SUPPORTING PEOPLE IN FINDING INFORMATION

HYBRID RECOMMENDER SYSTEMS AND GOAL-BASED STRUCTURING

MARK VAN SETTEN

Telematica Instituut,
On top of technology.

This publication is part of the Telematica Instituut's Fundamental Research Series. Telematica Instituut is a unique, co-operative venture involving businesses, scientific research institutions and the Dutch government, which carries out research into telematics, a field also known as information and communication technology (ICT).

The institute focuses on the rapid translation of fundamental knowledge into market-oriented applications such as electronic business practice, electronic co-operation and electronic data retrieval. We are also involved in the development of middleware.

In a financial, administrative and substantive alliance, the parties involved work on a joint mission, namely the development and application of high-quality telematics knowledge. By combining forces in an international network of this kind, we will be in a better position to deploy the opportunities of telematics so that ICT can make a contribution to strengthening the innovative capacity of business and industry.

In addition, telematics will fundamentally change and facilitate our way of living, working and even spending our leisure time. In all of these activities, technology remains the servant of man: On top of technology. The Dutch government also recognizes the importance of information and communication technology, and awarded Telematica Instituut the title 'leading technological institute'.

www.telin.nl

Supporting People In Finding Information

Hybrid Recommender Systems and Goal-Based Structuring

Mark van Setten

Telematica Instituut

Enschede, The Netherlands, 2005

Telematica Instituut Fundamental Research Series, No. 016 (TI/FRS/016)

Cover Design: Studio Oude Vrielink, Losser and Jos Hendrix, Groningen
Book Design: Lidwien van de Wijngaert and Henri ter Hofte
Printing: Universal Press, Veenendaal, The Netherlands

Cover Photo: the photo on the cover is a scan of a painting made by the author himself. It depicts a piece of the *Posbank*, a beautiful nature reserve close to the village where the author was born. As one of the main themes in this thesis is personalization, a painting created by the author gives the thesis a more personal touch to the author.

Samenstelling promotiecommissie:

Voorzitter, secretaris:	prof. dr. ir. A.J. Mouthaan (Universiteit Twente)
Promotor:	prof. dr. ir. A. Nijholt (Universiteit Twente)
Assistent promotor:	dr. M.J.A. Veenstra (Telematica Instituut)
	dr. E.M.A.G. van Dijk (Universiteit Twente)
Leden:	prof. dr. C.A. Vissers (Universiteit Twente)
	prof. dr. W. Jonker (Universiteit Twente)
	prof. dr. L. Ardissono (Università degli Studi di Torino, Italy)
	prof. dr. S. Brinkkemper (Universiteit Utrecht)
	prof. dr. F.M.G. de Jong (Universiteit Twente)
Paranimfen:	drs. M.L. Nahapiet en R.C. Waanders

ISSN 1388-1795; No. 016
ISBN 90-75176-89-9

Telematica Instituut, P.O. Box 589, 7500 AN Enschede, The Netherlands
E-mail: info@telin.nl; Internet: http://www.telin.nl
Telephone: +31 (0)53-4850485; Fax: +31 (0)53-4850400

SUPPORTING PEOPLE IN FINDING INFORMATION
HYBRID RECOMMENDER SYSTEMS AND GOAL-BASED STRUCTURING

PROEFSCHRIFT

ter verkrijging van
de graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
prof. dr. W.H.M. Zijm,
volgens besluit van het College voor Promoties
in het openbaar te verdedigen
op vrijdag 16 december 2005 om 13.15 uur

door

Mark van Setten
geboren op 25 december 1971
te De Steeg

Dit proefschrift is goedgekeurd door de promotor, prof. dr. ir. A. Nijholt en de assistent promotoren, dr. M.J.A. Veenstra en dr. E.M.A.G. van Dijk

Preface

During college, I never imagined that I would write a PhD thesis. My educational career has always been one of small steps. However, when provided with the opportunity to start a PhD while working at the Telematica Instituut, I accepted it and have enjoyed it immensely, even though there have been some rough times. I was provided with the opportunity to search for a topic that I believed was both interesting and necessary to research. I was soon drawn to the areas of information retrieval and user-centred design; in combination this lead to the topic of personalization and in particular the question of how to help people find information that is of their interest. Throughout my research, I have tried to keep into mind that in the end, personalization research must lead to solutions that support real people in their struggle to find interesting information; people matter, both as a reason to do research for as well as a source of support. This thesis would have never been finished without the support of other people.

First of all, I would like to thank Mettina Veenstra for her support and daily guidance throughout these years. I have always valued her ideas, comments and guidance and I really appreciate the way she allowed me to explore my own path throughout this research, giving suggestions were appropriate. Also many thanks to Anton Nijholt, my PhD supervisor, for his support and for being able to ask me those questions that triggered me to think about alternatives and other ways of looking at the subject at hand. Betsy van Dijk has been extremely helpful when designing user experiments and helped me a lot when writing papers.

I also owe a lot to Jeroen van Barneveld, whom I had the honour of supervising when working on his master's thesis. He has done a wonderful job when we were investigating the user interface aspects in this research. I also value the work done by Jaap Reitsma and Peter Ebben when turning my experimental software into a professional toolkit for the development of

prediction engines. I am also grateful to Omroep.nl who have provided me with the necessary data for my experiments.

I also like to thank my colleagues at the Telematica Instituut for providing such a stimulating, interesting and fun environment to do my research in. Especially my office mate Margit for our many discussions, both on and off topic, Henk for reviewing a draft of my thesis, Harry for giving me insightful views on the media domain, Henny for her support on statistical issues and Martin for his help on programming issues. Daan and Ferial provided a lot of support at the start of my PhD research when I was trying to focus my research. Also thanks to my fellow PhD students working at the Telematica Instituut, some who are still working on their thesis and some who have finished before me.

This thesis would have never been realized without the support of my family and friends, especially Leon who has stood by my side all these years. Whatever happens, I know we will always be there for each other. Also a big thanks to my mother Ria and my brother Wilfred, who have always supported me in whatever I decided to do, even though they have been through rough times themselves. Also thanks to my father Jan, Lies and Ben and the rest of my family.

I am also grateful to Mary Black and Frances Black for their friendship and I feel blessed that they have allowed me to become part of their music. This has given me much pleasure and provided me with much needed distraction every now and then. To Rob, Marcel, Anke, Miranda, Frans and all my other friends, I say thank you for being there during this hectic time and for all the fun we had.

Mark van Setten
Enschede, the Netherlands, November 2005

Contents

Chapter 1

Introduction

The past decades have brought major technological developments that influence the daily lives of millions of people. One of these developments has been the Internet. The Internet allows people to easily and cheaply communicate with each other using technologies such as e-mail, voice-over-IP and instant messaging. Furthermore, it provides retailers with opportunities to reach larger customer populations using online stores; examples are Amazon.com, Bol.com and Walmart.com that sell a wide variety of products or specialized online shops such as Interflora.com selling flowers or internationalmale.com selling men's clothing. The Internet also offers individuals the opportunity to offer their goods for sale to large populations via auctioning websites such as eBay or Marktplaats.nl. Access to news sources has also been made easier and faster via news portals on the Internet, such as CNN online and the BBC news online.

This is all made possible as the Internet allows one to publish, distribute and search for information, products and services more easily and more cheaply. How easy it is to publish and distribute information is noticeable in the numerous websites created by individuals and companies about almost every topic imaginable and in the number of articles available in digital libraries such as those of the ACM, IEEE and the New York Public Library online. How much information and services are available via the Internet becomes apparent from the number of existing web pages; the exact number is unknown as it continuously changes, but on 11 May 2005, Google had indexed 8.058.044.651 pages.

In this thesis, we address solutions to help people find information, products and services that are interesting to them as the enormous availability is not only a blessing to those in search of something, it has also become a challenge to find that which is interesting within that vast amount of available information, products and services.

1.1 Background and focus

1.1.1 Information overload

In many situations, there are so many options to choose from that people feel lost and have difficulties making decisions; e.g. when buying a CD, there are not only a very large number of artists and albums to choose from, there are also numerous (electronic) shops that can be used to search for CD's and to buy CD's. How does someone know if he will like an artist he never heard of before and which album of that artist can best be bought? And even if it is clear which CD to buy, the question remains which shop is the most likely to have the CD in stock, is reliable and asks a competitive price for the CD. The same goes for buying other products, like books, movies or electronic equipment.

Also when searching for information about a topic on the Internet there is so much information available that it is difficult to find those web pages, books, papers, articles, music, videos etc. that are relevant to the topic one is searching for; e.g. most search engines on the Internet return hundreds to thousands of results on every query, while only a few of those results are really relevant to the searcher and that are not always the results at the top of the result list. Furthermore, what is relevant and interesting for one searcher may not be relevant and interesting to another searcher, even if they post the same query.

The problem of 'too much to choose from' not only occurs when searching for something, it also occurs when trying to manage the various streams of information that reach people daily, e.g. e-mail, news stories and television. In those streams, people find it increasingly difficult to separate the interesting or important from the non-interesting and irrelevant; e.g. distinguishing between interesting news stories and uninteresting ones or separating important e-mail from junk mail.

Too much choice not only leads to feelings of loosing control and being unable to handle the amount of information, it also leads to anxiety generated by worrying whether something interesting or important is being missed (Edmunds & Morris, 2000, page 19). This problem is often referred to as the *information overload* problem. Information overload refers to "the state of having too much information to make a decision or remain informed about a topic. Large amounts of historical information to dig through, a high rate of new information being added, contradictions in available information, a low signal-to-noise ratio making it difficult to identify what information is relevant to the decision, or the lack of a method for comparing and processing different kinds of information can all contribute to this effect" (Wikipedia, 2005a). Goulding (2001) argues that

those suffering information overload may suffer the same fate as the information poor (people who have no access to information): both are unable to make well-informed decisions; the information poor due to the lack of information and those who experience information overload as they are unable to separate relevant information from junk.

According to Bawden, Holtman & Courtney (1999, page 252-253) one single solution to the information overload problem does not suffice; only a combination of several solutions will help to solve the problem; both managerial and technical solutions. Solutions on the managerial side are concerned with educating people, ranging from teaching people simple skills such as how to handle incoming e-mail and teaching them to sparsely join mailing lists to teaching people how to determine what they need, where they need to look for information that meets those needs and how to evaluate the usefulness of retrieved information; the latter is often referred to as information literacy.

According to Bawden et al. (1999), technology is for a large part responsible for the information overload problem, both due to making it easier to publish information and thus increasing the amount of available information as well as making it easier for people to access information. However, Bawden et al. (1999) indicate that technology can also provide solutions. This thesis addresses several technical solutions that contribute to help people deal with the information overload problem by supporting them in finding interesting information.

1.1.2 Information retrieval

Finding information is studied in the research of the domain of information retrieval, which is "… the art and science of searching for information in documents, searching for documents themselves, searching for metadata which describes documents, or searching within databases, whether relational stand alone databases or hypertext networked databases such as the Internet or intranets, for text, sound, images or data" (Wikipedia, 2005b). When people search for information, their ultimate interest is not always in the information itself, but in the object that the information represents; e.g. a CD instead of the title and track list of the CD as it is listed in an online shop or a person instead of the profile describing that person. For this reason, in this thesis the generic term *item* is used.

Definition 1 Item

An item refers to the information about an object as well as the object itself, where an object can be an electronic document, a product, a person, a service or anything that can be represented by information.

An item can be regarded at various abstraction levels: an item, a subsection of an item or a collection of items. Depending on what someone is looking for, each of these abstraction levels can be referred to as the item; e.g. if one is looking for videos, an item refers to one video; if one is looking for video scenes, an item refers to one scene; if one is looking for video collections, an item refers to one collection.

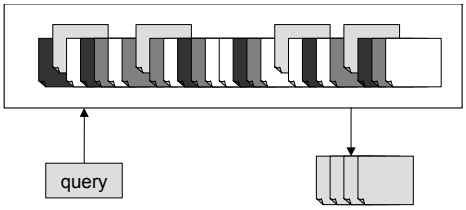


Figure 1-1 Information Retrieval

There are three approaches that people can use to find items: the retrieval approach, the filtering approach and the browsing approach. The retrieval approach is also called information retrieval (see *Figure 1-1*), where "the system is presented with a query by the user and [is] expected to produce information sources which the user finds useful" (Oard, 1997, page 2). The focus with information retrieval lies on the user's short-term and immediate interests; the user informs the information retrieval system what items he is looking for using some sort of explicit formulation: the query. With information retrieval, information is delivered using an information pull method: the user asks, the system provides. Well-known examples of information retrieval systems are Internet search engines and online telephone books.

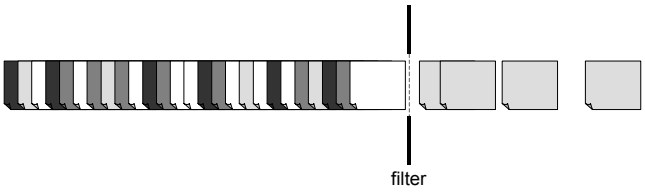


Figure 1-2 Information Filtering

Information systems that employ the filtering approach, called information filtering systems (see *Figure 1-2*), "… are typically designed to sort through large volumes of dynamically generated information and present the user with sources of information that are likely to satisfy his or her information requirement" (Oard, 1997, page 2). With information filtering, the source of information is often dynamic; new items are added or updated regularly and the user wants to be informed when new items appear that are of his interests. Examples of typical dynamic information sources where new information is added regularly are news websites, TV guides and e-mail. As

information filtering systems focus on informing users of new or updated items, they mainly address the user's long-term interests; interests that are stable over a longer period of time (Billsus & Pazzani, 1999); e.g. a person's interest in political news. This compared to short-term interests and immediate interests. Short-term interests are only valid for a short period of time, e.g. a person's interest in the weather at his holiday destination, while immediate interests are the interest of a person at a specific moment in time, e.g. the departure time of the next train to Amsterdam. Long-term interests of users can either be formulated explicitly by the user or be derived implicitly by the system. With information filtering, information is delivered using an information push method: the system sends new or updated information to the user, without the user having to ask for it.

Browsing (see *Figure 1-3*) is a process in which query formulation is tightly integrated with results presentation. Browsing is mostly used for explorative or muddled topical needs (Ingwersen, 1992, page 117); the user does not know precisely what he is looking for, hence, he explores the available information space in the hope to find something interesting. The most well known electronic browsing environment is the World Wide Web, where pages of information are linked together. People browse through this vast collection of information by following these links. With browsing, users implicitly formulate what they are looking for by clicking on links.

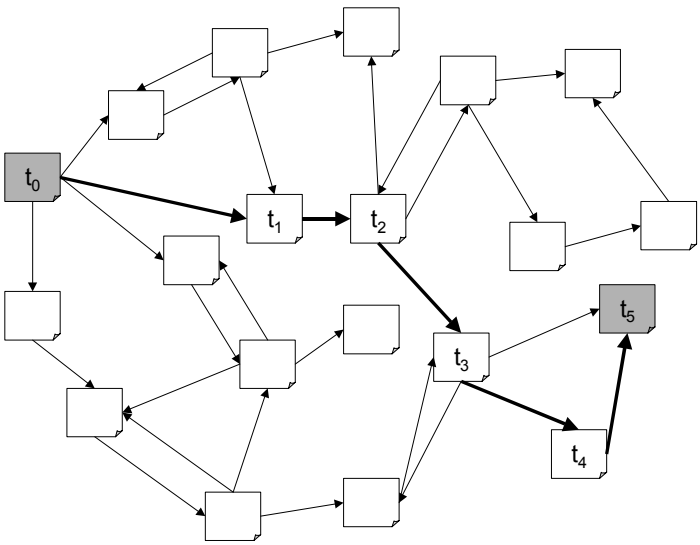


Figure 1-3 Browsing. User browses on a website from page t_0 to page t_5

The three retrieval approaches support people in finding interesting items by allowing them to search for or browse through items or to be kept up to date on new items. However, the approaches are only part of the solution to solve information overload; they can sometimes even increase the feeling

of information overload; e.g. even the best search engines on the Internet often return hundreds to thousands of results in response to a query, while only a handful of results are actually useful to the user; users still have to go through this set to determine how useful each result is to them.

Two characteristics of traditional information systems lie at the basis of this problem: they are objective and unaware of the semantics of the information and the user's request; when two users submit the same query, they receive the same results, even if their needs and interests are different; e.g. when asking for 'Mary Black', user A may be interested in the Irish singer 'Mary Black', while user B may be looking for the Mary Black Foundation in South Carolina, USA. Traditional information systems are not aware of the difference in interests of the two users and are also not aware of the semantic difference between the singer Mary Black and the foundation Mary Black. The same issue occurs when browsing, e.g. on the World Wide Web, all users see the same pages, with all the same links; that some links are more relevant or interesting to some users than to others is not taken into account. Even with standard information filtering techniques, e.g. term-frequency-inverse-document frequency (tf-idf) (Houseman & Kaskela, 1970), which do provide some level of subjectivity as each user can have his own profile which is used to filter information, objectivity is an issue: the same terms added to user profiles of different users are dealt with exactly the same.

1.1.3 Personalized information systems

This thesis does not focus on providing solutions to make information systems semantically aware, which is the topic of semantic web research (Berners-Lee, Hendler & Lassila, 2001); this thesis addresses solutions to make information systems aware of the interests and needs of their users and adapt their behaviour based on this knowledge about the user; i.e. making information systems more subjective. Such systems are called adaptive or personalized information systems; adaptive as they dynamically change their behaviour; personalized as they change their behaviour to fit the needs of each single individual person using the system.

As personalized information systems are concerned with adapting themselves and the information about items they retrieve to better fit the needs of the user, they need to have knowledge about the characteristics, interests, and preferences of their users. This knowledge about the user is often stored in so called user profiles, which are representations of the user in the system. The acquisition of knowledge about the user, what knowledge to acquire and how to model and represent this knowledge are all part of the research area of user modelling.

One way for personalized information systems to adapt to a user is to adapt the way information about items is structured. Information on the Internet is often not published as one big chunk, but rather as a set of related pieces of information, which are linked together using so-called hyperlinks. The way various pieces of information are grouped on one page and the links between multiple related pages forms the structure of a website. This structure can be adapted to better suit a person's needs, resulting in what is called adaptive hypermedia.

Another important aspect of a personalized information system that can be adapted to better fit the user's needs is the user interface. The user interface consists of those aspects of a system that are directly visible to the user and via which the user can interact with the system; it contains the way information and functionality is presented to the user, using devices such as displays and soundcards, and the way a user can interact with the system, using devices such as a mouse, a keyboard and voice input. Adapting the user interface means that the way information and functionality is presented and the way the user can interact with the system is different for each user. This topic is part of the research area of intelligent user interfaces.

Personalized information systems can also adapt the information retrieval process in order to better meet the needs of the user. In that case, the retrieval of information (either using the retrieval, filter or browsing approach) is made more personal by using knowledge about the user to determine what items are interesting for each user individually and to suggest the most interesting items to the user. This is the area of research that deals with recommender systems. Recommender systems can be defined as systems that "produce individualized recommendations as output or have the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options" (Burke, 2002, page 331).

Recommender systems, adaptive hypermedia and intelligent user interfaces correspond to three important processes (see *Figure 1-4* for our visualisation of these processes) within information systems: selection, structuring and presentation; a fourth process can also be distinguished, namely the transformation process.

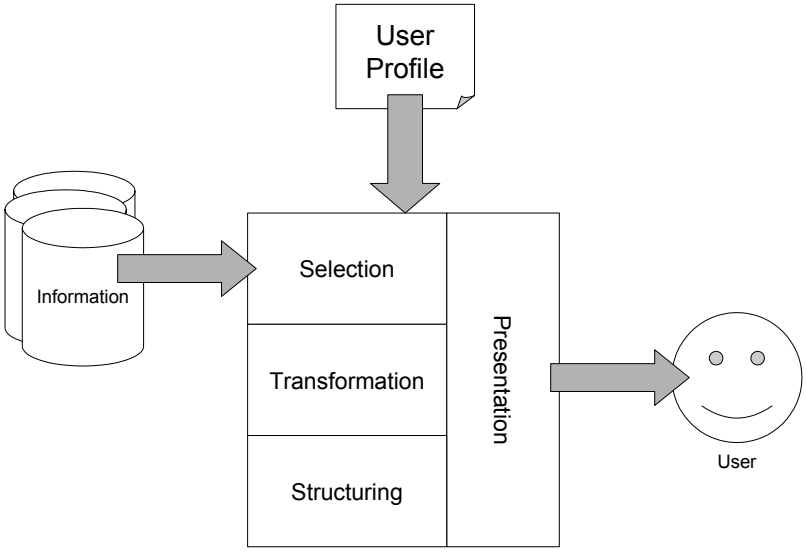


Figure 1-4 Processes in an information system where adaptation can take place

In each of the four processes adaptation can take place to provide results that better meet the needs of a user:

1. In the *selection* process, the information system determines what items are interesting or relevant enough for a user and removes all other items from the retrieved set of items.
2. In the *transformation* process, retrieved items are transformed into another form than in which they were retrieved; examples of possible transformations are translations, summarizing, modality transformations or quality changes. Transformations are optional and are not addressed any further in this thesis.
3. The *structuring* process is concerned with grouping items into coherent groups, sorting groups and items within groups and linking various items together to provide a navigation structure that best fits the users' needs.
4. The final process *presentation* is concerned with presenting the retrieved, structured and possibly transformed items to the user; it deals with issues such as layout, document formats, colours, fonts and presentation medium.

The order of the first three processes is not necessarily sequential; the order can vary and may iterate, although selection is always the first process, just as presentation is always the final process. Furthermore, each of the processes can put constraints on the other processes, e.g. when items have to be presented on a small handheld device with limited screen space, another way of structuring items may be required than when the same items have to be presented on a computer display.

As we try to find solutions to solve the information overload problem by helping people find those items that are of interest to them, our research focus lies mainly on recommender systems. Even though the main focus lies on recommender systems, aspects of user profiles, structuring and presentation are also addressed in this thesis as they are an essential part of a personalized information system and all contribute to provide support for users in finding interesting items. However, these processes are always addressed from the point of view of a recommender system; i.e. user profiles as a tool for maintaining knowledge about the interests of users during the selection process, structuring that is concerned with structuring items that a recommender system has determined as being of interest to the user in such a way that it makes it easier for people to find those items, and presentation as far as it concerns the presentation of recommendations and how people interact with them. For this reason, the objectives of this research cover the three main processes in an information system with recommender systems as a basis.

1.2 Objective and research questions

Main objective

The main objective of this research is to investigate and develop solutions that support people in finding interesting information.

As mentioned in the previous section, we use recommender systems to reach this objective. Several research groups and projects have been working on recommender systems and more specifically on recommendation techniques; techniques that determine which items are interesting for a user and that recommend those to the user. However, research focus has shifted from single recommendation techniques to combining recommendation techniques, into so-called hybrid recommender systems, in order to provide more accurate recommendations (Burke, 2002). Related work on recommender systems is described in chapter 2.

Research objective 1

Many hybrid recommenders have been developed for specific applications or domains; our first research objective is to develop a domain-independent framework that describes how to create hybrid recommender systems that can be tailored to various domains in order to provide more accurate recommendations. In order to reach this objective, the following research questions have to be answered:

1. What methods exist to combine recommendation techniques and which of these methods is best suited for developing a domain-independent framework that allows for the creation of hybrid recommenders that can be tailored to various domains?
2. What are the generic building blocks of recommender systems?

3. How can these generic building blocks be used to create hybrid recommender systems based on the chosen hybridization method?
4. Can hybrid recommender systems be developed using the framework and are recommendations made by those hybrid recommender systems accurate?

Recommendations as produced by recommender systems are not the only way to support people in finding interesting items. Such recommendations only focus on the user's long-term and short-term interests; also the immediate interests, the goals people have with the items, are important in deciding which items are interesting and which are not. For this reason, the second research objective is to determine how and if the goals of a user can be used in the recommendation process. In order to achieve this objective, the following research questions have to be answered:

Research objective 2

5. What are the goals that users have for which they want to use information?
6. How can the goals of a user be used within the recommendation process?
7. Does the introduction of goals in the recommendation process help people in finding interesting items?

As we will see in chapter 6, user goals are best used in the structuring process of an information system.

Whether an information system helps a person to find interesting items is also dependent on how the items found are presented to the user; i.e. the user interface of an information system. This leads to the third research objective: determine how the user interface elements that are specific to recommender systems should be designed. In order to achieve this objective, the following research questions have to be answered:

Research objective 3

8. What user interface elements are specific to recommender systems?
9. What are the guidelines for designing those user interfaces elements that are specific to recommender systems in order for a recommender system to be usable?

1.3 Approach

As this thesis focuses on three different aspects concerning recommender systems, three different research approaches are used; for each aspect an approach that best suits that specific aspect.

The approach for the first study is based on finding a generic model of recommendation techniques used in recommender systems by analyzing various recommendation techniques. This generic model is then used to

develop a framework that describes how to create hybrid recommender systems. The framework itself and several design choices within the framework are evaluated by developing recommender systems in two different domains; these recommender systems are then used in simulation experiments to show that the recommendations provided by these hybrid recommenders are more accurate than the recommendations made by the individual recommendation techniques.

The main research method for the study concerning the use of goals in recommender systems is the development and validation of a goal-based structuring method by combining two theories: the uses and gratification theory and the means-end approach. From this theory a set of hypotheses are derived that encompass our expectations that structuring items based on the user's goals makes it easier for people to find interesting items than when a structure is used that is not based on the user's goals. These hypotheses are tested in a between-subjects experiment. This study also investigates how easy it is for people to find interesting items when goal-based structuring is and is not combined with the use of recommendations.

The final study, which focuses on presentation aspects of recommender systems, employs an iterative user interface design method, using design techniques such as brainstorming, interactive design sessions, electronic surveys, heuristic evaluation and usability tests in order to derive guidelines for the design of the specific user interface elements of recommender systems. What these specific user interface elements are is derived from the first study concerning the framework for the development of hybrid recommender systems.

The order in which the three studies have been performed differs from the order in which they are presented in this thesis; structuring is presented before presentation, as that is the order in which these processes occur in an information system; however, structuring has been investigated after the user interface study as this reversal allowed us to use the results of the user interface study in the experiment concerning goal-based structuring; the goal-based structuring experiment required a working recommender system that had to be used by users for several weeks, which meant that a well designed and investigated user interface was necessary; the user interface study was independent of whether goal-based structuring was used or not as the focus in the user interface study was on those presentation aspects that are specific to recommender systems in general and not on the specific aspects of goal-based recommender systems.

1.4 Outline

The structure of this thesis is based on the three aspects of recommender systems investigated corresponding to the three major processes in an information system: selection (chapter 3, 4 and 5), structuring (chapter 6) and presentation (chapter 7).

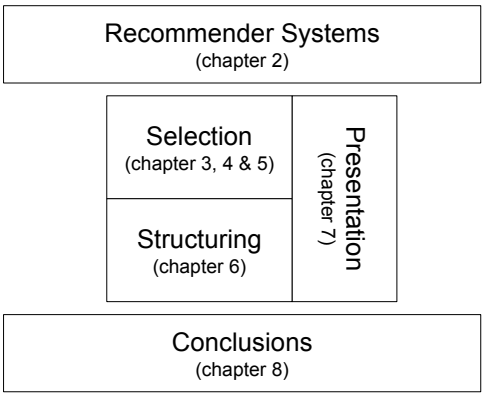


Figure 1-5 Outline of thesis based on three major processes

Chapter 2 provides a more detailed description of recommender systems, the techniques used within recommender systems to determine how interesting information is to a user, hybrid recommender systems and other research topics concerning recommender systems.

Chapter 3 introduces the Duine[1] prediction framework, which is our approach to create hybrid recommender systems in a domain-independent way, which can be tuned to specific application domains. Chapter 4 describes several design options with regard to decision approaches that can be used for recommender systems developed with the Duine prediction framework, while chapter 5 reports on experiments to validate that recommender systems created with the prediction framework are capable of providing accurate recommendations.

Chapter 6 describes how the goals of users can be used within recommender systems; it describes a way to structure items based on the goals people have with those items. The strength of this goal-based structuring method is investigated in a user study and is compared with the help people receive from traditional recommenders. This study shows that goal-based structuring helps people more with finding interesting items than recommendations do.

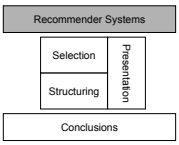
[1] Duine is the Irish Gaelic word for 'person' and has been the name of this research project. It is pronounced as /DIN-ah/.

Chapter 7 researches the presentation aspects of recommender systems, focused on presenting predictions and explanations and how to acquire explicit feedback from users. These aspects are investigated using an iterative user interface design method.

Chapter 8 concludes this thesis by combining and reflecting on the results of our framework to develop hybrid recommender systems, our goal-based structuring method and the results of the study concerning the user interface aspects of recommender systems.

Chapter 2

Recommender Systems



To help people deal with the information overload problem, this thesis addresses some technical solutions to help people to easily find items that are interesting to them. Our solutions cover the three main processes of a personalized information system, namely selecting, structuring and presenting items. Before discussing our specific solutions within these three processes in the next chapters, this chapter first focuses on one of the main type of systems that can be used to support people in finding interesting information, namely recommender systems. Various aspects of recommender systems are introduced (section 2.1), including several techniques to predict how interesting an item will be for a user (sections 2.2 and 2.3). This chapter concludes in section 2.4 with related research topics concerning recommender systems that are not specifically addressed in this thesis.

Resnick & Varian (1997) describe recommender systems as systems that acquire opinions about items from a community of users and that use those opinions to direct other users within that community to those items that are interesting for them. This description of a recommender system is directly related to the real-world concept of recommendations, where one person recommends something to another person. In the view of Resnick and Varian, a recommender system is only an intermediary in this process.

(Herlocker, 2000) describes a recommender system as a system that predicts what items a user will find interesting or useful. This definition of a recommender system is broader than the definition of Resnick and Varian; it does not imply that opinions of other people have to be used to recommend items; it also allows recommender systems to use other mechanisms to predict what users find interesting.

The view of recommender systems we take is based on the definition by Burke (2002, page 331):

Definition 2
Recommender System

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.

By this definition, a recommender system does not necessarily result in a list of recommendations, i.e. a list of only items that the user will find interesting, it can also provide other means of guiding a user to those items that are of interest to him; e.g. highlighting interesting items. The main concepts here are 'individualized' and 'personalized'; every user receives recommendations or is guided to interesting or useful objects in a way that best suits that user; based on his interests, his preferences and his characteristics.

This chapter describes recommender systems and provides the basis for discussing the domain-independent framework developed in this research to create hybrid recommender systems. Section 2.1 defines and discusses several key aspects of recommender systems, such as recommendations, predictions and ratings. Several techniques that can provide predictions for recommender systems are discussed in section 2.2, while section 2.3 describes the combination of multiple prediction techniques into hybrid recommender systems. In section 2.4, a discussion of related research topics concerning recommender systems that are not addressed in this thesis is provided.

2.1 Recommendations, predictions and ratings

Within the area of recommender systems, several concepts are used to denote different aspects of a recommender system: recommender, recommendation, prediction, certainty, user's interest, rating, predicted interest, predicted rating, actual interest, given rating, feedback, and prediction accuracy. This section defines these concepts and their relationships.

Definition 3
Recommender

An entity, person or software module, that produces recommendations as output or that has the effect of guiding users in a personalized way to interesting items.

A recommender is the active party in a recommender system that generates and provides the recommendations to a user; a recommender can be a piece of software (or software embedded in hardware) as well as a person; e.g. a TV recommender system can be an application that runs on a set top box connected to a TV that helps users to find interesting TV programs.

Recommendations can be delivered to people both in a pull-based as in a push-based manner. In pull-based recommenders, users explicitly make a

request for recommendations; in push-based recommenders, recommendations are sent to users without their specific request (although a prior subscription may be required). Schafer, Konstan & Riedl (2001) also distinguish a third type of delivery, namely passive or organic recommendations: the recommender provides recommendations within the natural context of the system; as part of the user experience; e.g. users can use an electronic TV guide normally by browsing through the guide; a passive recommender system would display predictions with each TV program where users are free to either base their decision on these predictions or to ignore them.

A special type of recommender system assists users in navigating through a complex item space: conversational recommender systems (Burke, Hammond & Young, 1997; McCarthy, Reilly, McGinty & Smyth, 2004). In conversational recommender systems, users interact more closely with a recommender by providing feedback – 'critiques' – on one aspect or a combination of aspects of a recommended item; e.g. a car recommender system suggested a Renault Megane to a user, however, the user informs the recommender that he wants a larger car; this feedback is the critique; the recommender then searches for similar cars to the suggested Renault Megane, but ones that are larger. As there is a lot of interaction between a conversational recommender and a user, conversational recommenders are less suitable for day-to-day decisions that users want to spend little time on; e.g. choosing what to watch on TV or which (newspaper) articles or e-mail to read. Conversional recommenders are best suited for recommending items that users rarely need to buy and where the user needs to choose one item, e.g. home appliances, cars, or houses. Another use of conversational recommenders is to help people find a specific item in an lesser known domain where the user can learn to understand the domain (Burke, 2000b) and choose between different aspects of items; e.g. searching for a restaurant in a town the user has never visited before or buying a digital camera for the first time. For such recommenders, it is difficult to learn the interests of people in such items; hence, the recommender and user need to work together to find the best matching item. Conversational recommender systems are not discussed any further in this thesis.

In the view of Resnick & Varian (1997) and Herlocker, Konstan, Terveen & Riedl (2004), the core task of a recommender is to "provide users with a ranked list of the recommended items, along with predictions for how much the user would like them".

Definition 4
Recommendation

An item or a list of items that is interesting to a user according to a recommender; the list only contains those items a recommender believes are interesting enough for the user.

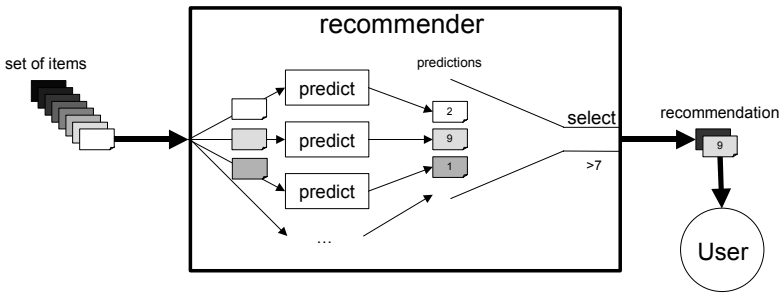


Figure 2-1
Recommender and predictions

Of all items that are fed into a recommender (as defined by Resnick & Varian and Herlocker et al.), a recommender returns only a subset of those items: those items it believes are interesting enough for its user. A recommender makes this decision (see *Figure 2-1*) by predicting how interesting each item is to the user. A recommendation then consists of the top predicted items (Herlocker, 2000). Jameson, Schäfer, Simons & Weis (1995, page 1886) say that "it is almost inevitable for an evaluation-oriented information provision to try to predict how the user would evaluate individual objects in the domain", were an evaluation-oriented information provision system is a systems where "the user has the goal of making evaluative judgements about one or more objects, where the system supplies the user with information to help him make these judgements"; recommender systems are evaluation-oriented information provision systems. Some recommenders allow a user to influence this selection process by, for example, allowing the user to set a threshold value that determines when items are interesting enough.

In the view of Burke (2002), recommender systems can also return all items with indications of how interesting each item is for the user instead of a subset; other mechanisms can then be applied to guide the user to the interesting items; e.g. using structuring or presentation techniques on the list of items (see chapters 6 and 7).

An example of a recommendation made by a TV recommender system is shown in *Figure 7-1*. This screenshot shows that the recommendation of this TV recommender systems consists for each channel of a set of TV programs that the recommender believes is interesting enough for its user and each program also has an indication (the number of stars) that gives the user an indication of how interesting each program is for him.

Figure 2-2 Recommendations made by a TV recommender system

As a "prediction involves the accurate anticipation of future (or as yet unobserved) events" (Neale & Liebert, 1986), a prediction in the domain of recommender systems is defined as:

Definition 5 Prediction

The anticipated interest of a user in one item.

Recommenders may or may not return predictions with recommendations; some recommenders simply return a list of items they recommend without giving any indication of the anticipated interest; other recommenders provide detailed information about the predictions.

Not every recommendation and prediction made by a recommender is made with the same amount of certainty or confidence (McNee, Lam, Guetzlaff, Konstan & Riedl, 2003).

Definition 6 Certainty

The degree of belief a recommender has in the accuracy of a prediction.

Certainty is based on the amount and quality of the available knowledge about the user and the item for which a prediction is made. When more and/or better quality knowledge is available to the recommender, the more confident a recommender can be in his predictions and recommendations, the more trust a user can place in the predictions and recommendations.

Besides the difference between recommendations and predictions, there also is a difference between interests and ratings and predicted interests and actual interests and predicted ratings and given ratings (see *Figure 2-3*). When discussing the differences between these concepts, notice that the discussion refers to one item: e.g. the predicted user's interest in one item, a rating for one item or the actual interest of the user in one item.

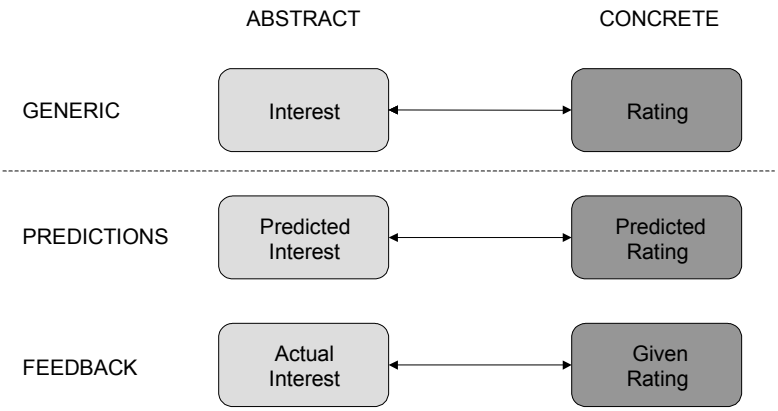


Figure 2-3 Abstract interests versus concrete ratings

Definition 7 User's interest

An abstract indication of how much a user appreciates an item.

A user's interest is an abstract concept as it is difficult for people to clearly and completely describe how much one likes or dislikes an item; especially a single item unrelated to other items; e.g. how much does one like an apple? People are better capable of providing an order between multiple items; e.g. one can like apples and oranges, but oranges more than apples. However, the larger the set of items the more difficult it becomes to specify this order. For this reason, an often-used way to concretize a user's interest is by using ratings:

Definition 8 Rating

A concrete value representing a user's interest, i.e. a concrete value that gives an indication of how much a user appreciates an item.

This concrete value is measured on a certain scale, e.g. 1 to 5 or -10 to 10, which can be presented to a user, e.g. using a number of icons such as the number of stars by the TV recommender system in *Figure 7-1*. An item can be given one rating for the whole item or ratings for various aspects of the item. The actual scale to use is determined by the designers of a recommender system (see also chapter 7.4).

Definition 9 Predicted interest

An anticipated abstract indication of how much a user appreciates an item.

A recommender tries to determine how much a user will appreciate each item and decides which item(s) it will recommend based on those predicted interests. However, as a predicted interest is only an abstract concept, a recommender needs a concrete representation of this abstract concept: a predicted rating.

Definition 10 Predicted rating

A concrete value representing the predicted user's interest.

Such a concrete value gives an indication of the anticipated appreciation that a user will have for an item, e.g. the system predicts a rating of 3 stars on a 5 star scale. A recommender can use this concrete value to compare multiple items and to recommend only those items that have a high enough predicted rating. The TV recommender in *Figure 7-1* only selected those programs that have at least 3 stars and presented the predicted rating using the stars to give the user an indication of what the recommender predicted as how much the user would appreciate each program.

Definition 11 Actual interest

The real appreciation that a user has for an item.

Definition 12 Given rating

A concrete value provided by a user that represents how much that user really appreciates an item.

Where a predicted interest is about the anticipated appreciation of a user for an item, the actual interest is the real appreciation of a user for an item. The given rating concretizes this real but abstract appreciation. This is the rating that a user gives to the recommender system, e.g. 4 stars on a 5 star scale or like versus dislike. Some recommender systems allow users to only give one rating per item; others allow users to rate various aspects of an item.

Definition 13 Feedback

The user's response to the recommendations and predictions made by the recommender.

Recommenders can learn from feedback provided by users in order to optimize their future recommendations and predictions; improved recommendations and predictions are a motivation for users to provide feedback. Giving items a rating is the most common type of feedback in recommender systems; more about feedback can be found in section 3.1.

Feedback is also used to measure the performance of a recommender; i.e. the prediction accuracy.

Definition 14 Prediction accuracy

The extent to which the predicted interest agrees with the actual interest of the user; i.e. the extent to which the predicted rating agrees with the given rating.

Prediction accuracy "measures how close the recommender system's predicted ratings are to the true user ratings" (Herlocker et al., 2004). The more accurate the predictions of a recommender are, the better the recommender supports users in finding interesting items. There are several

ways to quantize prediction accuracy using prediction accuracy measures; one of the most often used measures is discussed in section 5.1; other prediction accuracy measures can be found in (Herlocker et al., 2004).

2.2 Prediction techniques

In order for a recommender to determine a predicted rating for an item, recommenders require one or more algorithms that reason about the current user and the item; algorithms that calculate the predicted rating: so-called prediction techniques. The current user is that user for which a recommendation is being made; the other users are all users of a recommender system excluding the current user.

Definition 15 Prediction Technique

An algorithm that predicts how interested a user will be in an item by calculating a predicted rating.

Two major groups of prediction techniques can be identified:

1. *Social-based prediction techniques* analyze the behaviour and characteristics of users without using knowledge about the items; they use the known behaviour and characteristics of the current user and other users to deduce the predicted interest of the item for the current user.
2. *Information-based prediction techniques* analyze the item and other items and the knowledge about the current user to deduce the predicted interest of the item for the current user.

Social-based prediction techniques only require a unique value to identify each of the items for which predictions have to be made; no further knowledge about the items is required. This makes social-based prediction techniques domain-independent. Social-based prediction techniques are capable of providing diversity, also referred to the serendipity effect of social-based prediction techniques; they are capable of recommending items dissimilar to items a user has seen in the past (Smyth & Cotter, 2000), they can provide "cross-genre or 'outside the box' recommendation[s]" (Burke, 2002) as the predictions are not only based on past behaviour of the current user, but also on behaviour of other users. However, social-based techniques suffer from bootstrapping and sparsity problems. Bootstrapping is the problem that there is almost nothing known about a new user or a new item making it impossible to generate a prediction, while sparsity refers to the fact that not every user can give an opinion about every item, making it difficult to find relationships between users, between items and between users and items, while these relationship are necessary for generating predictions.

Information-based prediction techniques on the other hand are domain-dependent as they require content and/or metadata to analyze in order to generate predicted ratings; they are "at the mercy of the descriptive data available" (Burke, 2002). Because information-based prediction techniques are mainly based on item similarity, recommended items tend to be similar to items the user has seen in the past, leading to reduced diversity (Smyth & Cotter, 2000). These techniques are also called cognitive prediction techniques (Malone, Grant, Turbak, Brobst & Cohen, 1987).

Malone et al. also identify a third group of prediction techniques (although they examined information filtering in general), namely economic-based techniques. Such techniques are based on "costs-benefit assessments and explicit and implicit pricing mechanisms" (Malone et al., 1987, page 392). These techniques do not address the interests of a user with respect to the content of an item. They can be used by people as an additional piece of information besides the predicted interests when deciding which items to use or buy (although some people may not make the distinction between interests and economical issues explicit). Since the focus of the Duine prediction framework is on predicting the interest of a user in an item, economic-based techniques are not discussed further, but are of interest for future research.

Furthermore, Burke (2002) also identified knowledge-based and demographic-based recommendations. Demographic-based recommendations try to categorize a user into a certain demographic class according to some personal attributes (e.g. age, gender, occupation) and try to provide recommendations based on knowledge about the demographic classes. Knowledge-based recommenders use prior knowledge about the relationship between the user's needs and possible items to recommend. However, in our dichotomy of prediction techniques, demographic-based techniques are a type of social-based prediction techniques as they only employ information about the user and other users; in this case demographic classes of users. Knowledge-based techniques use knowledge about how certain items can fulfil the needs of users, making it a subclass of information-based prediction techniques.

In the next two sections, several types of social-based and information-based prediction techniques are described. For most prediction techniques, various algorithms exist. Since the specific algorithms are not important when describing the prediction framework (see the next chapter), they are not provided. When introducing experiments with the prediction framework (see chapter 5), the specific algorithms used will be described.

2.2.1 Social-based prediction techniques

Collaborative filtering

The basic idea behind collaborative filtering (also called social filtering) is that people who have rated the same items the same way in the past probably have the same taste. Based on this knowledge one can predict how much a person likes an unseen item when similar users have already rated that item (Resnick, Iacovou, Suchak, Bergstrom & Riedl, 1994; Shardanand & Maes, 1995; Breese, Heckerman & Kadie, 1998; Aggarwal, Wolf, Wu & Yu, 1999; Sarwar et al., 1998; Herlocker, 2000; Sarwar, Karypis, Konstan & Riedl, 2000). Collaborative filtering is one of the most researched prediction techniques for recommender systems.

Collaborative filtering basically consists of three steps (Herlocker, 2000). In the first step, the similarity between the current user and those other users who have rated the item for which a prediction is necessary is calculated based on how the current user and each of the other users have rated the same items in the past. The second step is to select a subset from all other users that have rated the item by selecting only the most similar users. The final step is to use the similarities and the ratings for the item of the selected similar users to calculate the predicted rating. Herlocker, Konstan & Riedl (2002) provides an overview of design choices and alternative algorithms in collaborative filtering.

Collaborative filtering is an accurate domain-independent prediction technique, especially for content that cannot easily and adequately be described by metadata (Melville, Mooney & Nagarajan, 2002), such as movies, music and entertainment programs on TV. Objective metadata can be associated with such items, such as title and author, however that metadata is not able to capture emotional and esthetical appeal for such types of items, which is indirectly captured by the opinions of users with similar tastes.

Item-item filtering

Where in collaborative filtering the idea is that people who have rated the same items the same way in the past probably have the same taste, the idea with item-item filtering is that items that have been rated the same way in the past are probably similar (Herlocker & Konstan, 2001; Linden, Smith & York, 2003). For example, Lord of the Rings – Fellowship of the Ring and Lord of the Rings – The Two Towers are similar as people tend to rate both movies in the same way: if you give the first a high rating, you probably also give the second a high rating, while if you rate the first low, you probably also rate the second low. Item-item filtering can be used for making predictions about an item the same way as case-based reasoning does (see

section 2.2.2), but with item-item filtering the similarity between items is calculated via the ratings given to those items and not based on features of the item.

As item-item filtering uses the same rating data as collaborative filtering, item-item filtering is also domain-independent and based on behavioural data of users.

Stereotypes and demographics

The use of stereotypes in user modelling has first been introduced in 1979 by Rich (1998). When predicting the behaviour of someone else, people often use stereotypes. Stereotypes contain a set of characteristics that describe a stereotypical user and a collection of aspects (e.g. behaviour, interests, actions) that such a stereotypical user generally exhibits. In recommender systems, the collection of aspects contains the interests of people in items. The characteristics are often based on demographic data, such as age, gender, occupation, and education. Ardissono et al. (2004) use stereotypes of TV viewers to determine the interests of users in genres of TV programs. Stereotypes "are very useful for application areas in which, quick but not necessarily completely accurate, assessments of the user's background knowledge are required" (Goren-Bar & Glinansky, 2004, page 150).

Stereotypes can be used in various ways in recommender systems, such as to bootstrap user profiles for new users (e.g. filling a user profile with a predefined set of keywords for information filtering) or to have only a limited number of user profiles based on stereotypes instead of modelling each user individually. Stereotypes can also be used in combination with collaborative filtering; instead of determining the similarity between users based on their ratings, similarity of users can be determined by the similarity of demographic or other stereotypical characteristics.

Popularity

Popularity prediction techniques use ratings of all users to predict how interesting an item is for one user. The more users who liked the item, the higher the predicted rating for that item will be. The most basic popularity prediction technique is the average rating of an item over all users. Herlocker (2000) describes a popularity-based prediction technique that takes into account that various people have different overall interests in a certain item; it calculates a deviation-from-mean average.

Average

A very basic prediction technique is to average all ratings the current user has given in the past. This average represents the overall interest of the user in items from the given recommender, e.g. the overall interest in TV

programs in a TV recommender system, the overall interest of movies in a movie recommender system, or the overall interest in books in a book recommender system.

2.2.2 Information-based prediction techniques

Information filtering

Information filtering is the process in which a system filters a vast amount of information and only delivers or recommends information to the user that is relevant or interesting to the user. As recommender systems are part of information filtering and retrieval (see chapter 1), information filtering techniques can also be used within recommender systems.

Information filtering originated in the domain of text retrieval. One of the earliest forms of electronic information filtering came from the work on Selective Dissemination of Information (SDI) by Houseman & Kaskela (1970). SDI was used in a system to keep scientists informed of new documents published in their expertise area. The most widely used information filtering technique is based on the term-frequency-inverse-document frequency (tf-idf) algorithm.

Pazzani & Billsus (1997) describe an information filtering approach for a recommender system that uses a Bayesian classifier (Duda & Hart, 1973) to determine the probability that a document, represented by a set of the k most informative words, is either interesting or not interesting for a user. The calculated probabilities can be used to rank and/or order the pages (Pazzani & Billsus, 1997) or can be used to create a predicted rating. They also experimented with several other information filtering algorithms: PEBLS, decision trees, Rocchio's algorithm using tf-idf and neural networks.

Case-based reasoning

Case-based reasoning (CBR) is based on the assumption that if a user likes a certain item he will probably also like similar items (Riesbeck & Schank, 1989). CBR as a prediction technique looks at all items a user has rated in the past and determines how similar they are to the current item. For those items that are similar enough, the old ratings are used to calculate a predicted rating for the new item.

Case-based reasoning is especially good in predicting how interested a user is in the same types of information or slightly different version of the same information. The key aspect of case-based reasoning is determining the similarity between two items. "What counts as similar depends on one's goals [...] typically, there are only a handful of standard goals in any given [...] domain" (Burke, 1999); however, these goals are domain-dependent

making the way to calculate similarity between items for those goals also domain-dependent.

The item-item filtering technique discussed in the previous section resembles case-based reasoning, except that with item-item filtering similarity between items is calculated using the ratings given by all users to items and is thus based on the behaviour of users not on the item itself.

Attribute-based prediction techniques

Jameson et al. (1995) use Multi-Attribute Utility Theory (MAUT) to determine how interesting an item is to a user. For each attribute of an item, the prediction technique has a value function that assigns a function value to the attribute based on the value of that attribute. Each attribute also has an importance weight that can vary per user. Based on the importance weights and function values, predictions can be generated.

Items are often grouped into one or more categories; e.g. genres of movies, TV programs, books or the product categories in shops. Category-based predictors, which are a subclass of attribute-based prediction techniques, use these categories to predict how interesting an item is for a user; e.g. genreLMS (van Setten, 2002) and the category approach described in Goren-Bar & Glinansky (2004).

2.3 Hybrid prediction techniques

Because every prediction technique has its own strengths and weaknesses, there is a need to combine different prediction techniques to increase the accuracy of recommender systems (Burke, 2002). The idea behind hybrid prediction techniques is that a combination of algorithms can provide more accurate recommendations than a single algorithm as disadvantages of one algorithm can be overcome by other algorithms. Malone et al. (1987) already mentioned the notion that a combination of approaches will most likely result in the most useful systems.

Burke defines a taxonomy of methods to combine recommendation techniques, i.e. combining techniques that result in a list of items that are interesting to the user according to the recommender. These methods are called hybridization methods. The hybridization methods described by Burke, not only apply to recommendation techniques; they can also be applied to prediction techniques.

2.3.1 Hybridization methods

The hybridization methods are: weighted, switching, mixed, feature combination, cascade, feature augmentation and meta-level.

Weighted
The predictions of several prediction techniques are combined by weighting the predicted ratings of the techniques to produce a single prediction. In its simplest form, the predicted ratings calculated for an item by the prediction techniques are averaged. More advanced forms may employ Bayesian networks, neural networks or other (non-)linear functions. Examples of systems that use weighted hybridization are the news recommender system P-Tango (Claypool et al., 1999), a restaurant recommender system (Pazzani, 1999), a TV recommender system (Buczak, Zimmerman & Kurapati, 2002) and another TV recommender system (Ardissono et al., 2004). In section 3.4.1, argumentation is provided why the weighted hybridization method is not always capable of providing more accurate predictions.

Switching
Depending on some criterion, the hybrid technique switches between the available prediction techniques. If one prediction technique is not capable of providing good predictions another prediction technique is used. Examples of switching recommender systems are the Daily Learner (Billsus & Pazzani, 2000), a product recommender system (Tran & Cohen, 2002) and the prediction framework described in the next chapter.

Mixed
The results of several prediction techniques are presented at the same time; instead of having just one prediction per item, each item has multiple predictions associated with it from various prediction techniques. Examples of recommender systems that employ the mixed hybridization method are PTV (Smyth & Cotter, 2000), ProfBuilder (Wasfi, 1999) and PickAFlick (Burke et al., 1997).

Feature combination
Features generated and normally used by a specific prediction technique are used in other prediction techniques. For example, the 'ratings of similar users' that is a feature of collaborative filtering is used in a case-based reasoning based prediction technique as one of the features to determine the similarity between items. (Basu, Hirsh & Cohen, 1998) describe a recommender system called Ripper that uses feature combination.

Cascade
The predictions or recommendations of one technique are refined by another prediction or recommendation technique; the first prediction technique outputs a coarse list of predictions, which is refined by the next

prediction technique. An example of a recommender system that use the cascade hybridization method is EntreeC (Burke, 2002).

Feature augmentation

Output from one prediction technique is used as an input feature for another technique. An example are the filterbots described in Sarwar et al. (1998). Filterbots are automated prediction techniques that generate ratings for each item based on some pre-defined criteria. These generated ratings are then used by a collaborative filter in the same way as ratings from real users, i.e. the filterbots are treated as normal users.

Meta-level

The internal model learned by one prediction technique is used as input for another. The difference with feature augmentation is that with feature augmentation the first technique outputs some features based on its internally learned model, these features are then used as input for the second technique; with meta-level hybridization, the entire model that is learned by the first technique is used as the input for the second technique, e.g. all weights learned by GenreLMS are used by an information filtering prediction technique where the genres are treated as keywords with the weights learned by GenreLMS. Examples of meta-level recommender systems are Fab (Balabanovic, 1997) and the system described by Condliff, Lewis, Madigan & Posse (1999).

2.3.2 Coupling

These various hybridization methods provide different levels of coupling between the prediction techniques, which is the amount of knowledge that is needed by a hybrid recommender about the internal workings of the used prediction techniques in order to combine those techniques.

Two extreme levels of coupling are:

- *Tight coupling*: two or more prediction techniques (A, B) are combined into a new prediction technique (C), where the new prediction technique has and uses knowledge about the internal workings of the combined techniques: $C = f(A, B)$ (see *Figure 2-4*).
- *Loose coupling*: when two or more prediction techniques (A, B) are loosely coupled into a new prediction technique (C), the new prediction technique only combines the results of the coupled techniques and has no knowledge about the internal workings of the combined techniques: $C = g(f(A), f(B))$ (see *Figure 2-5*).

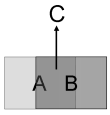


Figure 2-4 Tight coupling of prediction techniques

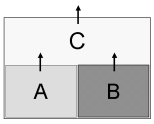


Figure 2-5 Loose coupling of prediction techniques

Coupling is not a dichotomy; tight coupling and loose coupling are just extreme levels of coupling; prediction techniques can be more or less coupled in a hybrid recommender (see *Figure 2-6*). Weighted hybridization

is very loosely coupled; there is no knowledge necessary about the behaviour of the prediction techniques as only the predicted ratings of the techniques are combined using weights. The only knowledge necessary is the scale on which each prediction technique outputs its predictions.

Switching is not strictly tightly coupled, although more than the weighted method; for switching, the hybrid technique needs to have knowledge about 'when each prediction technique is capable of providing good predictions'; this requires some level of knowledge about the behaviour of the prediction technique within the application domain.

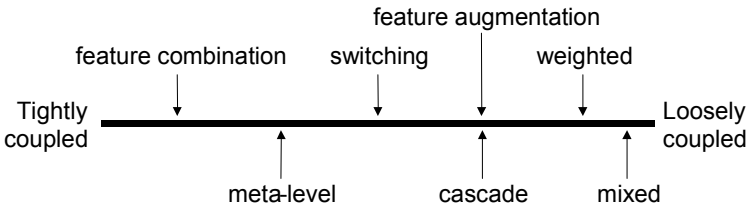


Figure 2-6 Level of coupling of hybridization methods

The mixed hybridization method requires no knowledge about the behaviour of the prediction techniques as the individual predictions of the techniques are all presented together.

The feature combination method on the other hand requires a lot of knowledge about the prediction techniques, the available features of the techniques and the meaning of each feature, as it needs to be able to combine and reason with those features.

The level of coupling for cascade is limited; prediction techniques later in the process only have to be able to understand the output of previous prediction techniques. Since the output of the previous prediction techniques is the same as of the later prediction techniques – a prediction – the level of knowledge required is limited.

Just like cascade, feature augmentation only needs to be able to understand the output of other prediction techniques; which in most cases are predictions.

Meta-level hybridization requires a reasonable amount of coupling as the entire model learned by one prediction technique is used as the input for another prediction technique; the technique that receives the model as an input, needs to be able to understand the meaning of the model and be able to reason with that model.

The Duine framework employs the switching hybridization method as it provides a balance between loose and strictly coupling, making it ideal for a domain-independent framework that can be implemented in various domains and which can be tuned to the domain in which it is implemented. The way in which switching hybridization is implemented in the framework is described in the next chapter.

2.4 Related topics concerning recommender systems

In this section, related topics in the research area of recommender systems are discussed. Although all the related issues are important for recommender systems, this thesis does not provide solutions for them. Research into these topics is either left to other research projects or left open for future research. However, clear relationships and implications of research in this thesis and these topics will be addressed in this thesis.

2.4.1 Cold-start problem

One of the problems of most recommender systems is the cold-start or bootstrapping problem (Maltz & Ehrlich, 1995); this refers to the situation where a recommender does not have enough information about a user or an item to make good recommendations for it. There are three types of cold-start problems: new user, new item and new system.

The new user cold-start problem occurs when a new user starts to use a recommender system; in the beginning, there is little known about this new user, making it difficult for many prediction techniques to generate accurate predictions and recommendations.

The new item cold-start problem occurs when a new item is added to a recommender system; some prediction techniques, such as collaborative filtering, are not capable of generating predictions for such items as there are no ratings given to them by users.

The new system cold-start problem is a magnified combination of both the new user and new item cold-start problem as in new recommender systems there is nothing known about the (new) users and there are no ratings for any of the items.

