Available online at http://www.journalcra.com

International Journal of Current Research
Vol. 3, Issue, 7, pp.199-203, July, 2011

INTERNATIONAL JOURNAL OF CURRENT RESEARCH

ISSN: 0975-833X

RESEARCH ARTICLE

WEBSITE PERSONALIZATION USING DATA MINING TECHNIQUES-COLLABORATIVE FILTERING

P.K. Srimani[1] and Srinivas, A[2]

[1]Former Chairman, Department of Computer Science and Maths, Bangalore University, India
[2]Faculty of Computer Science, Surana College, Bangalore, Karnataka, India

ARTICLE INFO

Article History:
Received 22nd April, 2011
Received in revised form
17th May, 2011
Accepted 19th June, 2011
Published online 16th Julv 2011

Key words:
Collaborative Filtering,
Recommender System,
Personalization,
Content Based Filtering,
Taste Library ASF,
(Apache Software Foundation).

ABSTRACT

Finding information on a large web site can be a difficult and time-consuming process. Recommender systems can help users find information by providing them with personalized suggestions. In this paper the creation of recommendation system was emphasized to achieve the personalization on the website. Recommender systems typically use techniques from collaborative filtering, in which proximity measures between users are formulated to generate recommendations, or content-based filtering, in which users are compared directly to items. Our approach used similarity measures between users. User-based collaborative filtering gave personalized recommendations by finding similar users. Item-Based collaborative filtering recommended similar items. Different algorithms were compared. However, the applied testing procedure did not employ equal conditions for both approaches. The aim of this report was to give an evaluation on both the approaches by employing a fair testing procedure on the data gathered. Test results and their dependency to the employed algorithms were interpreted. The experiments are carried out by building the recommendation engine through the Taste library in Java — a fast and flexible engine for collaborative filtering.

INTRODUCTION

As the World Wide Web continues to grow at an exponential rate, the size and complexity of many web sites grow along with it. For the users of these web sites it becomes increasingly difficult and time consuming to find the information they are looking for. To help users find the information that is in accordance with their interests a web site can be personalized. Today, personalization is something that occurs separately within each system that one interacts with. Recommender systems automate personalization on the Web, enabling individual personalization for each customer. Recommender systems enhance E-commerce sales in three ways:

- Browsers into buyers:
- Cross-sell:
- Loyalty

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) is the most successful recommendation technique to date.

***Corresponding author:** profsrimanipk@gmail.com, asrhod@gmail.com

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. In this paper a recommendation system (system that suggest items of interest to a user) using collaborative filtering which can be user based or item based was considered. As an experimental domain, a system that suggests movies to customers was considered based on information regarding what they have liked and disliked in the past. In carrying out the experiments “Taste library” in Java was used which an open-source recommendation library is using which one could create a basic recommender easily with collaborative filtering algorithms. Taste was added to the Apache’s Mahout Project. Taste has many standard collaborative filtering algorithms for developers to work on. What is good about Taste is that it can scale training session for slope one similarity calculations.

Existing Methodology Comprises the Following

Collaborative filtering approaches are often classified as memory-based or model-based. In the memory-based approach, all rating examples are stored as-is into memory (in contrast to learning an abstraction). In the prediction phase, similar users or items are sorted based on the memorized ratings. Based on the ratings of these similar users or items, a recommendation for the test user can be generated. Examples of memory-based collaborative filtering include user-based methods and item-based methods The advantage of the memory-based methods over their model-based alternatives is that less parameters have to be tuned; however, the data sparsity problem is not handled in a principled manner. In the model-based approach, training examples are used to generate a model that is able to predict the ratings for items that a test user has not rated before. Examples include decision trees, aspect models and latent factor models. The resulting 'compact' models solve the data sparsity problem to a certain extent. However, the need to tune an often significant number of parameters has prevented these methods from practical usage. Lately, researchers have introduced dimensionality reduction techniques to address data sparsity. However, as pointed out some useful information may be discarded during the reduction. In content-based filtering items are matched either to a user's interest profile or query on the basis of content rather than opinion. One strength of this approach over collaborative filtering is that as long as the system has some information about each item, recommendations can be made even if the system has received a small number of ratings, or none at all. The downside is that each item must be characterized with respect to the features that appear in a user's profile and, further, the profile of each user must be collected and modeled. Naturally, these descriptive features must, themselves, be acquired or engineered somehow. Recently a number of methods have been developed for the "collaborative filtering" or "social filtering" of information (Resnick et al. 1994; Shardanand & Maes 1995; Breeze et al. 1998). The main idea is to automate the process of "word-of-mouth" by which people recommend products or services to one another. If it is required to choose between a variety of options with which one has any experience, one will often rely on the opinions of others who do have such experience. However, when there are thousands or millions of options, like in the Web, it becomes practically impossible for an individual to locate reliable experts that can give advice about each of the options. By shifting from an individual to a collective method of recommendation, the problem becomes more manageable. The basic mechanism behind collaborative filtering systems is the following:

