Dissertation

Interactive recommendation applications for configurable products and services

Monika Mandl

Graz, 2012

Institute for Software Technology
Graz University of Technology

Supervisor/First reviewer: Univ.-Prof. Dipl.-Ing. Dr. techn. Alexander Felfernig
Second reviewer: Univ.-Prof. Dipl.-Ing. Dr. Martin Hitz

Abstract (English)

Nowadays, purchase decisions are increasingly made in online selling environments. Digital marketplaces provide a large amount of products and product information, but in contrast to customers of bricks and mortar stores, users of online selling applications are not supported by human sales experts. Therefore, many online retailers offer intelligent shopping support by integrating recommendation and configuration technologies in the online shop. These technologies help to improve the shopping experience by assisting the customer in the specification of individual requirements, as well as by providing a personalized product space. This thesis focuses on evaluating and improving different factors of intelligent online selling environments, that have an effect on users' satisfaction with the shopping process. Specifically, this work deals with three major topics: the *quality of recommendation algorithms*, the *quality of customers' subjective experiences*, and *the impact of psychological aspects of recommendations on human decision making.*

In this thesis, new algorithms are introduced, that improve the *prediction quality* of unit critiquing recommender systems. Results of an empirical experiment show that the new critiquing approaches have the potential to reduce the number of critiquing cycles in critiquing sessions. In order to improve the *subjective quality* of product configuration processes, this thesis introduces an approach to assist the customer in specifying his/her requirements, by combining knowledge-based configuration with collaborative recommendation algorithms. In this context, the RecoMobile configurator is introduced, a personalized configuration system for mobile phones and subscriptions, that integrates recommendation technologies to support the customer's preference construction process. An empirical evaluation of the RecoMobile product configurator points out important improvements in terms of, for example, user satisfaction and perceived configuration process quality.

To evaluate consumers' *subjective experiences* within an intelligent selling environment and the *impact of recommendation technologies on consumer decision making*, different user interface types for the RecoMobile configurator were developed. The results of an empirical study indicate that different presentation styles of personalized recommendations can have a significant impact on the perceived quality of the selection process, as well as on a users' trust in the product configurator.

Finally, this thesis points out potential dangers of recommendation technologies. An empirical study revealed that recommendations can lead to biases in consumer decision making. As a consequence of these results, we have to increasingly turn our attention to ethical aspects when implementing intelligent selling applications, since such decision biases can be exploited to mislead users and to increase sales of, for example, expensive products.

Abstract (German)

Heutzutage werden Kaufentscheidungen zunehmend in Online-Shops getroffen. Digitale Marktplätze bieten eine große Menge von Produkten und Produktinformationen an, aber im Gegensatz zu Kunden in konventionellen Geschäften, werden Kunden in Online-Shops nicht durch menschliche Experten unterstützt. Daher bieten viele Online-Händler durch die Integration von Empfehlungs- und Konfigurationstechnologien eine intelligente Kaufunterstützung an, um das Online-Kauferlebnis zu verbessern. Diese Technologien helfen dabei, einerseits Kunden in der Spezifikation ihrer individuellen Anforderungen zu unterstützen, und andererseits einen personalisierten Produktbereich zur Verfügung zu stellen. Der Fokus dieser Arbeit liegt in der Evaluierung und Verbesserung verschiedener Faktoren von intelligenten Online-Verkaufsumgebungen, die einen Einfluss auf die Kundenzufriedenheit mit dem Kaufprozess haben. Konkret befasst sich diese Arbeit mit drei Hauptthemen: der *Qualität von Empfehlungsalgorithmen*, der *Qualität der subjektiven Erfahrungen von Kunden*, und *den Auswirkungen von psychologischen Aspekten von Empfehlungen auf die Entscheidungsfindung*.

In dieser Arbeit werden neue Algorithmen vorgestellt, um die *Vohersagequalität* in Critiquing-Empfehlungssystemen zu verbessern. Die Resultate einer empirischen Studie zeigen, dass die neuen Ansätze die Anzahl der Critiquing-Zyklen in einer Critiquing-Session reduzieren können. Zur Verbesserung der *subjektiven Qualität* von Produkt-Konfigurationsprozessen wird ein Ansatz vorgestellt, der den Kunden bei der Spezifikation seiner Anforderungen unterstützt, indem wissensbasierte Konfiguration mit kollaborativen Empfehlungsalgorithmen kombiniert wird. RecoMobile, ein personalisiertes Konfigurationssystem für Mobiltelefone und Vertragsoptionen, integriert Empfehlungstechnologien zur Unterstützung der Bestimmung von Kundenanforderungen. Eine empirische Evaluierung von RecoMobile zeigt Verbesserungen z. B. bei der Benutzerzufriedenheit und bei der wahrgenommenen Konfigurationsprozessqualität auf.

Um die *subjektiven Erfahrungen* von Kunden in einer intelligenten Verkaufsumgebung und den *Einfluss von Empfehlungstechnologien auf den Entscheidungsprozess* zu evaluieren, wurden verschiedene Typen der Benutzeroberfläche für RecoMobile entwickelt. Die Ergebnisse einer empirischen Studie zeigen, dass unterschiedliche Anzeigearten von Empfehlungen einen erheblichen Einfluss auf die Zufriedenheit der Kunden mit der Qualität des Auswahlverfahrens, sowie auf das Vertrauen der Kunden in den Konfigurator haben können.

Letztendlich werden potentielle Gefahren von Empfehlungstechnologien aufgezeigt. Eine empirische Studie hat gezeigt, dass Empfehlungen zu einer Verzerrungswirkung der Kaufentscheidung führen können. Daher müssen wir unsere Aufmerksamkeit auf ethische Aspekte bei der Entwicklung von intelligenten Verkaufsumgebungen richten, da solche Verzerrungswirkungen ausgenutzt werden können, um Kunden zu täuschen, und den Umsatz von z. B. teuren Produkten zu erhöhen.

Acknowledgement

Thanks for the past, the present, and the future!

Monika Mandl
Graz, 2012

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Graz, ____________________________   ____________________________

Place, Date   Signature

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Graz, am ____________________________   ____________________________

Ort, Datum   Unterschrift

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The information-rich nature of the Internet makes it easy for companies to implement digital marketplaces, providing a large product assortment and a significant amount of product information. The rapid growth of e-commerce, and the ever-increasing amount of product choices have made it a difficult task to identify products that meet one's requirements. For example, at Amazon.com the search for a book of the category *fantasy* results in more than 70.000 items.

Research in the field of consumer behavior has shown that consumers are frequently overwhelmed by the amount of possible choice options – a phenomenon known as *information overload* (Jacoby et al., 1974; Häubl and Murray, 2003; Lee and Lee, 2004). In an analysis of hypertext systems published by Jeffrey Conklin in 1987 (Conklin, 1987), Conklin mentioned two problems that arise when information spaces grow: *disorientation* and *cognitive overhead*. He referred to cognitive overhead as "*the additional effort and concentration necessary to maintain several tasks or trails at one time*" (Conklin, 1987) and he defined disorientation as "*the tendency of users to lose their way in non-linear information*". Huffman and Kahn (Huffman and Kahn, 1998) state that users of online shops in many cases have the problem of not understanding the set of offered options in detail, and are often overwhelmed by the complexity of those options – a phenomenon well known as *mass confusion* (Huffman and Kahn, 1998).

In order to counteract these problems and to support customers in their product search, many online retailers integrate intelligent customer support into their websites (Sabin and Weigel, 1998; Schafer et al., 2001). *Intelligent selling systems* are applied to assist customers in specifying their requirements, as well as finding a product that matches their individual preferences. In the context of this thesis, the focus is set to two specific types of intelligent selling systems: *recommender systems* (see, for example, (Burke et al., 1999; Terveen and Hill, 2001; Adomavicius and Tuzhilin, 2005; Chen and Pu, 2012)) and *product configurations systems* (see, for example, (Sabin and Weigel, 1998; Blecker et al., 2004)).

Recommender systems personalize the shopping experience by providing predictions on products that may be of interest for the customer (Terveen and Hill, 2001). Formally, a recommendation problem can be formulated as follows:

Definition 1 (recommendation problem): let C be the set of all customers, and let I be the set of possible items to recommend. The recommendation problem is to identify a set of items $I(c) \in I$ for each customer $c \in C$ that satisfies the individual requirements.

Since it is impossible for customers to process the amount of information offered by online shops, recommender systems are an important instrument to guide the customer through a *personalized product space*.

Recommendation technologies can further be exploited to assist the consumer in the *preference construction process*. Consumers are rarely able to provide complete and accurate preferences at the beginning of their product search (Reilly et al., 2007b), but construct their preferences within the decision making process (Bettman et al., 1998). *Conversational recommender systems* support this constructive preference elicitation process by involving the customer in a dialog, where the user is enabled to refine his/her requirements (Burke et al., 1997). Critiquing-based recommender systems (McCarthy et al., 2005; Viappiani et al., 2007) apply this user interaction concept by calculating a set of recommendations in response to an initial (partial) user query. If the user is not satisfied with the results, it is possible to critique attributes of the example products. For example, the user might indicate that he/she is looking for a *cheaper* product than the actual recommendation. This interaction model allows users to learn and adapt their individual preferences according to concrete examples (Faltings et al., 2004).

The approach to exploit recommendation technologies to assist customers in specifying their requirements has become an important research area in the field of *product configuration systems* in the last years (see, for example, (Geneste and Ruet, 2001; Wicaksono et al., 2012)). Product configurators are an important tool to implement the concept of mass customization, that aims at producing individually customized products with near mass production efficiency (Pine, 1993; Tseng and Jiao, 2001). Product configuration systems support the idea of integrating customers in the design and production process by enabling them to configure complex products according to their individual requirements (Blecker et al., 2004; Yang et al., 2005). Instead of *choosing* a product, a user of a product configurator *specifies* a product by configuring the product attributes based on a set of available options. Therefore, contrary to recommender systems, where the recommendation calculation is based on an existing set of available items, configuration systems are used to *create* items from a set of well-defined components (Sabin and Weigel, 1998). A configuration problem can be formulated as follows:

Definition 2 (configuration problem): let D be the set of possible product attributes, and let I be the corresponding instantiations. C represents a set of constraints that restrict the set of possible solutions (for example, technical restrictions, or rules concerning the production process), and R defines the set of customer requirements. The configuration problem is to identify an instantiation $I(r_d)$ for each attribute $d \in D$ that satisfies the corresponding customer requirement $r_d \in R$, and that is consistent with the configuration constraints $c_d \in C$.

A major problem of product configuration systems is that the high diversity of options may outstrip a user's capability to fully explore them and make a buying decision (Streufert and Driver, 1965; Jacoby et al., 1974; Bettman, 1979; Malhotra, 1982). Research has shown that a personalization of product configuration processes through the *integration of recommendation* *technologies* can support customers in the specification of their requirements and thus can lead to a higher customer satisfaction with the configuration system (Geneste and Ruet, 2001; Coster et al., 2002; Felfernig et al., 2010b).

Intelligent selling applications such as recommender systems and product configuration systems, that aim to support customers in their online product search, have become important research areas over the last decade (Adomavicius and Tuzhilin, 2005; Heiskala et al., 2007; Wicaksono et al., 2012). The following sections motivate the importance of different factors of such intelligent selling applications, including the prediction accuracy of recommendation algorithms, the subjective quality of the application, and the influence of recommendations on human decision making. Furthermore, the research objectives and the contributions of this work are pointed out. An outline of the thesis closes this chapter.

1.1. Motivation

This thesis focuses on algorithms and techniques that support customers in their decision making process in intelligent online selling environments, where recommendation and configuration technologies are integrated in order to support the customer in constructing their preferences and finding their ideal items. In this context, this thesis covers the following challenges:

(1) improving the prediction quality of recommendation algorithms,
(2) improving users' subjective attitudes toward the acceptance of recommendation and configuration technologies, and
(3) analyzing the impact of recommendations on the consumer decision making process.

(1) In recommender applications, prediction algorithms are applied to find items that may be of interest for the user. Prediction can be defined as "*a value that expresses the predicted likelihood that a user will 'like' an item*" (Papagelis and Plexousakis, 2005). The items with the top n predictions are presented as recommendations to the user. Therefore improved prediction quality can lead to better recommendations for the user. Challenges like the *Netflix price*[1] highlight the importance of the development and improvement of recommendation algorithms. Different techniques have been developed for calculating accurate and efficient recommendations. The earliest recommender systems applied *collaborative approaches* (Goldberg et al., 1992) – in such systems the behavior or opinions of previous system users are exploited to find items that may be of interest for the current user. *Content-based algorithms* focus on finding items similar to those the user has liked in the past (Terveen and Hill, 2001). The user preferences are typically expressed by ratings the user has given on items, or by a purchase history. *Knowledge-based recommender systems* exploit explicitly defined requirements of the user, knowledge about the product assortment, and the dependencies between customer preferences and product attributes, to calculate personalized recommendations (Burke, 2000).

One of the main sources that have an influence on the prediction quality of recommender algorithms is the information about the user preferences. Many interactive recommender applications enable users to give feedback on recommendations in order to get a better understanding of their

[1] `http://www.netflixprize.com/`

preferences, and to improve the system's prediction quality (Smyth and McGinty, 2003; Faltings et al., 2004; Chen and Pu, 2007b).

(2) Beside the importance of the prediction quality of recommendation algorithms, various additional factors may have an effect on users' subjective attitudes toward the acceptance of recommender technologies. Examples for such factors are interface design elements and the robustness or trustworthiness of the system. Trust has been identified as a key factor that influences the subjective quality of interactive recommender applications (Jarvenpaa et al., 2000; Roy et al., 2001; Chen and Pu, 2005). Research has shown that the level of trust (confidence) in a recommender application is directly correlated with customer's *intention to purchase a product*, as well as with the *intention to use the system again* (Jarvenpaa et al., 2000; Roy et al., 2001; Chen and Pu, 2005). The research of Chen and Pu (Chen and Pu, 2005) indicates that in addition to security issues and privacy policies, there are system features related to the recommender's *competence*, for example, interface display techniques, or interaction models, that influence the trust building process of customers. The manipulation of user interface design elements, such as color, background, or fonts may also have an influence on consumer's *willingness to buy* (Eroglu, 2001; Eroglu et al., 2003). These findings point out, that it is not only important to implement algorithms that provide good recommendations, but also to establish an adequate format for presenting the recommendations.

(3) Although there exist a couple of theories that explain the existence of different types of decision biases in consumer decision making, few research focused on investigating the effect of recommendations on the decision making process of a user (Mandel and Johnson, 1998; Felfernig et al., 2007; Adomavicius et al., 2011). Since research has shown that consumers construct their preferences within the decision making process (Bettman et al., 1998), such decision biases can have a strong influence on the preference construction process, as well as on the final product choice. Felfernig et al. (Felfernig et al., 2007) investigated *serial position effects* in recommendation scenarios. Their findings indicate that the ordering of product attributes induce significant changes in the product selection behavior. Adomavicius et al. (Adomavicius et al., 2011) found that recommender's predictions can have a strong influence on users' rating behavior, causing an *anchoring effect*. They conducted three laboratory experiments to explore the effect of "system ratings" that indicate a prediction of how much the consumer will like an item. Their results suggest that user preferences can be significantly influenced by ratings provided by recommender applications. The results of the research of Mandel and Johnson (Mandel and Johnson, 1998) indicate that web page design can have an impact on users' perceived importance of product attributes and therefore on the final product choice – this is known as *priming effect*. In a user experiment, where participants had to specify the importance of attributes related to sofas, different web page styles were used to prime the attributes. Participants either saw a sofa advertisement with a blue background and clouds, that was designed to prime the attribute *comfort*, or with a green background with pennies, designed to prime the attribute *price*. Mandel and Johnson (Mandel and Johnson, 1998) demonstrated that different background designs may influence the preference construction process by raising the awareness, and therefore the importance of the primed attributes.

The existence of such biasing effects lead to potential dangers for recommendation technologies. If the user's opinion is biased in the preference construction process, this can lead to imprecise and erroneous requirements information, and as a consequence to less accurate algorithm prediction quality. Another problem is that decision biases can lead to unscrupulous business practices since online retailers could exploit these biases to increase sales for, for example, expensive products. As a consequence of these results, we have to increasingly turn our attention to the investigation of further biasing effects in the context of intelligent selling applications. An important research area in this context is the development of user interface types that either have the potential to counteract these biasing effects, or exploit the effects in a way, that lead to an improvement of the user's online shopping experience.

1.2. Research Objectives

In the previous section we have discussed challenges in the field of intelligent selling applications that are of special relevance in the context of this thesis. These challenges raised the following research questions (see Figure 1.1 for the system's main research and corresponding evaluation issues):

1. **How to improve the performance of unit critiquing?**
   Critiquing systems belong to the type of conversational recommender systems (McCarthy et al., 2004; Zhang and Pu, 2006; Viappiani et al., 2007). Users of such systems provide feedback on recommended items by critiquing product attributes (Burke, 2002b; Li and Pearl, 2006; Reilly et al., 2007a). An example of such a critique is "*I want a less expensive computer than this*". This interaction style enables users to construct their preferences within the recommendation process, and helps the system to adapt the user preference model (Ricci and Nguyen, 2007). *Unit critiquing* offers a very light-weight type of feedback, where users critique one single product attribute in each critiquing cycle. Recent research indicates that exploiting the critiquing experiences of previous users can lead to an improvement of the algorithm efficiency, and therefore to a reduction of the number of critiquing cycles in unit critiquing sessions (McCarthy et al., 2010). In this context we focused on the development of new approaches to collaborative unit critiquing with the aim to further improve the critiquing efficiency. This leads us to our first research question:

   (Q1) *How to improve the algorithm efficiency of unit critiquing and therefore reduce the number of critiquing cycles in critiquing sessions?*

2. **How to exploit recommendation technologies to improve the quality of product configuration processes?**
   Product configuration systems have been recognized as ideal tools to assist customers in configuring complex products according to their individual preferences. A side-effect of the high diversity of products offered by a configurator is that the complexity of the alternatives may outstrip a user's capability to explore them and make a buying decision (Malhotra, 1982). For example, when using the *Alienware* product configurator[2] to customize a computer, technical knowledge concerning computer components, such as processor, memory,

[2] `http://www.alienware.com`

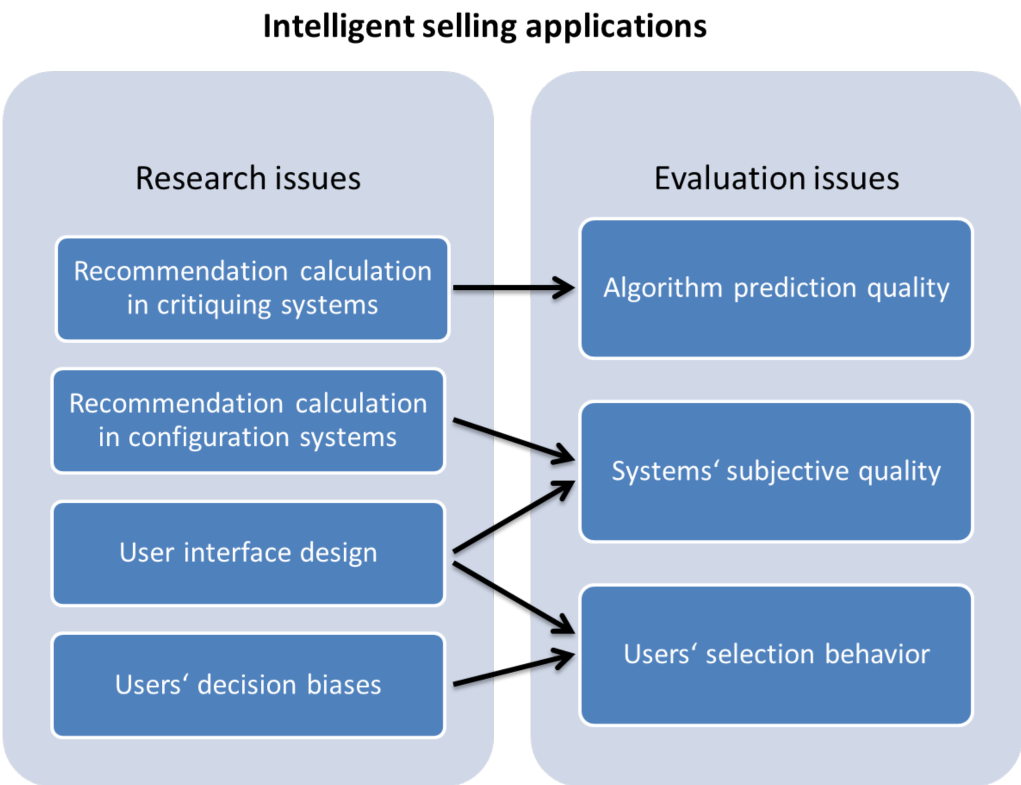

Figure 1.1.: The main research issues of the thesis and the corresponding evaluation issues.

and hard drive is needed. Research has shown that recommendation technologies can support customers in the specification of their requirements, and thus can help to achieve a higher customer satisfaction (Geneste and Ruet, 2001; Coster et al., 2002; Felfernig et al., 2010b). The integration of recommendation technologies with knowledge-based configuration is still in a very early stage. There exist some contributions that take into account the application of personalization technologies in the configuration context (see, for example, (Coster et al., 2002; Tiihonen and Felfernig, 2010; Falkner et al., 2011; Mandl et al., 2012b)). In this context we are interested in a user-centric evaluation of the subjective quality of recommendation techniques that assist customers in their preference construction process in the context of a configuration dialog. The research question is:

(Q2) *How to improve the subjective quality of configuration dialogs by integrating recommendation strategies in the product configuration process?*

3. **Evaluating the impact of different design alternatives for recommendations.** Research on consumer decision making has shown that consumers are influenced by the format of the information presented, and as a consequence use different decision-making strategies in different contexts (see, for example, (Asch, 1949; Payne, 1976; Bettman and Kakkar, 1977; Bettman et al., 1991)). The format of information presentation as well as elements of the user interface design can have a significant impact on user attitudes and perceptions of the trustworthiness of a system (see, for example, (Kim and Moon, 1998;

Roy et al., 2001; Eroglu, 2001; Eroglu et al., 2003)). In this context we are interested in answering the question if different styles to present personalized feature value recommendations (*defaults*) in product configurators have an impact on users' perceived quality of the configuration process, as well as on the decision making process. Our research addressed two relevant questions:

(Q3.1) *Do different default representation styles have an influence on a user's willingness to accept feature value recommendations?*
(Q3.2) *Do different default representation styles have an impact on users' subjective perception of the configuration system?*

4. **Evaluating the impact of recommendations on users' decision making process.** In the psychological literature there exist a couple of theories that explain the existence of different types of decision biases in consumer decision making (Payne, 1976; Bettman et al., 1991; Mandl et al., 2009, 2011a, 2012c). One example is the *Status Quo Bias* – research in the field of human decision making has revealed that people have a strong tendency to keep the status quo when choosing among alternatives (see, for example, (Samuelson and Zeckhauser, 1988; Kahneman et al., 1991; Ritov and Baron, 1992; Bostrom and Ord, 2006)). The focus of our research is set on answering the question whether a status quo bias exists in the context of product configuration systems where personalized defaults are presented to the user. Furthermore, we want to explore whether it is possible to reduce this biasing effect by providing a user interface where both, keeping and changing the status quo option, needs user interaction. This interface type was inspired by the research of Ritov and Baron, who argue that "*when both keeping and changing the status quo require action, people will be less inclined to err by favoring the status quo when it is worse*" (Ritov and Baron, 1992). Our research addressed two relevant questions:

(Q4.1) *Do users of product configuration systems experience a Status Quo Bias, i.e, is the choice behavior of users influenced by defaults?*
(Q4.2) *Is it possible to counteract the Status Quo Bias by providing an appropriate user interface?*

1.3. Contributions

One major contribution of this thesis is improving the quality of the preference construction process in intelligent selling applications. In this context, the research focus is set on conversational recommendation approaches, such as critiquing systems, as well as on product configuration systems. The algorithm *Nearest Neighbor Compatibility Critiquing* (Mandl and Felfernig, 2012a) has been developed within this work to increase the efficiency of unit critiquing. This algorithm focuses on finding a previous system user with a similar critiquing history compared to the current user (*nearest neighbor*), and recommend that item in nearest neighbor's critiquing history that best matches the current user's requirements. Furthermore, we have combined our new approach with conventional and experience-based critiquing to corresponding *ensemble-based critiquing variations* (Mandl and Felfernig, 2012a).

In order to improve the quality of product configuration processes, we combine knowledge-based configuration with collaborative recommendation algorithms (Felfernig et al., 2010a,b). In this context we have developed RecoMobile, which is a configuration system for mobile phones and subscriptions. Users of RecoMobile are pro-actively supported in their preference construction process by personalized defaults and minimal sets of changes in the case that no solution could be found. A collaborative ranking strategy is applied in order to present the results (products) to the user in a convenient way.

Furthermore, the presented work deals with customers' subjective experiences within an intelligent selling environment (Mandl et al., 2011c), and the impact of recommendations on human decision making (Mandl et al., 2011b). Different user interface types for the RecoMobile configurator were developed and evaluated in empirical studies. The versions differed in the extent of personalization, in the way the recommendations were displayed to the user, and in the extent of user interaction required to accept or reject the suggestions. Our studies aimed at analyzing users' subjective attitudes toward the acceptance of recommendation technologies in the context of a product configuration dialog, and at comparing the selection behavior of users of different configurator versions.

An overview of the research questions and the corresponding contributions of this thesis is given in Table 1.1.

Table 1.1.: Overview of the research questions and the corresponding contributions.

Research Questions	**Contributions**
(Q1) How to improve the algorithm efficiency of unit critiquing and therefore reduce the number of critiquing cycles in critiquing sessions?	In this work we present the *Nearest Neighbor Compatibility Critiquing* approach, an extension of the experience-based unit critiquing algorithm. Furthermore, we introduce ensemble-based critiquing variations, where we combined our new approach with existing unit critiquing strategies. To demonstrate improvements in recommendation efficiency we conducted an offline evaluation based on artificial user data. The results of our experiment indicate that our new nearest neighbor compatibility critiquing approach, as well as ensemble-based variations thereof have the potential to reduce the number of critiquing cycles in critiquing sessions, compared to existing unit critiquing strategies.

Table 1.1.: Overview of the research questions and the corresponding contributions.

Research Questions	**Contributions**
(Q2) How to improve the subjective quality of configuration dialogs by integrating recommendation strategies in the product configuration process?	We conducted an empirical study where the RecoMobile configurator was used to analyze users' subjective attitudes toward the acceptance of recommender technologies in the context of a configuration dialog. In our evaluation we compared a personalized version of the RecoMobile configurator (feature recommendation supported), with a version that does not include feature recommendations. The results of our study indicate, that the integration of recommendation techniques in a product configuration system show to be useful in terms of improving the user acceptance of the configurator interface.
(Q3.1) Do different default representation styles have an influence on a user's willingness to accept feature value recommendations?	We present the results of an empirical online study that analyzed the impact of different presentation styles of personalized recommendations in product configuration scenarios (Mandl et al., 2011c). The results revealed no significant difference concerning the selection behavior between the configurator versions. But we found a strong correlation between the number of selected or accepted recommendations, and users' satisfaction with the presentation of feature recommendations.
(Q3.2) Do different default representation styles have an impact on users' subjective perception of the configuration system?	To evaluate if different methods of presenting recommendations have an impact on perceived quality of the system, a post-study questionnaire was designed covering 8 subjective measures related to satisfaction, trust, confidence, and intentions of behavior (Mandl et al., 2011c). Participants of an online study were asked to fill out the questionnaire after interacting with the RecoMobile configurator. The results show that the method of presenting feature recommendations can have a significant impact on users' satisfaction with the overall perceived quality of the selection process, as well as on users' confidence and trust in the product configurator.

Table 1.1.: Overview of the research questions and the corresponding contributions.

Research Questions	**Contributions**
(Q4.1) Do users of product configuration systems experience a *Status Quo Bias*, i.e, is the choice behavior of users influenced by defaults?	To test the influence of defaults on the selection behavior of users an online survey was performed where different versions of the ReCoMobile configurator were used (Mandl et al., 2011b). These versions differed in the degree of personalization as well as in the degree of required user interaction. The results of our study showed that there exists a strong biasing effect even if uncommon values are presented as default values. Furthermore, we found that the status quo effect can, for example, be exploited to make users of a configuration system selecting more expensive solution alternatives.
(Q4.2) Is it possible to counteract the *Status Quo Bias* by providing an appropriate user interface?	In this context we have developed and evaluated an user interface design that supports an interaction type described by Ritov and Barron (Ritov and Baron, 1992). They suggest counteracting the status quo bias by presenting the options in such a way, that keeping as well as changing the status quo needs user input (Ritov and Baron, 1992). We detected that providing such an user interface type does not counteract the status quo bias.

1.4. Thesis Outline

This thesis consists of six chapters, which are organized as follows:

Chapter 1 introduces the motivation and research objectives of this work. Our research questions regarding recommender technologies and human decision making in intelligent selling environments are discussed. Finally, an overview of the structure of this thesis concludes the chapter.

Chapter 2 gives an introduction to the research field of recommender applications and product configuration systems. Section 2.1 gives an overview of different recommendation strategies, such as collaborative, content-based, and knowledge-based recommendation approaches. In Section 2.2 the concepts of critiquing systems are discussed, and Section 2.3 provides an overview of the concept of product configuration systems and strategies to integrate recommendation technologies in the product configuration process.

Chapter 3 provides an overview on related work in the field of consumer decision making, as well as in the field of evaluation techniques for recommender systems. In Section 3.1 different decision-making strategies are presented. Section 3.2 focuses on selected theories from decision psychology, and their potential impact on preference construction and choice processes in intelligent selling environments. Finally, Section 3.3 gives an overview of existing approaches to the evaluation of recommendation technologies, with respect to the types of questions (concerning algorithm prediction quality, systems' subjective quality, and users' selection behavior – see Figure 1.1) that can be answered by the evaluation approaches.

In **Chapter 4** results of our research are presented that aimed at improving the recommendation quality in the context of intelligent selling applications. The focus of Section 4.1 is set on specific algorithms that improve the algorithm efficiency of unit critiquing. We will discuss our new *Nearest Neighbor Compatibility Critiquing* approach, as well as different *ensemble-based variations.* Section 4.2 presents an approach to improve the subjective quality of configuration processes by combining knowledge-based configuration with collaborative recommendation algorithms. A short summary of the presented approaches concludes this chapter.

In **Chapter 5** we analyze user interface design and human decision making aspects in the context of recommendations scenarios that may affect consumer decision making. In Section 5.1 we present an experiment that aimed at investigating the impact of different *presentation styles* of personalized defaults. Section 5.2 presents a discussion of the results of an empirical study that had the goal to explore the impact of the *Status Quo Bias* in personalized product configuration scenarios. Section 5.3 summarizes our findings in the field of human decision making in intelligent selling environments.

Chapter 6 concludes this thesis. We reflect on our research questions and contributions in Section 6.1, and give an outlook on future research issues in Section 6.2.

Chapter 2

Preliminaries

Parts of the contents of this chapter have been published in (Mandl et al., 2009, 2011a, 2012b).

Intelligent selling systems have become an important instrument in digital marketplaces to support customers in their product search. Product configuration systems and recommender systems provide an effective strategy to personalize the online shopping experience for each customer (Linden et al., 2003; Blecker et al., 2004). In order to assist the customer in his/her product search, implicit or explicit information about the user preferences is collected (Larson and Smithwick, 2010; Jawaheer et al., 2010), and recommendation algorithms are applied to present a personalized product space to the customer. The purpose of this chapter is to introduce the background of such intelligent selling systems, which serves as a foundation of the work described in this thesis.

The remainder of this chapter is organized as follows. Section 2.1 provides an overview of recommendation technologies and approaches to categorize recommender applications. One possibility of categorizing recommender systems is by the used knowledge sources for finding items to recommend. In this context there exist three main categories: *collaborative recommender systems* collect and use knowledge about previous system users, for example, purchase histories or product ratings (see Section 2.1.1), *content-based recommender systems* exploit knowledge about the product assortment and matches that information against user preferences (see Section 2.1.2), and *knowledge-based recommender systems* are based on explicit knowledge about customers, the product assortment, and the dependencies between customer preferences and product attributes (see Section 2.1.3). Another possibility of the categorization of recommender systems is by the user interaction style supported by the application. This categorization focuses on recommender applications that collect explicitly defined user preferences. *Conversational recommender systems* allow users to give feedback on recommended items and therefore support an incremental construction of user preferences, whereas *single-shot recommender systems* make a set of recommendations in response to an initial user request without taking into account previous user history. In Section 2.2 we introduce the concepts of *critiquing systems*, which belong to the group of conversational recommender applications. Section 2.3 deals with *product configuration systems* which support the idea of integrating customers in the product design and production process. In contrast to recommender systems, where the recommendation calculation is based on an existing set of items, the available assortment in configuration systems is not restricted to a predefined set of items – a product configurator supports users in assembling a product instance from a generic product structure that satisfies the individual customer requirements (Sabin and Weigel, 1998).

One major drawback of product configuration systems is, that users are often overwhelmed by the complexity of available options (Huffman and Kahn, 1998). Therefore, Section 2.3 presents an overview of strategies for integrating product configuration systems with recommendation technologies to better support users of such systems.

2.1. Recommender Systems

Recommender systems have become an important instrument in e-commerce to assist customers in their search process by providing a personalized information filtering technique to find information or products that are likely to be of interest to the user. Burke defines recommender systems as "*any system that produces individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options*" (Burke, 2002a). Many successful Internet companies such as Amazon, Google, Pandora, Facebook, or Netflix, have integrated recommendation technologies in their websites to personalize the shopping experience and to help users to find items they might wish to examine or purchase.

Research has shown that when having to choose one option out of large set of alternatives, people often experience *information overload*, which means that they are overwhelmed by the amount of possible choice options (see, for example, (Jacoby et al., 1974; Lee and Lee, 2004)). Xiao and Benbasat (Xiao and Benbasat, 2007) indicate that recommender systems can reduce information overload, as well as the complexity of online searches. Especially for users who are not experts in the product domain, recommender systems can help to guide through the product space (Schafer et al., 2001). Pu and Chen suggest that recommender systems can provide user benefits in terms of more *efficiency* in finding preferential items, more *confidence* in making a purchase decision, and a potential chance to *discover something new* (Pu and Chen, 2010).

For online retailers the advantage lies in the fact that recommender systems can help to implement a *One-to-One Marketing* strategy (Peppers et al., 1996), an approach to personalize marketing by identifying and satisfying the customers' individual needs. The integration of recommendation technologies in an online shop creates a value-added relationship between the site and customers, and therefore can improve the loyalty of customers (Schafer et al., 2001).

In order to cover the diverse application areas, a variety of recommender strategies have been developed. There exist some approaches to categorize recommender applications. For example, Schafer et al. (Schafer et al., 2001) have developed a taxonomy for recommender systems, that classify their attributes into three categories: the *functional I/O*, the *recommendation method*, and *other design issues*, such as the degree of personalization. Another way is to differentiate recommender systems by the extent to which they engage in a preference elicitation dialog with users (Bridge and Kelly, 2005) (user interaction style – see Figure 2.1):

- **Single-shot systems**: The system delivers a set of recommendations in response to an initial user request. An example of this approach are web search engines (Ma et al., 2007).

In such systems each user interaction is typically treated independently without taking into account the previous user history (Ma et al., 2007). If the system user is not satisfied with the result he/she has to revise the initial user query to get new recommendations.

- **Conversational systems**: In conversational recommender systems it is possible to refine the user query by providing feedback to the recommendations (Viappiani et al., 2007). The user feedback of each recommendation cycle is used in the calculation of future recommendations. This incremental construction of user preferences helps users to quickly navigate to suitable products in the product space (Burke et al., 1997). Different strategies for capturing user feedback have been explored, which can be categorized in four forms: value elicitation, tweaking, preference-based, and ratings-based feedback (see, for example, (McGinty and Smyth, 2002; Smyth and McGinty, 2003; McCarthy et al., 2005)). *Value elicitation* and *tweaking* both operate on the attribute level, which means that users are asked to provide their preferences concerning the attributes of a recommended product (McGinty and Smyth, 2002). In value elicitation the customer has to provide a specific value for a specific attribute (McGinty and Smyth, 2002) (for example, *show me holidays with a price lower than €1.000*), whereas in tweaking users provide feedback by critiquing attributes of recommended items in a directional way (McCarthy et al., 2005) (for example, *show me holidays that are cheaper than the actual recommendation*).

  In *preference-based* and *ratings-based* feedback methods users express their preferences on the product level (McGinty and Smyth, 2002). Preference-based systems present a list of products that matches the user requirements and the user is asked to select the preferred item out of this list (McGinty and Smyth, 2002). In ratings-based systems users give ratings on items according to their preferences (Lemire and McGrath, 2005), and recommendations are based on these ratings.

Burke (Burke, 2002a) introduced an approach to categorize recommender systems based on the used knowledge sources for finding items to recommend (see Figure 2.1). A typical knowledge source exploited by recommender systems is the *knowledge about the user (customer)*. This knowledge can either be represented by implicit collected user information, such as the purchase history, or by explicitly defined customer requirements (Hu et al., 2008; Jawaheer et al., 2010). *Collaborative knowledge* consists of information collected from previous system users. The *content knowledge* contains information about available items and corresponding item attributes. The *knowledge about the domain* can be modeled by a set of domain constraints that describe relationships between customer preferences and the product attributes, as well as dependencies between product attributes. On the basis of these different knowledge sources, we distinguish three recommendation techniques: *collaborative*, *content-based*, and *knowledge-based recommendation*. In the following we will provide an overview of these recommendation techniques.

