A SEMINAR REPORT ON

Role of Recommendation Systems in Improvising Businesses

SUBMITTED BY
SHUBHAM VIJAY BOTHARA
ROLL NO. 306033

SEMINAR GUIDE
SHEETAL GIRASE

Department of Information Technology
MAHARASHTRA INSTITUTE OF TECHNOLOGY
PUNE 411038

2014-15

MAHARASHTRA ACADEMY OF ENGINEERING & EDUCATIONAL RESEARCH'S

MAHARASHTRA INSTITUTE OF TECHNOLOGY PUNE

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that SHUBHAM VIJAY BOTHARA ROLL NO. 306033 of T. E. IT successfully completed seminar on

ROLE OF RECOMMENDER SYSTEMS IN IMPROVISING BUSINESSES

to my satisfaction and submitted the same during the academic year 2014-15 towards the partial fulfillment of the degree of Bachelor of Engineering in Information Technology of University of Pune, under the Department of Information Technology, Maharashtra Institute of Technology, Pune.

Sheetal Girase
Assistant Professor
Seminar Guide

Dr. Debajyoti Mukhopadhyay
Dean (R&D), MIT Group of Institutions
Professor & Head of IT
Maharashtra Institute of Technology

Acknowledgements

I would like to express my gratitude and appreciation to all those who gave me the possibility to complete this report.

A special thanks to my seminar guide, **Sheetal Girase**, whose help, stimulating suggestions and encouragement, helped and guided me in writing this report and preparing the presentation.

I would also like to acknowledge with much appreciation the crucial role of **Vidya Deshpande** in coordinating the seminar.

Special thanks to thank my group mates, **Tanmay Bangale, Akshay Jadhav and Purvesh Kothari**, who helped and gave me suggestions on writing this report.

Shubham Bothara

Abstract

The amount of information available is immensely increasing and the processing of the relevant information to each user consequently requires huge cognitive efforts. In order to cope with this information overload problem, personalized information retrieval technology focusing on the user's exact needs is desired.

Recommendation systems are widely used on the Internet to assist customers in finding the products or services that best fit with their individual preferences. There exist two basic types of recommendation systems, namely Content-based filtering and Collaborative filtering, supported by Knowledge-based and Demographic recommendation systems. Content-based filtering methods examine items previously favored by the actual user, whereas collaborative filtering computes recommendations based on the information about similar items users (user-user collaborative filtering) or (item-item collaborative filtering).

Contents

List of Figures

List of Tables

List of Abbreviations

Sr. No.	Abbreviations		
1	RS	-	Recommendation Systems
2	CB	-	Content Based
3	TFIDF	-	Term Frequency * Inverse Document Frequency
4	CF	-	Collaborative Filtering
5	DM	-	Demographic
6	KB	-	Knowledge Based
7	STS	-	Social tagging systems

CHAPTER 1

INTRODUCTION TO PROJECT:

Recommendation System using Hybrid approach in Ubiquitous environment

1.1 Introduction to Project

With the rapid development and popularization of Internet, the information technology has penetrated into every corner of our social life. We are really in an "information explosion" era. In the face of the vast resources on the Internet, people often feel be at a loss what to do, not know what course to take, the emergence of the so-called "information overload" and "information lost" phenomenon. In view of all kinds of the information, only to rely on manual methods to collect and collate the required information is clearly not enough. So, automatic collection and collation of all kinds of information is a challenge and opportunity for development.

The hybrid recommendation approach is a promising in Recommendation Systems. To overcome the limitations of existing Recommendation Systems, we decided to follow a hybrid approach that combines both methods in a ubiquitous environment. It can provide with a lot of synergies compared to simple basic recommendation algorithms. Ubiquitous computing is a concept in software engineering where computing is made to appear everywhere and anywhere, i.e. ubiquitous computing can occur using any device, in any location, and in any format.

1.2 Motivation behind project topic

Although many different approaches to automated recommender systems have been developed within the past few years, the interest in this area still remains high. This is due to growing demand on practical applications, which are able to provide personalized recommendation and deal with information overload. With various technologies becoming available across platforms, ubiquitous computing must be must be a part of Recommender

Systems too. Recommender systems constitute an own independent research field, containing dozens of sub-domains and interdisciplinary aspects that are worth to be studied.

1.3 Objective(s) of the work

We aim to satisfy the following requirements:

- It should provide efficient way of recommending items to the user with very high recommendation accuracy.
- To provide ubiquitous environment for the user of the system.
- System should retrieve relevant results while solving the new-item problem, enhancing the variety in results.
- System should be stable. It should retrieve relevant results for the user irrespective of the environment.

1.4 Introduction to Seminar Topic

The Internet has no dearth of content. The amount of available information is immensely increasing and the processing of the relevant information to each user consequently requires huge cognitive efforts. The overwhelming amount of data necessitates mechanisms for efficient information filtering. The reliance on information in our daily lives is getting bigger. The challenge is in finding the right content for yourself: something that will answer your current information needs or something that you would love to read, listen or watch. In order to cope with this information overload problem, personalized information retrieval technology focusing on the user's exact needs is desired. In this Seminar we discuss various types of existing Recommendation Systems.

CHAPTER 2

LITERATURE SURVEY

Recommender Systems (*RS*) are defined as software tools and techniques providing users with suggestions for items a user may wish to utilise [1]. The concept of RSs generally grows out of the idea of information reuse, and persistent preferences. RSs are widely used on the Internet to assist customers in finding the products or services that best fit with their individual preferences. They represent a powerful method for enabling users to filter through large information and product spaces. A result from a RS is understood as a recommendation, an option worthy of consideration [3].

2.1 Need for Recommendation Systems

Historically, people have relied on recommendations and mentions from their peers or the advice of experts to support decisions and discover new material. If a person wishes to watch a movie, he/she relies on the review of critiques. The Collaborative Filtering (*CF*) based RS is based on similar notion. CF computes recommendations based on an assumption that if the individual agreed in past with some users, then the recommendations coming from these similar users should be relevant to the active user.

RSs have become extremely common in recent years, and are applied in various domains. The most popular RS applications are books, movies, music, restaurants, e-commerce, news etc. They assist users to analyse and help them analyse available books, articles, webpages, movies, music, restaurants, etc. to find the most valuable and valuable information for them. RSs are primarily directed towards users who lack sufficient experience or competence to evaluate the potentially overwhelming number of alternatives available [2].

