Chapter 16
Usability Guidelines for Product Recommenders Based on Example Critiquing Research

Pearl Pu, Boi Faltings, Li Chen, Jiyong Zhang and Paolo Viappiani

Abstract Over the past decade, our group has developed a suite of decision tools based on example critiquing to help users find their preferred products in e-commerce environments. In this chapter, we survey important usability research work relative to example critiquing and summarize the major results by deriving a set of usability guidelines. Our survey is focused on three key interaction activities between the user and the system: the initial preference elicitation process, the preference revision process, and the presentation of the systems recommendation results. To provide a basis for the derivation of the guidelines, we developed a multi-objective framework of three interacting criteria: accuracy, confidence, and effort (ACE). We use this framework to analyze our past work and provide a specific context for each guideline: when the system should maximize its ability to increase users' decision *accuracy*, when to increase user *confidence*, and when to minimize the interaction *effort* for the users. Due to the general nature of this multi-criteria model, the set of guidelines that we propose can be used to ease the usability engineering process of other recommender systems, especially those used in e-commerce environments. The ACE framework presented here is also the first in the field to evaluate the performance of preference-based recommenders from a user-centric point of view.

Designers can use these guidelines for the implementation of an effective and successful product recommender.

Pearl Pu · Li Chen · Jiyong Zhang
Human Computer Interaction Group, School of Computer and Communication Sciences,
Swiss Federal Institute of Technology in Lausanne (EPFL), CH-1015, Lausanne, Switzerland
e-mail: \{pearl.pu,li.chen,jiyong.zhang\}@epfl.ch

Boi Faltings
Artificial Intelligence Laboratory, School of Computer and Communication Sciences
Swiss Federal Institute of Technology in Lausanne (EPFL), CH-1015, Lausanne, Switzerland
e-mail: boi.faltings@epfl.ch

Paolo Viappiani
Department of Computer Science, University of Toronto, 6 King's College Road, M5S3G4, Toronto, ON, CANADA
e-mail: paolo.viappiani@gmail.com

F. Ricci et al. (eds.), *Recommender Systems Handbook*,
DOI 10.1007/978-0-387-85820-3_16,

16.1 Introduction

According to Jacob Nielsen, a well known usability researcher, the first law of e-commerce is that if users cannot find the product, they cannot buy it either.[1] The word "find" indeed defines a challenging task that e-commerce systems must support. It refers to the online retailer's ability to help users identify an ideal product that satisfies the user's needs (sometimes even unknown to him/herself) and inspire users to select the items recommended to them. This implies that the system must assist users to carry out not only a search but also a decision making task.

How do users actually face such tasks in the online environments? With increased competition, online retailers are offering a progressively large collection of available products. New items are added to their catalogs regularly, often to ensure that all of their direct competitors' products are included in their inventory as well. The task of locating a desired choice is directly dependent on the number of available options. Indeed with this "infinite" shelf space the user task is becoming daunting, if not impossible, for the average user. Under such circumstances, users are likely to employ one of two decision approaches. In the first case, they try to achieve high decision accuracy, but face the time-consuming task of sifting through all options and trading off the pros and cons between the various aspects of the products. Alternatively, they can adopt heuristic decision strategies and process information more selectively. Although they expend less effort in this case, these heuristic strategies can lead to decision errors and are likely to cause decision regret. Clearly, neither approach is ideal, since adopting one or the other implies a compromise on either decision accuracy or effort. According to [40], the tradeoff between accuracy and effort is an inherent dilemma in decision-making that cannot be easily reconciled.

Significant research has been performed to develop highly interactive and intelligent tools to assist users. As a result, preference-based recommenders have emerged and are broadly recognized as effective search and navigation mechanisms guiding users to find their preferred products in e-commerce and other demanding decision environments. In the past decade, we have developed the example critiquing method and a suite of decision tools based on this method to provide personalization and recommendation to the product search problem [43, 45, 51, 52]. More than a dozen user studies were carried out and published, which validated the method in various data domains: travel planning, apartment search, and product search for laptops, Tablet PCs, and digital cameras. However, the wide adoption of example critiquing in electronic commerce remains limited.

Our goal is to analyze and survey our past work related to example critiquing and synthesize the major results by deriving a set of usability guidelines. More specifically, we focus on three key interaction activities: the initial preference elicitation process, the preference revision process, and the presentation of the system's recommendation results. To provide a basis for the derivation of the guidelines, we investigate the objectives a product recommender must achieve in order to maximize

[1] Nielsen stated this law in his Alertbox in 2003. For details, please visit http://www.useit.com/alertbox/20030825.html.

user satisfaction and their willingness to use the system. A recommender's accuracy, i.e., how the system finds items that users truly want, has traditionally been an important and central aspect of recommenders. However, accuracy alone does not entirely capture user benefit without the consideration of ease of use (usability) [35]. People are known to have very limited cognitive resources and are not likely to achieve a high level of accuracy if the required effort is excessive. Finally, since product search is a decision process, the recommender must also help users achieve confidence that the products recommended to them are what they truly want. Instead of accuracy alone, we therefore propose a multi-objective framework, called ACE, for the derivation of our guidelines: 1) the system's ability to help users find their most preferred item (accuracy), 2) its ability to inspire users' confidence in selecting the items that were recommended to them (confidence), and 3) the amount of user effort it requires for achieving the relative accuracy (effort).

Notice that some users may be ready to spend a lot of effort to obtain a very accurate recommendation, while others may be ready to accept a lower accuracy to obtain a result quickly. While the design of a recommender clearly involves a tradeoff between accuracy and effort, what actually matters to the user is the tradeoff between *confidence* and effort: I am willing to put in more interaction effort only if I am increasingly convinced of the products that are recommended to me. The main challenge for recommender system design is to ensure that user confidence is sufficiently high to make them spend enough effort to reach an acceptable decision outcome.

This set of three requirements for deriving the guidelines can also be used as an evaluation framework to measure the usability of a product recommender. This chapter therefore contributes to the field in two main areas. From an academic point of view, the article establishes a novel set of user-centric criteria to evaluate the performance of preference-based recommenders. From a practical point of view, the chapter surveys the state of the art of example critiquing on three key interaction activities and derives a set of 11 usability guidelines that can be applied in a wider and more scalable way. Since these guidelines are derived from methods that have been validated in usability studies, practitioners can use them with confidence to enhance the usability engineering process, including design and testing, of a product recommender.

16.2 Preliminaries

16.2.1 Interaction Model

Preference-based recommenders suggest items to users based on their *explicitly* stated preferences over the attributes of the items. The various systems described in this chapter were designed using different architectures and evaluated using different data domains. However, they share some overall characteristics, especially

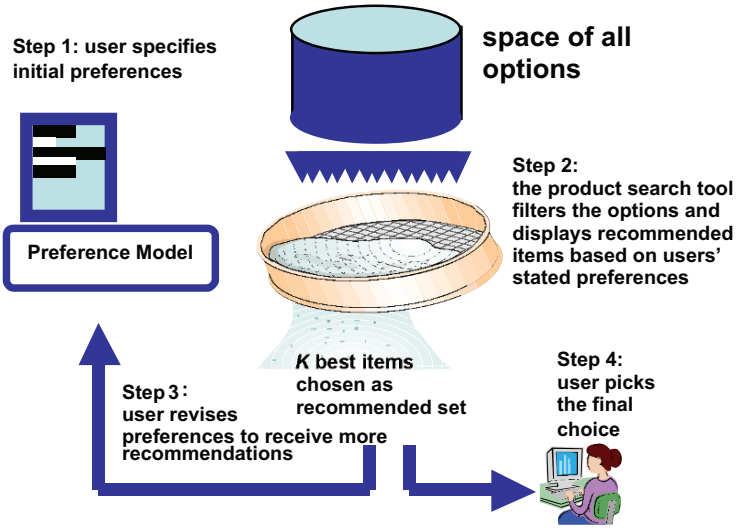


Fig. 16.1: The generic system-user interaction model of a preference-based recommender system.

concerning the interaction model. To form a nomenclature throughout the survey and the guideline derivation process, we present a generic model of interaction summarizing the steps of a recommender in Figure 16.1. A user starts his/her interaction process by stating a set of initial preferences via, for example, a graphical user interface. After obtaining that information, the system filters the space of options and selects the items to be recommended to users based on their stated preferences. This set is called the *recommendation set*. At that point, either the user finds her most preferred item in the recommendation set and thus terminates her interaction with the system, or she revises the preference model, using critiques such as "I would like a cheaper item", in order to obtain more accurate recommendations. This last user feedback step is called preference revision. As a result of the process, the user can either pick a single item, or construct a list of items known as the consideration set that might then be compared in further detail.

The recommender can fail if the user is not sufficiently confident that the process is indeed finding a suitable product, and therefore does not undertake sufficient preference revision cycles to give the system an accurate preference model. Our guidelines are designed to avoid this situation, and at the end of this chapter we provide a model that shows their rationale with respect to the user's decision process.

Once a recommendation set has been determined, a system may use various display strategies to show the results. A typical tool presents the user with a set of k items ($1 \leq k \leq n$, where n is the total number of products) in each of a total of m interactions. In each display of these k items, a user is identifying her target choice to be included in the consideration set. The more options are displayed, the more effort the user must expend to examine them. On the other hand, in a small display

set users could easily overlook their target choice and engage in more interaction cycles. This tradeoff of effort versus accuracy will be discussed in further detail when individual systems are presented.

The presented interactive components may not be simultaneously included in the same system by other tools. For example, the initial preference elicitation is an optional step when users are presented with a set of recommendations (e.g., best sellers in different categories) as soon as they visit a site. Other systems, on the other hand, may elicit users' initial preferences but do not provide the option to allow users to revise them.

16.2.2 Utility-Based Recommenders

Since example critiquing tools are based on multi-attribute utility theory, we provide an overview of the underlying recommendation algorithm.

The fundamental assumption underlying these recommenders is that people prefer items because of their attributes. Different values of an attribute correspond to different degrees of preference depending on the situation: for example, a large apartment is useful when there are guests, but not useful when it has to be cleaned. A user will determine preferences for an item by weighing the advantages and disadvantages of each feature according to how often it is beneficial and how often it is not. Thus, preference is a weighted function of attributes.

Formally, the preference-based recommendation problem can be formulated as a Multi-Attribute Decision Problem (MADP) $\Psi = \langle \mathbf{X}, \mathbf{D}, \mathbf{O}, \mathbf{P} \rangle$, where $\mathbf{X} = \{X_1, \cdots, X_n\}$ is a finite set of attributes that the product catalog has, $\mathbf{D} = D_1 \times \cdots \times D_n$ indicates the product domain space (each $D_i (1 \leq i \leq n)$ is a set of domain values for attribute X_i), $\mathbf{O} = \{O_1, \cdots, O_m\}$ is a finite set of available products that the system may provide, and $\mathbf{P} = \{P_1, \cdots, P_t\}$ denotes a set of preferences that the user may have. The objective of a MADP is to find a product (or products) that is (or are) most preferred by the user. A MADP can be solved by constraint-based approaches or utility-based approaches. Below we introduce the approach based on the multi-attribute utility theory (MAUT) to solve a given MADP. Please see [45] for the constraint-based approach.

