Intelligent Techniques for Web Personalization

Sarabjot Singh Anand[1] and Bamshad Mobasher[2]

[1] Department of Computer Science, University of Warwick, Coventry CV4 7AL, UK
s.s.anand@warwick.ac.uk

[2] Center for Web Intelligence, School of Computer Science, Telecommunications and Information Systems, DePaul University, Chicago, Illinois, USA
mobasher@cs.depaul.edu

Abstract. In this chapter we provide a comprehensive overview of the topic of Intelligent Techniques for Web Personalization. Web Personalization is viewed as an application of data mining and machine learning techniques to build models of user behaviour that can be applied to the task of predicting user needs and adapting future interactions with the ultimate goal of improved user satisfaction. This chapter survey's the state-of-the-art in Web personalization. We start by providing a description of the personalization process and a classification of the current approaches to Web personalization. We discuss the various sources of data available to personalization systems, the modelling approaches employed and the current approaches to evaluating these systems. A number of challenges faced by researchers developing these systems are described as are solutions to these challenges proposed in literature. The chapter concludes with a discussion on the open challenges that must be addressed by the research community if this technology is to make a positive impact on user satisfaction with the Web.

1 Introduction

The term *information overload* is almost synonymous with the Internet, referring to the sheer volume of information that exists in electronic format on the Internet and the inability of humans to consume it. The freedom to express oneself through publishing content to the Web has a number of advantages, however, the task of the consumer of this content is made more difficult not only due to the need to assess the relevance of the information to the task at hand but also due to the need to assess the reliability and trustworthiness of the information available.

Information retrieval technologies have matured in the last decade and search engines do a good job of indexing content available on the Internet and making it available to users, if the user knows exactly what he is looking for but often, search engines themselves can return more information than the user could possibly process. Also, most widely used search engines use only the content of Web documents and their link structures to assess the relevance of the document to the user's query. Hence, no matter who the user of the search engine is, if the same query is provided as input to the search engine, the results returned will be exactly the same.

The need to provide users with information tailored to their needs led to the development of various information filtering techniques that built profiles of users and attempted to filter large data streams, presenting the user with only those items that it believes to be of interest to the user.

The goal of personalization is to provide users with what they want or need without requiring them to ask for it explicitly [1]. This does not in any way imply a fully-automated process, instead it encompasses scenarios where the user is not able to fully express exactly what the are looking for but in interacting with an intelligent system can lead them to items of interest.

Intelligent Techniques for Web Personalization is about leveraging all available information about users of the Web to deliver a personal experience. The “intelligence” of these techniques is at various levels ranging from the generation of useful, actionable knowledge through to the inferences made using this knowledge and available domain knowledge at the time of generating the personalized experience for the user. As such, this process of personalization can be viewed as an application of data mining and hence requiring support for all the phases of a typical data mining cycle [2] including data collection, pre-processing, pattern discovery and evaluation, in an off-line mode, and finally the deployment of the knowledge in real-time to mediate between the user and the Web.

In this chapter we provide an overview of the topic of Intelligent Techniques for Web Personalization. In Section 2 we describe the process of personalization in terms of an application of a data mining to the Web. Section 3 provides a classification of approaches to Web personalization while in Section 4 we describe the data available for mining in the Web domain, specifically for the generation of user models. Section 5 describes the various techniques used in generating a personalized Web experience for users highlighting the advantages and disadvantages associated with each approach. Issues associated with current approaches to Web personalization are discussed in Section 6. The important issue of evaluating Web personalization is discussed in Section 7. Finally the chapter concludes in Section 8 with a discussion on the current state and future direction of research in Web personalization.

2 The Personalization Process

Personalization aims to provide users with what they need without requiring them to ask for it explicitly. This means that a personalization system must somehow infer what the user requires based on either previous or current interactions with the user. This in itself assumes that the system somehow obtains information on the user and infers what his needs are based on this information.

In the context of this book, we focus on personalization of the Web or more generally, any repository of objects (items) browseable either through navigation of links between the objects or through search. Hence, the domain we address includes Intranets and the Internet as well as product/service catalogues. More formally, we assume that we are given a universe of n items, $I = \{i_j : 1 \leq j \leq n\}$, and a set of m users, $U = \{u_k : 1 \leq k \leq m\}$, that have shown an interest, in the past, in a subset of the universe of items. Additionally, each user, u_k, may be described as a t-dimensional vector $(a_1^k, a_2^k, ..., a_t^k)$ and each item, i_j, by an s-dimensional vector $(b_1^j, b_2^j, ..., b_s^j)$. Further domain knowledge about the items, for example, in the form of an ontology, may also be available. We will assume the existence of a function $r_{u_k} : I \rightarrow [0,1] \cup \perp$ where $i_j = \perp$ signifies that the item i_j has not been rated by the user, u_k [1] that assigns a rating to each item in I. Let $I_k^{(u)}$ be the set of items currently unrated by the user u_k, i.e. $I_k^{(u)} = \{i_j : i_j \in I \wedge r_{u_k}(i_j) = \perp\}$. Similarly let $I_k^{(r)}$ be the set of items rated by the user u_k, i.e. $I_k^{(r)} = I - I_k^{(u)}$.

The goal of personalization is to recommend items, i_j, to a user u_a, referred to as the *active user*, where $i_j \in I_a^{(u)}$ that would be of interest to the user.

Central to any system capable of achieving this would be a user-centric data model. This data may be collected implicitly or explicitly but in either case must be attributable to a specific user. While this seems obvious, on the Web it is not always straightforward to associate, especially implicitly collected data with a user. For example, server logs provide a rich albeit noisy source of data from which implicit measures of user interest may be derived. Due to the stateless nature of the Web, a number of heuristics must be used along with technologies such as cookies to identify return visitors and attribute a sequence of behaviours to a single user visit/transaction [3].

Once the data has been cleansed and stored within a user-centric model, analysis of the data can be carried out with the aim of building a user model that can be used for predicting future interests of the user. The exact representation of this user model differs based on the approach taken to achieve personalization and the granularity of the information available. The task of learning the model would therefore differ in complexity based on the expressiveness of the user profile representation chosen and the data available. For example, the profile may be represented as vector of 2-tuples $u_k^{(n)}(< i_1, r_{u_k}(i_1) >, < i_2, r_{u_k}(i_2) >, < i_3, r_{u_k}(i_3) > .... < i_n, r_{u_k}(i_n) >)$ where i_j's $\in I$ and r_{u_k} is the rating function for user u_k. In the presence of a domain ontology, the user profile may actually reflect the structure of the domain [4], [5], [6]. Recently, there has been a lot of research interest in generating aggregate usage profiles rather than individual user profiles [7], that represent group behaviour as opposed to the behaviour of a single user. The distinction between individual and aggregate profiles for personalization is akin to the distinction between lazy and eager learning in machine learning.

The next stage of the process is the evaluation of the profiles/knowledge generated. The aim of this stage is to evaluate how effective the discovered knowledge is in predicting user interest. Common metrics used during this phase are coverage, mean absolute error and ROC sensitivity. See Section 7 for a more detailed discussion on evaluation metrics.

The deployment stage follows evaluation, where the knowledge generated and evaluated within the previous two stages of the process is deployed to generate recommendations in real-time as the users navigate the Web site. The key challenge at this stage is scalability with respect to the number of concurrent users using the system.

An essential, though often overlooked, part of the personalization process is the monitoring of the personalization. Anand et al. suggest that the success of the person-

[1] Note that a while we assume a continuous scale for rating, a number of recommender systems use a discrete scale. However, our formalisation incorporates this case as a simple linear transformation can be performed on the scale to the [0,1] interval.

alization should be based on lift in business process based metrics [8]. Other than just monitoring the effectiveness of the knowledge currently deployed, an essential aspect of monitoring the effect of personalization is profile maintenance. User interests are dynamic and their evolution must be detected and adapted to for effective personalization to take place. Additionally, personalization itself can influence user behaviour. Techniques for identifying this change and adapting the personalization system to it are not well understood, requiring further research.

In terms of the learning task, personalization can be viewed as a

- Prediction Task: A model must be built to predict ratings for items not currently rated by the user. Depending on whether the user ratings are numeric or discrete, the learning task can be viewed as a being one of regression or classification.
- Selection Task: A model must be built that selects the N most relevant items for a user that the user has not already rated. While this task can be viewed as one of post processing the list of predictions for items generated by a prediction model, the method of evaluating a selection based personalization strategy would be very different from that of a prediction based strategy (see Section 7).

3 Classifications of Approaches to Personalization

In this section we discuss various dimensions along which personalization systems can be classified based on the data they utilize, the learning paradigm used, the location of the personalization and the process that the interaction takes with the user.

3.1 Individual Vs Collaborative

The term personalization impresses upon the individuality of users and the need for systems to adapt their interfaces to the needs of the user. This requires data collected on interactions of users with the system to be modelled in a user-centric fashion. Typically, data is collected by the business with which the user is interacting and hence the business has access to data associated with all its customers.

A personalization system may choose to build an individual model of user likes and dislikes and use this profile to predict/tailor future interactions with that user. This approach commonly requires content descriptions of items to be available and are often referred to as *content-based filtering systems*. NewsWeeder [9] is an example of such a system that automatically learns user profiles for netnews filtering. In the case of NewsWeeder the user provides active feedback by rating articles on a scale of 1 to 5. The process of building a profile for a user requires the transformation of each article into a bag or words representation, with each token being assigned a weight using some learning method such as *tfidf* [10] or minimum description length [11]. The profile is then used to recommend articles to the user.

An alternative approach to recommendation is to not only use the profile for the active user but also other users with similar preferences, referred to as the active user's neighbourhood, when recommending items. This approach is referred to as *social or collaborative filtering*. An example of such a system is GroupLens, also aimed at recommending netnews articles [12]. GroupLens defines a user profile as an n-dimensional vector, where n is the number of netnews articles. If an articles has been rated by the user, its corresponding element in the vector contains the rating. Note that as opposed to content-based filtering, the actual content descriptions of the articles is not part of the profile. Articles not currently rated by the active user but rated highly by users in the neighbourhood of the active user are candidates for recommendation to the active user. While GroupLens only uses rating data, collaborative approaches that utilise both content and user rating data have also been proposed [13], [14].

A major disadvantages of approaches based on an individual profile include the lack of serendipity as recommendations are very focused on the users previous interests. Also, the system depends on the availability of content descriptions of the items being recommended. On the other hand the advantage of this approach is that it can be implemented on the client side, resulting in reduced worries for the user regarding privacy and improved (multi-site) data collection for implicit user preference elicitation.

The collaborative approach also suffers from a number of disadvantages, not least the reliance on the availability of ratings for any item prior to it being recommendable, often referred to as the new item rating problem. Also, a new user needs to rate a number of items before he can start to obtain useful recommendations from the system, referred to as the new user problem. These issues along with others such as sparseness are discussed in greater detail in Section 6.

3.2 Reactive Vs Proactive

Reactive approaches view personalization as a conversational process that requires explicit interactions with the user either in the form of queries or feedback that is incorporated into the recommendation process, refining the search for the item of interest to the user. Most reactive systems for personalization have their origins in case-based reasoning research [15], [16], [17]. Reactive systems can be further classified based on the types of feedback they expect from the user. Common feedback mechanisms used by these systems include value elicitation, critiquing/tweaking [17], rating and preference feedback [18]. Value elicitation and tweaking/critiquing are feature based approaches to feedback. While in value elicitation the user must provide a rating for each feature of each recommendation object presented to the user, based on its suitability to the users needs, in tweaking/critiquing the user only provides directional feedback (for example, "too high", "too low") on feature values for the recommended object. Rating and preference are feedback approaches at the object level. In rating based feedback, the user must rate all the recommendations presented to him, based on their 'fit' with his requirements. In preference feedback the user is provided with a list of recommendations and is required to choose one of the recommendations that best suits his requirement. The system then uses this feedback to present the user with other, similar objects. The iterations continue until the user finds an object of interest or abandons the search. Examples of such recommender systems include Entree [19], DIETORECS [20] and ExpertClerk [21]. For a more detailed discussion on these feedback mechanisms see [16], [17].

