UNIVERSITY OF MINNESOTA

This is to certify that I have examined this bound copy of a doctoral thesis by

John Benjamin Schafer

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Joseph A. Konstan

Name of Faculty Advisor

Signature of Faculty Advisor

Date

John T. Riedl

Name of Faculty Advisor

Signature of Faculty Advisor

Date

GRADUATE SCHOOL

MetaLens:
A Framework for Multi-source Recommendations

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

John Benjamin Schafer

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

July 2001

ACKNOWLEDGEMENTS

This thesis was possible thanks to the help and support of many people.

Thank you to my colleagues in the GroupLens research project and our predecessors. Specifically, thank you to Jon Herlocker for being the student leadership the group needed for so many years. Thank you to Dan Cosley, Nathan Good, Tony Lam, Sean McNee, and Badrul Sarwar for your support of and contributions to this research.

Thank you to my co-advisors, mentors, and, yes, friends, Professors Joseph Konstan and John Riedl. Both in and out of the classroom, you have demonstrated that teaching and learning go hand-in-hand and can occur when you least expect it. As I continue my career, I will always remember the many lessons that you have taught.

Thank you to the *original* Dr. Schafer – my father, John. Growing up, I learned what it means to be a teacher by watching you. If I enjoy my career only half as much as you did yours, I will be a rich man indeed. Thank you to the second Dr. Schafer – my brother, Joe. Your struggles with graduate school always occurred *just* far enough prior to mine that I recognized my own as simply part of the process. Thank you to my mother, Grace, the rest of the "educating" Schafers – Katie, Greg and Shelby – the Frohardts and the Olsons. Your love and support can't be measured or repaid. I hope you know how deeply it was appreciated.

Finally, thank you to "my girls." To my beautiful Margaret, thank you for your radiant smile that always cheered me in the closing days of this degree and gave me the perspective that I desperately needed. To my even *more* beautiful wife, Amy, you didn't complain once when we packed up and moved our lives to further my career (either in coming to Minneapolis or in leaving it). When the process seemed at its worst, that was when I could count most on your steadfast support. I owe you the world that you have made possible.

ABSTRACT

In a world where the number of choices can be overwhelming, recommender systems help users find and evaluate items of interest. They do so by connecting users with information regarding the content of recommended items or the opinions of other individuals. In this thesis, we focus on a new class of recommender systems called meta-recommenders. Meta-recommender systems build on existing recommender technologies by giving users control over the combination of rich recommendation data to yield more personalized recommendations.

The work presented in this thesis makes several significant contributions to the field of recommender systems. We begin by considering the technologies used in creating recommender systems and the variety of ways these technologies are applied and recommendations presented in e-commerce recommender applications. We use this information to create a taxonomy for recommender applications in e-commerce. We also consider correlations between the recommender application models used to recommend products and the sites that choose to implement them.

Next, we introduce meta-recommenders and present the MetaLens Recommendation Framework. This framework serves as a model for how meta-recommenders collect data and generate recommendations that users find understandable, usable, and helpful. A series of controlled use experiments indicate that users want these systems to provide recommendation data alongside the recommendation. Furthermore, when appropriate, users want control over which data is displayed.

Implementation studies show the development of three different recommender systems built within this framework. Analysis of public use of these systems demonstrates that users like, and often prefer, these systems to more “traditional” recommenders. While acceptance comes at a slow pace, users who customized a system were more likely to return to use the system again. Finally, while the quantity and type of recommendation data preferred varies widely from user to user, analysis demonstrates that users want access to as much recommendation data as possible. All told, these results provide a meaningful foundation for the design of future meta-recommenders.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1: Introduction

My poor generation, we're on for the ride,
an ocean of choices, pulled out on the tide.
We're handed a beach ball, and told to pick a side.
Drowned in information. My poor generation.
- "My Poor Generation," Moxy Früvous [51]

On a daily basis we are "drowned in information" as we choose from the overwhelming number of options in "an ocean of choices." To keep abreast of the latest developments in our career field we can choose from a multitude of journal articles, conference proceedings, magazines, textbooks, newsgroups, and web sites. During our personal time we must choose which television show to watch, which movie to see, which CD to listen to, or which book to read. The number of options from which to choose in each of these categories is more than we can possibly process. While the Internet is touted as "the great equalizer" [29], its development has only made the situation worse. Where we were previously limited to the journals carried by our library or the books available at our local bookstore, the Internet has given us access to hundreds of libraries around the world and bookstores that carry millions rather than thousands of titles. The number of choices has become overwhelming, causing a severe case of *information overload*. In the end, it has become impossible even to evaluate all of the information in a given category, let alone "consume" it all.

Fortunately, the same technology that has contributed to the problem has provided us with a portion of the solution. *Recommender Systems* have emerged as powerful tools for helping users reduce information overload. These systems use a variety of techniques to help users identify the items that best fit their tastes or needs.

The remainder of this chapter is organized as follows. First, we will define "recommender systems" as used in this thesis and briefly consider how they are designed to help users find items of interest. Second, we will introduce a new recommender system designed to provide users with personalized control over the combination of rich recommendation data from multiple information sources producing a single, and hopefully more informative, recommendation set. Third, we will provide a list of research challenges and a brief summary of the methods employed in addressing these challenges. Fourth, we follow with a list of the research contributions of this thesis and an outline of the remaining chapters.

1.1 Recommender Systems

According to Resnick and Varian [65], "in a typical recommender system people provide recommendations as inputs, which the system then aggregates and directs to appropriate recipients." This definition includes three classes of systems: suggestion systems, estimation systems, and comment systems. **Suggestion systems** provide a list of candidate items or recommendations. Suggestion systems include the "Your Recommendations" feature at Amazon.com [89], which provides a list of recommended books that a given consumer might like. **Estimation systems** provide an estimate of user preference on specific items or predictions. Estimation systems include the "User's Grade" feature at MovieFinder.com [99], which provides a recommendation for a given movie based on an aggregation of the opinions of users of the site. **Comment systems** provide access to textual recommendations of members of a community. Comment systems include the "Customer's Comments" feature at Amazon.com, which collates the textual recommendations of other users regarding specific products.

We extend the Resnick and Varian definition by using the term "recommender system" to refer not only to systems that specifically *recommend* items but also to those that help users *evaluate* items, such as **Feature-Search systems**. Feature-search systems provide users with the ability to explicitly express an interest in items with a particular set of features. Feature-search systems include the "Shopper" feature at carsdirect.com [90], which allows consumers to select options from a list of automobile features and dynamically see what effect this has on the list of available automobiles. While this system does not offer specific recommendations, it allows a user to receive pseudo-recommendations by identifying items that match the user's needs. These "recommendations" serve as an important first step in the decision-making process for an increasing number of users. In this thesis, we will use the term "recommender system" to refer to any system which provides a recommendation, prediction, opinion, or user-configured list of items that assists the user in evaluating items.

Numerous recommender systems have been built for both research and commercial applications, demonstrating that recommender systems can connect users with useful information. Although the algorithms behind these systems vary, most are based on one or more of three classes of technology. These consist of information filtering and retrieval, data mining, and collaborative filtering. A description of these technologies and a discussion of related prior work are presented in Chapter 2.

1.2 Meta-recommender Systems

This thesis introduces a new class of recommender systems which we classify as *meta-recommender systems*. These systems provide users with personalized control over the generation of a single recommendation list formed from the combination of rich recommendation data from multiple information sources and recommendation techniques. More specifically, we present the design and implementation of a meta-recommender framework which we have named the *MetaLens Recommendation Framework*.

The MetaLens Recommendation Framework (MLRF) is based on an evaluation system model which an increasing number of Internet sites use to help consumers find items of interest. These sites allow consumers to identify a domain of purchase (i.e., a digital camera, computer, or automobile) and narrow the list of products within that domain. Consumers may indicate the features in which they are interested, and the importance of these features in their final decision. Sites turn these requirements into search queries, use information filtering techniques over the attributes of known products in the category, and return ranked lists of "recommended" products. MLRF extends this model by evaluating how well individual consumers will *like* the items and including this evaluation in the recommendation process.

For this thesis, MLRF has been put to use in the domain of movies. As such, it provides recommendations for current theatrical movie releases by combining information filtering-based recommendations with both personalized and generic collaborative filtering-based recommendations. The user interface to MetaLens allows a user to indicate his preferences for specific content including genre, MPAA rating, distance to the theater, ticket price, and show time. Recommendations based on this data are combined with the user's personalized prediction profile, and a single, ranked recommendation list is provided.

1.3 Research Challenges

Three major research challenges were addressed in the completion of this thesis.

1.3.1 Challenge 1: What format should meta-recommendations take?

Meta-recommender systems are a new class of recommender systems. As such, we must carefully consider the recommendations these systems produce. The simplest recommendation from a meta-recommender takes the form of a ranked list of items that meet the requirements of the user. Such a recommendation, however, does not take advantage of the rich content that was used in the recommendation process. By displaying the list in a tabular format, it is possible to include additional information that may help the user make more informed decisions.

This challenge is addressed in Chapters 3, 4, and 6. Chapter 3 discusses current recommendation formats as practiced in electronic commerce recommendation applications. Chapter 4 provides the results of two controlled studies in which subjects were provided with varying quantities of recommendation data along with their recommendations. Results of these studies were used in the design of a publicly available meta-recommender system. Chapter 6 discusses use of this system based on an analysis of data logs tracking which features users included in the submission of their information requirements.

1.3.2 Challenge 2: Which interface do users prefer in a recommender system?

A core assumption in the creation of meta-recommender systems is that the current implementations of recommender systems provide users with incomplete recommendations. Most of these systems provide users with information that is only one piece of the puzzle. In order to visualize the entire picture users must interact with several systems and collect several of these pieces. This assumption leads to two questions. One, are users aware of this deficiency? Two, which interface provides users with the most complete view of the picture?

Chapter 5 addresses this challenge by providing the results of a controlled study. Subjects interacted with three combinations of recommender systems to solve scenarios, and the systems were evaluated based on an analysis of usage data and user-provided feedback.

1.3.3 Challenge 3: How do users interact with meta-recommender systems?

The design of any software system should be based, in part, on how users will interact with the system. By considering how users choose to interact with meta-recommenders, future implementations of these systems can be improved to provide for more natural interaction with the system. As we have built the first meta-recommenders, initial information about how users would choose to interact with such systems was non-existent. User studies provided initial feedback suggesting the nature of this interaction, but more extensive analyses were necessary to provide more detailed feedback on how users interact with meta-recommenders.

This challenge is addressed in Chapters 6 and 7. Chapter 6 provides results of the examination of usage logs to identify the patterns in user information needs at both a personal and a global level. Chapter 7 considers how the degree of information available to a user might affect his system usage and choices.

1.4 Research Methods

Research in this thesis was conducted using three experimental methods. These methods are introduced below and explained in more detail in the following chapters.

Research conducted to consider current practice was performed using data sampling and site analysis from common electronic commerce sites which sell products in a variety of domains. Careful consideration was made to ensure that those selected were a representative sampling of "legitimate" sites. A more detailed explanation is presented in Chapter 3.

Research conducted to answer questions involving the design of a meta-recommender and user preferences in recommender system presentation was performed using controlled experiments. Subjects were provided with different interfaces and different "usage scenarios." For each scenario, they were asked to use the interface to find a movie/theater/show time triple that they felt was appropriate for the usage scenario. Usage logs and user surveys were used to evaluate the effectiveness and user preferences for the different interfaces. More detailed explanations are made in Chapters 4 and 5.

Research conducted to consider user interaction with a meta-recommender was performed through the public deployment of three systems: MetaLens, MetaLite, and MetaClick. These systems built within the MetaLens Recommendation Framework were implemented as part of the MovieLens movie recommender operated by the GroupLens research group [98]. MovieLens has over 100,000 registered users who use the automated collaborative filtering based system to receive recommendations on theatrical releases and rental videos or DVDs. Analysis for this research was conducted by considering both web logs and human-centered usage logs. More detailed explanations are made in Chapters 6 and 7.

All studies involving human participation have been reviewed and approved by the Research Subjects Protection Program of the University of Minnesota (http://www.research.umn.edu/subjects/).

1.5 Contributions

While extensive analysis is reserved for the appropriate chapter, the essential contributions of this thesis are summarized below. In addition to chapter level analysis, the conclusion addresses and synthesizes these contributions in more detail.

1.5.1 Chapter 3

- An analysis of recommender system usage in the “leading” electronic commerce sites from several product domains.
- A taxonomy for recommender system applications in electronic commerce.
- Empirical results suggesting that application models may be affected by the domain of the products being sold.

1.5.2 Chapter 4

- An analysis of different models for presenting recommendations in a meta-recommender system for movies.

- Identification of the preferred model based on empirical user study data.

1.5.3 Chapter 5

- Empirical data which shows that users find the personalized control of meta-recommenders more helpful than "traditional" methods for solving common recommendation situations.

1.5.4 Chapter 6

- Identification of the important movie features that users want to incorporate in their information searches.
- Experimental evidence that frequent users of meta-recommenders will personalize their interface when given the opportunity.
- An analysis of the clusters of information needs met by meta-recommender systems.

1.5.5 Chapter 7

- Consideration of how the meta-recommender framework can be used to create different meta-recommender systems.
- Empirical data suggesting that different users find different meta-recommenders the most helpful.

1.6 Overview of Thesis

An outline of the remainder of the thesis can be found in Table 1.1.

Chapter	Chapter Description
1	**Introduction** (this chapter)
2	**Related Work.** Describes the core work previously conducted in the fields of study addressed in this thesis. More specific related works sections relating to the specific research questions are provided in Chapters 4-7.
3	**A Taxonomy for Recommender Systems.** Provides an extensive analysis of how recommender systems have been applied in the field of electronic commerce. A taxomony for these applications is built, which identifies six application models for recommender systems in electronic commerce. Patterns are presented which identify which application models exist in which product domains.
4	**Recommendation Design for Meta-recommenders.** Provides results of two controlled studies exploring the type and amount of information users wish to have included in recommendations from meta-recommenders.
5	**Comparing Recommender Systems.** Demonstrates that users find value in the recommendations provided by meta-recommenders.
6	**A Meta-recommender in the Wild.** Considers the patterns discovered in the usage of MetaLens – a meta-recommender system in the domain of movies.
7	**Meta* – Creating New Recommenders Using the MetaLens Recommendation Framework.** Presents the development of two additional recommenders using the MetaLens framework and early use statistics suggesting that different users find different recommenders the most helpful system.
8	**Conclusions**

Table 1.1: Thesis overview.

Chapter 2: Related Work

> There are only about 16 usable hours in a person's day, and having access to all the information in the world doesn't change that simple fact.
>
> - John Dvorak [30]

The issue of information overload is not a new problem [17], nor is the study of how to find usable and helpful information from an expanse of options [8], [28], [31], [53], [63], [68]. This thesis builds upon ideas in this prior work and is heavily influenced by its results. This chapter is organized as follows. First, we consider how traditional marketing methods provided a foundation for the growth of recommender systems as a marketing tool in e-commerce. Second, we provide a general overview of some of the fundamental research involving the technologies used in recommender systems. Discussion regarding work related to specific research questions is delayed until the appropriate chapters. Third, we discuss how we have built on this previous work to create this new class of recommender systems.

2.1 Electronic Commerce

In his book Mass Customization [57], Pine argues that companies need to shift from the old world of mass production where "standardized products, homogeneous markets, and long product life and development cycles were the rule" to the new world where "variety and customization supplant standardized products." Pine argues that building one product is no longer adequate. At a minimum, companies need to be able to develop *multiple* products that meet the *multiple* needs of *multiple* consumers. While e-commerce hasn't necessarily allowed businesses to manufacture more products, it has allowed them to provide consumers with more choices. Increasing choice, however, has also increased the amount of information that consumers must process before they are able to select which items meet their needs. To address this information overload, e-commerce stores are applying mass customization principles not to the products but to their presentation in the on-line store [58]. One way to achieve mass customization in e-commerce is the use of recommender systems.

2.1.1 How Sites Use Recommender Applications

Recommender systems are used by e-commerce sites to suggest products and to provide consumers with information to help them decide which products to purchase. Product recommendations can be based on the top overall sellers on a site, on the consumer's demographics, or on an analysis of the consumer's past buying behavior as a prediction for future buying behavior. The forms of recommendation include suggesting products, providing personalized product information, summarizing community opinion, and providing community critiques. Broadly, these recommendation techniques are part of personalization on a site because they help the site adapt itself to each user.

Personalization, to this extent, is one way Pine's ideas can be realized on the Web. Mass customization originally referred to the physical modification of products and services to make them fit each consumer's needs [57]. More recently, mass customization has evolved to encompass a wide range of methods for customizing the *consumer experience* [58]. The consumer experience includes the physical products, which can be customized in function or in appearance, and the presentation of those products, which can be customized automatically or with help from the consumer. Under this broader definition, recommender systems serve to support a customization of the consumer experience in the presentation of the products sold on a web site. In a sense, recommender systems allow for the creation of a new store personally designed for each consumer. Of course, in the virtual world, all that changes is the selection of products shown to the consumer, not an underlying physical store.

Recommender systems evolved in response to an increasing set of choices in products to buy and information to consume, combined with consumer frustration at a decreasing level of professional support for making these choices (i.e., fewer expert shopkeepers). These conditions created challenges for both consumers and merchandisers. Consumers experienced information overload and sought help in selecting from an overwhelming array of products while merchandisers lost their relationships with consumers and sought to rebuild and deepen those relationships by better helping consumers find products of interest.

Recommender systems responded directly to consumers, giving them independent advice modeled after informal "word of mouth." At the same time, new database marketing techniques, data mining, and targeted advertising assisted merchandisers by giving them tools to respond to consumer needs, understand consumer behavior, and best use the limited available consumer attention. This section briefly describes database marketing and targeted advertising technologies and their relationship to recommender systems.

Database marketing is an attempt by businesses to provide more personal service to their consumers. Neighborhood shopkeepers knew their regular visitors and could provide each one with personal assistance, services, and advice. Many businesses today cannot maintain that one-to-one human relationship because of the prevalence of much larger retail stores, low employee-to-consumer ratios, and high employee turnover. Some businesses responded by treating all consumers the same. Others used database marketing to divide consumers into segments based on demographic characteristics such as ZIP Code, income, and occupation and marketed to each segment as a group. In many implementations, consumers treated as part of a segment find that the business no longer understands their individual preferences, needs, or desires.

One-to-one marketing [56] attempts to overcome the impersonal nature of marketing by using technology to assist businesses in treating each consumer individually. Part of one-to-one marketing is the capture and use of consumer preferences (e.g., learning that a particular consumer always wants gifts shipped overnight or that a particular consumer collects an entire line of porcelain dolls). Another part involves changing business practices to use the consumer knowledge gathered by the business.

Recommender systems are a technology that helps merchandisers implement a one-to-one marketing strategy. The recommender system analyzes a database of consumer preferences to overcome the limitations of segment-based mass marketing by presenting each consumer with a personal set of recommendations. Of course, recommender systems are not a complete solution. It is still necessary to record and use additional data, such as preferred credit card and shipping address, to deliver complete one-to-one service to consumers.

Ad targeting, or more generally *offer targeting*, is an attempt to identify which consumers should be made an offer based upon their prior behavior. Traditional marketers watch for a given "event" in a consumer's life and then aim specific advertisements or offers at him. When a consumer applies for his first credit card, he begins receiving offers from numerous banks for their version of the card. When he purchases a house, he begins receiving offers for loan consolidation, second mortgages, life insurance and aluminum siding. When he has a child, he finds himself inundated with advertisements for everything from diapers and formula to book clubs and, once again, life insurance.

Offer targeting treats consumers as both individuals and members of a market group. Offers are typically made to all consumers whose names appear on a list (i.e., the "just acquired a mortgage" list). However, individual consumers are added and removed from these lists based on their individual behavior. Achieving a "life event" gets a consumer added to a list. Consumers who continue to ignore the offers will eventually be removed from the list.

Recommender systems are a technology that can help businesses decide to whom to make an offer. Such systems could suggest to search engines and advertising companies which advertisements or offers to display based on consumer behavior. Yahoo or Excite could use a recommender system to identify which banner ad to display based on which keywords the consumer queried, or to which subsection of the hierarchy a consumer navigated. Not surprisingly, consumers who enter the keywords "Buick Century" in a search engine may find a banner advertising the latest Buick product. In practice, however, these decisions are based on purchased target marketing less the search for "Buick Century" produce results that include a banner advertising a Ford.

2.1.2 Why Sites Use Recommender Applications

Recommender systems are similar to, but also different from, marketing systems and supply-chain decision-support systems. Marketing systems support the marketer in making decisions about how to market products to consumers, usually by grouping the consumers according to marketing segments and grouping the products in categories that can be aligned with the marketing segments. Marketing campaigns can then be run to encourage consumers in different segments to purchase products from categories selected by the marketer. By contrast, recommender systems directly interact with consumers, helping them find products to purchase that meet their needs or desires. Supply-chain decision-support systems help marketers make decisions about how many products to manufacture, and to which warehouses or retail stores to ship the products. These decision-support systems use analytic technology to predict how many of which products will be purchased in each location, so the right products are available for consumers to purchase. Many supply-chain decision-support systems answer questions about aggregates – of all the consumers in Minneapolis, how many will buy toothpaste in February? Recommender systems answer questions about individual consumers – which product will this consumer prefer to buy right now?

Recommender systems include processes that are conducted largely by hand, such as manually creating cross-sell lists, and actions that are performed largely by computer, such as collaborative filtering. We will refer to the latter as *automatic recommender systems*. They have been explicitly designed to take advantage of the realtime personalization opportunities of interactive e-commerce. Accordingly, the algorithms focus more on realtime recommendations and just-in-time learning than on model-building and execution. We study both manual and automatic recommender systems since each offers many interesting ideas about the presentation of recommendations to consumers. This chapter serves as an introduction to the elements of recommender systems and their application to e-commerce.

Recommender systems enhance e-commerce sales in three ways:

Converting Browsers into Buyers: Visitors to a Web site often browse the site without purchasing anything. Recommender systems can help consumers find products they wish to purchase.

Increasing Cross-sell: Recommender systems improve cross-sell by suggesting additional products for the consumer to purchase. If the recommendations are good, the average order size should increase. For instance, a site might recommend additional products in the checkout process, based on the products viewed by the consumer or those already in the shopping cart.

Building Loyalty: In a world where a site's competitors are only a click or two away, gaining consumer loyalty is an essential business strategy [61], [62]. Recommender systems improve loyalty by creating a value-added relationship between the site and the consumer. Sites invest in learning about their consumers, use recommender systems to operationalize that learning, and present custom interfaces that match consumer needs. Consumers repay these sites by returning to the ones that best match their needs. The more a consumer uses the recommender system – teaching it what he wants – the more loyal he is to the site. "Even if a competitor were to build the exact same capabilities, a customer … would have to spend an inordinate amount of time and energy teaching the competitor what the company already knows" [58]. Creating relationships between consumers can also increase loyalty, for consumers will return to the site where their friends are.

2.2 Information Filtering and Information Retrieval

The earliest "recommender systems" were information retrieval systems designed to fight information overload in textual domains. These systems not only find items of high interest but also eliminate those of low interest. While implemented with similar technology, Information Retrieval (IR) and Information Filtering (IF) are considered to be fundamentally different tasks [7].

2.2.1 Information Retrieval

Information retrieval methods are most frequently used in an attempt to satisfy ephemeral information needs using large, relatively static databases. [40], [66]. A user conducting a search in a digital library is considered to be performing an information retrieval task. The user provides a query indicating the keywords which fit his current information need. The search engine examines a previously built content index and *retrieves* the items in the library which match the keywords in the user's query.

Information retrieval is most typically used in domains with relatively static information stores.

Information retrieval interfaces most often take one of two forms. Traditionally such systems use a “form-fill-in query interface.” In these systems, users fill in one or more fields of a query form with descriptions of the items that will meet their needs. Upon completing the form, users submit the query and receive a list of items matching their query.

However, an increasing number of information retrieval systems are providing a mechanism for the input of dynamic or *direct-manipulation queries.* These systems follow a paradigm encouraged by Shneiderman [76]. His belief is that user interfaces for information visualization need to follow the *visual-information-seeking mantra* (“Overview first, zoom and filter, then details on demand”). These systems present users with adjustable sliders, buttons or checklists which users set to represent their query. Unlike form-based systems, these provide a dynamic retrieval list. That is, a user is provided with a retrieval list from the very beginning (“Overview first”). The effect of each modification the user makes to the query interface is immediately reflected in the retrieval list (“zoom and filter”). In doing so, users can focus on how items are “distributed” and what effect each query restriction has on the overall set of items matching query constraints (“details on demand”).

Commercial recommender applications which have been implemented using IR methods often use product catalog database searches to find products that meet the requirements of the consumer. These searches can either be performed in realtime or scheduled to be performed on a periodic basis. Furthermore, they can return results from actual database searches, or they can return pre-defined lists of recommendations culled from users and the “editors” of the site. Examples include the “Advisor” at drugstore.com [93], “Personal Shopper” at eBay [94], and “Shopper” at carsdirect.com.

The Advisor allows users to indicate their preferences when purchasing a product from a category such as “cold and flu remedies.” For example, a consumer might indicate the symptoms she wishes to relieve (runny nose and sneezing), the form in which she would like to administer the relief (caplets), and the "age group" for which the product is intended (adult). The Advisor recommends a list of products meeting these conditions. The Personal Shopper feature at eBay uses a technology derived from information retrieval systems. Users enter a set of keywords of their choosing which describe the type of products they are interested in purchasing. On a periodic basis, the system performs a search over all auctions at the site and sends the user an email with the results of this search. The Shopper feature at carsdirect.com uses a dynamic query approach to information retrieval. Users select and deselect from lists of automobile features and are immediately provided feedback regarding the number of cars which match their current requirements.

2.2.2 Information Filtering

Information filtering methods are most frequently used in an attempt to identify items that match relatively stable and specific information needs. These are most effective in domains where item sets are dynamic, such as those found in email systems or news services. For example, the user of a news service who chooses to register a profile with the service's notification feature is conducting an information filtering task. The user builds a profile – in essence, a persistent query – of the keywords in which she is interested. Alternatively, an information filtering system may observe what a user is "consuming" and automatically build a profile based on the common keywords from these items. As new items are added to the news service, the system *filters* the incoming information streams and notifies the user when it identifies an item that matches the profile she has built. Information filtering is predominantly used in domains with a rapid turnover or frequent additions.

The "Eyes" feature at Amazon.com is loosely based on a keyword filtering system. Consumers explicitly enter requests based upon author, title, subject, ISBN, or publication date – in essence, telling the system what to include in their interest profile. As books are added to the product catalog, Eyes compares their information vector with the interest profiles entered by a consumer. If a match is made, the consumer is notified via email of the new items.

2.2.3 Information Filtering and Retrieval Technology

Before an IR or IF system can attempt to match an item, that item must be "stored" in the computer. Unfortunately, the concepts in information items are rarely configured for easy representation in an information system. Indexing refers to the transformation from the information item to a searchable data structure [44]. First, the information item must be examined and the essential elements extracted. Previously, this examination was limited to a small subset of the entire information item: titles, abstracts, or author-provided keywords. However, the decreasing cost of computer power and storage has allowed these data representations to be more complex and include whole document text.

Once the essential information is extracted, a data structure representing this information must be created. One of the earliest systems, Cornell University's SMART system [16], [68], represented information using a *vector weighting* model. In vector weighting, the semantics of each information item are represented as a vector in which each position in the vector represents an information processing term. The value at each position in the vector can either be binary – indicating the presence or lack of an index term – or weighted – indicating the relative importance of the indexing term within the item. Other techniques used for data representation include *probabilistic weighting models* [25], [34] and *Bayesian models* [44].

Vector weighting systems form a representation of a document based on information extracted from the document. Vector weighting techniques are limited, however, in their scope of document structure as they capture only the limited set of words used in the system and do not represent how the words are related. *N-gram vectors* have evolved as a way to incorporate more of the document's structure and have proven to be less sensitive to spelling errors [11]. An n-gram is a sequence of n letters (typically a minimum of three). Text is converted to an n-gram distribution by counting the number of times each possible n-gram appears in the document. Since for each n there are a finite number of letter sequences of length n, all documents, regardless of domain, can use the same n-gram vector. Tauritz et al. use a weighted n-gram system to provide adaptive information filtering [78].

Information filtering agents are software programs that attempt to act intelligently on behalf of a user. In these types of systems, an initial profile is provided by the user. More importantly, however, these systems are automatically updated based on feedback about whether or not the user likes the items being passed on by the agent. Feedback can be explicitly provided by the user or inferred from positive evidence [74]. A variety of systems have been implemented in domains such as Usenet news [49], email [12], [21], Internet Relay Channels (IRC) [83], and the world wide web [50].

In the remainder of this thesis, we will refer to these technologies under the singular term "information filtering."

2.3 Data Mining

The term "data mining" refers to a broad spectrum of mathematical modeling techniques and software tools that are used to find patterns in sets of data. Recommender systems that incorporate data mining techniques make their recommendations based on knowledge learned from the actions and attributes of users. These systems are often based on the development of user profiles which can be persistent, based on demographic or item "consumption" history data, ephemeral, based on the actions during the current session, or both.

Brachman et al. describe knowledge discovery as an eight stage process [13]. These steps include getting to know the data, data acquisition, data integration, data cleaning, model development, data mining, testing and verification, and application. When creating a data mining-based recommender system, four of these – acquisition, cleaning, mining and verification – are particularly interesting.

Data mining-based recommender applications are common in the domain of electronic commerce. This is true, in part, due to the fact that data acquisition in e-commerce is less challenging than it is in other domains [42]. In contrast to other domains, the data collection process is more controlled; data is automatically being collected electronically, eliminating the degree of manual processing. Furthermore, many of these systems were designed with data mining in mind. This means that data models are more likely to be in a format ready for analysis. Finally, data that is difficult to collect in physical stores (consumer browsing data, non-purchased shopping carts, etc.)

is now quite accessible. Sites are able to store affordably and process extensively large quantities of data that may yield very detailed associations [27].

Although often overshadowed by the more alluring task of actual mining, data preparation is an essential component of the data mining process. Many of the recommendations in electronic commerce are based on analysis of browsing patterns extracted from web server logs. Consequently, these logs must be carefully processed in an attempt to separate individual users and distinct interaction sessions [24], [60].

At this point, data is ready for the actual mining process. Data mining processes generally fall into one of two categories – the development of associations rules and the development of classifiers.

One of the best-known examples of data mining is the discovery of association rules [2], [5], [54], [81]. In the domain of commerce, association rules are relationships between items that indicate a relationship between the purchase of one item and the purchase of another. These may include fine-grained purchase correlations (e.g. consumers who purchase a denim shirt have a high tendency to purchase a cartoon character tie also) or purchases involving temporal patterns (e.g. consumers who have purchased the novel *The Diamond Age* frequently follow this purchase with an interest in biographies about Alan Turing).

The discovery of these associations can help companies in several ways. Traditionally, this information was used to target advertising campaigns to the audience most receptive to the products. When applied in recommender systems, these association rules can be applied to identify in which items a user might be interested based on his demographics or his prior actions. For example, a visitor at the web site CDNOW [91] who is considering the Moxy Füvous album *Thornhill* can use the "Album Advisor" feature to receive recommendations for the albums *Fly* by The Dixie Chicks and the soundtrack to *Run Lola Run*. These recommendations are based on data mining analysis that detects these albums as common purchases made by CDNOW visitors who also bought *Thornhill*.

Classifiers are general computational models for assigning a category to an input. The inputs may be vectors of features for the items being classified or data about relationships among the items. The category is a domain-specific classification such as malignant or benign for tumor classification, approve or reject for credit requests, or intruder or authorized for security checks. One way to build a recommender system using a classifier is to use information about an item and a user as the input and to have the output category represent how strongly to recommend the item to the user. Classifiers may be implemented using many different machine-learning strategies including clustering [14], [39], [46], [80], Bayesian networks [18], [23], and neural networks [19]. In each case, the classifier is trained using a training set in which ground truth classifications are available. It can then be applied to classify new items for which the ground truths are not available. If subsequent ground truths become available, the classifier may be retrained over time.

Once generated, data mining results must be validated. Since "many discovered rules can be spurious, irrelevant, or trivial" [1], there is then a need to develop methods to separate the good rules from the bad. Historically, this separation has been performed by human domain experts. Unfortunately, this method does not scale well. If a business discovers only one rule per consumer the domain experts at a successful company like Amazon.com could still be overwhelmed trying to validate the millions of rules produced. Researchers have successfully identified a variety of methods for post-processing these profile rules. These include similarity-based grouping, template-based filtering, and redundant rule elimination. Combining these methods produced a system which successfully screened 98.5% of the rules from a test set of over 1 million discovered rules [1].

Several commercial recommender systems have their roots in the previously described techniques. Amazon.com relies on an analysis of sales data to identify likely groups for a given product or likely products for a given group. The Purchase Circles feature creates "top-10" lists for market segments based on a given geographic region, employer, educational institution, governmental or other organization. Amazon.com will provide a consumer with recommendations from the purchase circles into which it feels the consumer fits, or the user may select a group with which he feels a particular affinity. The Customers who Bought feature examines consumer purchase data to identify pairs of items which are frequently bought by the same user over a period of time or as a co-purchase. By using the current item at which a consumer is looking, Amazon.com can apply data mining techniques to suggest other books that the consumer might enjoy.

2.4 Collaborative Filtering

Collaborative filtering is an attempt to facilitate the process of "word of mouth." A user provides the system with evaluations of items that may be used to build a profile of her likes and dislikes. This process can be either implicit, by inferring interest based on the item a user views or purchases, or explicit, through the indication of a "rating" for items with which the user is familiar. In some systems, ratings can even take the form of text-based critiques. The simplest of collaborative filtering systems use all members of the system by aggregating all of the evaluations. These provide recommendations by creating ranked "Top N" lists which allow users to find items popular with the community at large, "Average User" scores for each item which provide statistical aggregation of the individual evaluations, or access to the text comments provided by other users.

Most personalized collaborative filtering systems trace their roots to Tapestry [32]. Tapestry is an *active collaborative filtering* system in which a user takes a direct role in the process of deciding whose evaluations are used to provide his recommendations. Operating in the domain of email and Usenet news postings, Tapestry allows the user to create rules or queries that indicate to which other evaluators (users) to listen. Such a system in e-commerce might allow a user to request "show me all books on 'agents' that Nathan has evaluated in which his evaluation contains the words 'outstanding' or 'top notch.'" As such, he is actively pulling recommendations to himself. Reversing the process, the active collaborative filtering system implemented by Maltz and Ehrlich [48] provides users with a system for explicitly recommending items to a specific group of users. As such, users are actively pushing recommendation to other users. Whether push or pull, active collaborative filtering systems work best in domains with small communities since users must be able to identify other users

An alternative lies in the study of *automated collaborative filtering*. In automated collaborative filtering systems, the underlying algorithm automatically handles the process of identifying whose evaluations to consider for each user. The more advanced systems attempt to personalize the process by forming an individualized neighborhood for each user. This neighborhood consists of a subset of users whose opinions are highly correlated with those of the original user. For example, such a system might detect that John and Ben both *liked Hoop Dreams* and hated *Pi*. If Ben just rented *Field of Dreams* and loved it, then this type of system would recognize that John is likely to love it also.

Several automated collaborative filtering systems were started almost simultaneously. The original GroupLens project [43], [64] provides automated neighborhoods for recommendations in Usenet news. Its aim is to help people find articles they will like in the huge stream of available netnews articles. Users rate articles, and GroupLens automatically recommends other articles to them. Users of GroupLens learn to prefer articles with high predictions as indicated by time spent reading. In 1996, GroupLens expanded their recommender systems to include MovieLens, an application of the GroupLens collaborative filtering engine for the domain of movies [26], [35], [36], [70], [71].

Prior to the introduction of MovieLens, Video Recommender [37] also made recommendations on movies. Research on Video Recommender showed that personalized recommendations from collaborative filtering provided a substantial improvement over the use of movie critics. This was an important contribution to the field because it validated that although traditional critics provide an important service, their opinions alone do not make adequate recommendations.

Ringo [75] uses collaborative filtering techniques to provide users with recommendations about audio CDs. When first joining the system, a user is given an initial list of 125 artists to rate. Once the initial profile is established, the user can ask Ringo to suggest new artists or albums she will like. The system can also make predictions about a specific artist or CD. In addition, Ringo has support for message boards on which users can discuss their music tastes. However, these message boards are independent of the recommender system.

While the previous examples all rely on explicit ratings, PHOAKS [79] shows how implicit ratings can be used to create a recommender system. It examines Usenet news postings to find "endorsements" of web sites. It then creates a listing of the top web sites endorsed in each newsgroup. External validation has shown that this technique is effective at identifying popular sites.

More recent collaborative filtering has included the development of interfaces for explaining recommendations [36], [77] and an effort to create systems that are “bigger, stronger, faster.” Much of this latter research has focused on the development of new algorithmic models for producing recommendations. These include rule induction [22], clustering [80], graph theory [3], latent semantic indexing [9], [10], and singular value decomposition [71].

Several collaborative filtering-based recommender systems have proven quite popular at existing e-commerce sites. MYCDNOW uses a mixture of explicit and implicit ratings to provide consumers with personalized recommendations. Consumers explicitly indicate which albums they own and which artists are their favorites. Purchases from CDNOW are entered automatically into the “own it” list. Although “own it” ratings are initially treated as an indication of positive likes, consumers may later distinguish between “own it and like it” and “own it but dislike it.” When a consumer requests recommendations the system will predict six albums the consumer might like based on what her “neighbors” frequently own.

2.5 Hybrid Systems

2.5.1 The Problem

As researchers and e-commerce sites have studied different recommender system technologies, many have realized that no single technology works in all situations. Each has domains in which it is superior, while each has situations in which it is rendered virtually useless. As we report on research crucial in the development of meta-recommenders, it is important to consider and understand the strengths and weaknesses of each of the technologies used in current recommender systems.

Information filtering techniques build a profile of user preferences that is particularly valuable when a user encounters new content that has not been rated before. An avid Woody Allen fan doesn't need to wait for reviews to decide to see a new Woody

Allen film, and a person who hates horror films can as quickly dismiss a new horror film without regret. Furthermore, IF techniques do not depend on having other users in the system, let alone users with similar tastes. Information filtering techniques can be effective, but they suffer certain drawbacks. First, because most IF techniques are based on keyword analysis, they require a source of content information and are thus ineffective in domains without textual descriptions such as music[1]. Second, IF techniques fail to provide much in the way of serendipitous discovery. For example, a profile that learns a user likes Woody Allen would likely never discover a non-Woody Allen drama that just happens to appeal greatly to most Woody Allen fans. Finally, keyword analysis can not measure an item's quality. The previously mentioned profile can only notify a user that a new Woody Allen film has been released. It can not predict if the user will actually like that movie.

One important advantage of collaborative filtering is that it *does not* consider the content of the items being recommended. Rather than map users to items through "content attributes" or "demographics," CF treats each item and user individually. Accordingly, it becomes possible to discover new items of interest simply because other people liked them; it is also easier to provide good recommendations even when the attributes of greatest interest to users are unknown or hidden. For example, many movie viewers may want to see "a movie that makes me feel good" or "a smart, funny movie" as opposed to a movie staring a particular actor or from a particular genre. At the same time, CF's dependence on human ratings can be a significant drawback. For a CF system to work well, several users must evaluate each item; even then, new items cannot be recommended until some users have taken the time to evaluate them. These limitations, often referred to as the *sparsity* and *first-rater problems*, cause trouble for users seeking obscure movies (since nobody may have rated them) or advice on movies about to be released (since nobody has had a chance to evaluate them).

The majority IF and CF-based recommender systems produce realtime, on-line associations. These recommenders use a single-phase, lazy learning approach in which

---

[1] While some would argue that textual descriptions of music *can* be generated, most would agree that what a person likes in music can rarely be captured in words.

they build and update the model while making recommendations in realtime. Data mining-based recommenders differ from these systems largely because they often require two phases. In the learning phase, the data mining system analyzes the data and builds a model of consumer behavior (e.g., association rules). This phase is often very time-consuming and may require the assistance of human analysts. After the model is built, the system enters a use phase in which the model can be rapidly and easily applied to consumer situations. One of the challenges in implementing data mining within organizations is creating the organizational processes that successfully transfer the knowledge from the learning phase into practice in the use phase. Human "operators" must decide how the model will be used in the recommendation process. Often the human overhead required in this phase is too limiting to the overall effectiveness of the system. Additionally, data mining based systems often suffer from the difficulty in tuning minimum support and confidence. Without appropriate values for these thresholds, systems produce either too many or too few rules or have particularly poor recommendation performance [47].

2.5.2 The Solution

Researchers studying the limitations of recommendation technology have suggested that a solution can be formed through combinations of methods. It is believed that by using the strengths of one technology to offset the weaknesses of another, a better system can be built. These hybrid systems serve as precursors to the meta-recommender systems considered in this thesis.

Fab [6] maintains user profiles of interest in web pages using information filtering techniques, but uses collaborative filtering techniques to identify profiles with similar tastes. As a user visits and rates web pages, Fab maintains a keyword vector-based profile for the user. Fab creates groups of users with similar profile vectors and uses their vectors in the generation of "collection agents." These agents evaluate new web pages and recommend those that make it through the "filter" to members of that agent's group. While most CF systems match a user to a single group, Fab matches users with multiple groups based on the combinations of interest in the user's profile vector. Fab-like systems are limited by the strength of their information filtering techniques. Inaccurate content profiles create inaccurate correlations which means inaccurate group formation.

The impact of inaccurate content profiles can be greatly reduced by supplementing CF methods with information filtering agents or *filterbots.* These rate items based on item attributes with implementations which can range from simple text-based agents [69] to personalized agents [33]. Research from the GroupLens project has shown that filterbots improve recommendation quality and coverage in sparse domains by acting as "super-raters" or consumers with the ability to rate every item in the database. The advantage that filterbots have over Fab-like systems is that the content-filters are only a portion of the data used in making collaborative filtering decisions. Theoretically, if bad content-filters are created, they will generate poor user-to-user correlations, and the CF system will select the good content filters instead.

The SmartPad supermarket product recommender system [45] suggests new or previously unpurchased products to shoppers creating shopping lists on a personal digital assistant (PDA). The SmartPad system considers a consumer's purchases across a store's product taxonomy. Recommendations of product subclasses are based upon a combination of class and subclass associations drawn from information filtering and co-purchase rules drawn from data mining. Product rankings within a product subclass are based upon the products' sales rankings within the user's consumer cluster, a less personalized variation of collaborative filtering.

ProfBuilder [86] uses both information filtering and collaborative filtering-based techniques to recommend web pages. As users visit web sites, they provide explicit feedback regarding the quality of the pages they viewed. ProfBuilder uses these evaluations to produce a CF-based recommendation. At the same time, it uses keyword analysis of the pages visited to provide implicit information regarding the user's content interests. ProfBuilder uses these keywords to produce a separate, IF-based recommendation list. The results of each method are displayed in a single interface but are reported in unique recommendation lists.

The Krakatoa Chronicle [41] is an interactive, personalized newspaper which decides story placement based on a combination of user and community scores. User scores are calculated using keyword vector profiles of explicit “seed” information (based on keywords provided by the user to bootstrap the system) and implicit feedback (based on user interaction with the articles in the document). Community scores are calculated by averaging the user scores across the community. While there is the potential for the creation of personalized, collaborative filtering based communities in Krakatoa, the actual implementation uses the average of all users of the system. Krakatoa’s layout control feature provides users with the ability to dynamically personalize the parameters that affect the arrangement and distribution of articles in the document. Parameters consist of a sensitivity factor (high scoring articles are allocated more space), a density factor (how many articles per page), and a combination factor (the ratio with which user and community scores are combined).