2.1.1. Collaborative Recommender Systems

The term "*Collaborative Filtering*" was coined by the developers of *Tapestry*, the first recommender system (Goldberg et al., 1992). The basic idea of the Tapestry mail system was to make information filtering more effective by involving humans in the filtering process (Goldberg et al., 1992). Users

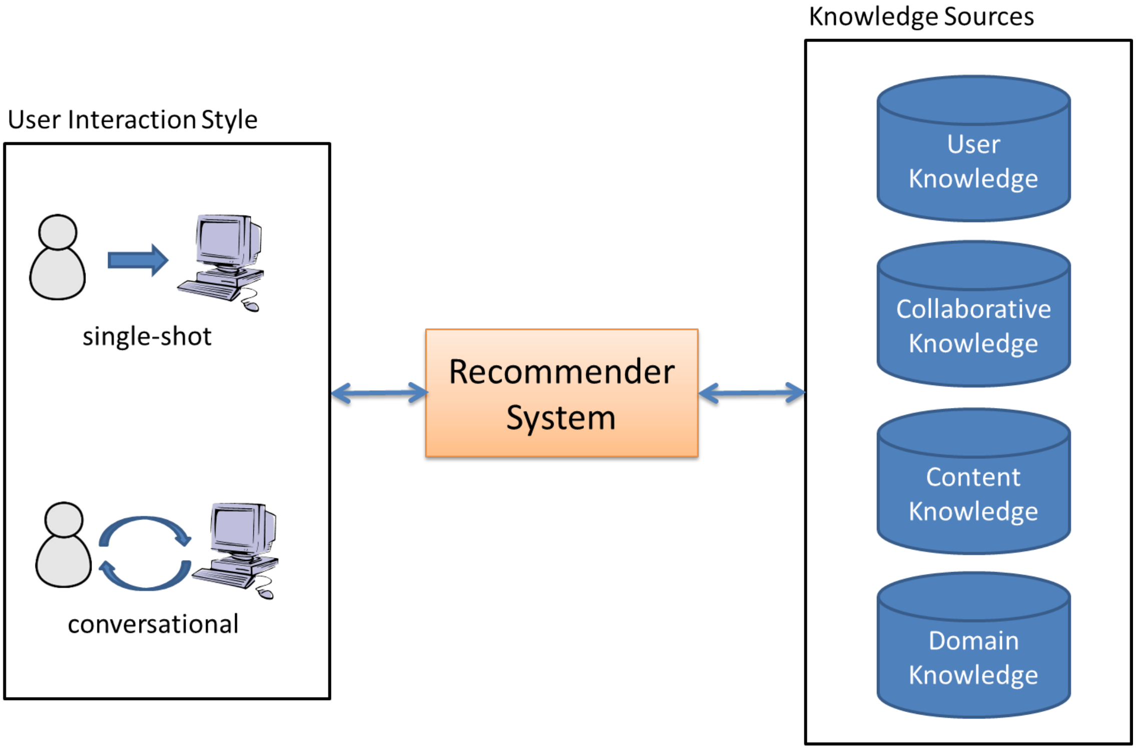

Figure 2.1.: User interaction styles of recommender systems and knowledge sources exploited by recommender algorithms to calculate recommendations for the user.

of the system were allowed to rate messages (”good”, ”bad”, free text annotation) and therefore it was possible to manage documents based not only on the content, but also on the opinions of users who had already read them.

In collaborative recommender systems information about the behavior or opinions of other system users are exploited to find items that may be of interest for the current user. The user interests are typically expressed by ratings on items, or by a purchase history. The core technique of such a system is to match people with similar interests and then make recommendations on this basis (Herlocker et al., 2000; Terveen and Hill, 2001). For example, if *user A* and *user B* have a similar rating or purchase history, and *user A* purchases or rates an item *user B* has not seen yet, a collaborative recommender will recommend this item to *user B*. An example of a collaborative recommender system is *last.fm*[1], a web-based system where people share their tastes regarding music and then receive music recommendations on the basis of other users with similar music tastes. This key technology of collaborative filtering, to exploit user preferences for the calculation of recommendations, reflects a common social practice: seek for and rely on the experience and opinions of other people. Typically such systems do not use any information about the attributes of items for the recommendation calculation, and therefore no deep knowledge about the product assortment is needed (see Figure 2.2). This means, an online retailer does not need to enter detailed and up-to-date information about the item's attributes into the recommender

[1] `http://www.last.fm`

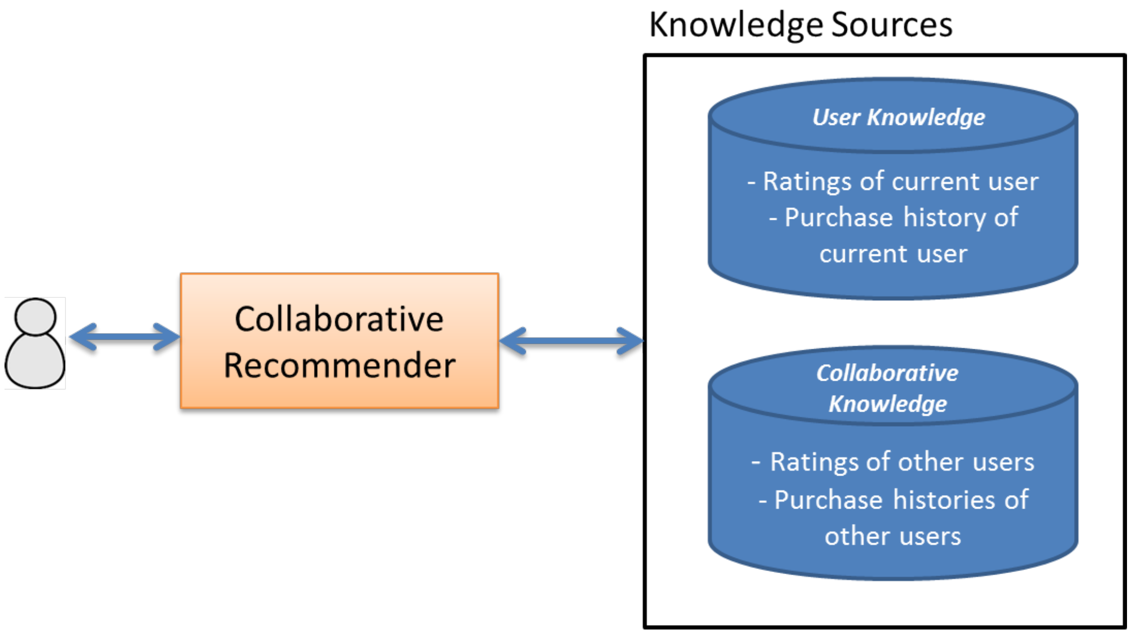

Figure 2.2.: Collaborative Recommender – exploited knowledge sources for calculating collaborative recommendations.

system, which can be a very time intensive task – for example, at Pandora[2], a popular personalized internet radio service, the characteristics of a song are specified by 400 attributes.

There exist two basic variants of collaborative recommender algorithms:

- *User-based collaborative filtering* uses information about the behavior of other users, such as users' purchasing habits or product ratings, to find users similar to the current user (*nearest neighbors*) and recommend items that were interesting to them (Herlocker et al., 1999; Resnick et al., 1994). This approach is also known as *neighborhood-based collaborative filtering* (Bell and Koren, 2007).
- *Item-based collaborative filtering* focuses on finding items similar to those, the current user has already purchased or rated. Such systems attempt to find other items with similar ratings, or items that users tend to purchase together. This kind of recommendation algorithm is also known as *shopping cart recommendation* (Linden et al., 2003). The advantage of the item-based approach is that the algorithm operates on a small set of items (only those items the user has rated) and therefore represents a very fast solution (Linden et al., 2003).

As an example, Table 2.1 contains item ratings from three system users, *user1*, *user2*, and *user3*. Applying the user-based strategy to Table 2.1 to find a recommendation for *user3*, the first step is to calculate the similarity between users. Breese et al. (Breese et al., 1998) provide a good overview of collaborative filtering strategies and an empirical evaluation of the predictive accuracy of the various algorithms. A well known correlation measure is the *Pearson's correlation coefficient* – see Formula 2.1. In this Formula $r_{x,i}$ specifies the rating of user x for item i (for example, *user3*

[2] `http://www.pandora.com`

Table 2.1.: Collaborative filtering – sample rating statistic (1=bad; 5=very good).

	item1	item2	item3	item4	item5
user1	1	1	-	2	-
user2	4	4	1	-	5
user3	-	5	1	-	-

rated *item2* with a "5" on a 1-to-5 rating scale), and $\overline{r_x}$ specifies the average rating score of user x ($\overline{r_{\text{user1}}}$=1.33; $\overline{r_{\text{user2}}}$=3.50; $\overline{r_{\text{user3}}}$=3 for rating entries in Table 2.1).

$$sim(x,y) = \frac{\sum_i^n (r_{x,i} - \overline{r_x})(r_{y,i} - \overline{r_y})}{\sqrt{\sum_i^n (r_{x,i} - \overline{r_x})^2}\sqrt{\sum_i^n (r_{y,i} - \overline{r_y})^2}} \tag{2.1}$$

The similarity calculation for the user ratings shown in Table 2.1 will identify *user2* as nearest neighbor of *user3* in our scenario (*user2* has similar ratings on *item2* and *item3* compared to *user3*). Therefore *item1* and *item5* will be selected as recommendations for *user3*, since both received an excellent rating by *user2*, and they have not been rated yet by *user3*.

The item-based approach calculates similarities of items instead of users – for each of the user's purchased or rated items the algorithm focuses on finding items that were purchased together, or items that were rated similar (Linden et al., 2003). A well-known measure in this context is the *Cosine similarity measure* (see Formula 2.2), that calculates the similarity of two vectors $\vec{a}$ and $\vec{b}$ based on the angle between them. From Table 2.1 we can extract, for example, the rating vectors (1,4,0) for *item1*, (1,4,5) for *item2*, and (0,1,1) for *item3*. We can see that *item2*, which received a high rating from *user3*, received similar ratings as *item1* from *user1* and *user2*. Therefore the item-based approach will select *item1* as recommendation for *user3*.

$$sim(\vec{a},\vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| * \left|\vec{b}\right|} \tag{2.2}$$

At *Amazon.com* both approaches are integrated at the website[3] (Linden et al., 2003) (see Figure 2.3). The section '*Customers Who Bought This Item Also Bought*' on a product information side lists user-based recommendations, whereas item-based recommendations are given in the "*Frequently Bought Together*" section (Linden et al., 2003).

Collaborative recommender systems allow acceptable recommendation accuracy for frequently bought products such as music, or books (Felfernig et al., 2007). A major drawback of collaborative filtering is that such a system must be initialized with a large amount of data, because the quality of collaborative filtering suffers in case of sparse preference databases – this problem is known as *sparsity problem* (Burke, 2000; Debnath et al., 2008; Papagelis et al., 2005). Another well known problem of collaborative recommender systems is the "*cold start problem*", that deals with the fact that, if a new item is added to the product database, it can not be recommended until someone has rated it (also known as *first rater problem*) (Schein et al., 2002; Kumar and Bhatia, 2012). The similar problem applies to new users of the system – new users cannot get accurate

[3] `http://www.amazon.com`

Figure 2.3.: Collaborative filtering at Amazon.com.

recommendations until they have rated or purchased a considerable number of items. In order to tackle these problems, various *hybrid recommendation strategies* have been developed, that combine different recommendation approaches (see, for example, (Balabanovic and Shoham, 1997; Schein et al., 2002; Burke, 2002a)).

2.1.2. Content-Based Recommender Systems

The basic strategy of the content-based approach is to recommend items similar to those items the user has liked in the past (Pazzani and Billsus, 1997; Terveen and Hill, 2001; Pazzani and Billsus, 2007). The calculation of recommendations is based on the correlation between the attributes of items and the user's preferences. An important knowledge source of content-based recommenders is the item description (see Figure 2.4). Each item is typically represented by a feature vector or an attribute profile, that hold numeric or nominal values describing certain aspects of the item, such as the price (Debnath et al., 2008). For example, a book might be described by its genre, the author, etc. – Table 2.2 gives a simple example of structured data representing a book database. Each book is described by the same set of attributes, and there is a known set of values that can be applied to these attributes. Often a weighting scheme is included to specify the importance of the attributes. For example, when choosing a movie, the genre might be more important for the user than the actors. The authors of (Debnath et al., 2008) introduced a method for determining feature weights on the basis of a collaborative social network graph of items. They demonstrated the effectiveness of feature weighting by comparing their approach with a pure content based method where equal weights are considered for all features. The results of their study indicate

Table 2.2.: Content-based filtering – book database and sample rating statistic (1=bad; 5=very good).

Product			Rating		
Title	Genre	Author	user1	user2	user3
The Da Vinci Code	Thriller	Dan Brown	5	-	3
The Lord of the Rings	Fantasy	J. R. R. Tolkien	2	5	-
Steve Jobs	Biography	Walter Isaacson	2	1	4
The Firm	Thriller	John Grisham	-	3	2
Harry Potter	Fantasy	J. K. Rowling	-	-	4

that the integration of feature weighting techniques can improve the quality of recommendation results (Debnath et al., 2008).

The first recommender systems were designed for recommending unstructured, text-based items, such as mails and news articles (see, for example, (Goldberg et al., 1992)). In such recommendation scenarios, techniques from information retrieval are applied to create a structured representation of text documents. A well known approach is to calculate *tf*idf* (term frequency*inverse document frequency) weighting factors, that specify the importance of a word in a document out of a document collection – see Formula 2.3 (Pazzani and Billsus, 2007). In this Formula, $tf_{t,d}$ represents the frequency of a term t in the document d, df_t refers to the number of documents that contain the term, and N is the number of documents in the collection. A high *tf*idf* weight of a term t indicates that this term specifies a central topic of the document (Pazzani and Billsus, 2007).

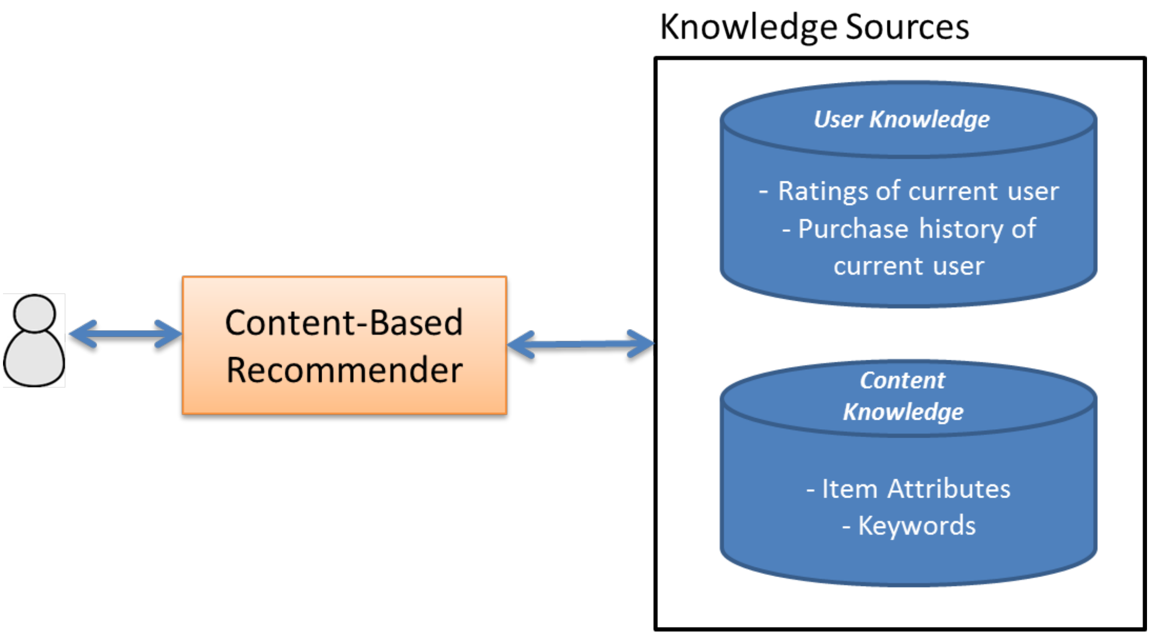

Figure 2.4.: Content-Based Recommender – exploited knowledge sources for calculating content-based recommendations.

$$w(t,d) = \frac{tf_{t,d}\ log\left(\frac{N}{df_t}\right)}{\sqrt{\sum_i (tf_{t_i,d})^2\ log\left(\frac{N}{df_{t_i}}\right)^2}} \tag{2.3}$$

Another important knowledge source of content-based filtering are the user preferences (see Figure 2.4). A user's preferences are typically expressed by entries in a user profile (Bridge and Kelly, 2005), that represent already purchased or rated items. Figure 2.5 shows an example of a recommender system that collects user ratings on items. Jinni[4] is a popular movie recommender which implements the content-based approach. Jinni categorizes movies into "genes" according to attributes such as mood, plot, genres, and style. The recommendation system learns the "*movie personality*" of users, which is based on ratings of search results, and compares the genes of all movie titles in the system's catalog along with the user preferences to calculate recommendations. To rate a movie, a 11-point scale is provided, ranking from *Terrible* to *Must See* (see Figure 2.5). Furthermore, if the user has not seen the movie yet, it is possible to specify if he/she will or will not see this movie in the future.

To calculate more targeted recommendations, the preferences in a user model can be extended by additional information, for example, viewed items, or demographic information (Linden et al., 2003). For example, if a middle-aged user of a movie recommender is interested in fantasy movies, the recommender can propose the movie "*The Lord of the Rings*". For a ten year old user "*Harry Potter*" would be a better recommendation.

The focus of a content-based recommendation algorithm is set on finding items that best match the preferences of the current user. For example, in Table 2.2 we can see that *user1* gave a high rating on the book *The Da Vinci Code* – therefore we can conclude that this user prefers *thrillers* or books from the author *Dan Brown*. If we want to calculate a recommendation for *user1*, a simple content-based strategy is to recommend books out of the preferred categories, which the user has not rated yet. In our case the book *The Firm* will represent a good recommendation, since it's genre belongs to the category *thriller*. A number of different techniques from machine learning and information retrieval are applied in content-based recommender systems to learn and adapt preferences through feedback the user provides on items, and to filter items for those that satisfy the user preferences (for example, decision trees, nearest neighbor methods, linear classifier) – see, for example, (Pazzani and Billsus, 2007) for an overview of classification learning algorithms in content-based recommenders.

Since content-based recommender systems calculate their recommendations based upon the description of an item, and a profile of the user's preferences, there exist no "*new item problem*" like in the collaborative approach. But similar to collaborative recommendation systems there exists the "*new user problem*" – the prediction quality of a content-based recommender algorithm depends on the number of rated or purchased items (Pazzani and Billsus, 2007). Another drawback of the content-based strategy is the suppression of serendipity – recommendations are restricted to items similar to those already rated or purchased by the user (Balabanovic and Shoham, 1997), which prevents users from discovering new items.

[4] http://www.jinni.com

Figure 2.5.: Jinni – a content based movie recommender that collects user ratings to learn the "*movie personality*" of users.

2.1.3. Knowledge-Based Recommender Systems

Traditional approaches to recommendation (collaborative filtering (Goldberg et al., 1992; Konstan et al., 1997; Linden et al., 2003), content-based filtering (Pazzani and Billsus, 1997, 2007), and different hybrid variants thereof (Burke, 2002a)), are well applicable for recommending quality and taste products such as movies, books, groceries, music, or news (Felfernig et al., 2007). Especially in the context of high-involvement products such as computers, cars, apartments, or financial services, those approaches are less applicable. For example, apartments are not bought very frequently – consequently the corresponding items will not receive a critical mass of ratings needed for making reasonable predictions. For example, Bell and Koren (Bell and Koren, 2007) propose to use the 100 nearest neighbors in their collaborative filtering recommendation approach. Furthermore, a low frequency of user ratings would require to take into consideration a rather long time period of gathering ratings – this would make it infeasible for collaborative and content-based filtering algorithms to derive meaningful predictions, since both approaches are based on the existence of a purchase or rating history.

Especially in domains where traditional recommendation approaches are not the first choice, knowledge-based recommendation technologies come into play (Burke, 2000; Felfernig and Burke, 2008). Knowledge-based recommender applications are exploiting explicitly defined requirements of the user and additionally dispose of deep knowledge about the underlying product assortment (see Figure 2.6). In knowledge-based recommender systems more complex user interaction is

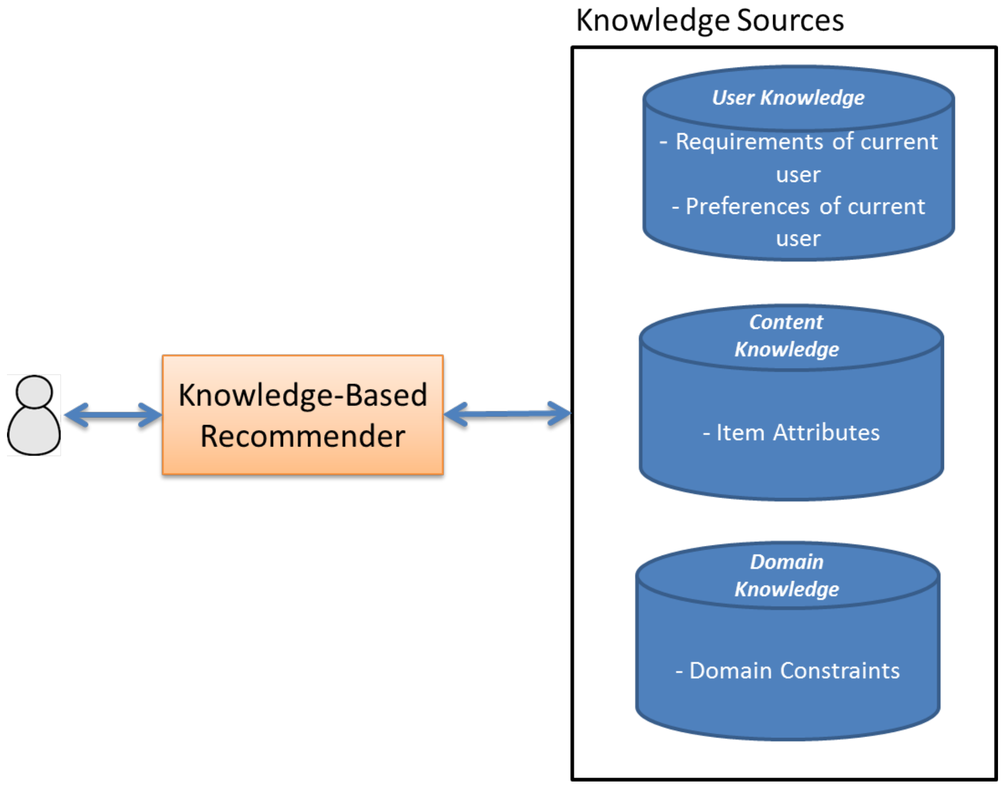

Figure 2.6.: Knowledge-Based Recommender – exploited knowledge sources for calculating knowledge-based recommendations.

needed to gather information about user preferences compared to content-based and collaborative recommenders. In content-based and collaborative systems the preferences are typically expressed in terms of a rating or purchase history, whereas in knowledge-based recommenders the user is involved in a preference elicitation process. The *single-shot* approach to determine user requirements is simply to ask the user about his/her preferences. In *conversational* interaction styles users typically interact with the system through a feedback dialog on recommendations.

Based on explicitly defined user preferences, knowledge-based systems generate recommendations by reasoning about which items satisfy the user's requirements (Burke et al., 1999). Different strategies are applied to rank items according to the user preferences – for example, in *constraint-based systems* domain expertise is modeled through constraints that are used to determine a set of items that may be of interest for the customer. For example, if the user of a computer recommender specifies that he/she wants to play games with the laptop, all products with an onboard video card should be excluded from the recommendation list. Thus, knowledge-based recommender applications exploit knowledge sources that are typically not available in collaborative and content-based filtering scenarios. A direct consequence of the availability of deep knowledge about the product assortment and explicitly defined customer requirements is that no ramp-up problems occur (Burke, 2000; Felfernig and Burke, 2008). The other side of the coin is that – due to the explicit representation of recommendation knowledge in a recommender knowledge base – knowledge-based recommenders cause so-called *knowledge acquisition bottlenecks*: knowledge engineers and domain experts have to invest considerable time efforts in order to develop and keep those knowledge bases up-to-date.

The major difference between filtering-based recommendation approaches and knowledge-based recommendation (Burke, 2000; Felfernig and Burke, 2008) is that the latter use explicit knowledge about customers, the product assortment, and the dependencies between customer preferences and the product attributes (see Figure 2.6). The rules for the identification of a solution are explicitly defined and thus allow the derivation of intelligent and deep explanations regarding the recommendation results. Since advisory knowledge is represented in the form of variables and constraints we are able to automatically determine repair actions in situations where no solution can be found for the given set of customer requirements (Felfernig et al., 2004, 2009). Knowledge-based recommendation problems can be defined on the basis of simple conjunctive database queries, as well as on the basis of so-called constraint satisfaction problems (CSPs) (Tsang, 1993). Figure 2.7 presents an example of a knowledge-based recommender application that has been developed for one of the largest financial service providers in Austria. Such an application guides a user (repeatedly) through the following phases:

1. *Requirements specification* (Phase I.): in the first phase users are interacting with the recommender application in order to identify and specify their requirements. Examples for such requirements in the financial services domain are *the investment period should be below four years*, *the profit per year should be more than 5 percent*, or *the recommended items should not contain shares*.
2. *Repair of inconsistent requirements* (Phase II.): in the case that the recommender application is not able to identify a solution, it proposes a set of repair actions (change proposals for requirements) that (if accepted by the user) can guarantee the identification of a recommendation. An example of such an infeasibility in the financial services domain is a *low willingness to take risks* combined with *high return rates of the investment*. Another example of an infeasibility is the combination of *high return rates* and *short investment periods*.
3. *Result presentation* (Phase III.): if the requirements can be fulfilled, the recommender application presents a set of product alternatives. These alternatives are typically ranked on the basis of a utility function (for a detailed example see (Felfernig and Burke, 2008)) and are either presented as an ordered list or on a product comparison page.
4. *Explanations* (Phase IV.): For each of the identified and presented product alternatives the customer can activate a corresponding explanation as to why this product has been recommended. Each explanation consists of argumentations that relate specified user requirements with the corresponding product properties. An example of an explanation is *we recommend this product since it supports the specified investment period and additionally provides reasonable return rates with low risks*.

2.2. Critiquing Systems

Critiquing-based recommender systems adopt a conversational interaction style – users of such systems provide feedback by critiquing attributes of recommended items in a directional way

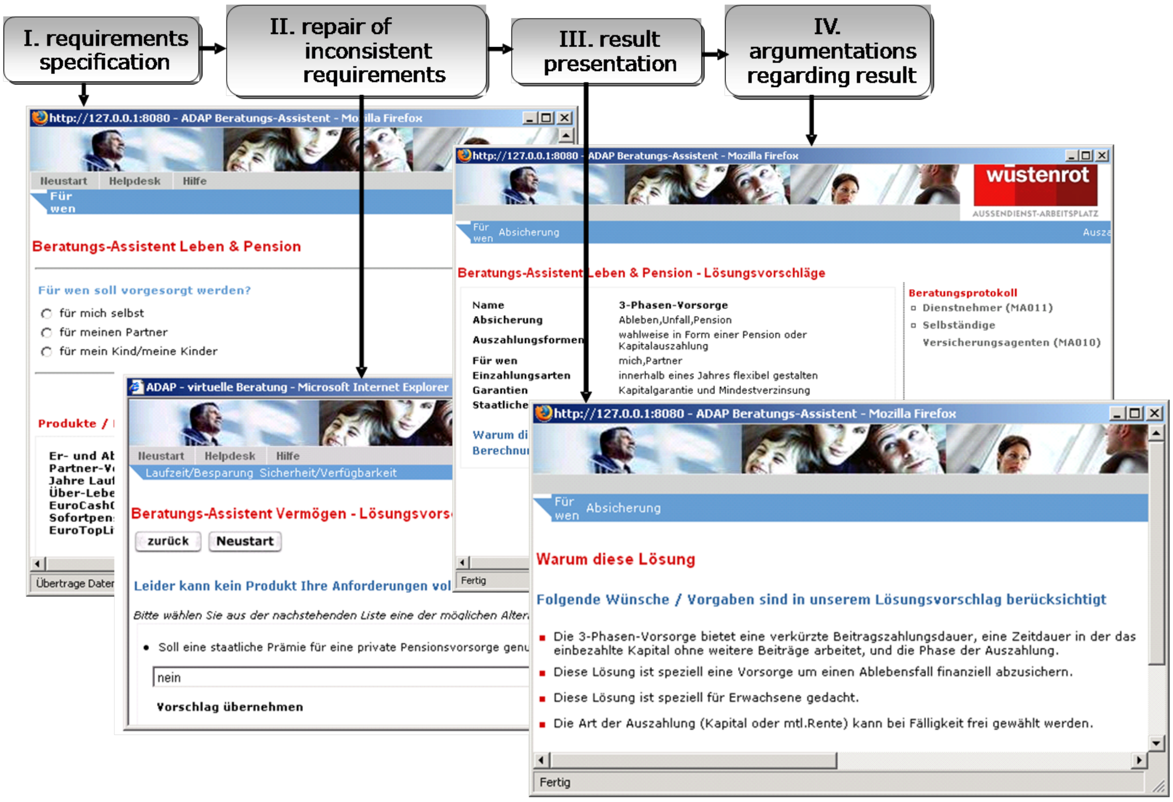

Figure 2.7.: Example knowledge-based recommender application. A typical recommendation process consists of the phases requirements specification (I.), repair of inconsistent requirements (II.), presentation of recommended products (III.), and explanation of each selected product (IV.).

(McCarthy et al., 2005; Viappiani et al., 2007). Figure 2.8 exemplifies the basic user-system interaction steps in a critiquing-based recommender system. The system displays a set of k example candidates in response to an initial user query. If the user can't find his/her target item among these examples, he/she can critique attributes of the example products. The user's critiques are then used to refine the user preference model and to calculate new recommendations. The critiquing-based recommendation approach has been realized in knowledge- (Burke et al., 1997), content- (McCarthy et al., 2004), preference- (Zhang and Pu, 2006), and user-based recommender systems (McCarthy et al., 2010).

The first system that integrated this example critiquing strategy was RABBIT, an intelligent database assistant (Frederich et al., 1982). The purpose of the RABBIT system was to assist users in formulating a database query by supporting two techniques of human remembering: *descriptive retrieval* and *retrieval by instantiation* (Frederich et al., 1982). *Descriptive retrieval* refers to the fact that people retrieve information from their memory by iteratively constructing partial descriptions of the desired target item (Bobrow and Collins, 1975; Williams and Hollan, 1981), and *retrieval by instantiation* explains that the information retrieved in each iteration of the retrieval process represents an instantiation, i.e., an example item suggested by the partial

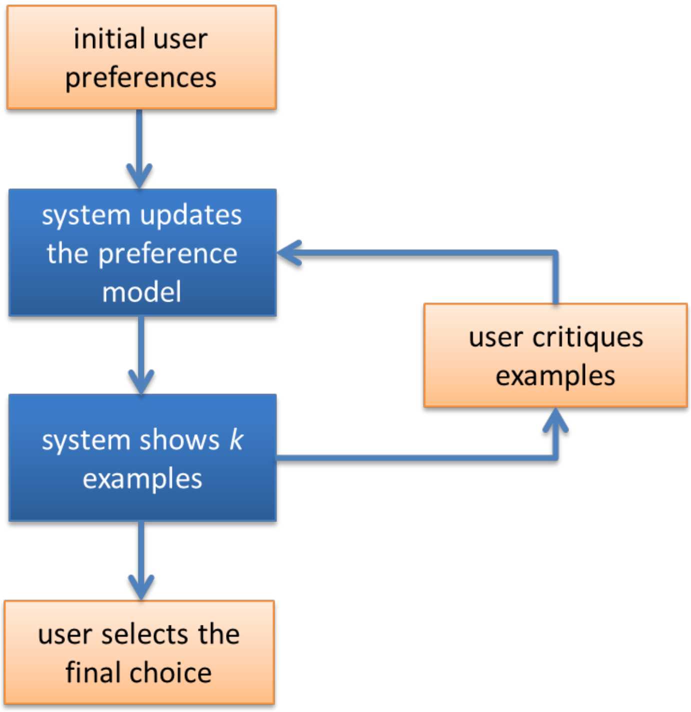

Figure 2.8.: The user interaction model of a critiquing-based recommender system.

descriptions (Williams and Hollan, 1981).

A major advantage of critiquing systems is, that the user can constrain a particular product attribute without providing a specific value for it (Reilly et al., 2007a). For example, a user of a holiday recommender might express that he/she wants to see a holiday package that is *cheaper* than the actual recommendation by critiquing the corresponding *price* attribute. Figure 2.9 shows an example of a critiquing-based user interface. *Movie Tuner* is integrated in the *Movielens* webpage[5] and allows users to adjust their movie selection by requesting more or less of specific qualities, for example, "more action" or "less violent".

Research on human decision making has shown that consumers show an adaptive and constructive decision behavior (Bettman et al., 1998). The critiquing-based approach supports this incremental preference construction process by allowing its users to learn and adapt their preferences according to concrete examples (Zhang and Pu, 2006). Users are rarely able to provide complete and accurate preferences at the begin of a recommendation session and therefore the feedback can be inconsistent and contradictory (Reilly et al., 2007b). This fact is typically taken into account by an incremental adaption of the user preference model. Furthermore, critiquing-based recommender systems do not require users to specify preferences about attributes they might not be sure about. A user's critiques on the recommendations of each critiquing cycle are used for refining the user preference model. The user's updated preferences are then used to find recommendations that are more likely to be of interest to the user (Chen and Pu, 2007b). Research has shown that example critiquing interfaces enable users to perform decision tradeoff tasks more

[5] http://www.movielens.org/

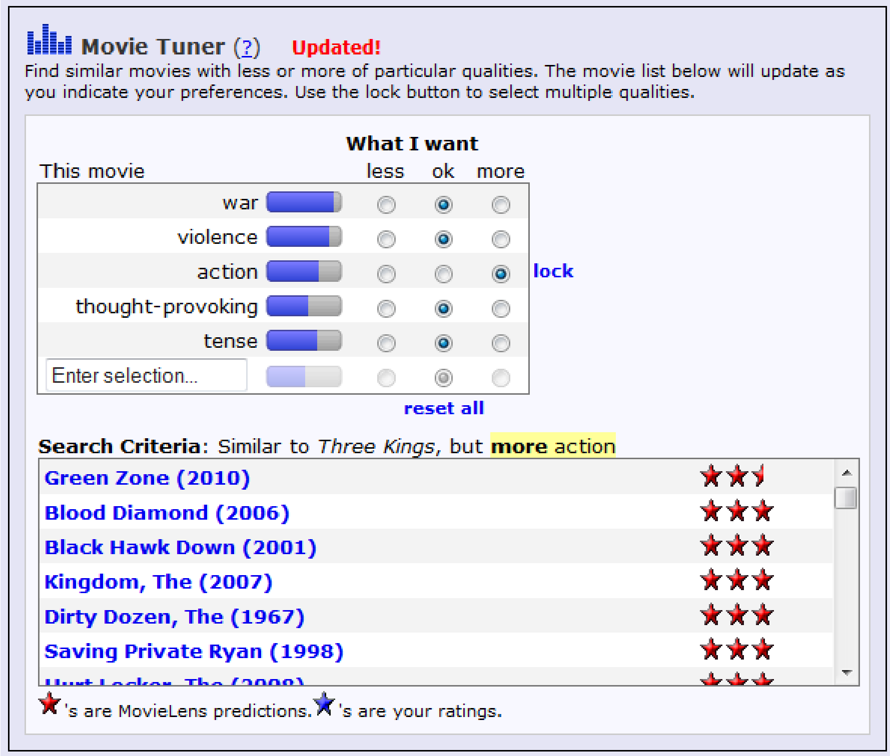

Figure 2.9.: Movielens' *Movie Tuner* critiquing interface.

efficiently with fewer errors compared to non-critiquing interfaces such as, for example, ranked lists (Pu and Kumar, 2004; Pu and Chen, 2005).

There exist different approaches to integrate critiquing in a decision support tool. One is the *unit critiquing* approach (Burke et al., 1996; McCarthy et al., 2010) where users apply one critique on one single product attribute in each critiquing cycle. Typically unit critiques are expressed by "more", "less", or "other". Another critiquing approach is to provide *compound critiques*, which give the possibility to critique multiple product attributes during one critiquing cycle (Burke et al., 1997; Reilly et al., 2004, 2005; Zhang and Pu, 2006). An example of such a compound critique in the laptop domain could be "*Less Memory, Cheaper and Different Processor*". Section 4.1 gives a more detailed overview of existing critiquing algorithms. An in-depth discussion of critiquing techniques can be found in (Chen and Pu, 2012).

2.3. Product Configuration Systems

Configuration systems have a long tradition as a successful application area of Artificial Intelligence, see, for example, (Barker et al., 1989; Mittal and Frayman, 1989; Stumptner, Stumptner; Sabin and Weigel, 1998; Fleischanderl et al., 1998). On an informal level, configuration can be interpreted as a "*special case of design activity where the artifact being configured is assembled from instances of a fixed set of well-defined component types which can be composed conforming to a set of constraints*" (Sabin and Weigel, 1998). Constraints can be represented by rules related to, for example, "*technical restrictions, economic factors and restrictions according to the production process*" (Felfernig, 2004).