Proactive approaches on the other hand learn user preferences and provide recommendations based on the learned information, not necessarily requiring the user to provide explicit feedback to the system to drive the current recommendation process. Proactive systems provide users with recommendations, which the user may choose to select or ignore. The users feedback is not central to the recommendation process as is the case in reactive systems. Examples of proactive systems include the recommendation engine at Amazon.com [22] and CDNOW, Web mining based systems such as [23], [24], [25], GroupLens [26], MovieLens [27] and Ringo [28].

3.3 User Vs Item Information

Personalization systems vary in the information they use to generate recommendations. Typically, the information utilized by these systems include:

- Item Related Information: This includes content descriptions of the items being recommended and a product/ domain ontology
- User Related Information: This includes past preference ratings and behaviour of the user, and user demographics

Systems that use item related information generally deal with unstructured data related to the items [29], [9]. Once this data has been processed, into relational form such as a bag-of-words representation commonly used for textual data, a user profile is generated. The profile itself may be individual as in the case of NewsWeeder [9] or based on group behaviour [13].

Most systems that use user related information, tend to be based on past user behaviour such as the items they have bought or rated (implicitly or explicitly) in the past. Fewer systems use demographic data within the recommendation process. This is due to the fact that such data is more difficult to collect on the Web and, when collected, tends to be of poor quality. Also, recommendations purely based on demographic data have been shown to be less accurate than those based on the item content and user behaviour [30]. In his study of recommender systems, Pazzani collected demographic data from the home pages of the users rather than adding the additional burden on the user to provide data specifically for the system. Such data collection outside of a controlled environment would be fraught with difficulties. In Lifestyle Finder [31], externally procured demographic data (Claritas's PRIZM) was used to enhance demographic attributes obtained from the user, through an iterative process where the system only requests information pertinent to classifying the user into one of 62 demographic clusters defined within the PRIZM classification. Once classified, objects most relevant to that demographic cluster are recommended to the user.

In addition to systems that depend solely on item related or user related information, a number of hybrid systems have been developed that use both types of information. Section 5.4 discusses these systems in greater detail. An example of such a system is the bibliographic system proposed by Haase et al. [5]. In addition to data on user behaviour, two domain ontologies are also available to the system describing the content of the items in a more structured form than that used by NewsWeeder. Hasse et al. define a user model based on user expertise, recent queries, recent relevant results (implicitly obtained by user actions on previous recommendations), a vector of weights for content features and a similarity threshold.

3.4 Memory Based Vs Model Based

As described in Section 2, the process of personalization consists of an offline and online stage. The key tasks during the offline stage are the collection and processing of data pertaining to user interests and the learning of a user profile from the data collected. Learning from data can be classified into memory based (also known as lazy) learning and model based (or eager) learning based on whether it generalizes beyond the training data when presented with a query instance (online) or prior to that (offline).

Traditional Collaborative filtering (see Section 5.2) and content based filtering based systems (see Section 5.1) that use lazy learning algorithms [32], [33] are examples of the memory-based approach to personalization, while item-based and other collaborative filtering approaches that learn models prior to deployment (see Section 5.3) are examples of model-based personalization systems.

As memory based systems simply memorise all the data and generalize from it at the point of generating recommendations, they are more susceptible to scalability issues. Section 6.3 discusses some of the solutions proposed in literature to address the scalability of memory based personalization systems. As the computationally expensive learning occurs offline for model-based systems, they generally tend to scale better than memory based systems during the online deployment stage. On the other hand, as more data is collected, memory based systems are generally better at adapting to changes in user interests compared to model based techniques that must either be incremental or be rebuilt to account for the new data.

Memory based systems generally represent a user profile using a vector representation though more expressive representations such as associative networks [34] and ontological profiles [35] have also been proposed.

3.5 Client Side Vs Server Side

Approaches to personalization can be classified based on whether these approaches have been developed to run on the client side or on the server-side. The key distinction between these personalization approaches is the breadth of data that are available to the personalization system. On the client side, data is only available about the individual user and hence the only approach possible on the client side is *Individual*.

On the server side, the business has the ability to collect data on all its visitors and hence both Individual and Collaborative approaches can be applied. On the other hand, server side approaches generally only have access to interactions of users with content on their Web site while client side approaches can access data on the individuals interactions with multiple Web sites.

Given these characteristics, most client side applications are aimed at personalized search applicable across multiple repositories [36], [37]. The lack of common domain ontologies across Web sites, unstructured nature of the Web and the sparseness of available behavioral data currently reduce the possibilities for personalization of navigational as opposed to search based interactions with the Web.

4 Data

Explicit data collection has typically been modelled as ratings of items, personal demographics and preference (including utility) data. Preference data refers to information that the user provides that can help the system discern which items would be useful to the user. When declared explicitly it can take the form of keywords/product categories (e.g. genres in movie/music databases) or values for certain attributes that describe the objects (e.g. cotton as the preferred material in an apparel store). Utility data refers to information regarding how the user would measure the fit of the objects recommended with his requirements. For example, if two suppliers for the same product exist, with supplier A providing the product at a premium rate over supplier B but with the advantage of free insurance for a predefined period, different users will have different thresholds for the extra cost of purchasing the product from supplier A [38], [39]. We refer to data that defines these preferences as utility data. Rating data may take the form of a discrete numeric value or an unstructured textual form such as reviews of products. While using numeric values is computationally easier to leverage, they are also less reliable as users associate these discrete values subjectively, for example, three stars according to one user may be equivalent to two stars for another user.

Implicit data collection refers to any data that can be collected on the user unobtrusively by "watching" their interaction with the system. Once again the objective is to obtain ratings from various discernable actions of the user. The actions and the associated inferences are dependent on the type of system being personalized. For example, in the Web domain in general, the linger time [2] is taken to be an implicit indicator of interest in the object [26]. Additionally, in an e-commerce context, actions such as adding an item to the basket, purchasing an item, deleting an item from the basket can all imply differing levels of interest in the item [40] as could bookmarking of pages [41], visit frequency, following/passing over a link and saving a page on a news/content site [42]. Claypool et al. [43] evaluated a number of possible implicit interest indicators and concluded that linger time and amount of scrolling can be useful indicators of interest. They also provided a useful categorization of interest indicators.

One issue with implicit data collection is that most observations are positive in nature and it is up to the system to use some heuristics to decide on what defines a negative observation. For example, the use of the back button after the user spends only a short time on a page can be inferred as being a negative observation or the choosing of a document from a list may render the other items in the list as being classified as not interesting [44], [45]. Even when certain negative actions are observed such as the deletion of an item from a shopping trolley, heuristics must be used to decide on how the initial interest in an item, i.e. inserting of the product in the shopping basket, must be amended when the item is deleted from the basket. Schwab et al. [46] propose a system that only employs positive feedback data to avoid the use of such heuristics. Hotle and Yan [47] showed that implicit negative feedback data can greatly improve the effectiveness of a conversational recommendation system, however, care must be taken in deciding what feedback can be attributed as being negative.

It is worth noting at this point that some of the implicit interest indicators used in these evaluations required data to be collected on the client side, while other data can be collected on the Web server, albeit with some inaccuracy, servicing the user request.

Explicit data input has a cost associated with it as it requires users to detract from their principle reason for interacting with the system and provide data, the benefits of which are intangible to the user. A number of studies carried out by the IBM User Interface Institute in the early 1980's confirm that, in general, users are motivated to get

[2] The time spent viewing an item and its associated content.

started with using a system and do not care about spending time up front on setting up the system, as is required by personalization systems that are dependent on explicit data being provided by the user. Carroll and Rosson [48] refer to this phenomenon as the "paradox of the active user" as users would save time in the long term by taking some initial time to optimize the system but that's not how people behave in the real world. While the studies were not aimed at personalization systems per se, the conclusion of the studies that engineers must not build products for an idealized rational user, rather they must design for the way users actually behave is just as valid for personalization systems. Studies in personalization show that without tangible benefits for the user, the user tends to read a lot more documents than they bother ranking [49]. By generating data that indicates a users interest in an object without the user needing to provide this information would result in more data and a reduction in sparsity, that exists especially in large information resources, typical of the Web. Additionally, privacy concerns also imply that users on the Internet tend to only provide accurate information that is deemed essential. Berendt and Teltzrow [50] suggest that users on the Internet exhibit varying degrees of privacy concerns and a large percentage of users would be happy to impart with various degrees of private information based on the perceived benefit to them in doing so. An interesting implication for designing personalization systems.

5 Personalization Techniques

In this section we describe the various approaches used for generating a personalized Web experience for a user.

5.1 Content-Based Filtering

Content based filtering systems have their roots in information retrieval. The approach to recommendation generation is based around the analysis of items previously rated by a user and generating a profile for a user based on the content descriptions of these items. The profile is then used to predict a rating for previously unseen items and those deemed as being potentially interesting are presented to the user. A number of the early recommender systems were based on content-based filtering including Personal WebWatcher [45], InfoFinder [51], NewsWeeder [9], Letizia [44] and Syskill and Webert [52]. Mladenic [53] provides a survey of the commonly used text-learning techniques in the context of content filtering, with particular focus on representation, feature selection and learning algorithms.

Syskill and Webert learns a profile from previously ranked Web pages on a particular topic to distinguish between interesting and non-interesting Web pages. To learn the profile, it uses the 128 most informative words, defined using expected information gain, from a page and trains a naïve Bayes classifier to predict future, unseen pages as potentially interesting or not for the user. The user may provide an initial profile for a topic, which in the case of Syskill and Webert, requires the definition of conditional probabilities for each word, given a page that is (not) interesting to the user. As pages get rated, these initial probabilities are updated, using conjugate priors [54], to reflect the rating of the pages by the user.

Rather than requiring the user of explicitly rate documents, Letizia uses implicit indicators of interest coupled with *tfidf* to compute content similarity between previosuly browsed interesting pages and candidate pages in the proximity of the users current browsing activity. To maximise the added value of the system, as opposed to the depth-first search carried out by most Web users, Letizia carries out a breadth first search of the hyperlinked documents, maintaining a list of documents that it believes to be relevant to the user.

Schwab et al. [46] propose the use of a naïve Bayes and nearest neighbor approach to content based filtering to build a user profile from implicit observations. In their approach they specifically abstain from using any heuristics for assigning certain observations as negative feedback, instead modifying the use of nearest neighbor and naïve Bayes to deal with only positive observations through the use of distance and probability thresholds. They also proposed a novel approach to feature selection based on the deviation of feature values for a specific user from the norm.

The main drawback of content-based filtering systems is their tendency to overspecialize the item selection as recommendations are solely based on the users previous rating of items, resulting in recommended items being very similar to previous items seen by the user. User studies have shown that users find online recommenders most useful when they recommend unexpected items [55], alluding to the fact that the overspecialization by content-based filtering systems is indeed a serious drawback. One approach to dealing with this problem is to inject some form of diversity within the recommendation set (see Section 6.5).

5.2 Traditional Collaborative Filtering

Goldberg et al. [56] first introduced collaborative filtering as an alternative to content based filtering of a stream of electronic documents. The basic idea as presented by Goldberg et al. was that people collaborate to help each other perform filtering by recording their reactions to e-mails in the form of annotations.

The application of this technology for recommending products has gained popularity and commercial success [57]. In a recommendation context, collaborative filtering works as described below.

Users provide feedback on the items that they consume, in the form of ratings. To recommend items to the active user, u_a, previous feedback is used to find other likeminded users (referred to as the user's neighbourhood). These are users that have provided similar feedback to a large number of the items that have been consumed by u_a. Items that have been consumed by likeminded users but not by the current user are candidates for recommendation. The assumption made by these systems is that users that have had common interests in the past, defined by feedback on items consumed, will have similar tastes in the future.

The rating data that is input to a collaborative filtering system is often referred to as a ratings matrix where each column is associated with an item in I, and each row contains the ratings of the items by an individual user.