2.2 Tasks of Recommendation Systems

Herlocker has a list of eleven popular tasks a RS can assist to implement: [1]

- **Find Some Good Items**: The RS recommends a featured list of ranked items found that fit the user's requirements. This is the main recommendation task that many commercial systems address.
- **Find All Good Items**: In some cases, the RS recommends a list of all the items that satisfy all the criteria the user set from the item database. This is used when the number of items is relatively small or in a critical situation like in medical applications.
- **Annotation in Text**: Based on current context, the RS recommends a list of items according to the user's long term preference. E.g. A certain TV series on a certain channel can be recommended according to the user's long term viewing habits.
- **Recommend A Sequence**: Here the idea is that instead of recommending single items, a sequence is recommended that is pleasing is a whole. E.g. A music playlist.
- **Recommend A Bundle**: The RS recommends a list of related items that can work together to serve a purpose better for the user. Typically when you buy a camera, you may consider buying a memory card, a pouch and complete the purpose of the camera.
- **Just Browsing**: For users that browse without a prominent purpose, the RS's task is to assist the user to browse items within the scope that are interesting to the user at that specific browsing session.
- **Find Credible Recommender**: There are users that are skeptical of the recommendation yielded by the system. They do not trust the system. The RSs task is then to allow the user to test the system's behavior.
- **Improve The Profile**: The system can take inputs from the user about his/her likes and dislikes, in general, explicit preference information. This is necessary to provide personalized recommendations. E.g. Amazon Betterizer.
- **Express Self**: Some users care little about recommendations, but it is important for them to be able to express their opinions and beliefs of certain item. A comment section is where the system can take such inputs, and also the satisfaction it creates can leverage as a motivation for purchasing the item related.
- **Help Others**: Certain users may wish to leave a full review or rating of the item as they believe their contribution will benefit the community. And that can be hugely motivational for other potential buyers to set their minds.

- **Influencing Others**: Certain users could be exclusively influential, trying to convince other users buying or nor buying the product. Even malicious user can fall into this category.

2.3 Data Collection in Recommendation Systems

When building a model from a user's profile, a distinction is often made between explicit and implicit forms of data collection.

Examples of explicit data collection include the following:

- Asking a user to rate an item.
- Asking a user to search.
- Asking a user to rank a collection of items from favourite to least favourite.
- Presenting two items to a user and asking him/her to choose the better one of them.
- Asking a user to create a list of items that he/she likes.

Examples of implicit data collection include the following:

- Observing the items that a user views in an online store.
- Analysing item/user viewing times.
- Keeping a record of the items that a user purchases online.
- Obtaining a list of items that a user has listened to or watched on his/her computer. Analysing the user's social network and discovering similar likes and dislikes

2.4 Types of Recommendation Systems

In order to implement its core function, identifying the useful items for the user, a RS must predict that an item is worth recommending. In order to this, the system must be able to predict the utility of some of them, or at least compare the utility of some items, and then decide what items to recommend based on the comparison [2]. According to the taxonomy provided by [3], RSs are distinguished into six different classes based on their knowledge source primarily as shown in Figure 2.1.

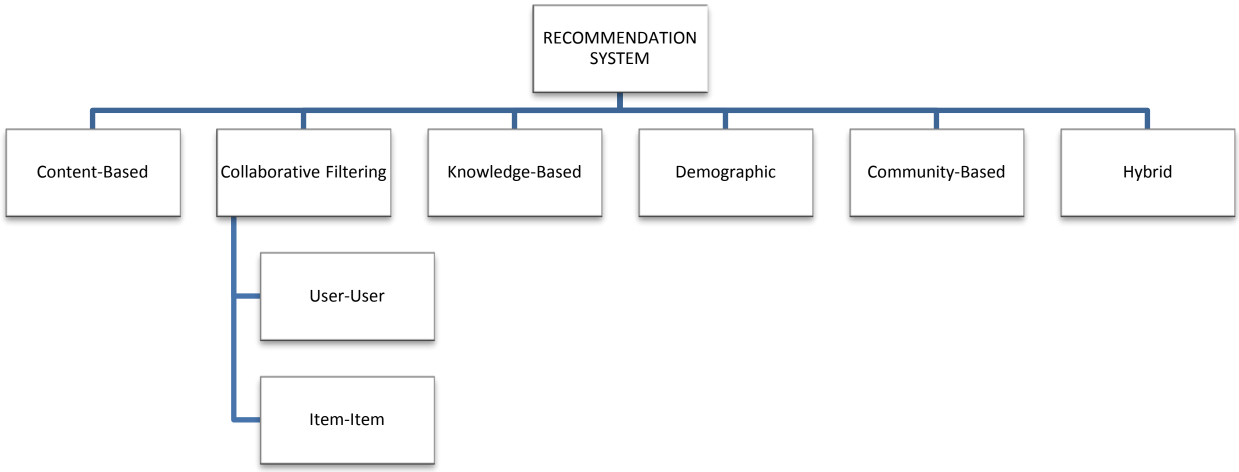

Figure 2.1 Types of Recommendation Systems

In subsequent chapters we shall discuss all the types mentioned in Figure 2.1

CHAPTER 3

CONTENT-BASED RECOMMENDATION SYSTEMS

Content Based (*CB*) RS are based on a description of the item and a profile of the user's preference. They try to recommend items similar to those a given user has liked in the past [8]. A user profile is built to indicate the type of item this user likes. These algorithms try to recommend items that are similar to those that a user liked in the past or is examining in the present. Various candidate items are compared with items previously rated by the user and the best-matching items are recommended. The result is a relevance judgment that represents the user's level of interest in that object [8].

If a profile accurately reflects user preferences, it is of tremendous advantage for the effectiveness of an information access process. For instance, it could be used to filter search results by deciding whether a user is interested in a specific Web page or not and, in the negative case, preventing it from being displayed. This approach has its roots in information retrieval and information filtering research.