- a large group of people's preferences are registered;
- using a similarity metric, a subgroup of people is selected whose preferences are similar to the preferences of the person who seeks advice;
- a (possibly weighted) average of the preferences for that subgroup is calculated;
- The resulting preference function is used to recommend options on which the advice-seeker has expressed no personal opinion as yet.

MATERIALS AND METHODS

Collaborative filtering is a mapping of two high dimensional spaces as shown in the figure below.

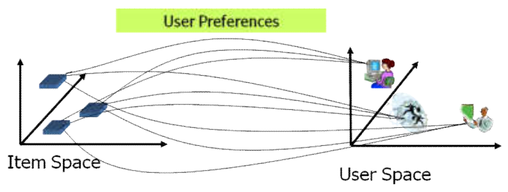


Fig. 1: Collaborative filtering

The goal of a collaborative filtering algorithm is to suggest new items or to predict the utility of a certain item for a particular user based on the user's previous likings and the opinions of other like-minded users. In a typical CF scenario, there is a list of *m* users $\mathcal{U} = \{u_1, u_2, \ldots, u_m\}$ and a list of *n* items $\mathcal{I} = \{i_1, i_2, \ldots, i_n\}$. Each user u_i has a list of items I_{ui}, which the user has expressed his/her opinions about. Opinions can be explicitly given by the user as a *rating score*, generally within a certain numerical scale, or can be implicitly derived from purchase records, by analyzing timing logs, by mining web hyperlinks and so on. Note that $I_{u_i} \subseteq \mathcal{I}$ and it is possible for I_{ui} to be a *null-set*. There exists a distinguished user $u_a \in \mathcal{U}$ called the *active user* for whom the task of a collaborative filtering algorithm is to find an item likeliness that can be of two forms.

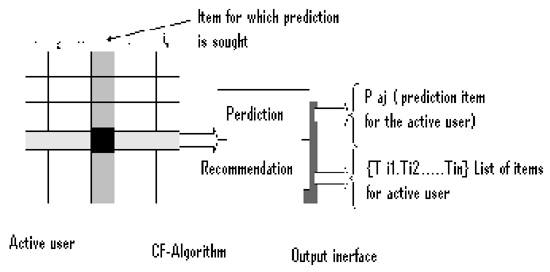

Item for which prediction is sought
Perdiction
Recommendation
P aj (prediction item for the active user)
{T i1.Ti2.....Tin} List of items for active user
Active user
CF-Algorithm
Output inerface

Figure 2: Collaborative filtering process

The above figure shows the schematic diagram of the collaborative filtering process.

DOMAIN DESCRIPTION

Collaborative filtering is demonstrated using the filtering techniques in the domain of movie recommendation. Visitors go to a movie's page to check out the plot description and will be shown a list of similar movies. These are other movies that were watched by people who also rented that specific movie. Because of space- and time-constraints, this application only provides a view on the dataset and the recommended movies. The dataset contains rating data provided by each user for various movies. User ratings range from zero to five stars. Zero stars indicate extreme dislike for a movie and five stars indicate high praise. To have a quicker turn-around time for our experiments, only a subset of the Each Movie dataset is used. The movies and their ratings originate from the Movielens dataset of the Grouplens research group from the University of Minnesota. There are datasets containing 100.000, 1 million and 10 million ratings.