Hybrid recommender systems, as discussed in the previous section, are one solution to the cold-start problem as these systems incorporate multiple prediction techniques, decreasing the probability that none of the prediction techniques is capable of providing an accurate prediction. However, cold-start issues can still occur. Middleton, Shadbolt & De Roure (2004) describe a solution in which they use ontological user profiles to bootstrap users profiles which are created by using web proxies to unobtrusively monitor a user's web browsing behaviour.

Another possibility to solve the cold-start problem is the use of stereotypical user profiles (Ardissono et al., 2004) that are inferred from prior information about users; e.g. categorization of users and their interests based on surveys.

Baudisch & Brueckner (2002) address the cold-start problem in a TV recommender system by allowing users to use an online TV guide without having to provide explicit interests or to give ratings, but by allowing users

to browse through and search within the TV guide and to build a user profile unobtrusively in the background by monitoring this usage; this implicitly acquired feedback is used at a later stage to generate recommendations.

In this thesis, the cold-start problem is not addressed directly; the Duine prediction framework described in the next chapter, which can be used to create a hybrid recommender system, allows for the incorporation of prediction techniques that can predict the interests of users for an item with limited knowledge about either the user or an item. Such techniques are included when validating the framework in chapter 5.

2.4.2 Evaluation

As recommender systems provide subjective results, an important issue concerning recommender systems is how to evaluate recommender systems; how can the effectiveness of recommender systems best be evaluated? How to measure the prediction accuracy of recommender systems? Where non-personalized information systems can measure prediction accuracy and performance objectively, personalized systems need measures that take into account the subjective nature of their results. Herlocker et al. (2004) provide an overview of the currently available methods to evaluate recommender systems taking into account different ways that users can interact with and receive recommendations. In chapter 5, one of these evaluation methods is applied to evaluate the prediction framework discussed in the next chapter.

2.4.3 Group recommendations

Most recommender systems have been primarily focused on providing recommendations to individual users. However, in some domains it is just as important to be able to recommend items to a group of people; e.g. recommending what movie to watch tonight with a group of friends; what restaurant to eat in with a group of colleagues; what TV programs to watch tonight with the family.

One of the first examples of a group recommender system was the MusicFX system which "adjusts the selection of music playing in a fitness centre to best accommodate the music preferences of the people working out at any given time" (McCarthy & Anagnost, 1998, page 363); however, MusicFX is not a real recommender in the sense that it does not provide individualized recommendations nor does it guide users to interesting items; MusicFX automatically selects a radio station based on the predictions concerning genres using the explicitly provided interests of the fitness centre members for these genres.

O'Connor, Cosley, Konstan & Riedl (2001) discuss five design issues concerning group recommender systems: what is the nature of a group (ephemeral or persistent, public or private)? How do groups form and evolve (who creates, who can join and how are they managed)? How is privacy handled within a group (concerning joining and during membership)? How to form recommendations for groups (the social value function and the algorithmic implementation)? What interfaces support group recommendations (group-only, composite or individually focused interfaces)?

O'Connor et al. (2001) basically distinguish two approaches to forming group recommendations: merging the individual user profiles into one group profile (or manually creating a group profile) and use this group profile in the same way as if it were a single user to get recommendations or to get recommendations for each group member and merge the resulting recommendations. Goren-Bar & Glinansky (2004) describe a TV recommender that linearly combines the user profiles of individual TV viewers into one group profile and uses this group profile to recommend TV programs. Masthoff (2004) investigated various strategies for selecting (a sequence of) TV programs for a group based on the predicted interests of each individual user. Ardissono, Goy, Petrone, Segnan & Torasso (2003) create recommendations for tourist attractions for group of tourists by first generating rakings for each individual subgroup (their level of focus is not on individuals, but on subgroups within groups) and then combining the rakings of the subgroups into a raking for the group as a whole.

In this thesis, the focus lies on recommendations for individual users. Masthoff (2004) indicates that before group recommendations can be made, first the interests of the individual users need to be determined; good individual recommendations lead to better group recommendations; both research topics are important and strengthen each other.

2.4.4 Privacy, security and trust

Recommender systems use personal information of users, such as user characteristics, their interests and opinions in the form of ratings for items, to provide services that better meet the needs of each individual user. “The other edge of the sword is that recommender systems provide perfect tools for marketers and others to invade user’s privacy” (Riedl, 2001, page 29). Hence, according to Riedl (2001) privacy is an important issue to take into account, not only from a moral viewpoint, but also from a legal viewpoint; users must know or be able to trust that their privacy is guaranteed by a recommender system. Schreck (2003) also notices that people may have personal demands concerning their privacy.

Security is an important factor to ensure that personal data remains personal and is only used by those who have a right to access and use that data. Schreck (2003) discusses several security methods to allow users to remain anonymous and to assure that personal data is secure during transport. Even though a recommender system can have the best security methods in place, it is still the user who must decide whether to trust the recommender system and the people behind the system; trust that security methods are indeed implemented and that these methods are capable of keeping their personal data safe.

This type of trust is called system / impersonal trust (Abdul-Rahman & Hailes, 1997). Another type of trust is the context-specific interpersonal trust, where the user has to trust someone else regarding one specific situation; e.g. does a user trust a review written by another user for an item. O'Donovan & Smyth (2005) have used the concepts of context-specific interpersonal trust and system / impersonal trust to design a collaborative filtering algorithm that takes into account trust.

Another way to increase the trust a user has in recommendations is by providing good explanations concerning the predictions. Zimmerman & Kurapati (2002) use explanations based on items or aspects of items that the user already knows to increase the trust users have in the recommendations of a TV recommender. Sinha & Swearingen (2002) examined how the interaction design of recommenders systems, including how different recommender system elements, such as transparency of recommendations and familiarity with recommendations compared to the recommendations of new unknown items, influenced the trust people had in the recommender system; they found that transparency and the inclusion of known items in the recommendations increased the trust of users in recommender systems.

In order to increase trust in systems that use personal information, an industry standard called P3P has been developed that provides an automated way for users to gain more control over the use of personal information on Web sites they visit (see http://www.w3.org/P3P). The idea behind P3P is that each website or service describes how it deals with personal information in a standardized way, which is made public in a privacy profile and published together with the website or service. Web browsers or other P3P compliant client applications can read this privacy profile and compare it with the privacy preferences of the user; based on this comparison the browser or client application then informs the user of any possible privacy issues and asks the user how he wishes to proceed.

Another security issue for recommender systems is the possibility of attackers that try to bias the recommendations made by a recommender system in order steer the system to provide recommendations that are favourable for them; e.g. to try to get their products recommended more

often than their competitor's products. Burke, Mobasher, Zabicki & Bhaumik (2005) describe seven different types of attack that may compromise a recommender system.

This thesis does not address any privacy, security or trust issues, although the prediction framework takes into account explanations regarding predictions and chapter 7 investigates the presentation of explanations in the user interface of a recommender system.

2.4.5 Cross-domain recommendations

Most recommender systems so-far only focused on recommending items within one domain; only movies, only TV programs; only CD's etc. Some exceptions are webstores like Amazon.com that recommend various product types; such recommenders mainly use social-based prediction techniques, which are more easy to use across multiple domains than information-based prediction techniques. Since social-based prediction techniques are not dependent on metadata of items (except for a unique identifier), they can easily be applied to a heterogeneous set of items. Information-based prediction techniques have the disadvantage of needing to encode the interests of user's for which they use domain-specific knowledge.

One group of recommender systems that may be able to cross domains are meta-recommenders. "These systems present recommendations fused from 'recommendation data' from multiple information sources" (Schafer, Konstan & Riedl, 2002, page 43). Although the concept of meta-recommenders seems to be useful for cross-domain recommendations, the meta-recommender investigated by Shafer et al. only provide recommendations within one domain using multiple recommenders and information sources to provide 'more pieces of the puzzle'; a movie recommender system that uses a recommender based on collaborative filtering, a recommender that recommends local cinema's and a recommender that uses movie critics reviews for movies.

One of the key research issues concerning cross-domain recommendations is the semantic alignment of concepts from various domains; how to relate something learned about a user in one domain to concepts in another domain. This issue is being researched by the semantic web community (Berners-Lee et al., 2001).

2.4.6 User interfaces

An aspect that is not only important for recommender systems, but for any system is the interface between user and system. This thesis addresses some user interface issues that are specific to recommenders systems, namely the presentation of predictions, the presentation of explanations and the way

users can give feedback in the form of ratings. However, there are also other user interface issues that are specific to recommender systems.

One such issue is how to elicit ratings and interests from new users (McNee, Lam, Konstan & Riedl, 2003). As recommender systems perform best when they have a certain minimum amount of ratings of a user, it is important to quickly get enough information about a new user, but on the other hand, it is also important to make it easy for the user to start using the recommender system.

Another issue in recommender systems is the confidence (also called certainty) a recommender has in its predictions and recommendations. Although the confidence of a recommender system in predictions is addressed in the next chapter, the presentation of confidence values is not directly addressed in this thesis. However, presenting explanations about how predictors arrived to their predictions, which has a similar function as confidence values, is addressed in chapter 7. McNee et al. (2003) investigated the issue of presenting confidence values.

Where most research into recommender systems' user interfaces focuses on graphical user interfaces, Wärnestål (2005) investigates the use of natural language dialogues to recommend items.

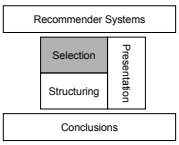
2.5 Conclusions

In this chapter, recommender systems have been explored by defining various aspects of recommender systems, such as the difference between recommendations, predictions and ratings. Also various techniques to predict how interested people are in items have been discussed, with a focus on social-based and information-based techniques. Furthermore, an overview of different hybridization methods to combine multiple prediction techniques has been given. Finally, an overview of related research topics concerning recommender systems has been provided.

Our first research objective is to develop a domain-independent framework that describes how to create hybrid recommender systems that can be tailored to various domains in order to provide more accurate recommendations. As the switching hybridization method is the most suited for a domain-independent framework, the next chapter introduces our domain-independent framework that is based on switching hybridization: the Duine prediction framework. Chapter 4 examines an important design choice when developing recommender systems based the framework, while chapter 5 discusses the results of experiments with the Duine prediction framework. Chapters 6 and 7 address the other two research objectives that are concerned with the use of goals in recommender systems and the user interface elements specific to recommender systems.

Chapter 3

Prediction Framework



One of the solutions to support people in their search for interesting items is to use recommender systems; systems that help in the selection process and that predict how interested a person will be in a certain item and that uses these predictions to recommend the most interesting items or guide people to the interesting items. The previous chapter introduced recommender systems and discussed several important aspects of recommender systems. It also discussed several techniques that try to determine the anticipated interest of a user in an item, i.e. prediction techniques: some by reasoning about the information, some by using social inferences. The concept of using multiple prediction techniques together to determine how interested an item is for a user has also been discussed, so-called hybrid recommender systems. The main advantage of hybrid recommender systems is that the weakness of one prediction technique can be overcome by other prediction techniques resulting in more accurate predictions.

As our first research objective is to develop a domain-independent framework that describes how to create hybrid recommender systems that can be tailored to various domains in order to provide more accurate recommendations, this chapter will describe that framework, called the Duine prediction framework. The Duine prediction framework describes a way to create prediction engines that is based on a switching hybridization approach called predictions strategies; prediction strategies select between various prediction techniques in order to improve the accuracy of predictions. Switching hybridization, as discussed in the previous chapter, has been chosen as the main hybridization method as it provides the best balance between loose coupling and tight coupling making it ideal for a domain-independent framework of which the resulting recommender systems have to be tailored to various domains.

Section 3.1 introduces the generic concept of a predictor that is used as a basis for the introduction of our view on combining multiple prediction

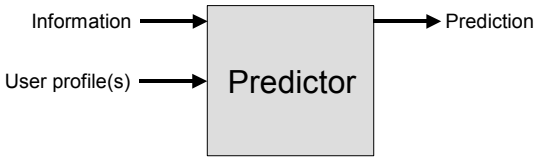
techniques: prediction strategies (section 3.2) and prediction engines (section 3.3). The process to generate predictions using the introduced framework is described in section 3.4. Before concluding this chapter in section 3.6, two possible extensions to the framework are described in section 3.5, showing the openness of the framework.

Parts of this chapter have already been published in van Setten, Veenstra, Nijholt & van Dijk (2003) and van Setten, Veenstra & Nijholt (2002).

3.1 Generic predictor model

Even though there are various types of prediction techniques, it is possible to create a generic model due to the basic nature of each prediction technique: each technique can calculate a predicted rating, simply called a *prediction,* of an item for a given user, based on knowledge stored in the *user profile*, *information* about the item and *profiles of other users*. This creates the basis of the generic predictor model (see *Figure 3-1*).

Figure 3-1 Basic predictor model



In this model, a prediction technique is generalized to a predictor, as later on another type of predictor will be introduced (see also section 3.2):

Definition 16 Predictor

An entity that predicts how interested a user will be in an item.

Information

Predictions can be generated for various types of items, e.g. a product, a book, a person, scientific paper, a movie, a TV program, a news article, a web service. As information contains the metadata of an item and/or the content of an electronic item, it is the information that is used by a predictor, especially information-based predictors, to make a prediction of how interesting the item will be to the user.

Some predictors need access to parts of the information, e.g. category-based predictors need access to the categories an item belongs to; other predictors need access to the full content, e.g. information filtering; and some predictors only need access to an attribute which value uniquely identifies the item, e.g. collaborative filtering.

User profile

A user profile is the representation of the user in the system; it describes the user of a recommender. A predictor can use both the profile of the user for whom a prediction must be made and the profiles of other users when making a prediction for the current user. A prediction technique such as genreLMS only needs the profile of the current user, while a technique such as collaborative filtering requires access to the profiles of all other users.

Prediction

Predictions are the output of a predictor and reflect the anticipated interest of a user in one item. Predictions consist of a predicted rating and a certainty value that represents the amount of certainty the predictor has in the prediction.

In order to compare and combine predictions from different predictors, it is necessary for each predictor to normalize its predictions and certainty to the same scale. For the predicted ratings, often-used normalization ranges are the bipolar range of [-1,1] and the unipolar range of [0,1]. The bipolar range goes from −1 representing absolutely not interesting to +1 representing absolutely interesting with 0 representing a neutral or indifferent interest. The unipolar range goes from 0 representing absolutely not interesting to +1 representing absolutely interesting with 0.5 representing a neutral or indifferent interest.

Certainty is expressed on the unipolar range of [0,1] but often presented to the user as a percentage ranging from 0% representing complete uncertainty to 100% representing complete certainty. Normalized predictions can be presented on different scales to users depending on the system and domain (see section 7.2.2).

Feedback

Most predictors are capable of learning from users in order to optimize future predictions. They learn from *feedback* provided by users.

There are two ways to acquire feedback from users: by analyzing the usage behaviour, which is called implicit feedback (Lieberman, 1995), and by using explicit relevance feedback (Rocchio, 1965; O'Riordan & Sorensen, 1995). With implicit feedback, the recommender system gathers information about a user's actions while using the system. These can range from global actions, such as the amount of time spent reading or using an item, to detailed actions such as each button click. Such actions are used to infer how interested the user is in the item. Implicit feedback lessens the burden on the user by inferring the user's interests from the user's behaviour instead of having the user provide the feedback (Goren-Bar & Glinansky, 2004); e.g. when users record a TV program using TiVo, this action is treated as implicit feedback that the user likes that TV program

(Ali & van Stam, 2004). With explicit feedback, in contrast, a user explicitly evaluates the relevance of the item by providing a rating.

As predictors need to learn from the feedback of users in order to improve their predictions, feedback is also incorporated in the generic predictor model (see *Figure 3-2*).

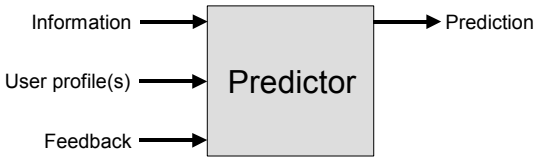


Figure 3-2 Feedback added to the predictor model

In the Duine prediction framework, feedback consists of a given rating on the same normalized scale as the predicted rating and a certainty value indicating the trust the system should place in the given rating.

When a user explicitly provides feedback, the given rating normally has a certainty of 100% as the given rating represents the actual interest of the user; "explicit feedback is much more accurate [than implicit feedback]" (Goren-Bar & Glinansky, 2004, page 151). However, some argue that users do not always tell the truth when rating items or providing information about themselves (Rich, 1998); people even show a certain inconsistency when rating the same items twice (Hill, Stead, Rosenstein & Furnas, 1995). For this reason, one can use a lower level of certainty, e.g. 90%, for explicitly given feedback.

As implicit feedback is not directly acquired from the user, it is necessary to give implicit feedback a lower certainty than explicit feedback. The amount of certainty depends on the actions observed and how indicative these actions are for the actual interest of the user; this depends on the domain. More details about the use of feedback and certainty in the learning process of predictors are provided in section 3.4.4.

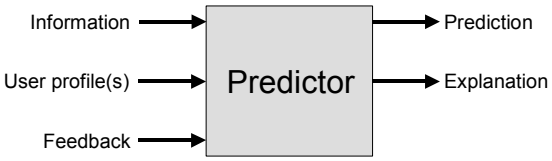
Explanations

Optionally, predictors can provide *explanation* data. Explanations provide transparency by providing the reasoning and data behind a prediction and can increase the acceptance of recommenders (Herlocker, Konstan & Riedl, 2000). Explanations help users to decide whether to accept a prediction or not and to understand the reasoning when a prediction is inaccurate. This in turn helps to increase the trust a user has in a recommender.

Explanations can be given in various forms, such as detailed explanations based on those attributes of an item that are important to the user, and in different formats, such as text, tables or graphs. Each prediction technique often has a form of explaining its predictions that is more suited to that technique, e.g. collaborative filtering can best used statistical information

about how similar users rated the item, while techniques that base themselves on attributes of the content, such as the multi-attribute utility theory techniques, can use these attributes, their values and importance to the user in its explanations.

Figure 3-3 Explanations added to the predictor model



Validity indicators

In order to make informed decisions about when predictors are useable, each predictor in the Duine prediction framework exposes so-called *validity indicators*. These validity indicators allow the framework to employ the switching hybridization method.

Definition 17 Validity indicator

A feature of a predictor that provides information about the state of the predictor that can be used to determine how useful the predictor will be in predicting the user's interests.

The state of a predictor is described by the amount and quality of knowledge that is available to the predictor, more specifically the knowledge that is used by the predictor to base its predictions on.

Validity indicators are analogous, although developed independently, to the concept of reliability indicators as described by Toyama & Horvitz and Bennett, Dumais & Horvitz whom use these indicators to combine multiple visual tracking algorithms (Toyama & Horvitz, 2000) and text classification algorithms (Bennett, Dumais & Horvitz, 2002). Ardissono et al. (2004) use the confidence each prediction technique has in its own predictions as a kind of validity indicators; these confidence values are used as weights to integrate the predictions of multiple techniques, i.e. weighted hybridization.

Because of differences in predictors, most predictors have unique and different validity indicators. E.g. where collaborative filtering provides 'the number of similar users that rated the item', CBR provides 'the number of similar rated items by the user'. Validity indicators can have different types of values; e.g. the validity indicator 'the number of similar users that rated the item' from the collaborative filtering prediction technique is an integer value, while the validity indicator 'certainty' from the GenreLMS prediction technique is a decimal value; the prediction technique AlreadyRated even has a Boolean validity indicator 'Known'. Validity indicators of various

predictors used in experiments with the Duine prediction framework are described in detail in section 5.2.2.

Validity indicators are the sixth and final aspect of the generic predictor model. The full model is shown in *Figure 3-4*.

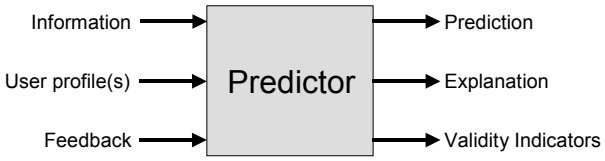


Figure 3-4 Full predictor model

3.2 Prediction strategies

The central concept in combining multiple predictors using the switching hybridization method is the prediction strategy.

Definition 18 Prediction Strategy

A predictor that generates an interest prediction for an item and user, not using an algorithm but by selecting between and/or combining predictors based on the most up-to-date knowledge about the current user, other users, the information, other information and the system.

To make decisions about which predictors to use and how to combine them, a prediction strategy uses the validity indicators of the predictors that are used within the prediction strategy; these indicators provide up-to-date knowledge about the relevancy of each predictor for the current prediction request. The way a prediction strategy makes a decision is called the prediction strategy decision approach. These prediction strategy decision approaches handle any conflict that may occur when multiple predictors are capable of providing a prediction; the decision approaches decide when to use which predictor. Examples of decision approaches are decision trees and rules, artificial neural networks, and case-based reasoning. Chapter 4 describes various prediction strategy decision approaches in more detail.

Black box perspective

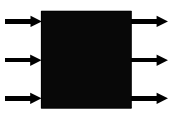


Figure 3-5 Black box perspective of a prediction strategy

From a black box perspective, prediction strategies are no different than prediction techniques. Both predict the interest in an item for a user, both can learn from feedback provided by the user, both can provide explanations about their working, and both can expose validity indicators on which decisions can be made about their suitability for a certain prediction: both are predictors (see *Figure 3-6*). For this reason, the generic

model of a predictor applies to both prediction techniques and prediction strategies.

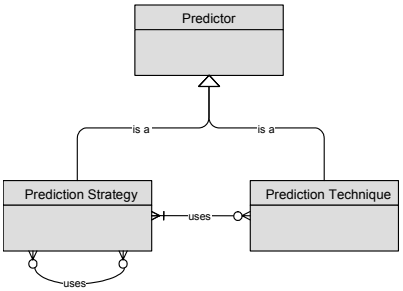


Figure 3-6 Predictor: a super class for prediction techniques and prediction strategies

Transparent box perspective

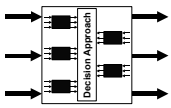


Figure 3-7 Transparent box perspective of a prediction strategy

When looking into the black box, prediction techniques actually generate predictions based upon the user profile and information, whereas prediction strategies only choose one or more predictors (prediction techniques and/or other prediction strategies) to generate the predictions for them using a prediction strategy decision approach.

Nested prediction strategies

As prediction strategies are also predictors it is possible to nest prediction strategies when necessary; use prediction strategies within other prediction strategies. This, for example, allows a strategy to be created that can be used as a fallback for other strategies; when various strategies are not capable of providing a prediction they can all use that fallback strategy to generate a prediction.

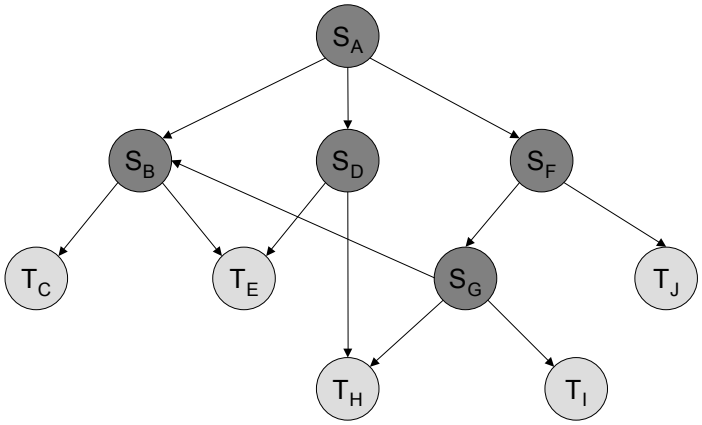


Figure 3-8 Nested prediction strategies are oriented acyclic graphs

The structure of (nested) prediction strategies is an oriented acyclic graph (see *Figure 3-8*), where all the local sinks are prediction techniques (T) and all the regular nodes are prediction strategies (S). One prediction strategy is the global source of the graph (S_A) and is the starting point for prediction requests. The fact that prediction strategies are oriented acyclic graphs is important for learning from feedback (see section 3.4.4).

3.3 Prediction engines

Besides the concepts prediction technique, prediction strategy and predictor, there is also another main concept in the Duine prediction framework, namely the prediction engine.

Definition 19 Prediction Engine

A software component that processes every incoming request for predictions and is the main entry point for processing feedback.

A prediction engine is the interface for predictions to other components of a personalized information system; e.g. recommender systems. A prediction engine directs prediction requests to the global source of a prediction strategy graph. Prediction engines have four tasks:

1. *Separating a request for multiple items into single item prediction requests*. Where predictors work on generating a prediction for a single item, prediction engines can process requests for multiple items. It is a task of a prediction engine to convert the one request for a set of multiple items into multiple requests for a single item and to later on combine the prediction results into a set again.
2. *Separate a request for multiple users into requests for individual users*. Some prediction engines are not only capable of predicting how interesting items are for an individual user, they can also predict how interesting a set of items is for a group of users (O'Connor et al., 2001). O'Connor et al. (2001) describe two broad approaches for group recommendations: the first is merging the user profiles of the individuals and recommending items based on the merged profile or secondly, recommend items for each individual user and merge the results. There are several methods for merging the user profiles and integrating individual predictions into group predictions (Masthoff, 2004). The prediction framework supports both approaches. It is a task of a prediction engine to either merge the user profiles and use this profile when calling the main predictions strategy or to gather predictions for each individual user of the group and then combine these predictions into predictions for the group. However, the issue of group recommendations still requires more research, especially to determine

the domain specific requirements of groups. Group predictions are not further discussed in this thesis.

3. *Direct requests for various classes of items to the correct prediction strategies.* When a prediction engine supports various classes of items at the same time, it is a task of a prediction engine to direct the requests for each class of items to the main prediction strategy that belongs to that class; e.g. predictions for books should be made by a prediction strategy that is tailored to provide the best predictions for books and not by a prediction strategy that is tailored to provide predictions for music. This also means that each prediction strategy and the prediction techniques used within that strategy are focused on only one class of items and they do not have any knowledge about the interests of users in other classes.
4. *Process feedback gathered from users.* Feedback acquired from users, either explicitly or implicitly provided, must be provided to each predictor in the prediction strategy graph, so every predictor is allowed to learn from the feedback to improve its future predictions. The order in which predictors are allowed to learn from the feedback is important (see section 3.4.4). Similarly as with prediction requests for various classes of items, it is a task of the prediction engine to propagate feedback about a certain class of items to those prediction strategies that are capable of dealing with that class.

The relations between a prediction engine, prediction strategies, prediction techniques and predictors are summarized in *Figure 3-9*.

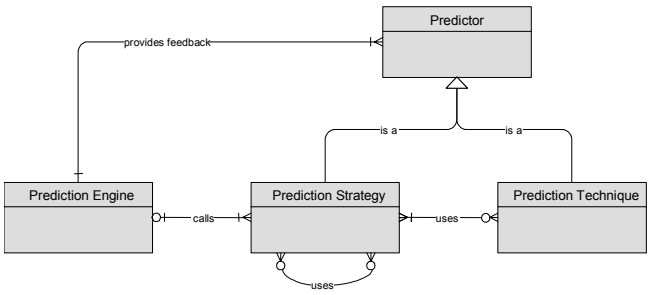


Figure 3-9 Relations between prediction engine, prediction strategy, prediction technique and predictor

3.4 Prediction process

So far, the structural aspects of the Duine prediction framework have been discussed. This section addresses the dynamic aspects of the prediction framework: the prediction process. The prediction process is described for a single prediction for a single item for a single user; separating combined

prediction requests in single prediction requests (e.g. for multiple items, for multiple users) has already been addressed when discussing prediction engines.

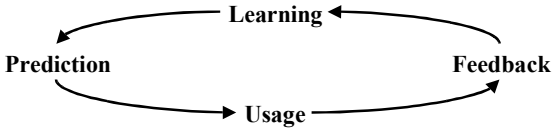


Figure 3-10 Prediction Process

The prediction process consists of four main phases (see *Figure 3-10*):

1. *The prediction phase*: a prediction is generated for an item for a user.
2. *The usage phase*: the user and/or other parts of a recommender system use the generated predictions.
3. *The feedback phase*: after the user has used one or more items, he can provide feedback to the system about his actual interest in the items.
4. *The learning phase*: the system learns from the feedback provided by the user, which is used to increase the accuracy of future predictions.

These four phases do not have to follow each other immediately. It is possible that there are longer periods of time between phases; e.g. after a prediction has been generated and presented to the user, it may take a few hours or days before the users provides feedback on the prediction (if feedback is provided at all).

3.4.1 The prediction phase

The prediction process consists of two sub processes: selecting one or more predictors that are best suited to provide a prediction and combining the predictions of the selected predictors when multiple predictors have been selected.

Selecting predictors

In the Duine prediction framework, the selection of predictors is the core of the framework and is based upon the validity indicators of predictors. The method to select which predictors are best suited to provide a prediction depends on the decision approach used in the prediction strategy. Various decision approaches are discussed in the next chapter.

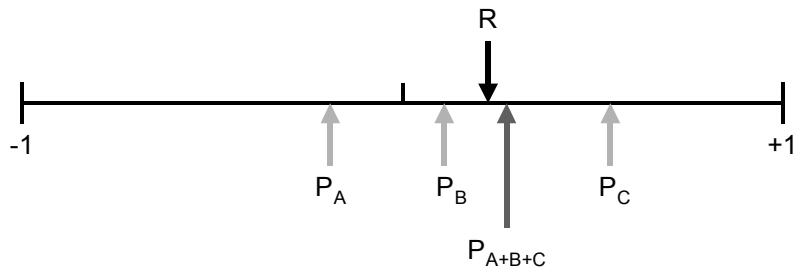
Combining predictors

Besides employing a switching hybridization method, combining the output of (a few selected) predictors into one prediction using a weighted hybridization method may also increase the accuracy of a prediction engine (see *Figure 3-11*). The prediction of the combined predictors (P_{A+B+C}) can

be closer to the rating a user would give (R) than the predictions of the individual predictors (P_A, P_B, P_C).

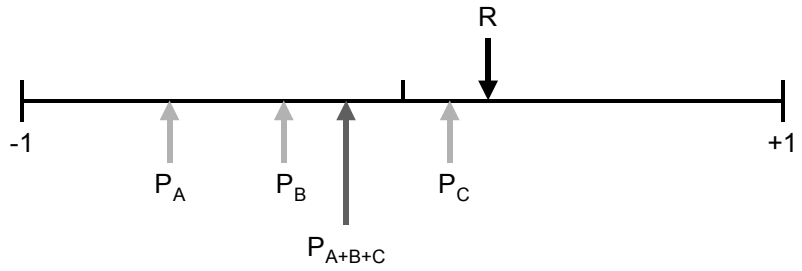
Various statistical central tendency methods can be used to combine predictions; e.g. mean, median, mode, trimean, weighted mean. Statistical central tendency methods are used to estimate the actual value of a parameter from a set of different samples; estimating the actual rating from a set of predictions.

Figure 3-11 Why combining predictors could increase prediction accuracy



However, the combination of predictions may also decrease the accuracy of a prediction engine. One of the properties of statistical central tendency methods is that the estimated value is always situated between the lowest and the highest sample, e.g. between the lowest and highest prediction. When all predictors produce predictions that are on the same side of the rating the user would give, all too low or all too high, the combined prediction will be less accurate than the prediction of the best individual predictor (see *Figure 3-12*).

Figure 3-12 Why combining predictor could decrease prediction accuracy



Experiments with the Duine prediction framework have shown that combining predictions actually decreased the accuracy of the prediction engine in the used datasets. With most prediction requests (in about 75% of the cases) the individual predictors all predicted on the same side of the given rating.

This observation can be attributed to the fact that all predictors base their predictions on the same user profile; the same known interests of the user and the same ratings provided by the user. This means that the

predictions from the various predictors for the same prediction request cannot be regarded as independent samples, weakening the power of the central tendency methods.

Besides using central tendency methods to combine multiple predictions also other methods can be used such as neural networks or Bayesian networks. Some research has been done in combining predictions using neural networks (Zimmerman et al., 2004); this is not further investigated in this thesis.

3.4.2 The usage phase

After the prediction engine has returned the item set with added predictions, a decision can be made based upon those predictions. The three most common types of usage for predictions are filtering, sorting and presenting; combinations of these three are also possible.

Filtering

Most recommender systems remove those items that are not interesting enough for a user from the item set; they only return the set of items that are of interest to a user: the recommended items. Filtering can both be done by a recommender system or by a user; a recommender system removes uninteresting item before presentation, a user ignores or deletes uninteresting items. Even though a recommender system may filter the item set, this can still be based on input from a user; e.g. a threshold provided by a user, indicating which predicted rating values are interesting enough and which are not.

Sorting

When sorting the set of items using predictions, the most interesting items are placed at the top of the list and the least interesting items at the bottom of the list; items are sorted on predicted ratings in descending order (if the user wants to have the least interesting items returned, the set is sorted in ascending order). Sorting is part of the structuring process; chapter 6 discusses structuring as way to support users in finding interesting items in more detail.

Presenting

Predictions can be presented in various ways, such as using an icon or a number of icons to indicate the predicted rating, using low- and highlighting of items or colouring items. In chapter 7 various presentation forms are discussed and preferences of users for these forms are examined.

3.4.3 The feedback phase

When a user has access to items with predictions, he will make the final decision of which item(s) to use. This choice is already an indication of the user's actual interests and can be used by recommender systems to learn from in order to improve future predictions. Feedback is an indication by a user about his real interest in an item. As discussed in section 3.1, there are broadly two types of feedback: implicit feedback and explicit feedback.

When using feedback, either implicitly acquired or explicitly provided, a recommender has to determine how certain it can be that the acquired feedback really represents the user's actual interest. Implicit feedback has a lower level of certainty than explicit feedback. For example, if someone has the TV tuned to the same program for more than an hour, this does not necessarily mean that he likes that program; he could as easily have been called away and forgot to switch of the TV; if he had stayed watching, he might have switched to another program. However, if that same user explicitly rated that TV program as being fun, a recommender system is much more certain that this program is indeed of interest to the user. Explicit feedback has a higher certainty than implicit feedback.

3.4.4 The learning phase

The goal of learning in prediction engines is to provide more accurate predictions on future prediction requests. A prediction engine first allows each predictor to learn from the feedback provided by the user. Than it stores the feedback in the user profile.

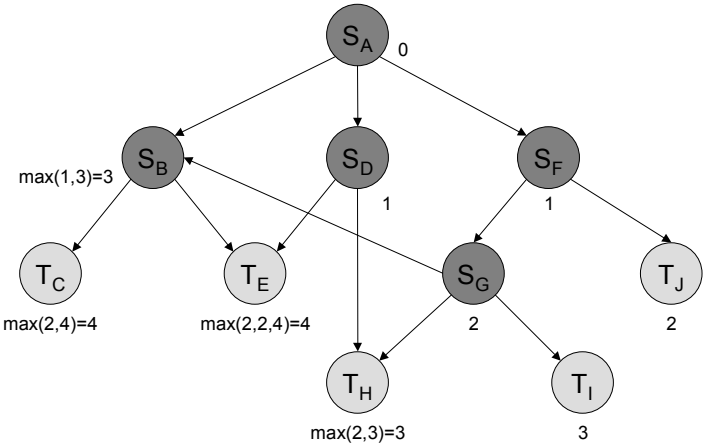
Feedback cannot be propagated to the predictors via the hierarchy of strategies as this causes prediction techniques and prediction strategies that are used in multiple prediction strategies to receive the same feedback multiple times, which can disrupt learning algorithms. For this reason, a prediction engine should use the 'maximum path – breadth first technique.'

'Breadth first' because higher levels need to learn before lower levels, as the learning process may depend on the predictions and validity indicators of lower levels, which change after a technique or strategy on the lower level has learned from feedback. It is the task of a prediction strategy to predict when to use with child predictor; it learns which child predictor it should have used when it receives feedback from the user for an item. The strategy can then asks all its child predictors to generate a prediction, which it can compare with the received feedback. If child predictors would learn the feedback before the prediction strategy, the prediction strategy would be learning incorrectly as the predictions of the child predictors would then also be based on that piece of feedback.

'Maximum path' is used in order to prevent a technique or strategy that is used on multiple levels to process feedback before one of his parents. A

path on a graph is a sequence $\{x_0, x_1, \ldots, x_n\}$ such that (x_1, x_2), (x_2, x_3), ..., (x_{n-1}, x_n) are edges of the graph (Weisstein, 2004). The length of a path is the number of edges between the first node x_0 and the last node x_n. Every path in a feedback graph starts with the main prediction strategy (S_A in example 1).

Example 1 Maximum path – breadth first method



In this example, a prediction strategy A uses three other prediction strategies: B, D and F. Prediction strategy B uses two prediction techniques, namely C and E. Prediction technique E is also used by prediction strategy D, which also uses prediction technique H. Prediction strategy F uses another prediction strategy G and a prediction techniques J. Prediction strategy G uses the prediction techniques H and I and also prediction strategy B.

If prediction technique H would be allowed to learn before prediction strategy D, the prediction of H would change; hence prediction strategy D could not learn how well H predicted before it learned from the feedback, hence D would observe a better prediction from H than it actually provided before it learned from the given feedback.

When applying the maximum path – breadth first method, first the maximum path of each predictor is calculated (as indicated in the figure); e.g. the length of the path between A and H is 2 via D but when going via F and G the length between A and H is 3, which is longer.

Then using breadth first, the predictors are allowed to learn in the order from least depth till highest depth. This means that feedback in this example is processed in the following order: A, D, F, G, J, B, H, I, C, E

Prediction technique learning

Instance-based prediction techniques do not have to learn explicitly from feedback provided by a user as they use the ratings that are stored by the prediction engine in the user profile. With instance-based learning that what has been learned is represented as a collection of prototypes stored in

the same format as it has been acquired for learning (Quinlan, 1993b), i.e. the feedback for a prediction technique: the given rating.

Model-based prediction techniques on the other hand do have to explicitly update their model by learning from the feedback. In model-based learning, that what has been learned is represented in some language that is richer than the format of the learning examples. The learning method constructs some kind of explicit generalization of the training cases, rather than allowing generalization to flow implicitly from the learning examples (Quinlan, 1993b).

For instance-based prediction techniques the ratings are stored in the profile of the user; model-based prediction techniques store their own models in the user profile. These models are not accessible for other prediction techniques, unless they explicitly share the same model.

Prediction strategy learning

Depending on the used prediction strategy decision approach, prediction strategies can also learn from feedback provided by the user. Prediction strategies improve their prediction accuracy by learning better when to use which predictors. Not every type of prediction strategy decision approach is capable of learning, e.g. a manually created decision rule-based approach is unable to update its rules automatically, while a neural network based approach is able to update its model. As choosing a prediction strategy decision approach is an important decision when designing a prediction engine based on the Duine prediction framework, chapter 4 discusses prediction strategy decision approaches in more detail.

Certainty and learning

Certainty is not only used when processing feedback; certainty can also be used to learn how accurate knowledge about a user stored in the user profile is. There are five events that influence certainty in knowledge about a user: learning, reinforcement, degradation, removal and contradiction (see *Figure 3-13* and *Figure 3-14*).

When something new is *learned* about a user, the level of certainty depends on the reliability of the source of that information (explicitly or implicitly acquired). When evidence is acquired that confirms something already known about a user, the certainty in that piece of knowledge can be increased (*reinforcement*). However, when over a longer period of time no more evidence is found that confirms a piece of knowledge about a user, certainty in that piece of knowledge can slowly be decreased (*degradation*) until at some point certainty about that piece of knowledge becomes so low that it can be *removed* from the user profile. Degradation is a way to support slow shifts in interests of people (Lam, Mukhopadhyay, Mostafa & Palakal, 1996).

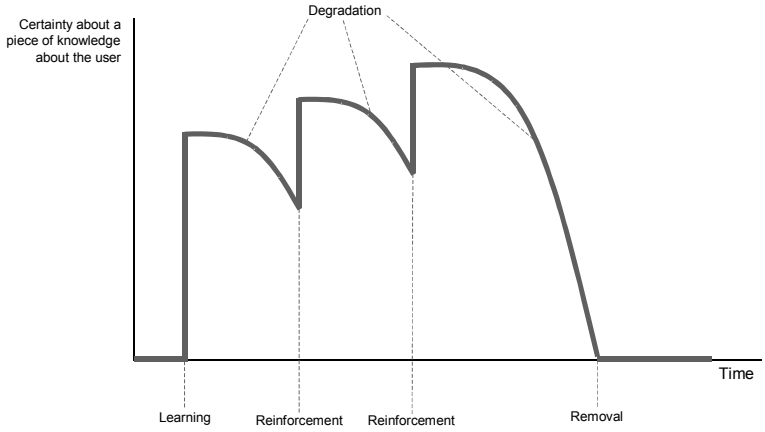


Figure 3-13 Certainty learning, reinforcement, degradation and removal

It may even be possible that acquired feedback *contradicts* something already known about a user. A contradiction should result in a major decrease in certainty about that piece of knowledge and may even trigger an immediate removal of that piece of knowledge. Contradictions support abrupt change in interests or erroneous feedback.

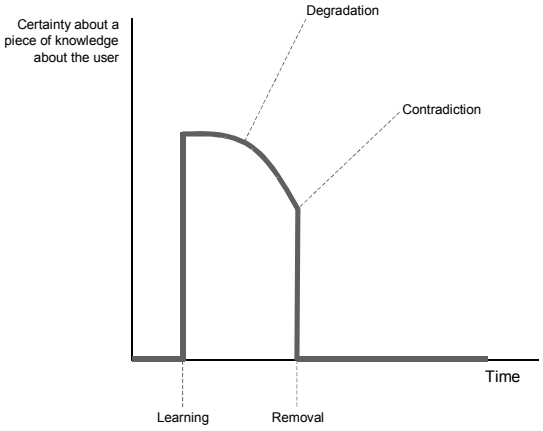


Figure 3-14 Contradictions in learning knowledge about the user

When feedback has been processed the whole prediction cycle is complete. New prediction requests can be made at the prediction engine, starting a whole new prediction cycle. It is not always necessary to acquire and process feedback before making other prediction requests. However, acquiring feedback and learning, i.e. closing the loop, is important in order to keep improving the accuracy of predictions and to allow predictors to detect changes in interests of the user.

3.5 Framework extension

In this chapter, the basic Duine prediction framework has been described. This domain-independent framework allows for the creation of prediction engines in various domains. However, the framework is extensible; where necessary, it is possible to extend the framework with additional decision data; two such extensions to the framework have been investigated. The first, tuning parameters, makes the framework more flexible and easier to implement in various domains. The second, context-awareness, introduces a new source of information on which to make more accurate predictions taking into account the current user's context.

These two extensions to the basic Duine prediction framework are not described in the remainder of this thesis, but are only provided as examples of how the framework can be extended.

3.5.1 Tuning parameters

Several prediction techniques contain a generic algorithm that can be tailored to specific situations. For example, in the second step of collaborative filtering the set of most similar users to the current user has to be chosen. This choice is often made by using a correlation weight threshold (Herlocker et al., 2002). The optimal value of this threshold differs per system. Such a threshold is called a *tuning parameter*.

Definition 20 Tuning parameter

A value that does not change the conceptual behaviour of a predictor but which can be used to optimize the prediction accuracy of a predictor for a specific system.

Not only prediction techniques can employ tuning parameters, also prediction strategies can use them; e.g. instead of using hard coded values in the conditions of a decision rule-based prediction strategy, which are compared with validity indicators, tuning parameters can also be used; this makes a prediction strategy tailorable.

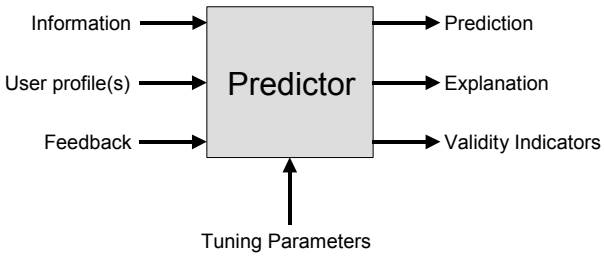


Figure 3-15 Tuning parameters as an extension of the generic predictor model

3.5.2 Context-awareness

Context is any information that can be used to characterize the situation of an entity. An entity is any person, place or object that is considered relevant to the interaction between a user and an application, including the user and application themselves (Dey, 2000). Examples of contextual information are location, time, proximity, user status and network capabilities. "A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task" (Dey, 2000). The key goal of context-aware systems is to provide a user with relevant information and/or services based on his current context. This goal matches with the goal of recommender systems. Both context-aware systems and recommender systems are used to provide users with relevant information and/or services; the first based on the user's context; the second based on the user's interests. Therefore, the next logical step is to combine these two systems.

In van Setten, Pokraev & Koolwaaij (2004) we describe the integration of context-awareness and recommendations in a mobile tourist application, including the use of predictors that provide predictions based on contextual information. For this, the generic predictor model has been extended with *contextual information*.

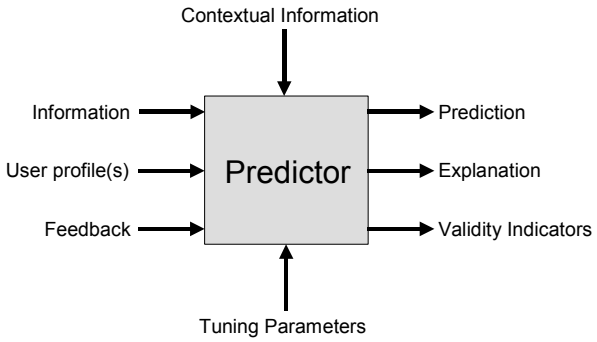


Figure 3-16 Contextual information as an extension of the generic predictor model

3.6 Conclusion

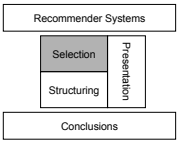
In this chapter, the Duine prediction framework has been introduced. This framework provides a way to create prediction engines using multiple prediction techniques via prediction strategies that employ the switching hybridization method. These prediction engines can be used in recommender systems that help people in finding information that is of interest to them. Although the framework itself is domain-independent, it

allows for domain-specific implementations of prediction engines that are optimized to a domain, providing accurate predictions and recommendations. This prediction framework addresses the first research objective of this research.

The next chapter goes into more detail on an important design aspect of the framework, namely the prediction strategy decision approach to use in a prediction strategy. Such a decision approach has to decide which predictors are best suited for a specific prediction request. Empirical evidence of the applicability and strength of the Duine prediction framework is provided in chapter 5, where experiments with the framework are described.

Chapter 4

Prediction Strategy Decision Approaches



In order to help people find items that are of interest to them within the enormous amount of information, products and services that are available nowadays, this thesis focuses on personalized information systems that adapt themselves and the items they retrieve to better fit the needs of the user. Within the selection process of personalized information systems, which is the focus of chapters 2 till 5, the focus lies on recommender systems and specifically on the process in which a prediction is made about how interesting a user will find a specific item. The other processes within a personalized information system, structuring and presentation, are discussed in chapters 6 and 7 respectively.

The previous chapter introduced the Duine prediction framework, which allows for the creation of prediction engines that incorporate multiple predictors by using prediction strategies that employ the switching hybridization method. For each prediction request, switching hybridization chooses one prediction technique to provide the prediction. Predictions of multiple techniques are not combined as we have shown that in most cases a combination results in worse predictions. In other words, in prediction strategies, the most important decision to make with every prediction request is which predictor to use for generating a prediction. Several methods can be used to make this decision: the so-called prediction strategy decision approaches.

This chapter first introduces prediction strategy decision approaches (section 4.1). Then four major machine learning techniques that can be used as prediction strategy decision approaches are discussed in sections 4.2 to 4.5 including how they can be used as a prediction strategy decision approach. We furthermore argue in section 0 why we only use manually-created decision rules and case-base reasoning as prediction strategy

decision approaches for the validation of the Duine prediction framework in the next chapter. Section 4.7 summarizes and concludes this chapter.

Parts of this chapter have already been published in van Setten, Veenstra, Nijholt & van Dijk (2004).

4.1 Decision approaches

In order for prediction strategies to make decisions about which predictor is best suited to provide a prediction, a prediction strategy needs knowledge about predictors. Such knowledge can either be provided manually by experts or by using automated decision approaches that allow a prediction strategy to teach itself when to use which predictor.

When domain experts create a knowledge base (see *Figure 4-1*), they, or special knowledge engineers, must formalize what is known about the strengths and weaknesses of each predictor and when each predictor is best suited to provide predictions; such formalisation must be represented in a way that can be understood by machines. "A knowledge engineer must understand enough about the domain in question to represent important objects and relationships" (Russell & Norvig, 1995).

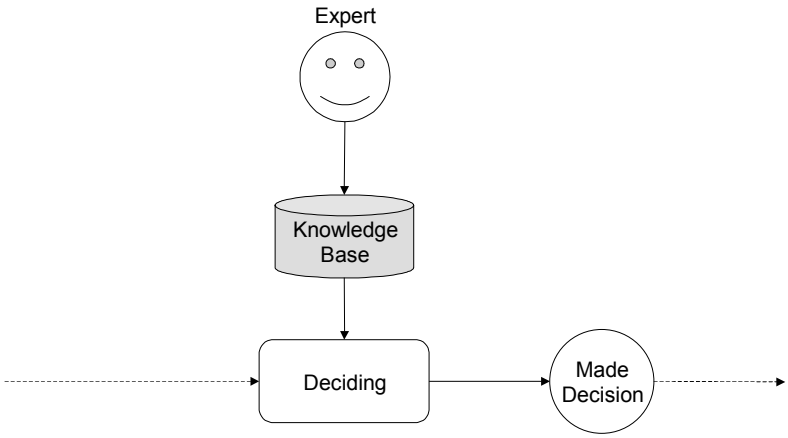


Figure 4-1 Knowledge base created by domain experts

Automated decision approaches employ machine learning techniques. Machine learning is an area of artificial intelligence that develops techniques that allow computers to learn. "Machine learning is concerned with the question of how to construct computer programs that automatically improve with experience" (Mitchell, 1997). The manner in which knowledge is stored on which decisions are based also varies. Knowledge can either be stored as instances or as a model.

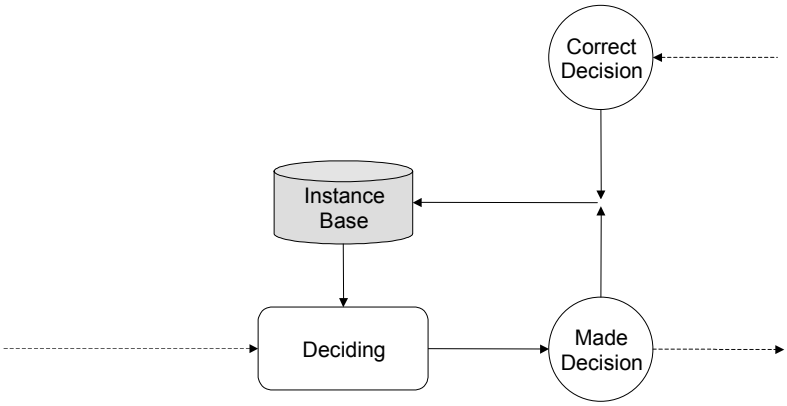


Figure 4-2 Instance based learning

Instance-based knowledge uses examples to describe what is known; these examples are called instances or cases (see *Figure 4-2*). When making a decision, all the known instances are used (Witten & Frank, 2000). Instances describe situations in which a decision had to be made and what the correct decision was for that situation. One might argue that with instance-based knowledge a system does not learn, it just stores instances. The burden of deriving knowledge from instances lies in the decision process itself not in a separate learning process. Instances can be provided by experts and/or be acquired during usage of a system based on feedback from users (automatically acquired instances).

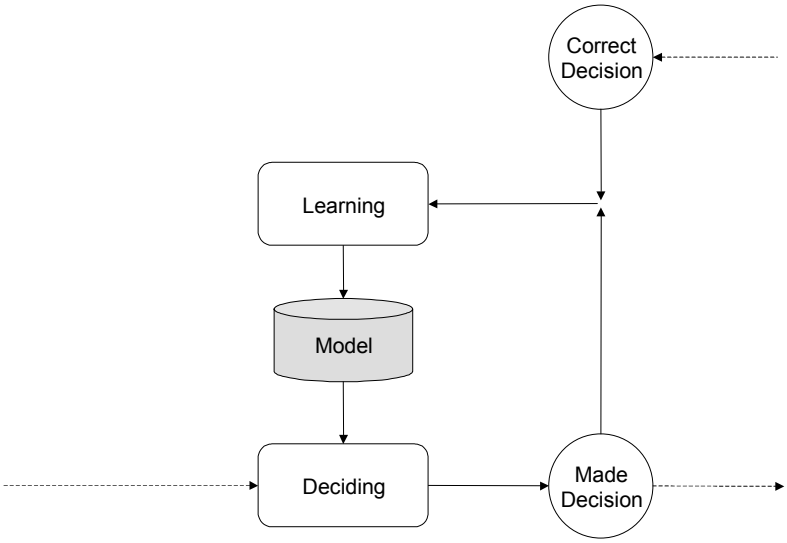


Figure 4-3 Model based learning

Instead of storing instances, a model can also be used to describe knowledge that is used to make a decision (Witten & Frank, 2000). When using a model to represent knowledge, instances can still be used, but only to create or update a model from those instances (see *Figure 4-3*); the burden of deriving knowledge from instances now lies in the learning process, not in the decision process. The model is used in the decision making process to make a decision. Notice that knowledge bases created manually by experts can be both a model as well as a set of instances (see *Figure 4-1*).

In order for systems to keep making correct decisions, it is necessary to update the knowledge; to allow the system to keep learning. There are two ways in which knowledge can be kept up-to-date: using staged updates and dynamic updates. If a knowledge base is not kept up-to-date one speaks of a static knowledge base.

With staged updates the knowledge base, either a set of instances or a model, is updated or re-created outside the system and then the old knowledge base is replaced by the new knowledge base. One could argue that there are no static knowledge bases as every system can be updated by replacing its knowledge base. However, the major difference is whether facilities exist in a system that allow staged updates to take place regularly and perhaps even automatically.

Dynamic updating allows systems to update their knowledge base with every new example encountered during usage of the system; either the example is added to the set of instances or the model is updated. Typically, instance-based systems are dynamically updated, while model-based systems are updated in stages or kept static; however, exceptions occur; the set of instances from some instance-based systems are only extended every now and then, while some model-based systems update their model with every new instance encountered. Manually created knowledge bases or models are often kept static.

In the remainder of this chapter, four specific machine learning techniques are discussed: decision trees and decision rules (section 4.2), case-based reasoning (section 4.3), backpropagation artificial neural networks (section 4.4) and Bayesian probabilities (section 4.5). These four have been chosen as they are four major machine learning and knowledge representation techniques and they fully cover the three aspects discussed that distinguish these techniques: manually created versus automatically learned (learning), instance-based versus model-based (type) and no updates (i.e. static), staged updates or dynamic updates (updating), see *Table 4-1*.

Table 4-1 Machine learning techniques

Decision Technique	Learning	Type	Updating
Decision Trees and Decision Rules	Manual	Model	Static or Staged
	Automatic		
Case-Based Reasoning	Automatic	Instance	Static, Staged or Dynamic
Backpropagation Artificial Neural Network	Automatic	Model	Static or Staged
Bayesian Probabilities	Automatic	Instance	Dynamic
		Model	Static or Staged

Before discussing these four techniques and how they can be used as prediction strategy decision approaches, a description is given of how automated decision approaches can teach themselves in general which predictor to use in which situations; decision approaches are in fact meta-learners. Also an issue that is of major concern for using automated learning techniques in prediction strategies is discussed: noise.

4.1.1 Meta-learners

A predictor tries to predict how interested a user is in an item by learning the interests of the user; a predictor is a learner. A prediction strategy decision approach tries to predict when to use which predictor; it learns the strengths and weaknesses of each predictor; i.e. of each learner. Thus, decision approaches for prediction strategies are meta-learners (Chan & Stolfo, 1993).

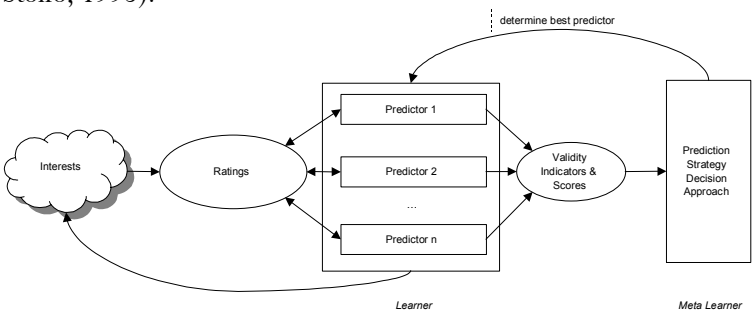


Figure 4-4 Prediction strategies as meta-learners

Both learners and meta-learners have to learn from data (see *Figure 4-4*). Predictors learn from the ratings and other feedback provided by users. Prediction strategy decision approaches cannot learn from these ratings; they do not need to know about users' interests, they need to know how well predictors perform under various circumstances.

A prediction strategy decision approach can only learn how well the different predictors performed when feedback has been received from the user. This feedback represents the actual interest of the user, which the

decision approach can compare with the predictions of the individual predictors in order to determine which predictor(s) retrospectively predicted best.

As validity indicators provide information that can be used to decide whether to employ a predictor or not (see section 3.1), these indicators can also be used by decision approaches to describe the circumstances of a prediction request. The outcome of a predictor is a score indicating the predicted interest of the user; the outcome of the decision within a prediction strategy is not the prediction itself but data that shows which predictor can best be used in that situation; either a reference to the best predictor or an indication of how well each predictor is expected to perform for the prediction request. The prediction error of a predictor can be used as this performance indicator for a certain request. The prediction error is the difference between the prediction and the given rating of the user; the lower the prediction error the more suitable that predictor was, in retrospect, to generate the prediction.

Summarized, prediction strategy decision approaches can learn from each prediction request for which feedback has been received; the data from which they learn consists of the validity indicators of each predictor and either a reference to the best predictor or for each predictor the prediction error for that request.

4.1.2 Noise

For any learning algorithm, noise is an issue. Noise for learning algorithms occurs when examples have exactly the same attribute values but different results (Russell & Norvig, 1995). For a prediction technique this means that the same item received different ratings by the same user. For prediction strategies this means that all validity indicators of the used predictors have the same values in two situations, indicating exactly similar prediction request circumstances, but some instances state that predictor A should be used, while other instances state that predictor B should be used.

For automated prediction strategy decision approaches noise is not only a greater problem than for predictors, but also a greater problem than for most meta-learners in domains that do not try to predict human interests; prediction strategy decision approaches have to deal with multiple and cumulative sources of noise. The first three sources of noise, discussed below, already provide difficulties for interest predictors; since decision approaches learn about predictors, they also create problems for these meta-learners:

– *Ambiguity in people's interests*: The interests of people change over time and due to variations in context (Lam et al., 1996), such as emotional or physiological state, making a person's observable interests seem

ambiguous. Since it is very difficult, if not impossible, to fully capture all the circumstances under which a rating has been given, gathered ratings also contain this apparent ambiguity. E.g. when a certain user is relaxed he may rate a comedy show on TV with 5 stars, while that same user may rate that same TV program with only 2 stars if he is stressed.