Multi-Attribute Utility Theory (MAUT)

The origin of utility theory dates back to 1738 when Bernoulli proposed his explanation to the St. Petersburg paradox in terms of the utility of monetary value [4]. Two centuries later it was von Neumann and Morgenstern (1944) who revived this method to solve problems they encountered in economics [71]. Later, in the early 1950s, in the hands of Marschak [29] and of Herstein and Milnor [21], the Expected Utility Theory was established on the basis of a set of axioms that form the basis of the von Neumann Morgenstern theorem (VNM Theorem) [39, 59].

In the 1970s Keeney and Raiffa [24] extended utility theory to the case of multiple attributes. The main idea of multi-attribute utility theory is to represent user preferences as utility functions of the attribute values.

Let the symbol $\succeq$ denote the user's preference order, e.g. $A \succeq B$ means "A is preferred or indifferent to B". According to Utility Theory, for a given MADP, there exists a utility function $U : O \to \Re$, such that for any two possible products O and $\bar{O} \in O$,

$$O \succeq \bar{O} \Longleftrightarrow U(O) \geq U(\bar{O}) \tag{16.1}$$

More specifically, a product O can be represented by a set of attribute values $\langle X_1 = x_1, \cdots, X_n = x_n \rangle$ (in short as $\langle x_1, \cdots, x_n \rangle$), thus the above formula can be rewritten as

$$\langle x_1, \cdots, x_n \rangle \succeq \langle \bar{x}_1, \cdots, \bar{x}_n \rangle \Longleftrightarrow U(\langle x_1, \cdots, x_n \rangle) \geq U(\langle \bar{x}_1, \cdots, \bar{x}_n) \tag{16.2}$$

If the utility function is given, the degree of preference for each product is characterized as a numerical utility and the preference order of all products is given by these utility values.

Finding the proper utility function U to represent users' preferences precisely is a challenging task. While in theory the utility function can be used in any style to represent user preferences, a special case is commonly used to reduce computation effort. If the attributes are mutually preferentially independent[2] based on the utility theory, the utility function has the additive form as follows:

$$U(\langle x_1, \cdots, x_n \rangle) = \sum_{i=1}^{n} w_i v_i(x_i) \tag{16.3}$$

where v_i is a value function of attribute X_i with range $[0,1]$, and w_i is the weight value of X_i satisfying $\sum_{i=1}^{n} w_i = 1$. In other words, the utility function for a product O is the weighted sum of the utility functions for each of its attributes. The weight value for each attribute can be given as default value $1/n$, and we can allow the user to specify the weight values of some attributes. The value function v_i can be determined to satisfy the user's preferences related to the attribute X_i . Usually a linear function with the form $v_i = ax_i + b$ is enough to represent the user's preference on each attribute.

Once the utility function for each product is determined, we are able to rank all the products based on their overall utilities and select the top K products with highest utilities as the recommendation set. In practice, we assume that the attributes of any product are mutually preferentially independent, so the additive form of the utility function can always be applied.

[2] An attribute X is said to be preferentially independent of another attribute Y if preferences for levels of attribute X do not depend on the level of attribute Y. If Y is also preferentially independent of X, then the two attributes are said to be mutually preferentially independent. See more details in [22].

16.2.3 The Accuracy, Confidence, Effort Framework

As mentioned in the introduction, our search for a multi-objective requirement framework is to find a basis for the derivation of the design guidelines. We look for criteria that a product recommender must satisfy in order to achieve maximum user satisfaction and their willingness to use the system. We give a more precise definition of the ACE (Accuracy, Confidence, Effort) framework as well as ways of measuring these variables.

Accuracy refers to the objective accuracy of a recommender. For rating-based systems, the most often used measure is the mean absolute error (or MAE) [1]. It is measured by an offline procedure known as leave-one-out on a previously acquired dataset. Leave-one-out involves leaving one rating out and then trying to predict it with the recommender algorithm being evaluated. The predicted rating is then compared with the real rating and the difference in absolute value is computed. The procedure is repeated for all the ratings and an average of all the errors is called the Mean Absolute Error.

Recommendations generated by utility-based systems are based on users' preference profiles. Since such profiles cannot be simulated, offline methods to measure accuracy are not possible. One method commonly used in this field is the switching task as defined in [48]. It measures how often users' truly preferred items are selected from the ones recommended to them. For example, if 70 out of 100 users found their preferred items during the recommendation process without changing their decisions after the experiment administrator presented all of the available options to them, then we say the accuracy of the system is 70%. This implies an experimental procedure where each user first interacts with a recommender to pick items for her consideration set. In a second phase of this procedure, the experiment administrator will show all of the available items and then ask her whether she finds the items in the consideration set still attractive. If she switches to other items, the system has failed to help her make an accurate decision. Such procedures were already employed in consumer decision research to measure decision quality, known as the switching task [20]. The fraction of users that switch to other items, called the switching rate, gives the only precise account of decision accuracy for a personalized recommender tool. However, as they are very time consuming, switching tasks are only performed in carefully defined empirical studies.

User confidence, the second evaluation criterion in our ACE framework, is the system's ability to inspire users to select the items recommended to them. It is a subjective variable and can only be assessed using a post-study questionnaire that asks users to indicate their agreement with statements such as: *I am confident that the items recommended to me are the ones I look for*. When the objective accuracy cannot be feasibly obtained, user confidence can be used to assess the perceived accuracy of a system. Perceived accuracy is strongly correlated to actual accuracy, but also influenced by other aspects of the recommender: whether the user is given enough possibilities and sufficiently involved in preference eliciation and revision, and whether the display of results convinces the user that the best results are being found.

By *user effort*, we refer to the actual task time users take to finish an instructed action, such as the preference elicitation and preference revision procedures and the time they take to construct the consideration set. As an alternative to elapsed time, we can also measure the number of interaction cycles for any of the interaction activities since this measure is independent of the working habits of individual users.

Throughout this chapter, we will analyze the 3 major components of example-based recommender systems: preference elicitation, revision and result display. We use accuracy, confidence and effort to evaluate different techniques and derive a set of 11 guidelines for the design of successful example-based recommender systems, and compare them against a model of user behavior that provides a unified motivation.

We believe that confidence is a major factor for the success of any recommender system, and therefore suggest that the ACE framework could be useful to evaluate other types of recommender systems as well.

16.2.4 Organization of this Chapter

We structure the guidelines according to the generic components of the model presented in Figure 16.1: initial preference elicitation (step 1), preference revision (step 4), and display strategy (step 2). The rest of this article is organized as follows: section 16.3 (Related Work) reviews design guidelines for preference elicitation and personalized product recommender tools in other fields; Section 16.4 (Initial Preference Elicitation) presents guidelines for motivating users to state their initial preferences as accurately as possible using principles from behavioral decision research; Section 16.5 (Stimulating Preference Elicitation with Examples) continues the discussion on preference elicitation and identifies concrete methods to help users state complete and sound preferences; Section 16.6 (Preference Revision) describes strategies to help users resolve conflicting preferences and perform tradeoff decisions; Section 16.7 (Display Strategies) presents guidelines for device display strategies that achieve a good balance between minimizing user's information processing effort and maximizing decision accuracy and user confidence; Section 16.8 presents a model of user behavior that provides a rationale for the guidelines; Section 16.9 concludes this article.

16.3 Related Work

16.3.1 Types of Recommenders

Two types of recommender systems have been broadly recognized in the field relative to the way systems gather and build user preference profiles: those based on

users' explicitly stated preferences (called preference-based recommenders) and those based on users navigation or purchase behaviors (called behavior-based recommenders). For the treatment of behavior-based recommenders, which generate recommendations based on users' accumulated interaction behaviors such as the items users have examined and purchased, please refer to [75], and to [26, 55] for demographic-based recommenders. Four types of preference-based recommenders exist: rating-based, case-based, utility-based and critiquing-based. Please see other ways to classify recommender systems ([1, 7]).

16.3.2 Rating-based Systems

Users explicitly express their preferences (even though they may not know it) by giving either binary or multi-scale scores to items that they experienced. Either the system proposes a user to rate a set of items or users will select on their own a set of items to rate. These initial ratings constitute the user profile. Systems that fall into this category are most commonly known as collaborative recommenders due to the fact that the user is recommended items that people with similar tastes and preferences liked in the past. For this reason, this type of system is also called social recommender. The details of how the collaborative algorithms work can be found in [1]. Lately some websites, such as tripadvisor.com, started collecting users' ratings on multiple attributes of an item to obtain a more refined preference profile.

16.3.3 Case-based Systems

This type of system recommends items that are similar to what users have indicated as interesting. A product is treated as a case having multiple attributes. Content-based [1] and case-based technologies [7] are used to analyze the attribute values of available products and the stated preferences of a user, and then identify one or several best-ranked options according to a ranking scheme.

16.3.4 Utility-based Systems

Utility-based recommenders, such as example critiquing recommenders, propose items based on users' stated preferences on multi-attribute products. Multi-attribute products refer to the encoding scheme used to represent all available data with the same set of attributes $\{a_1, \cdots, a_k\}$ where each attribute a_i can take any value v, from a domain of values $d(a_i)$. For example, a data set comprising all digital cameras in an e-store can be represented by the same set of attributes: manufacturer, price, resolution, optical zoom, memory, screen size, thickness, weight, etc. The list of at-

tributes as well as the domain range varies among product domains. We assume that users' preferences depend entirely on the values of these attributes so that two items that are identical in all attributes would be equally preferred. Furthermore, products considered here, such as digital cameras, portable PCs, or apartments, demand a significant financial commitment. They are called high involvement products because users are expected to possess a reasonable amount of willingness to interact with the system, participate in the selection process and expend a certain amount of effort to process information [62]. Users are also expected to exhibit slightly more complex decision behaviors in such environments than they would in selecting a simpler item, such as a book, a DVD, or a news article.

Tools using these technologies have also been referred to as knowledge-based recommenders [7] and utility-based decision support interface systems (DSIS) [62]. Utility refers to multi-attribute utility theory that such technologies use to calculate a product's suitability to a user's stated preferences. A related technology, specializing in searching configurable products, uses constraint satisfaction technology [45]. The difference between utility- and case-based systems lies in the notion of utility. While the weight of user preference is important for UBR, it is ignored in CBR. Further, the notion of value tradeoff is an essential part of decision making for UBRs. Essentially UBRs recommend decisions, rather than just similar products. See more details about this topic in the section on tradeoff reasoning.

16.3.5 Critiquing-based Systems

Both case- and utility-based recommenders can be improved by adding the additional interaction step of critiquing. A critiquing-based product recommender simulates an artificial salesperson that recommends options based on users' current preferences and then elicits their feedback in the form of critiques such as "I would like something cheaper" or "with faster processor speed." These critiques help the agent improve its accuracy in predicting users' needs in the next recommendation cycle. For a user to finally identify her ideal product, a number of such cycles are often required. Since users are unlikely to state all of their preferences up front, especially for products that are unfamiliar to them, the preference critiquing agent is an effective way to help them incrementally construct their preference model and refine it as they see more options.