RECOMMENDER SYSTEM FRAMEWORK

In this paper, collaborative techniques were applied, specifically as the information filtering tools for the proposed framework. Collaborative or social-based filtering retrieved the information for a particular user by referring to other user evaluations on the information content. The method of mining user access patterns based on the association rule mining was applied as the collaborative filtering technique. The overall process for designing and implementing a recommender system could be comprised in 5 steps.

- *Data Collection* included the collection of data sets for executing the data mining algorithms.
- *Data Preprocessing* included cleaning and transforming the collected data sets into the formats which were suitable for the data mining algorithms. It also included the data reduction and selection techniques to improve the efficiency of the data mining algorithms.
- *Information Filtering via Data Mining* was the core process of the recommender system framework, where the data sets were analyzed and the data mining algorithms were applied as the information filtering tools to generate and discover any useful and interesting recommended outputs.
- *Database Design and Implementation* ensured the efficiency of data and information access and retrieval, the database for the recommender system was designed and implemented for all related data.
- *User Interface Design and Implementation* acted as an intermediary between the users and the recommender system. This step involved the design and implementation of a Web (i.e., HTTP) server which received the user's requests via WWW, processes the requests by accessing the database, and responds by returning the results to the users. The user interface provided a recommendation function with the user personalization technique by requiring each user to log into the system in order to keep track of the preferences.

MAHOUT (TASTE) – RECOMMENDATION ENGINE

Apache Mahout is a new open source project by the Apache Software Foundation (ASF) with the primary goal of creating scalable machine-learning algorithms that are free to use under the Apache license. The project is entering its second year, with one public release under its belt. Mahout contains implementations for clustering, categorization, CF, and evolutionary programming.

Building a Recommendation Engine

Mahout currently provided tools for building a recommendation engine through the Taste library — a fast and flexible engine for CF. Taste supported both user-based and item-based recommendations and provides many choices for making recommendations, as well as interfaces for us to define our own. Taste provides a rich set of components from which one can construct a customized recommender system from a selection of algorithms. Taste was designed to be enterprise-ready; it's designed for performance, scalability and flexibility. Taste is not just for Java; it can be run as an external server which exposes recommendation logic to your application via web services and HTTP. Top-level packages define the Taste interfaces to these key abstractions:

- Data Model
- User Similarity and Item Similarity
- User Neighborhood
- Recommender

Sub packages of *org.apache.mahout.cf.taste.impl* hold implementations of these interfaces. These are the pieces to build the recommendation engine. For the academically inclined, Taste supports both memory-based and item-based recommender systems, slope one recommenders, and a couple other experimental implementations. It does not currently support model-based recommenders.

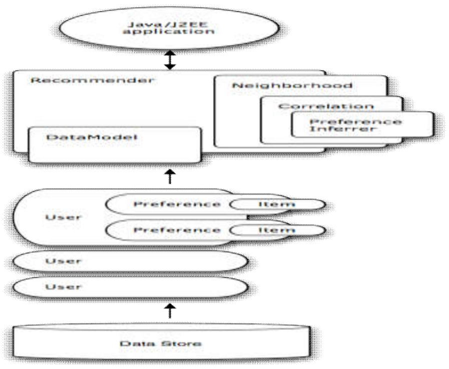


Fig. 3: Architecture

This diagram shows the relationship between various Taste components in a user-based recommender. An item-based recommender system was similar except that there were no Preferred Inferences or Neighborhood algorithms involved (*B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-Based Collaborative Filtering Recommendation Algorithms*). A Recommender is the core abstraction in Taste. Given a DataModel, it can produce recommendations. Applications will most likely use the Generic User Based Recommender implementation or Generic Item Based Recommender, possibly decorated by Caching Recommender.

