Recommender Systems Handbook

Chapter · October 2010

DOI: 10.1007/978-0-387-85820-3_1

3 authors:

Francesco Ricci
Free University of Bozen-Bolzano
284 PUBLICATIONS **12,666** CITATIONS

SEE PROFILE

Lior Rokach
Ben-Gurion University of the Negev
353 PUBLICATIONS **16,273** CITATIONS

SEE PROFILE

Bracha Shapira
Ben-Gurion University of the Negev
192 PUBLICATIONS **7,286** CITATIONS

SEE PROFILE

Chapter 1
Introduction to Recommender Systems Handbook

Francesco Ricci, Lior Rokach and Bracha Shapira

Abstract Recommender Systems (RSs) are software tools and techniques providing suggestions for items to be of use to a user. In this introductory chapter we briefly discuss basic RS ideas and concepts. Our main goal is to delineate, in a coherent and structured way, the chapters included in this handbook and to help the reader navigate the extremely rich and detailed content that the handbook offers.

1.1 Introduction

Recommender Systems (RSs) are software tools and techniques providing suggestions for items to be of use to a user [60, 85, 25]. The suggestions relate to various decision-making processes, such as what items to buy, what music to listen to, or what online news to read.

"Item" is the general term used to denote what the system recommends to users. A RS normally focuses on a specific type of item (e.g., CDs, or news) and accordingly its design, its graphical user interface, and the core recommendation technique used to generate the recommendations are all customized to provide useful and effective suggestions for that specific type of item.

RSs are primarily directed towards individuals who lack sufficient personal experience or competence to evaluate the potentially overwhelming number of alter-

Francesco Ricci
Faculty of Computer Science, Free University of Bozen-Bolzano, Italy e-mail: fricci@unibz.it

Lior Rokach
Department of Information Systems Engineering, Ben-Gurion University of the Negev, Israel e-mail: liorrk@bgu.ac.il

Bracha Shapira
Department of Information Systems Engineering, Ben-Gurion University of the Negev, Israel e-mail: bshapira@bgu.ac.il

F. Ricci et al. (eds.), *Recommender Systems Handbook*,
DOI 10.1007/978-0-387-85820-3_1,

native items that a Web site, for example, may offer [85]. A case in point is a book recommender system that assists users to select a book to read. In the popular Web site, Amazon.com, the site employs a RS to personalize the online store for each customer [47]. Since recommendations are usually personalized, different users or user groups receive diverse suggestions. In addition there are also non-personalized recommendations. These are much simpler to generate and are normally featured in magazines or newspapers. Typical examples include the top ten selections of books, CDs etc. While they may be useful and effective in certain situations, these types of non-personalized recommendations are not typically addressed by RS research.

In their simplest form, personalized recommendations are offered as ranked lists of items. In performing this ranking, RSs try to predict what the most suitable products or services are, based on the user's preferences and constraints. In order to complete such a computational task, RSs collect from users their preferences, which are either explicitly expressed, e.g., as ratings for products, or are inferred by interpreting user actions. For instance, a RS may consider the navigation to a particular product page as an implicit sign of preference for the items shown on that page.

RSs development initiated from a rather simple observation: individuals often rely on recommendations provided by others in making routine, daily decisions [60, 70]. For example it is common to rely on what one's peers recommend when selecting a book to read; employers count on recommendation letters in their recruiting decisions; and when selecting a movie to watch, individuals tend to read and rely on the movie reviews that a film critic has written and which appear in the newspaper they read.

In seeking to mimic this behavior, the first RSs applied algorithms to leverage recommendations produced by a community of users to deliver recommendations to an active user, i.e., a user looking for suggestions. The recommendations were for items that similar users (those with similar tastes) had liked. This approach is termed collaborative-filtering and its rationale is that if the active user agreed in the past with some users, then the other recommendations coming from these similar users should be relevant as well and of interest to the active user.

As e-commerce Web sites began to develop, a pressing need emerged for providing recommendations derived from filtering the whole range of available alternatives. Users were finding it very difficult to arrive at the most appropriate choices from the immense variety of items (products and services) that these Web sites were offering.

The explosive growth and variety of information available on the Web and the rapid introduction of new e-business services (buying products, product comparison, auction, etc.) frequently overwhelmed users, leading them to make poor decisions. The availability of choices, instead of producing a benefit, started to decrease users' well-being. It was understood that while choice is good, more choice is not always better. Indeed, choice, with its implications of freedom, autonomy, and self-determination can become excessive, creating a sense that freedom may come to be regarded as a kind of misery-inducing tyranny [96].

RSs have proved in recent years to be a valuable means for coping with the information overload problem. Ultimately a RS addresses this phenomenon by pointing

a user towards new, not-yet-experienced items that may be relevant to the users current task. Upon a user's request, which can be articulated, depending on the recommendation approach, by the user's context and need, RSs generate recommendations using various types of knowledge and data about users, the available items, and previous transactions stored in customized databases. The user can then browse the recommendations. She may accept them or not and may provide, immediately or at a next stage, an implicit or explicit feedback. All these user actions and feedbacks can be stored in the recommender database and may be used for generating new recommendations in the next user-system interactions.

As noted above, the study of recommender systems is relatively new compared to research into other classical information system tools and techniques (e.g., databases or search engines). Recommender systems emerged as an independent research area in the mid-1990s [35, 60, 70, 7]. In recent years, the interest in recommender systems has dramatically increased, as the following facts indicate:

1. Recommender systems play an important role in such highly rated Internet sites as Amazon.com, YouTube, Netflix, Yahoo, Tripadvisor, Last.fm, and IMDb. Moreover many media companies are now developing and deploying RSs as part of the services they provide to their subscribers. For example Netflix, the online movie rental service, awarded a million dollar prize to the team that first succeeded in improving substantially the performance of its recommender system [54].
2. There are dedicated conferences and workshops related to the field. We refer specifically to ACM Recommender Systems (RecSys), established in 2007 and now the premier annual event in recommender technology research and applications. In addition, sessions dedicated to RSs are frequently included in the more traditional conferences in the area of data bases, information systems and adaptive systems. Among these conferences are worth mentioning ACM SIGIR Special Interest Group on Information Retrieval (SIGIR), User Modeling, Adaptation and Personalization (UMAP), and ACM's Special Interest Group on Management Of Data (SIGMOD).
3. At institutions of higher education around the world, undergraduate and graduate courses are now dedicated entirely to RSs; tutorials on RSs are very popular at computer science conferences; and recently a book introducing RSs techniques was published [48].
4. There have been several special issues in academic journals covering research and developments in the RS field. Among the journals that have dedicated issues to RS are: AI Communications (2008); IEEE Intelligent Systems (2007); International Journal of Electronic Commerce (2006); International Journal of Computer Science and Applications (2006); ACM Transactions on Computer-Human Interaction (2005); and ACM Transactions on Information Systems (2004).

In this introductory chapter we briefly discuss basic RS ideas and concepts. Our main goal is not much to present a self-contained comprehensive introduction or survey on RSs but rather to delineate, in a coherent and structured way, the chapters

included in this handbook and to help the reader navigate the extremely rich and detailed content that the handbook offers.

The handbook is divided into five sections: techniques; applications and evaluation of RSs; interacting with RSs; RSs and communities; and advanced algorithms.

The first section presents the techniques most popularly used today for building RSs, such as collaborative filtering; content-based, data mining methods; and context-aware methods.

The second section surveys techniques and approaches that have been utilized to evaluate the quality of the recommendations. It also deals with the practical aspects of designing recommender systems; describes design and implementation considerations; and sets guidelines for selecting the more suitable algorithms. The section also considers aspects that may affect RS design (domain, device, users, etc.). Finally, it discusses methods, challenges and measures to be applied in evaluating the developed systems.

The third section includes papers dealing with a number of issues related to how recommendations are presented, browsed, explained and visualized. The techniques that make the recommendation process more structured and conversational are discussed here.

The fourth section is fully dedicated to a rather new topic, exploiting user-generated content (UGC) of various types (tags, search queries, trust evaluations, etc.) to generate innovative types of recommendations and more credible ones. Despite its relative newness, this topic is essentially rooted in the core idea of a collaborative recommender,

The last selection presents papers on various advanced topics, such as: the exploitation of active learning principles to guide the acquisition of new knowledge; suitable techniques for protecting a recommender system against attacks of malicious users; and RSs that aggregate multiple types of user feedbacks and preferences to build more reliable recommendations.

1.2 Recommender Systems Function

In the previous section we defined RSs as software tools and techniques providing users with suggestions for items a user may wish to utilize. Now we want to refine this definition illustrating a range of possible roles that a RS can play. First of all, we must distinguish between the role played by the RS on behalf of the service provider from that of the user of the RS. For instance, a travel recommender system is typically introduced by a travel intermediary (e.g., Expedia.com) or a destination management organization (e.g., Visitfinland.com) to increase its turnover (Expedia), i.e., sell more hotel rooms, or to increase the number of tourists to the destination [86]. Whereas, the user's primary motivations for accessing the two systems is to find a suitable hotel and interesting events/attractions when visiting a destination.

In fact, there are various reasons as to why service providers may want to exploit this technology:

- *Increase the number of items sold.* This is probably the most important function for a commercial RS, i.e., to be able to sell an additional set of items compared to those usually sold without any kind of recommendation. This goal is achieved because the recommended items are likely to suit the user's needs and wants. Presumably the user will recognize this after having tried several recommendations[1]. Non-commercial applications have similar goals, even if there is no cost for the user that is associated with selecting an item. For instance, a content network aims at increasing the number of news items read on its site.
  In general, we can say that from the service provider's point of view, the primary goal for introducing a RS is to increase the conversion rate, i.e., the number of users that accept the recommendation and consume an item, compared to the number of simple visitors that just browse through the information.
- *Sell more diverse items.* Another major function of a RS is to enable the user to select items that might be hard to find without a precise recommendation. For instance, in a movie RS such as Netflix, the service provider is interested in renting all the DVDs in the catalogue, not just the most popular ones. This could be difficult without a RS since the service provider cannot afford the risk of advertising movies that are not likely to suit a particular user's taste. Therefore, a RS suggests or advertises unpopular movies to the right users
- *Increase the user satisfaction.* A well designed RS can also improve the experience of the user with the site or the application. The user will find the recommendations interesting, relevant and, with a properly designed human-computer interaction, she will also enjoy using the system. The combination of effective, i.e., accurate, recommendations and a usable interface will increase the user's subjective evaluation of the system. This in turn will increase system usage and the likelihood that the recommendations will be accepted.
- *Increase user fidelity.* A user should be loyal to a Web site which, when visited, recognizes the old customer and treats him as a valuable visitor. This is a normal feature of a RS since many RSs compute recommendations, leveraging the information acquired from the user in previous interactions, e.g., her ratings of items. Consequently, the longer the user interacts with the site, the more refined her user model becomes, i.e., the system representation of the user's preferences, and the more the recommender output can be effectively customized to match the user's preferences.
- *Better understand what the user wants.* Another important function of a RS, which can be leveraged to many other applications, is the description of the user's preferences, either collected explicitly or predicted by the system. The service provider may then decide to re-use this knowledge for a number of other goals such as improving the management of the item's stock or production. For instance, in the travel domain, destination management organizations can decide to advertise a specific region to new customer sectors or advertise a particular

[1] This issue, convincing the user to accept a recommendation, is discussed again when we explain the difference between predicting the user interest in an item and the likelihood that the user will select the recommended item.

type of promotional message derived by analyzing the data collected by the RS (transactions of the users).