16.3.6 Other Design Guidelines

This chapter derives a set of usability design guidelines based on our recent and related works in the domain of interaction technologies for preference-based search and recommender tools. Therefore, we will not review the related works that contribute to the accumulated list of guidelines in this section. Rather, discussions of

these works will be provided throughout the chapter in areas that correspond to the established guidelines. We will, however, describe two papers which share our goal of deriving good design guidelines for decision support systems. The first cited work proposes a set of recommendations derived from marketing research in order to increase users' motivation to interact with the recommender agent of an online store and its website. The second cited work describes a list of "building code" guidelines for an effective preference construction procedure in decision problems involving higher-stake outcomes.

Based on a critical but well-justified view of preference-based product search and recommender tools available at the time, Spiekermann and Paraschiv proposed a set of nine design recommendations to augment and stimulate the interaction readiness between a user and such systems [62], but there is no overlap between these recommendations and our guidelines.

Much of the basis for these recommendations are insights from marketing literature relative to information search, and perceived risk theory that defines a user's readiness to interact with a product recommender as her motivation to reduce the functional, financial, and emotional risks associated with the purchase decision. The design recommendations were therefore derived from methods concerned with reducing user risk in all of these dimensions. Compared to our work, this is a "horizontal" approach that covers the general design of an entire e-commerce website, in which the recommender agent is the principal technical component. We perform an in-depth examination of the recommender engine's interaction technologies using a more "vertical" approach, ensuring that consumers are offered the optimal usability support for preference elicitation and decision making when interacting with the product recommender tools.

Although the subject matter is more concerned with making high-stake decisions, the valuation process described by Payne et al. [41] is similar to our goal of addressing the needs and preferences of a consumer facing a purchase decision. Their work challenged many of the traditional assumptions about well-defined and preconceived preferences and strongly promoted a theory of preference construction. Under that new framework, the authors proposed a set of "building codes" (similar to guidelines) to help professional decision makers establish value functions and make high quality decisions. Their discussions on the nature of human preferences, the way people construct and measure preferences, and how to face tradeoffs to obtain rational decisions are especially influential to our work. References to the specific details of this and other works in behavior decision research will be given at relevant areas in this chapter.

16.4 Initial Preference Elicitation

Example critiquing systems are decision tools to help people find multi-attribute products such as flights, digital cameras, tablet PCs, etc. We use $P = \{(a_i, w_i)\}$ where $1 \leq i \leq n$ to specify a user's preferences over a total of n attributes of her

desired product for utility-based recommenders. Furthermore, a_i represents the desired characteristics on the i^{th} attribute and w_i the degree to which such characteristics should be satisfied. This model is also known as the value function in [24]. It is called the preference model in most systems discussed here. Some methods assume the same weights for all attributes and therefore do not elicit such information [9, 53]. Preference elicitation (also known as query specification) is the initial acquisition of this model for a given user.

It would seem apparent that a user's preferences could be elicited by simply asking her to state them. Many online search tools use a form-filling type of graphical user interface or a natural language dialog system to collect such information. Users are asked to state their preferences on every aspect, such as departure and arrival date and time, airlines, intermediate airports, etc., and are given the impression that all fields must be filled. We call such approaches non-incremental since all preferences must be obtained up-front.

To understand why this simple-minded approach does not work, we turn to behavior decision theory literature to understand the nature of user preference expression. According to the adaptive decision theory [40], user preferences are inherently adaptive and constructive depending on the current decision task and environment. Due to this nature, users may lack the motivation to answer demanding initial elicitation questions prior to any perceived benefits [62], and they may not have the domain knowledge to answer the questions correctly. In another words, if a system imposes a heavy elicitation process in the beginning, the preferences obtained in this way are likely to be uncertain and erroneous.

Similar literature reveals that users' preferences are context-dependent and are constructed gradually as a user is exposed to more domain information regarding his/her desired product [40, 41]. For example, Tversky et al. reported a user study about asking subjects to buy a microwave oven [66]. Participants were divided into 2 groups with 60 users each. In the first group, each user was asked to choose between an Emerson priced at $110 and a Panasonic priced at $180. Both items were on sale, and these prices represented a discount of one third off the regular price. In this case only 43% of the users chose the more expensive Panasonic at $180. A second group was presented with the same choices except with an even more expensive item: a $200 Panasonic, which represented a 10% discount from its original $220 price. In this context, 60% of the users chose the Panasonic priced at $180. In other words, more subjects prefer the same item just because the context has changed. This finding demonstrates that people are not likely to reveal their preferences as if they were innate to them, but construct them in an incremental and adaptive way based on contextual information.

To verify if these classical behavioral theories shed light on user preference expression in online environments, we conducted some empirical studies. 22 subjects were asked to interact with a preference elicitation interface [67]. The average user stated only preferences on 2.1 attributes out of a total of 10 when they had the possibility to freely state all preferences. Their preferences only increased to 4.19 attributes at the time of selecting the final choice.

Another study was conducted soon afterwards to further confirm the finding that users were unlikely to state all of their preferences in the beginning. It compared how users perform product search tasks in terms of decision accuracy and effort while interacting with a non-incremental procedure versus the incremental one [68]. With the former procedure, users were required to specify all of their preferences in a single graphical user interface (called form filling), whereas with the latter approach, each preference was constructed by a user. All 40 users were randomly and evenly divided into two groups, and each group was assigned to one system (either non-incremental or incremental approach) to evaluate. In the non-incremental approach, users stated on average 7.5 preferences on the 10 attributes, while the results for the incremental approach remained the same. However, the non-incremental approach had an accuracy of only 25%, while the incremental method achieved 70% accuracy with comparable user effort. That is, only 25% of users found the target products when they had to state preferences in the non-incremental form-filling style. Thus, while the non-incremental approach may produce the required data, the quality of what has been elicited may be questionable. There is no guarantee that users will provide correct and consistent answers on attributes for which their preferences are still uncertain.

Similar findings were reported in preference elicitation for collaborative filtering based recommender systems. McNee et al. compared three interface strategies for eliciting movie ratings from new users [34]. In the first strategy, the system asked the user to rate movies that were chosen based on entropy comparisons to obtain a maximally informative preference model. That is, the system decided which movies users should initially rate. In another strategy, users were allowed to freely propose movies they wanted to rate. In a mixed strategy, the user had both possibilities. A total of 225 new users participated in the experiment which found that the user-controlled strategy obtained the best recommendation accuracy compared to the other two strategies in spite of a lower number of ratings completed by each user (14 vs. 36 for the system-controlled interface). Furthermore, the user-controlled interface was more likely to motivate users to return to the system to assign more ratings. This demonstrates that a higher level of user control over the amount of interaction effort gives rise to more accurate preference models.

Incremental elicitation methods, as described in [1] and [5], can be used to revise users' stated preferences (or value functions) in general cases. Other incremental methods improve decision quality in specific areas. In [6, 36, 42], researchers addressed preference uncertainty by emphasizing the importance of displaying a diverse set of options in the early stage of user-system interaction. Faltings et al. [13, 69, 70] described ways to stimulate users to express more preferences in order to help them increase decision accuracy. Pu and Faltings showed how additional preference information can be acquired via preference revision, especially as users perform tradeoff tasks [45]. More detail on these topics will be given in Sections 16.5.2 and 16.6.

Based on both classical and recent empirical findings, we have derived several design guidelines concerning the initial preference elicitation process:

Guideline 1 (any effort): *Consider novice users' preference fluency. Allow them to reveal preferences incrementally. It is best to elicit initial preferences that concern them the most and choose an effort level that is compatible with their knowledge and experience of available options.*

A rigid elicitation procedure obtains users' preferences using a system predesigned order of elicitation. When users are forced to formulate preferences in a particular order or using attributes that do not correspond to their actual objectives, they can fall prey to incorrectly formulating *means objectives* that prevent them from achieving their fundamental objectives [23]. For example, when planning a trip by airplanes a user may be prompted by the system to first enter her preferred airline company before being allowed to state her preferences on the flight. Thus, a user with the fundamental objective of flying at a certain hour has to formulate a different objective, the airline company to use, as a *means* of achieving the fundamental objective. This new objective is therefore called a means objective. To correctly translate the true objective into means objectives, the user needs to have detailed knowledge of the product offering, in this case the flight time tables offered by the different airlines. Since her knowledge in this area can be poor in the beginning, the system may fail to find the most optimal results.

Thus we propose

Guideline 2 (any order): *Consider allowing users to state their preferences in any order they choose.*

An elicitation procedure can also be too rigid by not allowing users to state preferences on a sufficiently rich set of attributes. Consider this example [70]: in a travel planning system, suppose that the user's objective is to be at his destination at 15:00, but that the tool only allows search by the desired departure time. The user might erroneously believe that the trip requires a plane change and takes about 5 hours, thus forming a means objective of a 10:00 departure in order to answer the question. However, the best option might be a new direct flight that leaves at 12:30 and arrives at 14:30.

Means objectives are intermediary goals formulated to achieve fundamental decision objectives. Assume that a user has a preference on attribute a_i (e.g., the arrival time), but the tool requires expressing preferences on attribute a_j (e.g., the departure time). Using beliefs about the available products, the user will estimate a transfer function $t_i(a_j)$ that maps values of attribute a_i to values of attribute a_j (e.g., arrival time = departure time + 5 hours) The true objective $p(a_i)$ is then translated into the means objective $q(a_j) = p(t(a_j))$. When the transfer function is inaccurate, the means objective often leads to very inaccurate results.

Note also that unless there is a strong correlation between attributes a_i and a_j, an accurate transfer function may not even exist. A recent visit to a comparison shopping website (www.pricegrabber.com) showed that the site does not include

the weight attribute of Tablet PCs in the search field, even though such information is encoded in their catalog. For many portability conscious users, the weight of a Tablet PC is one of the most important decision parameters. When unable to examine those products directly based on the weight attribute, the consumer might infer that weight is correlated to the screen size and perhaps the amount of storage in the hard drive. Consequently, she may consider searching for a Tablet PC with less disk space and a smaller screen size than she actually desires. These users could potentially make unnecessary sacrifices because many manufacturers are now able to offer light-weight Tablet PCs with considerable disk space and comfortable screen sizes.

Thus we propose

Guideline 3 (any preference): *Consider allowing users to state preferences on any attributes they choose.*

When designing interfaces based on these three guidelines, a good balance between giving the user the maximum amount of control, yet not overwhelming her with interface complexity is necessary. We recommend the use of adaptive interfaces, where users can click on attributes for which they want to state preferences or leave them unclicked or on default values if they do not have strong preferences at that point. Alternatively, designers can use a "ramp-up" approach for the initial query specification, where users specify preferences over a small number of attributes initially and are prompted to specify more once some results are displayed.

16.5 Stimulating Preference Expression with Examples

An effective elicitation tool should collect users' preferences in an incremental and flexible way. In practice, we are also interested in mechanisms that stimulate users to state preferences.

Undesirable means objectives arise mainly due to users' unfamiliarity with available options. At the same time, it has been observed in behavior theory that people find it easier to construct a model of their preferences when considering examples of actual options [41]. According to Tversky [65], people do not maximize a precomputed preference order, but construct their choices in light of the available options. These classical theories seem to justify why example critiquing is an effective method for building preference elicitation tools [43, 47].

