Artificial Intelligence Review **19:** 285–330, 2003.

A Taxonomy of Recommender Agents on the Internet

MIQUEL MONTANER, BEATRIZ LÓPEZ and JOSEP LLUÍS DE LA ROSA

Agents Research Laboratory, Institut d'Informàtica i Aplicacions, Universitat de Girona, Campus Montilivi, 17071 Girona, Spain. E-mails: mmontane@eia.udg.es; blopez@eia.udg.es; peplluis@eia.udg.es

Abstract. Recently, Artificial Intelligence techniques have proved useful in helping users to handle the large amount of information on the Internet. The idea of personalized search engines, intelligent software agents, and recommender systems has been widely accepted among users who require assistance in searching, sorting, classifying, filtering and sharing this vast quantity of information. In this paper, we present a state-of-the-art taxonomy of intelligent recommender agents on the Internet. We have analyzed 37 different systems and their references and have sorted them into a list of 8 basic dimensions. These dimensions are then used to establish a taxonomy under which the systems analyzed are classified. Finally, we conclude this paper with a cross-dimensional analysis with the aim of providing a starting point for researchers to construct their own recommender system.

Keywords: agents, information filtering, personalization, profile exploitation, profile generation, profile maintenance, recommender systems, taxonomy, user modelling

1. Introduction

The development of the Internet has resulted in a global information society with a growing number of users around the world. Yet, because of this avalanche of information at our doors, there is a rapidly increasing difficulty in finding what we want when we need it and in a manner which best meets our requirements. Users are constantly confronted with situations in which they have too many options to choose from, where they need help to explore and to filter out their preferences from the myriad possibilities. Internet Search Engines, designed originally to be helpful, now commonly find many thousands of potentially relevant sites, thus losing their usefulness.

Recently, in the Artificial Intelligence community, there has been a great deal of work on how AI can help to solve this problem. The idea of personalized search engines, intelligent software agents, and recommender systems has been widely accepted among users who require assistance in searching, sorting, classifying, filtering and sharing the vast amount of information now available on the Web. A combination of modelling the preferences

of particular users, building content models, and modelling social patterns in intelligent agents (Maes 1994) would provide users with a means for managing information in a rational way, thus helping them to overcome the information overload.

The kind of personalized agent which develops will depend on the requirements of the application. It is therefore essential that we identify different design possibilities and investigate their properties. In this paper, we carry out a comprehensive and systematic study of recommender agents. Analyzing the different systems, we have identified a list of 8 dimensions in which recommender agents can differ and possible values for these dimensions, therefore providing a taxonomy.

The intention of this paper is to present state-of-the-art elements organized into a simple classification, explain the methods used and describe their advantages and disadvantages. Thus, the main purpose is to provide a starting point for researchers to construct their own recommender agents.

This paper is organized as follows. First, we present the dimensions that constitute the taxonomy, which we group in two blocks: dimensions regarding profile generation and maintenance and dimensions related to profile exploitation. In Sections 3 and 4 we provide the classification of the systems according to dimensions of profile generation and maintenance and profile exploitation respectively. We continue by performing a cross-dimensional analysis in Section 5 and end with Section 6 in which several conclusions are presented.

2. The Taxonomy

In a relatively short time, several recommender agents have been developed and there is a wide variety of such systems, all of which take advantage of a particular set of AI techniques. We have followed two main approaches in this study of recommender agents: spatial and functional. The spatial approach produces a classification of agents according to the application domain (see Table 1 for the domains of the various systems analyzed). The functional approach produces a classification based on the different task-achievement techniques used in the system. This latter approach allows us to study the systems systematically and consequently, we have spent more time on it.

Consistently, when analyzing how a personal agent makes recommendations or assesses a user, the key issue is the user profile. User profile generation and maintenance requires five design decisions which constitute the first five dimensions of our taxonomy: the profile representation technique, the technique used to generate the initial profile, the source of the relevance feedback which represents the user interests, the profile learning

Table 1. Domain of the analyzed systems

NAME	REFERENCES	DOMAIN
ACR News	Mobasher et al. 2000	Netnews filtering
Amazon	Amazon 2001	E-commerce
Amalthaea	Moukas 1997	Web recommender
Anatagonomy	Skagami et al. 1997	Personalized newspaper
Beehive	Huberman and Kaminsky 1996	Sharing news
Bellcore Video Recom	Hill 1995	Movie recommender
Casmir	Berney and Ferneley 1999	Document recommender
CDNow	CDnow	E-commerce
Fab	Balabanovic and Shokam 1997	Web recommender
GroupLens	Resnick et al. 1994	Netnews recommender
ifWeb	Minio and Tasso 1996; Asnicar and Tasso 1997	Web recommender
InfoFinder	Krulwich and Burkey 1995, 1996	Information recommender
INFOrmer	Riordan and Sorensen 1995; Sorensen et al. 1997	Netnews filtering
Krakatoa Chronicle	Kamba et al. 1995	Personalized newspaper
LaboUr	Schwab et al. 2001	Document recommender
Let's Browse	Lieberman et al. 1999	Web recommender
Letizia	Lieberman et al. 1995	Web recommender
LifeStyle Finder	Krulwich 1997	Purchase, travel and store recommender
MovieLens	Good et al. 1999	Movie recommender
News Dude	Billsus and Pazzani 1999	Netnews recommender
NewsWeeder	Lang 1995	Netnews recommender
NewT	Sheth and Maes 1993	Netnews filtering
Personal WebWatcher	Mladenic 1996	Web recommender
PSUN	Sorensen and McElligot 1995	Netnews recommender
Re:Agent	Boone 1998	E-mail filtering
Recommender	Basu et al. 1998	Movie recommender
Ringo/FireFly	Shardanand 1994; Shardanand and Maes 1995	Music recommender
SIFT Netnews	Yan and Garcia-Molina 1995	Netnews filtering
SiteIF	Stefani and Strappavara 1998	Web recommender
Smart Radio	Hayes and Cunningham 1999, 2000	Music lists recommender
Syskill & Webert	Pazzani et al. 1996; Pazzani and Billsus 1997	Web recommender
Tapestry	Goldberg 1992	E-mail filtering
Webmate	Chen and Sycara 1998	Web recommender
WebSail	Chen et al. 2000	Web search filtering
WebSell	Cunningham et al. 2001	Purchase recommender
Websift	Cooley 1999	Web recommender
WebWatcher	Armstrong et al. 1995; Joachims et al. 1997	Web recommender

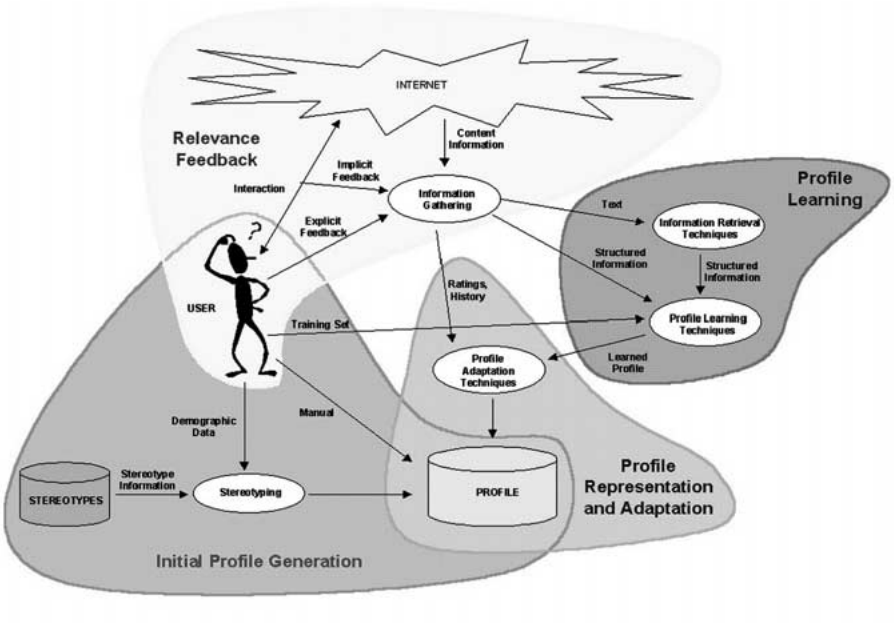

Figure 1. Profile generation and maintenance.

technique and the profile adaptation technique. Figure 1 shows the relationships between these techniques in the generation and maintenance of user profiles.

The *profile representation* is the first step to take into account in a recommender agent since the other techniques depend on it. Once this step is decided, the other techniques can be defined. A recommender agent cannot begin to function until the user profile has been created. Furthermore, the system needs to know as much as possible from the user in order to provide him/her with satisfactory results from the very beginning. Therefore, systems need to use a suitable technique in order to generate an accurate *initial profile*.

To generate and maintain the user profile, the system needs relevant information about the user's interests. When users interact with a computer, they provide a great deal of information about themselves. Successful interpretation of these data streams is necessary for computers to tailor themselves to each individual's behavior, habits and knowledge. As for the interaction of the user with these applications, the system can gather relevance feedback to learn his tastes, interests and preferences. *Relevance feedback* is then a main dimension for recommender agents. Typically, the feedback, given explicitly or implicitly by the user, has no sense in itself. Therefore, there is a need for a *profile learning technique* which extracts the relevant information and structures this information depending on the representation of the profile.

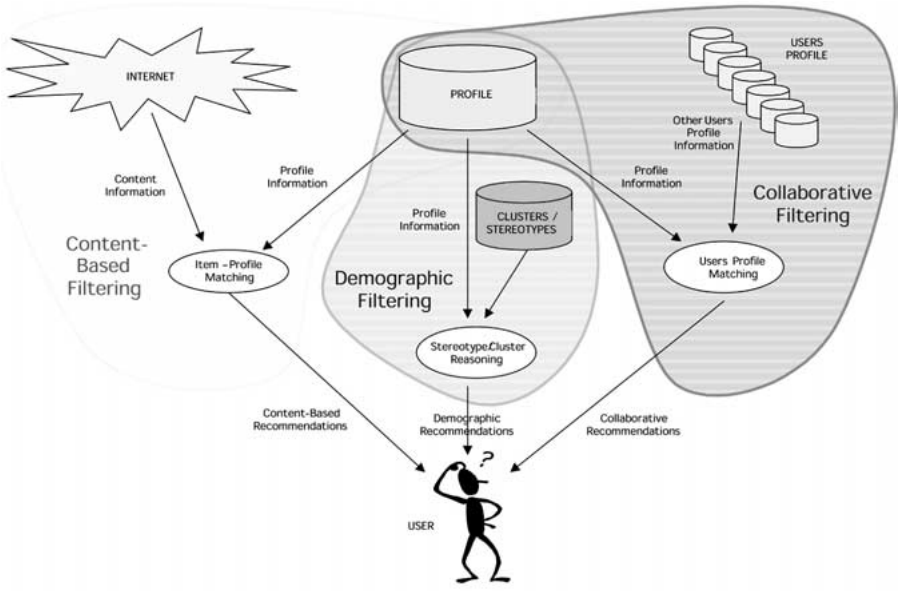

Figure 2. Profile exploitation for recommendation.

User tastes usually change as time goes on. Therefore, the user profile should also be changed in order to retain the desired accuracy in its exploitation. Hence, the need for a technique to *adapt the user profile* to new interests and to forget old ones.

Once there is a user profile available, personal agents exploit it to recommend either products or actions to a user. Intelligent agents make decisions according to the information available. Such information includes data about items as well as different profiles of other agents on the web. Since there is so much information, a fundamental issue is to select the most appropriate information with which to make decisions. In other words, an information filtering method is essential. There are three information filtering approaches for making recommendations: demographic filtering, content-based filtering and collaborative filtering. Demographic filtering uses descriptions of people to learn the relationship between a particular item and the type of people who like it. Content-based filtering uses descriptions of the content of the items to learn the relationship between a single user and the description of the items. Several *user profile-item matching* methods can be used in order to compare the user's interests and the items. Collaborative filtering uses the feedback from a set of people concerning a set of items in order to make recommendations, but ignores the content of the items. Various methods are used by the systems to *match user profiles* and find users with similar interests.