DATA MODEL

A Data Model is the interface to information about user preferences. An implementation might *draw this data from any source, but a database is the most likely source. Taste provides MySQL JDBC Data Model to access preference data from a database via JDBC, though many* applications will want to write their own. Taste also provides a File Data Model. There are no abstractions for a user or item in the object model (not anymore). Users and items are identified solely by an ID value in the framework. Further, this ID value must be numeric; it is a Java long type through the APIs. A Preference object or Preference Array object encapsulates the relation between user and preferred items (or items and users preferring them). *User Similarity* defines a notion of similarity between two Users. This is a crucial part of a recommendation engine. These are attached to a Neighborhood implementation. *Item Similarities* are analogous, but find similarity between Items. In a user-based recommender, recommendations are produced by finding a "neighborhood" of similar users near a given user. A UserNeighborhood defines a means of determining that neighborhood -for example, nearest 10 users. Implementations typically need a UserSimilarity to operate.

TOOLS & REQUIREMENTS

Required

1. Java / J2SE 6.0

Optional

Apache Ant 1.5 or later and Maven 2.0.10 or later, if one wants to build from source or build examples. (Mac users note that even OS X 10.5 ships with Maven 2.0.6, which will not work.) Taste web applications require a Servlet 2.3+ container, such as Jakarta Tomcat. It may in fact work with older containers with slight modification. *MySQL JDBC Data Model* implementation required a MySQL 4.x (or later) database. Again, it may be made to work with earlier versions or other databases with slight changes.

IMPEMENTATION

Figure 3 shows the network architecture diagram, figure 4 and 5 shows class diagram and Sequence diagram respectively.

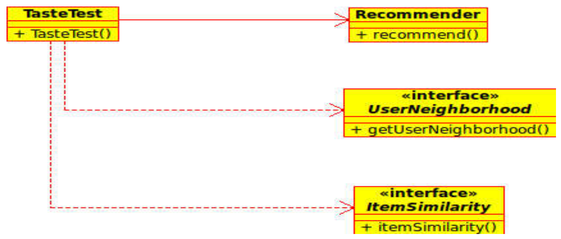


Fig.4: Class Diagram

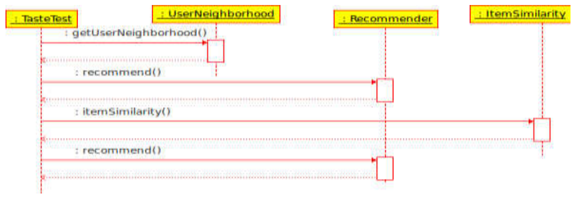


Fig. 5: Sequence Diagram

User Data

user id	*item id*	*rating*	*timestamp*
196	242	3	881250949

Item Data

movie id	*movie title*	*video release date*	*UR L*	*Unknown*

Figure 6: User Date and Item Data

The user interface screen shots are shown in the figure 7 and figure 8 respectively under the Appendix section

RESULTS AND DISCUSSION

Recommender systems research used several types of measures for evaluating the quality of a recommender system. They can be mainly categorized into two classes:

- *Statistical accuracy metrics* evaluate the accuracy of a system by comparing the numerical recommendation scores against the actual user ratings for the user-item pairs in the test dataset. *Mean Absolute Error* (MAE) between ratings and predictions was a widely used metric. MAE was a measure of the deviation of recommendations from their true user-specified values. For each ratings-prediction pair $<p_i,q_i>$ this metric treats the absolute error between them i.e., $|p_i-q_i|$ equally. The MAE was computed by first summing these absolute errors of the N corresponding ratings-prediction pairs and then computing the average. Formally,

$$MAE = \frac{\sum_{i=1}^{N} |p_i - q_i|}{N}$$

The lower the MAE, the more accurately the recommendation engine predicted the user ratings. *Root Mean Squared Error* (RMSE), and *Correlation* were also used as statistical accuracy metric
- *Decision support accuracy metrics* evaluated how effective a prediction engine was at helping a user to select high-quality items from the set of all items. These metrics assumed the prediction process as a binary operation-either items were predicted (good) or not (bad). With this observation, an item which had a prediction score of 1.5 or 2.5 on a five-point scale was irrelevant while user considered being only predictions with score 4 or higher. The most commonly used decision support accuracy metrics were *reversal rate*, *weighted errors* and *ROC sensitivity*.