We mentioned above some important motivations as to why e-service providers introduce RSs. But users also may want a RS, if it will effectively support their tasks or goals. Consequently a RS must balance the needs of these two players and offer a service that is valuable to both.

Herlocker et al. [25], in a paper that has become a classical reference in this field, define eleven popular tasks that a RS can assist in implementing. Some may be considered as the main or core tasks that are normally associated with a RS, i.e., to offer suggestions for items that may be useful to a user. Others might be considered as more "opportunistic" ways to exploit a RS. As a matter of fact, this task differentiation is very similar to what happens with a search engine, Its primary function is to locate documents that are relevant to the user's information need, but it can also be used to check the importance of a Web page (looking at the position of the page in the result list of a query) or to discover the various usages of a word in a collection of documents.

- *Find Some Good Items:* Recommend to a user some items as a ranked list along with predictions of how much the user would like them (e.g., on a one- to five-star scale). This is the main recommendation task that many commercial systems address (see, for instance, Chapter 9). Some systems do not show the predicted rating.
- *Find all good items:* Recommend all the items that can satisfy some user needs. In such cases it is insufficient to just find some good items. This is especially true when the number of items is relatively small or when the RS is mission-critical, such as in medical or financial applications. In these situations, in addition to the benefit derived from carefully examining all the possibilities, the user may also benefit from the RS ranking of these items or from additional explanations that the RS generates.
- *Annotation in context:* Given an existing context, e.g., a list of items, emphasize some of them depending on the user's long-term preferences. For example, a TV recommender system might annotate which TV shows displayed in the electronic program guide (EPG) are worth watching (Chapter 18 provides interesting examples of this task).
- *Recommend a sequence:* Instead of focusing on the generation of a single recommendation, the idea is to recommend a sequence of items that is pleasing as a whole. Typical examples include recommending a TV series; a book on RSs after having recommended a book on data mining; or a compilation of musical tracks [99], [39].
- *Recommend a bundle:* Suggest a group of items that fits well together. For instance a travel plan may be composed of various attractions, destinations, and accommodation services that are located in a delimited area. From the point of view of the user these various alternatives can be considered and selected as a single travel destination [87].

- *Just browsing:* In this task, the user browses the catalog without any imminent intention of purchasing an item. The task of the recommender is to help the user to browse the items that are more likely to fall within the scope of the user's interests for that specific browsing session. This is a task that has been also supported by adaptive hypermedia techniques [23].
- *Find credible recommender:* Some users do not trust recommender systems thus they play with them to see how good they are in making recommendations. Hence, some system may also offer specific functions to let the users test its behavior in addition to those just required for obtaining recommendations.
- *Improve the profile:* This relates to the capability of the user to provide (input) information to the recommender system about what he likes and dislikes. This is a fundamental task that is strictly necessary to provide personalized recommendations. If the system has no specific knowledge about the active user then it can only provide him with the same recommendations that would be delivered to an "average" user.
- *Express self:* Some users may not care about the recommendations at all. Rather, what it is important to them is that they be allowed to contribute with their ratings and express their opinions and beliefs. The user satisfaction for that activity can still act as a leverage for holding the user tightly to the application (as we mentioned above in discussing the service provider's motivations).
- *Help others:* Some users are happy to contribute with information, e.g., their evaluation of items (ratings), because they believe that the community benefits from their contribution. This could be a major motivation for entering information into a recommender system that is not used routinely. For instance, with a car RS, a user, who has already bought her new car is aware that the rating entered in the system is more likely to be useful for other users rather than for the next time she will buy a car.
- *Influence others:* In Web-based RSs, there are users whose main goal is to explicitly influence other users into purchasing particular products. As a matter of fact, there are also some malicious users that may use the system just to promote or penalize certain items (see Chapter 25).

As these various points indicate, the role of a RS within an information system can be quite diverse. This diversity calls for the exploitation of a range of different knowledge sources and techniques and in the next two sections we discuss the data a RS manages and the core technique used to identify the right recommendations.

1.3 Data and Knowledge Sources

RSs are information processing systems that actively gather various kinds of data in order to build their recommendations. Data is primarily about the items to suggest and the users who will receive these recommendations. But, since the data and knowledge sources available for recommender systems can be very diverse, ultimately, whether they can be exploited or not depends on the recommendation

technique (see also section 1.4). This will become clearer in the various chapters included in this handbook (see in particular Chapter 11).

In general, there are recommendation techniques that are knowledge poor, i.e., they use very simple and basic data, such as user ratings/evaluations for items (Chapters 5, 4). Other techniques are much more knowledge dependent, e.g., using ontological descriptions of the users or the items (Chapter 3), or constraints (Chapter 6), or social relations and activities of the users (Chapter 19). In any case, as a general classification, data used by RSs refers to three kinds of objects: items, users, and transactions, i.e., relations between users and items.

Items. Items are the objects that are recommended. Items may be characterized by their complexity and their value or utility. The value of an item may be positive if the item is useful for the user, or negative if the item is not appropriate and the user made a wrong decision when selecting it. We note that when a user is acquiring an item she will always incur in a cost, which includes the cognitive cost of searching for the item and the real monetary cost eventually paid for the item.

For instance, the designer of a news RS must take into account the complexity of a news item, i.e., its structure, the textual representation, and the time-dependent importance of any news item. But, at the same time, the RS designer must understand that even if the user is not paying for reading news, there is always a cognitive cost associated to searching and reading news items. If a selected item is relevant for the user this cost is dominated by the benefit of having acquired a useful information, whereas if the item is not relevant the net value of that item for the user, and its recommendation, is negative. In other domains, e.g., cars, or financial investments, the true monetary cost of the items becomes an important element to consider when selecting the most appropriate recommendation approach.

Items with low complexity and value are: news, Web pages, books, CDs, movies. Items with larger complexity and value are: digital cameras, mobile phones, PCs, etc. The most complex items that have been considered are insurance policies, financial investments, travels, jobs [72].

RSs, according to their core technology, can use a range of properties and features of the items. For example in a movie recommender system, the genre (such as comedy, thriller, etc.), as well as the director, and actors can be used to describe a movie and to learn how the utility of an item depends on its features. Items can be represented using various information and representation approaches, e.g., in a minimalist way as a single id code, or in a richer form, as a set of attributes, but even as a concept in an ontological representation of the domain (Chapter 3).

Users. Users of a RS, as mentioned above, may have very diverse goals and characteristics. In order to personalize the recommendations and the human-computer interaction, RSs exploit a range of information about the users. This information can be structured in various ways and again the selection of what information to model depends on the recommendation technique.

For instance, in collaborative filtering, users are modeled as a simple list containing the ratings provided by the user for some items. In a demographic RS, socio-demographic attributes such as age, gender, profession, and education, are used. User data is said to constitute the user model [21, 32]. The user model profiles the

user, i.e., encodes her preferences and needs. Various user modeling approaches have been used and, in a certain sense, a RS can be viewed as a tool that generates recommendations by building and exploiting user models [19, 20]. Since no personalization is possible without a convenient user model, unless the recommendation is non-personalized, as in the top-10 selection, the user model will always play a central role. For instance, considering again a collaborative filtering approach, the user is either profiled directly by its ratings to items or, using these ratings, the system derives a vector of factor values, where users differ in how each factor weights in their model (Chapters 5 and 4).

Users can also be described by their behavior pattern data, for example, site browsing patterns (in a Web-based recommender system) [107], or travel search patterns (in a travel recommender system) [60]. Moreover, user data may include relations between users such as the trust level of these relations between users (Chapter 20). A RS might utilize this information to recommend items to users that were preferred by similar or trusted users.

Transactions. We generically refer to a transaction as a recorded interaction between a user and the RS. Transactions are log-like data that store important information generated during the human-computer interaction and which are useful for the recommendation generation algorithm that the system is using. For instance, a transaction log may contain a reference to the item selected by the user and a description of the context (e.g., the user goal/query) for that particular recommendation. If available, that transaction may also include an explicit feedback the user has provided, such as the rating for the selected item.

In fact, ratings are the most popular form of transaction data that a RS collects. These ratings may be collected explicitly or implicitly. In the explicit collection of ratings, the user is asked to provide her opinion about an item on a rating scale. According to [93], ratings can take on a variety of forms:

- Numerical ratings such as the 1-5 stars provided in the book recommender associated with Amazon.com.
- Ordinal ratings, such as "strongly agree, agree, neutral, disagree, strongly disagree" where the user is asked to select the term that best indicates her opinion regarding an item (usually via questionnaire).
- Binary ratings that model choices in which the user is simply asked to decide if a certain item is good or bad.
- Unary ratings can indicate that a user has observed or purchased an item, or otherwise rated the item positively. In such cases, the absence of a rating indicates that we have no information relating the user to the item (perhaps she purchased the item somewhere else).

Another form of user evaluation consists of tags associated by the user with the items the system presents. For instance, in Movielens RS (http://movielens.umn.edu) tags represent how MovieLens users feel about a movie, e.g.: "too long", or "acting". Chapter 19 focuses on these types of transactions.

In transactions collecting implicit ratings, the system aims to infer the users opinion based on the user's actions. For example, if a user enters the keyword "Yoga" at

Amazon.com she will be provided with a long list of books. In return, the user may click on a certain book on the list in order to receive additional information. At this point, the system may infer that the user is somewhat interested in that book.

In conversational systems, i.e., systems that support an interactive process, the transaction model is more refined. In these systems user requests alternate with system actions (see Chapter 13). That is, the user may request a recommendation and the system may produce a suggestion list. But it can also request additional user preferences to provide the user with better results. Here, in the transaction model, the system collects the various requests-responses, and may eventually learn to modify its interaction strategy by observing the outcome of the recommendation process [60].

1.4 Recommendation Techniques

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 do 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 this comparison. The prediction step may not be explicit in the recommendation algorithm but we can still apply this unifying model to describe the general role of a RS. Here our goal is to provide the reader with a unifying perspective rather than an account of all the different recommendation approaches that will be illustrated in this handbook.

To illustrate the prediction step of a RS, consider, for instance, a simple, non-personalized, recommendation algorithm that recommends just the most popular songs. The rationale for using this approach is that in absence of more precise information about the user's preferences, a popular song, i.e., something that is liked (high utility) by many users, will also be probably liked by a generic user, at least more than another randomly selected song. Hence the utility of these popular songs is predicted to be reasonably high for this generic user.

This view of the core recommendation computation as the prediction of the utility of an item for a user has been suggested in [3]. They model this degree of utility of the user u for the item i as a (real valued) function $R(u,i)$, as is normally done in collaborative filtering by considering the ratings of users for items. Then the fundamental task of a collaborative filtering RS is to predict the value of R over pairs of users and items, i.e., to compute $\hat{R}(u,i)$, where we denote with $\hat{R}$ the estimation, computed by the RS, of the true function R. Consequently, having computed this prediction for the active user u on a set of items, i.e., $\hat{R}(u,i_1),\ldots,\hat{R}(u,i_N)$ the system will recommend the items $i_{j_1},\ldots,i_{j_K}$ ($K \leq N$) with the largest predicted utility. K is typically a small number, i.e., much smaller than the cardinality of the item data set or the items on which a user utility prediction can be computed, i.e., RSs "filter" the items that are recommended to users.