This method is called example critiquing since users build their preferences by critiquing the example products that are shown to them. Users initially state preferences on any number of attribute values that they determine to be relevant. From that point on, the system engages the user in successive cycles of "examples and critiques": it displays a set of example products and elicits user feedback in the form of critiques such as "I like this laptop computer, but with more disk space". The

critiques determine the set of examples to display next. This interaction terminates when users are able to identify their preferred products.

Users can quickly build their preferences by critiquing the example products shown to them. As users only have to state critiques rather than preferences, the model requires little effort from users. Most importantly, the example critiquing paradigm appears to satisfy both the goal of educating users with available options and the goal of stimulating them to construct their preferences in the context of given examples.

We will review a suite of critiquing systems and derive guidelines that illustrate the most effective components of this method. For additional information of this subject, please refer to the chapter by Lorraine McGinty and James Reilly entitled "On the Evolution of Critiquing Recommenders" (in this book).

Example critiquing was first mentioned in [72] as a new interface paradigm for database access, especially for novice users to specify queries. Recently, example critiquing has been used in two principal forms by several researchers: those supporting product catalog navigation and those supporting product search based on an explicit preference model.

In the first type of system, for example the FindMe systems [8, 9], search is described as a combination of search and browsing called assisted browsing. The system first retrieves and displays the best matching product from the database based on a user's initial query. It then retrieves other products based on the user's critiques of the current best item. The interface implementing the critiquing model is called tweaking, a technique that allows users to express preferences with respect to a current example, such as "look for an apartment similar to this, but with a better ambiance." According to this concept, a user navigates in the space of available products by tweaking the current best option to find her target choice. The preference model is implicitly represented by the current best product, i.e., what a user chooses reflects her preference of the attribute values. Reilly et al. have recently proposed dynamic critiquing [53] based on some improvements of the tweaking model. In addition to the unit-value tweaking operators, compound critiques allow users to choose products which differ from the current best item in two or more attribute values. For example, the system would suggest a digital camera based on the initial query. It also recommends cameras produced by different manufacturers, with less optical zoom, but with more storage. Compound critiques are generated by the Apriori algorithm [2] and allow users to navigate to their target choice in bigger steps. In fact, users who more frequently used the compound critiques were able to reduce their interaction cycles from 29 to 6 in a study involving real users [32].

In the second type of example-critiquing systems, an explicit preference model is maintained. Each user feedback in the form of a critique is added to the model to refine the original preference model. An example of a system with explicit preference models is the SmartClient system used for travel planning [43, 63]. It shows up to 30 examples of travel itineraries as soon as a set of initial preferences have been established. By critiquing the examples, users state additional preferences. These preferences are accumulated in a model that is visible to the user through the interface (see the bottom panel under "Preferences" of Figure 6 in [64]) and can be

revised at any time. ATA [28], ExpertClerk [60], the Adaptive Place Advisor [16], and the incremental dynamic critiquing systems function similarly [30]. The advantage of maintaining an explicit model is to avoid recommending products which have already been ruled out by the users. Another advantage is that a system can suggest products whose preferences are still missing in the stated model, as is further discussed in section 16.5.2.

Therefore, to educate users about the domain knowledge and stimulate them to construct complete and sound preferences, we propose the following guideline:

Guideline 4: *Consider showing example options to help users gain preference fluency.*

16.5.1 How Many Examples to Show

Two issues are critical in designing effective example-based interfaces: how many examples and which examples to show in the display. Faltings el al. investigated the minimum number of items to display so that the target choice is included even when the preference model is inaccurate [14]. Various preference models were analyzed. If preferences are expressed by numerical utility functions that differ from the true preferences by a factor of at most ε and they are combined using either the weighted sum or the min-max rule, then

$$t = \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^d \tag{16.4}$$

where d is the maximum number of stated preferences, and t is the number of displayed items so that the target solution is guaranteed to be included. Since this number is independent of the total number of available items, this technique of compensating inaccurate preferences by showing a sufficient number of solutions scales to very large collections. For a moderate number (up to 5) of preferences, the correct amount of display items typically falls between 5 and 20. When the preference model becomes more complex, inaccuracies have much larger effects. A much larger number of examples is required to cover the model inaccuracy.

16.5.2 What Examples to Show

The most obvious examples to include in the display are those that best match the users' current preferences. However, this strategy proves to be insufficient to guarantee optimality. Since most users are often uncertain about their preferences and are more likely to construct them as options are shown to them, it becomes important

for a recommender system to guide the user to develop a preference model that is as complete and accurate as possible. However, it is important to keep the initiative to state more preferences on the user's side. Therefore we call examples chosen to stimulate users to state preferences *suggestions*. We present two suggestion strategies: diversity- and model-based techniques.

The ATA system was the first to show suggestions [28], which were extreme-valued examples where some attributes, for example departure time or price, took extreme values such as earliest or cheapest. However, a problem with this technique is that extreme options are not likely to appeal to many users. For example, a user looking for a digital camera with good resolution might not want to consider a camera that offers 4 times the usual resolution but also has 4 times the usual weight and price. In fact, a tool that suggests this option will discourage the user from even asking for such a feature, since it implies that high resolution can only exist at the expense of many other advantages.

Thus, it is better to select the suggestions among examples that are already good given the currently known preferences, and focus on showing *diverse* rather than extreme examples. Bradley and Smyth were the first to recognize the need to recommend diverse examples, especially in the early stage of using a recommender tool [6]. They proposed the bounded greedy algorithm for retrieving the set of cases most similar to a user's query, but at the same time most diverse among themselves. Thus, instead of picking the k best examples according to the preference ranking $r(x)$, a measure $d(x,Y)$ is used to calculate the relative diversity of an example x from the already selected set Y according to a weighted sum

$$s(x,Y) = \alpha r(x) + (1-\alpha)d(x,Y) \tag{16.5}$$

where α can be varied to account for varying importance of optimality and diversity. For example, as a user approaches the final target, α can be set to a higher value (e.g. 0.75 in the experiment setup) so that the system emphasizes the display set's similarity rather than diversity. In their implementations, the ranking $r(x)$ is the similarity $sim(x,t)$ of x to an ideal example t on a scale of 0 to 1, and the relative diversity is derived as

$$d(x,Y) = 1 - \frac{1}{|Y|}\sum_{y\in Y} sim(x,y) \tag{16.6}$$

The performance of diversity generation was evaluated in simulations in terms of its relative benefit, i.e. the maximum gain in diversity achieved by giving up similarity [61]. Subsequently, McSherry has shown that diversity can often be increased without sacrificing similarity [36]. A threshold t was fixed on the ranking function, and then a maximally diverse subset among all products x for which $r(x) > t$ was selected. When k options are shown, the threshold might be chosen as the value of the k-th best option, thus allowing no decrease in similarity, or at some value that does allow a certain decrease.

We thus propose the following guideline:

Guideline 5: *Consider showing diverse examples to stimulate preference expression, especially when users are still uncertain about their final preferences.*

The adaptive search algorithm used in [33] alternates between a strategy that emphasizes similarity and one that emphasizes diversity to implement the interaction "show me more like this" by varying the α in the ranking measure. At each point, a set of example products is displayed and the user is instructed to choose her most preferred option among them. Whenever the user chooses the same option twice consecutively, the system considers diversity when proposing the next examples in order to refocus the search. Otherwise, the system assumes that the user is making progress and it continues to suggest new options based on optimality. Evaluations with simulated users show that this technique is likely to reduce the length of the recommendation cycles by up to 76% compared to the pure similarity-based recommender.

More recent work on diversity was motivated by the desire to compensate for users' preference uncertainty [42] and to cover different topic interests in collaborative filtering recommenders [74]. For general preference models, it is less clear how to define a diversity measure. Viappiani et al. considered the user's motivation to state additional preferences when a suggestion is displayed [69, 70, 50]. A suggestion is a choice that may not be optimal under the current preference model, but should provide a high likelihood of optimality when an additional preference is added. For example, a user may add "an apartment with a balcony" preference after seeing examples of such apartments. 40 (9 females) subjects from 9 different nationalities took part in a user study to search for an apartment. The experiment's results show that the use of suggestions almost doubled decision accuracy and allowed the user to find the most preferred option 80% of the time. A user is likely to be opportunistic and will only bother to formulate new preferences if she believes that this might lead to a better choice. Thus, they propose the following *look-ahead principle* [69, 70, 50]:

Guideline 6: *Consider suggesting options that may not be optimal under the current preference model, but have a high likelihood of optimality when additional preferences are added.*

The look-ahead principle can be applied to constructing model-based suggestions by explicitly computing, for each attribute a_i, a difference measure $diff(a_i,x)$ that corresponds to the probability that a preference on this attribute would make option x most preferred. Items are then ranked according to the expected difference measure over all possible attributes:

$$F_a(x) = \sum_{a_i \in A} P_{a_i} diff(a_i,x) \tag{16.7}$$

where P_{a_i} is the probability that the user is motivated to state a preference on attribute a_i. Such probabilities are summed over all attributes for which the user has not yet expressed a preference. The best suggestions to display are therefore those items possessing the highest probability of becoming optimal after considering hidden preferences. It is possible to adapt these techniques to generate a set of suggestions that jointly maximize the probability of an optimal item. More details are given in [13, 50]. To investigate the importance of suggestions in producing accurate decisions, several empirical user studies were carried out [50, 69, 67]. One was conducted in an unsupervised setting, where users' behavior was monitored on a publicly accessible online system. The scientists conducting the experiment collected logs from 63 active users who went through several cycles of preference revision. Another study was carried out in a supervised setting. The scientists recruited 40 volunteers and divided them into two groups. One group evaluated the interface with model-based suggestions, and another group evaluated the one without. Both user studies showed the significant effects of using these model-based suggestions: users who used the suggestion interfaces stated significantly more preferences than those who did not (an increase of 2.09 preferences vs. only 0.62 without suggestions, $p < 0.01$, in supervised studies [69, 67], and an increase of 1.46 vs. 0.64 without suggestions, $p < 0.002$, for online users [69, 50]) and users who used the suggestion interfaces also reached significantly higher decision accuracy (80 vs. 45 percent without suggestions, $p < 0.01$, in supervised user studies [50]).

16.6 Preference Revision

Preference revision is the process of changing one or more desired characteristics of a product that a user has stated previously, the degree to which such characteristics should be satisfied, or any combination of the two. In [48] 28 subjects (10 females) were recruited to participate in a user study in which the user was asked to find his or her most preferred apartment from a list of available candidates. The user's preferences could be specified on a total of six attributes: type, price, area, bathroom, kitchen and distance to work place. Each participant was first asked to make a choice, and then used the decision aid tool to perform tradeoffs among his/her preferences until the desired item was chosen. In this user study, every user changed at least one initial preference during the entire search process for finding a product. Many users change preferences because there is rarely an outcome that satisfies all of the initial preferences. Two frequently encountered cases often require preference revision: 1) when a user cannot find an outcome that satisfies all of her stated preferences and must choose a partially satisfied one, or 2) when a user has too many possibilities and must further narrow down the space of solutions. Even though both activities can be treated as the process of query refinement, the real challenge is to help users specify the correct query in order to find the target item. Here we present a unified framework of treating both cases as a *tradeoff process* because finding an

acceptable solution requires choosing an outcome that is desirable in some respects but perhaps not so attractive in others.