Table 2. Dimensions of the taxonomy

TAXONOMY OF RECOMMENDER AGENTS	
Profile generation and maintenance	Profile exploitation
User profile representation	Information filtering method
Initial profile generation	User profile-item matching technique
Profile learning technique	User profile matching technique
Relevance feedback	Profile adaptation technique

In terms of profile exploitation, then, three main dimensions characterize intelligent recommender agents: the information filtering method (demographic, content-based and collaborative), the item-profile matching (when content-based) and the user profile matching techniques (when collaborative). See Figure 2 for a general view.

All in all, we have identified, from a functional viewpoint, 8 classification dimensions for recommender agents, 5 in terms of profile generation and maintenance and 3 in terms of profile exploitation (see Table 2). We will now go on to discuss these in further detail.

3. Profile Generation and Maintenance

Five design decisions should be taken to generate and maintain a user profile: the representation, the technique to generate the initial profile, the source of the relevance feedback which represents the user interest, the profile learning technique and the profile adaptation technique (see Figure 1).

3.1. *User profile representation*

Constructing accurate profiles is a key task since the system's success will depend, to a large extent, on the ability to represent the user's current interests. Accurate profiles are vital for both the content-based component (to insure recommendations are appropriate) and the collaborative component (to insure that users with similar profiles are indeed similar).

Several approaches have been taken to represent user profiles, such as a history of purchases, web navigation or e-mails, an indexed vector of features, a n-gram, a semantic network, an associative network, a classifier including neural networks, decision trees, inducted rules or Bayesian networks, a matrix of ratings and a set of demographic features. Table 3 shows the user profile representation techniques used by the different systems analyzed.

Table 3. Profile representation technique of the systems

NAME	TECHNIQUE
ACR News	Frequent itemsets, URL clusters
Amazon	Purchase history with ratings
Amalthaea	Weighted feature vector
Anatagonomy	Weighted feature vector
Beehive	Clusters (weighted feature vector)
Bellcore Video Recom	User-item ratings matrix
Casmir	Weighted feature and document network
CDNow	Purchase history with ratings
Fab	Weighted feature vector
GroupLens	User-item ratings matrix
ifWeb	Multivalued weighted attributes, weighted semantic network
InfoFinder	Decision tree
INFOrmer	Weighted associative network
Krakatoa Chronicle	Weighted feature vector
LaboUr	Probabilistic feature vector, boolean feature vector
Let's Browse	Weighted reature vector
Letizia	Weighted freature vector
LifeStyle Finder	Demographic features
MovieLens	Weighted feature vector, inducted rules
News Dude	Short-term: weighted, long-term: boolean feature vector
NewsWeeder	Weighted feature vector
NewT	Weighted feature vector
Personal WebWatcher	Probabilistic feature vector
PSUN	Weighted n-grams
Re:Agent	Weighted feature vector, neural network
Recommender	Inducted rules
Ringo/FireFly	User-item ratings matrix
SIFT Netnews	Boolean feature vector, weighted feature vector, decision tree
SiteIF	Weighted semantic networks
Smart Radio	User-item ratings matrix
Syskill & Webert	Probabilistic feature vector, boolean feature vector, decision tree, weighted feature vector
Tapestry	Indexed messages and annotations
Webmate	Weighted feature vector
WebSail	Boolean feature vector
WebSell	Interesting/not interesting products
Websift	Inducted rules, patterns, statistics
WebWatcher	Boolean feature vector

3.1.1. *History-based model*

Some systems keep a list of purchases, the navigation history in WWW or the content of e-mail boxes as a user profile. Additionally, it is also common to keep the relevant feedback of the user associated with each item in the history.

A historical approach is most commonly used in e-commerce, in which systems keep a list of purchased products and user ratings, as a user profile. This is the case in the two most popular state-of-the-art recommender systems in e-commerce: Amazon.com (Amazon 2001) and CDNow.com (CDNow 2001). A similar approach is used in WebSell (Cunningham et al. 2001), in which the profile is defined by using two lists, one with purchased products rated as *interesting* and another with *uninteresting* ones. Another approach is implemented in Tapestry (Goldberg et al. 1992), an e-mail filtering system which builds a profile while keeping track of messages and annotations given by the user.

3.1.2. *Vector space model*

In the vector space model, items are represented with a vector of features, usually words or concepts, with an associated value. This value can be a Boolean or a real number. The Boolean value represents the presence of the value of the feature, and the real number represents the frequency, relevance or probability of the feature, which is calculated using information indexing techniques (see Section 3.3.2).

For example, Webmate (Chen and Sycara 1998) utilizes a multiple feature vectors representation. The basic idea is to represent each document as a vector in a vector space so that documents with similar content have similar vectors. Each dimension of the vector space represents a word and its weight, calculated as a combination of the statistics term frequency (see Section 3.3.2).

3.1.3. *Weighted n-grams*

In weighted n-grams, items are represented with a net of words with weights in the nodes and edges. For example, PSUN (Sorensen and McElligot 1995), based on the assumption that words tend to occur one after another a significantly high number of times, extracts fixed length consecutive series of n characters and organizes them with weighted links representing the co-occurrence of different words. Therefore, the structure achieves a context representation of the words.

3.1.4. *Weighted semantic networks*

Semantic networks (Potter and Trueblood 1988) are able to store the meanings of words, so the human-like use of these meanings is possible. Minio and Tasso, in the ifWeb system (Minio and Tasso 1996), implement such

an approach. In IfWeb, a semantic network base contains a collection of semantic networks describing a typical pattern of topics of interest to the user. The Stefani and Strapparava approach in the SiteIF system (Stefani and Strappavara 1998) represents every node as a word or an interesting concept and the arcs between nodes are the co-occurrence relation between two words; every node and every arc has a weight representing a different level of interests to the user.

3.1.5. *Weighted associative networks*

An associative network consists of a set of nodes which represent primary terms, concepts or words, in which a user is interested. A set of weighted links establishes the organization of these terms into relevant phrases. Associative networks differ from the semantic networks because semantic networks have different generic link types such as synonymy, superclass-subclass, and also possibly disjunctive and conjunctive sets of links. In contrast, associative networks have only a single link type, a weighted edge, the semantics being implicit in the structure of the network and the parameters associated with the processing (Riordan and Sorensen 1995).

3.1.6. *Classifier-based models*

Systems using a classifier as a user profile learning technique retain the structure of the classifier as a profile. This is the case in neural networks, decision trees, inducted rules and Bayesian networks.

A neural network is a network of input and output cells, based upon neuron functions in the brain. Neural networks create a compact representation that responds to queries quickly. However, they can be slow to train. For example, Re:Agent (Boone 1998) filter e-mails through a neural network previously trained with feature vectors of past messages.

A decision tree is another way to classify data. It consists of a set of nodes and a set of directed edges that connect the nodes (tree structure). The internal nodes represent questions about the parameters, and the edges represent answers to those questions, i.e. values for the parameters. The leaf nodes represent a final decision. For example, InfoFinder (Krulwich and Burkey 1996) recommends documents based on a decision tree.

Association rules have been used for many years in merchandizing, both to analyze patterns of preference across products, and to recommend products to consumers based on other products they have selected. Association rules can form a compact representation of preference data which improve efficiency of storage as well as performance. For example, an association rule expresses the relationship that a certain movie is often purchased along with others (Basu et al. 1998).

A Bayesian network is a directed acyclic graph in which nodes represent propositional variables and arcs represent dependencies (Jensen 1996). A node's value is a function of the values of the nodes it depends upon. Leaf nodes represent propositions, which can be determined by observation. The resulting model is very small, very fast and essentially as accurate as nearest neighbors methods (Breese et al. 1998).

3.1.7. *User-item ratings matrix*

Some collaborative filtering systems maintain a user-item ratings matrix as a user profile. The user-item ratings matrix contains historical user ratings on items. Each cell (u,i) of the matrix contains a rating representing the evaluation of the user u to the item i, and an empty value if there is no evaluation.

These systems do not use any learning profile technique (see Section 3.3.1) but bring together all the processes in the user profile matching techniques (see Section 4.3).

3.1.8. *Demographic features*

Demographic filtering systems create a user profile through stereotypes. Therefore, the user profile representation is a list of demographic features which represent the kind of user. None of these systems use any learning profile technique (see Section 3.3.1) but they bring together all the processes in stereotype reasoning (Kobsa et al. 2001).

3.2. *Initial profile generation*

It is desirable to learn as much as possible from the user so that the recommender agents provide satisfactory results from the very beginning. However, the user is not usually willing to spend much time in defining his interests to create his profile. Moreover, users' interests may change over time, making the profiles difficult to maintain. For these reasons, starting up and maintaining user profiles is a difficult aspect in the design and development of intelligent agent systems. The degree of automation in the acquisition of user profiles can range from manual input, to semi-automatic procedures (stereotyping and training sets), to the automatic recognition by the agents themselves. Table 4 shows the initial profile generation techniques used by the different systems analyzed.

3.2.1. *Empty*

Some systems do not bother with the initial profile and start with an empty profile structure (e.g., Chen and Sycara 1998; Balabanovic and Shoham

Table 4. Initial profile generation technique of the systems

NAME	TECHNIQUE
ACR News	Training set
Amalthaea	Manual
Amazon	Empty
Anatagonomy	Empty
Beehive	Empty
Bellcore Video Recom	Training set
Casmir	Unknown
CDNow	Empty
Fab	Empty
GroupLens	Empty
ifWeb	Training set, stereotyping
InfoFinder	Training set
INFOrmer	Training set
Krakatoa Chronicle	Empty
LaboUr	Training set
Let's Browse	Training set
Letizia	Empty
LifeStyle Finder	Stereotyping
MovieLens	Training set
News Dude	Training set
NewsWeeder	Training set
NewT	Training set
Personal WebWatcher	Manual
PSUN	Training set
Re:Agent	Manual, training set
Recommender	Training set
Ringo/FireFly	Training set
SIFT Netnews	Training set
SiteIF	Empty
Smart Radio	Training set
Syskill & Webert	Manual, stereotyping
Tapestry	Empty
Webmate	Empty
WebSail	Empty
WebSell	Empty
Websift	Training set
WebWatcher	Manual

1997; Cunningham et al. 2001). There is no initial phase, the profile structure is filled through an automatic recognition method when the user begins interaction with the system.

3.2.2. *Manual*

A manual system asks users to register their interests in the form of keywords, topics and so on. One of the advantages of this method is the transparency of the system behavior. When items have been delivered to a user, he/she can usually easily guess why each item was delivered. One problem with this method though, is that it requires much effort on the part of the user. Another problem is that people cannot necessarily specify what they are interested in, because their interests are sometimes still unknown.

3.2.3. *Stereotyping*

Stereotyping is based on the fact that creating an initial model is, in a sense, a classification problem, aimed at generating initial predictions about the user (Kobsa et al. 2001). The user model is initiated by classifying users in stereotypical descriptions (Rich 1979), representing the features of classes of users. The use of stereotypes in computer systems for maintaining models of their users was introduced by Rich with the Grundy system. Typically, the data used in the classification is demographic and the user is asked to fill out a registration form: record data (name, address, etc.), geographic data (area code, city, etc), user characteristics (age, sex, etc.), psychographic data (e.g., lifestyle), etc.

An example is the method implemented by Krulwich in the LifeStyle Finder (Krulwich 1997) which uses a commercially available database of demographic data which encompasses the interests of people nationwide.

The main shortcoming of this technique is the difficulty of acquiring personal data from the users. Internet users normally avoid engaging in a relationship with Internet sites. This is mostly due to a lack of faith in the privacy policy of today's web sites. Normally, users either withhold personal data or provide false data.