To achieve its goal of providing useful recommendations, a collaborative filtering system must provide algorithms for achieving the following:

– a metric for measuring similarity between users, for neighbourhood formation
– a method for selecting a subset of the neighbourhood for prediction
– a method for predicting a rating for items not currently rated by the active user

A number of metrics have been proposed for measuring the similarity between users including Pearson and Spearman Correlation [12], the cosine angle distance [58], Entropy, Mean-squared difference and constrained Pearson correlation [28]. The most commonly used metric is the cosine angle which has been shown to produce the best performance. It is calculated as the normalized dot product of user vectors:

$$sim(u_a, u_b) = \frac{u_a \cdot u_b}{\|u_a\|^2 . \|u_b\|^2}$$

Once the similarity of the active user with all other users has been computed, a method is required to calculate the ratings for each item $i_j \in I_a^{(u)}$. The most commonly used approach is to use the weighted sum of rank

$$r_{u_a}(i_j) = \overline{r}_{u_a} + \frac{\sum_{u_k \in U_j} sim(u_a, u_k) \times (r_{u_k}(i_j) - \overline{r}_{u_k})}{\sum_{u_k \in U_j} sim(u_a, u_k)}$$

where $U_j = \{u_k \mid u_k \in U \wedge u_k(i_j) \neq \perp\}$ and $\overline{r}_{u_a}$ and $\overline{r}_{u_k}$ are the average ratings for users u_a and u_k respectively.

As the number of users and items increases, this approach becomes infeasible. Other than performance considerations, there is also a case to be made for reducing the size of the neighborhood with respect to the accuracy of the recommendations [59] as with a majority of neighbors not similar to the current user, the noise generated by their ratings can reduce the accuracy of the recommendations. Hence a method is required to select a subset of users, defining the neighborhood of the current user. Only users in the active users neighbourhood are then used to predict item ratings. Two approaches have been used in literature to select the neighborhood. One is based on a threshold on the similarity value [28] and the other uses a threshold on the number of neighbors, irrespective of the similarity value, which is traditionally used by the k-nearest neighbor approach to lazy learning. One of the problems with using a threshold on similarity is that as the number of items increases, the sparsity of the active user's neighbourhood increases, reducing the coverage of the recommender system. On the other hand, when using a fixed number of neighbours, the accuracy of the predictions will be low for users that have more unique preferences.

A number of variants have been proposed to the basic collaborative filtering process described above. First, Herlocker [60] proposed the use of a significance weighting that incorporated a measure of how dependable the measure of similarity between two users is. The idea behind this weight was the fact that, in traditional collaborative filtering, two users would be considered equally similar whether they had two items rated in common or whether it was fifty. Intuitively, this would seem strange as in the first case we are basing the similarity measurement on a very small amount of data. Empirical evaluation carried out by Herlocker et al. suggested that neighbors based on these small samples were bad predictors of the interests of the active user. As a result, they proposed a significance measure that associated a weight in the unit interval to each user, based on how many items were involved in the similarity calculation. Both, the similarity metric and the significance weight were used when generating the active user's neighbourhood.

Secondly, traditional collaborative filtering gives an equal importance to all items within the similarity calculation. Noting that not all items are equally informative, Herlocker et al. [60] proposed the introduction of a variance weighting to take into account the variability of values within a single column of the ratings matrix. A low variance would suggest that most users have a similar rating for the item and as such the item is less effective in discriminating between users and should therefore have little effect of the similarity calculation between two users. Breese et al. proposed the use of inverse user frequency where items less frequently rated were given a lower weight [59]. They also proposed case amplification that heightened the weight associated with those users that had a similarity, to the active user, close to 1.

Finally, to deal with the fact that ratings are inherently subjective and users tend to have different distributions underlying their item ratings, normalization of ratings provided by each user was proposed by Resnick et al. [12]. Rankings were scaled based on their deviations from the mean rating for the user. An alternative method for performing the scaling of ratings is to compute z-scores to also take into account the differences in spread of the ratings [60].

While collaborative filtering is commercially the most successful approach to recommendation generation, it suffers from a number of well known problems including the cold start/latency problem (see Section 6.1) and sparseness within the rating matrix (see Section 6.2). Traditional collaborative filtering also suffers from scalability issues (see Section 6.3). More recently, malicious attacks on recommender systems [61] (see Section 6.9) have been shown to affect traditional user-based collaborative filtering to a greater extend than model based approaches such as item-based collaborative filtering.

5.3 Model Based Techniques

Model based collaborative filtering techniques use a two stage process for recommendation generation. The first stage is carried out offline, where user behavioral data collected during previous interactions is mined and an explicit model generated for use in future online interactions. The second stage, is carried out in real-time as a new visitor begins an interaction with the Web site. Data from the current user session is scored using the models generated offline, and recommendations generated based on this scoring. The application of these models are generally computationally inexpensive compared to memory-based approaches such as traditional collaborative filtering, aiding scalability of the real-time component of the recommender system.

Model generation can be applied to explicitly and implicitly obtained user behavioural data. While the most commonly used implicit data is Web usage data, data pertaining to the structure and content are also often used.

A number of data mining algorithms have been used for offline model building including Clustering, Classification, Association Rule Discovery, Sequence Rule Discovery and Markov Models. In this section we briefly describe these approaches.

Item-Based Collaborative Filtering. In item-based collaborative filtering the offline, model building, process builds an *item similarity matrix*. The item similarity matrix, IS, is an $n \times n$ matrix where $IS[j, t]$ is the similarity of items i_j and i_t. Rather than basing item similarity on content descriptions of the items, similarity between items is based on user ratings of these items, hence each item is represented by an m dimensional vector, and the similarity computed using metrics such as (adjusted-) cosine similarity and correlation-based similarity [62]. The recommendation process predicts the rating for items not previously rated by the user by computing a weighted sum of the ratings of items in the item neighbourhood of the target item, consisting of only those items that have been previously rated by the user.

The model itself can be rather large, being in $O(n^2)$. An alternative is to store only the similarity values for the k most similar items. k is referred to as the model size. Clearly as k becomes small, the coverage as well as accuracy of the model will reduce.

Evaluation of the item-based collaborative filtering approach [62] showed that item-based collaborative filtering approaches provide better quality recommendations than the user based approach for rating prediction.

Clustering Based Approaches. Two main approaches to clustering for collaborative filtering have been proposed. These are item-based and user-based clustering. In user-based clustering, users are clustered based on the similarity of their ratings of items. In item based clustering, items are clustered based on the similarity of ratings by all users in U. In the case of user-based clustering, each cluster centre $C_k^{(U)}$ is represented by an n-dimensional vector, $C_k^{(U)} = (ar_1, ar_2, ..., ar_n)$, where each ar_j is the average item rating for (or average weight associated with) item i_j by users in cluster k. In the case of item-based clustering the cluster centre is represented by an m-dimensional vector $C_k^{(I)} = (q_1, q_2, ..., q_m)$, where each q_i is the average ratings by user, u_i of items within the cluster.

In the case of Web usage or transaction data a number of other factors can also be considered in determining the item weights within each profile, and in determining the recommendation scores. These additional factors may include the link distance of pages to the current user location within the site or the rank of the profile in terms of its significance.

The recommendation engine can compute the similarity of an active user's profile with each of the discovered user models represented by cluster centroids. The top matching centroid is used to produce a recommendation set in a manner similar to that used in user-based collaborative filtering.

Various clustering algorithms have been used, including partitioning algorithms such as, K-means for item and user-based clustering [63], ROCK [64] for item-based clustering, agglomerative hierarchical clustering [64] for item-based clustering, divisive hierarchical clustering for user-based and item-based clustering [65], mixture resolving algorithms such as EM [66] to cluster users based on their item ratings [59] and Gibbs Sampling [59].

Motivated by reducing the sparseness of the rating matrix, O'Connor and Herlocker proposed the use of item clustering as a means for reducing the dimensionality of the rating matrix [64]. Column vectors from the ratings matrix were clustered based on their similarity, measured using Person's correlation coefficient, in user ratings. The clustering resulted in the partitioning of the universe of items and each partition was treated as a separate, smaller ratings matrix. Predictions were then made by using traditional collaborative filtering algorithms independently on each of the ratings matrices.

Kohr and Merialdo proposed the use of top-down hierarchical clustering to cluster users and items. Clustering results in two cluster hierarchies, one based on the item ratings by users and the other based on the user ratings of items [65]. For the active user, the predicted rating for an item is generated using a weighted average of cluster centre coordinates for all clusters from the root cluster to appropriate leaf node of each of the two hierarchies. The weights are based on the intra-cluster similarity of each of the clusters.

Association and Sequence Rule Based Approaches. Association and Sequence rule discovery [67], [68] techniques were initially developed as techniques for mining supermarket basket data but have since been used in various domains including Web mining [69]. The key difference between these algorithms is that while association rule discovery algorithms do not take into account the order in which items have been accessed, sequential pattern discovery algorithms do consider the order when discovering frequently occurring itemsets. Hence, given a user transaction $\{i_1, i_2, i_3\}$, the transaction supports the association rules $i_1 \Rightarrow i_2$ and $i_2 \Rightarrow i_1$ but not the sequential pattern $i_2 \Rightarrow i_1$.

The discovery of association rules from transaction data consists of two main parts: the discovery of frequent itemsets [3] and the discovery of association rules from these frequent itemsets which satisfy a minimum *confidence* threshold.

Given a set of transactions T and a set $I = \{I_1, I_2, \ldots, I_k\}$ of itemsets over T. The $support$ of an itemset $I_i \in I$ is defined as

$$\sigma(I_i) = \frac{|\{t \in T : I_i \subseteq t\}|}{|T|}$$

An association rule, r, is an expression of the form $X \Rightarrow Y\ (\sigma_r, \alpha_r)$, where X and Y are itemsets, $\sigma_r = \sigma(X \cup Y)$ is the support of $X \cup Y$ representing the probability that X and Y occur together in a transaction. The confidence for the rule r, α_r, is given by $\sigma(X \cup Y)/\sigma(X)$ and represents the conditional probability that Y occurs in a transaction given that X has occurred in that transaction.

Additional metrics have been proposed in literature that aim to quantify the interestingness of a rule [70], [71], [72] however we limit our discussion here to support and confidence as these are the most commonly used metrics when using association and sequence based approaches to recommendation generation.

The discovery of association rules in Web transaction data has many advantages. For example, a high-confidence rule such as $\{$special-offers/, /products/software/$\} \Rightarrow$ $\{$shopping-cart/$\}$ might provide some indication that a promotional campaign on software products is positively affecting online sales. Such rules can also be used to optimize the structure of the site. For example, if a site does not provide direct linkage between two pages A and B, the discovery of a rule $\{$A$\} \Rightarrow \{$B$\}$ would indicate that providing a direct hyperlink might aid users in finding the intended information.

The result of association rule mining can be used in order to produce a model for recommendation or personalization systems [73,74,75,76]. The *top-N* recommender

[3] Itemsets which satisfy a minimum *support* threshold.

systems proposed in [76] uses association rules for making recommendations. First all association rules are discovered from purchase data. Customer's historical purchase information is then matched against the left-hand-side of the rule in order to find all rules supported by a customer. All right-hand side items from the supported rules are sorted by confidence and the first N highest ranked items are selected as the recommendation set.

One problem for association rule recommendation systems is that a system cannot give any recommendations when the dataset is sparse. In [73] two potential solutions to this problem were proposed. The first solution is to rank all discovered rules calculated by the degree of intersection between the left-hand-side of rule and a user's active session and then to generate the top k recommendations. The second solution is to utilize collaborative filtering: the system finds "close neighbors" who have similar interest to a target user and makes recommendations based on the close neighbor's history. In [74] a collaborative recommendation system was presented using association rules. The proposed mining algorithm finds an appropriate number of rules for each target user by automatically selecting the minimum support. The recommendation engine generates association rules for each user, among both users and items. If a user minimum support is greater than a threshold, the system generates recommendations based on user association, else it uses item association.