As mentioned above, some recommender systems do not fully estimate the utility before making a recommendation but they may apply some heuristics to hypothe-

size that an item is of use to a user. This is typical, for instance, in knowledge-based systems. These utility predictions are computed with specific algorithms (see below) and use various kind of knowledge about users, items, and the utility function itself (see section 1.3) [25]. For instance, the system may assume that the utility function is Boolean and therefore it will just determine whether an item is or is not useful for the user. Consequently, assuming that there is some available knowledge (possibly none) about the user who is requesting the recommendation, knowledge about items, and other users who received recommendations, the system will leverage this knowledge with an appropriate algorithm to generate various utility predictions and hence recommendations [25].

It is also important to note that sometimes the user utility for an item is observed to depend on other variables, which we generically call "contextual" [1]. For instance, the utility of an item for a user can be influenced by the domain knowledge of the user (e.g., expert vs. beginning users of a digital camera), or can depend on the time when the recommendation is requested. Or the user may be more interested in items (e.g., a restaurant) closer to his current location. Consequently, the recommendations must be adapted to these specific additional details and as a result it becomes harder and harder to correctly estimate what the right recommendations are.

This handbook presents several different types of recommender systems that vary in terms of the addressed domain, the knowledge used, but especially in regard to the recommendation algorithm, i.e., how the prediction of the utility of a recommendation, as mentioned at the beginning of this section, is made. Other differences relate to how the recommendations are finally assembled and presented to the user in response to user requests. These aspects are also discussed later in this introduction.

To provide a first overview of the different types of RSs, we want to quote a taxonomy provided by [25] that has become a classical way of distinguishing between recommender systems and referring to them. [25] distinguishes between six different classes of recommendation approaches:

Content-based: The system learns to recommend items that are similar to the ones that the user liked in the past. The similarity of items is calculated based on the features associated with the compared items. For example, if a user has positively rated a movie that belongs to the comedy genre, then the system can learn to recommend other movies from this genre. Chapter 3 provides an overview of content-based recommender systems, imposing some order among the extensive and diverse aspects involved in their design and implementation. It presents the basic concepts and terminology of content-based RSs, their high level architecture, and their main advantages and drawbacks. The chapter then surveys state-of-the-art systems that have been adopted in several application domains. The survey encompasses a thorough description of both classical and advanced techniques for representing items and user profiles. Finally, it discusses trends and future research which might lead towards the next generation of recommender systems.

Collaborative filtering: The simplest and original implementation of this approach [93] recommends to the active user the items that other users with similar tastes liked in the past. The similarity in taste of two users is calculated based on

the similarity in the rating history of the users. This is the reason why [94] refers to collaborative filtering as "people-to-people correlation." Collaborative filtering is considered to be the most popular and widely implemented technique in RS.

Chapter 4 presents a comprehensive survey of neighborhood-based methods for collaborative filtering. Neighborhood methods focus on relationships between items or, alternatively, between users. An item-item approach models the preference of a user to an item based on ratings of similar items by the same user. Nearest-neighbors methods enjoy considerable popularity due to their simplicity, efficiency, and their ability to produce accurate and personalized recommendations. The authors will address the essential decisions that are required when implementing a neighborhood-based recommender system and provide practical information on how to make such decisions.

Finally, the chapter deals with problems of data sparsity and limited coverage, often observed in large commercial recommender systems. A few solutions to overcome these problems are presented.

Chapter 5 presents several recent extensions available for building CF recommenders. Specifically, the authors discuss latent factor models, such as matrix factorization (e.g., Singular Value Decomposition, SVD). These methods transform both items and users to the same latent factor space. The latent space is then used to explain ratings by characterizing both products and users in term of factors automatically inferred from user feedback. The authors elucidate how SVD can handle additional features of the data, including implicit feedback and temporal information. They also describe techniques to address shortcomings of neighborhood techniques by suggesting more rigorous formulations using global optimization techniques. Utilizing such techniques makes it possible to lift the limit on neighborhood size and to address implicit feedback and temporal dynamics. The resulting accuracy is close to that of matrix factorization models, while offering a number of practical advantages.

Demographic: This type of system recommends items based on the demographic profile of the user. The assumption is 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 [59].

Knowledge-based: 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. Notable knowledge-based recommender systems are case-based [22, 87]. In these systems a similarity function estimates how much the user needs (problem description) match the recommendations (solutions of the problem). Here the similarity score can be directly interpreted as the utility of the recommendation for the user.

Constraint-based systems are another type of knowledge-based RSs (Chapter 6). In terms of used knowledge, both systems are similar: user requirements are col-

lected; repairs for inconsistent requirements are automatically proposed in situations where no solutions could be found; and recommendation results are explained. The major difference lies in the way solutions are calculated. 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.

Knowledge-based systems tend to work better than others at the beginning of their deployment but if they are not equipped with learning components they may be surpassed by other shallow methods that can exploit the logs of the human/computer interaction (as in CF).

Community-based: 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". [8, 14]. Evidence suggests that people tend to rely more on recommendations from their friends than on recommendations from similar but anonymous individuals [103]. This observation, combined with the growing popularity of open social networks, is generating a rising interest in community-based systems or, as or as they usually referred to, social recommender systems [34]. 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. In fact these RSs are following the rise of social-networks and enable a simple and comprehensive acquisition of data related to the social relations of the users.

The research in this area is still in its early phase and results about the systems performance are mixed. For example, [34, 64] report that overall, social-network-based recommendations are no more accurate than those derived from traditional CF approaches, except in special cases, such as when user ratings of a specific item are highly varied (i.e. controversial items) or for cold-start situations, i.e., where the users did not provide enough ratings to compute similarity to other users. Others have showed that in some cases social-network data yields better recommendations than profile similarity data [37] and that adding social network data to traditional CF improves recommendation results [36]. The chapter 20 provides a survey of the findings in this field and analyzes current results.

Hybrid recommender systems: These RSs are based on the combination of the above mentioned techniques. 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. Given two (or more) basic RSs techniques, several ways have been proposed for combining them to create a new hybrid system (see [25] for the precise descriptions).

As we have already mentioned, the context of the user when she is seeking a recommendation can be used to better personalize the output of the system. For example, in a temporal context, vacation recommendations in winter should be very different from those provided in summer. Or a restaurant recommendation for a

Saturday evening with your friends should be different from that suggested for a workday lunch with co-workers.

Chapter 7 presents the general notion of context and how it can be modeled in RSs. Discussing the possibilities of combining several context-aware recommendation techniques into a single unified approach, the authors also provide a case study of one such combined approach.

Three different algorithmic paradigms for incorporating contextual information into the recommendation process are discussed: reduction-based (pre-filtering), contextual post filtering, and context modeling. In reduction-based (pre-filtering) methods, only the information that matches the current usage context, e.g., the ratings for items evaluated in the same context, are used to compute the recommendations. In contextual post filtering, the recommendation algorithm ignores the context information. The output of the algorithm is filtered/adjusted to include only the recommendations that are relevant in the target context. In the contextual modeling, the more sophisticated of the three approaches, context data is explicitly used in the prediction model.

1.5 Application and Evaluation

Recommender system research is being conducted with a strong emphasis on practice and commercial applications, since, aside from its theoretical contribution, is generally aimed at practically improving commercial RSs. Thus, RS research involves practical aspects that apply to the implementation of these systems. These aspects are relevant to different stages in the life cycle of a RS, namely, the design of the system, its implementation and its maintenance and enhancement during system operation.

The aspects that apply to the design stage include factors that might affect the choice of the algorithm. The first factor to consider, the application's domain, has a major effect on the algorithmic approach that should be taken. [72] provide a taxonomy of RSs and classify existing RS applications to specific application domains. Based on these specific application domains, we define more general classes of domains for the most common recommender systems applications:

- Entertainment - recommendations for movies, music, and IPTV.
- Content - personalized newspapers, recommendation for documents, recommendations of Web pages, e-learning applications, and e-mail filters.
- E-commerce - recommendations for consumers of products to buy such as books, cameras, PCs etc.
- Services - recommendations of travel services, recommendation of experts for consultation, recommendation of houses to rent, or matchmaking services.

As recommender systems become more popular, interest is aroused in the potential advantages in new applications, such as recommending friends or tweets to

follow as in www.tweeter.com. Hence, the above list cannot cover all the application domains that are now being addressed by RS techniques; it gives only an initial description of the various types of application domains.

The developer of a RS for a certain application domain should understand the specific facets of the domain, its requirements, application challenges and limitations. Only after analyzing these factors one could be able to select the optimal recommender algorithm and to design an effective human-computer interaction.

Chapter 11 of this handbook provides guidelines for matching the application domain to the recommendation technique. Burke and Ramezani in their chapter provide a new classification of recommender systems. Unlike former classifications of RSs (such as [25, 94, 3, 7]), Burke and Ramezani take an AI-centric approach, and focus on the knowledge sources required for different recommendation approaches, and the constraints related to them as a primer guideline to choosing the algorithm. The chapter discusses the applicability of various recommendation techniques for different types of problems and suggests decision-making guidelines in selecting these techniques.

The chapter explicitly aims at system implementers as "recommenders" for the right recommendation approach. The authors describe the knowledge sources that are available to a recommender systems in different domains and identify what knowledge sources are required for each recommendation technique. This implies that the design of a recommender system should first emphasize the analysis of the available sources of knowledge, and then decide about the algorithm accordingly.

Another example of the need to adjust the recommender approach to the domain is described in Chapter 12, which deals with recommender systems for technology-enhanced learning (TEL). TEL, which generally covers technologies that support all forms of teaching and learning activities, aims at designing, developing and testing new methods and technologies to enhance learning practices of both individuals and organizations. TEL may benefit greatly from integrating recommender systems technology to personalize the learning process and adjust it to the user's former knowledge, abilities and preferences. The chapter presents the particular requirements of RSs for TEL; the user tasks that are supported in TEL settings; and how these tasks compare to typical user tasks in other RSs. For example, one particular user task for TEL –"find novel resources"– attempts to recommend only new or novel items. Or, to cite another example, – "find new pathways" – is concerned with recommending alternative pathways through the learning resources. The chapter presents an analysis of the filtering approaches that could be useful for TEL along with a survey of existing TEL systems illustrating the recommendation techniques that have been deployed in these systems.

Chapter 10 discusses practical aspects of RS development and aims at providing practical guidelines to the design, implementation and evaluation of personalized systems. Besides the prediction algorithm, many other factors need to be considered when designing a RS. Chapter 10 lists some of these elements: the type of target users and their context; the devices that they would use; the role of the recommendation within the application; the goal of the recommendation; and, as mentioned previously, the data that is available.

The authors propose to build a model of the environment based on three dimensions: system users; the characteristics of the data; and the overall application. The recommender system design will be based on this model. The authors illustrate their guidelines and the model on a news recommendation system that they have developed.