Like Krakatoa, Tango [20] recommends articles in an online newspaper. It does so by creating separate recommendations from a collaborative filter and an information filter and merging these using a separate combination filter. Unlike Krakatoa, Tango’s collaborative filter provided true personalization of the community used. Rather than using a “fixed” ratio for the averaging of the recommendations provided by the two filters, the combination filter employed by Tango uses per-user, per-article weights. The calculation of these weights takes into account the degree of confidence each filter has in a particular document’s recommendation, as well as error analysis for each filter’s past performance for the user in question.

Nakamura and Abe [52] describe a system for the automatic recording of programs using a personal video recorder (Tivo, UltimateTV, etc.). They implement a set of “specialist” algorithms that use probabilistic estimation to produce recommendations which are both content-based (based on information about previously recorded shows from the electronic program guide) and collaborative (based on the viewing patterns of similar users). Their system also incorporates an intelligent scheduling algorithm. In most other domains, although based on the system’s recommendations, the final action is taken by the user. With a personalized video system, the action is taken by the video recorder. In principle at least, the recorder can take only a limited number of actions (record too many shows and the storage drive will fill up). Thus, the decision to take action must include information regarding not only which shows are worth recording but also resource allocation (will doing so prevent me from recording a "better" show in a few hours?).

2.6 Meta-recommender Systems

In Chapter 1 we defined meta-recommenders as systems providing users with personalized control over the generation of a single recommendation list formed from a combination of rich recommendation data using multiple information sources and recommendation techniques. In doing so, meta-recommenders extend the concepts introduced by these hybrid systems.

Recall that ProfBuilder uses different recommendation technologies on different data sources to generate a pair of recommendation lists. Hybrid systems such as SmartPad, Digital Video, Fab, and Filterbots use similar methods but extend this concept by integrating an algorithm to merge the recommendation lists. Tango and Krakatoa provide a user partial access to his information filter. Users are given the ability to provide keywords of positive interest which can affect the type of documents returned from the information filter. Krakatoa even provides users access to the combination filter. Through the use of an on-screen slider, users may dynamically adjust the ratio in which the information and collaborative filters are combined.

	Recs. From		**Single**	**User Control Over**	
	CF	**IF**	**Rec List**	**IF Rec**	**Combination**
ProfBuilder	●	●			
SmartPad	●	●	●		
Digital Video	●	●	●		
Fab	●	●	●		
Filterbots	●	●	●		
Tango	●	●	●	◐	
Krakatoa	◐	●	●	◐	●
MetaLens	●	●	●	●	●

Table 2.1: How MetaLens builds on hybrid systems.

As will be discussed in more detail in Chapter 4, the MetaLens Recommendation Framework serves as an extension of hybrid systems by building on the best elements of each of these systems (Table 2.1). MLRF uses multiple recommendation technologies to generate scores from multiple sources of recommendation content. MLRF provides users with explicit control over a personalized collaborative filter and complete access to the construction of the information and combination filters. It is our belief that this access provides users with the ability to receive more complete and meaningful recommendations under situations involving a wide range of information needs.

Chapter 3: A Taxonomy for Recommender Systems[2]

> Our vision is that if we have 20 million customers, we should have 20 million stores.
>
> –Jeff Bezos, CEO of Amazon.com™ [87]

In previous chapters, we have addressed three classes of technology used in recommender systems and some of the relevant research regarding these technologies. While this discussion is an important first step, there is far more to the development of a recommendation application than deciding which algorithm to use. In fact, the same algorithm could be used to produce recommender systems with very different appearances. For example, consider an information retrieval algorithm. This algorithm could be incorporated as part of a dynamic search interface that helps users find items to evaluate. It could also be implemented as part of a recommender system suggesting "If you like this book, then you might like these other books." In the latter case, the algorithm would automatically search for books with similar keywords in the subject description. Similarly, two recommender systems with nearly identical appearances could be powered by very different algorithms. For example, while the "If you like" system just discussed could be run by an information retrieval algorithm, it could also be based on a data mining algorithm searching co-purchase data. Because of the great variability in recommender systems, it is important to consider the different dimensions upon which they may be classified. This chapter presents the results of an analysis of recommender systems implemented at electronic commerce sites and the construction of a taxonomy for such systems.

This chapter is divided into the following sections. First, we consider twenty recommendation applications at six e-commerce sites and identify what these systems have in common as well as what separates them from each other. Second, we present a taxonomy for recommender applications, classifying them based on the inputs to the recommender process, the method used to generate recommendations, the outputs of the recommendation process to the consumer, and the degree of personalization. Third, we

---

[2] Portions of this chapter have been previously published as a conference paper in the ACM Conference on Electronic Commerce [72] and as a journal article in Data Mining and Knowledge Discovery [73].

examine the patterns that emerge from the taxonomy and identify six models of recommender applications. These six models are currently the dominant uses of recommender systems in e-commerce. Fourth, we describe an analysis of approximately 450 recommendation applications from nearly 150 e-commerce sites across ten product domains. We consider how the products being sold may affect the recommender applications implemented at the site.

3.1 Recommender Applications in Electronic Commerce

In the following section, we present six e-commerce businesses that use one or more variations of recommender system technology in their web sites. For each site, and each variation, we give a brief description of the features of the system. In later sections we refer to these examples as we explain the types of recommendations provided, the type of technology used, and the types of information gathered. For organizational purposes these sites have been alphabetized. The figures referenced in this section are located in Appendix I.

3.1.1 Amazon.com

Amazon.com™ got its start in 1995 as an Internet-based bookseller. They have since expanded to offer the Earth's Biggest Selection™ of products, including free electronic greeting cards, online auctions, CDs, videos, DVDs, toys and games, and electronics. The following section will focus on recommender systems in the *book* section of Amazon.com.

Customers Who Bought: Like many e-commerce sites, Amazon.com is structured with an information page for each book, giving details of the text and purchase information. The Customers Who Bought feature (Figure I.1) is found on the information page for each book in their catalog. It is, in fact, two separate recommendation lists. The first recommends books frequently purchased by consumers who purchased the selected book. The second recommends authors whose books are frequently purchased by consumers who purchased works by the author of the selected book.

Your Recommendations: Amazon also encourages direct feedback from consumers about books they have read. Consumers rate books they have read on a 5-point scale from "hated it" to "loved it." After rating a sample of books, users may request recommendations for books that they might like. At that point, a half dozen non-rated texts are presented that correlate with the user's indicated tastes. Figure I.2 shows a sample screen from Your Recommendations.

Eyes[3]: The Eyes feature (Figure I.3) allows consumers to be notified via email of new items that have been added to the Amazon.com catalog. Users enter requests based upon author, title, subject, ISBN, or publication date information. Users can use both simple and more complex Boolean-based criteria (AND/OR) for notification queries. One of the interesting variations of the Eyes system allows requests to be entered directly from any search results screen, creating a persistent request based on the search.

Amazon.com Delivers: Amazon.com Delivers (Figure I.4) is based on a newsletter model for marketing. Consumers select checkboxes to choose from a list of specific genres (Oprah books, biographies, cooking). Periodically the editors at Amazon.com send their latest recommendations by email to subscribers in each category.

Bookstore Gift Ideas: The Gift Ideas feature allows consumers to receive recommendations from editors. Users pick a category of books for which they would like some suggestions. By navigating to that section of the "Gift Department," they can view a general list of recommendations created by the editors of Amazon.com. They also can select to view recommendations in one of a predefined list of categories including Globetrotter, Entrepreneur, and Teens (Figure I.5). In many ways this serves as an online version of the Amazon.com Delivers feature. However, consumers can be provided with recommendations anonymously since there is no need to register with the site as there is with Delivers.

Customer Comments: The Customer Comments feature allows consumers to view text recommendations provided by other consumers. Located on the information page for each book is a list of 1-5 star ratings and written comments provided by

---

[3] Replaced by "Amazon.com Alerts"

consumers who have read the book in question and submitted a review. Users have the option of incorporating these recommendations into their purchase decision. Furthermore, users can “rate the comments.” With each comment is the yes or no question “Did this comment help you?” Results are tabulated and reported such as “5 of 7 people found the following review helpful” (Figure I.6).

Purchase Circles: The Purchase Circles feature (Figure I.7) allows consumers to view the "top-10" list for a given geographic region, company, educational institution, government or other organization. For example, a user could request to see what books are the bestsellers for consumers at Oracle, MIT, or residents of New York City. Purchase Circles provide another “fellow consumer” form of recommendation by allowing a user not only to see what others are reading but also to personalize the recommendations by allowing her to select a “domain” with which she associates herself. A user can view Purchase Circles by navigating to the Circle that interests her.

3.1.2 CDNOW

CDNOW “is a leading online music destination, offering the ultimate connection to the world of music.” [91] Launched in 1994, CDNOW has frequently been among the first to offer its users a variety of web-based innovations including sound samples, encoded music for online delivery, and many of the personalization features discussed in this section.

Album Advisor: The Album Advisor feature of CDNOW works in three different modes. The first two are similar to the Customers Who Bought feature of Amazon.com. Users locate the information page for a given album or artist. The system then recommends ten other albums related to the album or artist in question. Results are presented as “Customers who bought X also bought items in set S” or “Customers who bought items by Y also bought set T” (Figure I.8). The third mode works as a gift “advisor.” Users type in the name of up to three artists, and the system returns a list of ten albums CDNOW considers similar in style and taste to the artists in question.

Related Artists: The Related Artists feature of CDNOW (Figure I.9) works on the assumption that if a consumer likes a given performer, there is a group of artists with similar styles that she will also like. Users locate an artist and select the Related Artists link. Upon doing so, they are provided with a list of these artists who are considered to be "similar artists" and a list of artists who are considered to be among the "roots and influences" for the selected artist.

Buyer's Guides[4]: The Buyer's Guide feature at CDNOW allows consumers to receive recommendations based on a particular genre of music. Users browse a list of genres provided by the site, including categories such as British Invasion, Big Chilling, and The 80s Fan (Figure I.10). Selecting one of the links from this list takes the user to a new list of albums the editors consider the essential part of this genre.

Artist Picks: The Artist Picks feature provides recommendations directly from the artists. Each week a different artist lists the albums that shaped his or her taste as well as what is currently in their CD player.

Top 100: Traditionally, hype and "bestseller" status have been used by commerce sites to make recommendations to their consumers. The Top 100 feature allows visitors to CDNOW to receive this type of recommendation (Figure I.11). The 100 are drawn from the sales figures of the site and can theoretically be continuously upgraded to reflect actual sales.

My CDNOW: My CDNOW enables consumers to set up their own music store based on albums and artists they like. Users indicate which albums they own and which artists are their favorites. Purchases from CDNOW are entered automatically into the "own it" list. Although "own it" ratings are initially treated as an indication of positive likes, users can go back and distinguish between "own it and like it" and "own it but dislike it." When a user requests recommendations, the system predicts six albums she might like based on what she already owns. The user can provide feedback by selecting "own it," "move to wish list" or "not for me" for any of the albums in her prediction list. The albums recommended change based on the feedback. Figure I.12 shows a sample screen from My CDNOW.

---

[4] Replaced by "Gift Guide."

3.1.3 drugstore.com

Created in 1999, drugstore.com™ is a retail store and information site for health, beauty, wellness, personal care, and pharmacy products.

Advisor: The Advisor feature at drugstore.com allows consumers to indicate their preferences when purchasing a product from a category such as "suncare" or "cold and flu remedies." For example, in the latter, a user might indicate the symptoms he wishes to relieve (runny nose and sneezing), the form in which he wants the relief (caplets) and the "age" of patient to whom he want to administer the product (adult). Upon receiving this information, the Advisor returns a list of recommended products meeting these conditions.

Test Drives: In the Test Drives feature, a team of consumer volunteers is sent a new product. These "fellow consumers" provide reviews of the product including a star rating and text comments (Figure I.13).

3.1.4 eBay

Founded in 1995 as a way to connect collectors of Pez dispensers, eBay is the world's largest online trading community.

Feedback Profile: The Feedback Profile feature at eBay allows both buyers and sellers to contribute to feedback profiles of other consumers with whom they have done business. The feedback consists of a satisfaction rating (satisfied/neutral/dissatisfied) as well as a specific comment about the other consumer. Feedback is used to provide a recommender system for purchasers as well as buyers, who are able to view the profile of the other individual. This profile consists of the distribution of satisfaction ratings for the past 7 days, the past month, and the past 6 months, as well as an overall summary (Figure I.14). Upon further request, consumers can browse the individual ratings and comments concerning the seller/buyer (Figure I.5).

Personal Shopper[5]: The personal shopper feature of eBay allows a consumer to indicate an item he is interested in purchasing. The user defines a search based on a set of keywords of his choosing, including his price limit. For a pre-selected length of time

---

[5] Replaced by "Favorite Searches."

(30, 60, or 90 days), the site regularly performs the consumer's search over all auctions at the site and sends him an email with the results of this search (Figure I.16).

3.1.5 MovieFinder.com

MovieFinder is the movie site maintained by E! Online. Founded in 1996, E! Online contains entertainment news, information and reviews and is a subsidiary of E! Networks (available on many U.S. cable and satellite systems).

Users Grade/Our Grade: Both the Users Grade and the Our Grade features report a letter grade recommendation to the consumer. The Users Grade feature allows consumers to register with the site and give letter grades (A-F) to the movies they have seen. These grades are then averaged over all consumers and reported as the Users Grade. The Our Grade feature provides consumers with a grade from the editors of E! Online. Thus, consumers viewing the information page for *Crouching Tiger, Hidden Dragon* might find that it gets a grade of A from the editors and a grade of A- from other consumers who have rated it (Figure I.17).

Top 10: The Top 10 feature at E! Online allows consumers to get recommendations from the editors in a category of their choice. Consumers select a category from a list of previously defined categories such as chick flicks (Figure I.18), sex scenes, and movies from books. Selecting a list takes the user through descriptions of the top ten movies in that category as defined by one of the editors of E! Online.

3.1.6 Reel.com

Reel.com™ provides movie-related information and products. "Reel.com was named the most popular Web site for gathering information about films playing in movie theaters, according to a consumer survey conducted by Greenfield Online and ASI Entertainment (March 2000)." [100].

Movie Matches: Similar to Amazon.com's Customers Who Bought, Reel.com's Movie Matches provides recommendations on the information page for each movie. These recommendations consist of "close matches" and/or "creative matches." Each set contains up to a dozen hyperlinks to the information pages for each of these "matched" films. The hyperlinks are annotated with one-sentence descriptions of how the new movie is similar to the original movie in question (e.g. "Darker thriller raises similarly disturbing questions…"). Figure I.19 shows a sample screen from Movie Matches.

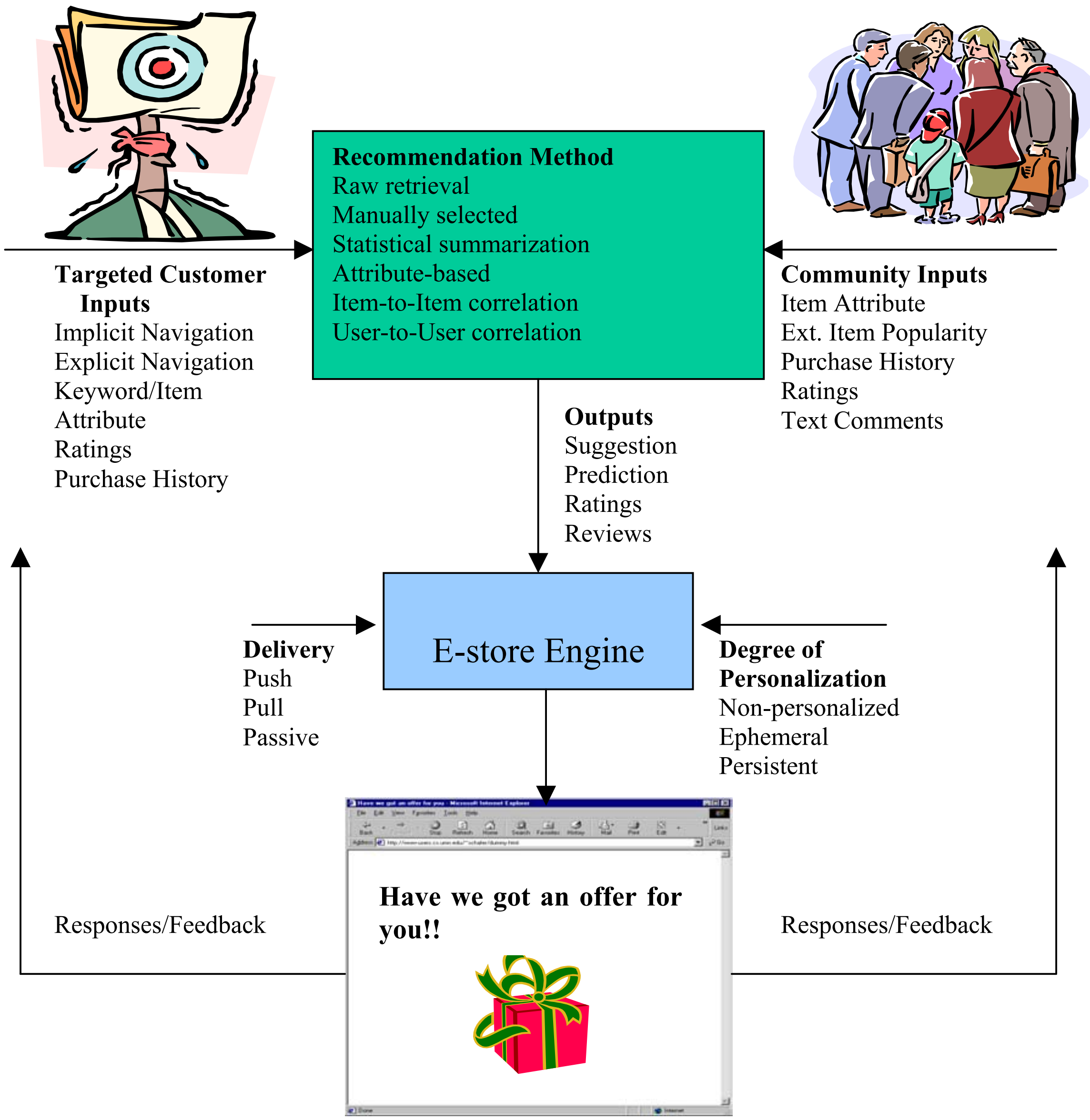


Figure 3.1: The recommendation process in electronic commerce.

Application Model	Functional I/O			Recommendation Method
	Targeted Consumer Input	Community Input	Output	
Broad Rec. List				
Amazon Bookstore Gift Ideas	Explicit navigation	Item attributes Ext. item popularity	Suggestion	Manual selection
Amazon Purchase Circles	Explicit navigation	Purchase history	Suggestion	Statistical summarization
CDNOW Buyer's Guides	Explicit navigation	Item attributes Ext. item popularity	Suggestion Reviews	Manual selection
CDNOW Artist Picks	None	None	Suggestion	Manual selection
CDNOW Top 100	None	Purchase history	Suggestion	Statistical summarization
MovieFinder Top 10	Explicit navigation	Item attributes Ext. item popularity	Suggestion	Manual selection
Feature Search Rec. List				
Drugstore Advisor	Keyword/Attribute	Item attributes	Suggestion Reviews	Attribute-based Manual selection
Comments and Ratings				
Amazon Customer Comments	Implicit navigation	Ratings Text comments	Prediction Ratings Reviews	Statistical summarization
Drugstore Test Drives	Explicit navigation	Ratings Text comments	Prediction Ratings Reviews	Statistical summarization
eBay Feedback Profile	Implicit navigation	Ratings Text comments	Prediction Ratings Reviews	Statistical summarization
Notification Service				
Amazon Eyes	Keyword/Attribute	Item attributes	Suggestion	Attribute-based
Amazon.com Delivers	Keyword/Attribute	Item attributes	Reviews	Manual selection
eBay Personal Shopper	Keyword/Attribute	Item attributes	Suggestion	Attribute-based
Product-Associated				
Amazon Customers who Bought	Implicit navigation	Purchase history	Suggestion	Item-to-Item correlation
CDNOW Album Advisor -Single Item	Explicit navigation	Purchase history	Suggestion	Item-to-Item correlation
CDNOW Album Advisor -Multiple Item	Keyword/Attribute	Purchase history	Suggestion	Item-to-Item correlation
CDNOW Related Artists	Explicit navigation	Item attributes	Suggestion	Manual selection
MovieFinder Users/Our Grade	Implicit navigation	Ratings	Prediction	Statistical summarization
Reel.com Movie Matches	Implicit navigation	Item attributes	Suggestion	Item-to-Item correlation
Deep Personalization				
Amazon Your Recommendations	Purchase History Ratings	Ratings Purchase history	Prediction Suggestion	User-to-User correlation
My CDNOW	Purchase history Ratings	Purchase history	Suggestion	User-to-User correlation

Table 3.1: Taxonomy of recommender applications: Functional I/O and Recommendation Method.

Application Model	Design Issues	
	Degree of Personalization	Delivery
Broad Rec. List		
Amazon Bookstore Gift Ideas	Non-personalized	Pulling unordered list and expert narrative
Amazon Purchase Circles	Non-personalized	Pulling ordered list
CDNOW Buyer's Guides	Non-personalized	Pulling unordered list
CDNOW Artist Picks	Non-personalized	Pulling expert narrative
CDNOW Top 100	Non-personalized	Pulling ordered list
MovieFinder Top 10	Non-personalized	Pulling unordered list
Feature Search Rec. List		
Drugstore Advisor	Ephemeral	Pulling unordered list and expert narrative.
Comments and Ratings		
Amazon Customer Comments	Non-personalized	Passive delivery of comments, individual ratings of other customers, and a predicted rating.
Drugstore Test Drives	Non-personalized	Pulling comments, individual ratings of other customers, and a predicted rating.
eBay Feedback Profile	Non-personalized	Passive delivery of comments, individual ratings of other customers, and a predicted rating.
Notification Service		
Amazon Eyes	Persistent	Pushing single recommendation
Amazon.com Delivers	Persistent	Pushing expert narrative
eBay Personal Shopper	Persistent	Pushing unordered list
Product-Associated		
Amazon Customers who Bought	Ephemeral	Passive delivery of unordered list
CDNOW Album Advisor -Single Item	Ephemeral	Passive delivery of unordered list
CDNOW Album Advisor -Multiple Item	Ephemeral	Pulling unordered list
CDNOW Related Artists	Ephemeral	Pulling unordered list
MovieFinder Users/Our Grade	Ephemeral	Passive delivery of predicted rating
Reel.com Movie Matches	Ephemeral	Passive delivery of unordered list
Deep Personalization		
Amazon Your Recommendations	Persistent	Pulling (un)ordered list
My CDNOW	Persistent	Passive delivery of unordered list

Table 3.2: Taxonomy of recommender applications: Design Issues.

3.2 A Taxonomy

Using the previous examples, we have developed a taxonomy for e-commerce recommender applications that separates their attributes into three categories: the functional I/O, the recommendation method, and other design issues. Recommender applications combine inputs about the consumer in question with those about product and user communities to generate recommendations. Sites then use decisions about personalization level and delivery method to transform these into specific recommendation packages. Feedback to these recommendations may generate additional inputs for future recommendations. Figure 3.1 illustrates this process.

These three categories are not independent; as Tables 3.1 and 3.2 illustrate, certain design choices require specific outputs. Similarly, certain outputs can be produced only by some of the recommendation methods. Furthermore, our taxonomy includes only recommenders that attempt to help individual consumers based on preference data, such as interest information about products or purchase history. We do not consider methods that recommend according to less personal information, such as segmentation, demographic or psychographic or purely business information. Finally, we do not claim that the taxonomy is complete. Rather, it represents the range of e-commerce applications in use at the time of this writing. We fully expect new I/O, methods, and designs to emerge. We do, however, expect the basic structure of the taxonomy to remain useful as new practices are integrated into it.

3.2.1 Functional I/O

To simplify the process, we begin by concerning ourselves only with the data flowing into and out of these systems. Each system takes in a collection of inputs that may include consumer preference data, attribute data, and other correlates. Since this covers a large space of data, we additionally divide these inputs to indicate their origin – inputs about the targeted consumer (i.e., about the consumer for whom we are making recommendations) vs. general inputs regarding the community of other consumers. Recommender applications use these inputs to produce output recommendations for other items. Analysis of these I/O produced the following dimensions.

3.2.1.1 Targeted Consumer Inputs

Inputs about the targeted consumer are fed into the recommendation process to provide personalized recommendations. An application that uses no inputs about the targeted consumer can produce only non-personal recommendations. Adding one or more types of inputs allows the recommender application to personalize recommendations based on the consumer's current activity, the consumer's long-term preferences, or both. While there are multiple ways of categorizing the inputs from the targeted consumer, one compelling set of categories evolves from the consumer’s approach toward providing the input.

While many recommender applications are still global in nature, more are beginning to respond to the consumer’s current state by using the consumer's current navigation to provide context for the production or refinement of recommendations. Consumer behaviors interpreted for this input include both actions the consumer would have performed in exactly the same way even if he were unaware of the recommender system and actions the consumer performs for the sole purpose of enhancing the recommendations. **Implicit navigation** inputs are, generally, inferred from the consumer’s behavior without the consumer's awareness of their use for recommendation processes. This input may include the *specific item* or items that the consumer is currently viewing or those items in the consumer's shopping cart. For example, Amazon.com uses the particular book that a consumer is browsing to recommend a set of additional books considered in some way similar to the currently viewed text. This input may also include the *category or feature* to which the consumer has navigated. In doing so, e-merchants hope these applications will help convince the browser that the initial product is worthwhile – if he likes the “similar” items – thus helping sell multiple products at once.

In contrast, **explicit navigation** inputs are intentionally made by the consumer with the purpose of informing the recommender application of his or her preferences. To offer these, sites provide the consumer with a finite set of attribute choices as navigational links. For example, a consumer using MovieFinder’s Top 10 feature is provided with a hyperlinked list of top ten lists produced by the editors. By navigating to a list of interest, the consumer can get recommendations for products in a fairly specific category. Despite differences in the configuration of these systems, from a consumer's point of view, he is simply navigating.

In some cases, input from the consumer can not be limited to a single category or item of interest. In these cases, applications may use **keywords and item attributes**, either explicitly from a search or implicitly, derived from the items being viewed. In either case, these keywords and attributes are interpreted as input that models the consumer's current interests. For example, consumers using the Advisor at drugstore.com provide information about their wants and needs before receiving recommendations for products such as cold and flu remedies. Systems using these types of inputs replace the feel of navigation with the feel of searching.

The targeted consumer may provide the most helpful and explicit inputs in the form of ratings of items she has consumed. In an ideal situation, consumers are presented with a representative sample of items from the e-merchant's database and are asked to indicate their preference for each of the representative items. This can consist of numerical ratings ("rate each on a scale of 1-5"), or a simple binary rating ("did you like this?"). Consumers who create a personalized My CDNOW are given the opportunity to indicate explicitly the albums that they already own, separating them into the ones they like and the ones they wish they had never purchased. In doing so, the consumer uses a process that feels like neither navigating nor searching. Rather, the feel is that of configuring. The consumer is providing data to the site to allow the business to provide a more personalized experience.

Rather than asking consumers to provide explicit ratings, some sites use the targeted consumer's **purchase history** as an implicit form of ratings. These provide lists of items for which the consumer has expressed a very concrete preference. For example, once a consumer sets up her My CDNOW account, all additional purchases are recorded in the "bought it and liked it category." This input, however, has no real "feel" to the consumer. She is simply using the site. Good implementations of purchase histories recognize that they are related to ratings and allow the consumer to enter them in "ratings mode." For example, users of My CDNOW can review their ratings (including those entered implicitly through purchase) and change the "liked it" to "hated it."

3.2.1.2 Community Inputs

Community inputs include a broad range of data regarding how multiple individuals in the community, or the community as a whole, perceive items. Inputs that reflect overall community opinions include **item attribute** assignments that assign community-based labels and categories to items. For example, many attributes such as film genre and book categories reflect the consensus of the broader society. Similarly, **external item popularity** may reflect popularity in broader communities such as global box-office sales or national bestseller lists. In manually selected recommendation lists such as CDNOW's Buyer's Guides, it is presumed that editors are taking into account more than site sales figures to generate their list of most popular products. Finally, just as we described using the purchase history of an individual consumer as a set of implicit ratings about products, we can use the **community purchase history** to do the same. These can be mined as individual purchase histories to discover similarities and draw conclusions about sales trends or item similarity (Album Advisor) or aggregated to produce site-specific top seller lists (Purchase Circles).

While the previous community inputs are tied to the community as a whole, other inputs are directly associated with individual members of the recommender community. Several sites encourage **text comments** from their users. Systems such as drugstore.com's Test Drives gather comments about a single product and present these as a means to facilitate the decision-making process. While text comments are helpful, they require a fair amount of processing by the targeted consumer. The consumer must read each paragraph and interpret to what degree it is positive or negative. To simplify this process, most sites offering the opportunity for the community to write text comments also encourage the members to indicate some form of numerical score or **ratings**. Just as recommender systems can use the ratings of the targeted consumer, they can also gather the ratings of all consumers to provide data for use in producing recommendations.

Most of the sites in our survey appear to be using largely site-specific data about their consumers, combined with both site-specific and syndicated data about their products. Item attributes are usually syndicated through services that publish digital catalogs comprising categorizations and descriptions of products. For instance, book vendors often use third party genre and keyword classifications. These third party attributes are often supplemented with a smaller amount of site-specific data. External item popularity is nearly always syndicated to provide a broad measure of consumer interest. Community purchase information is always site-specific, based on purchase behaviors of groups of consumers from the site. Community text comments and ratings are primarily site-specific. In principle these data could be shared between sites, but we do not know of examples of such sharing to date.

When syndicated data about products are obtained, it must be unified with data from the site, such as the site's catalog and editor's assessments. Syndicated data about individual consumers can also be purchased and used; this is common with mailing lists and demographic data. In this case, the unification is even more challenging than when using product data because the providers of the data – the consumers –often resist the unification. (For instance, consumers may provide false information to protect their privacy.) Unification of syndicated data has the potential to enhance recommendations for consumers; whether or not it will be common in practice is still an open issue.

3.2.1.3 Outputs

Output recommendations of specific items vary in type, quantity, and look of the information provided to the consumer. The most common type of output can be considered a **suggestion.** This often takes the form of "try this," or simply placing "this" in the web page viewed by the user. The simplest form of "this" is the recommendation of a single item. By recommending only a single item, the e-merchant increases the chance that the consumer will seriously consider the item since the recommendation takes little time to process. However, it also places all of the risk on a single recommendation, which may be rejected because the consumer already owns the item or has other outside knowledge. Targeted advertising also generally results in an individual recommendation as do "check-out coupon" promotions and some other systems designed to elicit up-selling. More commonly, recommender systems provide a set of suggestions for a consumer in a particular context. Some application designers prefer to leave the list unordered to avoid giving the impression that a particular recommendation is the best one. Unordered lists may avoid premature consumer dismissal of an entire set of recommendations based on rejection of the first one. (Of course, every list has some order; many "unordered" lists are deliberately presented in another order, such as alphabetical order, to avoid being misinterpreted as best-first.) Other applications rank recommended items. The structure of an ordered list provides extra information that may be helpful to consumers.

Several recommender algorithms present consumers with a **prediction** of the rating they would give to an item. These estimates can be presented as personalized estimates for individual consumers or as non-personalized estimates for typical community members. These predicted ratings can help consumers understand the strength of a recommendation. Predicted ratings can be displayed in the context of individual recommendations or lists of recommendations, or they can be displayed in the context of general item information. MovieFinder's "Customer Grade/Our Grade" feature provides two different predictions (community and editorial) on an A to F scale that are presented as a user browses the information screen for a movie of her selection.

When communities are small or community members are well known, it may be useful to display the individual **ratings** of community members to allow the targeted consumer to draw her own conclusion about the strength of a recommendation. This technique is particularly valuable when the consumer can select known community members or when the ratings are accompanied by **reviews**. Reviews are an example of recommendations that contain evaluations that are not completely machine-understandable. Indeed, unlike other recommendation techniques, it is difficult to distinguish text comments that recommend for and against a particular item, though, as previously mentioned, many systems that use text comments also ask reviewers to include a numerical rating. Presenting text comments to consumers provides them with an understanding of why a particular item should be favored or disfavored, and comments may be the only way to help a consumer navigate through substantial disagreement among people who have previously agreed. Amazon.com and eBay both help people evaluate items (books and commerce partners) by presenting text comments and ratings in a non-personalized way. That is, each consumer sees the same, complete set of comments. It is possible to select or order comments based on a consumer's history of agreement with the commenter, but we are unaware of any e-commerce applications that do so. By doing so, these systems would be providing commenters with “credentials” – some indication that this person’s comments hold value. The closest step we've seen is a level of "meta-rating" on Amazon.com where readers of comments can rate the comments themselves. Due to the sparsity of consumer comments, there is as yet no mechanism for automatically using those ratings to create personalized sets of comments.

3.2.2 Recommendation Method

In the previous section, we focused on the data used and generated in the recommendation process. In this section, we provide an overview of the specific processes used in actual e-commerce recommender systems. We should point out that individual systems may actually use a combination of these processes. Each category discussed here represents a family of algorithms and approaches. Breese et al. [15] compare a variety of algorithms for recommendation generation, and Herlocker et al. [35] provide a detailed comparison of user-to-user correlation algorithms.

The **raw retrieval** "null recommender" system provides consumers with a search interface through which they can query a database of items. In this case, recommendation is a “binary,” syntactic process whereby the system "recommends" whatever the consumer has requested. While not technically a recommender application, such an application may appear as one to consumers. For example, when a consumer asks a music site for albums by "The Beatles," the system returns a list of Beatles albums that may be helpful and may indeed lead the consumer to an album of which he was previously unaware. Raw retrieval systems are ubiquitous in e-commerce applications.

Applications that value personality over personalization may create sets of recommendations that have been **manually selected** by editors, artists, critics, and other experts. These "human recommenders" identify items based on their own tastes, interests, and objectives and create a list of recommended items available to community members. Their recommendations are often accompanied by text comments that help other consumers evaluate and understand the recommendation. For example, consumers using the MovieFinder Top 10 lists select a particular “genre” for which they would like recommendations – for example, “chick flicks.” They are provided with a list manually compiled by an editor listing what she considers to be the top ten chick flicks of all time. The process does not use computer computation at all but simply reproduces what could appear in a list on the wall of any video store. This process most closely mimics traditional critics and editors, including both potential insight and potential bias. Although not included in our examples, an increasing number of sites allow any community member to establish recommendation lists (e.g. Amazon.com’s Listmania Lists allow users to create and share a list of items sold through Amazon.com with other users of the site).

In cases where personalization is impractical or unnecessary, recommender applications can very efficiently provide **statistical summaries** of community opinion. These summaries include within-community popularity measures (e.g., percentage of people who like or purchase an item) and aggregate or summary ratings (e.g., number of people who recommend an item, average rating for an item). They include systems such as eBay's customer feedback, which provides average ratings of buyers and sellers. Prospective sellers and buyers can consult the average and the individual evaluations but cannot see the rating by "users I've agreed with." While these summaries provide only non-personalized recommendations, they are popular because they are easy to compute, and they can be used in non-customized environments such as physical store displays.

Recommendations based on the syntactic properties of the items and consumer interests in those properties use **attribute-based** recommendation technologies. Though the simplest attribute-based recommendation is raw retrieval, true "recommenders" that use attributes model consumer interests beyond a simple query. For example, a consumer who is browsing in the "country music" section of a music store and who has several "$9.99 special" compact disks in her shopping cart might receive recommendations for discount country CDs. Other attribute-based recommenders use consumer profiles that indicate likes or dislikes to make recommendations to the consumer. For example, the same music store may learn that a particular consumer only buys discounted CDs or that another consumer never buys music from the 1970s.

Other applications use **item-to-item correlation** to identify items frequently found in "association" with items in which a consumer has expressed interest. Association may be based on co-purchase data, preference by common consumers, or other measures. In its simplest implementation, item-to-item correlation can be used to identify "matching items" for a single item, such as other clothing items that are commonly purchased with a pair of pants. More powerful systems match an entire set of items, such as those in a consumer's shopping cart, to identify appropriate items to recommend. Item-to-item correlation recommender applications usually use current purchases or other current interests rather than long-term consumer history, which makes them particularly well-suited for recommending gifts. A consumer merely needs to identify some other items liked by the recipient to elicit gift recommendations tailored to the recipient rather than the giver.

Finally, recommender systems using **user-to-user correlation** recommend products to a consumer based on the correlation between that consumer and other consumers who have purchased products from the e-commerce site. This technology is often called "collaborative filtering" because it originated as an information filtering technique that used group opinions to recommend information items to individuals [37], [43], [64], [75]. My CDNOW is a system that uses user-to-user correlations to identify a community of consumers who tend to own and like the same sets of CDs. The principle is that if several members of my community owned and liked the latest Sting album, then it is highly likely that I will too. Though we use the word correlation in the name of this technique, thus hinting at nearest-neighbor techniques based on linear correlation, the technique can be implemented with many other technologies as well [15].

One important issue when considering the recommendation method is whether the computation can be performed entirely online while the Web store is interacting with the consumer, or whether parts of the computation must be performed offline for performance reasons. Online recommendations are preferred because they can respond immediately to the consumer's preferences. Most of the recommender processes mentioned above can be performed entirely online. Raw retrieval, manual selection, statistical summarization, and attribute-based are all simple computations that are usually performed during consumer interaction. Item-to-item correlation and user-to-user correlation are computationally more intensive and often require an offline component to prepare a model that can be executed efficiently online. One challenge in designing the model-building is to ensure the resulting online system is as responsive as possible to interactive input from the user.

3.2.3 Other Design Issues

3.2.3.1 Degree of Personalization

Recommender applications may produce recommendations at varying degrees of personalization. The degree of personalization encompasses several factors including both the accuracy and the usefulness of recommendations. Accuracy measures how correct the system is while usefulness includes such factors as *serendipity*–whether the system provides valuable but unexpected recommendations–and *individualization*–whether the system provides different recommendations to different people – measures which are both important. An accurate system that only recommends consensus bestsellers provides less value than a system that can find and recommend more obscure books of interest to particular users. Similarly, a system that recommends obscure books, but that is rarely correct, would not be used for long. While personalization is a continuum across several dimensions, we find it useful to identify three common levels specifically.

When recommender applications provide identical recommendations to each consumer, the application is classified as non-personalized. The specific recommendations may be based on manual selection, statistical summarization, or other techniques. Many of the e-commerce recommendation examples are non-personalized. Top-sellers, editor choices, average ratings, and unfiltered consumer comments all present the same recommendations to each user of the system.

Recommenders that use current consumer inputs to customize the recommendation to the consumer’s current interests provide ephemeral personalization. This is a step above non-personalized recommenders because it provides recommendations that are responsive to the consumer's navigation and selection.

Particular implementations may be more or less personal, however. A recommender application with a high degree of ephemeral personalization would be one that uses an entire current browsing session or shopping cart to recommend items. Conversely, a recommender application that simply attaches recommendations to the current item is nearly non-personalized. Ephemeral personalization is usually based on item-to-item correlation, attribute-based recommendation, or both. Examples of ephemeral personalization include CDNOW's multi-item Album Advisor and certain versions of drugstore.com’s Advisors. Both take information provided by the consumer at recommendation time and return a list of suggestions from that ephemeral context.

The most highly-personalized recommender applications use persistent personalization to create recommendations that differ for different consumers, even when they are looking at the same items. These persistent recommenders employ user-to-user correlation, attribute-based recommendation using persistent attribute preferences, or item-to-item correlation based on persistent item preferences. They require consumers to maintain persistent identities but reward them with the greatest level of personal recommendation. Examples of persistent personalization include My CDNOW, which uses user-to-user correlation, and Amazon.com's Eyes and eBay's Personal Shopper, which use persistent attribute recommendation.

3.2.3.2 Delivery

Matching the delivery of recommendations to the consumer's activity is a critical design decision in e-commerce recommender systems, just as it is in traditional marketing. In fact, e-commerce provides close analogues to traditional solicitation and retail models. Marketers have long used direct mail and outbound telemarketing in an attempt to get consumers to initiate a new buying session. Push technologies have the benefit of reaching out to a consumer when the consumer is not currently interacting with the e-merchant. In e-commerce applications, e-mail is the most commonly used push technology for delivering recommendations. Sending recommendations, and perhaps promotional offers, invites the consumer to return to the e-merchant. Indeed, today's technology allows consumers to click on a link in the e-mail message and be taken directly to the recommended product on-line.

Applications using pull technologies allow the consumer to control when recommendations are displayed. They make the consumer aware that recommendations are available (e.g., by displaying a link to them) but do not actively display recommendations until the consumer requests them. This request may appear in different contexts, such as a request to evaluate a specific product, a request to find a gift, or a request for recommendations in a category. Some early applications used pull delivery because recommendation computation was expensive and could slow down the interactivity of a site. Today, pull delivery is a design choice for applications where types of recommendations are considered peripheral (e.g., top-10 lists or gift recommendations) rather than integrated into the application.

Sometimes referred to as "organic" recommendations, passive delivery presents the recommendation in the natural context of the rest of the e-commerce application. Examples of passive recommendation include displaying recommendations for products related to the current product (Amazon.com's Customers Who Bought feature), displaying recommendations for products related to the topic of a text article (CDNOW's Artist Picks), and displaying recommendations in the context of exploration (drugstore.com's Advisors). Passive recommendation has the advantage of reaching the consumer at the time when the consumer is already receptive to the idea. Indeed, e-commerce sites often use passive recommendation as part of the ordering process, suggesting upgraded shipping options, for example, at the time when the consumer is completing a purchase (where it is much more effective than asking about shipping on a link off the home page). A possible disadvantage of passive recommendations is that consumers may not actively notice them, but we are not aware of any research that suggests that noticing recommendations explicitly makes them more effective than having them as part of the overall experience.

We find that the preferred methods of delivery are changing. Early applications focused on push and pull delivery because of performance and the desire to show consumers that because "they care" they are actively recommending. More recently, applications have been shifting to passive and push delivery – passive delivery on their web site with pushed recommendations to bring consumers back.

3.3 Six Recommender Models and Why Sites Use Them

Section 3.2 classifies e-commerce recommendation applications by the functional inputs and outputs to the application, the recommendation method, and other design issues. Figures 3.2 and 3.3 reveal six patterns in e-commerce recommender application designs, each addressing different business goals. This section identifies the six business goals and the application models used to address them. Designers of e-commerce recommender applications can use these models as examples of already-proven solutions to be emulated or as a base from which to explore as-yet-untested recommender application models that may address new business needs.

3.3.1 Helping New or Infrequent Visitors: Broad Recommendation Lists

One of the key challenges for e-commerce sites is to engage visitors – especially new or infrequent visitors – before they leave to visit another site. For sites that list thousands to millions of different products, this challenge is particularly acute; they must not only engage the visitor but also keep him from getting lost and frustrated. Nearly every site visited has some form of broad recommendation list designed to direct consumers towards engaging products. These lists typically allow the targeted consumer to use current navigation to pull non-personalized suggestions. These include overall bestsellers, bestsellers in a category, editor and expert recommendations, and other collections of products selected either manually or through simple statistical summarization. In essence, these recommendation lists replace the tabletop displays, endcaps, and large product displays in physical stores. Whichever technique is used, the lists help orient users who might otherwise leave before finding compelling products.