In [75] a scalable framework for recommender systems using association rule mining was proposed. The proposed recommendation algorithm uses an efficient data structure for storing frequent itemsets, and produces recommendations in real-time, without the need to generate all association rules from frequent itemsets. In this framework, the recommendation engine based on association rules matches the current user session window with frequent itemsets to find candidate pageviews for giving recommendations. Given an active session window w and a group of frequent itemsets, we only consider all the frequent itemsets of size $|w| + 1$ containing the current session window. The recommendation value of each candidate pageview is based on the confidence of the corresponding association rule whose consequent is the singleton containing the pageview to be recommended. In order to facilitate the search for itemsets (of size $|w| + 1$) containing the current session window w, the frequent itemsets are stored in a directed acyclic graph, called a *Frequent Itemset Graph*. The Frequent Itemset Graph is an extension of the lexicographic tree used in the "tree projection algorithm" [77]. The graph is organized into levels from 0 to k, where k is the maximum size among all frequent itemsets. Given an active user session window w, sorted in lexicographic order, a depth-first search of the Frequent Itemset Graph is performed to level $|w|$. If a match is found, then the children of the matching node n containing w are used to generate candidate recommendations.

When discovering sequential patterns from Web logs, two types of sequences are identified: Contiguous or Closed Sequences and Open Sequences [69]. Contiguous sequences require that items appearing in a sequence rule appear contiguously in transactions that support the sequence. Hence the contiguous sequence pattern $i_1, i_2 \Rightarrow i_3$ is satisfied by the transaction $\{i_1, i_2, i_3\}$ but not by the transaction $\{i_1, i_2, i_4, i_3\}$, as i_4 appears in the transaction between the items appearing in the sequence pattern. On the other hand, both transactions support the rule if it were an open sequence rule.

Given a transaction set T and a set $S = \{S_1, S_2, \ldots, S_n\}$ of frequent (contiguous) sequential patterns over T, the support of each S_i is defined as follows:

$$\sigma(S_i) = \frac{|\{t \in T : S_i \text{ is (contiguous) subsequence of } t\}|}{|T|}$$

The confidence of the rule $X \Rightarrow Y$, where X and Y are (contiguous) sequential patterns, is defined as

$$\alpha(X \Rightarrow Y) = \frac{\sigma(X \circ Y)}{\sigma(X)},$$

where $\circ$ denotes the concatenation operator. The Apriori algorithm used in association rule mining can also be adopted to discover sequential and contiguous sequential patterns. This is normally accomplished by changing the definition of support to be based on the frequency of occurrences of subsequences of items rather than subsets of items [78].

To aid performance of the recommendation process, sequential patterns are typically stored in the form of a single trie structure with each node representing an item and the root representing the empty sequence. Recommendation generation can be achieved in $O(s)$ by traversing the tree, where s is the length of the current user transaction deemed to be useful in recommending the next set of items. Mobasher et al. [79] use a fixed size sliding window, of size m, over the current transaction for recommendation generation. Hence the maximum depth of the tree required to be generated is m+1. The size of the trees generated during the offline mining can be controlled by setting different minimum support and confidence thresholds.

An empirical evaluation of association and sequential pattern based recommendation showed that site characteristics such as site topology and degree of connectivity can have a significant impact on the usefulness of sequential patterns over non-sequential (association) patterns [80]. Additionally, it has also been shown that contiguous sequential patterns are particularly restrictive and hence are more valuable in page prefetching applications rather than in recommendation generation [79].

A technique related to the use of sequential rules is that of modeling Web interactions as Markov Chain models. A Markov model is represented by the 3-tuple $\langle A, S, T\rangle$ where A is a set of possible actions, S is the set of all possible states for which the model is built and T is the Transition Probability Matrix that stores the probability of performing an action $a \in A$ when the process is in a state $s \in S$. In the context of recommendation systems, A is the set of items and S is the visitor's navigation history, defined as a k-tuple of items visited, where k is referred to as the order of the Markov model. As the order of the Markov model increases, so does the size of the state space, S. On the other hand the coverage of that space, based on previous history, reduces, leading to an inaccurate transition probability matrix. To counter the reduction in coverage, various Markov models of differing order can be trained and used to make predictions. The resulting model is referred to as the All-Kth-Order Markov model [81]. The downside of using the All-Kth-Order Markov model is the large number of states. Also, the issue regarding the accuracy of transition probabilities especially for the higher order Markov models is not addressed. Selective Markov models that only store some of the states within the model have been proposed as a solution to this problem [82]. A post pruning approach is used to prune out states that cannot be expected to be accurate predictors. Three pruning approaches based on the support, confidence and estimated error were proposed.

Rather than pruning states as a post process, sequence rule discovery and association rule discovery algorithms actively prune the state space during the discovery process using support. A further post pruning, based on confidence of the discovered rules, is also carried out. Hence the Selective Markov model is analogous to sequence rule discovery algorithms. Note however that the actual pruning process based on confidence proposed by Deshpande and Karypis [82] is not the same as that carried out during sequence rule discovery. Evaluation of Selective Markov models showed that up to 90% of states can be pruned without a reduction in accuracy. In fact some improvements in model accuracy resulted from pruning.

Graph Theoretic Approaches. Aggarwal et al. proposed a graph theoretic approach to collaborative filtering in which ratings data is transformed into a directed graph, nodes representing users and edges representing the predictability of a user based on the ratings of another user [83]. A directed edge exists from user u_i to u_j if user u_j predicts user u_i. To predict if a particular item, i_k, will be of interest to user u_i, assuming i_k has not been rated by the user, the shortest path from u_i is calculated to any user, say u_r, who has rated i_k and a predicted rating for i_k by u_i is generated as a function of the path from u_i to u_r.

Mirza et al. provide a framework for studying recommendation algorithms by graph analysis [84]. In their framework, ratings data is represented as a bipartite graph $G = \langle U \cup I, E \rangle$ with nodes representing either users or items, while edges represent ratings of items by users. A social network is constructed using the concept of a jump which is defined as a mapping from the ratings data to a subset of $U \times U$. Mirza et al. define a number of different types of jump, the simplest being a skip, results in an edge between two users if there exists at least one item that both of them have rated. In general, different social networks emerge based on the definition of the jump used. Mirza describes a number of ways in which jumps can be defined [85]. One such jump that mirrors traditional collaborative approaches to recommendation is the hammock jump, which requires a user defined parameter, w, known as the *hammock width*. For an edge to exist between two users u_k and u_l within the resulting social network, the hammock width must be less than or equal to $\mid I_k^{(r)} \cap I_l^{(r)} \mid$. The skip is, therefore, a special case of the hammock jump with hammock width 1. A third graph, called a recommender graph is then defined as a bipartite directed graph $G_R = \langle U \cup I, E_R \rangle$, with nodes $i_k \in I$ restricted to having only incoming edges. The shortest path from a user, u_i to an item in the graph can then be used to provide the basis for recommendations.

5.4 Hybrid Techniques

Other than the approaches discussed above, a number of hybrid approaches to personalization have also been proposed. These hybrid recommenders have been motivated by the observation that each of the recommendation technologies developed in the past have certain deficiencies that are difficult to overcome within the confines of a single recommendation approach. For example, the inability of collaborative filtering approaches to recommend new items items can be solved by coupling it with a content based recommendation approach. Not surprisingly, the most common form of hybrid recommender combines content based and collaborative filtering. An example of such a system is Fab [14], a recommendation system for Web content. Fab consists of a number of collection and selection agents. Collection agents are responsible for gathering pages pertaining to a small set of topics of interest of users. As the topics are based on user interests these may evolve with time to reflect the changing interests of the system's users. The selection agents select a set of pages for specific users out of the overall set of pages collected by the collection agents. The user rates each page presented to him by the selection agent. Each user has its own selection agent that contains a profile based on keywords contained in pages that have been previously rated by the user. Ratings for individual pages are also passed back to the original collection agents that can refine their own collection profile. Note that the collection agents profile is based on ratings from various users as opposed to just one user as is the case for the selection agent. The collaborative component of the system is based on the definition of a neighbourhood for each user within which pages rated highly are shared.

Another form of hybrid recommender that has recently been gaining a lot of attention is that which combines item ratings with domain ontologies (see Section 6.7).

More generically, Pazzani showed that combining various recommendations generated using different information sources such as user demographics, item content and user ratings (collaboratively) increases the precision of the recommendations [30].

Based on their study on the impact of site characteristics on the usefulness of sequential patterns over non-sequential (association) patterns [80], Nakagawa and Mobasher [86] proposed a hybrid recommendation system that switched between different recommendation systems based on the degree of connectivity of the site and the current location of the user within the site. Evaluation of this approach revealed that the hybrid model outperformed the base recommendation models in both precision and coverage.

Burke provides a comprehensive analysis of approaches to generating hybrid recommendation engines [87].

6 Issues

The study of recommendation systems over the last decade have brought to light a number of issues that must be addressed if these systems are to find acceptance within the wider context of personalized information access. In this section we discuss these issues. Along with a description of the issue we also discuss solutions that have been proposed to date to resolve them.

6.1 The Cold Start and Latency Problem

Personalization systems expect to have some information available on the individual users so that they can leverage this information to present items of interest to the user in future interactions. Hence, a new user with no interaction history poses a problem to the system as it is unable to personalize its interactions with the user. This is often referred to as the *new user problem*. The lack of useful interactions may put the user off the system before the system is able to gather the data it requires to start personalizing its interactions with the user.

A similar issue is posed by the introduction of a new item. When a new item becomes available, the lack of rating data means that systems that depend on item ratings solely (for example, collaborative filtering based approaches) cannot recommend the new item before a considerable history of has been collected. This is often referred to as the *New Item or Latency* problem. A collaborative filtering system provides no value to the first user in a neighborhood to rate an item. This need for altruistic behavior can further delay the introduction of a new item into the recommendation process [49].

The new user problem is even more acute at the point of time when a collaborative system is initially installed as not only is rating data not available for a single user but there is no rating data for any users of the system which is referred to a the *Cold Start* problem.

An approach often used to alleviate the new user/ item problem has been to use hybrid recommendation techniques, typically those that combine collaborative techniques with content based filtering techniques [88], or those based on demographic profiling [31].

Massa and Avesani propose the incorporation of a Web of trust within the recommendation process and show that using this additional information can be very effective in addressing the new user problem [89]. However, this does assume the existence of a Web of trust which in itself may not be available.

Middleton et al. [4] propose the use of an external ontology as seed knowledge for a recommender system as a solution to the cold start problem. The Quickstep recommender system developed by Middleton et al. aims to provide academics with recommendations of papers of interest. Feedback from the academics is incorporated into an ontology based user profile. To avoid the cold start problem, Quickstep uses information from the research publication and personnel database of the academic institution to populate an initial profile for the user. This approach obviously assumes the availability of an external ontology that may not always be available.

Haase et al. [5] approach the cold start problem by reusing the properties of a peer-to-peer network using profiles of similar peers in the semantic neighborhood to initialize the profile of a new peer.

6.2 Data Sparseness

Sparsity refers to the fact that as the number of items increases, even the most prolific users of the system will only explicitly or implicitly rate a very small percentage of all items. As a result, there will be many pairs of customers that have no item ratings in common and even those that do will not have a large number of common ratings. The nearest neighbor computation resulting from this fact will not be accurate and hence a low rating for an item would not imply that similar items will not be recommended [90]. To counter the effect of an increasing number of items, for collaborative filtering to provide accurate predictions, the number of users required to rate a sizeable number of items will be much higher than that required when the number of items is small.

Sarwar et al. [49] evaluated the benefit of using simple information filtering bots on Usenet news to generate ratings for new items published. The bots generated ratings for items based on the correctness of spellings, length of article and length of the included message. The value of these bots was evaluated in various Usenet news groups. The filter bots are treated in similar manner to ordinary users and hence their ratings are only used when these filterbots are in the neighbourhood of a current user. Good et al. [27] extended this research by using a number of information filtering agents in the domain of movie recommendation that used genre, cast and keywords for generating ratings. Some of these bots included a learning component for example, a bot that used inductive logic programming [91] to learn a model for predicting ratings based on genre and keywords. Good et al. also suggested a number of ways in which ratings from individual bots could be combined with user ratings to generate rating predictions.