3.2.4. *Training set*

The training set is a collection of user interaction examples which is used to infer the initial user profile. One practical way to establish the training set is to ask the user to rate some concrete examples as relevant or irrelevant to their interests (e.g., Sorensen and McElligot 1995; Boone 1998). A similar approach is to ask the user to rate a set of predefined examples (e.g., Good et al. 1999; Shardanand 1994). In both cases, once the user has given the appropriate information, the system processes the data with one of the learning techniques explained in Section 3.3.

This mode has the advantage of simplified handling. It has the disadvantage, and the danger, that someone has to select the examples which are not always representative and the results are less precise. Some of the systems using this technique are ACR News (Mobasher et al. 2000); FireFly (Shardanand and Maes 1995) and LaboUr (Schwab et al. 2001).

3.3. *Profile learning techniques*

The previous section described sources of information potentially representative of user interests, mainly the training sets. Profile learning techniques build a user profile through these data. These techniques can be seen as a preliminary step in representing a user profile.

It is important to note that when the learning data is composed of text without structure, it is necessary to pre-process the information in order to get structured, relevant information. Some systems simply use an information indexing technique to build a profile and represent it as a structure of indexed words, although information indexing techniques cannot be considered artificial intelligence techniques.

Some systems have an off-line phase during which they learn a model of a user behavior, and then an online phase during which they apply the model in real time. Most systems, however, use a lazy learning approach (online), in that they build and update the model while making recommendations in real time. Offline learning methods may prove practical for environments in which knowledge of consumer preferences changes slowly with respect to the time needed to build the model, but they are not suitable for environments in which consumer preference models must be updated rapidly or frequently.

In this section, we will first briefly explain some typical systems for which profile learning techniques are not necessary. Then, we summarize the information retrieval techniques used to preprocess information. Finally, we look at the most commonly used profile learning techniques are reviewed: clustering and classifiers. Table 5 shows the profile learning techniques used by the different systems analyzed.

3.3.1. *Not necessary*

Some systems keep information directly acquired from the system as a user profile, therefore, they do not need a profile learning technique. There are three main kinds of systems do not need a profile learning method:

- Systems which acquire user profile information from a database. For instance, electronic commerce systems (Amazon 2001; CDNow 2001; Cunningham et al. 2001) which extract the information from a database of products and keep a purchase list as a profile (see Section 3.1.1).

Table 5. Profile learning technique of the systems

NAME	TECHNIQUE
ACR News	Induction rule learning, clustering
Amazon	Not necessary
Amalthaea	Feature selection (stemming), information indexing (TF-IDF)
Anatagonomy	Information indexing (TF-IDF)
Beehive	Clustering
Bellcore Video Recom	Not necessary
Casmir	Simple positive reinforcement, simple positive reinforcement with query keyword overriding, positive and negative reinforcement, positive and negative reinforcement with query keyword overriding
CDNow	Not necessary
Fab	Information indexing (TF-IDF)
GroupLens	Not necessary
ifWeb	Feature selection (stop-words, stemming, . . .)
InfoFinder	Feature selection (heuristics), decision tree (ID3)
INFOrmer	Feature selection (stop-words, stemming, . . .)
Krakatoa Chronicle	Information indexing (TF-IDF)
LaboUr	Information indexing (Boolean)
Let's Browse	Information indexing (TF-IDF)
Letizia	Information indexing (TF-IDF)
LifeStyle Finder	Not necessary
MovieLens	Information indexing (TF-IDF), induction rule learning (Ripper)
News Dude	Short-term: information indexing (TF-IDF), information indexing (Boolean)
NewsWeeder	Information indexing (TF-IDF), MDL
NewT	Feature selection (stop-words, stemming), information indexing (TF-IDF)
Personal WebWatcher	Information indexing (TF-IDF)
PSUN	Feature selection (stemming), n-gram induction
Re:Agent	Feature selection (stop-words), information indexing (TF-IDF), clustering, neural network
Recommender	Induction rule learning (Ripper)
Ringo/FireFly	Not necessary
SIFT Netnews	Information indexing (Boolean), information indexing (TF-IDF)
SiteIF	Feature selection (stop-words, stemming, . . .)
Smart Radio	Not necessary
Syskill & Webert	Feature selection (stop-words), information indexing (Boolean), information indexing (TF-IDF), decision tree (ID3)
Tapestry	Not necessary
Webmate	Information indexing (TF-IDF)
WebSail	Information indexing (TF-IDF)
WebSell	Not necessary
Websift	Induction rule learning
WebWatcher	Information indexing (TF-IDF), Winnow, WordStat, random

- Collaborative filtering systems (Goldberg et al. 1992; Resnick et al. 1994; Shardanand and Maes 1995) which keep a matrix with the user-item ratings as a profile (see Section 3.1.7).
- Systems which create an initial profile through stereotyping (see Section 3.2.3) and do not modify it (Krulwich 1997). This is the case in demographic filtering systems (see Section 4.1.1).

Systems that do not need a profile learning technique concentrate information filtering tasks on the profile-item (see Section 4.2) or profile-profile (see Section 4.3) matching techniques.

3.3.2. *Structured information retrieval techniques*

When information has no structure (e.g. text), some kind of pre-processing step is needed to extract structured relevant information. Typically, this process comprises two main steps: feature selection and information indexing.

Feature selection can be achieved through different approaches which reduce the number of words: stop-words, pruning, stemming, etc (see Salton and McGill 1983).

Information indexing uses the frequency word occurrence to calculate the potential relevance of an item. TF-IDF is one of the most successful and best-tested techniques. A document is represented as a vector of weighted terms (see Section 3.1.2). The computation of the weights reflects empirical observations regarding text. Terms frequently appearing in a document (TF = term-frequency), but rarely on the outside (IDF = inverse-document-frequency), are more likely to be relevant to the topic of the document.

TF-IDF has two popular variants: the boolean method and the probabilistic method. The boolean method is a simplistic approach where the profile is represented as a vector of words with a boolean value indicating their presence in the text. The probabilistic method is a generalization of the exact match technique. In this method, documents are ranked by the probability that they satisfy the information need rather than by making a shape decision. Bayesian inference networks have proven to be a useful technique for computing this probability (see Section 3.3.4).

3.3.3. *Clustering*

The basic idea of this technique is clustering similar user information into groups based on data. Then, these clusters are matched against actual information to conclude whether it is interesting or not.

For example, in ACR News (Mobasher et al. 2000) user transactions are mapped into a multi-dimensional space as vectors of URL references. This space is partitioned into clusters (*usage clusters*) representing a group of transactions that are similar, based on co-occurrence patterns of URL

references. Finally, clusters are matched against an active user session to recommend interesting URLs.

Traditional collaborative filtering techniques are often based on matching the current user profile against clusters of similar profiles obtained by the system over time from other users (see Section 4.3.1.2).

3.3.4. *Classifiers*

Classifiers are general computational models for assigning a category to an input. To build a recommender system using a classifier means using information about the item and the user profile as input, and having the output category represent how strongly to recommend an item to the user. Classifiers may be implemented using many different machine learning strategies including neural networks, decision trees, association rules and Bayesian networks.

Learning in neural networks is achieved by training the network with a set of data. Each input pattern is propagated forward through the network and active output cells represent the interest of the user. When an error is detected, it is propagated backward adjusting the cell parameters to reduce the error, thus achieving learning. For instance, Jennings and Higuchi employed a neural network for constructing a user's profile (Jennings and Higuchi 1993).

Decision tree learning is a method for approximating discrete-valued target functions, in which the learned function is represented by a decision tree. The learned trees can also be represented as a set of if-then rules. Decision tree learners build a decision tree by recursively partitioning examples into subgroups, obtaining source classes of items which can be classified, for example, into *interesting* and *not interesting* (Krulwich and Kurkey 1995). The most widely-used decision tree learner applied to profiling is the ID3 (Quinlan 1983).

The discovery of association rules by inductive learning (Mobasher et al. 2000) is one of the best-known classifier examples. The association rule discovery methods initially find groups of items occurring frequently together in many transactions. Such groups of items are referred to as frequent item sets. Association rules capture the relationships among these items based on their patterns of co-occurrence across transactions. Association rules can form a very compact representation of preference data which may improve efficiency of storage as well as performance. Some examples of inductive learning techniques are Ripper (Cohen 1995b), Slipper (Cohen and Singer 1999), CN2 (Clark and Niblett 1989) and C4.5rules (Quinlan 1994).

A Bayesian network learner algorithm is applied to a set of training data, searching over various model structures in terms of dependencies for each item. In the resulting network, each item will have a set of parent items that are the best predictors of its votes. The model can be built off-line in a matter of hours. Thus, this technique may prove practical for environments in which

knowledge of consumer preferences changes slowly with respect to the time needed to build the model.

3.4. *Relevance feedback*

Human interests change as time passes. For example, a new father may be interested in infant care just after childbirth, but this interest gradually decreases over time. Therefore, the recommender agent needs up-to-date information to update the user profile automatically. In this section, several ways to obtain this information, which we call relevance feedback, are presented. Then, in Section 3.5 we will see how to use this information to update user profiles.

Typically, it is possible to distinguish two kinds of relevance feedback: positive information (items liked by the user) and negative information (i.e., inferring features which the user is not interested in (Holte and Yan 1996)). The authors claim that negative information produces a dramatic improvement in the system's performance. However, there are a few systems which cannot take into account negative information because the system's accuracy is likely to decrease (e.g., Schwab et al. 2000). So, it all depends on the system.

The two most common ways to obtain relevance feedback is to use information given explicitly or to get information observed implicitly from the user's interaction. Moreover, some systems propose implicit/explicit hybrid approaches. Table 6 shows the relevance of feedback techniques used by the different systems analyzed.

3.4.1. *No feedback*

Some systems do not update the user profile automatically and, therefore, they do not need relevance feedback. For example, all the systems which update the user profile manually (see Section 3.5.1). Of course, systems which never modify the profile do not need relevance feedback either.

For instance, SIFT Netnews (Yan and Garcia-Molina 1995) creates an initial profile of the user and does not update it automatically over time. However, the user can modify his profile manually.

3.4.2. *Explicit feedback*

In several systems, users are required to explicitly evaluate items. These evaluations indicate how relevant or interesting an item is to the user, or how relevant or interesting the user thinks an item is to other users (Rich 1979).

There are three main approaches to get explicit relevance feedback: like/dislike (e.g., Chen et al. 2000; Billsus and Pazzani 1999), ratings (e.g., Shardanand and Maes 1995; Moukas 1997) and text comments (e.g., Resnick et al. 1994; Goldberg 1992). In like/dislike systems, users are required to

Table 6. Relevance feedback technique of the systems

NAME	TECHNIQUE
ACR News	Implicit (navigation history)
Amazon	Explicit (ratings), implicit (purchase history)
Amalthaea	Explicit (ratings)
Anatagonomy	Explicit (ratings), implicit (scrolling, enlarging)
Beehive	Implicit (mail history)
Bellcore Video Recom	Explicit (ratings)
Casmir	Explicit (ratings)
CDNow	Explicit (ratings), Implicit (purchase history)
Fab	Explicit (ratings)
GroupLens	Explicit (ratings, text comments), implicit (time spent)
ifWeb	Explicit (ratings)
InfoFinder	Explicit (ratings)
INFOrmer	Explicit (ratings)
Krakatoa Chronicle	Explicit (ratings), implicit (saving, scrolling, time spent, maximizing, resizing, peeking)
LaboUr	Implicit (links, time spent)
Let's Browse	Implicit (links, time spent)
Letizia	Implicit (links, time spent)
LifeStyle Finder	Explicit (ratings), implicit (purchase history)
MovieLens	Explicit (ratings)
News Dude	Explicit (like/dislike, I already know this, tell me more)
NewsWeeder	Explicit (ratings)
NewT	Explicit (like/dislike)
Personal WebWatcher	Implicit (links)
PSUN	Explicit (ratings)
Re:Agent	No feedback
Recommender	Explicit (ratings)
Ringo/FireFly	Explicit (ratings)
SIFT Netnews	Explicit (like/dislike)
SiteIF	Implicit (links)
Smart Radio	Explicit (ratings), implicit (saving)
Syskill & Webert	Explicit (ratings)
Tapestry	Explicit (like/dislike, text comments), implicit (forwarding)
Webmate	Explicit (like/dislike)
WebSail	Explicit (like/dislike)
WebSell	Explicit (unknown)
Websift	Implicit (navigation history)
WebWatcher	Explicit (goal reached), implicit (links)

explicitly judge items on a binary scale, i.e., classify an object as "*interesting*" or "*not interesting*", as "*relevant*" or "*not relevant*" or as "*like*" or "*hate*". The ratings approach requires users to provide a judgement on a discrete scale. The rating scale is typically numeric (e.g., the web bookstore Amazon.com offered users the opportunity of rating books in various categories on a 5-point scale) or symbolic with mapping to a numeric scale (e.g., in Syskill & Webert (Pazzani et al. 1996) users have the possibility of rating a Web page as "*hot*", "*lukewarm*", or "*cold*"). Finally, several sites encourage textual comments from their users (e.g., Grouplens (Resnick et al. 1994) and Tapestry (Goldberg et al. 1992)). Systems gather comments about a single item and present them to the users as a means of facilitating the decision-making process. While textual comments are helpful, they require a fair amount of processing by the targeted user. Users must read text and interpret to what degree it is positive or negative.