Another important issue related to the practical side of RS deployment is the necessity of evaluating them. Evaluation is required at different stages of the systems life cycle for various purposes [25, 1]. At design time, evaluation is required to verify the selection of the appropriate recommender approach. In the design phase, evaluation should be implemented off-line and the recommendation algorithms are compared with user interactions. The off-line evaluation consists of running several algorithms on the same datasets of user interactions (e.g., ratings) and comparing their performance. This type of evaluation is usually conducted on existing public benchmark data if appropriate data is available, or, otherwise, on collected data. The design of the off-line experiments should follow known experiment design practices [11] in order to ensure reliable results.

Evaluation is also required after the system has been launched. The algorithms might be very accurate in solving the core recommendation problem, i.e., predicting user ratings, but for some other reason the system may not be accepted by users, e.g., because the performance of the system is not as expected. At this stage it is usually beneficial to perform on-line evaluation with real users of the system and analyze system logs in order to enhance system performance. In addition, most of the algorithms include parameters, such as weights thresholds, the number of neighbors, etc., requiring constant adjustment and calibration.

Another type of evaluation is a focused user study that can be conducted when the on-line evaluation is not feasible or too risky. In this type of evaluation, a controlled experiment is planned where a small group of users are asked to perform different tasks with various versions of the system. It is then possible to analyze the users performance and to distribute questionnaires so that users may report on their experience. In such experiments it is possible to collect both quantitative and qualitative information about the systems.

Evaluation is also discussed in Chapter 12 in the context of TEL systems. The authors provide a detailed analysis of the evaluation methods and tools that can be employed for evaluating TEL recommendation techniques against a set of criteria that are proposed for each of the selected components (e.g., user model, domain model, recommendation strategy and algorithm).

Chapter 8 details three types of experiments that can be conducted in order to evaluate recommender systems. It presents their advantages and disadvantages, and defines guidelines for choosing the methods for evaluation them. Unlike existing discussions of evaluation in the literature that usually speaks about the accuracy of an algorithms prediction [25] and related measures, this chapter is unique in its approach to the evaluation discussion since it focuses on property-directed evaluation. It provides a large set of properties (other than accuracy) that are relevant to the systems success. For each of the properties, the appropriate type of experiment and

relevant measures are suggested. Among the list of properties are: coverage, cold start, confidence, trust, novelty, risk, and serendipity.

When discussing the practical aspects of RSs, it may be beneficial to analyze real system implementations. The idea is to test theoretically intuitive assumptions in order to determine if they work in practice. The major problem that one must face in this case comes from the fact that the owners of commercial RSs are generally unwilling to reveal their practices and there are only relatively few opportunities for such cooperation.

Chapter 9 reports on such an opportunity and describes the operation of a real RS, illustrating the practical aspects that apply to the implementation stage of the RS development and its evaluation. This description focuses on the integration of a RS into the production environment of Fastweb, one of the largest European IP Television (IPTV) providers. The chapter describes the requirements and considerations, including scaling and accuracy, that led to the choice of the recommender algorithms. It also describes the off-line and on-line evaluations that took place and illustrates how the system is adjusted accordingly.

1.6 Recommender Systems and Human Computer Interaction

As we have illustrated in previous sections, researchers have chiefly been concerned with designing a range of technical solutions, leveraging various sources of knowledge to achieve better predictions about what is liked and how much by the target user. The underlying assumption behind this research activity is that just presenting these correct recommendations, i.e., the best options, should be enough. In other words, the recommendations should speak for themselves, and the user should definitely accept the recommendations if they are correct. This is clearly an overly simplified account of the recommendation problem and it is not so easy to deliver recommendations.

In practice, users need recommendations because they do not have enough knowledge to make an autonomous decision. Consequently, it may not be easy for them to evaluate the proposed recommendation. Hence, various researchers have tried to understand the factors that lead to the acceptance of a recommendation by a given user [105, 30, 24, 97, 33].

[105] was among the first to point out that the effectiveness of a RS is dependent on factors that go beyond the quality of the prediction algorithm. In fact, the recommender must also convince users to try (or read, buy, listen, watch) the recommended items. This, of course, depends on the individual characteristics of the selected items and therefore on the recommendation algorithm. The process also depends, however, on the particular human/computer interaction supported by the system when the items are presented, compared, and explained. [105] found that from a users perspective, an effective recommender system must inspire trust in the system; it must have a system logic that is at least somewhat transparent; it should point users towards new, not-yet-experienced items; it should provide details

about recommended items, including pictures and community ratings; and finally, it should present ways to refine recommendations.

[105] and other similarly oriented researchers do not diminish the importance of the recommendation algorithm, but claim that its effectiveness should not be evaluated only in terms of the accuracy of the prediction, i.e., with standard and popular IR metrics, such as MAE (Mean Absolute Error), precision, or NDCG (Normalized Discounted Cumulative Gain) (see also Chapters 8 5, 9). Other dimensions should be measured that relate to the acceptance of the recommender system and its recommendations. These ideas have been remarkably well presented and discussed also by [33]. In that work the authors propose user-centric directions for evaluating recommender systems, including: the similarity of recommendation lists, recommendation serendipity, and the importance of user needs and expectations in a recommender.

Following the remarks made in [105], let us introduce some important points raised by HCI research that are further discussed in this handbook.

1.6.1 Trust, Explanations and Persuasiveness

First of all let us focus on trust. There are two different notions of trust that are discussed in this handbook: trust about the other users of the recommender and trust about a system's recommendations.

Chapter 20 focuses on the first notion and considers a class of recommender systems termed "social recommender systems". These systems attempt to generate more useful recommendations derived from information about user profiles and relationships between users that nowadays can be found virtually everywhere; e.g. in social networking sites such as Facebook, LinkedIn and MySpace. Since trust-based recommender systems mainly exploit the trust relationships found in these social networking sites to build new recommendation algorithms (e.g., [34]), they still operate on the core rating prediction problem but use trust relationships. The main claimed advantage is that users will be aware of the nature of the recommendations, i.e., how they have been identified, and will tend to place greater trust in these recommendations. In other words, the mutual trust of users can be exploited also for increasing the trust in the system.

Trust in system recommendations is discussed in Chapter 15. In this chapter the main scope is actually the role of explanations in RSs and trust emerges as one out of seven roles that can be played by explanations in RSs. These roles are: transparency - explaining how the system works; scrutability - allowing users to tell the system it is wrong [50]; trust - increasing user confidence in the system; effectiveness - helping users make good decisions; persuasiveness - convincing users to try or buy; efficiency - helping users make decisions faster; and satisfaction - increasing the ease of use or enjoyment.

This chapter also illustrates a range of approaches for building explanations. In the collaborative filtering style, i.e., the explanation is of the form "Other users similar to you liked this item". In content-based style explanations, the item's attributes

which most affected the item to be recommended to the user are illustrated. For example, in a movie recommendation, an explanation may be of the form "This movie was recommended because it stars Bruce Wills who you seem to like", or "Item X was recommended because of features A and B which are shared by items Y and Z, which you rated highly". In case-based style explanations, the system refers to items that are similar to the recommended one, for example, "The item was recommended because you said you own item X" or "These items are recommended based on your most recently viewed items". And finally, in knowledge-based style explanations, the system explains the differences between the recommended item and another item and how it serves the user's goal: "This room has an ocean view and is larger than the previous recommended room, which will make it more romantic as you requested".

Moving back to trust, we see that it serves as a means of obtaining the main goal of the recommender, i.e., to convince the user to accept the recommendations and try out one of the recommended items. This issue is ultimately related to the persuasiveness of the full RS, i.e., how the various elements of the RS, including what and how an item is recommended, actually operate during the human/computer interaction. This topic is discussed in the Chapter 14. Here the authors stress that a recommendation is seen as credible advice and is actually taken into account not only because of the user's perceptions of the recommendation but also due to the fundamental role of the system which is perceived as an advice-giver. Indeed, the literature about persuasion suggests that people are likely to accept recommendations from credible sources and we therefore conclude that the credibility of the RS is vital to increasing the likelihood of recommendation acceptance. Hence, the authors discuss how the credibility of RSs can be enhanced, providing a synopsis of credibility-related research.

1.6.2 Conversational Systems

Another severe limitation of many algorithmic approaches to RSs is due to the fact that these algorithms have been designed to collect all the input data only once. They then terminate their job by returning their recommendations. In many cases, this model is not effective since users may not be fully aware of their preferences until they have interacted to a certain extent with the system and roughly understand the range of alternatives. Or they may want to browse several alternative options before being convinced that some of the recommendations may suit them. There is also the possibility that the system may be initially wrong in its suggestions and the user may be willing to provide additional information that can fix these problems, and eventually obtain some better recommendations.

These aspects have been stressed and tackled by researchers engaged in following a line of research that is commonly known as "conversational RSs" [27, 110, 67, 60]. Conversational RSs use a diverse range of techniques for rating prediction or ranking. However, they all try to support an interactive process where both the user and

the system may query or provide information to the other partner. The critical issue here is how to design the dialogue, i.e., the conversational strategy and what actions the user and the system must perform in the various stages of the interaction. The supported dialogue must be effective, i.e., the user should terminate the conversation with a solution of the task (e.g., book a flight) and in a quick way (small number of conversational steps). In this handbook two chapters deal with this important topic.

Chapter 13 provides a comprehensive account of the research conducted in critiquing-based systems. Critiquing-based interfaces, or dialogue models, given an initial set of user preferences (e.g., preferred values for some item features) present to the user recommended items and support the user in formulating "critiques", such as "Show me more like item A, but cheaper".

Critiquing-based systems have attracted great interest in domains where there is a need for more sophisticated and interactive decision/recommendation support systems, such as in travel applications [88, 32, 100], or computer systems [82, 83]. Critiquing-based systems were initially designed as effective approaches to user preference elicitation problems, but have now become important for some additional motivations or applications, such as group recommendations, mixed-initiative recommendations, adaptive user interface, recommendation explanation, mobile recommenders.

Another approach related to conversational systems is preference-based [67]. Preference-based are similar to critiquing-based approaches since they present upfront the user with some recommendations, which are not considered to be the best but then let the user express preferences about some items. This additional information is used to refine the system representation of the user's preferences (user model) enabling the system to generate new and better recommendations.

Chapter 16 surveys these novel methods and systems focusing on three facets of the user-system interaction of such preference-based recommenders: initial preference elicitation; preference revision; and presentation of recommendation results. This chapter derives from the analysis of some systems as a collection of usability guidelines that can be applied in a wide and scalable way. Moreover, to select the guidelines, the authors do not focus on accuracy alone, but take into account that humans have limited cognitive resources and are not likely to achieve a high level of accuracy if the required effort is excessive. They identify and select methods that produce high recommendation accuracy involving an effort level that users are willing to make.

Previously mentioned approaches (critiquing- and preference-based) have been mostly applied to case-based reasoning systems [22], where the retrieval component is based on a similarity metric. In such cases, a query can always retrieve and rank all the products contained in the catalogue since a product is always, to some extent, similar to a probe product (query). If the query language supports other constraints (e.g. equality or range constraints) the query may fail to return a product satisfying the query [47, 71, 31]. In this case several techniques have been proposed for repairing the query by relaxing the minimum amount of constraints to make it satisfiable. This topic is also covered in a chapter dedicated to constraint-based RSs (Chapter 6).

1.6.3 Visualization

We have highlighted so far some HCI issues that have been tackled in RS research and which are discussed in this handbook. In summary, we have noted that how the system presents and visualizes the computed recommendation is obviously a critical factor for the acceptance of the recommendations and the RS.