- *Inconsistent ratings*: People are not always consistent in expressing their interests using ratings (Hill et al., 1995); this inconsistency is also a source of noise in the gathered set of ratings. E.g. at some moment a certain user rates TV program X with 4 stars; two months later he rates TV program Y that he likes just as must as TV program X but he give Y 3.5 stars instead of 4 stars, not because he likes Y less, but because he did not think about or remember that he rated TV program X with 4 stars.
- *Uncertainty*: When the user provides explicit feedback, the recommender system can be almost certain that the provided feedback is correct (excluding the other mentioned sources of noise). However, with implicit feedback there is a higher level of uncertainty that can cause noise in the data. For example, when a recommender system monitors the TV watching behaviour of a person and it sees that the TV has been tuned to one program for the whole duration of that program, it might deduce that the user likes the program. However, it could also have been the case that the TV was left one while the user was talking to a neighbour at the front door. In the experiments described in the next chapter, we will only use ratings that have been provided explicitly by the user, i.e. eliminating this source of noise.
- *Observation errors*: Plain observation errors also occur; e.g. a user wants to rate an item with four stars, but mistakenly clicks on 3 stars without noticing his mistake; errors in transferring a rating from the user's machine to a user profile database; errors in the storage of the given ratings. Although these errors do not occur very often, if they occur, they are a source of noise in the gathered data.

Due to these sources of noise, interest predictors experience difficulties in providing equally accurate predictions under the same circumstances (reflected by the same validity indicator values), which leads to noise in the set of data used for prediction strategy decision approaches to learn from. However, decision approaches also have to deal with an additional source of noise:

- *Imperfect predictors*: Since there is no theoretical basis for any existing interest predictor that can fully explain the reasons for a person to be interested in certain items, predictors have a fundamental imperfection that results in a variation of prediction quality under similar observable circumstances. E.g. in both situation A and B, the validity indicator of

collaborative filtering shows that there are 250 people with similar taste to the current user that have already rated the movie for which a prediction must be made. In situation A, the resulting prediction matches the actual rating of the user perfectly, while in situation B it is completely wrong. When examining the two situations more closely it appears that the movie in situation B has one actor that is really disliked by the user, a situation that could not be detected by collaborative filtering.

Prediction techniques based on well-founded theories may be able to provide more accurate and solid predictions than the currently available techniques; one such theory is that people are interested in items based on the goals they have for an item; this theory is investigated in chapter 6.

All these sources lead to a noisy set of data from which prediction strategy decision approaches have to learn; prediction requests that are described by the same set of validity indicator values result in different predictors being the best predictors for that request. Good prediction strategy decision approaches have to be able to deal with this high level of noise.

The next section discusses the four major identified machine learning techniques and how they can be used as prediction strategy decision approaches. The techniques are also evaluated on their capabilities of learning from noisy data sets.

4.2 Decision trees and decision rules

Decision tree learning is "a method for approximating discrete-valued target functions, in which the learned function is represented by a decision tree" (Mitchell, 1997). Each node in a decision tree represents some attributes and each branch from a node corresponds with a possible value for that attribute. When trying to classify a certain instance, one starts at the root of the decision tree and moving down the branches of the tree according to the values of the attributes until a leaf node is reached. The leaf nodes represent the results of the decision tree. An abstract example of a decision tree is shown in *Figure 4-5*.

Another way to represent a decision tree is using decision rules (Witten & Frank, 2000). Decision rules generally take the form of IF … THEN … rules; e.g. IF attribute A = value X AND attribute B = value Y THEN result K

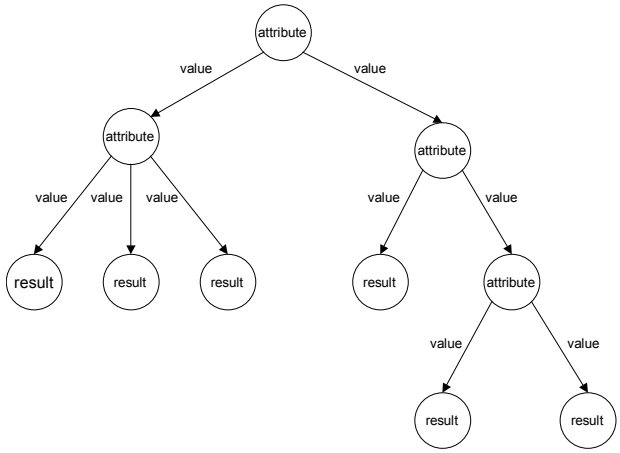


Figure 4-5 Example of a decision tree

Decision trees and decision rules can be either manually created by experts or learned automatically from a set of examples using algorithms such as ID3 (Quinlan, 1986) or C4.5 (Quinlan, 1993a). Experts often describe their knowledge about a decision process using decision trees and decision rules, as they are easy to interpret by other people. As decision trees and rule sets contain the knowledge on which decisions are based and not a set of instances, decision trees and decision rules are model-based.

Automatically learning decision trees and rules requires a fair amount of time, especially if the set of data to learn from increases in size (Witten & Frank, 2000); this means that decision trees and rules are best learned once and be kept static or be updated in stages. Manually created decision trees or decision rules can only be used statically or be updated in stages as they require expert involvement, which cannot be provided in online systems.

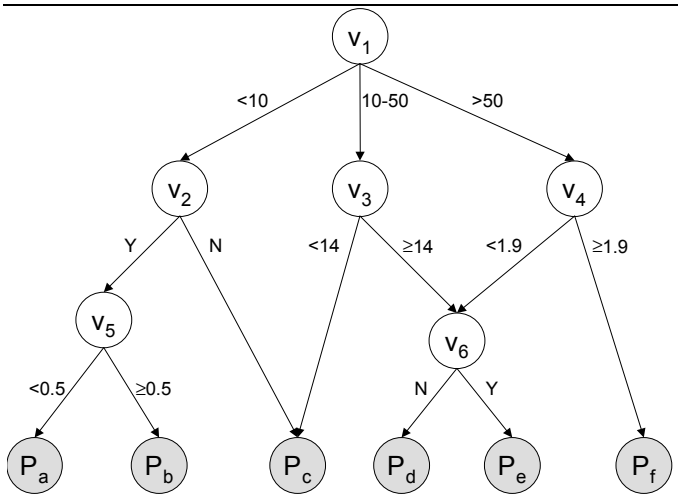
4.2.1 Decision trees and decision rules in prediction strategies

When employing a decision tree in a prediction strategy, nodes represent validity indicators and each branch from a node represents a value or range of values for that validity indicator. The leaf nodes of the decision tree represent predictors (see *Example 2* on the next page).

For manually created prediction strategies, decision rules are very useful as they are easy to interpret. When enough knowledge about the working of predictors is available they are also easy to create. However, learning a decision tree from a set of instances is more difficult, especially due to the high levels of noise. Although training data for learning decision trees may contain errors (Mitchell, 1997), the high levels of noise that prediction strategy decision approaches have to deal with, provide significant problems for decision tree learning algorithms. Furthermore, the standard decision tree learning algorithms cannot handle real valued attributes (Mitchell,

1997), which most validity indicators are. Extensions to the basic decision tree learning algorithms can be made to handle real valued attributes by segmenting the attribute's value range (Mitchell, 1997); however, high levels of noise severely complicate segmentation as noise blurs the borders between segments.

Example 2 Example decision tree and decision rules for a prediction strategy



IF $V_1 < 10$ AND $V_2 = Y$ AND $V_5 < 0.5$ THEN P_a
IF $V_1 < 10$ AND $V_2 = Y$ AND $V_5 \geq 0.5$ THEN P_b
IF ($V_1 < 10$ AND $V_2 = N$) OR ($V_1 \geq 10$ AND $V_1 \leq 50$ AND $V_3 < 14$) THEN P_c
IF (($V_1 \geq 10$ AND $V_1 \leq 50$ AND $V_3 \geq 14$) OR ($V_1 > 50$ AND $V_4 < 1.9$)) AND $V_6 =$ N THEN P_d
IF (($V_1 \geq 10$ AND $V_1 \leq 50$ AND $V_3 \geq 14$) OR ($V_1 > 50$ AND $V_4 < 1.9$)) AND $V_6 =$ Y THEN P_e
IF $V_1 > 50$ AND $V_4 \geq 1.9$ THEN P_f

4.3 Case-based reasoning

Case-based reasoning (CBR) is a method to solve new problems by adapting solutions that were used to solve past problems (Riesbeck & Schank, 1989). A case is a contextualized piece of knowledge representing an experience. It contains the past lesson that is the content of the case and the context in which the lesson can be used (Watson, 1997). When solving a problem with CBR, one searches for past cases that are analogous to the current case. The solutions of the most analogous past cases are then used to create a solution for the current case.

CBR has already been used as a prediction technique, e.g. in the PTV

system (Smyth & Cotter, 2000) (see also section 2.2.2). As a prediction technique, CBR searches for items that the user has already rated and that are similar to the item for which a prediction is required. Based on the ratings of the most similar previously rated items, a prediction is generated.

CBR is an instance-based machine learning technique, which can be used statically and with stages or dynamic updates. In the case of a static approach, a fixed set of cases is used, while with a dynamic approach, the set of cases is constantly updated based on the identification of new cases and solutions during usage of the system; with static updates instances are gathered and added to the case-base every now and then. The most common types are static case-bases and dynamic case-bases.

4.3.1 Case-based reasoning in prediction strategies

When using CBR as a prediction strategy, a case represents a specific prediction request for which feedback of the user has been received. As validity indicators provide information that can be used to decide whether to employ a predictor or not, these indicators can also be used by CBR to describe the case of a prediction request.

Prediction strategies can build a case-base in two ways. Either by creating one case-base using all validity indicators of all predictors per request or by creating different case-bases, one per predictor, where each description consists only of the validity indicators of that predictor. In the situation of one case-base, the outcome of each case is a reference to the best performing predictor of that case; the outcome of each case when using a case-base per predictor is the prediction error of that predictor.

For prediction strategies, it is best to use different case-bases per predictor. When only one case-base is used, issues arise when predictors are added to, changed in or removed from the strategy. All the old cases would become invalid because they are based on a set of predictors that no longer exists. Different case-bases per predictor allow the set of predictors to remain flexible; only for the new or changed predictor must the case-base be rebuilt.

Furthermore, when using one case-base, a much larger case-base is necessary before accurate decisions can be made by the strategy. This is due to the fact that the probability that a similar set of validity indicators occurs for one technique is much higher than the probability that a similar set of indicators occurs for multiple techniques.

When using a case-base for each predictor, a case is described by:

{validity indicator 1, validity indicator 2, …, validity indicator N → prediction error}

A few examples of case descriptions for the CBR prediction strategy decision approach are shown in *Table 4-2*.

Table 4-2 Example case descriptions

Number of similar items > 0.5	Number of similar items > 0.7	Number of similar items > 0.9	Prediction Error
4	8	2	0.3452
15	12	6	0.0453
8	2	0	0.8232
6	7	2	0.3582
…	…	…	…

For every prediction request, the goal of the CBR-based prediction strategy is twofold. First, determine the expected error of each predictor using those stored cases that have similar validity indicators as the current situation. Second, choose the predictor with the least expected error as the one that should provide the prediction.

As CBR does not try to derive a model from the set of instances, but uses all the instances (cases) to determine the most similar set of instances for a given request and then uses that set to determine the most appropriate predictor, we expect that CBR can deal with the additional levels of noise that prediction strategy decision approaches have to deal with, as noise will be averaged out when calculating the expected error.

4.4 Backpropagation artificial neural network

Artificial Neural Networks (ANN) are inspired by the workings of the brain, which consists of a very complex web of vast numbers of interconnected neurons; it is estimated that the human brain consists of approximately 10^{11} neurons, each connected to about 10^{4} other neurons (Mitchell, 1997). Each neuron is a simple system that has one or more inputs and one or more outputs. The basic function of each neuron is to take a number of real-valued inputs and to calculate a linear combination of these inputs; if the result is greater than some threshold the perceptron outputs a value of 1 otherwise 0.

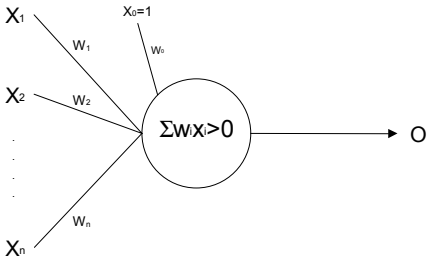


Figure 4-6 Basic unit of an artificial neural network: the perceptron

The basic component of an artificial neural network is the perceptron (see *Figure 4-6*). The perceptron has a number of input variables (x_1, x_2 to x_n). For each of the inputs, the perceptron has an associated weight (w_1, w_2 to w_n) and a weight (w_0) that is not related to any input variable ($x_0 = 1$); this unrelated input variable represents the threshold value. The perceptron multiplies each input variable with its associated weight, including w_0 and sums it together. If the result is greater than 0, then the output node is positively activated ($o = 1$), otherwise the output node is negatively activated ($o = -1$) or not activated ($o = 0$). A perceptron is capable of learning by updating the weights according to what the output ought to have been. Perceptrons can only learn linearly separable functions (Russell & Norvig, 1995).

Multilayer ANNs are capable of learning non-linear functions. However, when using the simple threshold function for each perceptron, even a multilayer ANN still can only learn a linear function (Mitchell, 1997). One solution is to use a sigmoidal threshold function instead of a discrete threshold function, which results in a perceptron with a smoothed threshold function (see *Figure 7-1*). An ANN with three layers of units using sigmoidal threshold functions can approximate any arbitrary function (Cybenko, 1988).

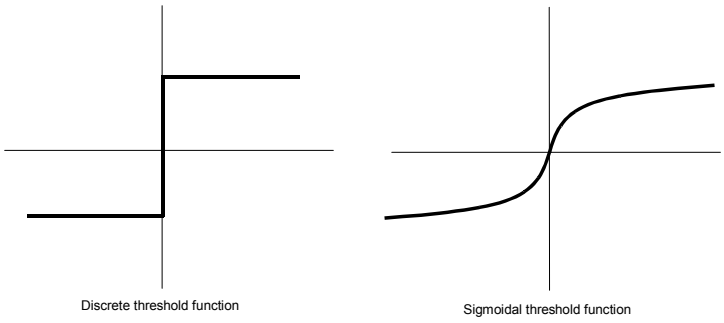
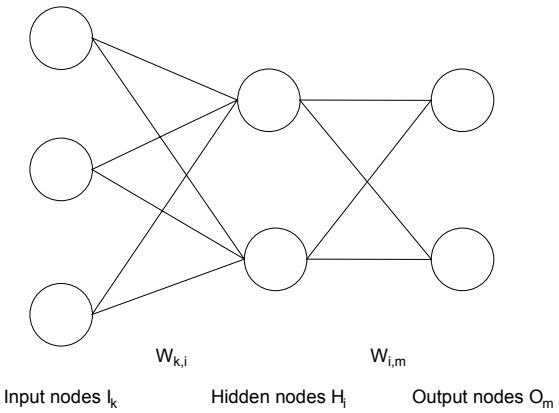


Figure 4-7 Discrete versus sigmoidal threshold function

An ANN learns from a set of instances containing input values and corresponding output values; the network learns by adjusting the weights on the connections between the nodes. One of the most widely used types of

ANN is the backpropagation network. A backpropagation ANN (see *Figure 4-8*) contains one input layer that represents the input values (I_k), one output layer that represents the output values (O_m) and one (or more) hidden layers that represent more complex relationships between the input and output nodes (H_i). The weights of the connections between the input and hidden nodes ($W_{k,I}$) and between the hidden en output nodes ($W_{i,m}$) reflect the strength of the connections between the nodes. These weights are updated when learning from the training set of instances using the backpropagation learning algorithm (Mitchell, 1997).

Figure 4-8
Backpropagation artificial neural network



As an ANN stores knowledge in the weights of the connections between the nodes, ANN are model-based learners. Learning an backpropagation ANN requires instances from the training set to be processed multiple times by the network (Russell & Norvig, 1995); this makes the time to learn an ANN long; hence, backpropagation ANN are not suited for dynamic updates, only for static usage or staged updates.

Backpropagation ANN can both be used for problems in which training data is noisy and complex (such as inputs from cameras and microphones) and to problems with more symbolic representations, in which decision tree learning is also often used (Mitchell, 1997).

4.4.1 Backpropagation neural networks in prediction strategies

When employing a backpropagation ANN in a prediction strategy, several issues have to be dealt with. The first step is to determine the inputs and output of the network. The inputs of the ANN are the validity indicators, which have to be encoded in real numbers in a fixed range (Russell & Norvig, 1995). Just like CBR, ANN can also be used in two ways as a prediction strategy decision approach: one integrated network containing all predictors or one network per predictor.

When an integrated network is used, all validity indicators for all predictors are used as input nodes while each output node represents one predictor (see *Figure 4-9*). For each prediction request, the values of the validity indicators are fed into the network, which activates the input nodes, hidden nodes and results in the activation of that output node that represents the predictor that is best suited to provide the prediction. Which output node is activated depends on the learned weights that describe the relationship between the validity indicators and the output nodes via the hidden nodes. Informing the network which output node should have been activated enables the learning process that updates the weights.

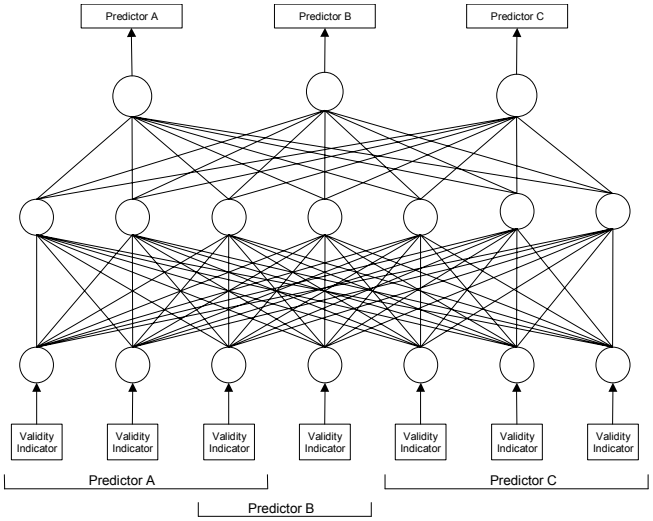


Figure 4-9 One artificial neural network as a prediction strategy decision approach

When using multiple networks in a prediction strategy decision approach, one network for each predictor, the input nodes of each network are the validity indicators of that predictor and the output node represents the expected error of that predictor (see *Figure 4-10*). For each prediction request, all the networks of all the predictors are fed with the values of their validity indicators and each network calculates the expected error for its predictor. The prediction strategy then chooses that predictor with the lowest expected prediction error.

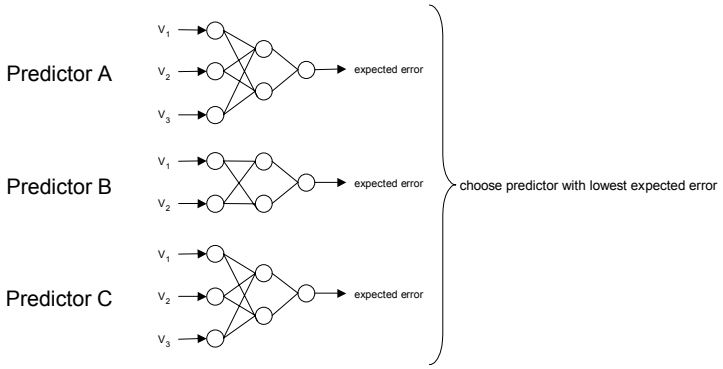


Figure 4-10 Multiple artificial neural networks as a prediction strategy decision approach

These two ways to use ANN are very similar to the two ways to use CBR. The same advantages and drawbacks that have been mentioned for CBR also apply to these two approaches for ANN. The main difference between CBR as a prediction strategy decision approach and ANN as a prediction strategy decision approach is that CBR uses the raw instances to reason, while ANN create an intermediate model from the instances and then use that model for reasoning.

4.5 Bayesian probability

"Probability provides a way of summarizing the uncertainty that comes from our laziness and ignorance" (Russell & Norvig, 1995); laziness in the sense that it is often not possible to have a complete set describing the full knowledge necessary to make a decision; ignorance either because one does not have a complete theory for a domain or it may not be possible to acquire all data necessary for making a decision. Probability describes the chance that a certain event occurs; e.g. 40% chance that is will start to rain today. A probability of 0 represents the belief that something is false, while a probability of 1 (or 100%) represents the belief that something is true.

Probabilities depend on evidence found that support a certain probability calculation. Probabilities without any evidence are called prior or unconditional probabilities; e.g. without any evidence, the prior probability that it will start to rain is 40%; written as P(rain) = 0.4. Probabilities based on evidence are called posterior or conditional probabilities. "All probability statements must [..] indicate the evidence with respect to which the probability is being assessed" (Russell & Norvig, 1995); e.g. the posterior probability that it will start to rain given the fact that the sky is filled with dark clouds is 85%; P(rain | sky filled with dark clouds) = 0.85.

Using Bayes' rule, one can calculate the posterior probability using the prior probability of both the evidence P(A) and the event P(B) and the posterior probability of the evidence occurring given the event P(A|B). Bayes' rule states that (Russell & Norvig, 1995; Mitchell, 1997):

$$P(B \mid A) = \frac{P(A \mid B)P(B)}{P(A)}$$

It is often easier to directly calculate or estimate the posterior probability of evidence given the occurrence of an event from a dataset than to calculate or estimate the posterior probability of an event given the evidence; e.g. based on a historical set of data containing weather situations, one can easily calculate in how many instances there was a sky filled with dark clouds when it started to rain.

Bayesian probabilities can be used either as a model in prediction strategy decision approaches or by using the instances. When used as a model, all probabilities are either specified by experts or calculated offline and these probabilities are used when a decision has to be made. It is also possible to calculate the probabilities on the fly from a set of data, resulting in an instance-based decision approach.

When a model of probabilities is used, this model can either be kept static or be updated in stages; when probabilities are calculated on the fly from the available set of instances, updates are dynamic if that set of instances is extended when new instances are discovered during usage of the system.

4.5.1 Bayesian probability in prediction strategies

When using probabilities in prediction strategies, one determines which predictor has the highest chance of providing the best prediction; calculating the probability that a certain predictor should be used given the set of validity indicators of that predictor; e.g. P(predictor | V), where $V = \{ v, v_2, \ldots, v_n \}$. When choosing the predictor that has the highest chance of providing the best prediction, Bayes' rule can be used to calculate the posterior probability of each predictor being the best predictor for the given prediction request.

A disadvantage of using probabilities is that a decision about whether a specific predictor should be used cannot be made independently of other predictors. Where calculating the expected error of a predictor using CBR or an ANN can be solely based on knowledge learned about that specific predictor, probability calculations need to take into account the relative probability of a predictor compared to other predictors; e.g. the prior probability of using a predictor, P(predictor X), is calculated as the number

of times predictor X was the best performing predictor over all other predictors (best meaning the lowest absolute error). This means that a change in a predictor or the set of predictors means that all previously calculated probabilities become invalid; a new or changed predictor, no matter how accurate, will be in a disadvantage compared to the other unchanged predictors as they have already been used before and thus a higher prior probability.

Example 3 Calculating posterior probabilities

V_1	V_2	V_3	Predictor
Yes	5	A	X
Yes	25	B	X
No	10	A	Z
No	10	A	Z
Yes	5	B	Z
Yes	5	C	Z
No	10	C	Y
No	25	B	Y
No	25	A	Z
No	5	A	X

Prior Probabilities:

$P(V_1=Yes) = 4/10 = 0.4$ $P(V_1=No) = 6/10 = 0.6$

$P(V_2=5) = 4/10 = 0.4$ $P(V_2=10) = 3/10 = 0.3$ $P(V_2=25) = 3/10 = 0.3$

$P(V_3=A) = 5/10 = 0.5$ $P(V_3=B) = 3/10 = 0.3$ $P(V_3=C) = 2/10 = 0.2$

$P(X) = 3/10 = 0.3$ $P(Y) = 2/10 = 0.2$ $P(Z) = 5/10 = 0.5$

Calculating the posterior probability of each predictor if one only knows $V_1=Yes$:

$P(X|V_1=Yes) = P(V_1=Yes|X)P(X) / P(V_1=Yes) = 2/3 * 0.3 / 0.4 = 0.5$

$P(Y|V_1=Yes) = P(V_1=Yes|Y)P(Y) / P(V_1=Yes) = 0/2 * 0.2 / 0.4 = 0$

$P(Z|V_1=Yes) = P(V_1=Yes|Z)P(Z) / P(V_1=Yes) = 2/5 * 0.5 / 0.4 = 0.5$

Calculating the posterior probability of each predictor if one knows $V_1=Yes$, $V_2=10$, $V_3=A$ (Notice that this specific instance has not occurred before):

$P(X|V_1=Yes \wedge V_2=10 \wedge V_3=A) = P(X)P(V_1=Yes|X)P(V_2=10|X)P(V_3=A|X) =$
$= 0.3 * 2/3 * 0/3 * 2/3 = 0$

$P(Y|V_1=Yes \wedge V_2=10 \wedge V_3=A) = P(Y)P(V_1=Yes|Y)P(V_2=10|Y)P(V_3=A|Y) =$
$= 0.2 * 0/2* 1/2 * 0/2 = 0$

$P(Z|V_1=Yes \wedge V_2=10 \wedge V_3=A) = P(Z)P(V_1=Yes|Z)P(V_2=10|Z)P(V_3=A|Z) =$
$= 0.5 * 2/5 * 2/5 * 3/5 = 0.048$

Example 3 shows how to calculate the posterior probability of each predictor based on a small set of instances given the value of one validity indicator and given the values of all three validity indicators where that

combination has never occurred before (using the conditional independence assumption (Russell & Norvig, 1995)). In situations where the combination of validity indicator values already occurred before, one can directly calculate the probability for each predictor in the subset of those instances with that combination of validity indicator values.

4.6 Decision approaches used for validation

In order to validate the Duine prediction framework, it is not necessary to examine every possible prediction strategy decision approach in the validation experiments. When one tries to optimize a prediction engine for a specific domain, one may want to compare the various available prediction strategy decision approaches in order to choose the best performing approach in that specific domain. However, for validation of the framework, this is not necessary. For this reason, only two decision approaches have been examined in the validation experiments: manually created decision rules and CBR.

These two approaches have been chosen as they represent two extreme decision approaches: manual static model-based versus automated dynamic instance-based. Decisions made by these approaches are also easily interpretable by humans, making it possible to learn from them in order to better understand the relationships between predictors.

4.7 Conclusions

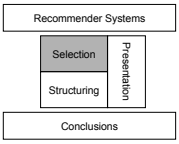
In this chapter, an important design choice for designing prediction engines has been investigated: the choice of which prediction strategy decision approach to use in prediction strategies that determine which predictor to use for a certain prediction request. Four types of machine learning techniques, decision trees and decision rules, case-based reasoning, backpropagation artificial neural networks and Bayesian probabilities, have been discussed that can be used to make this decision. The differences between these machine learning techniques have been discussed including ways to apply each of the machine learning techniques as a prediction strategy decision approach.

In order to determine if prediction engines created with the Duine prediction framework can indeed help people in finding interesting items, prediction engines have been developed for two different systems: a movie recommender system and a personalized electronic TV program guide. These two systems and the results of experiments with the prediction engines of these two systems are discussed in the next chapter. In these

experiments, two prediction strategy decision approaches have been used and compared: manually created decision rules and case-based reasoning. The results of how well each prediction strategy decision approach performs and the comparison of the two approaches are also discussed in the next chapter.

Chapter 5

Validating the Prediction Framework



To determine if the Duine prediction framework as described in chapter 3 can indeed be used to develop prediction engines that can help people in finding interesting items the framework has to be validated. Predicting how interested a user is in an item is part of the selection process of personalized information systems. Validation means to demonstrate that it is possible to create prediction engines based on this domain-independent framework, where the prediction engines are tailored to the specific domains in which they must function. The predictions made by these prediction engines have to be better than any individual prediction technique as the main idea behind the framework is to use prediction strategies that switch between multiple prediction techniques in order to improve the accuracy of the predictions.

In the previous two chapters, the Duine prediction framework has been described and various decision approaches to implement prediction strategies have been introduced. This chapter provides the validation of the Duine prediction framework by providing the results of experiments with two personalized information systems: a movie recommender system and personalized electronic TV guide. In these experiments, two distinct prediction strategy decision approaches are used: decision rules and case-based reasoning (CBR); the first being a model-based approach where the model is manually created by experts, the second being an instance-based approach that learns automatically.

First, a way to measure the performance of prediction engines is described in section 5.1. The set up of the experiments, including a description of the two datasets used, is provided in section 5.2. In section 5.3, the results of the experiments using a rule-based prediction strategy are analyzed and discussed, while in section 5.4 the results of the CBR-based prediction strategy are examined. The results of the two prediction strategy

decision approaches are compared in section 5.5. This chapter ends with conclusions concerning the Duine prediction framework in section 5.6.

Parts of this chapter have already been published in van Setten et al. (2004) van Setten et al. (2003) and van Setten et al. (2002).

5.1 Validation measures

Experimentally validating the Duine prediction framework requires instruments to measure and compare the performance of prediction engines. In order to help users with finding interesting items, the most important aspect of the performance construct is *prediction accuracy*, which has already been introduced in section 2.1. The more accurate predictions are, the better the recommender knows the user, and the better the recommender can help people in finding interesting items.

Another aspect of performance is *prediction stability*, which refers to the consistency of a recommender in the accuracy of its predictions; the level of variation in prediction accuracy. Users cannot rely on predictions of a recommender in which prediction accuracy varies a lot. Of two recommenders with equal prediction accuracy, the most stable recommender is preferred as users can better rely on that recommender. However, prediction accuracy is more important than stability; a recommender with very stable but inaccurate predictions provides a worse user experience than a recommender with more accurate but instable predictions. For this reason, the focus of this chapter is on prediction accuracy.

Prediction speed is another aspect of the performance of recommender systems. Prediction speed is the time it takes for a recommender system to generate a prediction for an item for a user. Although important, without prediction quality and stability even the fastest recommender system cannot help people in finding interesting items. As the focus of this thesis is on helping people to find interesting items, prediction speed (including scalability) is not examined in more detail.

Herlocker et al. (2004) suggest that a fourth performance aspect may also be important, namely the *explanatory power* of a recommender system: how well can a recommender system explain the reasons behind a prediction to its user? As explaining predictions is still under investigation in other research projects, explanatory power is not tackled in this chapter, although it will be briefly addressed in chapter 7 as part of the presentation of predictions.

The focus in validating the Duine prediction framework is on prediction accuracy.

5.1.1 Measuring prediction accuracy

Herlocker et al. (2004) provide a thorough investigation of evaluation metrics for various types of recommender systems. As the Duine prediction framework focuses on providing predictions for each item that can be displayed to a user, referred to as annotation in context by Herlocker et al., prediction accuracy is the most important metric. Measuring prediction accuracy is "necessarily limited to a metric that computes the difference between the predicted rating and true rating" (Herlocker et al., 2004).

Another group of accuracy measures is based on ranking measures, which are applicable to recommender systems that return a limited list of suggested items instead of a prediction for each item. For such measures, the order of items within the list becomes important and the location of errors within the list; e.g. an error higher in the list is more important than an error at the bottom of the list. See Herlocker et al. (2004) for more details on such ranking measures.

Mean absolute error

Mean absolute error is the most used metric to measure prediction accuracy of prediction engines (Breese et al., 1998; Herlocker et al., 2002; Shardanand & Maes, 1995).

Mean absolute error, abbreviated to *mae,* over a set of n given ratings is measured by the average absolute deviation between the predicted ratings p_i and the given ratings r_i. The lower the mean absolute error, the more accurate a prediction engine is.

$$mae = \frac{\sum_{i=1}^{n} |p_i - r_i|}{n}$$

According to Herlocker et al., one of the benefits of mean absolute error is that it has well defined statistical properties that make it possible to test the significance of differences in mean absolute errors of two systems, e.g. using a paired samples t-test.

Herlocker at al. only investigated the accuracy of different variants of the same type of prediction technique (collaborative filtering). When investigating the accuracy of different types of predictors, it is also necessary to take into account whether a predictor is capable of providing a prediction for certain prediction requests or not. Mean absolute error can only be calculated for items that have both a given rating and a predicted rating. One metric to measure the capability of a predictor to generate predictions is the coverage metric (Herlocker, 2000). Coverage measures the percentage of items for which a prediction could be generated; i.e. the

number of items for which a prediction could be generated divided by the total number of items for which a prediction was requested.

However, in recommender systems that provide annotation in context via predictions it is necessary for users to receive an accurate prediction with every item that can be provided to them. This is even stricter for recommender systems that provide time sensitive information, such as recommender systems for electronic TV program guides (EPG) and auctioning sites. In time sensitive systems, items are only available for a limited amount of time or only at a specific time. If a recommender cannot generate a prediction for all items, it means that a user might miss interesting items. For this reason, a new metric for prediction accuracy has been created that is based on both mean absolute error and coverage: the global mean absolute error (*gmae*); this metric is always capable of calculating prediction accuracy, therefore it is called global.

Global mean absolute error is the same as the mean absolute error, except that when a predictor cannot generate a predicted rating the neutral rating (zero) is assumed. Without additional information, a user can only interpret an item without a predicted rating as being neither interesting nor not-interesting; i.e. neutral. Using a neutral value as a default rating has also been applied in the evaluation of a hybrid case-based reasoning and collaborative filtering recommender system (Burke, 2000a).

As both prediction techniques and prediction strategies are predictors, prediction accuracy can be measured for both by *gmae*. The lower the *gmae* is for a prediction technique, the better the prediction technique is in predicting; the lower the *gmae* is for a prediction strategy, the better that prediction strategy in determining which predictor to choose for generating a prediction.

Problems of mean absolute error metrics

McLaughlin & Herlocker (2004) argue that mean absolute error metrics conceal flaws in prediction techniques that consequently may hurt the user experience due to following characteristics of mean absolute error metrics:

1. During offline analysis, if there is no rating available for an item, it is impossible to use that item in the analysis; hence it has no effect on calculating *(g)mae*.
2. An error for a high rating has the same impact as an error of the same size for a low rating.
3. A prediction technique that can very accurately predict items with average or low ratings, but which is unable to accurately predict items with high ratings may show a good accuracy, while being unable to provide good recommendations to its user.

4. Mean absolute error cannot be used to measure the accuracy of prediction techniques that do not produce a predicted rating per item, but only result in a list of recommended items.

The first characteristic not only applies to mean absolute error but to any metric that is used for offline analysis. If no opinion of a user is known for an item, no metric can use that item in its evaluation, unless it makes assumptions like such an item has a neutral or negative rating. McLaughlin and Herlocker take the latter position when evaluating collaborative filtering algorithms using the precision metric. Any item that does not have a rating is assumed to be not relevant. They are aware that this results in a worst-case precision value instead of an accurate precision value. The only solution for this problem seems to be to ask users to rate every item in a dataset, which is practically impossible as it also means that those users have to know every item in the dataset. In the experiments described in this chapter, this issue has been partly addressed by including an experiment, the EPG experiment, in which all users were asked to rate every TV program they have an opinion about instead of only rating those TV programs they have actually watched.

The second and third characteristics of mean absolute error are indeed a problem in situations were a recommender systems tries to offer its user a top-N list of recommended items ("Find good items" task (Herlocker et al., 2004)). However, in systems where every item needs to get a predicted rating in order to be presented to a user ("Annotation in context" task (Herlocker et al., 2004)), these two characteristics are not a problem, but a necessity for a good accuracy metric as every item needs to get a predicted rating.

As the Duine prediction framework focuses on recommender systems that do provide predicted ratings per item, the last characteristics of mean absolute error metrics is not relevant for this research.

Natural upper bound

In order to determine how well a prediction strategy is performing on its own (not in comparison to other prediction strategies), it is important to know the natural upper bound (Bennett et al., 2002) of the accuracy measures. This natural upper bound defines the best possible prediction accuracy that can be achieved for a given dataset. For the coverage measure, the natural upper bound is also the theoretical upper bound of 100%, which means that always a predictor is chosen by the prediction strategy decision approach.

For the mean absolute error measures, the theoretical upper bound is zero, although practically this will be somewhat above zero as a predictor can "never be more accurate than the variance in a user's rating for the

same item" (Herlocker et al., 2004). A study by Hill et al. (1995) has shown that when users rate the same items twice at different occasions, there will be a difference between the ratings; it will be virtually impossible to ever achieve a perfect prediction engine with a *(g)mae* of zero.

However in prediction strategies, the natural upper bound is not always the same as the theoretical upper bound. For some prediction requests, none of the available predictors may be able to provide the correct prediction, which means that the theoretical upper bound of no error for that specific request cannot be achieved by the prediction strategy. To determine the natural upper bound for the *gmae* measure in a specific dataset, one needs to determine per prediction request what the best predictor would have been, which is the predictor that had the lowest absolute error. The natural upper bound for (a subset of) the dataset is the average for all prediction requests of this minimum possible error per prediction.

Rounding

Absolute error metrics take the absolute difference between the given rating of the user and the predicted rating; however, users generally rate items on an ordinal scale (e.g. 1,2,3,4,5), while predicted ratings are generally on a interval scale; e.g. a predicted rating of 2.34. In most studies that use mean absolute error metrics, the difference between these two scales is not taken into account. However, predicted ratings are often also presented to the user using the ordinal scale, e.g. as a number of stars; in that case, the predicted rating is rounded to the nearest value on the ordinal scale. Rounding the predicted ratings resembles the experience of the user more closely.

In order for results to be compatible with other publications, results are first presented without rounding predicted ratings to the feedback scale; then the influence of rounding the predictions to the feedback scale is examined.

Correctness

Within a set of predictors, accuracy can also be evaluated by determining the correctness of each predictor, which indicates in how many instances a predictor was the most accurate predictor; i.e. the number of times the predicted rating p_x of predictor x had the lowest absolute error *ae* of all available predictors for that prediction request relative to the total number of prediction requests:

$$correctness_x = \frac{\#(p_x = \min(ae))}{\#\,prediction\,requests} \cdot 100\%$$

A predictor with 60% correctness means that that predictor was the most accurate predictor in 60% of the prediction requests. The higher the correctness percentage, the more often that predictor results in the most accurate prediction[2]. Notice that the sum of correctness of all predictors can be larger than 100% as multiple predictors can provide the most accurate prediction for a specific prediction request.

5.1.2 Measuring subjectively

Due to the problems of offline analysis as discussed in section 5.1.1, measuring accuracy objectively cannot fully reflect how good a recommender system performs in the eyes of its users. For this reason, one may also want to measure the performance of a recommender system as it is perceived by its users by asking the user's opinion about the recommender system.

As this type of measurement requires direct input of users, which can only be acquired after users have used a recommender system for some time, it cannot be part of any dataset. Any changes to the prediction engine can result in a differently perceived performance; this requires re-measuring the user's opinion. A way to measure accuracy subjectively would be to set up a between-subjects experiment where groups of users are offered different prediction engines during the same time period using the same set of content; comparing the opinions of users concerning accuracy between the different groups shows which prediction engine is perceived as most accurate.

Experimenting with various prediction strategies, including tuning, requires the use of the same set of content and the same user's ratings for that content so that accuracy can be objectively compared; this cannot be accomplished in an online recommender system; hence the focus of this chapter lies only on objectively measuring accuracy using offline analysis in two datasets.

5.2 Experimental setup

The hypothesis of the Duine prediction framework is that switching between several predictors increases the prediction accuracy of prediction engines. To validate the Duine prediction framework, this hypothesis has to be validated. Validating this hypothesis also shows that it is possible to create prediction engines based on the Duine prediction framework; i.e. prediction engines that use prediction techniques and prediction strategies

[2] Correctness can be used as a prior probability in Bayesian probability calculations.

that are both based on the generic predictor model, where prediction strategies make decisions about which predictor to use via the validity indicators of the predictors.

Several decision approaches can be used for prediction strategies (see chapter 4). Two of these approaches are examined in this thesis:

1. Manually created decision rules.
2. Case-based reasoning (CBR).

These two approaches have been chosen, as they are very different; manually created decision rules represent a situation in which expert knowledge is used, whereas case-based reasoning is an automatic learning technique. Furthermore, decision rules are expressed in a model, whereas case-based reasoning uses instances to base its decision on.

These decision approaches are first examined individually by discussing the various design alternatives for each approach, the chosen design, the results of the validation experiment and conclusions concerning the suitability of that approach as a prediction strategy decision approach. Secondly, the decision approaches are compared to each other to determine what decision approach is the most suitable to use in prediction strategies.

Before the experiments and results are discussed, datasets that have been used in the experiments are described.

5.2.1 Datasets

A dataset for measuring the accuracy of prediction engines requires at least the following data:

- A set of items, where each item is described by at least an item identifier.
- A group of users, where each user is described by at least a user identifier.
- A set of ratings from the users for the items using a triple of user identifier, item identifier and given rating; preferably extended with a timestamp or an indication of ordering in time.

Depending on the dataset, items and users can be described in more detail; the minimal, a set using only identifiers and given ratings, only allows for social-based prediction techniques to be used; when items are described in more detail, also information-based prediction techniques can be used. Two methods can be distinguished for the acquisition of datasets:

1. *Implicit acquisition*: data is gathered from users who use a recommender system for its recommendations. During usage, users can either explicitly provide their opinions about items (explicit feedback) or the

recommender system monitors their behaviour to implicitly determine a user's opinion (implicit feedback).
2. *Explicit acquisition*: data is gathered by asking people to go through a set of items and to rate those items for which they have an opinion. In this method, users do not receive any recommendations; they only give their opinions on items. Since there is no real usage involved, only explicit feedback can be acquired.

Both methods have advantages and disadvantages. The main drawback of explicit acquisition is that it does not represent a normal usage situation. On the other hand, it avoids one of the problems of implicit acquisition: people tend to only give feedback on incorrect predictions; imagine the perfect prediction engine, one where every prediction reflects the actual interest of the user. In such a system, users no longer need to give feedback, as predictions are already correct. From the user's point of view, this is an ideal situation, but not for research purposes, as there is no way to quantifiably validate the accuracy of such a prediction engine. However, the same problem occurs in less perfect, but more realistic, prediction engines; more realistic as it is very improbable that any recommender system will be able to predict the interests of users perfectly as even users themselves are incapable of fully describing their own interests; people even rate the same items differently at different times (Herlocker et al., 2004). In such systems, there is no way of knowing whether a prediction for which no feedback has been received was either correct, whether the user has not seen the information or whether he has no opinion about it (McLaughlin & Herlocker, 2004). By asking people to rate a list of items, there is either a large probability that the user has no opinion about those items for which no rating is available or that he has already rated that same item multiple times.

Explicit acquisition also partly addresses the first problem of using mean absolute error metrics to measure the accuracy of a prediction engine; during offline analysis, if there is no rating available for an item, it is impossible to use that item in the analysis; hence it has no effect on calculating *(g)mae*. When a dataset is acquired explicitly, a user has examined all items and given a rating for those items he has an opinion about, not only on those items that happened to be recommended by the original prediction engine of the live recommender system. This way, more ratings are acquired per user than via implicit acquisition.

The use of the Duine prediction framework has been examined in two systems; one is based on implicit acquisition, the other on explicit acquistion; the first is a system that recommends movies (MovieLens), the other a system that recommends TV programs (EPG dataset).

MovieLens

MovieLens is a movie recommender system (http://movielens.umn.edu) developed by the GroupLens group at the University of Minnesota (http://www.grouplens.org). MovieLens recommends movies to its users using collaborative filtering. The researchers at MovieLens have made two datasets publicly available. The first dataset consists of 100,000 ratings by 943 users for 1,682 movies gathered during 7 months in 1997 and 1998; the second dataset consists of 1,000,209 ratings by 6,040 users for 3,900 movies gathered from all users who joined the system in 2000 (during 34 months). For this experiment, the first dataset is used as it has been used by many other research projects into recommender systems, allowing for our results to be compared if required.

In MovieLens, data has been gathered via real usage of the system, i.e. implicit acquisition, where users explicitly rated movies that were recommended to them.

EPG

The electronic program guide (EPG) system has been developed specifically for this research, both to demonstrate the possibilities of recommender systems and in particular a recommender system developed using the Duine prediction framework and to gather data for validation purposes.

For this dataset, 24 people have been asked to rate four weeks of TV programs from Dutch television (broadcasted between 15 August 2002 and 14 September 2002); i.e. explicit acquisition. These four weeks contained 40,539 broadcasts from 47 different channels of which 12,445 broadcasts actually received at least one rating. Participants only rated those programs they had an opinion about, resulting in a total of 31,368 ratings. The four weeks include a transition from the summer TV season to the winter TV season at September 1st. This transition has been included deliberately as it helps to show that prediction strategies are capable of dealing with large changes in the information source.

Table 5-3 Comparing the two datasets

	MovieLens	EPG
Domain	Movies	TV programs
Data acquisition method	Implicit	Explicit
Number of users	943	24
Number of items	1,682	12,445 of 40,539
Number of ratings	100,000	31,368
Sparsity[3]	93.7%	89.5%
Time period	7 months	4 weeks

The use of implicit feedback has not been examined in these experiments for two reasons: the MovieLens dataset has been acquired by the University of Minnesota, which meant we had no control over the data acquisition method; for the EPG dataset, acquiring implicit feedback required specialized equipment to monitor what users were watching on TV; this hardware was not available and too expensive to build and distribute over multiple users.

5.2.2 Prediction techniques

In the MovieLens dataset and the EPG dataset the same basic prediction techniques have been used; although tuned to the specific domain, respectively the movie and TV domain. The used prediction techniques are:

- AlreadyRated
- UserAverage
- TopNDeviation
- Collaborative Filtering
- Case-based Reasoning
- GenreLMS
- SubGenreLMS
- Information Filtering

AlreadyRated

By definition AlreadyRated is not a predictor; it does not return a predicted rating of a user in an item; it only returns a value if a user has already rated that item: the previously given rating by that user which is not a prediction. However, AlreadyRated should be used in prediction strategies as users expect to see the given rating of an item the next time that item is retrieved, instead of a prediction that may have a different value than the given rating.

[3] Sparsity shows how much ratings are in a dataset compared to how much ratings could have been in the dataset and is calculated by 1-(ratings/(users*items))

Accuracy measurements for AlreadyRated have no meaning compared to accuracy measurements for real predictors as users tend not to re-rate items if that rating is still correct; any measured accuracy for AlreadyRated is mostly based on ratings for items that were wrongly rated, artificially decreasing the accuracy for AlreadyRated.

AlreadyRated has one validity indicator: *Known*, which returns true if a rating is known for an item and user, otherwise it returns false.

UserAverage

The prediction technique UserAverage returns the mean of all ratings provided by a user; i.e. the predicted value $p_{a,x}$ for user a for item x is calculated by the sum of all given ratings r of user a divided by the number of given ratings; the average rating is also often denoted by $\bar{r}$. This average represents the subjective neutral rating of a user:

$$p_{a,x} = \overline{r_a} = \frac{\sum_{i=1}^{n} r_{a,i}}{n}$$

UserAverage has one validity indicator, namely the number of items that have already been rated by the user: *NumberOfRatedItemsByUser*. When no items have been rated, UserAverage returns the prediction value of zero; the more items that have been rated, the better the predicted value represents the subjective neutral rating of the user.

TopNDeviation

TopNDeviation returns a prediction based on all n ratings from other users that already rated the item for which a prediction must be made. The exact algorithm used is the deviation-from-mean average over all users as described in (Herlocker, 2000). It takes the average rating of the user $\overline{r_a}$ (see UserAverage) as a basis and adds the mean deviation between the given rating for this item $r_{u,x}$ and the average rating $\overline{r_u}$ of all other users u that rated the item:

$$p_{a,x} = \overline{r_a} + \frac{\sum_{u=1}^{n} (r_{u,x} - \overline{r_u})}{n}$$

As TopNDeviation depends on others users that have rated the item, it has a validity indicator *NumberOfUsersThatRatedItem* that returns the number of users that have already rated the item.

Collaborative filtering

Collaborative filtering is based on the idea that people that have rated the same items the same way in the past probably also have similar interests patterns. Based on this knowledge one can predict how much a person likes an unseen item when similar users already rated that item. Collaborative filtering basically consists of two steps; the algorithms used for these two steps have been based on research by Herlocker (Herlocker, 2000).

In the first step, similarity between the current user and other users, who have rated the item for which a prediction is necessary, is calculated based on how the current user and the others have rated the same items in the past. To calculate the similarity between user *a* and user *b*, the Pearson correlation coefficient is used:

$$s_{a,b} = \frac{\sum_{i=1}^{n} \left[\left(r_{a,i} - \bar{r}_a \right) \left(r_{b,i} - \bar{r}_b \right) \right]}{\sqrt{\sum_{i=1}^{n} \left(r_{a,i} - \bar{r}_a \right)^2 \sum_{i=1}^{n} \left(r_{b,i} - \bar{r}_b \right)^2}}$$

Herlocker investigated several correlation measures and found that Pearson correlation resulted in the best prediction accuracy of collaborative filtering.

One optimization for similarity suggested by Herlocker has been incorporated in the collaborative filtering prediction technique. Similarity between two users is adjusted based on the number of rated items they have in common; as long as the two users do not have at least 50 items in common, their similarity value is linearly decreased by multiplying the Pearson correlation with the number of items in common divided by 50.

The second step is to use the similarity and the ratings of those similar users, for the item for which a prediction is necessary, to calculate the prediction for the current user, where σ_i is the standard deviation of the ratings of user *i*:

$$p_{a,x} = \bar{r}_a + \sigma_a \frac{\sum_{i=1}^{n} \left[\left(\frac{r_{i,x} - \bar{r}_i}{\sigma_i} \right) s_{a,i} \right]}{\sum_{i=1}^{n} s_{a,i}}$$

As collaborative filtering depends on the fact that the user has rated enough items, in order to find other similar users, and that there are enough similar users that have already rated the items, the collaborative filtering prediction technique has two validity indicators: *NumberOfRatedItemsByUser* which is the

same as the validity indicator of UserAverage and the *NumberOfSimilarUsers WhoRatedItem,* which indicates how many users have already rated the item and that have rated at least the same 50 items as the current user. However, since the second validity indicator already requires the current user to have rated at least 50 items, otherwise no other users can be found that have rated at least the same 50 items, the first validity indicator is redundant. However, in real-life production systems, the first validity indicator may be necessary as a first check for performance (speed) reasons; calculating how many items the current user has rated is less processing intensive than calculating how many users have rated at least the same 50 items.

Case-based reasoning

Case-Based reasoning (CBR) is based on the idea that if two items are similar and if a rating is known for one of them, the rating for the other will probably be the same. CBR is especially good in predicting how interested a user is in the same types of items, or slightly different versions of the same items. The CBR prediction technique looks at all items that the user has rated in the past and determines how similar they are to the current item. For those items that are similar enough (similarity $s_{i,x}$ equal or larger than threshold t), the old ratings are used to calculate a prediction for the new item by taking the weighted average of those ratings, using the similarity as a weight:

$$p_{a,x} = \forall i \in R_a \rightarrow s_{i,x} \geq t \frac{\sum r_{a,i} s_{i,x}}{\sum s_{i,x}}$$

The actual determination of how similar two items are is domain-dependent; in each domain a function for $s_{x,y}$ needs to be created that returns a value between [0..1] indicating how similar two items are; a similarity of 0 means that items are not similar at all; a similarity of 1 means that items are identical.

CBR has a validity indicator that returns the number of similar items that the user has already rated. As there are multiple levels of item similarity, this validity indicator can be calculated for several similarity thresholds; i.e. the number of similar items that are at least t similar to the current item. Hence, the validity indicator is called *NumberOfAlready RatedSimilarItemsWithSimilarity(t)*, e.g. *NumberOfAlreadyRatedSimilar ItemsWithSimilarity(0.7)*.

GenreLMS

GenreLMS learns how interested a user is in the genres or categories assigned to items (van Setten, 2002). The genre-learning technique calculates a prediction $p_{a,x}$ using a linear function over different genres for each user:

$$p_{a,x} = w_0 + \sum_{i=1}^{n} w_i g_i$$

For each genre i the algorithm has learned a weight w_i indicating the relative importance of each genre to the user, whereas w_0 is a constant value for the user. The extent (percentage) to which an item belongs to genre i is indicated by g_i, with:

$$\sum_{i=1}^{n} g_i = 1$$

If the metadata of an item does not describe to which extent a specific genre belongs to that item, but only that a genre does or does not belong to that item, g_i can be calculated by setting $g_i = 0$ for genres that do not belong to the item and by equally distributing each genre that belongs to the item (with m being the total number of genres that do belong to the item):

$$g_i = \frac{1}{m}$$

This algorithm describes how a prediction is generated when weights for each genre have been learned. Learning itself takes place the moment a user rates an item. The learning algorithm uses the basic least mean square method (Mitchell, 1997, page 11); each weight is updated using the difference between the actual rate r that is provided by the user and the predicted rate p, taking into account how much the genre belongs to the item g_i:

$$w_{i_{new}} = w_i + \mu(r - p)g_i$$

Here μ is a small constant determining the rate in which weights are updated. If this update moderator is chosen too low, it takes a long time before optimal weights are learned, whereas if the moderator is chosen too high, there is a risk of constantly overshooting the optimal weights.

The validity indicator of this technique is *Certainty*, which is the average certainty of each genre in the item as indicated by the user profile, which is a number between 0 and 1. Each time the user rates an item with a specific genre, the certainty of that genre is altered: when the rating is positive and

the weight of that genre in the user profile is positive or when the rating is negative and the weight of that genre in the user profile is negative, certainty is increased by 0.1, otherwise it is decreased with 0.1.

SubGenreLMS

This is the same technique as GenreLMS, using the same validity indicator. The only difference is that GenreLMS works on main genres (e.g. comedy, horror, action), whereas SubGenreLMS works on an alternate set of more finer-grained genres (English comedy, action movie, French Science-Fiction movie).

Information filtering

This prediction technique is also similar to GenreLMS, except that it uses all words (with stop words removed and stemmed) from the item descriptions and their frequency as weights in calculating the prediction and only those words for which an interest is known in the user profile. The validity indicator of this technique is *Certainty*, which is the average certainty for each word in the information item as indicated by the user profile, which is a number between 0 and 1. Learning the interests of the user in the words and the certainty of these interests is done as described for GenreLMS.

5.2.3 Validation process

To compare predictors, the set of ratings in each dataset is divided into different validation sets based on their distribution in time. Set A consists of the first set of ratings (in time), set B of the next set of ratings, etc. The EPG dataset has been divided into four sets, one set per week (with 8867, 8843, 6890 and 6768 ratings respectively), with the transition from summer to winter season between week 2 and 3. As no equivalent to a logical time period such as weeks exists for the movies in the MovieLens dataset, the set of ratings has been divided into sets of 10,000 ratings each. Set A consists of the first 10,000 ratings (in time), set B of the next 10,000 ratings, etcetera; resulting in ten sets for the MovieLens dataset.

When testing each set, the ratings of all previous sets are used for training. This way of validating through time, called x-validation, is preferred over randomly drawn train and test sets as x-validation represents the order in which users actually accessed and rated the items; allowing the validation to more closely represent the user's experience. Furthermore, train/test validation is not possible in the EPG dataset as that can result in having ratings of TV programs in the learn set which are broadcasted after TV programs in the test set for which predictions have to be generated.

For all sets in the EPG and MovieLens dataset, the *gmae* for each predictor has been calculated and paired samples T-tests have been performed to determine if differences between predictors are statistically significant (using a 95% confidence level, meaning that if $p < 0.05$ then differences are statistically significant).

The validation process is carried out using the following steps: The ratings provided by the users are fed to the system one by one, in the logical order of the system; in EPG the logical order is the order in which the programs were broadcasted, in MovieLens the order in which ratings were provided for movies. When a rating is provided during validation, first all predictors are invoked to provide a prediction for the current user and the current item. The absolute difference between the predicted rating calculated by each predictor and the given rating is the prediction error. After the errors for all predictors have been calculated, the given rating is provided as feedback to all predictors and is stored in the user profile. Several predictors use this given rating to learn. This way, when the next rating is processed, the system and all predictors know and have learned from all the previously processed ratings. This process is repeated for all ratings in the specific test set. At the end the *gmae* can be calculated by averaging all absolute errors that fall within each of the sets.

If the *gmae* of a prediction strategy is statistically better than each predictor it uses, it is a good prediction strategy; if the *gmae* of a prediction strategy is statistically equal to the best predictor, it is an adequate prediction strategy; if the *gmae* of a prediction strategy is statistically worse than the best predictor, it is a bad prediction strategy.

5.3 Rule-based prediction strategy

For the design of rule-based prediction strategies, there are no algorithms or functions for which a choice must be made for the implementation except the rules themselves; this is also the drawback of decision rules: expert knowledge is required. However, when designing a rule-based prediction strategy, some experimentation may be required to tune the thresholds of the decision rules. The decision rules used for the MovieLens and EPG datasets are described in the next sections.

5.3.1 MovieLens

Every rule-based prediction strategy has to start with the AlreadyRated prediction technique as users expect to see the rating they gave for an item they have rated in the past. However, this technique will never be used in

the validation experiment with the MovieLens dataset as that dataset does not contain multiple ratings for the same item by the same user.

In MovieLens, the Collaborative Filtering prediction technique is one of the major prediction techniques as this dataset has originally been gathered for experiments with collaborative filtering; research (Herlocker, 2000) and experimentation with the rule set has shown that Collaborative Filtering only works well when at least 20 similar users have already rated the item for which a prediction is required.

Case-based reasoning in MovieLens will not perform well as there is only a limited amount of metadata available about movies in the dataset, making it difficult to create an accurate similarity measure for CBR; only the title and genres are available. For this reason, the thresholds for using CBR have been set quite high; only if there is a lot of evidence that similar items have already been rated by the user, CBR will be used.

Experimentation by Herlocker (2000) has shown that TopNDeviation is good alternative for Collaborative Filtering in the MovieLens dataset; for this reason it is also used in the prediction strategy for MovieLens. Experimentation with the rule set has shown that a threshold of 10 users that have already rated that item provides good results.

The GenreLMS prediction technique is also a good predictor for the MovieLens dataset (van Setten, 2002). Experimentation has shown that with a certainty value of at least 0.30 the GenreLMS prediction technique provides good results within the prediction strategy. Tuning experiments with GenreLMS showed that an update moderator of $\mu = 0.14$ provides the most accurate predictions in the MovieLens dataset.

SubGenreLMS and Information Filtering are not used in the MovieLens prediction strategy, as the metadata of the movies does not contain subgenres and descriptions of movies.

UserAverage is used as the fallback predictor when no other prediction technique is expected to provide an accurate prediction.