The basic concept of *mass customization* is the production of customer-individual products with near mass production efficiency (Pine, 1993; Tseng and Jiao, 2001). This concept was later extended by methods for customizing also the consumer *experience* (Pine and Gilmore, 1999), which includes, for example, the presentation of products (Schafer et al., 2001). Following the paradigm of mass customization, the intelligent customizing of products and services is crucial for manufacturing companies to stay competitive. Product configuration systems support the idea of integrating the customer in the design and production process. Instead of *choosing* a product, a user of a product configurator *specifies* a product by configuring the product attributes based on a set of available options. Product configurators have been recognized as ideal tools to assist customers in configuring complex products according to their individual requirements (Blecker et al., 2004; Yang et al., 2005). Example domains where product configurators are applied are computers, cars, industrial machinery, financial services, and telecommunication switches. A popular example of an online configurator is the Dell product configurator[6]. After selecting a base product, customers can choose from many options (for example, processor, operating system) to customize their Dell product according to their individual preferences. Another example is the Nike company, which shows that this personalization aspect can also be applied to standardized products, such as footwear. On the website `www.nikeid.com` customers can select a basic shoe, that can be customized in terms of, for example, color, grip sole, and it is also possible to put a name, or a specific number on the shoes.

An important task typically supported by configurators is to check the consistency of user requirements with the knowledge base, such that incorrect quotations and orders can be avoided. Product configuration systems have many advantages such as the elimination of errors, shorter product delivery cycles, and an increased productivity of sales representatives (Barker et al., 1989; Heiskala et al., 2007).

Integrating Recommendation Technologies in Product Configuration Systems

Product configuration systems are increasingly being used by manufacturing companies to assist customers in specifying their requirements, and to find a product that matches their preferences. A side-effect of the high diversity of products offered by a configurator is that the complexity of the alternatives may outstrip a user's capability to explore them and make a buying decision. Previous research has shown that consumers are often overwhelmed by high variety categories because of the large amount of options to evaluate (Malhotra, 1982). Since humans have limited processing capacity (Streufert and Driver, 1965; Bettman, 1979), confronting consumers with too much information can lead to an information overload and therefore can result in decreased quality of decision performance (Jacoby et al., 1974).

Another challenge is that users often have no clear knowledge of which solution might fulfill their needs (Bettman et al., 1998; Piller and Franke, 2002). This fact is described by the *theory*

[6] `http://www.dell.com/`

of preference construction (Bettman et al., 1998), that explains that users typically do not know exactly which products or components they would like to have. As a consequence, users construct and adapt their preferences within the scope of a configuration process (Bettman et al., 1998).

Research has shown that recommendation technologies can support customers in the specification of their requirements and thus can help to achieve a higher customer satisfaction (Geneste and Ruet, 2001; Coster et al., 2002; Felfernig et al., 2010b). The integration of recommendation technologies with knowledge-based configuration is still in a very early stage. There exist some contributions that take into account the application of personalization technologies in the configuration context. Geneste and Ruet (Geneste and Ruet, 2001) introduce an approach to the integration of case-based reasoning methods (Kolodner, 1993; Smyth and Keane, 1996; McSherry, 2003) with constraint solving (Junker, 2004) with the goal to adapt nearest neighbors identified for the current problem. The work of Coster et al. (Coster et al., 2002) and Felfernig et al. (Felfernig et al., 2010b) focuses on the recommendation of product features and feature values, and strategies to integrate these recommendations in the user interface.

In the following we will discuss selected strategies to support users of product configuration systems by applying recommendation technologies.

Default Recommendations

A possibility to help the user identifying meaningful alternatives that are compatible with their current preferences is to provide *defaults*. Defaults in the context of interactive configuration dialogs are *preselected options used to express feature recommendations* (Mandl et al., 2011b). Providing defaults can help users to get a better understanding of the compatibility of already specified sub-configurations and possible instantiations of other features. Furthermore, defaults support the idea that users specify only those features which are relevant to them, and let the system find meaningful completions for configurations (Falkner et al., 2011).

There exist different approaches for recommending defaults:

- **Static default recommendation:** McSherry (McSherry, 2005) presents an approach to apply defaults by integrating the idea that there are product attributes whose values most users would prefer to maximize or minimize (*default preferences*). These attributes are denoted as *more-is-better* (MIB – for example, resolution, or optical zoom in the digital camera domain) or *less-is-better* (LIB – for example, price, or weight) attributes, and are combined to a *default query* (McSherry, 2005). This default query contains the default preferences, which mean the maximum values for MIB attributes and the minimum values for LIB attributes.

  Table 2.3 contains three example products from the mobile phone domain. *pModel* specifies the existing phone models, *pStyle* the phone styles, *pHSDPA* specifies the supported HSDPA data rate, *pGPS* whether the phone supports GPS navigation, and *pPrice* specifies the price of the phone. For our example phones the preference value for the attribute *pPrice* will be 69 (LIB – the cheapest availabe phone), and the preference value for the attribute *pHSDPA* will be 7.2 (MIB – the highest available data rate) – therefore the default query

Table 2.3.: Available phone models in working example.

pModel	pStyle	pHSDPA	pGPS	pPrice
p1	*bar*	*0*	*false*	*69*
p2	*clam*	*7.2*	*true*	*149*
p3	*clam*	*3.6*	*false*	*99*

is $Q = \{pPrice = 69; pHSDPA = 7.2\}$. For the non numerical attributes *pModel*, *pStyle*, and *pGPS* no assumptions can be made about the preferred values, and therefore these attributes do not appear in the default query. To calculate the similarity of a given item I to the default query Q with respect to a numeric attribute a, Formula 2.4 can be used for MIB attributes, and Formula 2.5 can be used for LIB attributes (McSherry, 2005). In this formulae, *max(a)* and *min(a)* refer to the maximum and minimum values of the attribute a in the case base, $\phi_a(Q)$ is the value of a in the default query Q, and $\phi_a(I)$ is the value of a for the item I.

$$sim_a(I,Q) = \frac{\phi_a(I) - \phi_a(Q)}{max(a) - min(a)} = \frac{\phi_a(I) - min(a)}{max(a) - min(a)} \tag{2.4}$$

$$sim_a(I,Q) = \frac{\phi_a(Q) - \phi_a(I)}{max(a) - min(a)} = \frac{max(a) - \phi_a(I)}{max(a) - min(a)} \tag{2.5}$$

Applying the concept of static default recommendation to our example phones in Table 2.3, we can calculate the similarities between the phones and the default query Q for the numeric attributes *pPrice* and *pHSDPA*. We treat the price attribute as a LIB attribute, and therefore the phone with the lowest price (*p1*) has the highest similarity value of 1, and the phone with the highest price (*p2*) has a similarity value of 0. Phone *p3* has a similarity value of 0.625. In the case of the attribute *pHSDPA* we use the formula for MIB attributes (Formula 2.4). Thus phone *p2* has a similarity of 1, since it supports the highest data rate, phone *p1* has a similarity value of 0, since it does not support HSDPA, and phone *p3* has a similarity of 0.5. Summing these similarity values up, will result in the highest similarity value of 1.125 for phone *p3*, and therefore the attributes of *p3* will be presented as *defaults*.

The idea of *default preferences* introduced by McSherry (McSherry, 2005) can be used as a good starting point for the elicitation of personal preferences.

- **Rule-based default recommendation:** The rule-based calculation of defaults takes into account explicitly defined rules, which are set into context with already specified user requirements (Falkner et al., 2011). In our example in the mobile phone domain, we can, for example, specify the rule, that if the customer indicates that he/she often wants to use the mobile phone for browsing the web, the attribute value for *HSDPA* should be set to the highest value:

  - *(webUse = often) → (pHSDPA=7.2) /* frequent web use requires a fast internet connection */*

  In contrast to the domain constraints in the product knowledge base, these rule-based recommendations can be changed by the customer. An example of a domain constraint in the mobile phone domain is, if the customer wants to use the mobile phone for browsing the web, the mobile phone must be able to connect to the internet:

– *(webUse != no) → (pHSDPA>0) /* web use requires an internet connection */*

- **Collaborative default recommendation:** In this approach the system exploits information about already completed configuration sessions from previous system users to recommend defaults for unspecified features to the current user. In Section 4.2.2 we discuss the application of *Nearest Neighbors* and *Naïve Bayes Voter* algorithms to the collaborative recommendation of feature values.

Recommendation of Repair Alternatives

In situations where no configuration can be found for a given set of customer requirements, a possible approach is to present repair alternatives to the user (Reiter, 1987; Felfernig et al., 2004, 2008, 2009). Repair alternatives are a set of user requirements which have to be changed such that a configuration solution can be identified. For example, the following set of customer requirements $C_R = \{r_1 : \mathit{styleReq{=}bar}, r_2 : \mathit{webUse{=}often}, r_3 : \mathit{GPSReq{=}true}\}$ does not allow the calculation of a solution in our working example (see Table 2.3). Therefore, we have to identify a minimal set of requirements that has to be changed in order to be able to restore consistency. Quite often there exists a large number of alternative repair actions for inconsistent customer requirements (Felfernig et al., 2009). The integration of personalization aspects can help to reduce the number of relevant repair alternatives. Felfernig et al. (Felfernig et al., 2009) introduced an approach that integrates collaborative problem solving techniques into the calculation of repair proposals in order to identify repairs similar to the original customer requirements. The results of their research showed that the personalized repair approach can significantly improve the prediction accuracy for interesting repair alternatives (Felfernig et al., 2009).

Ranking of Results

When the specified user requirements lead to a large number of possible solutions, ranking strategies are needed to present the results to the user in a convenient way. Research has shown that users are more likely to explore items that appear early in the results ranking – this effect is known as *primacy effect* (Asch, 1949; Murphy et al., 2006; Felfernig et al., 2007). There exist a number of literature which explain different result ranking approaches – see, e.g., (Lai et al., 2006; Felfernig et al., 2008, 2010b). In (Felfernig et al., 2010b) a similarity-based approach for the ranking of the products is introduced (see Section 4.2.2). Completed configurations from previous system users are used to calculate the similarity to the user's current configuration, so that phones from nearest configurations are shown first. Looking at Table 2.4, which contains three valid configurations from previous configuration sessions (referred to as $conf$) and a partially specified configuration of the active user $conf_u = \{r_0 : \mathit{styleReq{=}clam}, r_1 : \mathit{pHDSPA} = \mathit{true}\}$, phones of configurations $conf_2$ and $conf_3$ meet the customer requirements (see Tables 2.3 and 2.4). Since $conf_3$ is most similar to the user configuration the corresponding phone p_3 is ranked first. p_1 cannot be recommended because it does not match the customer requirements.

Table 2.4.: Example: valid configurations from previous sessions.

feature/configuration	**$conf_1$**	**$conf_2$**	**$conf_3$**	**$conf_u$**
f_1=pModel	*p1*	*p2*	*p3*	
f_2=pStyle	*bar*	-	*clam*	*clam*
f_3=pHDSPA	*false*	*true*	*true*	*true*
f_4=pGPS	-	*true*	*false*	
f_5=pPrice	*69*	*149*	*99*	

Another approach to calculate personalized item rankings and to take into account primacy effects in the presentation of result sets has been introduced in (Felfernig et al., 2008). The authors of (Felfernig et al., 2008) utilize the concepts of Multi-Attribute Utility Theory (MAUT) (von Winterfeldt and Edwards, 1986), and derive the importance of interest dimensions from customer requirements. Product alternatives are then evaluated according to these dimensions.

Lai et al. (Lai et al., 2006) propose a content-based relevance ranking strategy for web-based video search which takes into consideration *rich content*, such as semantic content descriptions and speech within the video. The results of their evaluation show that their approach achieves improved ranking performance.

Chapter 3

Related Work

Parts of the contents of this chapter have been published in (Mandl et al., 2009, 2011a, 2012c).

This chapter introduces related work in the field of consumer decision making as well as in the field of evaluation techniques for recommender systems. Both topics play a central role in this thesis. Section 3.1 presents different models of consumer decision making. Since user interface elements can have a major impact on the final outcome of users' decision process (Asch, 1949; Payne, 1976; Lussier and Olshavsky, 1979; Tversky and Kahneman, 1981; Bettman et al., 1991), the knowledge about different decision-making strategies has to be considered in the user interface design. Section 3.2 deals with selected theories from decision psychology and their relevance for intelligent selling applications.

Since this thesis aims at analyzing different factors of intelligent selling environments that may have an effect on users' satisfaction with the shopping process, different evaluation frameworks are needed to examine the corresponding evaluation issues. In the context of this thesis, we will present empirical experiments, that aimed at analyzing the quality of intelligent selling applications (see Section 4), as well as the influence of recommendations on human decision making aspects (see Section 5). Section 3.3 presents an overview of established strategies to evaluate recommendation technologies in different contexts.

3.1. Consumer Decision Making in Interactive Selling Environments

In contrast to customers of bricks and mortar stores, users of online selling environments are not supported by human sales experts. In such situations intelligent selling environments, such as recommender applications, help to personalize the shopping experience by identifying the products and/or services that fit the user's wishes and needs. In order to better assist consumers in buying decisions, and to increase the user satisfaction with the recommendation process, it is not only important to implement algorithms that provide good recommendations, but also to establish an adequate format for presenting the recommendations.

Research on consumer decision making has shown that consumers are influenced by the format of the information presented and as a consequence use different decision-making strategies in different contexts (see, for example, (Asch, 1949; Payne, 1976; Lussier and Olshavsky, 1979; Tversky and Kahneman, 1981; Bettman et al., 1991)). These decision strategies are explained in different models of human decision making. In the following section we provide an overview of selected models and discuss their importance for recommender system development.

Models of Consumer Decision Making

In the 18th century, economics began to explore knowledge about consumer decision-making processes. Nicholas Bernoulli developed the first consumer decision making theory. The basic assumption of this theory was that consumers make buying decisions based on the expected results of their purchases (Richarme, 2004). According to Bernoulli, consumers select that option which will provide maximum satisfaction. *Bernoulli's Utility theory* was later extended by John von Neumann and Oskar Morgenstern (Morgenstern and Neumann, 1944). In their *Expected Utility Theory* they introduced four axioms which define a *rational decision maker* (Morgenstern and Neumann, 1944):

- *Completeness*: a decision maker has well defined preferences,
- *Transitivity*: preferences are consistent,
- *Independence*: preferences hold independently of the outcome, and
- *Continuity*: given a middle option there is a "tipping point" between being better than and worse than this reference option.

Von Neumann and Morgenstern stated that the preferences of a rational decision maker are consistent and can be represented by a utility function. In the 1950s, Herbert Simon developed an alternative model of consumer decision making: "*Satisficing*" (Simon, 1956). This model takes into account the fact that consumers stop the decision making process when they have found a product they consider as good enough, rather than to identify the best solution. Simon argued that the idea of the rational decision maker (Morgenstern and Neumann, 1944) requires cognitive information processing skills that people do not possess (Simon, 1956). According to Simon, decision makers lack the ability and resources to arrive at the optimal solution and typically operate within a *bounded rationality* (Simon, 1956).

Since the 1960s various consumer decision-making models have been developed (Erasmus et al., 2001). The large number of different decision making models highlights the complexity of consumer choices. In the following we will discuss selected models with a special relevance in the context of recommender applications.

Traditional Economic Models

Based on the rationality aspects of Utility Theory (Morgenstern and Neumann, 1944), traditional economic models are assuming that all users are able to take decisions that are optimal, and that

Figure 3.1.: Traditional five step classification of the consumer decision process (Schiffman and Kanuk, 1997; Solomon, 2002) .

have been derived on the basis of rational and formal processes. Consumers are considered as rational decision makers who seek to maximize utility. Due to their wide ranging scope, these models are often labeled as "*grand models*" (Kassarjian, 1982). Among the best known are the *Nicosia Model* (Nicosia, 1966), the *Howard-Sheth-Model* (Howard and Sheth, 1969), and the *Engel, Kollat & Blackwell-Model* (Engel and Blackwell, 1986). In these models the consumer decision making process is reflected in terms of the traditional five step classification (Schiffman and Kanuk, 1997; Solomon, 2002) (see Figure 3.1).

An assumption of economic models is, that preferences remain stable, i.e., are not adapted within the scope of a decision process. However, it is a fact that preferences can be extremely unstable, for example, a customer who buys a car first sets the upper limit for the overall price to €20.000. This does not mean that the upper limit is strict, since the customer could change his/her mind and set the upper limit for the price to €25.000, simply because he/she detected additional technical features for which he/she is willing to pay the higher price, for example, high-quality headlights, park-distance control, satellite navigation, and rain-sensor for the windscreen wipers. Solely on the basis of this simple example we immediately see that preferences could change over time, i.e., are not stable within the scope of a recommendation process. This insight led to the development of new decision models – see, for example, (Payne et al., 1993; Bettman et al., 1998). The most important ones will be discussed in the following.

Effort Accuracy Framework

Following this model developed by Payne, Bettman, and Johnson (Payne et al., 1993), users are taking into account cost-benefit aspects. This basic assumption is similar to Simon's *satisficing theory* (Simon, 1956). Simon coined the term"*bounded rationality*" (Simon, 1956) to describe the "*human inability to logically evaluate decisions under conditions of uncertainty* (Chira et al., 2008). Caused by this inability the decision-maker tries to find a satisfactory solution rather than the optimal one – a strategy which Simon called "*Satisficing*" (Simon, 1956). A decision process is now characterized by a trade-off between the effort to take a decision and the expected quality of the decision.

The effort-accuracy framework is based on the fact that users (customers) show an adaptive decision behavior and select from an available set of different decision heuristics depending on the current situation (Payne et al., 1993). Criteria for the selection of a certain heuristic are on the one hand the needed decision quality, and on the other hand the (cognitive) efforts needed for successfully completing the decision task. This framework clearly differs from the above mentioned economic models of decision making. In those models, optimality plays a dominant role, and the efforts related to successfully completing a decision task are neglected. However, especially the effort for completing a decision task has to be taken into account as an important factor that determines whether the user is willing to apply the intelligent selling application or chooses a different provider.

Construction of Preferences

The concept of preference construction in human choice has been developed by Bettman, Luce, and Payne (Bettman et al., 1998). The basic idea of preference construction is that users tend to identify their preferences within the scope of a decision making process, but only in rare cases are able to state their preferences before the beginning of the process. Consequently, decision support should be more focused on *constructing* a consistent set of preferences than *eliciting* preferences from the user. The latter is still the predominantly supported type of decision process in many existing recommender applications (Bridge et al., 2005; Brusilovsky et al., 2007).

Since user preferences are constructed within the scope of a recommendation session, the design of the user interface can have a major impact on the final outcome of the decision process. For example, Mandel and Johnson (Mandel and Johnson, 1998) reported that web page design can have an impact on users' preferences by influencing attribute importance. In a user experiment they demonstrated that different background designs may influence the preference construction process by raising the awareness, and therefore the importance of the primed attributes (Mandel and Johnson, 1998).

3.2. The Impact of Recommender Systems on Consumer Decision Making

In a decision making situation, customers are often overwhelmed by the complexity of available alternatives (Huffman and Kahn, 1998). Research has shown that recommender systems can help to reduce the complexity of online searches as well as the information overload (Xiao and Benbasat, 2007). In order to improve the applicability of intelligent selling applications, we must integrate recommendation technologies with deep knowledge about human decision making. Such an integration can help to improve the perceived quality of the intelligent selling application for the user, as well as the predictability of decision outcomes (see the discussions in the following sections). In the following we will review selected theories from decision psychology with respect to their potential impact on preference construction processes. These theories have already shown to be of relevance for recommender applications – an overview is provided in Table 3.1.

3.2.1. Decoy Effects

Decoy products are items which are added to existing item sets with the purpose of increasing the attraction of defined target items (Felfernig et al., 2008)[1]. Depending on the similarity and

[1]Note that we use the *robot product domain* in the following examples.

Table 3.1.: Selected theories of decision psychology.

theory	explanation
decoy effects	the inclusion of decoy products to a result set can significantly change the outcome of the decision process (Huber et al., 1982; Simonson and Tversky, 1992; Yoon and Simonson, 2008; Teppan and Felfernig, 2009c).
serial position effects	information units at the beginning and the end of a list are analyzed and recalled significantly more often than those in the middle of a list - this has an impact on a user's selection behavior (Murphy et al., 2006; Felfernig et al., 2007, 2008).
framing	the way in which we describe a certain decision alternative can have a significant impact on the final decision (Marteau, 1989; Tversky and Kahneman, 1981, 1986; Levin et al., 1998; Jannach et al., 2011).
defaults	pre-selected decision alternatives have the potential to significantly change the outcome of a decision process (Samuelson and Zeckhauser, 1988; Ritov and Baron, 1992; Huffman and Kahn, 1998; Herrmann et al., 2007; Felfernig et al., 2010a,b).
anchoring	decision makers are influenced by random and uninformative starting points ("*anchors*") set for item attributes (Chapman and Johnson, 1999; Viappiani et al., 2007; Adomavicius et al., 2011).
social navigations	the user's navigation is driven by the actions or behaviors from other users (Dourish and Chalmers, 1994; Chalmers et al., 2004; Papagelis et al., 2008).

the respective inferiority or superiority of decoy items when compared to the other items in the choice set, decoys can significantly change the perception of the items and the choice set as a whole (Felfernig et al., 2008). In this context, the inferiority respectively superiority of items is measured by simply comparing the underlying properties of items with regard to their distance to the optimal value. For example, *robot X* dominates *robot Y* in the dimensions *price* and *reliability* if it has both, a lower price and a higher reliability. As a consequence, the inclusion of such decoy products can significantly influence the outcome of decision processes, and therefore has to be taken into account when implementing intelligent selling applications. The phenomenon that users change their selection behavior in the presence of additional inferior items is denoted as *decoy effect*. Decoy effects are of special relevance in the result presentation phase within a recommendation process and have been intensively investigated in different application contexts, see, for example, (Huber et al., 1982; Simonson and Tversky, 1992; Yoon and Simonson, 2008; Felfernig et al., 2008; Teppan and Felfernig, 2009b,c).

In the following subsections we will discuss different types of decoy effects and explain how these effects can influence the outcome of decision processes. Note that the existence of decoy effects provides strong evidence against the validity of traditional economic models of choice (Nicosia, 1966; Howard and Sheth, 1969; Engel and Blackwell, 1986), that suppose rational and optimal strategies in human decision making.

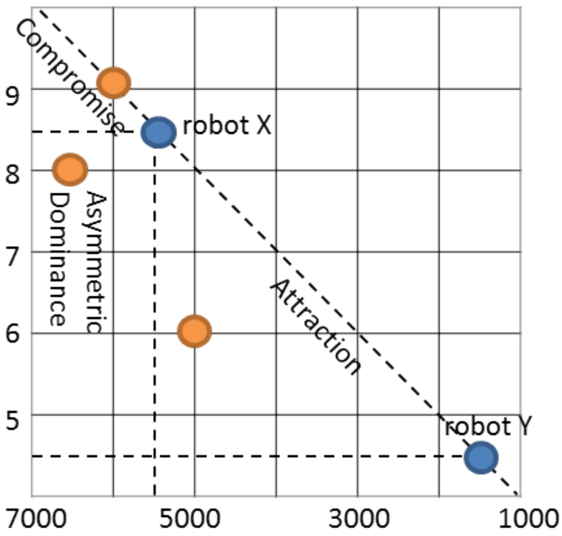

Figure 3.2.: Example decoy items in a two-dimensional product landscape – the inclusion of decoy products can significantly change the perception of items in an existing item set (Huber et al., 1982; Simonson, 1989).

Table 3.2.: Compromise effect – adding an extreme alternative (*robot D*) to a choice set will result in people favoring the "middle" choice (*robot X*) (Simonson and Tversky, 1992).

product (robot)	X 	Y 	D 
price [€ 0..10.000]	5.500	1.500	6.000
reliability [0..10]	8.5	4.5	9

Compromise Effects

Compromise effects (Simonson, 1989) denote one specific archetype of decoy effects which is shown in Table 3.2. The research of Simonson and Tversky (Simonson and Tversky, 1992) has demonstrated that adding an extreme alternative to a choice set will result in people favoring the "middle" choice, where attribute values are positioned between the values of the other alternatives. For example, the attractiveness of robot X compared to robot Y can be increased by adding robot D to the set of alternatives (see Table 3.2 and Figure 3.2). By the insertion of decoy robot D the comparison focus of the user is set to XD, since D is more similar to X than to Y (*similarity effect*). Since robot D (the decoy) is more extreme in both dimensions than robot X, as robot D has a higher reliability and is more expensive, robot X is mentally established as a (probably good) compromise between robot Y and robot D.

More formally, we can explain decoy effects as follows. Under the assumption that the probability of selection for item X out of the item set {X,Y} is equal to the probability of selection of Y out of {X,Y}, i.e., P(X,{X,Y}) = P(Y,{X,Y}), the addition of D causes a preference shift to X, i.e., P(Y,{X,Y,D}) < P(X,{X,Y,D}).

Table 3.3.: Asymmetric dominance effect (Huber et al., 1982) – decoy product *robot D* is completely dominated by *robot X* and only partially dominated by *robot Y*.

product (robot)	X 	Y 	D 
price [€ 0..10.000]	5.500	1.500	6.500
reliability [0..10]	8.5	4.5	8

Table 3.4.: Attraction effect (Simonson, 1989) – decoy product *robot D* is positioned between the target *robot X* and the competitor product *robot Y*.

product (robot)	X 	Y 	D 
price [€ 0..10.000]	5.500	1.500	5.000
reliability [0..10]	8.5	4.5	6

Asymmetric Dominance Effects

The second archetype of decoy effect is called asymmetric dominance (Huber et al., 1982). An example is depicted in Table 3.3. In this scenario, robot X dominates robot D in both attributes (price and reliability), whereas robot Y dominates robot D in only one dimension (the price – compare Figure 3.2). The addition of robot D to the set of {X,Y} can help to increase the share of X. In this context the comparison focus is set to XD (D is more *similar* to X than Y), which makes X the clear winner in the competition, i.e., P(Y,{X,Y,D}) < P(X,{X,Y,D}).

Attraction Effects

The third archetype of decoy effects is called attraction effect (Simonson, 1989). In this context, X appears to be only a little bit more expensive, and simultaneously has a significantly higher reliability compared to robot D (*tradeoff-contrast* – see Table 3.4 and Figure 3.2). In this scenario the inclusion of D can trigger an increased probability of selection for robot X, since X appears to be more attractive than D, i.e., P(Y,{X,Y,D}) < P(X,{X,Y,D}). The attraction effect moves the comparison focus to the combination of items XD since D is more similar to X than to Y (*similarity effect*). Note that both, compromise effects and attraction effects, are based on the ideas of tradeoff-contrast and similarity. The difference lies in the positioning of decoy items. In the case of the compromise effect, decoy products are representing extreme solutions (see Table 3.2), whereas in the case of the attraction effect decoy products are positioned between the target and the competitor product (see Table 3.4).

The Role of Decoy Effects in Recommendation Scenarios

If decoy items are added to a result set, this can change the selection probability for items that were included in the original result set. The occurrence of decoy effects has been shown in a number of empirical studies in application domains such as financial services, e-tourism, and even software agents (see, for example, (Huber et al., 1982; Simonson and Tversky, 1992; Teppan and Felfernig, 2009a)). The major possibilities of exploiting decoy effects in knowledge-based recommendation scenarios are the following:

- *Increased selection probability for target items*: as already mentioned, adding additional decoy items to a result set can cause an increased share of target items (Teppan and Felfernig, 2009b) (in our example denoted as robot X). This scenario definitely has ethical aspects to be dealt with since companies can potentially try to apply decoy effects for selling products that are maybe suboptimal for the customer.
- *Increased decision confidence*: beside an increase of the share of the target product, decoy effects can be exploited for increasing the decision confidence of a user (Teppan and Felfernig, 2009c). In this context, decoy effects can be exploited for resolving cognitive dilemmas which occur when a user is unsure about which alternative to choose from a given set of nearly equivalent alternatives.
- *Increased willingness to buy*: from empirical studies we know that a user's level of trust (confidence) in recommendations is directly correlated with the willingness to buy, i.e., increasing the level of trust directly means that the purchase probability can be increased as well (Chen and Pu, 2005).

The important question to be answered now is how to predict decoy effects within the scope of a recommendation scenario. Predicting the selection of products contained in the set of possible product alternatives (the consideration set CSet) requires the calculation of dominance relationships between the items contained in a result set. Exactly for this calculation different models have been developed (see, e.g., (Teppan and Felfernig, 2009a; Roe et al., 2001)) – the outcomes of each of these models are dominance relationships between the items in CSet. The calculation of such dominance relationships can be based on Formula 3.1, which is a simplified version of the approach introduced in (Teppan and Felfernig, 2009a). This formula allows the calculation of dominance relationships between different products in a consideration set, i.e., *d(u, CSet)* denotes the dominance of product u compared to all other items in CSet.

$$d(u,CSet) = \sum_{v \in CSet-\{u\}} \sum_{a \in properties} \sqrt{\frac{diff(u_a,v_a)}{a_{max}-a_{min}}} * sign(u_a,v_a) \tag{3.1}$$

$$diff(u_a,v_a) = |u_a - v_a| \tag{3.2}$$

$$sign(u_a,v_a) = \begin{cases} 1 & if\ u_a \leq v_a,\ a = price \\ 1 & if\ u_a \geq v_a,\ a = reliability \\ -1 & otherwise \end{cases} \tag{3.3}$$

Applying Formula 3.1 to the product set {X,Y,D} depicted in Table 3.3 results in the dominance values that are depicted in Table 3.5. For example, product v_1(Y) has a better price than product u (X; the target item) – the corresponding dominance value is -0.89, i.e., product u is inferior

Table 3.5.: Dominance values for u ∈ CSet for Table 3.3.

	u	$\mathbf{v_1}$	$\mathbf{v_2}$	**Sum**	**d(u,CSet)**
	X	**Y**	**D**		d(X,{X,Y,D})
price		-0.89	0.45	-0.44	
reliability		1	0.35	1.35	
					0.91
	Y	**X**	**D**		d(Y,{X,Y,D})
price		0.89	1.0	1.89	
reliability		-1	-0.93	-1.93	
					0.04
	D	**X**	**Y**		d(D,{X,Y,D})
price		-0.45	-1.0	-1.45	
reliability		-0.35	0.94	0.59	
					-0.86

regarding the attribute price. The sum of the attribute-wise calculated dominance values, i.e., d(u,CSet), provides an estimation of how dominant item u appears to be in the set of candidate items CSet. The values in Table 3.5 clearly show a dominance of item X over the items Y and D. The dominance relationships between items in a result set can be directly used by a corresponding configuration algorithm to calculate a choice set, such that the attractiveness of one option is increased (Teppan and Felfernig, 2009a).

3.2.2. Serial Position Effects

In 1949 Solomon Asch conducted an experiment on formations of personality impression (Asch, 1949). The results of this study showed that presenting adjectives describing a person in sequence, the same words could result in very different ratings of that person depending on the order in which the words were presented. A person described as "*intelligent, industrious, impulsive, critical, stubborn, envious*" was rated more positive by the participants, than a person described as "*envious, stubborn, critical, impulsive, industrious, intelligent*". This phenomenon is known as *primacy effect* and is explained through a memory advantage that early items in a list have (Crowder, 1976).

Murphy, Hofacker and Mizerski (Murphy et al., 2006) explored the importance of an item's list position in an online environment. In their experiment they manipulated the serial position of links on the website of a popular restaurant. The results of this study showed that visitors tended to click the link on first position most frequently. But there was also an increased tendency to click on the links at the end of the list. This is known as *recency effect*. The results go along with the findings of Hermann Ebbinghaus who first documented the *serial position effect* (Ebbinghaus and Clara, 1885) which describes the relationship between recall probability of an item and its position in a list (see Figure 3.3).

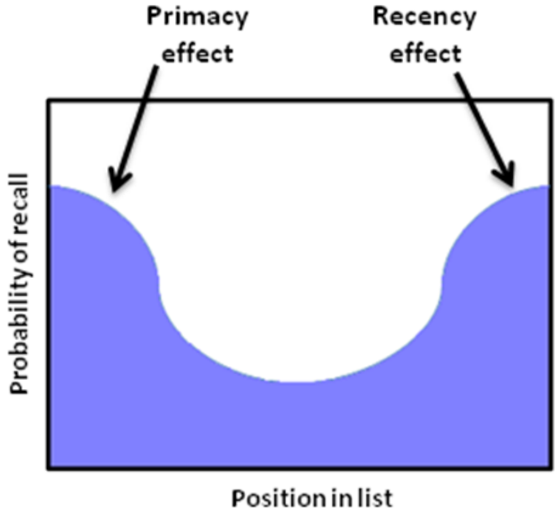

Figure 3.3.: Serial position effect – a term coined by Hermann Ebbinghaus (Ebbinghaus and Clara, 1885) that refers to the finding that items at the beginning and at the end of a list are more accurately recalled than those items positioned in the middle of a list.

The Role of Serial Position Effects in Recommendation Scenarios

Felfernig et al. (Felfernig et al., 2007) investigated serial position effects in knowledge-based recommendation scenarios, especially in the context of presenting product features. They conducted a study where participants were asked to choose a tent out of a set of tents (the tent, a participant would buy most likely in a real purchase situation). The position of product attributes used to describe the tents was varied. The results of this study showed significant changes in the product selection behavior that have been triggered by changed product attributes orderings.

The results of the studies (Asch, 1949; Murphy et al., 2006; Felfernig et al., 2007) illustrate the importance of an item's position in an ordered list. E-Commerce retailers utilize product recommendations as a targeted marketing tool to personalize the shopping experience for each customer (McSherry, 2004). In this context serial position effects play an important role in the ordering of products on result pages. As a consequence, recommender applications must be aware of the fact that different item rankings can trigger a different item selection behavior, and as well can increase or reduce a user's decision making effort.

Based on the results of their research, Murphy, Hofacker and Mizerski (Murphy et al., 2006) suggest to place the most important item on the first position, and to place another important item on the last position of a list (the process should be continued with the order of importance). An approach to calculate personalized item rankings, and to take into account primacy/recency effects in the presentation of result sets, has been introduced in (Felfernig et al., 2008). The authors of (Felfernig et al., 2008) utilize the concepts of Multi-Attribute Utility Theory (MAUT) (von Winterfeldt and Edwards, 1986) and derive the importance of interest dimensions from customer requirements. Product alternatives are then evaluated according to these dimensions.

We now want to discuss the concepts presented in (Felfernig et al., 2008) in more detail on

Table 3.6.: Scoring rules for product attribute *price*.

price	**economy**	**quality**
≤ 2000	10	3
$>2000, \leq 5000$	6	5
$>5000, \leq 8000$	4	7
$>8000, \leq 10000$	2	10

Table 3.7.: Scoring rules for product attribute *reliability*.

reliability	**economy**	**quality**
≤ 3	10	4
$>3, \leq 6$	6	7
$>6, \leq 10$	4	10

the basis of the following example. Let us assume that *economy* and *quality* have been defined as example interest dimensions for the robot product domain introduced in Section 3.2.1. In Tables 3.6 and 3.7 example scoring rules are defined, that describe the relationships between the robot attributes (*price* and *reliability*) and the corresponding interest dimensions (*economy* and *quality*). For example, Table 3.6 shows that an expensive robot has a low perceived value for interest dimension economy, and a high perceived value for interest dimension quality. Table 3.7 shows that a robot with low reliability has a high valence in interest dimension economy, and a low valence in interest dimension quality.

Let's assume a concrete customer (*customer 1*) with a higher interest in the dimension economy (importance of 0.7) compared to the dimension quality (importance of 0.3 – assuming that the sum of importance values is 1). A personalized product ranking can be calculated on the basis of Formula 3.4. In this formula *contribution(r,i)* defines the contribution of product r to the interest dimension i, and *interest(i)* shows the degree to which a specific customer is interested in dimension i.

$$productutility(r) = \sum_{i=1}^{n} contribution(r,i) * interest(i) \tag{3.4}$$

Applying the scoring rules of Tables 3.6 and 3.7 to the robots of Table 3.4 results in the item ranking shown in Table 3.8. This customer-specific ranking of products can now be used to identify an ordering of robots that takes into account primacy/recency effects. For this purpose a utility value has to be assigned to each list position. Based on the approach of Murphy, Hofacker and Mizerski (Murphy et al., 2006), the first and last position will get a higher utility value as the middle position (see Table 3.9).

In order to calculate a customer-specific product ordering taking into account primacy/recency effects, Formula 3.5 can be applied. In our simple example we have (3!)=6 possible combinations

Table 3.8.: Overall utilities of robots in Table 3.4.

robot	economy	quality	overall utility
X	4+4=**8**	7+10=**17**	8*0.7+17*0.3=**10.7**
Y	10+6=**16**	3+7=**10**	16*0.7+10*0.3=**14.2**
D	6+6=**12**	5+7=**12**	12*0.7+12*0.3=**12**

Table 3.9.: Utilities of product positions.

robot position	1	2	3
utility of position	3	1	2

of product sequences. We can use this formula to identify a corresponding product ordering.

$$orderutility_{r1...n} = \sum_{i=1}^{n} productutility(r_i) * positionutility(i) \tag{3.5}$$

In Formula 3.5, *productutility(r_i)* specifies the utility of a specific product r_i for a customer (in our case *customer 1* – see Table 3.8), and *positionutility(i)* defines the utility of a specific position i in the result list (see Table 3.9). As shown in Table 3.10, the listing with the highest utility is the one where robot Y is positioned at the first position (the most interesting option for the customer, since it represents a cheap solution), and robot D is placed on the last position.

