# A Hybrid Restaurant Recommender

Prerna Dwivedi
Mumbai, India

Nikita Chheda
Mumbai, India

## ABSTRACT
In any e-commerce application, the recommender systems play a vital role as they assist the prospective buyers in making proper decisions on the basis of the recommendations that the system provides. Recommender systems aim at providing the users with effective recommendations based on their intuitions and preferences. The two very old techniques commonly used for providing automated recommendations are collaborative filtering and knowledge based filtering techniques. However, both these techniques have certain drawbacks when used separately. In this paper, we propose architecture for designing hybrid recommender system that combines the advantages of both the techniques; thereby improving accuracy. The proposed approach uses a combination of personalised recommendations (based on individuals past behaviour), social recommendations (based on past behaviour of similar users) and item-based recommendations (based on restaurant database). This combination overcomes all the drawbacks that are faced when these techniques are used separately. In this paper, we have described the application of such a system within the domain of restaurants.

## General Terms
Hybrid Restaurant Recommender System, Data mining

## Keywords
Hybrid recommender system, restaurant recommender system, collaborative filtering and knowledge based recommender system, web log records

## 1. INTRODUCTION
Advancement in high speed network technology and the ongoing rapid expansion of the internet has increased the necessity of filtering the abundant information made available to the users; thus, increasing the necessity of developing recommender systems. Recommender system will provide users with options that are relevant to their interests and their previous preferences; thus, helping them navigate through large information spaces of insignificant and irrelevant data.
Two most commonly used techniques for building a recommender system are collaborative filtering and knowledge based filtering. The collaborative filtering approach makes prediction for a user based on the similarity between the interest profile of that user and those of other users. On the other hand, knowledge based recommender systems exploits their knowledge base of the product domain to generate recommendations to the user, by reasoning about what products meet the user's requirements. The hybrid restaurant recommender system that we have developed switches between the two techniques based on the situation the system is experiencing.
Restaurant recommender system has been made for a website that helps the users to order food from or gain information about any restaurant in any location. Considering the increased number of restaurants and fast food chains, finding a restaurant that suits the customer's requirements the best is difficult. The recommender system recommends the user a list of restaurants based on the user access pattern available in the web log records. It not only targets individual customers but also considers the preferences of others because they are in the same group or have preferred similar kinds of services. The idea is to capture, model, and analyze the behavioral patterns and profiles of users interacting with the Website. Web servers record and accumulate data about user interactions whenever requests for resources are received. Analyzing the Web access logs can help understand the user behavior and the web structure. The discovered patterns are represented as collections of pages that are frequently accessed by the user. This collection is then used to determine similarity between two users and the current user's own past preferences.
Implemented system uses various recommendation techniques to provide beneficial and accurate recommendations to the users; its evaluation has been conducted successfully with recommendation experiment and usability test.

## 2. BACKGROUND
In this section, we introduce the collaborative filtering and knowledge based filtering recommender systems. Furthermore, we discuss the positive and negative sides of the two systems and why the two approaches have been integrated to form a hybrid system.

### 2.1 Collaborative filtering
Collaborative filtering is one of the oldest filtering techniques that earlier recommender systems used. CF method of mining data involves filtering of information or patterns done using techniques involving collaboration among users' viewpoints, data sources, user ratings, etc. It was very much preferred technique as it requires users' active participation, an easy way to represent users' interests to the system. Hence the recommendations are always subject to change as it is very much dynamic. Dynamic means that it constantly encourages active user participation and their ratings, views about a particular item. The system aggregates data about customers' purchasing habits or preferences and make recommendations to other users based on similarity in overall patterns. For example, in the Ringo music recommender system, users who had expressed their musical preferences by rating various artists and albums could get suggestions of other groups and recordings that others with similar preferences also liked [3].
It should be obvious that CFRSs have the following advantages and shortcomings:
Advantages:

- They can make personalized recommendations.
- They are able to identify appropriate items to users.
- Their prediction quality improves over time as their databases of user preferences get larger and larger.