Motivated by the observation that as the number and diversity of items increases, it is less likely that a user's rating of an item will be affected by all other item ratings, for recommending Usenet news articles, Resnick et al. [12] showed that creating separate item partitions for each discussion group can improve performance of the recommender system. However, such a process is by its very nature not transferable to other domains, requiring a domain specific partitioning scheme to be devised for every new domain that the technique is applied to. O'Connor and Herlocker [64] investigated the use of item clustering to discover groups of items that show similar ratings behavior from users. Pearson correlation coefficient was used to compute the similarity between items. That is, two items were deemed as being similar if there was a strong correlation between the ratings of these items by users in general. Evaluation of this approach using MovieLens data however showed that while partitioning based on item clustering provides more accurate recommendations than random partitioning, genre based partitioning outperformed all of the item ratings based clustering approaches.

Goldberg et al. [92] proposed the use of a *gauge set* of items. This is a set of items that all users of the system must rate to seed the system. The gauge set provides the basis for a more accurate measurement of similarity between users as it would consist of a dense rating submatrix.

6.3 Scalability

Memory based approaches such as traditional collaborative filtering suffer from scalability issues as the number of users increases as well as an increasing number of candidate items.

A number of solutions have been proposed to deal with an increasing user base. The most widely used approach is to use a model-based approach to collaborative filtering rather than one that is memory based. An alternative is to limit the number of users that must be compared when making predictions for the active user. This can be achieved by either limiting the number of profiles stored (instance selection) or by indexing the user base and searching only a part of the whole user base for an active user (instance indexing).

Yu et al. [93] proposes an metric for use in instance selection, based on the information theoretic measure of mutual information, called relevance. The rationale behind instance selection is that, for a given target item, the rating of other items by a user should provide enough information to support the rating by the user of the target item. If this is not the case, then the user would probably not provide a useful basis for predicting the rank of the target item for the target user. Hence relevance of an instance (user) for predicting the rank, r, of a particular item, i, is calculated as $r(u, i) = \sum_{j \in J} I(V_i; V_j)$, where J is the set of all items other than j, rated by the user, u and $I(.,.)$ is the mutual information measure. Using the relevance metric, only the top N user instances are used for predicting the rating for the target item.

Chee et al. [94] propose the use of k-means to iteratively partition the rating matrix based on the rating similairty of users. The leaf nodes of the resulting binary tree consist of "cliques" of users with similar tastes. During prediction, the tree is navigated and similarity evaluated only within the clique of the active user.

Dealing with the issue of scalability with respect to the number of items is akin to feature subset selection and dimensionality reduction in machine learning. The most commonly used approach to dimensionality reduction applied to recommender systems have been singular value decomposition [95], [58] and principal component analysis [92]. Not only has singular value decomposition been shown to effectively reduce the dimensionality of the ratings matrix, but it has also been shown to improve accuracy of the recommendations when applied to less sparse ratings matrices through reduction in noise within the rating matrix. Approaches to incrementally build models based on singular value decomposition have also been investigated so as to avoid the expensive rebuilding of the model as new data becomes available [96], [97].

Tang et al. propose a the use of heuristics to limit the number of items considered [98]. For the movie recommendation domain they suggest using the temporal feature of items (year of release of a movie) to limit the set of candidate movies for recommendation.

6.4 Privacy

Currently U.S. laws impose little restrictions on private parties communicating information about people, leaving it up to the parties involved to define the extent of any such communication through a contract [99]. In particular, an online business may provide their customers with a privacy policy that would outline under what conditions, if at all, the business would share the information they hold about the customer. Breech of such a contract entitles the customer to bring a law suit on the business but not on any third party that has gained access to data as a consequence of the breech. In particular it is common place for a business suffering bankruptcy to sell the data they hold on their customers. Such a sale is currently supported by the law in the U.S [100].

Even if a business does not explicitly sell customer data, services such as collaborative filtering based recommender systems can be exploited to gain insights into individual customers preferences [101]. This is particularly true of users who rate products across different domains, referred to as straddlers. While such users are particularly desirable to enable collaborative systems to generate serendipitous recommendations, it also means that a user, who is obviously aware of their own preferences, or indeed an individual masquerading as a user with a certain set of preferences, could potentially gain insights into straddlers. Using a graph-theoretic representation for recommender systems, Ramakrishnan et al. [101] provide an analysis of the effect of two recommender system parameters, the hammock width and hammock path length, on the risk to straddlers. Their study concluded that a hammock width just below the value that splits the graph into a set of disconnected components carries the greatest risks for straddlers.

6.5 Recommendation List Diversity

While most research into recommending items has concentrated on the accuracy of predicted ratings, other factors have been identified as being important to users. One such factor is the diversity of items in the recommendation list. In a user survey aimed at evaluating the effect of diversification on user satisfaction, applied to item-based and user-based collaborative filtering, found that it had a positive effect on overall satisfaction even though accuracy of the recommendations was affected adversely [102]. The study further concluded that introducing diversity affects user satisfaction to a greater extent when item-based collaborative filtering is used, while it has no measurable affect on user-based collaborative filtering.

Smyth and McClave [103] proposed three approaches to introducing diversity into recommendation sets. The basic approach is to balance similarity of an item to the target with the diversity of the current items within the recommendation set. Diversity was measured as the average distance between the candidate recommendation and all items currently with the recommendation set. Ziegler proposed an approach to diversity maintenance [102] similar to the bounded greedy selection approach proposed by Smyth and McClave.

Sheth [104] modelled the information filtering task as a population of profiles, per user, that evolve using genetic operators of crossover and mutation. The profiles are generated using standard text mining functions such as *tfidf* on documents presented to the user by the profile and the relevance feedback received. While the crossover operator exploits the fitness of the current population of profiles, mutation is used to introduce some diversity into the population. Unlike other content based filtering approaches, as the profiles evolve through the use of the genetic operators, it is more likely that a level of serendipity can be maintained within the recommendation set.

6.6 Adapting to User Context

Personalization aims to “hide” the rigidity of the Internet by providing useful, contextually relevant information and services to the user. However, context as a concept has rarely been incorporated into personalization research. One of the reasons for this is that it is hard to arrive at a consensus of what defines context let alone modeling the concept. Lieberman and Selker provide a useful starting point for defining context, defining it as “everything that affects the computation except the explicit input and output” [105]. Unfortunately, this definition in itself does not make the modeling of context possible as we cannot consider all previous user interactions with a system as context for the current interaction and nor can we explicitly measure context, hence we must use current behavior to discover the user context and then use this context to predict the current behavior of the user so as to better service his requirements. If we assume that user behavior is predictable based on past interactions, we now must select only those previous interactions that were undertaken within the same context and use them to predict the needs of the user.

Contextual retrieval is also viewed as an important challenge in the information retrieval community [106]. Parent et al. [36] proposed a client-side Web agent that allows the user to interact with a concept classification hierarchy to define the context of the query terms provided. The agent uses portions of the hierarchy to expand the initial search query, effectively adding ‘user intent’ to the query. Sieg et al. [6] define context by the portions of the concept hierarchy (such as the Yahoo Directory) that match the user query. Each node of the concept hierarchy has a vector representation based on the documents contained in the node and all its subcategories. Previously accessed documents are clustered (an offline process) and the cluster centres form the user’s profile. When a query is issued, all clusters from the user profile that have a similarity with the query above a pre-defined threshold are selected. The query is matched against the concept hierarchy and a subset of nodes are chosen from the concept hierarchy that have a certain amount of similarity to the query. The selected clusters are then used to further refine the selection. In [35], user context is captured via nodes in a concept lattice induced from the original ontology and is updated incrementally based on the user’s interactions with the concepts of the ontology. Updates are initiated through the user selecting or deselecting concepts within the lattice that were considered to be of interest by the system based on the user’s long-term and short term memories. The context is represented as a pair of term vectors, one for the selected concepts and the other representing the deselected concepts.

6.7 Using Domain Knowledge

Dai and Mobasher [107] provide a framework for integrating domain knowledge with Web usage mining for user based collaborative filtering. They highlight that semantics can be integrated at different stages of the knowledge discovery process.

Mobasher et al. proposed the use of semantic knowledge about items to enhance item-based collaborative filtering [90]. Their approach is to represent the semantic knowledge about an item as a feature vector and calculate the similarity based on this information to other items. This item-similarity is then combined with rating similarity to get an overall measure of item similarity which is used to predict the rating by a user of a currently unrated item.

Cho and Kim [108] apply a product taxonomy with Web usage mining to reduce the dimensionality of the rating database when searching for nearest neighbours while Niu, Yan et al. [109] build customer profiles based on product hierarchy in order to learn customer preferences.

Middleton et al. use an ontological profile for a user within their research paper recommendation system, QuickStep [4]. The profile is based on a topic hierarchy alone. They also attempt to use externally available ontologies based on personnel records and user publications to address the cold-start problem for their recommendations system. The existence of such additional knowledge, while applicable in their specific application domain, cannot however be assumed in a general e-tailer scenario.

Haase et al. create semantic user profiles from usage and content information to provide personalized access to bibliographic information on a Peer-to-Peer bibliographic network [5]. The semantic user profile consists of the expertise, recent queries, recent relevant instances and a set of weights for the similarity function.

Ghani and Fano [29] proposed a recommender system based on a custom-built knowledge base of product semantics. The focus within the paper is on generating ”soft” attributes from online marketing text, describing the products browsed, and using them to generate cross category recommendations.

6.8 Managing the Dynamics in User Interests

Most personalization systems tend to use a static profile of the user. However user interests are not static, changing with time and context. Few systems have attempted to handle the dynamics within the user profile.

In NewDude [110] the user model consists of a short term interests and a long term interests model. The short term interests model is based on the n most recently rated stories. Each item (story) is represented as a term vector using *tfidf*. The similarity of the target item to items within the short term interest profile is computed using cosine similarity. If the similarity of the target item to another story in the short term interest profile is greater than a threshold value, it is deemed as being a known story and is therefore discarded. Alternatively, stories from the short term interest profile that have a similarity value greater than a threshold value are deemed to be in the neighbourhood of the target item and are used to predict a rating for the target item. If the target is deemed to be of interest to the user, it is recommended, alternatively it is discarded. If the neighbourhood of the target item within the short term interest profile is empty, the long term interest profile is used to classify the target item. The long term memory is based on the 150 most informative words appearing in the items and the model is based on the multinomial formulation of the naïve Bayes [111].

Rather than use a fixed number of most recent user interactions, Koychev and Schwab suggest the use of a continuous weighting function that associates a higher weight to more recent interactions with a user [112]. Tests using a linear weighting function showed some improvements in predictive accuracy.

An alternative approach is based on the evolution of a population of profiles per user [113], [104]. As interests of users change, profiles that better reflect their current interests become more prominent within the population. Moukas and Zacharia separate out the two roles of information filtering and discovery and describe a market-based control scheme to control the fitness of information and discovery agents.

6.9 Robustness

The dependence of personalization systems on item ratings provided by users and their use of these ratings in generating social recommendations also opens them to abuse. For example, an interested party may decide to influence item recommendations by inserting false ratings for a subset of items that they have an interest in. Attacks of this nature are referred to as shilling [4] [115] or profile injection [116].

Recent research has begun to examine the vulnerabilities and robustness of different recommendation techniques, such as collaborative filtering, in the face of shilling attacks [116,117,115,118]. O' Mahony [119] identify two key types of attacks

– Push: This is an attack aimed at promoting a particular item by increasing its ratings for a larger subset of users

[4] A shill is an associate of a person selling a good or service, who pretends no association and assumes the air of an enthusiastic customer [114].

– Nuke: This is an attack aimed at reducing the predicted ratings of an item so that it is recommended to a smaller subset of users

These attacks take the form of the insertion of a number of new users with a set of rating that either provide high or low ratings to particular items.

A number of different attack models have been identified in literature. The *sampling attack* [118] is primarily of theoretical interest as it requires the attacker to have access to the ratings database itself. The *random attack* [115] forms profiles by associating a positive rating for the target item with random values for the other items. The *average attack* [115] assumes that the attacker knows the average rating for each item in the database and assigns values randomly distributed around this average, except for the target item. These attacks have been found to be effective against user-based collaborative recommendation algorithms, but less so against item-based recommendation.