Explicit feedback has the advantage of simplicity. Several papers have demonstrated the high performance of systems using explicit relevance feedback (Salton and Buckley 1990; Buckley and Salton 1995). However, in practical applications, explicit feedback has three serious drawbacks:

- First, the relevance of information is always relative to the changing information need of a user, and information relevance judgements of individual items are typically assumed to be independent when, in fact, they are not (e.g., the third article presented on the same topic may simply be rated lower because the first two items satisfied information need and the user is judging incremental relevance at this point).
- Another problem is that numeric scales may not be adequate for describing the reactions humans have to items.
- The last problem is that computer users do not supply enough feedback, particularly negative feedback. Pazzani et al. report that only 15% of users would supply feedback even though they were encouraged to do so (Pazzani and Billsus 1997). Users are generally very reluctant to perform actions not directed towards their immediate goals if they do not receive immediate benefits, even when they would profit in the long run (Carroll 1987).

3.4.3. *Implicit feedback*

Implicit feedback means that the system automatically infers the user's preferences passively by monitoring the user's actions. Chatterjee et al. prove empirically in 1998 that the user's interests can be inferred from his behavior. Their results are important because motivating web consumers to provide personal data in an explicit way is proving very difficult. Conclusions about the user's interests should therefore not rely on user explicit feedback very

much, but rather take passive observations about users into account as far as possible.

Implicit feedback was defined some years ago by Rich (1979), and the first system was implemented by Mitchell et al. (1985). Since then, many systems have implemented implicit user feedback in their approaches (e.g., Stefani and Strappavara 1998; Schwab et al. 2001) and some systems have even combined it with explicit feedback (see hybrid approach in Section 3.4.4).

Most implicit methods obtain relevance feedback by analyzing the links followed by the user (e.g., Lieberman 1995; Mladenic 1996), by a history of purchases (e.g., Amazon 2001; CDNow 2001; Krulwich 1997), by the navigation history (e.g., Cooley et al. 1999; Mobasher et al. 2000), by e-mail boxes (e.g., Huberman and Kaminsky 1996) and by the time spent on a particular web page (e.g., Morita and Shinoda 1994; Konstan et al. 1997; Kobsa et al. 2001; Sakagami et al. 1997).

There are many other examples of confirmatory actions. For documents like Web pages, news articles or e-mail messages, it is interesting to find out if the user takes any further processing action, such as saving a document (Kamba et al. 1995), printing a document, bookmarking a Web page, deleting a document, replying to or forwarding an e-mail (Goldberg et al. 1992), or scrolling, maximizing, minimizing or resizing the window containing the document or the Web page (Kamba et al. 1995; Sakagami et al. 1997). Since these actions are performed under the control of the application, they can be registered and evaluated to learn the user's profile.

However, Kobsa et al. (2001) do not recommend a universal logging of usage data on the micro-interaction level, such as the tracking of mouse movements within applets, unless the purpose of the login has already been specified (e.g., for determining user's interest in page segments (Sakagami et al. 1997)). The amount of data collected is very large, the computation needed to derive recommendations for adaptations is extensive, and the confidence in the suitability of these adaptations is likely to be relatively low. However, experimentation with such data in smaller, laboratory contexts to drive the development of new methods in the area of implicit feedback seems promising.

3.4.4. *Hybrid approach*

The limited evidence available on implicit feedback suggests that it has great potential but its effectiveness remains unproven. As is common in many technologies, the best performing system results in combining several existing technologies. In this field, implicit feedback can be combined with explicit feedback systems in a hybrid system. Providing implicit feedback greatly decreases the user's efforts, whereas providing explicit feedback helps the system to infer user preferences accurately.

One approach with this combination involves using implicit data as a check on explicit ratings (Nichols 1997). For instance, if a user is explicitly rating an item, then there should be some implicit data to confirm that he has actually examined it. If there is no evidence to suggest this, then perhaps its rating should be ignored or reduced in importance. Conversely, an evaluation with a relatively long "examine time" may be increased in importance.

A different case is Anatagonomy (Sakagami et al. 1997). Giving explicit feedback is optional and should only be used when users wish to show explicit interest. WebWatcher (Joachims et al. 1997), LifeStyle Finder (Krulwich 1997), Krakatoa Chronicle (Kamba et al. 1995), GroupLens (Resnick et al. 1994), CDNow (CDNow 2001) and Amazon (Amazon 2001) also use hybrid relevance feedback.

3.5. *Profile adaptation techniques*

Since recommender agents typically involve interaction over long periods of time, user interests cannot be assumed to stay constant. This normally means that the most recent observations gathered through what we have called relevance feedback represent the user's current interests better than older ones. Therefore, there is a need for a technique that will adapt the user profile to new interests and forget old ones.

There are several approaches to this: manually (simply adding the new information), with a time window, aging examples, combining a short-term and a long-term model, a gradual forgetting function or the natural selection for ecosystems of agents. Table 7 shows the profile adaptation techniques used by the different systems analyzed.

3.5.1. *Manual*

In some systems, the user has to change the profile when he/she is interested in updating it. For instance, in Sift Netnews (Yan and Garcia-Molina 1995), when the user wants to include/exclude one of the interests contained in his profile, he has to modify it manually.

As in manual initial profile generation (see Section 3.2.2), this approach has two important problems: it requires much effort on the part of the user and people cannot necessarily specify what they are interested in because their interests are sometimes still unknown. Therefore, manual updating turns out to be difficult when requirements change quickly.

3.5.2. *Add new information*

This approach is the most commonly used in current systems. The idea is to update the user profile adding new information extracted from the user

Table 7. Profile adaptation technique of the systems

NAME	TECHNIQUE
ACR News	Add new information
Amalthaea	Natural selection, gradual forgetting
Amazon	Add new information
Anatagonomy	Add new information
Beehive	Add new information
Bellcore Video Recommender	Add new information
Casmir	Add new information
CDNow	Add new information
Fab	Natural selection
GroupLens	Add new information
ifWeb	Gradual forgetting
InfoFinder	Add new information
INFOrmer	Add new information
Krakatoa Chronicle	Add new information
LaboUr	Gradual forgetting
Let's Browse	Add new information
Letizia	Add new information
LifeStyle Finder	Add new information
MovieLens	Add new information
News Dude	Short-term and long-term models
NewsWeeder	Add new information
NewT	Natural selection
Personal WebWatcher	Add new information
PSUN	Natural selection
Re:Agent	Manual
Recommender	Add new information
Ringo/FireFly	Add new information
SIFT Netnews	Manual
SiteIF	Gradual forgetting
Smart Radio	Add new information
Syskill & Webert	Add new information
Tapestry	Add new information
Webmate	Add new information
WebSail	Add new information
WebSell	Add new information
Websift	Add new information
WebWatcher	Add new information

relevance feedback. Thus, the profile is adapted to the new user's interests. The main drawback, however, is that old interests are not forgotten.

3.5.3. *Gradual forgetting function*

The concept of gradual forgetting was introduced by Webb and Kuzmycz in 1996 with the main idea being that natural forgetting is a gradual process. Therefore, a gradual forgetting function can be defined. It should produce a weight for each observation according to its location in the course of time. Webb and Kuzmycz suggest a data aging mechanism which places an initial weight of 1 on each observation. A set proportion discounts the weight of every observation each time another relevant observation is incorporated into the model. Thus, the most recent observations become more "important", assuming they better represent the current users' interests than the older ones. Hence, the system becomes more noise resistant without losing its sensitivity to real changes in interest (Schwab et al. 2001).

Koychev proposes a linear gradual forgetting function (Koychev 2000), but it can be approximated with any function (e.g., logarithmic or exponential).

A particular case of the gradual forgetting function is the time window approach. It is the most frequently-used approach in dealing with the problem of forgetting old interests. It consists of learning the description of the user's interests from only the latest observations. The training examples are selected from a so-called time window, i.e. only the last examples are used for training (Mitchell et al. 1994). An improvement on this approach is the use of heuristics to adjust the size of the window according to the current predictive accuracy of the learning algorithm (Widmer and Kubat 1996).

Maloof and Michalski implemented a variation of the time window approach (Maloof and Michalski 2000). Instances older than a certain given age are deleted from the partial memory. Like the time window, the system only takes into account the last examples. However, this approach does forget observations outside the given window or older than a certain age.

3.5.4. *Natural selection*

The natural selection approach is associated with systems implementing an ecosystem architecture of agents based on genetic algorithms. An ecosystem of specialized agents competing in parallel, gives recommendations to the user. The ecosystem evolves in the following way: agents producing the best results are reproduced with the crossover and mutation operators and others are destroyed. Amalthaea (Moukas 1997), Fab (Balabanovic and Shoham 1997), NewT (Sheth and Maes 1993) and PSUN (Sorensen and McElligot 1995) use this approach.

4. Profile Exploitation

To recommend either products or actions to a user, an intelligent recommender agent makes decisions according to the information available (items, profiles of other agents on the web, etc). Thus, it is vital to select the most appropriate information with which to make decisions. Information filtering methods are based on user profile-item matching techniques and user profile matching techniques. So, regarding exploitation, three main dimensions characterize intelligent recommender agents: the information filtering method (demographic, content-based and collaborative), the item-profile matching (when content-based) and the user profile matching techniques (when collaborative). See Figure 2 for a general view.

4.1. *Information filtering methods*

There are three main information filtering methods: demographic, content-based and collaborative. Table 8 shows the information filtering techniques used by the various systems analyzed.

4.1.1. *Demographic filtering*

Demographic filtering approaches use descriptions of people to learn the relationship between a single item and the type of people who like it. The user profiles are created by classifying users in stereotypical descriptions (Rich 1979), representing the features of classes of users. Personal data about the user is required and is used to classify users in terms of these demographic data. Classifications are used as general characterizations for the users and their interests. Commonly, the personal data is asked of the user in a registration form (see Section 3.2.3). The resulting profiles span the range of information contained in the demographic database.

For instance, the method implemented by Krulwich in the LifeStyle Finder (Krulwich 1997) uses a demographic system called PRIZM from Claritas Corporation which divides the population of the United States into 62 demographic clusters according to their purchasing history, lifestyle characteristics and survey responses.

A demographic filtering system has two principal shortcomings:

- Demographic filtering is based on a generalization of the user's interests, so the system recommends the same items to people with similar demographic profiles. As every user is different, these recommendations prove to be too general.
- The demographic approaches do not provide any individual adaptation to interest changes. The user's interests tend to shift over time (Koychev 2000), so the user profile needs to adapt to change.