However, as collaborative filtering methods recommend items based on users' past preferences, new users will need to rate sufficient number of items to enable the system to capture their preferences accurately and thus provides reliable recommendations.
Similarly, new items also have the same problem. When new items are added to the system, they need to be rated by substantial number of users before they could be recommended to users who have tastes similar to the ones who rated them. Hence there is very little opportunity to refine searches.
Moreover, a CFRS may not behave properly when a user's interests change, since it makes recommendations based on the past interests of that user. For example, a book shopper, who always purchased computer books in the past, may find CFRS recommendations not very helpful when she is searching books for her children.

## 2.2 Knowledge Based Mining

Knowledge based Mining builds its recommendations based on the knowledge of the items explicitly. Unlike Collaborative filtering methods that used user's past history, this technique focuses on the knowledge stored at the back end about users and products to actively reason out what products meet the user's requirements. This method does not face the ramp-up problem like collaborative filtering method. In this technique, it makes implicit assumptions based on the user's demographic or behavioral patterns. The user is asked to give the first input in the form of his favorites or most likely used item. Then based on this input as well as the other user data (behavioral/demographic) present, a similar item that matches his first item is provided as recommendation. In this way, a tree is formed where the leaves are the recent recommendations or preferences and node is the old or already present data.
Thus, the recommendations are built on the large set of data collected through knowledge engineering. The PersonalLogic recommender system offers a dialog that effectively walks the user down a discrimination tree of product features [6]. Other systems have adapted quantitative decision support tools for this task.
The advantage of using this method of recommendation is that it is sensitive to preference changes and does not require prior knowledge about a new user. However, to make good recommendations, a KBRS must understand the product domain well. It must have knowledge of important features of the product, and be able to access the knowledge base where these important features are stored in an inferable way. Thus, a KBRS requires knowledge engineering with all of its attendant difficulties. Also, the data remains static and is re-used time and again when considering users' choices or preferences. Therefore, the recommendations are quite static.

## 2.3 Hybrid Approach

We saw what problems the two methods (CF and KB) face and how our method stands apart by overcoming them. Hybrid approach is a technique that uses both the techniques – collaborative and filtering, and switches between the two depending on the situation. For example, it will use the collaborative approach if the system knows the user well and knowledge based approach if the user is new to the system. In order to explain how the hybrid approach that our system uses is better and more effective than the above two techniques, we have performed comparative analysis of all the three methods, as shown below:

**Table 1. Comparison Table**

| Technique | Advantages | Disadvantages |
|---|---|---|
| Knowledge-based | 1. No ramp-up required | 1. Knowledge engineering |
| | 2. Sensitive to preferences changes | 2.Recommendations are static |
| Collaborative filtering | 1. Can identify niches precisely. | 1. Quality dependent on large historical data set. |
| | 2. Domain knowledge not needed. | 2. Subject to statistical anomalies in data. |
| | 3. Quality improves over time. | 3. Insensitive to preference recommendations. |
| | 4. Personalized changes | |
| The Hybrid method of filtering. | 1. Browsing patterns are considered for recommendation; hence there is no way in which user can manipulate data. | 1. Accuracy in recommendations comes at the stake of efficiency. Efficiency can still be improved. |
| | 2. No ramp up and shilling attacks. | |
| | 3. User preferences are updated regularly, so dependency on historical data is low. | |
| | 4. Recommendations are dynamic; it keeps changing as the user's preferences change. | |

## 3. PROPOSED ARCHITECTURE

Figure 1 shows the proposed architecture. In this section, we introduce an architecture that switches between the two types of recommender systems according to the situation currently being faced by the system. For new users, simple mode is used where the user enters his requirements and no past user history is required. For already existing users, advanced mode of recommendation is available which has three types of recommendation approaches. Recommendations based on user's history have been used to determine the user's preferences in previous sessions when he was searching for something similar. Recommendations based on similar users have been used for recommending restaurants to the user that were preferred by users possessing similar interests.