3.2.3. Framing

Framing effects occur when one and the same decision alternative is presented in different variants (Tversky and Kahneman, 1986). Tversky and Kahnemann presented a series of studies where they confronted participants with choice problems using variations in the framing of decision outcomes (Tversky and Kahneman, 1981). They reported that "*seemingly inconsequential changes in the formulation of choice problems caused significant shifts of preference*" (Tversky and Kahneman, 1981). An explanation for such choice reversals is given by *prospect theory* developed by Kahnemann and Tversky (Kahneman and Tversky, 1979). In this theory a value function is introduced for explaining decision making under risk, where negative outcomes have a higher impact compared to the positive ones (see Figure 3.4).

Levin, Schneider and Gaeth (Levin et al., 1998) introduced three dimensions in which framing

Table 3.10.: Overall utilities of possible robot sequences.

robot sequence (r_1-r_2-r_3)	overall utility
X-Y-D	10.7*3+14.2*1+12*2=**70.2**
X-D-Y	10.7*3+12*1+14.2*2=**72.5**
Y-X-D	14.2*3+10.7*1+12*2=**77.3**
Y-D-X	14.2*3+12*1+10.7*2=**76.0**
D-X-Y	12*3+10.7*1+14.2*2=**75.1**
D-Y-X	12*3+14.2*1+10.7*2=**71.6**

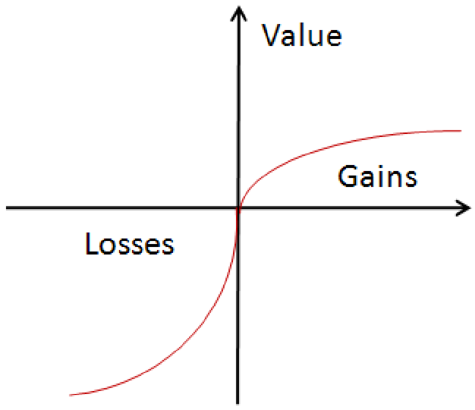

Figure 3.4.: Prospect theory's value function (Kahneman and Tversky, 1979) – evaluating losses and gains.

Table 3.11.: Framing types defined by Levin, Schneider and Gaeth (Levin et al., 1998).

Framing type	**Information framed**
risky choice framing	risk level of options
attribute framing	one attribute of an object
goal framing	goal of an action/behavior

manipulations can differ: what is framed, what is affected, and how the effect is measured. Based on this distinction, the authors identified three different types of framing effects (see Table 3.11).

The form of framing introduced by Tversky and Kahneman (Tversky and Kahneman, 1981) is categorized as *risky choice framing*, based on the fact that given options in a choice problem differ in their risk level. The same decision problem can be described as a choice between one sure and one risky loss (loss frame) as well as a choice between one sure and one risky gain (gain frame) (Pinón and Gärling, 2004). Research has identified a general preference shift from more risky choices (with more gains) to choices that avoid losses (Levin et al., 1998). The most famous example of this framing type is the *Asian disease problem* (Tversky and Kahneman, 1981). Participants of a study were confronted with the situation that 600 individuals are infected with a deadly disease. Two options with a different risk level to combat the disease where presented. Participants were divided into two groups. One group had to choose between options described in terms of lives saved (*positive frame*), and the other group had to choose between options based on a description of lives lost (*negative frame*) (Tversky and Kahneman, 1981)). The results of this study showed that participants preferred the risk-free alternative when the problem was framed positively ("*200 people will be saved*" is preferred to the option "*600 people will be saved with 1/3 probability and all people will die with a 2/3 probability*" (Tversky and Kahneman, 1981)). In contrast, participants in the negative frame tend to choose the risky alternative ("*1/3 probability that no one will die and 2/3 probability that all people will die*" is preferred to "*400 people will die*" (Tversky and Kahneman, 1981)).

The second framing type is called *attribute framing*. Here only one attribute or characteristic of an object is framed. For example, in a study conducted by Levin and Gaeth (Levin and Gaeth,

1988) a beef product labeled as "*75% lean*" was evaluated more favorable than a beef product labeled as "*25% fat*".

The third type of framing termed by Levin, Schneider and Gaeth (Levin et al., 1998) is *goal framing* where the information which is framed is the goal of an action or behavior. Ganzach and Schul (Ganzach and Schul, 1995) reported three experiments where the goal of the decision was framed either as acceptance or rejection. The participants were asked to choose one out of two candidates which they would accept/reject. The results of the experiments showed that goal framing influences the extent of processing of positive vs. negative information (Ganzach and Schul, 1995). If the decision problem was framed as an *acceptance decision*, participants were more likely to rely on positive information, whereas participants confronted with a *rejection decision* focused on the evaluation of negative information. An explanation for this phenomenon is given by the *confirmation bias*, a term coined by Peter Wason (Wason, 1960). The confirmation bias describes a tendency to make frame-compatible features more important in a choice problem.

Price framing

Another occurrence of framing is *price framing*, where the granularity of the presented price information is framed, which means that the price information is either presented in one attribute or distributed over several attributes (Bertini et al., 2006). Since only one attribute of the object is framed, price framing can be seen as a subclass of attribute framing in the categorization of Levin, Schneider and Gaeth (Levin et al., 1998). Bertini and Wathieu (Bertini et al., 2006) conducted a series of studies to investigate the effect of this framing type. They found that the price format influences users in the evaluation of product attributes. If the price information is provided for different subparts of a product, users tend to focus on evaluating those subparts with corresponding price information. If the product price on the contrary is represented by one attribute, users focus on evaluating other technical attributes (Bertini et al., 2006).

The Role of Framing Effects in Recommendation Scenarios

The framing of options or characteristics of options in a choice set can result in a shift of selection probability for items (Marteau, 1989; Tversky and Kahneman, 1981, 1986; Levin et al., 1998; Bertini et al., 2006). The implications of the above-mentioned framing types on user decision behavior are the following:

- *Risky choice framing*: Levin, Schneider and Gaeth (Levin et al., 1998) pointed out that in risky choice framing a positive frame typically enhances risk aversion. For example, a fund with a 95% probability of no loss is interpreted as a better solution compared to the same product described with a 5% probability of loss. In the context of recommender systems this framing type plays an important role in the presentation of product alternatives, as well as in the presentation of repair proposals for inconsistent requirements, since the way in which those alternatives are presented can significantly change a user's selection behavior.
- *Attribute framing*: a positive framing of an attribute of options leads to a more positive judgment of the options compared to negative frames. For example, Marteau (Marteau,

1989) demonstrated that people were more likely to attend medical procedures described by their survival rate rather than their mortality rate. In this context as well, attribute framing has to be taken into account when designing result (product) presentations in a recommender application.

- *Goal framing*: in goal framing a negatively framed message is more likely to lead to a negative response than a comparable positively framed message, as results of the research of Ganzach and Schul (Ganzach and Schul, 1995) shows. In recommender systems this fact is relevant in the requirements specification phase. For example, if the interface is requesting decisions regarding the inclusion of items, users will rather take into account positive properties and vice-versa if items should be excluded, uses will rather take into account negative (less preferred) item properties.
- *Price framing*: In the context of recommender systems, this framing type has to be considered in the product presentation since price framing can lead to a shift of a user's evaluation focus from quality attributes (for example, technical attributes of a digital camera) to price attributes and thus could significantly change the outcome of the decision process. This effect is, for example, exploited by discount airlines which typically give a separate listing for air fares and fees.

3.2.4. Defaults

Internet users are facing an ever increasing amount of product information. For example, at Pandora[2], a popular personalized internet radio service, the characteristics of a song are specified by 400 attributes. Since humans capacity to process information is limited, such a large amount of information can lead to an information overload (Häubl and Murray, 2003; Fitzsimons and Lehmann, 2004; Lee and Lee, 2004).

Early research in the field of consumer behavior indicated that confronting the consumer with too much information can result in a decreased quality of decision performance (see, for example, (Jacoby et al., 1974)). These traditional approaches studied the information overload effect by varying the number of alternatives in the choice set and the number of product attributes. The results of these studies showed an inverted-U-shaped relationship between the amount of information and decision quality (measured by the consumers' ability to make correct decisions among many different products (Lee and Lee, 2004)). Later research resulted in contrary results (see, for example, (Russo, 1974; Summers, 1974)). Russo (Russo, 1974) reanalyzed the data of the research of Jacoby, Speller and Kohn (Jacoby et al., 1974) and found no overload effect. Contrary to the original conclusions, Russo's results suggested that more information can help the consumer in making choices (Russo, 1974). Consequently both aspects seem to be important, i.e., the user must not be overloaded with too many (often non-understandable technical) aspects, but on the other hand must have available all the necessary information relevant for taking a decision. Huffman and Kahn (Huffman and Kahn, 1998) state that "*the key to customer satisfaction with the entire shopping interaction is to ensure that the customer is equipped to handle the variety*". A way to do this is given by Salgado (Salgado, 2006), who suggest to provide information about how the

[2] `http://www.pandora.com`

alternatives in the recommendation set were selected "*in a way that is convincing to customers*" (Salgado, 2006). Another possibility to support users in the specification of their requirements is to provide defaults (Felfernig et al., 2010a,b). Defaults in recommender systems are preselected options used to express personalized feature recommendations. For example, if the user is interested in reading emails with the phone, the recommended phone should support web access. Thus defaults are a means to help the user identifying meaningful alternatives that are compatible with their current preferences.

The Role of Defaults in Recommendation Scenarios

Especially for knowledge-based recommender applications defaults play a very important role since users tend to accept preset values compared to other alternatives (Samuelson and Zeckhauser, 1988; Ritov and Baron, 1992). An explanation model for this phenomenon is that users often tend to favor the status quo over alternatives often of equal attractiveness. Samuel and Zeckhauser (Samuelson and Zeckhauser, 1988) have shown this effect, known as *status quo bias*, in a series of experiments. Kahnemann, Knetsch and Thaler (Kahneman et al., 1991) argue that the status quo bias can be explained by a notion of *loss aversion*. They explain that the status quo serves as a reference point, and alternative options are evaluated in terms of gains and losses relative to the reference point. Felfernig et al. (Felfernig et al., 2010a,b) conducted a study to investigate the impact of personalized feature recommendations in a knowledge-based recommendation process (see, for example, Figures 4.4, 4.5,4.6,4.7). The Nearest Neighbors and Naïve Bayes Voter algorithms were used for the calculation of defaults (for more details see Section 4.2). The results of this research indicate that supporting users with personalized defaults can lead to a higher satisfaction with the recommendation process.

A major risk of defaults is that they could be exploited for misleading users and making them to choose options that are not really needed to fulfill their requirements. Ritov and Barron (Ritov and Baron, 1992) suggest counteracting the status-quo bias by presenting the options in such a way, that keeping as well as changing the status quo needs user input. They argue that "*when both keeping and changing the status quo require action, people will be less inclined to err by favoring the status quo when it is worse*" (Ritov and Baron, 1992). In Section 5.2 we will present the results of an empirical study, that aimed at analyzing such an interface type suggested by Ritov and Baron.

3.2.5. Anchoring

The anchoring effect is a cognitive bias that explains that decision makers are influenced by initial starting points ("*anchors*") (Tversky and Kahneman, 1974; Chapman and Johnson, 1999). This results in a tendency to under-adjust the anchoring attribute, so that the final decision is very close to the starting point (Chapman and Johnson, 1999). Tversky and Kahneman (Tversky and Kahneman, 1974) demonstrated this effect in a series of studies. They reported for example, that when asked to guess the percentage of African nations which are members of the United Nations, people who were first asked "*Was it more or less than 10%?*" guessed lower values (25% on average) than those who had been asked if it was more or less than 65% (45% on average).

Research has shown that anchoring affects a broad range of judgments, for example, answers to knowledge questions, monetary evaluations, and social judgments (Cervone and Peake, 1986; Jacowitz and Kahneman, 1995; Chapman and Johnson, 1999). Chapman and Johnson demonstrated through a series of experiments that the anchoring effect occurs even when the anchor is obviously random and irrelevant (Chapman and Johnson, 1999). Their results further suggest that anchoring does not only occur when anchoring attributes represent merely a number, but also when the attribute has multiple features (Chapman and Johnson, 1999). Jacowitz and Kahnemann (Jacowitz and Kahneman, 1995) introduce a technique for the quantitative study of anchoring effects in estimation tasks. Their method allows to measure the size of the anchoring effect as well as to compare results across different problems. They demonstrate that asking subjects questions like "*Is the quantity higher or lower than X?*" results in a surprisingly large anchoring effect. Their results also indicate that there exists an inverse relationship between the anchoring effect and the subjects' confidence in their judgments (Jacowitz and Kahneman, 1995). The authors of (Cervone and Peake, 1986) conducted two experiments that aimed at analyzing the anchoring effect in a judgmental process. The results of these experiments suggest that the anchoring effect strongly affects self-efficacy judgments (Cervone and Peake, 1986).

The Role of Anchoring in Recommendation Scenarios

Viappiani et al. (Viappiani et al., 2007) claim that the example-critiquing strategy used by critiquing based recommender systems can lead to an anchoring effect. They suggest to display alternative examples, in addition to the recommendation candidates, to enable the user to refocus the search in another direction (Viappiani et al., 2007).

(Adomavicius et al., 2011) explored whether predictions generated by recommender systems can cause an anchoring effect. The results of three experiments indicate that system recommendations can have a significant impact on the preference ratings of consumers. Adomavicius et al. (Adomavicius et al., 2011) point out the potential danger of these findings – this biasing effect can lead to unscrupulous business practices since online retailers could exploit the anchoring bias to increase the sales of, for example, expensive products.

3.2.6. Social Navigation

The term *social navigation* was coined by Dourish and Chalmers and describes that "*movement from one item to another is provoked as an artifact of the activity of another or a group of others*" (Dourish and Chalmers, 1994). Systems that integrate social navigation concepts exploit social practices and behavior to help users navigate and explore (Chalmers et al., 2004). Such systems typically illustrate other users' actions in the user interface and therefore raise users' awareness of other users with a similar searching goal (Papagelis et al., 2008). Amazon[3] utilizes the concept of social navigation through the "*Customers Who Bought This Item Also Bought*" recommendations (see Figure 2.3). Another example of social navigation is the ratings information integrated at the

[3] `http://www.amazon.com`

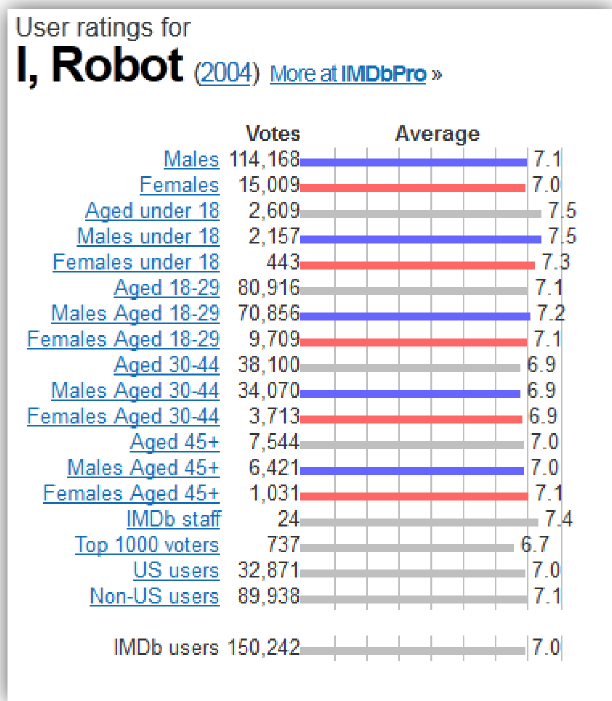

Figure 3.5.: *Social navigation* at the Internet Movie Database (IMDb) – ratings information for the movie "*I, robot*".

Internet Movie Database website (IMDb)[4]. For example, in Figure 3.5 we can see that the movie "*I, Robot*" was rated best by male users under 18.

Users of systems that integrate social navigation have to accept that their actions are visible to other users – this intrusion in users' privacy may result in a loss of trust in the system (Svensson et al., 2001). Erickson and Kellog (Erickson and Kellogg, 2000) introduced the concept of *social translucence* to establish design issues for digital systems that make social cues visible to users. To support social processes in users' search activities they define three characteristics of such systems: "*visibility, awareness, and accountability*" (Erickson and Kellogg, 2000). This concept points out that it is important that users know what social information is disclosed by the system and how it is used.

The Role of Social Navigation in Recommendation Scenarios

Svensson et al. (Svensson et al., 2001) developed a food receipt recommender that integrate social navigation concepts. In this systems users can group a set of receipts to a *recipe collection*, which is defined as a place "*where customers can meet, socialise and get recommendations*"(Svensson et al., 2001). As a result of a conducted user study, Svensson et al. (Svensson et al., 2001) point out that users perceive social navigation as increase in value for the service. Their results indicate that as long as actions are anonymous users do not mind to share their (food) preferences with

[4] `http://www.imdb.com/`

other system users. But a small group felt negatively influenced when actions are not anonymous. This individual differences are an important factor that must be taken into account in the design of recommender applications that support social navigation (Svensson et al., 2001).

Another important issue that has to be considered when designing social navigation systems is the *snowball effect* (Svensson et al., 2001). Svensson et al. describe this effect as "*where the social trails lead more and more users down a path they do not perceive valuable in the long run*" (Svensson et al., 2001). Svensson et al. explained that in their food receipt recommender, users were lured to move to recipe collections with many users (Svensson et al., 2001). In order to reduce the snowball effect, they suggest to create different visualizations of, for example, friends and unknown users, such that users can choose whom to follow, rather than follow an anonymous crowd (Svensson et al., 2001).

3.3. Evaluating Recommender Systems

The comparison of recommender algorithms is often based on offline metrics using techniques from machine learning and information retrieval (Papagelis and Plexousakis, 2005). Within an offline experiment typically the *prediction quality* of recommender algorithms is measured and compared. Prediction can be defined as "*a value that expresses the predicted likelihood that a user will 'like' an item*" (Papagelis and Plexousakis, 2005) and therefore improved prediction quality can lead to better recommendations for the user.

A major drawback of this evaluation strategy is that the performance of a recommender algorithm is typically evaluated on existing datasets. The authors of (Hayes et al., 2002) argue that "*it would be a mistake to assume that domain specific datasets such as those from the EachMovie and MovieLens projects can be used to test every Recommendation System*" (Hayes et al., 2002). Another limitation of this approach is that the recommendation problem is reduced to the problem of calculating ratings for items.

In addition to the importance of the prediction quality of the algorithm, recommender systems have various properties that may have an effect on user experience, such as interface design elements, robustness or trustworthiness of the system. For example, the results of the research of Mandel and Johnson (Mandel and Johnson, 1998) indicate that web page design can have an impact on users' perceived importance of product attributes and therefore on the final product choice. Chen and Pu (Chen and Pu, 2005) suggest that a user's level of trust (confidence) in recommendations is directly correlated with the willingness to return to the website and make a buying decision.

In recent years measuring the user experience during the recommendation process has increasingly gained importance in the evaluation process (Hayes et al., 2002; McNee et al., 2006; Pu and Chen, 2010). In the research community three different types of experiments to evaluate recommender systems have been established: *offline*, *online* and *user studies* (Shani and Gunawardana, 2009). In the following sections we provide an overview of these evaluation methods.

3.3.1. Offline Evaluation

Offline experiments are performed by using pre-collected or artificially generated data sets to simulate user interaction with the recommender system. This evaluation strategy is based on the assumption that the applied data sets can model reliable user behavior in the context of the test setting (Shani and Gunawardana, 2009).

Offline experiments are typically used for measuring the performance of recommender algorithms. Performance can be measured in terms of *prediction quality* – the likelihood that the algorithm calculates a recommendation that is of interest for the user. Performance measures in the field of recommender systems have their roots in machine learning and information retrieval. In order to apply these measures in a recommendation scenario, the items of the recommender engine have to be classified into four groups to build a so-called *confusion matrix* (see Table 3.12) (Kohavi and Provost, 1998). Recommended items that are of interest to the user are classified as true positive (*tp*), whereas recommended items that are uninteresting to the user are false positive (*fp*). An item is classified as false negative (*fn*) if it is interesting to the user but the engine does not recommend it. If the engine does not recommend an item uninteresting for the user it is classified as true negative (*tn*).

Table 3.12.: Classification of items used by a recommender engine (Shani and Gunawardana, 2009).

	Recommended	**Not recommended**
Used	True positive (tp)	False negative (fn)
Not used	False positive (fp)	True negative (tn)

Commonly used measures to estimate the quality of a recommendation algorithm are *accuracy* and *coverage*. Accuracy can be defined as the number of correct recommendations to all possible recommendations (see Formula 3.6), and coverage refers to the proportion of items that the system can recommend (see Formula 3.7) (Schulz and Hahsler, 2002).

$$prediction\ accuracy = \frac{\#\ tp + \#\ tn}{\#\ tp + \#\ fn + \#\ fp + \#\ tn} = \frac{\#\ of\ correct\ recommendations}{total\ possible\ recommendations} \tag{3.6}$$

$$prediction\ coverage = \frac{\#\ of\ recommendations}{total\ number\ ot\ items} \tag{3.7}$$

Well known measures in information retrieval are *precision* and *recall* – see Formulas 3.8 and 3.9 for their definition in the field of recommender systems (Shani and Gunawardana, 2009). Precision (see Formula 3.8) can be defined as "*the proportion of recommendations that are good recommendations*" (Harita et al., 2011) and recall (see Formula 3.9) can be defined as "*the proportion of good recommendations that appear in top recommendations*" (Harita et al., 2011).

$$precision = \frac{\#\ tp}{\#\ tp + \#\ fp} = \frac{\#\ of\ correctly\ recommended\ items}{total\ number\ of\ recommended\ items} \tag{3.8}$$

$$recall = \frac{\#\ tp}{\#\ tp + \#\ fn} = \frac{\#\ of\ correctly\ recommended\ items}{total\ useful\ items} \tag{3.9}$$

The fact that no real users are needed for offline evaluation makes it very easy to evaluate and compare recommender algorithms in terms of prediction quality. A major drawback of this evaluation type is, that there is no information about the influence of recommendations on users' decision making behavior.

We applied an offline evaluation framework in the study described in Section 4.1, that aimed at comparing the performance of our new unit critiquing approaches with existing critiquing strategies. For this purpose we performed an offline experiment which applied a leave-one-out approach (see, for example, (McCarthy et al., 2010)), where each case in the item case base is temporarily removed and used as *target* of a critiquing session (see Section 4.1.4 for more details).

3.3.2. Online Evaluation

Research has shown that user satisfaction with a recommender system does not necessarily correlate with algorithm accuracy (see, for example, (McNee et al., 2006)). In recent years, researchers have identified many properties that may have an impact on the success of a recommender system in addition to the importance of calculating accurate predictions. The format of information presentation as well as elements of user interface design can have a significant impact on user attitudes and perceptions of the trustworthiness of a system (see, for example, (Kim and Moon, 1998; Roy et al., 2001; Eroglu, 2001; Eroglu et al., 2003)).

Pu and Chen (Pu and Chen, 2010) developed an evaluation framework for recommender systems that aims at measuring the perceived qualities of the application – for example, the usability, interface qualities, and the users' satisfaction with the system – and the influence of these qualities on users' behavioral intentions. They elaborated 60 questions concerning users' subjective attitudes based on their experience towards a recommender application. Such questionnaires are typically presented after the participants interacted with the recommender application to measure users' general satisfaction with the system. Table 5.2 gives an example of evaluation questions.

Hayes et al. suggest that "*user satisfaction with a recommendation strategy can only be measured in an on-line context*" (Hayes et al., 2002). They propose an online evaluation framework for pairwise comparison of recommender algorithms. In online research, a recommender system is typically deployed as web application and the performance or quality of the system is tested by a large group of users. In such an online evaluation it is possible to measure *objective metrics* (for example, the willingness of users to rely on recommendations) as well as *subjective measurements* (for example, confidence metrics) by collecting implicit and explicit feedback from the users.

Many companies such as Amazon or Microsoft employ controlled online experiments (Kohavi et al., 2009). A random sample of system users are redirected to an experimenting website, where either the user interface or the recommendation algorithm differs from the original system. The user interaction with the alternative website is recorded in order to compare it with the original system (Kohavi et al., 2009).

Online experiments are a more expensive option compared to offline studies since a fully engineered recommendation system is needed, but such experiments can provide valuable information about users' satisfaction with the recommender application.

In the context of this work we performed three online experiments. The study described in Section 4.2 aimed at analyzing the subjective quality of the RecoMobile configurator, a personalized configuration system for mobile phones and subscriptions. In Section 5.1 we describe an experiment that aimed at investigating the impact of different presentation styles of personalized defaults, and in Section 5.2 we discuss an empirical study, that had the goal to explore the impact of the *Status Quo Bias* in product configuration scenarios where defaults are presented as recommendations to users.

3.3.3. User Experiments

In a user study a small group of test subjects is asked to interact with the recommender system in a controlled environment and report on their experience (Shani and Gunawardana, 2009). A popular method for performing a user experiment is the *thinking aloud protocol* (Lewis, 1982). The users are asked to think aloud while performing several specified tasks with the recommender application. Their comments and behavior are recorded during the experiment. While collecting *quantitative measures*, for example, which task was completed by the users, or the time needed to complete the tasks, it is also possible in such experiments to ask the user *qualitative questions*, for example, whether the recommendations were considered as helpful, or what is the user's subjective perception about the user interface (Shani and Gunawardana, 2009).

User studies are very expensive to conduct, requiring a significant amount of time and effort (Shani and Gunawardana, 2009). But the experimental settings of this evaluation type allow to observe the users' behavior while interacting with the recommendation system, and therefore user experiments can cover the widest set of questions compared to offline and online experiments (Shani and Gunawardana, 2009).

Chapter 4

Improving Recommendation Quality in Intelligent Selling Applications

Parts of the contents of this chapter have been published in (Felfernig et al., 2010a,b; Mandl and Felfernig, 2012a).

In this chapter results of our research are presented that aimed at improving recommendation quality in the context of intelligent selling applications. In Section 4.1 a new approach is introduced to calculate recommendations in a critiquing environment that reuses past users' sessions to identify critique-product correlations. We combine our approach with the conventional critiquing algorithm (Burke et al., 1996) and the experience-based technique (McCarthy et al., 2010) to an ensemble-based variation. To demonstrate improvements in recommendation efficiency we conducted an offline evaluation based on artificial user data.

In Section 4.2 we combine knowledge-based configuration with collaborative recommendation algorithms in order to improve the quality of product configuration processes by personalizing the configuration experiences for users. To demonstrate the applicability of our approach, we performed an online evaluation where both, implicit and explicit user feedback was collected.

4.1. Improving the Performance of Unit Critiquing

Critiquing-based recommender systems belong to the group of conversational recommender systems (Salamó et al., 2005; Viappiani et al., 2007). Users of such systems provide feedback by critiquing attributes of recommended items in a directional way (McCarthy et al., 2005). A major advantage of this feedback strategy is that the user can constrain a particular product attribute without providing a specific value (Salamó et al., 2005). For example, a user of a holiday recommender might express that he/she wants to see a holiday package that is *cheaper* than the actual recommendation by critiquing the corresponding *price* attribute. Research on human decision making has shown that users are rarely able to provide complete and accurate preferences at the beginning of a recommendation session (Reilly et al., 2007b), but become aware of their preferences when recommended products violate them (Pu and Faltings, 2000) – therefore the feedback can be inconsistent and contradictory (Reilly et al., 2007b). This fact is typically integrated in the incremental refinement of the user preference model (see, for example, (Reilly et al., 2005)).

There exist different approaches to integrate critiquing in a decision support tool. *Unit critiques* operate on a single product attribute in each critiquing cycle and typically facilitate a ”more”, ”less”, or ”other” type of feedback. *Compound critiques* provide the possibility to critique multiple product attributes during each critiquing cycle (McCarthy et al., 2005). Different knowledge sources are exploited to calculate such critiques. *Static compound critiques* are generated according to the system's knowledge of the product domain (see,for example, (Burke et al., 1997)) – the items are equipped with a fixed set of critiques. An example of such a compound critique is the *sportier* critique in the Car Navigator system (Burke et al., 1997), which implies several changes to the feature set: engine size, acceleration, and price are all increased. *Dynamic compound critiques* are generated according to the system's knowledge of the remaining items (see, for example, (McCarthy et al., 2004)). Such critiques are a selection of individual feature critiques that represents the differences between the actual recommendation and the remaining items (McCarthy et al., 2004). For example, a user in the PC domain can express that he/she is interested in a product that is cheaper and equipped with a faster CPU compared to the current recommendation by selecting the corresponding *lower price, faster CPU* compound critique. *Incremental critiquing* (Reilly et al., 2005) extends the dynamic critiquing approach by exploiting previous user critiques to calculate future recommendations. Zhang and Pu (Zhang and Pu, 2006) have introduced an approach where compound critiques are generated according to the system's knowledge of the user's preferences. The calculation of such compound critiques is based on the multi-attribute utility theory (MAUT) (Keeney and Raiffa, 1976). An in-depth discussion of critiquing techniques can be found in (Chen and Pu, 2012).

In this section we focus on improving the efficiency (in terms of the number of critiquing cycles) of unit critiquing. We introduce an extension of the *experience-based unit critiquing approach* developed by McCarthy, Salem, and Smyth (McCarthy et al., 2010). The basic idea of experience-based critiquing is that successful critiquing experiences – critiquing sessions which led to a purchase decision – may imply critiquing patterns that are similar to the current user's critiquing session, and therefore might help to short-cut the critiquing process for the current user (McCarthy et al., 2010). Experience-based critiquing systems search for the user with the most similar critiquing history compared to the current user (nearest neighbor) and use the corresponding accepted final item as new recommendation. We adopt this idea and recommend that item in nearest neighbor's critiquing history that best matches the current user's requirements. The new recommendation represents an item that has been presented to a previous user with a similar critiquing history but not necessarily the final purchase decision. We call this new approach *nearest neighbor compatibility critiquing*. To further reduce the number of interaction cycles needed to successfully complete a critiquing session, we combine this approach with conventional and experience-based critiquing to corresponding ensemble-based variations.

4.1.1. Unit Critiquing

Research has shown that unit critiques result in significant lower cognitive costs for users compared to compound critiques. The reason for this is, that it is more difficult to evaluate and understand compound critiques (McCarthy et al., 2005). Although the usage of compound critiques can result in shorter critiquing sessions users are more willing to apply unit critiques (McCarthy et al., 2005). In the following we will present basic concepts of existing approaches to unit critiquing.

Conventional Critiquing – Tweaking

One of the earliest systems that deployed the conventional critiquing approach were the FindMe Systems (Burke et al., 1996). In such systems the user critiques are handled as a *"show me more like this,* ***but****..."* type of feedback. When the user applies a critique c_i to a recommended item r_i the applied recommendation strategy is to find an item which is compatible with the current user critique c_i, and which is maximally similar to the critiqued item r_i. Algorithm 1 shows a simplified version of conventional critiquing. First the algorithm filters out those items that are incompatible with the current user critique and then selects the next recommendation from the remaining cases.

Algorithm 1 Conventional Critiquing

Input:

- c_u – current user's critique
- CB – item catalog
- r_i – actual recommendation

$CB' \leftarrow \{r \in CB \mid satisfies(r, c_u)\}$
$CB' \leftarrow$ *sort cases in* CB' *in decreasing order of their similarity to* r_i
$r_i \leftarrow$ *most similar item in* CB'
return r_i

Experience-Based Critiquing

The experience-based critiquing approach (McCarthy et al., 2010) is based on the idea of determining recommendations by exploiting information from the critiquing sessions of previous users. A user's critiquing session s_u can be defined as a sequence of recommendation critique pairs p_i ((McCarthy et al., 2010) – see Formulae 4.1 and 4.2).

$$s_u = \{p_1, ..., p_n\} \tag{4.1}$$

$$p_i = (r_i, c_i) \tag{4.2}$$

r_i represents the recommended item in critiquing cycle i, and c_i represents the critique that was applied to that item. A critique can be defined as a triple that is composed of the item's attribute f_i that is the focus of the critique, the value v_i of that attribute, and the *type* of the applied critique (typically <, >, =, <>, *accept*, where *accept* marks the item as final decision) ((McCarthy et al., 2010) – see Formula 4.3).

$$c_i = (f_i, v_i, type) \tag{4.3}$$

If the user applies a new critique to a recommended item the system typically checks this critique against the critiquing session of the user. If the new critique contradicts or refines an old critique, the old critique is deleted from the critiquing history before the new critique is added (McCarthy et al., 2010).

The experience-based critiquing approach (McCarthy et al., 2010) exploits information about critiquing experiences of past system users to calculate recommendations for the current user. The algorithm first extracts those previous sessions where the accepted final item is compatible with the current user's critique (see Algorithm 2). These previous user sessions are then ranked according to the similarity of their critiquing history to the (partial) critiquing history of the current user. The session of the top ranked candidate (nearest neighbor) is then used to recommend the corresponding accepted final item to the current user.

Algorithm 2 Experience Based Critiquing

Input:

- c_u – current user's critique
- s^u – current user's critiquing session
- S^P – previous users' critiquing sessions
- r_i – actual recommendation
- $r_{p,final}$ – accepted final item in session p
- t – threshold for number of overlapping critiques

$SP' \leftarrow \{s^p \in S^P \mid satisfies(r_{p,final}, c_u)\}$
$SP' \leftarrow sort\ cases\ in\ SP'\ in\ decreasing\ order\ of\ their\ similarity\ to\ s^u$
$s_i \leftarrow session\ with\ highest\ similarity\ in\ SP' > t$
$r_i \leftarrow r_{p,final}\ in\ s_i$
return r_i

After this introduction to the basic unit critiquing strategies, we will now focus on our new approach – *nearest neighbor compatibility critiquing.*

4.1.2. Algorithm: Nearest Neighbor Compatibility Critiquing

The incremental critiquing approach (Reilly et al., 2005) extends the ideas of conventional critiquing (Burke et al., 1996). The algorithm focuses on finding an item that is compatible with the current user critique and satisfies most of the user's previous critiques in the critiquing history (Reilly et al., 2005). We combine this approach with the ideas of experience-based critiquing (McCarthy et al., 2010) and exploit previous users' critiquing experiences to find that item in nearest neighbor's critiquing history that best matches the current user requirements. For each item that was critiqued in the nearest neighbor's critiquing session we calculate a compatibility score for the user's critiquing history. That item in the nearest neighbor's critiquing history that satisfies the actual user critique and that has the highest compatibility score will serve as recommendation to the current user. We denote this new approach *Nearest Neighbor Compatibility Critiquing*. In Algorithm 3 the basic steps of this critiquing approach are shown[1]. Note that in the case that there is no relevant item to recommend, we revert to conventional critiquing.

Algorithm 3 Nearest Neighbor Compatibility Critiquing

Input:

- c_u – current user's critique
- s^u – current user's critiquing session
- S^P – previous users' critiquing sessions
- CS – compatibility scores for items
- r_i – actual recommendation
- t – threshold for number of overlapping critiques

```
SP' ← sort cases in S^P in decreasing order of their similarity to s^u
s_i ← session with highest similarity in SP' > t
for each critique c_i in s^u do
  for each recommendation r_i in s_i do
    if satisfies(r_i, c_i) then
      CS ← updateCompatibilityScore(CS, r_i)
    end if
  end for
end for
CS' ← {r ∈ CS | satisfies(r_i, c_u)}
r_i ← item with highest compatibility score in CS'
return r_i
```

A simple example of the application of *nearest neighbor compatibility critiquing* is the following. Table 4.1 contains six example products (digital cameras). Table 4.2 contains three successful critiquing sessions s_1, s_2, and s_3 from previous users. Let us assume that the current user has already applied the critiques $\mathbf{s^u}$={c_0:*manufacturer !=HP*, c_1:*price>160*, c_2:*MPix<12*}. The nearest neighbor compatibility critiquing approach will identify session s_2 as the nearest neighbor session for this combination of critiques (the critiques on the attributes *manufacturer* and *MPix* are identical to the critiques in s^u). The next step is to find the item in the critiquing history of the nearest neighbor session that best satisfies the current user's critiques. In the nearest neighbor session s_2 products p2, p4, p1, and p3 were critiqued (see Table 4.2). p1, p3, and p4 satisfy the current user's critique on the manufacturer attribute (c_0:*manufacturer !=HP*), p1, p2, and p4 satisfy the critique on price (c_1:*price>160*), and p1 and p3 satisfy the critique on the MPix attribute (c_2:*MPix<12*). Therefore the nearest neighbor compatibility critiquing approach will present *p1* as recommendation since it best satisfies the current user's critiques.

[1] Note that in our implementation we used a simple similarity approach where only direct matches of critiques are considered as similar.