In this paper MAE was used as the evaluation metric to report prediction experiments because it was most commonly used and easiest to interpret directly

ADJUSTED COSINE (VS) PEARSON CORRELATION

Figure 6 shows the evaluated results regarding different similarity measures. The error was measured as MAE (mean absolute error). Note that the similarity measures were evaluated for the item-based approach as shown in the figure 1.

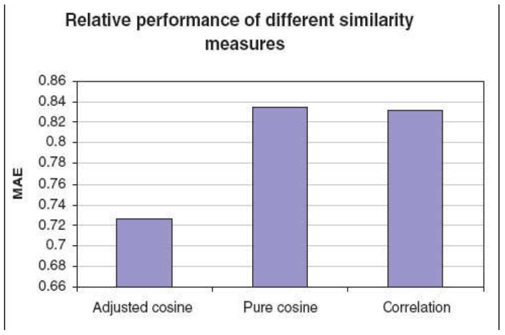
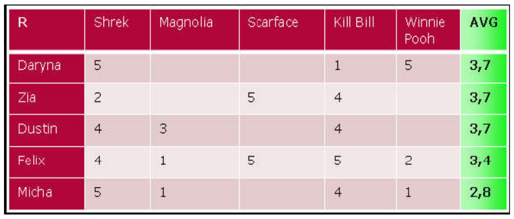


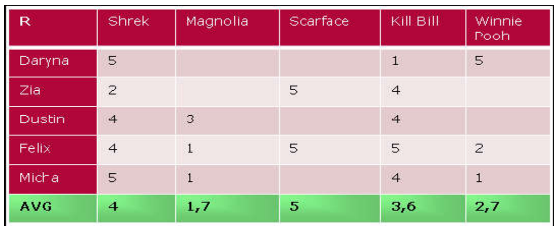
Figure 7: Relative Performance of different similarity measures

Pearson's Correlation and Adjusted Cosine were very similarly calculated but resulted in different absolute errors (Fig. 8 and 9)



R	Shrek	Magnolia	Scarface	Kill Bill	Winnie Pooh	AVG
Daryna	5			1	5	3,7
Zia	2		5	4		3,7
Dustin	4	3		4		3,7
Felix	4	1	5	5	2	3,4
Micha	5	1		4	1	2,8

Figure 8: An example table containing ratings and items average ratings, as used for Adjusted Cosine measure regarding User-User similarity



R	Shrek	Magnolia	Scarface	Kill Bill	Winnie Pooh
Daryna	5			1	5
Zia	2		5	4	
Dustin	4	3		4	
Felix	4	1	5	5	2
Micha	5	1		4	1
AVG	4	1,7	5	3,6	2,7

Fig. 9: An example table containing ratings and users average ratings, as used for Adjusted Cosine measure regarding Item-Item similarity

Technically this meant that data from every row was scaled to the values of its column, and vice versa. Semantically, the difference in the data meant that regarding the adjusted cosine calculation for User-User similarity, the average overall rating of the item (popularity) was taken into account. But for Item-Item similarity, the average user's rating (his general attitude) was considered. In contrast to this, Pearson correlation scales the values of a vector to the average of the vector. So for a row, the row average and not the column average was taken into account.

CONCLUSIONS AND FUTURE RESEARCH

The need for personalized website design has increased in recent years. The current approach for personalized website design was easily applied to websites due to their cost-effective features, but the current approach cannot easily provide a more refined personalized service because of its lack of a user information database. In this study, the design recommender system was investigated as a more advanced method for website design personalization and current recommender systems were discussed and recommendation techniques were identified, especially collaborative filtering, as a key technology for the creation of user-adaptive personalization service in websites, which produced personal recommendations by computing the similarity between a user's preference and the those of other users. The problematic issues for the collaborative filtering technique, such as the case when a new user with no or very few ratings cannot be reliably matched against other users were examined. To solve this new user problem, we used a demographic data instead of preference data. Based on these investigations, we presented a framework of a design recommender system for website personalization based on the collaborative filtering approach was presented. The work done has limitations and directions for further research as follows:

- This study aimed to suggest a basic concept and prototype to improve the current personalization service for website design by using exploratory research methods such as a literature review and a case study. Therefore, in-depth technical studies should follow for the more practical use of the proposed prototype.
- Our recommender system was based on collecting explicit user preference data by means of user's direct participations such as user ratings; however, implicit user preference data obtained by analyzing a user's usage and behavior should be mixed with explicit user data to build a more sensitive and effective recommender system and support lifecycle personalization. Therefore, our future work in this area is to include the implicit user preference data in our recommender system.

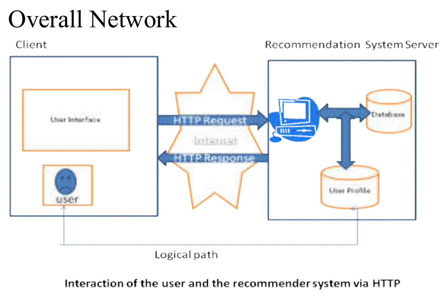


Fig. 10: Interaction of the user and the recommender system via HTTP.

COMPONENT DIAGRAM

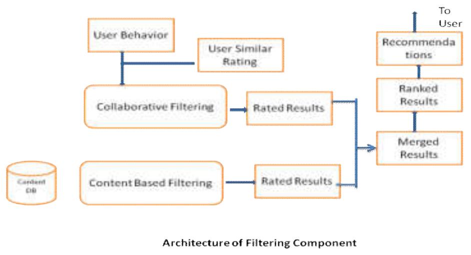
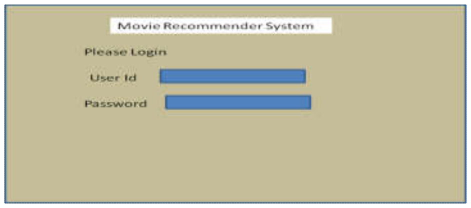


Fig. 11: Architecture of Filtering Component

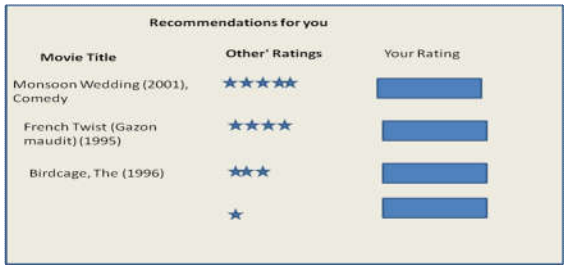


Movie Recommender System

Please Login

User Id

Password



Recommendations for you

Movie Title	Other' Ratings	Your Rating
Monsoon Wedding (2001), Comedy	★★★★★	
French Twist (Gazon maudit) (1995)	★★★★	
Birdcage, The (1996)	★★★	
	★	

Fig. 11 and 12. Recommender System Screen Shots

REFERENCES

Arun K Pujari 2006, Data Mining Techniques.

Geroge M.Marakas 2007, Modern Data Warehousing, Mining and Visualization, Core Concepts.

Jiawei Han and Micheline kamber, 2008, Data Mining Concept and Techniques.

Margaret H. Dunham , 2009, Data Mining Introductory and Advanced Topics.

Resnick et al. 1994; Shardanand & Maes 1995; Breeze et al. 1998 Collaborative Filtering and Social Filtering.

B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-Based Collaborative Filtering Recommendation Algorithms.

Springer Berlin/Heidelberg Web Personalization Techniques for E-commerce –

Book Series – Lecture Notes in Computer Science.

http://www.daniel-lemire.com/fr/documents/publications/webpaper.pdf

http://lucene.apache.org/mahout/taste.html

http://www.hindawi.com/journals/aai/2009/421425.html

http://www.lebensland.de/download/ibcf/ibcf.pdf

www.grouplens.org