Recommendations based on cuisine and location preference is used to determine user's past preferences and recommending him similar restaurants [9]. To handle new hotels, notifications are sent to the users about their opening. Each of these types has been described in detail below.
In order to implement these approaches, we made use of user access logs. These logs were used for determining user navigation patterns and stored in the database in the form required by the system. Also the ratings given by the user is stored in the database. These records are then processed to provide recommendations using different techniques.

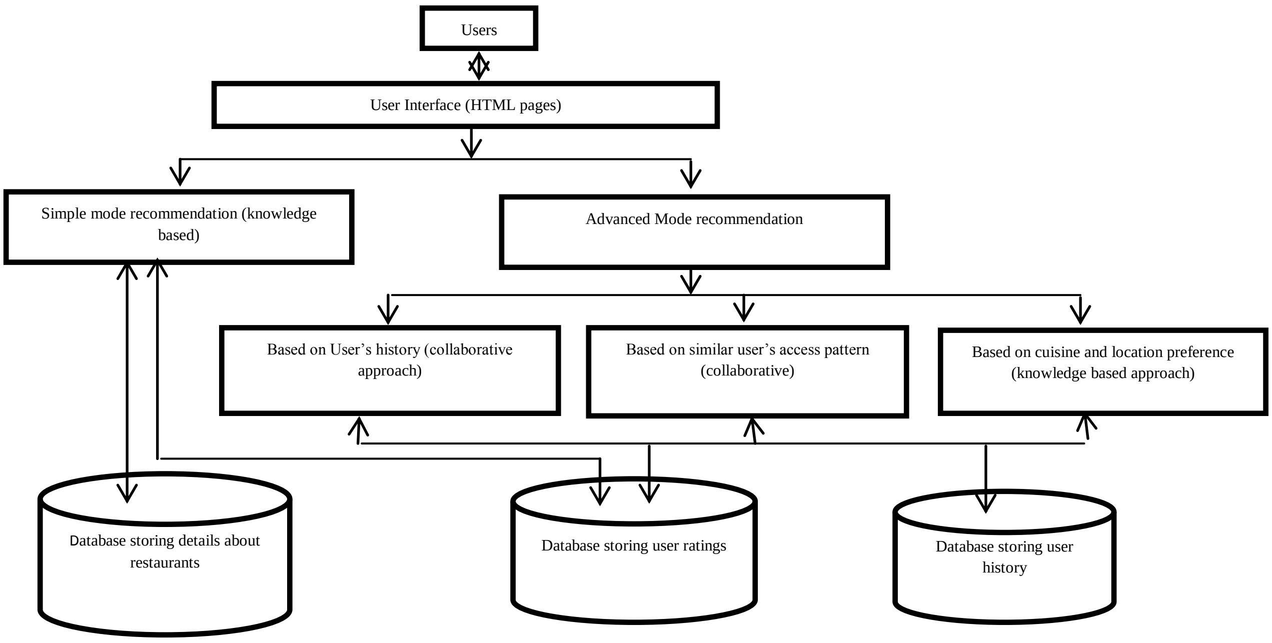
Users
User Interface (HTML pages)
Simple mode recommendation (knowledge based)
Advanced Mode recommendation
Based on User's history (collaborative approach)
Based on similar user's access pattern (collaborative)
Based on cuisine and location preference (knowledge based approach)
Database storing details about restaurants
Database storing user ratings
Database storing user history

**Fig 1: Architecture of Restaurant Recommender System**

The algorithms used to develop all the stages involved in developing recommender system have been described in the following subsections.

## 3.1 Extraction

Extraction is a process in which only relevant information is taken from the web log records and stored in the database. It includes the following steps:

**Step 1:** Every time a user logs out, his access patterns are retrieved from the web log records. Web log is an unprocessed text file which is recorded from the Apache Tomcat Server. Due to different server setting parameters, there are many types of web logs, but typically the log files share the same basic information, such as: client IP address, request time, requested URL, HTTP status code, referrer, etc. Web log consisting of 17 attributes was used. Fragment of web log from Apache Tomcat server is shown below:
Fields: date time c-ip cs-username s-sitename s-computername s-ip s-port cs-method cs-uri-stem cs-uri-query sc-status time-taken cs-version cs-host cs(User-Agent) cs(Referer).
**Step 2:** In order to store the user access patterns in the databases, the web logs need to be pre-processed so as to eliminate irrelevant information and store only the attributes required for providing effective recommendations. The data pre-processing process involves the four stages namely, data cleansing, user identification, session identification and content retrieval [8]. The output of one stage is an input to the next. The stages can be described as follows:

a) **Data Cleansing:** In this stage, unnecessary attributes were filtered and only the ones required were stored in the database. Since the username acts as a primary key, the username and the URL path were the only required fields.
b) **User identification:** Since URL rewriting is used, the name of the user appears in the URL path; hence, the username is available in the web log records whenever he logs in. Two users cannot have the same username and therefore, username is used for uniquely identifying users.
c) **Session Identification:** For every user a count is maintained where the count is equal to the number of times the user has visited the website. Every time the user logs into the website, the count is incremented by one.
d) **Content Retrieval:** Content retrieval is the stage in which the user request is processed to retrieve only the path accessed by the user.
Ex: Consider the query:
http//www.bookMyFood.com/hotelinfo.jsp?hid=65
here, the query retrieved will be processed to obtain only the hotel id 65 and this information will be stored in the user's records.

**Step 3:** The hotel id available after the content retrieval stage is then stored in the database. Each hotel name along with its details is associated with a unique hotel id and this hotel id is stored in the user's records whenever he visits the hotel's page. Moreover, two tables have been maintained. First one is to maintain the entire history of the user till date. Here, the username alone is the primary key. Another one is used to maintain the information about every session of all the users. In this table, the username along with session id acts as the primary key.

## 3.2 Recommendations

The system includes two modes of recommendations.
(a) Simple Mode Recommendation.
(b) Advanced Mode Recommendation

### *3.2.1 Simple Mode*
Here, the user can search for restaurants by specifying the search criteria. The user can enter the price range, location, cuisine and so on. The system will then return the results that best suit the user's requirements. The user can also ask the system to dilute some requirements by choosing options like 'find something cheaper' or 'consider nearby locations'. This mode has been made for new users whose access patterns are not available with the system. This mode enables users to search for restaurants if they have some specific requirements in this session that they normally do not prefer. List of restaurants that satisfy the entered preferences are displayed. The hotels with highest ratings are displayed first.

### *3.2.2 Advanced Mode*
Here the recommendation is done in 3 ways:

- **Based on similar user:** The access pattern of all the users is available in the database in the form of history. In order to determine the similar users of a particular user, LCS (Least Common Subsequence) algorithm has been used. This algorithm is explained as follows:

  1. The history of current user and all other users are taken from the database.
  2. A threshold of 70 per cent has been set. Two users will be considered similar users only if their access patterns match at least up to 70 per cent.
  3. The access pattern of the current user is then compared with the access pattern of other users one by one.
  4. Thus the similar user list of the current user is obtained.
  5. Then the access patterns of the similar users are checked to determine the hotels not yet visited by current user, but visited by similar users.
  6. These unvisited hotels are then recommended to the current user.
  7. The order in which the restaurants are recommended depends on the ratings given to the restaurants.

- **Based on User's own access pattern:** Here the following steps are followed:
  1. When the user logs in and starts browsing restaurants, his LSW (Live Session Window) is captured and the accessed restaurants are recorded by the system.
  2. This information is used for comparing his current session's browsing patterns with all the previous sessions.
  3. This method determines the restaurants preferred by the user in previous sessions when he had similar browsing patterns.
  4. Now the preferred restaurants which have not been visited in this session are recommended.

- **Based on cuisine and location:** This is done as follows:
  1. During registration, user is asked for his Favourite Cuisine and preferred location. This is saved in the database.
  2. When the user logs in for the first time, he gets recommendation about the favourite cuisine (ex: Punjabi) that he mentioned during registration.
  3. Now in the subsequent sessions, his history is checked. The types of hotels that he visited more are recorded (ex: Chinese) and accordingly his favourite cuisine is updated in the database (say from Punjabi to Chinese).
  4. Also the location the user visited maximum number of times is determined from the database records
  5. The restaurants providing the user's favourite cuisine at his favourite location is displayed.