Table 4.1.: Available digital cameras in working example.

	price	**manufacturer**	**MPix**
p1	*170*	*Canon*	*10*
p2	*250*	*HP*	*12*
p3	*150*	*Canon*	*8*
p4	*190*	*Nikon*	*12*
p5	*280*	*HP*	*16*
p6	*160*	*Nikon*	*10*

Table 4.2.: Example: successful critiquing sessions from previous users (s_1, s_2, and s_3), and the uncompleted critiquing session from the current user s^u.

user session	**critiqued product**	**critique**
$\mathbf{s_1}$	p3	c_0=(price, 150, >)
	p1	c_1=(manufacturer, Canon, !=)
	p4	c_2=(MPix, 12, =)
	p2	c_3=accept
$\mathbf{s_2}$	p2	c_0=(manufacturer, HP, !=)
	p4	c_1=(MPix, 12, <)
	p1	c_2=(price, 170, <)
	p3	c_3=accept
$\mathbf{s_3}$	p6	c_0=(price, 160, >)
	p4	c_1=(manufacturer, Nikon, !=)
	p2	c_2=(MPix, 12, >)
	p5	c_3=accept
$\mathbf{s^u}$	p5	c_0=(manufacturer, HP, !=)
	p6	c_1=(price, 160, >)
	p4	c_2=(MPix, 12, <)

4.1.3. Ensemble-based Variation

The basic idea of ensemble-based methods is to combine the results of several recommendation algorithms to improve the overall prediction quality (Rokach, 2009). Research has shown that this approach has the potential to outperform corresponding individual strategies – some of the top-ranked teams in the Netflix competition[2] applied an ensemble based approach (Bell et al., 2007; Piotte and Chabbert, Piotte and Chabbert; Töscher and Jahrer, 2008).

In the following we will describe our approach to combine different unit critiquing algorithms to an ensemble-based solution in order to reduce the number of critiquing cycles.

Identifying an Ensemble-Based Solution

In order to calculate an ensemble-based recommendation we combine *nearest neighbor compatibility critiquing* with *conventional critiquing* (Burke et al., 1996) and *experience-based critiquing* (McCarthy et al., 2010). We select the k best-ranked items of each algorithm and weight them

[2] `http://www.netflixprize.com/`

Table 4.3.: Product ranking of individual critiquing algorithms.

	ranking		
algorithm	**1**	**2**	**3**
conventional critiquing	*p6*	*p1*	*p3*
experience-based critiquing	*p3*	-	-
nearest neighbor compatibility critiquing	*p1*	*p3*	-

Table 4.4.: Weights of ranking positions.

ranking position	1	2	3
weight of ranking	100	10	1

according to their ranking positions in the individual algorithm rankings (see Formula 4.4).

$$ensembleranking_{a_i} = \sum_{i=1}^{n}\sum_{j=1}^{k} itemranking(a_{ij}) * rankingweight(j) \tag{4.4}$$

In Formula 4.4, *itemranking(*a_{ij}*)* specifies the item which is ranked on position *j* by algorithm *i* (see Table 4.3), and *rankingweight(j)* defines the weight of a specific ranking position *j* in the result list (see Table 4.4). For example, itemranking(2,1) is p3 since p3 has been ranked on position 1 by experience based critiquing and the weight of ranking position 1 is 100.

Let us assume that the current user $\mathrm{s^u}$ has applied the following critique to product *p4* in the current critiquing cycle: $\mathbf{c_a}$=$\{MPix<12\}$ (see Table 4.2). *Conventional critiquing* will rank product *p6* on the first position (see Table 4.3), since *p6* satisfies the user's critique, and it is most similar to the critiqued product *p4*. *Experience-based critiquing* will select product *p3* as top-ranked, since in session s_2 (session with the critiquing history most similar to the current session s^u), *p3* has been selected (e.g., purchased) by the user (see Table 4.2). *Nearest neighbor compatibility critiquing* will calculate *p1* as best recommendation, since it is that item in nearest neighbor session s_2, that best matches the current user's critiques. As shown in Table 4.5, the product with the highest ensemble-based ranking is product *p3*, which will be used as next recommendation to the current user.

For our evaluation we combine the individual algorithms to three ensemble-based critiquing variations (see Table 4.6): *variation 1* assembles *conventional* with *experience-based critiquing*, *variation 2* assembles *conventional* with *nearest neighbor compatibility critiquing*, and *variation 3*

Table 4.5.: Ensemble-based ranking of possible recommendations in the next critiquing cycle.

product	**overall rating**
p1	1*100+1*10 + 0*1= 110
p2	0*100+0*10 + 0*1= 0
p3	1*100+1*10 + 1*1= **111**
p4	0*100+0*10 + 0*1= 0
p5	0*100+0*10 + 0*1= 0
p6	1*100+0*10 + 0*1= 100

Table 4.6.: Ensemble based variations.

	conventional	experience-based	nearest neighbor compatibility
Variation 1	x	x	
Variation 2	x		x
Variation 3	x	x	x

includes all three algorithms.

4.1.4. Evaluation

In our evaluation we compare the ensemble-based solutions with the performance of the individual critiquing algorithms. We will use the well-known *Travel* dataset (available from `http://www.ai-cbr.org/`), that consists of over 1000 vacation cases. Each case is described in terms of 9 features – 6 nominal, and 3 numeric features – including *Price*, *Region*, *Transportation*, *Hotel*. On the basis of this dataset, we performed an offline experiment which follows the leave-one-out approach described by, for example, McCarthy, Salem, and Smyth (McCarthy et al., 2010). In this evaluation method each case in the item case base (the Travel dataset) is temporarily removed and used as *target* of a critiquing session. A critiquing session consists of two steps: In the first step the target case is extracted from the dataset and a random subset of the target item's features are taken to form the initial query. This query is used to find the first recommendation among the remaining cases in the case base. In the second step the target item is included in the case base again. We randomly select one of the nine features of the vacation cases that serves as focus attribute to critique. For each recommended case a critique is generated that is compatible with the target item. For example, if the current recommended holiday has a duration of 7 days and the target holiday has a duration of 14 days, a "*more*" critique is applied. The critiquing session terminates when the target item is selected as recommendation. Therefore the *number of critiquing cycles* that were needed until the target item was recommended serves as performance measure to compare the different critiquing strategies.

Experience-based critiquing as well as *nearest neighbor compatibility critiquing* reuse past critiquing sessions to calculate recommendations for the current user. For this purpose we use the critiquing sessions generated with the *conventional critiquing* algorithm. We generated three different initial queries for the 1024 travel cases in the dataset, and applied the resulting critiquing sessions as a session case base for the experience-based and nearest neighbor compatibility critiquing techniques.

4.1.5. Results

For our evaluation we recorded the number of critiquing cycles (session length) needed until the target item was recommended and averaged these results for each algorithm. Therefore the key performance measure is the *average number of critiquing cycles* (see Figures 4.1 and 4.2).

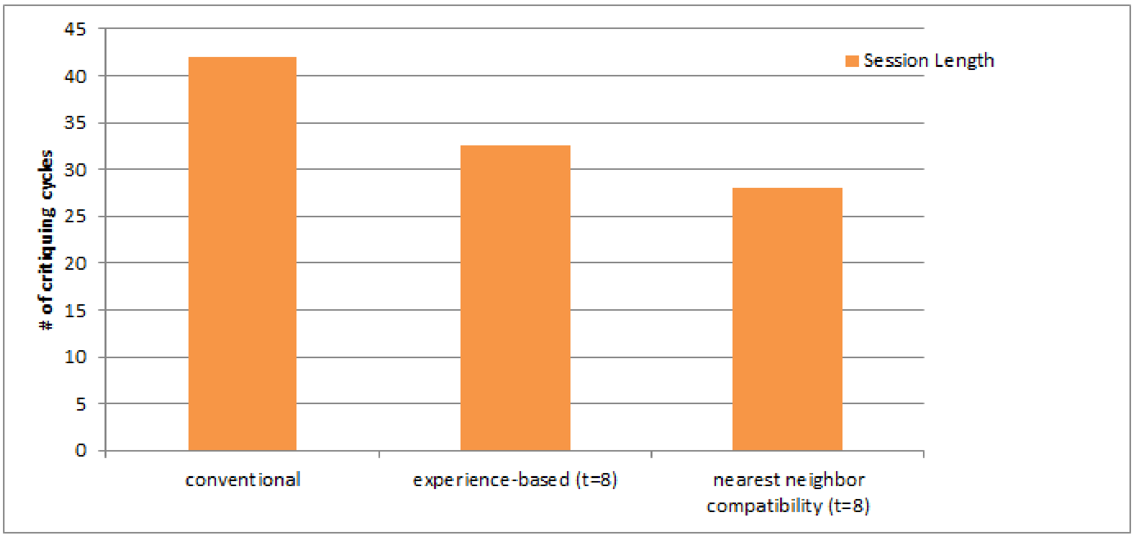

Figure 4.1.: Evaluation results for the individual critiquing algorithms – average session lengths (# critiquing cycles) recorded for each of the algorithms.

An average critiquing session where the conventional critiquing algorithm is used requires 42 critiquing cycles to recommend the target item (see Figure 4.1). As mentioned by McCarthy, Salem, and Smyth (McCarthy et al., 2010), for the experience-based approach the impact of the threshold value t (specifies the minimum amount of overlapping critiques) has a strong impact on the performance results of the algorithm. In our setting we found that the experience-based approach performed best with $t=8$ (critiquing sessions must have at least three overlapping critiques). This is consistent with the findings of McCarthy, Salem, and Smyth who indicate that thresholds of 8 and greater can lead to shorter average session lengths compared to conventional critiquing (McCarthy et al., 2010). Correspondingly, the performance of the *nearest neighbor compatibility approach* also depends on the selection of an appropriate threshold. Figure 4.1 exemplifies the results with threshold values 8 for the experience-based and the nearest neighbor compatibility approaches. With the experience-based algorithm 33 cycles, and with the nearest neighbor compatibility approach 28 cycles are needed on average. Our strategy to recommend that item that was of interest for the nearest neighbor user but not necessarily the user's final purchase decision can lead to better recommendations for the current user. Since our results are based on artificially generated data they have to be considered as preliminary and need to be verified in an online experiment with real users. But as McCarthy, Salem, and Smyth (McCarthy et al., 2010) point out, exploiting the information of past users' critiquing history can be promising in the field of critiquing systems, leading to less effort for users to find their target item.

The results of the ensemble-based variations are illustrated in Figure 4.2. If two algorithms are combined to calculate an ensemble-based recommendation, the number of critiquing cycles can be reduced – variation 1, where conventional critiquing is combined with experience based critiquing, shows a similar performance as our nearest neighbor compatibility approach (28 cycles are needed on average until the target item is recommended). Variation 2 – the combination of conventional and nearest neighbor compatibility critiquing – shows the best performance of the ensemble-based

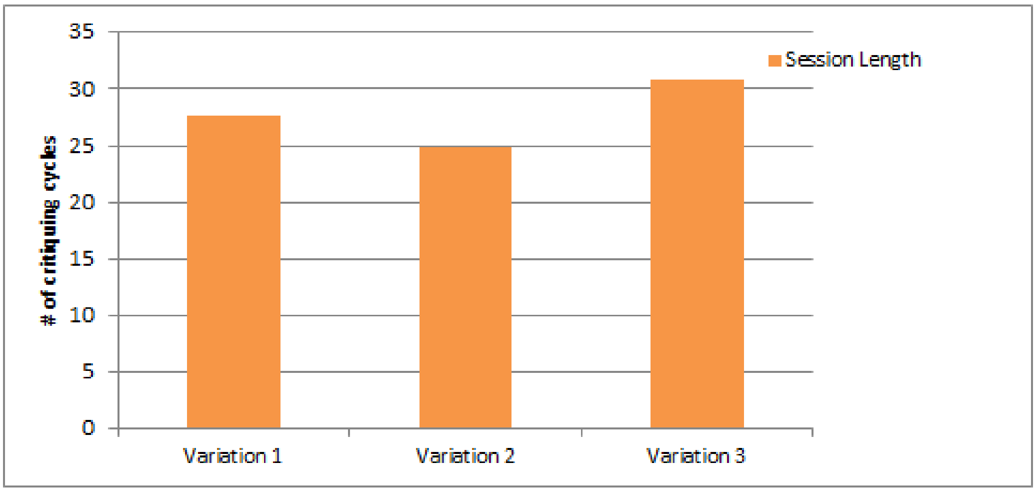

Figure 4.2.: Evaluation results for the ensemble-based critiquing variations – average session lengths (# critiquing cycles) recorded for each of the algorithms.

variations (25 cycles are needed on average until the target item is recommended). Variation 3 – the combination of all three individual algorithms – cannot take advantage of the item candidates recommended by the individual algorithms. This variation requires on average 31 critiquing cycles to recommend the target item.

4.1.6. Related Work

Conversational recommender systems help users to quickly navigate to suitable products in the product space by supporting an incremental construction of user preferences (Burke et al., 1996, 1997). Different strategies for capturing user feedback have been explored, which can be categorized in four types: value elicitation, tweaking/critiquing, preference-based, and ratings-based feedback (see, for example, (Smyth and McGinty, 2003)). We set the focus of this section on critiquing-based recommender systems. In such systems users provide feedback by critiquing particular product attributes of a presented recommendation in a directional way.

The simplest form of critiquing is *unit critiquing* (Burke et al., 1996; McCarthy et al., 2010), where users can constrain one single product attribute in each recommendation cycle. In the first systems that implemented this approach (for example, the *FindMe* systems (Burke et al., 1996)) the response to a user critique was to recommend a new item, that is compatible with the actual critique and that is maximally similar to the critiqued item. While this approach works well in domains that are reasonably sparse it can lead to protracted critiquing sessions in domains that are dense (Burke, 2002b). Burke explains that when the domain is dense, "*products that marginally meet the critique are not really very different from the source*" (Burke, 2002b). To overcome this problem, and to make larger jumps in the product space, the concept of *compound critiques* has been introduced. This critiquing type provides the possibility to critique multiple features within a single critiquing cycle (see, for example, (Burke et al., 1997; McCarthy et al., 2004; Zhang and Pu, 2006)). Research has shown that compound critiques have the potential to outperform the unit critiquing technique in terms of less critiquing cycles, but the application of compound critiques significantly increases the cognitive load for the user (McCarthy et al., 2005).

McCarthy, Salem, and Smyth (McCarthy et al., 2010) introduced a new approach to improve the efficiency of unit critiquing. They exploit critiquing experiences of past system users to calculate recommendations for the current user (*experience-based unit critiquing*). We adopted the idea of McCarthy et al. (McCarthy et al., 2010) and introduced a new approach (*nearest neighbor compatibility critiquing*) to item selection which can potentially reduce critiquing session length. In addition, we introduced the concept of ensemble-based critiquing, which has the potential to further reduce the number of needed critiquing cycles.

4.1.7. Conclusion

Conversational recommender systems allow users to learn and adapt their preferences according to concrete examples. Critiquing systems support such a conversational interaction style. Especially unit critiques offer a low cost feedback strategy for users in terms of the needed cognitive effort. In this section we presented an extension of the experience-based unit critiquing algorithm. The development of our new approach, which we denote as *nearest neighbor compatibility critiquing*, aimed at increasing the efficiency of unit critiquing. We combined our new approach with existing critiquing strategies to ensemble-based variations and presented the results of an empirical study that aimed at comparing the recommendation efficiency (in terms of the number of critiquing cycles) of ensemble-based solutions with individual critiquing algorithms. The results of our experiment indicate that our new nearest neighbor compatibility critiquing approach, as well as ensemble-based variations thereof have the potential to reduce the number of critiquing cycles in critiquing sessions.

4.2. RecoMobile

Mass customization aims at customizing products according to individual preferences and needs with near mass production efficiency (Pine, 1993; Tseng and Jiao, 2001). Following the paradigm of *mass customization*, the intelligent customizing of products and services is crucial for manufacturing companies to stay competitive. Configuration technologies are well established as a foundation of mass customization and support the manufacturing of highly-variant products under pricing conditions similar to mass production (Sabin and Weigel, 1998).

Although configuration systems have many advantages such as a significantly lower amount of incorrect quotations and orders, shorter product delivery cycles, and higher productivity of sales representatives (Barker et al., 1989), customers (users) are often overwhelmed by the complexity of offered alternatives. This phenomenon is well known as *mass confusion* (Huffman and Kahn, 1998). Another problem is described by the *theory of preference construction* (Bettman et al., 1998), that explains, that users typically do not know exactly which products or components they would like to have. As a consequence, users construct and adapt their preferences within the scope of a configuration process (Bettman et al., 1998).

A possibility to help the user identifying meaningful alternatives that are compatible with his/her current preferences, is to provide *defaults*. Defaults in the context of interactive product configuration dialogs are *preselected options used to express personalized feature recommendations* (Mandl et al., 2011b).

In order to improve the quality of configuration processes, we combine knowledge-based configuration (Stumptner, Stumptner) with collaborative recommendation algorithms (Goldberg et al., 1992; Herlocker et al., 2000; Terveen and Hill, 2001). In this context we present RecoMobile which is a personalized configuration system for mobile phones and subscriptions. The entry screen of RecoMobile is depicted in Figure 4.3. First, the user has to answer a few questions concerning some general attributes of the configuration domain (for example, the preferred phone style or preferences regarding the internet access). For the following questions regarding mobile subscription details, privacy settings, and the phone, the recommender proposes feature settings (*default proposals*) that are determined on the basis of user interactions of past configuration sessions (see Figure 4.4). After the specification of a set of requirements, the configuration system checks whether a solution (configuration) exists. In the case that no solution could be found, the system activates a diagnosis & repair component (Reiter, 1987; Felfernig et al., 2004, 2008, 2009) (see Figure 4.5). An example of such an infeasibility in the mobile phones domain is the combination of *the phone should have no web access* and *I want to read Emails with the phone*. In such a situation, the system activates a repair component that identifies minimal sets of changes such that the retrieval of at least one solution is possible. The layout of the RecoMobile phone selection page is depicted in Figure 4.6. The phone selection page enlists the set of phones that fulfill the given set of customer requirements. This set is ranked on the basis of collaborative similarity metrics (for details see the next section). For each mobile phone the user can activate a corresponding explanation page. In RecoMobile, the explanations are presented in the form of a detailed enlisting of those user requirements which are fulfilled by the specific mobile phone (see Figure 4.7). Finally, it is possible to select the preferred mobile phone and to finish the session.

In order to demonstrate the applicability of our approach, we present results of an empirical study that include important improvements in terms of, for example, *user satisfaction*, or *perceived configuration process quality*.

4.2.1. Configuring

In this section we will provide technical details, that help to understand how RecoMobile supports configuration tasks, and how the system determines repair alternatives in situations where no solution could be found.

Supporting configuration tasks

The task of identifying a configuration for a given set of specified customer requirements can be defined as follows (Felfernig et al., 2010a,b):

Definition 3 (configuration task): a configuration task can be defined as a constraint satisfaction problem (V, D, C). V = $\{v_0, v_1, ..., v_n\}$ represents a set of finite domain variables and D

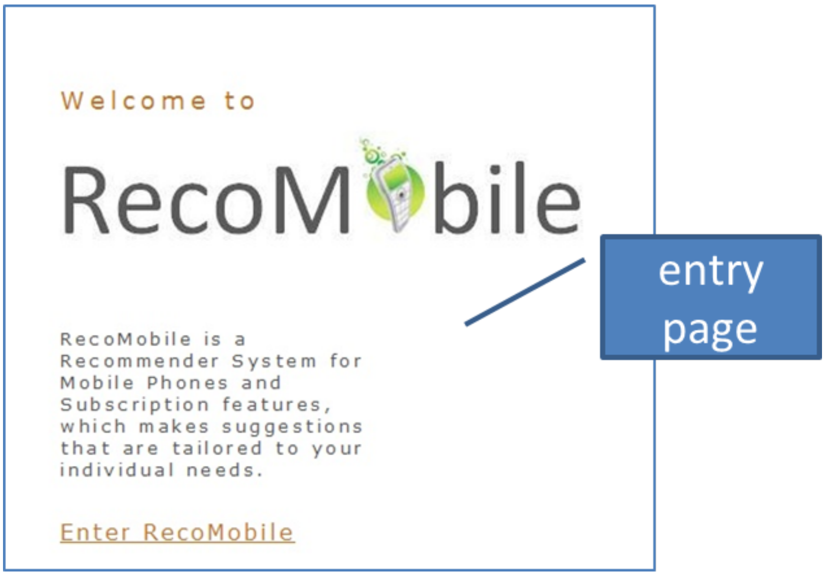

Figure 4.3.: RecoMobile Configurator – Entry Screen.

$= \{dom_0, dom_1, \ldots, dom_n\}$ represents a set of domains, where dom_i is assigned to the variable v_i. Finally, C $= C_{KB} \cup C_R$ represents a set of constraints, where $C_{KB} = \{c_0, c_1, \ldots, c_m\}$ represents the configuration knowledge base that restricts the possible combinations of values assigned to the variables in V and $C_R = \{r_0, r_1, \ldots, r_q\}$ represents a set of customer requirements.

A simple example of a configuration task is *V*=\{*styleReq, webUse, GPSReq, pModel, pStyle, pHSDPA, pGPS*\}, where *styleReq* expresses the user's preferred phone style, *webUse* specifies how often the user intends to access the Internet with the phone, and *GPSReq* specifies whether the user wants to use GPS navigation functionality. Table 4.7 specifies the existing phone models (*pModel*), their styles a.k.a. form factor (*pStyle*), whether the phone supports fast Internet access (*pHSDPA*), and whether the phone supports GPS navigation (*pGPS*). The respective domains are *D* = \{*dom(styleReq)* = \{*any, bar, clam*\}, dom(webUse) = \{*no, occasional, often*\}, dom(GPSReq) = \{*false, true*\}, dom(pModel) = \{*p1, p2, p3*\}, dom(pStyle) = \{*bar, clam*\}, dom(pHSDPA) = \{*false, true*\}, dom(pGPS) = \{*false, true*\}\}.

Table 4.7.: Available phone models in working example.

pModel	pStyle	pHDSPA	pGPS
p1	*bar*	*false*	*false*
p2	*clam*	*true*	*true*
p3	*clam*	*true*	*false*

Furthermore, we introduce a set of domain constraints $\mathrm{C_{KB}} = \{\mathrm{c_0}, \mathrm{c_1}, \mathrm{c_2}, \mathrm{c_3}\}$. Table 4.7 can be interpreted as a constraint in disjunctive normal form which yields $\mathrm{c_0}$. The remaining constraints represent the following domain properties:

- *c1: (webUse = often) → (pHSDPA=true) /* frequent web use requires a fast Internet connection */*

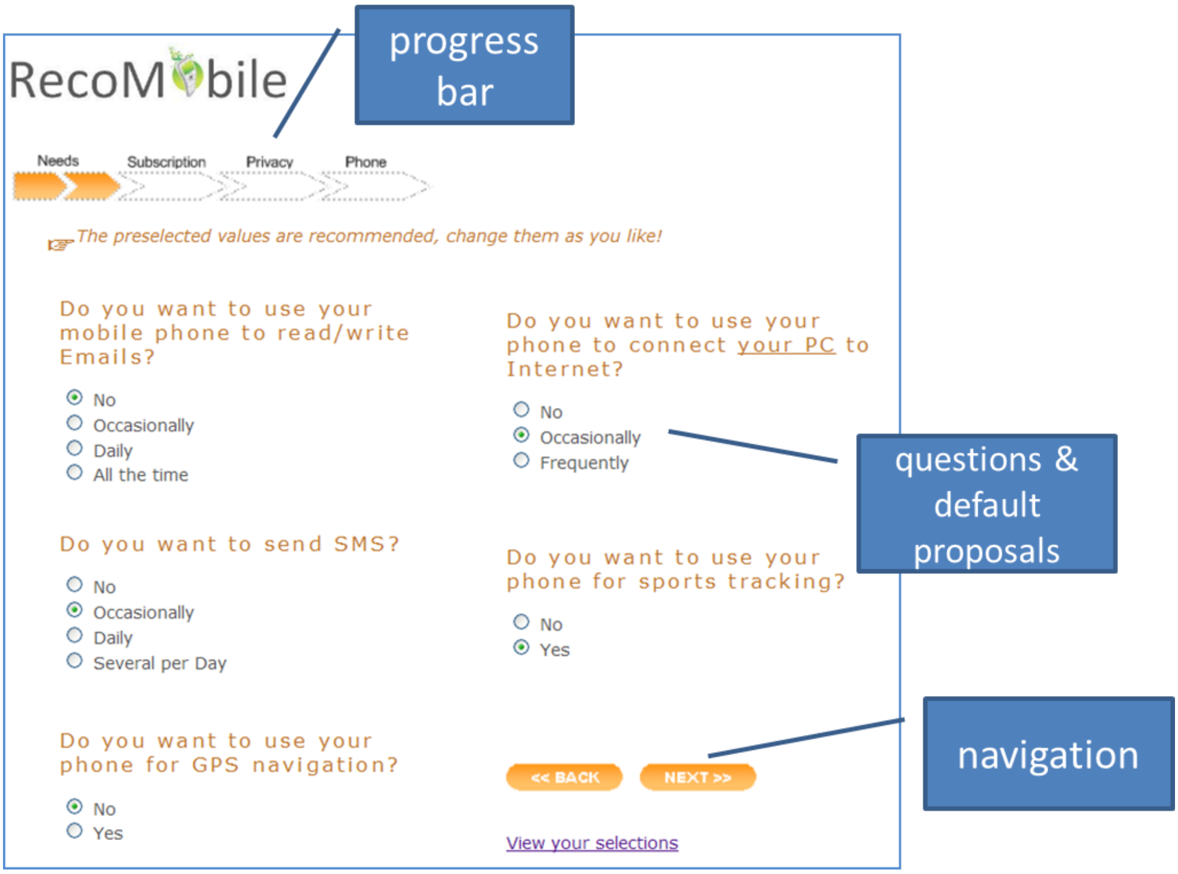

Figure 4.4.: Requirements specification in RECOMOBILE – personalized defaults are presented in order to proactively support users. These defaults are determined on the basis of the information from already completed recommendation sessions.

- *c2: (styleReq=any) OR (styleReq=pStyle) /* the phone should support the user's preferred phone style */*
- *c3: (GPSReq = true) → (pGPS = true) /* if GPS navigation is required, the phone must support it */*

Finally, an example of customer requirements is $C_R = \{r_0$:*styleReq=clam*, r_1:*webUse=often*, r_2:*GPSReq=false*$\}$.

On the basis of this definition of a configuration task we can now introduce the definition of a solution for a configuration task (also denoted as *configuration*) (Felfernig et al., 2010b):

Definition 4 (configuration): a solution (configuration) for a given configuration task (V, D, C) is represented by an instantiation $I = \{v_0 = i_0, v_1 = i_1, \ldots, v_n = i_n\}$, where $i_i \in dom_i$.

A configuration is *consistent*, if the assignments in I are consistent with the constraints in C. Furthermore, a configuration is *complete*, if all the variables in V have a concrete value. Finally, a configuration is *valid*, if it is both consistent and complete. An example of a valid configuration in our working example is: {*styleReq=clam, webUse=often, GPSReq=false, pModel= p3,*

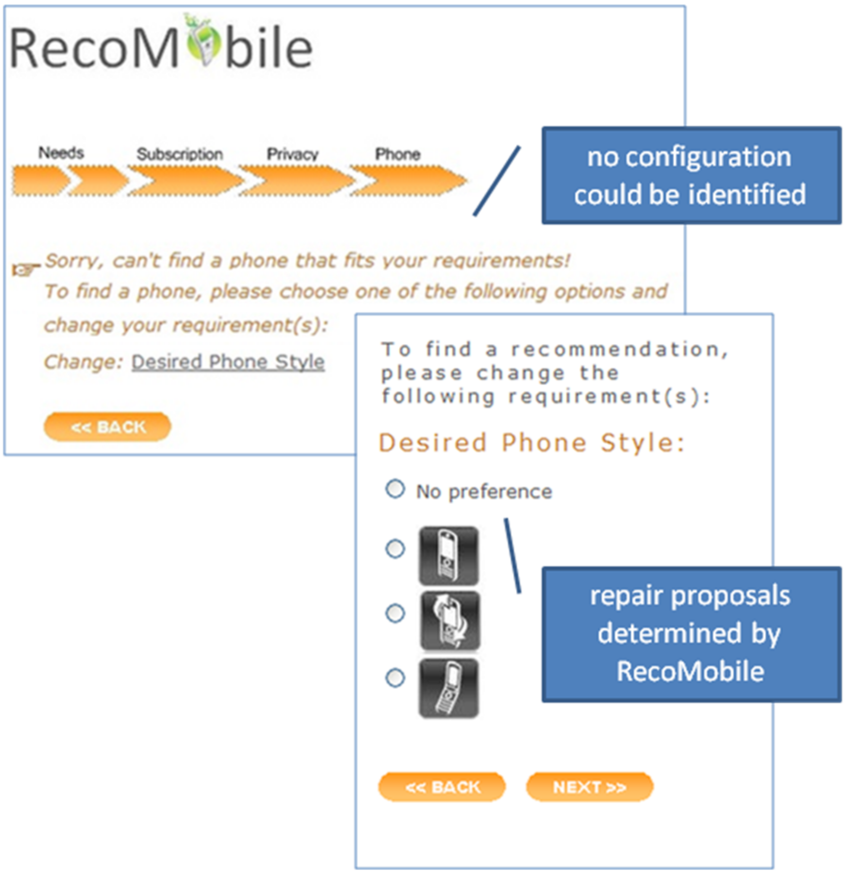

Figure 4.5.: Repair of inconsistent requirements – a repair component identifies minimal sets of changes such that the retrieval of at least one solution is possible.

pStyle=clam, pHDSPA=true, pGPS=false}.

Diagnosing inconsistent requirements

In situations where no configuration can be found for a given set of customer requirements, we have to activate a diagnosis functionality (Reiter, 1987; Felfernig et al., 2004, 2008, 2009). Let us assume the following set of customer requirements $\mathrm{C_R}$={$\mathrm{r_1}$:*styleReq=bar*, $\mathrm{r_2}$:*webUse=often*, $\mathrm{r_3}$:*GPSReq=true*}. The setting in $\mathrm{C_R}$ does not allow the calculation of a solution; consequently, we have to identify a minimal set of requirements that has to be changed in order to be able to restore consistency. We are interested in minimal changes since we want to keep the original set of requirements the same as much as possible. The calculation of a minimal set of requirements that has to be changed is based on the determination of conflict sets (Junker, 2004; Reiter, 1987). The following definition of a *conflict set* is based on (Junker, 2004):

Definition 5 (conflict set): A conflict set CS $\subseteq C_R$ is a subset of customer requirements, so that CS $\cup$ C_{KB} is inconsistent. A conflict set CS is *minimal*, if there does not exist a conflict CS' with CS' $\subset$ CS.

A diagnosis can be defined as follows (Felfernig et al., 2009):

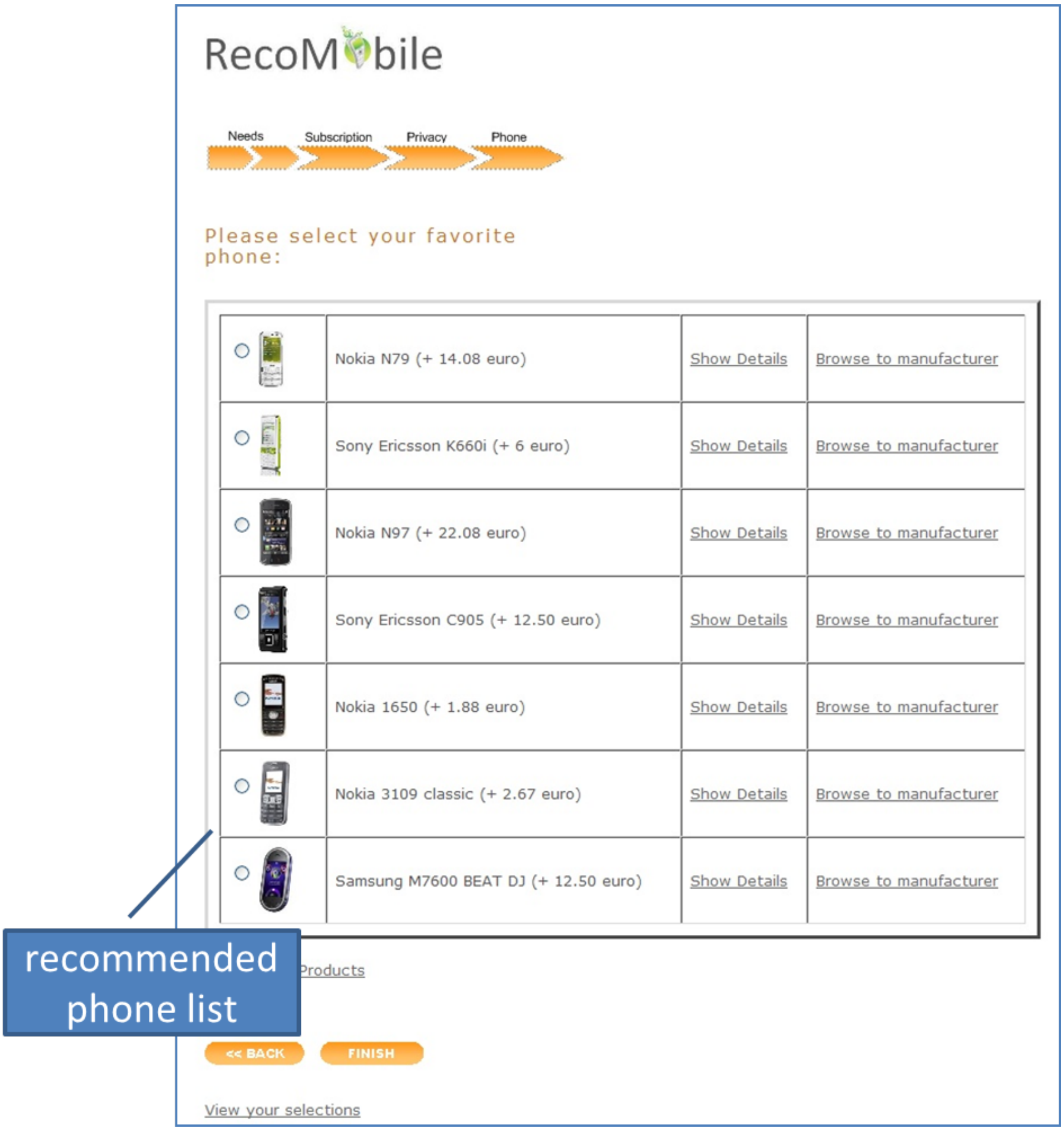

Figure 4.6.: Result presentation in RecoMobile – a set of phones that fulfill the specified requirements is presented to the user.

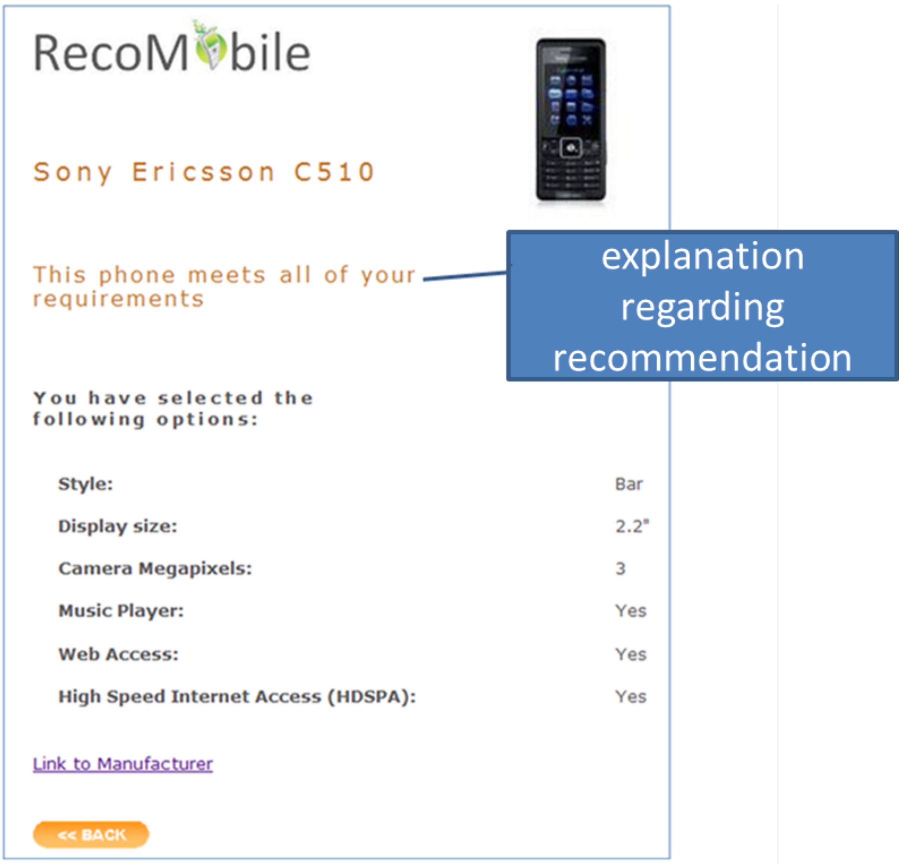

Figure 4.7.: A simple form of explanation – an enlisting of those user requirements which are fulfilled by a specific mobile phone.

Definition 6 (diagnosis): A diagnosis d $\subseteq C_R$ is a subset of customer requirements, so that C_R–d $\cup$ C_{KB} is consistent.

In our working example we can identify the two minimal conflict sets CS_1={r_1*:styleReq=bar*, r_2*:webUse=often*} and CS_2={r_1*:styleReq=bar*, r_3*:GPSReq=true*}. Both are conflict sets since {r_1, r_2} $\cup$ C_{KB} as well as {r_1, r_3} $\cup$ C_{KB} is inconsistent. Furthermore, both conflict sets are minimal since for both there does not exist a proper subset with the conflict set property. In order to restore consistency, we have to resolve each of the identified minimal conflict sets. A systematic way to this is to apply the concept of model-based diagnosis (Reiter, 1987). The core concept presented in (Reiter, 1987) is the *Hitting Set Directed Acyclic Graph* (HSDAG) algorithm, that is complete in the sense that all the existing diagnoses are found. The calculation of such a HSDAG is exemplified in Figure 4.8. The HSDAG algorithm assumes the existence of a corresponding conflict detection algorithm. In RecoMobile we apply a version of the *QuickXPlain* algorithm proposed by (Junker, 2004).