Presentation and explanation techniques are not easily separable; a good presentation technique is also capable of explaining recommendations but also in motivating the user to make further requests, including requests for explanations. One common aspect in the technologies presented so far is the fact that recommendations are presented as a list of items. The length of this list can vary but the output of the core recommendation algorithm is normally a ranked list and this has been always exploited in the presentation.

In this handbook we include a chapter that illustrates a presentation approach that deviates from this paradigm. In Chapter 17 the authors observe that much information is lost in the ranked list visualization approach, since two products, both of which match the user query or the user model, can differ from each other based on a completely different set of product characteristics. If one is using a two dimensional, map-based visualization of the recommendations, it is possible to retain part of this information. In the map, one can position, in a restricted area of the map, recommendations that are similar to each other. This chapter presents two approaches for building this two-dimensional map of the recommendations and discusses its advantages and disadvantages.

1.7 Recommender Systems as a Multi-Disciplinary Field

Designing and developing RSs is a multi-disciplinary effort that has benefited from results obtained in various computer science fields especially machine learning and data mining, information retrieval, and human-computer interaction. This is also clear in the chapters included in this handbook and the discussion presented above. Here we want to briefly address these relationships.

Machine learning and data mining, subfields of artificial intelligence, allow a computer to learn to optimally perform a certain task using examples, data or past experiences [109]. For example, data mining can be used to learn from transaction data that customers who bought “Da Vinci Code” also bought “The Five People You Meet in Heaven”. Consequently, recommendations can be constructed using the information provided by these associations.

Many RSs are centered around the use of various machine learning and data mining algorithms to predict user evaluations for items, or for learning how to correctly rank items for a user. Chapter 2 of this handbook provides an overview of the main data mining techniques used in the context of RSs preprocessing methods, such as: sampling or dimensionality reduction; classification techniques, such as Bayesian

networks and support vector machines; clustering techniques such as k-means algorithm; and association rules.

Other chapters that illustrate and exemplify the relationships between RSs and data mining are: Chapter 12, discussing the usage of active learning for selective information acquisition; Chapter 5, devoted to advanced optimization techniques for building rating prediction models; Chapter 7, presenting various rating prediction methods that exploit contextually tagged transactional data; Chapter 24, presenting data mining techniques that exploit the evaluations of items over several criteria to better predict the overall user evaluations; Chapter 25, focusing on data mining solutions to detect attacks to a recommender system and for building more robust algorithmic solutions; Chapter 4, illustrating various instance based-learning options currently used in collaborative filtering systems; Chapter 19 illustrating the use of data mining solutions operating on a multiway array or a hypergraph with hyperedges, i.e., (user, resource, tag) triples; Chapter 20 presenting various data mining solutions on trust networks.

Information retrieval (IR) aims to assist users in storing and searching various forms of content, such as texts, images and videos [63]. With IR tools, users can quickly find information that is both relevant and comprehensive for their needs. While IR did not begin with the Web, the WWW played an important role in establishing new ideas mainly due to the development of Web search engines.

Both IR and RSs are faced with similar filtering and ranking problems. IR generally focuses on developing global retrieval techniques, often neglecting the individual needs and preferences of users. Still [25] argues that recommender systems are not clearly separated from information retrieval. The "individualized" and "interesting and useful" criteria that RSs try to achieve are the core differences between RSs and information retrieval or search engines.

Recently, modern Web search engine have also relied on recommendation techniques to address Web search challenges and to implement advanced search features. For example, search engines recommend similar queries to the current user query. Various engines also attempt to apply some form of personalization by generating results to a user query that are not only relevant to the query terms but are also tailored to the users context (e.g., her location), and her search history.

Chapter 18 discusses the research goals of IR and personalized Web search from the RS perspective. The authors illustrate how techniques that originated in recent RS research may be applied to address search engine challenges. The chapter focuses on two promising ideas for search engines improvement: personalization and collaboration. The chapter describes a number of different approaches to personalizing Web searches by exploiting user preferences and context information to affect search results. In addition, the chapter discusses recent work in the area of collaborative information retrieval, which attempts to take advantage of the potential for cooperation between friends, colleagues or users with similar needs in implementing a variety of information-seeking tasks. This new line of research, termed social search, benefits from the social medium property of the Web in providing search results that are affected by the experience and preferences of similar users. The authors foresee a "convergence of recommender systems and search systems" and

believe that integrating these sources in search engine algorithms would result in highly satisfied users receiving the right information at the right time.

Other chapters that are related to IR research and illustrate techniques that are studied in this area include: Chapter 19, addressing problems related to the retrieval of tag-based information content and Chapter 3, presenting an overview of content-based approaches that are strongly rooted in current search engine technologies.

Finally, RSs are ultimately targeted to provide useful information to users and for that reason HCI plays a fundamental role in the ultimate acceptance of the computed recommendations. In fact, several field studies have clearly indicated that from a user's perspective, HCI aspects related to the usability of the system have a tremendous effect on the willingness of users to actually explore a systems recommendations and provide input to the system in return for more effective recommendations. These topics were discussed previously in Section 1.6.

1.8 Emerging Topics and Challenges

1.8.1 Emerging Topics Discussed in the Handbook

It is clear from the previous pages that RS research is evolving in many and diverse directions and new topics are emerging or becoming more important subjects of investigation. The reader is also referred to the proceedings of the last editions of the ACM RecSys conferences and several other excellent review papers for additional material [7, 3]. In this handbook we cover some of these topics. Indeed, several have been already presented, such as: context-aware recommender (Chapter 7); new visualization techniques (Chapter 17); community-based personalized search (Chapter 18); trust-based RS (Chapter 20). Other important topics are covered in the last two sections of this handbook and we want now to briefly introduce these chapters.

Chapter 19 presents social tagging systems (STS) a new RS-related topic that is emerging due to the growth of Web 2.0 applications. STS like Flickr, Bibsonomy, or Delicious, allow the ordinary user to publish and edit content as well as generate and share tags (i.e., free keywords). STS users are experiencing information overload problems since STS are used by millions of users who enter into the system uncontrolled content and tags that pose retrieving difficulties for traditional IR systems. Thus, RSs are required to assist users in finding relevant Information and some commercial STS are starting to offer recommendations (e.g., Delicious).

The chapter discusses the new challenges that RSs for STS face, such as new recommender tasks. These include not only traditional recommendations regarding content, but also recommendations for relevant tags and even other relevant users. Tag recommendation (i.e., recommending to the users relevant tags for an item), has different characteristics than traditional recommendations since the system can recommend recurrent tags, unlike traditional RSs that usually do not recommend the same item twice. In addition, RSs for STS deal with a three-dimensional prob-

lem (user, resource, tag), rather than the traditional two-dimensional problem (user, item), and this affects the complexity of the algorithms. The chapter includes a state-of-the-art survey about the new generation of RSs built to serve STS. It also details the challenges of deploying RS for real world STS, and offers new algorithms for dealing with the challenges of content in both STS and tag recommendation.

Chapter 21 deals with those situations when it would be good if the system could recommend information or items that are relevant to a group of users rather than to an individual. For instance, a RS may select television programs for a group to view or a sequence of songs to listen to, based on models of all group members. Recommending to groups is clearly more complicated than recommending to individuals. Assuming that we know precisely what is good for individual users, the issue is how to combine individual user models. In this chapter, the authors discuss how group recommendation works, what its problems are, and what advances have been made so far.

Chapter 22 discusses the ubiquitous issue of aggregating preferences, criteria or similarities. Normally such aggregation is done by using either the arithmetic mean or maximum/minimum functions. But many other aggregation functions which could deliver flexibility and adaptability, and ultimately more relevant recommendations, are often overlooked. In this chapter the authors review the basics of aggregation functions and their properties and present the most important families, including generalized means, Choquet and Sugeno integrals, ordered weighted averaging, triangular norms and conorms, as well as bipolar aggregation functions. Such functions can model various interactions between the inputs, including conjunctive, disjunctive and mixed behavior.

In Chapter 23, the authors focus on another fundamental problem of RSs, i.e., the need to actively look for new data during the operational life of the recommender. This issue is normally neglected on the assumption that there is not much space for controlling what data (e.g., ratings) the system can collect since these decisions are taken by the users when visiting the system. Actually, the RS provokes the users with its recommendations and many systems actually explicitly ask for user preferences during the recommendation process. Hence, by tuning the process, users can be pushed to provide a range of different information. Specifically they can be requested to rate particular items and the knowledge of the users opinions about these items could be estimated as particularly beneficial according to various criteria, e.g., to provide more diverse recommendations or simply to improve the prediction accuracy of the system for some users or for the whole population of users. At this point active learning comes in; it can augment RSs, helping users to become more self-aware of their own likes/dislikes, leading to more meaningful and useful questions. At the same time active learning can provide new information to the system that can be analyzed for subsequent recommendations. Hence, applying active learning to RSs enables personalization of the recommending process [61]. This is accomplished by allowing the system to actively influence the items the user is exposed to (e.g. the items displayed to the user during sign-up or during regular use), as well as by enabling the user to explore his/her interests freely.

Chapter 24 introduces another emerging topic, i.e., multi-criteria recommender systems. In the 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 users opinions may lead to more effective and accurate recommendations.

Chapter 24 provides an overview of multi-criteria RSs. First, it defines the recommendation problem as a multi-criteria decision-making problem and reviews methods and techniques that can support the implementation of multi-criteria recommenders. Then, it focuses on the category of multi-criteria rating recommender techniques that provide recommendations by modeling the users utility for an item as a vector of ratings along several criteria. A review of current algorithms that use multi-criteria ratings for calculating the rating prediction and generating recommendations is provided. The chapter concludes with a discussion on open issues and future challenges for these recommenders.

The last chapter of this handbook (Chapter 25) surveys articles dealing with security issues. This topic has become a major issue in the past few years. Recent works on the topic include [28, 45, 102, 112]. The chapter analyzes algorithms designed to generate more robust recommendations, i.e., recommendations that are harder for malicious users to influence. In fact, collaborative recommender systems are dependent on the goodwill of their users, i.e., there is an implicit assumption that users will interact with the system with the aim of getting good recommendations for themselves while providing useful data for their neighbors. However, users will have a range of purposes in interacting with RSs and in some cases, these purposes may be counter to those of the system owner or those of the majority of its user population. Namely these users may want to damage the Web site hosting the recommender or to influence the recommendations provided to visitors, e.g., to score some items better or worse rather than to arrive at a fair evaluation.

In this chapter the authors provide a model of efficient attacks, i.e., attacks that can, with relatively low cost, produce a large impact on system output. Since these attacks may very well be launched against a site, it makes sense to detect them so that countermeasures can be taken as soon as possible. At the same time, researchers have studied a number of algorithms that are intended to robustly withstand attacks and which have lower impact curves relative to efficient attacks. These approaches are also surveyed in this chapter. With the combination of these techniques, researchers have sought, not to eliminate attacks, but to control their impact to the point where they are no longer cost-effective.

1.8.2 Challenges

The list of newly emerging and challenging RS research topics is not limited to those described in the chapters that we have mentioned above. Moreover, covering all of them is not within the scope of this short introduction. The reader is referred to the final discussion sections in this handbook for other outstanding problems.

Below we briefly note additional challenging topics that we consider important for the development of the research on RSs and which are not covered in the handbook.