A *segmented attack* [120] associates the pushed item with a small number of popular items of similar type. It pushes an item to a targeted group of users with known or easily predicted preferences. Profiles are inserted that maximize the similarity between the pushed item and items preferred by the group. This attack model ensures that the pushed item will be recommended to those users that are its target segment. It is particularly effective against item-based recommendation algorithms to a degree that broader attacks are not. This attack also requires very limited knowledge about the system and the users. An attacker needs to know only a group of items well liked by the target segment and needs to build profiles containing only those items.

The study of attack models and their impact on recommendation algorithms can lead to the design of more robust and trustworthy personalization systems. The notion of trust, which is essential to the practical success of recommender systems, is further discussed below. Another important goal is the development of metrics to help quantify the effect of those attacks (see Section 7).

6.10 Trust

A user study conducted by Sinha et al. found that, in general, two types of recommendations need to be generated by a recommender system. These are *trust-generating recommendations* and *useful recommendations* [55]. They define trust-generating recommendations as items that the user has previously experienced and suggest that while these recommendations are not “useful” to the user, they build trust between the user and the system. They also found that users preferred using trusted sources for recommendations.

However, most collaborative filtering systems base the generation of recommendations simply on the similarity of the target users previous ratings with that of other users, not explicitly dealing with the issue of trust. This opens such systems to attacks such as shilling as described in Section 6.9.

Recently researchers have begun looking at how trust can be incorporated into the recommendations process [89], [121], [122]. Massa and Avesani propose the use of a “Web of trust”, a social network with users as nodes and directed weighted edges signifying a level of trust from one user to another. In their implementation of a trust-aware recommendation system, a user was allowed to rate not just items but also users based on the usefulness of their reviews/ ratings. Only users trusted by the target user were then employed within the recommendation generation process. Through the propagation of trust within the network, users not specifically rated by the target user may also participate in the recommendation process. Some of the additional advantages resulting from the use of such a trust based social network include alleviation of the new user problem commonly faced by traditional collaborative filtering systems as well as attack-resilience [121].

As opposed to depending on the user providing a Web of trust to the recommender system, O'Donovan and Smyth [122] investigated the possible learning of trust metrics from the ratings data available within the system itself. They defined two metrics for trust, one at the profile level and the other at the item level, based on the correctness of previous recommendations. The trust metric was combined with the similarity metric during neighborhood formulation.

Herlocker et al. investigated the ability of a recommender system to generate explanations for how individual recommendations were generated [123] . Three key points within the collaborative approach to recommendation generation that could provide useful information to be communicated to the user were identified as the user profile generation, neighbourhood formulation and neighbour rating combination for prediction.

They further identified the two key goals of generating explanations. The first was aimed at building trust with the user through provision of logical explanations for the recommendations generated. The second was aimed at providing the user with the ability to identify whether a recommendation is based on weak data. Additional benefits include improved data collection as a result of involving the user in the recommendation process and greater acceptance of the recommender as a decision aide. They further identified over twenty explanation interfaces and evaluated them using volunteer users of the MovieLens recommender. Of these the most valued feedback were histograms of ratings by neighbours, past performance of the recommender for the user and similarity to other items within the user's profile. Another useful finding of the study was that explanation interfaces must be simple to be successful and care must be taken not to overload to the user with information.

While explanations can be viewed positively by users, providing explanations such as ratings may further influence the user's own ratings as even simple feedback in terms of the predicted rating has been shown to influence the user's own rating [124].

7 Evaluation of Personalization Systems

Evaluation of personalization systems remains a challenge due to the lack of understanding of what factors affect user satisfaction with a personalization system. It seems obvious that a system that accurately predicts user needs and fulfils these needs without the user needing to expend the same resources in achieving the task as he would have, in the absence of the system, would be considered successful. Hence personalization systems have most commonly been evaluated is terms of the accuracy of the algorithms they employ.

Recent user studies have found that a number of issues can affect the perceived usefulness of personalization systems including, trust in the system, transparency of the recommendation logic, ability for a user to refine the system generated profile and diversity of recommendations [125], [126], [102].

For a business deploying a personalization system, accuracy of the system will be little solace if it does not translate into an increase in quantitative business metrics such as profits or qualitative metrics such as customer loyalty.

Hence the evaluation of personalization systems needs to be carried out along a number of different dimensions, some of which are better understood that others and have well established metrics available. The key dimensions along which personalization systems are evaluated include

- User Satisfaction
- Accuracy
- Coverage
- Utility
- Explainability
- Robustness
- Performance and Scalability

Attempts to measure user satisfaction range from using business metrics for customer loyalty such as RFM and life-time value through to more simplistic measures such as recommendation uptake. For example, the físchlár video recommendation system [127] implicitly obtains a measure of user satisfaction by checking is the recommended items were played or recorded.

As stated in Section 2, personalization can be viewed as a data mining task. The accuracy of models learned for this purpose can be evaluated using a number of metrics that have been used in machine learning and data mining literature such as mean absolute error (MAE) and area under the ROC curve, depending on the formulation of the learning task (see Section 2).

In the prediction task, MAE has been commonly used in collaborative filtering literature [28], [60], [93]. Other accuracy metrics used for the prediction task with numeric ratings include root mean squared error and mean squared error, that implicitly assign a greater weight to predictions with larger errors, and normalized mean squared error [92] that aims to normalize MAE across datasets with varying rating scales. Massa and Avesani suggest another variant of MAE called the mean absolute user error that calculates the mean absolute error for each user and then averages over all users [89]. This was based on their observation that recommender systems tend to have lower errors when predicting ratings by prolific raters rather than less frequent ones. This metric is particularly useful when the number of items in the test set per user varies, for example, if it is based on a percentage of items rated by a user.

Precision and Recall are standard metrics used in information retrieval. While precision measures the probability that a selected item is relevant, recall measures the probability that a relevant item is selected. Precision and recall are commonly used in evaluating the selection task [128], [58], [129]. The F1 measure that combines precision and recall, has also been used for this purpose task [130], [131].

Coverage measures the percentage of the universe of items that the recommendation system is capable of recommending. For the prediction task it is calculated as the percentage of unrated items, a rating for which can be predicted by the system. An alternative is to calculate coverage as a percentage of items of interest to a user rather than considering the complete universe of items [132].

Breese et al. suggested a metric based on the expected utility of the recommendation list [59]. The utility of each item is calculated by the difference in vote for the item and a "neutral" weight. The metric is then calculated as the weighted sum of the utility of each item in the list where the weight signifies the probability that an item in the ranked list will be viewed. This probability was based on an exponential decay. In the context of navigating a hyperlinked repository, other metrics have also been proposed that measure utility based on the distance of the recommended item from the current page referred to as navigation distance [133]. Another factor affecting utility of a recommendation is the novelty of the recommendation in the context of the overall recommendation list.

A number of metrics have been proposed in literature for evaluating the robustness of a recommender system. Each of these metrics attempt to provide a quantitative measure of the extent to which an attack can affect a recommender system. Stability of prediction [119] measures the percentage of unrated (user,items) pairs that have a prediction shift less that a predefined constant. Power of an attack [119] on the other hand measures the average change in the gap between the predicted and target rating for the target item. The target item is the item that the attack is attempting to push or nuke. The power of attack metric assumes that the goal of the attack is to force item ratings to a target rating value. Noting that the effect of an attack on an items current rating is not necessarily going to affect its ability to be recommended, Lam and Herlocker [61] proposed an alternative metric called the Change in Expected change in top-N occupancy. It is calculated as the average expected occurrence of the target items in the top-N recommendation list of users.

The performance and scalability dimension aims to measure the response time of a given recommendation algorithm and how easily it can scale to handle a large number of concurrent requests for recommendations. Typically, these systems need to be able to handle large volumes of recommendation requests without significantly adding to the response time of the Web site that they have been deployed on.

8 Conclusions and Future Directions

In this chapter, we have provided a comprehensive review of intelligent techniques for Web personalization. We have taken the view that Web personalization is an application of data mining and must therefore be supported during the various phases of a typical data mining cycle. We have described the various explicit and implicit data sources available along with the typical approaches used to transform this data into useful user profiles/models that can be used to generate recommendations. We have also described various approaches to generating recommendations from a set of user profiles/ models. Research into this topic has raised a number of interesting issues related to the personalization process. These are issues that need to be addressed by any personalization system that aims to provide robust, accurate and useful personalized content to its users. We also provide a description of the current understanding of how these systems should be evaluated, describing some of the most commonly used metrics within personalization literature.

While a lot has been achieved in the last decade of research into personalization, a number of challenges and open research questions still face researchers.

A key part of the personalization process is the generation of user models. Commonly used user models are still rather simplistic, representing the user as a vector of ratings or using a set of keywords. Even where more multi- dimensional information has been available, such as when collecting implicit measures of interest, the data has traditionally been mapped onto a single dimension, in the form of ratings. More expressive models need to be explored.

In particular profiles commonly used today lack in their ability to model user context and dynamics. Users rate different items for different reasons and under different contexts. The modelling of context and its use within recommendation generation needs to be explored further. Also, user interests and needs change with time. Identifying these changes and adapting to them is a key goal of personalization. However, very little research effort has been expended on this topic to date. This is partly due to the fact that at the deployment stage, the models used are static due to a trade-off between expressiveness of the profiles and scalability with respect to the number of concurrent personalization requests. Recently research has begun to explore user models that are based on ontological information. These richer profiles have shown promise in comparison to systems that limit user models to a vector representation. However this research is very much in its infancy and warrants further research.

Memory based approaches traditionally used for personalization suffer from scalability issues with respect to the size of the user base as well as the size of the universe of items. The applicability of research into instance selection for memory based learning [134] to collaborative filtering needs to be investigated. Also a number of indexing mechanisms based on similarity [135] have been proposed. The applicability of these to sparse data sets typically found in recommender systems needs to be investigated.

With regard to the robustness of recommenders, our understanding of attack models is still in its infancy as is our understanding of the extent to which these attacks affect the different approaches to developing recommender systems. Most studies have tended to evaluate the effect of these attacks on user-based and item-based collaborative filtering. More research needs to be carried out into how robust other model based and hybrid approaches to recommendation generation are to these attacks. Little work has been carried out into quantifying how difficult it would be to identify and prevent attacks from taking place. Data Mining has been applied successfully to network intrusion detection. Can similar techniques be applied to identifying attacks on recommender systems?

The ultimate goal of personalization is a lift in user satisfaction. However, most research into personalization has focussed evaluation on the accuracy of predicted ratings and little agreement has emerged as to what factors, other than prediction accuracy affect user satisfaction. Even less agreement exists with regard to how the effect of personalization on these factors should be measured. A lot more user studies need to be carried out to gain a better understanding of these issues. The development of more personalization exemplars with the necessary infrastructure to conduct large scale user testing is required. In addition to user satisfaction, more business oriented metrics need to be developed to measure the true economic benefit to businesses that deploy such systems.

User studies have shown explanability of recommendations as an important factor in user satisfaction, however, most systems for generating recommendations are hard to explain other than at the generic conceptual level. Explanation facilities developed in the context of knowledge based systems may provide some useful insights into how similar facilities can be developed for recommendation systems.

The use of trust within the computation of neighbourhoods has been shown to alleviate some of the issues associated with pure collaborative filtering such as the new user problem and robustness. However they require additional input from users in the form of trust networks. Some early work into using introspective learning for measuring trustworthiness of users within collaborative filtering has shown potential and warrants further investigation.

