similar movies of searched movie</td>
</tr>
<tr>
<td>Error info</td>
<td>-----</td>
</tr>
</tbody>
</table>
6.4.3 Test Case 4

This test case is used for testing if the ratings are updated properly. The user can rate the movies which he has not rated before. This test case is to ensure that the ratings are updated properly.
<table>
<thead>
<tr>
<th><strong>Table 6.4 Test case 4</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Test Name</strong></td>
</tr>
<tr>
<td><strong>Objective</strong></td>
</tr>
<tr>
<td><strong>Input</strong></td>
</tr>
<tr>
<td><strong>Expected output</strong></td>
</tr>
<tr>
<td><strong>Original output</strong></td>
</tr>
<tr>
<td><strong>Error info</strong></td>
</tr>
<tr>
<td><strong>Solution for error</strong></td>
</tr>
<tr>
<td><strong>Test performed by</strong></td>
</tr>
<tr>
<td><strong>Dated on</strong></td>
</tr>
</tbody>
</table>

Figure 6.4.1 shows the page where user can provide his rating. Figure 6.4.2 shows that the value has been updated.
Figure 6.4.1 Test case 4.1

Figure 6.4.2 Test case 4.2
6.4.5 Test Case 5

This test case is to check if the validations are properly made. When a user clicks on rate option without selecting the number ranging between 1-5 an error is prompted.

Table 6.5 Test case 5

<table>
<thead>
<tr>
<th>Test Name</th>
<th>Unit test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective</td>
<td>Check if Proper validations are made</td>
</tr>
<tr>
<td>Input</td>
<td>Movies data in files</td>
</tr>
<tr>
<td>Expected output</td>
<td>Error when movie not rated</td>
</tr>
<tr>
<td>Original output</td>
<td>Error when movie not rated</td>
</tr>
<tr>
<td>Error info</td>
<td>-----</td>
</tr>
<tr>
<td>Solution for error</td>
<td>-----</td>
</tr>
<tr>
<td>Test performed by</td>
<td>Shivani</td>
</tr>
<tr>
<td>Dated on</td>
<td>11-24-2014</td>
</tr>
</tbody>
</table>
Figure 6.5 shows that error is thrown when user clicks rating without selecting the number.

6.6 Test Case 6

This Test case is for error calculation to check the accuracy of the algorithms used. As mentioned in 4.1, The error rates are found on the command prompt by running the class, the mean error rates are displayed for five test cases for all the methods. For measuring the accuracy of the techniques(Pearson and Adjusted cosine), the data is divided into two parts base case and test case, where base case is used for machine learning to predict the ratings for a movie that is not listed in that base case list. The test case has the list of actual ratings which is used for comparing with the calculated values. I have designed a program which calculates the prediction for that unlisted movie in base case using both the techniques and compares its values to the actual rating present in the test case.
Figure 6.6.1 shows the base case data which is fed to the program for learning purpose. The first column represents user, second column shows the movie and the third column is for movie ratings. It can be noticed that user 1 has not given rating for movie 6.

![Figure 6.6.1 Base case](image-url)
The following Figure 6.6.2 is the actual data which is used to compare with the values obtained using the algorithms Pearson correlation and Adjusted cosine similarity.

Figure 6.6.2 Test Case
Figure 6.6.3 shows the program `rmse.java` which is used to test the accuracy of the algorithms. This program takes the data from base case, computes the recommendations using Pearson correlation technique and adjusted cosine technique. It calculates root mean square error for the actual values which are present in the test case file and the values obtained from these techniques.
The following figure 6.6.4 shows the root mean square errors of Adjusted cosine method and Pearson Correlation technique. It is seen that the error rate for Pearson is less than the error from Adjusted Cosine method, which indicates that Pearson correlation method is a better method for using in recommender systems.

Figure 6.6.4 rmse errors
IMPLEMENTATION RESULTS

7.1 Generating Recommendations

The main aim of the project is to provide recommendations to the users using two algorithms, Pearson Correlation and Adjusted cosine similarity methods.