Part of what makes broad recommendation lists so prominent is the low level of needed input; no personal information is needed, except for minimal ephemeral context about the category of interest to the consumer. Broad recommendation lists allow marketers to adjust pricing and inventory to match the recommendations since they can be assured that these recommendations will reach a large audience. Editors or experts can create text to surround broad recommendations to market the recommended products to consumers. The recommendations themselves can be delivered in several different ways, though most applications either place them on a home page or "category home" page or advertise them and have users directly select them.

3.3.2 Addressing Specific Needs: Feature-search Recommendation Lists

For consumers who are “just browsing” for something that might be of interest, the generalized recommendations on a broad range of concepts from broad recommendation lists are quite adequate. For consumers who need products with specific features, however, feature-search recommendation lists are more appropriate. Feature-search recommendation lists often allow the targeted consumer to provide explicit keyword and/or attribute information about the types of products for which he is searching. This information is used to pull ephemerally personalized lists of products and/or expert narratives concerning the set of products which meet the consumer’s requirements. These lists can be implemented easily in domains with concrete product attributes (such as recommendations for a red, boy’s, short-sleeved shirt) but have also been used in domains with more “fuzzy,” editor-defined attributes (such as recommendations concerning gifts appropriate for the birthday of a girl turning 4 years old who is interested in science).

3.3.3 Credibility Through Community: Customer Comments and Ratings

Retailers in general, and e-retailers in particular, must often overcome an image of low credibility. Consumers may feel that the site is interested only in making a sale, and therefore that it will present any "recommendation" or advertising necessary to induce them to make a purchase. While principles of one-to-one marketing suggest that it is in the retailer's interest to serve the interests of the consumer, stores must still leap over the credibility hurdle to move towards a one-to-one relationship. One way to do this is to collect reviews and ratings from members of the community at large. These systems use the targeted consumer’s current navigation to suggest which non-personalized reviews, ratings, and predictions to display passively. By building a "community center," sites allow consumers to communicate with each other and provide each other with advice and feedback on products.

These "grass roots" recommendations require little site-directed effort since the consumers do all of the evaluation. They also provide a high degree of credibility since consumers often are more likely to believe a set of other consumers than the marketer who makes money on the purchases. As a side benefit, these recommendations create a sense of community that can distinguish the site from others and thereby retain customers. Customer comment applications provide a summary of the ratings, either as an average or another figure representing the rate of positive and negative recommendations, and give the consumer an opportunity to read through the ratings and form her own opinion.

3.3.4 Inviting Consumers Back: Notification Services

Stores that know their consumers' interests can leverage that information by inviting them back to the store when products of interest arrive or are discounted. Notification services use keywords provided by the targeted consumer and attributes of the items being recommended to push persistent, personalized suggestions and can thereby build stronger consumer relationships. Many e-merchants allow consumers to describe the products they find interesting and then automatically notify them when such products are available. These notification services can provide a great service to the consumer, who becomes quickly aware of new products of interest, and can be very effective at bringing consumers back to the e-commerce site on a regular basis. The form of the descriptions can vary from a simple keyword or attribute query to a more complex specification that includes price ranges.

3.3.5 Cross-Selling: Product-Associated Recommendations

Suggestive selling is particularly effective when the seller knows the current interests of the buyer. Retailers arrange products to enhance cross-selling by placing complementary items in close proximity. On-line retailers are freed from physical layout and can directly suggest products related to the one a consumer is viewing. By using the targeted consumer's current navigation as an ephemeral indication of interest, such systems use item-to-item correlation and community purchase history to display suggestions to the targeted consumer passively.

Many different recommender applications use the context of a current product or several current products to recommend other products using a variety of recommender methods. This popularity is partly due to the variety of inputs that can be used to generate such recommendations, including anonymous purchase histories, consumer purchase histories, ratings, product attributes and expert opinions. Product-associated recommendations are particularly well-suited for passive delivery since they can be integrated into a product information page.

3.3.6 Building Long-Term Relationships: Deep Personalization

The goal of most retail businesses is to develop long-term relationships with consumers that lead to higher lifetime values and greater competitive barriers. Deep personalization, based on a consumer's history of preferences, purchases, or navigation, is the strongest and most difficult type of personalization to implement. Deep personalization is common already in web advertising and is becoming more widely used in e-commerce now that collaborative filtering recommendation engines are readily available. Deep personalization uses collaborative filtering's ability to match the targeted consumer's history with histories of other consumers to generate persistent, personalized suggestions or predictions. Deep personalization builds a consumer relationship over time, leveraging the history developed to provide increasingly better recommendations. Unlike notification services that require manual updating, deep personalization updates the user profile whenever the consumer interacts with the merchant. Deep personalization systems can use user-to-user correlation, attribute-based systems with a learning module to identify user interests, or a combination of the two.

3.4 Product Domains and Recommender System Usage

In our attempts to build a taxonomy for recommender systems, we restricted our study to a fairly limited number of sites. This allowed us to identify the models of recommender systems that sites are using. It did not, however, help us understand how prevalent these systems are or how the different models are being applied. The following section discusses the results of a more detailed study of how Internet sites are applying recommender systems.

3.4.1 Hypotheses

Rather than answering all of our questions, the construction of a taxonomy left us with one major question – "Are there trends in the types of Recommendation Systems provided by different types of e-commerce sites?" Upon further consideration, we felt that by studying different domains, the attributes of the products that they sell, and the types of recommender applications they choose, we could address this question and consider the following hypotheses.

Hypothesis 1: Different product domains focus on different recommender application models.

The data known about a product is extremely domain dependent. Furthermore, the gathering of particular types of consumer data is easier in certain domains. For example, consumers looking for books are more likely to perform keyword searches than those looking for banking services. Booksellers can choose to record this data as an implicit indication of interest and apply this data in a recommender system. Electronic banking sites, however, will likely need to consider a different method for gathering recommendation data.

It is expected that the presence of different types of data will drive the type of recommender system that will be most useful to the users of the site. The identification of these trends is extremely beneficial to the developer of a new e-commerce site. An understanding of where some sites have had success while others have failed may mean significant savings in time and money if new sites can avoid the mistakes of their predecessors.

Hypothesis 2: Product attributes affect the need for, and prevalence of, recommender systems.

While Hypothesis 1 may show us the results – different domains use different recommender systems – Hypothesis 2 starts to consider the cause. We wanted to consider what attributes of products might affect the need for a recommender system. In considering product attributes, we identified five to study: number of items at an average site, average cost per item, frequency of new items being added to the product catalog, degree of homogeneity of the items sold at a site, and the degree to which item selection was based on specific and tangible attributes of the product. These five attributes are explained in more detail later in this section. Specific hypotheses concerning these attributes are as follows:

- **Hypothesis 2.1**: As the number of items at a site increases, so will the need for a recommender system.

As the number of products at a site increases it becomes harder and harder to manually separate the "wheat from the chaff." Presumably, consumers who are trying to choose between thousands of potential products have different needs than those trying to decide between hundreds.

- **Hypothesis 2.2**: As the cost per item increases, so does the need for a recommender system.

We expect that consumers are more willing to take a risk on lower priced items. That is, a consumer is probably more willing to gamble on a $7.50 movie ticket than on a $1000 computer. We would expect that recommenders in domains with more expensive products are more likely to take the time to learn about a consumer's specific needs than those in inexpensive domains. In doing so, these systems provide users with the personalized information necessary to help them make their decisions.

- **Hypothesis 2.3:** As the amount of time between catalog updates decreases, the need for a recommender system will increase.

Finding what you are looking for can be difficult. However, if product catalogs remain relatively stable, consumers have more time to weed through the items being sold manually and determine what they will purchase. Conversely, domains with frequent catalog changes present a different view to consumers every time they visit the site. Because of this, they are more likely to need a recommender system to help them find items of interest.

- **Hypothesis 2.4**: The more homogeneous the products within a domain, the more likely it is to need a recommender system.

A user often makes his final purchase decision by eliminating those products that don't meet his needs. In some situations, the number of remaining choices is so few that the decision is relatively easy. In domains where products are very homogenous, the user is more likely to need a recommender system to help separate seemingly similar products.

- **Hypothesis 2.5**: The more that products are selected based on specific attributes, the less the need for a recommender system.

When specific product attributes are the basis for product selection, consumers are less likely to need recommender systems and more likely to need simple information retrieval systems that search product descriptions. When products are selected on "intangibles," however, the search for attributes in product descriptions is less helpful.

Hypothesis 3: There are correlations between product attributes and the recommender application models that are used to recommend among products with those attributes.

Not only do these five product attributes affect the need for a recommender system, but we also propose they will affect the type of recommender system that can solve the problem. For example, feature-search recommenders are more likely in domains where products are based on very specific, tangible attributes. Since feature-search recommenders require specific features as part of the search process, they will be particularly ineffective in domains where products are selected on intangible attributes. Although we expect these correlations to exist, specific sub-hypotheses were not proposed due to the number of required combinations.

3.4.2 Experimental Design

For this analysis, we selected ten product domains covering a variety of products, price ranges, and purposes (Table 3.3). Within these domains, we wanted to study established companies – those legitimate enough, and in existence long enough, to have enough resources available to *consider* recommender systems as part of their marketing strategy. We wanted to guarantee, however, that our site selection was not limited to the heavy hitters or those known to have recommender systems. After examining several of the "reputation" sites on the Internet (Power Rankings at Forrester Research, Bizrate, etc.), we chose to use the site lists at Gomez.com™ [97]. Primarily, we chose Gomez because the number of sites listed within a given product domain was appropriate (~15-

25). Where the site lists were too large (>30), we randomly sampled to ~15 sites. The result was an analysis of approximately 150 unique sites, implementing nearly 450 recommender systems. Analysis of a given site was restricted to the domain lists on which it appeared, and recommender systems were treated differently within each section. For example, while Barnes and Noble sells items from a variety of domains including books, software, and music, Gomez included Barnes and Noble on its book and music lists but left it off its software list. Analysis of the Barnes and Noble site was restricted to its two referenced domains, and the recommender systems within each were treated as separate systems.

For each of the sites selected, we inspected the site looking for visitor-identifiable recommender systems. This study consisted of exploring the sites but not purchasing from them. It is possible that there are additional recommender systems in these sites that do not become active until a visitor becomes a repeat consumer of the site. This study made no attempt to classify such systems. Furthermore, this study did not attempt to identify "hidden" recommendations. Consider a user logging on to a movie site. Featured movies on the front page may be generically selected for all users or may be selected specifically for that user based on her previous purchase history. Certainly, the latter would be considered a recommender system. However, detecting these sorts of "hidden" recommenders is difficult.

	Sites in original rankings	**Sites used in study**[6]
Apparel	23	23
Auction	11	10
Banking	60	15
Books	15	15
Cars	14	14
Computer	26	25
Electronics	13	13
Movies	22	21
Music	19	19
Toys	16	15

Table 3.3: Domain and site distribution.

[6] May be less than the number of sites in the original rankings due to site closures and/or mergers, or due to a reduction in sample size.

For each of the recommender systems at each of the sites studied, we identified its "values" for each of the six dimensions within our taxonomy as well as the application model being used. For each of the domains in our study, we identified its classification for each of five product/site attributes: number of unique items sold, average cost per item sold, frequency with which new items are added to the product catalog, homogeneity, and the degree to which product selection is based on specific attributes. When sites within a domain had a large difference, we attempted to find a reasonable common ground. A summarization of how domains were classified is in Table 3.4.

Number of products	
1K	Apparel, Banking, Cars
10K	Computers, Electronics, Toys
50K	Auction, Movies, Music
100K	Books
Average Cost[7]	
<$20	Books, Movies, Music
$20-50	Toys
$50-100	Apparel
$100-1000	Auctions, Computers, Electronics
$1000+	Cars
Frequency Of New Items	
Daily	Auctions
Weekly	Books, Movies, Music
Seasonal	Toys, Apparel
Short Cycle	Computers, Electronics
Long Cycle	Cars
Homogeneity	
Very Similar	Music, Cars, Movies
Similar	Books, Toys
Unsimilar	Electronics, Computers, Apparel
Very Unsimilar	Auctions
Tangibility	
Intangible	Auctions, Books, Movies, Music, Toys
Mostly Intangible	Apparel
Mostly Specific	Computers, Electronics
Specific Needs	Banking, Cars

Table 3.4: Product attribute classification of domains.

The **number of unique items** sold at a site within a given domains was based on the average number of products sold at *typical* sites within the domain (i.e., while eBay

[7] Due to the lack of "price," banking has not been included in the attribute analysis for average cost or homogeneity.

may have millions of products in their auctions, the typical auction site has in the tens of thousands). Values were estimated through catalog examination and self-reported numbers from sites.

The **average cost per item** sold within a domain was based on catalog sampling. Two sites were selected at random from each of the domains, and a sample of ten items was taken. Where appropriate this sample consisted of site-reported "bestsellers." Where that was not possible, or where that was inappropriate, a semi-random sampling was taken from the "featured products" lists at the site. Particular care was taken to make certain this was a representative sample. For example, many of the book sites report bestsellers only on their hard cover books. Using this as the sole sample would skew the average-item price higher than actuality. Similarly, many of the "featured item" lists either present items from one theme ("The best in laptop computers") or seem to push big ticket items. Whenever two sites produced substantially different averages (Amazon.com, selling mostly fiction, produced a much lower average price than fatbrain.com, selling mostly technical manuals), a third site was included.

The classification that a domain received for the **frequency with which new items are "released"** was based on observations regarding the frequency with which sites update their product catalogs. For example, new models of cars are released annually, and sites perform a minimum of catalog maintenance. Conversely, items may be added at auction sites 24 hours a day, and catalog maintenance is a continuous process.

Homogeneity of products is a relatively qualitative and subjective attribute. We understand that there are few attributes we can use to separate different CDs from each other, while there are quite a few to separate different items at auction sites. Unfortunately, quantifying this is difficult. In an effort to use an objective measure of homogeneity, we chose to classify products based on the standard deviation of the average item price. Thus, music is classified as very homogenous since most CDs sell for a similar price. On the other end of the extreme, products at auction sites are extremely non-homogenous since product prices range wildly from under a dollar to thousands of dollars. This method of determining homogeneity was selected over other methods due to the accessibility of the data.

The **tangibility of a domain**, or the degree to which products within a domain are selected based on specific attributes, was based on human analysis. Domains were classified based on the extent to which a consumer could identify his requirements when purchasing an item within the domain. For example, when a user needs to purchase a new car, she can state her needs in relatively specific terms – a mini-van with an automatic transmission and dual airbags that is intended more for town driving than highway driving. When she feels like seeing a movie, her "needs" may be intangible. She may settle for using a description as simple as, "something along the lines of a romantic comedy."

In planning this study, we had to consider the effect that data detail would have on study time. Based on the resources available during the completion of this thesis it was determined that the more general results available from a domain-based analysis were sufficient. Unfortunately, in collapsing the examined sites in a given domain into one common data point, we have compressed data and may have masked more detailed results. It is worth acknowledging that the alternative – gathering attribute values on a site-by-site level – may provide even more conclusive results that product attributes have an effect on the recommendation models sites implement. However, this study, as presented, provides a solid base for just such a study.

3.4.3 Results

Hypothesis 1 stated "**Different product domains focus on different recommender application models**." The general analysis of recommender system usage within the ten domains is summarized in Table 3.5. Several trends are observed.

- With the exception of banking sites, the majority of sites, regardless of domain, have incorporated at least one recommender system into their site.
- Most domains have an application model that is the clear favorite. Six of the ten domains have a "most common" model that dominates the next most common model by at least 25%.

- Most domains have one or more application models that are consistently not used. All 10 domains have at least one application model that no site has chosen to implement. Even if we ignore the rarest of the application models (deep personalization), nine of the 10 domains have at least one remaining model with not a single implementation.

	Sites used in study	Sites w/ 1+ RS	Broad Recs.	Feature Search	Customer Comments	Notification	Product Assoc.	Deep Person.
Apparel	23	78%	61%	22%	0%	9%	22%	4%
Auction	10	90%	90%	0%	90%	50%	10%	0%
Banking	15	7%	7%	0%	0%	0%	0%	0%
Books	15	100%	100%	0%	40%	40%	40%	7%
Cars	14	86%	21%	71%	21%	7%	0%	0%
Computer	25	84%	68%	16%	12%	12%	56%	0%
Electronics	13	100%	77%	15%	31%	0%	85%	0%
Movies	21	95%	81%	0%	43%	10%	48%	14%
Music	19	95%	95%	0%	53%	5%	63%	16%
Toys	15	93%	93%	33%	40%	0%	27%	7%
Total	170	84%	69%	15%	29%	12%	37%	5%

Table 3.5: Application model usage by domain.

These results each suggest that our hypothesis was correct. That is, different product domains focus on different recommender system application models.

Hypothesis 2 stated "**Product attributes affect the need for, and prevalence of, recommender systems.**" We proposed that *overall* recommender system usage should be affected by the five product attributes we selected. A detailed analysis of product attributes and recommender system usage is presented in Table 3.6.

As the number of products a consumer must choose from increases, so does the overall use of recommender systems. Prior logic suggested that for domains with few items (100s – 1000s), consumers were able to feel like they could build a realistic picture of the product catalog and identify the products suitable for their purposes. Therefore, the need for a recommender system is reduced. However, as the number of products increases (tens or hundreds of thousands), this becomes harder and harder to do manually, and even a simple recommender system becomes increasingly beneficial.

It is difficult to identify even a threshold effect in any of the remaining attributes. The easiest explanation is that there simply is no correlation between these attributes and their impact on the need for some form of recommender system. A second explanation is that while these attributes do have an affect on recommender system usage, their effects are based on combinations of attributes that our simple linear examination cannot detect. Finally, a third explanation is that considering these attributes over all models simultaneously does them a disservice. Perhaps, as Hypothesis 3 suggested, there are correlations between these product attributes and individual application models.

	Total	Sites w/ 1+ RS	Broad Recs.	Feature Search	Customer Comments	Notification	Product Assoc.	Deep Personal.
# of products								
1K	52	62%	35%	29%	6%	6%	10%	2%
10K	53	91%	77%	21%	25%	6%	55%	2%
50K	50	94%	88%	0%	56%	16%	46%	12%
100K *	15	100%	100%	0%	40%	40%	40%	7%
Avg Cost								
<$25	55	96%	91%	0%	45%	16%	51%	13%
$25-50	15	93%	93%	33%	40%	0%	27%	7%
$50-100 *	23	78%	61%	22%	0%	9%	22%	4%
$100-1000	48	90%	75%	13%	33%	17%	54%	0%
$1000+ *	14	86%	21%	71%	21%	7%	0%	0%
Freq. Of New								
Daily *	10	90%	90%	0%	90%	50%	10%	0%
Weekly	55	96%	91%	0%	45%	16%	51%	13%
Seasonal	38	84%	74%	26%	16%	5%	24%	5%
Short Cycle	53	68%	53%	11%	13%	6%	47%	0%
Long Cycle *	14	86%	21%	71%	21%	7%	0%	0%
Homogeneity								
Very Similar	54	93%	70%	19%	41%	7%	41%	11%
Similar	30	97%	97%	17%	40%	20%	33%	7%
Unsimilar	61	85%	67%	18%	11%	8%	49%	2%
Very Unsimilar*	10	90%	90%	0%	90%	50%	10%	0%
Tangibility								
Intangible	80	95%	91%	6%	50%	18%	41%	10%
Mostly Intangible *	23	78%	61%	22%	0%	9%	22%	4%
Mostly Specific	38	89%	71%	16%	18%	8%	66%	0%
Specific Needs	29	48%	14%	34%	10%	3%	0%	0%

Table 3.6: Application model usage by product attributes.
* Indicates a category containing only one domain

Hypothesis 3 stated "**There are correlations between product attributes and the recommender application models that are used to recommend among products with those attributes**." The following considers the correlations present with each application model.

As we previously mentioned, **broad recommendation lists** are the easiest to implement of all the application models. This is evident when examining the results in Table 3.5. Broad recommendation lists are the most common form of recommender system in eight of the ten domains and are the second most common in the remaining two.

Broad recommendation lists have strong correlations with the number of products and the frequency with which products are added to the system. As the number of products available and frequency with which they become available increases, so does the trend for sites to implement this application model. It is not surprising that these two attributes coincide. The more frequently new products are released, the more likely it is that a product catalog contains large quantities of them. Similarly, as hypothesized earlier, it is not surprising that as more products are available, sites choose to implement a simple recommender system to help their consumers find products to purchase.

Broad recommendation lists have a mild correlation with the average cost of a product. As cost increases, the use of broad recommender lists decreases. This contradicts our earlier hypothesis that as cost increases, so does the need for recommender systems. Perhaps this hypothesis should have stated that as cost increases, so does the need for *good* recommender systems. The recommendations provided by broad recommendation lists are simply too general and do not provide consumers with useful information. Thus, sites switch their energies to models that are more helpful.

Finally, broad recommendation lists have a mild correlation with the intangibility of the attributes used in selecting a product. Broad recommendation lists tend to be accessed through automatic, passive display or through following a single link. As such, it is easier for a user to navigate quickly to his desired movie (romantic comedy) than it is for him to navigate to his desired car.

Where broad recommendation lists attempt to make generalized recommendations on broad range of concepts, **feature-search recommendation lists** attempt to make very specific recommendations on very explicit concepts. In fact, feature search recommendation lists are frequently used to make recommendations to consumers looking for the previously mentioned mini-van. In essence, feature-search recommendation lists are a "converse" to broad recommendation lists. This concept becomes apparent when we examine the correlations between product attributes and the feature-search model.

Feature-search recommendation lists have a strong correlation with number of products and the frequency with which products are added into the catalog. As the number of products available and frequency with which they become available increases, sites decrease their implementation of this application model. As the number of products increases, not only does the amount of data increase, but the amount of data required to *differentiate* between the products increases as well. Consequently, as product numbers increase, data requirements increase. Similarly, the more frequently new items are being added to the system, the more human effort is involved in making sure that the data is entered properly. Thus, as product numbers and frequency increases, the overhead for maintaining a feature-search recommender becomes too much for sites to maintain efficiently.

Feature-search recommendation lists have mild correlations with average cost, homogeneity, and tangibility. As cost increases, so does the use of feature-search recommenders. Perhaps as price increases, the need to be certain of our final choice becomes more important. Feature-search recommendation lists provide the more detailed and precise recommendations required in this case. For product homogeneity there is a threshold effect that states that as long as the products are somewhat homogenous, feature-search recommendation lists are helpful. This is logical considering that it is easier for consumers to distinguish manually between non-homogenous items. Thus, there is a lower need for recommender systems in domains with non-homogenous products. For the exact same reason that broad recommendation lists were used where decisions are based on less tangible attributes it is logical that feature-search recommendations are more beneficial where decisions are based on specific attributes.

Correlations between attributes and the **customer comments and ratings** model are much weaker than correlations in the previous models. We see a distinct threshold effect between the number of products in a system and the presence of this model. This agrees with our prior logic in which we hypothesized that when there are too few products there simply isn't the need for recommender systems. Similarly, there is a mild threshold effect between frequency of new products and this model. It is likely that this results from the correlation between frequency and number of products discussed previously.

There is a minor negative correlation between the cost of products in a domain and the implementation of a customer comments application. That is, as price increases, the presence of this model decreases. Perhaps the readers of reviews are looking for some form of comparison between the reviewed product and other similar products. With more costly product domains, review writers are less likely to have knowledge of multiple products. Since reviews in more costly domains are lacking quality comparison data in either explicit or implicit format, users find the reviews less helpful, and such features fail.

Finally, there is a correlation between the extent to which decisions are based on less tangible attributes and the presence of a comments model. When writing prose-based comments, fellow consumers are much better able to capture these intangible attributes than in other systems. The one exception to this pattern is the domain of apparel in which no site has implemented a customer comments application. While this may indicate a missed opportunity, it is more probable that sellers in this domain have specifically rejected this model. It is likely that positive and negative comments in the domain of books are equally offsetting. However, it may take many positive comments to negate a single negative comment in the apparel industry. Furthermore, the apparel industry has long worked under the principle that designers should tell consumers what they should want to buy.

In Section 3.3.4 we claimed that sites use **notification services** to keep consumers coming back. They do this by inviting them to the store when new products of interest arrive. Thus, it is not surprising to observe that domains with a high frequency of product releases are more likely to use notification services. All sites on the Internet have to deal with the lack of a physical presence. Presumably, consumers are less likely to be "driving by" and "stopping in to see what is new." Sites selling products from domains with frequent additions of new items are more likely to offer consumers something that wasn't available during their last visit. Sites need to get consumers in the store, however, in order to entice them with the new products. Notification services give the sites an outlet for reminding consumers of the site's existence and a reason for returning.

In addition to a correlation with frequency of product releases, notification services have a positive correlation with number of overall products at the site. This follows our standard reasoning that the more products sold at a site, the more important it is to offer a feature which allows users to identify starting points for consideration of these products. Certainly, notification service provides such a starting point.

Recommender applications based on the model of **product-associated recommendations** are the applications showing the weakest correlations with any of the studied product attributes. Other than a *minor* threshold effect between these applications and the commonly linked "number of products at a site" and "frequency of new products," there are no correlations to discuss.

One explanation of this "phenomenon" is to consider the underlying technology that facilitates product-associated recommendations. These recommendations are normally based on association rules discovered during data mining. As we have discussed in prior chapters, data mining is becoming an increasingly easier process for e-commerce sites that take the time to gather and organize their data in a logical manner. Thus, it becomes relatively inexpensive to implement a recommendation application using product-associated recommendations. The lack of correlation data may suggest that sites choose to implement this model because they can, rather than because it meets the needs of their consumers.

Unfortunately, applications of **deep personalization** are the least implemented of the six models. Due to the relatively low numbers observed in this study, the validity of any correlations between product attributes and domains implementing deep personalization are too suspect to consider in this thesis.

3.5 Summary

We present several important results in this chapter. First, we discuss the development of a taxonomy for recommender applications in e-commerce. This taxonomy should be useful to two groups: academics studying recommender systems and implementers considering applying recommender systems in their site. For academics, the examples and taxonomies provide a useful initial framework within which their research can be placed. The framework will undoubtedly be expanded to include future applications of recommender systems. For implementers, the chapter provides a means of making choices among the available applications and technologies. An implementer can choose a moneymaking goal, select the interfaces that will help achieve that goal, and pick an implementation technique that supports the goal within the interface.

Second, we report on a study which finds correlations between the attributes of the products sold by sites and the application models used to recommend the products. While only partially validating our original hypotheses, this study produced fascinating results. The validation of Hypothesis 1 – different product domains focus on different recommender application models – suggests that there is more to implementing a recommender system than throwing one up on a site. These systems have requirements that a site must be able to meet and produce recommendations that must meet the end-user's needs. The validation of Hypothesis 3 – there are correlations between product attributes and the recommender application models that are used to recommend among products with those attributes – provides even more support for this belief. By beginning to identify which application models are used with which product attributes, we begin to consider how and why these application models actually work. Hypothesis 2.1 was the only sub-hypothesis of Hypothesis 2 that was validated. As discussed, this suggests that the interaction between product attributes and recommender system usage is more complex than we originally considered. A simple linear analysis of attributes vs. usage seems insufficient to yield meaningful results. The analysis does, however, lay a foundation for a more careful and complex analysis.

Chapter 4: Recommendation Design for Meta-recommenders

Previous chapters have discussed the technologies used in recommender applications as well as implementations from both the research community and electronic commerce. In Chapters 1 and 2, we introduced the concept of meta-recommender systems. However, an examination of the taxonomy created in Chapter 3 suggests that the business community has yet to implement a complete meta-recommender. This chapter is the first of several that discuss the design and implementation of meta-recommender systems.

In this chapter, we re-introduce the concept of meta-recommender systems by presenting the results of two controlled user studies which consider interface design. This chapter is organized as follows. First, we consider several issues integrally related to the development of meta-recommender systems. Second, we introduce the MetaLens Recommendation Framework. Third, we discuss MetaLens, the first of several meta-recommenders built within this framework. Fourth, we present the results of two studies on the design of a recommendation format for MetaLens.

4.1 Related Work

4.1.1 Customizable Portals

Portals are online gateways to the Internet. Typically, a portal consists of the most popular features of the Internet (a catalog of web sites, a search engine, or both) combined with email and/or news services. Portals are frequently intended to be the "point of entry" to the web for their users. Examples of some of the more popular portals are sites such as Yahoo, Excite, and AltaVista, as well as the "browser software" designed by service providers such as AOL.

Increasingly, these sites are providing their users with the ability to customize the information and layout of the portal. One example of this is "My Excite," [88] which allows a user to configure a front page to contain everything from scores of his favorite sports teams and the weather from cities of interest, to prices for his favorite stocks and news in one or more of several news categories (Figure 4.1). By providing users with customization features, the portals make themselves more useful to their users, making the users more likely to select the site as their front page, which increases page views, which, in turn, increases the site's potential for earning income.

In the development of meta-recommenders, valuable lessons can be learned by observing the design and implementation of these customizable portals. Portal sites quickly discovered that no one layout or set of information met the needs of all their users. By providing users with customization features, the portals provide a mechanism through which a user can configure the site to be more useful for him. One way to make a meta-recommender more helpful is for designers to take similar measures and provide customization features. Thus, a single recommender is more likely to meet the needs of a variety of users.

Figure 4.1: Customizable portal: My Excite.

4.1.2 Product Comparison Sites

What happens when a consumer only knows “I want a digital camera with a minimum of 1 MegaPixel of resolution and a 2x optical zoom"? One way for the consumer to get help is through the use of “product comparison sites.” These sites allow consumers to identify a domain of purchase (i.e., a digital camera, computer, or automobile) and narrow the list of products within that domain by indicating the features in which they would be interested, and the importance of these features in their final decision. Sites perform queries over the attributes of known products in the category and return ranked lists of “recommended” products.

Frictionless [94] often allows users to start their search by selecting from several predefined profiles. For example, a consumer interested in purchasing a new notebook computer can decide if his profile is closer to that of a “telecommuter, road warrior, or budget shopper” in addition to a truly customized profile. The consumer’s selection provides an initial set of features and weights which he can then modify as desired (for example, indicating that the laptop must have between 64 and 256 MB of RAM and should come with either a touch pad or a trackball). In addition to maintaining their own site, the Frictionless system is the engine that powers comparison shoppers at Lycoshop, Brodia, computer.com, and Wingspan.

Active Buyer’s Guide [88] not only allows users to enter preferences for the products which they are interested in purchasing but also provides what the creators refer to as “Adaptive Recommendation Technology.” This asks consumers to make a decision based on a set of “tradeoffs.” For example, when examining baby strollers, a consumer might be asked to rank his preference between a stroller with a “removable canopy and no front bar” or one with a “fixed canopy and a removable front bar.” In addition to maintaining their own site, Active Buyer’s Guide is the engine that powers comparison shoppers at Infoseek/GoNetwork, MySimon and DealTime.

The power of comparison sites becomes apparent when examining data regarding how consumers arrive at a given e-merchant. According to an Active Research study during the 1999 Christmas season, comparison sites were second only to portals as “traffic drivers” to e-merchants. For every consumer who entered a site directly, three entered the site through some form of comparison engine [38]. Since comparison sites are such popular and powerful features of the Internet, designers of recommender systems should consider what these systems provide users and whether or not current recommender systems provide similar functionality.

4.1.3 Data Fusion

"Data fusion" is a somewhat ambiguous term representing a variety of activities. However, it can be reasonably defined as "a formal framework … for the alliance of data originating from different sources. It aims at obtaining information of greater quality; the exact definition of 'greater quality' will depend upon the application." [85]. Data fusion is conducted in a variety of application fields including the connection of earth observation data from multiple sensors [92], media research [82], information management [4], and, most closely related to this thesis, the combination of search engine results in digital libraries [84].

For example, van der Putten [82] discusses ways in which data miners can generate more data to mine. He suggests that often in database marketing "elementary customer information resides in customer databases, but market survey data is only available for a subset or even a different sample of customers." He demonstrates that a nearest neighbor algorithm can connect these separate databases into a single, cohesive data set for use by data mining practitioners. This can be as simple as recognizing that User X in database1 and User Y in database2 are the same person or as complex as recognizing that database2 does not contain data for User X, but the data for User Z is an acceptable substitute. In the development of meta-recommenders, data fusion will become a particularly important step of the process. As the amount of recommendation data used by the meta-recommender system increases, it becomes more challenging to connect the data for each recommended item.

4.2 The MetaLens Recommendation Framework

Consider the following scenario. A user of MovieLens wants to take her 8-year old nephew to the movies. While she wants the movie to be something she might enjoy, she also has additional requirements. For example, she would prefer a comedy or family movie rated no “higher” than PG-13, containing no sex, violence or bad language, lasting less than two hours and, if possible, showing at a theater in her neighborhood. While MovieLens should be good at providing her with lists of movies she will like or giving her a personalized prediction for a specific movie, it will do so based on her long-standing collaborative filtering-based profile. That is, it may be biased towards the British art films or independent thrillers she frequently likes. MovieLens fails to provide her with an interface for expressing her ephemeral requirements. Because of this, she will need to consult several sources such as MovieLens, IMDB, Yahoo movies, and the theater listings in her local newspaper to gather enough information to make her choice.

The remainder of this thesis is built around the creation of meta-recommenders that help users faced with exactly this type of problem. More specifically, this thesis focuses on recommender systems constructed with the MetaLens Recommendation Framework (MLRF). This framework serves as a structure within which multiple meta-recommenders can be constructed. It does so through a three-layer process.

The **Data Layer** of the MetaLens Recommendation Framework is where data used in the recommendation process is acquired, fused, and stored for use by the computation layer. We have defined a meta-recommender as a system that uses “…a combination of rich recommendation data using multiple data sources.” In order to have access to multiple data sources, the data layer works with a series of data acquisition modules. Each gathers all or a portion of the recommendation data from a single data source. Additionally, data acquisition modules may need to perform some level of data fusion to ensure that all data is usable by the computation module. Data acquisition modules in MLRF take one of two formats: those that gather cached data, and those that gather runtime data.

Modules which collect cached data are generally used to gather non-personalized and relatively static data. New data items are searched for, and existing data items are updated on a set schedule. This data is then cached to provide more efficient data access. This is appropriate when users are accessing the same data set, and modifications to the data are relatively predictable.

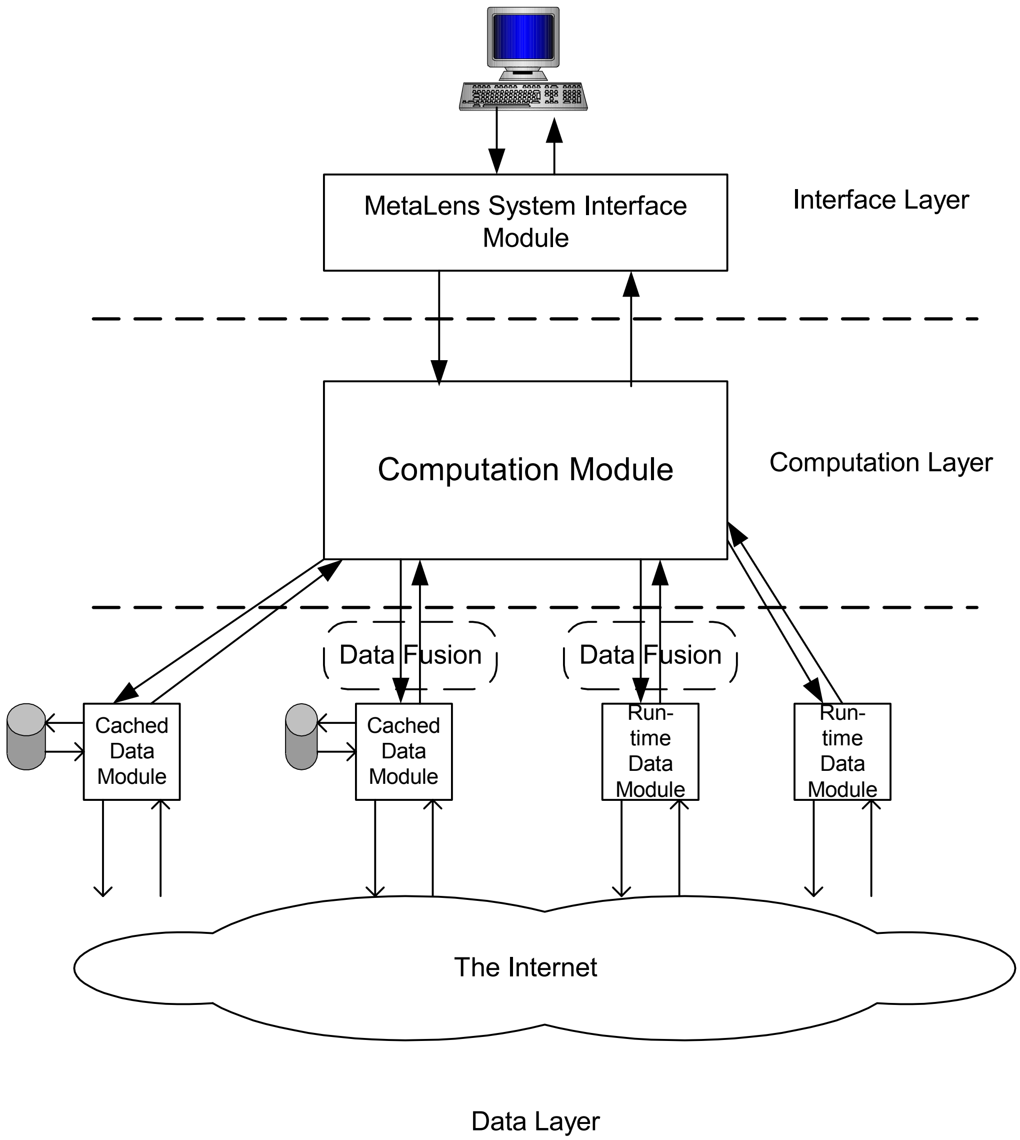


Figure 4.2: The MetaLens Recommendation Framework.

Modules which collect data at runtime are used to gather data which is personalized for the user or data which change frequently. When data is personalized for each user, the overhead is too large to warrant the caching of this data in advance. When data changes frequently, caching becomes impractical because the system would spend an inordinate portion of its resources updating data that may not even be accessed prior to the next update. Under either of these situations, it costs less in the long run to gather specific, up-to-date data *each* time a recommendation request is made.

The **Computation Layer** of the MetaLens Recommendation Framework is where the recommendation content from the data layer and the user's requirements from the interface layer are combined. The process produces recommendations consisting of an ordered list of recommended "items" and each item's corresponding score. These recommendations are returned to the interface layer for presentation to the user.

The **Interface Layer** of the MetaLens Recommendation Framework serves as the connection between the user of the recommender system and the computation layer. The visible portion of the interface layer consists of the user interface through which users can request and view recommendations. However, the more functional portion of the interface layer is hidden from the end user. This portion validates and translates the ephemeral and persistent user requirements, communicates this information to the computation module, and formats the returned recommendation list to provide useful recommendations to the user.

4.3 A First Meta-recommender: The MetaLens System

Although the MetaLens Recommendation Framework should be relevant in a variety of domains, it was initially used to implement a meta-recommender for the domain of movies. The MetaLens system was built to aid in scenarios like the one described in Section 4.2. It was designed within MLRF as a proof of concept for the recommendation framework and is used extensively in the remainder of this thesis. The following section explains the layers in MLRF in more detail by explaining how these layers are used in the development of MetaLens. Figure 4.3 provides a more detailed view of the layers as implemented by MetaLens. Figure 4.4 provides detail on the Yahoo modules described below.

Much like the user in our scenario makes her final choice by examining several movie data sources, MetaLens considers recommendation data from several Internet film sites to produce a single, merged, list of recommendations. Yahoo Movies serves as the primary data source for MetaLens, providing information concerning movies, theaters, and show times. The data about a particular movie is relatively static; for instance, a movie's MPAA rating seldom changes. Furthermore, new movies are regularly released on Fridays. Because of this, the majority of the recommendation data used by MetaLens is gathered is gathered offline by five cached data modules which collect information on a regular schedule to account for newly added entities and allow for potential changes in known entities.

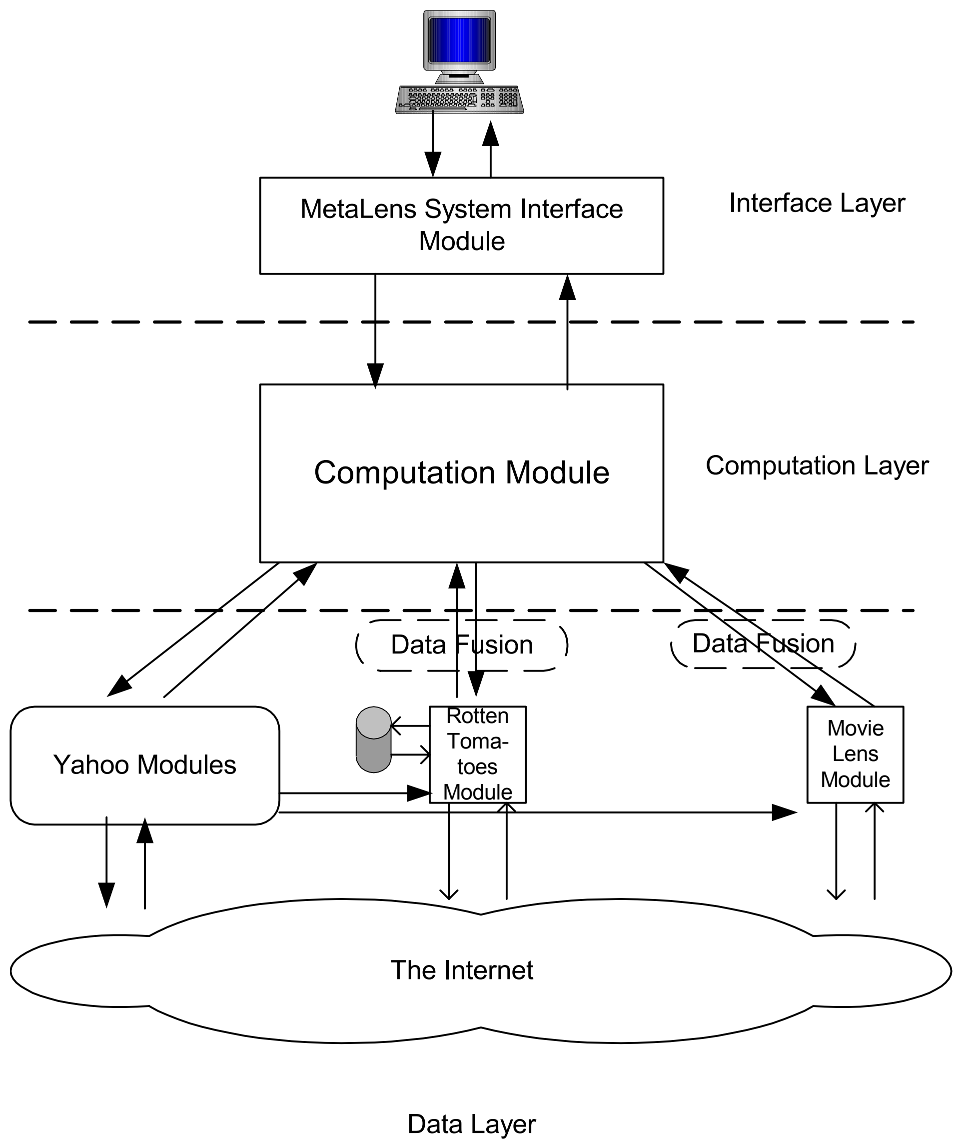


Figure 4.3: The MetaLens Recommendation Framework as applied to the MetaLens system.