Let us assume that the conflict detection algorithm returns CS_1={r_1, r_2} as the first conflict set. Since the algorithm of (Junker, 2004) guarantees minimality, we can resolve each conflict (represented by a conflict set) by deleting exactly one of its elements. For example, to resolve the conflict represented by CS_1, we can either delete the requirement (constraint) r_1 or the requirement

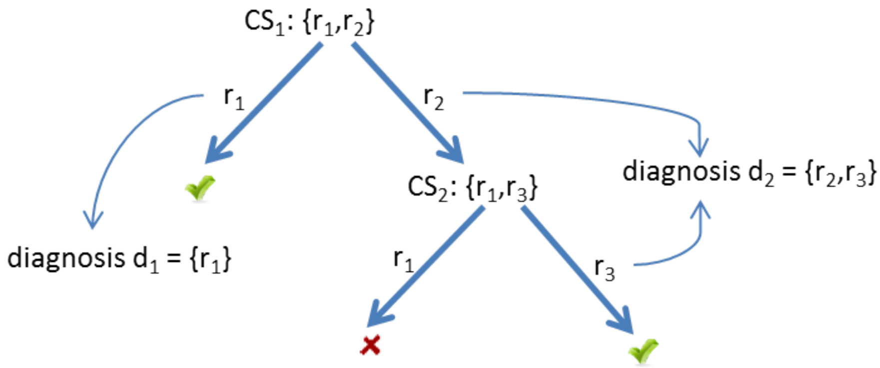

Figure 4.8.: Calculating diagnoses with the HSDAG algorithm (Reiter, 1987).

r_2. If we decide to delete r_1 from our conflict sets, no further conflict exists since we have deleted at least one element from each of the existing conflict sets. In this situation our conflict detection algorithm will not return further conflict sets and we have identified the first diagnosis which is $d_1=\{r_1\}$.

If we decide to delete r_2, the conflict detection algorithm will return $CS_2=\{r_1, r_3\}$ as the next conflict set, since the constraints (requirements) in $\{r_1, r_3\}$ are inconsistent with the given set of domain constraints $C_{KB} = \{c_0, c_1, c_2, c_3\}$. Again, we have two alternatives for resolving the conflict. If we choose r_1, we again have resolved all the existing conflicts, however $\{r_2, r_1\}$ would not be a minimal diagnosis since $\{r_1\}$ is already a diagnosis. Consequently, we do not have to further consider this path and can close it (no further minimal diagnoses will be found here) – such paths are labeled with $\times$ (see Figure 4.8). If we choose r_3, we have identified the second diagnosis which is $d_2=\{r_2, r_3\}$.

This way we are able to determine minimal sets of requirements that have to be changed in order to be able to find a solution – such alternatives are then displayed using the interface that is depicted in Figure 4.5.

4.2.2. Recommending Defaults

Beside the calculation of diagnoses in the case that no solution could be found, RecoMobile supports the *recommendation of feature values* (defaults). To calculate recommendations for feature values, valid configurations of previous sessions are stored in a database. On the basis of these configurations RecoMobile supports two basic algorithms: *Nearest Neighbors* (see (Tiihonen and Felfernig, 2010)) and *Naïve Bayes Voter* (Coster et al., 2002; Tiihonen and Felfernig, 2010). These algorithms are discussed in detail in the next sections.

Nearest Neighbor based feature value recommendation

The idea of Nearest Neighbor based feature recommendation is to find a configuration of a previous user which is closest to the active user's already specified requirements and to recommend feature values of that nearest neighbor for the remaining features. The distance between the already specified user requirements $conf_u$ and a neighbor configuration $conf_a$ is defined as the sum of individual distances (McSherry, 2003) between corresponding feature values $f_{i,u}$, and $f_{i,a}$, weighted by feature importance weights $w(f_i)$[3] (see Formula 4.5).

$$dist(conf_u, conf_a) = \sum_{f_i \in F_u} d_{f_i}(f_{i,u}, f_{i,a}) * w(f_i) \tag{4.5}$$

To calculate distances between feature values, RecoMobile applies the *Heterogeneous Value Difference Metric (HVDM)* from Wilson and Martinez (Wilson and Martinez, 1997) to cope with both symbolic and numeric features. The similarity of symbolic values in a domain is learned automatically. This is done by examining the probability that individual feature values contribute to the same classification of the samples - in our case classification of configurations. The higher the probability of a pair of feature values to be present in identically classified configurations, the more similar these feature values are considered (Wilson and Martinez, 1997).

A simple example of the application of the Nearest Neighbor based approach to the recommendation of feature values is the following. Table 4.8 contains three valid configurations *conf1*, *conf2*, and *conf3* from previous completed configuration sessions. Let us assume that the current user has already specified the requirements C_R={r_0:*styleReq=clam*, r_1:*webUse=often*}. Intuitively, the nearest neighbor for this combination of requirements is $conf_2$, since the feature values of *styleReq* and *webUse* are identical with the values specified in C_R. If we want to predict a value for the feature *GPSReq*, we would simply use the value specified in the configuration $conf_2$, which is *GPSReq=true*.

Naïve Bayes Voter based feature value recommendation

To calculate recommendations for individual feature values the Naïve Bayes Voter (Coster et al., 2002) applies the idea of the Bayes theorem (see Formula 4.6).

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)} \tag{4.6}$$

In the case of Naïve Bayes Voter, event A represents the fact that a configuration is the combination of feature values already specified by the user, and event B denotes the fact that a specific feature has value *v*. To simplify calculations the divisor P(A) is omitted in the case of Naïve Bayes voter, since P(A) is the same for all feature values that are compared. Thus, the predictor formula consists of two parts: a basic probability P(B) and a conditional probability P(A|B).

When determining the value to recommend for feature f_j, the predictor is determined for each possible value of the feature. The basic probability P(B) for value *v* of feature f_j is simply the proportion of configurations having a similar feature value.

[3]The weights for the RecoMobile features have been determined in a user study where participants had to estimate the importance of different phone features.

Table 4.8.: Example: valid configurations from previous sessions.

feature/configuration	**$conf_1$**	**$conf_2$**	**$conf_3$**	**$conf_u$**
f_1=styleReq	*bar*	*clam*	*clam*	*clam*
f_2=webUse	*often*	*often*	*occasional*	*occasional*
f_3=GPSReq	*false*	*true*	*false*	*true*
f_4=pStyle	*bar*	*clam*	*clam*	
f_5=pModel	*p1*	*p2*	*p3*	
f_6=pHDSPA	*false*	*true*	*true*	
f_7=pGPS	*false*	*true*	*false*	

Looking at Table 4.8, which contains 3 valid configurations from previous configuration sessions (referred to as *conf*) and a partial specified configuration of the active user ($conf_u$), the calculation of the basic probability for feature phone style (*pStyle*) for value *v=clam* results in P(pStyle=clam)=0.667 because 2 out of 3 previous users have chosen a phone of style *clam*.

The conditional probability part P(A|B) is the product of individual feature value probability estimates, which are calculated with formula 4.7.

$$m_{est}(N_c, N, p, m) = \frac{N_c + mp}{N + m} \tag{4.7}$$

N denotes the number of configurations in *conf* that have value *v* for feature f_j. These configurations are referred to as $conf_v$. N_c represents the number of configurations within $conf_v$ having an equal value as the user configuration *u* for feature f_i. *m* is the number of previous configurations and *p=1/m*. Applying this formula to our working example, we have to calculate the probability estimates for the three already specified features in context of f_4=clam. 2 out of 2 neighbors with *pStyle=clam* (*N*=2) have the same style requirement specification as our active user (f_1=clam) (N_c=2) which results in an m-estimate m_{est}(2, 2, 0.333, 3)=0.600. 1 out of 2 neighbors with *pStyle=clam* has specified the same value for the web use requirement as our active user (f_2=occasional) which results in an m-estimate m_{est}(1, 2, 0.333, 3)=0.400. For the GPS requirement we again have m_{est}(1, 2, 0.333, 3)=0.400 (1 out of 2 neighbors has specified the same value as the active user). Therefore the predictor pr(f_4,clam)= 0.667*(0.600*0.400*0.400) = 0.0640. Since the predictor for *pStyle=clam* is highest and consistent with already specified feature values of the active user, this value is recommended to the user.

Similarity-based ranking of phones

When the specified user requirements lead to a large number of possible solution rankings, strategies are needed to present the results to the user in a convenient way. Research has shown that users are more likely to explore items that appear early in the results ranking – this effect is known as *primacy effect* (Asch, 1949; Felfernig et al., 2007; Murphy et al., 2006). There exists a number of literature which explains different result ranking approaches – see, for example, (Felfernig et al., 2008, 2010b; Lai et al., 2006). In RecoMobile we follow a collaborative similarity-based approach. We determine the distance of each previous configuration to the user's current configuration, so that phones from nearest configurations are shown first. Phones that are compatible

Table 4.9.: Overview of hypotheses (H_1...H_9).

ID	Hypothesis
H_1	*personalized configurators increase the* ***confidence*** *of a user in the product decision*
H_2	*users of personalized configurators are more* ***satisfied*** *with the* ***configuration process quality***
H_3	*personalized configurators increase a user's* ***trust*** *in the presented configuration solution*
H_4	*personalized configurators* ***better support*** *users in finding the best options*
H_5	*the* ***probability of reusing*** *the configurator is higher with personalized versions*
H_6	*the* ***probability of recommending*** *the personalized configurator to other users is higher*
H_7	*a user's* ***expectations regarding the solution*** *are better fulfilled with personalized versions*
H_8	*personalized configurators trigger a higher* ***purchase probability*** *than non-personalized ones*
H_9	*the average* ***interaction time*** *per page is lower with personalized versions*

with user requirements are presented to the user. For example, if the customer requirements are C_R={r_0:*styleReq=clam*, r_1:*webUse=occasional*, r_2:*GPSReq = false*}, phones of configurations $conf_2$ and $conf_3$ meet the customer requirements. Since $conf_3$ is most similar to the user configuration, the corresponding phone p_3 is ranked first. p_1 cannot be recommended because its style does not match customer requirements.

After this introduction to the basic technologies integrated in the RecoMobile configurator, we will now focus on our empirical study. This study has been conducted with the goal to show the advantages of applying recommendation technologies in knowledge-based configuration settings.

4.2.3. User Study

In our evaluation we compare a personalized version of the RecoMobile configurator (feature recommendation supported – Version B) with a version that does not include feature recommendation (Version A). We focus on a comparison of Naïve Bayes voter based feature value recommendation with a corresponding non-personalized version. To compare these two versions we defined the hypotheses shown in Table 4.9. Note that the calculation of recommendations for feature settings in Version B was based on the data collected by configurator Version A. We used a user-centric evaluation framework and evaluated our hypotheses on the basis of a dataset collected within an online survey. This online survey was conducted at two Austrian universities (Graz and Klagenfurt), at the University of Bolzano, and the Helsinki University of Technology. N=546 subjects participated in the study (32.4% female and 67,6% male). In the scenario of the study, the participants had to decide which mobile phone (including the corresponding services) they would select. Each participant was assigned to one of the two configurator versions with the task to identify a mobile phone solution and to place a fictitious order.

We collected both implicit and explicit user feedback. After interacting with the configurator, the participants had to fill out a questionnaire (answers provided on a 10-point Likert scale) that helped us to evaluate the hypotheses {H_1, H_2, ..., H_8}. The questionnaire is based on an evaluation framework developed by Pu and Chen (Pu and Chen, 2010), that helps to measure users' subjective attitudes based on their recommender experience. Hypothesis H_9 was evaluated directly on the basis of session interaction data (session length).

4.2.4. Empirical Evaluation

The major results of evaluating the hypothesis are summarized in Table 4.10. For the p-values used in our analysis we applied a corresponding independent two-sample t-test which is supported by the given normal-distribution of the variables. The participants of the study were informed that they are participating in a "lottery drawing" if they provide their email address.

Table 4.10.: Evaluation results for hypotheses H_1...H_9 - Independent t-test. Significant differences are highlighted with bold typeface

ID	**non-pers. (Version A)**	**pers. (Version B)**	**significance**
H_1	5.30(2.92)	5.88(2.75)	$p \leq 0.15$
H_2	5.56(1.98)	6.52(1.97)	$\mathbf{p \leq 0.01}$
H_3	4.83(2.56)	5.38(2.63)	$p \leq 0.14$
H_4	5.0(2.66)	5.76(2.44)	$\mathbf{p \leq 0.05}$
H_5	4.37(3.09)	5.16(3.05)	$p \leq 0.09$
H_6	4.28(3.03)	5.11(2.98)	$p \leq 0.08$
H_7	4.65(2.31)	5.76(2.79)	$\mathbf{p \leq 0.05}$
H_8	4.30(3.43)	5.27(2.99)	$\mathbf{p \leq 0.05}$
H_9	3.27 min. (2.07)	3.07 min.(1.19)	$p \leq 0.25$

Regarding *user confidence*, hypothesis H_1 cannot be confirmed (see Table 4.10). Regarding the *overall quality of the configuration process* (H_2) users are significantly ($p \leq 0.01$) more satisfied with a configurator which supports recommendations for feature values. Such recommendations can provide support in situations where users are not sure about which value to select. In addition, they actively help users to keep partial configurations consistent, i.e., the probability of being confronted with repair situations is reduced. Regarding user's *trust* in the configurator, hypothesis H_3 cannot be confirmed on the basis of our statistical tests. Personalized configurators significantly better *support* users in finding the best options ($p \leq 0.05$, H_4). This effect can be explained by active user support provided by the automated generation of recommendations for feature settings. In addition, there is a tendency that a user's *willingness to reuse the configurator* is higher (H_5), if he or she used the personalized version ($p \leq 0.09$), and users of personalized configurators have a higher probability of *recommending the configurator application to other users* ($p \leq 0.08$, H_6). Personalized configurators are significantly better fulfilling a user's *expectations regarding the presented result* ($p \leq 0.05$, H_7), which indicates higher-quality configurations that have been found due to more active user support. Furthermore, personalized configurators are triggering a higher *purchase probability* ($p \leq 0.05$, H_8). Contrary to hypothesis H_9, there is no real tendency that *configuration sessions with personalized configurators are shorter.*

4.2.5. Related Work

Main-stream recommender applications are based on collaborative filtering (Konstan et al., 1997) and content-based filtering (Pazzani, 1999) approaches. These approaches are predominantly applied to quality and taste products – a very well known example is Amazon.com (Linden et al., 2003). The application of pure collaborative or content-based recommendation is the exception of the rule – in many cases only hybrid approaches can solve problems such as the ramp-up problem (for example, for a new user the recommender system does not dispose of rating data which makes the calculation of initial recommendations a challenging task). A discussion of this and further issues regarding the deployment of recommenders can be found in (Burke, 2002a).

Configuration systems have a long and successful history in the area of Artificial Intelligence (Barker et al., 1989; Fleischanderl et al., 1998; Mittal and Frayman, 1989; Sabin and Weigel, 1998). Although these systems support interactive decision processes with the goal to determine configurations that are useful for the customer, the integration of personalization technologies has been ignored with only a few exceptions – see, for example, (Coster et al., 2002; Geneste and Ruet, 2001). The RecoMobile configurator presented in this section has been developed and evaluated on the basis of the configuration concepts introduced in (Tiihonen and Felfernig, 2010). The goal of the work presented here was to implement and evaluate a system that integrates recommendation technologies that actively support users in a product configuration process.

The integration of recommendation technologies with knowledge-based configuration is still in a very early stage. Most of the existing commercial configuration environments are lacking of recommendation functionalities – the study presented in this chapter points out potentials for improvements. There exist some contributions that take into account the application of personalization technologies in the configuration context. (Geneste and Ruet, 2001) introduce an approach to the integration of case-based reasoning methods (Kolodner, 1993; McSherry, 2003; Smyth and Keane, 1996) with constraint solving (Junker, 2004) with the goal to adapt nearest neighbors identified for the current problem. There exist a couple of approaches that are similar to Geneste and Ruet (Geneste and Ruet, 2001) – see, for example, (Coster et al., 2002). All of those approaches do not provide a clear concept for enabling minimal changes and handling inconsistent feature value recommendations.

McSherry (McSherry, 2005) presents an approach to apply defaults in the preference elicitation process by integrating the idea that there are product attributes whose values most users would prefer to maximise or minimise. They called such attributes *more-is-better* (MIB) or *less-is-better* (LIB) attributes. The results of this research suggest that *default preferences* have the potential to dramatically increase recommendation efficiency (McSherry, 2005). Contrary to the RecoMobile configurator, there is no personalization aspect in this approach. For every user, the preference values of product attributes are automatically set to the lowest value of the attribute for an LIB attribute, and to the highest value of the attribute for a MIB-attribute (McSherry, 2005).

The comparison of recommender algorithms is often based on offline metrics using techniques from machine learning and information retrieval such as precision and recall (Shani and Gunawardana, 2009). A major problem of this evaluation type is that the performance of a recommender algorithm is evaluated on existing datasets. The authors of (Hayes et al., 2002) argue that "*it* *would be a mistake to assume that domain specific datasets such as those from the EachMovie and MovieLens projects can be used to test every Recommendation System*" (Hayes et al., 2002).

Another limitation of this approach is that the recommendation problem is reduced to the problem of calculating ratings for items the user has not seen so far. In recent years measuring the user experience during the recommendation process has increasingly gained importance in the evaluation process (Hayes et al., 2002; Pu and Chen, 2010; McNee et al., 2006). Hayes et al. suggest that "*user satisfaction with a recommendation strategy can only be measured in an on-line context*" (Hayes et al., 2002). They propose an online evaluation framework for pairwise comparison of recommender algorithms.

Pu and Chen developed an evaluation framework for recommender systems that aims at measuring the perceived qualities of a recommender and the influence of these qualities on users' behavioral intentions (Pu and Chen, 2010). We adopted this framework to measure users' subjective attitudes based on their experience in the context of a product configuration dialog where personalized value recommendations are presented to the users.

4.2.6. Conclusion

In this section we provided an overview of basic recommendation techniques that can be used in the context of configuring complex products and services. In our evaluation we compared a personalized version of the RecoMobile configurator (feature recommendation supported), with a version that does not include feature recommendation. The results of our study indicate that the integration of recommendation techniques in a product configuration system show to be useful in terms of improving the user acceptance of the configurator interface.

4.3. Summary

In this chapter we presented strategies to improve the recommendation efficiency in intelligent selling applications. We applied different evaluation techniques to demonstrate the applicability of our approaches.

In Section 4.1 we introduced a new unit critiquing approach which exploits the information of successfully completed critiquing sessions to find an item that best matches the current user's critiques. We combined our nearest neighbor compatibility approach with the conventional critiquing algorithm (Burke et al., 1996) and the experience-based approach (McCarthy et al., 2010) to ensemble-based critiquing variations and conducted an offline experiment to compare the performance of the different critiquing strategies. The results of our experiment indicate that our new nearest neighbor compatibility critiquing approach as well as ensemble-based variations thereof have the potential to reduce the number of critiquing cycles in critiquing sessions.

In Section 4.2 we presented an approach to the recommendation of feature values in the context of interactive product configuration dialogs. We introduced the RecoMobile configuration environment that supports the configuring of mobile phone packages (phone and corresponding subscription features). Users of RecoMobile are pro-actively supported by personalized defaults and minimal sets of changes in the case that no solution could be found. This application has been evaluated within the scope of an empirical online study. The results of this study show that our personalization approach allows significant improvements in the perceived quality of product configuration dialogs.

Chapter 5

Evaluating the impact of feature recommendations on decision making

Parts of the contents of this chapter have been published in (Mandl et al., 2011b,c).

In the psychological literature there exist a couple of theories that explain the existence of different types of decision biases in a consumer decision making situation. Consumers are influenced by the format of the information presented and as a consequence use different decision-making strategies in different contexts (see, for example, (Asch, 1949; Payne, 1976; Lussier and Olshavsky, 1979; Tversky and Kahneman, 1981; Bettman et al., 1991)). This implies that the design of the user interface can have a major impact on the final outcome of the decision making process. Furthermore, research has shown that system properties such as interface display techniques or interaction models, may have an effect on users' subjective attitudes toward the acceptance of recommender technologies (Eroglu, 2001; Eroglu et al., 2003; Chen and Pu, 2005). Therefore we must integrate recommendation technologies with deep knowledge about human decision making to improve the applicability of intelligent selling applications.

In this chapter we provide results of our research that aimed at analyzing different factors in the context of feature recommendations that may effect consumers' decision making. The online experiment described in Section 5.1 investigates the impact of *different presentation styles* of personalized default recommendations on the systems' subjective quality. In Section 5.2 we describe an empirical study which explores the impact of the *status quo bias* on human decision making in the context of a product configuration session. This bias deals with the fact that consumers are more likely to select an alternative that is designated as the status quo option (Samuelson and Zeckhauser, 1988; Kahneman et al., 1991; Ritov and Baron, 1992; Bostrom and Ord, 2006) – in our research represented by feature value recommendations.

5.1. Design Alternatives for Feature Recommendations

Research on consumer decision making has shown that the format of the information presented can have an impact on customers' decision-making strategies (see, for example, (Asch, 1949; Bettman et al., 1991; Payne, 1976; Bettman and Kakkar, 1977)). For example, the results of a study conducted by Asch (Asch, 1949) illustrate the importance of an item's position in an ordered list. The experiment demonstrated that when presenting adjectives describing a person in sequence, the same words could result in very different ratings of that person depending on the order in which the words were presented (Asch, 1949). This effect is known as *primacy effect* (Crowder, 1976; Murphy et al., 2006).

The format of information presentation as well as elements of user interface design can have a significant impact on user attitudes and perceptions of the trustworthiness of a system (see, for example, (Kim and Moon, 1998; Roy et al., 2001; Eroglu, 2001; Eroglu et al., 2003)). Eroglu et al. (Eroglu, 2001; Eroglu et al., 2003) examined the impact of task-relevant cues such as the description or pictures of the product, as well as system design elements, such as color, background, and fonts, on the outcomes of online shopping. They suggest that both environmental cues and design elements influence a consumer's *willingness* to make a buying decision (Eroglu, 2001; Eroglu et al., 2003). The results of a study conducted by Kim and Moon (Kim and Moon, 1998) showed that the manipulation of user interface design elements, such as title, menu, main clipart and color, can have an impact on users' *trust* in an electronic commerce system. Roy et al. (Roy et al., 2001) revealed a strong relationship between quality or usability of a system and the level of trust of customers in the system. Trust has been demonstrated as key factor to the success of e-commerce (Jarvenpaa et al., 2000; Grabner-Kräuter and Kaluscha, 2003; Chen and Pu, 2005). Empirical research has shown that the perceived trustworthiness of a system can have an influence on consumer's *purchase intention* as well as on the *intention to use the system again* (Jarvenpaa et al., 2000).

In this section we discuss the impact of different styles to present default values in product configurators. We present results of a case study that investigated whether different ways to represent defaults have an influence on the *users' acceptance* of proposed defaults, as well as on *subjective evaluations* of the product configurator. The evaluation criteria include one objective measure (number of accepted defaults) and subjective measures related to *satisfaction*, *trust*, *confidence*, and *intentions of behavior*.

5.1.1. User Study

We conducted an online study with the goal to compare different presentation styles for feature recommendations (defaults) in a product configuration system. For this purpose we used the RecoMobile prototype, a knowledge-based configuration system for mobile phones and services enriched with recommendation functionalities to predict useful feature settings for the user (see Section 4.2). Example pages of RecoMobile are depicted in Figures 4.4, 4.5, 4.6, 4.7.

We implemented three different versions of the RecoMobile user interface (see Table 5.1). These three versions differ in the way the defaults are displayed to the user, as well as in the

Table 5.1.: RecoMobile - Configurator versions in user study.

Version	Default Presentation	Explanation
A	Defaults without confirmation	Defaults are preselected by the system – acceptance of defaults does not require additional interaction (see Figure 5.1)
B	Defaults with confirmation	Defaults are preselected by the system – acceptance of defaults requires additional interaction in terms of a confirmation (see Figure 5.2)
C	Defaults to select	Defaults are displayed with a hint – acceptance of defaults requires additional interaction in terms of a selection (see Figure 5.3)

extent to which user interaction is required to select/reject the provided default suggestions. In Version A the recommended feature alternative is preselected by the system (see Figure 5.1). User interaction is only required to select a different alternative than the recommended one (low user involvement). In Version B the preselected feature recommenations are endowed with an "Accept-Button" so that both, acceptance and changing the alternative require user interaction (high user involvement - see Figure 5.2). In Version C the recommended feature alternative is displayed with a hint (see Figure 5.3), and the user has to select the desired feature alternative - even if the user wants to accept the recommendation.

Versions B and C are derived from the research work of Ritov and Baron (Ritov and Baron, 1992). They indicate that people have a strong tendency to accept preset values (known as *status quo bias*) compared to other alternatives (Ritov and Baron, 1992). In Section 5.2 we present the results of a online study that revealed a strong biasing effect even if uncommon values are presented as default options (the status quo options) (Mandl et al., 2011b). Therefore, a major risk of this status quo effect is that defaults could be exploited for misleading users, and making them to choose options that are, for example, not really needed to fullfill their requirements. Ritov and Baron (Ritov and Baron, 1992) suggest to present the options in such a way, that keeping as well as changing the status quo needs user input. They argue that "*when both keeping and changing the status quo require action, people will be less inclined to err by favoring the status quo when it is worse*" (Ritov and Baron, 1992).

5.1.2. Study Design

Our experiment addressed two relevant questions. (1) *Does the manipulation of the presentation style of defaults have an impact on users' perception of the configuration system*? (2) *Do different default representation styles have an influence on users' willingness to accept/select defaults*?

To address these questions, we conducted an online survey at the University of Klagenfurt. N=129 subjects participated in the study. Each participant was assigned to one of the three user interface versions of the configurator (see Table 5.1). The experiment was based on a scenario where the participants had to decide which mobile phone (including the corresponding services)

Figure 5.1.: The low user involvement version of RecoMobile - no interaction is required if users want to accept the provided defaults.

they would select. A post-study questionnaire was designed covering 8 subjective measures (see Table 5.2). The questionnaire is based on an evaluation framework developed by Pu and Chen (Pu and Chen, 2010). Each question had to be answered on a 11-point Likert scale.

The questionnaire is based on an evaluation framework developed by Pu and Chen (Pu and Chen, 2010), that helps to measure users' subjective attitudes based on their recommender experience

5.1.3. Evaluation

In the following we discuss the influence of different representation styles of defaults on users' perception of the configuration system, as well as on users' selection behavior.

Objective Measure

The objective measure aims at analyzing the default selection behavior in the different product configurator versions. The average number of defaults selected or accepted by users of different configurator versions are shown in Table 5.3. The results show that there is no significant difference between the configurator versions. Users of Versions A and B accepted on average roughly 8 out

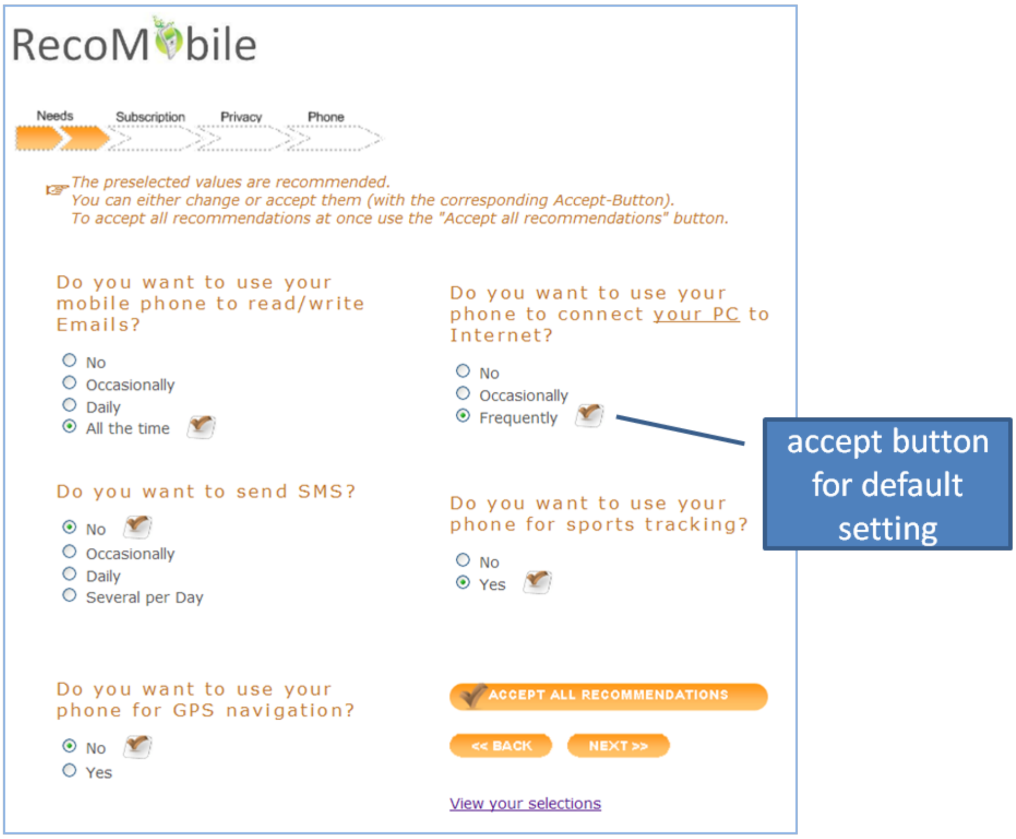

Figure 5.2.: The high user involvement version of RecoMobile - users of this version had to accept the provided defaults.

of the 15 recommended feature options (Version A - mean=8.02, Version B - mean=7.91). Users of Version C selected on average roughly 6 default proposals (Version C - mean=6.43).

Subjective Measures

The results of the evaluation of the subjective measures are shown in Tables 5.4 and 5.5. The average scores of user evaluations of the subjective measurements are shown in Figure 5.4.

An independent t-test was conducted to find significant differences between the configurator versions. Table 5.5 presents a comparison of users' evaluations of configurator versions with different styles of presenting defaults (significant differences are highlighted with bold typeface). The different methods of presenting recommendations have a significant impact on *perceived quality* of the system. Users of configurator Version C gave significantly lower scores (mean=4.41, Standard Deviation SD=3.135) than users of Version A (mean=6.30, SD=1.896), and B (mean=6.03, SD=2.467). The comparison on *confidence* shows significant differences (t=2.050, p=0.044) between users of Version B (mean=6.08, SD=2.882), who rated highest, and users of Version C (rated lowest - mean=4.68, SD=2.857). As for *trust* in the recommendations given by the system, there is a significant difference (t=2.558, p=0.013) between the rating of Version A (mean=5.47, SD=2.063), and Version C (mean=3.94, SD=2.628).

Table 5.2.: Evaluation questions.

Measurements	Questions
Satisfaction with quality	How satisfied were you with the overall quality of the selection process?
Confidence	How confident are you in having selected the most suitable phone and subscription?
Trust	How high is your degree of trust in the recommendations given by the system?
Use intention	How high is the probability that you would use the system again?
Reference intention	How high is the probability that you would recommend the system to another user?
Purchase intention	Assume that you need a new phone. How high is the probability that you would purchase the selected mobile phone?
Satisfaction with user interface	How satisfied were you with the presentation of feature recommendations?

Table 5.3.: Comparison of default selection behavior in different configurator versions. Mean=average number of defaults selected or accepted by users of corresponding configurator version.

Version	Mean	Std. Deviation
A	8.02	4.078
B	7.91	3.676
C	6.43	3.749

Figure 5.3.: Version C of the RecoMobile user interface – the user selects the desired alternative, even in the case of accepting the default proposal.

The comparison of the other measures (use intention, reference intention, purchase intention and satisfaction with user interface) revealed no further significant differences between the different user interface versions.

Version C was rated lowest in almost all subjective measures, except the *purchase intention*. Version B has a slightly lower mean value in this case (mean=3.72, SD=3.177), compared to Version C (mean=3.85, SD=3.076).

Relationships between objective and subjective variables

We wanted to identify potential correlations between the number of selected/accepted defaults and users' subjective perceptions of the configurator. Therefore we calculated correlations between the objective measure and each subjective measure. Table 5.6 presents the coefficient values by Pearson's Correlation. The results show that the number of selected/accepted defaults is significantly positively correlated with the perceived user interface quality. This suggests that users' willingness to accept defaults depends not only on the quality of the recommendations, but also on the *presentation style of defaults*.

Table 5.4.: Subjective Measures - Evaluation results. Mean=average rating of evaluation questions (see Table 5.2) by users of corresponding configurator version. SD=Standard Deviation.

	Version A		Version B		Version C	
Measurement	Mean	SD	Mean	SD	Mean	SD
Satisfaction with Quality	6.30	1.896	6.03	2.467	4.41	3.135
Confidence	5.70	2.322	6.08	2.882	4.68	2.857
Trust	5.47	2.063	4.94	2.366	3.94	2.628
Use intention	4.30	3.344	5.11	3.328	4.12	3.121
Reference intention	4.47	2.980	5.08	3.255	4.06	3.219
Purchase intention	5.17	3.485	3.72	3.177	3.85	3.076
Satisfaction with User Interface	5.23	3.081	4.50	3.185	4.18	3.451

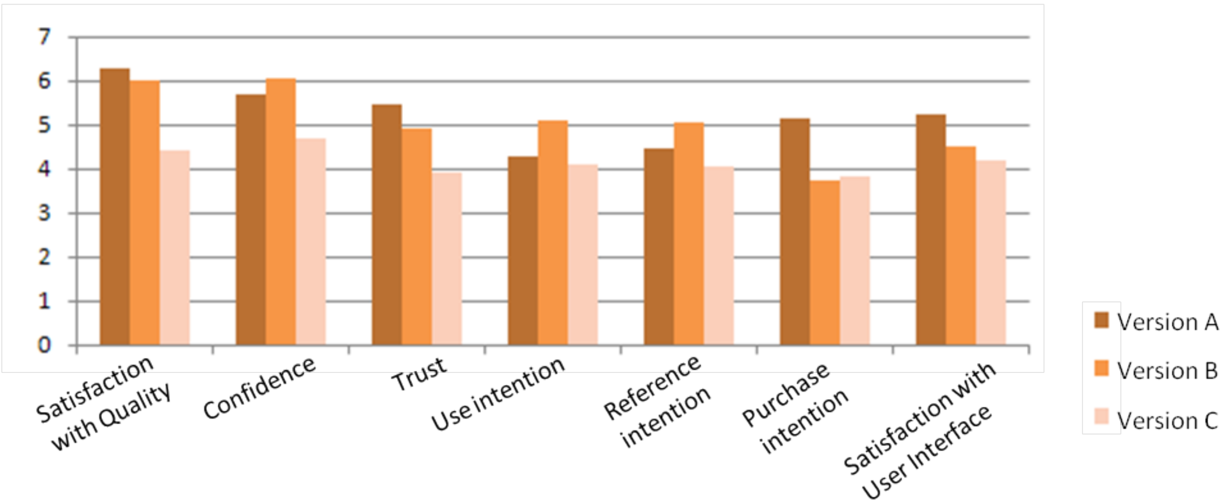

Figure 5.4.: Comparison of subjective evaluation results between different RecoMobile user interface versions - Mean Values (see Table 5.4).

Table 5.5.: Comparison of evaluation results of different configurator versions - Independent t-test. Significant differences are highlighted with bold typeface.

Measurement	Version A compared with Version B	Version A compared with Version C	Version B compared with Version C
Satisfaction with Quality	p=0.623	**p=0.005**	**p=0.020**
Confidence	p=0.559	p=0.119	**p=0.044**
Trust	p=0.348	**p=0.013**	p=0.097
Use intention	p=0.329	p=0.822	p=0.203
Reference intention	p=0.429	p=0.602	p=0.190
Purchase intention	p=0.083	p=0.114	p=0.862
Satisfaction with User Interface	p=0.348	p=0.204	p=0.685

Table 5.6.: Correlations between objective and subjective variables - Pearson's Correlation.

	Number of selected defaults
Satisfaction with Quality	-0.054 (p=0.591)
Confidence	0.027 (p=0.793)
Trust	-0.079 (p=0.437)
Use intention	-0.148 (p=0.142)
Reference intention	-0.066 (p=0.512)
Purchase intention	-0.050 (p=0.620)
Satisfaction with User Interface	**0.265 (p=0.008)**

5.1.4. Related Work

Product configuration systems are increasingly being used by manufacturing companies to assist customers in specifying their requirements and to find a product that matches their preferences. One major problem of configuration systems is the high diversity of offered products and product attributes. Previous research has shown that consumers are often overwhelmed in high variety categories because of the large amount of options to evaluate (Malhotra, 1982). Since humans have limited processing capacity (Streufert and Driver, 1965; Bettman, 1979), confronting consumers with too much information can lead to an information overload and therefore can result in decreased quality of decision performance (Jacoby et al., 1974). A possibility to overcome this problem in configuration systems is to personalize the system's user interface by providing feature recommendations matched to customer preferences (Felfernig et al., 2010b). The results of a study conducted by Felfernig et al. (Felfernig et al., 2010b) show a higher user satisfaction with the configuration process when personalized defaults are provided, compared to a non-personalized configurator version (for more details, see Section 4.2).

Although product configuration systems support interactive decision processes with the goal to determine configurations that are useful for the customer, the integration of human decision psychology aspects has been ignored with only a few exceptions. Human choice processes within a product configuration task have been investigated by, for example, Kurniawan, So, and Tseng (Kurniawan et al., 2006). They conducted a study to compare product configuration tasks (choice of product attributes) with product selection tasks (choice of product alternatives). Their results suggest that *configuring* products, instead of *selecting* products, can increase customer satisfaction with the shopping process (Kurniawan et al., 2006).