In the figure 7.1, we can see the recommendations generated for the user 450 using Pearson correlation analysis. The recommendations are obtained by clicking on the get recommendation link. Top 15 recommendations are displayed for the user. Apart from the recommendations the prediction values are also shown in each recommendation.

Figure 7.1 Generating recommendations 1
In the below figure 7.2, the recommendations generated using adjusted cosine similarity method is shown for user 450. The top 15 recommendations are shown using this method, apart from the recommendation the prediction value is also shown. The adjusted cosine similarity method results are shown in the above figure.

![Collaborative filtering - Mozilla Firefox](image)

**RECOMMENDATIONS USING COSINE SIMILARITY METHOD**

<table>
<thead>
<tr>
<th>Movie Name</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Close Shave, A (1995)</td>
<td>4.39</td>
</tr>
<tr>
<td>Boot, Dss (1981)</td>
<td>4.26</td>
</tr>
<tr>
<td>Some Folks Call It a Sling Blade (1993)</td>
<td>4.19</td>
</tr>
<tr>
<td>Waiting for Guffman (1996)</td>
<td>4.11</td>
</tr>
<tr>
<td>Third Man, The (1949)</td>
<td>4.09</td>
</tr>
<tr>
<td>Shall We Dance? (1996)</td>
<td>4.04</td>
</tr>
<tr>
<td>Kolya (1996)</td>
<td>3.98</td>
</tr>
<tr>
<td>Paths of Glory (1957)</td>
<td>3.92</td>
</tr>
<tr>
<td>Hoop Dreams (1994)</td>
<td>3.89</td>
</tr>
<tr>
<td>Grand Day Out, A (1992)</td>
<td>3.87</td>
</tr>
<tr>
<td>Jean de Florette (1986)</td>
<td>3.86</td>
</tr>
<tr>
<td>When We Were Kings (1996)</td>
<td>3.86</td>
</tr>
<tr>
<td>Nikita (La Femme Nikita) (1990)</td>
<td>3.83</td>
</tr>
<tr>
<td>Charade (1963)</td>
<td>3.81</td>
</tr>
<tr>
<td>This Is Spinal Tap (1984)</td>
<td>3.81</td>
</tr>
</tbody>
</table>

Figure 7.2 Generating recommendations 2
7.2 Viewing Search Results

In the figure 7.3, the search details are displayed for item searched using the textbox shown in the top right. The details of movie shown are count of the users who rated this movie, the average rating given to that movie, the rating given by the current user (if any). Rate option given to the user if he did not rate the movie.

Figure 7.3 Viewing search results 1
In the figure 7.4, apart from the movie details in the search option the similar movies are also presented to the user. The similar movies for the searched item are also presented using both the similarity methods. The top 15 movies similar to the searched item are displayed. Both the similarity methods are used for calculating the similarity for the searched item.

Figure 7.4 Viewing Search results 2
7.3 Calculating the Error Rates

In the above figure 7.5, the error rates are shown for all the similarity methods for prediction generation. This test is done on five test cases with 80-20 learning to test ratio. The above error rates which are well below the 1.0 indicate the effectiveness of the system. The main motive of a 'recommender system' is to provide accurate suggestions to a user. In my project I implemented two algorithms for providing recommendations. Calculating rmse values for these both techniques helps us in understanding which technique gives more accurate results.

As already mentioned in 4.2 and 6.6 the figure 7.5 shows values which are calculated for 4 different set of test cases. In all the cases Pearson correlation is found to be more accurate.
CONCLUSION

Two methods are used to predict ratings for a given item for a given customer. They are Pearson correlation coefficient, adjusted cosine similarity and. The root mean square error (RMSE) for all the two cases are compared and following comparison is made:

![Figure 8.1 Error rates]

Clearly, Pearson correlation is not as good as adjusted cosine similarity as explained in Test Case 6. The reason is that Pearson correlation gives a biased result whenever the number of co-rated items is less as shown in figure 8.1. For example, when comparing two items, one of which has hundred ratings as 5 and the other having just 5 ratings, all 5, their Pearson correlation
coefficient will come as 1 (which shows most similar) because it doesn’t take into consideration the fact that the number of ratings for the second one is less. Adjusted cosine takes into consideration the difference in rating habit of users and thus, subtracts the user average from the ratings.