- Scalability of the algorithms with large and real-world datasets. As the research on core techniques progresses and matures, it becomes clear that a fundamental issue for RSs is to determine how to embed the core recommendation techniques in real operational systems and how to deal with massive and dynamic sets of data produced by the interactions of users with items (ratings, preferences, reviews, etc.). A solution that works fine when tested off-line on relatively small data sets may become inefficient or even totally inapplicable on very large datasets. New approaches and large-scale evaluation studies are needed [91, 92, 33, 38, 116, 75, 75].
- Proactive recommender systems, i.e., recommenders that decide to provide recommendations even if not explicitly requested [90, 24, 62, 80]. The largest majority of the recommender systems developed so far follow a "pull" model [94]; where the user originates the request for a recommendation. In the scenarios emerging today, where computers are ubiquitous and users are always connected, it seems natural to imagine that a RS can detect implicit requests. It therefore needs to predict not only what to recommend, but also when and how to "push" its recommendations. In this way the RS can become proactive without being perceived as disturbing.
- Privacy preserving recommender systems [81, 26, 79, 56, 17, 28, 102, 16, 5, 53, 70, 114]. RSs exploit user data to generate personalized recommendations. In the attempt to build increasingly better recommendations, they collect as much user data as possible. This will clearly have a negative impact on the privacy of the users and the users may start feeling that the system knows too much about their true preferences. Therefore, there is a need to design solutions that will parsimoniously and sensibly use user data. At the same time these solutions will ensure that knowledge about the users cannot be freely accessed by malicious users.
- Diversity of the items recommended to a target user [104, 66, 69, 55, 54, 46, 119]. In a recommendation list, it is more likely that the user will find a suitable item if there is a certain degree of diversity among the included items. There is often no value in having perfect recommendations for a restricted type of product, unless the user has expressed a narrow set of preferences. There are many situations, especially in the early stage of a recommendation process, in which the users want to explore new and diverse directions. In such cases, the user is using the recommender as a knowledge discovery tool. The research on this topic is still in an

early stage, and there is a need to characterize the nature of this "diversity", i.e., whether we are looking for diversity among different recommendation sessions or within a session, and how to combine the diversity goal with the accuracy of the recommendation.

- Integration of long-term and short-term user preferences in the process of building a recommendation list [6, 40, 74]. Recommender systems may be divided in two classes: those that build a long-term profile, generated by aggregating all the user transaction data collected by the system (e.g., collaborative filtering) and those that are more focused on capturing the ephemeral preferences of the user, e.g., as in case-based approaches. Obviously both aspects are important and either the precise user task or the availability of items may come under consideration in resolving the preference integration problem. In fact, new research is required to build hybrid models that can correctly decide to drift or not toward the contingent user's preferences when there is enough evidence to suggest that the user's short-term preferences are departing from the long-term ones.
- Generic user models and cross domain recommender systems are able to mediate user data through different systems and application domains [41, 18, 52, 19, 20, 49, 15]. Using generic user model techniques, a single RS can produce recommendations about a variety of items. This is normally not possible for a general RS which can combine more techniques in a hybrid approach, but cannot easily benefit from user preferences collected in one domain to generate recommendations in a different one.
- Distributed recommender systems that operate in open networks [38, 116, 92, 113, 17, 102]. The computational model of the largest majority of RSs adheres to a typical client-server architecture, where the user-client requests recommendations to the server-recommender which replies with the suggestions. This is clearly a severe limitation and suffers from all the classical problems of centralized systems. The emerging scenario of grid or cloud computing can become an excellent opportunity to implement more robust and flexible computational models for RSs.
- Recommender that optimize a sequence of recommendations [120, 99, 10, 59, 61, 107, 106]. We mentioned already that conversational RSs have emerged in the attempt to improve the quality of recommendations provided by the systems based on a simpler approach: a one-time request/response. Conversational RSs can be further improved by implementing learning capabilities that can optimize not only the items that are recommended but also how the dialogue between the user and the system must unfold in all possible situations.
- Recommenders designed to operate in mobile devices and usage contexts [117, 98, 55, 51, 4, 115, 111, 57, 29, 9, 77, 76, 89, 73, 44, 95, 13]. Mobile computing is emerging as the most natural platform for personal computing. Many recommendation requests are likely to be made when the user is on the move, e.g., at shops or hotels in a visited city. This necessitates "mobilizing" the user interface and to design computational solutions that can efficiently use the still limited resources (computational power and screen size) of the mobile devices.

Finally, before ending this introduction, we want to present some additional challenges that were discussed in a tutorial held at the latest RecSys conference in New York, October 22-25, 2009 [http://recsys.acm.org/tutorial3.pdf]. John Riedl (University of Minnesota), Todd Beaupre (Yahoo!) and John Sanders (Netflix) mentioned eight important challenges for the research on recommender systems: transparency, exploration versus exploitation, guided navigation, time value, user action interpretation, evaluating recommenders, scalability, academic/industry partnerships.

Some of these issues have already been discussed in this introduction. For example, transparency was introduced when we discussed the role of the explanation of a recommendation, and we stressed the important role it plays in order to present a recommendation as more acceptable for the user. Also the evaluation of RSs, i.e., the range of possible and important dimensions that can be measured in an evaluation is a topic fully addressed in another chapter (Chapter 8).

The time value of recommendations is also partially discussed in our remarks about context-aware recommenders (Chapter 7). However, the challenge refers to the fact that a given set of recommendations may not be applicable forever but there could be a time interval when these items can be recommended. This is clear, for instance, when it comes to news items; people want to be informed about the most recent events and news cannot be meaningfully recommended even one day after the initial announcement.

Exploration vs. exploitation is touched upon in active learning (Chapter 23). This challenge refers to the fundamental dilemma that a designer must properly tackle, i.e., whether to keep recommending items that the system can now identify as good recommendations, given the data currently available for the system or to further explore user preferences (e.g., asking to rate additional and particular items) in order to build newer and possibly even better recommendations in the future.

indexGuided navigation Guided navigation refers to combining classical recommendation lists, i.e., suggestions with tools that let the user navigate more autonomously in the space of possible options. User action interpretation refers to the possibility that in addition to explicit ratings there could be many more actions performed by the user operating the recommender that can be detected, analyzed and used to build a better prediction model. The idea is that every single user action should be exploited in the recommendation process. But it is challenging to interpret the user's actions, i.e., the intent behind an action, and there are actions that should be discarded because they were not produced by genuine users, such as, actions performed by different users on the same browser, or false and malicious registrations or data or log data caused by robots or crawlers.

Scalability was also mentioned earlier. We stress again that this is clearly an issue about which discussion is missing in the current literature since it has been mostly investigated by practitioners.

Finally the discussion in that workshop became quite animated when the matter of cooperation between industry and academia was touched upon. Industry has specific problems but is not making them clearly visible. This is happening for many reasons, including the need to not disclose to competitors critical information. Con-

versely, academia is looking for problems that can be tackled in a framework of the resources and time available to them and will generally address a topic only if it is likely to have an impact in the scientific community. This has made and will make industry-academic cooperation difficult. But RSs is a research field that requires new concrete challenges and there is a real risk of stagnation if we fail to tackle the useful but risky challenges in favor of solved or mature problems.

We hope that this handbook, as a useful tool for practitioners and researchers, will contribute to further develop knowledge in this exciting and useful research area. In this way we believe that we can reduce the risk that these two groups will follow different roads. Currently the research on RSs has greatly benefited from the combined interest and efforts that industry and academia have invested in this field. We therefore wish the best to both groups as they read this handbook and we hope that it will attract even more researchers to work in this highly interesting and challenging field.

References

1. Adomavicius, G., Sankaranarayanan, R., Sen, S., Tuzhilin, A.: Incorporating contextual information in recommender systems using a multidimensional approach. ACM Trans. Inf. Syst. **23**(1), 103–145 (2005)
2. Adomavicius, G., Tuzhilin, A.: Personalization technologies: a process-oriented perspective. Commun. ACM **48**(10), 83–90 (2005)
3. Adomavicius, G., Tuzhilin, A.: 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 **17**(6), 734–749 (2005)
4. Ahn, H., Kim, K.J., Han, I.: Mobile advertisement recommender system using collaborative filtering: Mar-cf. In: Proceedings of the 2006 Conference of the Korea Society of Management Information Systems, pp. 709–715 (2006)
5. Aïmeur, E., Brassard, G., Fernandez, J.M., Onana, F.S.M.: Alambic : a privacy-preserving recommender system for electronic commerce. Int. J. Inf. Sec. **7**(5), 307–334 (2008)
6. Aimeur, E., Vézeau, M.: Short-term profiling for a case-based reasoning recommendation system. In: R.L. de Mántaras, E. Plaza (eds.) Machine Learning: 2000, 11th European Conference on Machine Learning, pp. 23–30. Springer (2000)
7. Anand, S.S., Mobasher, B.: Intelligent techniques for web personalization. In: Intelligent Techniques for Web Personalization, pp. 1–36. Springer (2005)
8. Arazy, O., Kumar, N., Shapira, B.: Improving social recommender systems. IT Professional **11**(4), 38–44 (2009)
9. Averjanova, O., Ricci, F., Nguyen, Q.N.: Map-based interaction with a conversational mobile recommender system. In: The Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, 2008. UBICOMM '08, pp. 212–218 (2008)
10. Baccigalupo, C., Plaza, E.: Case-based sequential ordering of songs for playlist recommendation. In: T. Roth-Berghofer, M.H. Göker, H.A. Güvenir (eds.) ECCBR, *Lecture Notes in Computer Science*, vol. 4106, pp. 286–300. Springer (2006)
11. Bailey, R.A.: Design of comparative experiments. Cambridge University Press Cambridge (2008)
12. Balabanovic, M., Shoham, Y.: Content-based, collaborative recommendation. Communication of ACM **40**(3), 66–72 (1997)
13. Bellotti, V., Begole, J.B., hsin Chi, E.H., Ducheneaut, N., Fang, J., Isaacs, E., King, T.H., Newman, M.W., Partridge, K., Price, B., Rasmussen, P., Roberts, M., Schiano, D.J., Walen-