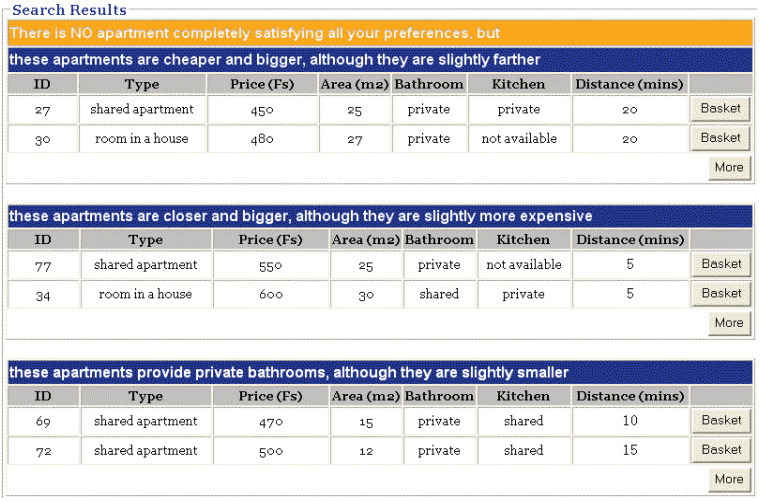
16.6.1 Preference Conflicts and Partial Satisfaction

A user who inputs a query for a spacious apartment with a low price range and obtains "nothing found" as a reply, learns very little about how to state more suitable preferences.

The current industry practice manages preference conflicts by browsing-based interaction techniques. A user is only allowed to enter her preferences one at a time starting from the point where all of the product space is available. As she specifies more preferences, she essentially drills down to a sub product space until either she selects her target in the displayed options or no product space remains. For example, if someone desires a notebook with minimal weight (less than 2 kilos), then after specifying the weight requirement, she is only allowed to choose those notebooks weighing less than 2 kilos. If the price of these lightweight notebooks is very high, she is likely to miss a tradeoff alternative that may weigh 2.5 kilos and cost much less. This interaction style has become very popular in comparison shopping websites (see www.shopping.com, www.pricegrabber.com, www.yahoo.shopping.com). As the system designers have prevented users from specifying conflicting preferences, this interaction style is very limited. Users are unable to specify contextual preferences and especially tradeoffs among several attributes. If a user enters the set of preferences successively for each attribute, the space of matching products could suddenly become null with the message "no matching products can been found." At this point, the user may not know which attribute value to revise among the set of values that she has specified so far, requiring her to backtrack several steps and try different combinations of preference values on the concerned attributes.

A more sensible method, such as the one used in SmartClient [43, 64], manages a user's preference conflicts by first allowing her to state all of her preferences and then showing her options that maximally satisfy subsets of the stated preferences based on partial constraint satisfaction techniques [15]. These maximally satisfied products educate users about available options and facilitate them in specifying more reasonable preferences. In the same spirit, McCarthy et al. propose to educate users about product knowledge by explaining the products that do exist instead of justifying why the system failed to produce a satisfactory outcome [31]. FindMe systems rely on background information from the product catalog and explain preference conflicts on a higher level [8, 9]. In the case of a user wanting both a fuel-efficient and high-powered car, FindMe attempts to illustrate the tradeoff between horsepower and fuel efficiency. This method of showing partially satisfied solutions is also called soft navigation by Stolze [63].

To convince users of the partially satisfied results, we can also adopt the approach used by activedecision.com. It not only shows the partial solutions, but also explains in detail how the system satisfies some of users' preferences and not oth-



Search Results

There is NO apartment completely satisfying all your preferences. but

these apartments are cheaper and bigger, although they are slightly farther

ID	Type	Price (Fs)	Area (m2)	Bathroom	Kitchen	Distance (mins)	
27	shared apartment	450	25	private	private	20	Basket
30	room in a house	480	27	private	not available	20	Basket

More

these apartments are closer and bigger, although they are slightly more expensive

ID	Type	Price (Fs)	Area (m2)	Bathroom	Kitchen	Distance (mins)	
77	shared apartment	550	25	private	not available	5	Basket
34	room in a house	600	30	shared	private	5	Basket

More

these apartments provide private bathrooms, although they are slightly smaller

ID	Type	Price (Fs)	Area (m2)	Bathroom	Kitchen	Distance (mins)	
69	shared apartment	470	15	private	shared	10	Basket
72	shared apartment	500	12	private	shared	15	Basket

More

Fig. 16.2: Partially Satisfied Products in an Organization Interface.

ers. A qualitative user survey about such explanation mechanisms was conducted in the form of a carefully constructed questionnaire, based on a series of hypotheses and corresponding applicable questions. 53 participants completed the survey, and most of them strongly agreed that the explanation components are more likely to inspire their trust in the recommended solutions [10]. In addition, an alternative explanation technique, the organization interface where partially satisfied products are grouped into a set of categories (Figure 16.2), was preferred by most subjects, compared to the traditional method where each item is shown along with an explanation construct [10]. A follow-up comparative user study (with 72 participants) further proved that this interface method can significantly inspire competence-induced user trust in terms of the user's perceived competence, intention to return and intention to save effort (see some details of the experiment in section 7.3) [49].

Guideline 7: *Consider resolving preference conflicts by showing partially satisfied results with compromises clearly explained to the user.*

16.6.2 Tradeoff Assistance

As catalogs grow in size, it becomes increasingly difficult to find the target item. Users may achieve relatively low decision accuracy unless a tool helps them efficiently view and compare many potentially interesting products. Even though a recommender agent is able to improve decision quality by providing filtering and

comparison matrix components [20], a user can still face the bewildering task of selecting the right items to include in the consideration set.

Researchers in our group found that online tools could increase the level of decision accuracy by up to 57% by helping users select and compare options which share tradeoff properties [48]. 28 subjects (10 females) took part in the experiment; each of the participants was first asked to make a choice, and then use the decision aid tool to perform a set of tradeoff navigation tasks. The results showed that after a user has considered an item as the final candidate, the tool can help her/him reach higher decision accuracy by prompting them to see a set of tradeoff alternatives. The same example critiquing interfaces as discussed in Section 5 can be used to assist users to view tradeoff alternatives, for example, "I like this portable PC, but can I find something lighter?" This style of interaction is called tradeoff navigation and is enabled by the "modify" widget together with the "tweaking panel" (see Figure 4 in [47]). Tweaking (used in FindMe [8, 8]) was the first tool to implement this tradeoff assistance. It was originally designed to help users navigate to their targets by modifying stated preferences, one at a time. Example critiquing (used in SmartClient [48, 47]) is more intentional about its tradeoff support, especially for tradeoffs involving more than two participating attributes. In a single interaction, a user can state her desire to improve the values of certain attributes, compromise on others, or any combination of the two.

Reilly et al. introduced another style of tradeoff support with dynamic critiquing methods [53]. Critiques are directional feedback at the attribute level that users can select in order to improve a system's recommendation accuracy. For example, after recommending a Canon digital camera, the system may display "we have more matching cameras with the following: 1) less optimal zoom and thinner and lighter weight; 2) different manufacturer and lower resolution and cheaper; 3) larger screen size and more memory and heavier." Dynamic critiquing is an approach of automatically generating useful compound critiques so that users can indicate their preference on multiple attributes simultaneously. The experiment in [53] shows that the dynamic critiquing approach has the ability to reduce the interaction session length by up to 40% compared to the approach with only unit critiques.

Although originally designed to support navigation in recommender systems, the unit and compound critiques described in [53] correspond to the simple and complex tradeoffs defined in [47]. They are both mechanisms to help users compare and evaluate the recommended item with a set of tradeoff alternatives. However, the dynamic critiquing method provides system-proposed tradeoff support because it is the system which produces and suggests the tradeoff categories, whereas example critiquing provides a mechanism for users to initiate their own tradeoff navigation (called user motivated critiques in [11]).

A recent study from our group compared the performance of user-motivated vs. system-proposed approaches [11]. A total of 36 (5 females) volunteers participated in the experiment. It was performed in a within-subjects design, and each participant was asked to evaluate two interfaces with the respective two approaches one after the other. All three evaluation criteria stated in section 2.2 were used: decision accuracy, user interaction effort and user confidence. The results indicate that

the user-motivated tradeoff method enables users to achieve a higher level of decision accuracy with less cognitive effort, mainly due to its flexibility in allowing users to freely combine unit and compound critiques. In addition, the confidence in a choice made with the user-motivated critique method is higher, resulting in users' increased intention to purchase the product they have found and return to the agent in the future. We thus propose:

Guideline 8: *In addition to providing the search function, consider providing users with tradeoff assistance in the interface using either system-proposed or user motivated approaches. The latter is likely to provide users with more flexibility in choosing their tradeoff desires and thus enable them to achieve higher decision accuracy and confidence.*

16.7 Display Strategies

At least three display strategies are currently employed in preference-based search and recommender tools: recommending items one at a time, showing top k matching results (where k is a small number between 3 and 30), or displaying products with explanations on how ranking scores are computed. We discuss these various strategies using the accuracy, confidence, and effort evaluation framework discussed in Section 2.

16.7.1 Recommending One Item at a Time

The advantage of such recommender systems is that it is relatively easy to design the display, users are not likely to be overwhelmed by excessive information, and the interface can be easily adapted to small display devices such as mobile phones. The obvious disadvantage is that a user may not be able to find her target choice quickly. As mentioned in Section 5, a novice user's initial preferences are likely to be uncertain. Thus the initially recommended results may not include her target choice. Either a user has to interact with the system much longer due to the small result set, or if a user exhausts her interaction effort before reaching the final target, she is likely to achieve very low decision accuracy. Thus we propose:

Guideline 9: *Showing one search result or recommending one item at a time allows for a simple display strategy which can be easily adapted to small-display devices; however, it is likely to engage users in longer interaction sessions or only allow them to achieve relatively low decision accuracy.*

16.7.2 Recommending K best Items

Some product search tools present a set of top-k alternatives to the users. We call this style of display the k-best interface. Commercial tools employing this strategy can be found at ActiveDecision.com ($k > 10$). Academic prototypes include those used by SmartClient ($7 \leq k \leq 30$) [14, 47], ATA ($k = 3$) [28], ExpertClerk ($k = 3$) [60], FirstCase ($k = 3$) [37] and TopCase ($k = 3$) [38].

When k approaches 10, the issue of ordering the alternatives becomes important. The most commonly used method is to select the best k items based on how well they match users' stated preferences using utility scores (see multi-attribute utility theory [24]). We can also use the "k nearest neighbor" retrieval algorithm (or simply k-NN) [12] to rank the k items, such as those used in the case-based reasoning field [25]. The k items are displayed in descending order from the highest utility score or rank to the lowest (activedecision.com, SmartClient). This method has the advantages of displaying a relatively high number of options without overwhelming the users, preselecting the items based on how well they match the stated preferences of a user, and achieving relatively high decision accuracy [48, 47].