3.1 Methodology

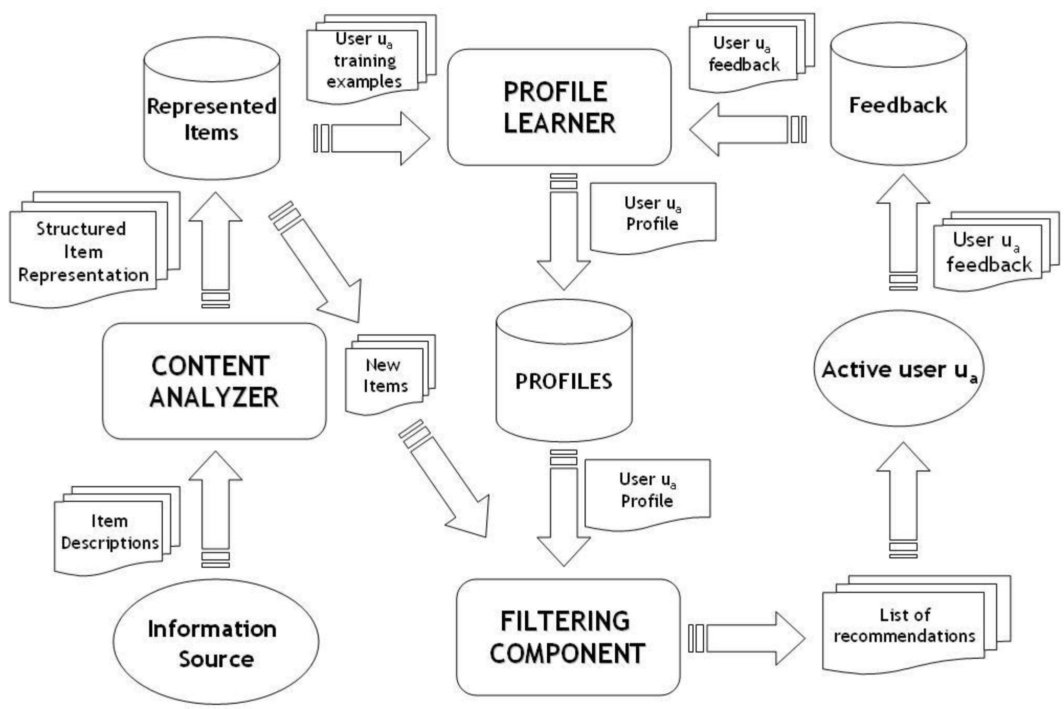

Figure 3.1 High Level Architecture of a Content-Based Recommender System [1]

The high level architecture of a content-based Recommendation system is depicted in Figure. 3.1. The recommendation process is performed in three steps, each of which is handled by a separate component:

- **Content Analyser**: When information is unstructured (e.g. text), some kind of pre-processing step is needed to extract structured relevant information. The main responsibility of content analyser is to represent the content of items coming from different information sources in a form suitable for the next processing steps. Data items are analysed by feature extraction techniques in order to shift item representation from the original information space to the target one. This representation is the input to the Profile Learner and Filtering Component.
- **Profile Learner**: The Profile Learner collects tries to generalize the data based on user's preferences, in order to construct the user profile.
- **Filtering Component**: The Filtering Component compares the user's profile representation against that of items to be recommended. The result is a binary or continuous relevance judgment. The matching is realized by computing the cosine similarity between the prototype vector and the item vectors.

To extract the features of the items in the system, an item presentation algorithm is applied. A widely used algorithm is the **TFIDF** representation (also called vector space representation). TFIDF stands for Term Frequency * Inverse Document Frequency. It is a numerical statistic that is intended to reflect how important a word is to a document in a collection.

- Term Frequency: Number of occurrences of a term in the document (can be a simple count)
- Inverse Document Frequency: How few documents contain this term. Typically log (no. of documents / no of documents with term)

The universe of all possible keywords of items defines a **Content Space**. Each keyword is a dimension of that content space. Each item has a position in that space; that position defines a vector. Each user has a taste profile that is also a vector in that space. The match between user preference and items is measured by how closely the two vectors align.

Consider the Figure. 3.2 depicting the content space in two dimensions –Action and Romance in a movie. Now in the very simplest model, if there is a movie with only action and no romance it would be on X-Axis. Similarly, if there is a movie with only romance and no action it would be on X-Axis. Now, consider movies Movie 1 and Movie 2 with more action and less romance and more romance and less action respectively. We can express each of these movies as a vector in graphical notation.

So this is the concept of putting things in a space. The way the vector space model becomes interesting is that we could also put user tastes in the same space. And you can compute how much the user is likely to enjoy a particular movie by the angle between the movie, and the user's taste profile. The cosine angle between the user's taste profile and Movie 1 is greater than the cosine angle between the user's taste profile and Movie 2. Hence there is a higher probability for the user to like Movie 1 than Movie 2.

This can also be thought of as the dot product between the user's taste profile vector and any Movie vector; greater the value of dot product higher the probability for the user to like the movie.

This same concept works even as you go from two dimensions to three to 300, to 3,000, and beyond.

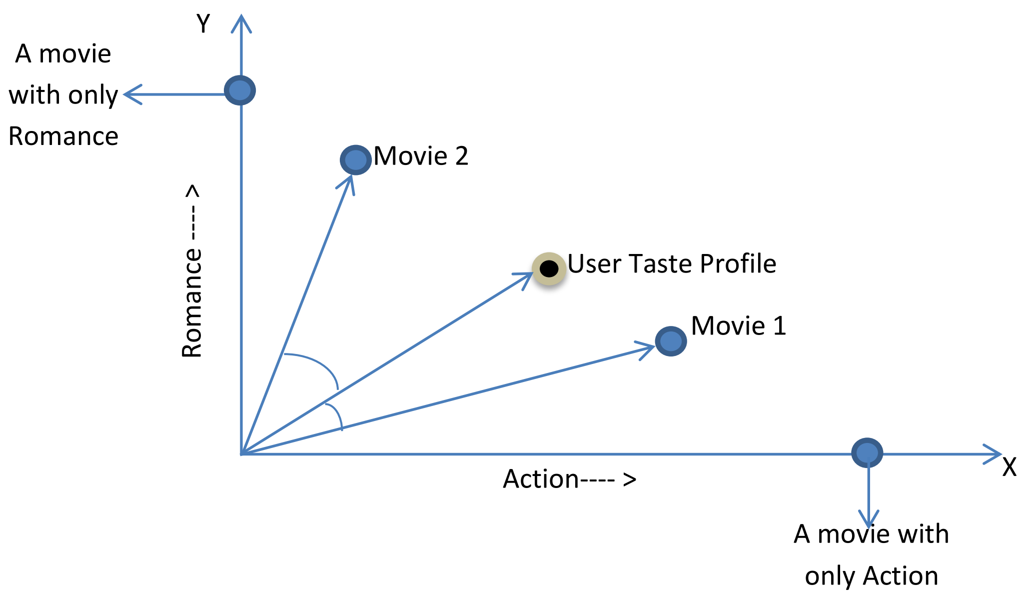

Figure 3.2 Content Space of Movies Based On Two Genres (Dimensions): Action and Romance

To create user profile, the system mostly focuses on two types of information:

- A model of the user's preference.
- A history of the user's interaction with the Recommendation system.