The decision rule set for the prediction strategy of MovieLens is summarized using pseudo-code:

```
IF AlreadyRated.Known = true THEN USE AlreadyRated
ELSEIF CollaborativeFiltering.NumberOfSimilarUsersWhoRatedItem ≥ 20
        THEN USE CollaborativeFiltering
ELSEIF CBR.NumberOfAlreadyRatedSimilarItemsWithSimilarity(0.9) ≥ 100 OR
        CBR.NumberOfAlreadyRatedSimilarItemsWithSimilarity (0.7) ≥ 500 OR
        CBR.NumberOfAlreadyRatedSimilarItemsWithSimilarity (0.5) ≥ 1100 THEN USE CBR
ELSEIF TopNDeviation.NumberOfUsersThatRatedItem ≥ 10 THEN USE TopNDeviation
ELSEIF GenreLMS.Certainty ≥ 0.30 THEN USE GenreLMS
ELSE USE UserAverage
```

5.3.2 EPG

Using similar arguments as for the MovieLens prediction strategy, a rule-based prediction strategy has been created for the EPG dataset. As more metadata about TV programs is available in the EPG dataset than metadata about movies in the MovieLens dataset, information-based prediction techniques such as CBR and Information Filtering can be given a more prominent role in the prediction strategy.

As the metadata of TV programs not only contains data about genres, but also about subgenres, a prediction strategy has been created that is used within the main TV strategy, one that switches between GenreLMS and SubGenreLMS or combines their predicted ratings into one prediction, depending on the validity indicator *Certainty* of both predictors. Tuning experiments resulted in update moderators of $\mu = 0.075$ for GenreLMS, $\mu = 0.145$ for SubGenreLMS and $\mu = 1.07$ for Information Filtering in the EPG dataset.

In pseudo-code, the rule-based prediction strategy for the EPG dataset used in the validation experiments is:

TV Strategy

```
IF AlreadyRated.Known = true THEN USE AlreadyRated
ELSEIF CollaborativeFiltering.NumberOfRatedItemsByUser ≥ 25 AND
        CollaborativeFiltering.NumberOfSimilarUsersWhoRatedItem ≥ 20
        THEN USE CollaborativeFiltering
ELSEIF CBR.NumberOfAlreadyRatedSimilarItemsWithSimilarity (0.9) ≥ 1 OR
        CBR.NumberOfAlreadyRatedSimilarItemsWithSimilarity (0.7) ≥ 5 OR
        CBR.NumberOfAlreadyRatedSimilarItemsWithSimilarity (0.5) ≥ 11 THEN USE CBR
ELSEIF InformationFiltering.Certainty > 0.5 THEN USE InformationFiltering
ELSEIF GenreLMS.Certainty ≥ 0.35 OR  SubGenreLMS.Certainty ≥ 0.35 THEN USE GenreStrategy
ELSEIF TopNDeviation.NumberOfUsersThatRatedItem ≥ 10 THEN USE TopNDeviation
ELSE USE UserAverage
```

Genre Strategy

```
IF GenreLMS.Certainty ≥ 0.35 AND
   SubGenreLMS.Certainty ≥ 0.35 THEN
   USE WeightedAverageOf(GenreLMS, SubGenreLMS)
ELSEIF GenreLMS.Certainty ≥ 0.35 THEN USE GenreLMS
ELSEIF SubGenreLMS.Certainty ≥ 0.35 THEN USE SubGenreLMS
```

5.3.3 Results

The results of the experiments with the rule-based prediction strategies in the two datasets are discussed by first providing the non-rounded prediction accuracy results including an analysis of the number of times

each predictor within a strategy has been used followed by an analysis of the correctness percentages of the prediction strategies and its predictors; then the effect of rounding predictions to the ordinal feedback scale is examined. Finally, conclusions concerning the use of manually created decision rules as a prediction strategy are provided.

Prediction accuracy in the EPG dataset

The non-rounded prediction accuracy results in the EPG dataset are displayed in *Table 5-4* (the best prediction technique per set is displayed in italics, the best predictor per set is displayed in bold). These results show that the rule-based prediction strategy provides the most accurate predictions of all predictors in all four weeks and in the overall dataset. The difference with all other predictors is significant in all four weeks and for the overall dataset ($p<0.01$). Overall, the rule-based prediction strategy resulted in an 11.6% increase in prediction accuracy compared to the best prediction technique. The results also show that the natural upper bound is much lower than the prediction accuracy of the rule-based prediction strategy, indicating that there is room for improvement (58% below the best prediction technique).

Table 5-4 Non-rounded prediction accuracy of predictors in the EPG dataset (measured with *gmae*, the lower the value the better)

	Overall	Week 1	Week 2	Week 3	Week 4
AlreadyRated	0.6126	0.6086	0.6084	0.6234	0.6126
CBR	*0.2376*	*0.3530*	*0.1899*	*0.2171*	*0.1693*
GenreLMS	0.3975	0.4105	0.3751	0.4036	0.4035
SubGenreLMS	0.4129	0.4558	0.3982	0.4105	0.3781
WeightedAverageOf(GenreLMS, SubGenreLMS)	0.3870	0.4211	0.3662	0.3850	0.3716
InformationFiltering	0.2919	0.3679	0.2598	0.2769	0.2495
CollaborativeFiltering	0.4820	0.4850	0.4764	0.4852	0.4822
UserAverage	0.5398	0.5301	0.5294	0.5512	0.5544
TopNDeviation	0.5241	0.5199	0.5202	0.5286	0.5300
Rule-based prediction strategy	**0.2100**	**0.2935**	**0.1774**	**0.1933**	**0.1601**
Accuracy increase	11.6%	16.9%	6.6%	11.0%	5.4%
Natural upper bound	0.0997	0.0997	0.0572	0.0649	0.0525

Table 5-5 Number of times predictors are used by the rule-based prediction strategy in the EPG dataset

	Overall	Week 1	Week 2	Week 3	Week 4
AlreadyRated	1212 (3.86%)	505 (5.70%)	303 (3.43%)	242 (3.51%)	162 (2.39%)
CBR	22348 (71.24%)	4471 (50.42%)	6920 (78.25%)	5263 (76.39%)	5694 (84.13%)
GenreLMS	3850 (12.27%)	2467 (27.82%)	675 (7.63%)	456 (6.62%)	252 (3.72%)
SubGenreLMS	73 (0.23%)	52 (0.59%)	13 (0.15%)	4 (0.06%)	4 (0.06%)
WeightedAverageOf(GenreLMS, SubGenreLMS)	913 (2.91%)	437 (4.93%)	241 (2.73%)	132 (1.92%)	103 (1.52%)
InformationFiltering	2703 (8.62%)	767 (8.65%)	661 (7.47%)	746 (10.83%)	529 (7.82%)
CollaborativeFiltering	0 (0.00%)	0 (0.00%)	0 (0.00%)	0 (0.00%)	0 (0.00%)
UserAverage	258 (0.82%)	161 (1.82%)	29 (0.33%)	44 (0.64%)	24 (0.35%)
TopNDeviation	11 (0.08%)	7 (0.04%)	1 (0.01%)	3 (0.04%)	0 (0.00%)
Total	31368 (100%)	8867 (100%)	8843 (100%)	6890 (100%)	6768 (100%)

When examining how often the various prediction techniques have been used by the rule-based prediction strategy (see *Table 5-5*), it shows that, as one would expect, the best prediction techniques with the lowest *gmae*, such as CBR, GenreLMS and Information Filtering have been used most often by the prediction strategy. It also shows that Collaborative Filtering and TopNDeviation have never or almost never been used; this was to be expected: as there are only 24 users in this dataset, the probability that the validity indicator thresholds for Collaborative Filtering and TopNDeviation are exceeded is small.

The results also show that the increase in prediction accuracy of the rule-based prediction strategy is larger in the first and third week compared to the second and fourth week. In the first week, users are new to the recommender; a lot of prediction techniques still have to learn from the users. In this week, the best prediction technique CBR has too little TV programs in its case-base to know what TV programs users are interested in; the rule-based prediction strategy relies on other prediction techniques until CBR has enough knowledge about the user's interests to provide accurate predictions. A similar situation occurs in the third week; instead of new users, the transition of the summer TV season to the winter TV season takes place, which means that a lot of new TV programs are introduced; for several prediction techniques, such as CBR, this means that it has to learn anew what users think about these programs. The rule-based prediction

strategy is again capable of relying on other prediction techniques, such as Information Filtering, to provide accurate predictions.

Furthermore, the results shows that there may be room for optimization in for example the GenreStrategy. In the current rule-based prediction strategy, GenreLMS is used more often than the weighted average of GenreLMS and SubGenreLMS, while the latter provides more accurate predictions in weeks 2, 3 and 4. However, further optimization of the rule-based prediction strategy is not discussed; this prediction strategy already shows that rule-based prediction strategies can improve the prediction accuracy compared to prediction techniques in the EPG dataset.

The results also show that although the prediction accuracy of the rule-based prediction strategy decreases with 8.96% from week 2 to week 3 (the transition of the summer TV season to the winter TV season), it does not decrease as much as the best prediction technique CBR, which decreases with 14.32%. The rule-based prediction strategy is more stable than the CBR prediction technique as the rule-based prediction strategy was able to use other predictors instead of CBR at the beginning of week 3; e.g. Information Filtering has been used more often in week 3 than in week 2, while CBR has been used less often in week 3 than in week 2.

The correctness percentages of each predictor in the EPG dataset based on the non-rounded predictions (see *Table 5-6*) show that the rule-based prediction strategy is not the predictor with the highest correctness percentage in each of the four weeks nor for the overall dataset; the CBR prediction technique has the highest correctness percentage (statistically significant, $p<0.01$). However, the prediction accuracy results show that the rule-based prediction strategy is more accurate than CBR. This discrepancy can be attributed to the fact that in those instances that CBR is not the best predictor, e.g. in 46.1% of the predictions in the overall dataset, the *gmae* of those 46.1% is much worse (*gmae*=0.4794) than the *gmae* of the rule-based prediction strategy when it is not the best predictor (49.1% of the instances in the overall dataset with an *gmae* of 0.3915); i.e. in situations when CBR is the best predictor it predicts the interests of a user very well, however, in situations where CBR is not the best predictor it provides bad predictions; the rule-based prediction strategy may have a lower correctness percentage, in situations where it is not the best predictor it predicts more adequately than CBR as a prediction strategy falls back on other predictors.

Table 5-6 Correctness percentages of the non-rounded predictors in the EPG dataset

	Overall	Week 1	Week 2	Week 3	Week 4
AlreadyRated	24.8%	25.8%	24.9%	24.1%	24.0%
CBR	***53.9%***	***47.3%***	***58.0%***	***54.7%***	***56.4%***
GenreLMS	15.5%	20.1%	14.3%	14.1%	12.5%
SubGenreLMS	7.8%	6.3%	7.4%	8.7%	9.5%
WeightedAverageOf(GenreLMS, SubGenreLMS)	9.7%	13.7%	8.0%	8.9%	7.4%
InformationFiltering	18.6%	16.2%	18.9%	19.9%	20.0%
CollaborativeFiltering	11.5%	13.6%	11.2%	11.0%	9.4%
UserAverage	4.6%	6.8%	4.1%	3.6%	3.1%
TopNDeviation	20.8%	22.2%	20.2%	21.1%	19.2%
Rule-based prediction strategy	50.9%	44.5%	53.6%	52.4%	54.3%

Table 5-7 Rounded prediction accuracy of predictors in the EPG dataset (*gmae*)

	Overall	Week 1	Week 2	Week 3	Week 4
AlreadyRated	0.6126	0.6086	0.6084	0.6234	0.6126
CBR	*0.2280*	*0.3458*	*0.1778*	*0.2090*	*0.1588*
GenreLMS	0.3795	0.3944	0.3542	0.3861	0.3862
SubGenreLMS	0.3979	0.4435	0.3817	0.3961	0.3610
WeightedAverageOf(GenreLMS, SubGenreLMS)	0.3673	0.4032	0.3445	0.3650	0.3523
InformationFiltering	0.2728	0.3543	0.2367	0.2567	0.2293
CollaborativeFiltering	0.4770	0.4779	0.4724	0.4803	0.4782
UserAverage	0.5366	0.5249	0.5270	0.5499	0.5508
TopNDeviation	0.5199	0.5135	0.5151	0.5290	0.5254
Rule-based prediction strategy	**0.1972**	**0.2830**	**0.1610**	**0.1819**	**0.1475**
Accuracy increase	13.5%	18.2%	9.4%	13.0%	7.2%
Natural upper bound	0.0606	0.0888	0.0477	0.0572	0.0440

When examining the prediction accuracy of the rule-based prediction strategy when rounded predictions are used (see *Table 5-7*), the same conclusions concerning the accuracy of predictors can be drawn compared to using the non-rounded predictions; of all prediction techniques CBR is the most accurate, while the rule-based prediction strategy is capable of providing more accurate predictions than any of the prediction techniques (statistically significant, $p<0.01$). Most predictors also appear to be more accurate when predictions are rounded than when they are not rounded.

When examining the correctness percentages for the rounded predictions (see *Table 5-8*), the rule-based prediction strategy is more correct than any of the predictors used within the strategy (statistically significant, $p<0.01$). Also the prediction accuracy of CBR when that predictor is not the best predictor is higher (25.8% of the instances with an *gmae* of 0.7371) than the prediction accuracy of the rule-based prediction strategy when the strategy is not the best predictor (23.1% of the instances with an *gmae* of 0.6859). Rounding the predictions to the user's feedback scale amplifies the errors: the errors of incorrect predictions are higher than when predictions are not rounded.

Table 5-8 Correctness percentages of the rounded predictors in the EPG dataset

	Overall	Week 1	Week 2	Week 3	Week 4
AlreadyRated	28.3%	30.8%	28.0%	27.4%	26.6%
CBR	*74.2%*	*62.3%*	*79.2%*	*76.3%*	*81.0%*
GenreLMS	49.7%	52.1%	51.3%	47.9%	46.5%
SubGenreLMS	45.0%	41.7%	45.6%	44.8%	48.4%
WeightedAverageOf(GenreLMS, SubGenreLMS)	49.4%	49.9%	50.5%	48.3%	48.4%
InformationFiltering	62.1%	54.7%	65.4%	64.0%	65.9%
CollaborativeFiltering	34.5%	37.5%	32.8%	34.0%	33.2%
UserAverage	23.5%	29.3%	22.3%	20.8%	20.2%
TopNDeviation	38.0%	40.6%	37.0%	37.2%	36.6%
Rule-based prediction strategy	**76.9%**	**68.1%**	**80.4%**	**78.8%**	**81.9%**

Summarized, in the EPG dataset, the rule-based prediction strategy is a good prediction strategy as it is capable of providing more accurate predictions than the individual prediction techniques, independently of whether predictions are rounded to the user's feedback scale or not.

Prediction accuracy in the MovieLens dataset

The non-rounded prediction accuracy results in the MovieLens dataset are provided in *Table 5-9*. The results show that the rule-based prediction strategy is always the best predictor, both in all ten subsets and in the overall dataset. The difference with all other predictors is significant in almost all subsets and for the overall dataset ($p<0.05$), except in the subsets 60001-70000 and 90001-100000; the difference with GenreLMS in these subsets is not significant; of all prediction techniques, GenreLMS is the best prediction technique in these subsets. Although the increase in prediction accuracy is statistically significant, the overall increase is only 1.5% compared to the best overall prediction technique TopNDeviation.

Table 5-9 Prediction accuracy of non-rounded predictors in the MovieLens dataset (*gmae*)

	Overall	1-10000	10001-20000	20001-30000	30001-40000	40001-50000
CBR	0.4570	0.4527	0.4595	0.4538	0.4421	0.4622
GenreLMS	0.3958	*0.3965*	0.4014	0.3878	0.3845	0.3994
CollaborativeFiltering	0.4112	0.4145	0.4088	0.3994	0.3975	0.4144
UserAverage	0.4221	0.4143	0.4183	0.4100	0.4079	0.4284
TopNDeviation	*0.3919*	0.4128	*0.3912*	*0.3811*	*0.3758*	*0.3906*
Rule based strategy	**0.3859**	**0.3910**	**0.3848**	**0.3754**	**0.3734**	**0.3868**
Accuracy increase	1.5%	1.4%	1.6%	1.5%	0.6%	1.0%
Natural Upper Bound	0.2248	0.2081	0.2218	0.2200	0.2180	0.2315

	50001-60000	60001-70000	70001-80000	80001-90000	90001-100000
CBR	0.4651	0.4732	0.4369	0.4565	0.4678
GenreLMS	0.4064	***0.4026***	0.3850	0.3949	***0.3991***
CollaborativeFiltering	0.4165	0.4348	0.3955	0.4086	0.4223
UserAvgerage	0.4298	0.4471	0.4080	0.4185	0.4386
TopNDeviation	*0.3983*	0.4103	*0.3713*	*0.3855*	0.4026
Rule based strategy	**0.3944**	**0.4026**	**0.3680**	**0.3842**	**0.3980**
Accuracy increase	1.0%	0.0%	0.9%	0.3%	0.3%
Natural Upper Bound	0.2336	0.2326	0.2215	0.2291	0.2315

In the EPG dataset there is one predictor that is very dominant, namely CBR, which has a low *gmae*; i.e. it is a very accurate predictor. However, in the MovieLens dataset most predictors have similar levels of prediction accuracy and the best prediction technique in the subsets differs in the MovieLens datasets: sometimes GenreLMS and sometimes TopNDeviation is the best prediction technique. The most used prediction technique is TopNDeviation, which is used in 79.6% of all predictions (see *Table 5-10*); the prediction accuracy of the second best prediction technique, GenreLMS, differs only 1% from the prediction accuracy of TopNDeviation; the difference with the worst prediction technique is 16.6%. In the EPG dataset, the prediction accuracy of the worst prediction technique, AlreadyRated, differs with 157.8% from the prediction accuracy of CBR; i.e. accuracy of predictors in MovieLens is very similar.

Table 5-10 Number of times predictors are used by the rule-based prediction strategy in the MovieLens dataset

	Overall	1-10000	10001-20000	20001-30000	30001-40000	40001-50000
CBR	113 (0.11%)	0 (0.00%)	0 (0.00%)	1 (0.01%)	0 (0.00%)	0 (0.00%)
GenreLMS	10671 (10.67%)	5006 (50.06%)	1694 (16.94%)	906 (9.06%)	594 (5.94%)	548 (5.48%)
CollaborativeFiltering	6758 (6.76%)	4 (0.04%)	43 (0.43%)	231 (2.31%)	575 (5.75%)	839 (8.39%)
UserAverage	2826 (2.83%)	1464 (14.64%)	423 (4.23%)	273 (2.73%)	114 (1.14%)	127 (1.27%)
TopNDeviation	79632 (79.63%)	3526 (35.26%)	7840 (78.40%)	8589 (85.89%)	8717 (87.17%)	8486 (84.86%)
Total	100000 (100%)	10000 (100%)	10000 (100%)	10000 (100%)	10000 (100%)	10000 (100%)

	50001-60000	60001-70000	70001-80000	80001-90000	90001-100000
CBR	0 (0.00%)	54 (0.54%)	43 (0.43%)	3 (0.03%)	12 (0.12%)
GenreLMS	443 (4.43%)	514 (5.14%)	381 (3.81%)	231 (2.31%)	354 (3.54%)
CollaborativeFiltering	539 (5.39%)	728 (7.28%)	1143 (11.43%)	793 (7.93%)	1863 (18.63%)
UserAverage	128 (1.28%)	94 (0.94%)	79 (0.79%)	68 (0.68%)	56 (0.56%)
TopNDeviation	8890 (88.9%)	8610 (86.10%)	8354 (83.54%)	8905 (89.05%)	7715 (77.15%)
Total	10000 (100%)	10000 (100%)	10000 (100%)	10000 (100%)	10000 (100%)

Similar results as for the EPG dataset concerning correctness can be found in the MovieLens dataset (see *Table 5-11*). Here CBR is again the predictor with the highest correctness percentage; this higher correctness is statistically significant ($p<0.05$) in the overall dataset; in the subsets 1-10000, 10001-2000, 40001-50000, 60001-70000 and 80001-90000 the difference is statistically significant ($p<0.05$) except with TopNDeviation where there is no significant difference. However, in the 71.9% that CBR is not the best predictor it provides bad predictions: $gmae=0.5757$ over the whole dataset, compared to $gmae=0.4274$ for the 72.1% of the instances that the rule-based prediction strategy is not the best predictor.

Notice that the correctness percentages for all predictors in the MovieLens dataset are similar and they are lower than the correctness percentages in the EPG dataset: in the MovieLens prediction engine, there is no prediction technique that outperforms the other prediction techniques like CBR does in the EPG dataset.

Table 5-11 Correctness percentages of the non-rounded predictors in the MovieLens dataset

	Overall	1-10000	10001-20000	20001-30000	30001-40000	40001-50000
CBR	**29.1%**	**29.8%**	**29.2%**	**30.7%**	**29.4%**	**27.9%**
GenreLMS	23.6%	21.0%	22.4%	23.1%	23.7%	23.6%
CollaborativeFiltering	17.8%	19.3%	18.5%	18.0%	18.2%	16.8%
UserAverage	16.7%	17.7%	16.6%	17.1%	16.7%	16.9%
TopNDeviation	27.9%	**29.6%**	**29.6%**	29.1%	27.7%	**27.0%**
Rule-based prediction strategy	23.9%	25.4%	26.2%	26.3%	23.9%	23.2%

	50001-60000	60001-70000	70001-80000	80001-90000	90001-100000
CBR	**29.8%**	**28.2%**	**28.7%**	**29.4%**	**28.0%**
GenreLMS	22.6%	26.5%	24.3%	23.0%	25.9%
CollaborativeFiltering	18.0%	17.3%	18.2%	17.3%	17.0%
UserAverage	16.9%	15.6%	17.0%	16.0%	16.8%
TopNDeviation	28.0%	**27.0%**	26.1%	**29.1%**	26.0%
Rule-based prediction strategy	24.0%	23.2%	21.8%	23.6%	21.8%

Rounding the predictions to the user's feedback scale also shows higher prediction accuracy (see *Table 5-12*) and correctness percentages (see *Table 5-13*) in the MovieLens dataset for all predictors. The rule-based prediction strategy is still the most accurate predictor; the prediction accuracy is statistically different from the other predictors ($p<0.05$), except in the subsets 60001-70000 and 90001-100000 where it is not statistically different from GenreLMS and in the subset 80001-90000 where it is not statistically different from TopNDeviation. The increase in overall prediction accuracy of the rule-based prediction strategy over the best prediction technique is still only 1.7%.

Table 5-12 Prediction accuracy of rounded predictors in the MovieLens dataset (*gmae*)

	Overall	1-10000	10001-20000	20001-30000	30001-40000	40001-50000
CBR	0.4488	0.4431	0.4526	0.4472	0.4338	0.4540
GenreLMS	0.3791	*0.3802*	0.3876	0.3710	0.3666	0.3818
CollaborativeFiltering	0.3946	0.3986	0.3925	0.3830	0.3819	0.3958
UserAverage	0.4066	0.3998	0.4035	0.3939	0.3942	0.4089
TopNDeviation	*0.3750*	0.3987	*0.3749*	*0.3622*	*0.3592*	*0.3746*
Rule based strategy	**0.3685**	**0.3748**	**0.3676**	**0.3561**	**0.3560**	**0.3705**
Accuracy increase	1.7%	1.4%	2.0%	1.7%	0.9%	1.1%
Natural Upper Bound	0.2028	0.1830	0.1981	0.1983	0.1961	0.2098

	50001-60000	60001-70000	70001-80000	80001-90000	90001-100000
CBR	0.4556	0.4663	0.4265	0.4494	0.4601
GenreLMS	0.3890	***0.3871***	0.3675	0.3791	***0.3813***
CollaborativeFiltering	0.4019	0.4151	0.3767	0.3912	0.4099
UserAvgerage	0.4161	0.4289	0.3900	0.4034	0.4271
TopNDeviation	*0.3821*	0.3929	*0.3534*	***0.3670***	0.3856
Rule based strategy	**0.3788**	**0.3859**	**0.3491**	**0.3663**	**0.3805**
Accuracy increase	0.9%	0.3%	1.2%	0.2%	0.2%
Natural Upper Bound	0.2128	0.2100	0.2013	0.2070	0.2116

Table 5-13 Correctness percentages of the rounded predictors in the MovieLens dataset

	Overall	1-10000	10001-20000	20001-30000	30001-40000	40001-50000
CBR	56.6%	54.9%	55.5%	56.1%	57.5%	56.9%
GenreLMS	66.4%	62.8%	64.0%	67.0%	67.2%	67.2%
CollaborativeFiltering	63.6%	60.0%	63.3%	64.5%	64.5%	65.1%
UserAverage	61.3%	58.8%	61.1%	62.7%	62.1%	62.7%
TopNDeviation	68.0%	61.3%	**67.6%**	69.1%	69.1%	69.2%
Rule-based prediction strategy	**68.8%**	**64.1%**	**68.0%**	**70.1%**	**69.6%**	**69.7%**

	50001-60000	60001-70000	70001-80000	80001-90000	90001-100000
CBR	57.5%	55.4%	59.8%	57.0%	55.8%
GenreLMS	66.5%	**66.5%**	68.4%	66.9%	**67.9%**
CollaborativeFiltering	64.1%	61.8%	66.2%	64.4%	62.5%
UserAverage	61.5%	59.4%	63.5%	62.0%	59.6%
TopNDeviation	**68.4%**	66.7%	71.3%	**69.7%**	67.7%
Rule-based prediction strategy	**68.8%**	**67.5%**	**72.0%**	**69.8%**	**68.4%**

The correctness percentages for the rounded predictions show that the rule-based prediction strategy now has the highest correctness percentages; this difference is statistically significant ($p<0.05$) except in the subsets 10001-20000, 50001-60000, 60001-70000, 80001-90000 and 90001-100000 where it is not statistically different from one other predictor (see *Table 5-13* for the details).

Summarized, in the MovieLens dataset, the rule-based prediction strategy is an adequate to good prediction strategy as it is sometimes capable of providing more accurate predictions than the individual

prediction techniques, while in other situations it provides predictions of equal accuracy as the best prediction technique; this is independent of whether predictions are rounded to the user's feedback scale or not. However, the increase in prediction accuracy is only small.

5.3.4 Rule-based prediction strategy conclusions

Both the prediction accuracy results for the EPG dataset and the prediction accuracy results for the MovieLens dataset show that rule-based prediction strategies are indeed capable of providing more accurate predictions by switching between the available predictors; hence supporting the hypothesis of the Duine prediction framework that switching between several predictors increases the prediction accuracy of prediction engines.

The significant increase of prediction accuracy in the EPG dataset is much higher (11.6%) than the significant increase of prediction accuracy in the MovieLens dataset (1.5%). This can be attributed to the fact that in the EPG dataset there is a dominator predictor, which allows the prediction strategy to base itself on this dominant predictor and use the other predictors as fallback; in the MovieLens dataset this is not the case, which makes the prediction strategy switch between predictors of similar overall accuracy, even though it bases itself mainly on one predictor, which lowers the probability of decreasing the overall prediction accuracy.

Another difference between the EPG dataset and the MovieLens dataset that contributes to the difference in prediction accuracy increase is that in the EPG dataset both social-based and information-based prediction techniques are used due to the available metadata about TV programs; in the MovieLens dataset mainly social-based prediction techniques are used as there is not enough metadata available about the movies to create effective information-based prediction techniques. If the MovieLens dataset can be extended with richer metadata concerning the movies, better information-based prediction techniques can be created, allowing the prediction strategy to use both social-based and information-based predictors; we leave this open for research.

Rounding predictions to the same ordinal scale as which users use to rate items increases the prediction accuracy and correctness of prediction strategies and prediction techniques. However, this is an artificial increase and can only be attributed to the different measuring scale, not to improvements in the predictors. We recommend using rounded predictions in prediction accuracy measurements when rounded predictions are presented to the user; this more closely simulates the experience of the user.

Rule-based prediction strategies provide more accurate predictions than any other prediction technique, as they are capable of switching between

predictors. A downside of rule-based prediction strategies is that expert knowledge is required for the design of the rules and fine-tuning the rules requires experimentation. For this reason, in the next section a prediction strategy decision approach is examined that teaches itself when to use which predictor.

5.4 Case-based reasoning based prediction strategy

Due to the need of expert knowledge for the design of rule-based prediction strategies, it is interesting to determine if automated prediction strategy decision approaches are equally or even better capable of switching between predictors in a prediction strategy. In this section, case-based reasoning as a prediction strategy decision approach is examined.

When designing a prediction strategy using case-based reasoning, three major design decisions have to be made:

1. How to determine analogous cases?
2. Which similar cases to select?
3. How to limit the size of the case-base?

5.4.1 Determining analogous cases

The key element of CBR is to determine which old cases are analogies of the current case (Watson, 1997); which old prediction requests for which feedback has been received are analogies of the current prediction request. Traditional CBR systems calculate the distance between two cases; old cases with the least distance from the current case are retrieved to determine the outcome for the current case.

Distances are determined based on the attributes of a case. If CBR is used as a prediction technique, the attributes of a case are the metadata attributes of an item, such as author, genre, title, duration, etc. However, if CBR is used as a prediction strategy decision approach, the attributes of the case are the attributes of the predictors, i.e. the validity indicators. With CBR as a prediction strategy decision approach, the outcome of each case is the accuracy of that predictor for that specific case.

The differences in distance of the retrieved cases are taken into account: the closer the case is to the current situation, the more important that case should be in determining the outcome for the current request. This means that selected analogous cases have to be weighted according to their closeness to the current case.

Using distances as weights is difficult due to the non-fixed upper level of distance; a distance of zero means identical cases and the higher the distance the less identical cases are, but the maximum possible distance is

unknown or may differ per distance measure. However, similarity measures are mostly in the range of [0, 1] (Watson, 1997); one means that two cases are exactly the same and zero means that two cases are completely different. Similarity values can be directly used as weights to represent the importance of a case. As traditional CBR systems use distances, it is necessary to convert distances into similarities.

Case-based reasoning distance measures

The four most frequently used distance measures in CBR systems are (Mendes, Mosley, & Watson, 2002):

1. Unweighted Euclidean distance
2. Weighted Euclidean distance
3. Maximum measure
4. Mean squared difference

The *unweighted Euclidean distance (ued)* between two cases a and b is calculated by summing the squared differences between all attributes x of the two cases and taking the square root of this sum:

$$d_{a,b} = \sqrt{\sum_{i=0}^{n} (x_{a,i} - x_{b,i})^2}$$

The *weighted Euclidean distance (wed)* between two cases a and b is the same as the unweighted Euclidean distance, except that each attribute has a certain weight w_i which is used as a multiplier for the difference of the attribute between the two cases:

$$d_{a,b} = \sqrt{\sum_{i=0}^{n} w_i (x_{a,i} - x_{b,i})^2}$$

By using weights, a distinction between more important and less important attributes can be made.

The distance with the *maximum measure (mms)* is determined by the attribute that has the largest distance and is determined by the formula:

$$d_{a,b} = \sqrt{\max((x_{a,i} - x_{b,i})^2)}$$

With *mean squared difference (msd)*, the distance is calculated by taking the average of the sum of the squared differences between all case attributes:

$$d_{a,b} = \frac{\sum_{i=0}^{n}(x_{a,i} - x_{b,i})^2}{n}$$

Besides these four basic CBR distance measures, also an information retrieval measure has been explored, namely cosine similarity. The *cosine similarity (cs)* measure (Salton & McGill, 1983) is the only of these functions that directly results in a similarity value instead of a distance. With cosine similarity, the values of the attributes are treated as a vector. Each case is described by one vector; the similarity between two cases is determined by the cosine of the angle between the two vectors representing that case. Cosine similarity can be calculated by:

$$s_{a,b} = \frac{\vec{A} \cdot \vec{B}}{\left|\vec{A}\right|\left|\vec{B}\right|} = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} x_{a,i}x_{b,j}}{\sqrt{\sum_{i=0}^{n} x_{a,i}^{\ 2}}\sqrt{\sum_{j=0}^{m} x_{b,j}^{\ 2}}}$$

Distance to similarity measures

Three different types of conversion functions for the conversion of distance to similarity have been explored: linear functions, sigmoidal functions and the inverted function. Of course, other functions could also have been examined, however, these function represent three major types of functions, namely linear, quick and smooth descending functions.

The linear function *linear(d, m)* converts the distance *d* to similarity *s* linear over the distance from 0 to *m*, where *m* represents an upper boundary after which all cases can be regarded as too non-similar; distances equal to or larger than *m* all have a similarity of 0:

$$if\ d < m\ then\ s = \frac{(m-d)}{m}\ else\ s = 0$$

A sigmoidal function is often used in machine learning for smooth transitions (Mitchell, 1997). The sigmoidal function *sigmoidal(d, k, m)* has a tuning parameter *k,* which is used to determine the flatness of the smoothing function. The sigmoidal function, including transformations of the function to place it in the distance domain of [0, *m*] and in the similarity range of [0, 1] is:

$$s = \frac{1}{1 + e^{k(d - \frac{1}{2}m)}}$$

Using $0.5m$ assures that similarity is 0.5 halfway between zero distance and the maximum affordable distance m. This is not strictly necessary; one might decide to place the similarity midpoint anywhere between 0 and m. However, in order to compare the smoothening effects of a sigmoidal function to linear functions that all have a midpoint at $0.5m$, the midpoint for the sigmoidal functions have been set at $0.5m$.

The inverted function *inverted(d)* does not have its similarity midpoint at $0.5m$. The function used is the inverted function, transposed to have a value of 1 at a distance of 0:

$$s = \frac{1}{(d + 1)}$$

The difference between the three conversion functions is visualized in *Figure 5-1*.

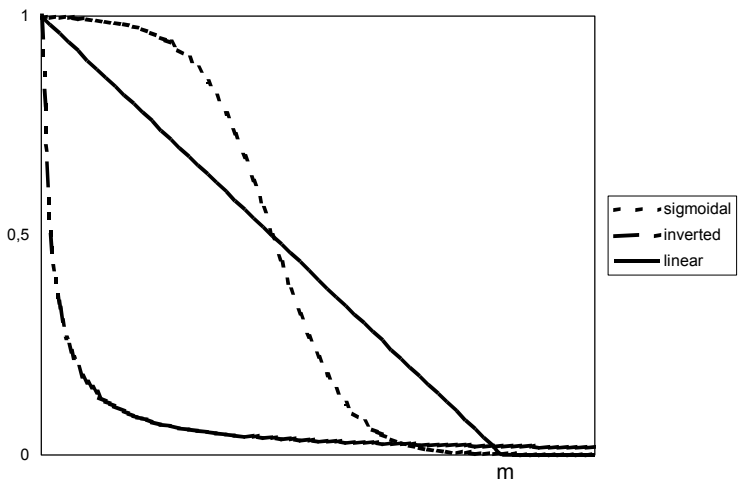


Figure 5-1 Distance to similarity conversion functions

5.4.2 Selecting similar cases

When the similarity of the current case with the cases in the case-base have been calculated, the next decision is which of the cases in the case-base have to be chosen to base the result for the current case on. Possible selection options are:

1. *All cases*: in this situation, all cases are used and the similarity of each case with the current case is used as a weight in calculating the result for the current case.
2. *Eldest most similar n cases*: when there are multiple cases that are the most similar to the current case, only the eldest *n* of those cases are used; e.g. use the three eldest most similar cases.
3. *Newest most similar cases*: when there are multiple cases that are the most similar to the current case, only the newest *n* of those cases are used; e.g. use the two newest most similar cases.
4. *All cases where similarity exceeds a threshold*: only those cases that have at least a similarity with the current case that is equal to or larger than a predefined threshold are used; the similarity of those cases are then used as a weight in calculating the result for the current case; e.g. use all cases that have a similarity of at least 0.7.

When small sets of data are used, only a limited set of similar cases, one, two or three cases, are necessary to derive the possible outcome for a new case (Mendes et al., 2002). In larger case-bases, thresholds or all cases are used to determine a set of most similar cases.

5.4.3 Limiting case-base sizes

One of the drawbacks of using CBR as a prediction strategy is the scalability issue of CBR: the larger the case-base, the more time it takes to make a decision; for every prediction request, similarity has to be calculated between the current request and all cases in the case-base. For this reason, it can be necessary to limit the size of the case-base. Different methods for limiting the case-base size can be used, such as first-in/first-out, least used or least recently used.

However, removing old cases from the case-base has a risk: some of the removed cases may represent situations that, although they do not occur often, they do occur every now and then. An example is a new user registering in the system. Because little to nothing is known about a new user, several prediction techniques that perform well for established users may not be able to providing accurate predictions for a new user. If the case-base is too small, cases representing such an event may have been removed by the time such an event reoccurs.

Because the frequency in which special situations occur differs per system, the optimal size of the case-base also differs per system; the more frequently special situations occur, the smaller the necessary case-base size. For this reason, it is necessary to experiment with different case-base sizes in order to determine the optimal size: optimal with regard to both prediction accuracy and prediction speed. However, as this research focuses

on prediction accuracy and not on prediction speed, the optimal case-based size is determined based on prediction accuracy only.

5.4.4 Choices for CBR strategy

The first decision when using CBR as a prediction strategy is whether to use one case-base for all predictors or to use separate case-bases for each predictor. As discussed in section 4.3.1, separate case-bases are used, as that not only keeps the prediction strategy more flexible, it also increases the probability that a similar set of cases can be found.

Several experiments have been performed to determine the optimal CBR parameters for both datasets; i.e. what conversion function, distance measure and case selection method to use; limiting the case-based size is discussed later in this section.

While exploring different similarity functions for the EPG dataset, it was quickly discovered that steeply descending similarity functions for all prediction techniques provide the best results, e.g. inverted, linear(d, m and sigmoidal(d, k, m) where $m \leq 4$. With such small distance ranges, especially when combined with a threshold value, both inverted and sigmoidal functions can easily be approximated by computationally simpler linear functions. Such steeply descending similarity functions mean that only old cases that are very close to the current case can be used to determine the outcome for the current case.

Of all distance measures, the unweighted Euclidean distance measure is always one of the best, no matter what distance to similarity function was used; this can be attributed to the fact that most predictors have only one validity indicator. Of the different distance to similarity conversion functions, a linear function with a low value for m performs best also taking into account the computational simplicity of a linear function. In order to determine what value to use for m and similarity threshold t, running several simulations resulted in $m=2$ and $t>0.70$ providing the best results.

In the MovieLens dataset, it was again found that the unweighted Euclidean distance provides the best results, however, this time in combination with a linear function with higher values for m. The best performing linear function had a value of $m=60$ when combined with a similarity threshold of $t>0.70$.

5.4.5 Results

Prediction accuracy in the EPG dataset

The results of the CBR-based prediction strategy in the EPG dataset shows that the CBR-based prediction strategy outperforms the best prediction technique in each of the four weeks and in the overall dataset (see *Table 5-*

14, bold indicates the statistically best predictor). The prediction accuracy of all the prediction techniques are of course the same as with testing the rule-based prediction strategy (consult *Table 5-4* for those values). The differences with the best prediction technique are statistically significant ($p<0.05$). The overall increase in prediction accuracy is 12.5%.

Table 5-14 Prediction accuracy of non-rounded CBR-based prediction strategy (*gmae*) in the EPG dataset

	Overall	Week 1	Week 2	Week 3	Week 4
CBR-based strategy	**0.2080**	**0.3034**	**0.1710**	**0.1861**	**0.1535**
Best prediction technique	0.2376	0.3530	0.1899	0.2171	0.1693
Accuracy increase	12.5%	14.1%	10.0%	14.3%	9.3%

An analysis of the number of times the various predictors have been used by the CBR-based prediction strategy (see *Table 5-15*) shows that the CBR-based strategy used various prediction techniques and used the best techniques, such as CBR and Information Filtering, the most; i.e. it is capable of discriminating between accurate and less accurate predictors.

These results show that in the EPG dataset, CBR as a prediction strategy decision approach results in a good prediction strategy; it is capable of providing better predictions than the prediction techniques. In section 5.5, the CBR-based prediction strategy will be compared with the rule-based prediction strategy.

The results also show that the increase in prediction accuracy of the CBR-based prediction strategy is larger in the first and third week compared to the second and fourth week. In the first week, users are new to the recommender; a lot of prediction techniques still have to learn from the users. In this week, the best prediction technique CBR has not learned enough about what TV programs users are interested in; the CBR-based prediction strategy can rely on other prediction techniques until CBR has enough knowledge about the user's interests to provide accurate predictions. A similar situation occurs in the third week; instead of new users, the transition of the summer TV season to the winter TV season takes place, which means that a lot of new TV programs are introduced; for several prediction techniques, such as CBR, this means that it has to learn anew what users think about these programs. The CBR-based prediction strategy is again capable of relying on other prediction techniques, such as Information Filtering, to provide accurate predictions.

Table 5-15 Number of times predictors are used by the non-rounded CBR-based prediction strategy in the EPG dataset

	Overall	Week 1	Week 2	Week 3	Week 4
AlreadyRated	0 (0.00%)	0 (0.00%)	0 (0.00%)	0 (0.00%)	0 (0.00%)
CBR	18034 (57.49%)	3919 (44.20%)	5803 (65.62%)	4103 (59.55%)	4209 (62.19%)
GenreLMS	1926 (6.14%)	912 (10.29%)	413 (4.67%)	361 (5.24%)	240 (3.55%)
SubGenreLMS	26 (0.08%)	17 (0.19%)	6 (0.07%)	2 (0.03%)	1 (0.01%)
WeightedAverageOf(GenreLMS, SubGenreLMS)	885 (2.82%)	414 (4.67%)	194 (2.19%)	149 (2.16%)	128 (1.89%)
InformationFiltering	8451 (26.94%)	2315 (26.11%)	2095 (23.69%)	2037 (29.56%)	2004 (29.61%)
CollaborativeFiltering	324 (1.03%)	193 (2.18%)	77 (0.87%)	39 (0.57%)	15 (0.22%)
UserAvg	1391 (4.43%)	780 (8.80%)	247 (2.79%)	195 (2.83%)	169 (2.50%)
TopNDeviation	331 (1.06%)	317 (3.58%)	8 (0.09%)	4 (0.06%)	2 (0.03%)
Total	31368 (100%)	8867 (100%)	8843 (100%)	6890 (100%)	6768 (100%)

The correctness percentages in the EPG dataset (see *Table 5-16*) show that the CBR-based prediction strategy does not have the highest correctness percentage; i.e. it is not the best predictor in most of the cases. However, it provides more accurate predictions than the best prediction technique, CBR, as in those situations where the strategy is not the best predictor it has a much better prediction accuracy ($gmae$=0.3785) than when the best technique, CBR, is not the best predictor ($gmae$=0.4794).

Table 5-16 Correctness percentages of the non-rounded CBR-based prediction strategy in the EPG dataset

	Overall	Week 1	Week 2	Week 3	Week 4
CBR-based strategy	49.8%	40.6%	53.9%	52.4%	54.0%
Best prediction technique	**53.9%**	**47.3%**	**58.0%**	**54.7%**	**56.4%**

When the predictions of the CBR-based prediction strategy are rounded to the user's feedback scale in the EPG dataset, the prediction accuracy results (see *Table 5-17*) show that the CBR-based prediction strategy is still more accurate than the best prediction technique in the overall dataset and each of the four weeks; differences are statistically significant ($p<0.01$). The overall increase in prediction accuracy is 15.6%.

Table 5-17 Prediction accuracy of the rounded CBR-based prediction strategy

	Overall	Week 1	Week 2	Week 3	Week 4
CBR-based strategy	**0.1926**	**0.2907**	**0.1530**	**0.1713**	**0.1373**
Best prediction technique	0.2280	0.3458	0.1778	0.2090	0.1588
Accuracy increase	15.6%	15.9%	13.9%	18.1%	13.6%

The correctness percentages (see *Table 5-18*) show that the rounded CBR-based prediction strategy is the best predictor in 77.6% of the prediction requests; the difference with the correctness percentages of the other predictors is statistically significant ($p<0.01$).

Table 5-18 Correctness percentages of the rounded CBR-based prediction strategy in the EPG dataset

	Overall	Week 1	Week 2	Week 3	Week 4
CBR-based strategy	**77.6%**	**66.7%**	**81.7%**	**80.4%**	**83.8%**
Best prediction technique	74.2%	62.3%	79.2%	76.3%	81.0%

Summarized, in the EPG dataset, the CBR-based prediction strategy is a good prediction strategy as it is capable of providing more accurate predictions than the individual prediction techniques, independently of whether predictions are rounded to the user's feedback scale or not.

Prediction accuracy in the MovieLens dataset

The non-rounded prediction accuracy results of the CBR-based prediction strategy in the MovieLens dataset (see *Table 5-19*, bold indicates the statistically better predictor) show that the CBR-based prediction strategy does not always provide more accurate predictions than the best prediction technique; only in the first three sets and the last set and looking at the overall dataset does the CBR-based prediction strategy provides more accurate predictions than the best prediction technique (statistically significantly, $p<0.05$); in all other cases there is no significant difference with the best prediction technique; i.e. it does not perform better or worse than the best prediction technique. However, the best prediction technique differs per subset; sometimes GenreLMS is the best prediction technique, other times TopNDeviation (see *Table 5-9*); the CBR-based prediction strategy is capable of dealing with this variation, a single prediction technique is not.

Table 5-19 Prediction accuracy of non-rounded CBR-based prediction strategy (*gmae*) in the MovieLens dataset

	Overall	1-10000	10001-20000	20001-30000	30001-40000	40001-50000
CBR-based strategy	**0.3882**	**0.3926**	**0.3870**	**0.3775**	**0.3776**	**0.3919**
Best prediction technique	0.3919	0.3965	0.3912	0.3811	**0.3758**	**0.3906**
Accuracy increase	1.0%	1.0%	1.1%	0.9%	-0.5%	-0.3%

	50001-60000	60001-70000	70001-80000	80001-90000	90001-100000
CBR-based strategy	**0.4004**	**0.4007**	**0.3735**	**0.3863**	**0.3941**
Best prediction technique	**0.3983**	**0.4026**	**0.3713**	**0.3855**	0.3991
Accuracy increase	-0.5%	0.5%	-0.6%	-0.2%	1.3%

The overall increase in prediction accuracy is only 1% and in some subsets there is even a decrease in prediction accuracy (though not statistically significant); indicating that the CBR-based prediction strategy is not as effective in the MovieLens dataset as it is in the EPG dataset.

An analysis of the number of times the various predictors have been used by the CBR-based prediction strategy in the MovieLens dataset (see *Table 5-20*) shows that the CBR-based prediction strategy used various prediction techniques, with a large focus on GenreLMS and TopNDeviation; the two predictors that are the most accurate of all prediction techniques (see *Table 5-9*).

Similar to the non-rounded prediction accuracy results, the correctness percentages also show that the CBR-based prediction strategy does not really outperform the best prediction technique (see *Table 5-21*). In none of the ten subsets is the correctness percentage of the CBR-based prediction strategy better than that of the best prediction technique; in seven of the subsets it is statistically equal to the best prediction technique; in the other three subsets it is statistically lower.

Table 5-20 Number of times predictors are used by the non-rounded CBR-based prediction strategy in the MovieLens dataset

	Overall	1-10000	10001-20000	20001-30000	30001-40000	40001-50000
CBR	402 (0.40%)	31 (0.31%)	45 (0.45%)	35 (0.35%)	7 (0.07%)	5 (0.05%)
GenreLMS	39250 (39.25%)	5793 (57.93%)	3634 (36.34%)	3133 (31.33%)	3331 (33.31%)	4065 (40.65%)
CollaborativeFiltering	5488 (5.49%)	194 (1.94%)	108 (1.08%)	309 (3.09%)	408 (4.08%)	563 (5.63%)
UserAverage	3809 (3.81%)	2796 (27.96%)	502 (5.02%)	163 (1.63%)	81 (0.81%)	48 (0.48%)
TopNDeviation	51050 (51.05%)	1185 (11.85%)	5711 (57.11%)	6360 (63.60%)	6173 (61.73%)	5319 (53.19%)
Total	100000 (100%)	10000 (100%)	10000 (100%)	10000 (100%)	10000 (100%)	10000 (100%)

	50001-60000	60001-70000	70001-80000	80001-90000	90001-100000
CBR	17 (0.17%)	88 (0.88%)	108 (1.08%)	19 (0.19%)	47 (0.47%)
GenreLMS	3731 (37.31%)	4329 (43.29%)	4094 (40.94%)	3440 (34.40%)	3700 (37.00%)
CollaborativeFiltering	742 (7.42%)	584 (5.84%)	641 (6.41%)	595 (5.95%)	1344 (13.44%)
UserAverage	53 (0.53%)	62 (0.62%)	33 (0.33%)	22 (0.22%)	49 (0.49%)
TopNDeviation	5457 (54.57%)	4937 (49.37%)	5124 (51.24%)	5924 (59.24%)	4860 (48.60%)
Total	10000 (100%)	10000 (100%)	10000 (100%)	10000 (100%)	10000 (100%)

Table 5-21 Correctness percentages of the non-rounded CBR-based prediction strategy in the MovieLens dataset

	Overall	1-10000	10001-20000	20001-30000	30001-40000	40001-50000
CBR-based strategy	27.6%	24.8%	28.0%	**29.1%**	**27.6%**	**26.7%**
Best prediction technique	**29.1%**	**29.8%**	**29.6%**	**30.7%**	**29.4%**	**27.9%**

	50001-60000	60001-70000	70001-80000	80001-90000	90001-100000
CBR-based strategy	26.8%	**29.0%**	**27.6%**	**28.7%**	**27.7%**
Best prediction technique	**29.8%**	**28.2%**	**28.7%**	**29.4%**	**28.0%**

Although the prediction accuracy of the CBR-based prediction strategy improves when predictions are rounded to the user's feedback (see *Table 5-22*), it improves less than the used prediction techniques. With rounded predictions, the CBR-based prediction strategy is only statistically more accurate ($p<0.05$) in four out of the ten subsets; in two subsets and overall it is statistically equal to the best prediction technique and in four of the subsets the prediction accuracy is statistically even less. Where the 1% increase in prediction accuracy is statistically significant when predictions are not rounded, the 1% increase is statistically no longer significant when predictions are rounded.

Table 5-22 Prediction accuracy of the rounded CBR-based prediction strategy in the MovieLens dataset

	Overall	1-10000	10001-20000	20001-30000	30001-40000	40001-50000
CBR-based strategy	**0.3713**	**0.3762**	**0.3708**	**0.3590**	0.3614	0.3758
Best prediction technique	**0.3750**	0.3802	0.3749	**0.3622**	**0.3592**	**0.3746**
Accuracy increase	1.0%	1.1%	1.1%	0.9%	-0.6%	-0.3%

	50001-60000	60001-70000	70001-80000	80001-90000	90001-100000
CBR-based strategy	**0.3853**	**0.3852**	0.3542	0.3690	**0.3761**
Best prediction technique	**0.3821**	0.3871	**0.3534**	**0.3670**	0.3813
Accuracy increase	-0.8%	0.5%	-0.3%	-0.6%	1.4%

The correctness percentages of the rounded CBR-based prediction strategy in the MovieLens dataset (see *Table 5-23*) shows that there is little significant difference between the CBR-based prediction strategy and the best prediction technique in most of the subsets (seven out of ten and overall); the CBR-based prediction strategy is only in two subsets, 1-10000 and 90001-100000, more often correct (statistically significant, $p<0.05$) than the best prediction technique; in one subset, 50001-60000, the CBR-based prediction strategy is even worse.

Table 5-23 Correctness percentages of the rounded CBR-based prediction strategy in the MovieLens dataset

	Overall	1-10000	10001-20000	20001-30000	30001-40000	40001-50000
CBR-based strategy	**68.2%**	**63.7%**	**67.7%**	**69.6%**	**68.6%**	**68.6%**
Best prediction technique	**68.0%**	62.8%	**67.6%**	**69.1%**	**69.1%**	**69.2%**

	50001-60000	60001-70000	70001-80000	80001-90000	90001-100000
CBR-based strategy	67.6%	**67.2%**	**71.0%**	**69.3%**	**69.1%**
Best prediction technique	**68.4%**	**66.7%**	**71.3%**	**69.7%**	67.9%

All these results show that by definition the CBR-based prediction strategy is a good prediction strategy in MovieLens when prediction are not rounded to the user's feedback scale, however, when looking in more detail, it is only an adequate prediction strategy in six out of the ten subsets; i.e. most of the times the CBR-based prediction strategy provides predictions of similar quality as the best prediction technique; sometimes it provides better predictions. When predictions are rounded to the user's feedback scale, it is only an adequate prediction strategy and sometimes even a bad prediction strategy.

The fact that CBR-based prediction strategy does not work as well in the MovieLens dataset as in the EPG dataset can be contributed to the fact that the prediction techniques for the EPG dataset have a much higher spread in their prediction accuracy than the prediction techniques for the MovieLens dataset; the predictors in the MovieLens dataset have similar levels of prediction accuracy compared to the predictors in the EPG dataset. Spread is defined as the difference between the prediction accuracy of the best performing prediction technique and the prediction accuracy of the worst

performing prediction technique. The average spread over the 10 subsets in MovieLens is 0.0678, while the average spread over the four weeks in the EPG dataset is 0.3076. This means that the expected errors calculated by the CBR-based prediction strategy in the MovieLens dataset tend to be situated close together, making the probability of a wrong decision larger as the decisions of CBR are based on these expected errors. Since the spread is higher in the EPG dataset, the probability of a wrong decision is smaller.

5.4.6 Limiting case-based sizes

The EPG dataset

In order to determine the impact of limiting the size of the case-bases, which improves the prediction speed and scalability of a CBR-based prediction strategy, several experiments have been performed with several sizes using the first-in/first-out method. First-in/first-out is used because it allows the system to keep learning from recent situations; recent situations have a higher probability of resembling future prediction requests than older cases. The results, as displayed in *Table 5-24*, show that limiting the case-base size can even improve prediction accuracy. Limiting the case-base size to 12500 seems to provide the best results; the difference with using no limit for the case-base size is statistically significant ($p<0.01$). However, the gain in prediction accuracy is only 0.6% over the whole four weeks.

Table 5-24 Results of limiting the case-base size in the EPG dataset (bold indicates better predictions than no limit; prediction accuracy is not rounded to the user's feedback scale)

Case-base size	Overall	Week 1	Week 2	Week 3	Week 4
No limit	0.2080	0.3034	0.1710	0.1861	0.1535
1000	0.2142	0.3056	0.1784	0.1965	0.1593
2500	0.2130	0.3047	0.1772	0.1917	0.1614
5000	0.2099	**0.3029**	0.1743	0.1881	0.1568
7500	0.2083	0.3043	0.1728	**0.1842**	0.1535
10000	**0.2076**	0.3043	0.1720	**0.1831**	**0.1523**
12500	**0.2068**	0.3043	**0.1708**	**0.1830**	**0.1506**
15000	**0.2071**	0.3043	0.1713	**0.1841**	**0.1502**
17500	**0.2074**	0.3043	0.1712	**0.1853**	**0.1501**

We hypothesize that the removal of old cases made the strategy more accurate, since these cases represented old situations that did not occur again in the system. Furthermore, some prediction techniques behave differently early on in a system than they do later on, even under the same conditions according to the validity indicators. For example, in two situations A and B, a validity indicator of collaborative filtering indicates

that there are 40 similar users that have rated the item for which a prediction is necessary; however, in situation A – early in the system – the similarity of these 40 users is based on less rated items by each user than in the later situation B; hence the probability that collaborative filtering provides an accurate prediction is higher in situation B than in situation A[4].

However, the improved effect of limited case-base sizes may also be influenced by the time sensitive characteristic of the EPG dataset. Because all users started using the system at the beginning of week 1, no special situations like a new user occurred in the later weeks. On the other hand, there is one special occasion in the dataset between week 2 and week 3: at that time almost all channels changed their programming drastically because at that time the new TV season started; this made existing users similar to new users. The limited case-base size did not have any negative effects on the prediction accuracy; on the contrary, prediction accuracy increased more with limited case-base sizes after the second week than with unlimited sizes.

The MovieLens dataset

Limiting the size of the case-base in MovieLens, appears to improve the prediction accuracy of the CBR-based prediction strategy only slightly with a case-base size of 40000; the overall *gmae* of the dataset using this case-base size is lower than when no limit is used; however this improvement is not statistically significant ($p>0.05$).

Table 5-25 Results of limiting the case-base size in MovieLens (bold indicates better predictions than no limit; prediction accuracy is not rounded to the user's feedback scale)

	Overall	1-10000	10001-20000	20001-30000	30001-40000	40001-50000
No Limit	0.3882	0.3926	0.3870	0.3775	0.3776	0.3919
10000	0.3887	0.3926	**0.3863**	0.3807	0.3779	0.3929
20000	0.3885	0.3926	0.3870	0.3784	0.3783	0.3924
30000	0.3883	0.3926	0.3870	0.3775	**0.3767**	0.3925
40000	**0.3880**	0.3926	0.3870	0.3775	0.3776	**0.3914**
50000	0.3883	0.3926	0.3870	0.3775	0.3776	0.3919
60000	0.3882	0.3926	0.3870	0.3775	0.3776	0.3919
70000	**0.3881**	0.3926	0.3870	0.3775	0.3776	0.3919
80000	**0.3881**	0.3926	0.3870	0.3775	0.3776	0.3919
90000	**0.3881**	0.3926	0.3870	0.3775	0.3776	0.3919

[4] An alternative way to deal with this situation would be to introduce an additional validity indicator for collaborative filtering, one that indicates how many rated items have been used in calculating the similarity between this user and other users.

	50001-60000	60001-70000	70001-80000	80001-90000	90001-100000
No Limit	0.4004	0.4007	0.3735	0.3863	0.3941
10000	**0.3993**	0.4024	**0.3724**	0.3876	0.3948
20000	**0.4003**	**0.4003**	0.3740	0.3872	0.3941
30000	0.4006	**0.4002**	0.3748	0.3867	0.3941
40000	0.4008	**0.3988**	0.3742	0.3872	**0.3933**
50000	0.4006	0.4010	0.3738	0.3869	0.3941
60000	0.4004	**0.4005**	0.3738	0.3865	**0.3938**
70000	0.4004	0.4007	0.3735	**0.3862**	**0.3935**
80000	0.4004	0.4007	0.3735	**0.3862**	**0.3939**
90000	0.4004	0.4007	0.3735	0.3863	**0.3937**

The small spread of prediction accuracy in the predictors used in the CBR-based prediction strategy in MovieLens is also the reason why limiting the case-base size does have little effect in comparison with limiting the case-base size in the EPG dataset; this small spread in prediction accuracy of the predictors does not change when a smaller case-base size is used.

5.4.7 Case-based reasoning based prediction strategy conclusions

The prediction accuracy results in the EPG dataset show that CBR as a prediction strategy decision approach is capable of providing more accurate prediction than any of the prediction techniques; it is a good prediction strategy. However, in the MovieLens dataset the results show that in that dataset CBR as a prediction strategy decision approach is only an adequate to good prediction strategy, which is caused by the small spread in prediction accuracy of the prediction techniques in the MovieLens dataset, which makes it more difficult for CBR to discriminate between the prediction techniques.