dowski, A.: Activity-based serendipitous recommendations with the magitti mobile leisure guide. In: M. Czerwinski, A.M. Lund, D.S. Tan (eds.) CHI, pp. 1157–1166. ACM (2008)
14. Ben-Shimon, D., Tsikinovsky, A., Rokach, L., Meisels, A., Shani, G., Naamani, L.: Recommender system from personal social networks. In: K. Wegrzyn-Wolska, P.S. Szczepaniak (eds.) AWIC, *Advances in Soft Computing*, vol. 43, pp. 47–55. Springer (2007)
15. Berkovsky, S.: Mediation of User Models: for Enhanced Personalization in Recommender Systems. VDM Verlag (2009)
16. Berkovsky, S., Borisov, N., Eytani, Y., Kuflik, T., Ricci, F.: Examining users' attitude towards privacy preserving collaborative filtering. In: International Workshop on Data Mining for User Modeling, at User Modeling 2007, 11th International Conference, UM 2007, Corfu, Greece, June 25, 2007, Proceedings (2007)
17. Berkovsky, S., Eytani, Y., Kuflik, T., Ricci, F.: Enhancing privacy and preserving accuracy of a distributed collaborative filtering. In: RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems, pp. 9–16. ACM Press, New York, NY, USA (2007)
18. Berkovsky, S., Kuflik, T., Ricci, F.: Cross-technique mediation of user models. In: Proceedings of International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems [AH2006], pp. 21–30. Dublin (2006)
19. Berkovsky, S., Kuflik, T., Ricci, F.: Mediation of user models for enhanced personalization in recommender systems. User Modeling and User-Adapted Interaction **18**(3), 245–286 (2008)
20. Berkovsky, S., Kuflik, T., Ricci, F.: Cross-representation mediation of user models. User Modeling and User-Adapted Interaction **19**(1-2), 35–63 (2009)
21. Billsus, D., Pazzani, M.: Learning probabilistic user models. In: UM97 Workshop on Machine Learning for User Modeling (1997). URL `http://www.dfki.de/~bauer/um-ws/`
22. Bridge, D., Göker, M., McGinty, L., Smyth, B.: Case-based recommender systems. The Knowledge Engineering review **20**(3), 315–320 (2006)
23. Brusilovsky, P.: Methods and techniques of adaptive hypermedia. User Modeling and User-Adapted Interaction **6**(2-3), 87–129 (1996)
24. Bulander, R., Decker, M., Schiefer, G., Kolmel, B.: Comparison of different approaches for mobile advertising. Mobile Commerce and Services, 2005. WMCS '05. The Second IEEE International Workshop on pp. 174–182 (2005)
25. Burke, R.: Hybrid web recommender systems. In: The Adaptive Web, pp. 377–408. Springer Berlin / Heidelberg (2007)
26. Canny, J.F.: Collaborative filtering with privacy. In: IEEE Symposium on Security and Privacy, pp. 45–57 (2002)
27. Carenini, G., Smith, J., Poole, D.: Towards more conversational and collaborative recommender systems. In: Proceedings of the 2003 International Conference on Intelligent User Interfaces, January 12-15, 2003, Miami, FL, USA, pp. 12–18 (2003)
28. Cheng, Z., Hurley, N.: Effective diverse and obfuscated attacks on model-based recommender systems. In: RecSys '09: Proceedings of the third ACM conference on Recommender systems, pp. 141–148. ACM, New York, NY, USA (2009)
29. Church, K., Smyth, B., Cotter, P., Bradley, K.: Mobile information access: A study of emerging search behavior on the mobile internet. ACM Trans. Web **1**(1), 4 (2007)
30. Cosley, D., Lam, S.K., Albert, I., Konstant, J.A., Riedl, J.: Is seeing believing? how recommender system interfaces affect users' opinions. In: In Proceedings of the CHI 2003 Conference on Human factors in Computing Systems. Fort Lauderdale, FL (2003)
31. Felfernig, A., Friedrich, G., Schubert, M., Mandl, M., Mairitsch, M., Teppan, E.: Plausible repairs for inconsistent requirements. In: Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI'09), pp. 791–796. Pasadena, California, USA (2009)
32. Fisher, G.: User modeling in human-computer interaction. User Modeling and User-Adapted Interaction **11**, 65–86 (2001)
33. George, T., Merugu, S.: A scalable collaborative filtering framework based on co-clustering. In: Proceedings of the 5th IEEE Conference on Data Mining (ICDM), pp. 625–628. IEEE Computer Society, Los Alamitos, CA, USA (2005)

34. Golbeck, J.: Generating predictive movie recommendations from trust in social networks. In: Trust Management, 4th International Conference, iTrust 2006, Pisa, Italy, May 16-19, 2006, Proceedings, pp. 93–104 (2006)
35. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using collaborative filtering to weave an information tapestry. Commun. ACM **35**(12), 61–70 (1992)
36. Groh, G., Ehmig, C.: Recommendations in taste related domains: collaborative filtering vs. social filtering. In: GROUP '07: Proceedings of the 2007 international ACM conference on Supporting group work, pp. 127–136. ACM, New York, NY, USA (2007)
37. Guy, I., Zwerdling, N., Carmel, D., Ronen, I., Uziel, E., Yogev, S., Ofek-Koifman, S.: Personalized recommendation of social software items based on social relations. In: RecSys '09: Proceedings of the third ACM conference on Recommender systems, pp. 53–60. ACM, New York, NY, USA (2009)
38. Han, P., Xie, B., Yang, F., Sheng, R.: A scalable p2p recommender system based on distributed collaborative filtering. Expert systems with applications (2004)
39. Hayes, C., Cunningham, P.: Smartradio-community based music radio. Knowledge Based Systems **14**(3-4), 197–201 (2001)
40. He, L., Zhang, J., Zhuo, L., Shen, L.: Construction of user preference profile in a personalized image retrieval. In: Neural Networks and Signal Processing, 2008 International Conference on, pp. 434–439 (2008)
41. Heckmann, D., Schwartz, T., Brandherm, B., Schmitz, M., von Wilamowitz-Moellendorff, M.: Gumo - the general user model ontology. In: User Modeling 2005, 10th International Conference, UM 2005, Edinburgh, Scotland, UK, July 24-29, 2005, Proceedings, pp. 428–432 (2005)
42. Herlocker, J., Konstan, J., Riedl, J.: Explaining collaborative filtering recommendations. In: In proceedings of ACM 2000 Conference on Computer Supported Cooperative Work, pp. 241–250 (2000)
43. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Transaction on Information Systems **22**(1), 5–53 (2004)
44. Horozov, T., Narasimhan, N., Vasudevan, V.: Using location for personalized POI recommendations in mobile environments. In: Proc. Int'l Sym. Applications on Internet, pp. 124–129. EEE Computer Society (2006)
45. Hurley, N., Cheng, Z., Zhang, M.: Statistical attack detection. In: RecSys '09: Proceedings of the third ACM conference on Recommender systems, pp. 149–156. ACM, New York, NY, USA (2009)
46. Hwang, C.S., Kuo, N., Yu, P.: Representative-based diversity retrieval. In: Innovative Computing Information and Control, 2008. ICICIC '08. 3rd International Conference on, pp. 155–155 (2008)
47. Jannach, D.: Finding preferred query relaxations in content-based recommenders. In: 3rd International IEEE Conference on Intelligent Systems, pp. 355–360 (2006)
48. Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: Recommender Systems An Introduction. Cambridge University Press (2010)
49. Jessenitschnig, M., Zanker, M.: A generic user modeling component for hybrid recommendation strategies. E-Commerce Technology, IEEE International Conference on **0**, 337–344 (2009). DOI http://doi.ieeecomputersociety.org/10.1109/CEC.2009.83
50. Kay, J.: Scrutable adaptation: Because we can and must. In: Adaptive Hypermedia and Adaptive Web-Based Systems, 4th International Conference, AH 2006, Dublin, Ireland, June 21-23, 2006, Proceedings, pp. 11–19 (2006)
51. Kim, C.Y., Lee, J.K., Cho, Y.H., Kim, D.H.: Viscors: A visual-content recommender for the mobile web. IEEE Intelligent Systems **19**(6), 32–39 (2004)
52. Kobsa, A.: Generic user modeling systems. In: P. Brusilovsky, A. Kobsa, W. Nejdl (eds.) The Adaptive Web, *Lecture Notes in Computer Science*, vol. 4321, pp. 136–154. Springer (2007)
53. Kobsa, A.: Privacy-enhanced personalization. In: D. Wilson, H.C. Lane (eds.) FLAIRS Conference, p. 10. AAAI Press (2008)
54. Koren, Y., Bell, R.M., Volinsky, C.: Matrix factorization techniques for recommender systems. IEEE Computer **42**(8), 30–37 (2009)

55. Kramer, R., Modsching, M., ten Hagen, K.: Field study on methods for elicitation of preferences using a mobile digital assistant for a dynamic tour guide. In: SAC '06: Proceedings of the 2006 ACM symposium on Applied computing, pp. 997–1001. ACM Press, New York, NY, USA (2006)
56. Lam, S.K., Frankowski, D., Riedl, J.: Do you trust your recommendations? an exploration of security and privacy issues in recommender systems. In: G. Müller (ed.) ETRICS, *Lecture Notes in Computer Science*, vol. 3995, pp. 14–29. Springer (2006)
57. Lee, H., Park, S.J.: Moners: A news recommender for the mobile web. Expert Systems with Applications **32**(1), 143 – 150 (2007)
58. Linden, G., Smith, B., York, J.: Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing **7**(1), 76–80 (2003)
59. Mahmood, T., Ricci, F.: Towards learning user-adaptive state models in a conversational recommender system. In: A. Hinneburg (ed.) LWA 2007: Lernen - Wissen - Adaption, Halle, September 2007, Workshop Proceedings, pp. 373–378. Martin-Luther-University Halle-Wittenberg (2007)
60. Mahmood, T., Ricci, F.: Improving recommender systems with adaptive conversational strategies. In: C. Cattuto, G. Ruffo, F. Menczer (eds.) Hypertext, pp. 73–82. ACM (2009)
61. Mahmood, T., Ricci, F., Venturini, A., Höpken, W.: Adaptive recommender systems for travel planning. In: W.H. Peter OConnor, U. Gretzel (eds.) Information and Communication Technologies in Tourism 2008, proceedings of ENTER 2008 International Conference, pp. 1–11. Springer, Innsbruck (2008)
62. Mahmoud, Q.: Provisioning context-aware advertisements to wireless mobile users. Multimedia and Expo, 2006 IEEE International Conference on pp. 669–672 (2006)
63. Manning, C.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
64. Massa, P., Avesani, P.: Trust-aware collaborative filtering for recommender systems. In: Proceedings of the International Conference on Cooperative Information Systems, CoopIS, pp. 492–508 (2004)
65. McCarthy, K., Salamó, M., Coyle, L., McGinty, L., Smyth, B., Nixon, P.: Group recommender systems: a critiquing based approach. In: C. Paris, C.L. Sidner (eds.) IUI, pp. 267–269. ACM (2006)
66. McGinty, L., Smyth, B.: On the role of diversity in conversational recommender systems. In: A. Aamodt, D. Bridge, K. Ashley (eds.) ICCBR 2003, the 5th International Conference on Case-Based Reasoning, pp. 276–290. Trondheim, Norway (2003)
67. McGinty, L., Smyth, B.: Adaptive selection: An analysis of critiquing and preference-based feedback in conversational recommender systems. International Journal of Electronic Commerce **11**(2), 35–57 (2006)
68. McNee, S.M., Riedl, J., Konstan, J.A.: Being accurate is not enough: how accuracy metrics have hurt recommender systems. In: CHI '06: CHI '06 extended abstracts on Human factors in computing systems, pp. 1097–1101. ACM Press, New York, NY, USA (2006)
69. McSherry, D.: Diversity-conscious retrieval. In: S. Craw, A. Preece (eds.) Advances in Case-Based Reasoning, Proceedings of the 6th European Conference on Case Based Reasoning, ECCBR 2002, pp. 219–233. Springer Verlag, Aberdeen, Scotland (2002)
70. McSherry, F., Mironov, I.: Differentially private recommender systems: building privacy into the net. In: KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 627–636. ACM, New York, NY, USA (2009)
71. Mirzadeh, N., Ricci, F.: Cooperative query rewriting for decision making support and recommender systems. Applied Artificial Intelligence **21**, 1–38 (2007)
72. Montaner, M., López, B., de la Rosa, J.L.: A taxonomy of recommender agents on the internet. Artificial Intelligence Review **19**(4), 285–330 (2003)
73. Nguyen, Q.N., Ricci, F.: Replaying live-user interactions in the off-line evaluation of critique-based mobile recommendations. In: RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems, pp. 81–88. ACM Press, New York, NY, USA (2007)