- The Yahoo ZIP Code module runs on an "as needed" basis. The module, similar to each of the modules described in this section, consists of a script in perl which constructs an URL representing where information about theaters in each of the ~35,000 known ZIP Codes in the United States is located on the Yahoo site. These URLs are passed one at a time to a sub-process which creates an HTTP connection with the Yahoo web site. The HTML returned by the connection is parsed, and the data is extracted based on observed patterns in the construction of the HTML[8]. In this case, the data consists of information concerning the theaters considered by Yahoo to be within driving distance. Upon completion, the module generates a list of all known theaters (~5000).

- The Yahoo theater module is also run on an "as needed" basis. It uses the theater information found by the ZIP Code module to gather the needed information for each theater. This includes location, contact information, and special accommodations offered by the theater.
- The Yahoo show time module gathers the movies and show times for each theater in the country on a weekly basis. Show times are available from Yahoo in two formats: per theater and per ZIP Code. Rather than searching each of the ~5000 theaters individually, a minimum spanning set of ZIP Codes consisting of ~725 ZIP Codes is calculated after each cache update of the Theater module. This set consists of the minimum number of ZIP Codes necessary to gather all of the show time data used by MLRF.
- The Yahoo movie module uses the movie list generated by the show time module to gather specific movie information about each movie showing in the country during the upcoming week (typically 225 plus or minus 25). This information includes genre, MPAA rating, people involved with the film, a synopsis, and other common movie features.
- The Rotten Tomatoes module uses the movie list generated by the Show time module to gather specific critical review information about each movie. This information includes the number of critics rating the movie, the number of critics favorably reviewing the movie, the number of top critics reviewing the movie, and the number of top critics favorably reviewing the movie.

[8] Because of this, these modules are sensitive to modifications in the web site. Numerous times during the experimentation for this thesis, MLRF was rendered useless by a site upgrade at one of the sites used by the framework.

In addition to the cached-data modules, MLRF uses a single, runtime module. The MovieLens module gathers personalized prediction information from MovieLens. Because a prediction for a given movie is personalized for each user, and because this prediction can change at any moment (based on the input of additional movie ratings from either the user in question or other users like him), it is not appropriate to cache the data provided by MovieLens. Instead, the most up-to-date predictions for a user are gathered *each* time the computation module wants to create a recommendation list for that user.

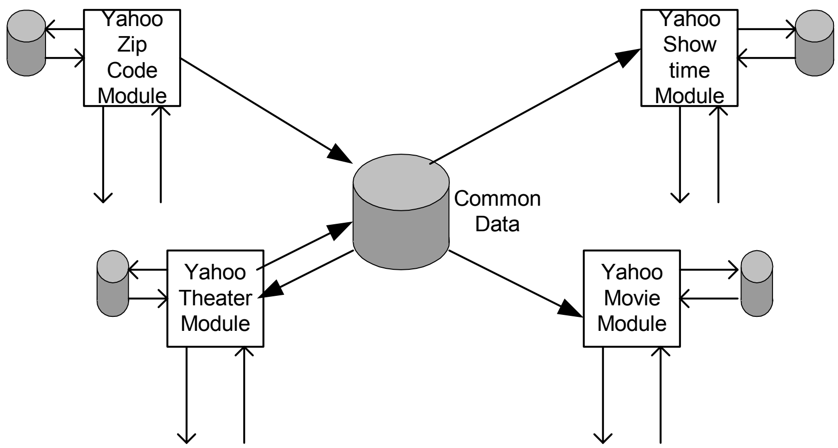


Figure 4.4: The Yahoo modules

The Rotten Tomatoes and MovieLens modules must also negotiate a data fusion process. While each of these three sites lists the title of each movie, subtle variations in title format (*The Thomas Crown Affair* vs. *Thomas Crown Affair, T*he) and different releases of movies with the same name (Is that the 1999 or the 1968 version of *The Thomas Crown Affair*?) make fusing the data a non-trivial problem. The majority of the data used by the recommendation framework comes from Yahoo. In order to restrict the amount of data fusion required and limit the impact if the fusion process fails, MLRF uses the identification numbers assigned by Yahoo. The data fusion process is relatively similar for both the Rotten Tomatoes and the MovieLens modules.

- Each Yahoo movie title is converted to a title search string appropriate for the module. This consists of removing all stop words (i.e. and, or, the) and special punctuation (i.e. ':', ',', '"') and formatting the remaining words for the given module.

- The formatted search string is submitted to the site.
- The results of the title search are validated by hand (Figure 4.5). Due to the relatively small number of title strings searched for each week, it is a minor task to have a human "operator" validate each search. One of three things can happen.
  - ♦ First, if only one matching title is found, the title and year for the movie from Yahoo Movies are displayed with the title and year for the matched movie. The operator is given the opportunity to override the match (happens rarely when two movies with the same title have been released in different years, yet only one is in the searched site's database).

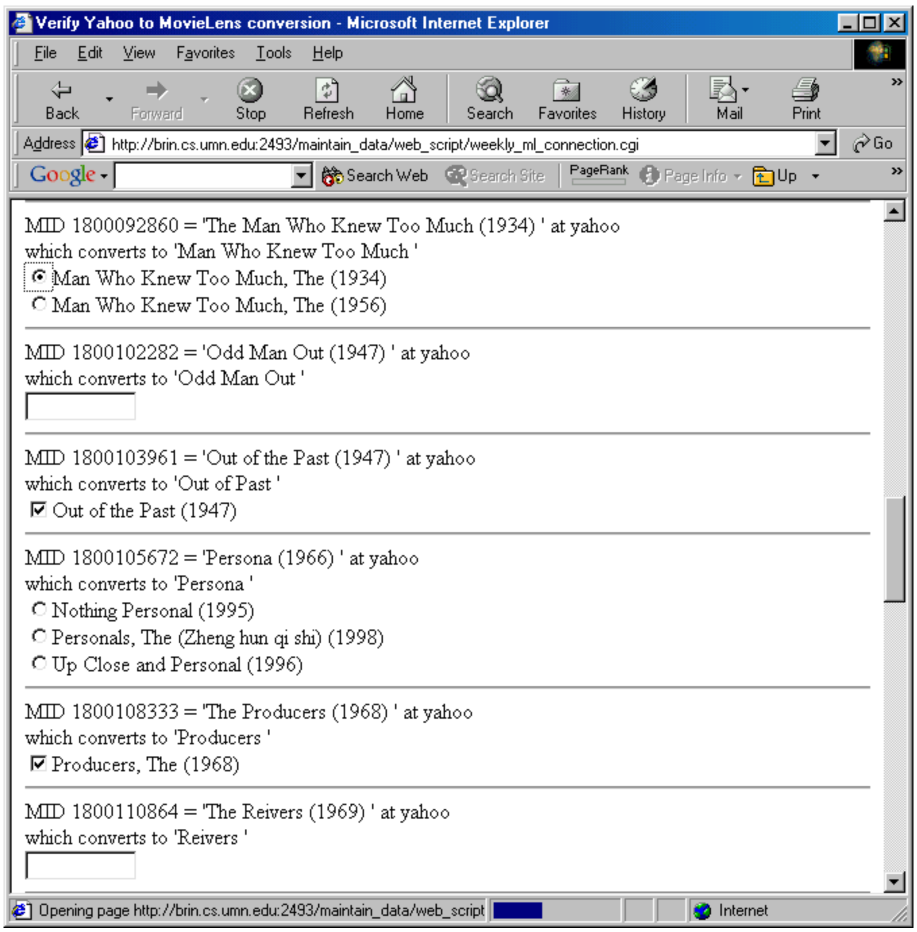


Figure 4.5: Data fusion manual validation process for MovieLens.

  - ♦ Second, if several matching titles are found the title and year for the movie from Yahoo Movies are displayed with the title and year pairs for the matched movies. The operator is given the opportunity to select the "correct" one if it is present.

- ♦ Third, if no matching title is found the title and year for the movie from Yahoo Movies are displayed along side a text box. In a separate window, the operator may search the site in question by hand (trying different variations of the Yahoo title), and if a match is found, the operator may enter the id number.

- Upon completion of the hand verification, a file is created which maps each Yahoo id to its corresponding Rotten Tomatoes or MovieLens id.

The base algorithm employed by the computation layer is based on an Extended Boolean Information Retrieval algorithm proposed by Salton et al. [67]. They propose this algorithm as a way to rank partial matches in Boolean queries in the domain of document retrieval. In traditional Boolean retrieval, the keyword query "Computer AND Science" will not return documents containing only the word "computer." However, Salton et al. propose that in many situations this document is better than documents containing neither of these keywords. Thus, their algorithm returns this first document higher than these "null" documents but lower than documents containing both keywords. Additionally, it provides a capability to weight each of these keywords. For example, users may indicate that a document containing only the word "computer" should be treated more favorably than a document containing only the word "science."

This algorithm is an ideal initial choice for meta-recommenders. Consider the task of selecting a movie to see. In essence, a user submits a query that says "I want a movie that is a comedy or family movie rated no "higher" than PG-13, containing no sex, violence or bad language, lasting less than two hours and, showing at a theater in my neighborhood." A traditional Boolean query of these requirements will return only movies matching ALL of these features. Most users, however, will settle for a movie matching a majority of these features. As applied in the computation layer, this algorithm treats the recommendation process as the submission of an AND joined information retrieval query using Equation 4.1. In this equation, I is the item being evaluated (a movie, theater, show time triple), Q is the "query" provided by the user, w_a is the weight associated with "feature a" by the user, and d_a is the degree to which the feature matches the user's query.

$$Similarity(I,Q) = 1 - \sqrt{\frac{\sum_{a \in features} w_a^2 (1-d_a)^2}{\sum_{a \in features} w_a^2}} \quad \text{(Equation 4.1)}$$

The value of d_a is calculated as follows:

- For features that match a requirement on a single option (i.e. "the movie should be less than 130 minutes in length"), each item is represented by a binary score of 1 or 0. For example, a movie less than 130 minutes is represented by a score of 1 while a movie greater than 130 minutes is represented by a score of 0.
- For features that can match on one of several options (i.e. "the movie should be either comedy or family movie"), each item is represented by a standard Boolean score based on the submission of an OR joined query on the requested options. For example, a movie with one of its genres listed as "comedy" is represented by a score of 1. A movie with one of its genres listed as "comedy" and one of its genres listed as "family" is represented by a score of 1. A movie with none of its genres listed as "comedy" or "family" is represented by a score of 0. In fact, this procedure fails to use some of the power of the Extended Boolean Information Retrieval algorithm. The base algorithm is designed to score items with two or more items in an OR joined string higher than items containing only one of the items in that string. However, this distinction was deemed to be irrelevant in this recommendation domain. That is, to most users a "family comedy" is no better a match than simply a "family" movie. Thus a standard Boolean OR is used instead.

- For features in which the input value is a numerical score (i.e. the MovieLens predicted rating, the average user score), each item is represented by a normalized score from 0 to 1 inclusive. For example, MovieLens predictions range from 1 to 5 stars. A 5 star movie is normalized to a score of 1. A 1 star movie is normalized to a score of 0. A 3.5 star movie is normalized to a score of 0.625.

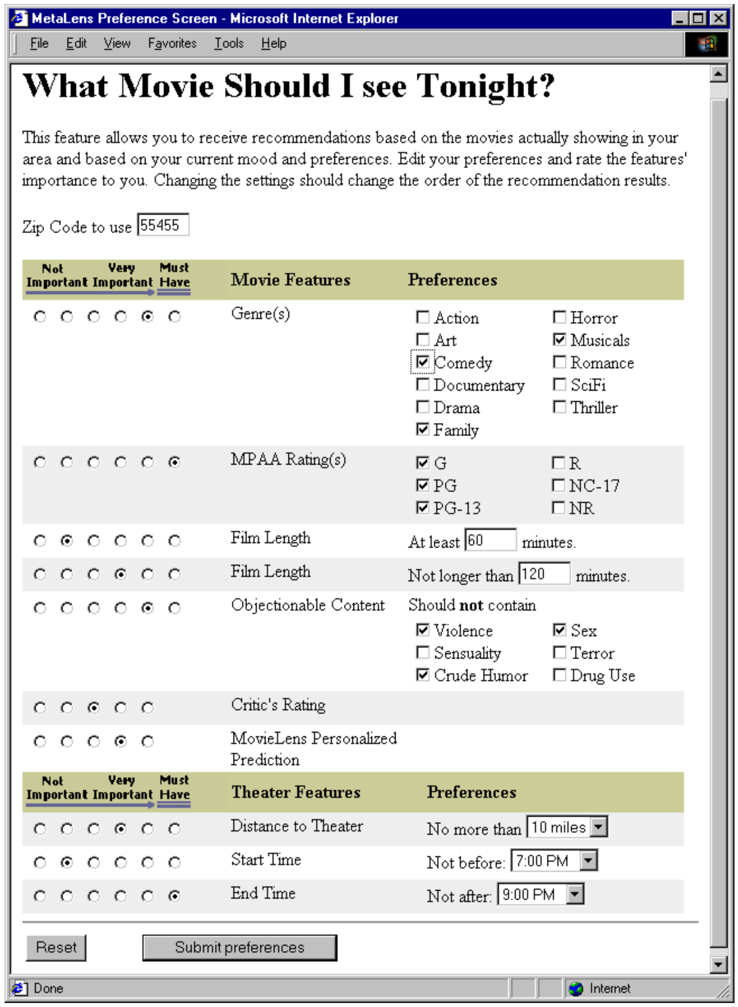
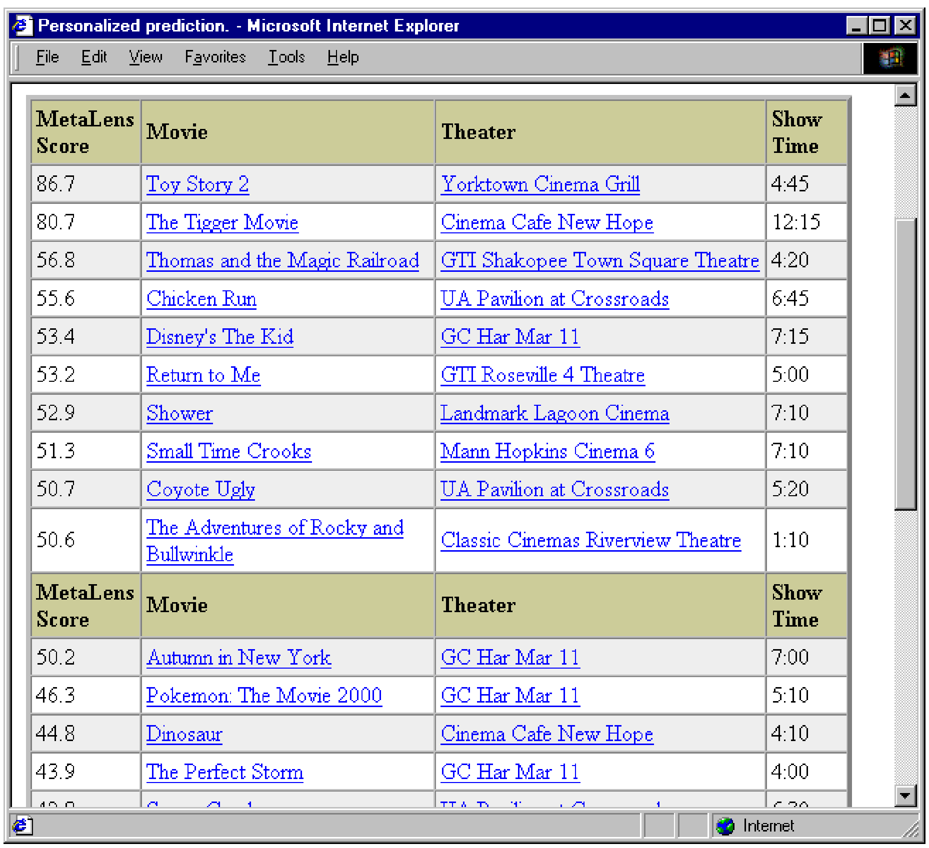


Figure 4.6: MetaLens preferences screen

The user interface portion of the interface layer consists of two screens. On the preferences screen (Figure 4.6), users indicate their ephemeral "requirements" for the type of movie they would like to see. They do this by providing information on what factors they consider important and how important it is that the recommended movie match each factor. As an example, Figure 4.6 might represent the configuration of the user in our previous scenario. When a user submits his preferences, the interface layer validates the information provided, formats it, and transfers control to the computation layer.

In order to make recommendations, the computation layer needs information concerning the theater, movie, and show time information for the user's provided ZIP Code. It requests this information from the data layer. The data layer gathers the information, either from the local cache or through runtime data acquisition as previously described. This data is returned to the computation layer, which converts the data to item match scores for each item (d_a in the previous algorithm), calculates a "Similarity" score, and returns a ranked list of these items and scores to the interface layer.



MetaLens Score	Movie	Theater	Show Time
86.7	Toy Story 2	Yorktown Cinema Grill	4:45
80.7	The Tigger Movie	Cinema Cafe New Hope	12:15
56.8	Thomas and the Magic Railroad	GTI Shakopee Town Square Theatre	4:20
55.6	Chicken Run	UA Pavilion at Crossroads	6:45
53.4	Disney's The Kid	GC Har Mar 11	7:15
53.2	Return to Me	GTI Roseville 4 Theatre	5:00
52.9	Shower	Landmark Lagoon Cinema	7:10
51.3	Small Time Crooks	Mann Hopkins Cinema 6	7:10
50.7	Coyote Ugly	UA Pavilion at Crossroads	5:20
50.6	The Adventures of Rocky and Bullwinkle	Classic Cinemas Riverview Theatre	1:10
MetaLens Score	Movie	Theater	Show Time
50.2	Autumn in New York	GC Har Mar 11	7:00
46.3	Pokemon: The Movie 2000	GC Har Mar 11	5:10
44.8	Dinosaur	Cinema Cafe New Hope	4:10
43.9	The Perfect Storm	GC Har Mar 11	4:00

Figure 4.7: MetaLens recommendation screen (Default format)

The interface layer first trims the recommendation list to contain only the highest rated triple for each movie – that is, each movie is recommended once in conjunction with the theater and show time that best fits the user's requirements. It then checks to see what additional requirements the system may have ("only display the top-10 movies") and returns this information to the user interface. The recommendation screen (Figure 4.7) displays an ordered list of recommendations by identifying the highest rated triple for each movie. Thus, according to the recommendations in Figure 4.7, MetaLens recommends that the user in our scenario should take her nephew to see the 4:45 showing of *Toy Story 2* at the Yorktown Cinema Grill.

The interface layer can also communicate directly with the data layer. For example, movie and theater names in the recommendation screen are presented as hyperlinks. Clicking one of these links spawns a separate browser window displaying detailed information about the item selected. This data is requested by the interface layer directly from the data layer.

4.4 Experiment One: Recommendation Format

Experiment One was designed to consider Research Challenge 1, "What format should meta-recommendations take[9]?" Consider the recommendations presented in Figure 4.7. While the interface shows that MetaLens finds *Toy Story 2* a slightly better choice than *The Tigger Movie*, it provides the user with no information to help her decide to take this recommendation. A skeptical user might want to validate that *Toy Story 2* is indeed the better choice. An inquisitive user might wonder why MetaLens finds these two much stronger choices than *Thomas* and *Chicken Run* – two movies which, on the surface, would also seem like reasonable alternatives to this scenario.

This research question addresses what, if any, information users would like to see displayed with their recommendations. To answer this, we identified four formats for displaying MetaLens' recommendations[10]:

Default – The "bare bones" format seen in Figure 4.7. Users are provided a ranked list of movie/theater/show time triples, and each triple's corresponding MetaLens score. No additional information is provided.

[9] While it is equally important to consider what format the preferences interface should take, we chose to delay this research. It is our belief that no matter how good the interface for indicating preferences, users won't use a system if they don't find the recommendations helpful. Thus, an initial design of the preferences interface (explained in later sections) was selected based on "common sense" and commercial comparison-shopping sites such as Active Buyer's Guide and Frictionless.

All – The opposite of the Default format, this format displays each movie or theater's values for each of the features considered in the recommendation process.

Custom – This format displays a subset of the information used in the recommendation process. One way to incorporate the lessons learned from customizable portals is to allow users to customize which values are displayed. A user's selection is provided through a "what information" screen appearing between the preferences and recommendation screens.

Automatic – This format also displays a subset of the information used in the recommendation process. Which information is displayed is based on an assumption that any feature weighted highly is important and should appear with the recommendations.

4.4.1 Hypotheses

Prior to beginning Experiment One, we proposed the following hypotheses concerning Research Question 1.

Hypothesis 4: In increasing order, users will prefer the Default, All, Custom, and Automatic formats.

We hypothesized that the Default format doesn't provide enough information to users. While additional information is readily available through the movie or theater links, this requires seemingly unnecessary effort. Conversely, while the All format provides additional information about which the users care, it also provides information about which they do not. In essence, the All format introduces information overload into a system designed to fight information overload. Based on this reasoning, we hypothesized that the Custom and Automatic formats would be the preferred formats. Each provides a subset of the information available, presumably, a subset containing information about which the user cares. We hypothesized that because the Automatic format provides the additional information with less effort required by the user, it would be preferable to the Custom format.

Hypothesis 5: Users with little prior knowledge of the recommended items will prefer recommendation formats providing more recommendation data.

---

[10] The names used to identify recommendation formats are used for clarity of explanation in this and future discussions. Such names were never used with research subjects.

It is presumed that users come to a system such as MetaLens with prior knowledge of many of the movies being recommended. Users may recall advertisements, reviews, and recommendations from friends for a number of the movies recommended. Because of this, users may be less likely to need additional information displayed with the recommendations. For example, many users are likely to know enough about the movies recommended in Figure 4.2 to know that the first few are children's movies and likely to be rated G or PG. Similarly, many are likely to suspect that *Coyote Ugly* is not appropriate for children. Thus, the need for displaying the MPAA information may be reduced for informed users. What happens, however, when users have little to no prior knowledge about the items being recommended? Does prior knowledge change which recommendation formats users prefer? It was our hypothesis that users without prior knowledge of the movies being recommended would show a higher interest in the "all" and "custom" formats because these two formats provide users greater access to the information they may not already have.

4.4.2 Experimental Design

Subjects for this experiment were selected from the pool of active and established users of MovieLens. Members in this category had been members of MovieLens for a minimum of three months prior to the experiment's start date, had visited MovieLens a minimum of three times during that period, and had provided the system with at least ten ratings. A random sampling of 125 qualified subjects were sent email invitations to participate in this online study. Respondents were sent the URL for an experimental server and told the entire process would take 30-60 minutes.

Upon completion of consent and instructions, subjects were asked to complete four tasks. For each task, the subject was presented with a scenario representing a situation for which they might be attempting to select a movie showing in their local theaters. These consisted of a random ordering of the following scenarios:

Scenario A: It is guys/girls-night out – you are going out with a group of several close, same gender friends. Pick the movie that the group should go see.

Scenario B: Your 8 year old nephew is visiting you. Pick a movie that is age appropriate, but that you might still enjoy.

Scenario C: You are setting up a "first date." Pick an "appropriate" movie for just such an occasion.

Scenario D: You have the opportunity to go out by yourself. Pick a movie that you might see if you have no one else to worry about.

Subjects used the MetaLens preference (Figure 4.8) screen to indicate their requirements for the given scenario. Preferences were gathered for eight data points. These consisted of genre, MPAA rating, film length, objectionable content, distance to the theater, start/end time, a critic's rating, and the subject's personalized MovieLens prediction for each movie. Upon submission of their preferences, subjects were presented with the top ten items from their recommendation list presented in one of the four recommendation formats (Figure 4.9). These too were randomly ordered such that each subject saw each of the four recommendation formats.

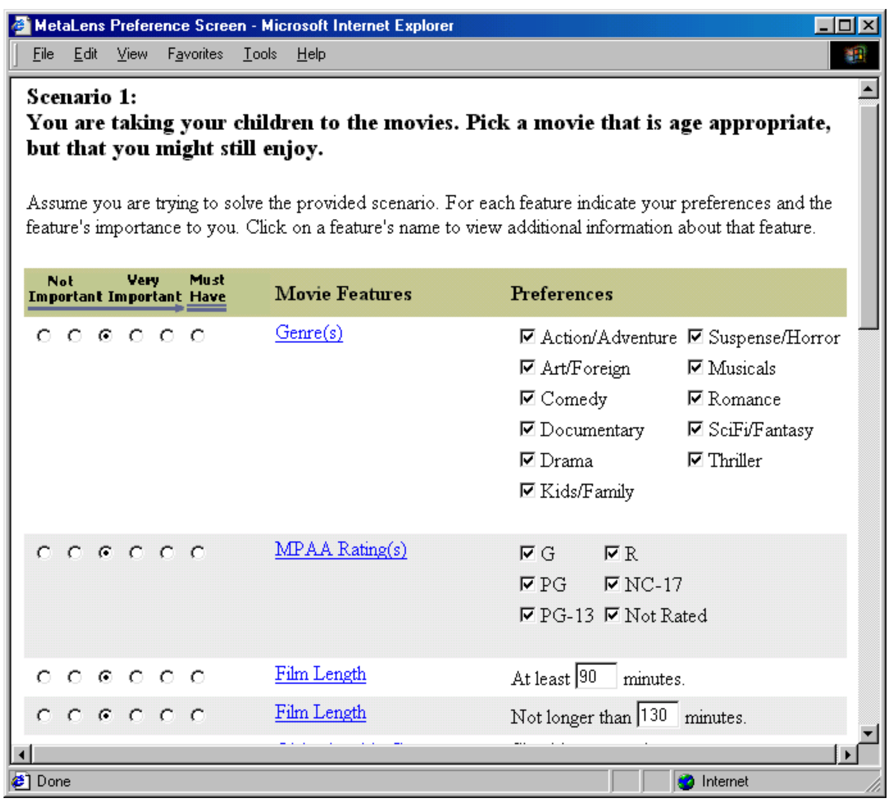


Figure 4.8: Experimental MetaLens preferences screen.

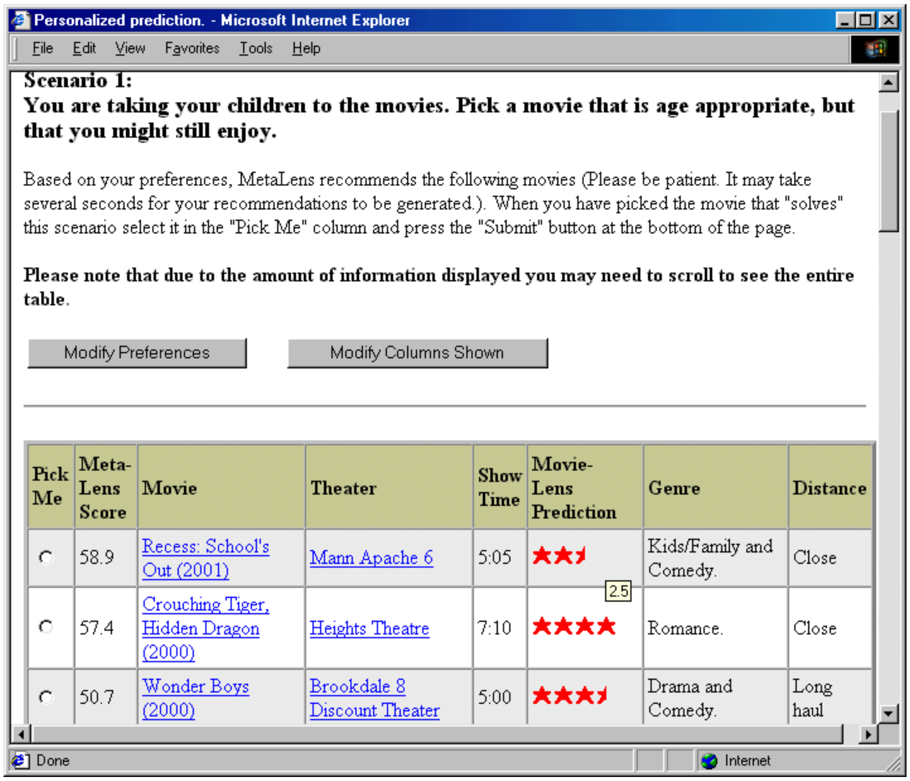


Figure 4.9: Experimental MetaLens recommendation screen.

Subjects were allowed to ask for additional information about any of the recommended movies or theaters. As previously described, selecting the hyperlink of the item in question spawns a separate browser window containing additional information about the item. From the recommendation screen, subjects could return to the preferences screen and reconfigure and resubmit their preferences. If they were currently viewing the Custom recommendation format, they were also given the option to re-select which information was in the recommendation table. To finish the task, subjects were asked to select a movie triple they felt “fit the scenario.” (Recall that each movie appears only once in the recommendation list with the highest rated theater and show time completing the triple.)

Between tasks, subjects were asked to complete a task survey, which asked them to answer three questions regarding the task and recommendation format they had just completed. These consisted of “scaled score” questions concerning how confident they were that the movie selected fit the scenario, how helpful the recommendation format was, and how much they had to rely on additional information.

Upon completion of all four tasks, subjects were asked to complete an exit survey. This consisted of several screens where they provided a unique rank for each of the four recommendation formats (from least to most helpful), free form comments on what they liked or disliked about their top and bottom choices, scaled scores on the MetaLens system in general, and scaled scores regarding the presented scenarios.

In addition to the specific survey answers requested from subjects, logging was built into the system to track the time required to complete each task,and the number of times a subject requested additional information screens.

Subjects were randomly assigned to one of two experimental groups. Group A represented "uninformed users." Movie titles were scrambled on recommendation and information pages. While it may have been possible to figure out what a scrambled movie was by examining the data on its information page, subjects were discouraged from doing so. Instructions asked members of Group A to approach the experiment as though they were selecting from a set of unknown films. Group B represented "informed users." Movie titles were presented normally, and subjects were instructed to use their prior knowledge when making decisions. In addition to these variations, subjects from Group A received an additional screen of questions in their exit survey. This screen identified the unscrambled title of the four movies they had selected, and subjects were asked once again to indicate their confidence that each movie fit the corresponding scenario.

4.4.3 Metrics

Results for this study are based on the comparison of a variety of measured quantities and subject-provided scores. When comparing scores provided by each subject, the mean differences were compared using a pairwise T-test. When comparing quantities or scores between members of the two research groups, an independent sample T-test was used. Mean differences with p-values greater than 0.05 are not considered statistically significant and are not discussed in the following sections.

4.4.4 Results

Of the 125 users invited to participate in this experiment, forty-nine test subjects consented to participate and completed the experiment. To answer the research questions, we evaluated these 49 subjects on seven measured, independent variables. These consisted of time to complete task, requests for additional information, subject-reported scores for confidence, format helpfulness, and dependence on external information, subject ratings of recommendation formats, and revisions to confidence score for subjects from Group A.

Three distinct classes of results are worth considering. First, to answer research question one, we attempted to identify differences in the mean score recorded for each of these seven variables based on the recommendation format presented. Second, to answer research question two, we attempted to identify differences in the mean score for each of these variables based on the experimental groups. Finally, we attempted to show that there were few correlations between the remaining variables—mainly task order and scenario presented – and these independent variables.

Hypothesis 4 stated, "**In increasing order, users will prefer the Default, All, Custom, and Automatic formats**." In the process of designing a useful meta-recommender, we wanted to consider which recommendation format subjects found most helpful. At the completion of the study, subjects were asked to provide each of the four recommendation formats with a unique ranking from least helpful to most helpful. These rankings were converted to numerical scores from zero to three respectively. The mean average ranking across all 49 subjects is presented in Table 4.1. Furthermore, this relative ranking was also the most common. Twenty-seven of 49 subjects ranked the formats in this order while thirteen more provided an ordering which reversed a single, consecutive pair.

Format	Average Ranking	Helpfulness	Confidence	Time to Complete Task
Default	0.16 (0.51)	3.28 (1.19)	3.48 (1.06)	225 (142)
Automatic	1.22 (0.69)	3.88 (0.73)	3.72 (1.01)	218 (148)
Custom	2.04 (0.68)	4.02 (0.65)	3.91 (0.86)	272 (154)
All	2.57 (0.74)	4.09 (0.72)	3.77 (0.87)	250 (127)

Table 4.1: Experiment One survey results. [Mean (Std. Dev.)]

These results were not what we had originally hypothesized. Although we were correct in proposing that the default format would be the least helpful (44 of 49 subjects reported it as such), we were wrong about the relationships between the remaining three. We had proposed that Automatic would be considered most helpful with Custom being ranked a close second. This hypothesis is incorrect on two counts. First, neither of these formats was reported most helpful. In fact, a majority of subjects ranked the All format as most helpful (34 of 49 subjects). Our hypothesis was also incorrect on the relational ordering of these two formats. In fact, subjects reported that they found Custom more helpful than Automatic.

One could argue that the prior results are "tainted" by the fact that the experimental design asked subjects to provide a unique ranking for the four recommendation formats while providing rather limited interaction with each of the formats. To provide this ranking, subjects must have had enough memory of each of the four formats to be able to produce a meaningful way to separate and rank them. While the consistency of the data suggests this was not an issue, we can partially check the validity by comparing these with the subject-reported scores of confidence and helpfulness of each of the recommendation formats. Since these values were gathered immediately after each task, the recommendation format should have been fresh in the subject's mind.

Subjects reported a lower helpfulness score when completing the task providing the Default recommendation format. Table 4.1 shows that subjects provided an average helpfulness score of 3.28 (on a scale of 1 to 5) for the Default format compared to a score of 3.88 for the Automatic format. Furthermore, both of these were considered less helpful than the All or Custom formats (with scores of 4.09 and 4.02 respectively).

With nearly similar results, subjects reported an average confidence score of 3.48 with the default format compared to averages of 3.72, 3.77, and 3.91 for the "my high," "all," and "custom" formats. Likely this is because the amount of information that is reported in the default format is minimal. We might assume that to overcome this subjects would be required to rely on previous knowledge or spend time searching the additional information screens to gain back their confidence. However, analysis of the data does not suggest that subjects were doing this. That is, subjects were not reporting higher "used extra" scores, nor were they using the movie links with any higher frequency. Thus, it is not surprising that they would be less confident with their decision.

Finally, we considered the amount of time required to complete the task based on the recommendation format presented in the task. Automatic and Default require less time than the All or Custom formats. It would seem as though tasks using the Default format should take more time to complete than the others since subjects would need to do additional research or, at the very least, rack their brain for prior knowledge. On the contrary, prior reported observations noted that subjects did not report relying any more heavily on prior knowledge with the Default format, nor did they use any additional requests for information. These observations, combined with this timing information, imply that the lack of information seemed to frustrate subjects. Rather than take the time to weed through the additional information, they simply made a decision – a choice that they admit causes them to be less confident with the outcome. In fact, this result seems to imply that when they have "good information" subjects take the time to think about the problem and when they have "bad information" they simply "come up with an answer."

Hypothesis 5 posed that "**Users with little prior knowledge of the recommended items will prefer recommendation formats providing more recommendation data.**" The fundamental result in the previous section was the ranking of the four recommendation formats. We note that this ranking is consistent between informed and uninformed users (Table 4.2). This finding contradicts our hypothesis since both groups of subjects found a given format equally helpful.

	Group A (25 subjects)	**Group B (24 subjects)**
Default	0.12 (0.44)	0.208 (0.59)
Automatic	1.08 (0.57)	1.375 (0.77)
Custom	2.04 (0.54)	2.042 (0.81)
All	2.76 (0.52)	2.375 (0.88)

Table 4.2: Experiment One recommendation format rankings. [Mean (Std. Dev.)]

Important differences between the experimental groups do exist, however. Probably the most extreme was the number of requests for additional information. Members of Group A (uninformed users) followed links to movie pages an average of two times per task. Conversely, subjects in Group B (informed users) used these links only 0.2 times per task. This is not entirely unexpected since members of Group B were expected to bring prior knowledge to the process. They were likely to recognize the movies being recommended, and many of the factors being used to differentiate between the movies were already in the back of their minds. Subjects from Group A, on the other hand, were less able to bring prior knowledge into the process and were more likely to need to follow links to discover information they could use to choose their solution.

Directly related to this is the time that was required to identify a solution. Subjects from Group A required approximately 30% longer to identify a "solution" to each scenario. While subjects from Group B were able to complete scenarios in 3.5 minutes on average, subjects from Group A required nearly 4.5 minutes. This can best be attributed to the additional time it took to analyze the information screens.

Next, subjects from Group A were consistently less confident with their selected movie than members of Group B. While members of Group A averaged a confidence score of 3.0 (corresponding with a response of being "neither certain or uncertain" about their selection), members of Group B averaged a score of nearly 3.5. This is somewhat unexpected when you consider that members of both groups had identical tools available to them. While members of Group A lacked previous knowledge they could bring into the process, one would assume that they could make up for this lack of knowledge through the supplemental information screens. The fact that they are unable to gain enough knowledge to bring their confidence to levels equal to that of their informed colleagues implies that perhaps there is more to the decision-making process than factual knowledge. Although movie distributors have counted on it for years, perhaps subjective factors such as a "gut reaction" when viewing a trailer play a much larger role than we had previously thought.

Supporting this is the fact that members of Group A consistently reported that they used less outside information in making their decisions. The question posed to them clearly indicates that they should include both prior knowledge and knowledge gained from the movie information pages when reporting outside knowledge, yet members of Group A report an average "outside knowledge" score of only 2.9 compared to a score of

3.3 for their peers in Group B. This too supports the belief that there are subjective factors that influence the final decision about the movies that fit the scenarios. Perhaps members of Group A felt that there was some outside knowledge that they might normally have used, which they were unable to include when they are unable to recognize the movies from which they were selecting.

Recall that subjects from Group A were asked to select a solution from scrambled movie titles. During the exit survey, subjects in this group were shown the unscrambled title of the film they had selected for each scenario and given the opportunity to modify their confidence. Subjects were far more likely to make a significant change in their confidence score (changes of +/-2) when the recommendation was provided through the Default format. When using the Default format, ten subjects made significant changes in their confidence. This compares to significant changes from seven subjects when receiving the All format, and four subjects each with Custom and Automatic.

Data indicates that the viewing of additional information screens by subjects in Group A had an effect on confidence. Those who used these screens were less likely to revise their confidence score than those who did not use these screens (21 revisions vs. 39). When they did revise their score, they were more likely to raise their confidence level (71% vs 51%). Finally, no matter which direction they changed their confidence score, they changed it by a much smaller amount (delta 0.73 vs 1.35). All of this suggests that subjects who make themselves informed prior to making a decision feel they make better decisions. Since one of the benefits of meta-recommenders is the availability of recommendation data, system designers need to consider ways in which they can make this data accessible to the users. Doing so may provide users with a system in which they feel more confident with their decisions.

Two **other results** were detected that, although not directly affecting the research questions in this study, are worth mentioning. First, it was observed that the order of the tasks has a direct effect on the amount of *time* required to identify a "solution." Regardless of which scenario or recommendation was used or which test group they represented, subjects required nearly 320 seconds to tell MetaLens their requirements, analyze the movies in the recommendation list, and select the movie they would pick to solve the first task. Subjects required just 230 seconds to complete the second and third tasks and had streamlined the process to approximately 180 for the final task. These results are not surprising as none of the subjects had an opportunity to use this interface prior to the experiment. Perhaps these time differences can be attributed to a learning curve.

Another explanation is that subjects start to recognize movies in later tasks. This results in a lower need to perform additional analysis, thus decreasing the time needed to solve the task. This, however, was *not* supported by the data. In fact, data regarding the order of tasks and the number of times a subject requested additional information shows the exact opposite. Across tasks two through four, subjects averaged a rather consistent 1.3 requests per task. This is likely explained through the observation that the scenarios were rather different. Since requirements changed significantly from one scenario to another, subjects were seeing different movies in each of their top-10 lists. Thus, even in later stages they were still in need of additional information concerning movies about which they were unsure. However, this same figure is a remarkably low 0.5 requests per person during task one. One explanation is that subjects were so busy learning the new system that they forgot they could get additional information by following the movie links.

The second result is that the specific scenario with which a subject was working may have an effect on their confidence with their final selection. Subjects reported the highest confidence levels (4.11 on a rating of 1-5) when selecting a movie for the "nephew" scenario. At the other extreme, subjects reported the lowest confidence levels (3.42) when completing the "first date" scenario. Confidence levels averaged an in-between 3.67 for the "same gender group" and "self" scenarios.

This result may be explained by considering the "factors" that influence which movie a subject chooses. The factors that influence a subject's decision when taking her nephew to the movies are likely few, and the distinctions are quite plain (she would *never* consider an R rated war film with violence and language). Thus, she only has to examine two or three factors. Once it is confirmed that these are met, she can be relatively confident that the movie will be appropriate. On the other hand, there are so many factors to take into account with the "first date" scenario that a subject is less confident he has selected the right film. In an interesting anecdote, numerous subjects used the comment section of this experiment to report being "old marrieds" who could not recall what they would want in a date film.

4.5 Experiment Two: Meta-recommenders With More Data

Results from Experiment One indicated that subjects prefer to see all the data used in the recommendation process. While this seems reasonable when doing so entails adding eight columns of recommendation data to the recommendation table, what happens when doing so means adding 16 or 24 columns of recommendation data? Experiment Two was designed to retest the hypothesis when considering a more extensive recommender.

4.5.1 Hypothesis

Prior to performing Experiment Two, a modified Hypothesis 4 was proposed:

Hypothesis 4b: Users will prefer the Automatic and Custom formats to either of the "All" formats.

The logic behind this modified hypothesis was identical to that used to support Hypothesis 4. With a more extensive recommender, the "All" formats (described in more detail in the following section) were expected to provide too much information in a system designed to fight information overload. Based on this reasoning, we felt that the Custom and Automatic formats would be the preferred formats.

4.5.2 Experimental Design

Experiment Two differed from Experiment One in three fundamental ways. First, the data included in recommendations was doubled. The previous data was expanded to include a non-personalized average user rating, film distributor, release date, special accommodations at the theater (for handicapped or hearing impaired consumers), and information regarding whether or not tickets to a particular show time were discounted over normal ticket prices. Additionally, the single critic's rating from Experiment One was replaced with four critical review data points representing the percentage of critics giving the movie a "thumbs up," the percentage of major market critics liking the movie, and thresholds for the minimum number of critics in each of these two categories.

Second, the four recommendation formats were modified slightly. The Custom and Automatic formats remained unchanged. However, since subjects in Experiment One reported the default format unhelpful, we chose to eliminate it. That leaves the format that subjects found the "most helpful" – the "All" format. It is worth questioning whether "all" refers to legitimately all the data used in any recommendation process, or whether we happened to "get lucky" in Experiment One and hit on an appropriate set of eight data points. To test this, we replaced "default" and "all" with these two variations. We will refer to these formats[11] as "New All" and "Old All."

Finally, Experiment Two was run through the GroupLens experimental infrastructure. The infrastructure provides a mechanism for qualifying registered users of MovieLens for experiments. In order to qualify for Experiment Two, subjects needed to have had no fewer than ten and no more than 4000 ratings in the MovieLens system and have been a member for a minimum of three months prior to the start of the experiment. Subjects who participated in Experiment One were excluded from participation in Experiment Two. There were not separate experimental groups for subjects in Experiment Two. Since all subjects were provided full access to movie and theater titles, and since it is assumed that subjects have at least partial prior knowledge about these movies and theaters, all subjects were treated as "informed users" (identical interaction to subjects from Group B of Experiment One).