References

1. Mulvenna, M., Anand, S.S., Buchner, A.G.: Personalization on the net using web mining. Communication of ACM **43** (2000)
2. Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., Wirth, R.: Crisp-dm 1.0: Step-by-step data mining guide. http://www.crisp-dm.org (2000)
3. Cooley, R., Mobasher, B., Srivastava, J.: Data preparation for mining world wide web browsing patterns. Knowledge and Information Systems **1** (1999)
4. Middleton, S.E., Shadbolt, N.R., Roure, D.C.D.: Ontological user profiling in recommender systems. ACM Transactions on Information Systems **22** (2004) 54–88
5. Haase, P., Ehrig, M., Hotho, A., Schnizler, B.: Personalized information access in a bibliographic peer-to-peer system. In: Proceedings of the AAAI Workshop on Semantic Web Personalization, AAAI Workshop Technical Report (2004) 1–12
6. Sieg, A., Mobasher, B., Burke, R.: Inferring user's information context: Integrating user profiles and concept hierarchies. In: Proceedings of the 2004 Meeting of the International Federation of Classification Societies. (2004)
7. Mobasher, B., Dai, H., Luo, T., Sung, Y., Nakagawa, M., Wiltshire, J.: Discovery of aggregate usage profiles for web personalization. In: Proceedings of the Web Mining for E-Commerce Workshop (WebKDD'2000), held in conjunction with the ACM-SIGKDD Conference on Knowledge Discovery in Databases (KDD'2000). (2000)
8. Anand, S.S., Mulvenna, M., Chevalier, K.: On the deployment of web usage mining. In Berendt, B., Hotho, A., Mladenic, D., van Someren, M., Spiliopolou, M., Stumme, G., eds.: Web Mining: From Web to Semantic Web. LNAI 3209. Springer-Verlag (2004) 23–42
9. Lang, K.: Newsweeder: Learning to filter netnews. In: Proceedings of the 12th International Conference on Machine Learning. (1995)
10. Salton, G.: Developments in automatic text retrieval. Science **253** (1991) 974–980
11. Rissanen, J.: Modelling by shortest data description. Automatica **14** (1978) 465–471
12. Resnick, P., Iacovou, N., Sushak, M., Bergstrom, P., , Riedl, J.: Grouplens: An open architecture for collaborative filtering of netnews. In: Proceedings of the 1994 Computer Supported Collaborative Work Conference. (1994)
13. Mobasher, B., Dai, H., Luo, T., Sung, Y., Zhu, J.: Integrating web usage and content mining for more effective personalization. In: Proceedings of the International Conference on E-Commerce and Web Technologies. (2000)

14. Balabanovic, M., Shohan, Y.: Fab: Content-based, collaborative recommendation. Communications of the ACM **40** (1997) 66–72
15. Burke, R.: Knowledge-based recommender systems. Encyclopedia of Library and Information Systems **69** (2000)
16. McGinty, L., Smyth, B.: Improving the performance of recommender systems that use critiquing. In: Intelligent Techniques in Web Personalisation. LNAI. Springer-Verlag (2005)
17. Lorenzi, F., Ricci, F.: Case-based recommender systems: a unifying view. In: Intelligent Techniques in Web Personalisation. LNAI. Springer-Verlag (2005)
18. McGinty, L., Smyth, B.: Comparison-based recommendation. In: Proceedings of the 6th European Conference on Case-based Reasoning. (2002)
19. Burke, R., Hammond, J., Kulyukin, V.A., Lytinen, S.L., Tomuro, N., Schoenberg, S.: Question answering from frequently asked question files. AI Magazine **18** (1997) 57–66
20. Fesenmaier, D.R., Ricci, F., Schaumlechner, E., Wober, K., Zanella, C.: Dietorecs: Travel advisory for multiple decision styles. In Frew, A.J., Hitz, M., O'Connor, P., eds.: Information and Communication Technologies in Tourism. Springer-Verlag (2003) 232–241
21. Shimazu, H.: Expertclerk: A conversational case-based reasoning tool for developing salesclerk agents in e-commerce webshops. Artificial Intelligence Review **18** (2002) 223–244
22. Linden, G., Smith, B., York, J.: Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing (2003) 76–80
23. Mobasher, B., Cooley, R., Srivastava, J.: Automatic personalization based on web usage mining. Communication of ACM **43** (2000)
24. Mobasher, B., Dai, H., Luo, T., , Nakagawa, M.: Effective personalization based on association rule discovery from web usage data. In: Proceedings of the 3rd ACM Workshop on Web Information and Data Management (WIDM01), held in conjunction with the International Conference on Information and Knowledge Management. (2001)
25. Eirinaki, M., Vlachakis, J., Anand, S.S.: Ikum: An integrated web personalization platform based on content structures and usage behaviour. In: Intelligent Techniques in Web Personalisation. LNCS. Springer-Verlag (2005)
26. Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R., Riedl, J.: Applying collaborative filtering to usenet news. Communications of the ACM **40** (1997) 77–87
27. Good, N., Schafer, J.B., Konstan, J., Borchers, A., Sarwar, B., Herlocker, J., Riedl, J.: Combining collaborative filtering with personal agents for better recommendations. In: Proceedings of the 1999 Conference of the American Association of Artificial Intelligence (AAAI-99). (1999) 439–446
28. Shardanand, U., Maes, P.: Social information filtering: Algorithms for automating word of mouth. In: Proceedings of CHI. (1995) 210–217
29. Ghani, R., Fano, A.: Building recommender systems using a knowledge base of product semantics. Accenture Technology Labs (2002)
30. Pazzani, M.: A framework for collaborative, content-based and demographic filtering. Artificial Intelligence Review **13** (1999) 393–408
31. Krulwich, B.: Lifestyle finder: Intelligent user profiling using large-scale demographic data. AI Magazine **18** (1997) 37–45
32. Maes, P.: Agents that reduce work and information overload. Communications of the ACM **37** (1994) 30–40
33. Yang, Y.: Expert network: Effective and efficient learning from human decisions in text categorization and retrieval. In: Proceedings of the 7th International ACM-SIGIR Conference on Research and Development in Information Retrieval. (1994) 13–22
34. O'Riordan, A., Sorensen, H.: An intelligent agent for high-precision text filtering. In: Proceedings of the 4th International Conference on Information and Knowledge Management. (1995) 205–211

35. Sieg, A., Mobasher, B., Burke, R., Prabhu, R.G., Lytinen, S.: Representing user information context with ontologies. In: Proceedings of HCI International Conference. (2005) 210–217
36. Parent, S., Mobasher, B., Lytinen, S.: An adaptive agent for web exploration based on concept hierarchies. In: Proceedings of the 9th International Conference on Human Computer Interaction. (2001)
37. Cassel, L., Wolz, U.: Client side personalization. In: Proceedings of the Second DELOS Network of Excellence Workshop on Personalization and Recommender Systems in Digital Libraries. (2001)
38. Guttman, R., Maes, P.: Agent-mediated integrative negotiation for retail electronic commerce. In: Proceedings of the Workshop on Agent Mediated Electronic Trading. (1998)
39. Murthi, B., Sarkar, S.: The role of the management sciences in research on personalization. Review of Marketing Science Working Papers **2** (2002)
40. Nichols, D.M.: Implicit rating and filtering. In: Proceedings of the fifth DELOS Workshop on Filtering and Collaborative Filtering. (1998) 31–36
41. Rucker, J., Polanco, M.: Siteseer: Personalized navigational for the web. Communications of the ACM **40** (1997) 73–76
42. Lieberman, H.: Autonomous interface agents. In: Proceedings of the ACM Conference on Human Factors in Computing Systems. (1997) 67–74
43. Claypool, M., Le, P., Waseda, M., Brown, D.: Implicit interest indicators. In: Proceedings of the 6th International Conference on Intelligent User Interfaces. (2001) 33–40
44. Lieberman, H.: Letizia: An agent that assists web browsing. In: Proceedings of the 14th International Joint Conference in Artificial Intelligence. (1995) 924–929
45. Mladenic, D.: Personal web watcher: Implementation and design. Technical Report IJS-DP-7472, Department of Intelligent Systems, J. Stefan Institute, Slovenia (1996)
46. Schwab, I., Kobsa, A., Koychev, I.: Learning about users from observation. In: Adaptive User Interfaces: Papers from the 2000 AAAI Spring Symposium. (2000)
47. Holte, R.C., Yan: Inferring what a user is not interested in. In: Lecture Notes In Computer Science (Proceedings of the 11th Biennial Conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence). Springer-Verlag (1996) 159–171
48. Carroll, J., Rosson, M.B.: The paradox of the active user. In Carrol, J.M., ed.: Interfacing Thought: Cognitive Aspects of Human-Computer Interaction. MIT Press (2005)
49. Sarwar, B.M., Konstan, J.A., Borchers, A., Herlocker, J., Miller, B., Riedl, J.: Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In: Computer Supported Cooperative Work. (1998) 345–354
50. Berendt, B., Teltzrow, M.: Addressing users' privacy concerns for improving personalization quality: Towards an integration of user studies and algorithm evaluation. In: Intelligent Techniques in Web Personalisation. LNAI. Springer-Verlag (2005)
51. Krulwich, B., Burkey, C.: Learning user information interests through extraction of semantically significant phrases. In: Proceedings of the AAAI Spring Symposium on Machine Learning in Information Access. (1996)
52. Pazzani, M., Billsus, D.: Learning and revising user profiles: The identification of interesting web sites. Machine Learning **27** (1997) 313–331
53. Mladenic, D.: Text-learning and related intelligent agents: A survey. IEEE Intelligent Agents (1999) 44–54
54. Heckerman, D.: A tutorial on learning with bayesian networks. Technical Report MSR-TR-95-06, Microsoft Corporation (1995)
55. Sinha, R., Swearingen, K.: Comparing recommendaions made by online systems and friends. In: Proceedings of Delos-NSF Workshop on Personalisation and Recommender Systems in Digital Libraries. (2001)

56. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using collaborative filtering to weave an information tapestry. Communications of the ACM **35** (1992)
57. Schafer, J., Konstan, J., , Riedl, J.: Recommender systems in e-commerce. In: Proceedings of the ACM Conference on Electronic Commerce. (1999)
58. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.: Application of dimensionality reduction in recommender system - a case study. In: ACM WebKDD 2000 Web Mining for E-Commerce Workshop. (2000)
59. Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence. (1998) 43–52
60. Herlocker, J., Konstan, J., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: Proceedings of the 1999 Conference on Research and Development in Information Retrieval. (1999)
61. Lam, S., Riedl, J.: Shilling recommender systems for fun and profit. In: Proceedings of the 13th international conference on World Wide Web. (2004) 393–402
62. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th International World Wide Web Conference. (2001)
63. Ungar, L., Foster, D.P.: Clustering methods for collaborative filtering. In: Proceedings of the Workshop on Recommendation Systems. (1998)
64. O'Connor, M., Herlocker, J.: Clustering items for collaborative filtering. In: Proceedings of ACM SIGIR'99 Workshop on Recommender Systems: Algorithms and Evaluation. (1999)
65. Kohrs, A., Merialdo, B.: Clustering for collaborative filtering applications. In: Computational Intelligence for Modelling, Control & Automation. IOS Press (1999)
66. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society **39** (1977) 1–38
67. Agrawal, R., Imielinski, T., Swami, A.: Mining associations between sets of items in massive databases. In: Proceedings of the ACM-SIGMOD 1993 International Conference on Management of Data. (1993) 207–216
68. R. Agrawal, R.S.: Mining sequential patterns. In: Proceedings of the International Conference on Data Engineering (ICDE). (1995)
69. Baumgarten, M., Buchner, A.G., Anand, S.S., Mulvenna, M.D., Hughes, J.: User-driven navigation pattern discovery from internet data. web usage analysis and user profiling. In Masand, B., Spiliopoulou, M., eds.: Web Usage Analysis and User Profiling: Proceedings of the WEBKDD'99 Workshop. Lecture Notes in Computer Science 1836. Springer-Verlag (2000) 74–91
70. Padmanabhan, B., Tuzhilin, A.: Unexpectedness as a measure of interestingness in knowledge discovery. Decision Support Systems **27** (1999) 303–318
71. Silberschatz, A., Tuzhilin, A.: What makes patterns interesting in knowledge discovery systems. IEEE Transactions on Knowledge and Data Engineering **8** (1996) 970–974
72. Tan, P., Kumar, V., Srivastava, J.: Selecting the right objective measure for association analysis. Information Systems **29** (2004) 293–313
73. Fu, X., Budzik, J., Hammond, K.J.: Mining navigation history for recommendation. In: Proceedings of the 2000 International Conference on Intelligent User Interfaces, New Orleans, LA, ACM Press (2000)
74. Lin, W., Alvarez, S.A., Ruiz, C.: Efficient adaptive-support association rule mining for recommender systems. Data Mining and Knowledge Discovery **6** (2002) 83–105
75. Mobasher, B., Dai, H., Luo, T., Nakagawa, M.: Effective personalization based on association rule discovery from web usage data. In: Proceedings of the 3rd ACM Workshop on Web Information and Data Management (WIDM01), Atlanta, Georgia (2001)