The research of (Kamali and Loker, 2002) and (Huffman and Kahn, 1998) was aimed at investigating the influences of user involvement on consumer satisfaction in a configuration environment. Kamali and Loker (Kamali and Loker, 2002) explored consumer involvement in the customization process by creating three levels of interactive design involvement. The results of their research revealed a higher consumer satisfaction with the website's navigation as user involvement increased. Huffman and Kahn (Huffman and Kahn, 1998) conducted a study to explore the relationship between the number of choices during product configuration, and the user satisfaction with the configuration process. Their results showed that both, the way the information is presented, and the type of customer input to the information gathering process, influence customer satisfaction when being confronted with many choice alternatives. From their results they concluded that customers might be overwhelmed when being confronted with too many choices (Huffman and Kahn, 1998).

McNee et al. (McNee et al., 2003) explored user interfaces that integrate users in the selection of items that are used to develop the initial user model. From their experiments they conclude that such interfaces can affect user loyalty in a positive way (McNee et al., 2003).

5.1.5. Conclusion

We presented the results of an empirical study that analyzed the impact of different presentation styles of personalized recommendations in product configuration scenarios. For this purpose we implemented three different user interface versions for the RecoMobile prototype, a mobile phone and subscription configurator system. The versions differed in the way the recommendations were displayed to the user and in the extent of user interaction required to accept/reject the suggestions.

The results show that the method of presenting feature recommendations can have a significant impact on *users' satisfaction* with the overall perceived quality of the selection process as well as on *users' confidence* and *trust* in the product configurator. Furthermore, we found a strong correlation between the number of selected or accepted recommendations and users' satisfaction with the presentation of feature recommendations. Our results suggest that the method of presenting personalized defaults is an important factor that influences users' satisfaction with the configuration process.

5.2. Status Quo Bias

A number of research in the field of human decision making have revealed that people have a strong tendency to keep the status quo when choosing among alternatives (see, for example, (Samuelson and Zeckhauser, 1988; Kahneman et al., 1991; Ritov and Baron, 1992; Bostrom and Ord, 2006)). Samuelson and Zeckhauser (Samuelson and Zeckhauser, 1988) explored this effect, known as *status quo bias*, in a series of labor experiments. Their results implied that an alternative was significantly more often chosen when it was designated as the status quo. They also showed that the *status quo effect* increases with the number of alternatives.

Kahnemann, Knetsch and Thaler (Kahneman et al., 1991) argue that the *status quo bias* can be explained by a notion of *loss aversion*. They explain that the status quo serves as a neutral reference point, and users evaluate alternative options in terms of gains and losses relative to the reference point (Kahneman et al., 1991). Since individuals tend to regard losses as more important than gains in decision making under risk (i.e., alternatives with uncertain outcomes) (Kahneman and Tversky, 1979) the possible disadvantages when changing the status quo appear larger than possible advantages.

A major risk of preset values is that they could be exploited for misleading users and making them to choose options that are not really needed to fulfill their requirements. Bostrom and Ord defined the status quo bias as "*a cognitive error, where one option is incorrectly judged to be better than another because it represents the status quo*" (Bostrom and Ord, 2006). Ritov and Barron (Ritov and Baron, 1992) suggest counteracting the status quo bias by presenting the options in such a way, that keeping as well as changing the status quo needs user input. They argue that "*when both keeping and changing the status quo require action, people will be less inclined to err by favoring the status quo when whether is worse*" (Ritov and Baron, 1992).

In this section, we want to focus on answering the question whether a status quo bias exists in the context of product configuration systems, and if it is possible to reduce this biasing effect by providing an interface supporting such an interaction type described by Ritov and Barron (Ritov and Baron, 1992).

5.2.1. User Study

One major problem of configurator systems is the high diversity of offered products. Users are often overwhelmed by the complexity of the alternatives, a phenomenon well known as *mass confusion* (Huffman and Kahn, 1998). A possibility to help the user identifying meaningful alternatives that are compatible with their current preferences is to provide defaults. Defaults in the context of interactive configuration dialogs are *preselected options used to express personalized feature recommendations* (Mandl et al., 2011b).

A major risk of defaults is that they can cause a *Status Quo Effect* and therefore make users choose options that are not really needed to fulfill their requirements. In the following we present a study where we used the RecoMobile prototype (see Section 4.2) to explore whether there exists a status quo bias in the context of product configurations systems. RecoMobile is a knowledge-based configuration system for mobile phones and services enriched with recommendation functionalities to predict useful feature settings (defaults) for the user (Felfernig et al., 2010b). Example pages of RecoMobile are depicted in Figures 4.4, 4.5, 4.6, and 4.7.

5.2.2. Study Design

Our experiment addressed two relevant questions. (1) *Are users of product configuration systems influenced by default settings even if they are uncommon?* (2) *Is it possible to counteract the status quo bias by providing a configuration interface where both keeping and changing the presented default settings needs user interaction?*

To test the influence of uncommon defaults on the selection behavior of users we differentiate between three basic versions of RecoMobile (see Table 5.7). Users of Version A got a RecoMobile version were no defaults were set (see Figure 5.5). Out of this collected data we selected for each feature (presented as questions within a configuration session) the alternative which was chosen least often and used it as defaults for Version B and C. These two versions differ in the way to which extent user interaction is required. In Version B user interaction is only required when the customer wants to change the recommended default setting (low user involvement - see Figure 5.1). In Version C the acceptance as well as the changing of the default settings requires user interaction (high user involvement - see Figure 5.2).

Table 5.7.: RecoMobile – Configurator versions in user study.

Version A	no defaults	no defaults were presented to the user
Version B	defaults without confirmation	unusual defaults were presented to the user
Version C	defaults with confirmation	unusual defaults were presented to the user which he/she had to confirm

We conducted an online survey at the Graz University of Technology. N=143 subjects participated in the study. Each participant was assigned to one of the three configurator versions (see Table 5.7). The experiment involved a hypothetical scenario where the participants had to decide which mobile phone (including the corresponding services) they would select.

Figure 5.5.: Version A of RecoMobile - no defaults are provided.

5.2.3. Evaluation

In our evaluation we compared the data of the configurator version without default settings (Version A - see Table 5.7) with the data collected in Version B and C. For each feature we conducted a chi-square test to compare the selection behavior of the users. For many of the features we could observe significant differences in the selection distribution. An overview of the selection behavior in different configurator versions is given in Table 5.8 and Table 5.9.

Table 5.8.: Comparison of the value selection behavior in different configurator versions – Version A compared with Version B. Significant differences are highlighted with bold typeface.

Feature	Significance
use phone to read and write Emails	p=0.079
use phone to connect PC to web	**p=0.025**
use phone to read and write SMS	p=0.302
use phone for sports tracking	p=0.211
use phone for navigation	p=0.099
monthly minutes package	p=0.235
free sms messages included	p=0.323
selected data package	**p=0.001**
mobile antivirus service	**p=0.008**
mobile messenger	p=0.629
display phone number to receiver	p=0.100
publish phone number in phone book	**p=0.032**
prohibit charged services for countries	p=0.260
prohibit charged services for calls	**p=0.014**
prohibit charged services for sms	**p=0.009**

As an example, the evaluation results regarding the feature *Which charged services should be prohibited for SMS?* are depicted in Figure 5.6. For this feature the default in Version B and C was set to alternative 3 - *Utility and Entertainment* - (that option which was chosen least often in Version A). In both versions the default setting obviously had a strong impact on the selection behavior of the users. Only 2 % of the users of Version A selected option 3, whereas in Version B 24 % chose this default option. The interesting result is that the version with high user involvement (Version C) did not counteract the status quo bias. 25.6 % of the users of Version C selected the default alternative. Contrary to the assumption of Ritov and Baron (Ritov and Baron, 1992) people tend to stick to the status quo (the default option) even when user interaction is required to accept it.

In Figure 5.7 another example is shown for the feature *Which data package do you want?* The default in Version B and C was set to option 5 (*2048 kbit/s (+ 29.90 euro)*), which was the most expensive alternative of this feature. In Version A 4 % of the users decided to choose this option - the mean value for the expenses for this attribute is 5.5 Euro (see Table 5.10). In Version B 16 % and in Version C 18.6 % of the users retained the status quo alternative. The mean value for the data package expenses in Version B is 12.8 Euro, and in Version C 13.3 Euro. This example shows that the status quo effect can be exploited to make customers spend more money. Here again the status quo effect was not suppressed in Version C, where people had to confirm the default setting.

5.2.4. Related Work

Research in the field of human decision making has revealed that people have a strong tendency to keep the status quo when choosing among alternatives (see, for example, (Samuelson and Zeckhauser, 1988; Kahneman et al., 1991; Ritov and Baron, 1992; Bostrom and Ord, 2006)). This decision bias has firstly been reported by Samuelson and Zeckhauser (Samuelson and Zeckhauser,

Table 5.9.: Comparison of the value selection behavior in different configurator versions – Version A compared with Version C. Significant differences are highlighted with bold typeface.

Feature	Significance
use phone to read and write Emails	p=0.133
use phone to connect PC to web	p=0.193
use phone to read and write SMS	p=0.395
use phone for sports tracking	p=0.825
use phone for navigation	p=0.392
monthly minutes package	**p=0.014**
free sms messages included	**p=0.032**
selected data package	**p=0.004**
mobile antivirus service	**p=0.002**
mobile messenger	p=0.643
display phone number to receiver	p=0.090
publish phone number in phone book	p=0.497
prohibit charged services for countries	p=0.107
prohibit charged services for calls	**p=0.011**
prohibit charged services for sms	**p=0.020**

Table 5.10.: Mean values for the monthly data package expenses.

	Mean Value
Version A	5.528 Euro
Version B	12.844 Euro
Version C	13.281 Euro

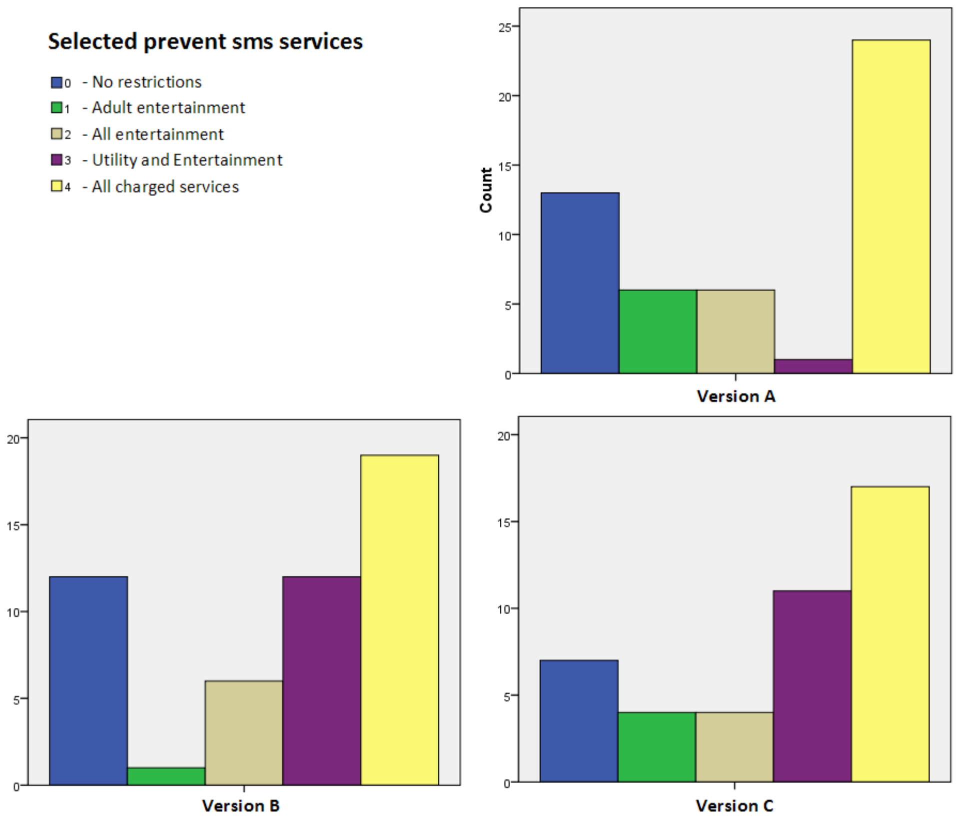

Figure 5.6.: Selections for *prohibit charged services for SMS* - the results of the conducted chi-square test show that the underlying distributions differ significantly (p=0.009 for Version A compared with Version B, p=0.020 for Version A compared with Version C).

1988). To our knowledge, such decision biases have not been analyzed in detail in the context of interactive configuration scenarios. The goal of our work was to investigate whether the status quo effect also exists in product configuration systems. In Section 4.2 we introduced an approach to integrate recommendation technologies with knowledge-based configuration (Felfernig et al., 2010b). The results of this research indicate that supporting users with personalized feature recommendations (defaults) can lead to a higher satisfaction with the configuration process. The work presented in this section is a logical continuation of the work presented in Section 4.2, which extends the impact analysis of personalization concepts to the psychological phenomenon of decision biases.

In the psychological literature there exist a couple of theories that explain the existence of different types of decision biases (see Section 3.1). In the context of our empirical study we could observe a status quo bias triggered by feature value recommendations, even if uncommon values are used as defaults. Another phenomenon that influences the selection behavior of consumers is known as *asymmetric dominance effect* (Huber et al., 1982) which belong to the group of *decoy effects* (Huber et al., 1982; Simonson and Tversky, 1992; Yoon and Simonson, 2008; Teppan and Felfernig, 2009c) (see Section 3.2.1). According to this theory consumers show a preference change

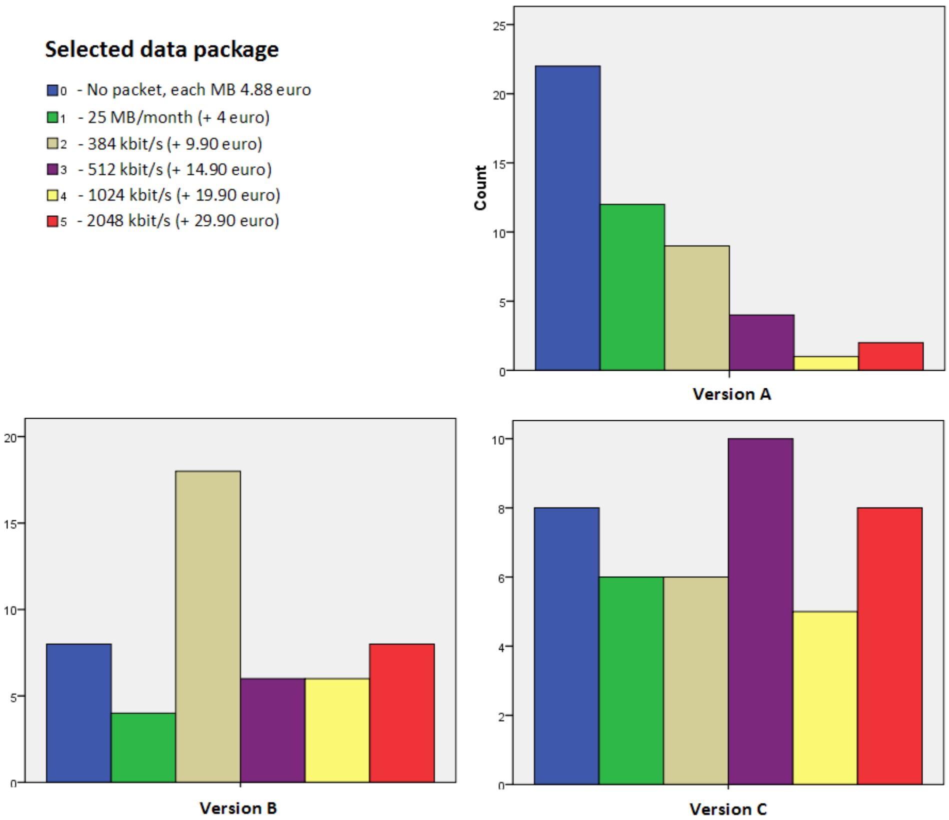

Figure 5.7.: Selections for *Monthly data package* - the results of the conducted chi-square test show that the underlying distributions differ significantly (p=0.001 for Version A compared with Version B, p=0.004 for Version A compared with Version C).

between two options when a third asymmetrically dominating option is added to the consideration set. An asymmentrically dominated option is completely dominated by (i.e., inferior to) one item in the consideration set but not by another. The insertion of such an alternative into the consideration set can increase the percentage of consumers who choose the dominating option compared to a situation where the asymmetrically dominated option is absent (Huber et al., 1982). Decoy effects have been intensively investigated in different application contexts, see, for example, (Huber et al., 1982; Simonson and Tversky, 1992; Yoon and Simonson, 2008; Teppan and Felfernig, 2009b,c; Felfernig et al., 2008).

The *Framing effect* (Marteau, 1989; Tversky and Kahneman, 1981, 1986; Levin et al., 1998; Jannach et al., 2011) – see Section 3.2.3 – describes the fact that presenting one and the same decision alternative in different variants can lead to choice reversals and therefore to the selection of different alternatives. Tversky and Kahnemann (Tversky and Kahneman, 1981) have shown that effect in a series of studies where they confronted participants with choice problems using variations in the framing of decision outcomes. They reported that "*seemingly inconsequential changes in the formulation of choice problems caused significant shifts of preference'*" (Tversky and Kahneman, 1981). Frame analysis has been applied to different application contexts, such as social movement, political opinion formation, or in the behavioral finance domain (Marteau, 1989;

Tversky and Kahneman, 1981, 1986; Levin et al., 1998; Jannach et al., 2011).

5.2.5. Conclusion

In this section we have presented the results of an empirical study that had the goal to analyze the impact of the status quo bias in product configuration scenarios where defaults are presented as recommendations to users. The results of our study show that there exists a strong biasing effect even if uncommon values are presented as default values. Our findings show that, for example, status quo effects make users of a configuration system selecting more expensive solution alternatives. As a consequence of these results, we have to increasingly turn our attention to ethical aspects when implementing product configurators, since it is possible that users are mislead simply by the fact that some defaults are representing expensive solution alternatives, but are maybe not needed to fulfill the given requirements. Finally, we detected that providing a user interface, were both, keeping and changing the provided defaults, needs user interaction (we called this the *high involvement user interface*) does not counteract the status quo bias. Our future work will include the investigation of additional decision phenomena in the context of knowledge-based configuration scenarios (for example, framing or decoy effects).

5.3. Summary

In this chapter we have presented the results of our research that aimed at analyzing different factors that may effect humans decision making. The online study described in Section 5.1 aimed at exploring the impact of the format of information presentation in product configuration scenarios where defaults are presented as recommendations to users. Defaults are preselected options used to express personalized feature recommendations which can lead to a higher satisfaction with the configuration process (see Section 4.2). We used three different user interfaces of a product configurator, which differ in the way the defaults are displayed to the user, as well as in the extent to which user interaction is required to select/reject the provided default suggestions. These different representation styles induced significant differences in users' satisfaction with the overall perceived quality of the selection process, as well as in users' confidence and trust in the product configuration system.

In the experiment described in Section 5.2 we revealed a strong biasing effect when presenting default recommendations in the context of a product configuration scenario. Our experiment illustrated that such default recommendations can cause a status quo bias, and make users, for example, selecting expensive solution alternatives. Furthermore, we found that different user interfaces, that support different user interaction styles, do not counteract the status quo bias.

The results of our experiments highlights the importance of regarding the processes of human decision making in the context of recommendation scenarios to improve the usability and applicability of intelligent selling applications.

Chapter 6

Conclusions and Future Work

To support users of intelligent selling environments in their decision making process, it is important to consider different factors, such as the prediction accuracy of recommender algorithms, user interface design elements, as well as psychological effects. This thesis introduces and evaluates approaches to assist customers in their preference construction process in the context of conversational recommender systems, as well as in product configuration systems. We further explored effects of presenting recommendations on consumer decision making strategies. In the following we reflect on our research questions and contributions. This chapter concludes with an outlook on future research issues.

6.1. Conclusion

In the following we provide an overview of our research questions (for a detailed discussion see Section 1.2), and our contributions to answer these questions.

Research Question Q1:

How to improve the algorithm efficiency of unit critiquing and therefore reduce the number of critiquing cycles in critiquing sessions?
In the context of this work we have developed new approaches to unit critiquing. In Section 4.1.2 the *Nearest Neighbor Compatibility Critiquing* algorithm is introduced, that focuses on recommending items that have been presented to previous users with a similar critiquing history to the current user. This approach is further combined with conventional and experience-based unit critiquing to corresponding ensemble-based variations (see Section 4.1.3). The results of an empirical study indicate that our new nearest neighbor compatibility critiquing approach as well as ensemble-based variations thereof have the potential to reduce the number of critiquing cycles in critiquing sessions. This reduction means a cost saving for users, since less critiquing cycles are needed to find the target product. A detailed discussion of our evaluation concerning the improvements to unit critiquing is presented in Section 4.1.4 and Section 4.1.5.

Research Question Q2:

How to improve the subjective quality of configuration dialogs by integrating recommendation strategies in the product configuration process?
The RecoMobile configurator presented in Section 4.2 has been developed and evaluated on the basis of the configuration concepts introduced in (Tiihonen and Felfernig, 2010). The goal of our work was to implement and evaluate a system that integrates recommendation technologies in the preference construction process. Users of RecoMobile are pro-actively supported by personalized defaults (see Section 4.2.2), and minimal sets of changes in the case that no solution could be found (see Section 4.2.1). We see our contribution as a first one on the way towards more intelligent recommender user interfaces, that know more about the user, and also know how to exploit this knowledge for improving the quality of intelligent selling applications in different dimensions such as prediction accuracy or overall satisfaction with the application. Section 4.2.4 presents a detailed discussion of improvements achieved by integrating recommendation strategies in configuration dialogs. The evaluation of the RecoMobile configurator indicates, that the integration of recommendation techniques in a product configuration system is useful in terms of improving the user acceptance of the configurator interface.

Research Question Q3.1:

Do different default representation styles have an influence on a user's willingness to accept feature value recommendations?
In Section 5.1 the results of an empirical study are presented that analyzed the impact of different presentation styles of personalized recommendations in product configuration scenarios. For this purpose we implemented three different user interface versions of RecoMobile. The versions differed in the way the recommendations were displayed to the user, and in the extent of user interaction required to accept/reject the suggestions. Our results indicate that there is no significant difference between the selection behavior of users of different configurator versions. But we found a strong correlation between the number of selected or accepted recommendations, and users' satisfaction with the presentation of feature recommendations. Our results therefore suggest that the method of presenting personalized defaults is an important factor that influences users' satisfaction with the configuration process. For example, the RecoMobile user interface version where the user has to select the desired feature alternative even in the case that the user wants to accept the recommendation, was rated lowest in almost all subjective measures. A detailed discussion of our evaluation concerning the impact of different representation styles of defaults is given in Section 5.1.3.

Research Question Q3.2:

Do different default representation styles have an impact on users' subjective perception of the configuration system?
To evaluate if different methods of presenting defaults have a impact on perceived quality of the system, participants of an empirical online study had to fill out a post-study questionnaire after

interacting with the RecoMobile prototype (see Section 5.1). This questionnaire is based on an evaluation framework developed by Pu and Chen (Pu and Chen, 2010) and is designed to cover eight subjective measures related to satisfaction, trust, confidence, and intentions of behavior. Our findings revealed that the method of presenting defaults can have a significant impact on users' satisfaction with the overall perceived quality of the selection process, as well as on users' confidence and trust in the product configurator (see Section 5.1.3).

Research Question Q4.1:

Do users of product configuration systems experience a *Status Quo Bias*, i.e, is the choice behavior of users influenced by defaults?
In Section 3.1 and Section 3.2 we have presented a selected set of decision-psychological phenomena – a number of related empirical studies clearly show the importance of taking into account such theories when implementing a recommender application. In Section 5.2 we have presented an empirical study that had the goal to investigate the effect of the status quo bias in the context of a product configuration system where feature value recommendations (defaults) are provided. The results of our study show that there exists a strong biasing effect even if uncommon values are provided as default settings. This result goes along with the findings of Bostrom and Ord, who state that people incorrectly judge a status quo alternative to be better than another (Bostrom and Ord, 2006). Our findings further revealed that it is even possible to exploit the status quo effect to make consumers spend more money. This scenario definitely has ethical aspects to be dealt with since companies can potentially try to mislead users by setting defaults to expensive alternatives that are not really needed to fulfill the customer requirements. A detailed discussion of our evaluation concerning the impact of feature value recommendations on human selection behavior is given in Section 5.2.3.

Research Question Q4.2:

Is it possible to counteract the *Status Quo Bias* by providing an appropriate user interface?
For our empirical study described in Section 5.2 we have developed and evaluated interface design types that supports an interaction type described by Ritov and Barron (Ritov and Baron, 1992). Contrary to the assumption of Ritov and Baron (Ritov and Baron, 1992), we found that user interfaces where both keeping and changing the provided defaults needs user interaction did not counteract the status quo bias (see Section 5.2.3).

6.2. Future Work

In the psychological literature there exist a couple of theories that explain the existence of different types of decision biases in a consumer decision making situation. Consumers are influenced by the format of the information presented and as a consequence use different decision-making strategies in different contexts (see, for example, (Asch, 1949; Payne, 1976; Bettman and Kakkar, 1977;

Bettman et al., 1991)). The results of a study conducted by Kim and Moon (Kim and Moon, 1998) showed that the manipulation of user interface design elements, such as title, menu, main clipart and color, can have an impact on users' *trust* in the system. The research of Eroglu at al. (Eroglu, 2001; Eroglu et al., 2003) revealed that the manipulation of user interface design elements, such as color, background, or fonts may also have an influence on consumer's *willingness to buy*. These findings imply that the design of the user interface can have a major impact on the final outcome of the decision making process. Our future work will include the investigation of additional decision phenomena in the context of interactive recommendation scenarios (for example, framing, anchoring, or decoy effects – see Section 3.2). In the remainder of this chapter relevant topics for future research are discussed.

Repair actions

Repair actions help users to get out of the so-called "no solution could be found" dilemma (Felfernig et al., 2009) (see Section 2.1.3). If a given set of requirements does not allow the calculation of a recommendation there exist potentially many different alternative combinations of repair actions (exponential in the number of requirements (O'Sullivan et al., 2007)) that resolve the current conflict. As a consequence, it is not possible to present the complete set of possible repair actions and we have to select a subset that best fits with the requirements of the user. An approach to personalize the selection of repair actions has been introduced in (Felfernig et al., 2009; Schubert et al., 2011). A major goal for future work is to extend these approaches by additionally taking into account different types of decision biases that potentially occur in the repair selection process. For example, (Kivetz and Simonson, 2000) investigated the impact of incomplete information on consumer choice. The results from a series of studies indicate that choosing from sets with missing information can influence consumers tastes and purchase decisions (Kivetz and Simonson, 2000). The presentation of repair actions can be seen as such a set with missing information, since repair alternatives typically include only a subset of the product attributes. In this context we want to investigate if the inclusion or exclusion of attributes in the repair set can have an impact on the decision making process.

Serial position effects on result pages in the context of knowledge-based recommender systems have been demonstrated by Felfernig et al. (Felfernig et al., 2007). We want to investigate whether such effects also occur when presenting repair actions. Finally, we are interested in the existence of decision biases when using *defaults* in a repair scenario to express personalized recommendations for repair alternatives.

Result pages

In the context of result pages the focus of our future work will be set to the investigation of the existence of *decoy effects* (see Section 3.2.1). In this context we are interested in answering questions regarding the upper bound for the number of products such that decoy effects still occur. Furthermore, we are interested in interrelationships between item distances (typically calculated by different types of similarity metrics (McSherry, 2004)) and the existence of decoy effects. A further question is whether we have to cluster target, competitor, and decoy items or whether decoy effects still occur if the positioning distance between items is increased. Another challenging question is whether there exists an interrelationship between different types of decoy effects, for example, do the asymmetric dominance effect and the compromise effect compensate each other or is there a kind of ”synergy effect” in terms of even more significant selection shifts?

Critiquing systems

Critiquing-based recommender applications (Reilly et al., 2007a; Chen and Pu, 2007a) adapt a conversational interaction style, where users provide feedback by critiquing attributes of recommended items in a directional way. Critiques are a natural way to support users in item selection processes without forcing them to explicitly specify values for certain item properties. Especially users who are non-experts in the product domain prefer a navigation process where they are articulating requirements on a more abstract level such as *lower price* or *higher resolution*.

Viappiani et al. (Viappiani et al., 2007) indicate that the example-critiquing strategy used by critiquing based recommender systems can lead to an anchoring effect. In this context a major goal for future work is to conduct a user experiment to investigate the anchoring effect in the context of a critiquing scenario.

Compound critiques

In order to fasten the interaction with a critique-based recommender application, prototype systems have been developed that support the articulation of so-called compound critiques (McCarthy et al., 2004; Zhang and Pu, 2006), i.e., critiques that include two or more change requests regarding basic product properties. An example of such a compound critique is *lower price and higher resolution*. A typical critiquing-based recommender presents a list of alternative compound critiques to the user. In this context, we are interested in answering the question whether decoy effects and serial position effects also occur in the selection of compound critiques.

Critiquing algorithms

The introduced unit critiquing approaches (see Section 4.1) have been evaluated in an offline experiment where artificially generated data was used. This evaluation method has several drawbacks. The major problem is that the randomly selection of critiques might not reflect realistic user preferences. An evaluation and comparison of different critiquing approaches with real users lies within future work. An important factor in this context is the subjective quality of the different critiquing algorithms. Furthermore, the impact of different similarity measures to compare user critiques has to be considered in a real user environment.

In the evaluation of ensemble based critiquing variations (see Section 4.1.5), the ensemble based variation composed by three individual techniques performed worse, compared to combinations of two algorithms. An in-depth analysis of these results, to see how the individual algorithms move towards the target item and therefore influence the ensemble-based critiquing solution, lies within future work.

Integration of recommendation technologies in product configuration systems

We interpret the work presented in Section 4.2 as a first major step toward the integration of recommendation technologies with knowledge-based configuration. The presented empirical study shows various opportunities for improving state-of-the-art configuration technologies and our goal for future work is to extend the current concepts to other application scenarios. We want to support the recommendation of products whose structure varies significantly as the result of the configuration task, which means that we have to augment our approach with similarity (diversity) measures that take into account compositional (part-of) and connection (port) structures as well.

Bibliography

Adomavicius, G., Bockstedt, J., Curley, S., and Zhang, J. 2011. Recommender systems, consumer preferences, and anchoring effects. In *RecSys'2011 Workshop on Human Decision Making in Recommender Systems*. 35–42. (Cited on pages 4, 37, and 49.)

Adomavicius, G. and Tuzhilin, A. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering 17,* 6, 734–749. (Cited on pages 1 and 3.)

Asch, S. 1949. Forming impressions of personality. *Journal of Abnormal and Social Psychology 41,* 6, 258–290. (Cited on pages 6, 31, 33, 34, 41, 42, 74, 81, 82, and 101.)

Balabanovic, M. and Shoham, Y. 1997. Fab: Content-based, collaborative recommendation. *Communications of the ACM 40,* 3, 66–72. (Cited on pages 19 and 21.)

Barker, V. E., O'Connor, D. E., Bachant, J., and Soloway, E. 1989. Expert systems for configuration at digital: Xcon and beyond. *Communications of the ACM 32,* 3, 298–318. (Cited on pages 27, 28, 65, and 77.)

Bell, R. M. and Koren, Y. 2007. Improved neighborhood-based collaborative filtering. In *1st KDDCup'07*. San Jose, California. (Cited on pages 17 and 22.)

Bell, R. M., Koren, Y., and Volinsky, C. 2007. The bellkor solution to the netflix prize. Tech. rep., AT & T Labs – Research. (Cited on page 60.)

Bertini, M., Wathieu, L., and of Research, H. B. S. D. 2006. *The framing effect of price format*. Working paper, Division of Research, Harvard Business School. (Cited on page 46.)

Bettman, J., Johnson, E., and Payne, J. 1991. Consumer decision making. *Handbook of Consumer Theory and Research*, 50–84. (Cited on pages 6, 7, 33, 34, 81, 82, and 102.)

Bettman, J. and Kakkar, P. 1977. Effects of information presentation format on consumer information acquisition strategies. *Journal of Consumer Research 3,* 4, 233–240. (Cited on pages 6, 82, and 101.)

Bettman, J. R. 1979. Memory factors in consumer choice: A review. *The Journal of Marketing 43,* 2, 37–53. (Cited on pages 2, 28, and 89.)

Bettman, J. R., Luce, M. F., and Payne, J. W. 1998. Constructive consumer choice processes. *Journal of Consumer Research 25,* 3, 187–217. (Cited on pages 2, 4, 26, 28, 29, 35, 36, and 65.)

Blecker, T., Abdelkafi, N., Kreutler, G., and Friedrich, G. 2004. Product configuration systems: State of the art, conceptualization and extensions. In *Proceedings of the Eight Maghrebian Conference on Software Engineering and Artificial Intelligence (MCSEAI)*. 25–36. (Cited on pages 1, 2, 13, and 28.)

Bobrow, D. G. and Collins, A., Eds. 1975. *Representation and Understanding: Studies in Cognitive Science*. Academic Press, Inc., Orlando, FL, USA. (Cited on page 25.)

Bostrom, N. and Ord, T. 2006. The reversal test: eliminating status quo bias in applied ethics. *Ethics 116,* 4, 656–679. (Cited on pages 7, 81, 90, 91, 93, and 101.)

Breese, J. S., Heckerman, D., and Kadie, C. 1998. Empirical analysis of predictive algorithms for collaborative filtering. *Proceedings of the 14th conference on Uncertainty in Artificial Intelligence 461,* 8, 43–52. (Cited on page 17.)

Bridge, D., Göker, M. H., McGinty, L., and Smyth, B. 2005. Case-based recommender systems. *The Knowledge Engineering Review 20,* 3, 315–320. (Cited on page 36.)

Bridge, D. and Kelly, J. P. 2005. Diversity-enhanced conversational collaborative recommendations. In *Proceedings of the 16th Irish Conference on Artificial Intelligence and Cognitive Science*. 29–38. (Cited on pages 14 and 21.)

Brusilovsky, P., Kobsa, A., and Nejdl, W., Eds. 2007. *The Adaptive Web: Methods and Strategies of Web Personalization.* Lecture Notes in Computer Science. Springer. (Cited on page 36.)

Burke, R. 2000. In *Encyclopedia of Library and Information Systems*. Vol. 69. New York, NY, USA, 180–200. (Cited on pages 3, 18, 22, 23, and 24.)

Burke, R. 2002a. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction 12,* 4, 331–370. (Cited on pages 14, 15, 19, 22, and 77.)

Burke, R., Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., and Sartin, M. 1999. Integrating knowledge-based and collaborative-filtering recommender systems. In *Proceedings of the Workshop on AI and Electronic Commerce. AAAI 99*. Orlando, Florida, 69–72. (Cited on pages 1 and 23.)

Burke, R. D. 2002b. Interactive critiquing for catalog navigation in e-commerce. *Artificial Intelligence Review 18,* 3-4, 245–267. (Cited on pages 5 and 64.)

Burke, R. D., Hammond, K. J., and Young, B. C. 1996. Knowledge-based navigation of complex information spaces. In *Proceedings of the thirteenth national conference on Artificial intelligence - Volume 1*. AAAI'96. AAAI Press, 462–468. (Cited on pages 27, 55, 57, 58, 60, 64, and 78.)

Burke, R. D., Hammond, K. J., and Young, B. C. 1997. The findme approach to assisted browsing. *IEEE Expert 12,* 4, 32–40. (Cited on pages 2, 15, 25, 27, 56, and 64.)

Cervone, D. and Peake, P. K. 1986. Anchoring, efficacy, and action: The influence of judgmental heuristics on self-efficacy judgments and behavior. *Journal of Personality and Social Psychology 50,* 3, 492–501. (Cited on page 49.)

Chalmers, M., Dieberger, A., Höök, K., and Rudström, r. 2004. Social Navigation and Seamful Design. *Cognitive Studies 11,* 3, 1–11. (Cited on pages 37 and 49.)

Chapman, G. B. and Johnson, E. J. 1999. Anchoring, activation, and the construction of values. *Organizational Behavior and Human Decision Processes 79,* 2, 115–153. (Cited on pages 37, 48, and 49.)

Chen, L. and Pu, P. 2005. Trust building in recommender agents. In *1st International Workshop on Web Personalization, Recommender Systems and Intelligent User Interfaces (WPRSIUI'05).* 135–145. (Cited on pages 4, 40, 51, 81, and 82.)

Chen, L. and Pu, P. 2007a. The evaluation of a hybrid critiquing system with preference-based recommendations organization. In *Proceedings of the 2007 ACM conference on Recommender systems.* ACM, New York, NY, USA, 169–172. (Cited on page 103.)

Chen, L. and Pu, P. 2007b. Preference-based organization interfaces: Aiding user critiques in recommender systems. In *Proceedings of the 11th international conference on User Modeling (UM '07).* Springer-Verlag, Berlin, Heidelberg, 77–86. (Cited on pages 4 and 26.)

Chen, L. and Pu, P. 2012. Critiquing-based recommenders: survey and emerging trends. *User Modeling and User-Adapted Interaction 22,* 1-2, 125–150. (Cited on pages 1, 27, and 56.)