## 3.3 New Hotel Notification
The manager who is the system admin has the privilege to add a hotel, delete a hotel and update menu. A separate table is maintained for the new hotels. Whenever new hotel information is added by the manager, the system will include the hotel into the 'new hotels' as well as 'all hotels' list. The hotel information remains in the new hotel table for a period of 2 months. Notifications/ Advertisements of a new hotel are sent to the users whose records include the cuisine provided by the hotel, as their favourite cuisine.

## 4. DISCUSSION
We now have a fair idea about how collaborative and knowledge-based recommender systems work. Instead of asking the user to enter the data every time, our system performs the required operations at the back end without user's knowledge. These operations are performed on user browsing patterns. The user will be asked to enter information only if he needs something totally different from his usual preferences or if the user is new. In other words, our system only performs mining on web log records but is still capable of performing all the tasks done by the previous techniques. This feature makes our technique different from the existing techniques. We shall now discuss some of the most important weaknesses of the previous techniques [4] and also how our system tries to overcome these.

## 4.1 Data Sparseness
Consider any e-commerce system based on collaborative filtering where the number of users and items is very large. Therefore, these systems are based on very large datasets. Hence the user item matrix could be very large. There could be cases where users must have not rated all the items in the database. As a result, most of the cells of the user-item matrix are left empty as the users have not liked the item or may have not bothered to provide the ratings for that particular item. This could lead to large data sparseness i.e. the number of empty cells in the matrix. In our system, we have calculated the similarity between the users based on their browsing patterns and not the ratings only. We have used LCS algorithm for this which is very efficient as it regards the two users similar only if their browsing path matches up to threshold value T. (in our system, T>=70%). This means that the system does not rely completely on users to input some information in order to work efficiently. Hence the problem of sparseness does not arrive at all. Moreover, the recommendations are dynamic and the information about the users is regularly updated by the system according to his changing browsing patterns.

## 4.2 Cold Start
Second major weakness is that as collaborative filtering methods recommend items based on users' past preferences, the very first users or absolutely new users will need to rate sufficient number of items to enable the system to capture their preferences accurately. Hence, there is very little data to work on at the beginning in order to provide reliable

recommendations. Similarly, new items also have the same problem. When new items are added to system, they need to be rated by substantial number of users before they could be recommended to users who have tastes similar to the ones who rated them [4]. Hence there is very little opportunity to refine searches. This is known as cold start problem. In our system, in order to overcome this problem, simple mode of recommendations has been provided where the user can specify his requirements. Also user preferences are taken at the time of registration which is used to provide recommendations. Notifications are sent to the users whenever a new hotel opens. In other words, our recommender system has taken care of all faults in previous systems.

## 4.3 Shilling attacks

In most of the rating based recommendation systems we come across something called as "Shilling attacks, where people may give lots of positive ratings for their own items and negative ratings for their competitors. It then becomes a necessity for the collaborative filtering systems to use precautions to discourage such kind of manipulations. In our system, as we do not solely provide recommendations on the basis of user ratings, this hurdle is very much avoided. We use the user's browsing patterns for recommending and also the similarity between the users is checked for against all the users in the database. So, even if the small sets of users continuously browse their own restaurants, our algorithm for recommendation is flexible and dynamic; thus, other users are not affected as the system checks for the similarity of the user with all the users in the database. Hence, this shilling attack is not possible in our method of Recommendation.

## 4.3 Static Recommendations

In knowledge based recommender systems, the knowledge about the product is used and the product that best meets the user's requirements is returned. However, every time the user has to enter his requirements and the system does not study the user's behavior to provide personalized recommendations in subsequent sessions. To overcome this problem, recommendations based on user's past history are also provided so that the accuracy of recommendations is improved with increased number of visits.