Rounding predictions of the CBR-based prediction strategy to the user's feedback scale has no influence on the prediction accuracy in the MovieLens dataset and shows an improved prediction accuracy in the EPG dataset. However, this is an artificial increase and not caused by improved decisions of the CBR-based prediction strategy.

Limiting the case-base size improved the prediction accuracy of the CBR-based prediction strategy in the EPG dataset; in the MovieLens dataset it did not increase the prediction accuracy, but it neither decreased prediction accuracy (using optimal case-base sizes). As a limited case-base size is preferred for the prediction speed and scalability of a prediction strategy, it is recommended to limit the case-base size in CBR-based prediction strategies.

Even though the EPG dataset only consists of four weeks of data, the transition of the summer TV season to winter TV season within the dataset shows that the CBR-based prediction strategy is capable of handling such changes in the set of items. We expect that if more weeks of data would have been available, the prediction strategy would remain more accurate than any of the individual prediction techniques; the strategy would rely a lot on CBR as a prediction technique for those programs that are regularly on TV, while using other predicting techniques, such as Information Filtering or the strategy that combines GenreLMS and SubGenreLMS for new programs or programs that are not that often on TV. And if data of more users had been available in the dataset, prediction techniques such as Collaborative Filtering and TopNDeviation would have been used more often by the CBR-based prediction strategy.

5.5 Comparing prediction strategy decision approaches

The results of the two examined prediction strategy decision approaches in the EPG dataset are summarized in *Table 5-26*; the results of the best non-rounded prediction strategy version of the two approaches are compared to each other. The differences between the rule-based prediction strategy and the CBR-based prediction strategy with a case-base size of 12500 are statistically significant ($p<0.01$); both in the overall dataset as within each of the four weeks.

Table 5-26 Accuracy results of the two prediction strategies in the EPG dataset (bold indicates the best prediction strategy in that week)

	Overall	Week 1	Week 2	Week 3	Week 4
Rule-based prediction strategy	0.2100	**0.2935**	0.1774	0.1933	0.1601
CBR-based prediction strategy (case-base size=12500)	**0.2068**	0.3043	**0.1708**	**0.1830**	**0.1506**
Accuracy increase (from rule-based to CBR-based)	1.5%	-3.7%	3.7%	5.3%	5.9%

These results show that the CBR-based prediction strategy is the best prediction strategy in the EPG dataset, although the increase in prediction accuracy from the rule-based prediction strategy to the CBR-based prediction strategy is only 1.5% for the overall dataset. However, in the first week, the rule-based prediction strategy is statistically more accurate than the CBR-based prediction strategy; this can be attributed to the fact that the CBR-based prediction strategy needs to build up a case-base; i.e. it takes time to learn before CBR can make good decisions about when to use

which predictor. In the later weeks, the CBR-based prediction strategy is more accurate than the rule-based prediction strategy, even up to 5.9% in the fourth week.

The prediction accuracy of the two examined non-rounded prediction strategy decision approaches in the MovieLens dataset is summarized in *Table 5-27*; again, the results of the best prediction strategy version of the two approaches are compared to each other. The differences are not always statistically significant; in some subsets there is statistically no difference ($p>0.05$) in the prediction accuracy of the two approaches: in sets 1-10000 and 20001-30000.

Table 5-27 Accuracy results of the two prediction strategies in the MovieLens dataset (bold indicates the best prediction strategy in that set)

	Overall	1-10000	10001-20000	20001-30000	30001-40000	40001-50000
Rule-based prediction strategy	**0.3859**	**0.3910**	**0.3848**	**0.3754**	**0.3734**	**0.3868**
CBR-based prediction strategy (case-base size=40000)	0.3880	**0.3926**	0.3870	**0.3775**	0.3776	0.3914
Accuracy increase (from rule-based to CBR-based)	-0.5%	-0.4%	-0.6%	-0.6%	-1.1%	-1.2%

	50001-60000	60001-70000	70001-80000	80001-90000	90001-100000
Rule-based prediction strategy	**0.3944**	0.4026	**0.3680**	**0.3842**	0.3980
CBR-based prediction strategy (case-base size=40000)	0.4008	**0.3988**	0.3742	0.3872	**0.3933**
Accuracy increase (from rule-based to CBR-based)	-1.6%	0.9%	-1.7%	-0.8%	1.2%

These results show that most of the times the rule-based prediction strategy is equally or more accurate than the CBR-based prediction strategy; only in two of the ten subsets is the CBR-based prediction strategy more accurate (60001-70000 and 90001-100000). The rule-based prediction strategy is only 0.5% more accurate compared to the CBR-based prediction strategy in the overall dataset.

As discussed in section 5.4.5, the CBR-based prediction strategy is not a good prediction strategy in the MovieLens dataset due to the fact that the various predictors used within the strategy are all very similar; which is revealed in the low spread in prediction accuracy between the predictors. The CBR-based prediction strategy has difficulties in discriminating

between the predictors based on their expected prediction error. The rule-based prediction strategy on the other hand, uses a fixed set of decision rules, which are not affected by the low spread in prediction accuracy of the predictors; it just makes its decisions based on the values of the validity indicators not on expected errors.

5.6 Conclusions

In this chapter, the Duine Prediction Framework has been validated by examining two different prediction strategy decision approaches: one manual static model-based using decision rules and one automated dynamic instance-based using case-based reasoning.

Experiments have shown that manually-created rule-based prediction strategies are capable of providing more accurate predictions by switching between the available predictors; hence supporting the hypothesis of the Duine prediction framework that switching between several predictors increases the prediction accuracy of prediction engines.

Another set of experiments showed that using CBR as a prediction strategy decision approach can also provide more accurate predictions by switching between the available predictors; however, in datasets that have predictors with similar levels of prediction accuracy, CBR is only an adequate prediction strategy; i.e. sometimes it provides more accurate predictions than the individual prediction techniques, but most of the times it provides predictions of similar prediction accuracy as the best individual prediction technique.

One of the main benefits of using CBR instead of the manually-created rules is that no expert knowledge is required to create the prediction strategies. A downside of using CBR as a prediction strategy is the performance and scalability penalty of CBR; e.g. a simulation using the CBR-based prediction strategy without limiting the case-base size was 7.7 times slower than a simulation using the rule-based prediction strategy in the MovieLens dataset. Rule-based systems are very fast and scalable because they are model-based. Since CBR is instance-based, the prediction speed of CBR-based strategies depends on the size of the case-bases. In some systems, limiting the size of the case-bases not only improves speed and makes the system more scalable, it can also improve the prediction accuracy of a CBR-based prediction strategy.

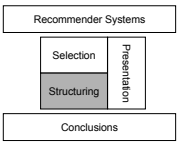
Based on these finding, we conclude that the Duine prediction framework, which uses a switching hybridization method based on validity indicators of prediction techniques, can indeed be used to create prediction engines for recommender systems that are capable of providing more accurate predictions than any of the individual prediction techniques. This

means that prediction engines created with the Duine prediction framework can be used in the selection process of personalized information systems in order to help people to find interesting items within the vast amount of information, products and services that is available on today.

With these results, our first research objective has been reached. This research objective focused on the selection process of personalized information systems. In the next chapter, the focus is on the structuring process of personalized information systems, were the use of goals in structuring items will be investigated.

Chapter 6

Goal-Based Structuring



Supporting users in finding interesting items is one of the key solutions in overcoming the information overload problem. In the previous chapters, the focus lay on selecting interesting items for a user using a framework that allows for the creation of prediction engines that combine multiple prediction techniques in order to predict how interesting items will be for a user. However, predicting and selecting are not the only way to support users in finding interesting items; structuring and presentation are also part of the solution and need to be taken into account when designing a personalized information system (see section 1.1.3).

Current prediction techniques only address the user's short-term and long-term interests, not the user's immediate interests; immediate interests are the interests a user has at a specific moment in time, while short-term interests are interests someone has for a couple of days or weeks; long-term interests are interests people have for a longer period of time (years). In this chapter, a structuring method is described that complements the Duine prediction framework and which provides support for users in finding interesting items also taking into account their interests at a specific moment: a goal-based structuring method. This structuring method is described in detail in this chapter and we validate whether goal-based structuring actually helps people to find interesting items via an experiment with an electronic TV program guide.

In section 6.1, the use of goals in recommender systems is introduced. Section 6.1.2 discusses how a recommender can determine what the current goal(s) of a user are in order to use these goals in its recommendations. The use of goals as a way to structure recommendations is then validated in an experiment. The hypotheses that are tested in the experiment and the design of the experiment are discussed in section 6.2, while the experimental system is described in section 6.3. Section 6.4 describes and analyzes the sample of participants of the experiment, while the hypotheses are tested in section 6.5. In section 6.6, details concerning

the gains and efforts of using predictions and goal-based structuring are examined. This chapter is concluded in section 6.7.

A condensed version of this chapter has also been accepted to be published in van Setten, Veenstra, Nijholt & van Dijk (to be published).

6.1 Using goals in recommender systems

6.1.1 Information retrieval and goals

Information retrieval can be regarded as a decision process; for each piece of information a decision is made whether it matches the user's query or not. Recommenders make a slightly different decision: for each piece of information a decision is made whether it is interesting enough for the user or not. An interesting aspect of looking at recommenders as decision makers is that a common aspect in various decision-making theories can be used; most decision-making theories agree that people try to achieve goals when making a decision. Kass & Finin (1988) define a goal as some state of affairs a user wishes to achieve.

The bounded rationality theory, a theory first introduced by H.A. Simon in 1956, describes decision making "as a search process that is guided by [...] values or a goal variable that must be reached or surpassed by a satisfactory decision alternative" (Selten, 2001, page 13-14). According to the rational choice theory, which originated from economics research, "individuals are seen as motivated by the wants or goals that express their 'preferences'" (Scott, 2000). The means-end approach, which tries to explain the decision making process of a consumer when buying a product, describes how people try to achieve goals by looking at the consequences of product attributes and not by looking at the attributes themselves (Reynolds & Olson, 2001). The differences between the various decision-making theories lies in the way they perceive how people make decisions in order to reach their goals.

If we assume that these theories are correct, it means that for items recommended by recommender systems, people also have goals they want to achieve with those items. If a recommender is aware of these goal(s), it can use this knowledge to provide better recommendations.

6.1.2 Determining a user's current goals

Determining a user's current goal can be accomplished in three ways depending on where the decision effort is placed. On the one extreme, a recommender can ask a user to *specify* his current goal(s) and recommend items belonging to that goal; this puts all the effort on the user. This

assumes that people are capable of and willing to make their goal(s) explicit and that they are capable of articulating these goals; this is not always the case (Kass & Finin, 1988). For these reasons, this option is not been examined any further in this research.

On the other end, a recommender can try to *predict* the user's current goal(s). This is comparable with predicting how interesting an item is; all the effort to make this decision is put on the recommender. This requires more knowledge about a user and his context than is currently possible to acquire; e.g. in the TV domain, factors such as the emotional and physiological state are important indicators for a user's goal when watching TV (Zillmann & Bryant, 1986, page 303-324). A good starting point for research into determining the user's current goals is the research track that tries to determine what a user is trying to achieve when using a software application in order to provide him with targeted help, e.g. Horvitz, Breese, Heckerman, Hovel & Rommelse (1998). One has to take into account though that trying to determine what a user tries to achieve in a software application takes place in a confined and structured environment, while trying to determine the user's current goal(s) when using an information system is less structured and more open; this requires detailed knowledge about the user and his context. Although recommenders can certainly benefit from acquiring such detailed user and context knowledge and reasoning about what there current goal(s) might be, we leave this open for future research and first focus on investigating if using goals will actually help users in finding interesting items.

Finally, a combination of predicting and specifying can be used, where the effort is shared between the user and the recommender: the recommender *structures* the items into different groups that correspond to the different possible goals users may have. The user then picks that group that best matches his current goal. The difference between using goals via structuring versus having a user explicitly specify his goals is similar to the difference between browsing and querying; with querying the user receives only the items exactly matching his query, whereas with browsing the user is able to navigate to the required items while adjusting and/or refining his goal(s) in the process. Having a user specify his goal(s) to a recommender results in a set of items that only matches the specified goal(s); when using goals to structure recommendations, the user is able to navigate through the recommended items, meanwhile adjusting and/or refining his goal(s) based on the items presented using the goal-based structure.

Structuring recommendations according to the possible goals does not require knowledge about the current goals of the user, it only requires that a recommender knows the possible goals users may have in the domain in which the recommender operates and determine which of these goals each item would achieve for the user; e.g. in the TV domain it is necessary to

know the possible goals people have for watching TV and what goal(s) each TV program will help a specific user to achieve. Goal-based structuring allows a recommender to support the user by using goals, while leaving the final decision about the user's current goal(s) to the user.

Determining the possible goals people can have in a certain domain is a topic that is also being researched in the mass-communication domain using the uses and gratification theory.

6.1.3 Uses and gratification theory

The uses and gratification theory can help to determine the goals of people when accessing information. In 1959, Elihu Katz (1959) first introduced the uses and gratifications theory (Severin & Tankard, 2001, page 293). This theory states that people choose the types of media (TV, newspapers, radio, etc.) that they will expose themselves to based on certain gratifications or some sense of personal satisfaction that they expect to receive; i.e. individuals actively seek out media and items that provide them with useful information or psychological gratifications, such as entertainment or emotional comfort, and avoid media and items with displeasing characteristics (Cooper, Roter & Langlieb, 2000). The main opinion at that time (the needle point theory) was that publishers decide in what media to publish items and that those for whom those items are intended will receive them; the receiving party has no choice in the process. The uses and gratification theory takes into account the choice options of individuals; people decide for themselves what media to access for what items. Later application of the uses and gratification theory went beyond the choice of what media people choose to access certain items; the focus shifted to the choices of people within and over types of media; the gratifications people receive from items.

According to the uses and gratification theory "communication behaviour, including the selection and use of the media, is goal-directed, purposive, and motivated" (Severin & Tankard, 2001, page 527). Furthermore, "people initiate the selection and use of communication vehicles" (Severin & Tankard, 2001, page 527) and "a host of social and psychological factors guide, filter, or mediate communication behaviour" (Severin & Tankard, 2001, page 528); i.e. individuals decide upon which media and items to access based on their personal goals and social and psychological factors. Although recommender systems can support people in the process of selecting items, in the end it is the user who chooses what items he will access, not the recommender.

Uses and gratification theory helps to determine the gratifications people expect to receive from using certain items. However, gratifications are not the same as goals. Gratifications are what the user experiences after

using items; goals are what the user would like to achieve using items. In an ideal situation, the experienced gratifications are sufficient to meet the goal(s) of the user (see *Figure 6-1*).

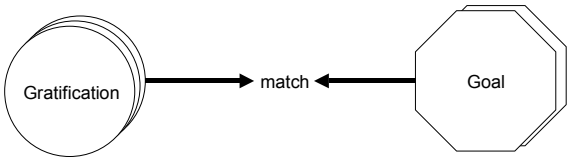


Figure 6-1
Gratifications received should match the goal(s) of the user

Knowing the possible gratifications users may receive from using items within a domain is not enough for a recommender to structure items according to goals; recommenders must also be able to determine which goal(s) can be achieved for a user by a specific item. One of the decision-making theories, the means-end approach, provides useful insights into the relationship between items and achieving goals.

6.1.4 Means-end approach

Decisions made by users about what information to access are similar to decisions made by consumers when buying products or services. In both cases, people have to make a choice between alternatives (including doing nothing); between different products or services or between different information items. One might argue that there are no costs involved in accessing information, which is clearly the case when buying products; however, even for some pieces of information, people have to pay some price; furthermore, people also have to invest time to access and use information. Recommenders support this choice by amplifying the differences between alternatives that are important to the user.

Reynolds & Olson (2001) describe a conceptual framework, called the means-end approach, for understanding how consumers use choice criteria in their decisions between alternatives. The basic assumption is that people decide between alternatives based on the anticipated consequences of each alternative and not on the direct attributes of an alternative. "Attributes, taken alone, have no consequences, and thus have no relevance. Consequences only occur when the consumer buys and consumes (or uses) the product and thereby experiences the consequences of use" (Olson & Reynolds, 2001, page 15). In recommender systems, the "attributes" concept of the means-end approach is equivalent with the item and its metadata for which a prediction must be made.

The means-end approach is based on a set of assumptions that guide the means-end approach (Reynolds & Olson, 2001):

- In order to fulfil their needs or to remove a deficiency, people can choose between different alternatives. This need or deficiency can be seen as a problem that can be solved by making a decision between alternatives.
- In the means-end approach, the focus lies on the consequences of using or applying a chosen alternative; consequences can be achieving a goal or a sub goal of the actual goal.
- Consequences can be either positive or negative, and both are important for making a decision. A person seeks positive consequences, while avoiding negative consequences; i.e. people select the alternative that maximizes the positive consequences and minimizes the negative ones.
- The most direct consequences are the ones that are functional. These consequences are the direct cause of using or experiencing the chosen alternative. Other, also relevant, consequences are more of an emotional nature, namely the psychosocial (psychological and social) consequences.
- The connections or linkages between product attributes and the consequences carry most of the meaning in making a choice between alternatives.
- Different people can value consequences differently. This makes it important to identify those consequences that are most important to a person for a specific decision.
- Intentional, conscious decision-making: Every choice between alternatives is a conscious, voluntary and intentional choice, even though a choice can become a habit (once there was a first time when the specific choice was made). This is an implicit assumption in the means-end approach; people can always choose between at least two alternatives (even if the choice is between doing something or doing nothing).

The most basic means-end model consists of attributes that lead to consequences when the product is used; these consequences contribute to the values or goals of the user (see *Figure 6-2*).

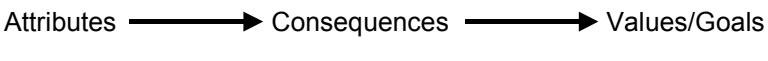


Figure 6-2 Basic means-end model

Consequences of accessing information depend on the item itself and the person who accesses the item. E.g. for one person watching a certain comedy on TV may result in that that person is being entertained, while another person may become irritated watching that same comedy. Whether consequences are positive or negative depends on the goals and values of the person; the consequences of using an item should match with the goals of the user.

The concept of consequences is similar to the concept of gratifications in the uses and gratification theory; consequences are that what happens to a user when using products or information; gratifications are that what a user experiences from using items. However, gratifications are only the positive subset of all possible consequences; as people seek positive consequences and try to avoid negative consequences (Reynolds & Olson, 2001, page 10) gratifications that match the user's goal are more important in the recommendation process than negative consequences which people try to avoid.

The means-end approach indicates that people make decisions based on the consequences of using items, not on the attributes of items and also not explicitly on the goals they want to achieve. The goals to achieve are implicit in the decision process, the consequences or gratifications are explicit; hence, goal-based structuring methods can better employ the explicit consequences or gratifications than the implicit goals: goal-based structuring should be done on gratifications not on goals. This results in a means-end model for goal-based structuring that describes how item attributes lead to one or more gratifications by using an item; these gratifications should match the goal(s) the user wants to achieve (see *Figure 6-3*); attributes of an item like title and genre are objective, while gratifications that are received when an item is used are subjective. E.g. a TV program with attributes like "Comedy" and "American", will lead for some people to the gratification "mood improvement" when watched, while for others a TV program with attributes like "Comedy" and "British" would lead to that gratification. Depending on the person, either the first or the latter program should be watched when the goal is to improve his or her mood.

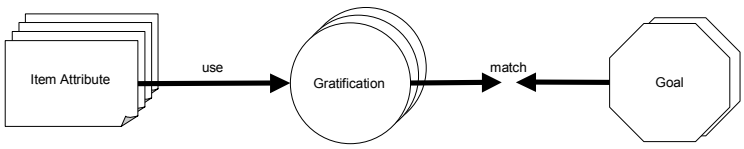


Figure 6-3 Means-end model for goal-based structuring

The next sections validate this model, focusing on whether using goal-based structuring actually helps users in finding interesting items and how this compares to the help users get from using recommendations in the form of predictions.

6.2 Validation of goal-based structuring

Just like the validation of the Duine prediction framework, the validation of goal-based structuring takes place in the domain of EPG recommender systems. Before discussing how to validate goal-based structuring, it is first important to notice that TV viewers can use a TV in two different ways (see section 7.3.3): the first way is to plan a couple of hours to watch TV and then watching TV according to this plan; the second way is to switch on the TV and to look for an interesting program that is on at that moment by channel surfing. An electronic program guide (EPG) mainly supports the first type of interaction where it is important what will be on TV later; with channel surfing it is not important what is on TV later, only what is on TV now.

The second type of interaction can be supported to a certain extent by an EPG using a "Now on TV" section showing all programs that are on TV at that moment. Another concept that supports the second type of interaction would be an "intelligent channel surfer": when a user turns on the TV, it automatically switches to the channel with the most interesting program for that user (the one with the highest predicted rating); the channel up and down buttons do not navigate through the channels in their programmed order, but using the order of the predicted ratings of the programs that are currently on. However, this concept is subject for future research.

When validating goal-based structuring in an EPG, the focus lies on the first type of interaction, planning a couple of hours watching TV.

6.2.1 Hypotheses

The main hypothesis when validating the use of goals in a recommender system is that using predictions and goal-based structuring both help users in finding interesting items. This hypothesis has been refined in a set of five hypotheses that are tested in the experiment:

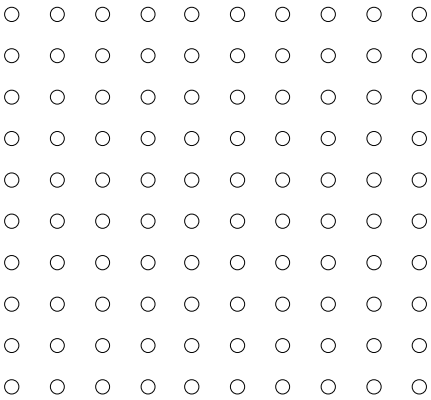
Hypothesis 1:

> *Using predictions for items makes it easier for users to find interesting items than using no predictions.*

When receiving a set of items without predictions, the user has to examine all items to determine which items are interesting for him; the use of predictions removes this burden from the user; hence it becomes easier for a user to find interesting items when predictions are used than when no predictions are used. E.g. in *Figure 6-4* there are 100 items in a set. Without using predictions, the user has to examine all 100 items in order

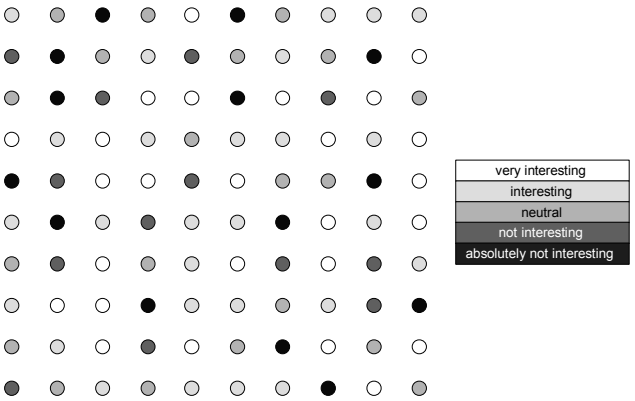
to determine which item he will use (or he may examine less and take the first acceptable item, which may not be the best possible item).

Figure 6-4 Possible items for a user without any predictions or structuring



However, when using predictions, there are fewer items to examine. In *Figure 6-5* only 26 items have a prediction labelled 'very interesting'. However, the user still has to locate these 26 items within all 100 items.

Figure 6-5 Possible items with predictions but without goal-based structuring



Hypothesis 2:

> *Structuring items based on the user's goals makes it easier for users to find interesting items than using structures that are not based on the user's goals.*

Structuring a set of items provides order. When a structure is based on the goal(s) of a user, the user only has to go through the group(s) of items that match his current goal(s); when a non-goal based structure is used, a user still has to examine items from multiple groups to identify those items that

match his current goal(s). For example, if the goal of a user in *Figure 6-6* is C than he only has to examine the 20 items that match goal C.

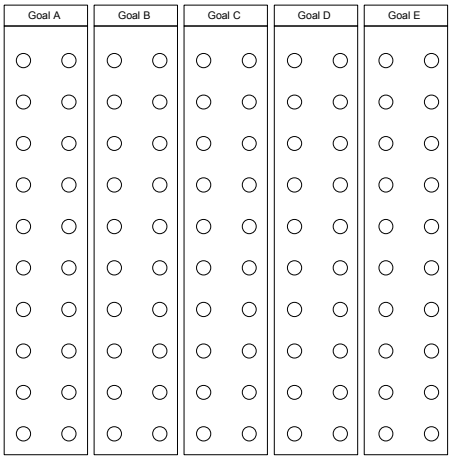


Figure 6-6 Possible items without predictions but with goal-based structuring

Hypothesis 3:

> *Using both predictions and structuring items based on the user's goals makes it easier for users to find interesting items than using no predictions and no structures that are based on the user's goals.*

When both predictions and a goal-based structure are used, the number of items that a user has to examine is reduced even more. A user only has to examine those items that match his goal and that have a high prediction. For example, in *Figure 6-7* the user that wants to achieve goal C only has to examine the 5 very interesting items within goal C.

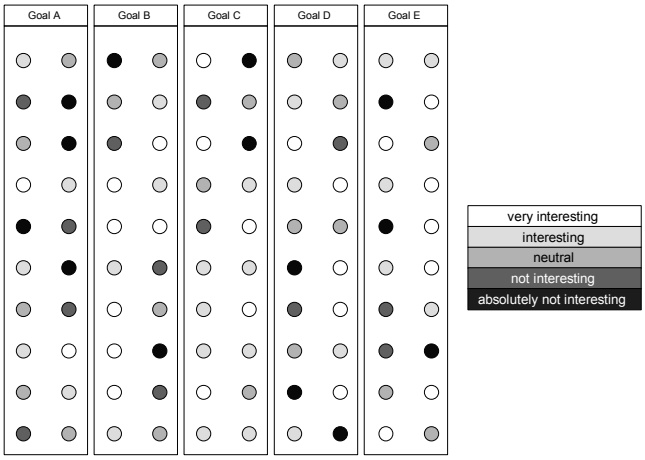


Figure 6-7 Possible items with predictions and goal-based structuring

When besides using a goal-based structure, items are also sorted (which is also a type of structuring) within each goal using the predictions, a user can find interesting items even faster (see *Figure 6-8*).

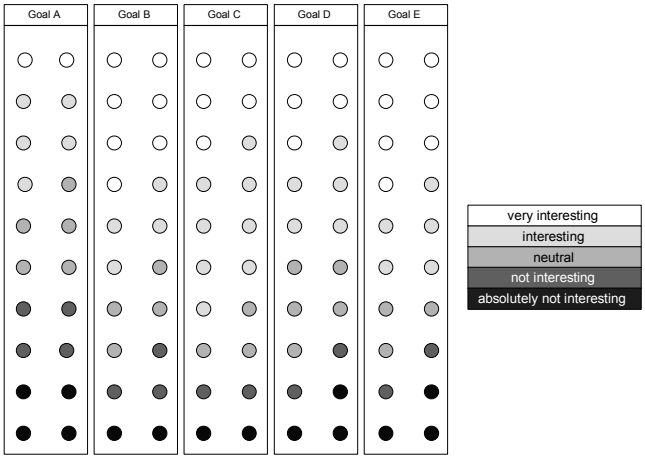


Figure 6-8 Possible items with predictions, a goal-based structure and sorted on predictions within the goal-based structure.

When assuming that goal-based structuring and the use of predictions can enhance each other, one can derive from hypotheses 1, 2 and 3 that:

Hypothesis 4:

> *Using both predictions and structuring items based on the user's goals makes it easier for users to find interesting items than using only predictions.*

Hypothesis 5:

> *Using both predictions and structuring items based on the user's goals makes it easier for users to find interesting items than using only structures that are based on the user's goals.*

To test these hypotheses, it is first necessary to make "how easy it is for people to find interesting items" measurable.

6.2.2 Measuring "how easy it is to find interesting items"

"How easy it is to find interesting items" is a complex construct that can be interpreted in several ways, such as the speed in which interesting items are found, the ease-of-use and how helpful the system is in finding interesting items. Several studies have researched ways to measure how helpful technology is to its users, including aspects such as speed, ease-of-use, and usefulness. Venkatesh, Morris, Davis & Davis (2003) compared various studies and integrated them into a unified theory of acceptance and use of technology; this theory measures the success of technology by measuring the intention that users will actually use new technology after deployment. Intention is measured, not real usage, as in most cases one wants to determine the probable success of new technology before introduction; this is also the case in this experiment; the system is only experimental and not available to the general public.

The intention users have for using a certain type of EPG is an indication of how good that EPG is in helping them find interesting TV programs. The intention to use an EPG that does not help users in finding interesting programs easily will be lower than the intention to use an EPG that does help users in finding interesting programs easily. For this reason, in our experiment we will measure the intention that users will use an EPG.

Venkatesh et al. (2003) examined eight different models that try to explain those factors that influence the acceptance by users of technology. Based on these eight models they formulated and empirically validated a unified model that integrates elements across these eight models. Their unified model is shown in *Figure 6-9*.

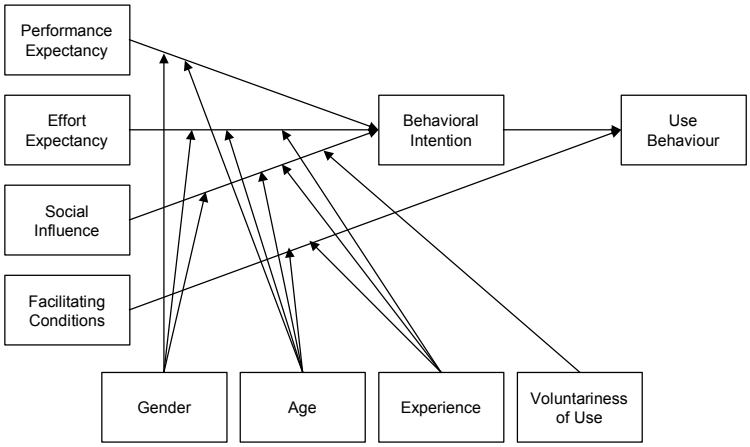


Figure 6-9 Unified Theory of Acceptance and Use of Technology

This model describes the four core determinants of intention and usage of new technology:

1. *Performance expectancy*: "the degree to which an individual believes that using the system will help him or her attain gains in job performance" (Venkatesh et al., 2003, page 447).
2. *Effort expectancy*: "the degree of ease associated with the use of the system" (Venkatesh et al., 2003, page 450).
3. *Social influence*: "the degree to which an individual perceives that important others believe he or she should use the new system" (Venkatesh et al., 2003, page 451).
4. *Facilitating conditions*: "the degree to which an individual believes that an organizational and technical infrastructure exists to support the use of the system" (Venkatesh et al., 2003, page 453).

As "facilitating conditions will not have a significant influence on behavioural intention […] [but] do have a direct influence on usage …" (Venkatesh et al., 2003, page 454) it is not necessary to measure the facilitating conditions in this experiment, as only the intention to use a certain type of EPG is measured, not the actual usage after the experiment.

There are four moderators that influence the relationship between the four core determinants and the intention and usage: gender, age, experience and voluntariness of use. Regarding the voluntariness of use, Venkatesh et al. notice that "none of the social influence constructs are significant in voluntary contexts" (Venkatesh et al., 2003, page 451). Since the use of the EPG is voluntary, the social influence determination is not relevant for this experiment. The gender and age moderators are of influence on the relationship between performance expectancy and intention (Venkatesh et al., 2003, page 450). The gender, age and

experience moderators are of influence on the relationship between effort expectancy and intention (Venkatesh et al., 2003, page 450).

As it is not necessary to take into account social influence and facilitating conditions in this experiment, a limited model of factors that influence the acceptance by users of the various EPGs can be used (see *Figure 6-10*).

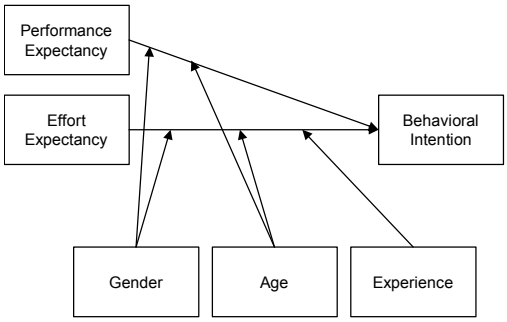


Figure 6-10 Limited UTAUT model for goal-based structuring experiment

The result of using this limited UTAUT model is that "how easy it is for users to find interesting items" is measured by behavioural intention. To be able to explain and understand the reasons behind the intention of participants in the experiment, it is also necessary to measure performance expectancy, effort expectancy, gender, age and experience with using EPGs.

As UTAUT and the studies it has been based on all focused on professional environments, it is necessary to translate the concrete measures of UTAUT to the home environment where concepts like tasks and job performance have little meaning; these have been translated to concepts like "finding interesting and fun TV programs" and "increased chances of a fun and interesting evening".

UTAUT uses four statements to measure performance expectancy (Venkatesh et al., 2003, page 460); translated to the home environment they are:

1. The EPG helps me to find interesting and fun TV programs.
2. Due to the EPG, I can find interesting and fun TV programs faster.
3. Due to the EPG, I watch less TV programs that disappoint me than without this online EPG.
4. Using the EPG increased my chances of a fun and interesting evening of watching TV.

UTAUT also uses four statements to measure effort expectancy (Venkatesh et al., 2003, page 460); the translated four statements are:

1. It is easy to learn the possibilities of the EPG.
2. The use of the EPG is clear and understandable.

3. The EPG is easy to use.
4. Learning to use the EPG is simple.

Intention is measured using the following translated statement:
1. If the EPG would become available as a real system, I intend to use the EPG.

UTAUT measures intention, performance expectancy and effort expectancy using a 7 point scale that measures the level of agreement to the statements with 1 being the negative end and 7 being the positive end of the scale (Venkatesh et al., 2003, page 438). Davis (1989) labels these 7 points as extremely unlikely, quite unlikely, slightly unlikely, neither, slightly likely, quite likely and extremely likely. We provided the participants with the statements mentioned above using seven radio buttons where the first was labelled "completely disagree" and the last was labelled "completely agree" (the five options in between were not explicitly labelled). One could argue whether this scale is an interval scale (1 to 7) or an ordinal scale (completely disagree to completely agree); i.e. if one believes that an agreement score of 4 is twice that of 2, the scale would be an interval, if not the scale is an ordinal scale. For ordinal scales, non-parametric test have to be used, for interval scales parametric tests can be used. Due to the grey area of the measuring scale used in this experiment, both non-parametric and parametric tests are used to examine intention, performance expectancy and effort expectancy.

For all tests a 95% confidence level is used to determine if differences are statistically significant. As the five hypotheses are one-sided - the expectation is that using predictions and/or goal-based structuring increases the intent to use an EPG - all significance values for hypothesis testing are based on one-tailed probabilities. As no hypotheses have been defined concerning performance expectancy and effort expectancy, significant differences for these measures are tested with two-tailed probabilities.

6.2.3 Gratifications for watching TV

The five hypotheses are examined in the domain of electronic program guides for TV. According to the means-end approach, TV viewers do not choose programs based on the attributes of the program, but on the anticipated consequences (gratifications) of watching a certain program; the attributes are used to determine these anticipated consequences.

The uses and gratification theory has been used to learn why people use certain items (Rubin, 2002), which resulted in typologies of uses and gratifications. Several investigations have been performed to discover the gratifications of media use; some tried to identify high-level gratifications,

also called orientations (McDonald, 1990), that describe an averaged attitude. Rubin (2002) describe two main orientations: ritualized use, using a medium more habitually to consume time and for diversion, and instrumental use, seeking certain media content for informational reasons. Other high-level gratifications have been identified by McDonald (1990), McQuail, Blumer and Brown in Severin & Tankard (2001), and by Katz, Gurevitch and Hass, also in Severin & Tankard (2001). These typologies are all based on high-level needs for media access.

Others have investigated need gratifications in specific domains such as TV (Lee & Lee, 1995; Weaver III, 2003; Donohew, Palmgreen & Rayburn II, 1987), websites (Eighmey & McCord, 1998) and Internet use (Stafford, Stafford & Schkade, 2004).

For our experiments with a TV recommender, the study of Lee & Lee (1995) is the most relevant. Results from this study have the highest level of detail of all TV gratification studies and the results encompass the results of the other TV gratification studies. This study started with 18 focus groups, followed by a quantitative survey of a national probability sample of 1872 television viewers in the US. This survey resulted in the identification of six gratification factors for watching TV:

1. *Committed/ritualized viewing*: planning an evening filled with favoured programs provides people with the enjoyment of anticipation.
2. *Mood improvement*: by watching TV, people can relax, relieve stress and escape everyday troubles, which improves their moods.
3. *Informational/cognitive benefit*: TV also keeps people up-to-date on events going on in the world (both locally as globally) and it provides people with a source for self-education and "food for thought".
4. *Social learning*: watching TV can also be used for self-examination and guidance through identification with people and situations on TV that are similar to ones own life.
5. *Social grease*: TV also has a role to smooth interpersonal relations. People that have seen the same programs have a topic to discuss, something to talk about.
6. *An engrossing different world (escapism)*: instead of being drawn by the similarities with ones own life, some TV programs allow people to "escape" to a different world in which they experience things they never would experience in the real world.

The gratifications used in the experiment to determine whether goal-based structuring actually help users in finding information are based on these six gratifications; however, the informational/cognitive benefit gratification has been divided into two separate gratifications as we believe that there is a difference between being informed about events and learning new things; learning something new does not necessarily include recent events that took

place in the world, while being informed about events does not imply that something is learned from those events.

The gratifications used have also been assigned more meaningful labels that better reflect the goals these gratifications serve and which are easier to understand for users:

1. Programs to keep up with (committed/ritualized viewing)
2. Improving my mood (mood improvement)
3. To be kept up-to-date (informational)
4. Learning new things (cognitive benefit)
5. Learning from others (social learning)
6. Watching what my friends watch (social grease)
7. To loose myself in a program (an engrossing different world (escapism))

6.2.4 Alternative structuring methods

In the experiment, goal-based structuring is compared to two other structuring methods:

1. Channel-based structuring
2. Genre-based structuring

Traditional paper TV guides group their programs by the channels on which they are broadcast; in Europe, most EPGs display the channels as columns and programs are sorted on time within these columns; however, in the US channels are displayed as rows and programs are sorted on time within these rows. Grouping programs on channel is also used in almost every existing EPG. For this reason, channel-based structuring is used in our experiment as the structuring method that represents the situation in which a structure is used that does not reflect user goals; one may argue that some channels are inherently goal-based due to their thematic programming; e.g. documentary channels and news channels. However, as channel-based structuring is the most widely known and used form of structuring, it is the best structuring method to use as the basis situation (control group) to which other structuring methods are compared.

Although the means-end approach indicates that people choose programs based on anticipated consequences instead of attributes of a TV program, it might be possible that making these consequences explicit is too unfamiliar to people. To anticipate this possibility, another way of structuring is also used, namely one using an attribute that is not a gratification in itself, but one that gives a indication of what gratifications to expect: the main genre of a TV program (see section 6.3 for details about the relationship between genres and gratifications). Genres are a way of implicitly structuring on goals.

6.2.5 Experimental design

To determine the effect of using predictions and/or goal-based structuring on how easy it is for people to find interesting items, i.e. testing the five hypotheses, an experiment with an EPG recommender has been designed to measure this effect. The independent variables in this experiment are "the use of predictions" and "the type of structuring"; these two variables are manipulated in the EPG to measure the effect on the dependent variable "how easy it is for people to find interesting TV programs".

The independent variable "using predictions" is a binary variable: predictions are used or predictions are not used (also called personalized versus non-personalized). As mentioned in the previous section, three types of structures are compared in this experiment: non goal-oriented structuring (channel-based), implicitly structuring on goals using an attribute that gives a good indication of what gratifications to expect (the main genre) and explicitly structuring on goals using gratifications (goal-based).

The three moderators that influence the intention of people accepting and using new technology (gender, age and experience) are possible classificatory variables that can be used in the experimental design; classificatory variables are variables that divide the population into two or more classes and which are not manipulated by the treatment (Neale & Liebert, 1986). Especially experience can be of great influence on the results of the experiment; the intention of people who never used an EPG to use a specific type of EPG will contain both their intention towards that specific EPG and their intention towards using EPGs in general; for people who already use EPGs their intention will only consist of the intention to use that specific type of EPG. For this reason, experience is used as a classificatory variable; people are assigned to groups taking into account their experience with using EPGs. To keep the experiment manageable, gender and age are not included as classificatory variables as their effects are less apparent, but their values will be acquired for checking their influence afterwards.

Factorial design

The two independent variables, "using predictions (yes/no)" and "the type of structuring (channel/genre/goal)" combined with the classificatory variable "experience (yes/no)" result is a mixed factorial design of 2 x 3 x 2. In a factorial design "two or more treatments, events, or characteristics are independently varied in a single piece of research." (Neale & Liebert, 1986, page 161). However, since only the first two variables are treatment variables, the last being a classificatory variable, the experimental design can be limited to a 2 x 3 (predictions x grouping) factorial design (see *Table 6-*

1) in which people are assigned to experimental groups taking into account their experience with using EPGs; each group receives a similar distribution of experienced and non-experienced participants. Each experimental group represents another type of guide based on the type of structuring and whether predictions are used or not.

Table 6-1 Factorial design

		Using Predictions	
		No	Yes
Structuring	Channel	1	4
	Genre	2	5
	Goal	3	6

Type of experiment

An experiment can be performed in two ways: a within-subjects experiment and a between-subjects experiment.

Within-subjects experiments are experiments in which a comparison is made between scores obtained before a treatment has occurred with scores obtained after the treatment has occurred from the same participants (Neale & Liebert, 1986). For this experiment, it would mean that participants use a certain type of EPG for some time, such as one with no predictions and a channel-based structure, after which the intention of using that EPG is measured. That same group then receives another type of EPG, such as one with no predictions and a goal-based structure. After using that other EPG for some time, intention for using that EPG is measured. The differences in the scores of the two measurements give an indication of which EPG is perceived as better.

With between-subjects experiments, comparisons are made between the scores of two separate groups that differ in whether or not they received a certain treatment (Neale & Liebert, 1986). For this experiment, it would mean that one group of participants would receive a certain type of EPG, such as one with no predictions and a channel-based structure, while another group of participants receives another type of EPG at the same time, such as one with no predictions and with a goal-based structure. After using these EPGs, in both groups the intention of using the EPG is measured. The differences in the scores between the two groups give an indication of which EPG is helping users better in finding interesting items.

Our experiment is based on the between-subjects approach; the within-subjects approach is less suitable due to possible learning effects that can occur when using different EPGs over time. Also the influence of external factors (such as differences in TV programming) provides a greater threat to the within-subjects experiments than to the between subjects experiment.

Summarised, the experiment is a between-subjects experiment consisting of six experimental groups based on the type of structuring and whether predictions are used or not to which participants are randomly assigned taking into account their experience in using EPGs.

6.3 Experimental system

The experimental EPG that has been developed to validate the five hypotheses encompasses six types of guides conforming to the six experimental groups. The look and feel and how users have to interact with the EPG is the same for all six guides, except for functionality that is specific to using predictions or a certain type of structuring; e.g. presentation of predictions and functionality to provide ratings are only present in guides that use predictions. Even for EPGs that do not use predictions, the recommender system is instructed to generate predictions in order to keep the processing time for all types of EPG similar, even though these predictions are never presented to the user. Screenshots of each type of guide can be found in Appendix A.

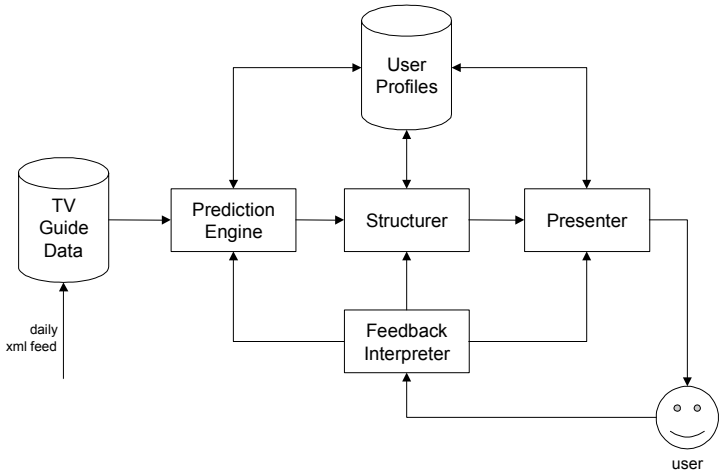


Figure 6-11 High level architecture of the experimental system

A high level architecture of the experimental system is shown in *Figure 6-11*. Every evening omroep.nl (the online TV guide of the Dutch public broadcasting companies) sends an XML file with all TV programs for the next three days to our experimental system, which imports these TV programs into the TV guide database. When a user requests a TV guide, the system retrieves the requested TV programs from the database and requests the prediction engine to calculate a prediction for each TV program and to provide data for explaining the reasons behind those predictions. The

structurer groups and sorts the TV programs based on the structuring method that is assigned to the user (which is stored in his user profile). The presenter creates the presentation of the TV programs and returns (a set of) web pages that can be displayed in the user's browser. The presentation has been based on the results of our study into user interfaces for recommender systems as described in chapter 7. All three components use the user's profile to retrieve and store preferences and/or interests of the user.

If the structuring method is explicitly goal-based, a goal represented by a gratification is assigned to each TV program. As the TV guide data provided by omroep.nl does not contain information about goals and because goal(s) of a TV program can differ between users, the experimental system uses two methods to assign goals:

1. Assignment of goals based on the main genre.
2. Explicit assignment of goals by the user as stored in his user profile.

Table 6-2 Mapping of main genres to goals

Goal	Main Genre
Mood improvement	Amusement Children Animation Comedy Music
Informational	Current Affairs Sports
Cognitive benefit	Nature Informative Documentary Science Other
Social learning	Religious Art/Culture
Escapism	Crime Serial/Soap Movie Erotic
Social Grease	Combined watch lists of a user's buddies
Committed/Ritualized Viewing	Only explicitly assigned TV programs

For new users and for TV programs that a user has not seen before, a goal is determined based on the main genre of the TV program, using the mapping as shown in *Table 6-2*. The list of genres is based on the list used by the public broadcasting companies and is part of the TV guide data provided by omroep.nl. If a user does not agree with these assigned gratifications, he can assign one or more other gratifications to that TV program, which is consequently stored in the user's profile. The next time that that TV

program (or other episodes of that TV program) appears in the EPG, those explicitly assigned gratifications will be used instead of the gratifications derived from the main genre.

The only exceptions to the genre-based assignment of gratifications are the committed/ritualized viewing gratification and the social grease gratification. A TV program is only assigned to the committed/ritualized gratification if the user has explicitly assigned that program to this gratification. The social grease gratification is filled with programs by combining all the watch lists of the user's buddies (independent of whether buddies use the same type of EPG); users can invite other people to take part in the experiment and become their buddies; a watch list is a list of TV programs that a user has selected from the whole TV guide which he or she intents to watch.

As mentioned in section 6.2.2, even though facilitating conditions are not of influence on the intent to use an EPG, they are important for the success of the experiment. For this reason, extensive help was made available to participants of the experiment in the form of detailed help files within the EPG. Every user also received an introductory e-mail message explaining the working of the guide that was assigned to that user. Finally, e-mail support was available to answer any questions participants may have had. In all communications, we tried not to influence participants and made sure that participants were not aware of the existence of other types of guides; however, one specific user in experimental group 1 (channel-based, no predictions) had so many questions and wishes, including the wish for a personalized guide, that we moved him to a special group that had access to all types of guides but of which the results have not been used in testing the hypotheses (see the next section).

6.4 Sampling

6.4.1 The sample

In the three and a half months that the EPG was online, 320 people created an account. These participants have been randomly assigned to one of the six experimental groups taking into account their experience in using EPGs. A special group has been used for colleagues and friends of the researchers and others whom already knew about the purpose of the experiment; the results of this group (consisting of 18 people) have not been used to test the hypotheses and are not included in the 320 accounts. Participation was voluntary; the only incentive for participants was that they were able to win a gift certificate of 50 euro.

After the three and a half months, all participants were asked to complete a survey in which their intention to keep using the EPG was measured (including performance expectancy and effort expectancy); 114 participants completed the survey. Even though this is a high dropout rate, the dropout rate is fairly distributed over the six experimental groups (see *Table 6-3*) according to a chi-square test: group 1: 67%, group 2: 58%, group 3: 69%, group 4: 66%, group 5: 67%, group 6: 68% ($\chi^2 = 1.067; df = 5; p = 0.9570$ two-tailed). Although it is not possible to know the real reason for this high dropout rate, it does not influence the results as the type of guide did not influence the dropout rates.

Table 6-3 Drop out rates

Experimental group	Dropout rate (percentage)
1	67%
2	58%
3	69%
4	66%
5	67%
6	68%

While examining these 114 surveys, it was discovered that five participants indicated that they were never able to use or had never used the EPG at all (they did not want to use Internet Explorer to access the EPG[5]). As the opinions of these users were not based on usage of the EPG, these five surveys have been excluded from analysis, resulting in 109 valid surveys. These 109 participants are distributed over the six experimental groups as described in *Table 6-4*. The participants are fairly distributed over the experimental groups ($\chi^2 = 1.257; df = 5; p = 0.939$ two-tailed).

Table 6-4 Distribution of participants over the six experimental groups

Experimental group	Frequency	Percentage
1	17	15.6%
2	22	20.2%
3	16	14.7%
4	19	17.4%
5	18	16.5%
6	17	15.6%
Total	109	100.0%

[5] The EPG has been developed for Internet Explorer as this browser was the most widely used browser at the time of the experiment; also some of the functionality to be tested required extensive client-side scripting, which would have been difficult and costly to develop for multiple browsers at the same time.

As the survey results from these six groups are to be compared for hypothesis testing, it is important that participants are well distributed over these six groups according to their gender, age and experience in using EPGs (see section 6.2.2). If distribution is not fair for one of these moderators, it is not possible to compare the six groups without explicitly taking these moderators into account; if distribution is fair, the six groups can be compared directly.

Gender distribution over experimental groups

The participants who completed a valid survey are distributed over the six experimental groups according to gender as described in *Table 6-5*:

Table 6-5 Gender distribution over the experimental groups

Experimental group	Gender		Total
	Female	Male	
1	3	14	17
2	8	14	22
3	4	12	16
4	6	13	19
5	6	12	18
6	6	11	17
Total	33	76	109

A chi-square test shows that the probability that the observed differences can be contributed to coincidence is 82.4% ($\chi^2=2.179$;df=5;p=0.824 two-tailed); i.e. participants are fairly distributed over the six experimental groups according to their gender.

Age distribution over experimental groups

To analyse the age of participants, participants are divided into three age groups:

- Junior: younger than 30 years of age.
- Medium: from 30 to 49 years of age.
- Senior: 50 years or older.

The participants who completed a valid survey are distributed over the six experimental groups according to their age as described in *Table 6-6*.

A chi-square test shows that the probability that the observed differences can be contributed to coincidence is 90.1% ($\chi^2=4.845$;df=10; p=0.901 two-tailed); i.e. the participants are fairly distributed over the six experimental groups according to their age.

Table 6-6 Age distribution over the six experimental groups

Experimental group	Age			Total
	Junior	Medium	Senior	
1	4	10	3	17
2	9	10	3	22
3	6	7	3	16
4	4	11	4	19
5	7	10	1	18
6	5	10	2	17
Total	35	58	16	109

Experience distribution over experimental groups

Although experience in using EPGs has been taken into account when assigning people to experimental groups, it is important to examine whether the subset of participants who completed the survey is still fairly distributed over the six experimental groups according to their experience in using EPGs. The participants who completed a valid survey are distributed over the six experimental groups according to prior experience in using EPGs as described in *Table 6-7*.

Table 6-7 Experience distribution over the six experimental groups

Experimental group	Experience		Total
	Low	High	
1	10	7	17
2	15	7	22
3	10	6	16
4	11	8	19
5	15	3	18
6	12	5	17
Total	73	36	109

A chi-square test shows that the probability that the observed differences can be contributed to coincidence is 60% ($\chi^2 = 3.656$;df=5;p=0.600 two-tailed); i.e. the participants are fairly distributed over the six experimental groups according to their prior experience in using EPGs.

As all three moderators, gender, age and experience, are fairly distributed over the six experimental groups, the results of these six groups can be compared when testing hypotheses without explicitly taking into account any of these moderators.

6.4.2 Generalizability

Multiple acquisition methods have been used to acquire participants in order to get a representative sample of TV guide users in the Netherlands, making it possible to generalize the results of the sample to the whole population: a banner in the online TV guide of the public broadcasting companies, flyers distributed in several major cities, mouth-on-mouth advertising and an invitation to a diverse group of Internet users from the Kenniswijk project in the city of Eindhoven. This way of sampling allowed us to find a representative group of participants including people with and without experience in using EPGs.

Even though there was a high dropout rate (as shown in the previous section), comparing the sample of 109 participants who completed the survey to all 320 people who registered to use the EPG (see *Table 6-8*) shows that the sample of 109 is a good representative of all registered accounts as far as the three moderators are concerned (gender $\chi^2=3.730;df=1;p=0.053$ two-tailed, age $\chi^2=0.1017;df=2;p=0.950$ two-tailed, experience $\chi^2=0.0446;df=1;p=0.833$ two-tailed); this makes the final sample of 109 participants just as representative as the group of all 320 registered participants.

Table 6-8 Comparing distribution of all participants to the sample

		All 320 participants	Sample of 109
Gender	Male	78%	70%
	Female	22%	30%
Age	Junior	33%	32%
	Medium	53%	53%
	High	14%	15%
Experience	Low	66%	67%
	High	34%	33%

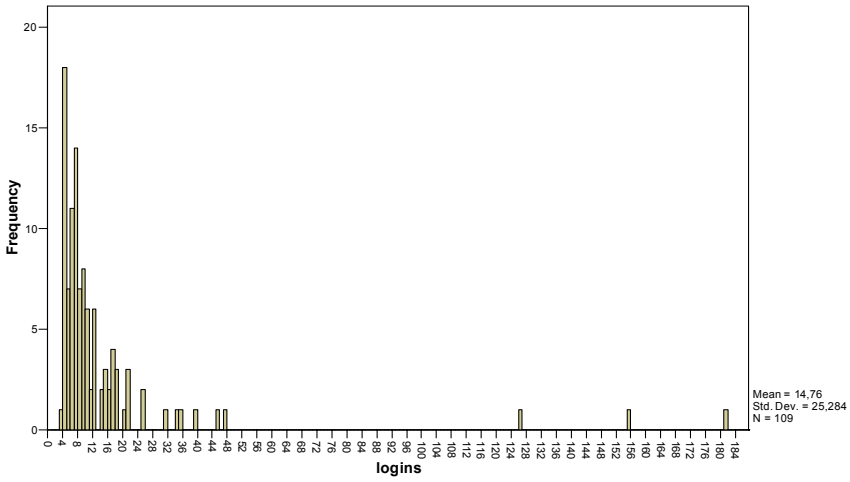
6.4.3 Weighting the cases

When testing the five hypotheses and examining performance expectancy and effort expectancy, cases will be weighted using the number of times a participants has used the EPG. There are two reasons to weight cases using the number of logins:

1. Frequent users are better capable to determine if the EPG helped them in finding interesting TV programs than users who only used the EPG a few times.
2. Due to the limited number of valid surveys some statistical tests will not be able to find any significant differences, even if there are any; e.g. a chi-square test between the six experimental groups and non-weighted intent shows that that 100% of the cells in the cross-table for this test

have an expected count of less than 5; i.e. there is not enough data to successfully apply the chi-square test. Weighting the cases can solve this issue, but only if a meaningful weight is assigned, otherwise the gained significance will be meaningless.

Figure 6-12 Histogram of the number of logins



Frequency of use is measured by the number of times a participant logged into the EPG. The usage logs of the experiment show that some participants used the EPG more than others. As there are a few extreme outliers in the number of times people logged into the EPG (see *Figure 6-12*), the number of logins have been mapped onto six groups, were the group number is used as a weight for opinions of participants in that group:

1. Number of logins ≤5.
2. Number of logins >5 and ≤10.
3. Number of logins >10 and ≤15.
4. Number of logins >15 and ≤20.
5. Number of logins >20 and ≤25.
6. Number of logins >25.

Without this mapping, the three extreme outliers would have dominated the results too much. The histogram of the number of login groups (nlogins) is shown in *Figure 6-13*, which has a similar shape to the non-grouped frequency of use, without having extreme outliers.

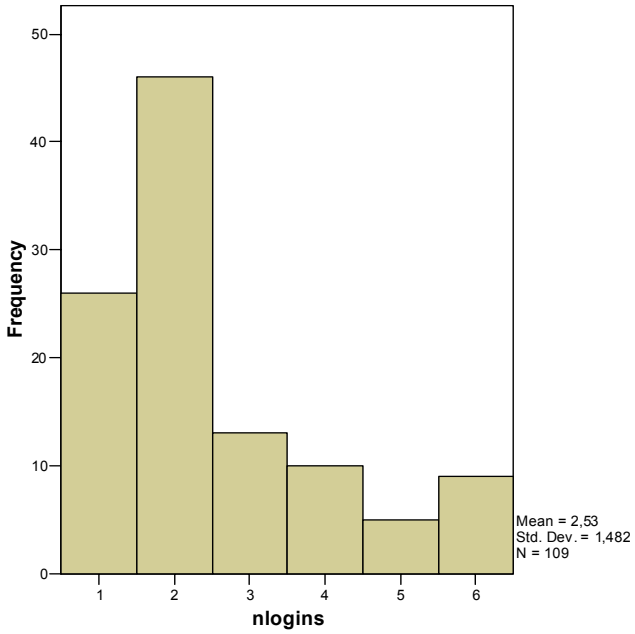


Figure 6-13 Histogram of the number of login groups

The number of login groups can only be used if there is no relationship between the intent of people to use the EPG and the number of logins; if such a relationship would exist, e.g. people who used the EPG more often have a higher intent to use the EPG in the future than people who used the EPG only a few times, it would bias the results. A correlation test shows that there is hardly any correlation between the number of login groups and intent ($r^2=0.000256$). The number of login groups (nlogins) are also fairly distributed over the six experimental groups; a chi-square test shows that there is a 76.5% probability that any difference occurred by coincidence ($\chi^2=19657;df=25;p=0.765$ two-tailed); i.e. it is safe to use the number of logins to weight the survey cases when testing the hypotheses and examining performance expectancy and effort expectancy.