Table 8. Information filtering method of the systems

NAME	METHOD
ACR News	Content-based filtering
Amazon	Hybrid
Amalthaea	Content-based filtering
Anatagonomy	Hybrid
Beehive	Collaborative filtering
Bellcore Video Recom	Collaborative filtering
Casmir	Hybrid
CDNow	Hybrid
Fab	Hybrid
GroupLens	Collaborative filtering
ifWeb	Content-based filtering
InfoFinder	Content-based filtering
INFOrmer	Content-based filtering
Krakatoa Chronicle	Hybrid
LaboUr	Hybrid
Let's Browse	Content-based filtering
Letizia	Content-based filtering
LifeStyle Finder	Demographic filtering
MovieLens	Hybrid
News Dude	Content-based filtering
NewsWeeder	Hybrid
NewT	Content-based filtering
Personal WebWatcher	Hybrid
PSUN	Content-based filtering
Re:Agent	Content-based filtering
Recommender	Hybrid
Ringo/FireFly	Collaborative filtering
SIFT Netnews	Content-based filtering
SiteIF	Content-based filtering
Smart Radio	Collaborative filtering
Syskill & Webert	Content-based filtering
Tapestry	Collaborative filtering
Webmate	Content-based filtering
WebSail	Content-based filtering
WebSell	Hybrid
Websift	Hybrid
WebWatcher	Hybrid

Nevertheless, demographic information can be a useful technique if combined with other approaches.

4.1.2. *Content-based filtering*

Content-based filtering approaches recommend items for the user based on the descriptions of previously evaluated items. In other words, they recommend items because they are similar to items the user has liked in the past. User profiles are created using features extracted from these items (see Section 3.3) and each user is assumed to operate independently.

The input data most often take the form of samples of the user's interests or preferences in a given area, and the profile is a generalization of these data which can be used generatively to carry out tasks on behalf of the user. These profiles are then used to find or recognize other items likely to be of interest. Different methods are used by the systems to match a user profile with new items and decide whether they are interesting to the user (see Section 4.2).

In Syskill & Webert (Pazzani et al. 1996), the user rates a number of Web documents from a content domain on a binary "hot" and "cold" scale. Based on these ratings, it computes the probabilities of words being in hot or cold documents. Lieberman developed the system Letizia (Lieberman 1995), which assists a user in Web browsing. Letizia tries to anticipate interesting items on the Web which are related to the user's current navigation context. For a set of links, Letizia computes a preference ranking based on a user profile. This profile is a list of weighted keywords, each one indicating the relevance of the words found on the pages. Personalized WebWatcher (Mladenic 1996) observes the individual user's choices of links on Web pages in order to recommend links on other Web pages he/she may visit later. The user does not have to provide explicit ratings. Instead, visited links are taken as positive examples, non-visited links as negative ones.

A purely content-based filtering system has several shortcomings:

- Content-based approaches are based on objective information about the items. This information is automatically extracted from various sources (e.g., Web pages) or manually introduced (e.g., product database). However, selecting one item or another is based mostly on subjective attributes of the item (e.g., a well-written document or a product with a spicy taste). Therefore, these attributes, which better influence the user's choice, are not taken into account.
- Another problem, which has been studied extensively, is over-specialization. Content-based filtering techniques have no inherent method for generating serendipitous finds. The system recommends more of what the user has already seen and indicated a liking for. When the system can only recommend items scoring highly against a user

profile, the user is restricted to seeing items similar to those already rated. In practice, additional hacks are often added to introduce some element of serendipity, in effect injecting a note of randomness.

– With the pure content-based approach, a user's own ratings are the only factor influencing future performance. However, only a few ratings are provided due to both the reluctance of users to perform actions not directed towards their immediate goals when immediate benefits are not forthcoming (Carroll and Rosson 1987), and the low interaction of the user with the system. Therefore, the recommendation quality is not very precise.

Nevertheless, these shortcomings can be solved by combining the content-based approach with the collaborative filtering approach (see hybrid filtering method in Section 4.1.4).

4.1.3. *Collaborative filtering*

The collaborative filtering technique matches people with similar interests and then makes recommendations on this basis. Recommendations are commonly extracted from the statistical analysis of patterns and analogies of data extracted explicitly from evaluations of items (ratings) given by different users or implicitly by monitoring the behavior of the different users in the system. This approach is very different from content-based filtering, the other most commonly used approach. Rather than recommending items because they are similar to items a user has liked in the past, items are recommended based on other user's preferences. Rather than computing the similarity of items, the similarity among users is computed. In collaborative filtering a user's profile consists simply of the data the user has specified. This data is compared to those of other users to find overlaps in interests among users. These are then used to recommend new items. Typically, for each user a set of "nearest neighbors" is defined using the correlation between past ratings. Scores for unseen items are predicted using a combination of the scores from the nearest neighbor. This approach requires less computation than the previous one because it doesn't have to reason with the user data and it clearly leverages the commonalties between users.

Tapestry (Goldberg et al. 1992) is one of the earliest implementations of collaborative filtering based recommender systems. This system relied on the explicit opinions of people from a close-knit community, such as an office workgroup. Another popular system is GroupLens (Konstan et al. 1997), which computes correlation between readers of Usenet newsgroups by comparing their ratings of news articles. The ratings of an individual user are used to find other users with similar ratings, and their ratings are processed to predict the user's interest in new articles.

Terveen and Hill (2001) claim three essentials are needed to support collaborative filtering: many people must participate (increasing the likelihood that any one person will find other users with similar preferences), there must be an easy way to represent a user's interests in the system, and the algorithms must be able to match people with similar interests. These three elements are not that easy to develop and produce the main shortcoming of collaborative filtering systems:

- The early-rater problem: when a new item appears in the database there is no way it can be recommended to a user until more information is obtained through another user either rating it or specifying which other items it is similar to.
- The sparsity problem: the goal of collaborative filtering systems is to help people focus on reading documents (or consuming items) of interest. As with the previous shortcoming, if the number of users is small relative to the volume of information in the system (because there is a very large or rapidly changing database) there is a danger of the coverage of ratings becoming too sparse, thinning the collection of recommendable items. Also, sparsity poses a real computational challenge as it becomes harder to find neighbors and harder to recommend items since too few people have given ratings.
- Another logic problem is that for a user whose tastes vary from the norm there will not be any other users who share his or her particular likes and dislikes, leading to poor recommendations.
- The difficulty of achieving a critical mass of participants makes collaborative filtering experiments expensive. Collaborative filtering systems require data from a large number of users before being effective as well as requiring a large amount of data from each user while limiting their recommendations to the exact items specified by those users.
- The critical dependency on the size and composition of the user population also influences a user's group of nearest neighbors. In a situation in which feedback fails to cause this group of nearest neighbors to change, expressing dislike for an item will not necessarily prevent the user from receiving similar items in the future. Furthermore, the lack of access to the content of items prevents similar users from being matched unless they have rated the exact same items.

Herlocker et al. also introduced the problem of lack of transparency in the collaborative filtering systems (Herlocker et al. 2000). Collaborative systems today are black boxes, computerized oracles which give advice but cannot be questioned. A user is given no indicators to consult in order to decide when to trust a recommendation and when to doubt one. These problems have prevented acceptance of collaborative systems in all but low-risk content domains since they are untrustworthy for high-risk content domains.

Nevertheless, these shortcomings can be solved by combining the collaborative filtering approach with the content-based filtering approach (see hybrid approach in next section).

4.1.4. *Hybrid approach*

Hybrid systems exploit features of content-based and collaborative filtering, since they will almost certainly prove to be complementary. On the one hand, purely collaborative systems solve the shortcomings of the purely content-based systems. The first shortcoming of content-based systems is the lack of subjective data about the items. In a collaborative system, the community of users can offer this kind of data explicitly. Subjective data can be an opinion of one item offered by a trusted friend. For instance, you can buy a spicy product because a user with similar tastes has recommended it to you. Another shortcoming of content-based systems is the lack of novelty. A perfect content-based technique would never find anything novel, limiting the range of applications for which it would be useful. Collaborative filtering techniques excel at identifying novelty using other users' recommendations and you can receive items dissimilar to those seen in the past. Finally, content-based systems lack user ratings to represent the user's interests. Collaborative systems can complete the user information with another user's experience as a basis. For instance, if you are very similar to another user and you have not rated a product, the system can use the other user's ratings to complete your interests.

On the other hand, purely content-based systems solve the shortcomings of the purely collaborative systems, the first of which is the early-rater problem. With content-based methods, new items can be recommended on the basis of their content, without the need for explicit ratings. Another advantage is that content-based systems can recommend to a user with unusual tastes without the need for a similar user, eliminating the sparcity problem of collaborative approaches. Finally, the number of participants is not important in content-based systems because they do not depend on population.

Thus, both content-based and collaborative filtering contribute to the other's effectiveness, avoiding the limitations mentioned for each system and allowing an integrated system to achieve both reliability and serendipity. Several papers have attested to the high performance of hybrid systems (e.g., Pazzani 1999; Good et al. 1999).

Fab (Balabanovic and Shoham 1997), LaboUr (Schwab et al. 2001) and WebSell (Cunningham et al. 2001) propose a very simple method for combining the two approaches: user profiles based on content analysis are maintained and closely compared to determine users with similar preferences for collaborative recommendation.

4.2. *User profile – item matching*

Typically, the user profile is used to recommend new items considered relevant to the user. Content-based filtering systems use direct comparison between the user profile and new items. Thus, a user profile-item matching technique is needed. Several techniques are studied here, whose aims are to automate the process of classifying items as relevant/not relevant, by computing comparisons between the representation of the user's interests and the representation of the items.

In the systems we analyzed, the user profile – item matching techniques used are: a simple keyword matching, the cosine similarity, the nearest neighbor and typical classifiers. Table 9 shows the user profile – item matching techniques used by the different systems analyzed.

4.2.1. *Standard keyword matching*

Standard keyword matching consists of a simple count of the terms which are present simultaneously in the current description of the new item and in the user profile. However, this model has some problems with the synonymy and plural meanings of some words.

An example is SiteIF (Stefani and Strappavara 1998) which implements a standard keyword matching algorithm that consists of checking, for every word in the representation of the document, whether the context in which it occurs has been already found in previously visited documents and already stored in the semantic network.

4.2.2. *Cosine similarity*

Cosine similarity comes from information retrieval research and is used in systems with simple user profile representation (Salton and Buckley 1988; Buckley et al. 1996; Yan and Garcia-Molina 1995; Chen et al. 2000). An early similarity formula was used by Salton in the SMART system (Salton and McGill 1983). Salton treated the index (user profile) and the search query (new item) as n-dimensional vectors (see Section 3.1.2). The cosine formula calculates the cosine of the angle between the two vectors. As the cosine approaches "1", the two vectors become coincident. If the two vectors are totally unrelated, they will be orthogonal and the value of the cosine is "0". Moreover, the square of the cosine of the angle (easily computed as the normalized inner product of the two vectors) can be used to rank the items.

4.2.3. *Nearest neighbor*

Nearest neighbor algorithms are based on computing the distance from the interested item to either the rest of the items or the classes of items in a user profile. This kind of algorithm (Duda and Hart 1973) operates by storing all

Table 9. User profile-item matching technique of the systems based on content-based filtering

NAME	TECHNIQUE
ACR News	Itemset and cluster similarity matching
Amalthaea	Cosine similarity
Amazon	Unknown
Anatagonomy	Cosine similarity
Casmir	Pre-search request based collaboration, post-search informing
CDNow	Unknown
Fab	Cosine similarity
ifWeb	Standard keyword matching
InfoFinder	Boolean search query string
INFOrmer	Graph comparison
Krakatoa Chronicle	Cosine similarity
LaboUr	Bayesian classifier, nearest neighbor
Let's Browse	Cosine similarity
Letizia	Cosine similarity
MovieLens	Cosine similarity, inducted rules
News Dude	Short-term: nearest neighbor (cosine similarity); long-term: naive Bayesian classifier
NewsWeeder	Cosine similarity
NewT	Cosine similarity
Personal WebWatcher	Naive Bayesian classifier
PSUN	Graph comparison
Re:Agent	Nearest neighbor, neural network
Recommender	Inducted rules
SIFT Netnews	Dot product
SiteIF	Standard keyword matching
Syskill & Webert	Naive Bayesian classifier, nearest neighbor, PEBLS, cosine similarity, decision tree
Webmate	Cosine similarity
WebSail	TW2
WebSell	CBR with nearest neighbor (Pearson r correlation)
Websift	Inducted rules and pattern matching
WebWatcher	Cosine similarity

examples in the training set; that is, all items in the user profile. To learn the interest of an item, the algorithm assigns it to the class of the closest example. Depending on the item representation, the function to compute the distance can be a simple keyword matching or a weighted comparison (Schwab et al. 2001).