The RS creates a content-based profile of users based on a weighted vector of item features. The weights denote the importance of each feature to the user and can be computed from individually rated content vectors using a variety of techniques. Simple approaches use the average values of the rated item vector while other sophisticated methods use machine learning techniques such as Bayesian Classifiers, cluster analysis, decision trees, and artificial neural networks in order to estimate the probability that the user is going to like the item.

3.2 Advantages:

- **User Independence**: Content-based Recommendations exploit solely ratings provided by the active user to build her own profile.
- **Transparency**: Explanations on how the RS works can be provided by explicitly listing content features or descriptions that caused an item to occur in the list of recommendations. Those features are indicators to consult in order to decide whether to trust a recommendation.
- **New Item**: Content-based recommenders are capable of recommending items not yet rated by any user. As a consequence, they do not suffer from the new-item problem, which affects Collaborative RS.

3.3 Limitations

- **Limited Content Analysis**: Content-based recommendation system cannot provide suitable suggestions if the analysed content does not contain enough information to discriminate items the user likes from items the user does not like.
- **Over-Specialization**: The system is limited to recommending content of the same type as the user is already using, the value from the recommendation system is significantly less than when other content types from other services can be recommended. Content-based recommenders have no inherent method for finding something unexpected. This drawback is also called serendipity problem.
- **New User**: Enough ratings have to be collected before a content-based recommender system

can really understand user preferences and provide accurate recommendations. Therefore, when few ratings are available, as for a new user, the system will not be able to provide reliable recommendations.

3.4 Examples

- **Pandora Radio**: Pandora uses the properties of a song or artist in order to select a "station" that plays music with similar properties. User feedback is used to refine the station's results, deemphasizing certain attributes when a user "dislikes" a particular song and emphasizing other attributes when a user "likes" a song
- **Internet Movie Database (*IMDb*)**: The IMDb is an online database of information related to films, television programs, and video games, including cast, production crew, fictional characters, biographies, plot summaries, trivia and reviews. It uses this information to recommend movies to a user. The IMDb Top 250 list is a listing of the top rated 250 films of all-time, based on ratings by the registered users of the website using the methods describe

CHAPTER 4

COLLABORATIVE-FILTERING RECOMMENDATION SYSTEMS

Collaborative Filtering (*CF*) is a popular recommendation algorithm that bases its predictions and recommendations on the ratings or behavior of other users in the system. It is the process of filtering information or patterns using techniques which involves collaboration among various agents, viewpoints, data sources, etc. [1]. The information domain for a CF system consists of users which have expressed preferences for various items. The fundamental assumption behind this method is that users' past agreement can be used to predict their future agreement. This implies that

- Users' tastes are either individually stable or move in sync with each other
- The RS has scope within a domain of agreement

So, for example, if user A and user B have a purchase history that overlaps strongly and user A has recently bought an item that B has not yet been, the basic rationale is to propose this item also to B.

4.1 Methodology

Typical workflow of a collaborative filtering system is as under:

- A user expresses his or her preferences by rating items (e.g. articles, videos or books) of the system. These ratings can be viewed as an approximate representation of the user's interest in the corresponding domain.
- The system computes correlations of the users' ratings against other users and finds the people with most similar tastes.
- The system recommends the items that similar users (neighbors) have rated highly but not yet being rated by this user.

The collaborative filtering technique recommends items based on User-User based approach, Item-Item based approach and User – Item based approach [4].

4.1.1 User-User Based Approach

User–user CF is a straightforward algorithmic interpretation of the core premise of collaborative filtering: find other users whose past rating behavior is similar to that of the current user and use their ratings on other items to predict what the current user will like.

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5	Movie 6
User 1	2	5	4	2	?	
User 2	1	5	4		5	
User 3	5	5	5	5	5	5
User 4	2	4		3		
User 5	5	3	5	3		5
User 6	1	5				1
User 7	2		5		5	

Table 4.1 Sample Dataset of Ratings

From Table 4.1 we can infer that User 1, User 4, User 6 and User 7 have similar tastes while User 1 and User 2 have dissimilar tastes. From the values available we can recommend User 1 Movie 5 as User 7 recommends movie 5 while User 2 advises against it.

To generate predictions or recommendations for a user *u*, user–user CF first uses *s* to compute a neighborhood $N \subseteq U$ of neighbors of *u*. Once *N* has been computed, the system combines the ratings of users in *N* to generate predictions for user *u*'s preference for an item *i*. This is typically done by computing the weighted average of the neighboring users' ratings *i* using similarity as the weights:

$$P_{a,i} = \overline{r_a} + \frac{\sum_{u=1}^{n} \left(r_{u,i} - \bar{r}_u\right) * w_{a,u}}{\sum_{u=1}^{n} w_{a,u}}$$

Where,

$P_{a,i}$ - Prediction for user a of ith item,

$\overline{r_a}$– Average rating by user a,

$r_{u,i}$ – Rating by uth user of ith item,

$w_{a,u}$ – Weight describing agreement between user a and user u.

4.1.2 Item-Item Based Approach:

User–user collaborative filtering, while effective, suffers from scalability problems as the user base grows. In user-user CF, searching for the neighbors of a user is an O(|U|) operation (or worse, depending on how similarities are computing — directly computing most similarity functions against all other users is linear in the total number of ratings) [7]. To extend collaborative filtering to large user bases and facilitate deployment on e-commerce sites, it was necessary to develop more scalable algorithm.

Item–item collaborative filtering, also called item-based collaborative filtering, takes a major step in this direction and is one of the most widely deployed collaborative filtering techniques today. A version of it is used by Amazon.com at this time [5]. Rather than using similarities between users' rating behavior to predict preferences, item– item CF uses similarities between the rating patterns of items. If two items tend to have the same users like and dislike them, then they are similar and users are expected to have similar preferences for similar items. In its overall structure, therefore, this method is similar to earlier content-based approaches to recommendation and personalization, but item similarity is deduced from user preference patterns rather than extracted from item data.