Other than the three differences just explained, Experiment Two was identical to Experiment One. Subjects still completed four tasks using each of the four scenarios from Experiment One. Subjects were given the opportunity to indicate their preferences for the given scenario using the MetaLens preference screen. Upon submission of their preferences, subjects were presented with a top-10 list presented in one of four randomly assigned recommendation formats such that each subject saw each of the four recommendation formats. Subjects were allowed to return to the preferences screen and reconfigure and resubmit their preferences. To finish the task, subjects were asked to select a triple they felt "fit the scenario." Subjects completed the same surveys between tasks and the same exit survey.

4.5.3 Metrics

Results for this study are based on the comparison of a variety of measured quantities and subject provided scores. When comparing scores provided by each subject, the mean differences were compared using a pairwise T-test. Mean differences with p-values greater than 0.05 are not considered statistically significant and are not discussed in the following sections.

4.5.4 Results

Of the 75 users invited to participate[12], 32 consented and completed the experiment. Results were analyzed in a manner similar to Experiment One and are presented in the following section.

Recall that Hypothesis 4b proposed that "**Users will prefer the Automatic and Custom formats to either of the "All" formats.**" At the completion of the study, subjects were asked to provide each of the four recommendation formats with a unique ranking from least helpful to most helpful. The mean average ranking across the 32 subjects is presented in Table 4.3. Unlike the prior study, results here do not provide a definitive ranking. Although the Custom format was reported preferable to the Automatic, Old All, and New All formats, the remaining three formats were considered equally helpful by subjects.

This result seems to validate partially the working hypothesis of this experiment. Recall that we originally hypothesized that subjects would prefer the Custom and Automatic formats over the All format since the later would provide too much

---

[11] Again, format titles are used for the convenience of discussion and were not used with research subjects.

[12] We originally invited sixty users to participate in Experiment Two (approximately half that were invited in Experiment One). It was our assumption that we would receive roughly the same completion rate as that received in Experiment One. In doing so, Experiment Two would have approximately the same number of users in its single experimental group as either of the experimental groups from Experiment One. When we received a particularly poor response rate, fifteen more users were invited to participate. These fifteen users had a nearly 100% response rate, yielding a slightly higher number of users than originally intended. However, users invited in both batches were selected at random based on the same critera. It is not believed that this strange set of acceptance rates affected the outcome.

information to process. In hindsight, we realize that, with its limited data, the All used in the initial experiment was not overwhelming. In redesigning for Experiment Two, we created a much larger recommender system using twice the amount of data. We believed this would cause the New All to become overwhelming and that this portion of our original hypothesis would be validated.

However, note the change in relationship between the Custom format and Old All. Recall that Experiment One found that subjects reported All (Old All in this experiment) to be more helpful than the Custom format. While those same two formats were used in Experiment Two, we found the relationship had switched. While this may be surprising, it can be explained. In Experiment One, subjects likely recognized that Old All represented the complete set of data for the recommendation process. When subjects were asked to select which of the eight features they wanted displayed with recommendations, it is likely that a majority of the eight would be selected[13]. Thus, it was more convenient to use the All format since no configuration was needed to get nearly the same format. To put this in information retrieval terms, selecting All provided 100% retrieval with only a minor hit to relevance. However, when those same eight data points are applied to the second experiment, it is likely they do not have the same degree of "fit" with what the subjects would select when given the ability to customize. That is, not only is there data the subject wouldn't have normally chosen, there is data that the subject would have chosen that is not displayed (from the newly added data). To put this in information retrieval terms, selecting Old All likely provides sub-optimal retrieval and relevance. In short, while Old All provided convenient access to meaningful information in the Experiment One, it had lost its meaning during Experiment Two.

Format	Avg. Ranking	Helpfulness	Confidence	Time to Complete Task
Custom	2.16 (0.88)	3.41 (1.04)	3.50 (1.16)	385 (421)
Automatic	1.31(1.12)	3.41 (1.04)	3.63 (1.04)	257 (187)
Old All	1.41 (1.10)	3.71 (1.00)	3.66 (1.15)	242 (157)
New All	1.13 (1.13)	3.13 (1.16)	3.44 (1.34)	297 (328)

Table 4.3: Experiment Two survey results. [Mean (Std. Dev.)]

[13] Due to a recording error, the specific features selected by users of the Custom format were not recorded. This speculation is based on observation rather than data.

Recall that subjects were asked to provide ratings on several issues upon completion of each task. These included confidence and helpfulness. While subjects reported the custom format as being most helpful in direct comparisons, the individual helpfulness scores show a slightly different picture. Subjects reported the Old All format as more helpful than the New All format. This direct relationship is not surprising. That is, New All will contain too much information to be helpful. Subjects will spend too much time finding the data that they need amongst the data they don't need. However, the fact that subjects did not report Custom or Automatic as being more helpful than New All is surprising.

Results from Experiment One suggested that subjects were equally confident with their decision when made with any of the formats providing some form of additional information (in other words, any format other than the Default format). Results from Experiment Two confirm this observation. All four formats in Experiment Two provided some form of additional information, and subject-reported confidence scores are similar.

Finally, we consider the time required completing a task given each recommendation format. Recall that, in Experiment One, tasks using the Custom or Old All formats required more time to complete than tasks with other formats. In Experiment Two, tasks using the Custom format took longer to complete than tasks with the Old All format. While it is not completely surprising that Custom requires more time (since subjects must take the time to indicate which information they want to view), it is surprising that its relationship with Old All has changed since, with respect to each other, these have not changed between the two experiments. One explanation for this is that used in explaining the change in helpfulness. That is, while Old All had meaning in Experiment One, it does not in Experiment Two. Thus, it is possible that subjects did not understand why they were receiving recommendations with this subset of information. Rather than analyzing what's there or requesting additional information, they simply made a decision and moved on.

As with Experiment One, several **additional results** were detected that are worth mentioning. It was observed that the order of the tasks has a direct affect on the amount of time required to identify a "solution." (Table 4.4) While consecutive tasks are not different, all other pairs are. Again, these time differences can most likely be attributed to a learning curve.

	Time to Complete Task
Task 1	477 (307)
Task 2	333 (428)
Task 3	201 (119)
Task 4	176 (96)

Table 4.4: Experiment Two task completion time. [Mean (Std. Dev.)]

While the prior section reported no relationship between a subject's confidence in her movie selection and recommendation format, we have noticed marginal relationships between confidence and scenario. Table 4.5 summarizes confidence levels from both experiments.

Scenario	**Confidence - Experiment One**	**Confidence – Experiment Two**
Same Gender Group	3.67 (0.86)	3.44 (1.08)
Nephew	4.11 (0.99)	3.31 (1.26)
First Date	3.42 (0.73)	3.72 (1.05)
Self	3.67 (1.12)	3.75 (1.27)

Table 4.5: Scenario effects on confidence. [Mean (Std. Dev.)]

Experiment One reported that subjects indicated the highest confidence when picking a movie to view with their nephew and the lowest confidence when picking a movie for a first date. In Experiment Two we discovered very different relationships. Subjects reported higher confidence when deciding on a date movie than they did when deciding on a movie for a same gender group of friends or for their nephew. Notice that our subjects' confidence for the nephew and "same gender" scenarios has reversed. Unfortunately, we are unable to explain this occurrence.

4.6 Summary

4.6.1 Validity Between Experiments

The values of three variables common to both experiments changed between Experiments One and Two. The first two involve subject-provided helpfulness scores. The scores for both the Automatic and the Custom recommendation formats were lower in Experiment Two. On initial inspection this seems suspect since it appears that neither format should have changed between the two experiments. Both allow for the selection of some subset of the data used in the recommendation process, including all or none of the data. However, it is important to recall that the amount of data used in the recommendation process doubled between Experiments One and two. Thus, even though subjects were allowed to pick which data to display when using the custom format, their number of choices has doubled in Experiment Two. Presumably, subjects may be interested in some larger subset of data since there is more from which to choose. While this data may in fact help the subject make more informed decisions, the interesting data may still cause a sense of information overload and thus cause the interface to be considered less helpful. This provides a real “catch 22” for the interface designer. While subject surveys suggest that subjects want decisions based on as much data as possible (subjects had little problem suggesting additional data they would like incorporated into future versions of MetaLens), the addition of this data may cause the interface to be less helpful overall.

The third variable change comes when comparing the mean ranking assigned to All/Old All. Recall that the Old All format in Experiment Two is identical to the All format in Experiment One. Thus, the fact that the mean ranking of Old All was lower than the mean ranking of All suggests that there was a fundamental change in the approach to this identical format. It is likely that subjects in Experiment One recognized that this represented all the data used in producing recommendations and were willing to do the little bit of scrolling to see everything. In Experiment Two, those data points represent a seemingly arbitrary subset of the data used in recommendations. The lowering in ranking suggests that subjects no longer found this set as meaningful, and its higher rank was often assigned to other formats.

4.6.2 Hypothesis 4 Revisited

Hypotheses 4 and 4b were proposed to address the question “In what format should recommendations appear?” When combined, the results of these two experiments provide us with an interesting answer to this question that suggests that Hypothesis 4 was only partially correct. Recall that our initial hypothesis stated that we expected users to find the Default format the least helpful. This result was so conclusively confirmed in

Experiment One that we felt free to eliminate this as an option in Experiment Two. Our initial hypothesis also stated that we expected users to find the All format nearly equally as poor. Although not true in Experiment One, we began to see this happening in Experiment Two. It would appear that we underestimated the threshold at which All became overwhelming. We were closer to that threshold with the sixteen "variables" used in Experiment Two than we were with the eight in Experiment One.

Finally, recall that our initial hypothesis stated that we expected users to find the Automatic and Custom formats nearly equally helpful with Automatic being slightly more popular due to the fact it required less effort from the user. This turned out to be incorrect. In both experiments subjects stated they preferred the Custom format to the Automatic format. This might be explained with two, potentially complementary explanations.

First, it is likely that our assumption about how Automatic should work was off the mark. Analysis of results from Experiment Two shows that subjects ask to see an average of 8.6 features in their Custom format. Had Automatic been assigned to build the recommendations for those same tasks, an average of 5.0 features would have been displayed. Treating the features returned by the Automatic format as documents in an information retrieval problem, our Automatic technique produces a precision of 80% but a recall of only 47%. It is evident that our method of selecting which features are displayed in the Automatic format does not match subject's expectations.

While we did not spend time analyzing how to "fix" the Automatic format, two ideas come quickly to mind. First, we may have selected a poor inclusion threshold. We chose to display any feature whose weight was greater than 0.5 (the top three of six selections on the weight scale). One solution may be to reconsider the threshold at which we include features. However, this method continues to assume that the information in highly rated features is informative to the subject. Alternately, perhaps we should assume that high importance leads to low variance. Consider a user who states that MPAA rating is very important. It is reasonable to expect that most of the top-10 movies already fit the user's selection of acceptable ratings. Thus, displaying that information may not provide any value in the decision-making process. An alternate way to determine what information to display would be to determine which features have variance. For example, determining that there is great variability in the objectionable content of the top ten movies might indicate that displaying this feature would be extremely helpful in making the final decision. Either way, our hypothesis that Automatic should be more useful to users than Custom may be valid given a better interpretation of how the Automatic format should work.

A second way to explain why subjects prefer the Custom format despite its requiring more effort is that users actually *want* control over what is displayed in their recommendations. Just as we argue that MetaLens is an improvement on prior meta-recommenders because it gives users control over how the recommendations are produced, we might argue that MetaLens should allow users to have control over how recommendations are displayed. The Custom format provides users with that level of control.

While either one of these explanations may be valid independently or in tandem, we have chosen to base future designs on the assumption that users want the control over how their recommendations are displayed. While future work may explore better "algorithms" for Automatic recommendation formats, we chose to focus our exploration on how users interact with meta-recommenders.

Chapter 5: Comparing Recommender Systems

In previous chapters we have proposed that, for many common scenarios, current recommender systems do not provide users with adequate recommendations. Users must "build" recommendations for these scenarios by hand-gathering information from existing sources of recommender data. We have proposed that meta-recommenders can simplify this information-gathering process to make solving the scenario easier on the user. This chapter discusses a user-centered study that focuses on whether subjects in an experimental setting find a meta-recommender more helpful than "traditional" systems offering access to the same data.

5.1 Introduction

It is common for researchers in the field of recommender systems to publish results indicating which recommendation algorithm is most accurate within their specific system [15], [36], [45], [55], [75]. Two of these studies provide particularly interesting results.

Breese et al., have conducted perhaps the most complete algorithmic comparison research to date by comparing six recommender algorithms, from four experimental protocols, implemented in each of three problem domains [15]. Their results showed that Bayesian-clustering and nearest neighbor methods outperform other techniques, although the preferred technique depends largely on the application area and the nature of the dataset.

While Breese et al. compared a variety of algorithms, Herlocker compared evaluation metrics [36]. The lack of standard evaluation metrics has caused confusion as different researchers compare different systems using different evaluation metrics. Herlocker identified six "user tasks." These summarize the goals users have in using a recommender system. Furthermore, he identified seven classes of evaluation methods and plotted pairwise combinations of specific metrics as they evaluated results from 1280 different "recommender systems." Due to the strong correlations visible in these graphs, he concluded that as long as evaluation methods were appropriate for the same user task, their conclusions would be relatively comparable.

Unfortunately, these studies are limited in that they make conclusions based on post hoc analysis and do not include users in the evaluation process. Often these studies are based on results that state that some metric indicates that one algorithm provides slightly better accuracy than another algorithm. For example, research conducted by the GroupLens Research Project finds that the Mean Absolute Error of a good movie recommender is approximately 0.80-0.75. As researchers, we would get excited by showing that a new algorithm could lower this error to 0.70. We must question, however, whether these small improvements are significant if users are unable to detect changes or express a meaningful preference for a "better" system.

5.2 Experiment Three: A User-based Comparison of Recommenders

5.2.1 Hypothesis

Experiment Three was designed to consider Research Challenge 2, "Which interface do users prefer in a recommender system?" Prior to conducing Experiment Three, we proposed the following hypothesis:

Hypothesis 6: Users will find a meta-recommender more helpful than "traditional" forms of recommender systems.

It is our belief that meta-recommenders provide the "best of both worlds." They allow recommendations to be based on persistent knowledge about the user, and they allow the user to input ephemeral requirements specific to the current information need. Better yet, they do so by providing the users with specific control over how this recommendation data is combined. Confirmation of this hypothesis suggests that meta-recommenders have successfully combined the benefits inherent in these traditional systems.

5.2.2 Experimental Design

Experiment Three was conducted within the GroupLens experimental infrastructure using settings identical to those from Experiment Two. (That is, participants had been members of MovieLens for at least three months with between ten and 4000 ratings). Subjects who participated in either of the previous experiments were excluded from participation in Experiment Three, and 125 subjects were invited to participate.

Upon completion of consent and instructions, subjects were assigned to one of six experimental groups (Table 5.1). Each group was presented with a different ordering of three recommender interfaces. For each interface, subjects were asked to complete three tasks. For each task, subjects were presented with a scenario representing a situation for which they would be attempting to select a movie from those showing in their local theaters. The nine scenarios used in the study (Figure 5.2) were divided into three sets of three. Each set was randomly assigned to an interface, and the ordering of the scenarios within the set was randomized. "Similar" scenarios (i.e. scenarios where children are involved) were placed in different task sets to prevent the results from being tainted by having similar scenarios appear with the same interface.

	Interface order
Group 0	MetaLens, ContentLens, MovieLens++
Group 1	MetaLens, MovieLens++, ContentLens
Group 2	ContentLens, MetaLens, MovieLens++
Group 3	ContentLens, MovieLens++, MetaLens
Group 4	MovieLens++, ContentLens, MetaLens
Group 5	MovieLens++, MetaLens, ContentLens

Table 5.1: Experiment Three experimental group differences.

Interface one, referred to as MovieLens++ (ML++), attempts to mimic "traditional" interfaces by providing users with an experience most similar to how they would currently solve such scenarios. That is, they can access personalized recommendations through sites such as MovieLens, but these come with little to no information about the content of the movies being recommended. Users must coordinate these recommendations with outside information in an attempt to produce an informed decision about what movies best fit their needs.

Subjects interact with ML++ through a base window presenting the specific scenario, buttons to "MovieLens" and "Movie Listings," and form fields to enter the movie, theater and show time that they feel match the scenario. (Figure 5.1). Selecting the "MovieLens" button produces a separate window showing a scaled down version of MovieLens (Figure 5.2). Subjects have access to the standard top-5 lists provided automatically by MovieLens, the title search feature, the genre/date search feature (Figure 5.3), and the "Movie Guru film" and DVD reviews. Other features such as links to IMDB, group ratings, and the ability to rate previously seen movies are disabled for the purposes of the experiment. Selecting the "Movie Listings" button produces a separate window showing a scaled-down version of Yahoo Movies (Figure 5.4). Users may view movie and theater listings for any ZIP Code and Yahoo defined distance (a five scale textual description of the distance from the ZIP Code to the theater) or search for specific movie titles. Movies and theaters in either of these features are hyperlinks to information pages for the appropriate entity. A subject may interact with one or both "sub-interfaces," using any of the features within each sub-interface, and in any order he chooses. Once the subject has identified a "solution" to the scenario, he must return to the base window and enter the appropriate movie and theater information.

Scenario A_1	It is guys/girls-night out — you are going out with a group of several same-gender friends. Pick the movie that the group should go see.
Scenario A_2	Your 8 year old nephew is visiting you. Pick a movie that is age appropriate, but that you might still enjoy.
Scenario A_3	You are planning a *first date*. Pick an appropriate movie for just such an occasion.
Scenario B_1	You have the opportunity to go out by yourself. Pick a movie that you might see if you have no one else to worry about.
Scenario B_2	You are taking your children to the movies. Pick a movie that is age appropriate, but that you might still enjoy.
Scenario B_3	You are taking your spouse/partner to the movies. Pick an appropriate movie for just such an occasion.
Scenario C_1	It is a rainy Saturday afternoon. Choose a movie that would be a fun way to kill the day.
Scenario C_2	You are going to the movies with your parents. Select a movie that everyone should enjoy.
Scenario C_3	You are planning a movie night for a group from work. Pick an appropriate movie for just such an occasion.

Table 5.2: Experiment Three scenarios.

It is important to note that in order to track data appropriately and prevent subjects from moving to "unauthorized" sites, only limited versions of MovieLens and Yahoo movies are provided to the user. While we felt these limited versions provide the functionality used by most users in solving these types of tasks, results of usage with ML++ should be construed as only an approximation for how users *actually* interact with current systems.

Interface two, referred to as MetaLens, attempts to provide users with a well-designed meta-recommender system. It provides the technology to combine the personalized recommendations from MovieLens with information filtering-based recommendations. To do so, it uses a meta-recommender system designed from the results of the experiments described in Chapter 4.

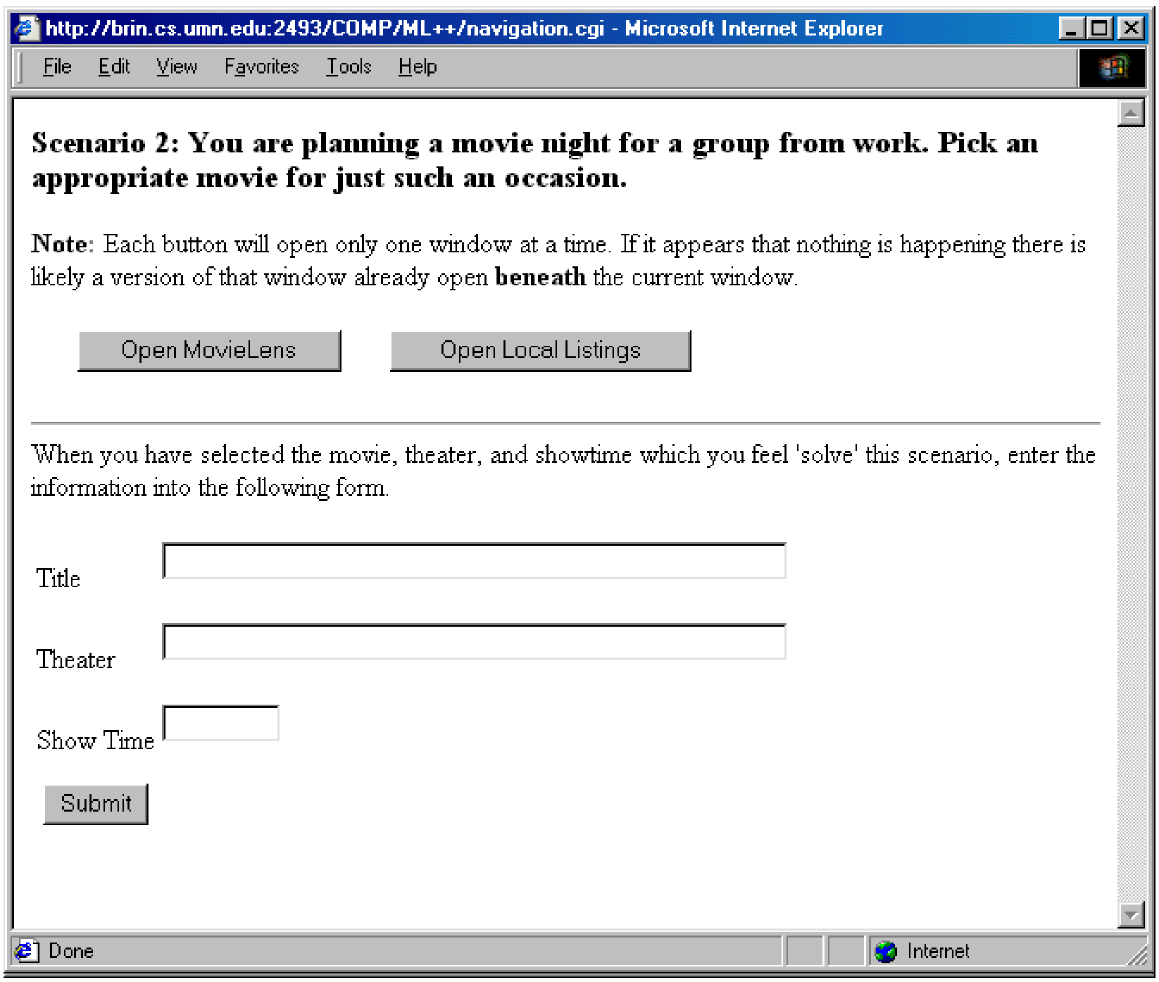


Figure 5.1: MovieLens++ base screen.

Figure 5.2: MovieLens++ MovieLens screen.

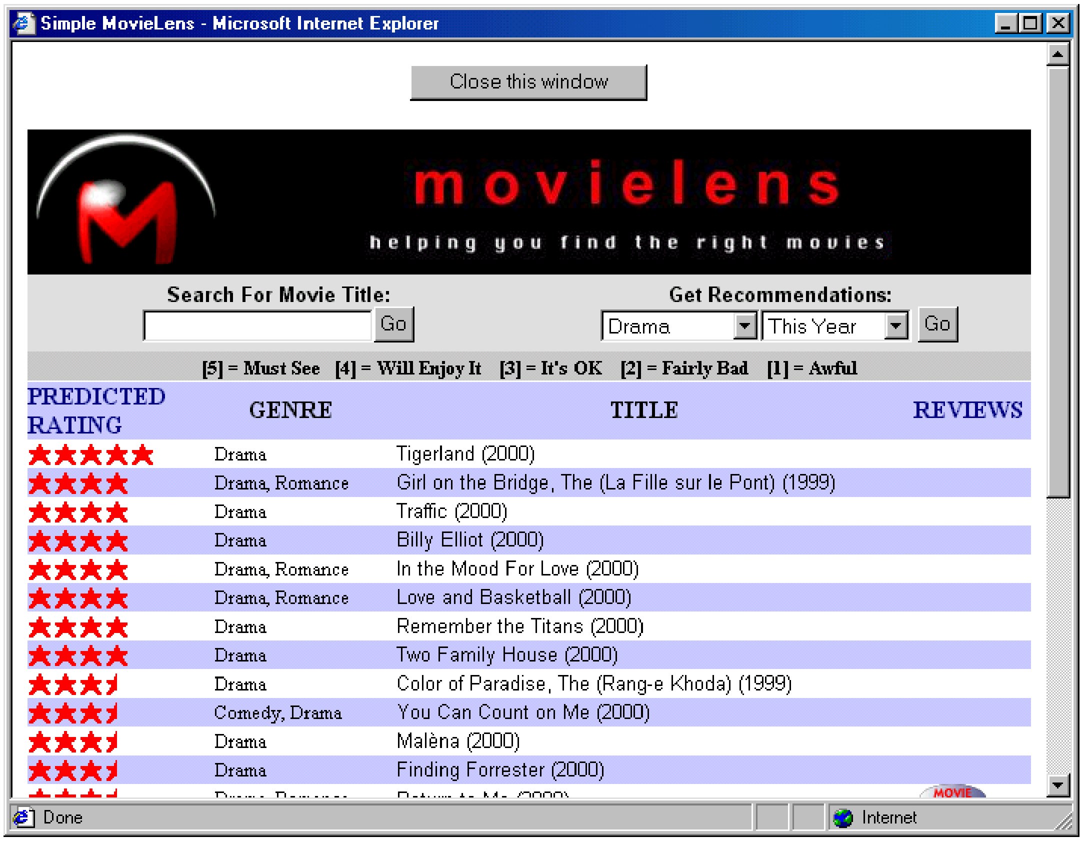


Figure 5.3: MovieLens++ MovieLens search results.

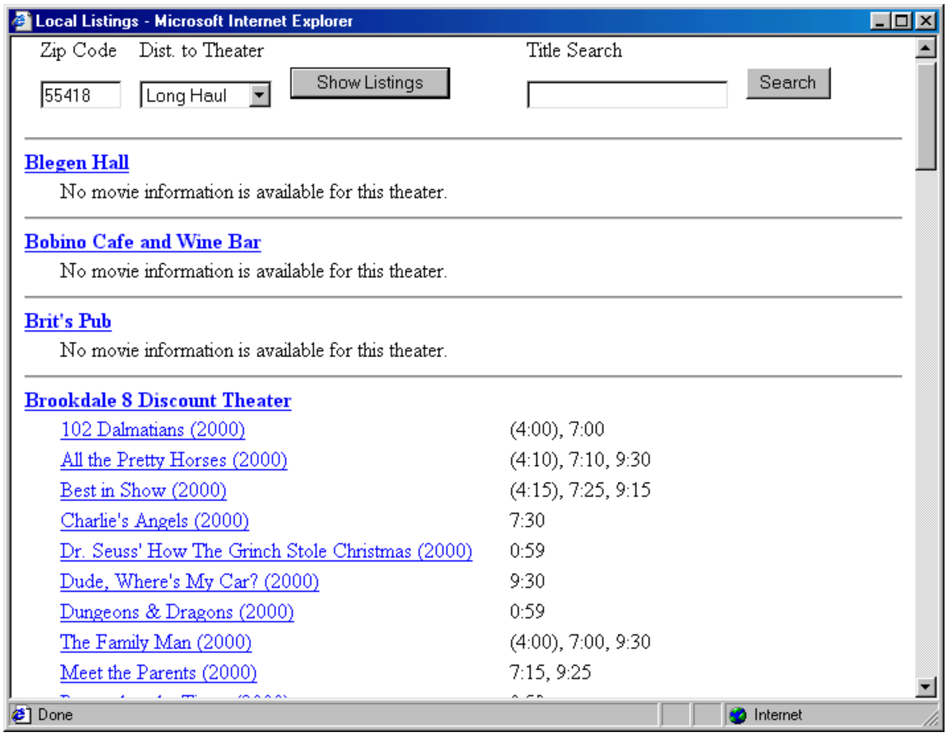


Figure 5.4: MovieLens++ movie listings.

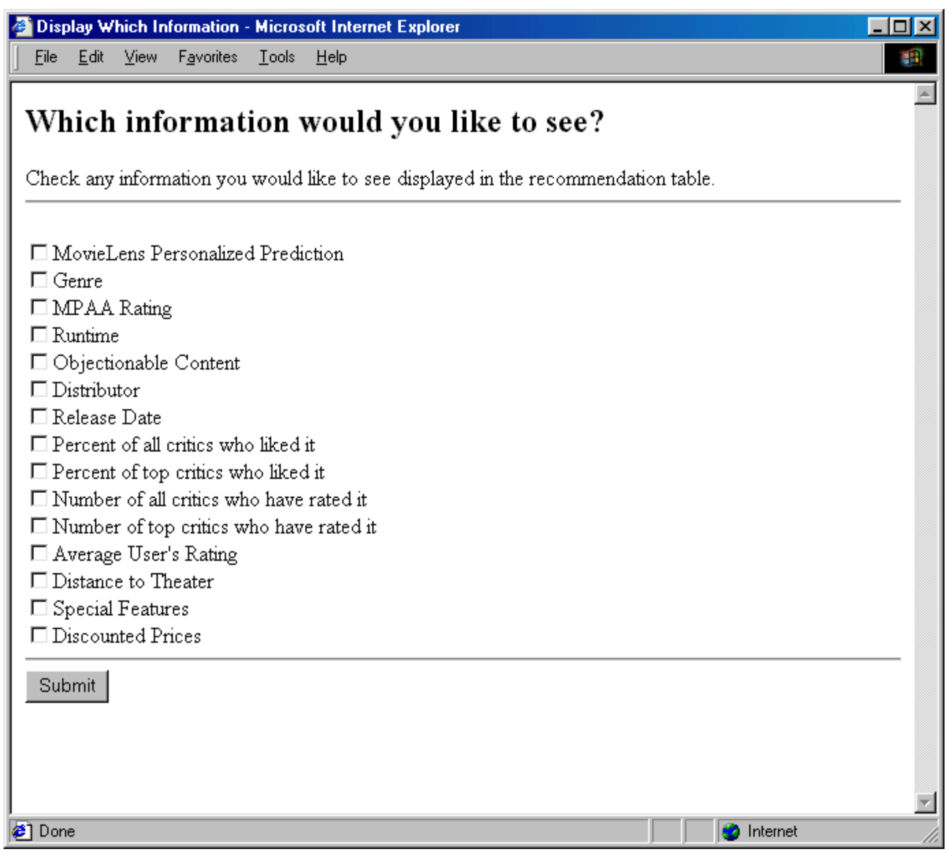


Figure 5.5: MetaLens "What Information" screen.