PEBLS (Cost and Salzberg 1993) is a nearest neighbor algorithm which makes use of a modification of the value difference metric (MVDM) for computing the distance between two examples.

LaboUr (Kamba et al. 1995), News Dude (Billsus and Pazzani 1999) and WebSell (Cunningham et al. 2001) are different examples of the nearest neighbor algorithm.

A particular case of the application of the nearest neighbor technique is Case-Based Reasoning (CBR). User profiles are represented by a collection of past experiences and, to recommend a new item, a wide amalgam of similarity measures to past items can be applied. The similarity encodes the knowledge that will assess whether an item suits the user's interests. Retrieval and adaptation techniques from CBR have become very important techniques for developing intelligent recommendation agents (Cunningham et al. 2001). For instance, WEBSELL (Cunningham et al. 2001) applies CBR with a similarity measure based on Pearson r correlation.

4.2.4. *Classification*

Systems based on content-based filtering can handle the recommendation task as a classification task. Based on a set of item features, the system tries to induce a model for each user which allows him/her to classify unseen items into two or more classes. The typical classification categories are *interesting* and *not interesting* (Billsus and Pazzani 1999), but the algorithm can classify items into any set of classes (e.g., relevant, undefined, not relevant). This means that the user profile is represented as a classifier: a neural network, decision tree, inducted rules or a Bayesian network (see Section 3.1.6).

For instance, Re:Agent (Boone 1998) implemented a neural network to divide several folders of e-mail into two categories: "work" and "other". Syskill & Webert (Pazzani et al. 1996) used a decision tree to classify Web pages into interesting/not interesting. Recommender (Basu et al. 1998) implemented a rule induction method to classify movies.

4.3. *User profile matching*

Systems based on collaborative filtering match people with similar interests and then make recommendations on this basis. Generally speaking the process of computing a recommendation consists of three steps: find similar users, create a neighborhood and compute a prediction based on selected neighbors.

4.3.1. *Find similar users*

Standard similarity measures are used to compute the distance between the current user's representation and the representation of a set of users. The commonest techniques used to compute the similarity between users are nearest neighbor, clustering and classifiers. Table 10 shows the user profile matching techniques used by the different systems analyzed.

Table 10. Techniques used by systems based on collaborative filtering to find similar users

NAME	TECHNIQUE
Amazon	Unknown
Anatagonomy	Cosine similarity
Beehive	Sharing news among users of the same cluster
Bellcore Video Recom	Nearest neighbor (Pearson r correlation)
Casmir	Pre-search request based collaboration, post-search informing
CDNow	Unknown
Fab	Cosine similarity
GroupLens	Nearest neighbor (Pearson r correlation)
Krakatoa Chronicle	Cosine similarity
LaboUr	Clustering (nearest neighbor – Pearson r correlation)
MovieLens	Cosine similarity
NewsWeeder	Cosine similarity
Personal WebWatcher	Naive Bayesian classifier
Recommender	Inducted rule execution
Ringo/FireFly	Nearest neighbor (mean squared differences, Pearson r correlation, constrained Pearson r correlation, artist-artist)
Smart Radio	Nearest neighbor (Pearson r correlation)
Tapestry	Tapestry query language
WebSell	CBR with nearest neighbor (Pearson r correlation)
Websift	Rule execution and pattern matching
WebWatcher	Cosine similarity

It is important to note that, in smaller applications, the set of users, among whom similarity is being computed, may be all users; in larger systems, statistical sampling methods are used to find a representative subset for which similarity is computed. In general, systems cannot work with large sets of data containing all the users and features, since the performance of the system will gradually break down. Several studies have been performed in order to reduce the data dimensionality, as for example Hofmann and Puzicha (1999) and Hayes et al. (2001).

4.3.1.1. *Nearest neighbor.* Nearest neighbor algorithms are applicable as a user profile-item matching technique (see Section 4.2) as well as a method to find similar users. In the latter case, nearest neighbor algorithms are based on computing the distance between consumers based on their preference history. Predictions of how much a user will like an item are computed by taking the

weighted average of the opinions of a set of nearest neighbors for that product. Nearest neighbor algorithms have the advantage of being able to incorporate the most up-to-date information rapidly, but the search for neighbors is slow in large databases. Herlocker et al. compare different nearest neighbor techniques and, as conclusions, show the results of these techniques in a specific framework and the suitability of each in different recommendation systems (Herlocker et al. 1999).

In general, two approaches are used in current systems to calculate the similarity between users: cosine similarity and correlation. Cosine similarity is applied in a way similar to the user profile-item matching technique (see Section 4.2.2) and users are compared to other users by the use of two vectors.

Regarding correlation, it is easy to define similarity measures between two user profiles working with databases of user ratings for items in which users indicate their interest in an item on a numeric scale. The typical correlation measures used in the systems analyzed are the Pearson r correlation coefficient (proposed by Shardanand and Maes (1995)) and the Spearman rank correlation coefficient (proposed by Herlocker et al. (1999)).

Another approach based on correlation between users is the entropy-based uncertainty measure. The measure of association based on entropy uses conditional probability techniques to measure the reduction in entropy of the active user's ratings which results from knowing another user's ratings. Herlocker et al. have shown that entropy has not shown itself as performing as well as Pearson r Correlation (Herlocker et al. 1999). Shardanand and Maes, in addition to Pearson r Correlation and Constrained Pearson r Correlation, use the Mean Squared Differences algorithm, which performs well compared to the Pearson r Correlation (Shardanand and Maes 1995). Another, more complicated approach, is explained in (Greening 1997).

4.3.1.2. *Clustering.* Some years ago, the user modelling community proposed a stereotype approach (Rich 1979). During the development stage of a system, user subgroups are identified and typical characteristics of members of these subgroups determined. During the run-time of the system, the user is assigned to one or more of these predefined user groups and their characteristics attributed to the user. The need for an empirically based pre-definition of these stereotypes is an evident disadvantage. As an alternative, the Doppelganger system used clustering mechanisms to find user groups dynamically, based on all available individual user models (Orwant 1995). Explicitly represented user models can be clustered and the descriptions of the clusters can be used like predefined stereotypes. Once the clusters are created, predictions for an

individual can be made by averaging the opinions of the other users in that cluster.

Clustering techniques usually produce less-personal recommendations than other methods, and in some cases, the clusters have less accuracy than nearest neighbor algorithms (Breese et al. 1998). Once the clustering is complete, however, performance can be very good, since the size of the group being analyzed is much smaller.

4.3.1.3 *Classification.* Collaborative filtering can be seen as a classification task (Billsus and Pazzani 1998) as well as part of content-based filtering (see Section 4.2.4). In collaborative filtering, where we want to infer item interest to a user, based on similarity with other users, typically, the initial data exists in the form of a sparse matrix (see Section 3.3.4), where rows correspond to users, columns correspond to items and the matrix entries are ratings. Note that "sparse" in this context means that most elements of the matrix are empty, because every user typically rates only a very small subset of all possible items. The prediction task can now be seen as filling in the missing matrix values. Since we are interested in learning personalized models for each user, we associate one classifier with every user. This model can be used to predict the missing values for one row in our matrix.

Some examples of classifiers are implemented in Basu et al. (1998), Good et al. (1999) and Billsus and Pazzani (1998).

4.3.2. *Create a neighborhood*

When systems look for similar users, they form a neighborhood of the most similar users to the target user. Generally, two techniques have been used to determine how many neighbors to select: the correlation-thresholding technique and the best-n-neighbors technique.

The correlation-thresholding technique is to set an absolute correlation threshold, where all neighbors with an absolute correlation greater than given thresholds are selected. Setting a high threshold limits the neighborhood to containing very good correlates, but for many users high correlates are not available, resulting in a small neighborhood which cannot provide prediction coverage for many items.

The best-n-neighbors technique is to pick the best-fixed number of users. This technique performs reasonably well, as it does not limit prediction coverage. However, picking a larger number of users will result in too much noise for those who have high correlates. Picking a smaller number can cause poor predictions for those users who do not have any high correlates.

A different approach has been proposed in Herlocker et al. (1999) for neighborhood formation based on the centroid. The first step is picking the closest user to the target user and calculate the centroid. Then, other users are

included in the neighborhood based on the distance to the centroid, which is recalculated each time a new user is added. Basically, this algorithm allows the nearest neighbors to affect the formation of the neighborhood and can be beneficial for very sparse data sets.

4.3.3. *Computing a prediction based on selected neighbors*

The final step is to derive the recommendations from the neighborhood of users. Once the neighborhood has been selected, the ratings from those neighbors are combined to compute a prediction, after possibly scaling the ratings to a common distribution. Different techniques are used in current systems: the most-frequent item recommendation, the association rule-based recommendation and the weighted average of ratings.

The most-frequent item recommendation looks into the neighborhood and scans through the user's interests extracting the most frequently selected items. After all the neighbors have been accounted for, the system sorts the items according to frequency and simply returns the n most frequent items not yet selected by the active user as recommendation.

The association rule-based recommendation infers rules previously generated from the neighborhood instead of using the entire population of users. Note that, considering only a few neighbors may not generate strong enough association rules in practice, which, in consequence, may result in insufficient items to recommend. The number of items can be augmented by using a scheme in which the rest of the items, if necessary, are computed by using the most frequent item algorithm.

Another way to combine all the neighbor's ratings into a prediction is to compute a weighted average of the ratings using the correlation as the weight. The basic weighted average makes an assumption that all users give ratings of approximately the same distribution.

The approach taken by GroupLens (Resnick et al. 1994) was to compute the average deviation of a neighbor's rating from that neighbor's mean rating, where the mean rating is taken over all items the neighbor has rated. The justification for this approach is that users may rate distributions centered on different points.

5. Cross-Dimensional Analysis

We have analyzed of 37 systems following a functional approach that allows us to draw up a general taxonomy comprising 8 dimensions and based on two main groups: user profile generation and maintenance, and user profile exploitation techniques. We then carried out a cross-dimensional analysis using the results of the spatial approach (see Table 1), that is, a cross-

dimension analysis among all the recommender systems of the same domain. From such an analysis, we have detected common patterns in web recommender systems, e-commerce recommender systems, item recommender systems and news recommender systems.

First, Table 11 illustrates a cross-dimension analysis among web recommender systems. Some common features come up when we look at the different systems. First of all, we can conclude that most of the systems need feature selection and information indexing techniques to extract relevant information from text. Therefore, web recommender systems represent profiles as feature vectors, learn profiles from text through feature selection and TF-IDF techniques and match profiles with new items through cosine similarity technique. Another feature to stress is the information filtering method: web recommender systems analyze the content of web pages before recommendation, therefore implementing a content-based filtering method. Some of them, however, take advantage of collaborative techniques to improve their results.

Second, Table 12 shows the cross-dimension analysis among the three e-commerce recommender systems analyzed in this paper. We can arrive at several relevant conclusions from this table. First, these systems do not need feature selection and information indexing techniques because the source of the information is a structured database of products. Thus, they represent the user profile as a history of *interesting/not interesting/purchased* products. Such profiles grow enormously as time passes, but e-commerce recommender systems are not interested in reducing the size of the profile because they do not want to lose information in a contraction process. So, a profile learning technique is not needed and they do not apply a profile adaptation technique to forget old interests. The large size of the user profile requires an advanced user profile–item matching technique, since the success of the recommendations depends on it. We think that this is the heart of e-commerce systems and therefore big e-commerce companies, such as Amazon and CDNow, do not publish which method they are using. Finally, it is important to note that these systems take advantage of the collaborative world, apart from the content analysis, to improve the quality of their recommendations.