6.5 Analysis of intent

The responses concerning intent have been summarized in *Table 6-9*. *Figure B-1* till *Figure B-11* in Appendix B show the histograms of intent for each of the six experimental groups, for combined non-personalized and personalized EPGs and for the combined EPGs for each structuring method. A quick look at the responses on intent for each of the experimental groups shows that the opinions of participants on their

intention to use their assigned EPG in the future varies a lot; e.g. although 18 people in experimental group 4 do really intend to use their EPG (intent 6 or 7), there are also 16 people who really do not intend to use that same EPG (intent 1 or 2).

Table 6-9 Cross-tab of intent and experimental group (weighted with nlogins)

		Experimental group						Total
		Without Predictions			With Predictions			
		Channel	Genre	Goal	Channel	Genre	Goal	
		Group 1	Group 2	Group 3	Group 4	Group 5	Group 6	
intent	1	13	11	3	3	7	2	39
	2	14	6	2	13	4	5	44
	3	2	7	5	6	8	4	32
	4	0	1	10	3	1	7	22
	5	2	9	17	10	6	1	45
	6	8	16	2	13	15	3	57
	7	7	0	3	5	6	16	37
Total		46	50	42	53	47	38	276

In order to test the hypotheses, it is necessary to determine whether there are significant shifts in intention between the various types of EPG. The means, standard deviation and confidence intervals provide a first indication of the influence of using predictions and the structuring method on intent. *Table 6-10* lists the means and standard deviation for each of the six EPGs, including the combined means and standard deviation for the personalized versus non-personalized EPGs and the three different structuring methods.

Table 6-10 Mean and standard deviation of intent for the various types of EPGs

Structuring	Predictions	Mean	Std. Deviation
Channel	No	3.35	2.368
	Yes	4.19	1.912
	Total	3.80	2.166
Genre	No	3.78	2.023
	Yes	4.36	2.090
	Total	4.06	2.066
Goal	No	4.29	1.470
	Yes	4.92	2.123
	Total	4.59	1.826
Total	No	3.79	2.023
	Yes	4.45	2.040
	Total	4.12	2.055

Figure 6-14 till *Figure 6-16* show the confidence intervals for each of the six experimental groups and the confidence intervals of intention for the three structuring methods and the personalized versus non-personalized EPGs. The means and confidence intervals indicate that using predictions and goal-based structuring have a positive influence on the intention to use a EPG. The detailed analysis of this influence is described in the next sections by testing the five hypotheses.

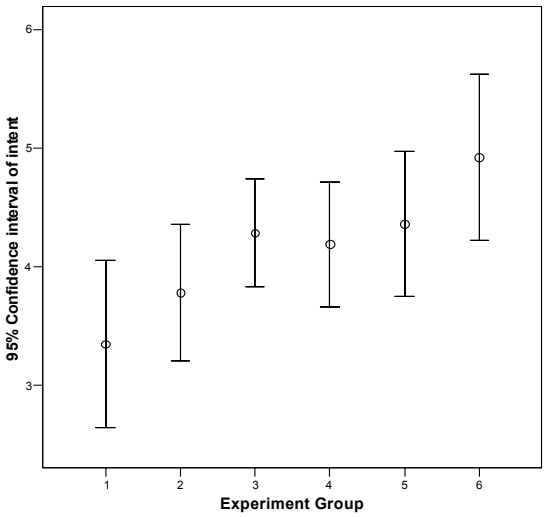


Figure 6-14 Confidence intervals of intent for each of the six experimental groups.

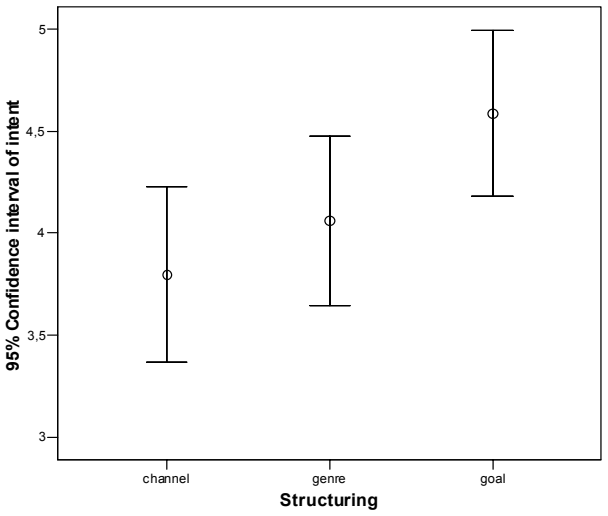


Figure 6-15 Confidence intervals of intent for the channel, genre and goal-based structured EPGs

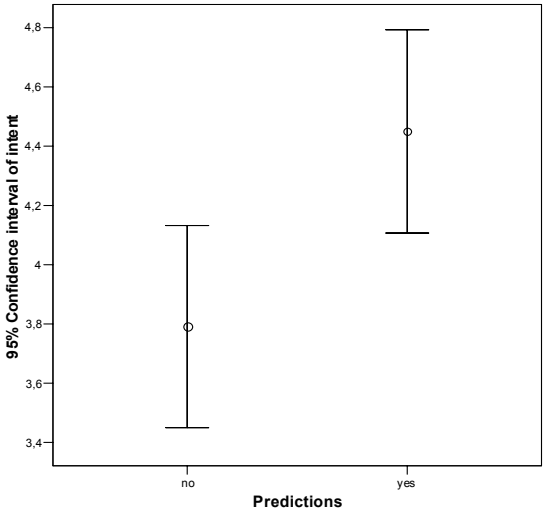


Figure 6-16 Confidence intervals of intent for the non-personalized and personalized EPGs

6.5.1 Hypothesis 1: Predictions

The first hypothesis states that using predictions for items makes it easier for users to find interesting items than using no predictions. As described in section 6.2.2, how easy it is to find interesting items is measured by the intention of users to use a specific EPG. To test this hypothesis, a comparison is made between the intention of users of non-personalized EPGs (independent of the structuring method), in which no predictions are used, and the intention of users of personalized EPGs (also independent of the structuring method), in which predictions are used. This hypothesis can be formally described as:

H_0: intent(predictions) = intent(no predictions)
H_a: intent(predictions) > intent(no predictions)

A detailed Mann-Whitney test shows that participants who used an EPG with predictions have a significantly higher intent (mean rank=152.19) to use that EPG in the future than those who used a EPG without predictions (mean rank=124.81); the probability that this increase occurred by coincidence is only 0.2% (U=7632.5;p=0.002 one-tailed).

An independent samples t-test also shows that using predictions has indeed a significant effect (t=-2.696; p=0.0035 one-tailed) on intent; *Figure 6-16* shows that participants who used a personalized EPG have a higher intent to keep using that EPG than participants who used a non-personalized EPG.

Both the parametric independent samples t-test as the non-parametric Mann-Whitney test reject H_0, hence confirming H_a: the intent of participants to use a personalized EPG is significantly higher than the intent of participants to use a non-personalized EPG. These results confirm the hypothesis that using predictions for items makes it easier for users to find interesting items in EPGs than using no predictions.

6.5.2 Hypothesis 2: Goal-based structuring

The second hypothesis states that structuring items based on the user's goals makes it easier for users to find interesting items than using structures that are not based on the user's goals. How easy it is to find interesting items is again measured by the intention of users to use a specific EPG. To test this hypothesis, a comparison is made between the intention of users of channel-based EPGs, genre-based EPGs (implicit goals) and goal-based EPGs (explicit goals) independently of whether predictions are used or not. The formal hypotheses tested are:

For implicit goals:
H_0: intent(genre-based) = intent(channel-based)
H_a: intent(genre-based) > intent(channel-based)

For explicit goals:
H_0: intent(goal-based) = intent(channel-based)
H_a: intent(goal-based) > intent(channel-based)

Another interesting test is to determine if the use of explicit goals (goal-based guide) has a significant influence on intent compared to using implicit goals (genre-based guide):

H_0: intent(goal-based) = intent(genre-based)
H_a: intent(goal-based) > intent(genre-based)

The first Mann-Whitney test between channel-based guides and genre-based guides shows that although the mean rank of intent of the genre-based guides (mean rank=101.36) is higher than the mean rank of the channel-based guides (mean rank=95.7), this difference is not statistically significant (U=4524.5;p=0.239 one-tailed). The probability that this difference occurred by coincidence is 23.9%. An independent sample t-test between channel-based and genre-based EPGs also shows that the increase in the mean of intent from channel-based guides (mean=3.80) to genre-based guides (mean=4.06) is not significant (t=-0.872;p=0.192 one-tailed). Neither the parametric nor the non-parametric test can reject H_0 for implicit goals; i.e. H_a has to be rejected, indicating that there is no

significant difference between the intent to use genre-based guides compared to the intent to use channel-based guides; i.e. implicit goal-based structuring does not significantly increase the intent to use an EPG over traditional channel-based structuring.

The second Mann-Whitney test between channel-based guides and goal-based guides shows that there is a significant difference between the two types of guide ($U=3128.5;p=0.0075$ one-tailed). As the mean rank of intent of the goal-based guides (mean rank=100.39) is higher than the mean rank of the channel-based guides (mean rank=81.60), there is a significant higher intent to use goal-based guides than channel-based guides. An independent samples t-test between channel-based guides and goal-based guides also shows that the increase in the mean of intent from channel-based guides (mean=4.06) to goal-based guides (mean=4.59) is significant ($t=-2.598;p=0.005$ one-tailed). Both the parametric as the non-parametric test reject H_0 for explicit goals, confirming H_a that the intent to use goal-based guides is significantly higher than the intent to use channel-based guides; i.e. explicit goal-based structuring does significantly increase the intent to use an EPG over traditional channel-based structuring.

The third Mann-Whitney test between genre-based guides and goal-based guides shows that although the mean rank of intent of the goal-based guides (mean rank=95.28) is higher than the mean rank of the genre-based guides (mean rank=83.82), this difference is not statistically significant ($U=3378.0;p=0.067$ one-tailed). However, according to an independent samples t-test between genre-based guides and goal-based guides, the increase in the mean of intent from genre-based guides (mean=4.06) to goal-based guides (mean=4.59) is significant ($t=-1.775;p=0.039$ one-tailed). As the non-parametric test cannot reject H_0, while the parametric test can, it is not possible to either confirm or accept H_a, which states that the mean of intent for goal-based guides is higher than for genre-based guides. This indicates that the intent to use genre-based guides is in between channel-based and goal-based guides, i.e. implicit goals are situated between using no goals and using goals explicitly. This can also be seen in *Figure 6-15*.

These results confirm the second hypothesis under a condition: structuring items based on the user's goals indeed makes it easier for users to find interesting items than using structures that are not based on the user's goals, but only when goals are used explicitly.

6.5.3 Hypothesis 3: Predictions and goal-based structuring

The third hypothesis states that using both predictions and structuring items based on the user's goals makes it easier for users to find interesting

items than using no predictions and no structures that are based on the user's goals. In this situation, non-personalized channel-based guides are compared with personalized goal-based guides; i.e. comparing experimental group 1 with experimental group 5 (implicit goals) and experimental group 6 (explicit goals). The formal hypotheses tested are:

For implicit goals:
H_0: intent(experimental group 5) = intent(experimental group 1)
H_a: intent(experimental group 5) > intent(experimental group 1)

For explicit goals:
H_0: intent(experimental group 6) = intent(experimental group 1)
H_a: intent(experimental group 6) > intent(experimental group 1)

A Mann-Whitney test shows that the increase in intent from experimental group 1 (mean rank=41.40) to experimental group 5 (mean rank=52.48) is significant (U=823.5;p=0.022 one-tailed). An independent samples t-test also shows that the means of intent between experimental group 1 (mean=3.35) and experimental group 5 (mean=4.36) is statistically different (t=-2.190;p=0.016 one-tailed).

A Mann-Whitney test between experimental group 1 and experimental group 6 shows that the increase in intent from experimental group 1 (mean rank=34.48) to experimental group 6 (mean rank=52.21) is also significant (U=505.0;p=0.0005 one-tailed). An independent samples t-test confirms that the increase in the mean of intent from experimental group 1 (mean=3.35) to experimental group 6 (mean=4.93) is statistically significant (t=-3.174; p=0.001 one-tailed).

Both the parametric tests as the non-parametric tests reject both H_0 hypotheses and thus confirming both H_a hypotheses, i.e. the intent of experimental group 5 and the intent of experimental group 6 to use the EPG are both significantly higher than the intent of experimental group 1. These results confirm the hypothesis that using both predictions and structuring items based on the user's goals (both implicit and explicit goal-based structuring) makes it easier for users to find interesting items than using no predictions and no structures that are based on the user's goals.

6.5.4 Hypothesis 4: Goal-based structuring over predictions

The fourth hypothesis states that using both predictions and structuring items based on the user's goals makes it easier for users to find interesting items than using only predictions. In this situation, personalized channel-based guides are compared with personalized goal-based guides; i.e. comparing experimental group 4 with experimental group 5 (implicit goals)

and experimental group 6 (explicit goals). The formal hypotheses tested are:

For implicit goals:
H_0: intent(experimental group 5) = intent(experimental group 4)
H_a: intent(experimental group 5) > intent(experimental group 4)

For explicit goals:
H_0: intent(experimental group 6) = intent(experimental group 4)
H_a: intent(experimental group 6) > intent(experimental group 4)

A Mann-Whitney test between experimental group 4 and experimental group 5 shows that there is no significant increase in intent from personalized channel-based guides (mean rank=48.92) to personalized genre-based guides (mean rank=52.29) ($U=1161.5;p=0.278$ one-tailed). Also the independent samples t-test between the intent of experimental group 4 (mean=4.19) and experimental group 5 (mean=4.36) shows no significant increase in intent ($t=-0.432;p=0.333$ one-tailed).

A Mann-Whitney test between experimental group 4 and experimental group 6 shows that there is a significant increase in intent from personalized channel-based guides (mean rank=41.35) to personalized goal-based guides (mean rank=52.49) ($U=760.5;p=0.022$ one-tailed). Also the independent samples t-test between experimental group 4 and experimental group 6 shows a significant increase in intent from experimental group 4 (mean=4.19) to experimental group 6 (mean=4.92) ($t=-1.721;p=0.045$ one-tailed).

For implicit goals, these results confirm H_0 and reject H_a: there is no significant increase in intent from a personalized channel-based guide to a personalized genre-based guide. For explicit goals, the results reject H_0, confirming H_a that there is a significant increase in intent from personalized channel-based guides to personalized goal-based guides. This confirms the hypothesis that using predictions in combination with structuring items based on the user's goals makes it easier for users to find interesting items than using only predictions, but only when goals are used explicitly.

6.5.5 Hypothesis 5: Predictions over goal-based structuring

The fifth hypothesis states that using both predictions and structuring items based on the user's goals makes it easier for users to find interesting items than using only structures that are based on the user's goals. For this hypothesis, non-personalized genre-based and non-personalized goal-based guides are compared with personalized genre-based and personalized goal-based guides; i.e. comparing experimental group 2 with experimental group

5 (implicit goals) and experimental group 3 with experimental group 6 (explicit goals). The formal hypotheses tested are:

For implicit goals:
H_0: intent(experimental group 5) = intent(experimental group 2)
H_a: intent(experimental group 5) > intent(experimental group 2)

For explicit goals:
H_0: intent(experimental group 6) = intent(experimental group 3)
H_a: intent(experimental group 6) > intent(experimental group 3)

A Mann-Whitney test between experimental group 2 and experimental group 5 shows that there is no significant increase in intent from non-personalized genre-based guides (mean rank=44.70) to personalized genre-based guides (mean rank=53.57) (U=960.0;p=0.056 one-tailed). Also an independent samples t-test between experimental group 2 (mean=3.78) and experimental group 5 (mean=4.36) shows no significant increase in intent (t=-1.393;p=0.0835 one-tailed).

A Mann-Whitney test shows that the increase in intent from non-personalized goal-based guides (mean rank=36.92) to personalized goal-based guides (mean rank=44.46) is not significant (U=647.5;p=0.070 one-tailed). Also the independent samples t-test between experimental group 3 (mean=4.29) and experimental group 6 (mean=4.92) shows no significant increase in intent (t=-1.568;p=0.0605 one-tailed).

The parametric and non-parametric tests confirm both H_0 hypotheses and thus reject both H_a hypotheses, i.e. there is no significant increase in intent from non-personalized genre-based guides to personalized genre-based and from non-personalized goal-based guides to personalized goal-based guides. This shows that adding predictions to an implicit goal-based guide or explicit goal-based guide does not significantly increase the intention of usage. As adding explicit goal-based structuring to both non-personalized and personalized channel-based guides does increase the intention of usage significantly (see sections 6.5.3 and 6.5.4), it can be concluded that adding goal-based structuring to an EPG has a greater influence on the intention of usage than adding predictions. Only adding predictions to a non-personalized channel-based guide significantly increases the intent of usage (Mann-Whitney U=936.0; p=0.0215 one-tailed and an independent samples t-test t=-1.954; p=0.027 one-tailed).

This rejects the hypothesis that using both predictions and structuring items based on the user's goals makes it easier for users to find interesting items than using only structures that are based on the user's goals.

6.6 Effort expectancy and performance expectancy

Intention has been used to measure "how easy it is to find interesting items", which is based on the unified theory of acceptance and use of technology (UTAUT) (Venkatesh et al., 2003) (see section 6.2.2). This theory also states that performance expectancy and effort expectancy influence the intention of accepting technology. Performance expectancy is defined as "the degree to which an individual believes that using the system will help him or her attain gains in job performance" (Venkatesh et al., 2003, page 447) while effort expectancy is defined as "the degree of ease associated with the use of the system" (Venkatesh et al., 2003, page 450).

In the experiment, performance expectancy and effort expectancy have also been measured explicitly; this not only allows the verification of the stated influence of performance expectancy and effort expectancy on intention as stated in UTAUT, it also provides more insights in the gains and efforts people expect from predictions and goal-based structuring.

Performance expectancy and effort expectancy have been measured by asking participants how much they agreed to eights statements using the same seven-point scale from 'totally disagree' to 'totally agree' as has been used to measure intention; these are the eight (two groups of four) statements as mentioned in section 6.2.2.

6.6.1 Relationship of performance expectancy and effort expectancy with intent

The first verification of the UTAUT theory is to validate whether the four measures for performance expectancy and the four measures for effort expectancy do indeed each measure one concept; respectively performance expectancy and effort expectancy. A factor analysis on both sets of four measures shows that each set indeed measures one concept: 76% of the variance between the four measures of performance expectancy is explained by one component, while 79% of the variance between the four measures of effort expectancy is explained by one component.

The second verification concerns the stated relationship of performance expectancy and effort expectancy with intent. Regression analysis shows that the relationship between the set of four measures for performance expectancy and the set of four measures for effort expectancy explains 64.0% of the variance of intent ($r^2 = 0.640$). An ANOVA analysis also confirms that the linear combination of performance expectancy and effort expectancy significantly ($p = 0.000$) explains the variance of intent. Both results confirm the relationship of performance expectancy and effort expectancy with intent as described by UTAUT. This relationship means that by analyzing the responses concerning performance expectancy and

effort expectancy, better insight can be acquired about the reasons for participants' responses on intent.

6.6.2 Performance expectancy

The first statement concerning performance expectancy is the most related to the five hypotheses as it directly asks how people think that the EPG helps them in finding interesting TV programs. When examining the influence of the three structuring methods on how people think that the EPG helps them in finding interesting TV programs, there is only a significant increase between the channel-based guides and goal-based guides, not between the channel-based and genre-based guides or between the genre-based and goal-based guides (see *Figure 6-17*). A Mann-Whitney test shows that the increase from channel-based guides to goal-based guides is significant (U=3080.0;p=0.009 two-tailed) as well as an independent samples t-test (t=-2.511;p=0.013 two-tailed).

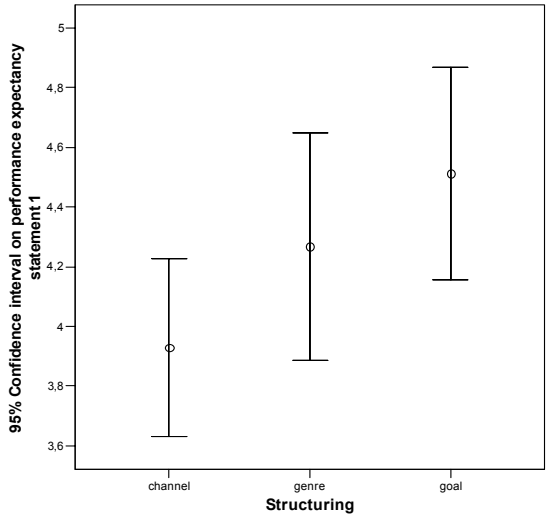


Figure 6-17 Confidence intervals per structuring method for the first statement concerning performance expectancy

When examining the influence of using predictions on how people think that the EPG helps them in finding interesting and fun TV programs, there is no significant difference between using predictions or not (see *Figure 6-18*); nor the Mann-Whitney test nor the independent samples t-test shows any significant difference.

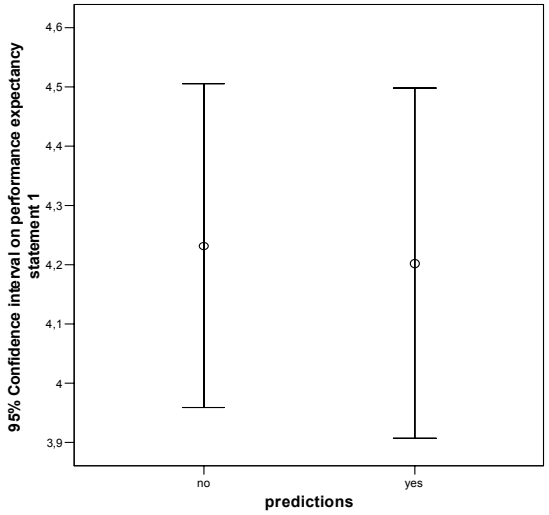


Figure 6-18 Confidence intervals for personalized versus non-personalized guides for the first statement concerning performance expectancy

A detailed analysis between the six experimental groups (see *Figure 6-19*) shows that there is a significant increase on how people think that the EPG helps them in finding interesting and fun TV programs from both a personalized and a non-personalized channel-based guide (experimental group 1 and experimental group 4) to a personalized goal-based guide (experimental group 6). A Mann-Whitney test shows that the increase between experimental group 1 and experimental group 6 is significant (U=584.5;p=0.007 two-tailed), which is confirmed by an independent samples t-test (t=-2.406;p=0.018 two-tailed). Similar results are found between experimental group 4 and experiment 6: Mann-Whitney U=759.0;p=0.037 two-tailed and independent samples t-test t=-2.191; p=0.031 two-tailed.

An interesting observation is that the increase between experimental group 1 (non-personalized channel-based guide) and experimental group 3 (non-personalized goal-based guide) is statistically not significant (Mann-Whitney U=790.0;p=0.133 two-tailed and independent samples t-test t=-1.329;p=0.187 two-tailed), even though the increase in intent between these two groups is significant (Mann-Whitney U=735.0;p=0.050 two-tailed and independent samples t-test t=-2.207; p=0.030 two-tailed). This illustrates that intent measures more than just performance expectancy.

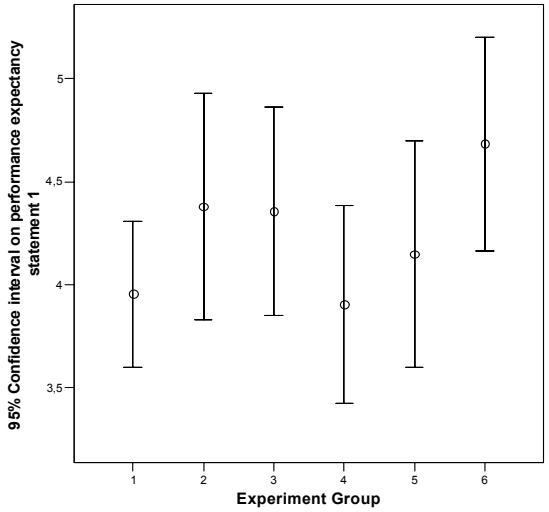


Figure 6-19 Confidence intervals for the experimental groups for the first statement concerning performance expectancy

When examining the other three statements concerning performance expectancy, no significant influence of the structuring method or using predictions on performance expectancy is found. For the second variable, this indicates that although people do believe that goal-based structuring helps them in finding interesting TV programs (especially when combined with personalization), they do not believe that this will help them find these programs any faster. Notice, this statement measures expected and subjective speed, which is something different than objective speed. Measuring objective speed requires controlled laboratory experiments, which is subject for future research.

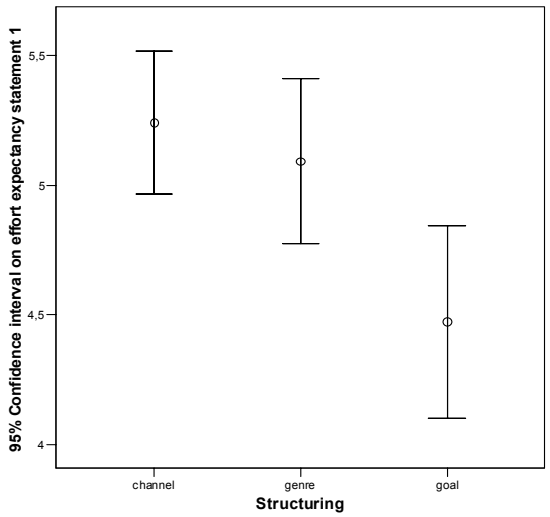
The final two statements used to measure performance expectancy both measure an indirect consequence of using an EPG, namely TV watching experience, i.e. watching less disappointing programs and having a fun time watching TV. No significant differences have been found for these statements, which might be attributed to the fact that TV watching experience is influenced by more factors than the EPG only, e.g. a TV has to be shared with other family members who also influence what is going to be watched, or other activities influence what is actually watched and how much it is enjoyed.

Concluding performance expectancy, people do believe that goal-based structuring helps them in finding interesting TV programs (especially when combined with the use of predictions) but they do not believe that this will help them find these programs any faster or that it will actually influence their TV watching experience.

6.6.3 Effort expectancy

When examining the influence of the three structuring methods on how much effort people believe it will take them to learn the possibilities of the EPG[6], there is no significant difference between channel-based and genre-based guides (see *Figure 6-20*). A Mann-Whitney test shows that the decreases in effort expectancy between channel-based and goal-based guides is significant (U=2937.0;p=0.002 two-tailed), which is confirmed by an independent samples t-test (t=3.370;p=0.001 two-tailed). A Mann-Whitney test between genre-based and goal-based guides shows that this decrease in effort expectancy is also significant (U=3035.0;p=0.010 two-tailed), which is confirmed by an independent samples t-test (t=2.528;p=0.012 two-tailed). These decreases mean that people believe it takes more effort to learn goal-based guides than channel or genre-based guides; i.e. explicit goal-based structuring takes more effort to learn than non-goal based structuring or implicit goal-based structuring.

Figure 6-20 Confidence intervals per structuring method for the first statement of effort expectancy



When examining the influence of using predictions on how much effort people believe it will take them to learn the possibilities of the EPG, there is no significant difference between personalized guides and non-personalized guides (see *Figure 6-21*); i.e. people believe that it does not take more effort to learn personalized guides than non-personalized guides.

[6] The first and last statements concerning effort expectancy are about learning: learning the possibilities and learning to use the EPGs. As both variables show similar results, only the results of the first statement are reported.

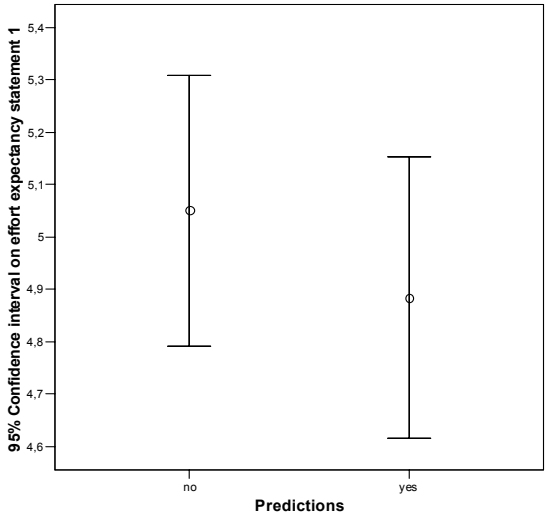


Figure 6-21 Confidence intervals for personalized versus non-personalized guides for the first statement of effort expectancy

The observation that goal-based structuring takes more effort to learn and that using predictions does not influence the learning effort is also confirmed when examining the differences in effort expectancy between the six experimental groups separately (see *Figure 6-22*). Between the experimental groups there are significant differences between experimental group 1 and experimental group 6 (Mann-Whitney U=532.0;p=0.001 two-tailed and independent samples t-test t=3.180;p=0.002 two-tailed), between experimental group 2 and experimental group 6 (Mann-Whitney U=602.5;p=0.003 two-tailed and independent samples t-test t=3.127;p=0.002 two-tailed) and between experimental group 4 and experimental group 6 (Mann-Whitney U=615.5;p=0.001 two-tailed and independent samples t-test t=3.148;p=0.002 two-tailed). A Mann-Whitney test also shows a significant difference between experimental group 5 and experimental group 6 (U=616.0;p=0.011 two-tailed), which is not confirmed by an independent samples t-test (t=1.882;p=0.063 two-tailed). These results confirm that personalized goal-based guides take more effort to learn than other personalized and non-personalized guides (except for a non-personalized goal-based guide).

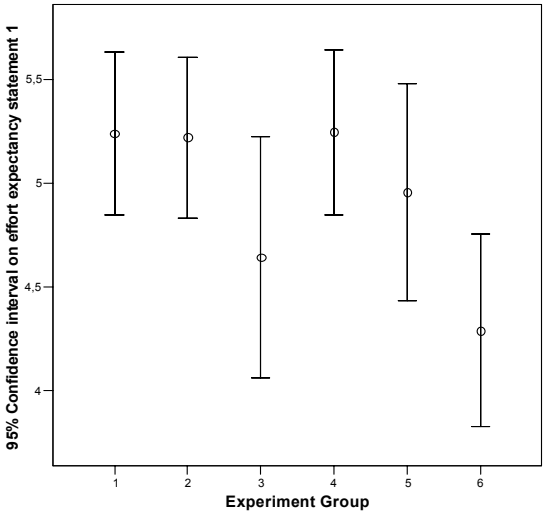


Figure 6-22 Confidence intervals per experimental group for the first statement of effort expectancy

The second statement for effort expectancy measures whether people believe that the interaction with the EPG is clear and understandable. Although the means of effort expectancy concerning a clear and understandable interaction show a slight decrease over the types of structuring (see *Figure 6-23*), there is no significant difference between any of the structuring methods, i.e. the type of structuring does not influence how clear and understandable an EPG is.

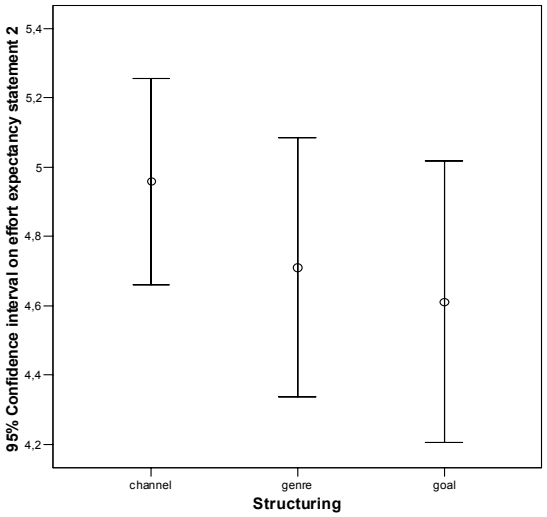


Figure 6-23 Confidence intervals per structuring method for the second statement of effort expectancy

When investigating the influence of using predictions on whether the interaction with the EPG is clear and understandable, both a Mann-

Whitney test and an independent samples t-test show that there is a significant difference between non-personalized and personalized EPGs (Mann-Whitney U=8031.0;p=0.021 two-tailed and independent samples t-test t=-2.432;p=0.016 two-tailed); i.e. people believe that the interaction with personalized EPGs is clearer and better understandable than with non-personalized EPGs (see also *Figure 6-24*).

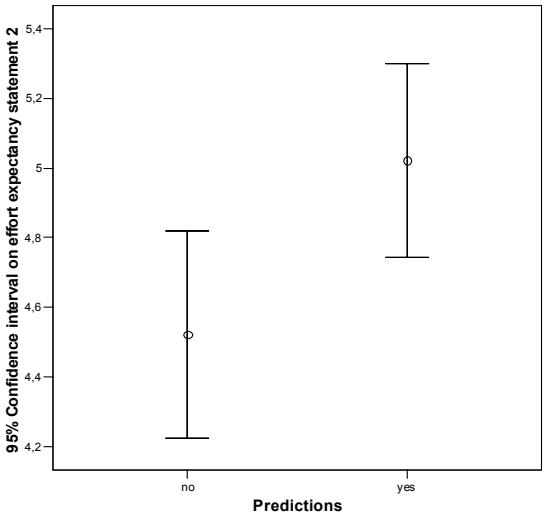


Figure 6-24 Confidence intervals for personalized versus non-personalized guides for the second statement of effort expectancy

When examining the differences between the six experimental groups in detail (see *Figure 6-25*), an interaction can be found between using predictions and goal-based structuring. A personalized goal-based EPG (experimental group 6) is significantly less clear and understandable than a personalized channel-based EPG (experimental group 4) (Mann-Whitney U=764.5;p=0.042 two-tailed and independent samples t-test t=2.339;p=0.022 two-tailed) and a personalized genre-based EPG (experimental group 5) (only significant using a independent samples t-test t=2.445;p=0.017 two-tailed not using a Mann-Whitney test U=686.5;p=0.060 two-tailed). This interaction between goal-based structuring and using predictions is also apparent from an ANOVA analysis. Structuring alone does not have a significant influence on explaining the variation of effort expectancy concerning an EPG being clear and understandable (F=0.929;df=2;p=0.396) while using predictions does (F=4.495;df=1;p=0.035) and the combination of predictions and structuring even more (F=4.910;df=2;P=0.008); i.e. although using predictions does make the interaction clearer and better understandable,

goal-based structuring combined with predictions makes the interaction less clear and more difficult to understand.

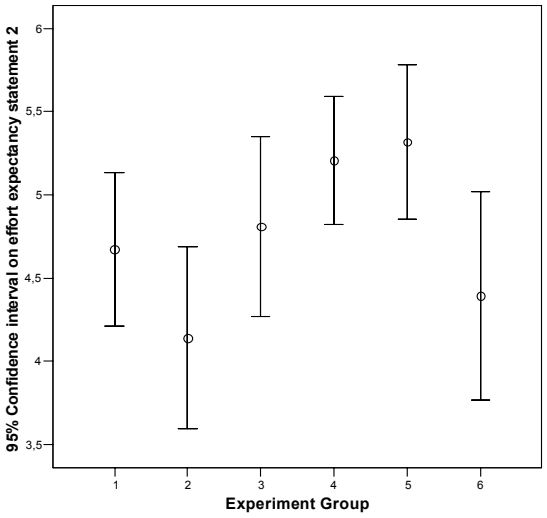


Figure 6-25 Confidence intervals per experimental group for the second statement of effort expectancy

The final aspect (third statement) of effort expectancy is ease-of-use. All tests show that there is no significant difference between the different structuring methods, between personalized or non-personalized guides or between any of the experimental groups (see *Figure 6-26*). There only appears to be a significant difference between experimental group 3 (non-personalized goal-based guide) and experimental group 5 (personalized genre-based guide) and only on according to a Mann-Whitney test ($U=747.5;p=0.043$ two-tailed) while an independent samples t-test shows no significant difference ($t=-1.263;p=0.210$ two-tailed); however, these two experimental groups have no direct relation concerning structuring or the use of predictions. This shows that neither structuring nor using predictions has any effect on the ease-of-use of EPGs.

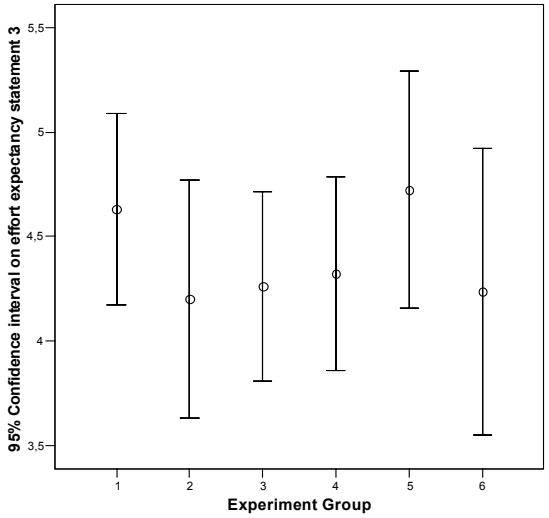


Figure 6-26 Confidence intervals per experimental group on the third statement of effort expectancy

Concluding effort expectancy, people believe it takes more effort to learn explicitly goal-based structured EPGs than channel or implicitly goal-based EPGs independently of whether predictions are used or not. However, people believe that using predictions makes the interaction clearer and better understandable, except when explicit goal-based structuring is combined with predictions, which makes the interaction less clear and more difficult to understand; the latter can probably be attributed to the learning curve of explicit goal-based guides. People also believe that neither the various types of structuring nor using predictions has any effect on the ease-of-use of an EPG.

6.7 Conclusions

In this chapter, a structuring method has been described that complements the Duine prediction framework and which provides support for users in finding interesting items also taking into account their interests at a specific moment: a goal-based structuring method. Not only has this structuring method been described in detail, it has also been tested whether goal-based structuring actually helps people in finding interesting information easier using an experiment with an electronic TV program guide. The experiment resulted in the following confirmation and rejection of the five hypotheses concerning the use of predictions and goal-based structuring in EPGs:

– *Confirmed hypothesis 1:* Using predictions for items makes it easier for users to find interesting items than using no predictions.

- *Confirmed hypothesis 2:* Structuring items based on the user's goals makes it easier for users to find interesting items than using structures that are not based on the user's goals, especially when goals are used explicitly.
- *Confirmed hypothesis 3:* Using both predictions and structuring items based on the user's goals makes it easier for users to find interesting items than using no predictions and no structures that are based on the user's goals.
- *Confirmed hypothesis 4:* Using predictions in combination with structuring items based on the user's goals makes it easier for users to find interesting items than using only predictions, but only when using goals explicitly.
- *Rejected hypothesis 5:* Using both predictions and structuring items based on the user's goals does not make it easier (nor harder) for users to find interesting items than using only structures that are based on the user's goals.

From these results, one can conclude that structuring EPGs using a goal-based structuring method makes it easier for users to find interesting items, especially if the goals are used explicitly; this is independent of whether predictions are used or not. Predictions on their own will only make it easier for people to find interesting items when they are added to a channel-based non-personalized EPG; however, adding predictions to goal-based EPGs (either implicitly or explicitly goal-based) does not have a significant influence (neither positive nor negative) on how easy it is for people to find interesting items. The latter may be caused by the fact that in the current TV guide only a limited number of programs are associated with each goal, reducing the need for additional support in the form of predictions.

The analysis of the effect of using predictions and goal-based structuring on performance expectancy and effort expectancy shows that goal-based structuring helps people to better find interesting TV programs to watch, but it will not influence how fast people expect to find these items nor will it influence people's TV watching experience. Goal-based structuring has a higher learning curve than non goal-based structures. We believe this can be attributed to the fact that people are forced to make their goals for watching TV explicit, which is something that most people are not used to when watching TV. In other domains, e.g. buying an electronic product, making ones goals explicit is more common, which may reduce the need for people to get used to goal-based structuring.

Furthermore, interaction with a personalized EPG is clearer and better to understand than with a non-personalized EPG, except when combined with goal-based structuring, which can be contributed to the learning curve of goal-based structuring. Goal-based structuring and predictions have no

effect on the ease-of-use of EPGs. These conclusions are all under the condition that people are willing to learn to use new types of EPGs.

However, due to the large diversity in opinions about the intent to use each of the different types of EPGs with or without predictions, we believe that it is wise to give people a choice in what structuring method to use (or allow them to switch between methods) and whether to use predictions or not. This allows people to use that structuring method that best suites their personal preferences and needs and allows people to get used to goal-based structuring and predictions at their own pace and decide if and when to switch to goal-based structuring.

As both parametric and non-parametric tests led to the same results, the conclusions drawn from the experiment are valid independently of whether the used measuring scale is interpreted as ordinal or interval. However, looking back, it would have been better if the measuring scale would have been designed more clearly as being either ordinal or interval; future experiments should make this distinction more clearly during the design of the experiment.

The results also provide support for the unified theory of acceptance and use of technology. The set of statements used to measure performance expectancy really does measure one concept; the same has been shown for the set of statements that measure effort expectancy. Furthermore, the relationship of performance expectancy and effort expectancy with intent is also confirmed.

The results of the experiment also support our model that combines the means-end approach and the uses and gratification theory. The explicit use of gratifications in EPGs resulted in a higher intention to use the EPG than using attributes of TV programs such as channel and genre; i.e. people indeed make decisions based upon the expected gratifications of watching a TV program and how these gratifications match their goals. By making these gratifications explicit, people are better supported in finding those TV programs that are of interest to them. The results even show that structuring on gratifications has a greater effect on helping people to find interesting items than using recommendations in the form of predictions.

As the mapping from an attribute of a TV program (the main genre) to gratifications formed the basis to assign TV programs to gratifications, and structuring on gratifications resulted in a higher intent to use the EPG, the stated linkage between attributes and consequences in the means-end approach is also confirmed. We believe that future research in recommender systems should focus more on understanding the linkage between item attributes and the gratifications they have for a user than on trying to optimize algorithms that try to predict how interesting an item will be for a user based on their short- and long term interests in the form of predicted ratings; people are better supported by a recommender that is

capable of successfully determining what gratifications a certain item will give to an individual user. We expect that this linkage is more detailed than just a mapping based on genres; e.g. a comedy seems to naturally belong to the mood improvement gratification; however, there are different types of comedy, such as British and American comedies; for someone an American comedy may lead to an improvement in his mood, while a British comedy does not; furthermore, even within American comedies there may be comedies that that person does not like; e.g. an actor that a person does not like stars in that comedy. These examples illustrate that a linkage from an item to a gratification is more fine-grained than just a mapping on genres.

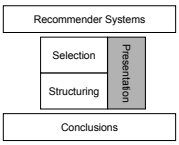
Understanding this linkage can also help recommenders with explaining their recommendations (Herlocker et al., 2000). Most explanation methods try to translate algorithmic aspects into user understandable explanations, e.g. "there are 75 users with a similar taste in TV programs who also liked this item" or "this TV program is similar to program x and z that you also liked". Ardissono et al. (2003) explicitly use item attributes when creating explanations in their INTRIGUE system to create better understandable explanations; they use those item attributes that have a significant meaning to the user. Understanding the linkage between attributes and gratifications may even provide additional support in explaining recommendations, e.g. "This TV program will improve your mood as it is an American comedy; furthermore there were 75 users with a similar taste who also liked this TV program."

Based on these results, we conclude that using goal-based structuring is extremely important to support users in finding interesting items, even more important than recommendations in the form of predictions.

This chapter addressed the structuring process of a personalized information system. The final process of personalized information systems, presenting the set of information and predictions, is addressed in the next chapter.

Chapter 7

Presenting Predictions



In order for information systems to better support people in finding interesting items, they have to learn about their users and adapt their behaviour to each individual user, so-called personalized information systems. So far, this thesis dealt with solutions to help people find interesting items by researching and developing solutions for two of the three main processes in personalized information systems, where this thesis focused on recommender systems as a specific type of personalized information system: selection and structuring. For the selection process, this resulted in a domain-independent framework to develop prediction engines for recommender systems that can be tailored to specific domains. The prediction engines developed with this framework provide more accurate predictions than any other prediction technique by using prediction strategies that switch between multiple prediction techniques. For the structuring process, a goal-based structuring method has been developed that structures items based on the goals that people can have for items. Experiments with the goal-based structuring method have shown that structuring items based on the user's goals indeed help people to find interesting items, even more than using recommendations in the form of predictions.

This chapter deals with the third main process of a personalized information system: the presentation of selected and structured items, which is part of the field of user interfaces and interaction design. This chapter only focuses on those aspects that are specific to the presentation of recommendations in the form of predictions and how users interact with recommendations and predictions. Where the previous chapters provided more generic solutions for building prediction engines and for structuring items, this chapter is more tightly coupled with the domain in which much of our research took place, namely the domain of electronic TV guides. This coupling is a direct result of the process of user interface and interaction design, which always has a tight coupling with the application for which a

user interface is designed. However, where possible, generalization of results is discussed.

Of the generic predictor model as described in chapter 3, there are three aspects that are part of the interaction with the user, namely predictions, explanations and feedback (see *Figure 7-1*); user profiles and the validity indicators are internal parts of a recommender (though they can be influenced by the feedback of the user); information is of course presented to the user, however, this is not specific to recommender systems.

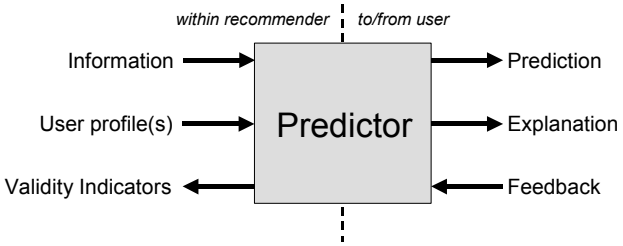


Figure 7-1 Parts of the generic predictor model that is part of the user interaction and specific to recommender systems.

The focus lies on investigating how people prefer predictions and explanations to be presented and how they prefer to provide feedback to the recommender system. The interface aspects of recommender systems have been examined in the domain of electronic program guides for TV, just like predicting user interests and structuring items.

An iterative user interface design process has been employed (see section 7.1), which started with an analysis of users, tasks and existing recommender systems (section 7.2), followed by a brainstorm and interactive design session with users (section 7.3). Based on the results of these sessions, an online survey has been conducted to investigate preferences of people for specific user interface elements (section 7.4), resulting in a first prototype. Usability experts evaluated this prototype using heuristic evaluation (section 7.5). The redesigned prototype has been tested with users in usability testing rounds (section 7.6), resulting in the final prototype. This chapter concludes (section 7.7) with how the results of this research have been used in the EPG experiment described in the previous chapter.

Parts of this chapter have also been published in Barneveld & van Setten (2004).

7.1 Designing a usable interface for predictions

7.1.1 Iterative design

In the past, software design and user interfaces were driven by the new technologies of the time. This is called system- or technology-driven design. Users were not taken into account much in the design. They were given software functions with whatever interface developers were able to come up with.

However, research has shown that it is very important to actually consult users or to involve them in the design process rather than designing for a fictitious user. As Spolsky (2001) puts it: "At a superficial level we may think we're designing for users, but no matter how hard we try, we're designing for who we think the user is, and that means, sadly, that we're designing for ourselves…" In the early 1980s, focus therefore shifted towards user-centred design (Norman & Draper, 1986), in which the usability for end-users is a primary design goal. Designing usable products usually involves four main phases (Faulkner, 2000; Nielsen, 1993):

- Analysis of tasks and users.
- Usability specification in which a number of (measurable) goals are identified.
- The actual design of the product.
- Evaluation of the usability of the design.

To obtain the highest possible level of usability, design and evaluation usually take place iteratively. The purpose of reiteration is to overcome the inherent problems of being unable to fully capture all requirements in all details by cycling through several designs, incrementally improving upon the current product with each pass (Dix, Finlay, Abowd, & Beale, 1998). Tognazzini (2000) states that iterative design, with its repeating cycle of design and testing, is the only validated method in existence that will consistently produce successful results, i.e. usable interfaces. Iterative testing is necessary because one cannot always be certain that modifications will actually improve the usability of a product. Changes can sometimes introduce new problems, which can only be detected by retesting (Lindgaard, 1994; Nielsen, 1993).

While user-centred design puts users as the focus of design considerations, their role was still quite passive, namely that of a target for user task analysis and requirements gathering. Following the "Scandinavian" approach to software systems design (Floyd, Mehl, Reisin, Schmidt & Wolf, 1989; Ehn, 1992), part of the human-computer interaction community recently moved to a new framework called participatory design (Muller &

Kuhn, 1993). In this, users are considered to be active participants and partners in the design process (Mandel, 1997).

In order to acquire proper understanding of their wishes and demands as well as a feeling of what their ideas are for our TV recommender systems' user interface, users have been involved especially in the first phases of the design process, resulting in rough designs by users and detailed opinions on interface elements. In the later phases, users have been involved in the validation of the detailed designs, but not in the design process itself as full participatory design can be very costly and time consuming; it also asks a lot from the users involved.

7.1.2 The user interface design process

Several design and evaluation techniques can be used during the cycles of an iterative user interface design process. Depending on the iteration phase, some techniques are more suitable than others. Techniques such as brainstorming and interactive design sessions are well suited for gaining global insight into the wishes, demands and ideas of the target users in early stages of the design phase. At intermediate stages, techniques focused on specific details and design questions (such as surveys) are more suitable. Techniques that evaluate the complete design come into play during the last stages of the design process. The iterative nature of the entire process makes it possible to return to techniques previously used in order to re-investigate design decisions. The user interface design process for this research consisted of the following activities:

- Analysis of the tasks, users and interfaces of existing systems (see section 7.2).
- A brainstorming session was organized with various TV viewers to explore their expectations of a user interface of a TV recommender system, and an interactive design session was held resulting in a number of crude mock-ups created by the TV viewers themselves (section 7.3).
- An interactive online survey was conducted among a group of users to investigate various widgets for visualizing the three user interface aspects (section 7.4). Based on the brainstorming results, the interactive design session and the survey, an initial prototype was developed for the TV recommender systems' user interface.
- Using heuristic evaluation methods, the first prototype was evaluated together with usability experts (section 7.5). The prototype was improved based on the results of this evaluation.
- Various iterations of usability tests were conducted with users (section 7.6).
- Based on the results of the various design and evaluation steps, a final prototype was developed (section 7.6.3).

7.2 Analysis

A user interface design process starts with a thorough analysis to define the tasks that need to be facilitated, the users and what users want and need. Two approaches have been employed: a formal task and user analysis (Dix et al., 1998; Lindgaard, 1994; Nielsen, 1993) and an analysis of existing EPGs, TV systems and other recommender systems. Results of the task analysis are not discussed here in more detail as they are reflected in the three aspects of recommender systems that were selected based on the results of the task analysis: presenting predictions, presenting explanations and acquiring feedback.

7.2.1 User analysis

A typical user of a TV recommender system is familiar with the concept of colour television and knows how to operate a television with a remote control. Our target group consists of users between roughly 15 and 60 years of age; interfaces for children need to take their different needs and behaviours into account, while older people may have difficulties dealing with the new technologies that TV recommender systems are based on; separate research is therefore necessary to determine good user interfaces for TV recommender systems for children and elderly. Because TV is used by a wide variety of people with various backgrounds, it must be possible for a wide variety of users with varying levels of education and experience to use the interface of a TV recommender system.

7.2.2 Existing systems

Because several EPGs, interactive TV systems and other recommender systems already exist, it was not necessary to start our design from scratch. The systems examined include: omroep.nl (www.omroep.nl), tvgids.nl (www.tvgids.nl), DirectTV (www.directtv.com), YourTV (www.yourtv.com.au), PTVplus (www.ptvplus.com), Tivo (www.tivo.com), TVScout (Baudisch & Brueckner, 2002), a prototype EPG by Philips (Gutta et al., 2000; Zimmerman & Kurapati, 2002), Sony EPG (www.sony.co.uk/digitaltelevision/products), Movielens (movielens.umn.edu), Netflix (www.netflix.com), Amazon (www.amazon.com), Libra (www.cs.utexas.edu/users/libra), Yahoo Launch (launch.yahoo.com), Jester (shadow.ieor.berkeley.edu/humor), Epinions (www.epinions.com) and imdb (www.imbd.com).

Based on the analysis of these systems, a set of factors have been identified that appear to influence the design of the three interface aspects of a recommender system. For the presentation of a prediction, these factors are:

- *Presentation form*: this is the visual concept used to present a prediction; e.g. the use of a bar, a number of symbols or a numerical score.
- *The scale of the prediction*: Continuous versus discrete, range (e.g. 1 to 5 or 0 to 10), precision (e.g. {1, 2, 3, 4, 5} or {1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5}), and symmetric versus asymmetric (e.g. -2 to 2 versus 1 to 5).
- *Visual symmetry or asymmetry*: even though a scale may be symmetric, a prediction can still be presented asymmetrically; e.g. a scale of -2 to 2 can be presented by five thumbs, with the third thumb representing the neutral value zero.
- *Use of colour to represent a prediction*: some systems use different colours to distinguish between lower and higher predicted ratings.

The factors for user feedback are the same as for predictions, with two additions:

- *Scale used for prediction and feedback*: Is the scale used for feedback the same as that for presenting predictions?
- *Integration of prediction and feedback*: To what extent is the presentation of feedback integrated with the presentation of predictions?

Identified factors for explanations are:

- *Level of detail*: how detailed is an explanation, e.g. is it only coarse or does it include a lot of examples and detailed descriptions of the reasoning?
- *System transparency*: does an explanation reflect the internal working of a predictor?
- *Modality*: what modalities are used to present explanations? E.g. text, graphs, tables, images, spoken language.
- *Integration with predictions*: is an explanation presented directly with a prediction or must the user specifically ask for an explanation?

The preferences of TV viewers regarding these three main aspects of a TV recommender system's user interface and their different impact factors have been investigated. First, a brainstorming session was organized followed by an interactive design session.

7.3 Brainstorming and interactive design sessions

The purpose of the brainstorm session was to explore users' basic expectations for user interfaces of TV recommender systems.

7.3.1 Approach

Potential users with no specific knowledge of recommender systems were invited to participate in the brainstorm and interactive design session. A total of 19 people participated in two sessions. The group consisted of 9 males and 10 females with various backgrounds, between the ages of 20 and 56. All people participated on a voluntary basis. To ensure that older people with relatively little knowledge of new computing applications would not be intimidated by younger people with more technical experience, the session was divided into two separate groups: one for younger participants and one for participants older than 45. The same approach for identifying user expectations was used in both groups.

In order to ensure that the participants would not be influenced, they did not receive any special instructions about recommender systems, except for an introductory general explanation about such systems. None of the results of the analysis as described in section 7.2 were provided to any of the participants beforehand. The session started with a brainstorming phase on the user interface. Ideas on the three main topics were generated, written down, and posted visibly for every participant. These ideas were then clustered to get a better overview. After a short discussion on the various ideas during which new ideas could still be added, groups of three to four participants were formed for the design session. Each group was asked to design and present a mock-up TV recommender systems' user interface, based on the ideas from the brainstorming session that they liked most. At the very least, the user interface had to be able to present a set of recommendations, give users a way to provide feedback on recommendations, and allow them to obtain an explanation of why a certain recommendation was made.

7.3.2 Results of the brainstorming session

The initial brainstorming resulted in a broad collection of ideas and suggestions for TV recommender systems' user interfaces. The results have been grouped based on the three main user interface aspects of recommender systems; a group for ideas that went beyond these three aspects has also been added. As expected, the ideas and comments resulting from this session were rather broad:

- *Predictions*: The user should always be in full control. If desired, the user should be able to turn off recommendations. The user should have influence on a range of settings, including the level of personalization, the number of recommendations, and their level of detail.
- *Feedback*: Providing feedback on recommended items should be as unobtrusive as possible. It should be easy and quick, and should require only a small amount of effort by the user. Implicitly generated feedback

would be preferable, for instance by measuring the viewing time of certain programs or analyzing uttered comments on programs.
- *Explanations*: Explanations based on peer users' interests and on similarities with the user's favourite programs were both considered to be interesting. Not everyone wants to see explanations all the time. For this reason, explanations should only be given when requested, and they should be easy to interpret. Textual explanations should be short; most participants preferred visual explanations such as charts.

In addition to ideas and comments on the three main aspects of a TV recommender, some more general ideas arose:
- The TV recommender system should be available on a variety of devices, such as personal computers, PDAs/handhelds, mobile phones and TVs. This would enable the user to consult the recommender system irrespective of his/her location.
- Watching TV is seen as a social activity. The possibility of multiple people watching TV and controlling the recommender system should be taken into account.
- Integration with an EPG that offers information on all TV programs, and not only on recommended programs, is desirable.

7.3.3 Results of interactive design session

The design process for mock-ups for a TV recommender systems' user interface resulted in a wide variety of drawings and descriptions (two mock-ups are shown in *Figure 7-2* and *Figure 7-3*). Several important similarities between the different mock-ups could be observed:
- Although participants stated that a TV recommender system should be available on a range of different devices, almost all mock-ups were based on a TV with a remote control as input device. One group proposed the use of a separate device (a hybrid of a PDA and a tablet PC) that facilitated the TV recommender systems' user interface and that could simultaneously be used to operate the TV.
- Every mock-up sorted or grouped recommendations by genre, while some provided alternative sorting options by time, channel, etc.

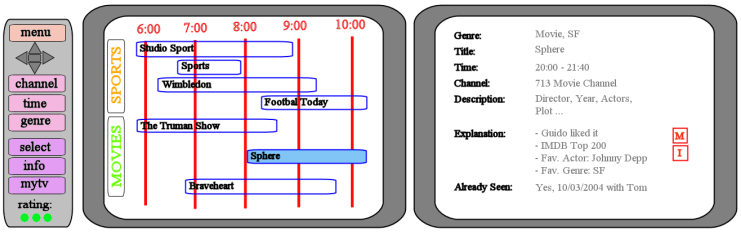


Figure 7-2 Mockup of a TV recommender interface

– As some participants remarked during the initial brainstorm session, a TV recommender systems' user interface should ideally facilitate the use by groups, because watching television is often a social event. A mockup reflecting this idea is shown in *Figure 7-3*.

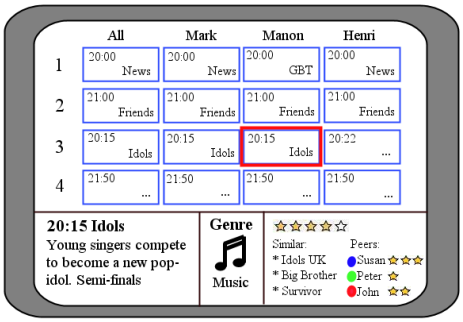


Figure 7-3 Mockup of a TV recommender interface for groups

– In virtually all mock-ups, the initiative for displaying recommendations lies at the side of the user. One group proposed unsolicited recommendations (pop-ups in the bottom of the TV screen or via instant messaging mechanisms on a PDA or mobile phone) to alert the user of a recommended TV program that is about to be aired.
– Most of the mock-ups provided a way for users to supply feedback on the recommended items. Most common was a 5-point scale ranging from 1 to 5 (visually asymmetric; the symmetry of the scale was not mentioned in the mock-ups), operated by the remote control. Other options included a sliding continuous scale and voice recognition.