Pu and Kumar compared an example critiquing based system ($k = 7$ rank ordered by utility scores) with a system using the ranked list display method ($k = n$ rank ordered on user selected attribute values such as price) [47, 51]. 22 volunteers participated in the user study. Each of them was asked to test two interfaces (example critiquing and ranked list) in random order by performing a list of given tasks. The results showed that while users performed the instructed search tasks more easily using example critiquing (less task time and smaller error rate, with statistical significance) and achieved higher decision accuracy [48, 51], more of them expressed a higher level of confidence that the answers they found were correct for the ranked list interface. Further analysis of users' comments recorded during the user study revealed that the confidence issue depends largely on the way items were ordered and how many of them were displayed. Many users felt that the EC system (displaying only 7 items) was hiding something from them and that the results returned by the EC interface did not correspond to their ranking of products. With the help of a pilot study, it was observed that users generally did not scroll down to view the additional products displayed, but their confidence level increased and the interaction time was not affected. Therefore we suggest the following guideline for the top-k display strategy:

Guideline 10: *Displaying more products and ranking them in a natural order is likely to increase users' sense of control and confidence.*

16.7.3 Explanation Interfaces

When it comes to suggesting decisions, such as which camera to buy, the recommender system's ability to establish trust with users and convince them of its recommendations is a crucial factor for success. Researchers from our group started investigating the user confidence issue and other subjective factors in a more formal framework involving trust relationships between the system and the user. It is widely accepted that trust in a technological artifact (like the recommender agent) can also be conceptualized as competence, benevolence, and integrity, similar to trust in a person. Trust is further seen as a long term relationship between the user and the organization that the recommender system represents [10]. When a user trusts a recommender system, she is more likely to purchase items and return to the system in the future. A carefully designed qualitative survey with 53 users revealed that an important construct of trust formation is an interface's ability to explain its results [10], as mentioned in section 6.1.

The explanation interface can be implemented in various ways. For example, ActiveDecision.com uses the tool tip with a "why" label to explain how each of the recommended products matches a user's stated preferences, similar to the interface shown in Figure 16.3. Alternatively, it is possible to design an organization-based explanation interface where the best matching item is displayed at the top of the interface along with several categories of tradeoff alternatives [49]. Each category is labeled with a title explaining the characteristics of the items the respective category contains (Figure 16.4).

In order to understand whether the organization interface is a more effective way to explain recommendations, a significant-scale empirical study was conducted to compare the organization interface with the traditional "why" interface in a within-subjects design. A total of 72 volunteers (19 females) were recruited as participants in the user study. The results showed that the organization interface significantly increases user perception of its competence, which more effectively inspires uses' trust and enhances their intention to save cognitive effort and use the interface again in the future [49]. Moreover, the study found that the actual time spent looking for a product did not have a significant impact on users' subjective perceptions. This indicates that less time spent on the interface, while very important in reducing decision effort, cannot be used alone in predicting what users may subjectively experience. Five principles for the effective design of organization interfaces were developed and an algorithm was presented for generating the content of such interfaces [49]. Here we propose:

Guideline 11: *Consider designing interfaces that explain how ranking scores are computed because they are likely to inspire user trust.*

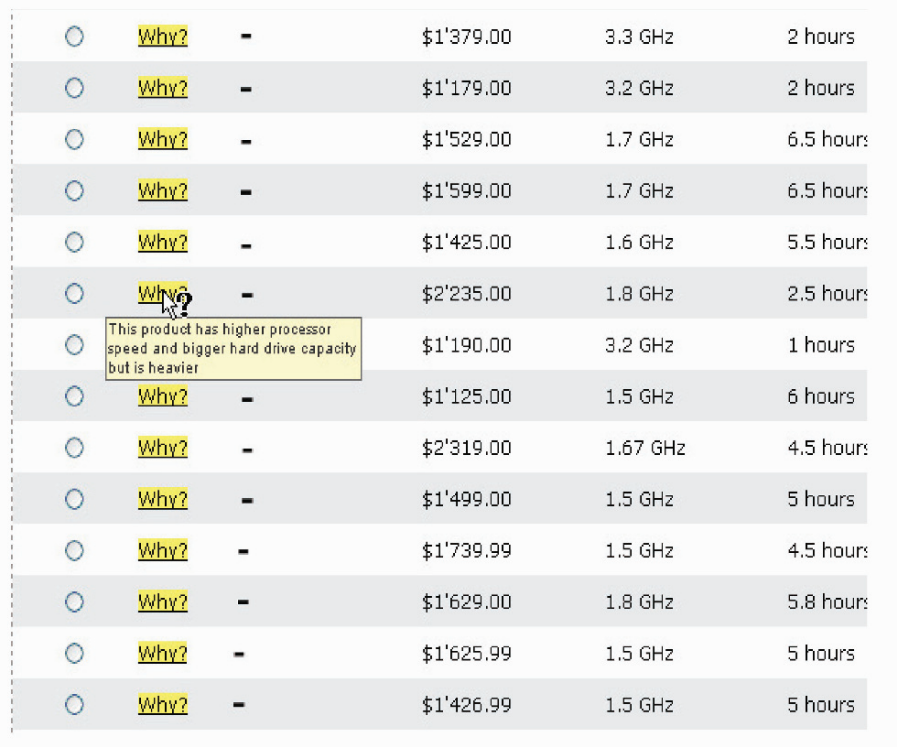
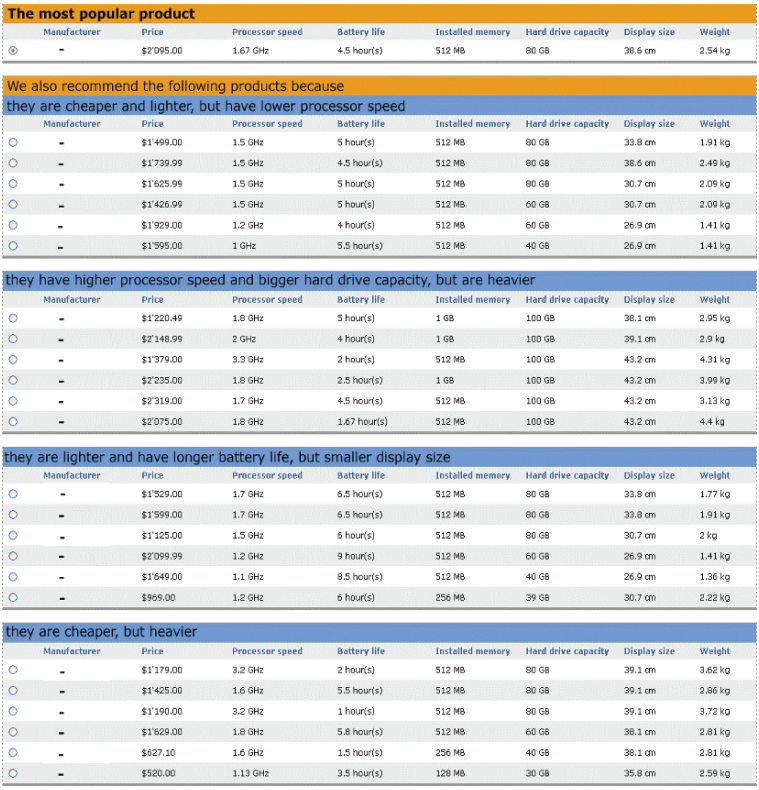


Fig. 16.3: A generic recommendation interface with simple "why" labels.

16.8 A Model for Rationalizing the Guidelines

We have developed a set of guidelines to ensure the design of usable product search tools relying on a general framework of three evaluation criteria: (i) decision accuracy, (ii) user interaction effort, and (iii) user decision confidence. We now develop a model of user behavior that allows us to rationalize the guidelines as a means of optimizing recommender system performance. We first consider the fact that product search tools must serve the needs of a significant and heterogeneous user population. They must adapt to the characteristics of individual users, in particular to their willingness to put in continuous interaction effort in order to obtain their desired results. In our theoretical model, we let e be the user's interaction effort, measured in interaction steps. With this effort, users hope to obtain an increasingly high confidence that they are finding the best option. We characterize this user confidence numerically as the accuracy that the user believes the system to have achieved, called the perceived accuracy a.

As each individual user has different aspirations for effort and perceived accuracy, we characterize each user by the three following parameters:



The most popular product

	Manufacturer	Price	Processor speed	Battery life	Installed memory	Hard drive capacity	Display size	Weight
◉	-	$2'095.00	1.67 GHz	4.5 hour(s)	512 MB	80 GB	38.6 cm	2.54 kg

We also recommend the following products because

they are cheaper and lighter, but have lower processor speed

	Manufacturer	Price	Processor speed	Battery life	Installed memory	Hard drive capacity	Display size	Weight
○	-	$1'499.00	1.5 GHz	5 hour(s)	512 MB	80 GB	33.8 cm	1.91 kg
○	-	$1'739.99	1.5 GHz	4.5 hour(s)	512 MB	80 GB	38.6 cm	2.49 kg
○	-	$1'625.99	1.5 GHz	5 hour(s)	512 MB	80 GB	30.7 cm	2.09 kg
○	-	$1'426.99	1.5 GHz	5 hour(s)	512 MB	60 GB	30.7 cm	2.09 kg
○	-	$1'929.00	1.2 GHz	4 hour(s)	512 MB	60 GB	26.9 cm	1.41 kg
○	-	$1'595.00	1 GHz	5.5 hour(s)	512 MB	40 GB	26.9 cm	1.41 kg

they have higher processor speed and bigger hard drive capacity, but are heavier

	Manufacturer	Price	Processor speed	Battery life	Installed memory	Hard drive capacity	Display size	Weight
○	-	$1'220.49	1.8 GHz	5 hour(s)	1 GB	100 GB	38.1 cm	2.95 kg
○	-	$2'148.99	2 GHz	4 hour(s)	1 GB	100 GB	39.1 cm	2.9 kg
○	-	$1'379.00	3.3 GHz	2 hour(s)	512 MB	100 GB	43.2 cm	4.31 kg
○	-	$2'235.00	1.8 GHz	2.5 hour(s)	1 GB	100 GB	43.2 cm	3.99 kg
○	-	$2'319.00	1.7 GHz	4.5 hour(s)	512 MB	100 GB	43.2 cm	3.13 kg
○	-	$2'075.00	1.8 GHz	1.67 hour(s)	512 MB	100 GB	43.2 cm	4.4 kg

they are lighter and have longer battery life, but smaller display size

	Manufacturer	Price	Processor speed	Battery life	Installed memory	Hard drive capacity	Display size	Weight
○	-	$1'529.00	1.7 GHz	6.5 hour(s)	512 MB	80 GB	33.8 cm	1.77 kg
○	-	$1'599.00	1.7 GHz	6.5 hour(s)	512 MB	80 GB	33.8 cm	1.91 kg
○	-	$1'125.00	1.5 GHz	6 hour(s)	512 MB	80 GB	30.7 cm	2 kg
○	-	$2'099.99	1.2 GHz	9 hour(s)	512 MB	60 GB	26.9 cm	1.41 kg
○	-	$1'649.00	1.1 GHz	8.5 hour(s)	512 MB	40 GB	26.9 cm	1.36 kg
○	-	$969.00	1.2 GHz	6 hour(s)	256 MB	39 GB	30.7 cm	2.22 kg

they are cheaper, but heavier

	Manufacturer	Price	Processor speed	Battery life	Installed memory	Hard drive capacity	Display size	Weight
○	-	$1'179.00	3.2 GHz	2 hour(s)	512 MB	80 GB	39.1 cm	3.62 kg
○	-	$1'425.00	1.6 GHz	5.5 hour(s)	512 MB	80 GB	39.1 cm	2.86 kg
○	-	$1'190.00	3.2 GHz	1 hour(s)	512 MB	80 GB	39.1 cm	3.72 kg
○	-	$1'629.00	1.8 GHz	5.8 hour(s)	512 MB	60 GB	38.1 cm	2.81 kg
○	-	$627.10	1.6 GHz	1.5 hour(s)	256 MB	40 GB	38.1 cm	2.81 kg
○	-	$520.00	1.13 GHz	3.5 hour(s)	128 MB	30 GB	35.8 cm	2.59 kg

Fig. 16.4: The more trust-inspiring organization interface.