**FUTURE ENHANCEMENTS**

Also, I feel that RMSE is not the right way to measure the accuracy of a recommender system. It is a topic of hot debate in recommender system research and we second that. Recommender systems were originally meant for showing users items that they did not know were there. It is about discovery. Using RMSE, all we are testing is how accurate we are in predicting something which has already been rated. This is an effective way of testing if our algorithms are working correctly, but this is not how a recommender system should be judged. We should have statistical measures to correctly judge user satisfaction, the fact that when did he “discover” things he was not aware of and if he/she was happy/unhappy about it.

I would like to say that similarity computation is the most expensive part of the collaborative filtering process and it is an unsolved problem. The rating data is always sparse because if the rating data is not sparse and full of data, then, the recommendation process becomes trivial. There are several other ways to compute similarity, like trust-based collaborative filtering, where there is a circle of trust for all users and only that circle is asked for recommendations. There is one scenario where a few experts are only used to rate movies and their opinion is used for all others.
REFERENCES


[7] Scalability-Wikipedia, the free encyclopedia

[8] Correlation and dependence- Wikipedia, the free encyclopedia

[9] Statistical dependence: Independence, Correlation and dependence, Copula, Spearman’s rank correlation coefficient

[10] “A Collaborative Filtering Recommendation Algorithm based on User Clustering and Item Clustering”. SongJie Gong, Zhejiang Business Technology Institute, 2010


[12] “A Framework for Collaborative, Content-Based and Demographic Filtering”. Micheal Pazzani, University of California, 2004


5.3 Sample Coding

public void calcPearson()
{
    int i,j;
    int itemsrateduser[][]=col.getItemsrate();
    for(i=0;i<ITEMSTOTAL;i++)
    {
        for(j=i;j<ITEMSTOTAL;j++)
        {
            float num=(float) 0.0;
            float denom1=(float) 0.0;
            float denom2=(float) 0.0;
            int coratecount=0;
            float iavg=col.getItemaverage(i);
            float javg=col.getItemaverage(j);
            for(int k=0;k<col.getItemratecount(i);k++)
            {
                int m=itemsrateduser[i][k]-1;
                if(rate[m][j]!=0)
{ 

    num+=(rate[m][i]-iavg)*(rate[m][j]-javg);

    denom1+=(rate[m][i]-iavg)*(rate[m][i]-iavg);

    denom2+=(rate[m][j]-javg)*(rate[m][j]-javg);

    coratecount++;

}

pearson[i][j]=(float) (num/Math.sqrt(denom1*denom2));

if(coratecount<15){pearson[i][j]=pearson[i][j]*coratecount/15;}

if(coratecount==0)pearson[i][j]=-1;

pearson[j][i]=pearson[i][j];

}

}

public void calcAdjustedcosine()
{

    int i,j;

    int itemsrateduser[i][]=col.getItemsrate();

    for(i=0;i<ITEMSTOTAL;i++)

    {

        for(j=i;j<ITEMSTOTAL;j++)

            

}
{
    float num=(float) 0.0;
    float denom1=(float) 0.0;
    float denom2=(float) 0.0;
    int coratecount=0;
    for(int k=0;k<col.getItemratecount(i);k++)
    {
        int m=itemsrateduser[i][k]-1;
        float uavg=col.getUseraverage(m);
        if(rate[m][j]!=0)
        {
            num+=(rate[m][i]-uavg)*(rate[m][j]-uavg);
            denom1+=(rate[m][i]-uavg)*(rate[m][i]-uavg);
            denom2+=(rate[m][j]-uavg)*(rate[m][j]-uavg);
            coratecount++;
        }
    }
    adjustcosine[i][j]=(float) (num/Math.sqrt(denom1*denom2));
    if(coratecount<15){adjustcosine[i][j]=adjustcosine[i][j]*coratecount/15;}
    if(coratecount==0)adjustcosine[i][j]=-1;
    adjustcosine[j][i]=adjustcosine[i][j];
} }