Third, Table 13 shows the cross-dimension analysis among item (e.g., movies, music, ...) recommender systems. The first conclusion that we can extract from this table is that all the systems take advantage of the collaborative world to improve their recommendations. In addition, most of the systems only recommend items based on other users opinions and these systems therefore represent the user profile as a user-item ratings matrix. Using this matrix as a user profile means there is no need for a profile-learning method and recommender agents match user profiles by the nearest neighbor technique.

Table 11. Cross-dimension analysis among Web recommender systems

NAME	REPRESENT	INITIAL	LEARNING	FEEDBACK	ADAPTION	FILTERING	MATCHING
Amalthaea	Feature vector	Manual	F.S., TF-IDF	Ratings	Natural selection GFF	Content	Cosine similarity
Fab	Feature vector	Empty	TF-IDF	Ratings	Natural selection	Hybrid	Cosine similarity
ifWeb	Semantic network	Training set, stereotyping	F.S.	Ratings	GFF	Content	Keyword matching
Let's Browse	Feature vector	Training set	TF-IDF	Links, time	Add new	Content	Cosine similarity
Letizia	Feature vector	Empty	TF-IDF	Links, time	Add new	Content	Cosine similarity
Personal WebWatcher	Feature vector	Manual	TF-IDF	Links	Add new	Hybrid	Bayesian classifier
SiteIF	Semantic network	Empty	F.S.	Links	GFF	Content	Keyword matching
Syskill & Webert	Feature vector, decision tree	Manual, stereo-typing	F.S., TF-IDF, ID3	Ratings	Add new	Content	Bayesian classifier, PEBLS, cosine similarity, decision tree
Webmate	Feature vector	Empty	TF-IDF	Like/dislike	Add new	Content	Cosine similarity
WebSail	Feature vector	Empty	TF-IDF	Like/dislike	Add new	Content	TW2
Websift	Rules, patterns	Training set	Rule induction	History	Add new	Hybrid	Rules, pattern matching
WebWatcher	Feature vector	Manual	TF-IDF	Links	Add new	Hybrid	Cosine similarity

Table 12. Cross-dimension analysis among e-commerce recommender systems

NAME	REPRE-SENT	INITIAL	LEARNING	FEED-BACK	ADAP-TION	FILTER-ING	MATCHING
Amazon	History	Empty	Not necessary	Ratings, history	Add new	Hybrid	Unknown
CDNow	History	Empty	Not necessary	Ratings, history	Add new	Hybrid	Unknown
WebSell	History	Empty	Not necessary	Unknown	Add new	Hybrid	CBR

However, it is very important that users provide relevance feedback to fulfill the matrix of the profile. Looking at the table, we can conclude that all the item recommender systems request ratings as relevance feedback. We also notice that all the item recommender systems add new information to the profile and never forget past item interests. However, we believe that a technique that will adapt the user profile as time passes, forgetting old interests, is needed in this domain.

Finally, Table 14 shows the cross-dimension analysis among news recommender systems. Like web recommender systems, news recommender systems need feature selection and information indexing techniques to extract relevant information from text. Hence, they represent the user profile with a feature vector, or some more complex structure, such as an associative network or an n-gram. Therefore, most of them learn user profiles through feature selection and TF-IDF techniques.

We did not detect any other pattern. We believe that this lack of common features lies in the fact that most systems have been developed following ad hoc approaches to satisfy specific application requirements. In this sense, we think that the taxonomy provided in this paper could be a useful guide for researchers contributing to the future development of new recommender agents.

6. Conclusions

The unceasing growth of the Internet and its environment has brought the need for new technology to help users to find what they are looking for. The combination of modelling particular user preferences, building content models and modelling social patterns in intelligent agents seems to be an ideal

Table 13. Cross-dimension analysis among item recommender systems

NAME	REPRESENT	INITIAL	LEARNING	FEEDBACK	ADAPTION	FILTERING	MATCHING
Bellcore Video	Ratings matrix	Training set	Not necessary	Ratings	Add new	Collaborative	Nearest neighbor
LifeStyle Finder	Demographic features	Stereo-typing	Not necessary	Ratings, history	Add new	Demographic	Demographic reasoning
MovieLens	Feature vector, rules	Training set	TF-IDF, rule learning	Ratings	Add new	Hybrid	Cosine similarity
Recommender	Rules	Training set	Rule Learning	Ratings	Add new	Hybrid	Rules
Ringo/FireFly	Ratings matrix	Training set	Not necessary	Ratings	Add new	Hybrid	Nearest neighbor
Smart Radio	Ratings matrix	Training set	Not necessary	Ratings, implicit	Add new	Collaborative	Nearest neighbor

Table 14. Cross-dimension analysis among news recommender systems

NAME	REPRESENT	INITIAL	LEARNING	FEEDBACK	ADAPTION	FILTERING	MATCHING
ACRNews	Clusters	Training set	Rule learning, clustering	History	Add new	Content	Cluster matching
Anatagonomy	Feature vector	Empty	TF-IDF	Ratings	Add new	Hybrid	Cosine similarity
Beehive	Clusters (feature vectors)	Empty	Clustering	History	Add new	Collaborative	Cluster matching
GroupLens	Ratings matrix	Empty	TF-IDF	Ratings, text, time	Add new	Collaborative	Nearest neighbor
INFOrmer	Associative network	Training set	F.S.	Ratings	Add new	Content	Graph comparison
Krakatoa Chronicle	Feature vector	Empty	TF-IDF	Ratings, implicit	Add new	Hybrid	Cosine similarity
News Dude	Feature vector	Training set	TF-IDF	Like/dislike	Short/long-term models	Content	Nearest neighbor, Bayesian classifier
NewsWeeder	Feature vector	Training set	TF-IDF, MDL	Ratings	Add new	Hybrid	Cosine similarity
NewT	Feature vector	Training set	F.S., TF-IDF	Like/dislike	Natural selection	Content	Cosine similarity
PSUN	N-grams	Training set	F.S., n-gram induction	Ratings	Natural selection	Content	Graph comparison
SIFT Netnews	Feature vector, decision tree	Training set	TF-IDF	Like/dislike	Manual	Collaborative	Dot product

solution. This paper has tried to gather together the state-of-the-art elements in recommender agents on the Internet.

There are other papers that have dealt with state-of-the-art recommender systems (e.g., Sarwar et al. 2000; Pretschner and Gauch 1999; Terveen and Hill 2001; Kobsa et al. 2001). Schafer et al., in particular, present a taxonomy of recommender systems in the e-commerce field, classifying the techniques used into three dimensions (Schafer et al. 2001). In this paper, we present a more complete, up-to-date taxonomy of general intelligent recommender agents on the Internet. We have analyzed 37 systems, using a functional approach, and have divided our taxonomy into two main groups: user profile generation and maintenance, and user profile exploitation techniques. From this we got a basic list of 8 dimensions and we have explained, within each of these dimensions, all the techniques used in the systems we analyzed. We followed that with a cross-dimensional analysis which we hope gives designers a set of clues that will help them in their work to develop new systems according to their requirements.

To sum up, the taxonomy provides a comprehensive explanation of current recommender agents which we hope will contribute towards progress in this area of research.

Acknowledgements

Miquel Montaner wishes to express sincere appreciation to the people at Intelligent Software Agents Group in the Robotics Institute of the Carnegie Mellon University, where the major part of this paper was developed during a research stage. Special gratitude to Professor Katia Sycara, head of the group, who accepted this author as research visitor.

This research has been developed within the DAF-DAF Project supported by the CICYT grant DPI2001-2094-C03-01.

References

Amazon (2001). http://www.amazon.com.

Armstrong, R., Freitag, D., Joachims, T. & Mitchell, T. (1995). WebWatcher: A Learning Apprentice for the World Wide Web. In *1995 AAAI Spring Symposium on Information Gathering from Heterogeneous Distributed Environments*.

Asnicar, F. & Tasso, C. (1997). IfWeb: A Prototype of User Models Based Intelligent Agent for Document Filtering and Navigation in the World Wide Web. In *Proceedings of UM'97*. Sardinia, Italy: Chia Laguna.

Balabanovic, M. & Shoham, Y. (1997). Combining Content-Based and Collaborative Recommendation. *Communications of the ACM*.

Basu, C., Hirsh, H. & Cohen, W. (1998). Recommendation as Classification: Using Social and Content-Based Information in Recommendation. In *Proceedings of AAAI'98*, 714–720.

Berney, B. & Ferneley, E. (1999). Casmir: Information Retrieval Based on Collaborative User Profiling. In *Proceedings of PAAM'99*, 41–56. Lancashire: The Practical Application Company Ltd.

Billsus, D. & Pazzani, M. J. (1998). Learning Collaborative Information Filters. In *Proceedings of the International Conference on Machine Learning*. Madison, WI: Morgan Kaufmann Publishers.

Billsus, D. & Pazzani, M. J. (1999). A Hybrid User Model for News Classification. In *Proceedings of UM'99*, 99–108. Wien, New York: Springer-Verlag.

Boone, G. (1998). Concept Features in RE:Agent, an Intelligent Email Agent. In *The Second International Conference on Autonomous Agents (Agents '98)*. Minneapolis/St. Paul.

Breese, J., Heckerman, D. & Kadie, C. (1998). Empirical Analysis of Predictive Algorithms for Collaborative Filtering. *Uncertainty in Artificial Intelligence: Proceedings of the 14th Conference*, 43–52. San Francisco: Morgan Kaufmann.

Buckley, C. & Salton, G. (1995). Optimization of Relevance Feedback Weights. In Fox, E., Ingwersen, P. & Fidel, R. (eds.) *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 351–357.

Buckley, C., Singhal, A., Mitra, M. & Salton, G. (1996). New Retrieval Approaches Using SMART. In *Proceedings of TREC-4*. NIST Special Publication.

Carroll, J. & Rosson, M. B. (1987). The Paradox of the Active User. In Carroll, J. M. (ed.) *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction*, 26–28. Cambridge, MA: MIT Press.

CDNow (2001). http://www.cdnow.com.

Chatterjee, P., Hoffman, D. L. & Novak, T. P. (1998). Modeling the Clickstream: Implications for Web-Based Advertising Efforts. *Working Paper*. Vanderbilt University.

Chen, L. & Sycara, K. (1998). Webmate: A Personal Agent for Browsing and Searching. In *Proceedings of AGENTS '98*, 132–139. ACM.

Chen, Z., Meng, X., Zhu, B. & Fowler, R. (2000). WebSail: From On-Line Learning to Web Search. In *Proceedings of the 2000 International Conference on Web Information Systems Engineering*.

Clark, P. & Niblett, T. (1989). The CN2 Induction Algorithm. In *Machine Learning*, Vol. 3, 261–283. The Netherlands: Kluwer Academic Publishers.

Cohen, W. (1995). Fast Effective Rule Induction. In *Proceedings of ML95*, 115–123. San Francisco: Morgan Kaufmann.

Cohen, W. & Singer, Y. (1999). A Simple, Fast, and Effective Rule Learner. In *Proceedings of AAAI-99*, 335–342.

Cooley, R., Tan, P. N. & Srivastava, J. (1999). WebSift: The Web Site Information Filter System. In *Proceedings of the 1999 KDD Workshop on Web Mining*. San Diego, CA: Springer-Verlag.

Cost, S. & Salzberg, S. (1993). A Weighted Nearest Neighbor Algorithm for Learning with Symbolic Features. *Machine Learning* **10**: 57–78.

Cunningham, P., Bergmann, R., Schmitt, S., Traphoner, R., Breen, S. & Smyth, B. (2001). WebSell: Intelligent Sales Assistants for the World Wide Web. In *E-2001*.

Duda, R. & Hart, P. (1973). *Pattern Classification and Scene Analysis*. New York: John Wiley & Sons, ISBN-0471223611.