- *Confidence threshold* θ*:* the amount of perceived accuracy that is required for the user to be satisfied with the current result of the search process and buy the product;
- *Effort threshold* ε: the amount of effort the user is willing to spend to obtain a recommendation;
- *Effort increment threshold* δ: the amount of additional perceived accuracy that is required to motivate the user to spend an additional interaction cycle.

A poorly designed tool can lose users due to an insufficient level of confidence for the confidence threshold, or an increase in confidence that is too small to justify the interaction effort. Figure 16.5 shows the perceived accuracy achieved by a hypothetical recommender tool as a function of effort, measured in interaction cycles. The confidence threshold is indicated by a horizontal dashed line, and the user will not buy the product unless the perceived accuracy exceeds it. The effort threshold ε and effort increment threshold δ characterize the amount of effort a user is willing to spend. It is characterized by another dashed line labelled as the *effort limit* in Fig-

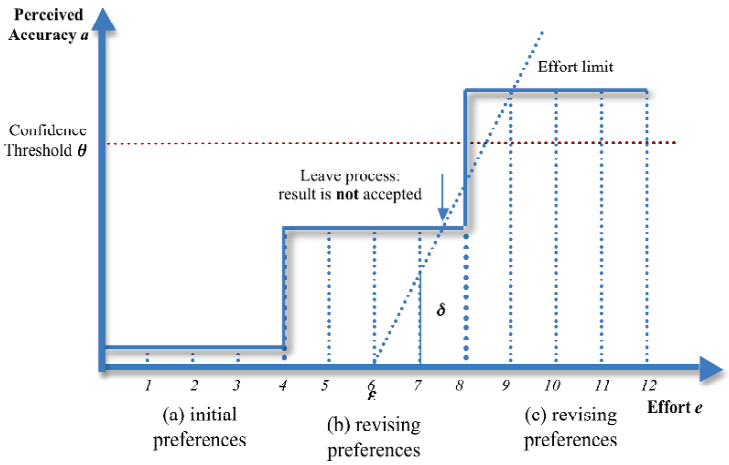


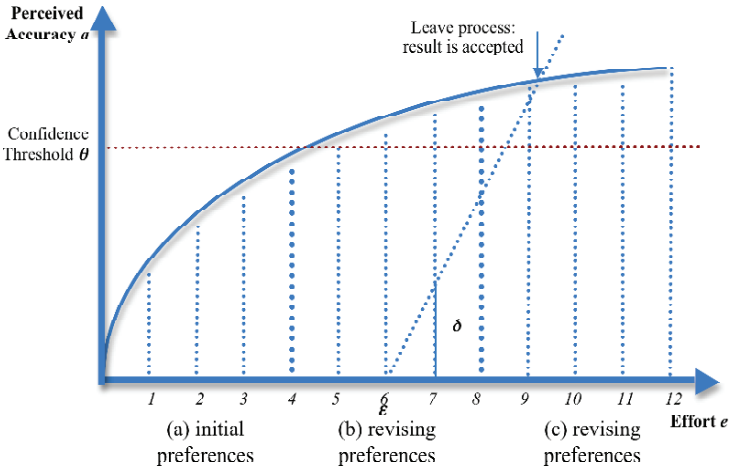
Fig. 16.5: Perceived accuracy as a function of interaction effort for an interview-based tool.

ure 16.5. The user will leave the process as soon as the accuracy/effort curve falls below the effort limit.

Figure 16.5 assumes a tool where users are asked to specify their preferences in a dialogue interface consisting of a list of questions. Such an interface requires a significant amount of effort of preference elicitation before any result is shown, and thus the perceived accuracy remains low until the first results are obtained. The process then proceeds by stages of revising preferences and obtaining successively better perceived accuracy in a stepwise manner. Note that in this example, the user will leave the process before reaching the confidence threshold and the interaction has thus not been a success. With slightly more patience, the user could have obtained a result above the confidence threshold and with acceptable effort, namely at the third intersection of the effort limit line with the curve. However, since this was not apparent to the user, this point will not be reached. Similar problems can occur with other interface designs that do not pay attention to ensuring a significant increase in perceived accuracy per effort invested.

To avoid this pitfall, the tool should ensure that the perceived accuracy is a concave function of effort, as shown in Figure 16.6. Such a function exploits a user's effort threshold to the best possible degree: if the increase in perceived accuracy becomes insufficient to keep the user interested in using the tool, she will not interact with the tool at a later stage either.

A concave function could only be achieved by ensuring instant feedback to the user's effort, and by placing the steps that result in the greatest increase in perceived accuracy at the beginning of the interaction. An early increase in perceived accuracy also serves to convince the user to stay with the system longer, as indicated by the dashed line in Figure 16.6. Instant feedback is ensured by example-based



Guidelines for (a)	Guidelines for (b)	Guidelines for (c)
1. Any effort; 2. Any order; 3. Any preferences.	4. Showing example options; 5. Showing diverse examples; 6. Suggesting options with look-ahead principle.	7. Preference conflict management; 8. Tradeoff assistance.
Guidelines for (a) – (c) 9. Showing one search result at a time is good for small-display devices, but it is likely to achieve relatively low decision accuracy; 10. Displaying more products and ranking them in a natural order is likely to increase users' sense of control and confidence; 11. Designing interfaces which are capable of explaining how ranking scores are computed can inspire user trust.		

Fig. 16.6: Perceived accuracy as a function of interaction effort for an example-based tool, and guidelines that apply to achieving the desired concave shape in the different stages.

interaction and the general guidelines 9-11 of showing multiple solutions in a structured and confidence-inspiring way. In general, it can be assumed that users will themselves choose to add the information that they believe to maximize their decision accuracy, and so user initiative is key to achieving a concave curve. In the first phase (a), the system can achieve the biggest accuracy gains by exploiting users' initial preferences. However, it is important at this stage to avoid asking them questions that they cannot accurately answer (guideline 1). Furthermore, the curve can be made steeper by letting the user formulate these initial preferences with as little effort as possible. We therefore derived guidelines (2 and 3) to make this possible.

Once these initial preferences have been obtained, the biggest increase in perceived accuracy during phase (b) can be obtained by completing the initial prefer-

ences with others of which the user was not initially aware. This can be stimulated by showing examples (guideline 4), and by choosing them to specifically educate the user about available options (guidelines 5 and 6). This provides the main cost-effect tradeoff for the second phase of a typical interaction. Finally, in the third phase (c) the set of preferences can be fine-tuned by adjusting their relative weights and making tradeoffs. This can be supported by tools that show partial solutions (guideline 7) and actively support decision tradeoffs among preferences (guideline 8).

As the tool cannot verify when the user transitions between the phases, and in fact the transition may be gradual, it should provide continuous support for each of them, but always encourage actions that are likely to increase perceived accuracy as much as possible. Thus, adjustment of tradeoff weights should be shown less prominently than the possibility to add new preferences.

These requirements are best addressed by the example-based recommender tools as described in Section 5. More precisely, the incremental establishment and refinement of the user's preference model increases the true decision accuracy. To keep the user engaged in the interaction process and convinced to accept the result of the search, this true accuracy must also be perceived by the user. This is supported by showing several results at the same time (guideline 9), which tends to correct inaccuracies, by providing structure to their display (guideline 10), and by providing explanations (guideline 11). These are necessary elements to motivate users to put in enough effort in achieving an accurate decision as much as possible.

16.9 Conclusion

This chapter presents eleven essential guidelines that should be observed when designing interactive preference-based recommender systems. In presenting and justifying the guidelines, we provided a broad and in-depth review of our prior work related to *example critiquing* regarding user interaction issues in recommender systems. Most importantly, a framework of three evaluation criteria was proposed to determine the usability of such systems: decision accuracy, user confidence, and user interaction effort (ACE). Within this framework, we have selected techniques, which have been validated through empirical studies, to demonstrate how to implement the guidelines. Emphasis was given to those techniques that achieve a good balance on all of the criteria. Adopting these guidelines, therefore, should significantly enhance the usability of product recommender systems and consequently the wide adoption of such systems in e-commerce environments.

References

1. Adomavicius, G., Tuzhilin, A., Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data*

Engineering 17 (6) (2005) 734-749.
2. Agrawal, R., Imielinski, T., Swami, A., Mining association rules between sets of items in large databases. *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, ACM Press, 1993, 207216.
3. Belkin, N.J., Croft, W.B., Information filtering and information retrieval: two sides of the same coin? *Communications of the ACM 35 (12)* (1992) 29-38.
4. Bernoulli, D., Exposition of a new theory on the measurement of risk (original 1738). *Econometrica 22 (1)* (1954) 23-36.
5. Blythe, J., Visual exploration and incremental utility elicitation. *Proceedings of the 18th National Conference on Artificial Intelligence*, AAAI press, 2002, 526-532.
6. Bradley, K., Smyth, B., Improving recommendation diversity. *Proceedings of the 12th Irish Conference on Artificial Intelligence and Cognitive Science*, 2001, 85-94.
7. Burke, R., Hybrid recommender systems: survey and experiments. *User Modeling and User-Adapted Interaction 12 (4)* (2002) 331-370.
8. Burke, R., Hammond, K., Cooper, E., Knowledge-based navigation of complex information spaces. *Proceedings of the 13th National Conference on Artificial Intelligence, AAAI press*, 1996, 462-468.
9. Burke, R., Hammond, K., Young, B., The FindMe approach to assisted browsing. *IEEE Expert: Intelligent Systems and Their Applications 12 (4)* (1997) 32-40.
10. Chen, L., Pu, P., Trust building in recommender agents. *Proceedings of the Workshop on Web Personalization, Recommender Systems and Intelligent User Interfaces at the 2nd International Conference on E-Business and Telecommunication Networks*, 2005, 135-145.
11. Chen, L., Pu, P., Evaluating critiquing-based recommender agents. *Proceedings of Twenty-first National Conference on Artificial Intelligence (AAAI-06)*, 2006, 157-162.
12. Cover, T.M., Hart, P.E., Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, IT-13, 1967, 21-27.
13. Faltings, B., Pu, P., Torrens, M., Viappiani, P., Designing example-critiquing interaction. *Proceedings of the 9th International Conference on Intelligent User Interfaces (IUI'04)*, ACM Press, 2004, 22-29.
14. Faltings, B., Torrens, M., Pu, P., Solution generation with qualitative models of preferences. *Computational Intelligence 20 (2)* (2004), 246-263.
15. Freuder, E.C., Wallace, R.J., Partial constraint satisfaction. *Artificial Intelligence 58 (1-3)* (1992) 21-70.
16. Goker, M., Thompson, C., The adaptive place advisor: a conversational recommendation system. *Proceedings of the 8th German Workshop on Case Based Reasoning*, 2000.
17. Goldberg, D., Nichols, D., Oki, B.M., Terry, D., Using collaborative filtering to weave an information tapestry. *Communications of the ACM (35) 12*, Special issue on information filtering (1992) 61-70.
18. Good, N., Schafer, J.B., Konstan, J.K., Borchers, A., Sarwar, B.M., Herlocker, J.L., Riedl, J., Combining collaborative filtering with personal agents for better recommendations. *Proceedings of the 16th National Conference on Artificial Intelligence (AAAI'99)*, AAAI press, 1999, 439-446.
19. Ha, V.A., Haddawy, P., Problem-focused incremental elicitation of multi-attribute utility models. In Shenoy, P. (ed.), *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence (UAI'97)*, 1997, 215-222.
20. Haubl, G., Trifts, V., Consumer decision making in online shopping environments: the effects of interactive decision aids. *Marketing Science 19 (1)* (2000) 4-21.
21. Herstein, I.N., Milnor, J., An axiomatic approach to measurable utility. *Econometrica 21 (2)* (1953) 291-297.
22. Hill, W., Stead, L., Rosenstein, M., Furnas, G., Recommending and evaluating choices in a virtual community of use. *Proceedings of the CHI '95 Conference on Human Factors in Computing Systems*, 1995, 194-201.
23. Keeney, R.L., *Value-Focused Thinking: A Path to Creative Decision Making*, Harvard University Press (1992).