As a user rates and re-rates items, their rating vector will change along with their similarity to other users. Finding similar users in advance is therefore complicated. A user's neighborhood is determined not only by their ratings but also by the ratings of other users, so their neighborhood can change as a result of new ratings supplied by any user in the system. For this reason, most user–user CF systems find neighborhoods at the time when predictions or recommendations are needed. In systems with a sufficiently high user to item ratio, however, one user adding or changing ratings is unlikely to significantly change the similarity between two items, particularly when the items have many ratings. Therefore, it is reasonable to pre-compute similarities between items in an item–item similarity matrix. The rows of this matrix can even be truncated to only store the k most similar items. As users change ratings, this data will become slightly stale, but the users will likely still receive good recommendations and the data can be fully updated by re-computing the similarities during a low-load time for the system.

4.1.2 Advantages

- **No Serendipity**: Serendipity is a measure "how surprising the recommendations are" [4]. Unlike CB Can identify cross-genre items.
- **Adaptive**: quality improves over time.
- Implicit feedback sufficient

4.1.3 Limitations

- **Cold Start**: These systems often require a large amount of existing data on a user in order to make accurate recommendations.
- **Scalability**: In many of the environments in which these systems make recommendations, there are millions of users and products. Thus, a large amount of computation power is often necessary to calculate recommendations.
- **Sparsity**: The number of items sold on major e-commerce sites is extremely large. The most active users will only have rated a small subset of the overall database. Thus, even the most popular items have very few ratings.

4.1.4 Examples

- **Amazon (User - Item CF)**
- **Facebook (User - User CF)**: Facebook uses CF to recommend new friends, groups, and other social connections by examining the network of connections between a user and their friends.

CHAPTER 5
ALTERNATIVE RECOMMENDATION SYSTEMS

As seen in earlier chapters, CB and CF suffer from a few limitations. Hence alternative types of RSs are needed. Some of them are explored in this chapter.

5.1 Knowledge-Based Recommendation Systems

Knowledge-based systems recommend items based on specific domain knowledge about how certain item features meet users' needs and preferences and, ultimately, how the item is useful for the user [2]. In some sense, all recommendation techniques could be described as doing some kind of inference. Knowledge-based approaches are distinguished in that they have functional knowledge: they have knowledge about how a particular item meets a particular user need, and can therefore reason about the relationship between a need and a possible recommendation. Compared to collaborative filtering and content-based filtering, knowledge-based recommenders have no cold-start problems since requirements are directly elicited within a recommendation session. However, no advantage without disadvantage, knowledge-based recommenders suffer from the so-called knowledge acquisition bottleneck in the sense that knowledge engineers must work hard to convert the knowledge possessed by domain experts into formal, executable representations. There are two types of Knowledge based RS's:

- Case Based
- Constraint Based

In terms of used knowledge, both systems are similar: user requirements are collected; repairs for inconsistent requirements are automatically proposed in situations where no solutions could be found; and recommendation results are explained [2]. The difference between the two types is the manner in which the solutions are generated. Case-based recommenders determine recommendations on the basis of similarity metrics whereas constraint-based recommenders predominantly exploit predefined knowledge bases that contain explicit rules about how to relate customer requirements with item features.

5.1.1 Advantages

- Sensitive to users changes of preference
- Can map from user needs to products

5.1.2 Limitations

- Suggestion ability static (does not learn)
- Knowledge engineering required.

5.2 Demographic Recommendation Systems

This type of RS recommends items based on the demographic profile of the user. It is based on the principle that different recommendations should be generated for different demographic niches. Many Web sites adopt simple and effective personalization solutions based on demographics. For example, users are dispatched to particular Web sites based on their language or country. Or suggestions may be customized according to the age of the user. While these approaches have been quite popular in the marketing literature, there has been relatively little proper RS research into demographic systems [2].

5.2.1 Advantages

- **No Serendipity**: Serendipity is a measure "how surprising the recommendations are" [4]. Unlike CB Can identify cross-genre items.
- **Adaptive**: quality improves over time.
- Implicit feedback sufficient

5.2.2 Limitations

- **Cold Start**: These systems often require a large amount of existing data on a user in order to make accurate recommendations.
- **Scalability**: In many of the environments in which these systems make recommendations, there are millions of users and products. Thus, a large amount of computation power is often necessary to calculate recommendations.

5.3 Community-Based Recommendation Systems

This type of system recommends items based on the preferences of the users friends. This technique follows the epigram “Tell me who your friends are, and I will tell you who you are”. [2]. It has been observed that people tend to rely more on recommendations from their friends rather than on recommendations from similar but anonymous individuals. With this observation and the growing popularity of open social networks, there is a rising interest in community-based systems or, as or as they usually referred to, social RSs. This type of RSs models and acquires information about the social relations of the users and the preferences of the user's friends. The recommendation is based on ratings that were provided by the user's friends. The research in this area is still in its early phase and results about the systems performance are mixed.

5.4 Hybrid Recommendation Systems

Hybrid RSs combine two or more recommendation techniques to gain better performance with fewer of the drawbacks of any individual one. It tries to achieve synergy between different RSs. A hybrid system combining techniques A and B tries to use the advantages of A to fix the disadvantages of B. For instance, CF methods suffer from new-item problems, i.e., they cannot recommend items that have no ratings. This does not limit content-based approaches since the prediction for new items is based on their description (features) that are typically easily available. According to [3] there are various hybridization methods as described in Table 5.1

Hybridization method	**Description**
Weighted	The scores (or votes) of several recommendation techniques are combined together to produce a single recommendation.
Switching	The system switches between recommendation techniques depending on the current situation.
Mixed	Recommendations from several different recommenders are presented at the same time
Feature combination	Features from different recommendation data sources are thrown together into a single recommendation algorithm.
Cascade	One recommender refines the recommendations given by another.
Feature augmentation	Output from one technique is used as an input feature to another.
Meta-level	The model learned by one recommender is used as input to another.

Table 5.1 Hybridisation Methods [3]

Given two (or more) basic RSs techniques, several ways have been proposed for combining them to create a new hybrid system. We have considered combinations between 2 basic RSs in Table 5.2.

There are four hybridization techniques that are order-insensitive: Weighted, Mixed, Switching and Feature Combination. They do not depend on the order they are applied: a CN/CF mixed system would be no different from a CF/CN one. Such redundant combinations are marked in gray.

Some combinations do not provide any significant improvements. Such insignificant combinations are marked black. The cascade, augmentation and meta-level hybrids are ordered, i.e. they depend on the order they are applied For example, a feature augmentation hybrid that used a content-based recommender to contribute features to be used by a second collaborative process, would be quite different from one that used collaboration first.