Goldberg, D., Nichols, D., Oki, B. M. & Terry, D. (1992). Using Collaborative Filtering to Weave an Information Tapestry. *Communications of the ACM* **35**: 61–70.

Good, N., Schafer, J., Konstan, J., Borchers, A., Sarwar, B., Herlocker, J. & Riedl, J. (1999). Combining Collaborative Filtering with Personal Agents for Better Recommendations. In *Proceedings of AAAI*, Vol. 35, 439–446. AAAI Press.

Greening, D. (1997). Building Consumer Trust with Accurate Product Recommendations. *Likeminds White Paper LMWSWP-210-6966*.

Hayes, C. & Cunningham, P. (1999). Smart Radio – a Proposal. In *Trinity College Dublin, Computer Science*, Technical Report, TCD-CS-1999-24.

Hayes, C. & Cunningham, P. (2000). Smart Radio: Building Music Radio on the Fly. In *Proceedings of Expert Systems 2000 (ES2000)*. Cambridge, UK.

Hayes, C., Cunningham, P. & Smyth, B. (2001). A Case-Based Reasoning View of Automated Collaborative Filtering. In *Trinity College Dublin, Computer Science*, Technical Report, TCD-CS-2001-09.

Herlocker, J., Konstan, J., Borchers, A. & Riedl, J. (1999). An Algorithmic Framework for Performing Collaborative Filtering. In *Proceedings of the 1999 Conference on Research and Development in Information Retrieval*.

Herlocker, J., Konstan, J. & Riedl, J. (2000). Explaining Collaborative Filtering Recommendations. In *Proceedings of ACM 2000 Conference on Computer Supported Cooperative Work*.

Hill, W., Stead, L., Rosenstein, M. & Furnas, G. (1995). Recommending and Evaluating Choices in a Virtual Community of Use. In *Proceedings of CHI'95*, 194–201. Denver.

Hofmann, T. & Puzicha, J. (1999). Latent Class Models for Collaborative Filtering. In *Proceedings of IJCAI'99*, 688–693. Stockholm, ISBN 1-55860-613-0.

Holte, R. C. & Yan, N. Y. (1996). Inferring What a User Is Not Interested In. In *AAAI Spring Symp. on Machine Learning in Information Access*. Stanford.

Huberman, B. & Kaminsky, M. (1996). Beehive: A System for Cooperative Filtering and Sharing of Information. *Technical Report, Dynamics of Computation Group*. Palo Alto, CA: Xerox, Palo Alto Research Center.

Jennings, A. & Higuchi, H. (1993). A User Model Neural Network for a Personal News Service. *User Modeling and User-Adapted Interaction* **3**: 1–25.

Jensen, F. V. (1996). *An Introduction to Bayesian Networks*. New York: Springer.

Joachims, T., Freitag, D. & Mitchell, T. (1997). WebWatcher: A Tour Guide for the World Wide Web. In *Proceedings of IJCAI'97*, 770–775. Nagoya, Japan.

Kamba, T., Bharat, K. & Albers, M. C. (1995). The Krakatoa Chronicle – an Interactive, Personalized, Newspaper on the Web. In *Proceedings of the Fourth International World Wide Web Conference*, 159–170.

Kobsa, A., Koenemann, J. & Pohl, W. (2001). Personalized Hypermedia Presentation Techniques for Improving Online Customer Relationships. *The Knowledge Engineering Review* **16**: 111–155.

Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L. & Riedl, J. (1997). Grouplens: Applying Collaborative Filtering to Usenet News. *Communications of the ACM* **40**: 77–87.

Koychev, I. (2000). Gradual Forgetting for Adaptation to Concept Drift. In *Proceedings of ECAI 2000 Workshop Current Issues in Spatio-Temporal Reasoning*.

Krulwich, B. (1997). LifeStyle Finder: Intelligent User Profiling Using Large-Scale Demographic Data. *AI Magazine* **18**(2): 37–45.

Krulwich, B. & Burkey, C. (1995). ContactFinder: Extracting Indications of Expertise and Answering Questions with Referrals. *Working Notes of the 1995 Fall Symposium on Intelligent Knowledge Navigation and Retrieval*, 85–91. Technical Report FS-95-03, The AAAI Press.

Krulwich, B. & Burkey, C. (1996). Learning User Information Interests Through Extraction of Semantically Significant Phrases. In *Proceedings of the AAAI Spring Symposium on Machine Learning in Information Access*. Stanford, CA.

Lang, K. (1995). NewsWeeder: Learning to Filter News. In *Proceedings of the 12th International Conference on Machine Learning*, 331–339. Lake Tahoe, CA.

Lieberman, H. (1995). Letizia: An Agent that Assists Web Browsing. In *Proceedings of the IJCAI'95*, 924–929.

Lieberman, H., Van Dyke, N. W. & Vivacqua, A. S. (1999). Let's Browse: A Collaborative Web Browsing Agent. In *Proceedings of International Conference on Intelligent User Interfaces*, 924–929.

Maes, P. (1994). Agents that Reduce Work and Information Overload. *Communications of the ACM* **37**(7): 30–40.

Maloof, M. A. & Michalski, R. S. (2000). Selecting Examples for Partial Memory Learning. *Machine Learning* **41**: 27–52.

Minio, M. & Tasso, C. (1996). User Modeling for Information Filtering on Internet Services: Exploiting an Extended Version of the UMT Shell. In *UM96 Workshop on User Modeling for Information Filtering on the WWW*. Kailua-Kona, Hawaii.

Mitchell, T., Caruana, R., Freitag, D., McDermott, J. & Zabowski, D. (1994). Experience with a Learning Personal Assistant. *Communications of the ACM* **37**(7): 81–91.

Mitchell, T. M., Mahadevan, S. & Steinberg, L. (1985). Leap: A Learning Apprentice for VLSI Design. In *Proceedings of IJCAI'85*, 573–580. Los Altos, CA: Morgan Kaufmann.

Mladenic, D. (1996). Personal WebWatcher: Implementation and Design. *Technical Report IJS-DP-7472, Department of Intelligent Systems*. Slovenia: J. Stefan Institute.

Mobasher, B., Cooley, R. & Srivastava, J. (2000). Automatic Personalization Based on Web Usage Mining. *Communications of the ACM* **43**(8).

Morita, M. & Shinoda, Y. (1994). Information Filtering Based on User Behaviour Analysis and Best Match Text Retrieval. In *Proceedings of SIGIR'94*, 272–81. Dublin, Ireland: Springer-Verlag.

Moukas, A. (1997). Amalthaea: Information Filtering and Discovery Using a Multiagent Evolving System. *Journal of Applied AI* **11**(5): 437–457 (Dublin, Ireland, Springer-Verlag).

Nichols, D. M. (1997). Implicit Rating and Filtering. In *Proceedings of 5th DELOS Workshop on Filtering and Collaborative Filtering*, 31–36.

Orwant, L. J. (1995). Heterogeneous Learning in the Doppelganger User Modelling System. *User Modelling and User Adapted Interaction* **4**(2): 107–130.

Pazzani, M. (1999). A Framework for Collaborative, Content-Based and Demographic Filtering. *Artificial Intelligence Review*.

Pazzani, M. & Billsus, D. (1997). Learning and Revising User Profiles: The Identification of Interesting Web Sites. *Machine Learning* **27**: 313–331 (Kluwer Academic Publishers).

Pazzani, M., Muramatsu, J. & Billsus, D. (1996). Syskill & Webert: Identifying Interesting Web Sites. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 54–61.

Potter, G. & Trueblood, R. (1988). Traditional, Semantic, and Hyper-Semantic Approaches to Data Modeling. *IEEE Computer* **21**(6): 53–63.

Pretschner, A. & Gauch, S. (1999). Ontology-Based Personalized Search. In *Proceedings of ICTAI'99*, 391–398.

Quinlan, J. R. (1983). Learning Efficient Classification Procedures and Their Application to Chess End Games. In Michalski, R. S., Carbonell, J. G. & Mitchell, T. M. (eds.) *Machine Learning: An Artificial Intelligence Approach*, 463–482.

Quinlan, J. R. (1994). The Minimum Description Length Principle and Categorical Theories. In *Proceedings of ML'94*. San Mateo: Morgan Kaufmann.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. & Riedl, J. (1994). Grouplens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of ACM CSCW'94*, 175–186.

Rich, E. (1979). User Modeling via Stereotypes. *Cognitive Science* **3**: 329–354.

Riordan, A. & Sorensen, H. (1995). An Intelligent Agent for High-Precision Information Filtering. In *Proceedings of the CIKM-95 Conference*.

Sakagami, H., Kamba, T. & Koseki, Y. (1997). Learning Personal Preferences on Online Newspaper articles for User Behaviors. In *Proc. 6th Int. World Wide Web Conference*, 291–300.

Salton, G. & Buckley, C. (1988). Term-Weighting Approaches in Automatic Text Retrieval. *Information Processing and Management* **24**(5): 513–523.

Salton, G. & Buckley, C. (1990). Improving Retrieval Performance by Relevance Feedback. In Spark Jones and Willet (eds.) *Readings in Information Retrieval*, Vol. 24, No. 5, 513–523. San Francisco, CA: Morgan Kauffman.

Salton, G. & McGill, M. (1983). *Introduction to Modern Information Retrieval*. New York, NY: McGraw-Hill Publishing Company.

Sarwar, B. M., Karypis, G., Konstan, J. A. & Riedl, J. (2000). Analysis of Recommender Algorithms for E-Commerce. In *ACM E-Commerce 2000 Conference*.

Schafer, J. B., Konstan, J. & Riedl, J. (2001). Electronic Commerce Recommender Applications. *Journal of Data Mining and Knowledge Discovery* **5**: 115–152.

Schwab, I., Kobsa, A. & Koychev, I. (2000). Learning about Users from Observation. In *AAAI 2000 Spring Symposium: Adaptive User Interface*.

Schwab, I., Kobsa, A. & Koychev, I. (2001). Learning User's Interests Through Positive Examples Using Content Analysis and Collaborative Filtering. *Submitted*.

Shardanand, U. (1994). Social Information Filtering for Music Recommendation. MIT EECS M. Eng. thesis, also TR-94-04. *Learning and Common Sense Group*. MIT Media Laboratory.

Shardanand, U. & Maes, P. (1995). Social Information Filtering: Algorithms for Automating 'Word of Mouth'. . . . In *Proceedings of CHI'95*, 210–217.

Sheth, B. & Maes, P. (1993). Evolving Agents for Personalitzed Information Filtering. In *Proceedings of the Ninth Conferece on Artificial Intelligence for Applications*. IEEE Computer Society Press.

Sorensen, H. & McElligot, M. (1995). PSUN: A Profiling System for Usenet News. In *CKIM '95 Workshop on Intelligent Information Agents*.

Sorensen, H., Riordan, A. O. & Riordan, C. O. (1997). Profiling with the INFORMER Text Filtering Agent. *Journal of Universal Computer Science* **3**(8): 988–1006.

Stefani, A. & Strappavara, C. (1998). Personalizing Access to Web Wites: The SiteIF Project. In *Proceedings of HYPERTEXT'98*.

Terveen, L. G. & Hill, W. (2001). Beyond Recommender Systems: Helping People Help Eachother. In Carroll, J. (ed.) *HCI in the New Millennium*. Addison Wesley.

Webb, G. & Kuzmycz, M. (1996). Feature Based Modelling: A Methodology for Producing Coherent, Consistent, Dynamically Changing Models of Agents' Competencies. *User Modelling and User-Adapted Interaction* **5**: 117–150.

Widmer, G. & Kubat, M. (1996). Learning in the Presence of Concept Drift and Hidden Contexts. *Machine Learning* **23**: 69–101 (Kluwer Academic Publishers).

Yan, T. W. & Garcia-Molina, H. (1995). Sift – a Tool for Wide-Area Information Dissemination. In *Proceedings of the 1195 USENIX Technical Conference*, 177–186.