24. Keeney, R.L., Raiffa, H., *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*, New York: Wiley (1976).
25. L, J., Kolodner. *Case-Based Reasoning*. San Mateo, CA: Morgan Kaufmann (1993).
26. Krulwich, B., Lifestyle finder: intelligent user profiling using large-scale demographic data. *Artificial Intelligence Magazine 18 (2)* (1997) 37-45.
27. Lang, K., Newsweeder: learning to filter news. *Proceedings of the 12th International Conference on Machine Learning*, 1995, 331-339.
28. Linden, G., Hanks, S., Lesh, N., Interactive assessment of user preference models: the automated travel assistant. *Proceedings of the 6th International Conference on User Modeling (UM'97)*, New York: Springer Wien New York, 1997, 67-78.
29. Marschak, J., Rational Behavior, Uncertain Prospects, and Measurable Utility. *Econometrica 18 (2)* (1950) 111-141.
30. McCarthy K., McGinty L., Smyth, B., Reilly, J., A live-user evaluation of incremental dynamic critiquing. *Proceedings of the 6th International Conference on Case-Based Reasoning (ICCBR'05)*, 2005, 339-352.
31. McCarthy K., Reilly, J., L. McGinty, Smyth, B., Thinking positively explanatory feedback for conversational recommender systems. *Proceedings of the Workshop on Explanation in CBR at the 7th European Conference on Case-Based Reasoning (ECCB'04)*, 2004, 115-124.
32. McCarthy K., Reilly, J., L. McGinty, Smyth, B., Experiments in dynamic critiquing. *Proceedings of the 10th International Conference on Intelligent User Interfaces (IUI'05)*, New York: ACM Press, 2005, 175-182.
33. McGinty L., Smyth, B., On the role of diversity in conversational recommender systems. *Proceedings of the 5th International Conference on Case-Based Reasoning (ICCBR'03)*, 2003, 276-290.
34. McNee S.M., Lam, S.K., Konstan, J., Riedl, J., Interfaces for eliciting new user preferences in recommender systems. *Proceedings of User Modeling Conference*, Springer, 2003, 178–187.
35. McNee S.M., Riedl, J., Konstan, J., Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems (CHI '06)*, ACM, New York, NY, 1097-1101.
36. McSherry, D., Diversity-conscious retrieval. In, Craw, S., Preece, A. (eds.), *Proceedings of the 6th European Conference on Advances in Case-Based Reasoning*, London: Springer-Verlag, 2002, 219-233.
37. McSherry, D., Similarity and compromise. *Proceedings of the 5th International Conference on Case-Based Reasoning (ICCBR'03)*, Springer-Verlag, 2003, 291-305.
38. McSherry, D., Explanation in recommender systems. *Workshop Proceedings of the 7th European Conference on Case-Based Reasoning (ECCBR'04)*, 2004, 125-134.
39. Mongin, P., *Expected Utility Theory*. Handbook of Economic Methodology, Edward Elgar, 1998, 342-350.
40. Payne, J.W., Bettman, J.R., Johnson, E.J., *The Adaptive Decision Maker*, Cambridge University Press (1993).
41. Payne, J.W., Bettman, J.R., Schkade, D.A., Measuring constructed preferences: towards a building code. *Journal of Risk and Uncertainty* 19 (1999) 243-270.
42. Price, B., Messinger, P.R., Optimal recommendation sets: covering uncertainty over user preferences. *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI'05)*, 2005, 541-548.
43. Pu, P., Faltings, B., Enriching buyers' experiences: the SmartClient approach. *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI'00)*, New York: ACM Press, 2000, 289-296.
44. Pu, P., Faltings, B., Torrens, M., User-involved preference elicitation. *Working Notes of the Workshop on Configuration. Eighteenth International Joint Conference on Artificial Intelligence (IJCAI'03)*, 2003, 56-63.
45. Pu, P., Faltings, B., Decision tradeoff using example-critiquing and constraint programming. *Constraints: an International Journal 9 (4)* (2004) 289-310.

46. Pu, P., Faltings, B., Torrens, M., Effective interaction principles for online product search environments. *Proceedings of the IEEE/WIC/ACM International Joint Conference on Intelligent Agent Technology and Web Intelligence*, 2004, 724-727.
47. Pu, P., Kumar, P., Evaluating example-based search tools. *Proceedings of the 5th ACM Conference on Electronic Commerce (EC'04)*, ACM Press, 2004, 208-217.
48. Pu, P., Chen, L., Integrating tradeoff support in product search tools for e-commerce sites. *Proceeding of the 6th ACM Conference on Electronic Commerce (EC'05)*, ACM Press, 2005, 269-278.
49. Pu, P., Chen, L., Trust building with explanation interfaces. *Proceedings of the 11th International Conference on Intelligent User Interface (IUI'06)*, 2006, 93-100.
50. Pu, P., Viappiani, P., Faltings, B., Stimulating decision accuracy using suggestions. *SIGCHI conference on Human factors in computing systems (CHI'06)*, 2006, 121-130.
51. Pu P., Chen L. and Kumar P., Evaluating Product Search and Recommender Systems for E-Commerce Environments. *Electronic Commerce Research Journal* 8(1-2), June 2008, 1-27.
52. Pu P., Chen L., User-Involved Preference Elicitation for Product Search and Recommender Systems. *AI Magazine* 29(4), Winter 2008, 93-103.
53. Reilly, J., K. McCarthy, L. McGinty, Smyth, B., Dynamic critiquing. *Proceedings of the 7th European Conference on Case-Based Reasoning (ECCBR'04)*, Springer, 2004, 763-777.
54. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J., GroupLens: an open architecture for collaborative filtering of Netnews. *CSCW '94: Conference on Computer Supported Cooperative Work*, ACM, 1994, 175-186.
55. Rich, E., User modeling via stereotypes. *Cognitive Science 3* (1979) 329-354.
56. Rokach, L., Maimon, O., Averbuch, M., Information Retrieval System for Medical Narrative Reports, Lecture Notes in Artificial intelligence 3055, page 217-228 Springer-Verlag, 2004.
57. Schafer, J.B., Konstan, J.A., Riedl, J., Recommender systems in e-commerce. *Proceedings of the ACM Conference on Electronic Commerce*, ACM, 1999, 158-166.
58. Shardanand, U., Maes, P., Social information filtering: algorithms for automating "Word of Mouth". *Proceedings of the Conference on Human Factors in Computing Systems (CHI '95)*, 1995, 210-217.
59. Schoemaker, P., The Expected Utility Model: Its Variants, Purposes, Evidence and Limitations. *Journal of Economic Literature 20 (2)* (1982), 529-563.
60. Shimazu, H., ExpertClerk: navigating shoppers' buying process with the combination of asking and proposing. *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI'01)*, 2001, 1443-1448
61. Smyth, B., P. McClave. Similarity vs. diversity. *Proceedings of the 4th International Conference on Case-Based Reasoning (ICCBR'01)*, Springer-Verlag, 2001, 347-361.
62. Spiekermann, S., Parachiv, C., Motivating human-agent interaction: transferring insights from behavioral marketing to interface design. *Journal of Electronic Commerce Research 2 (3)* (2002) 255-285.
63. Stolze, M., Soft navigation in electronic product catalogs. *International Journal on Digital Libraries 3 (1)* (2000) 60-66.
64. Torrens, M., Faltings, B., Pu, P., SmartClients: constraint satisfaction as a paradigm for scaleable intelligent information systems. *International Journal of Constraints 7 (1)* (2002) 49-69.
65. Tversky, A., *Contrasting rational and psychological principles in choice. Wise Choices: Decisions, Games, and Negotiations*, Boston, MA: Harvard Business School Press (1996) 5-21.
66. Tversky, A., Simonson, I., Context-dependent preferences. *Management Science 39 (10)* (1993) 1179-1189.
67. Viappiani, P., Faltings, B., V. Schickel-Zuber, Pu, P., Stimulating preference expression using suggestions. Mixed-Initiative Problem-Solving Assistants, *AAAI Fall Symposium Series*, AAAI press, 2005, 128-133.
68. Viappiani, P., Faltings, B., Pu, P., Evaluating preference-based search tools: a tale of two approaches. *Proceedings of the Twenty-first National Conference on Artificial Intelligence (AAAI-06)*, 2006, 205-210.

69. Viappiani, P., B. Faltings and Pu, P., Preference-based Search using Example-Critiquing with Suggestions. *Journal of Artificial Intelligence Research (JAIR)*, 27, 2006, 465-503.
70. Paolo Viappiani, Pearl Pu, and Boi Faltings. Preference-based Search with Adaptive Recommendations. *AI Communications* 21(2-3), 2008, 155-175.
71. J. von Neumann, Morgenstern, O., *The Theory of Games and Economic Behavior*, Princeton University Press (1944).
72. Williams, M.D., Tou, F.N., RABBIT: an interface for database access. *Proceedings of the ACM '82 Conference*, ACM Press, 1982, 83-87.
73. Zhang, J., Pu, P., Performance evaluation of consumer decision support systems. *International Journal of E-Business Research 2 (2006)* Idea Group Publishing.
74. Ziegler, C.N., S.M. McNee, Konstan, J.A., Lausen, G., Improving recommendation lists through topic diversification. *Proceedings of the 14th International World Wide Web Conference (WWW'05)*, 2005, 22-32.
75. Zukerman, I., Albrecht, D.W., Predictive statistical models for user modeling. *User Modeling and User-Adapted Interaction 11 (1-2)* (2001) 5-18.