Among the various mock-ups, two main interaction types could be distinguished. The first is based on the assumption that a user wishes to plan a couple of hours of TV watching. Recommended programs can be selected and placed in a 'personal EPG' or 'watch list'. More detailed information on recommended TV programs can be obtained, and these programs can be rated when watched. The mock-up in *Figure 7-2* is an example of an interface of this type. The second type of interaction is based on the idea that a user wants to watch a TV program that best fits his interests right now (immediate interests). These mock-ups provide a simpler type of interaction because fewer actions have to be performed: only the programs currently being aired are listed. In the prototype design, it has been attempted to offer both tasks within a single interface.

As the brainstorming and design sessions provided global guidelines for designing the user interface of a TV recommender system, the next step was to investigate the preferences of users for the three aspects in more detail. This was done by means of an online survey.

7.4 On-line survey about interface widgets

The analysis of existing (TV) recommender systems and the ideas generated in the brainstorm session provided some directions and guidelines for the user interface of a TV recommender system. Based on the analysis, a variety of possible interface widgets with different parameters were identified for visualizing each of the three interface aspects of recommender systems. Their usefulness was investigated in detail using an interactive online survey.

7.4.1 Evaluation by online survey

In order for participants to make a well-founded choice between different interface options for the interface widgets, an interactive online survey has been created instead of a paper survey. In this survey, participants could easily try the different widgets and were thus better able to determine which ones they preferred the most. The survey also supported branching: after certain choices participants received extra questions, or questions about parameters were tailored to the answers already given. The survey was completed by 106 people (43 female, 63 male) ranging from 15 to 70 years of age (average age 33) and with different types of education and occupation. Of those people, only 5.6% had ever used an EPG on TV while 29.9% had used an online TV guide on the Internet. Most people used paper TV guides (58% regular TV guides and 48% program listings in newspapers). Note that participants could select multiple sources for their TV program information. The survey questions can be accessed online at http://tiv.telin.nl/duine/tv/survey

7.4.2 Results

Predictions

The survey results indicate that most participants prefer either to have predictions integrated into a normal EPG (59%) or to have two separate views (39%): one with the normal EPG and one with recommendations. Only 2% believed that a list of recommendations alone would be enough; i.e. people want to be able to make the final decision about what TV programs to watch and not have the recommender system make all decisions for them.

Participants could also choose between four different interface elements to present predictions (see *Figure 7-4*): a group of symbols where more symbols express a higher prediction, a thermometer-like bar, a numerical score, and a smiling/sad face symbol. Most participants opted for the group of symbols (69%), with the bar in second place (19%). The main reason people gave for this choice was that both the group of symbols and the bar

provide a clear and orderly presentation of a prediction while allowing for easy comparison between multiple predictions.

Figure 7-4 Interface elements presenting predictions: group of symbols, bar, numerical score and smiling/sad face symbol

Of those who preferred a group of symbols, most participants liked to have stars presenting the prediction (85%), while only a few (7%) opted for thumbs. The others had no opinion or provided their own suggestions. When asked about the number of symbols that should be used in presenting a prediction, the majority chose a scale of 5 symbols (89%), by which 63% indicated that the centre symbol (e.g. three stars) should be seen as a neutral value. A neutral centre value indicates that most participants prefer a symmetrical scale for a prediction (using both positive and negative values and with equal lengths for the positive and negative sides), but an asymmetrical visual representation.

The use of colour in presenting the predictions was valued as an improvement by 91% of the participants. They noted that colour improves transparency, is more orderly and distinct, and provides a quicker overview of the predictions. However, attention should be devoted to colour-blindness; the presentation of predictions must be clear, even for people who cannot distinguish between certain colours.

Participants were also asked which colour they believed should be used to express that a program fits their interests poorly, neutrally or well. Most people associated red with a predicted low interest (57%) and associated orange (31%), yellow (26%) and blue (19%) with a predicted neutral interest. The prediction that the user would find the program interesting was predominantly associated with green (62%), although some people also indicated that red (15%) or yellow (14%) might be used. When prompted to select colour triplets for expressing predicted low-neutral-high user interests, the most popular combinations were red-yellow-green (15%), red-orange-green (15%) and red-blue-green (13%).

It can be concluded from these results that people prefer conventional and well-established patterns for presenting predictions: one to five stars to present a prediction (with three stars being neutral), and colour combinations that resemble those of traffic lights. Please note that this preference may be culturally influenced: when different established patterns exist in other cultures, it might be best to use those patterns instead.

Feedback

Although implicit feedback was preferred by the participants of the brainstorming session, the survey focused on explicit feedback because

explicit feedback is reflected in the user interface while implicit feedback is not.

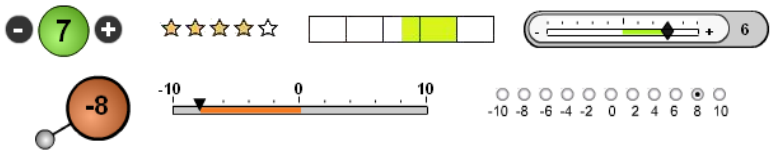


Figure 7-5 Various widgets for giving feedback. From left to right and top to bottom: numerical score with plus and minus buttons, group of stars, rating bar, rating slider with numeric score, volume knob, simple rating slider, and radio buttons

When presented with six different widgets for providing explicit feedback (see *Figure* 7-5), participants' stated preferences were less in agreement than for the elements representing predictions. The three most popular widgets were the ratings slider (26%), the group of stars (24%) and the numerical score with plus and minus buttons (21%). The results were also inconclusive regarding their preference for a symmetric rating scale (that has both positive and negative numbers) or an asymmetric scale (with only positive numbers): 48% preferred a symmetric scale, 43% the asymmetric scale; the remainder did not have a preference.

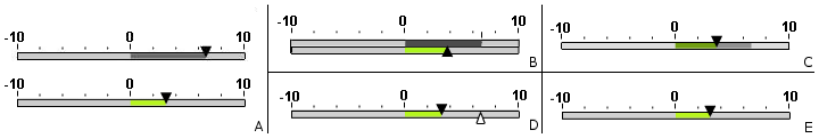


Figure 7-6 Five levels of combining feedback and prediction widgets, ranging from completely separated (A) to full integrated (E)

When asked whether the feedback widget should be separated from or combined with the presentation of the prediction, 55% chose to have the two combined, while only 33% preferred to separate the two completely (others were indifferent). Although most participants preferred integration, about 53% opted for loose integration only (widget B in *Figure* 7-6) in such a way that the combined widgets for feedback and predictions could still be identified separately; this way, a user can still see the original prediction when providing feedback. The other respondents selected one of the three other integration options; the more integrated, the less participants preferred them.

We believe that the same scale should be used for the presentation of predictions and user feedback, because there was no clear preference for a symmetric or an asymmetric feedback scale, and participants preferred to have the presentation of the prediction and feedback loosely integrated into a single widget, and consistency is an important generic usability requirement (Shneiderman, 1998). When looking at the granularity of the rating scale, it appears that people like a low to medium number of values. On both the symmetric and asymmetric scales, a range of 10 had a large preference (65% on the asymmetric scale and 21% on the symmetric scale), although on the symmetric scale, a range of 20 had the largest preference (33%). This might again be culturally influenced, in this case by Dutch

school grades that are on a 10-point scale. The range of 20 on the symmetric scale had a maximum of +10 and a minimum of -10. When providing feedback, participants also preferred the use of colour in the feedback widget (82%).

From these results and the general consistency principle, it can be concluded that it is best to use the same type of presentation and scale for predictions and feedback, namely a symmetrical scale mapped onto 5 stars. Because feedback requires a granularity of at least 10, half stars should also be supported. The neutral value should be the median of the range, i.e. 2.5 stars. For consistency reasons, the range for predictions should be the same as that for feedback (otherwise when a user gives a feedback of 3.5 stars for a program, the same program could be recommended to him with 3 or 4 stars). Furthermore, the feedback widget should be loosely integrated with the presentation of the prediction and should use colour to present the given feedback value.

Explanations

Participants indicated that a recommender ought to be able to explain its predictions (45% indicated that it was important and 28% that is was very important). Most participants (56%) prefer clear explanations, without wanting to know much about the inner working of the prediction engine. However, there are some who prefer more detailed explanations (22%), while others prefer minimal explanations (22%).

To determine what types of explanations people trust the most, we provided four different types. The first explanation was based on the similarity between the recommended TV program and another TV program the user liked: "you will like 'Angel' because you also like 'Buffy The Vampire Slayer'". This explanation was preferred by 25%. The second explanation was based on what the user's friends thought about the program: "Your friends Frances, Margit and Jeroen liked this program". Only 6% of the participants trust this explanation, which was explained by one of the participants as "although they are my friends, it does not mean that we have the same taste". Most people (34%) preferred the third type of explanation, which was based on the idea of collaborative filtering: "people who have tastes similar to yours liked this program". Also explanations based on program aspects, such as actors, genres or the director, were preferred by many participants (32%), e.g. "This movie is directed by Steven Spielberg and Tom Cruise plays one of the main characters".

When looking at the modality for presenting explanations, we offered participants three different modalities: a graph, a table and a textual explanation, and asked them to choose the preferred modality. Most participants opted for the graph (46%) or table (44%), while very few preferred the textual explanation (2%); the remainder had no preference.

This result confirms Herlocker's et al. (2000) findings regarding the modality of explanations.

90% of the participants preferred receiving an explanation only when they explicitly requested one and not automatically with every prediction. Only 6% wanted to see explanations with all predictions, while 4% did not want to see explanations at all.

It can be concluded from this survey that people find it important for a recommender system to be able to explain its predictions, although only when requested. Explanations themselves must be clear without too much detail about the inner working of the prediction engine. There is no clear preference for the type of explanations, although explanations based on people's friends are trusted the least. This also implies that it is possible for prediction strategies to use the explanations of its predictors where each predictor provides its own type of explanation. The modality of explanations should at least contain a graph or table and not only textual information, because graphs and tables allow people to quickly understand explanations.

7.4.3 First prototype

Based on the results of the brainstorming session and the interactive on-line survey, a first prototype of the user interface for the TV recommender was developed (see *Figure C-1*, *Figure C-2* and *Figure C-3* in Appendix C). In this design, predictions are presented by a group of stars. The scale of the prediction consists of five stars with a granularity of 0.5 stars, where 2.5 stars, being the median of the five stars, represents the neutral value, making it numerically symmetric, but visually asymmetric. The traffic-light pattern of red-yellow-green is also used within the presentation of the predictions as fill colour for the stars (e.g. in *Figure C-2* the program "De Bovenman" has 4.5 green stars, while the program "Kruispunt" has only a half red star). Feedback is given using the same scale as that for the predictions: the feedback widget is a combination of five stars with precision 0.5 and a rating bar below the original prediction, thus providing redundancy of interaction. Explanations are presented on the feedback pop-up screen by a short textual description and a graph that depends on the used prediction technique(s). We developed the recommender system's interface to be used on a tablet PC because we believe that such mobile devices (either in this form or in the form of PDA's or mobile phones), with integrated remote control functionality for the TV, will become more common in the future. The main advantage of using an EPG on mobile devices is that it allows people to access the EPG without disturbing other viewers who are watching TV, which is the case if the EPG has to be

accessed on the TV screen. However, the user interface can also be used on a regular PC without any changes.

Because the focus of this research was placed on predictions, feedback and explanations, we only used a table view EPG layout in this prototype and did not investigate alternatives, such as a grid layout or 3-D layouts, e.g. time pillars (Pittarello, 2004). These could provide better program-guide layouts when many more channels are available to the user. Other aspects, such as group recommendations (Masthoff, 2004) and interfaces for multiple devices, have also not been studied.

7.5 Heuristic evaluation of the first prototype

A formal heuristic evaluation involves having a small set of evaluators (often usability experts) examine and judge the user interface with recognized usability principles (or heuristics). With heuristic evaluation it is possible to identify many usability problems early in the design phase (Nielsen, 1993).

Although the focus of this research was placed on the three main aspects of recommendations, the heuristic evaluation also gave us insight into usability issues that affect the entire user interface of the TV recommender.

7.5.1 Heuristics

The heuristics that were used in evaluating the prototype are based on research by Harst & Maijers (1999), Shneiderman (1998) and Nielsen (1993). The heuristics used are:

- Provide task suitability
- Employ consistency throughout the interface
- Evoke a strong sense of controllability
- Reduce the user's short-term memory load
- Provide effective feedback and error messages
- Provide useful, straightforward and well-designed (on-line) help and documentation
- Be considerate in layout and aesthetics
- Use colour with thought.

7.5.2 Results

Two usability experts, two cognitive psychologists with experience in user interface design, who where not part of the research team performed a heuristic evaluation on the prototype. They discovered the following design issues:

- Ensure a consistent way of accessing detailed information about a TV program, both when opening the feedback (pop-up) screen and when

opening the explanation screen. In our prototype, clicking on the program title or short description would open a pop-up window with detailed information, while clicking on the prediction opened a pop-up that allowed the user to provide feedback and see the explanation. This might confuse users. In the revised design, clicking anywhere on the TV program creates a pop-up window in which the detailed information, the feedback and the explanation can be accessed separately using tabs. The tab that is displayed still depends on the location of the click. This form of presentation allows users to easily switch between the three aspects.

– Provide clear visible clues of what actions a user can perform. In our interface, users could drag programs to their own "watch list". Although the location where the user had to place the stylus in order to drag the program was clearly marked (*Figure 7-7*), it could be made clearer by changing the cursor symbol when it is above or near such a handle, thus making its affordance more easily visible.

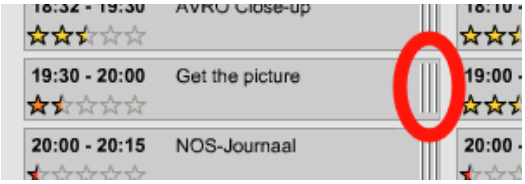


Figure 7-7 Visual indication that a program can be dragged

– Make the functionality of buttons very clear. Our EPG has two display modes: one in which the times of programs on the different channels are not synchronized (as shown in *Figure C-2* in Appendix C), and one in which the times are synchronized in blocks of one hour. To switch between these two modes, users had to activate or de-activate a clock-like symbol that was unclear and wrongly positioned. In the new design, a checkbox was used instead that is located just below the time selection field.
– Clearly show what information is currently displayed. In our first prototype, the date of the TV shows currently being displayed was only visible in the drop-down fields in the selection column. It should also be visible at the top of the listed programs.
– Bring explanations to the point; elaborate explanations are more difficult for users to understand.
– Make the explanations consistent with the presentation of predictions. Predictions in our prototype have a granularity of 0.5. However, explanations described the average interests with a different granularity, e.g. 1.6 stars. The granularity of the two should match.
– In the first prototype, the user had to press a save button after providing feedback to store the rating. According to one of the usability experts,

this was unnecessary as using the feedback widget should be enough: users should not have to press an extra button. The save button was therefore removed and the application saved the rating automatically.
- Allow users to scroll to other channels instead of requiring them to select channels from a pull-down menu. As this creates a completely different grid-like display of TV programs, it was decided to wait for the results of the usability tests before making such a drastic change.
- The experts disagreed about the use of colours for the predictions and feedback: one expert found the colours (traffic-light model) non-intuitive and unclear while another expert found them to be intuitive and very useful, as they made high predictions more easily detectable. Because the survey also indicated a preference for the traffic-light model, it was decided to leave it untouched until the usability tests gave a more definite answer.

The experts also provided insights into the positive aspects of the design. They believed that most goals of TV viewers are easy to achieve using the interface. Sometimes several different actions lead to the same result (redundancy), meaning that no extra shortcuts were needed. Both experts also indicated that they felt they were in control of the system. They believed that users would have little difficulty using the interface and would also have a feeling of control. The layout was perceived as generally clear and logical. However, some minor changes were recommended, e.g. in the placement of labels and the alignment of certain interface elements. These changes were taken into account in the subsequent prototype. Experts commended the sparse and hence effective use of colours. General items, such as backgrounds, tabs and buttons, are displayed in neutral (grayish) colours, while more important or highlighted items, such as predictions and genre indications, are shown in more striking colours.

7.6 Usability testing

Figure C-4 to *Figure C-7* in Appendix C show the new prototype, in which the most important problems that were uncovered in the heuristic evaluation have been addressed while preserving the strong points of the interface. With this new version, two series of usability tests have been performed with five users each. Usability testing with real users is the most important evaluation method. In a certain sense, it is irreplaceable because it provides direct information about how people use computers and what their exact problems are with the concrete interface being tested (Nielsen, 1993).

Dumas & Redish (1993) state that usability tests share the following characteristics:

- The primary goal is to improve the usability of the product; each successive test will have more specific goals.
- The participants represent real users and do real tasks.
- Everything participants do and say should be observed and recorded.

The resulting data is analyzed, the real problems are diagnosed and changes to fix those problems are recommended.

7.6.1 Setup of the usability test

The first usability test was conducted with three male and two female participants in individual sessions. One participant was in the age group of 15-20, two were 21-30, one was 31-45 and one was older than 45. All participants were familiar with the usage of TVs and had used a PC before: some had limited PC experience, others average. They were provided with a tablet PC containing the TV recommender. Before starting the session, participants were allowed to practice the use of the tablet PC with a stylus as an input device by playing a few games.

All actions performed by the participants were recorded on a VCR by capturing the image of the tablet PC. The participants were asked to go through several assignments on their own, without any help from or communication with the observer, and to think aloud. To ensure that the participants had real tasks when using the personalized EPG, the assignments included questions they had to answer, e.g. "How well do you think the program 'Newsradio' suits your interests, according to the system? (in your own words)".

Participants were clearly instructed that we were evaluating the user interface and not them, so that if they were unable to carry out an assignment it was not their fault, but a fault of the interface. In order to assess the perceived quality of the user interface, participants were asked to fill out a small questionnaire (16 questions on a 5-point scale).

Prior to the usability test, the following quantitative usability goals were defined:

- All participants must be able to perform all assignments on their own, without intervention by the observer.
- Each assignment must be completed within a specified time, which was determined by measuring our own use of the system (adding a safety margin because we were well acquainted with the interface) and based on a few small tests with various people that were unfamiliar with the experiment and who had not participated in the brainstorm or the interactive design sessions. Participants were not aware of this predefined maximum time; they could continue until the assignment

was completed, or abort the current assignment if they felt the system was not responding properly.

The qualitative usability goals were (which are based on the usability of those user interface aspects that are specific to recommender systems and some generic qualitative usability goals):

- The user interface must be easy to use.
- The interface should be intuitive.
- How the system presents a prediction and the meaning of the prediction should be clear.
- It should be easy for users to provide feedback on predictions.
- It should be simple for them to find explanations of predictions, and these explanations should be easy to understand.

7.6.2 Results of the first usability test

All participants performed all assignments without help from the observer. However, not all participants accomplished all assignments within the predefined maximum time (all reported times are true "interaction times" and do not include time spent reading the question). In particular, the following problems were identified:

- In the used prototype, the stars of a listed program turned white to indicate that this was not a prediction but feedback previously provided by the user for that same program. This appeared to be unclear: it took three participants more than the target of 60 seconds to figure out how the interface displayed this information.
- Users could drag programs to their watch lists by clicking on a handle pane next to each listing (see *Figure* 7-7) and then dragging the listing(s) to their watch lists. Based on the heuristic evaluation, we had already changed the mouse cursor symbol to indicate that the user could initiate a drag operation when hovering over this area. Participants nevertheless assumed that a program could be dragged by clicking at any point in its display area. It took two participants more than the target of 90 seconds to complete the assignment of adding four programs to their watch list.
- Finally, knowing how to find out which programs are in a certain genre was not intuitive (once again it took two participants more than the target of 90 seconds to complete this assignment the first time it was encountered in the task list). However, when the same task was encountered a second time in the task list, all participants completed this assignment well within the maximum time allotted.

The measured times also indicate that participants quickly learned how to use the interface. For instance, it took the five participants an average of 49

seconds to highlight genres the first time they had to do this. On a second occasion, it took them only 19 seconds. All participants were able to find out how to deal with this particular aspect of the interface, and easily remembered and applied this knowledge later.

Decreasing execution times for similar tasks were also seen in assignments in which participants had to drag programs to their watch lists. For the first program, it took them an average of 120 seconds; the second time only 12 seconds. Because the average time for completing this assignment the first time greatly exceeded the maximum allowable time limit (90 seconds for all four programs), the way programs could be dragged to the watch list was changed: dragging can now be initiated by clicking anywhere in the display area of a program, rather than using a dedicated handle only; the handle itself is maintained to indicate that the program item could be dragged.

Presentation of predictions

All participants instantly understood the meaning of the stars that indicated their predicted interest in a particular program. Also, when looking for more information on a certain program, they intuitively clicked on the program in question. Participants agreed that the interface clearly indicated whether or not a program would meet their interests (score 4.2 out of 5 {5,5,4,3,4}). The use of colours (green, yellow and red stars) was seen as explanatory and clarifying (score 4.6 out of 5 {4,5,4,5,5}). This calmed the concern that arose in the heuristic evaluation; users do appreciate the use of colours for presenting predictions.

In the design, the difference between a prediction and a program for which the user had already provided feedback was expressed by replacing the prediction with the feedback of the user, and visually changing the colour of the stars to white. This only appeared to be clear to two of the participants. One of the other three noticed it later in the test. This has been made clearer in the next version of the prototype, by adding a small icon of a person beside the stars if the rating was based on feedback given by the user (the colour still changes to white) and by making it clearer when providing feedback (see next section).

Providing feedback on predictions

All participants were able to quickly access the part of the interface with which they could give feedback on a prediction. The way to do this with the feedback widget was purposely kept redundant: users could use the slider or directly click on the stars. Three participants used the slider only, one participant clicked on the stars only and one participant used both options.

After rating a program in a pop-up window, four out of five participants were insecure about how to close the window. One participant pressed the

"Reset" button, while others eventually used the "X" button in the top-right corner of the pop-up. One of the participants reopened the pop-up window in order to check if his feedback was properly saved. During the discussion, four participants indicated that they expected some explicit feature to save their feedback, such as a save button. The lack of specific feedback from the system on their actions resulted in insecurity. This finding is in contradiction with the opinion of one of the usability experts in the heuristic evaluation. It appears that although it takes an extra action, users prefer to be sure that their feedback is saved; hence the save button was reintroduced. The save button is only enabled when the user has given or changed a rating. Pressing the button changes two visual states: the stars of the feedback widget change by turning the colour of the stars to white (the same colour that is used for the stars in the program listing for a program the user had already given feedback on) and the save button becomes disabled.

According to the final questionnaire, four participants agreed that giving feedback on a prediction takes little effort (average score of 4.75 {5,4,5,-,5}out of a maximum of 5) while 1 participant was indecisive about this matter.

Explanations

All participants were able to quickly access the part of the interface in which they could find explanations about a prediction. This was also confirmed by the final questionnaire, in which all participants agreed that the explanations were easy to find (score 4.8 {5,5,5,5,4} out of 5). Participants also indicated that explanations were visualized in a good way (score 5 {5,5,5,5,5} out of 5) and that the explanations serve their purpose well, because they clarify the prediction (score 4.6 {5,4,5,4,5} out of 5). Participants also indicated that they found the explanations to be relatively credible (score 3.8 {4,3,4,5,3} out of 5). However, some participants indicated that they would like more detailed explanations. The survey results also indicated that some people prefer minimal explanations, while others prefer more details. Therefore, we changed the prototype so that users can ask for more details on explanations when desired.

Interaction with various interface components

In general, participants indicated that the interface was easy to use (score 4.2 {4,5,4,4,4} out of 5) and that they were in control of it (score 4.6 {5,5,4,4,5} out of 5). This conclusion is also supported by the measured times it took participants to complete the assignments.

Separating the interface into different main functions on tabbed "file cards" (see *Figure C-5*, *Figure C-6* and *Figure 7-1* in Appendix C) also appeared to be a good design decision. All participants managed to find

information on these file cards quickly, and knew intuitively how to use the tabs. The pop-up window with extended program information, feedback and explanations appeared in a fixed position relative to the program on which the user clicked. Some participants mentioned that this obstructed the programs listed below the selected program, and suggested making the window dragable. This was changed in the follow-up prototype.

In the heuristic evaluation, one of the experts indicated that users should be able to scroll through the various channels, instead of selecting sets of channels from a pull-down menu. The usability tests did not indicate that users had any problems with selecting the channels from a pull-down menu; however, this does not mean that pull-down menus are better than scrolling. This issue is left open for research, as it is not related to the main user interface elements for recommender systems.

7.6.3 Iteration

Because some of the changes to the original prototype were not trivial (e.g. how user ratings are saved and how they are visually presented), iterative design theory requires another evaluation test, which should focus on the revised parts of the interface. Another usability test was therefore performed that was similar to the one described in the previous section. Five people, who did not participate in the previous usability test nor in the brainstorm or interactive design session, were asked to participate (one in the age group of 15-20, two in the group 21-30, one in the group 31-40 and one well above 41). The usability goals of this test corresponded with the usability goals of the previous test but focused on the changed aspects of the interface.

This second evaluation attested significant improvements in the usability of the prototype. All participants were able to perform all assignments within the estimated time limit without help from the observer. Measured times for completing the assignments show that the changes made to the prototype greatly simplify the tasks that proved to be too difficult in the first usability test. Dragging four programs of their own choice to the watch list took participants an average of 79 seconds (compared to 137 seconds in the first usability test). Participants felt that dragging programs could be done very intuitively because a drag action could be initiated from any point on a TV program display. Another important result was that participants instantly recognized the programs they had given feedback on; they all understood that the presence of a white person-like icon, which was added to this last prototype, indicating that they had given feedback on that particular program. This was a considerable improvement, considering that users had trouble figuring out what programs they had rated previously during the first usability test; this took them an average of 117 seconds.

During the second usability test, the same task was completed with an average of 8 seconds.

Results of the second evaluation indicate that the usability problems identified during the first test were resolved. No new usability problems were identified, which is why no further adjustments to the prototype were necessary.

7.7 Future evaluation and research

The usability tests described in the previous section were the last tests performed on the prototype interface. However, before a TV recommender system and its user interface as described in this chapter can be marketed commercially, more extensive usability tests should be performed involving usage in real-life household settings with a substantially larger number of users and over a longer period of time. The usage on multiple devices, individual user characteristics (such as colour blindness), and integration with devices such as digital video recorders should also be taken into account. Additional usability problems can then be uncovered that remain unnoticed in a more laboratory-like environment as was used for this research.

The results of this investigation into the presentation and interaction aspects of recommender systems have been applied in the EPG design used in chapter 6. However, not the full prototype design has been transferred to that experimental system as some contextual differences had to be taken into account. The major difference was that the EPG prototype resulting from the iterative design session was focused on using a tablet PC in the living room. As it was not possible to provide all participants of the experiment in chapter 6 with a tablet PC, that EPG was offered as a website on the Internet. For that reason, some of the interaction possibilities natural for a tablet PC (e.g. drag-and-drop using a stylus) were both difficult to implement in a web-based system and are also not commonly used in websites. Screenshots of the user interface as used in the experiment as described in chapter 6 can be found in Appendix A.

7.8 Conclusions

In this chapter, the presentation of selected and structured items has been investigated. Presentation is the third main process of personalized information systems, where this thesis focused on recommender systems as a specific type of personalized information system. The first two processes,

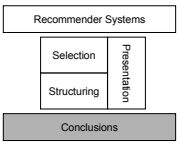
selection and structuring, have already been discussed in the previous chapters of this thesis.

A user-centred design study has shown that for the presentation of predictions, it is best to use a traditional form of presentation, namely five stars with a half star granularity where the centre of the five stars represents a neutral value and using a colour scheme that follows the well-known traffic light pattern. The study also showed that people believe that recommender systems should be able to clearly explain its predictions, but only on request, and without too much detail about the inner workings of the prediction engine. There is no clear preference for the type of explanation, which means that each prediction technique can have its own type of explanation. Explanations should be presented using graphs or tables; text should only be used in addition to graphs or tables. When users have to provide explicit feedback, it is best to use the same type of presentation and scale that is used to present predictions. The feedback widget should be loosely integrated with the presentation of the prediction, so the original prediction can still be distinguished from the feedback that the user provides.

With these user interface design guidelines, all three objectives of this thesis, which followed the three main processes of personalized information systems, have been reached. The next chapter summarizes all the results, discusses the limitations of the research and results in this thesis and provides directions for future research concerning recommender systems.

Chapter 8

Conclusions



The main objective of the research described in this thesis has been to investigate and develop technical solutions that support people in finding interesting information in order to help them overcome the information overload problem. Solutions have been investigated and developed for the three main processes in personalized information systems: the selection process which has been addressed in chapter 3 to 5, the structuring process which has been addressed in chapter 6, and the presentation process which has been addressed in chapter 7.

This chapter summarizes the results of this research in section 8.1 for each of these processes. In section 8.2, we discuss the limitation of this research by reflecting on the generalizability of the results, by putting the results into perspective and by discussing necessities for recommender systems research. In section 8.3, future research directions that result from this research are discussed after which we conclude this thesis in section 8.4.

8.1 Research results

8.1.1 Prediction framework

Supporting people in the information selection process means selecting items that are of interest to them, which is the focus of recommender systems. One aspect of recommender systems is to predict how interesting an item is to a user. Instead of focusing on specific techniques that can predict how interesting an item is to a user, we focused on using multiple prediction techniques, so called hybrid recommender systems. Hybrid recommender systems are aimed at providing more accurate predictions than recommender systems using only one prediction technique. Many hybrid recommender systems have been developed for specific applications

or domains; our research objective was to develop a domain-independent framework that describes how to create hybrid recommender systems that can be tailored to various domains in order to provide more accurate recommendations.

Switching hybridization

Based on an analysis of several hybridization methods, a switching hybridization method has been chosen for the framework as this method provides the best balance between loose coupling, where only the results of the used prediction techniques are combined without having any knowledge about the internal workings of the prediction techniques, and tight coupling, where a lot of knowledge about the internal workings of prediction techniques is used to combine the prediction techniques. This makes switching ideal for a domain-independent framework that can be implemented in various domains and then tuned to the domain in which it is implemented, while still providing enough benefits of hybridization to result in an increase in prediction accuracy.

Duine prediction framework

An analysis of various prediction techniques has lead to the development of a generic model for a predictor, which is an entity that predicts how interested a user will be in an item. A predictor takes user profiles, which contain knowledge the system has about a user, and information about an item as input and results in a prediction and an explanation about this prediction. A predictor can also receive feedback from users on its predictions so it can learn to improve its predictions. Every predictor also exposes so-called validity indicators, which provide information about the amount and quality of knowledge that is available to the predictor on which it bases its predictions; these validity indicators can be used to determine how useful the predictor will be in predicting the user's interests in an item.

Switching hybridization is implemented in the framework via prediction strategies. Prediction strategies are predictors that generate an interest prediction for an item and user, not by using an algorithm like prediction techniques do, but by selecting between and/or combining predictors based on the most up-to-date knowledge about the current user, other users, the information, other information and the system. This up-to-date knowledge is captured via the validity indicators of predictors.

Prediction strategy decision approaches

Prediction strategies can employ various approaches to make a decision about which predictor to use for a specific prediction request. Some of these approaches are capable of learning from feedback received from users on predictions, while other approaches are static. Four different decision

approaches (decision rules, case-based reasoning (CBR), backpropagation artificial neural networks and Bayesian probabilities) have been investigated on their theoretical applicability in prediction strategies.

Experiments

In order to determine whether recommender systems can be developed using the framework, prediction engines based on the framework have been developed for two systems: one electronic TV program guide (EPG) and one movie recommender system (MovieLens). Two distinct prediction strategy decision approaches have been used in the validation experiments for each of these two systems: manually created decision rules that are model-based and case-based reasoning that is automated and instance-based. The goal of these experiments has been to demonstrate that the framework can be used to develop prediction engines for recommender systems and that predictions made by these engines are accurate.

Accuracy results

In both systems, the EPG and MovieLens, the rule-based prediction strategy provided significantly more accurate predictions than each of the prediction techniques used within the prediction strategy. However, the increase in prediction accuracy in the EPG system was larger than the increase in prediction accuracy in MovieLens. Rule-based prediction strategies provide more accurate predictions than any other prediction technique, as they are capable of switching between predictors; when one predictor is not capable of providing accurate predictions it can switch to other predictors. A downside of rule-based prediction strategies is that expert knowledge is required to design the rules, while fine-tuning the rules requires experimentation. This drawback can be overcome by using a prediction strategy decision approach that teaches itself when to use which predictor.

The experiment with case-based reasoning as a prediction strategy decision approach has shown that in the EPG system, the CBR-based prediction strategy is capable of providing significantly more accurate predictions than the prediction techniques used by the strategy. However, in MovieLens the CBR-based prediction strategy only provides predictions that are just as accurate as the best prediction techniques used in the strategy. This is caused by the small spread in prediction accuracy of the prediction techniques used in MovieLens, which makes it more difficult for case-based reasoning to discriminate between prediction techniques.

A comparison of the prediction accuracy results of the rule-based prediction strategies and the CBR-based prediction strategies has shown that in the EPG system, the CBR-based prediction strategy is significantly more accurate than the rule-based prediction strategy, except at the start of the usage period as in that period CBR still has to build up a case-base.

However, in the MovieLens systems, the rule-based prediction strategy is almost always more accurate than the CBR-based prediction strategy, which can be attributed to the difficulties of CBR to learn from the prediction techniques in MovieLens which all have similar prediction accuracy levels.

This comparison indicates that it depends on the used prediction techniques whether manually created rule-based prediction strategies or CBR-based prediction strategies can best be used; if the prediction accuracy levels of the used prediction techniques are sufficiently diverse, CBR can be used as a prediction strategy; however, in other situations manually created rule-based prediction strategies are to be preferred. When the scalability issues of CBR are also taken into account, our recommendation is to use manually created rule-based prediction strategies, unless no expert knowledge is available to design these rules.

Conclusion

Based on these findings, we conclude that the Duine prediction framework, which uses a switching hybridization method based on validity indicators of predictors, can indeed be used to create prediction engines for recommender systems that are capable of providing more accurate predictions than any of the individual prediction techniques.

8.1.2 Goal-based structuring

Current prediction techniques only address the user's long-term and short-term interests, not their immediate interests; i.e. the goals that people have for the items they are looking for. For this reason, we have also investigated how the goals of users can best be used in recommender systems.

Determining user goals

Three ways in which the goals of a user can be determined and used in a recommender system have been investigated. As having a recommender system predict the goals of a user is not possible given the current state of the art on acquiring context information about a user (such as emotional state) and the fact that people are not always capable of explicitly specifying their goals, a combination of predicting and specifying is the best solution: structuring items based on the possible goals items can fulfil for a user. A recommender predicts what possible goals each item can fulfil for a specific user, assigns these goals to the item, and groups the items based on these possible goals; the user then examines the groups and decides which goal best matches his current need and picks an item from that group.

Gratifications

The assignment of goals to items is based on a combination of the means-end approach and the uses and gratification theory. This combined theory describes that there is a link between the attributes of an item and the uses and gratifications that people receive from that item; these uses and gratifications need to match with the goal(s) of a user. This link between item attributes and gratifications is subjective as different people can receive different gratifications from the same item.

Experiment

In an experiment with an EPG, TV programs were either structured traditionally using channels, implicitly on goals using an attribute that gives an indication of a goal (genres) or explicitly on goals via the uses and gratifications. Within each of these structuring methods, some guides used predictions while others did not; this resulted in six types of TV guides that were used in a between-subjects experiment.

Results

The experiment has shown that structuring items based on the user's goals makes it easier for people to find interesting items, especially if goals are used explicitly. Predictions on their own only make it easier for people to find interesting items when they are added to traditional structures (such as a channel-based structure in an EPG). Adding predictions to goal-based structuring does not have any effect on how easy it is for people to find interesting items as the goal-based structure already provides enough support in finding interesting items. Goal-based structuring has a higher learning curve than structures that are not based on the user's goals; we believe this can be attributed to the fact that people are forced to make their goals explicit, which is something that most people are probably not used to.

Goal-based structuring does not have any effect on the ease-of-use or how clear and understandable a recommender system is, whereas predictions improve the clarity and how understandable recommender systems are.

Although the experiment shows that goal-based structuring supports people in finding interesting items, there are some people who do not like goal-based structuring. In combination with the higher learning curve of goal-based structuring, we therefore recommend that recommender systems allow people to choose and switch between various structuring methods: such as traditional channel-based structuring and genre-based structuring, including goal-based structuring. That way, people can get used to goal-based structuring at their own pace and decide if and when to switch to goal-based structuring.

Conclusion

Based on the results concerning the use of goals in recommender systems, we conclude that using goal-based structuring is important to successfully support users in finding interesting items. Goal-based structuring is even more successful in helping people find interesting items than recommendations in the form of predictions.

8.1.3 Presenting predictions

In the last process within a recommender system before the user receives the recommendations, items and associated predictions are presented to the user; this is part of the user interface of a recommender system. This research investigated the design of those aspects of the user interface that are specific to recommender systems. These aspects have been derived from the generic model of a predictor: the presentation of predictions, the presentation of explanations and the way users provide feedback on predictions.

A user-centred design study within the domain of EPGs has shown that for the presentation of predictions, it is best to use a traditional form of presentation, namely presenting the prediction using five stars with a half star granularity, where the centre of the five stars represents a neutral value. When presenting these five stars, colours enhance the understanding of the prediction when the used colour scheme follows the well-known traffic light pattern where red – yellow/orange – green represents negative – neutral – positive predictions.

The study also confirmed that people believe that recommender systems should be able to clearly explain its predictions, but only on request. Explanations should not provide too much detail about the inner workings of the prediction engine. There is no clear preference for the type of explanation (similarity with known items, opinions of friends, based on attributes of an item, and opinions of people with similar taste) which means that each prediction technique can have its own type of explanation, one that best fits the underlying idea of the prediction technique. Explanations should be presented using graphs or tables; text should only be used in addition to graphs or tables.

Most people prefer the use of implicit feedback (monitoring the behaviour of the user) as it requires the least effort from the user. However, when providing explicit feedback, it is best to use the same type of presentation and scale that is used to present predictions, namely five stars with a granularity of half stars using the traffic light scheme for colouring the stars. The feedback widget should be loosely integrated with the presentation of the prediction, so the original prediction can still be distinguished from the feedback that the user provides.

These findings can be used as guidelines when designing user interfaces for recommender systems.

8.2 Limitations of research

8.2.1 Generalizability

The results of this research are first of all valid within the domain in which they have been investigated: the domain of electronic TV program guides and movies as far as the Duine prediction framework is concerned. However, we expect that most of the results are also valid in other domains.

Duine prediction framework
Besides using electronic TV program guides, the Duine prediction framework has also been applied and investigated using a movie recommender system and a recommender that recommends points of interest in a tourist application (although prediction accuracy measurements for the latter have not been performed). We expect that the principle of using prediction strategies that decide which predictors to use to provide a prediction based on the most up-to-date knowledge about the user, other users, the information, other information and the system itself remains applicable in other domains, especially since the framework has been developed based on the generic aspects of prediction engines.

Goal-based structuring
We expect that structuring items based on the expected goals in the form of uses and gratifications will also help people in finding interesting items in other domains than EPGs as the idea of goal-based structuring has been based on the means-end approach and the uses and gratification theory, which are both applicable and have been investigated in various other domains; i.e. people also have goals they want to meet in other domains. E.g. when selecting music, there are several uses and gratifications people can have for playing music, such as using music as background, wanting to sing-along, using music for a formal diner, using music for a party, changing one's mood etc. As the actual uses and gratifications will be different in other domains, uses and gratification research is necessary to determine the possible uses and gratifications people can have for items within a domain.

Presenting predictions
The user-centred design study into the user interface aspects of recommender systems has confirmed one of the propositions that is often

used in user interface design, namely "keep it simple"; users prefer that user interface elements are based on simple and well-known patterns instead of fancy designs that they have to get used to. As the concept of using stars to represent ratings has been used before in other domains, we expect that the guidelines for the design of those user interface elements for recommender systems established in this research also apply to domains beyond electronic TV program guides.

8.2.2 In perspective

Prediction strategy decision approaches

Only two prediction strategy decision approaches have been examined in the experiments with the Duine prediction framework, namely manually created decision rules and case-based reasoning. The other discussed learning techniques, Bayesian probabilities and neural networks, have not been used in the experiments as the goal of these experiments has been to demonstrate that the framework can be used to develop prediction engines for recommender systems and that predictions made by these engines are accurate, not to investigate all possible prediction strategy decision approaches. Our expectation is that neither Bayesian learning nor neural networks will provide more accurate predictions than rule-based prediction strategies and CBR-based strategies.

Bayesian probabilities resembles the way in which case-based reasoning works in a prediction strategy; with case-based reasoning, the expected error is calculated based on the errors of the predictors in those cases that have similar validity indicator values, whereas with Bayesian probabilities, a probability is calculated based on the number of times a predictor was the best predictor given the values of validity indicators; i.e. Bayesian probabilities is coarser in its calculations than case-based reasoning.

Neural networks on the other hand try to learn more fine-grained relationships between the validity indicator values and which predictor should be used (or what error each predictor will have given the validity indicator values). Although neural networks are capable of handling noisy data we expect that it will be difficult for a neural network to converge to a set of weights when there is a lot of contradiction in the data to learn from. As shown in the experiments, some predictors have very similar levels of prediction accuracy, which increases the probability of contradictions in the data from which a neural network has to learn, which we expect will make it more difficult for neural networks to converge to a set of weights. Of course, only experimentation with Bayesian probabilities and neural networks can determine their usefulness as a prediction strategy decision approach.

Case-based reasoning is situated in between the coarse learning methods of Bayesian probabilities and the more finer-grained learning of neural networks. For this reason, we expect that case-based reasoning will be more accurate than both Bayesian probabilities and neural networks. Of course, experiments using these other two prediction strategy decision approaches have to confirm these expectations.

Predictions versus goal-based structuring

Although prediction engines based on the Duine prediction framework improve the accuracy of predictions compared to the individual predictors used within a prediction engine, our findings also show that goal-based structuring has a larger influence on how easy it is for people to find interesting items than the use of predictions; this puts the increased prediction accuracy of prediction engines based on the framework in perspective. Before striving for marginal increases in prediction accuracy using hybridization methods, it is more important to provide good ways to structure items based on the user's goals.

We do however expect that in other situations predictions added to a goal-based structure will provide an additional benefit. In our experiment, when a goal-based structure was used, the number of items within each goal was limited; when the number of items within a goal increases, users will need additional help in distinguishing between interesting and uninteresting items within that goal; interest predictions can provide this additional support. Although it was not significant, the results of the experiment also show a minor increase in intent to use an EPG when predictions were added to a goal-based structure, indicating that predictions do have some use to people.

Users and datasets

During this research, we noted that one of the key aspects in successful recommender systems' research is having access to a group of real users that can use a recommender system, give feedback on recommendations and who can be invited to participate in experiments. Without such a group of users, research never passes the theoretical stage, as no real world validation is possible. As the main objective of recommenders is to support people in finding interesting items, it is extremely important to validate algorithms and ideas with real users.

Developing, deploying and maintaining a recommender system is not only costly, it also requires access to a rich set of information that is of interest to a group of users. These obstacles make it worthwhile to team up with content providers and/or to share datasets within the research community. One way of sharing access to user data is by making datasets publicly available, like has been done with the MovieLens datasets, the

Jester joke dataset, the EachMovie dataset and the BookCrossing dataset. For this reason, the EPG dataset used in validating the prediction framework has also been made publicly available at http://duine.telin.nl.

One must however be aware that every dataset has been collected with a specific focus in mind, which can be incompatible with the research questions that have to be answered in other studies. Datasets are ideal for comparing results between research projects and between versions of algorithms; however, having direct access to a real group of users remains a must in order to address issues that go beyond a fixed dataset.

8.3 Directions for future research

In the area of recommendations and predictions, there are several avenues of research that still need to be addressed and which have been mentioned before in the various chapters of this thesis, such as group recommendations, cross-domain recommendations, taking into account economical aspects in predictions, the scalability of predictors and effective ways to evaluate recommender systems. However, as goal-based structuring seems to have a larger impact on how easy it is for people to find interesting items than predictions, which is an outcome of this research, more focus in recommender systems' research should be placed on using goals of the user in recommender systems.

As the linkage from item attributes to gratifications, which in turn lead to the fulfilment of goals, is the basis for assigning items to gratifications, this linkage is extremely important. We recommend that future research in recommender systems should focus on understanding this linkage: how do item attributes lead to gratifications for a user?

Gratification prediction techniques

In order for recommender systems to learn about the linkage between items and gratifications, it might be possible to use techniques similar to prediction techniques that calculate a predicted rating for users for an item; e.g. an adapted version of collaborative filtering can be used to predict what gratifications a specific item will have for a user based on the gratifications assigned to that item by similar users, case-based reasoning can be used to predict to what gratifications an item belongs based on the gratifications assigned to similar items by the user or implicit goals such as categories or genres can be used to assign gratifications to items of which little is known. To convert prediction techniques into gratification prediction techniques, these prediction techniques have to be adapted: instead of predicting a value (the predicted rating) they now need to predict classes (the uses and

gratifications) and instead of predicting just one value, they should now be capable of predicting one or more classes.

Hybrid gratification prediction techniques

Hybrid methods to predict what gratifications belong to an item for a user can also be investigated. A framework similar to the Duine prediction framework for creating hybrid prediction engines can be developed, where prediction strategies are used to decide which gratification prediction technique is best suited to predict which gratifications can be assigned to an item, based upon up-to-date knowledge about the user, other users, the item, other items and the system.

Evaluation of gratification assignments

Parallel to research into gratification prediction techniques, research is also necessary into evaluation methods to objectively measure how accurate gratification prediction techniques are. As gratification prediction techniques have similarities with text classification algorithms, where a piece of text is assigned to one or more classes, a starting point is to investigate how text classifications algorithms are evaluated; however, the subjective nature of gratification prediction techniques has to be taken into account; i.e. some of the problems that occur when evaluating prediction techniques that result in a predicted rating due to subjectivity also apply to gratification prediction techniques.

Predicting user goals

As it is currently very difficult to predict what the goals of a person are at a specific moment, due to difficulties in sensing all aspects of a person such as his emotional and physiological state, research into techniques to determine such aspects is also needed; this is part of the research areas of ambient intelligence and context-awareness. Once it becomes possible to predict a user's current goal(s), recommender systems will be able to better support people in finding interesting items by emphasizing those items that have gratifications assigned to them that match their current goal(s).

Presentation of gratifications

Although the presentation of predictions has been addressed in this thesis, goal-based structuring has raised new questions for the user interface of recommender systems, namely what are the best ways to present gratifications? Issues such as labelling gratifications, icons to represent gratifications and how users can give feedback on assigned gratifications are topics that need to be investigated.

Gratifications in group recommender systems
Goal-based structuring also provides new ways to think about group recommender systems. Instead of trying to find one or more items that are interesting to a group of users based on their short-term interests and long-term interests, we expect that it is also beneficial to take into account the goals of each individual user besides the overall goal of the group; i.e. trying to find items that best match the goals of all users in the group within the boundaries of the group goal.

Generalizability of goal-based structuring
To determine the generalizability beyond TV recommender systems, goal-based structuring also needs to be applied and investigated in other domains. This will require the determination of gratifications people receive from items in a domain, after which the linkage between item attributes and those gratifications has to be investigated and validated in experiments with users.

8.4 In conclusion

In this thesis, technical solutions have been investigated that help people overcome the information overload problem with a focus on the three main aspects of information systems, namely the selection of items, the structuring of items and the presentation of items. The developed solutions in the form of the Duine prediction framework, the goal-based structuring method and the guidelines for the user interface aspects of recommender systems all contribute to develop better recommender systems that can help people to find interesting items. This research also opened up a new avenue of research in the domain of recommender systems where goals of the user become an integral part of a recommender system.

Appendix

A

Screenshots Goal-Based Experiment

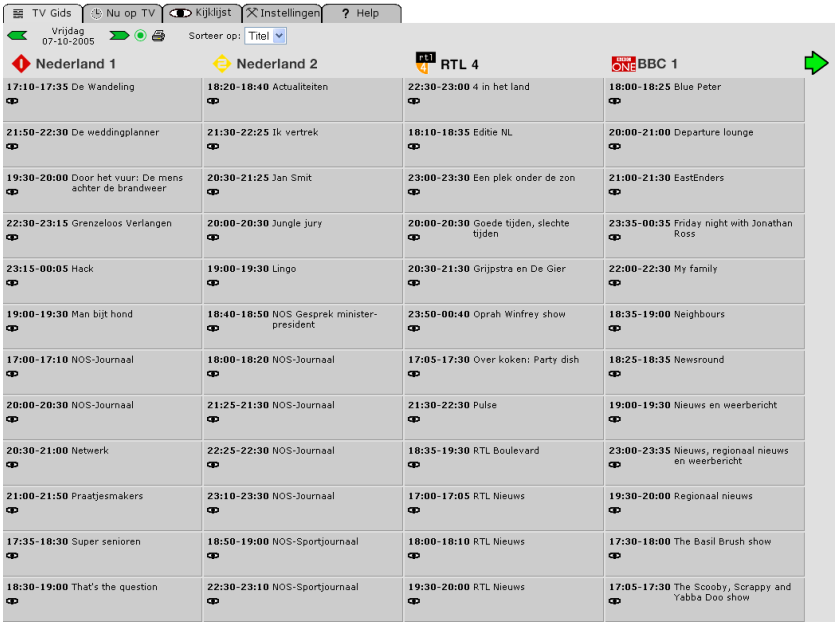


Figure A-1 Channel-based EPG without predictions (experimental group 1)

Figure A-2 Genre-based EPG without predictions (experimental group 2)

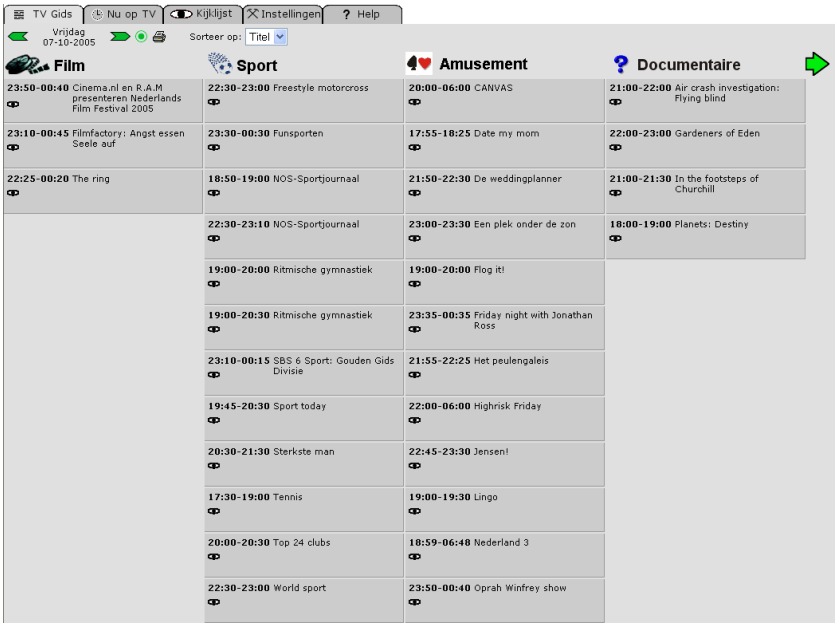
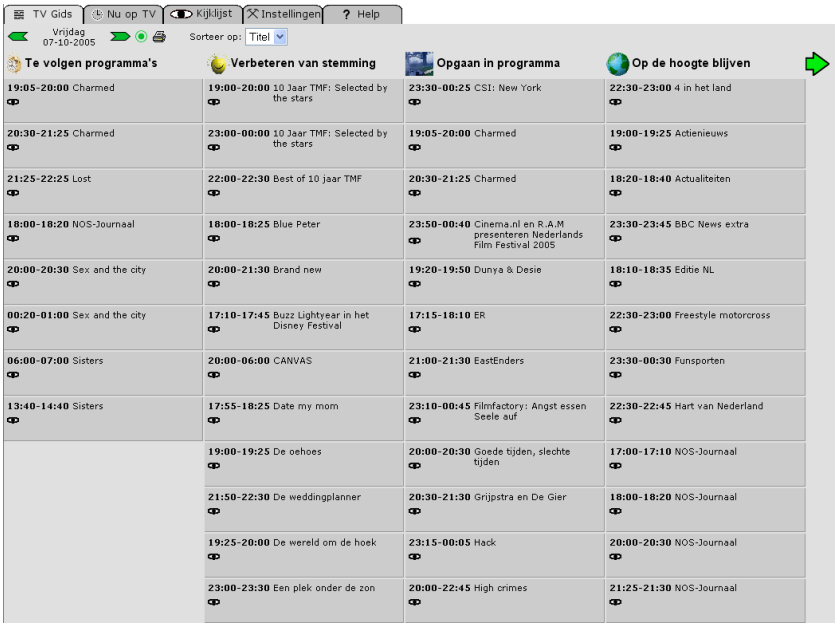


Figure A-3 Goal-based EPG without predictions (experimental group 3)[1]



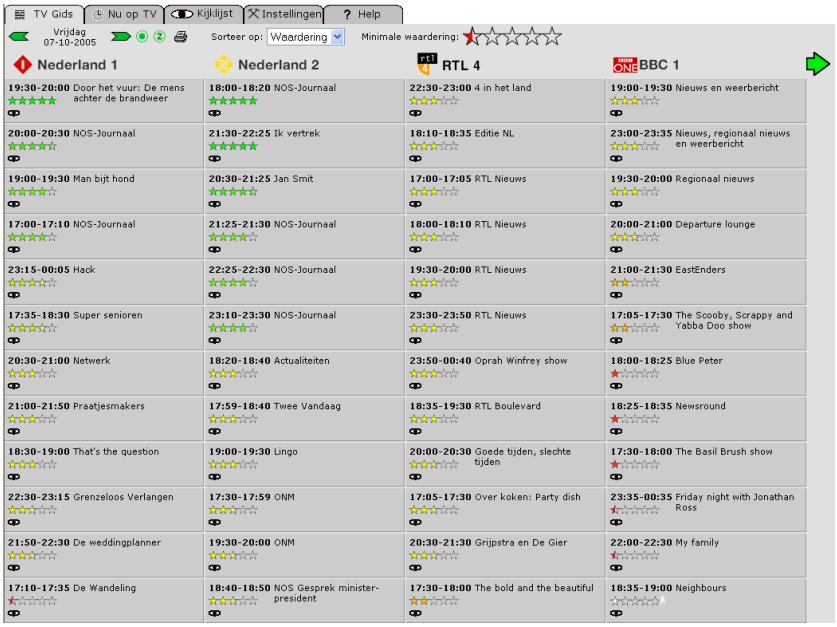


Figure A-4 Channel-based EPG with predictions (experimental group 4)

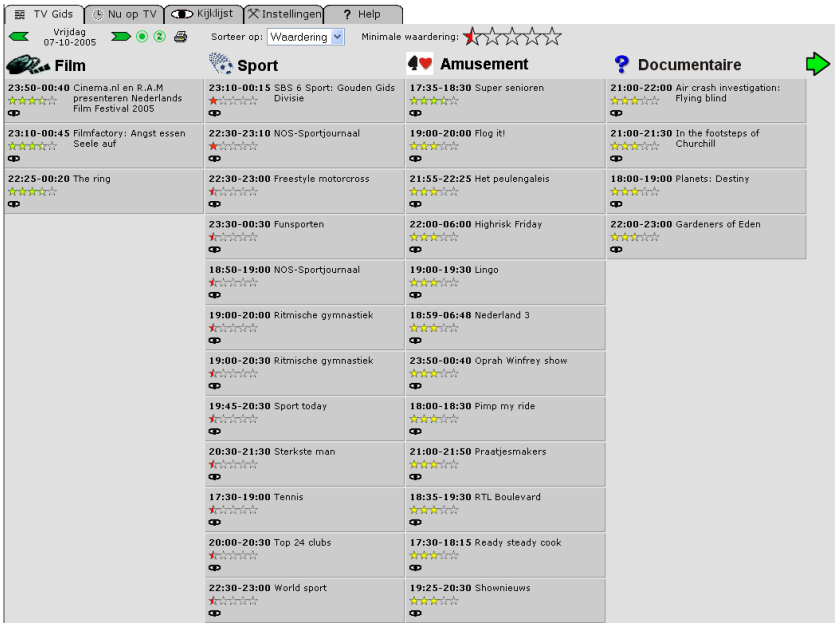
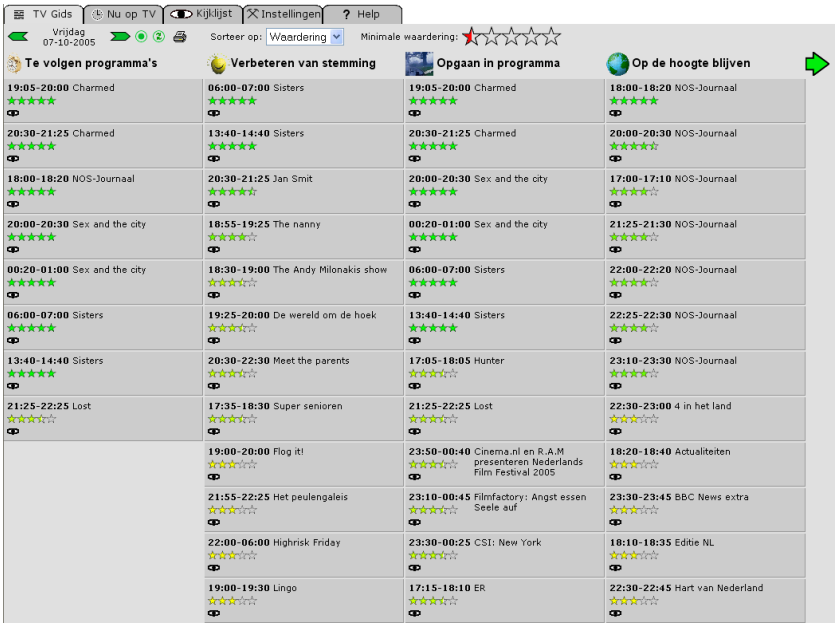


Figure A-5 Genre-based EPG with predictions (experimental group 5)

Figure A-6 Goal-based EPG with predictions (experimental group 6)[1]



[1] English translation of columns in Figure A-3 and Figure A-6:
- Te volgen programma's = Programs to keep up with
- Verbeteren van stemming = Improving my mood
- Opgaan in programma = To loose myself in a program
- Op de hoogte blijven = To be kept up-to-date

Appendix **B**

Histograms of Intent

Figure B-1 Histogram of intent for experimental group 1: Channel-based EPG without predictions