Chira, I., Adams, M., and Thornton, B. 2008. Behavioral bias within the decision making process. *Journal Of Business & Economics Research 6,* 8, 11–20. (Cited on page 35.)

Conklin, J. 1987. Hypertext: An introduction and survey. *IEEE Computer 20,* 9, 17–41. (Cited on page 1.)

Coster, R., Gustavsson, A., Olsson, T., and Rudstroem, A. 2002. Enhancing web-based configuration with recommendations and cluster-based help. In *Proceedings of the AH'2002 Workshop on Recommendation and Personalization in eCommerce.* 30–39. (Cited on pages 3, 6, 29, 72, 73, and 77.)

Crowder, R. 1976. *Principles of learning and memory.* The Experimental psychology series. Lawrence Erlbaum Associates. (Cited on pages 41 and 82.)

Debnath, S., Ganguly, N., and Mitra, P. 2008. Feature weighting in content based recommendation system using social network analysis. In *Proceedings of the 17th international conference on World Wide Web.* ACM, New York, NY, USA, 1041–1042. (Cited on pages 18, 19, and 20.)

Dourish, P. and Chalmers, M. 1994. Running out of space: Models of information navigation. *Proceedings of HCI'94*. Short paper presented at HCI'94. (Cited on pages 37 and 49.)

Ebbinghaus, H. and Clara, E. B. 1885. *Memory: A Contribution to Experimental Psychology.* Originally published in New York by Teachers College, Columbia University. (Cited on pages xi, 41, and 42.)

Engel, J. and Blackwell, R. 1986. *Consumer behavior.* Dryden Press series in marketing. Dryden Press. (Cited on pages 35 and 37.)

Erasmus, A., Boshoff, E., and Rousseau, G. 2001. Consumer decision-making models within the discipline of consumer science: a critical approach. *Journal of Family Ecology and Consumer Sciences 29,* 1, 82–90. (Cited on page 34.)

Erickson, T. and Kellogg, W. A. 2000. Social translucence: an approach to designing systems that support social processes. *ACM Transactions on Computer-Human Interaction 7,* 1, 59–83. (Cited on page 50.)

Eroglu, S. 2001. Atmospheric qualities of online retailing a conceptual model and implications. *Journal of Business Research 54,* 2, 177–184. (Cited on pages 4, 7, 53, 81, 82, and 102.)

Eroglu, S., Machleit, K., and Davis, L. 2003. Empirical testing of a model of online store atmospherics and shopper responses. *Psychology and Marketing 20,* 2, 139–150. (Cited on pages 4, 7, 53, 81, 82, and 102.)

Falkner, A. A., Felfernig, A., and Haag, A. 2011. Recommendation technologies for configurable products. *AI Magazine 32,* 3, 99–108. (Cited on pages 6, 29, and 30.)

Faltings, B., Pu, P., Torrens, M., and Viappiani, P. 2004. Designing example-critiquing interaction. In *Proceedings of the 9th international conference on Intelligent user interfaces (IUI 2004).* ACM, 22–29. (Cited on pages 2 and 4.)

Felfernig, A. 2004. Effort estimation for knowledge-based configuration systems. In *Proceedings of the ECAI 2004 Workshop on Configuration.* 1–6. (Cited on page 28.)

Felfernig, A. and Burke, R. 2008. Constraint-based recommender systems: technologies and research issues. In *Proceedings of the 10th International Conference on Electronic Commerce.* ICEC '08. ACM, New York, NY, USA, 1–10. (Cited on pages 22, 23, and 24.)

Felfernig, A., Friedrich, G., Gula, B., Hitz, M., Kruggel, T., Melcher, R., Riepan, D., Strauss, S., Teppan, E., and Vitouch, O. 2007. Persuasive recommendation: Exploring serial position effects in knowledge-based recommender systems. In *Proceedings of the 2nd International Conference of Persuasive Technology (Persuasive 2007).* 283–294. (Cited on pages 4, 31, 37, 42, 74, and 102.)

Felfernig, A., Friedrich, G., Jannach, D., and Stumptner, M. 2004. Consistency-based diagnosis of configuration knowledge bases. *Artificial Intelligence 152,* 2, 213–234. (Cited on pages 24, 31, 66, and 69.)

Felfernig, A., Friedrich, G., Schubert, M., Mandl, M., Mairitsch, M., and Teppan, E. 2009. Plausible repairs for inconsistent requirements. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence.* Pasadena, California, USA, 791–796. (Cited on pages 24, 31, 66, 69, and 102.)

Felfernig, A., Gordea, S., Jannach, D., Teppan, E., and Zanker, M. 2007. A short survey of recommendation technologies in travel and tourism. *ÖGAI Journal 25,* 2, 17–22. (Cited on pages 18 and 22.)

Felfernig, A., Gula, B., Leitner, G., Maier, M., Melcher, R., S., S., and E., T. 2008. A dominance model for the calculation of decoy products in recommendation environments. In *AISB 2008 Symposium on Persuasive Technology*. 43–50. (Cited on pages 36, 37, and 96.)

Felfernig, A., Gula, B., Leitner, G., Maier, M., Melcher, R., and Teppan, E. 2008. Persuasion in knowledge-based recommendation. In *PERSUASIVE '08 Proceedings of the 3rd international conference on Persuasive Technology*. Springer-Verlag, Berlin, Heidelberg, 71–82. (Cited on pages 31, 32, 37, 42, and 74.)

Felfernig, A., Mandl, M., Tiihonen, J., and Schubert, M. 2010a. Personalized product configuration. *Proceedings of the MKWI'10 Workshop on Planning and Scheduling*, 2251–2263. (Cited on pages 8, 37, 48, 55, and 66.)

Felfernig, A., Mandl, M., Tiihonen, J., Schubert, M., and Leitner, G. 2010b. Personalized user interfaces for product configuration. *Proceedings of the 15th International Conference on Intelligent User Interfaces (IUI'10)*, 317–320. (Cited on pages 3, 6, 8, 29, 31, 37, 48, 55, 66, 68, 74, 89, 91, and 95.)

Felfernig, A., Teppan, E., Friedrich, G., and Isak, K. 2008. Intelligent debugging and repair of utility constraint sets in knowledge-based recommender applications. In *Proceedings of the 13th international conference on Intelligent user interfaces*. IUI '08. 217–226. (Cited on pages 31, 66, and 69.)

Fitzsimons, G. J. and Lehmann, D. R. 2004. Reactance to recommendations: When unsolicited advice yields contrary responses. *Marketing Science 23,* 1, 82–94. (Cited on page 47.)

Fleischanderl, G., Friedrich, G. E., Haselböck, A., Schreiner, H., and Stumptner, M. 1998. Configuring large systems using generative constraint satisfaction. *IEEE Intelligent Systems 13,* 4, 59–68. (Cited on pages 27 and 77.)

Frederich, N. T., Michael, D. W., Richard, F., Austin, H., and Thomas, M. 1982. Rabbit: An intelligent database assistant. In *AAAI-82 Proceedings*. 314–318. (Cited on page 25.)

Ganzach, Y. and Schul, Y. 1995. The influence of quantity of information and goal framing on decision. *Acta Psychologica 89,* 1, 23–36. (Cited on pages 46 and 47.)

Geneste, L. and Ruet, M. 2001. Experience based configuration. In *17th International Conference on Artificial Intelligence, Workshop on Configuration*. 4–10. (Cited on pages 2, 3, 6, 29, and 77.)

Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. 1992. Using collaborative filtering to weave an information tapestry. *Communications of the ACM 35,* 12, 61–70. (Cited on pages 3, 15, 20, 22, and 66.)

Grabner-Kräuter, S. and Kaluscha, E. A. 2003. Empirical research in on-line trust: a review and critical assessment. *International Journal of Human-Computer Studies 58,* 6, 783–812. (Cited on page 82.)

Harita, M., Shveta Kundra, B., Punam, B., and V. S., D. 2011. Collaborative personalized web recommender system using entropy based similarity measure. *International Journal of Computer Science Issues 8,* 6, 231–240. (Cited on page 52.)

Häubl, G. and Murray, K. B. 2003. Preference construction and persistence in digital marketplaces: The role of electronic recommendation agents. *Journal of Consumer Psychology 13,* 1, 75–91. (Cited on pages 1 and 47.)

Hayes, C., Massa, P., Avesani, P., and Cunningham, P. 2002. An on-line evaluation framework for recommender systems. In *In Workshop on Personalization and Recommendation in E-Commerce*. Springer Verlag, 50 – 59. (Cited on pages 51, 53, 77, and 78.)

Heiskala, M., Tiihonen, J., Paloheimo, K.-S., and Soininen, T. 2007. *Mass Customization with Configurable Products and Configurators: A Review of Benefits and Challenges.* Mass Customization Information Systems in Business. (Cited on pages 3 and 28.)

Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. 1999. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval.* ACM Press, New York, NY, USA, 230–237. (Cited on page 17.)

Herlocker, J. L., Konstan, J. A., and Riedl, J. 2000. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work.* CSCW ’00. ACM, New York, NY, USA, 241–250. (Cited on pages 16 and 66.)

Herrmann, A., Heitmann, M., and Polak, B. 2007. The power of defaults. *Absatzwirtschaft 6,* 46–47. (Cited on page 37.)

Howard, J. and Sheth, J. 1969. *The theory of buyer behavior.* The Wiley marketing series. Wiley. (Cited on pages 35 and 37.)

Hu, Y., Koren, Y., and Volinsky, C. 2008. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining.* ICDM ’08. IEEE Computer Society, 263–272. (Cited on page 15.)

Huber, J., Payne, J. W., and Puto, C. 1982. Adding asymmetrically dominated alternatives - violations of regularity and the similarity hypothesis. *Journal of Consumer Research 9,* 1, 90–98. (Cited on pages xi, xiii, 37, 38, 39, 40, 95, and 96.)

Huffman, C. and Kahn, B. E. 1998. Variety for sale: Mass customization or mass confusion? *Journal of Retailing 74,* 4, 491–513. (Cited on pages 1, 14, 36, 37, 47, 65, 89, 90, and 91.)

Jacoby, J., Speller, D., and Berning, C. 1974. Brand Choice Behavior as a Function of Information Load: Replication and Extension. *The Journal of Consumer Research 1,* 1, 33–42. (Cited on page 14.)

Jacoby, J., Speller, D., and Kohn, C. 1974. Brand Choice Behavior as a Function of Information Load. *Journal of Marketing Research 11,* 1, 63–69. (Cited on pages 1, 2, 28, 47, and 89.)

Jacowitz, K. and Kahneman, D. 1995. Measures of anchoring in estimation tasks. *Personality and Social Psychology Bulletin 21,* 11, 1161–1166. (Cited on page 49.)

Jannach, D., Zanker, M., Felfernig, A., and Friedrich, G. 2011. *Recommender Systems An Introduction.* Cambridge University Press. (Cited on pages 37, 96, and 97.)

JARVENPAA, S. L., TRACTINSKY, N., AND VITALE, M. 2000. Consumer trust in an internet store. *Information Technology and Management 1,* 1-2, 45–71. (Cited on pages 4 and 82.)

JAWAHEER, G., SZOMSZOR, M., AND KOSTKOVA, P. 2010. Comparison of implicit and explicit feedback from an online music recommendation service. In *Proceedings of the 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems.* HetRec '10. ACM, New York, NY, USA, 47–51. (Cited on pages 13 and 15.)

JUNKER, U. 2004. Quickxplain: preferred explanations and relaxations for over-constrained problems. In *Proceedings of the 19th national conference on Artifical intelligence.* AAAI'04. AAAI Press, 167–172. (Cited on pages 29, 69, 71, and 77.)

KAHNEMAN, D., KNETSCH, J. L., AND THALER, R. H. 1991. Anomalies: The endowment effect, loss aversion, and status quo bias. *Journal of Economic Perspectives 5,* 1, 193–206. (Cited on pages 7, 48, 81, 90, and 93.)

KAHNEMAN, D. AND TVERSKY, A. 1979. Prospect theory: An analysis of decision under risk. *Econometrica 47,* 2, 263–291. (Cited on pages xii, 44, 45, and 90.)

KAMALI, N. AND LOKER, S. 2002. Mass customization: On-line consumer involvement in product design. *Journal of Computer-Mediated Communication 7,* 4. (Cited on page 89.)

KASSARJIAN, H. H. 1982. The development of consumer behavior theory. *Advances in Consumer Research 9*, 20–22. (Cited on page 35.)

KEENEY, R. L. AND RAIFFA, H. 1976. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs.* Wiley series in probability and mathematical statistics. Wiley. (Cited on page 56.)

KIM, J. AND MOON, J. Y. 1998. Designing towards emotional usability in customer interfaces – trustworthiness of cyber-banking system interfaces. *Interacting with Computers 10,* 1, 1–29. (Cited on pages 6, 53, 82, and 102.)

KIVETZ, R. AND SIMONSON, I. 2000. The effects of incomplete information on consumer choice. *Journal of Marketing Research 37,* 4, 427–448. (Cited on page 102.)

KOHAVI, R., LONGBOTHAM, R., SOMMERFIELD, D., AND HENNE, R. M. 2009. Controlled experiments on the web: survey and practical guide. *Data Min. Knowl. Discov. 18,* 1, 140–181. (Cited on page 53.)

KOHAVI, R. AND PROVOST, F. 1998. *Glossary of Terms.* Editorial for the Special Issue on Applications of Machine Learning and the Knowledge Discovery Process. (Cited on page 52.)

KOLODNER, J. 1993. *Case-based reasoning.* Morgan Kaufmann Series in Representation and Reasoning Series. Morgan Kaufmann Publishers. (Cited on pages 29 and 77.)

KONSTAN, J. A., MILLER, B. N., MALTZ, D., HERLOCKER, J. L., GORDON, L. R., AND RIEDL, J. 1997. Grouplens: applying collaborative filtering to usenet news. *Communications of the ACM 40,* 3, 77–87. (Cited on pages 22 and 77.)

KUMAR, A. AND BHATIA, M. 2012. Community expert based recommendation for solving first rater problem. *International Journal of Computer Applications 37,* 10, 7–13. (Cited on page 18.)

KURNIAWAN, S. H., SO, R., AND TSENG, M. 2006. Consumer decision quality in mass customization. *International Journal of Mass Customisation 1,* 2-3, 176–194. (Cited on page 89.)

LAI, W., HUA, X.-S., AND MA, W.-Y. 2006. Towards content-based relevance ranking for video search. In *Proceedings of the 14th annual ACM international conference on Multimedia.* MULTIMEDIA '06. ACM, New York, NY, USA, 627–630. (Cited on pages 31, 32, and 74.)

LARSON, K. AND SMITHWICK, D. 2010. Beyond the configurator: Collecting accurate data for an architectural design recommendation engine. *Working paper, MIT Department of Architecture.* (Cited on page 13.)

LEE, B.-K. AND LEE, W.-N. 2004. The effect of information overload on consumer choice quality in an on-line environment. *Psychology and Marketing 21,* 3, 159–183. (Cited on pages 1, 14, and 47.)

LEMIRE, D. AND MCGRATH, S. 2005. Implementing a rating-based item-to-item recommender system in php/sql. Tech. rep., Ondelette.com. (Cited on page 15.)

LEVIN, I. P. AND GAETH, G. J. 1988. How consumers are affected by the framing of attribute information before and after consuming the product. *Journal of Consumer Research 15,* 3, 374–378. (Cited on page 45.)

LEVIN, I. P., SCHNEIDER, S., AND GAETH, G. 1998. All frames are not created equal: A typology and critical analysis of framing effects. *Organizational Behavior and Human Decision Processes 76,* 2, 149–188. (Cited on pages xiii, 37, 44, 45, 46, 96, and 97.)

LEWIS, C. H. 1982. Using the "Thinking Aloud" Method In Cognitive Interface Design. Tech. rep., IBM. (Cited on page 54.)

LI, C. AND PEARL, P. 2006. Evaluating critiquing-based recommender agents. In *Proceedings of the 21st national conference on Artificial intelligence.* AAAI'06. 157–162. (Cited on page 5.)

LINDEN, G., SMITH, B., AND YORK, J. 2003. Amazon.com recommendations: item-to-item collaborative filtering. *Internet Computing, IEEE 7,* 1, 76–80. (Cited on pages 13, 17, 18, 21, 22, and 77.)

LUSSIER, D. A. AND OLSHAVSKY, R. W. 1979. Task complexity and contingent processing in brand choice. *Journal of Consumer Research 6,* 2, 154–65. (Cited on pages 33, 34, and 81.)

MA, Z., PANT, G., AND SHENG, O. R. L. 2007. Interest-based personalized search. *ACM Transactions on Information Systems 25,* 1, 1–18. (Cited on pages 14 and 15.)

MALHOTRA, N. K. 1982. Information load and consumer decision making. *Journal of Consumer Research 8,* 4, 419–430. (Cited on pages 2, 5, 28, and 89.)

MANDEL, N. AND JOHNSON, E. 1998. Constructing preferences online: Can web pages change what you want? *unpublished working paper*, 1–37. (Cited on pages 4, 36, and 51.)

MANDL, M. AND FELFERNIG, A. 2012a. Improving the performance of unit critiquing. *Proceedings of the 20th conference on User Modeling, Adaptation, and Personalization – to appear*. (Cited on pages 7 and 55.)

Mandl, M., Felfernig, A., and Schubert, M. 2009. Consumer decision making in knowledge-based recommendation. In *Proceedings of the 5th International Conference on Active Media Technology*. AMT '09. Springer-Verlag, Berlin, Heidelberg, 69–80. (Cited on pages 7, 13, and 33.)

Mandl, M., Felfernig, A., and Teppan, E. 2012c. *Consumer Decision Making on Configuration*. Configuration systems: Technologies and Business Cases – to appear. (Cited on pages 7 and 33.)

Mandl, M., Felfernig, A., Teppan, E., and Schubert, M. 2011a. Consumer decision making in knowledge-based recommendation. *Journal of Intelligent Information Systems 37,* 1, 1–22. (Cited on pages 7, 13, and 33.)

Mandl, M., Felfernig, A., and Tiihonen, J. 2011c. Evaluating design alternatives for feature recommendations in configuration systems. *E-Commerce Technology, IEEE International Conference*, 34–41. (Cited on pages 8, 9, and 81.)

Mandl, M., Felfernig, A., and Tiihonen, J. 2012b. *Personalized configuration*. Configuration systems: Technologies and Business Cases – to appear. (Cited on pages 6 and 13.)

Mandl, M., Felfernig, A., Tiihonen, J., and Isak, K. 2011b. Status quo bias in configuration systems. *Proceedings of the 24th international conference on Industrial engineering and other applications of applied intelligent systems*, 105–114. (Cited on pages 8, 10, 29, 66, 81, 83, and 91.)

Marteau, T. M. 1989. Framing of information: Its influence upon decisions of doctors and patients. *British Journal of Social Psychology 28,* 1, 89–94. (Cited on pages 37, 46, and 96.)

McCarthy, K., McGinty, L., and Smyth, B. 2005. Dynamic critiquing: An analysis of cognitive load. In *Proceedings of the 16th Irish Conference on Artificial Intelligence and Cognitive Science*. 19–28. (Cited on pages 2, 15, 25, 55, 56, 57, and 65.)

McCarthy, K., Reilly, J., McGinty, L., and Smyth, B. 2004. On the Dynamic Generation of Compound Critiques in Conversational Recommender Systems. In *Proceedings of the 3rd International Conference on Adaptive Hypermedia and Web-Based Systems (AH04)*. Springer-Verlag, 176–184. (Cited on pages 5, 25, 56, 64, and 103.)

McCarthy, K., Reilly, J., McGinty, L., and Smyth, B. 2005. Experiments in dynamic critiquing. In *Proceedings of the 10th international conference on Intelligent user interfaces*. IUI '05. ACM, New York, NY, USA, 175–182. (Cited on page 15.)

McCarthy, K., Salem, Y., and Smyth, B. 2010. Experience-based critiquing: Reusing critiquing experiences to improve conversational recommendation. In *ICCBR*. Lecture Notes in Computer Science, vol. 6176. Springer Berlin / Heidelberg, 480–494. (Cited on pages 5, 25, 27, 53, 55, 56, 57, 58, 60, 62, 63, 64, 65, and 78.)

McGinty, L. and Smyth, B. 2002. Comparison-based recommendation. *Lecture Notes in Computer Science 2416,* 1, 731–737. (Cited on page 15.)

McNee, S. M., Lam, S. K., Konstan, J. A., and Riedl, J. 2003. Interfaces for eliciting new user preferences in recommender systems. In *Proceedings of the 9th international conference on User modeling*. UM'03. Springer-Verlag, Berlin, Heidelberg, 178–187. (Cited on page 90.)

McNee, S. M., Riedl, J., and Konstan, J. A. 2006. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI '06 extended abstracts on Human factors in computing systems*. ACM, New York, NY, USA, 1097–1101. (Cited on pages 51, 53, and 78.)

McSherry, D. 2003. Similarity and Compromise. In *Proceedings of the Fifth International Conference on Case-Based Reasoning (ICCBR-03)*. Springer-Verlag, 291–305. (Cited on pages 29, 73, and 77.)

McSherry, D. 2004. Maximally successful relaxations of unsuccessful queries. In *15th Conference on Artificial Intelligence and Cognitive Science*. 127–136. (Cited on pages 42 and 102.)

McSherry, D. 2005. Incremental Nearest Neighbour with Default Preferences. In *Proceedings of the 16th Irish Conference on Artificial Intelligence and Cognitive Science*. 9–18. (Cited on pages 29, 30, and 77.)

Mittal, S. and Frayman, F. 1989. Towards a Generic Model of Configuration Tasks. In *11th International Joint Conference on Artificial Intelligence (IJCAI 1989)*. 1395–1401. (Cited on pages 27 and 77.)

Morgenstern, O. and Neumann, J. V. 1944. *Theory of Games and Economic Behavior*. Princeton University Press. (Cited on page 34.)

Murphy, J., Hofacker, C. F., and Mizerski, R. 2006. Primacy and recency effects on clicking behavior. *Journal of Computer-Mediated Communication 11,* 2, 522–535. (Cited on pages 31, 37, 41, 42, 43, 74, and 82.)

Nicosia, F. 1966. *Consumer decision process: Marketing and advertising implications*. Prentice-Hall behavioral sciences in business series. Englewood Cliffs, N.J. : Prentice-Hall. (Cited on pages 35 and 37.)

O'Sullivan, B., Papadopoulos, A., Faltings, B., and Pu, P. 2007. Representative explanations for over-constrained problems. In *Proceedings of the National Conference on Artificial Intelligence*. American Association for Artificial Intelligence, Menlo Park, United States, 323–328. (Cited on page 102.)

Papagelis, A., Papagelis, M., and Zaroliagis, C. 2008. Enabling social navigation on the web. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*. IEEE Computer Society, Washington, DC, USA, 162–168. (Cited on pages 37 and 49.)

Papagelis, M. and Plexousakis, D. 2005. Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents. *Engineering Applications of Artificial Intelligence 18,* 7, 781–789. (Cited on pages 3 and 51.)

Papagelis, M., Plexousakis, D., and Kutsuras, T. 2005. Alleviating the sparsity problem of collaborative filtering using trust inferences. In *Proceedings of the Third international conference on Trust Management*. iTrust'05. Springer-Verlag, Berlin, Heidelberg, 224–239. (Cited on page 18.)

Payne, J. W. 1976. Task complexity and contingent processing in decision making: An information search and protocol analysis. *Organizational Behavior and Human Decicion Processes 16,* 2, 366–387. (Cited on pages 6, 7, 33, 34, 81, 82, and 101.)

Payne, J. W., Bettman, J. R., and Johnson, E. J. 1993. *The adaptive decision maker*. New York: Cambridge University Press. (Cited on page 35.)

Pazzani, M. and Billsus, D. 1997. Learning and revising user profiles: The identification of interesting web sites. *Machine Learning 27,* 3, 313–331. (Cited on pages 19 and 22.)

Pazzani, M. J. 1999. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review 13,* 5-6, 393–408. (Cited on page 77.)

Pazzani, M. J. and Billsus, D. 2007. Content-based recommendation systems. In *The Adaptive Web*. Lecture Notes in Computer Science, vol. 4321. Springer-Verlag, Berlin, Germany, Chapter 10, 325–341. (Cited on pages 19, 20, 21, and 22.)

Peppers, D., Rogers, M., and Rogers, M. 1996. *The one to one future: building relationships one customer at a time*. One to One Future Series. Currency Doubleday. (Cited on page 14.)

Piller, F. and Franke, N. 2002. Configuration toolkits for mass customization: Setting a research. *Arbeitsbericht Nr 33 Okt 2002 des Lehrstuhls für Allgemeine und Industrielle Betriebswirtschaftslehre der Technischen Universität München 33*. (Cited on page 28.)

Pine, B. 1993. *Mass customization – The New Frontier in Business Competition*. Harvard Business School Press. (Cited on pages 2, 28, and 65.)

Pine, B. and Gilmore, J. 1999. *The experience economy: work is theatre & every business a stage*. Harvard Business School Press. (Cited on page 28.)

Pinón, A. and Gärling, T. 2004. *Effects of mood on adoption of loss frame in risky choice*. Göteborg psychological reports. Göteborg University. (Cited on page 45.)

Piotte, M. and Chabbert, M. The pragmatic theory solution to the netflix grand prize, in: Netflix prize documentation. Tech. rep., Pragmatic Theory Inc. (Cited on page 60.)

Pu, P. and Chen, L. 2005. Integrating tradeoff support in product search tools for e-commerce sites. In *Proceedings of the 6th ACM conference on Electronic commerce*. EC '05. ACM, New York, NY, USA, 269–278. (Cited on page 27.)

Pu, P. and Chen, L. 2010. A user-centric evaluation framework of recommender systems. In *Proceedings of the ACM RecSys 2010 Workshop on User-Centric Evaluation of Recommender Systems and Their Interfaces (UCERSTI)*. 14–21. (Cited on pages 14, 51, 53, 75, 78, 84, and 101.)

Pu, P. and Faltings, B. 2000. Enriching buyers' experiences: the smartclient approach. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. CHI '00. ACM, New York, NY, USA, 289–296. (Cited on page 56.)

Pu, P. H. Z. and Kumar, P. 2004. Evaluating example-based search tools. In *Proceedings of the 5th ACM conference on Electronic commerce*. EC '04. ACM, New York, NY, USA, 208–217. (Cited on page 27.)

Reilly, J., McCarthy, K., McGinty, L., and Smyth, B. 2004. Dynamic critiquing. In *Advances in Case-Based Reasoning*. Lecture Notes in Computer Science, vol. 3155. Springer Berlin / Heidelberg, 37–50. (Cited on page 27.)

Reilly, J., McCarthy, K., McGinty, L., and Smyth, B. 2005. Incremental critiquing. In *Research and Development in Intelligent Systems XXI*. Springer London, 101–114. (Cited on pages 27, 56, and 58.)

Reilly, J., Zhang, J., McGinty, L., Pu, P., and Smyth, B. 2007a. A comparison of two compound critiquing systems. In *In Proceedings of the 12th International Conference on Intelligent User Interfaces (IUI'07)*. ACM Press, 317–320. (Cited on pages 5, 26, and 103.)

Reilly, J., Zhang, J., McGinty, L., Pu, P., and Smyth, B. 2007b. Evaluating compound critiquing recommenders: a real-user study. In *Proceedings of the 8th ACM conference on Electronic commerce*. EC '07. ACM, New York, NY, USA, 114–123. (Cited on pages 2, 26, and 56.)

Reiter, R. 1987. A theory of diagnosis from first principles. *Artificial Intelligence 32,* 1, 57–95. (Cited on pages xii, 31, 66, 69, 71, and 72.)

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. 1994. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*. CSCW '94. ACM, New York, NY, USA, 175–186. (Cited on page 17.)

Ricci, F. and Nguyen, Q. N. 2007. Acquiring and revising preferences in a critique-based mobile recommender system. *IEEE Intelligent Systems 22,* 3, 22–29. (Cited on page 5.)

Richarme, M. 2004. Consumer decision-making models, strategies, and theories, oh my! *Decision Analyst*. (Cited on page 34.)

Ritov, I. and Baron, J. 1992. Status-quo and omission biases. *Journal of Risk and Uncertainty 5,* 1, 49–61. (Cited on pages 7, 10, 37, 48, 81, 83, 90, 91, 93, and 101.)

Roe, R. M., Busemeyer, J. R., and Townsend, J. T. 2001. Multialternative decision field theory: a dynamic connectionist model of decision making. *Psychol Rev 108,* 2, 370–92. (Cited on page 40.)

Rokach, L. 2009. Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography. *Computational Statistics & Data Analysis 53,* 12, 4046–4072. (Cited on page 60.)

Roy, M. C., Dewit, O., and Aubert, B. 2001. The impact of interface usability on trust in web retailers. *Internet Research 11,* 5, 388–398. (Cited on pages 4, 7, 53, and 82.)

Russo, J. E. 1974. More information is better: A reevaluation of jacoby, speller and kohn. *Journal of Consumer Research 1,* 3, 68–72. (Cited on page 47.)

Sabin, D. and Weigel, R. 1998. Product configuration frameworks-a survey. *IEEE Intelligent Systems 13,* 4, 42–49. (Cited on pages 1, 2, 14, 27, 65, and 77.)

Salamó, M., Smyth, B., McCarthy, K., Reilly, J., and McGinty, L. 2005. Reducing critiquing repetition in conversational recommendation. In *In Proceedings of the IJCAI'2005 Workshop on Multi-Agent Information Retrieval and Recommender Systems.* 55–61. (Cited on page 55.)

Salgado, M. 2006. Choosing to have less choice. Working paper, Fondazione Eni Enrico Mattei. (Cited on pages 47 and 48.)

Samuelson, W. and Zeckhauser, R. 1988. Status quo bias in decision making. *Journal of Risk and Uncertainty 1,* 1, 7–59. (Cited on pages 7, 37, 48, 81, 90, and 93.)

Schafer, J. B., Konstan, J. A., and Riedl, J. 2001. E-commerce recommendation applications. *Data Min. Knowl. Discov. 5,* 1-2, 115–153. (Cited on pages 1, 14, and 28.)

Schein, A. I., Popescul, A., Ungar, L. H., and Pennock, D. M. 2002. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval.* SIGIR '02. ACM, New York, USA, 253–260. (Cited on pages 18 and 19.)

Schiffman, L. and Kanuk, L. 1997. *Consumer behavior.* Prentice Hall. (Cited on pages xi and 35.)

Schubert, M., Felfernig, A., and Reinfrank, F. 2011. Reaction: Personalized minimal repair adaptations for customer requests. In *Flexible Query Answering Systems.* Lecture Notes in Computer Science, vol. 7022. Springer Berlin / Heidelberg, 13–24. (Cited on page 102.)

Schulz, A. G. and Hahsler, M. 2002. Evaluation of Recommender Algorithms for an Internet Information Broker based on Simple Association Rules and on the Repeat-Buying Theory. In *Fourth WebKDD Workshop: Web Mining for Usage Patterns & User Profiles*. Edmonton, Canada, 100–114. (Cited on page 52.)

Shani, G. and Gunawardana, A. 2009. Evaluating recommendation systems. *Recommender Systems Handbook*, 1–41. (Cited on pages xiii, 51, 52, 54, and 77.)

Simon, H. A. 1956. Rational choice and the structure of the environment. *Psychological Review 63,* 2, 129–38. (Cited on pages 34 and 35.)

Simonson, I. 1989. Choice based on reasons: The case of attraction and compromise effects. *Journal of Consumer Research 16,* 2, 158–74. (Cited on pages xi, xiii, 38, and 39.)

Simonson, I. and Tversky, A. 1992. Choice in context: Tradeoff contrast and extremeness aversion. *Journal of Marketing Research 29,* 3, 281–295. (Cited on pages xiii, 37, 38, 40, 95, and 96.)

Smyth, B. and Keane, M. T. 1996. Using adaptation knowledge to retrieve and adapt design cases. *Journal of Knowledge-based Systems 9,* 2, 127–135. (Cited on pages 29 and 77.)

Smyth, B. and McGinty, L. 2003. An Analysis of Feedback Strategies in Conversational Recommender Systems. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Cognitive Science (AICS-2003).* 211–216. (Cited on pages 4, 15, and 64.)

Solomon, M. 2002. *Consumer behavior: buying, having, and being.* Prentice Hall international series in marketing. Prentice Hall. (Cited on pages xi and 35.)

STREUFERT, S. AND DRIVER, M. J. 1965. Conceptual structure, information load and perceptual complexity. *3,* 6, 249–250. (Cited on pages 2, 28, and 89.)

STUMPTNER, M. An overview of knowledge-based configuration. *AI Communications 10,* 2, 111–125. (Cited on pages 27 and 66.)

SUMMERS, J. O. 1974. Less information is better? *Journal of Marketing Research 11,* 4, 467–468. (Cited on page 47.)

SVENSSON, M., HÖÖK, K., LAAKSOLAHTI, J., AND WAERN, A. 2001. Social navigation of food recipes. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. CHI '01. ACM, New York, NY, USA, 341–348. (Cited on pages 50 and 51.)

TEPPAN, E. AND FELFERNIG, A. 2009a. The asymmetric dominance effect and its role in e-tourism recommender applications. In *Wirtschaftsinformatik'09*. Springer, 791–800. (Cited on pages 40 and 41.)

TEPPAN, E. C. AND FELFERNIG, A. 2009b. Calculating decoy items in utility-based recommendation. In *IEA/AIE*. Lecture Notes in Computer Science, vol. 5579. Springer, 183–192. (Cited on pages 37, 40, and 96.)

TEPPAN, E. C. AND FELFERNIG, A. 2009c. Impacts of decoy elements on result set evaluations in knowledge-based recommendation. *International Journal of Advanced Intelligence Paradigms 1,* 3, 358–373. (Cited on pages 37, 40, 95, and 96.)

TERVEEN, L. AND HILL, W. 2001. Beyond recommender systems: Helping people help each other. In *HCI in the New Millennium*. Addison-Wesley, 487–509. (Cited on pages 1, 3, 16, 19, and 66.)

TIIHONEN, J. AND FELFERNIG, A. 2010. Towards recommending configurable offerings. *International Journal of Mass Customisation 3,* 4, 389–406. (Cited on pages 6, 72, 77, and 100.)

TÖSCHER, A. AND JAHRER, M. 2008. The bigchaos solution to the netflix prize 2008. Tech. rep., commendo research & consulting. (Cited on page 60.)

TSANG, E. P. K. 1993. *Foundations of constraint satisfaction*. Computation in cognitive science. Academic Press. (Cited on page 24.)

TSENG, M. AND JIAO, J. 2001. Mass customization. In *Handbook of Industrial Engineering*, G. Salvendy, Ed. Wiley-Interscience. (Cited on pages 2, 28, and 65.)

TVERSKY, A. AND KAHNEMAN, D. 1974. Judgment under Uncertainty: Heuristics and Biases. *Science 185,* 4157, 1124–1131. (Cited on page 48.)

TVERSKY, A. AND KAHNEMAN, D. 1981. The framing of decisions and the psychology of choice. *Science, New Series 211,* 4481, 453–458. (Cited on pages 33, 34, 37, 44, 45, 46, 81, 96, and 97.)

TVERSKY, A. AND KAHNEMAN, D. 1986. Rational choice and the framing of decisions. *Journal of Business 59,* 4, 251–278. (Cited on pages 37, 44, 46, 96, and 97.)

VIAPPIANI, P., PU, P., AND FALTINGS, B. 2007. Conversational recommenders with adaptive suggestions. In *Proceedings of the 2007 ACM conference on Recommender systems*. RecSys '07. ACM, New York, NY, USA, 89–96. (Cited on pages 2, 5, 15, 25, 37, 49, 55, and 103.)

von Winterfeldt, D. and Edwards, W. 1986. *Decision Analysis and Behavioral Research*. Cambridge University Press. (Cited on pages 32 and 42.)

Wason, P. C. 1960. On the failure to eliminate hypotheses in a conceptual task. *The Quarterly Journal of Experimental Psychology 12,* 3, 129–140. (Cited on page 46.)

Wicaksono, H., Schubert, V., Rogalski, S., Laydi, Y. A., and Ovtcharova, J. 2012. Ontology-driven requirements elicitation in product configuration systems. In *Enabling Manufacturing Competitiveness and Economic Sustainability*, H. A. ElMaraghy, Ed. Springer Berlin Heidelberg, 63–67. (Cited on pages 2 and 3.)

Williams, M. and Hollan, J. 1981. The process of retrieval from very long term memory. *Cognitive Science 5,* 2, 87–119. (Cited on pages 25 and 26.)

Wilson, D. R. and Martinez, T. R. 1997. Improved heterogeneous distance functions. *Journal of Artificial Intelligence Research (JAIR) 6,* 1, 1–34. (Cited on page 73.)

Xiao, B. and Benbasat, I. 2007. E-commerce product recommendation agents: Use, characteristics, and impact. *MIS Quarterly 31,* 1, 137–209. (Cited on pages 14 and 36.)

Yang, Y., Zhang, X., Liu, F., and Xie, Q. 2005. An internet-based product customization system for cim. *Robotics and Computer-Integrated Manufacturing 21,* 2, 109–118. (Cited on pages 2 and 28.)

Yoon, S. and Simonson, I. 2008. Choice set configuration as a determinant of preference attribution and strength. *Journal of Consumer Research 35,* 2, 324–336. (Cited on pages 37, 95, and 96.)

Zhang, J. and Pu, P. 2006. A comparative study of compound critique generation in conversational recommender systems. In *Proceedings of the 4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH2006)*. Springer, 234–243. (Cited on pages 5, 25, 26, 27, 56, 64, and 103.)