	Weighted	**Mixed**	**Switching**	**Feature Combination**	**Cascade**	**Feature Aug.**	**Metalevel**
CF/CN	P-Tango	PTV, ProfBuilder	DailyLearner	(Basu, Hirsh & Cohen 1998)	Fab	Libra	
CF/DM	(Pazzani 1999)						
CF/KB	(Towle& Quinn2000)		(Tran&Cohen, 2000)				
CN/CF							Fab,(Condliff, et al. 1999), LaboUr
CN/DM	(Pazzani 1999)			(Condliff, et al.1999)			
CN/KB							
DM/CF							
DM/CN							
DM/KB							
KB/CF					EntreeC	GroupLens (1999)	
KB/CN							
KB/DM							

(CF = collaborative, CN = Content-based, DM = Demographic, KB = Knowledge Based)

Table 5.2 Possible and Actual (or Proposed) Recommendation Hybrids [3]

CHAPTER 6

CASE STUDY:

User-Item Collaborative Filtering by Amazon

To understand the practical implementation of RS, the working of Amazon's personalised Recommendation System is considered. Amazon implements a unique model of Collaborative Filtering: The User – Item CF Model.

6.1 Introduction

Amazon.com is one of the leading e-commerce stores. The Amazon.com Web site includes functionality for allowing users to search, browse, and make purchases from an online catalog of several million book titles, music titles, video titles, and other types of items. On Amazon.com Recommendation algorithms are used to provide a personalised experience to every user. To provide such experience it collects extensive amount of data about its user. Nearly any action taken by you while you are logged in is stored for future use. Thanks to browser cookies, it can even maintain records on anonymous shoppers, eventually linking the data to a customer profile when the anonymous shopper creates an account or signs in.

As the existing recommendation algorithms could not scale to Amazon.com' tens of millions of customers and products, so they invented their own algorithm, item-to-item collaborative filtering. It scales to massive data sets and produces high-quality recommendations in real time.

The method of recommending items to users from a database of items, suggested by Amazon comprises:

- maintaining item selection histories of each of a group of users of a server system that provides functionality for browsing and selecting items from an electronic catalog of items, each item selection history corresponding to, and identifying items selected by, a particular user [11].
- Collectively analyzing at least the item selection histories of the plurality of users, as collected over a period of time, in an off-line processing mode to generate a plurality of data

values that represent degrees to which specific items in the electronic catalog are related [11].

- Storing a selected subset of the plurality of data values in a mapping structure that maps items to related items [11].
- For each of a plurality of users of the electronic catalog, using the mapping structure, including the data values stored therein, to generate personalized recommendations of items within the catalog [11].

6.2 Methodology

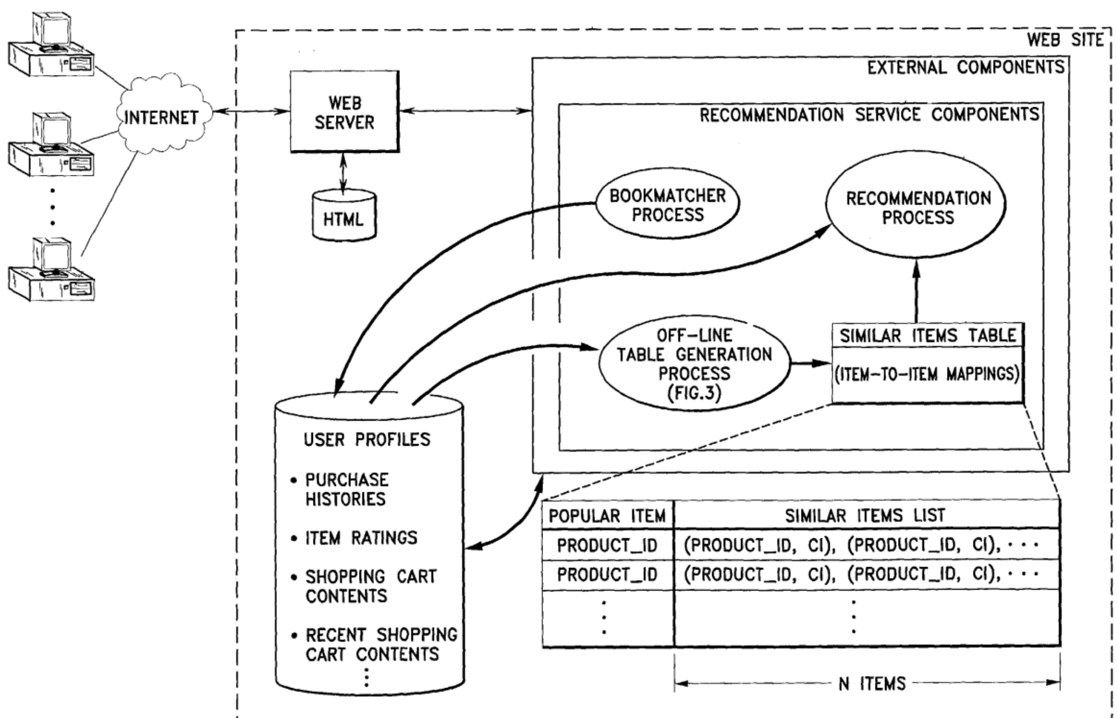

Figure 6.1 Functioning of Amazon.com

Figure 6.1 illustrates Amazon's recommendation service. The arrows in Figure 6.1 show the general flow of information that is used by the Recommendation Service.

The Web site includes a "user profiles" database which stores account-specific information about users of the site. When you visit Amazon's website searching for a product it asks you to rate particular product on a five-star scale, and it also notes which product images you enlarge, which you look at multiple times, which you place on a wish list, and which you actually buy. It also tracks what products are on your screen at the time as well as others you look at during your session. The retailer uses the path you've travelled through its website - the pages you've viewed and items you've clicked on—to suggest complementary products and it combines your purchase data with your ratings to build a profile of your long-term preferences.

Recommendations are generated by the recommendation services are returned to the Web server, which incorporates the recommendations into personalized Web pages transmitted to users. The recommendation service components include a BookMatcher application**.** Users of the BookMatcher service are provided the opportunity to rate individual book titles from a list of popular titles. The book titles are rated according to the following scale:

1 = Bad!

2 = Not for me

3 = OK

4 = Liked it

5 = Loved it

Users can also rate book titles during ordinary browsing of the site. As depicted in Figure 6. 1, the BookMatcher application records the ratings within the user's items rating profile. The BookMatcher application uses the users' item ratings profiles to generate personal recommendations.