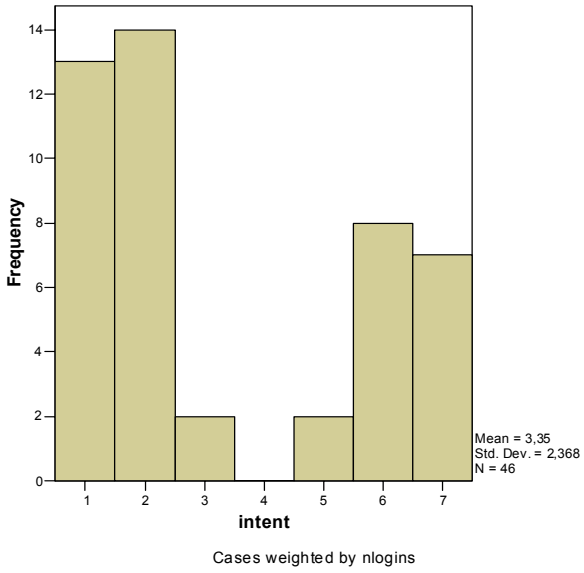


Figure B-2 Histogram of intent for experimental group 2: Genre-based EPG without predictions

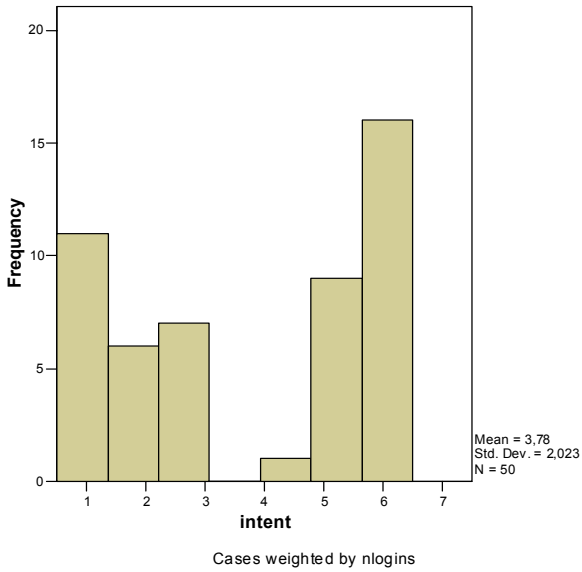
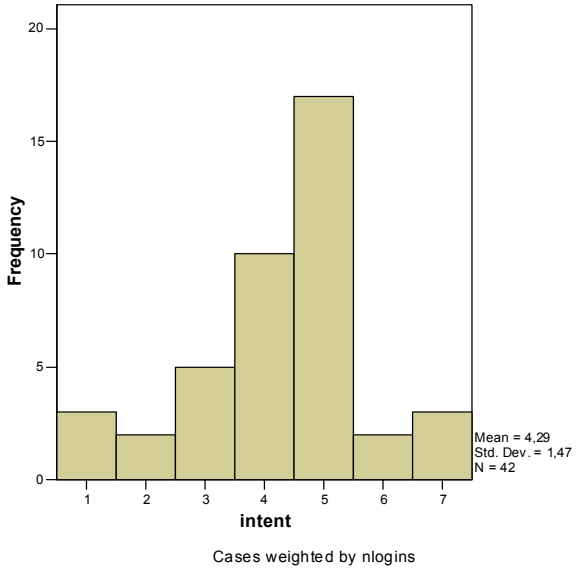


Figure B-3 Histogram of intent for experimental group 3: Goal-based EPG without predictions



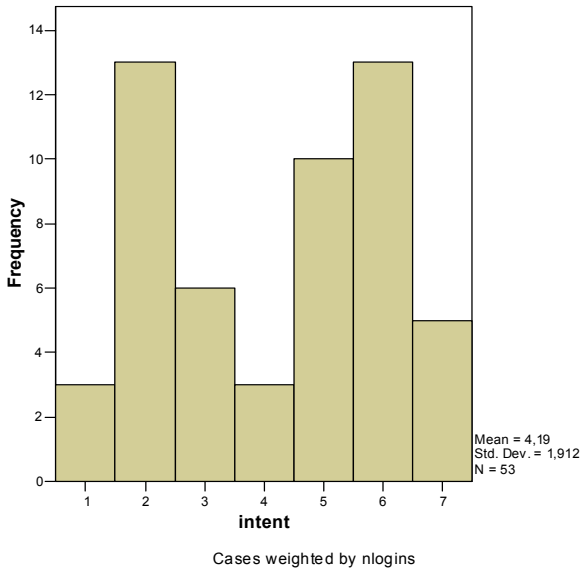


Figure B-4 Histogram of intent for experimental group 4: Channel-based EPG with predictions

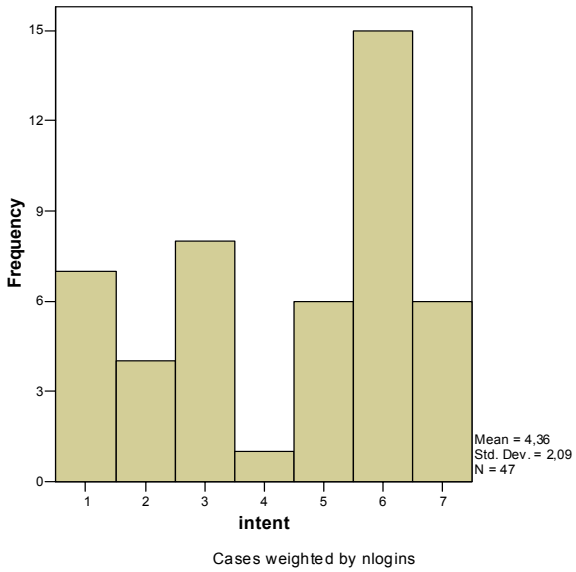


Figure B-5 Histogram of intent for experimental group 5: Genre-based EPG with predictions

Figure B-6 Histogram of intent for experimental group 6: Goal-based EPG with predictions

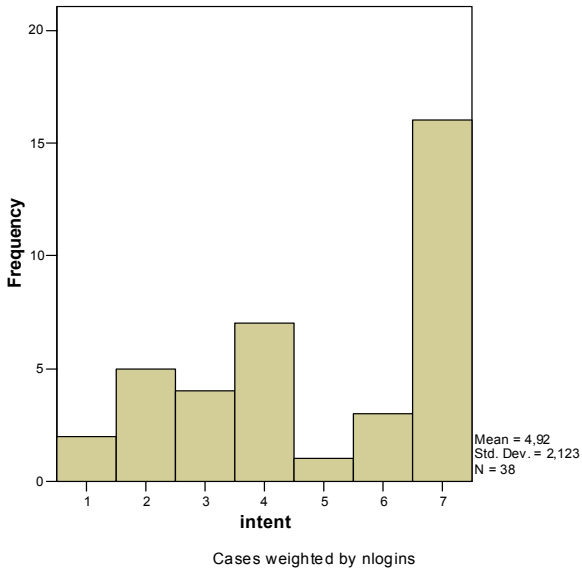


Figure B-7 Combined histogram of intent for EPGs without predictions

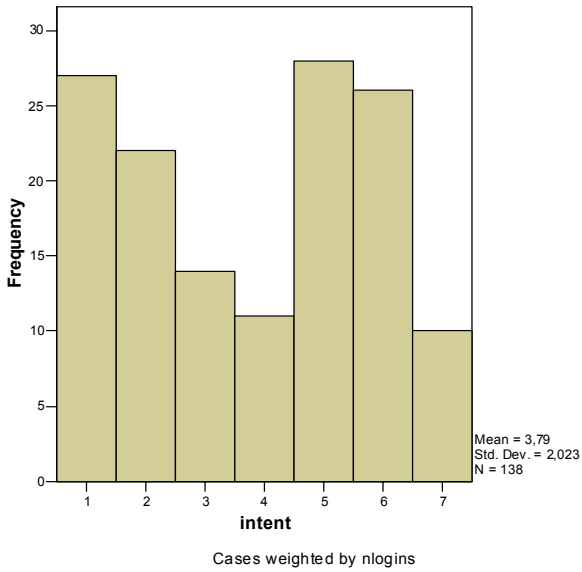


Figure B-8 Combined histogram of intent for EPGs with predictions

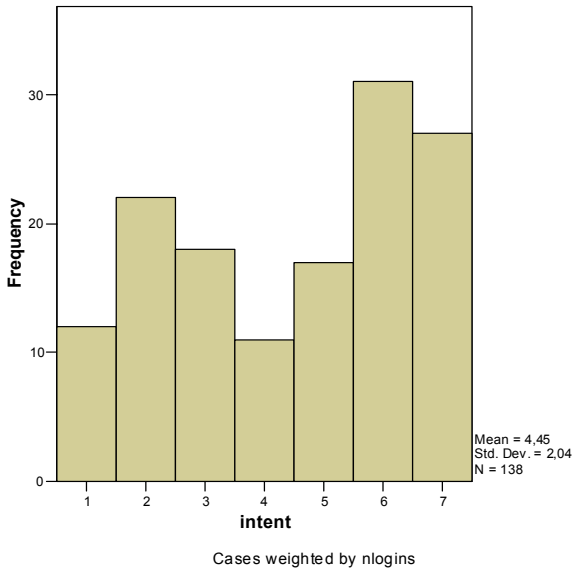


Figure B-9 Combined histogram of intent for channel-based EPGs

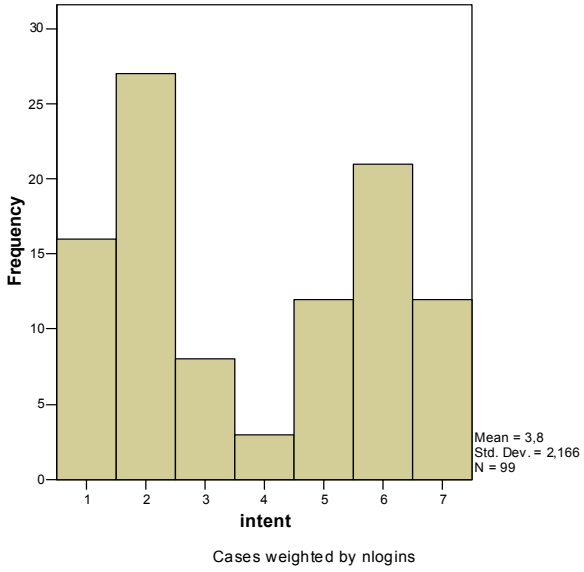


Figure B-10 Combined histogram of intent for genre-based EPGs

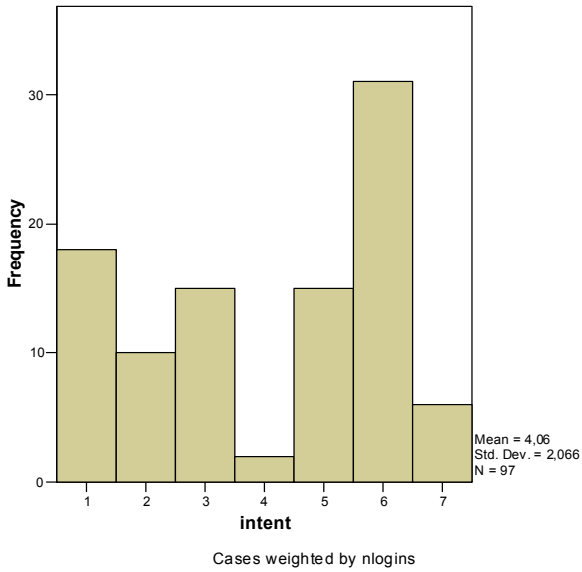
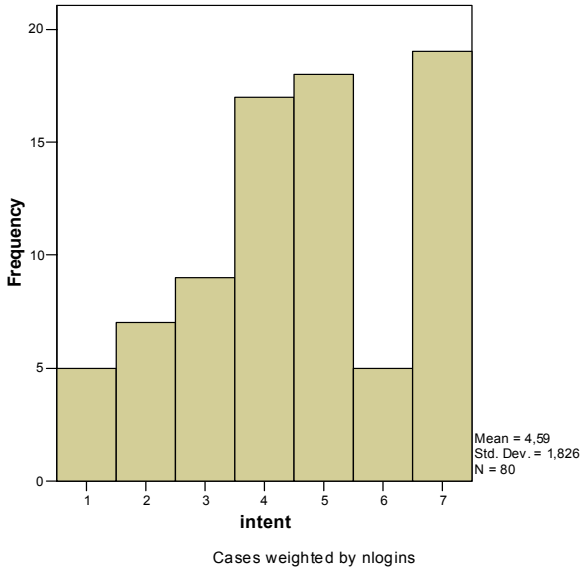


Figure B-11 Combined histogram of intent for goal-based EPGs



Appendix C

Prototype Screenshots

Figure C-1 First prototype: pop-up for feedback and explanations

Figure C-2 First prototype: main screen

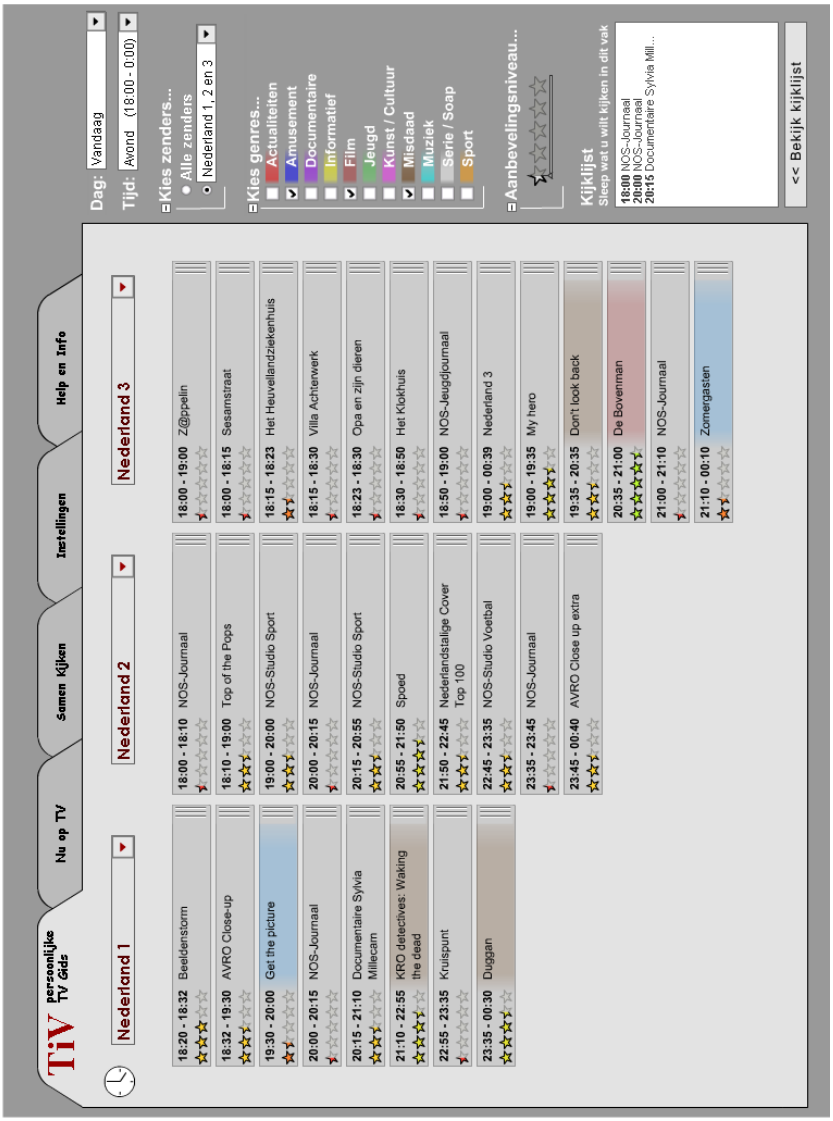


Figure C-3 First prototype: pop-up of detailed TV program description.

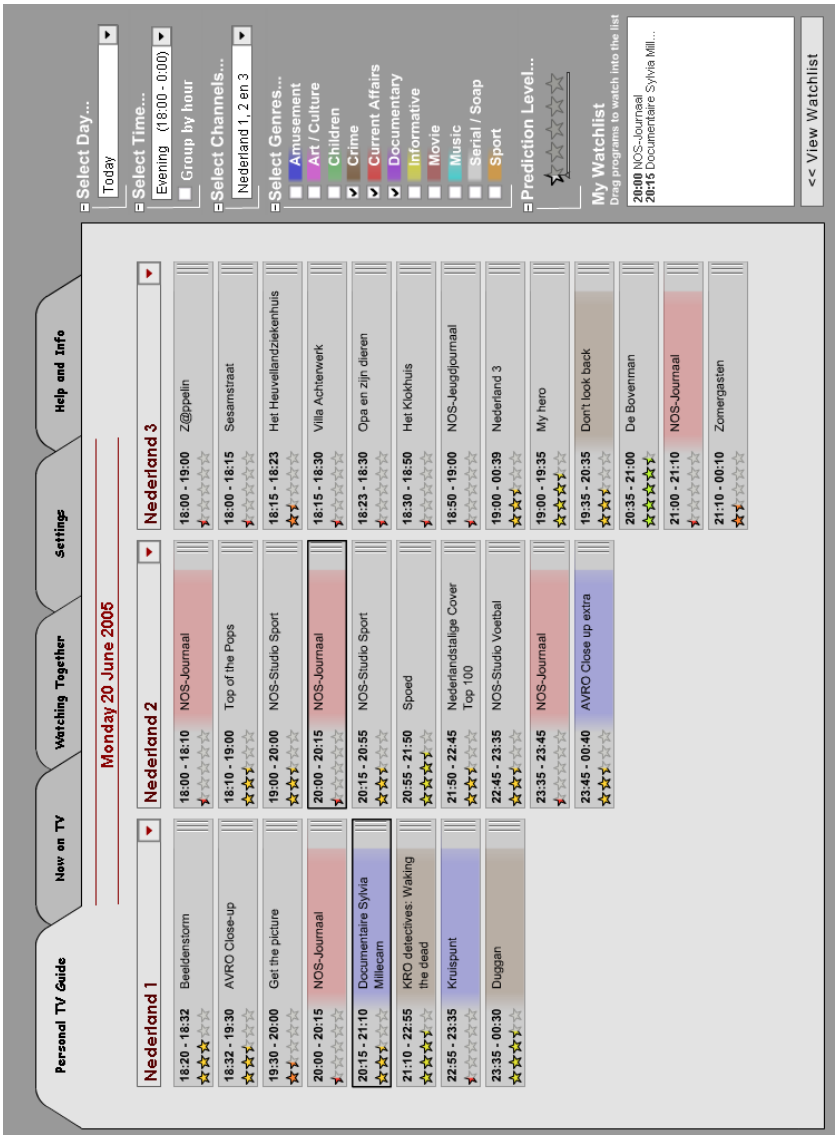


Figure C-4 Improved prototype before usability tests: main screen

Figure C-5 Improved prototype before usability tests: pop-up with detailed TV program description tab active

Figure C-6 Improved prototype before usability tests: pop-up with prediction tab active

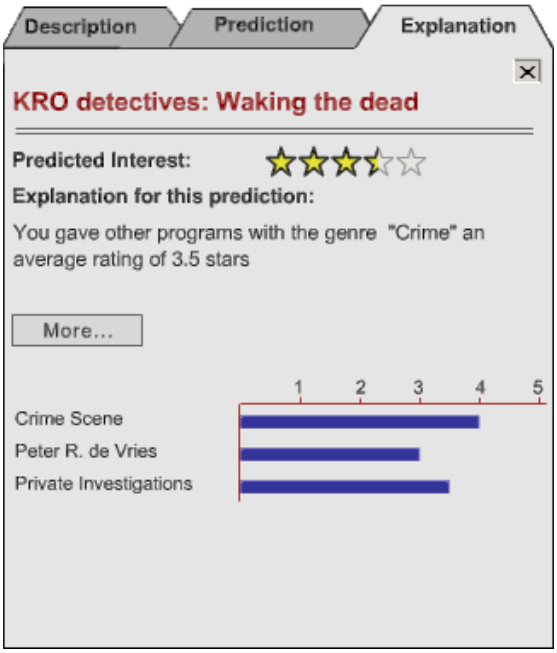


Figure C-7 Improved prototype before usability tests: pop-up with explanation tab active

Summary

The Internet has provided people with the possibility to easily publish and search for information. This resulted in an enormous amount of available information, products and services that also made it a challenge to find that what is really interesting to a person. Finding something really interesting is like searching for the proverbial needle in a haystack. This thesis addresses solutions to support people in finding interesting items by focusing on information systems that automatically learn and adapt their behaviour in order to support their users. The solutions provided in this thesis correspond to the three main processes in personalized information systems: selecting, structuring and presenting items.

The selection of items is concerned with determing what items are interesting to a user, or more precisely predicting how interesting each item is to a specific user. This process is part of so-called recommender systems. For recommender systems, there exist several techniques to predict how interested a user will be in an item. These techniques can be divided in two major groups, namely techniques that base their predictions on the contents of an items and knowledge about the user and techniques that base their predictions on the behaviour of other users and characteristics of those other users, without looking into the items itself.

As every prediction technique has its own strengths and weaknesses, the most accurate predictions can be achieved by combining prediction techniques. The Duine prediction framework described in this thesis provides a way to create prediction engines based on the use of prediction strategies. Prediction strategies generate a prediction for an item and a user by selecting and/or combining prediction techniques based on the most up-to-date knowledge about the current user, other users, the information, other information and the system itself. Prediction strategies can use various decision approaches to learn about and determine when to use which prediction technique. Experiments have shown that manually-created rule-based prediction strategies are capable of providing more accurate

predictions than the individual prediction techniques. On the other hand, case-based reasoning as a decision approach provides more accurate predictions in some datasets, while in other datasets the accuracy levels are comparable with the best prediction techniques. Based on these experiments one can conclude that using prediction strategies can indeed increase the accuracy of predictions in recommender systems, and thus better support people in finding interesting items.

Selecting items for a user by predicting how interested the user will be in those items is not the only way to support people in finding interesting information, products and/or services. The way items are structured when presented to the user is also an important factor. Where interest predictions take into account a user's short-term and long-term interests, the structuring method described in this thesis takes into account a user's immediate interests. This structuring method is based on the possible goals that users can have for items. In an experiment, it was investigated how goal-based structuring helps people in finding interesting items in comparison with using interest predictions. The results show that structuring items based on the user's goals as well as using interest predictions both make it easier for people to find interesting items. The results also show that goal-based structuring helps people more than interest predictions. However, people do need time to get used to goal-based structuring as it makes goals explicit while goals are normally implicit to people.

In personalized information systems, also the presentation of items is an important aspect of how easy it is for people to find interesting items. In this thesis, the user interface aspects of those parts that are specific to the interest predictions of recommender systems have been investigated. The aspects are the presentation of predictions, the presentation of explanations and the way people can provide feedback to a recommender system. A user-centered design study within the domain of electronic television guides resulted in several guidelines for these three user interface aspects.

The solutions developed and described in this thesis, in the form of the Duine prediction framework, the goal-based structuring method and the guidelines for the user interface aspects of recommender systems all contribute to develop better recommender systems that can help people to find interesting items. This research furthermore opened up a new avenue of research in the domain of recommender systems where goals of the user become an integral part of a recommender system.

Samenvatting

Het Internet maakt het makkelijker om informatie te publiceren en op te zoeken. Dit heeft geresulteerd in een enorme hoeveelheid beschikbare informatie, producten en diensten, waardoor het voor veel mensen een uitdaging is om datgene te vinden wat echt interessant is. Het is als zoeken naar de spreekwoordelijke naald in een hooiberg. Dit proefschrift beschrijft oplossingen om mensen te helpen bij het vinden van interessante objecten en richt zich daarbij op lerende informatiesystemen die zichzelf aanpassen om gebruikers beter van dienst te zijn. De oplossingen zijn verwant met de drie belangrijkste processen in gepersonaliseerde informatiesystemen: het selecteren, structureren en presenteren van objecten.

Het selecteren van objecten richt zich op het bepalen welke objecten interessant zijn voor een persoon, oftewel voorspellen hoe interessant ieder object is voor een specifieke gebruiker. Dit is onderdeel van zogeheten aanbevelingssystemen. Er zijn verschillende technieken waarmee een aanbevelingssysteem de interesse van een gebruiker in een object kan voorspellen. Deze technieken zijn in twee categorieën te verdelen, namelijk technieken die hun voorspellingen baseren op de inhoud van een object en kennis over de gebruiker en technieken die hun voorspellingen baseren op het gedrag van andere gebruikers en de eigenschappen van andere gebruikers, zonder naar de inhoud van de objecten zelf te kijken.

Aangezien iedere voorspellingstechniek zijn eigen voor- en nadelen heeft, kunnen accuratere voorspellingen worden verkregen door het combineren van technieken. Het Duine voorspellingsraamwerk, zoals beschreven in dit proefschrift, beschrijft een manier om voorspellingscomponenten te creëren die gebruik maken van voorspellingsstrategieën. Voorspellingsstrategieën genereren een voorspelling voor een object en een gebruiker door het selecteren en/of combineren van voorspellingstechnieken op basis van de meeste actuele kennis over de huidige gebruiker, andere gebruikers, informatie over het object, informatie over andere objecten en het systeem zelf.

Voorspellingsstrategieën kunnen op verschillende manieren leren en beslissen wanneer welke voorspellingstechniek gebruikt moet worden. Experimenten hebben aangetoond dat handmatig gecreëerde regelgebaseerde voorspellingsstrategieën accuratere voorspellingen opleveren dan losse voorspellingstechnieken. Case-based reasoning als voorspellingsstrategie daarentegen levert in sommige datasets wel accuratere voorspellingen op, maar in andere datasets is deze niet accurater dan de beste voorspellingstechniek. Op basis van de experimenten kan wel worden geconcludeerd dat voorspellingsstrategieën de accuraatheid van voorspellingen vergroten in aanbevelingssystemen, die daardoor beter in staat zijn om mensen te helpen bij het vinden van interessante objecten.

Het selecteren van objecten door het voorspellen van de interesse van de gebruiker in die objecten is niet de enige manier om mensen te ondersteunen bij het vinden van interessante informatie, producten en/of diensten. Ook de manier waarop objecten worden gestructureerd bij het presenteren aan de gebruiker is een belangrijke factor. Waar voorspellingen de korte- en lange termijn interesses van mensen in acht nemen, richt de structureringsmethode zoals beschreven in dit proefschrift zich op de onmiddellijke interesses van een persoon. Deze structureringsmethode is gebaseerd op de mogelijke doelen die iemand kan hebben met objecten. In een experiment is onderzocht op welke manier doelgericht structureren mensen helpt ten opzichte van het voorspellen van interesses. De resultaten tonen aan dat zowel doelgerichte structuren als voorspellingen het mensen makkelijker maken om interessante objecten te vinden. Maar doelgericht structureren blijkt mensen meer te helpen dan voorspellingen. Wel hebben mensen tijd nodig om te wennen aan doelgerichte structuren omdat daarbij doelen worden geëxpliciteerd die normaal impliciet blijven.

De presentatie van objecten is ook een belangrijk aspect bij het mensen makkelijk maken om interessante objecten te vinden. In dit proefschrift worden die interfaceaspecten onderzocht die specifiek zijn voor aanbevelingssystemen. Dit zijn de presentatie van voorspellingen, de presentatie van verklaringen en de manier waarop mensen hun mening over een object kenbaar kunnen maken. Een gebruikersgericht ontwerponderzoek met elektronische televisiegidsen heeft geresulteerd in een aantal richtlijnen voor deze drie interfaceaspecten.

De oplossingen die ontwikkeld zijn en beschreven staan in dit proefschrift, zoals het Duine voorspellingsraamwerk, de doelgerichte structureringsmethode en de richtlijnen voor de interfaceaspecten dragen allen bij aan het ontwikkelen van betere aanbevelingssystemen die mensen kunnen helpen bij het vinden van interessante objecten. Daarnaast is ook een nieuwe onderzoeksrichting opengelegd omtrent aanbevelingssystemen waarbij de doelen van gebruikers een integraal onderdeel gaan vormen van aanbevelingssystemen.

References

Abdul-Rahman, A. & Hailes, S. (1997). A distributed trust model. In *New Security Paradigms 1997* (pp. 48-60).

Aggarwal, C.C., Wolf, J.L., Wu, K.L. & Yu, P.S. (1999). Horting Hatches an Egg: A New Graph-Theoretic Approach to Collaborative Filtering. In *Proceedings of KDD-99* (pp. 201-212). San Diego, CA: ACM.

Ali, K. & van Stam, W. (2004). TiVo: Making Show Recommendations Using a Distributed Collaborative Filtering Architecture. In *Proceedings of KDD 2004* (pp. 394-401). Seattle, Washington: ACM.

Ardissono, L., Gena, C., Torasso, P., Bellifemine, F., Chiarotto, A., Difino, A. et al. (2004). User Modeling and Recommendation Techniques for Personalized Electronic Program Guides. In L. Ardissono, A. Kobsa & M.T. Maybury (Eds.), *Personalized Digital Television: Targeting Programs to Individual Viewers*. Dordrecht, the Netherlands: Kluwer Academic Publishers.

Ardissono, L., Goy, A., Petrone, G., Segnan, M. & Torasso, P. (2003). INTRIGUE: Personalized Recommendation of Tourist Attractions for Desktop and Handset Devices. *Applied Artificial Intelligence, 17,* 687-714.

Balabanovic, M. (1997). Content-Based, Collaborative Recommendation. *Communications of the ACM, 40,* 66-72.

Barneveld, J. & van Setten, M. (2004). Designing Usable Interfaces for TV Recommender Systems. In L. Ardissono, A. Kobsa & M.T. Maybury (Eds.), *Personalized Digital Television: Targeting Programs to Individual Viewers* (pp. 259-285). Dordrecht, the Netherlands: Kluwer Academic Publishers.

Basu, C., Hirsh, H. & Cohen, W. (1998). Recommendation as Classification: Using Social and Content-Based Information in Recommendation. In *Proceedings of the 15th National Conference on Artificial Intelligence* (pp. 714-720). Madison, WI.

Baudisch, P. & Brueckner, L. (2002). TV Scout: Lowering the entry barrier to personalized TV program recommendation. In P. De Bra, P. Brusilovsky & R. Conejo (Eds.), *Adaptive Hypermedia and Adaptive Web-Based Systems* (pp. 58-68). Malaga, Spain: Springer.

Bawden, D., Holtman, C. & Courtney, N. (1999). Perspectives on information overload. *Aslib Proceedings, 51,* 249-255.

Bennett, P.N., Dumais, S.T. & Horvitz, E. (2002). Probabilistic Combination of Text Classifiers Using Reliability Indicators: Models and Results. In *Proceedings of SIGIR'02* (pp. 207-214). Tampere, Finland: ACM.

Berners-Lee, T., Hendler, J. & Lassila, O. (2001). The Semantic Web. *Scientific American, May*.

Billsus, D. & Pazzani, M.J. (1999). A personal news agent that talks, learns and explains. In *Proceedings of Autonomous Agents'99* (pp. 268-275). New York: ACM.

Billsus, D. & Pazzani, M.J. (2000). User modeling for adaptive news access. *User Modeling and User-Adapted Interaction, 10,* 147-180.

Breese, J.S., Heckerman, D. & Kadie, C. (1998). Emperical Analysis of Predictive Algorithms for Collaborative Filtering. In *Proceedings of the 14th conference on Uncertainty in Artificial Intelligence*. Madison, WI: Morgan Kaufmann Publisher.

Buczak, A.L., Zimmerman, J. & Kurapati, K. (2002). Personalization: Improving Ease-of-Use, Trust and Accuracy of a TV Show Recommender. In L. Ardissono & A.L. Buczak (Eds.), *Proceedings of Personalisation in Future TV'02* (pp. 3-12). Malaga, Spain: University of Malaga.

Burke, R. (1999). The Wasabi Personal Shopper: A Case-Based Recommender System. In *Proceedings of the 11th Conference on Innovative Applications of Articifial Intelligence* (pp. 844-849). American Association for Artificial Intelligence.

Burke, R. (2000a). A Case-Based Reasoning Approach to Collaborative Filtering. In E. Blanzieri & L. Portinale (Eds.), *Advances in Case-Based Reasoning (5th European Workshop, EWCBR 2000)* (pp. 370-379). New York: Springer-Verlag.

Burke, R. (2000b). Knowledge-Based Recommender Systems. In A. Kent (ed.), *Encyclopedia of Library and Information Systems, Vol. 69, Supplement 32*. New York: Marcel Dekker

Burke, R. (2002). Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction, 12,* 331-370.

Burke, R., Hammond, K. & Young, B. (1997). The FindMe Approach to Assisted Browsing. *IEEE Expert, 12,* 32-40.

Burke, R., Mobasher, B., Zabicki, R. & Bhaumik, R. (2005). Identifying Attack Models for Secure Recommendations. In M.van Setten, S.M. McNee, & J.A. Konstan (Eds.), *Proceedings of the workshop Beyond Personalization 2005: A Workshop on the Next Stage of Recommender Systems Research* (pp. 19-25). San Diego.

Chan, P.K. & Stolfo, S.J. (1993). Toward Parallel and Distributed Learning by Meta-Learning. In *Working Notes AAAI Work. Knowledge Discovery in Databases* (pp. 227-240).

Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D. & Sartin, M. (1999). Combining Content-Based and Collaborative Filters in an Online Newspaper. In *SIGIR'99 Workshop on Recommender Systems: Algorithms and Evaluation*. Berkeley, CA.

Condliff, M.K., Lewis, D.D., Madigan, D. & Posse, C. (1999). Bayesian Mixed-Effects Models for Recommender Systems. In *SIGIR'99 Workshop on Recommender Systems: Algorithms and Evaluation*. Berkeley, CA.

Cooper, C.P., Roter, D.L. & Langlieb, A.M. (2000). User Entertaintment Television to Build a Context for Prevention New Stories. *Preventive Medicine, 31,* 225-231.

Cybenko, G. (1988). *Continuous valued neural networks with two hidden layers are sufficient.* Medford, MA: Tufts University.

Davis, F.D. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. *MIS Quarterly,* 318-340.

Dey, A.K. (2000). *Providing Architectural Support for Building Context-Aware Applications.* College of Computing, Georgia Institute of Technology, Georgia, USA.

Dix, A.J., Finlay, J.E., Abowd, G.D. & Beale, R. (1998). *Human-Computer Interaction*. (2nd ed.) London: Prentice Hall, Europe.

Donohew, L., Palmgreen, P. & Rayburn II, J.D. (1987). Social and Psychological Origins of Media Use: A Lifestyle Analysis. *Journal of Broadcasting & Electronic Media, 31,* 255-278.

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

Dumas, J. & Redish, J.C. (1993). *A Practical Guide to Usability Testing*. New Jersey, USA: Ablex Publishing Corporation.

Edmunds, A. & Morris, A. (2000). The problem of information overload in business organisations: a review of the literature. *International Journal of Information Management,* 17-28.

Ehn, P. (1992). Scandinavian Design: On Participation and Skill. In P.S. Adler & T.A. Winograd (Eds.), *Usability: Turning Technologies into Tools* (pp. 96-132). New York: Oxford University Press.

Eighmey, J. & McCord, L. (1998). Adding Value in the Information Age: Uses and Gratification of Sites on the World Wide Web. *Journal of Business Research, 41,* 187-194.

Faulkner, X. (2000). *Usability Engineering*. Hamsphire, UK: Palgrave.

Floyd, C., Mehl, W.M., Reisin, F.M., Schmidt, G. & Wolf, G. (1989). Out of Scandinavia: Alternative Approaches to Software Design and System Development. *Human-Computer Interaction, 4,* 253-350.

Goren-Bar, D. & Glinansky, O. (2004). FIT-recommeding TV programs to family members. *Computers and Graphics, 28,* 149-156.

Goulding, A. (2001). Information poverty or overload? *Journal of Librarianship and Information Science, 33,* 109-111.

Gutta, S., Kurapati, K., Lee, K.P., Martino, J., Milanski, J., Schaffer, D. et al. (2000). TV Content Recommender System. In *Proceedings of 17th National Conference on AI* (pp. 1121-1122). Austin.

Harst, G. & Maijers, R. (1999). *Effectief GUI-ontwerp*. Schoonhoven, The Netherlands: Academic Service.

Herlocker, J. (2000). *Understanding and Improving Automated Collaborative Filtering Systems.* University of Minnesota, Minnesota.

Herlocker, J. & Konstan, J.A. (2001). Content-Independent Task-Focused Recommendation. *IEEE Internet Computing, 5,* 40-47.

Herlocker, J., Konstan, J.A. & Riedl, J. (2000). Explaining Collaborative Filtering Recommendations. In *Proceedings of CSCW'2000* (pp. 241-250). Philadelphia, PA: ACM.

Herlocker, J., Konstan, J.A. & Riedl, J. (2002). An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information Retrieval, 5,* 287-310.

Herlocker, J. L., Konstan, J.A., Terveen, L.G. & Riedl, J. (2004). Evaluating Collaborative Filtering Recommender Systems. *ACM Transactions on Information Systems, 22,* 5-53.

Hill, W., Stead, L., Rosenstein, M. & Furnas, G.W. (1995). Recommending and evaluating choices in a virtual community of use. In *Proceedings of ACM CHI'95 Conference on Human Factors in Computer Systems* (pp. 194-201). New York: ACM.

Horvitz, E., Breese, J.S., Heckerman, D., Hovel, D. & Rommelse, K. (1998). The Lumière Project: Bayesian User Modeling for Inferring the Goals and Needs of Software Users. In *Proceedings of 14th Conference on Uncertainty in Artificial Intelligence* (pp. 256-265). Madison, WI: Morgan Kaufmann.

Houseman, E.M. & Kaskela, D.E. (1970). State of the art of selective dissemination of information. *IEEE Trans.Eng.Writing Speech III,* 78-83.

Ingwersen, P. (1992). *Information Retrieval Interaction*. London: Taylor Graham.

Jameson, A., Schäfer, R., Simons, J. & Weis, T. (1995). Adaptive Provision of Evaluation-Oriented Information: Tasks and Techniques. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence* (pp. 1886-1893). San Mateo, CA: Morgan Kaufmann.

Kass, R. & Finin, T. (1988). Modeling the User in Natural Language Systems. *Computational Linguistics, 14,* 5-22.

Katz, E. (1959). Mass communication research and the study of popular culture. *Studies in Public Communication, 2,* 1-6.

Lam, W., Mukhopadhyay, S., Mostafa, J. & Palakal, M. (1996). Detection of shifts in user interests for personalized information filtering. In *Proceedings of the 19th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 317-325). Zurich, Switzerland: ACM.

Lee, B. & Lee, R.S. (1995). How and Why People Watch TV: Implications for the future of interactive television. *Journal of Advertising Research, November/December,* 9-18.

Lieberman, H. (1995). Letizia: an agent that assists Web browsing. In *Proceedings of the fourteenth International Conference on AI* (pp. 924-929). Montréal, Canada.

Linden, G., Smith, B. and York, J. (2003). Amazon.com Recommendation: Item-to-Item Collaborative Filtering. *IEEE Internet Computing, January-February,* 76-80.

Lindgaard, G. (1994). *Usability Testing and System Evaluation*. London: Chapman & Hall.

Malone, T.W., Grant, K.R., Turbak, F.A., Brobst, S.A. & Cohen, M.D. (1987). Intelligent information sharing systems. *Communications of the ACM, 30,* 390-402.

Maltz, D. & Ehrlich, E. (1995). Pointing the way: Active collaborative filtering. In *Proceedings of the CHI'95 Human Factors in Computing Systems* (pp. 202-209). New York: ACM.

Mandel, T.W. (1997). *Elements of User Interface Design*. New York: John Wiley & Sons Inc.

Masthoff, J. (2004). Group modeling: selecting a sequence of television items to suit a group of viewers. In L. Ardissono, A. Kobsa, & M.T. Maybury (Eds.), *Personalized Digital Television: Targeting Programs to Individual Viewers*. Dordrecht, the Netherlands: Kluwer Academic Publishers.

McCarthy, J.F. & Anagnost, T.D. (1998). MusicFX: An Arbiter of Group Preferences for Computer Supported Collaborative Workouts. In *Proceedings of CSCW 98* (pp. 363-372). Seattle, Washington, USA: ACM.

McCarthy, K., Reilly, J., McGinty, L. & Smyth, B. (2004). On the Dynamic Generation of Compound Critiques in Conversational Recommender Systems. In P. De Bra & W. Nejdl (Eds.), *Proceedings of Adaptive Hypermedia and Adaptive Web-Based Systems 2004* (LNCS 3137 ed., pp. 176-184). Eindhoven, the Netherlands: Springer.

McDonald, D.G. (1990). Media Orientation and Television News Viewing. *Journalism and Mass Communication Quaterly, 67,* 11-20.

McLaughlin, M.R. & Herlocker, J. (2004). A Collaborative Filtering Algorithm and Evaluation Metric that Accuractely Model the User Experience. In *Proceedings of SIGIR'04* (pp. 329-336). Sheffield, South Yorkshire, UK: ACM.

McNee, S.M., Lam, S.K., Guetzlaff, C., Konstan, J.A. & Riedl, J. (2003). Confidence Displays and Training in Recommender Systems. In *Proceedings of INTERACT '03 IFIP TC13 International Conference on Human-Computer Interaction* (pp. 176-183). Zurich, Switzerland: IOS Press.

McNee, S.M., Lam, S.K., Konstan, J.A. & Riedl, J. (2003). Interfaces for Eliciting New User Preferences in Recommender Systems. In P. Brusilovsky, A. Corbett & F. de Rosis (Eds.), *Proceedings of User Modeling 2003* (LNAI 2702 ed., pp. 178-187). Johnstown, PA, USA: Springer.

Melville, P., Mooney, R.J. & Nagarajan, R. (2002). Content-Boosted Collaborative Filtering for Improved Recommendations. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-2002)* (pp. 187-192). Edmonton, Canada: American Association for Artificial Intelligence.

Mendes, E., Mosley, N. & Watson, I. (2002). A Comparison of Case-Based Reasoning Approaches to Web Hypermedia Project Cost Estimation. In *Proceedings of WWW2002* (pp. 272-280). Honolulu, Hawaii, USA: ACM.

Middleton, S.E., Shadbolt, N.R., & De Roure, D.C. (2004). Ontological User Profiling in Recommender Systems. *ACM Transactions on Information Systems, 22,* 54-88.

Mitchell, T.M. (1997). *Machine learning*. New york, USA: McGraw-Hill.

Muller, M. & Kuhn, S. (1993). Special Issue on Participatory Design. *Communications of the ACM 36*.

Neale, J.M. & Liebert, R.M. (1986). *Science and Behavior: An Introduction to Methods of Research*. (3 ed.) Englewood Cliffs, New Jersey: Prentice-Hall, Inc.

Nielsen, J. (1993). *Usability Engineering*. San Francisco: Morgan Kaufmann Publishers.

Norman, D.A. & Draper, S.W. (1986). *User Centered System Design: New Perspectives on Human-Computer Interaction*. New Jersey: Lawrence Erlbaum.

O'Connor, M., Cosley, D., Konstan, J.A. & Riedl, J. (2001). PolyLens: A Recommender System for Groups of Users. In *Proceedings of ECSCW 2001, Bonn, Germany (*pp. 199-218).

O'Donovan, J. & Smyth, B. (2005). Trust in Recommender Systems. In *Proceedings of IUI'05* (pp. 167-174). San Diego, CA, USA: ACM.

O'Riordan, A. & Sorensen, H. (1995). An intelligent agent for high-precision text filtering. In *Proceedings of CIKM'95* (pp. 205-211).

Oard, D.W. (1997). The state of the art in text filtering. *User Modeling and User-Adapted Interaction, 7,* 141-178.

Olson, J.C. & Reynolds, F. (2001). The Means-End Approach to Understanding Consumer Decision Making. In F. Reynolds & J.C. Olson (Eds.), *Understanding Consumer Decision Making - The Means-End Approach to Marketing and Advertising Strategy* (pp. 3-20). Mahwah, New Jersey: Lawrence Erlbaum Associates, Inc.

Pazzani, M.J. (1999). A Framework for Collaborative, Content-Based and Demographic Filtering. *Artificial Intelligence Review, 13,* 393-408.

Pazzani, M.J. & Billsus, D. (1997). Learning and revising user profile: The identification of interesting web sites. *Machine Learning, 27,* 313-331.

Pittarello, F. (2004). The Time-Pillar World. In L. Ardissono, A. Kobsa, & M.T. Maybury (Eds.), *Personalized Digital Television: Targeting Programs To Individual Viewers* (pp. 287-320). Dordrecht, The Netherlands: Kluwer Academic Publishers.

Quinlan, J.R. (1986). Induction of Decision Trees. *Machine Learning, 1,* 81-106.

Quinlan, J.R. (1993a). *C4.5: Programs for Machine Learning*. San Mateo, CA, USA: Morgan Kaufmann.

Quinlan, J.R. (1993b). Combining Instance-Based and Model-Based Learning. In Utgoff (Ed.), *Proceedings ML'93* (pp. 236-243). San Mateo, CA, USA: Morgan Kaufmann.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. & Riedl, J. (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of CSCW'94* (pp. 175-186). Chapel Hill, NC, USA: ACM.

Resnick, P. & Varian, H.R. (1997). Recommender Systems. *Communications of the ACM, 40,* 56-58.

Reynolds, T.J. & Olson, J.C. (2001). *Understanding Consumer Decsion Making - The Means-End Approach to Marketing and Advertising Strategy*. Mahwah, New Jersey: Lawrence Erlbaum Associates, Inc.

Rich, E. (1998). User modeling via stereotypes. In M.T. Maybury & W. Wahlster (Eds.), *Readings in intelligent user interfaces* (pp. 329-341). San Francisco: Morgan Kaufmann Publishers Inc.

Riedl, J. (2001). Personalization and Privacy. *IEEE Internet Computing, 5,* 29-31.

Riesbeck, C.K. & Schank, R. (1989). *Inside Case-Based Reasoning*. Northvale, NJ: Lawrence Erlbaum Associates.

Rocchio, J.J. (1965). Relevance feedback in information retrieval. In G. Salton (Ed.), *Scientific Report ISR-9 (Information Storage and Retrieval) to the National Science Foundation* (pp. XXIII-1-XXIII-11).

Rubin, A.M. (2002). The Uses-and-Gratifications Perspective of Media Effects. In J. Bryant & D. Zillmann (Eds.), *Media Effects: Advances in Theory and Research* (2 ed., pp. 525-548). New Jersey: Lawrence Erlbaum Associates.

Russell, S. & Norvig, P. (1995). *Artificial Intelligence: A Modern Approach*. Englewood Cliffs, New Jersey: Prentice-Hall, Inc.

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

Sarwar, B.M., Karypis, G., Konstan, J.A. & Riedl, J. (2000). Application of Dimensionality Reduction in Recommender System - A Case Study. In *Proceedings of WebKDD 2000 Web Mining for E-Commerce Workshop.* ACM.

Sarwar, B.M., Konstan, J.A., Borchers, A., Herlocker, J., Miller, B. & Riedl, J. (1998). Using filtering agents to improve prediction quality in the GroupLens research collaborative filtering system. In *Proceedings of CSCW'98* (pp. 345-354). ACM.

Schafer, B., Konstan, J. A. & Riedl, J. (2002). Meta-recommendation Systems: User-controlled Integration of Diverse Recommendations. In *Proceedings of CIKM'02* (pp. 43-51). McLean, Virginia, USA: ACM.

Schafer, J.B., Konstan, J.A., & Riedl, J. (2001). E-Commerce Recommendation Applications. *Journal of Data Mining & Knowledge Discovery,* 115-153.

Schreck, J. (2003). *Security and Privacy in User Modeling*. Dordrecht, The Netherlands: Kluwer Academic Services.

Scott, J. (2000). Rational Choice Theory. In G. Browning, A. Halcli, N. Hewlett & F. Webster (Eds.), *Understanding Contemporary Society: Theories of The Present*. Sage Publications.

Selten, R. (2001). What Is Bounded Rationality? In G. Gigerenzer & R. Selten (Eds.), *Bounded Rationality: The Adaptive Toolbox* (pp. 13-36). Cambridge, Massachusetts, London, UK: The MIT Press.

van Setten, M. (2002). Experiments with a recommendation technique that learns category interests. In *Proceedings of IADIS WWW/Internet 2002*. Lisabon, Portugal.

van Setten, M., Pokraev, S. & Koolwaaij, J. (2004). Context-Aware Recommendations in the Mobile Tourist Application COMPASS. In *Proceedings of Adaptive Hypermedia 2004* (pp. 235-244). Eindhoven, The Netherlands.

van Setten, M., Veenstra, M. & Nijholt, A. (2002). Prediction Strategies: Combining Prediction Techniques to Optimize Personalisation. In L. Ardissono (Ed.), *Proceedings of the workshop "Personalization in Future TV'02" at Hypermedia'2002* (pp. 23-32). Malaga, Spain.

van Setten, M., Veenstra, M., Nijholt, A. & van Dijk, B. (2003). Prediction Strategies in a TV Recommender System: Framework and Experiments. In *Proceedings of IADIS WWW/Internet 2003* (pp. 203-210). Faro, Portugal: IADIS.

van Setten, M., Veenstra, M., Nijholt, A. & van Dijk, B. (2004). Case-Based Reasoning as a Prediction Strategy for Hybrid Recommender Systems. In J. Favela, E. Menasalvas & E. Chávez (Eds.), *Advances in Web Intelligence - Proceedings of the Atlantic Web Intelligence Conference 2004* (pp. 13-22). Cancun, Mexico: Springer.

van Setten, M., Veenstra, M., Nijholt, A. & van Dijk, B. (to be published). Goal-based Structuring in Recommender Systems. *Interacting with Computers*. Elsevier.

Severin, W.J. & Tankard, J.W. Jr. (2001). *Communcation Theories*. (5 ed.) University of Texas at Austin: Addison Wesley Longman, Inc.

Shardanand, U. & Maes, P. (1995). Social information filtering: algorithms for automated "Word of Mouth". In *Proceedings of Human factors in computing systems 1995* (pp. 210-217). New York: ACM.

Shneiderman, B. (1998). *Designing the User Interface, Strategies for Effective Human-Computer Interaction*. (3rd ed.) Longman, USA: Addison Wesley.

Sinha, R. & Swearingen, K. (2002). The Role of Transparency in Recommender Systems. In *CHI '02 extended abstracts on Human factors in computing systems* (pp. 830-831). Minneapolis, Minnesota, USA: ACM.

Smyth, B. & Cotter, P. (2000). A personalised TV listings service for the digital TV age. *Knowledge-Based Systems, 13,* 53-59.

Spolsky, J. (2001). *User Interface Design for Programmers*. Berkeley, USA: Apress.

Stafford, T.F., Stafford, M.R., & Schkade, L.L. (2004). Determining Uses and Gratifications for the Internet. *Decision Sciences, 35,* 259-288.

Tognazzini, B. (2000). If They Don't Test, Don't Hire Them. http://www.asktog.com/columns/037TestOrElse.html [On-line].

Toyama, K. & Horvitz, E. (2000). Bayesian Modality Fusion: Probabilistic Integration of Multiple Vision Algorithms for Head Tracking. In *Proceedings of ACCV 2000, Fourth Asian Conference on Computer Vision*.

Tran, T. & Cohen, R. (2002). Hybrid Recommender Systems for Electronic Commerce. In *Knowledge-Based Electronic Markets, Papers from the AAAI Workshop* (pp. 78-83). Menlo Partk, CA: AAAI Press.

Venkatesh, V., Morris, M.G., Davis, G.B. & Davis, F. D. (2003). User Acceptance of Information Technology: Toward a Unified View. *MIS Quarterly, 27,* 425-478.

Wasfi, A. M. (1999). Collecting User Access Patterns for Building User Profiles and Collaborative Filtering. In *Proceedings of the 1999 International Conference on Intelligent User Interfaces* (pp. 57-64). Redondo, CA.

Watson, I.D. (1997). *Applying Case-Based Reasoning: Techniques for Enterprise Systems*. San Francisco, CA, USA: Morgan Kaufmann Publishers, Inc.

Wärnestål, P. (2005). Modeling a Dialogue Strategy for Personalized Movie Recommendations. In M.van Setten, S.M. McNee & J.A. Konstan (Eds.), *Proceedings of the Beyond Personalization 2005 workshop on the Next Stage of Recommender Systems Research* (pp. 77-82). San Diego, CA, USA.

Weaver III, J.B. (2003). Individual differences in television viewing motives. *Personality and Individual Differences, 35,* 1427-1437.

Weisstein, E.W. (2004). "Path". From MathWorld - A Wolfram Web Resource. http://mathworld.wolfram.com/Path.html [On-line].

Wikipedia (2005a). Wikipedia on Information Overload. http://en.wikipedia.org/wiki/Information_overload [On-line].

Wikipedia (2005b). Wikipedia on Information Retrieval. http://en.wikipedia.org/wiki/Information_retrieval [On-line].

Witten, I. H. & Frank, E. (2000). *Data mining: practical machine learning tools and techniques with Java implementations*. San Diego, California, USA: Morgan Kaufmann Publishers.

Zillmann, D. & Bryant, J. (1986). Exploring the entertainment experience. In J. Bryant & D. Zillmann (Eds.), *Perspectives on Media Effects* (pp. 303-324).

Zimmerman, J. & Kurapati, K. (2002). Exposing Profiles to Build Trust in a Recommender. In *Extended Abstracts Proceedings of Conference on Human Factors in Computer Systems (CHI'2002)* (pp. 608-609). Minneapolis, Minnesota: ACM.

Zimmerman, J., Kurapati, K., Buczak, A. L., Schaffer, D., Gutta, S. & Martino, J. (2004). TV Personalization System: Design of a TV Show Recommender Engine and Interface. In L. Ardissono, A. Kobsa, & M.T. Maybury (Eds.), *Personalized Digital Television: Targeting Programs to Individual Viewers* (pp. 27-51). Dordrecht, the Netherlands: Kluwer Academic Publishers.

Index

The entries refer to the pages where the term is either introduced or defined or discussed.

Telematica Instituut Fundamental Research Series

001 G. Henri ter Hofte, *Working apart together: Foundations for component groupware*
002 Peter J.H. Hinssen, *What difference does it make? The use of groupware in small groups*
003 Daan D. Velthausz, *Cost-effective network-based multimedia information retrieval*
004 Lidwien A.M.L. van de Wijngaert, *Matching media: information need and new media choice*
005 Roger H.J. Demkes, *COMET: A comprehensive methodology for supporting telematics investment decisions*
006 Olaf Tettero, *Intrinsic information security: Embedding security issues in the design process of telematics systems*
007 Marike Hettinga, *Understanding evolutionary use of groupware*
008 Aart T. van Halteren, *Towards an adaptable QoS aware middleware for distributed objects*
009 Maarten Wegdam, *Dynamic reconfiguration and load distribution in component middleware*
010 Ingrid J. Mulder, *Understanding designers, designing for understanding*
011 Robert J.J. Slagter, *Dynamic groupware services: modular design of tailorable groupware*
012 Nikolay K. Diakov, *Monitoring distributed object and component communication*
013 Cheun N. Chong, *Experiments in rights control expression and enforcement*
014 Cristian Hesselman, *Distribution of multimedia streams to mobile Internet users*
015 Giancarlo Guizzardi, *Ontological Foundations for Structural Conceptual Models*

See also: http://www.telin.nl/publicaties/frs.htm

About the author

Mark van Setten has been a researcher at the Telematica Instituut since 1999. His research activities include search and retrieval, personalization, recommender systems, user interface design and user-centred design. He received his master's degree in business information science with honours from the University of Twente in 1997, where he graduated on research into design methods for interaction design. He also holds a bachelor's degree in business economics.

The author has contributed to the design and development of personalized applications and recommender systems in various projects, including GigaPort, the Dutch initiative for the next generation Internet, Freeband Impulse, a Dutch initiative to raise knowledge regarding modern telecommunications within Dutch knowledge institutions to an international top level, and MultimediaN, the Dutch initiative for the research and development of high quality multimedia applications, with a focus on media and information intensive businesses. Furthermore, together with the University of Minnesota he organized the ACM workshop Beyond Personalization 2005 on the next stage of recommender systems research.

Supporting People In Finding Information

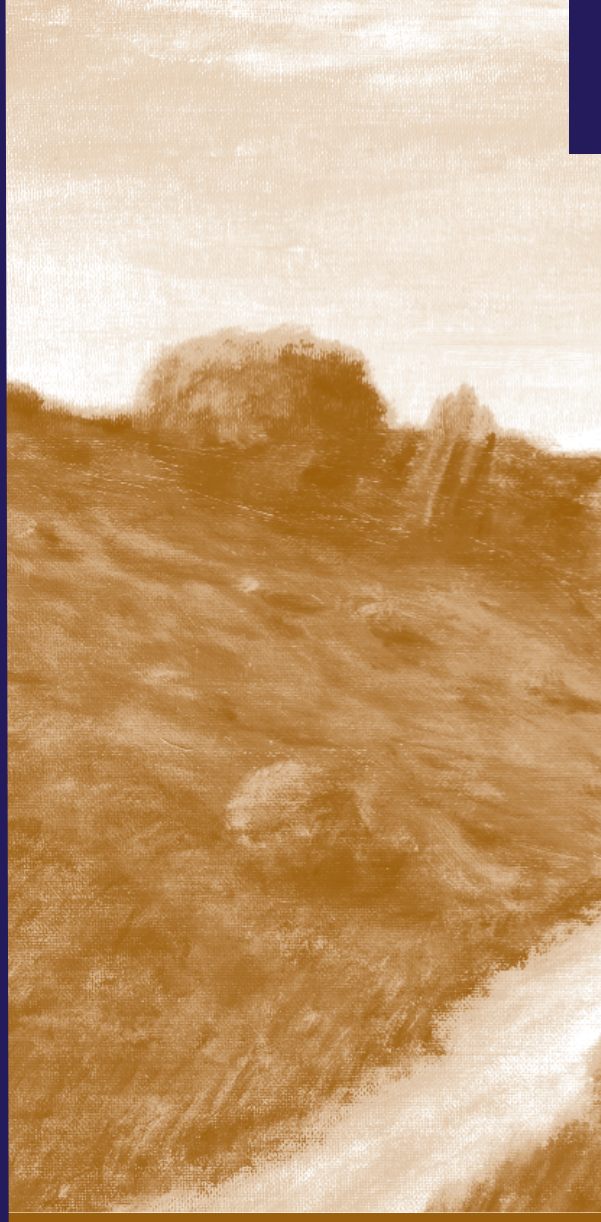
Hybrid Recommender Systems and Goal-Based Structuring

Mark van Setten

The Internet has provided people with the possibility to easily publish and search for information. This resulted in an enormous amount of online available information, products and services that made it a challenge for people to find what is really interesting to them.

Supporting People In Finding Information addresses solutions that can be used to support people in finding interesting items. It provides a framework for the development of hybrid recommender systems that select prediction techniques based on the most up-to-date knowledge, thus increasing the accuracy of recommender systems. It also describes a structuring method that makes it easier for people to find interesting information by structuring items according to the possible goals people have.

Finally, it addresses those user interface aspects that are specific to recommender systems, namely presenting predictions, presenting explanations and providing feedback.



ISBN: 90-75176-89-9

Telematica Instituut