## 5. TOOLS USED FOR IMPLEMENTATION

- **JSP/SERVLETS:**

A Servlet is an object that receives a request and generates a response based on that request. The basic servlet package defines Java objects to represent servlet requests and responses, as well as objects to reflect the servlet's configuration parameters and execution environment. The package javax.servlet.http defines HTTP-specific subclasses of the generic servlet elements, including session management objects that track multiple requests and responses between the Web server and a client. Servlets may be packaged in a WAR file as a Web application.

Servlets can be generated automatically from Java Server Pages (JSP) by the Java Server Pages compiler. The difference between Servlets and JSP is that Servlets typically embed HTML inside Java code, while JSPs embed Java code in HTML.

In our project, with the help of java servlets, we perform the process of extraction as soon as the user logs out. The details of the user's session are extracted from the log records and stored in the database. Clusters of similar users are created and recommendation is done online i.e. whenever the user asks for recommendation. Moreover, with the help of servlets, strings of access patterns of the user in different sessions are compared. Also, cuisine and location preferences have been determined using the same. In other words, all the three types of recommendations have been implemented using servlets. JSP has been used for database connectivity and session management.

- **HTML:**

HTML is used to develop all static pages for our website. CSS was used in addition to give a better look to GUI pages and thus further enhancement to the pages was done.

- **Apache-Tomcat Server**:

This is the local server that we have used to run our application on the browsers. We have chosen this Server because it is used for running JSP/servlets and also allows us to record and store all the access log files on which we perform our basic mining tasks**.**

- **MS-SQL 2005:** These were used for storing user and hotel information respectively at the back end.

## 6. CONCLUSION

In this paper, we propose architecture for a hybrid restaurant recommender system. A novel approach for recommendations of unvisited restaurants has been implemented. Our system selects one of the two approaches; collaborative filtering and knowledge based approach, depending on the situation. Collaborative filtering enables personalized recommendations and does not require domain knowledge. On the other hand, knowledge based approach does not face ramp – up. Thus, the plus points of both the systems are present in our hybrid architecture. In all, three techniques of recommendation have been proposed keeping into mind user's intuition as well as his preferences. The practical implementation of proposed architecture shows that the recommendations obtained are highly accurate; but this accuracy comes at the stake of efficiency. Our proposed architecture faces two challenges: the computational cost and the search cost in a database of users' preferences that tends to get larger and larger. A possible solution to the former challenge is multi-threading. That is, we can use a lightweight process (or thread) to perform computation in the background, while the user is interacting with the main process in the foreground. A possible solution to the latter challenge is data partition. That is, the database can be partitioned into several sub-tables, using some category such as alphabetical ordering. Thus, a search can be done in an appropriate sub-table, rather than in the entire database. Future work may thus involve implementing the above solutions into the system and making it more efficient thereby improving the response time.

## 7. ACKNOWLEDGMENT

We take this opportunity to express our gratitude towards our project guide Mrs. Jyothi Rao of Department Of Computer Engineering, K.J. Somaiya College of Engineering, Mumbai for her constant encouragement and guidance.

## 8. REFERENCES

[1] P. Resnick and H. R. Varian. Recommender systems. Communications of the ACM, 40(3):56–58, 1997

[2] R. Burke. Integrating knowledge-based and collaborative-filtering recommender systems. Papers from The AAAI Workshop on Artificial Intelligence for Electronic Commerce WS-99-01, AAAI Press, Menlo Park, California, 1999

[3] M. Montaner, B. Lopez, and J. L. Dela. A taxonomy of recommender agents on the internet. Artificial Intelligence Review, 19:285–330, 2003

[4] "Developing a Restaurant Recommender System" by Fredrik Kalseth, Supervised by Marilyn Walker.

[5] 'Analysis of Recommender Systems' Algorithms' by Emmanouil Vozalis, Konstantinos G. Margaritis

[6] Personallogic recommender system: http://www.personallogic.com.

[7] 'Combining Collaborative Filtering and Knowledge-Based Approaches for Better Recommendation Systems', Thomas Tran School of Information Technology and Engineering University of Ottawa

[8] 'Web usage mining: Discovery and Applications of usage patterns from web data' by Jaideep Srivastava, Robert Cooley.

[9] 'Web Usage Mining' By *Bamshad Mobasher.*