74. Nguyen, Q.N., Ricci, F.: Conversational case-based recommendations exploiting a structured case model. In: Advances in Case-Based Reasoning, 9th European Conference, ECCBR 2008, Trier, Germany, September 1-4, 2008. Proceedings, pp. 400–414 (2008)
75. Papagelis, M., Rousidis, I., Plexousakis, D., Theoharopoulos, E.: Incremental collaborative filtering for highly-scalable recommendation algorithms. In: M.S. Hacid, N.V. Murray, Z.W. Ras, S. Tsumoto (eds.) ISMIS, *Lecture Notes in Computer Science*, vol. 3488, pp. 553–561. Springer (2005)
76. Park, M.H., Hong, J.H., Cho, S.B.: Location-based recommendation system using bayesian user's preference model in mobile devices. In: J. Indulska, J. Ma, L.T. Yang, T. Ungerer, J. Cao (eds.) UIC, *Lecture Notes in Computer Science*, vol. 4611, pp. 1130–1139. Springer (2007)
77. Park, S., Kang, S., Kim, Y.K.: A channel recommendation system in mobile environment. Consumer Electronics, IEEE Transactions on **52**(1), 33–39 (2006). DOI 10.1109/TCE.2006.1605022
78. Pazzani, M.J.: A framework for collaborative, content-based and demographic filtering. Artificial Intelligence Review **13**, 393–408 (1999)
79. Polat, H., Du, W.: Privacy-preserving collaborative filtering using randomized perturbation techniques. In: Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM 2003), 19-22 December 2003, Melbourne, Florida, USA, pp. 625–628 (2003)
80. Puerta Melguizo, M.C., Boves, L., Deshpande, A., Ramos, O.M.: A proactive recommendation system for writing: helping without disrupting. In: ECCE '07: Proceedings of the 14th European conference on Cognitive ergonomics, pp. 89–95. ACM, New York, NY, USA (2007). DOI http://doi.acm.org/10.1145/1362550.1362569
81. Ramakrishnan, N., Keller, B.J., Mirza, B.J., Grama, A., Karypis, G.: When being weak is brave: Privacy in recommender systems. IEEE Internet Computing **cs.CG/0105028** (2001)
82. Reilly, J., McCarthy, K., McGinty, L., Smyth, B.: Dynamic critiquing. In: Advances in Case-Based Reasoning, 7th European Conference, ECCBR 2004, Madrid, Spain, August 30 - September 2, 2004, Proceedings, pp. 763–777 (2004)
83. Reilly, J., Zhang, J., McGinty, L., Pu, P., Smyth, B.: Evaluating compound critiquing recommenders: a real-user study. In: EC '07: Proceedings of the 8th ACM conference on Electronic commerce, pp. 114–123. ACM, New York, NY, USA (2007)
84. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: Grouplens: An open architecture for collaborative filtering of netnews. In: Proceedings ACM Conference on Computer-Supported Cooperative Work, pp. 175–186 (1994)
85. Resnick, P., Varian, H.R.: Recommender systems. Communications of the ACM **40**(3), 56–58 (1997)
86. Ricci, F.: Travel recommender systems. IEEE Intelligent Systems **17**(6), 55–57 (2002)
87. Ricci, F., Cavada, D., Mirzadeh, N., Venturini, A.: Case-based travel recommendations. In: D.R. Fesenmaier, K. Woeber, H. Werthner (eds.) Destination Recommendation Systems: Behavioural Foundations and Applications, pp. 67–93. CABI (2006)
88. Ricci, F., Missier, F.D.: Supporting travel decision making through personalized recommendation. In: C.M. Karat, J.O. Blom, J. Karat (eds.) Designing Personalized User Experiences in eCommerce, pp. 231–251. Kluwer Academic Publisher (2004)
89. Ricci, F., Nguyen, Q.N.: Acquiring and revising preferences in a critique-based mobile recommender system. IEEE Intelligent Systems **22**(3), 22–29 (2007). DOI http://doi.ieeecomputersociety.org/10.1109/MIS.2007.43
90. Sae-Ueng, S., Pinyapong, S., Ogino, A., Kato, T.: Personalized shopping assistance service at ubiquitous shop space. Advanced Information Networking and Applications - Workshops, 2008. AINAW 2008. 22nd International Conference on pp. 838–843 (2008). DOI 10.1109/WAINA.2008.287
91. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Incremental singular value decomposition algorithms for highly scalable recommender systems. In: Proceedings of the 5th International Conference in Computers and Information Technology (2002)

92. Sarwar, B.M., Konstan, J.A., Riedl, J.: Distributed recommender systems for internet commerce. In: M. Khosrow-Pour (ed.) Encyclopedia of Information Science and Technology (II), pp. 907–911. Idea Group (2005)
93. Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S.: Collaborative filtering recommender systems. In: The Adaptive Web, pp. 291–324. Springer Berlin / Heidelberg (2007)
94. Schafer, J.B., Konstan, J.A., Riedl, J.: E-commerce recommendation applications. Data Mining and Knowledge Discovery **5**(1/2), 115–153 (2001)
95. Schifanella, R., Panisson, A., Gena, C., Ruffo, G.: Mobhinter: epidemic collaborative filtering and self-organization in mobile ad-hoc networks. In: RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems, pp. 27–34. ACM, New York, NY, USA (2008)
96. Schwartz, B.: The Paradox of Choice. ECCO, New York (2004)
97. van Setten, M., McNee, S.M., Konstan, J.A.: Beyond personalization: the next stage of recommender systems research. In: R.S. Amant, J. Riedl, A. Jameson (eds.) IUI, p. 8. ACM (2005)
98. van Setten, M., Pokraev, S., Koolwaaij, J.: Context-aware recommendations in the mobile tourist application compass. In: W. Nejdl, P. De Bra (eds.) Adaptive Hypermedia 2004, pp. 235–244. Springer Verlag (2004)
99. Shani, G., Heckerman, D., Brafman, R.I.: An mdp-based recommender system. Journal of Machine Learning Research **6**, 1265–1295 (2005)
100. Sharda, N.: Tourism Informatics: Visual Travel Recommender Systems, Social Communities, and User Interface Design. Information Science Reference (2009)
101. Shardanand, U., Maes, P.: Social information filtering: algorithms for automating "word of mouth". In: Proceedings of the Conference on Human Factors in Computing Systems (CHI'95), pp. 210–217 (1995)
102. Shokri, R., Pedarsani, P., Theodorakopoulos, G., Hubaux, J.P.: Preserving privacy in collaborative filtering through distributed aggregation of offline profiles. In: RecSys '09: Proceedings of the third ACM conference on Recommender systems, pp. 157–164. ACM, New York, NY, USA (2009)
103. Sinha, R.R., Swearingen, K.: Comparing recommendations made by online systems and friends. In: DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries (2001)
104. Smyth, B., McClave, P.: Similarity vs diversity. In: Proceedings of the 4th International Conference on Case-Based Reasoning. Springer-Verlag (2001)
105. Swearingen, K., Sinha, R.: Beyond algorithms: An HCI perspective on recommender systems. In: J.L. Herlocker (ed.) Recommender Systems, papers from the 2001 ACM SIGIR Workshop. New Orleans, LA - USA (2001)
106. Taghipour, N., Kardan, A.: A hybrid web recommender system based on q-learning. In: Proceedings of the 2008 ACM Symposium on Applied Computing (SAC), Fortaleza, Ceara, Brazil, March 16-20, 2008, pp. 1164–1168 (2008)
107. Taghipour, N., Kardan, A., Ghidary, S.S.: Usage-based web recommendations: a reinforcement learning approach. In: Proceedings of the 2007 ACM Conference on Recommender Systems, RecSys 2007, Minneapolis, MN, USA, October 19-20, 2007, pp. 113–120 (2007)
108. Takács, G., Pilászy, I., Németh, B., Tikk, D.: Scalable collaborative filtering approaches for large recommender systems. J. Mach. Learn. Res. **10**, 623–656 (2009)
109. Tan, P.N.: Introduction to Data Mining. Pearson Addison Wesley, San Francisco (2006)
110. Thompson, C.A., Goker, M.H., Langley, P.: A personalized system for conversational recommendations. Artificial Intelligence Research **21**, 393–428 (2004)
111. Tung, H.W., Soo, V.W.: A personalized restaurant recommender agent for mobile e-service. In: S.T. Yuan, J. Liu (eds.) Proceedings of the IEEE International Conference on e-Technology, e-Commerce and e-Service, EEE'04, pp. 259–262. IEEE Computer Society Press, Taipei, Taiwan (2004)
112. Van Roy, B., Yan, X.: Manipulation-resistant collaborative filtering systems. In: RecSys '09: Proceedings of the third ACM conference on Recommender systems, pp. 165–172. ACM, New York, NY, USA (2009)

113. Wang, J., Pouwelse, J.A., Lagendijk, R.L., Reinders, M.J.T.: Distributed collaborative filtering for peer-to-peer file sharing systems. In: H. Haddad (ed.) SAC, pp. 1026–1030. ACM (2006)
114. Wang, Y., Kobsa, A.: Performance evaluation of a privacy-enhancing framework for personalized websites. In: G.J. Houben, G.I. McCalla, F. Pianesi, M. Zancanaro (eds.) UMAP, *Lecture Notes in Computer Science*, vol. 5535, pp. 78–89. Springer (2009)
115. Wietsma, R.T.A., Ricci, F.: Product reviews in mobile decision aid systems. In: Pervasive Mobile Interaction Devices (PERMID 2005) - Mobile Devices as Pervasive User Interfaces and Interaction Devices - Workshop in conjunction with: The 3rd International Conference on Pervasive Computing (PERVASIVE 2005), May 11 2005, Munich, Germany, pp. 15–18. LMU Munich (2005)
116. Xie, B., Han, P., Yang, F., Shen, R.: An efficient neighbor searching scheme of distributed collaborative filtering on p2p overlay network. Database and Expert Systems Applications pp. 141–150 (2004)
117. Yuan, S.T., Tsao, Y.W.: A recommendation mechanism for contextualized mobile advertising. Expert Systems with Applications **24**(4), 399–414 (2003)
118. Zhang, F.: Research on recommendation list diversity of recommender systems. Management of e-Commerce and e-Government, International Conference on pp. 72–76 (2008)
119. Zhang, M.: Enhancing diversity in top-n recommendation. In: RecSys '09: Proceedings of the third ACM conference on Recommender systems, pp. 397–400. ACM, New York, NY, USA (2009)
120. Zhou, B., Hui, S., Chang, K.: An intelligent recommender system using sequential web access patterns. In: Cybernetics and Intelligent Systems, 2004 IEEE Conference on, vol. 1, pp. 393–398 vol.1 (2004)
121. Ziegler, C.N., McNee, S.M., Konstan, J.A., Lausen, G.: Improving recommendation lists through topic diversification. In: WWW '05: Proceedings of the 14th international conference on World Wide Web, pp. 22–32. ACM Press, New York, NY, USA (2005)