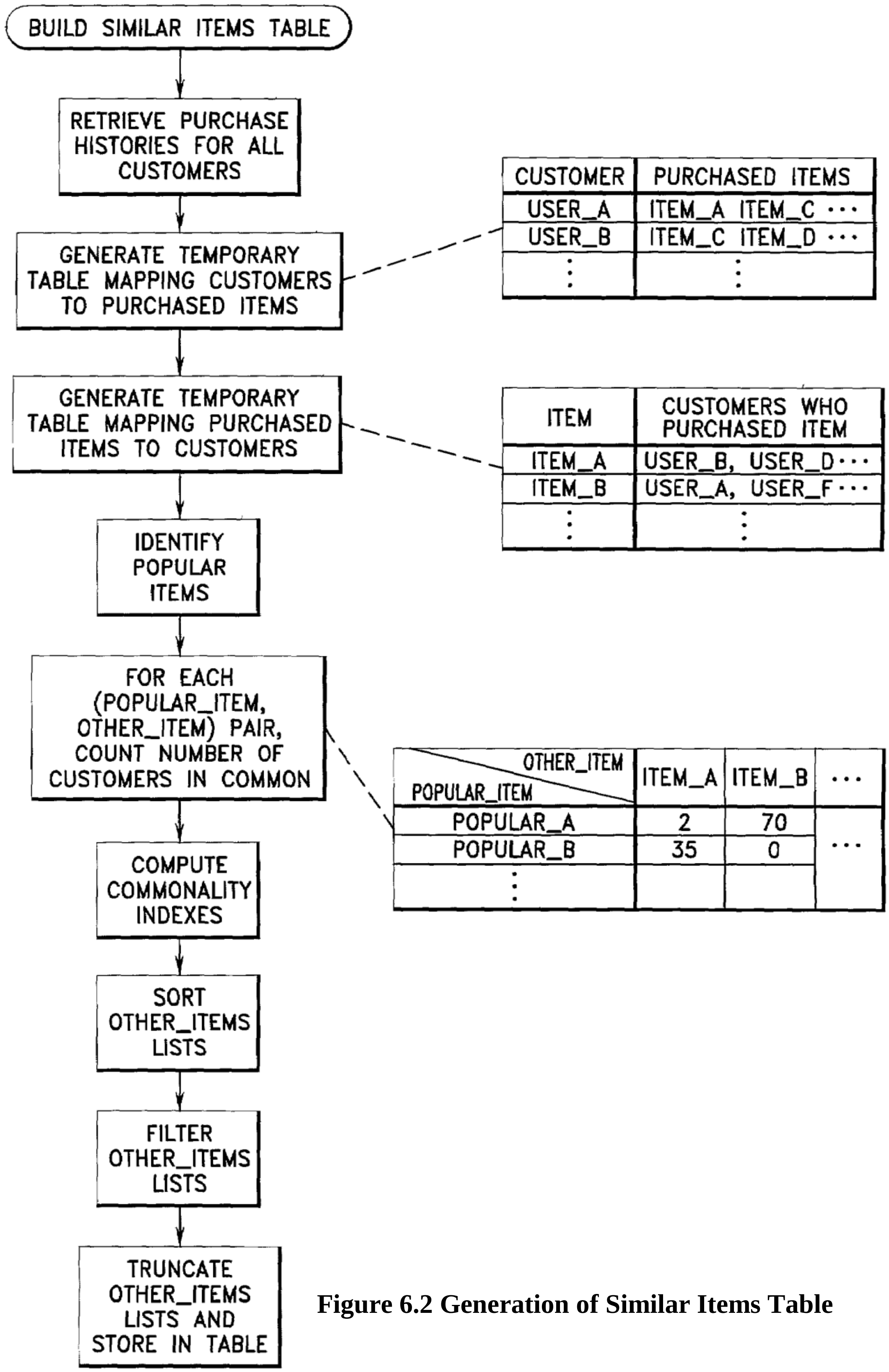

Figure 6.2 Generation of Similar Items Table

The recommendation services components also include a recommendation process, a similar items table, and an off-line table generation process, which collectively implement the Recommendation Service. As depicted by the arrows in Figure. 6.1, the recommendation process generates personal recommendations based on information stored within the similar items table and based on the items that are known to be of interest ("items of known interest") to the particular user.

The similar items table (Figure. 6.2) maps items to lists of similar items based at least upon the collective interests of the community of users. The similar items table is preferably generated periodically (e.g. once per week) by the off-line table generation process. This enables the personal recommendations to be generated rapidly and efficiently (such as in real-time in response to a request by the user), without sacrificing breadth of analysis.

The table generation process generates the table from data that reflects the collective interests of the community of users. Several different types of items (books, CDs, videos, etc.) are reflected within the same table, although separate tables could alternatively be generated for each type of item.

The Flowchart for the algorithm for generating personalized recommendations is shown Figure 6.3. As shown in Figure 6.3, the first step of the recommendations-generation process involves identifying a set of items that are of known interest to the user. The "knowledge" of the user's interest can be based on explicit indications of interest. E.g. the user rated the item highly) or implicit indications of interest like the user added the item to a shopping cart. Items that are not "popular items" within the similar items table can optionally be ignored during this step.

For each item of known interest, the service retrieves the corresponding similar items list from the similar items table, if such a list exists. If no entries exist in the table for any of the items of known interest, the process may be terminated; alternatively, the process could attempt to identify additional items of interest, such as by accessing other sources of interest information.