Figure 5.6: ContentLens recommendation screen

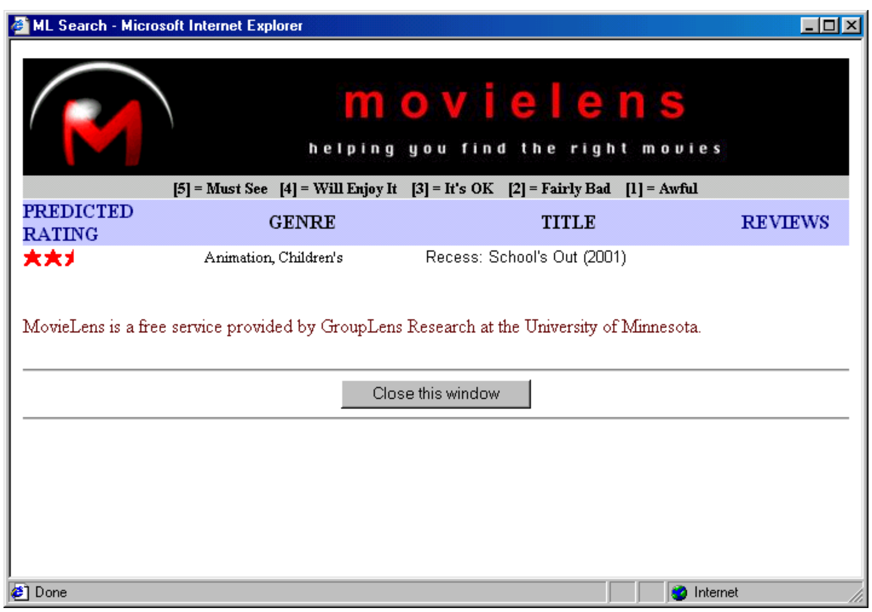


Figure 5.7: ContentLens “Show Me” feature

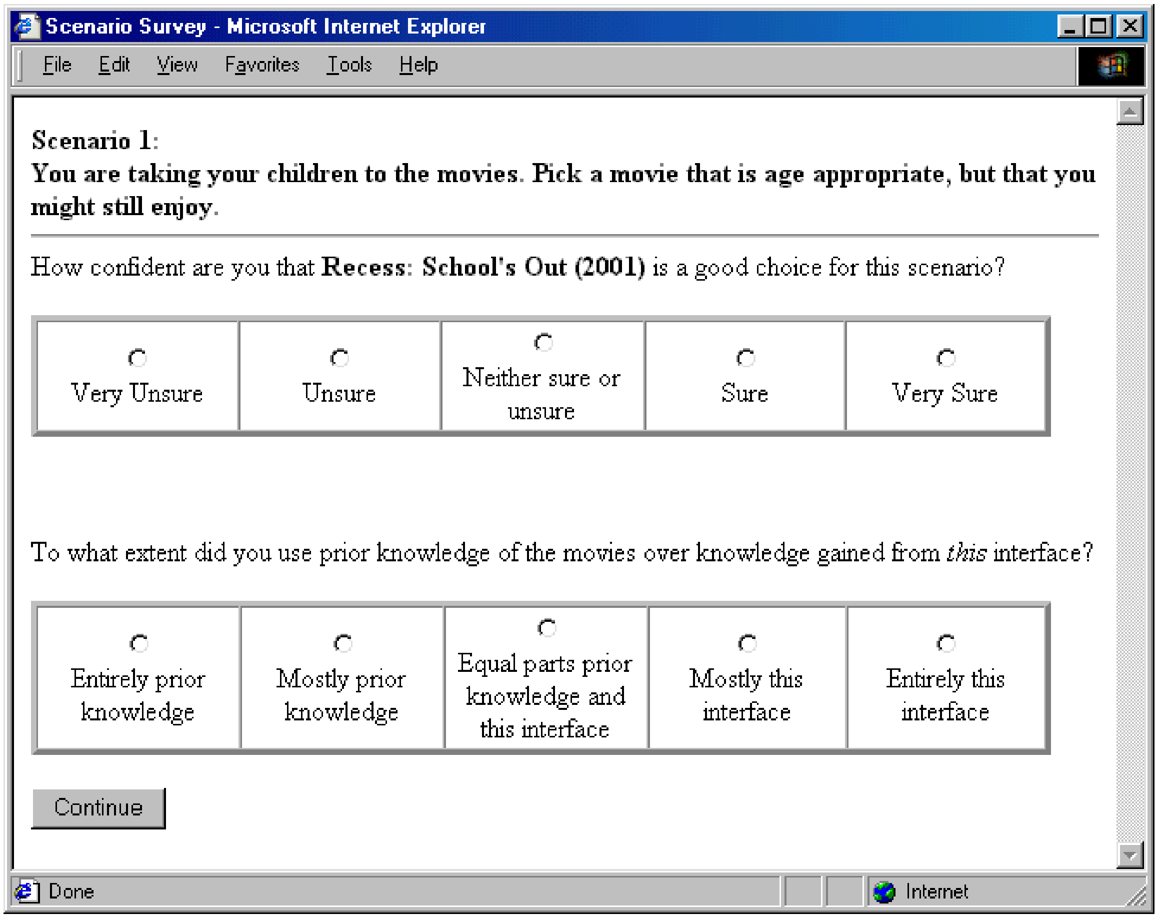


Figure 5.8: Task level survey

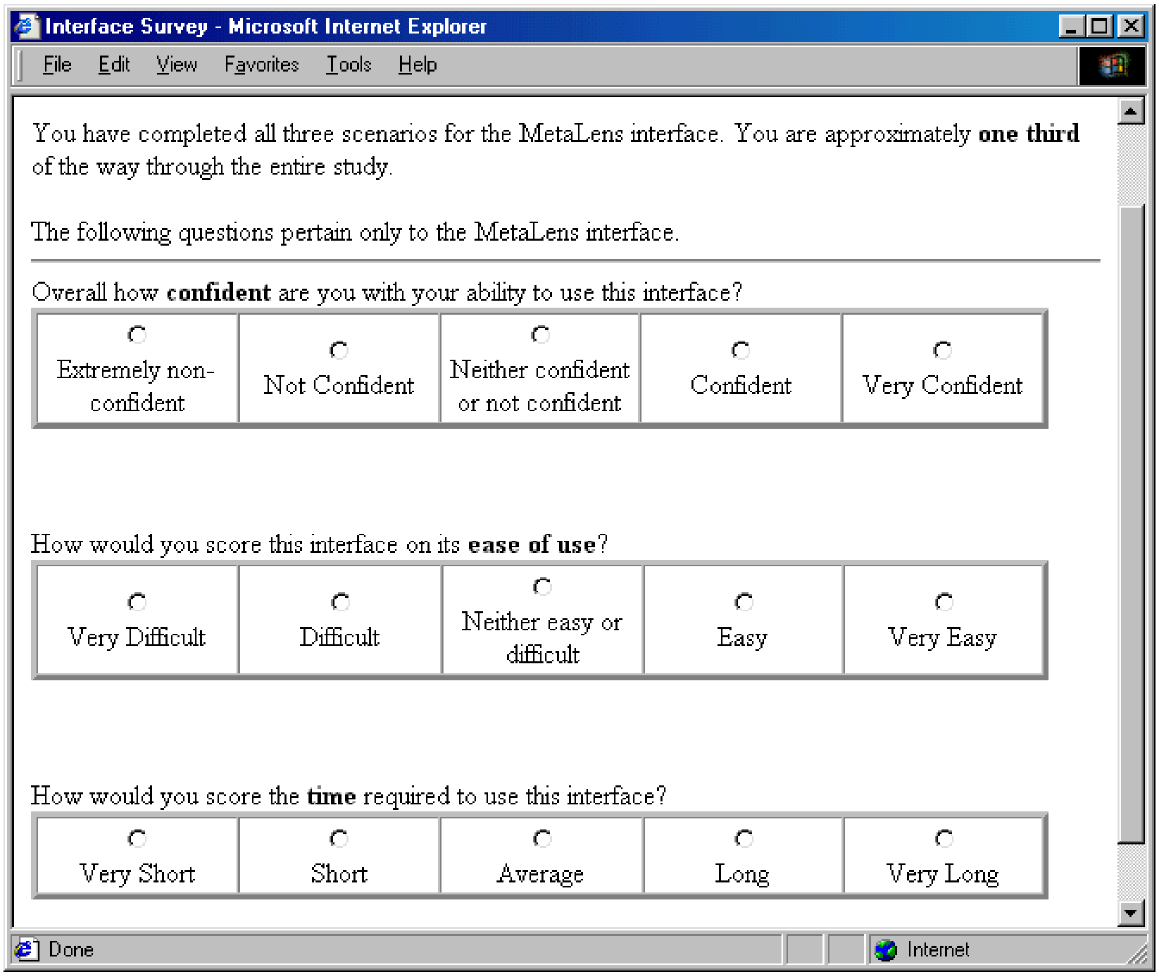


Figure 5.9: Interface level survey

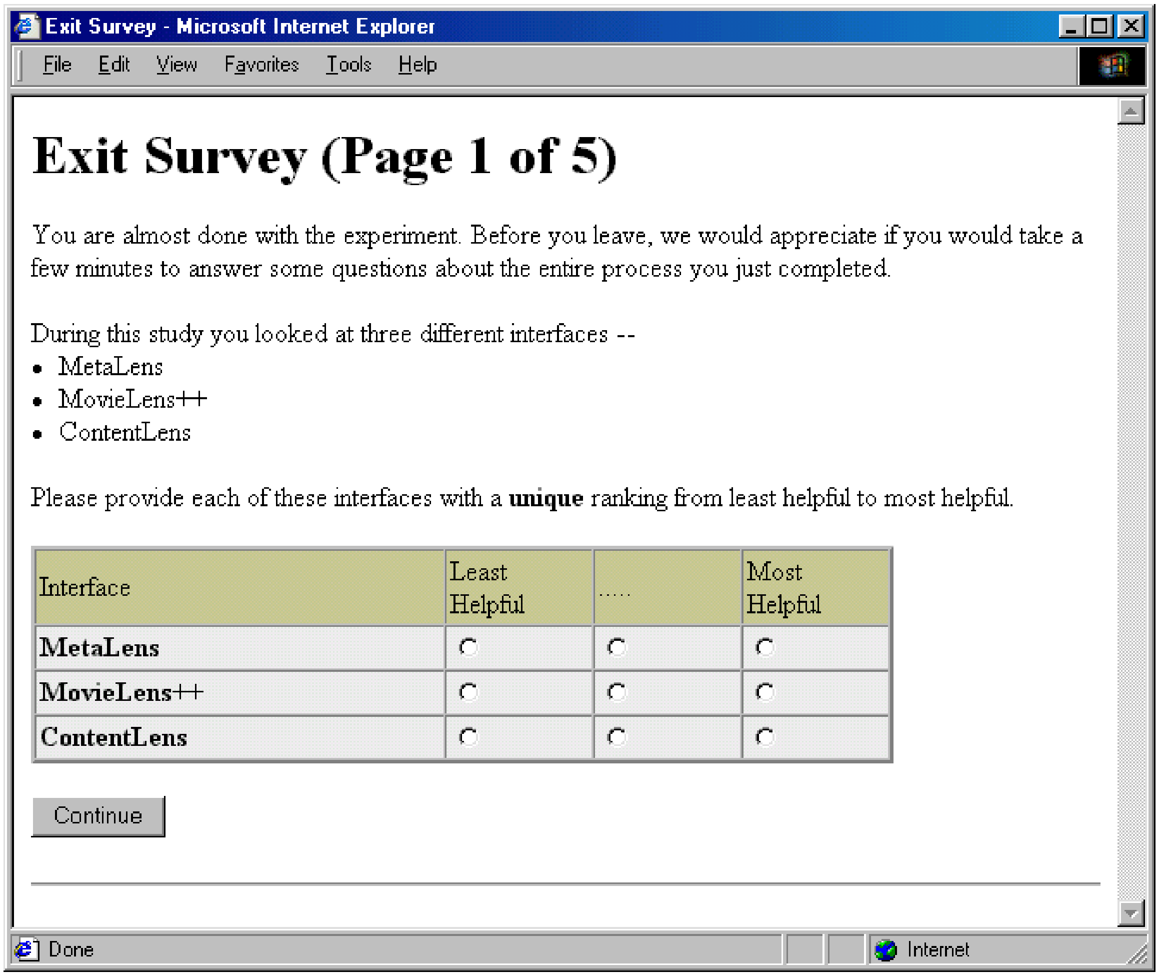


Figure 5.10: Exit survey page 1

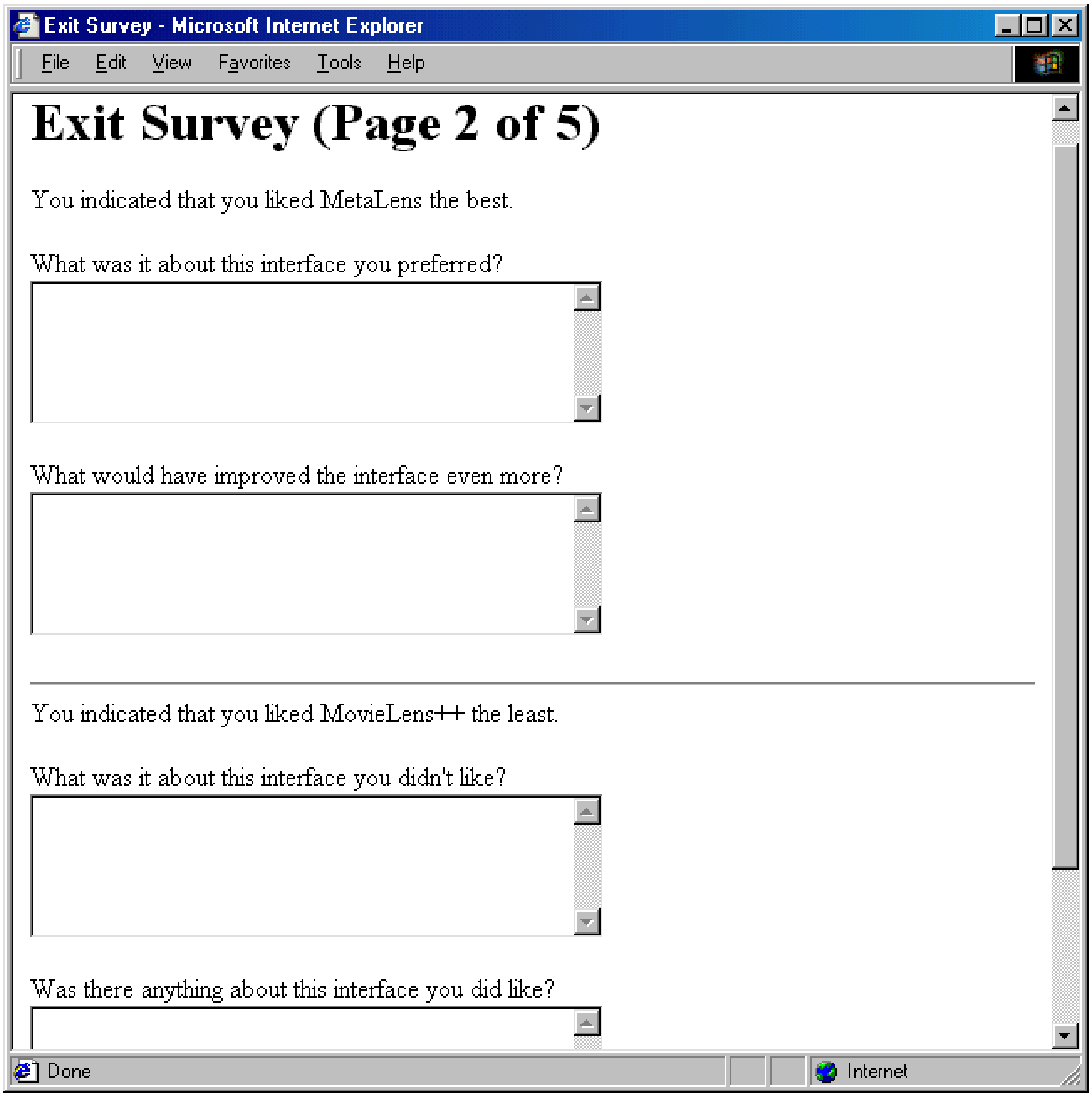


Figure 5.11: Exit survey page 2

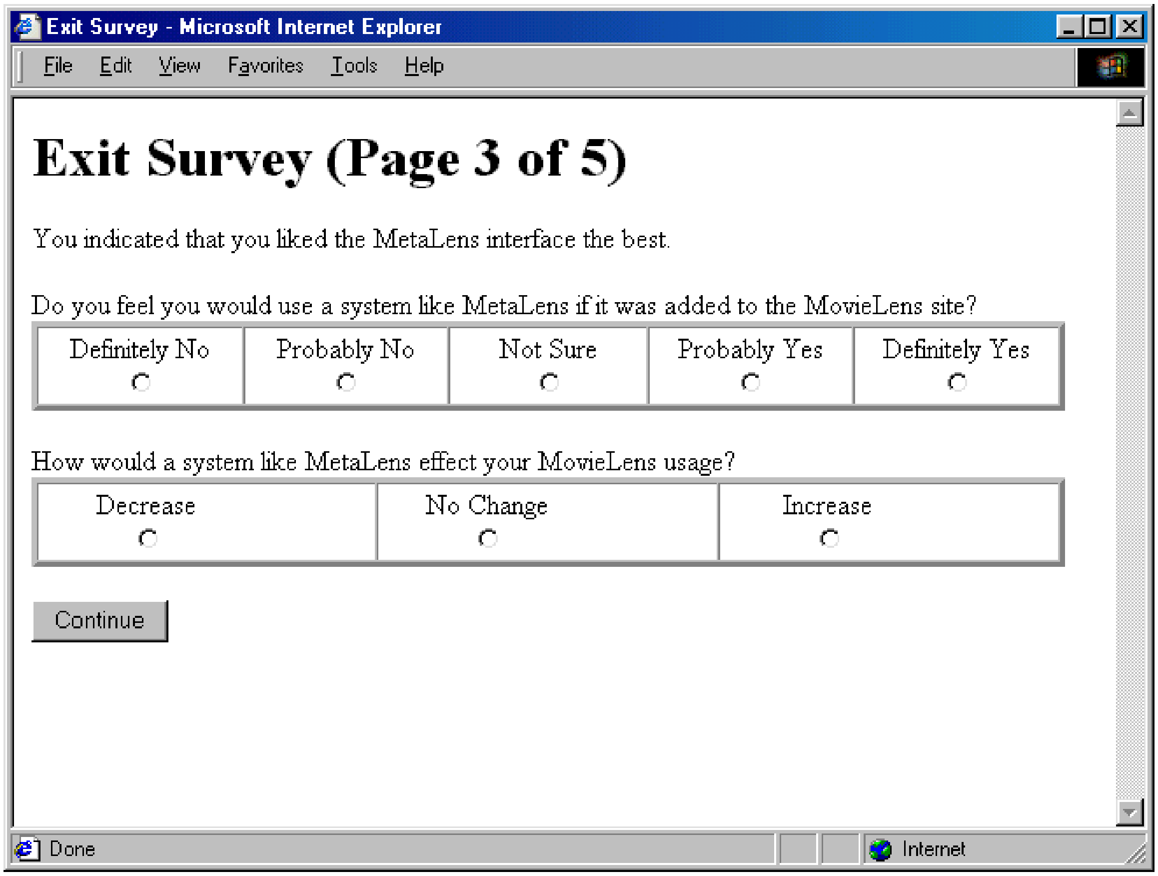


Figure 5.12: Exit survey page 3

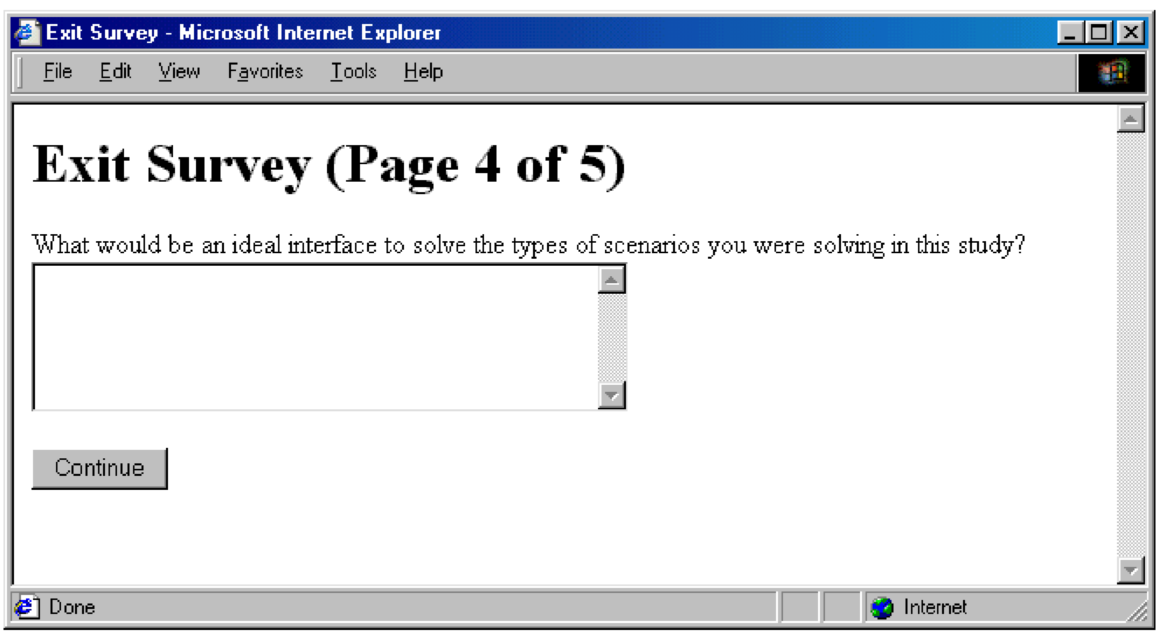


Figure 5.13: Exit survey page 4

Subjects use the MetaLens preference screen (Similar to Figure 4.6) to indicate their requirements for the given scenario. Preferences are gathered for the sixteen data points used in Experiment Two. Upon submission of their preferences, subjects are presented with a "What Information" screen (Figure 5.5) that allows them to request a subset of these sixteen data points to be displayed with the recommendations (the Custom format from Chapter 4). Submission of this information generates a third screen listing all of the items from their recommendation list (Similar to Figure 4.7). Recall that MetaLens provided only "top-10" results in previous experiments. Since MetaLens is being compared to more traditional systems in this experiment – systems which provide users full access to theater and movie listings – we felt it was important to provide subjects with complete access in MetaLens as well. Because of that, this implementation provides the user with the top rated movie triple for every movie in their neighborhood as opposed to the Top-10 as in Experiments One and Two.

Subjects are able to ask for a separate, "additional information" screen for any of the recommended movies or theaters by clicking on the item in question. From the recommendation screen, subjects may return to either the preferences or "what information" screens, reconfigure, and resubmit their preferences. To finish the task, subjects select a triple they feel satisfies the scenario.

Interface three, referred to as ContentLens (CL), creates a system somewhere between ML++ and MetaLens. ContentLens attempts to provide users with an interface similar to what they would use if a complete information recommender system existed for the domain of movies. It was introduced as a way to provide a non-meta-recommender system that at least partially automated the search process.

ContentLens gives subjects access to information filtering-based recommendations with a direct connection to collaborative filtering-based recommendations from MovieLens. The interaction that users have with ContentLens is nearly identical to the interaction they have with MetaLens. The main exception is that the ContentLens preference screen allows subjects to enter preferences for fifteen data points. These represent the sixteen data points used in Experiment Two less the

MovieLens personalized prediction. Subjects use a "what information" screen to select which items to display in the ContentLens recommendation list of the top movie-triples for all movies in their neighborhood. To provide users with access to MovieLens, the ContentLens recommendation format includes a "show me" (Figure 5.6) link for each movie on the recommendation list. Selecting this button creates a separate window displaying the results of a MovieLens title search for the movie in question (Figure 5.7).

Regardless of the interface under consideration, subjects were asked to complete a task survey between tasks (Figure 5.8). This survey asked them to provide answers to two questions regarding the task they just completed. These consisted of "scaled score" questions concerning their confidence that the movie selected fit the scenario and the level of outside information that they used.

Upon completion of all three tasks with a given interface, subjects were asked to complete an interface level survey (Figure 5.9). This survey asked them to provide answers to three questions regarding the interface with which they just finished working. These consisted of "scaled score" questions concerning their confidence in using the interface, the ease of use of the interface, and the time efficiency of the interface.

Upon completion of all nine tasks with each of the three interfaces, subjects were asked to complete an exit survey (Figure 5.10-5.13). This consisted of several screens where they provided a unique rank for each of the three interfaces (from least to most helpful), free form comments on what they liked or disliked about their top and bottom choices, and scaled scores regarding how the implementation of their preferred interface in MovieLens would affect their MovieLens usage.

In addition to the explicit factors requested of subjects, logging was built into the system to track which sub-interfaces and features they chose to use and the time required to complete each task.

5.2.3 Metrics

Results for this study are based on the comparison of a variety of measured quantities and user provided scores. Users' scores were compared using a pairwise T-test. Mean differences with p-values greater than 0.05 are not considered statistically significant and are not discussed in the following sections.

5.2.4 Results

Of the 125 users invited to participate in Experiment Three, sixty consented and completed the experiment. These subjects were evenly distributed among the six experimental groups presented in Table 5.1.

Subjects completed the experiment by participating in an exit survey which included providing a unique ranking of the three interfaces from least helpful (0) to most helpful (2). Table 5.3 shows that test subjects found MovieLens++ the least helpful and MetaLens the most helpful. In fact, exactly half of the sixty subjects provided this relative ordering.

These results give initial confirmation to our hypothesis that users will find MetaLens the most helpful interface. Not only was MetaLens the most helpful (57% of subjects), but subjects indicated that they do not care for the "traditional" way of solving these systems (ML++). A nearly equal number of subjects (63%) reported ML++ as the least helpful interface. Note that the mean average ranking for ContentLens is exactly 1.0, which corresponds with the number assigned to the middle-of-the-road interface. This means that, for those users who did not rank CL as the middle interface, equal numbers found it most helpful versus least helpful (14 subjects in each category).

	Ranking	**Int. Level Confidence**	**Int. Level Ease of Use**	**Int. Level Time score**	**Avg. time per task**
MovieLens++	0.57 (0.81)	3.87 (1.10)	3.03 (1.25)	3.07 (1.010)	168 (310)
ContentLens	1.00 (0.69)	4.20 (0.68)	3.73 (0.86)	3.10 (0.88)	225 (126)
MetaLens	1.43 (0.72)	4.32 (0.70)	3.78 (0.64)	3.00 (0.90)	248 (200)

Table 5.3: Experiment Three interface rankings and interface level survey results. [Mean (Std. Dev.)]

As with earlier experiments, rankings were gathered upon completion of the experiment and required subjects to be able to recall the interfaces with enough clarity to provide meaningful rankings of the interfaces. While the conclusions about these rankings are valid, scores provided during interface-level surveys – that is, provided by the user upon immediate completion of usage of a given interface – provide additional validity.

As can be seen in Table 5.3, subjects report lower confidence with movies selected while using the ML++ interface. Similarly, users reported a lower “ease of use” score while using ML++. These results support the prior conclusion that MovieLens++ is the least helpful interface. However, user scores neither help nor hurt our conclusion that MetaLens is the most helpful interface.

Next, recall that subjects completed three tasks with each interface. The average number of seconds required to complete the 180 tasks presented with each of the three interfaces is contained in the final column of Table 5.3. Users required less time to complete tasks while using the MovieLens++ interface. This is counter to our initial logic. That is, we supported our hypothesis by claiming that meta-recommenders like MetaLens were more efficient than traditional systems such as ML++ by automatically combining multiple data sources. We proposed that users would dislike ML++ because of the time required to make the manual combination. Perhaps users are overly simplifying the decision-making process with ML++. Rather than taking the time to analyze the multiple data sources, subjects considered only a subset of the data and made a decision based largely on prior knowledge.

There is support for this explanation. Recall that for each scenario completed users answered a task level survey. One of the two questions in this survey asked subjects to provide a score indicating how much their decision was based on interaction with the interface. Analysis of these scores shows that users reported a lower score for interface interaction when solving tasks with the ML++ interface (Table 5.4).

	Task Level Confidence	**Task Level Interface Interaction**
MovieLens++	4.00	2.79
ContentLens	4.07	3.06
MetaLens	4.12	3.26

Table 5.4: Experiment Three task level survey results.

This hypothesis was also based on the assumption that subjects would interact with *both* local movie listings and the MovieLens interface to compile the data from these multiple data sources. In fact, an analysis of which features subjects used with the ML++ interface shows that just 34% of tasks consulted both data sources (Table 5.5). This suggests that rather than take the time to gather all of the information to which they had access, subjects chose to browse the local listings and make a decision based on instinct rather than recommendations. Evidence suggests that this process is relatively quick but causes subjects to make decisions for which they are less confident (Table 5.2).

Interaction	**Tasks (from 180)**
At least one interaction with local listings[14]	154
Requested entire listings	145
Title search in local listing	61
Clicked for more info on movie	85
Clicked for more info on theater	54
At least one interaction with MovieLens	71
MovieLens Title Search	20
MovieLens Genre/Date Search	60
At least one interaction with BOTH	66

Table 5.5: Experiment Three MovieLens++ interface interaction.

Lastly, note that when asked to consider the amount of time spent with a given interface, subjects reported identical scores (Table 5.3). Why is it that subjects did not detect that ML++ took less time to use? Perhaps subjects knew that they were cutting corners by making decisions based on only a portion of the information. In an effort to account for this cheating they may be increasing their time score.

5.3 Summary

Confidence and ease of use scores indicate that subjects preferred MetaLens to more traditional methods such as combining MovieLens predictions with readily available movie content. While these scores cannot separate MetaLens and ContentLens, rankings of the systems provided by the subjects show that 57% of test subjects found MetaLens the most helpful interface. These results readily confirm our hypothesis that users recognized the need for meta-recommenders. The impact of this is tremendous. Nearly every measurement taken in this study indicates that subjects found MovieLens++ less helpful. Then why is it that so many of the current implementations stick with this

[14] The fact that for 26 of the 180 tasks, subjects were able to provide movie, theater, and show time information without consulting the local listings was originally thought to be an error in our data collection process. However, upon further examination, this figure was verified. Although subjects were required to enter this information, this data was not checked for its validity. For a portion of these 26, subjects appear to have entered invalid information. For others, it appears that subjects did enter valid information. Presumably, they did so using information learned during earlier tasks, or from prior knowledge they brought to the experiment as a whole.

insufficient design? It confirms our beliefs that there is a large gap between current recommender system design and the actual needs of users, and that meta-recommender systems will go a long way towards filling this gap.

Chapter 6: A Meta-recommender in the Wild

In previous chapters, we have discussed results of experiments designed to consider the interface features users find beneficial in a meta-recommender. In this chapter, we consider how these results were put to use in the design of a publicly available meta-recommender built within the MetaLens Recommendation Framework. Furthermore, we report on usage patterns observed during the first eight weeks of the public deployment of this system.

This chapter is organized as follows. First, we describe how the results from previous chapters were used in designing a public version of the MetaLens meta-recommender. Second, we propose several hypotheses concerned with a variety of factors present in Research Challenge 3. Third, we present the results of data analysis concerning usage of the MetaLens system.

6.1 Design Decisions

On February 22, 2001, a public version of MetaLens became available to all users of the MovieLens web site. While this version is similar to the test versions described in previous chapters, several important design decisions were made based on user comments and the results of previous experiments.

Users are invited to use MetaLens through a link on the MovieLens homepage (Figure 6.1). Prior to submitting queries to MetaLens, users are required to provide initial set up data. This data consists of the ZIP Code for which they wish to receive recommendations, theaters in that ZIP Code that they wish to exclude from recommendations, and the number of recommendations to display (currently limited to "top-10" or "all"). This data serves as an initial level of customization of MetaLens and is separated from query level customization under the assumption that these values will remain relatively constant. Users may access and modify these settings during any future MetaLens session (Figure 6.2) by following the "Your Settings" link available on most MetaLens screens.

Figure 6.1: MetaLens link from MovieLens.

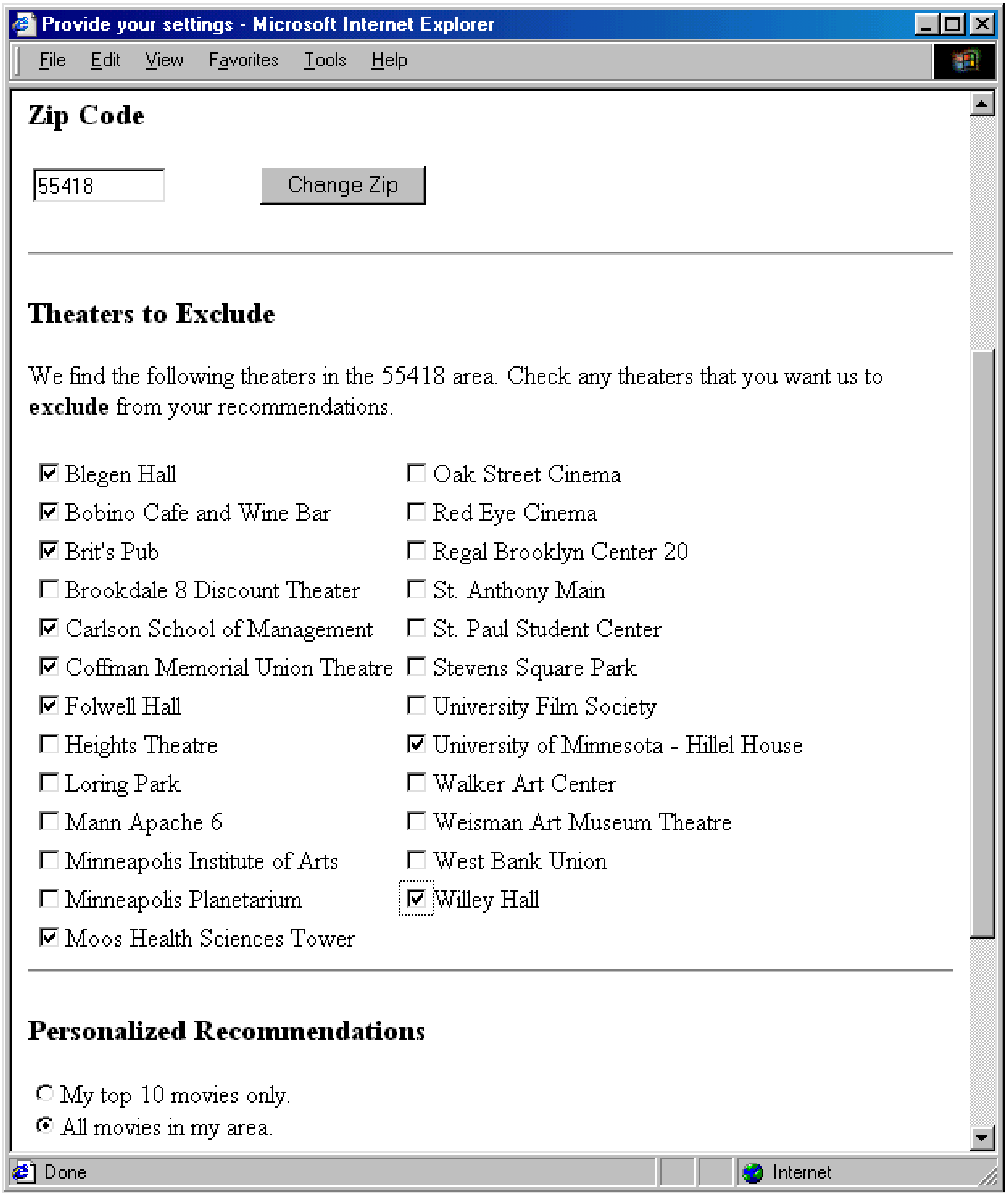


Figure 6.2: MetaLens "My Settings" page.

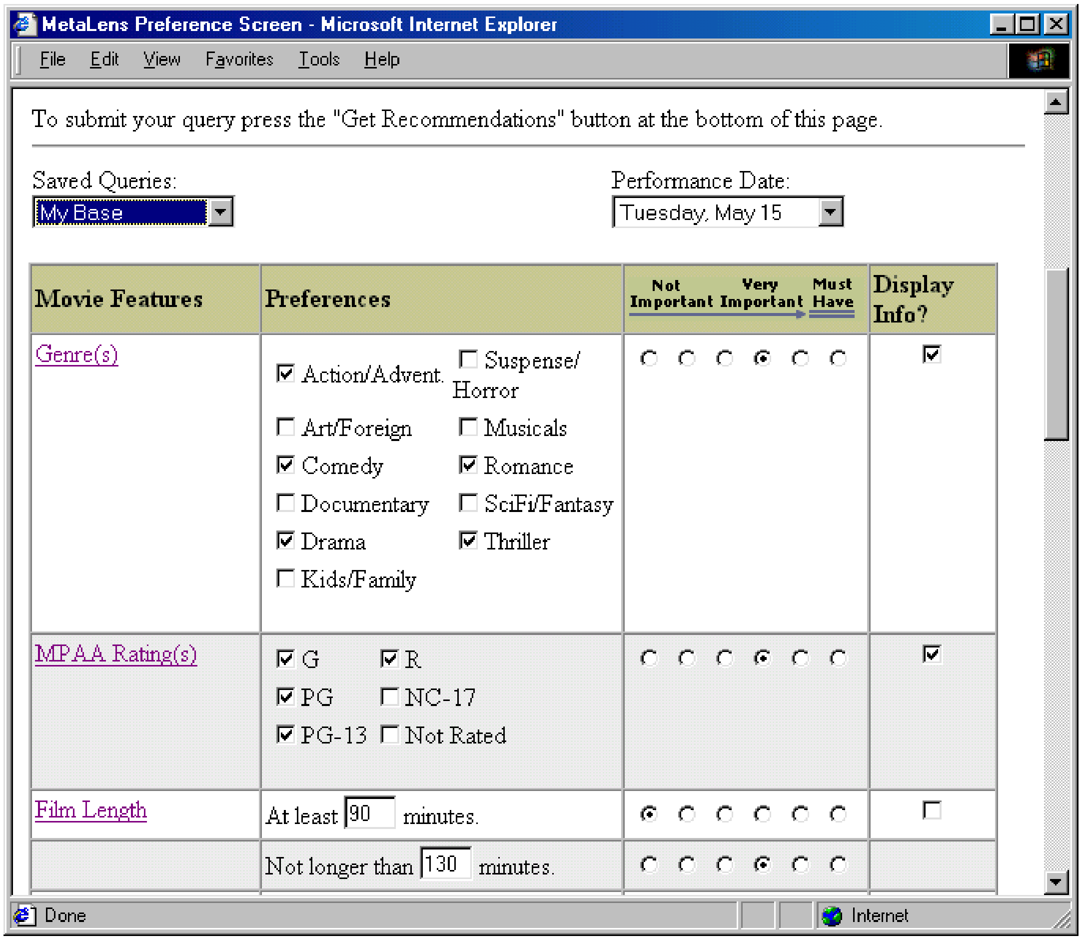


Figure 6.3: Preferences screen for live MetaLens.

Once a user has provided her initial set up values, she is taken to the preferences screen (Figure 6.3). This screen allows users to build the "query" sent to the meta-recommender engine. This query consists of information regarding which movie and theater (sub)features are important to the user (e.g., "I want to see a comedy or a family movie") and weights indicating how important it is that recommended movies match these features (e.g., "The movie I see *must* be one of the genres I selected"). Additionally, the user is provided with a drop down menu to select the date for which she would like to receive these recommendations. This menu provides access to all the days, in the current MetaLens Week (Thursday through Wednesday)[15].

[15] While it would be more useful to consistently provide access to showtimes for the upcoming seven days, data considerations prevent this. Most theaters change their movies on Fridays and operate on a Friday to Thursday week. Unfortunately, show time announcements for the upcoming weekend are rarely made more than a day or two in advance Thus, we are limited by the availability of this data. We elected to

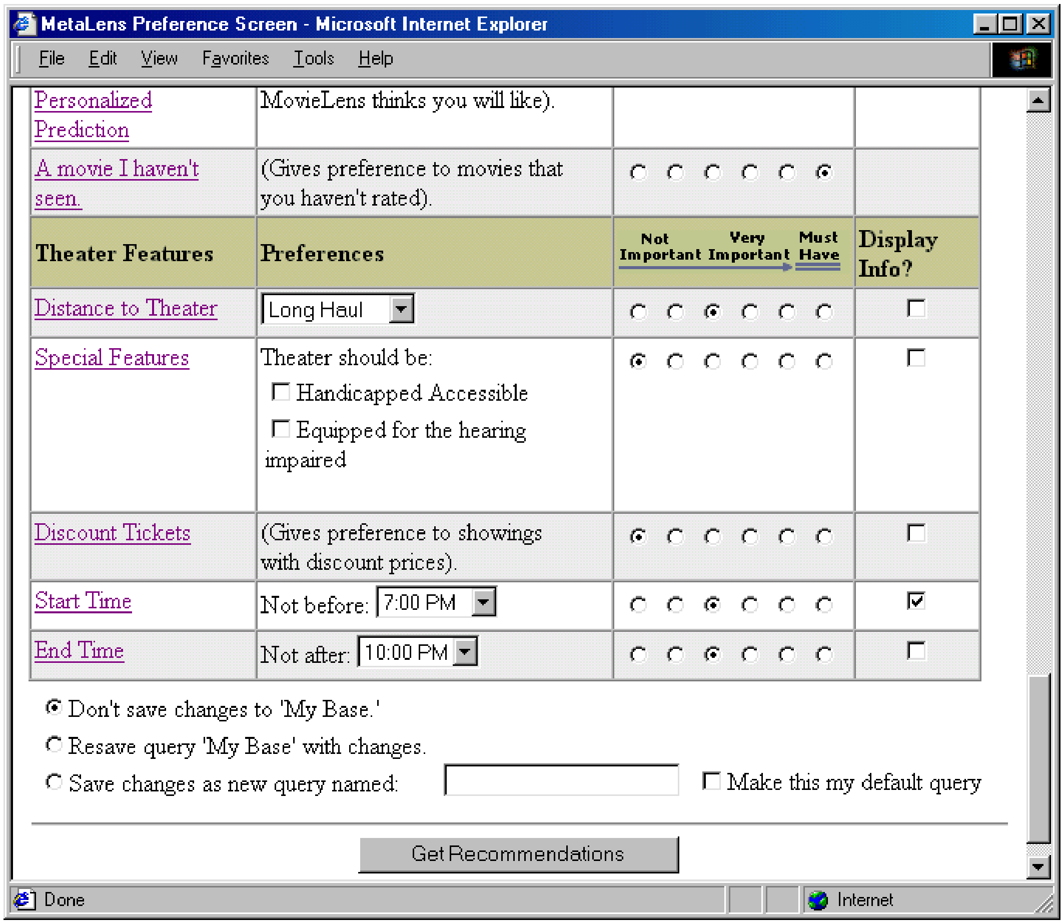


Figure 6.4: Options for creation of saved queries.

Based on results from Experiments One and Two, it was decided to use the Custom format for recommendations. In previous experiments, formatting information was gathered through a separate "what information" screen accessible between the preferences and recommendations screens. In the live version of MetaLens, formatting information is included as part of the preferences screen (Figure 6.3). Where appropriate, the user may select whether the values for each feature should be included on the recommendation screen.

Users have several customization options when submitting a query (Figure 6.4). The default selection for all query submissions is the "Don't Save" option. Selection of this option means that the query is submitted, but any changes to the query involved are not recorded. Alternatively, the user may provide a name for the query and select the

---

offset the MetaLens week slightly from the traditional "theater week" in an effort to provide users with as much advanced information as was realistically possible.

"Save As" option, which causes the query to be stored for future retrieval and use. If the query being submitted is based on one of the user's previously defined and saved queries, an additional option is available. The "Re-save" option updates the stored query based on any changes the user may have made[16]. Finally, once a user has saved at least one query, a checkbox appears which allows the user to indicate that the current query should be set as his default query.

When a user first uses MetaLens, each of the features on the preferences screen is set to the MetaLens Default value. Once a user has saved one or more queries, a second pull down menu appears at the top of the screen. This menu lists each of the queries available to the user – the MetaLens default query and each of his saved queries. The initial value of this menu is the user's default query (If none has been selected, then the MetaLens default is used). By selecting a different query from this menu, the weights, values, and display selections are automatically modified to reflect those recorded with the corresponding query.

The live version of MetaLens allows users to provide information concerning nineteen features of movies or theaters. In addition to the eighteen used in Experiment Three, a "movie I have not seen yet" item has been added (Figure 6.3). This feature allows users to downgrade movies they have already seen. The higher the weight provided by the user, the more the MetaLens score for a movie decreases when the movie has been rated. A "Must" weight causes such movies to be eliminated from the recommendation list.

6.2 Hypotheses

Research Challenge 3, "How do users interact with meta-recommender systems?", can be approached from a variety of angles. We proposed five hypotheses concerning this challenge.

[16] While users are provided the ability to modify and re-save existing queries, they are not given the opportunity to rename existing queries.

Hypothesis 7: Each feature will be considered important by users at one time or another.

Assume that all users have approximately the same requirements in selecting a movie. If this were the case, then there would be no need for meta-recommenders. Designers could simply build a system to look for the exact set of features that all users want. There would be no need to provide an individual user with the ability to customize how or even *what* recommendation data is combined to help her make her selection.

However, results from Experiments One and Two have indicated that users prefer personalized control over the combination of recommendation data. We propose that this is because a user's requirements change from day to day. What is important now may not be important tomorrow. Because of that, we expect that each of the pieces of recommendation data we have made available to users will be considered important at some point to some user.

Hypothesis 8: Certain sub-items *within* a feature category will be more commonly requested than others.

While a user's needs change frequently, there *are* general preferences that are constant. For example, while a user may *need* to see a Family movie tonight because he is taking out his nephew, over time he prefers to see Action films. We expect that these preferences are consistent enough over users as a whole, that we expect to see sub-items within a feature requested more frequently than others.

Hypothesis 9: As users become acclimated to the system, the frequency of tweaks will decrease.

We define a *query tweak* as the difference between two consecutive, same-session queries. For example, if a user submits a query with the weight of the starting time set to 0.5, and then resubmits the query with the weight set to 0.75, we would consider the second query to be a "tweak" and the starting time weight to be an element of that tweak.

One use for tweaks is to find out what effect a certain selection has on the final recommendations. For example, the previously described user may perform this tweak because she notices several highly ranked movies that did not meet her requested starting time. By submitting the tweak, she can get a better feel of whether the system is rating these movies highly because they so perfectly match her other requirements, or because the system is simply disregarding her requested starting time. As such, we would expect tweaks to be more frequent when users are new to the system and are trying to determine how and why the recommender system is making its recommendations.

Hypothesis 10: Given the opportunity, users will use "query profiles."

In Section 6.1 we described the ability for users to create and save queries for later retrieval. These "query profiles" become an important piece of the customization of MetaLens.

Anecdotally, it has been observed that users of systems like MovieLens often want a recommendation from the system with a recurring set of parameters – "I want to take my children to see a movie tonight." While the weights and selections within that request may change slightly, the "base" of the request is the same (a movie rated no higher than PG-13 and over by 9:30). However, we defended Hypothesis 7 by arguing that different users have different preferences. That is, a user without children will rarely need to use a system query regarding children. Query profiles allow users to configure the system to meet their individual preferences and even set up profiles for different daily moods. In doing so, they improve the system in the end by reducing the amount of future effort required to get recommendations.

Hypothesis 11: The profiles of many users will cluster as similar profiles.

We speculate that many users have similar requirements when they try to choose a movie for the evening. For example, a large number of users may have very similar profiles built around the idea that they are looking for a movie to which they can take their children. The detection of clusters of similar queries may help implementers design a new set of "default" queries which are more meaningful starting points for users. These new defaults may also help them design systems that require less effort from the users.

6.3 Results

Before addressing specific hypothesis, it is worth considering **general usage of MetaLens.** During the first eight weeks[17] in which MetaLens was a part of the MovieLens site, 1266 users followed the invitation link to try MetaLens. Of these, 838

[17] February 22 through April 18, 2001.

users "registered" (provided the initial personalization data) and submitted 1668 queries to MetaLens. The majority of these 838 users visited MetaLens only once during the 8-week period (Table 6.1). Furthermore, of the 1668 queries submitted to MetaLens, 148 (8.9%) were exactly the default query.

Sessions	Users
1	682
2	98
3	17
4	13
5	3
6	7
7	5
8	3
9	3
10-19	5
20-29	1
30-39	1

Table 6.1: Number of sessions per user.

Although we are interested in users in general, we are particularly interested in active users. We have defined active users as those who used MetaLens during three or more distinct sessions. Fifty-eight users (7% of all MetaLens users) fit into this category. For perspective, during the three calendar-month period that encompassed this study (February to April) 4724 users visited MovieLens. Of these, 724 visited the site during three or more sessions (15.2%). Of the 1668 total queries submitted to MetaLens, 603 (36.2%) were submitted by active users. Only 34 (5.6%) were exactly the default query, suggesting that active users are more likely than other users to modify the default query.

Hypothesis 7 stated "**Each feature will be considered important by users at one time or another**." Table 6.2 summarizes the weight assigned to each feature in the 1668 queries submitted to MetaLens. Observe that each of the features received the highest weight available in at least one query, and eighteen of the nineteen features received this weight in at least 1% of the queries. However, for eighteen of the nineteen features, the most commonly provided weight was the default set by the system. Furthermore, seventeen of the nineteen features had the *zero* weight as the second most common weight.

A different way of analyzing this is to consider the distribution of the "non-default" weights. That is, when users take the time to modify a feature's weight from its default value, do they tend to consider the feature important (raised from the default) or not important (lowered)? The results of this analysis are contained in Table 6.3. Notice that of the weights that were changed from the default setting, fifteen of nineteen features were more frequently lowered than raised. Across *all* features, a user is twice as likely to downgrade the default weight as she is to raise it.

	0	**0.25**	**0.5 (default)**	**0.75**	**1**	**Must (1)**	**Average Weight**
MovieLens	11	9	513	387	748	NA	0.78
Not Seen [18]	101	17	622	133	273	439	0.70
Genre	227	119	839	201	158	124	0.53
Cream Percent	274	126	827	181	142	118	0.50
Distance	319	175	932	123	79	40	0.43
Critic Percent	368	136	910	152	63	39	0.42
End Time	406	42	1033	76	85	24	0.42
Discount	480	30	932	115	49	62	0.40
Start Time	448	63	987	72	78	20	0.40
Average User	355	278	833	168	34	NA	0.39
MPAA	500	108	852	67	56	85	0.39
Cream Min.	496	94	933	66	55	24	0.37
Release	548	61	960	78	8	13	0.34
MinLength	576	88	880	54	52	18	0.34
Critic Min.	536	101	955	30	30	16	0.34
Content	634	43	923	24	26	18	0.32
MaxLength	664	106	822	30	30	16	0.30
Distributor	782	47	783	5	5	44	0.27
Special Accom.	841	9	816	1	0	1	0.25

Table 6.2: MetaLens distribution of feature weights – all users.

[18] "Not Seen" was added shortly after the original deployment of MetaLens and was not on option for 58 of the 1668 queries.

	Lowered	Raised
MovieLens	1.7%	98.3%
Not Seen	12.3%	87.7%
Genre	41.7%	58.3%
Cream Percent	47.6%	52.4%
Critic Percent	67.1%	32.9%
Distance	66.5%	33.5%
Discount	70.8%	29.2%
End Time	69.3%	30.7%
MPAA	75.0%	25.0%
Start Time	75.8%	24.2%
Average User	74.5%	25.5%
Cream Min.	80.3%	19.7%
MinLength	86.0%	14.0%
Release	84.3%	15.7%
Critic Min.	89.3%	10.7%
Content	90.9%	9.1%
MaxLength	91.0%	9.0%
Distributor	93.9%	6.1%
Special Accom.	99.8%	0.2%
All Features	67.0%	33.0%

Table 6.3: Direction of feature weight changes when modified – all users.

We get rather different results when we repeat this analysis for active users (Tables 6.4 and 6.5). While a comparable seventeen of the nineteen features received the highest weight available in at least one query, only fifteen received this weight in at least 1% of the queries. Furthermore, fewer than half (eight of nineteen) of the features have the default weight as the most commonly provided value. In fact, nine of the nineteen features had the *zero* weight as the most common weight. Finally, observe that of the weights that were changed from the default setting, sixteen of nineteen features were more frequently lowered than raised. Across *all* features, an active user is nearly three times as likely to downgrade the default weight as she is to raise it. Regardless of which set of data we consider, however, Hypothesis 7 is validated. That is, each feature was considered important by some user.

	0	0.25	0.5 (default)	0.75	1	Must (1)	Average Weight
MovieLens	4	1	104	75	419	NA	0.87
Not Seen [19]	61	2	152	11	114	235	0.75
Cream Percent	127	67	201	69	56	83	0.51
Genre	154	85	225	34	36	69	0.44
End Time	205	15	292	40	49	2	0.38
Critic Percent	199	52	267	56	14	15	0.36
MPAA	243	46	237	3	15	59	0.34
Discount	248	3	278	37	5	32	0.34
Start Time	244	26	251	38	44	0	0.34
Average User	181	135	202	66	19	NA	0.34
Distance	193	115	251	21	17	6	0.32
Cream Min.	292	31	209	24	41	6	0.29
Release	289	19	256	31	1	7	0.27
Distributor	310	17	234	0	0	42	0.27
Critic Min.	299	38	242	0	21	3	0.26
MinLength	311	26	249	1	10	6	0.25
Content	328	13	254	0	2	6	0.23
MaxLength	326	39	219	6	2	11	0.23
Special Accom.	397	1	205	0	0	0	0.17

Table 6.4: MetaLens distribution of feature weights – active users.

	Lowered	Raised
MovieLens	1.0%	99.0%
Not Seen	14.9%	85.1%
CreamPercent	48.3%	51.7%
Genre	63.2%	36.8%
End	70.7%	29.3%
CriticPercent	74.7%	25.3%
Start	76.7%	23.3%
Discount	77.2%	22.8%
Average User	78.8%	21.2%
MPAA	79.0%	21.0%
CreamMin	82.0%	18.0%
Dist	87.5%	12.5%
Distrib	88.6%	11.4%
Release	88.8%	11.2%
CriticMin	93.4%	6.6%
MaxLeng	95.1%	4.9%
MinLeng	95.2%	4.8%
Content	97.7%	2.3%
Special	100.0%	0.0%
All Features	72.4%	27.6%

Table 6.5: Direction of feature weight changes when modified – active users.

[19] "Not Seen" was added shortly after the original deployment of MetaLens and was not on option for 28 of the 603 queries.

Yet another way of analyzing "importance" is to consider which features users select to display with their recommendations. That is, to consider which features users selected when they customized their recommendations. The results of this analysis are in Table 6.6. This interpretation of importance also validates Hypothesis 7. That is, each feature is considered important enough by one or more users that they want to see information about it along with the recommendations.

	All Users	Active Users
AverageUser	31.1%	44.9%
Content	17.8%	26.2%
CreamMin	13.7%	17.9%
CreamPercent	29.0%	43.3%
CriticMin	13.4%	18.9%
CriticPercent	25.2%	40.3%
Discount	13.7%	20.4%
Dist	17.4%	20.6%
Distrib	9.8%	17.1%
End	13.2%	18.4%
Genre	43.1%	51.7%
MinLength	33.2%	44.8%
MovieLens	51.1%	73.8%
MPAA	34.1%	50.2%
Release	13.5%	16.3%
Special	1.5%	0.7%
Start	20.0%	30.5%

Table 6.6: Feature inclusion in the recommendation table.

Hypothesis 8 posed, "**Certain sub-items *within* a feature category will be more commonly requested than others**." The percentage of queries containing each of the sub-items for the five selection-based features is recorded in Table 6.7. Statistical analysis of the frequencies with which the sub-items within a content feature are selected indicates that this hypothesis is correct. For example, a comedy is certainly more commonly requested than a family movie. Paired-sample T-test analysis with $p \leq 0.05$ indicates that the most frequently requested genres are comedy and drama, followed by action, science fiction/thriller, romance/foreign, horror, documentary, musical, and lastly, family. A similar ranking exists for active users. Furthermore, comparable rankings exist for each of the remaining content features for both active and all users.

There are also distinct differences observed between active users and "non-active" users. For example, active users are more likely to be interested in movies distributed by Disney, Paramount, Sony, Touchstone, Universal Studios, Warner Brothers, or those in the "Others" category than those users not classified as active[20]. This particular trend can somewhat be predicted. The default MetaLens query has none of the distributors selected. If we assume that active users are more likely to change things from the default settings (an assumption confirmed later in this section) then we would assume that active users would be more likely to select one or more distributors. This would be reflected as a higher interest in distributors.