76. Sarwar, B.M., Karypis, G., Konstan, J., Riedl, J.: Analysis of recommender algorithms for e-commerce. In: Proceedings of the 2nd ACM E-Commerce Conference (EC'00), Minneapolis, MN (2000)
77. Agarwal, R., Aggarwal, C., Prasad, V.: A tree projection algorithm for generation of frequent itemsets. In: Proceedings of the High Performance Data Mining Workshop, Puerto Rico (1999)
78. Agrawal, R., Srikant, R.: Mining sequential patterns. In: Proceedings of the International Conference on Data Engineering (ICDE'95), Taipei, Taiwan (1995)
79. Mobasher, B., Dai, H., Luo, T., Nakagawa, M.: Using sequential and non-sequential patterns for predictive web usage mining tasks. In: Proceedings of the IEEE International Conference on Data Mining. (2002)
80. Nakagawa, M., Mobahser, B.: Impact of site characteristics on recommendation models based on association rules and sequential patterns. In: Proceedings of the IJCAI'03 Workshop on Intelligent Techniques for Web Personalization. (2003)
81. Pitkow, J., Pirolli, P.: Mining longest repeating subsequences to predict world wide web surfing. In: Proceedings of Second USENIX Symposium on Internet Technologies and Systems. (1999)
82. M. Deshpande, G.K.: Selective markov models for predicting web-page accesses. In: Proceedings SIAM International Conference on Data Mining. (2000)
83. Aggarwal, C.C., Wolf, J.L., Wu, K., Yu, P.: Horting hatches an egg: A new graph-theoretic approach to collaborative filtering. In: Proceedings of the ACM KDD Conference. (1999) 201–212
84. Mirza, B.J., Keller, B.J., Ramakrishnan, N.: Studying recommendation algorithms by graph analysis. Journal of Intelligent Information Systems **20** (2003) 131–160
85. Mirza, B.J.: Jumping connections: A graph theoretic model for recommender systems. MSc Thesis, Virginia Tech (2001)
86. Nakagawa, M., Mobahser, B.: A hybrid web personalization model based on site connectivity. In: Proceedings of the WebKDD Workshop at the ACM SIGKKDD International Conference on Knowledge Discovery and Data Mining. (2003)
87. Burke, R.: Hybrid systems for personalized recommendations. In: Intelligent Techniques in Web Personalisation. LNAI. Springer-Verlag (2005)
88. Smyth, B., Cotter, P.: A personlized television listing service. Communication of the ACM **43** (2000) 107–111
89. Massa, P., Avesani, P.: Trust-aware collaborative filtering for recommender systems. In: Proceedings of International Conference on Cooperative Information Systems. (2004)
90. Mobasher, B., Jin, X., Zhou, Y.: Semantically enhanced collaborative filtering on the web. In Berendt, B., Hotho, A., Mladenic, D., van Someren, M., Spiliopolou, M., Stumme, G., eds.: Web Mining: From Web to Semantic Web. LNAI 3209. Springer-Verlag (2004)
91. Cohen, W.: Fast effective rule induction. In: Proceedings of the 12th International Conference on Machine Learning. (1995)
92. Goldberg, K., Roeder, T., Gupta, D., Perkins, C.: Eigentaste: A constant time collaborative filtering algorithm. Information Retrieval **4** (2001) 133–151
93. Yu, K., Xu, X., Ester, M., Kriegel, H.P.: Feature weighting and instance selection for collaborative filtering: An information-theoretic approach. Knowledge and Information Systems **5** (2003)
94. Chee, S.H.S., Han, J., Wang, K.: Rectree: An efficient collaborative filtering method. In Carrol, J.M., ed.: Proceedings of the Third International Conference on Data Warehousing and Knowledge Discovery. LNCS 2114. Springer-Verlag (2001) 141–151
95. Pryor, M.H.: The effect of singular value decomposition on collaborative filtering. Dartmouth College, Computer Science Technical Report PCS-TR98-338 (1998)

96. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.: Incremental svd-based algorithms for highly scaleable recommender systems. In: Proceedings of the Fifth International Conference on Computer and Information Technology. (2002) 345–354
97. Brand, M.: Fast online svd revisions for lightweight recommender systems. In: Proceedings of the 3rd SIAM International Conference on Data Mining. (2003)
98. Tang, T., Winoto, P., Chan, K.C.C.: Scaling down candidate sets based on the temporal feature of items for improved hybrid recommendations. In: Intelligent Techniques in Web Personalisation. LNCS. Springer-Verlag (2005)
99. Volokh, E.: Personalization and privacy. Communication of the ACM **43** (2000) 84–88
100. Canny, J.: Collaborative filtering with privacy. In: Proceedings of the IEEE Security and Privacy Conference. (2002) 45–57
101. Ramakrishnan, N., Keller, B.J., Mirza, B.J., Grama, A.Y., Karypis, G.: Privacy risks in recommender systems. IEEE Internet Computing (2001) 54–62
102. Ziegler, C., McNee, S.M., Konstan, J.A., Lausen, G.: Improving recommendation lists through topic diversification. In: Proceedings of the 14th international conference on World Wide Web. (2005) 22–32
103. B.Smyth, McClave, P.: Similarity vs diversity. In: Proceedings of the 4th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development. (2001) 347 – 361
104. Sheth, B.: A learning approach to personlized information filtering. Masters Thesis, Massachusetts Institute of Technology (1994)
105. Lieberman, H., Selker, T.: Out of context: Computer systems that adapt to, and learn from, context. IBM Systems Journal **39** (2000) 617–632
106. Allan, J., Aslam, J., Belkin, N., Buckley, C., Callan, J., Croft, B., Dumais, S., Fuhr, N., Harman, D., Harper, D.J., Hiemstra, D., Hofmann, T., Hovy, E., Kraaij, W., Lafferty, J., Lavrenko, V., Lewis, D., Liddy, L., Manmatha, R., McCallum, A., Ponte, J., Prager, J., Radev, D., Resnik, P., Robertson, S., Rosenfeld, R., Roukos, S., Sanderson, M., Shwartz, R., Singhal, A., Smeaton, A., Turtle, H., Voorhees, E., Weischedel, R., Xu, J., C.Zhai: Challenges in information retrieval and language modelling: Report of a workshop held in the centre for intelligent information retrieval. ACM SIGIR Forum **37** (2002) 31–47
107. Dai, H., Mobasher, B.: A road map to more effective web personalization: Integrating domain knowledge with web usage mining. In: Proceedings of the International Conference on Internet Computing. (2003) 58–64
108. Cho, Y., Kim, J.: Application of web usage mining and product taxonomy to collaborative recommendations in e-commerce. Expert Systems with Applications **26** (2004) 233–246
109. Niu, L., Yan, X., , Zhang, C., , Zhang, S.: Product hierarchy-based customer profiles for electronic commerce recommendation. In: Proceedings of the 1st International Conference on Machine Learning and Cybernetics. (2002) 1075–1080
110. Billsus, D., Pazzani, M.J.: User modeling for adaptive news access. User Modelling and User-Adapted Interaction **10** (2000) 147–180
111. McCallum, A., Nigam, K.: A comparison of event models for naïve bayes text classification. In: AAAI/ICML-98 Workshop on Learning for Text Categorization, Technical Report WS-98-05, AAAI Press (1998)
112. Koychev, I., Schwab, I.: Adapting to drifting user's interests. In: Proceedings of ECML2000/MLNet workshop on Machine Learning in the New Information Age. (2000)
113. Moukas, A., Zacharia, G.: Evolving a multi-agent information filtering solution in amalthaea. In: Proceedings of the First International Conference on Autonomous Agents. (1997) 394–403
114. : Wikipedia: The free encyclopedia. http:\\en.wikipedia.org\wiki\Main_Page (2000)
115. Lam, S., Reidl, J.: Shilling recommender systems for fun and profit. In: Proceedings of the 13th International WWW Conference, New York (2004)

116. Burke, R., Mobasher, B., Zabicki, R., Bhaumik, R.: Identifying attack models for secure recommendation. In: Beyond Personalization: A Workshop on the Next Generation of Recommender Systems, San Diego, California (2005)
117. Burke, R., Mobasher, B., Bhaumik, R.: Limited knowledge shilling attacks in collaborative filtering systems. In: Proceedings of the 3rd IJCAI Workshop in Intelligent Techniques for Personalization, Edinburgh, Scotland (2005)
118. O'Mahony, M., Hurley, N., Kushmerick, N., Silvestre, G.: Collaborative recommendation: A robustness analysis. ACM Transactions on Internet Technology **4** (2004) 344–377
119. Mahony, M.O., Hurley, N., Kushmerick, N., Silverstre, G.: Collaborative recommendations: A robustness analysis. ACM Transactions on Internet Technologies **4** (2004) 344–377
120. Mobasher, B., Burke, R., Bhaumik, R., Williams, C.: Effective attack models for shilling item-based collaborative filtering systems. In: Proceedings of the WebKDD 2005 Workshop, in conjunction with the ACM SIGKKDD 2005, Chicago (2005)
121. Massa, P., Bhattacharjee, B.: Using trust in recommender systems: an experimental analysis. In: Proceedings of the 2nd International Conference on Trust Management. (2004)
122. Donovan, J.O., Smyth, B.: Trust in recommender systems. In: Proceedings of the 10th international conference on Intelligent user interfaces. (2005) 167–174
123. Herlocker, J., Konstan, J., Riedl, J.: Explaining collaborative filtering recommendations. In: Proceedings of ACM 2000 Conference on Computer Supported Cooperative Work. (2000) 241–250
124. Cosley, D., Lam, S.K., Albert, I., Konstan, J.A., Riedl, J.: Is seeing believing? how recommender interfaces affect users' opinions. In: Proceedings of the SIGCHI conference on Human factors in computing systems. (2003) 585–592
125. Swearingen, K., Sinha, R.: Beyond algorithms: An hci perspective on recommender systems. In: Proceedings of the ACM SIGIR Workshop on Recommender Systems. (2001)
126. Sinha, R., Swearingen, K.: The role of transaprency in recommender systems. In: CHI '02 extended abstracts on Human factors in computing systems. (2002) 830–831
127. Smeaton, A., Murphy, N., O'Connor, N.E., Marlow, S., Lee, H., McDonald, K., Browne, P., Ye, J.: The físchlár digital video system: a digital library of broadcast tv programmes. In: Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital Libraries. (2001) 312–313
128. Karypis, G.: Evaluation of item-based top-n recommendation algorithms. In: Proceedings of the tenth International conference on Information and knowledge management. (2001) 247–254
129. D.Billsus, Pazzani, M.J.: Learning collaborative information filters. In: Proceedings of ICML. (1998) 46–53
130. Mobasher, B., Dai, H., Luo, T., Nakagawa, M.: Discovery and evaluation of aggregate usage profiles for web personalization. Data Mining and Knowledge Discovery **6** (2002) 61–82
131. Basu, C., Hirsh, H., Cohen, W.: Recommendation as classification: Using social and content-based information in recommendation. In: Proceedings of the Recommender System Workshop. (1998) 11–15
132. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems **22** (2004) 5–53
133. Anderson, C., Domingos, P., Weld, D.: Adaptive web navigation for wireless devices. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence. (2001) 879–884
134. Wilson, D., Martinez, T.R.: Reduction techniques for instance-based learning algorithms. Machine Learning **38** (1997) 257–286
135. Daelemans, W., van den Bosch, A., Weijters, T.: Igtree: Using trees for compression and classification in lazy learning algorithms. Artificial Intelligence Review **11** (1997) 407–423