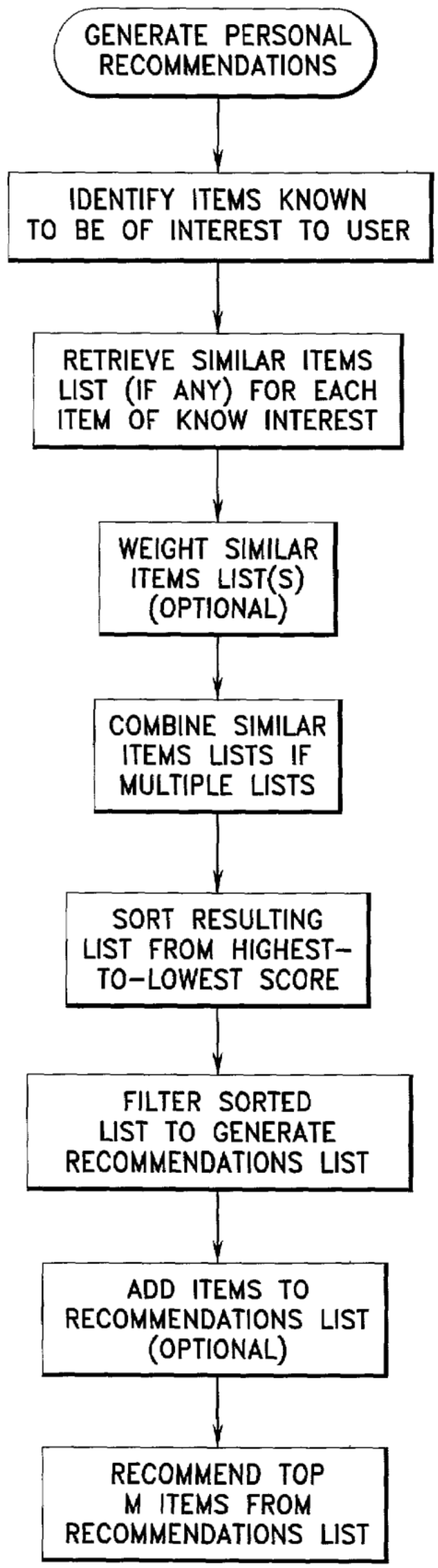

Figure 6.3 General Processing to Generate Personal Recommendations

Further on, the similar items lists are optionally weighted based on information about the user's affinity for the corresponding items of known interest. The similar items lists are generally weighted by multiplying the CI values of the list by a weighting value. The commonality index values as weighted by any applicable weighting value are referred to as “scores.”

If multiple similar items lists are retrieved in step, the lists are appropriately combined, such as by merging the lists while summing the scores of like items. The resulting list is then sorted in order of highest-to-lowest score. The sorted list is filtered to remove unwanted items. The items removed during the filtering process may include, for example, items that have already been purchased or rated by the user. The result is a list (“recommendations list”) of other items to be recommended to the user.

Then one or more additional items are optionally added to the recommendations list. The items added maybe selected from the set of items in the user's “recent shopping cart contents” list. As an important benefit of this step, the recommendations include one or more items that the user previously considered purchasing but did not purchase.

Finally, a list of the top N items from the recommendations list is returned to the Web server. The Web server incorporates this list into one or more Web pages that are returned to the user, with each recommended item being presented as a link to the item's product information page. The recommendations may alternatively be conveyed to the user by email, facsimile, or other transmission method. Further, the recommendations could be presented as advertisements for the recommended items. This general strategy is also used for generating Instant Recommendations and Shopping Cart Recommendations.

Performance

Generally to compute a recommendation time complexity $O\ (N^3)$ is required for a user-user based RS, where N is the no. of users. In case of Amazon, The offline computation of the similar-items table is extremely time intensive, with $O\ (N^2M)$ as worst case, where N is the no. of users and M the no. of items. In practice, however, it’s closer to $O(NM)$. Amazon is also certainly increasing online sales. It has been estimated that investments in recommenders bring in returns of 10 to 30 percent, thanks to the increased sales they drive.

CHAPTER 7

CONCLUSION AND RECENT TRENDS

RSs are evolving in many and diverse directions and new topics are emerging or becoming more important subjects of investigation. The various RS discussed above all have their advantages and limitations in performing their job. Most of the limitations in each one of the approaches can be complimented by the other. A good recommender system should be able to provide positive and relevant recommendations from time to time and also provide alternative recommendations to break the fatigue of the users seeing the same items in the recommendation list. Hence, hybrid systems seem to have an edge over others.

7.1 Recent Trends

As discussed above, RS are evolving. Some of the recent topics are discussed below:

7.1.1 Multi-criteria recommender systems

In majority of RSs the utility associated with an item is usually considered a single criterion value, e.g. an overall evaluation or rating of an item by a user. But recently this assumption has been judged as limited because the suitability of the recommended item for a particular user may depend on several aspects that the user can take into consideration when making his or her choice. The incorporation of multiple criteria that can affect the user's opinions may lead to more effective and accurate recommendations.

7.1.2 Social tagging systems (*STS*)

STS is a new RS-related topic that is emerging due to the growth of Web 2.0 applications. STS like Flickr or Delicious, allow the ordinary user to publish and edit content as well as generate and share tags. RSs are required to assist users in finding relevant information in STS. Tag recommendation, has different characteristics than traditional recommendations since the system can recommend recurrent tags, unlike traditional RSs. In addition, RSs for STS deal with a three-dimensional problem (user, resource, tag), rather than the traditional two-dimensional problem (user, item), and this affects the complexity of the algorithms.

CHAPTER 8
References

1. P. N. Vijaya Kumar, Dr. V. Raghunatha Reddy. "A survey on Recommender Systems (RSS) and Its applications", *IJIRCCE*, Vol. 2, Issue 8, pp. 5254-5260, Aug 2014.
2. *Recommender Systems Handbook,* 1st ed., Springer-Verlag New York, Inc. New York, NY,2010, pp 1-38.
3. R. Burke, "Hybrid web recommender systems", in *The adaptive web*, Springer Berlin / Heidelberg, 2007, pp. 377–408.
4. Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J. "An algorithmic framework for performing collaborative filtering". In: *SIGIR '99: Proc. of the 22nd Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pp. 230–237. ACM, New York, NY, 1999
5. Greg Linden, Brent Smith, and Jeremy York (2003). Amazon.com Recommendations Item-to-Item Collaborative Filtering [Online]. Available: http://computer.org/internet/
6. Sarwar, B., Karypis, G., Konstan, J., Reidl, J. "Item-based collaborative filtering recommendation algorithms". In:*WWW'01: Proc. of the 10th Int. Conf. onWorldWideWeb*, pp. 285–295. ACM, New York, NY, 2001
7. M. D. Ekstrand, J. T. Riedl and J. A. Konstan, "Collaborative Filtering Recommender Systems," in Human–Computer Interaction Vol. 4, No. 2 2010 © DOI: 10.1561/1100000009
8. Pasquale Lops, Marco de Gemmis, Giovanni Semeraro. "Content-based Recommender Systems: State of the Art and Trends", in *Recommender Systems Handbook*, Springer-Verlag New York, Inc. New York, NY 2011 pp. 73-105.
9. G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 6, pp. 734–749, 2005.
10. Prem Melville, Vikas Sindhwani "Recommender Systems," 2010 © Springer Science+Business Media, LLC. doi: 10.1007/978-0-387-30164-8_705
11. Amazon.com, Inc. "Personalized recommendations of items represented within a database," U.S. Patent 7 113 917, Sep, 26, 2006.