Sub-item	All Users	Active Users	Sub-item	All Users	Active Users
Action	93.3%	96.2%	Crude	7.1%	6.3%
Comedy	96.5%	97.5%	Drug	1.0%	0.2%
Documentary	80.0%	85.7%	Language	0.5%	0.2%
Drama	96.3%	97.2%	Nudity	1.6%	1.3%
Family	62.7%	66.8%	Sensuality	0.7%	0.2%
Foreign	86.0%	87.4%	Sex	1.6%	1.3%
Horror	83.9%	90.2%	Violence	2.9%	0.8%
Musical	72.6%	80.8%	Disney	3.8%	6.6%
Romance	87.5%	85.7%	Dreamworks	5.5%	6.8%
Science	90.9%	94.4%	Fox	5.5%	6.8%
Thriller	91.1%	93.9%	New Line	5.6%	6.8%
G	83.0%	90.2%	Other	6.2%	9.5%
PG	95.5%	98.0%	Paramount	5.2%	6.8%
PG-13	98.5%	99.2%	Sony	5.9%	8.6%
R	98.0%	98.5%	Touchstone	5.2%	6.8%
NC-17	90.6%	94.9%	Universal	5.0%	6.6%
NR	93.3%	95.4%	Warner	5.1%	6.8%
			Handi	0.4%	0.0%
			Hear	0.3%	0.2%

Table 6.7: Sub-item inclusion in queries.

However, this same logic fails when considering genre, MPAA rating, and objectionable content. When a difference exists between active users and non-active users for items in these content features, each of the differences is the opposite of what we would expect. For example, if active users are more likely to change settings from the default, then we would expect them to be more likely to *deselect* different genres or ratings (which are selected by default). However, in each case, these items are more frequently selected by the active users.

Hypothesis 9 suggested that "**as users become acclimated to the system, the frequency of tweaks will decrease.** " Of the 1668 queries submitted to MetaLens during

[20] This group is not directly reflected in Table 6.7.

the initial eight weeks of public use, 490 of these were tweaks. Recall that a session can have multiple queries. However, only sessions with two or more queries have a tweak. By definition, tweaks are all queries beyond the first query in a session. We have limited our analysis to tweaks submitted during the first nine sessions of each active user. The numbers of queries and tweaks submitted in sessions numbered beyond this are too few to be statistically significant. Table 6.8 lists the total number of queries submitted by active users in each of their first nine sessions as well as the percentage of these queries that were tweaks.

First, observe that the tweaks are decreasing with each session number. This is not completely surprising, however, since so are the total numbers of queries. Note that, in general, the percent of all queries that are tweaks is decreasing as time goes on. Thus, this data supports our hypothesis.

Session Number	Number of Sessions	Sessions w/ tweaks	Total Queries	Queries that are tweaks
1	58	31 (53.4%)	145	87 (60.0%)
2	58	26 (44.8%)	123	65 (52.8%)
3	58	18 (31.0%)	92	34 (37.0%)
4	36	5 (13.9%)	46	10 (21.7%)
5	27	5 (18.5%)	34	7 (20.6%)
6	23	4 (17.4%)	32	9 (28.1%)
7	15	3 (20.0%)	20	5 (25.0%)
8	11	2 (18.2%)	13	2 (15.4%)
9	8	1 (12.5%)	10	2 (20.0%)

Table 6.8: Query tweaks by session number.

Although not related to the original hypotheses, it is interesting to consider **positive vs. negative tweaks**. Each of these elements takes one of two forms. For example, one form of a positive element is a selection item (such as “Comedy” or “show the start time with my recommendations”) that did not exist in query one that was added to query two. The other form of a positive element is the increasing of a scalar item (increasing the weight for how important it is that the movie be a discounted showing, or adjusting the end time from 11 PM to midnight). Conversely, a negative item is either a selection item that is removed between queries, or the decreasing of a scalar item.

Item	Negative Tweak	Positive Tweak	Total
Distance Weight	46	34	80
Not Seen Weight	26	48	74
Genre Weight	30	37	67
MovieLens Display	17	49	66
Distance Value	29	34	63
MovieLens Weight	21	41	62
Release Value	22	39	61
Average User Weight	39	21	60
Cream Percent Weight	28	31	59
Critic Percent Weight	29	25	54
MPAA Weight	25	27	52
Average User Display	17	34	51
Start Time Value	28	23	51

Table 6.9: Most common tweak elements.

There were 2056 elements in the 490 tweaks performed by all users. These were divided almost equally between positive elements (1010) and negative elements (1046). Of the 82 items that could become an element in a tweak, those that are elements in at least 10% of the tweaks are displayed in Table 6.8. Notice that eight of the thirteen elements in the list represent changes in weight. This suggests that users observe that changes to weight have the largest potential impact on changes to MetaLens scores. Users leverage this characteristic of the recommendation algorithm to explore what impact certain changes will have (the rationale behind submitting tweaks). Furthermore, note that of the five popular tweak elements that are not changes in weight, three of these – distance value, MovieLens display, and Average User display – are also represented by popular weight tweaks. For example, the most common tweak element is a change to the distance weight. However, the fifth most common tweak element represents a change to the distance *value*. This dual appearance suggests that users found changes to these three values to be particularly important in making their decisions.

Finally, we consider which elements had the largest discrepancy between positive and negative tweaks. For example, changes in the weight assigned to the percentage of top critics who liked a movie (Cream Percent Weight in Table 6.8) are divided nearly equal between positive and negative tweaks. However, the "MovieLens Display" item (indicating that the user would like to have her MovieLens predictions displayed in the recommendation table) received positive tweaks (adding it to the query) nearly three times as often as it received negative tweaks. This suggests that users are particularly interested in MovieLens' prediction for the movies in question. They realize that the results of the current query would be more helpful if the MovieLens predictions are included with the recommendations. Table 6.10 lists the tweak elements with the largest discrepancy between positive and negative tweaks.

Item	Negative Tweaks	Positive Tweaks	Difference
MovieLens Display	17	49	+32
Not Seen Weight	26	48	+22
MovieLens Weight	21	41	+20
Average User Weight	39	21	-18
Discount Weight	29	11	-18
Release Value	22	39	+17
Average User Display	17	34	+17
End Time Value	13	29	+16
Start Time Weight	26	12	-14
Special Accom. Weight	27	14	-13
"Family"	20	7	-13
Distance Weight	46	34	-12
Content Weight	26	14	-12
Release Weight	28	17	-11
Cream Percent Display	10	20	+10
"Romance"	15	5	-10

Table 6.10: Tweak elements with the largest discrepancy between positive and negative tweaks.

Note the features that have more than one item included on this list. For example, both of the MovieLens-based items are heavily positive. We previously speculated that this was because users were much more likely to recognize the value of this piece of information and add it to subsequent queries. The release date makes the list twice but is split between a frequently positive item (Release Value) and a frequently negative item (Release Weight). Despite initial appearances, these actually are in agreement. A positive tweak for release value means that a user is increasing the amount of time during which the movie could have been released. The net impact is that the score for some older movies will rise slightly. Similarly, by decreasing the release weight, we are decreasing the impact of a movie *not* being released within this time interval. The net impact is that the score for some older movies will rise slightly. The one pairing that is slightly puzzling is the appearance of Average User Weight as a frequently negative tweak and Average User Display as a frequently positive tweak. This suggests that users are interested in knowing what random users think about a movie but aren't particularly interested in having that information included in the MetaLens-generated recommendation.

Hypothesis 10 stated that "**Given the opportunity, users will use 'query profiles.**'" Of the 838 users who submitted at least one query to MetaLens, 278 of them established 355 profiles. For future reference, we will refer to these users as "Power Users." The average power user created one profile. Table 6.11 lists the distribution of number of profiles created by power users.

Number of Profiles	Users
1	230
2	32
3	10
4	3
5	2
9	1

Table 6.11: Number of profiles per power user.

On initial inspection, this hypothesis appears to fail; only one third of all users created a request profile. However, as the number of visits by a user increases, so does the likelihood that he will become a power user. Among the 156 users who used MetaLens on two or more sessions, 87 (56%) were power users. Among the 58 active users, 44 (76%) were power users. This percentage is higher than that observed in the general population. Another way of looking at this is to consider the probability that a user has saved a profile. Over all users, a profile was saved during 22% of the sessions. If this distribution is consistent over active users, then we would expect 52.5% of users to have at least one profile when they obtained active user status. However, we observe that 41 of the 58 active users (70.7%) had saved at least one profile by the completion of their third session. Thus, we conclude that active users are more likely to be power users. Similarly, we observe that power users are more likely to be active users. While 16% of power users become active users, this rate is only 7% among users in general.

Finally, Hypothesis 11 predicted that "**the profiles of many users will cluster as similar profiles**." As a starting point for this analysis, let us consider users who began their query from the MetaLens Default query. Of the 82 *possible* changes these users could make, the average user made 18.44 (median = 19). The distribution of the number of changes is displayed in Figure 6.5.

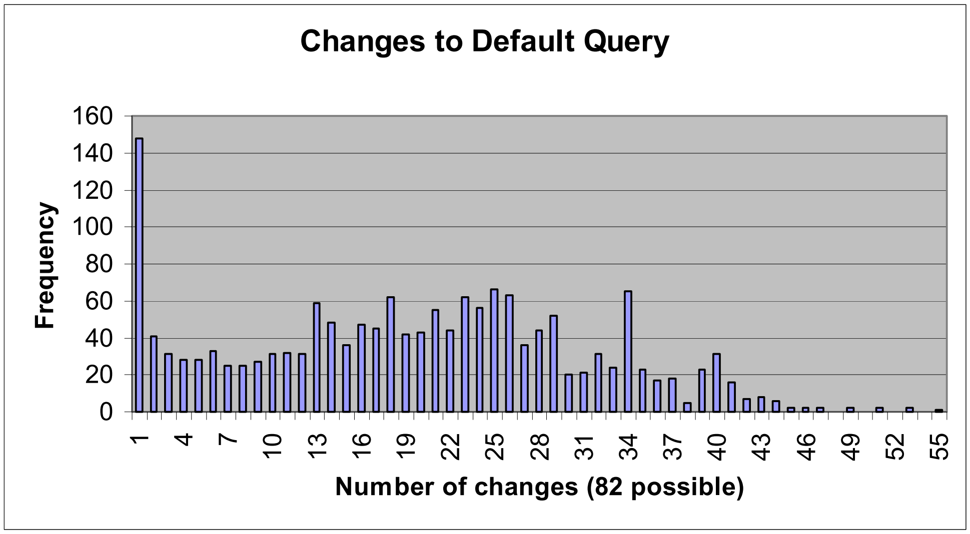


Figure 6.5: Number of changes made to the default query.

If we were to build a *single* new default based on the most common setting for each of these 82 points, it would be:

- MovieLens Weight = 1
- Special Accommodations Weight = 0
- All other weights = 0.5
- Display MovieLens Prediction
- Do not display other features
- Leave the default value of each feature as is.

Had we implemented a Default query with these "most common" values rather than those in the current Default, we would have decreased the average number of changes made by users to 18.26 (Median 18) changes.

In fact, this number is likely skewed higher than we would actually observe if this were implemented. Notice the striking similarity between this new Default and the current Default. It would be highly presumptuous of us to assume that this is because the current Default was so well selected. Instead, we must consider that users are highly likely to accept whatever default settings are put before them, and that the selection of system Default is important. This hypothesis is supported by the fact that nearly 9% of the queries submitted were the default query. The figure of 18.26 average changes is based on the fact that each of these 148 queries has moved from zero changes needed to three changes needed. It is probably a safe assumption that these queries would remain as the default query regardless of the settings present in the default. Taking it to the extreme, if we assume that *every* value that was set at the original default would change to the new default, we would create a situation where the average user would make 16.98 (median 17) changes.

However, the importance behind clusters is not coming up with a new single default but perhaps separating queries into several groups with similar requirements in the hopes of creating a set of new defaults. Rather than clustering all the queries submitted, we decided to limit this analysis to the profiles established by power users. Presumably, users who save a profile have spent a little more time constructing the query and have depended less on the previous Default query. (In fact, the average *profile* differs from the default query by 20.6 changes.) Thus, perhaps these profiles more accurately represent what users *really* want.

To produce clusters we used the K-means Clustering package provided by SPSS. As a starting comparison point, we asked it to produce just a single cluster. Taking the centroid over all power users produces a new default query where:

- MovieLens Weight = 1
- Not Seen Weight = Must (1)
- The weights for Special Accom., Distributor, MaxLength and MinLength = 0
- All other weights = 0.5
- Display MovieLens Prediction and Genre
- Do not display other features
- Release date value is set to “not important.”
- The remaining features are left as is.

The implementation of this as the Default query would create a situation in which the average user would need to make 18.97 changes while the average power user would need to make only 19.8 changes. While this is a minor improvement for power users, it is a slight deterioration for all users.

Recall our prior discussion concerning default settings likely remaining at the default. This being the case, we would estimate that the implementation of the centroid as the new Default query would produce a situation in which the average user would be required to make only 14.77 changes per query (median 15). Thus, cluster analysis may help us produce a Default query which is a real improvement over the current values.

One of the challenges in clustering is deciding when the appropriate number of clusters has been discovered. We used two metrics in making this decision. First, we considered how many changes the average power user would need to make to generate each of his profiles. The starting point in each case is the centroid of the cluster to which the profile is assigned. If the creation of an additional cluster reduces the overall number of changes required by only a small margin (1%), then it may be worth ceasing to increase the number of clusters.

However, a potential problem with clustering is related to the "overfitting" problem observed in Linear Regression models. For example, we could continue to cluster the 355 profiles until we started to approach 355 clusters. At this point nearly every cluster would consist of a single profile, the centroid for that cluster would be the profile itself, and the number of changes required would be zero. While this optimizes the previous metric, it is not a viable solution to the clustering problem. Thus, the second metric we use is to consider the size of the smallest cluster generated. If the size of the smallest cluster is less than 1% of the profiles being clustered, then it may be worth ceasing to increase the number of clusters.

We conducted k-means clustering for an increasing number of clusters. The results are listed in Table 6.12. Based on the thresholds for the two metrics chosen above, it was decided that five was an optimal number of clusters. The centroids for each of these five clusters are presented in Table 6.13. These results suggest that our original hypothesis was valid.

Number of Clusters	Total Changes	Size of the smallest cluster	Profiles per cluster
1	7050	355	355
2	6164	177	178,177
3	5889	52	119, 84, 52
4	5773	15	149, 98, 93, 15
5	5739	8	136, 124, 61, 26, 8
6	5664	2	130, 94, 75, 43, 11, 2
7	5671	2	143, 130, 60, 9, 8, 3, 2
..			
10	5473	1	98, 83, 52, 45, 48, 20, 5, 2, 1, 1

Table 6.12: Results of clustering the 355 query profiles.

Although the validation of this hypothesis is *interesting,* the fact that clusters exist doesn't become *helpful* until we can use these clusters to improve the recommender system. Unfortunately, attempts to extract the meaning of clusters were not productive. It was hoped that by examining the user-defined names of the profiles that cluster we would be able to extract some indication of what users were looking for. Unfortunately, the only pattern we were able to extract is that while a query's name may be helpful to its creator, it gives an outsider little clue as to the user's intentions with the query.

6.4 Summary

Analysis of usage logs indicates that there is a place in the recommendation community for meta-recommenders. Two important results come out of this analysis. First, each piece of recommendation data is considered helpful by some user of the system. Although many of these movie and theater features were considered unimportant by a large segment of the user base, each item was considered a "must" by a different segment. We proposed earlier that meta-recommenders help users evaluate and consolidate a large quantity of recommendation data. As such, it seems relevant to provide access to as much recommendation data as possible. This result suggests that designers must not treat lightly the decision to include or exclude certain pieces of recommendation data. Second, users who customized the system were more likely to be repeat users and repeat users were more likely to have customized the system. We reported earlier that one reason electronic commerce sites implement recommender applications is in an effort to increase consumer loyalty to the site. This being the case, we may conclude that if customization capabilities increase the likelihood of a user being a repeat user, they may also increase the user's loyalty to the site.

	1	2	3	4	5
Number of Profiles	124	136	61	26	8
Content Weight	0	0.5	0.5	0	0
CreamMin Weight	0	0.5	0.5	0	Must
CreamPercent Weight	0	0.5	0.5	1	Must
CriticMin Weight	0	0.5	0.5	0	Must
CriticPercent Weight	0	0.5	0.5	0.5	Must
Discount Weight	0	0.5	0.5	0	0
Distance Weight	0.5	0.5	0.5	0.5	0
Distrib Weight	0	0.5	0.5	0	0
End Weight	0	0.5	0.5	0	Must
Genre Weight	0.5	0.5	0.5	1	Must
MovieLens Weight	1	1	1	1	1
MPAA Weight	0	0.5	0.5	0.5	0
MaxLeng Weight	0	0.5	0.5	0	0
MinLeng Weight	0	0.5	0.5	0	0
Not Seen Weight	Must	0.5	0.5	Must	Must
Release Weight	0	0.5	0.5	0	0
Special Weight	0	0.5	0	0	0
Stars Weight	0	0.5	0.5	0	0.25
Start Weight	0	0.5	0.5	0	0
Genre Values	All	All	All	All except Family and Musical	All except Family and Musical
MPAA Values	All	All	All	All except G	All
Distributor Values	None	None	None	All except Other	None
Start Value	7 PM	7 PM	7 PM	7 PM	4 PM
End Value	11 PM	11 PM	2 AM	11 PM	11 PM
Release Value	14 days	14 days	No Care	No Care	No Care
Content Display					Yes
CreamMin Display					Yes
CreamPercent Display			Yes	Yes	Yes
CriticMin Display					Yes
CriticPercent Display			Yes	Yes	Yes
Discount Display					Yes
Dist Display			Yes	Yes	Yes
Distrib Display					Yes
End Display					Yes
Genre Display	Yes		Yes	Yes	Yes
MLpred Display	Yes		Yes	Yes	Yes
MPAA Display			Yes	Yes	Yes
MinLeng Display			Yes	Yes	Yes
Release Display					Yes
Stars Display			Yes	Yes	Yes
Start Display					Yes

Table 6.12: Cluster centroids – Values which were the default value in all clusters are excluded.

Chapter 7: Meta*: Creating New Recommenders Using the MetaLens Recommendation Framework

Several issues arose while analyzing the results discussed in Chapter 6. Questioning these led to the development of two new recommender systems built within the MetaLens Recommendation Framework. This chapter reports on the development of two new systems – MetaLite and MetaClick – which were built to test further several of our initial hypotheses and assumptions. This chapter is organized as follows. First, we discuss the areas of concern that came out of the analysis of the usage of MetaLens. Second, we present the design and implementation of MetaLite and MetaClick. Third, we report on the results of a user survey in which users provided feedback regarding MetaLens, and input on the design of MetaLite and MetaClick. Fourth, we present the results of usage analysis conducted over the first four weeks that these new systems were made publicly available.

7.1 Background

Several issues arose while examining the results presented in the previous chapter:

- Data logs indicated a poorer return rate than we had expected. Recall that the percentage of active users for MetaLens was roughly half the percentage observed at the MovieLens site.
- While Hypothesis 7 was validated, analysis of the data also indicated that, as a whole, users are interested in less data than we had originally expected. Recall that when users modify the interest weight for a feature from its default weight, they most commonly set this weight to zero (no interest) for seventeen of the nineteen features used in MetaLens (Table 6.2).

7.2 Design Decisions

MetaLite was developed to see if users would be interested in a meta-recommender with access to less information. In deciding which features to include in MetaLite, we originally proposed selecting the top five features. However, which features qualify as the "top five" varies depending on the metric used. If we consider the features provided the highest average weight, we arrive at MovieLens, genre, cream percent, “Not Seen,” and distance. If we consider the features receiving the most “Must” votes, we also get MovieLens, genre, cream percent, and “Not Seen.” However, distance is replaced by MPAA rating. In the end, rather than choose which metric to consider, we chose to implement MetaLite with these six features.

MetaLite uses the interface design used for MetaLens. Users interact with a preference screen, where they can provide weights and values for a set of features, and select which features to include on the recommendation screen. The underlying algorithm is identical to that used by MetaLens. In fact, MetaLite was completely built within the MetaLens framework. Other than a reduction in the number of features that users can incorporate into their queries, MetaLite is identical to MetaLens.

MetaClick was developed to see if users would be more interested in a meta-recommender requiring almost no input from the user. In considering the relatively low return rate experienced with MetaLens, one explanation is that users have a hard time translating their desires into categories of features. Perhaps users would be willing to give up some of the control over the formation of their recommendations in return for a simplified way to indicate their interests.

MetaClick consists of six single line descriptions of scenarios for the type of movie a user might be interested in viewing. Each scenario is connected with a “query profile” containing weights, values, and display information for the nineteen movie and theater values used by the original MetaLens framework. Users simply select which description best fits their mood for the evening (Figure 7.1). The profile corresponding to that description is sent to the MetaLens recommendation engine, and recommendations are returned as though the user had taken the time to configure MetaLens himself.

The query clusters extracted in Chapter 6 would be an ideal way to develop non-stereotyped profiles for MetaClick. However, the inability to extract meaningful information from these clusters regarding what the users were interested in prevented us from developing the necessary one-line descriptions. Instead, descriptions were selected based on feedback provided to exit surveys in Experiments One and Two concerning situations where users go to the movies. Each description’s corresponding query was selected based on admittedly stereotypical assumptions about what would be important to users.

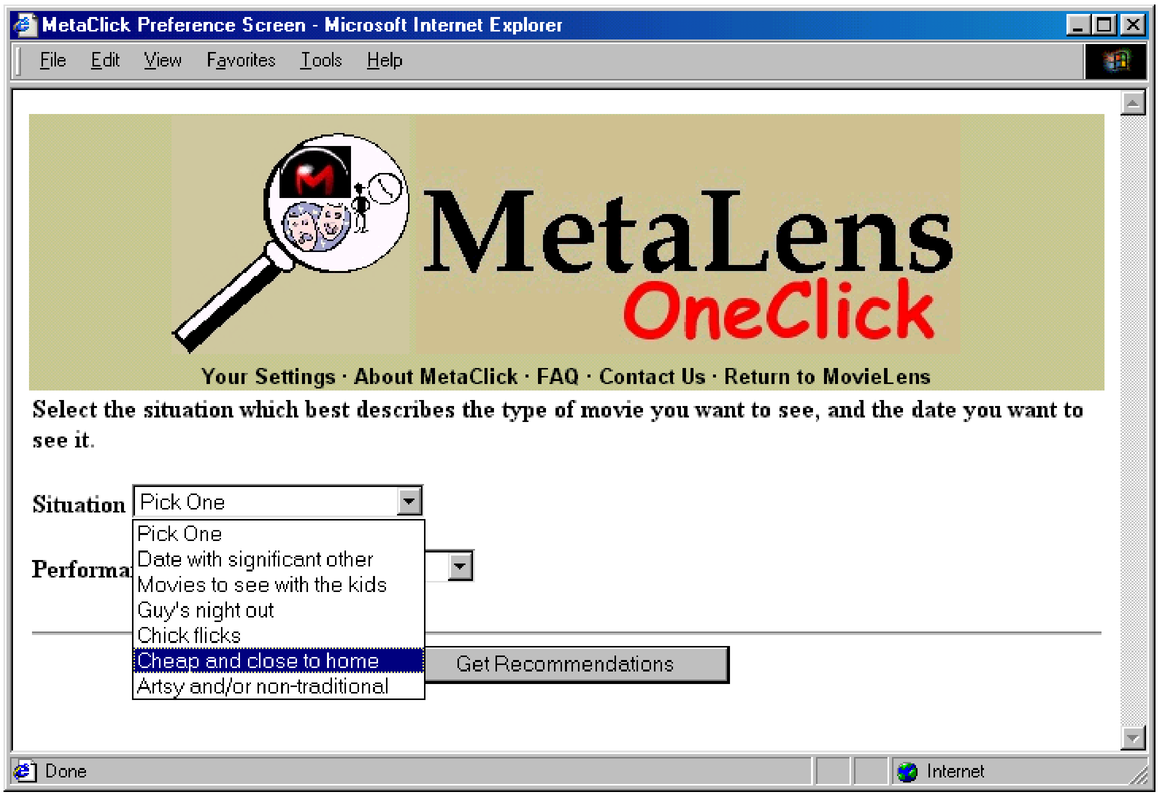


Figure 7.1: MetaClick "preferences" screen

7.3 The Survey

Over 100 "registrants" with MetaLens were sent email invitations to participate in an online survey. The list of users was nearly equally divided between active users and those who had visited the site once or twice. Of those sent invitations, 26 subjects accepted and completed the two-part survey.

In part one, subjects were asked several questions concerning their MetaLens usage. Question one asked subjects to indicate the number of times they had used MetaLens (Table 7.1). Although we originally planned on comparing user-reported usage with known usage, a scripting error left results anonymous.

Question two asked users to indicate what they felt were the strengths of MetaLens. Users were provided with a list of six potential strengths and were given the option to enter additional strengths as well (although none opted to do so). The average user selected 2.6 of the six strengths, with a majority of users indicating they felt MetaLens provided relevant recommendations. A majority felt that the data used in this recommendation process was one of the strengths as well. User responses are summarized in Table 7.2.

Times Used	**Number of respondents**
1	12
2	4
3	2
4	1
5 or more	7

Table 7.1: Number of times respondents reported using MetaLens.

Strength	**Number of respondents**
Data used in recommendation process	15
Relevant recommendations	15
Timely recommendations	12
Easy to use	10
Let's me indicate my mood	8
The ability to save queries	8

Table 7.2: User reported strengths of MetaLens.

Weakness	**Number of respondents**
Too much non-relevant data used	11
Difficult to use	4
Too slow	4
Recommendations not relevant	4
Not enough relevant data used	3
Theater/Distance info not relevant	3

Table 7.3: User reported weaknesses of MetaLens

Question three asked users to indicate what they felt were the weaknesses of MetaLens. Users were provided with a list of six potential weaknesses and were given the option to enter additional items as well (again, none opted to do so). The average user selected one of the six weaknesses. The most common response was that MetaLens uses too much non-relevant data. Furthermore, of the eleven users who provided this response, six also indicated that the data used by MetaLens was a strength. User responses are summarized in Table 7.3.

The fourth question provided subjects with a list of the nineteen features MetaLens uses to make its recommendations. Subjects were also given the option to enter one or more features separately. Subjects were asked to select up to five features that they felt were the most relevant to them. The results are displayed in Table 7.4.

Subjects indicated that the MovieLens prediction was an important feature to them. Recall that the usage statistics discussed in Chapter 6 reported this as well. However, note that several other features that are highly used among all users appear relatively lower on the ranked list of relevant feature. While results to this question could have been used to change the features selected for inclusion in MetaLite, it was decided that the relatively low number of respondents in this list made it less accurate than the analysis based on MetaLens users as a whole.

Feature	**Users selecting for their top 5**
*MovieLens Prediction	22
Average User Rating	15
*Genre	14
*Percent of TOP critics who liked	12
*Distance to Theater	9
Start	9
Percent of ALL critics who liked	8
*Eliminate Movies I have seen/rated	7
Number of TOP critics who liked	5
Release Date	3
Discounted Shows	3
*MPAA	2
End	2
Maximum Length	1
Distributor	1
Zip Code[21]	1
Minimum Length	0
Objectionable Content	0
Number of ALL critics who liked	0
Special Accommodations	0

Table 7.4: User reported top-five features.
(* = Top features based on actual MetaLens).

Part two of this survey provided users with access to the demonstration versions of MetaLite and MetaClick as described in Section 7.1. Users were given a brief description of MetaLite, were allowed to interact with it at will, and then were asked

[21] Entered manually by a subject.

several questions concerning their perceived use of the system. This process was then repeated for MetaClick.

For MetaLite, subjects were first asked to indicate if they would use MetaLite if it were part of MovieLens. Of the subjects responding, 23 of the 26 (88.4)% indicated that they would use MetaLite. Subjects were then asked to estimate their MetaLite usage compared to their MetaLens usage. Results of this comparison are presented in Table 7.5. Finally, users were given the opportunity to comment on MetaLite. Although few uses took advantage of this opportunity, comments were relatively evenly split between those who felt that MetaLite took away what they loved about MetaLens (access to all that data), and those who felt MetaLite improved on MetaLens by simplifying the process.

For MetaClick, subjects were once again asked to indicate if they would use the system. Of the subjects responding, only eighteen of the 26 (69.2%) indicated that they would use MetaClick. Subjects were then asked to estimate their MetaClick usage compared first to MetaLens and then to MetaLite. Results of these comparisons are contained in Table 7.6. Finally, users were given the opportunity to comment on MetaClick. This time, for every user who thought that MetaClick provided easy access to recommendations, two users felt it was based on stereotypes or criteria that didn't match their needs.

	MetaLens	**MetaLite**	**MetaClick**	**The Same**
MetaLens vs. MetaLite	8	13	NA	5
MetaLens vs. MetaClick	16	NA	8	2
MetaLite vs. MetaClick	NA	16	5	5

Table 7.5: User reported system preferences.

In an effort to better analyze the results of the pairwise comparisons summarized in Table 7.6, a ranking of the three systems was generated for each user based on the results of the three comparisons. Each time a system "won" a comparison, it received one point. Systems receiving a "same" score, each received one half a point. Based on this method, the three systems received rankings on a scale of zero (the system I would use the least) to two (the system I would use the most). Average rankings over all test subjects, as well as rankings based on the activity level of subjects are displayed in Table 7.6.

	MetaLens	MetaLite	MetaClick
All Users	1.06 (0.79)	1.31 (0.63)	0.63 (0.73)
Active Users	1.55 (0.64)	1.00 (0.47)	0.45 (0.83)
Non-Active Users	0.75 (0.66)	1.50 (0.73)	0.75 (0.66)

Table 7.6: Projected "rankings" of the three systems. [Mean (Std. Dev.)]

Among all users, MetaLite is ranked higher than MetaClick. This ranking is seen again when considering non-active users. In this case, MetaLite is ranked higher than either of the other two systems. However, when we consider active users, we see very different results. In fact, active users rank MetaLens higher than either of the other two systems. Finally, active users rank MetaLens higher than non-active users do while non-active users rank MetaLite higher than active users do.

We started this chapter by hypothesizing that part of the reason for the low return rate for people who try MetaLens is the fact that MetaLens overwhelms them or provides too much information that is not part of their decision-making process. These results suggest that this may be the case. When presented with a scaled-down version of MetaLens – MetaLite – these non-active users seem more interested in the use of meta-recommenders.

7.4 Usage Analysis

To study alternate implementations of recommenders built within the MetaLens framework, both MetaLite and MetaClick were added to the MovieLens web site. Both were left as originally implemented as described in Sections 7.1 and 7.2. The following section presents a brief analysis of usage patterns observed among users of MetaLite and MetaClick as well as patterns among users who tried combinations of the three systems.

7.4.1 MetaLite

During the first four weeks[22] that MetaLite was part of the MovieLens web site, 63 users followed the MetaLite link from the MovieLens home page. Of these, 36 (57.1%) chose to register if necessary (previously registered users were not required to reregister) and submit a query. These 36 users submitted a total of 62 queries of which twelve (19.4%) were the default query. Two of the 36 users of MetaLite used the system

[22] May 7 through June 4, 2001.

in three or more sessions, earning "active user" status. These two users, equaling 5.6% of all users, submitted a total of fifteen queries or 24.2% of all queries. This information is summarized in Table 7.7.

	Users who followed link	Users who submitted a query	Total Queries Submitted	Active Users (3 or more sessions)	Queries Submitted by active users
MetaLite	63	36 (57.1%)	62	2 (5.6%)	15 (24.2%)
MetaClick	133	86 (64.7%)	173	2 (2.3%)	24 (13.9%)
MetaLens	783	531 (67.8%)	1001	30 (5.6%)	295 (29.5%)

Table 7.7: Four week system usage.

Usage numbers for MetaLite are comparable to usage numbers generated during the first four weeks that MetaLens was part of the MovieLens web site. In fact, there are many similarities between MetaLite and MetaLens. Table 7.8 displays the distribution of feature weights in MetaLite and the average weight assigned. For comparison, the last column of the table displays the average weight assigned each feature during the initial eight week trial of MetaLens as previously reported in Chapter 6. Independent sample T-test analysis indicates that only genre weight has changed between MetaLite and MetaLens, suggesting that the importance of a feature is relatively constant regardless of how many features are included in the decision-making process.

	0	**0.25**	**0.5 (default)**	**0.75**	**1**	**Must (1)**	**Average Weight**	**8 Week MetaLens Avg.**
MovieLens	3	0	18	5	36	NA	0.79	0.78
Not Seen	6	0	26	3	8	19	0.68	0.70
Distance	8	10	36	5	1	2	0.44	0.43
Genre	15	3	35	2	4	3	0.43	0.53
Cream Percent	19	4	28	4	3	4	0.40	0.50
MPAA	23	1	32	2	0	4	0.35	0.39

Table 7.8: MetaLite feature weight distribution (shaded values indicate significant changes).

Table 7.9 displays the percentage of times the values for a feature were selected for inclusion in the recommendation table for both MetaLite and MetaLens. The number of times genre or MPAA rating was selected is less for MetaLite than MetaLens. This result is counter to what we expected to see. Results of Experiment One indicated that users prefer to see all available information when there are a limited number of features used in the decision-making process. That being the case, we would expect that the percentage of times a feature was selected for display in MetaLite would be greater than, or at least equal to, the percentage of times it was selected for display in MetaLens. However, the percentage of queries containing genre and MPAA Rating display requests is lower with MetaLite than with MetaLens. One explanation is that a significant number of users of MetaLite have enough experience with MetaLens to have developed a prior opinion about the added value provided by this information. As a result, they request this information at a lower rate than when they were first learning how to use MetaLens. An alternate explanation is that MetaLite users figured the system would accurately use their genre and MPAA requirements, and therefore they didn't need to see them.

	MetaLite	**MetaLens**
Cream Percent	27.4%	29.0%
Distance	14.5%	17.4%
Genre	27.4%	43.1%
MovieLens	50.0%	51.1%
MPAA	14.5%	34.1%

Table 7.9: Feature inclusion in the MetaLite recommendation table.

Finally, we considered user-provided "values" indicating a user's interest in sub-items within each of the features. User values for distance and Cream Percent (percentage of top critics liking the movie) are not different between MetaLens and MetaLite. The percentage of queries containing each of the sub-items for genre and MPAA Rating is displayed in Table 7.10. Observe that several of the genres and all of the MPAA ratings were requested fewer times by users of MetaLite than by users of MetaLens (the highlighted items). Consider that users of MetaLite have fewer decisions to make than users of MetaLens. Users have a total of thirty possible changes to make to the default query of MetaLite, compared to 82 for MetaLens. In both cases, these include de-selecting sub-items in the genre and MPAA Rating features. It is not surprising that we observe fewer instances of several of these sub-items in MetaLite. Presumably, users have more time to consider each change when faced with fewer changes.

Sub-item	MetaLite	MetaLens
Action	83.9%	93.3%
Comedy	85.5%	96.5%
Documentary	83.9%	80.0%
Drama	91.9%	96.3%
Family	61.3%	62.7%
Foreign	90.3%	86.0%
Horror	77.4%	83.9%
Musical	71.0%	72.6%
Romance	77.4%	87.5%
Science	79.0%	90.9%
Thriller	85.5%	91.1%
G	72.6%	83.0%
PG	80.6%	95.5%
PG-13	83.9%	98.5%
R	90.3%	98.0%
NC-17	80.6%	90.6%
NR	82.3%	93.3%

Table 7.10: MetaLite sub-item inclusion in queries.

7.4.2 MetaClick

During the four-week period previously described, 133 users followed the MetaClick link from the MovieLens home page. Of these, 86 (64.7%) chose to register (when necessary) and submit a query. These 86 users submitted a total of 173 queries. During this time, two users (2.3%) earned active user status and submitted 24 (13.9%) of the total queries. This information is included in Table 7.7. These numbers suggest that usage rates with MetaClick are poorer than those observed in MetaLens and MetaLite. However, lack of a controlled study leaves us uncertain.

Table 7.11 lists the number of times each profile was selected by users. Note that ten of these queries (5.7%) were submitted with the "Pick One" selection still showing in the profile pull down menu. Users who select this "option" are instructed to return to the selection screen and select a profile. Each of the ten users did submit a follow-up query. We can think of this as analogous to users who submit the default query with MetaLens or MetaLite. Presumably, users are more interested in seeing what the system does than actually getting the recommendations. Therefore, they simply submit the "preferences" screen as it initially appears. Unfortunately, with MetaClick this is an invalid submission. We find this particularly plausible because we note that the frequency of

"Pick One" submissions is nearly identical to the frequency of default query submissions in MetaLens.

Click Profile	**Number of Submitted Queries**
Pick One[23]	10
Date with significant other	46
Movies to see with the kids	3
Guy's night out	34
Chick flicks	18
Cheap and close to home	21
Artsy and/or non-traditional	41

Table 7.11: Number of times each profile was selected.

7.4.3 Multi-system Usage Analysis

We identified 75 users who used at least two of the three systems. Of these, 57 had tried two systems while the remaining eighteen had tried all three systems. Table 7.12 lists the number of users using each ordered combination of the three systems. It is worth pointing out that the order in which these systems are presented on the MovieLens screen has less effect on the order in which users tried the systems than the order in which they became available. MetaLens was available nine weeks prior to MetaClick and MetaLens. Thus it was the most common first system. However, while MetaLite appears prior to MetaClick on the MovieLens page, MetaClick is the more commonly requested second system.

Since MetaLens was available for nine weeks prior to the implementation of MetaLite and MetaClick, it was not surprising to discover that 74 of these 75 users had tried MetaLens. Of the 74, 67 had used MetaLens at least once prior to first trying one of the other systems. Of the remainder, seven tried MetaClick first followed by MetaLens while eight tried MetaClick followed by MetaLite.

Order in which systems were **first** used	Number of users
MetaLens, MetaClick	35
MetaLens, MetaLite	14
MetaLens, MetaLite, MetaClick	10
MetaLens, MetaClick, MetaLite	8
MetaClick, MetaLens	7
MetaClick, MetaLite	1

Table 7.12: Number of users using each ordered combination of meta-recommender systems.

[23] Users neglected to select one of the six options.

Analysis of the limited number of "active users" who have tried multiple systems provides an indicator to how usage data may shift as all three remain on the MovieLens site. Of the 75 users who have used at least two of the systems, thirty of these qualify as active users of MetaLens. Additionally, two are active users of MetaLite, and two are active users of MetaClick. However, these four users also are among the thirty active users of MetaLens.

Anecdotally, it is worth considering these "dual" active users. The system usage of these users is summarized in Table 7.13. The table lists the number of consecutive MetaLens sessions each user completed prior to the introduction of MetaLite and MetaClick. It also lists the system tried during their first "exploratory session" and the systems they have used since that session.

User	**Number of Previous MetaLens Sessions**	**System(s) tried during the exploratory session**	**Systems used since exploratory session**
MetaLite Active #1	9	MetaLite and MetaClick	MetaLite for four sessions, then back to MetaLens.
MetaLite Active #2	17	MetaLite	MetaLite for four sessions, then MetaLite and MetaLens alternately for six sessions.
MetaClick Active #1	8	MetaClick	MetaClick for two sessions.
MetaClick Active #2	2	All Three	MetaLens and MetaClick together in seven sessions.

Table 7.13: System use patterns of "dual active" users.

Note that of these four, one switched his preferred recommender to MetaClick, one returned to MetaLens, and the other two found two systems equally helpful. We find somewhat similar results when we consider the remaining 26 active users of MetaLens. Of these, two appear to have switched to one of the new systems (one each), while one alternated between MetaLens and MetaLite. Eleven users tried one or both of the new systems and then used MetaLens for all additional sessions although three of the eleven had only one subsequent MetaLens visit. Finally, the remaining twelve users established themselves as active users of MetaLens, tried one or both of the new systems, and did not return prior to completion of this analysis period. It is impossible to categorize these users.

7.5 Summary

At the beginning of this chapter, we addressed two concerns regarding initial MetaLens usage data which caused us to question the initial implementation of the MetaLens recommender system. These were a poor return rate and a high number of low-interest features. Results from initial analysis of two additional meta-recommenders using the MetaLens Recommendation Framework suggest that our initial design may have been adequate.

First, we expressed concern over a poor return rate. However, both MetaLite and MetaClick also suffered from poor initial return rates. MetaLite produced an active user rate identical to that of MetaLens while the MetaClick rate is only half of either of these systems. This suggests that return rate may be less a sign of bad design and more a sign of a slow acceptance rate for this new class of recommender systems. This is supported by reconsidering the "active user" rate of MetaLens. During the first fourteen weeks that MetaLens was part of the MovieLens site, 950 users submitted at least one query. Of those, 206 visited the site during three or more sessions. This yields an active user rate of 21.7%; much higher than the 7% rate over eight weeks that we reported in Chapter 6. In fact, this rate is higher than the 15.2% rate observed for MovieLens during a comparable thirteen week time period. To be fair to MetaLite and MetaClick, return rates over comparable periods of time need to be studied.

Next, we expressed concern over the high number of low-interest features. Recall that for seventeen of the nineteen features used in MetaLens, the most common, non-default weight selection was a zero weight. Turning this around, for only two features – MovieLens prediction and "Not Seen" – was the most common, non-default weighting something other than zero. We hypothesized that creating a system using a limited number of more popular features would provide a system in which features were weighted more highly, but this does not appear to be the case. Recall that average weight assigned to five of the six features present in MetaLite have remain unchanged from the average weight assigned in MetaLens, while the weight actually dropped for the sixth feature. Furthermore, Table 7.8 shows that while MovieLens prediction and "Not Seen" are once again most commonly assigned a high value, three of the remaining four continue to have zero as the most common, non-default value. The fourth feature, distance, has 0.25 as this value.

Finally, recall that survey results suggested active users of MetaLens preferred MetaLens over MetaLite and MetaClick while non-active users reported a preference for MetaLite. In general, we do observe the MetaLens preference when we consider actual use of these systems. A significant number of users tried MetaLens or MetaClick and have returned to using MetaLens as their primary meta-recommender. There is insufficient data to support this preference among non-active users.

All told, results from this chapter indicate that our initial MetaLens design was sound and that we simply need to provide a period of time for users to learn the system and recognize its value.

Chapter 8: Conclusions

> The real voyage of discovery consists not in seeking new landscapes but in having new eyes.
>
> -Marcel Proust

In a world where the number of choices can be overwhelming, recommender systems help users find and evaluate items of interest. As implemented in electronic commerce, they do so by connecting users with varying degrees of information regarding the content of recommended products and the opinions of other consumers who have purchased the items. In this thesis, we focused on a new class of recommender systems called meta-recommenders. Meta-recommender systems build on existing recommender technologies by giving users control over the combination of rich recommendation data to yield more personalized recommendations.

The work presented in this thesis focused on developing a meta-recommender framework that users find understandable, usable, and helpful. Controlled use experiments show that users prefer these systems and find them more helpful than traditional systems. Implementation studies show the development of three different systems built within the MetaLens Recommendation Framework and demonstrate that users will use these systems. Finally, data indicates that users who take the time to personalize a system to their liking find the system worth repeated use.

8.1 Contributions

The work presented in this thesis makes several significant contributions to the field of recommender systems. We began in Chapters 2 and 3 by considering several technologies used in creating recommender systems, and the variety of ways these technologies were being applied and recommendations presented in electronic commerce recommender applications. The range of combinations of systems has made it difficult at times for developers and researchers to define how their systems compare to others. To address this concern, we created a taxonomy for recommender applications in e-commerce. While this taxonomy will undoubtedly grow and adapt as new systems are developed, it provides a common basis for system definition and classification and a mechanism through which design choices and tradeoffs can be compared.

Based on this taxonomy, Chapter 3 also discussed results of a study which considered correlations between recommender application models and the sites that choose to implement them. This study confirmed that different product domains focus on different recommender application models and that there are correlations between product attributes and the recommender application models that are used to recommend these products. These results are a powerful foundation for future, more detailed studies. Furthermore, e-commerce application developers can benefit from this research by examining the attributes of the products for sale through their sites, and considering the applications used with products of similar classification.

Next, we introduced meta-recommenders and presented a framework for the collection of data and generation of recommendations within this new class of recommender system. In presenting this framework in Chapter 4, we have explained the challenges and requirements in creating meta-recommenders and have provided researchers and developers alike with a model upon which they may base their own designs. Additionally, by considering the implementation of three different recommender systems using this framework, we have shown that there are still design issues to be considered.

In addition to introducing a meta-recommender framework, we have presented work addressing three research challenges concerning meta-recommender systems as originally presented in Chapter 1. The first challenge was "**What format should meta-recommendations take?"** In answering this question, we provide developers with a proven starting point for the construction of future meta-recommender applications. While the simplest format presents only the items being recommended, this format fails to leverage the rich recommendation data used in making recommendation decisions. Chapter 4 presented results of two experiments considering whether users want this additional information and how best to deliver it within the meta-recommendation process. First, these results indicate that users want additional information. Users overwhelmingly indicatee that they found meta-recommendations containing only the recommended item to be insufficient. Second, these results indicated that when the amount of recommendation data was limited users would accept receiving all of it with recommendations. However, when the amount of recommendation data became more plentiful, users preferred being granted control over the selection of which portion of the data was received as part of the recommendations. While designed to fight information overload, a poorly designed or implemented recommender system can actually provide as much confusion as the original problem. By considering these findings, developers can increase the likelihood that users find their recommender systems helpful.

The second challenge addressed was "**Which interfaces do users prefer in a recommender system?**" In a market place where sites must always strive to best their competitors, knowing with which interfaces users prefer to interact may make the difference when it comes to survival. Most current recommender systems provide users with information that may be only a piece of a much larger puzzle. They require users to manually gather pieces from several systems in order to visualize the whole picture. In Chapter 5, we presented results of a user study that suggest that users feel current systems have provided them with incomplete recommendations. More specifically, results suggest that users are cognizant of the effort required to make informed decisions with most current recommender applications. Furthermore, they recognize the helpfulness of meta-recommender systems which help them bring together pieces of the puzzle. Knowing this, developers can choose to implement more helpful and complete systems. While our results suggest this can be accomplished through meta-recommenders, this information may challenge developers to consider other, more informative ways to present their users with recommendations and information.

Finally, we concluded the thesis by considering a third research challenge, "**How do users interact with meta-recommender systems?**" Meta-recommenders are a new class of recommender systems. As such, several lessons can be learned by considering user interaction with different types of systems. Chapters 6 and 7 presented analysis of several levels of interaction with three recommenders built within the MetaLens recommendation framework. The key lesson learned from these analyses is that users like, and in fact often prefer, meta-recommenders. When three variations of meta-recommenders were made publicly available, users found them to be worth repeated use. However, it is also worth noting that acceptance comes at a slow pace. Users need to gain confidence in a system and their ability to interact with the system. Additionally, we observed that users prefer to have the ability to customize meta-recommenders. Because meta-recommenders provide access to a large quantity of recommendation data, users want the ability to save general preferences concerning which data is important to them. These issues are related. In order to gain confidence in a system, users need to be able to feel it is working for them. By providing users with customization features, sites can help users feel that the system is *their* system. Finally, we identified that different users have different levels of expectation for what makes a complete recommendation. While some users want access to absolutely every piece of recommendation data available, others feel a limited subset is sufficient. Designers of recommender applications must keep each of these principles in mind while designing new systems.

8.2 Limitations

While the work presented in this thesis has made several contributions, it admittedly faces several limitations. First, while our results suggest that meta-recommenders are an extremely promising new class of recommender systems, these results are based on studies limited to the domain of movies. It remains to be seen whether the benefit and acceptance of these systems will transfer to other domains.

Second, meta-recommenders have yet to be proven in a large, real-world, runtime application. The recommender systems built within the MLRF served a very limited number of users and depended heavily on human interaction to validate pre-retrieved and cached data. Presumably, real world applications would need to be able to negotiate for much of the recommendation data at runtime. Our studies have not indicated whether or not such applications could do so in a timely manner.

Third, future applications of meta-recommenders will have to consider the issue of proprietary data. MetaLens gathers data from several Internet sites without asking for permission. For research purposes this falls into the realm of fair use. However, a commercial application of MetaLens could expose itself to legal troubles for similar actions. Although much of the data gathered from a site such as Yahoo Movies is publicly available Yahoo could argue that, as a collection, it is copyright protected.

Designers of future meta-recommenders will need to consider carefully the legality of their data sources.

Finally, even if meta-recommenders do make the transfer to other domains, recommender systems are a dynamic field, and they continue to evolve at an extremely rapid rate. It is uncertain whether the interest in meta-recommenders is due to a genuine belief by users that they provide more informative recommendations or due to the novelty of the systems.

8.3 User Control, Confidence, and Privacy

In introducing meta-recommenders we have stressed the issue of user control. Our discussion, however, has been restricted to user control over the recommendation process. We have not addressed the issue of a user's control over his personal data. At the root of this discussion is the concept of user confidence. This includes confidence that the recommendations being provided are unbiased, confidence that sites are not gathering data about the user without consent, and confidence that collected data will not be "used against them."

While users find value in control over the recommendation process, they must feel confident that the recommendations being produced are unbiased and based on their personal control. That is, a user must believe that the top items are being recommended because they best match his query and not because a distributor is trying to push a particular item. Future meta-recommenders, particularly in commercial applications, must consider mechanisms to assure users that recommendations are truly based on user control.

In an attempt to grant the users of a meta-recommender control over the recommendation process we have also provided the creators of such systems an opportunity to gather a significant amount of data about a user. To get meta-recommendations from a system like MetaLens a user must provide information concerning her preferences and desires. While some users are willing to allow sites to store this data to provide more customized service, others feel strongly that such an action is an invasion of their privacy. Designers of future meta-recommenders must consider how to walk this fine line of providing a user with the best service possible while allowing her to feel like she has retained control of her personal information.

Finally, even when a user consents to allow a site to build a profile about him, he may want assurances that the site will not use this information against him at a later time. For example, the fact that a given user frequently expresses an interest in science fiction movies could be sold to the SciFi Book of the Month Club for use in targeted advertising. Again, designers of future meta-recommenders must consider these issues and address ways to provide users with the control they want over both the recommendation process and the use of their personal data.

8.4 Future Work

We believe this thesis has laid a foundation for future research questions focusing on three areas of the recommendation process. These include system design, customization features, and user interface design.

System design issues focus on modifications and improvements to the MetaLens Recommendation Framework or future implementations of meta-recommenders. These include:

- *Which algorithms provide the most relevant ranking*? The extended Boolean retrieval algorithm currently used by the MetaLens framework has a number of variables that the application programmer can modify to tweak the accuracy of results. What settings provide better recommendation rankings? Are there other algorithms that can do the job in a more efficient or accurate manner?
- *What metric best evaluates the accuracy of recommendation rankings*? To test algorithms, we need to have a metric that can adequately evaluate the quality of the recommendation rankings produced by an algorithm. Most metrics will depend on an analysis of which items are "good" and which items are "bad." Acquiring this analysis may be difficult based on the complex nature of the data being used in the decision making process. However, the

validation of the recommendations being produced is an essential next step in validating the concept of meta-recommenders.

- *Can we build modular meta-recommenders*? Analysis of which features users include in their decision-making process suggests that different users have very different requirements. While one user will insist on having access to a particular feature, another will absolutely never use it. What interface changes need to be made to allow a user to "build" his own custom recommender based on the features he feels he is likely to use? What framework changes need to be made to support easy and quick changes to such a MetaCustom?
- *How do we balance data availability and system usefulness?* While experimental results indicate that users want access to as much data as possible, other results indicate that the addition of this data may cause the interface to be less helpful overall. What amount and set of data provides the highest combined access and system usefulness?

Customization issues are concerned with how we can make systems better by providing users with the ability to personalize the system to better fit their needs. These issues include:

- *How do we allow for user-defined feature inputs*? It is not uncommon for users of recommender systems to request that feature X be added into the system. A recommender system using the MetaLens framework should be able to handle inputs from any recommendation data source so long as the input is provided in a format that can be easily fused with the base data. However, the framework would need to be modified to allow for the dynamic input of data and the recommendation algorithm adjusted to handle a varying number of recommendation features.
- *How do we negotiate privileged inputs*? MetaLens depends on the receipt of recommendation data from third-parties. When this data is publicly accessible, there are few problems. However, many sites restrict data access

to registered users of the site. What protocol needs to be added to the MetaLens framework to allow for the acquisition of data by a registered user?

User Interface issues are concerned with ways to improve a user's interaction with the recommender system. These include:

- *How does the process change if we provide recommendations through different interfaces*? One of the advantages of meta-recommenders is the fact that they involve such a rich assortment of recommendation data. While the present interface matches other similar interfaces, user interface design experts like Shneiderman would argue that the current interface does not allow users to interact properly with the data.
- *Can an "Automatic" recommendation format that users like and trust be developed*? Earlier work suggested that users do not want to have the system determine which items to display with their recommendations. This may have been due more to a poor development of the Automatic format's algorithm and less to a user's need for control. Is there an underlying rationale behind what is displayed that can be extracted and used to create a helpful Automatic format?

Appendix I – Electronic Commerce Recommender Applications

Figure I.1: Amazon.com Customers Who Bought

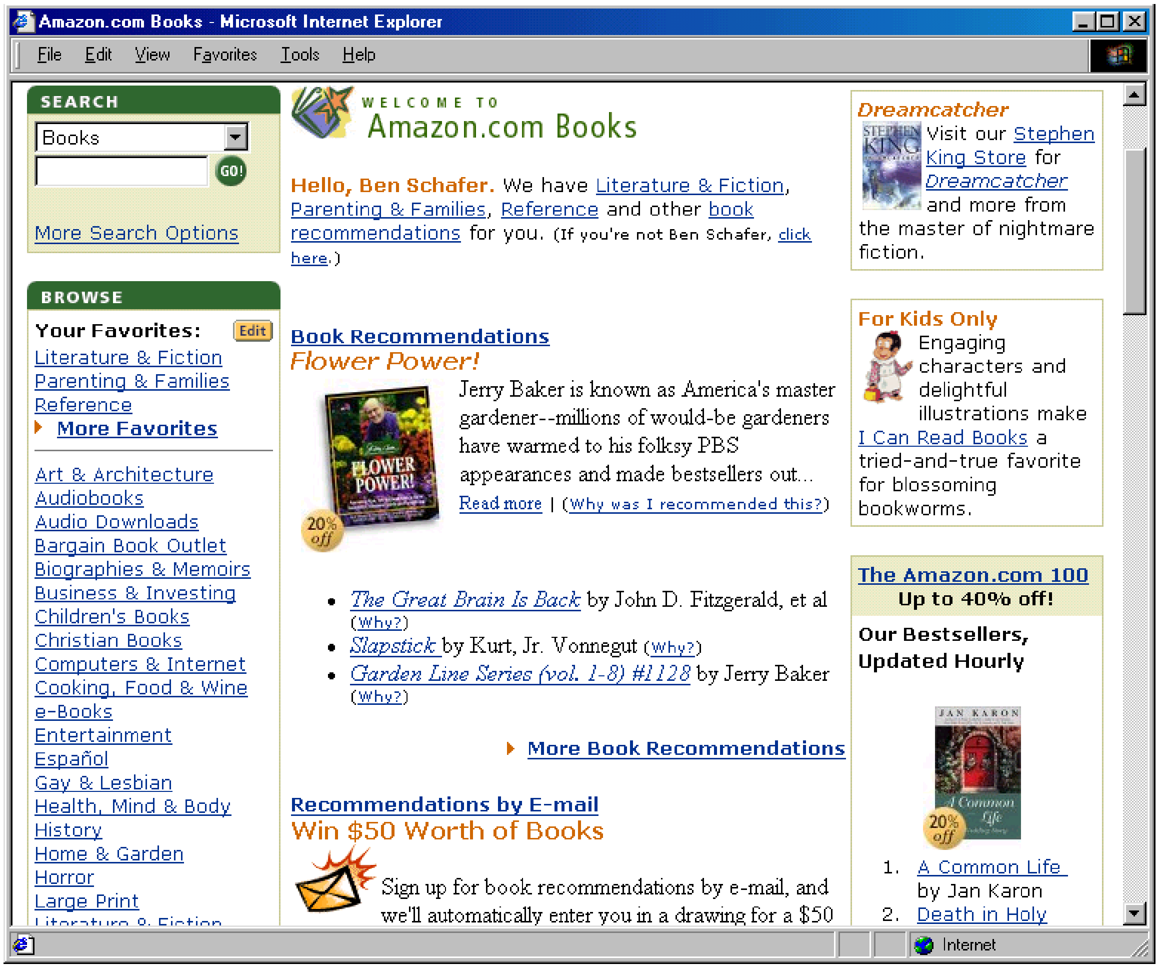


Figure I.2: Amazon.com Your Recommendations

Figure I.3: Amazon.com Alerts (formerly Eyes)

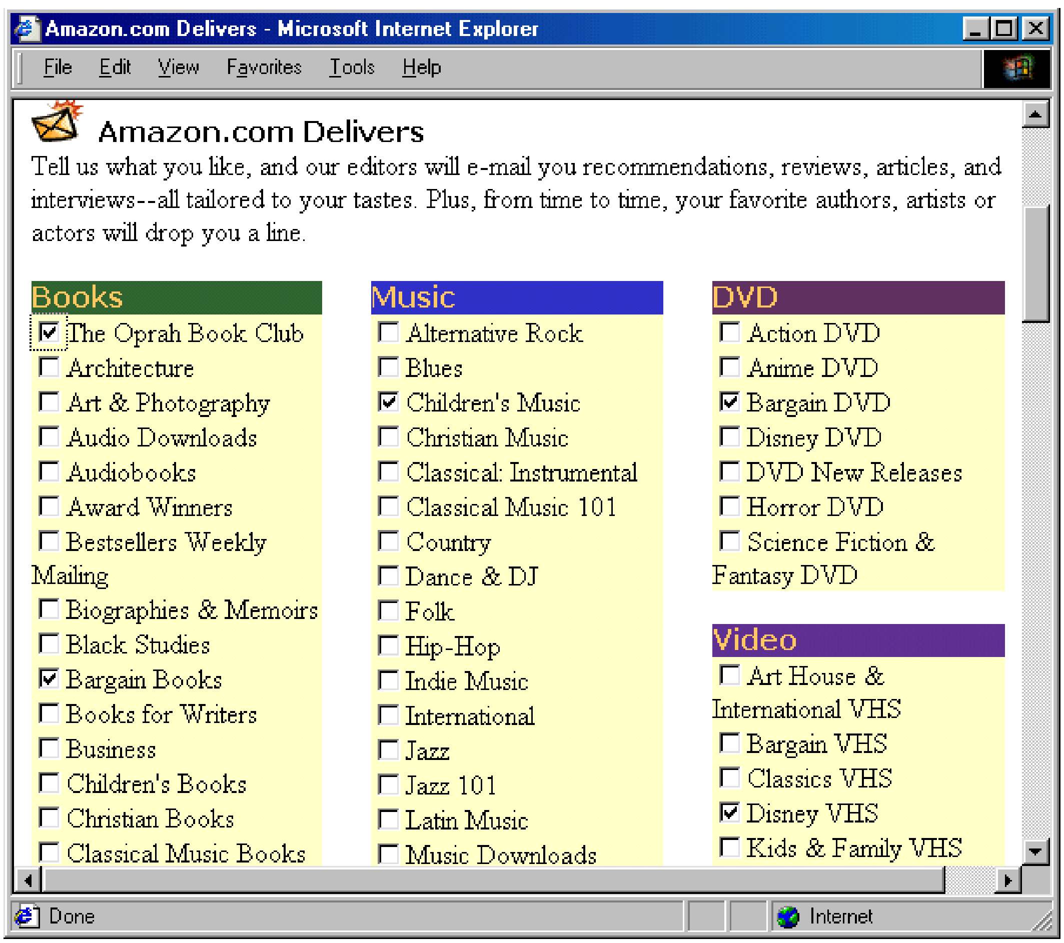


Figure I.4: Amazon.com Delivers

Figure I.5: Amazon.com Bookstore Gift Ideas

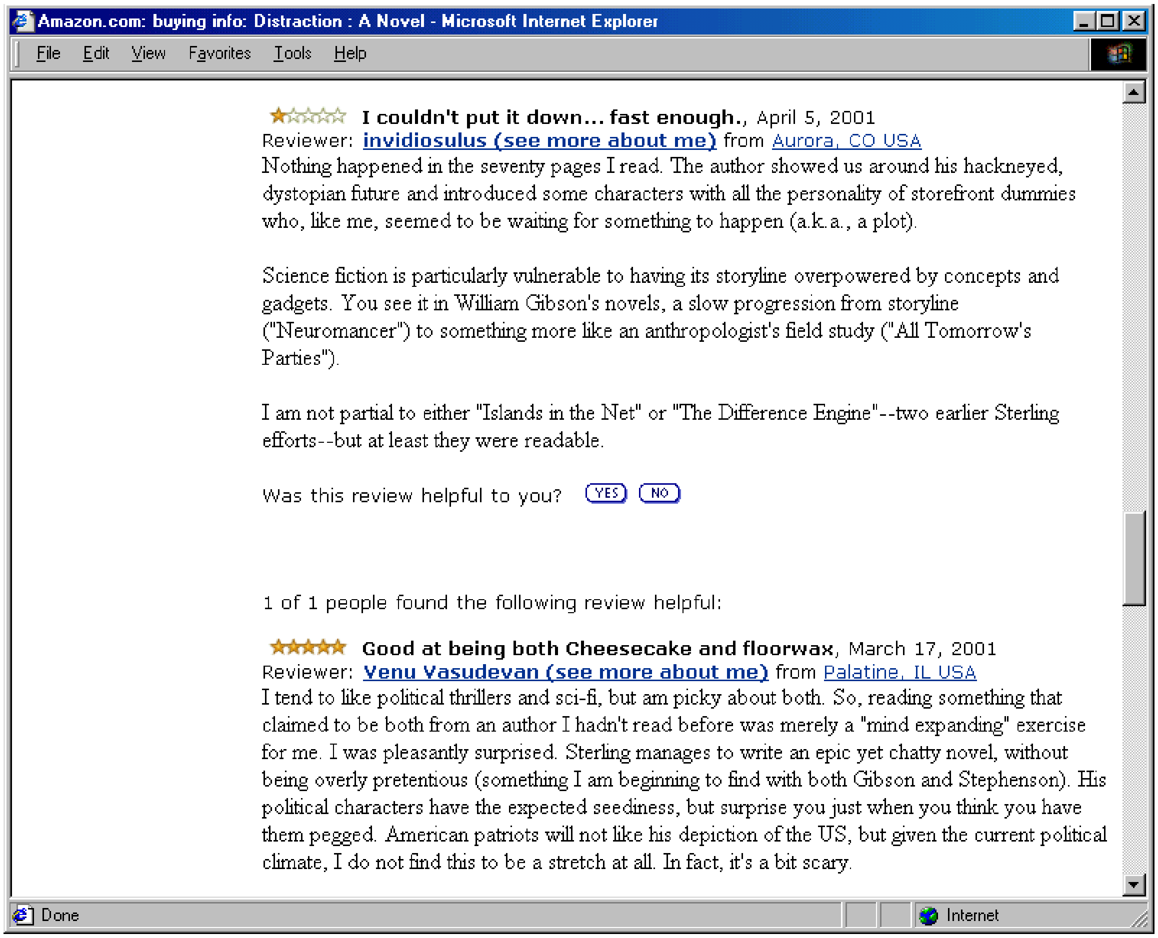


Figure I.6: Amazon.com Customer Comments

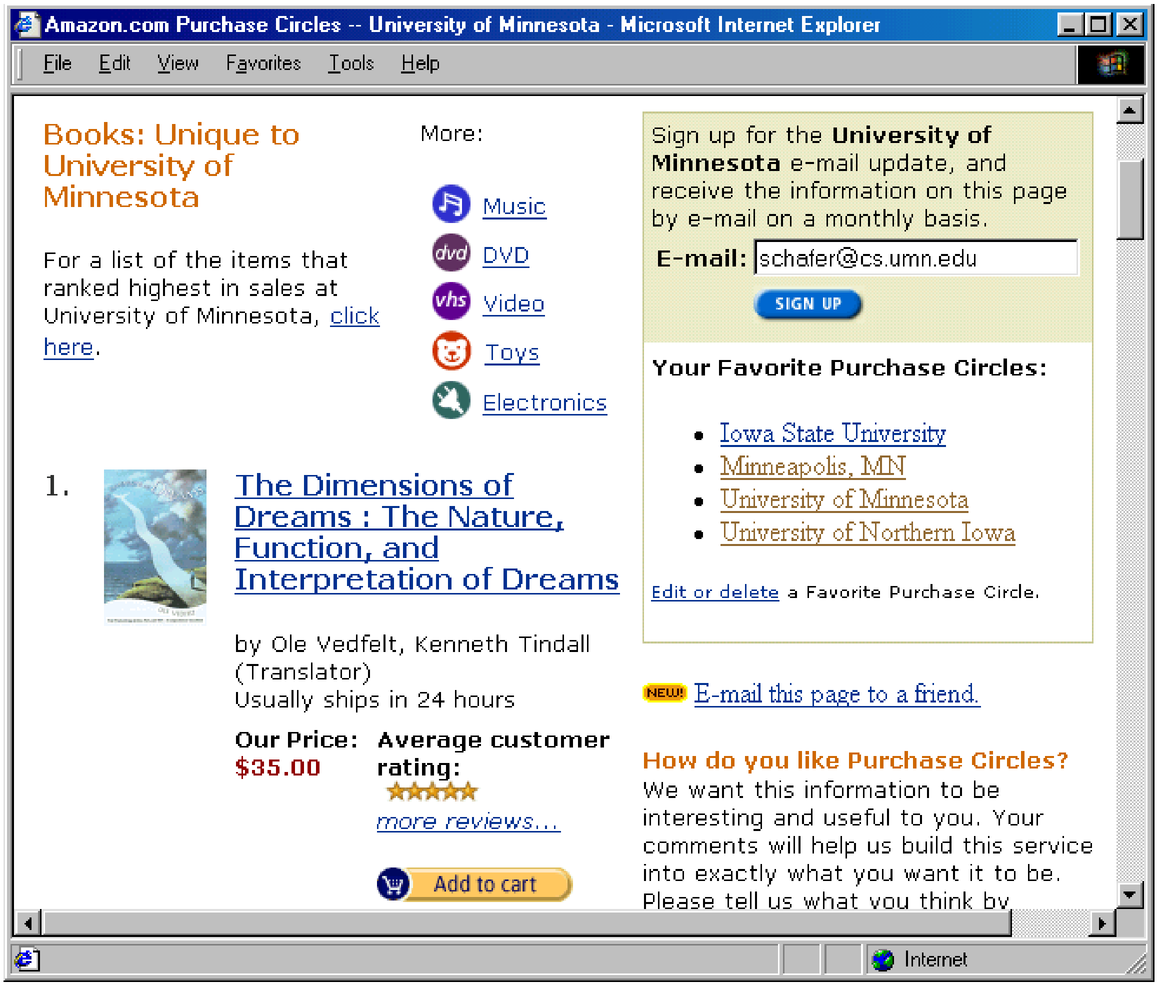


Figure I.7: Amazon.com Purchase Circles

Figure I.8: CDNOW Album Advisor

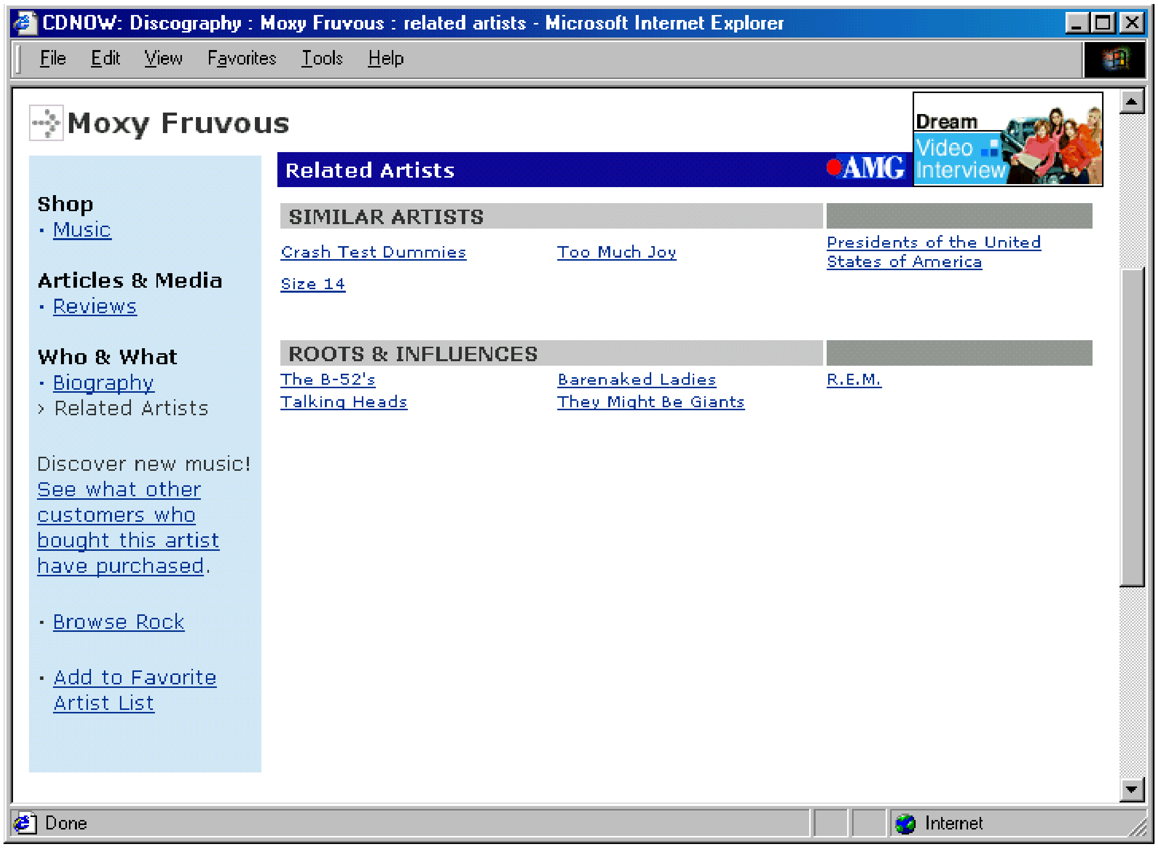


Figure I.9: CDNOW Related Artists

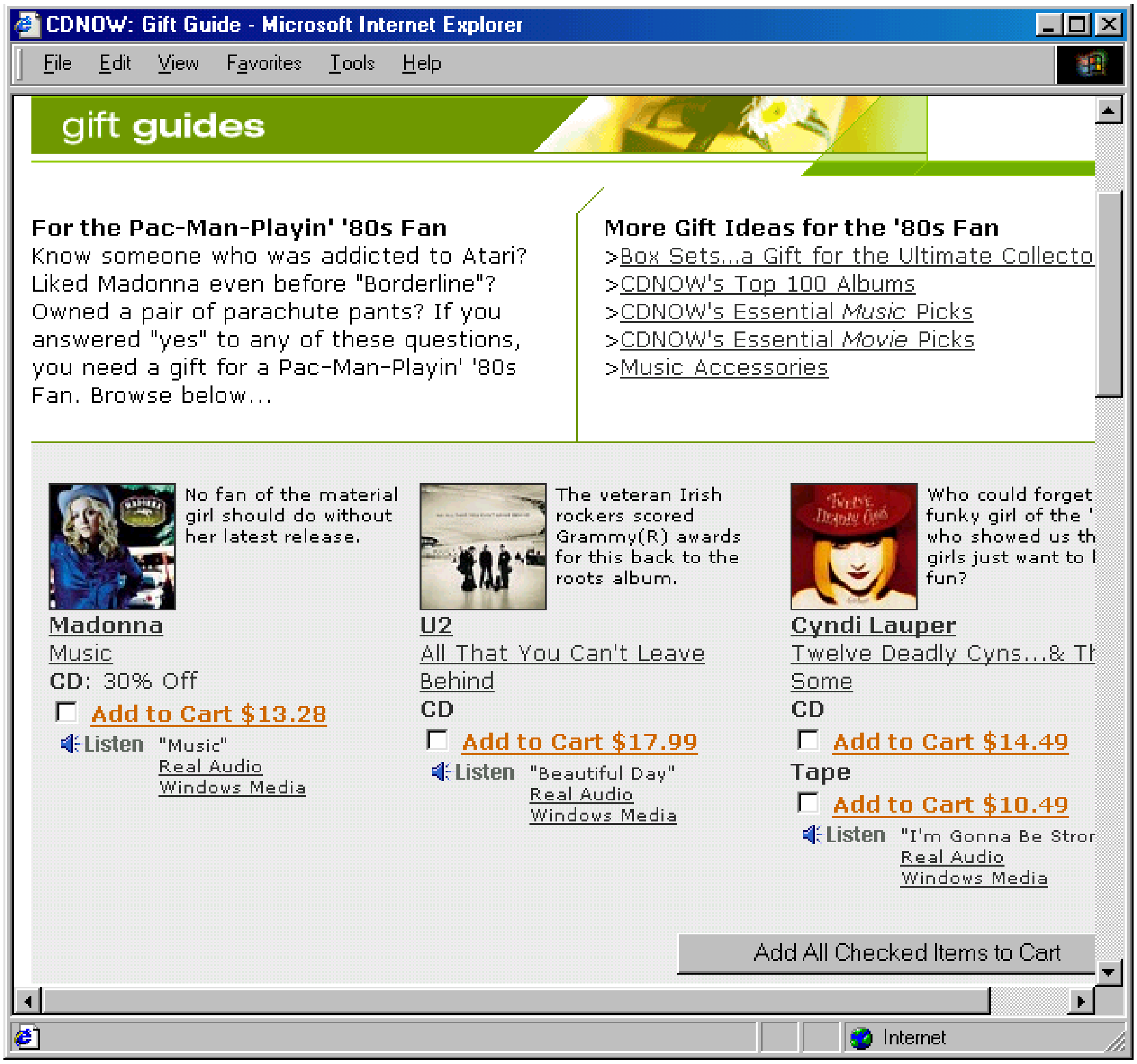


Figure I.10: CDNOW Gift Guide (formerly Buyer's Guide)

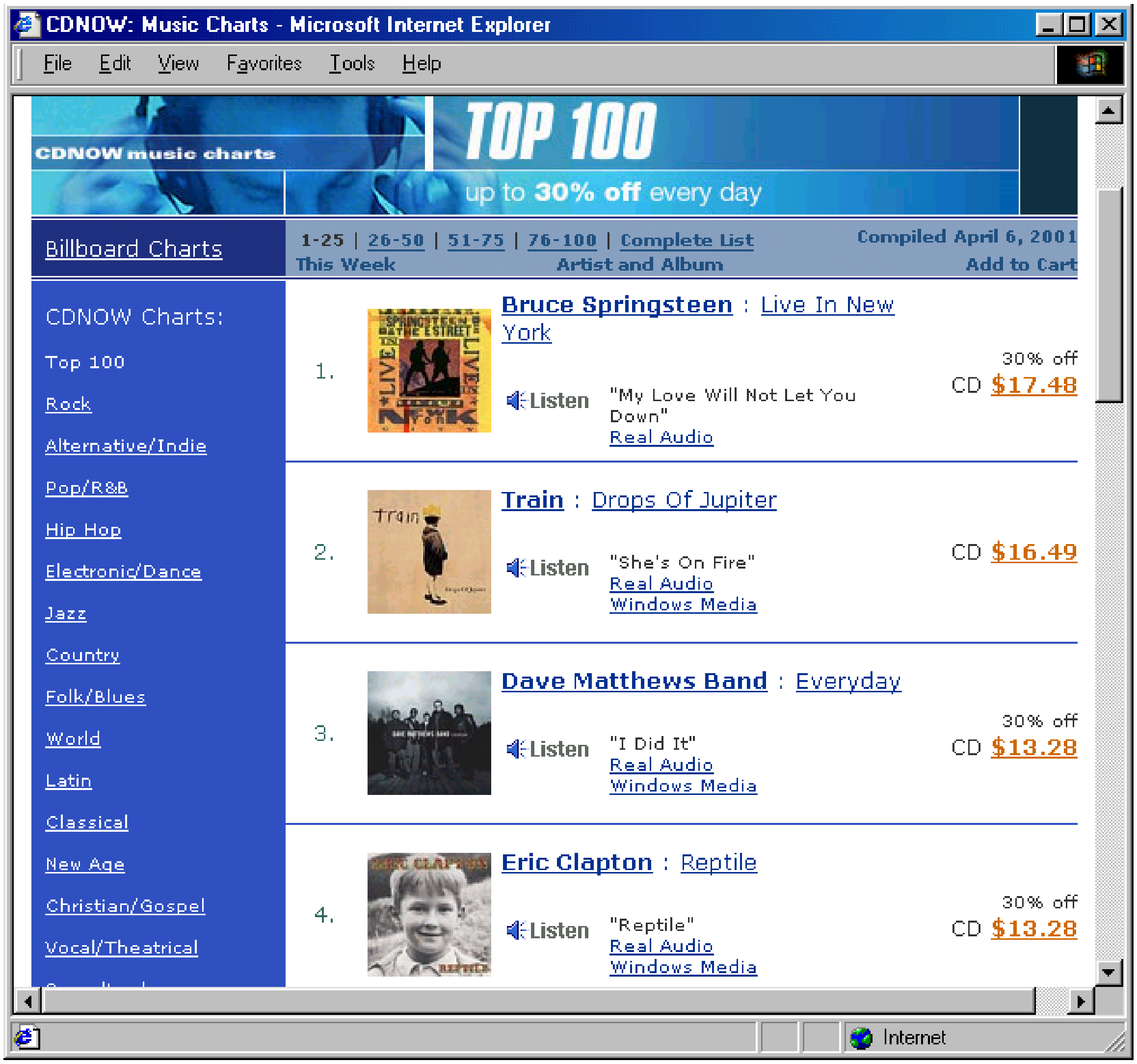


Figure I.11: CDNOW Top 100

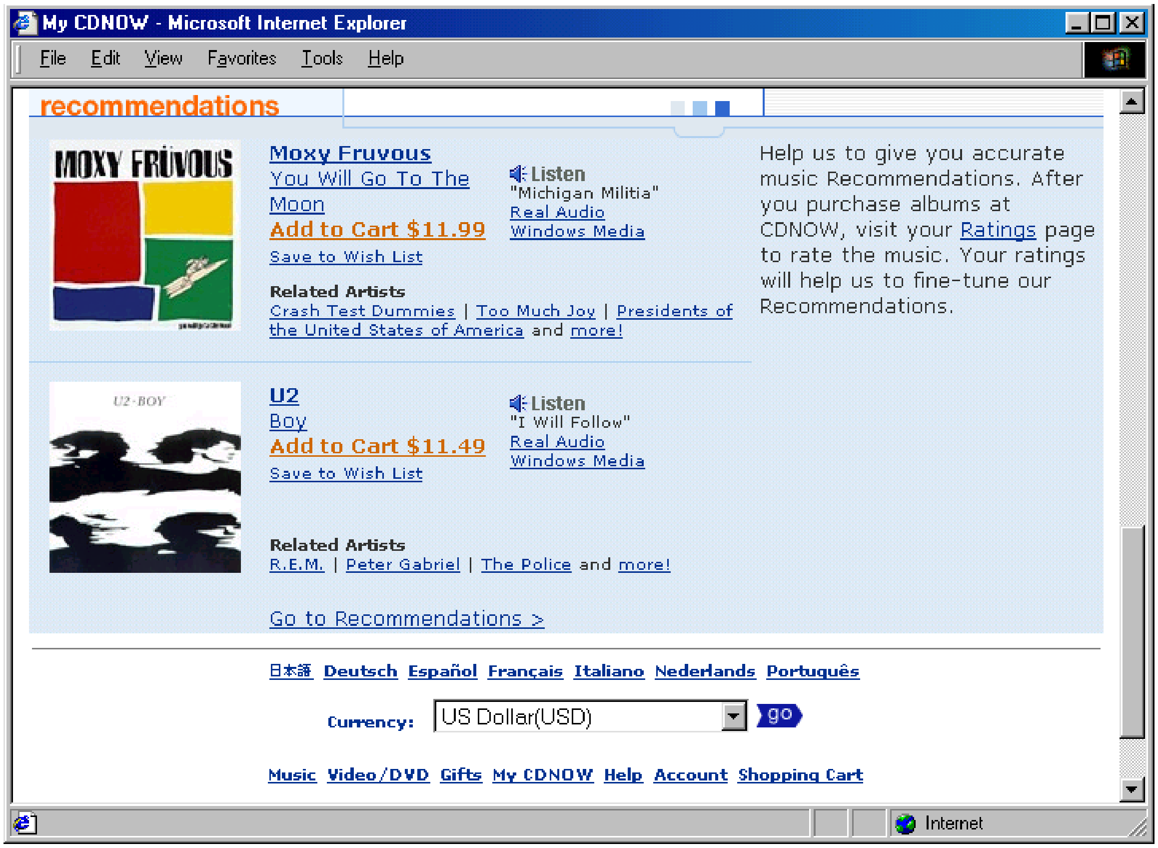


Figure I.12: My CDNOW

Figure I.13: drugstore.com Test Drives

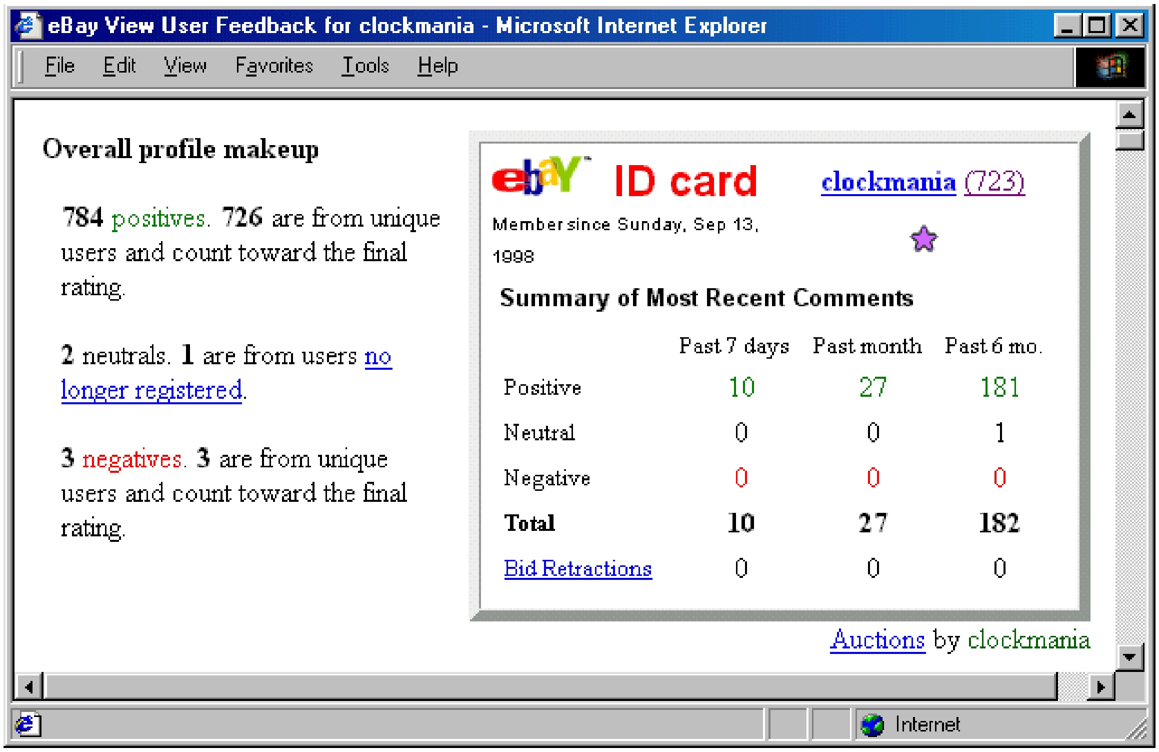


Figure I.14: eBay Feedback Profile summary

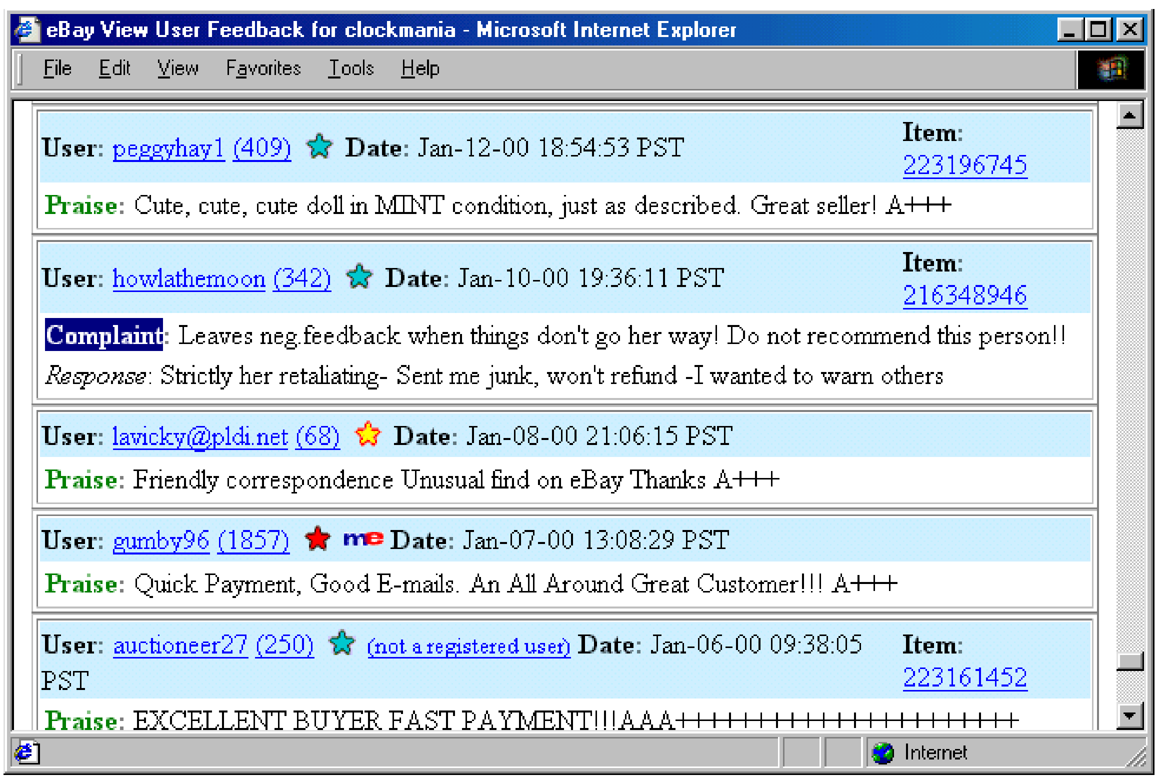


Figure I.15: eBay Feedback Profile comments

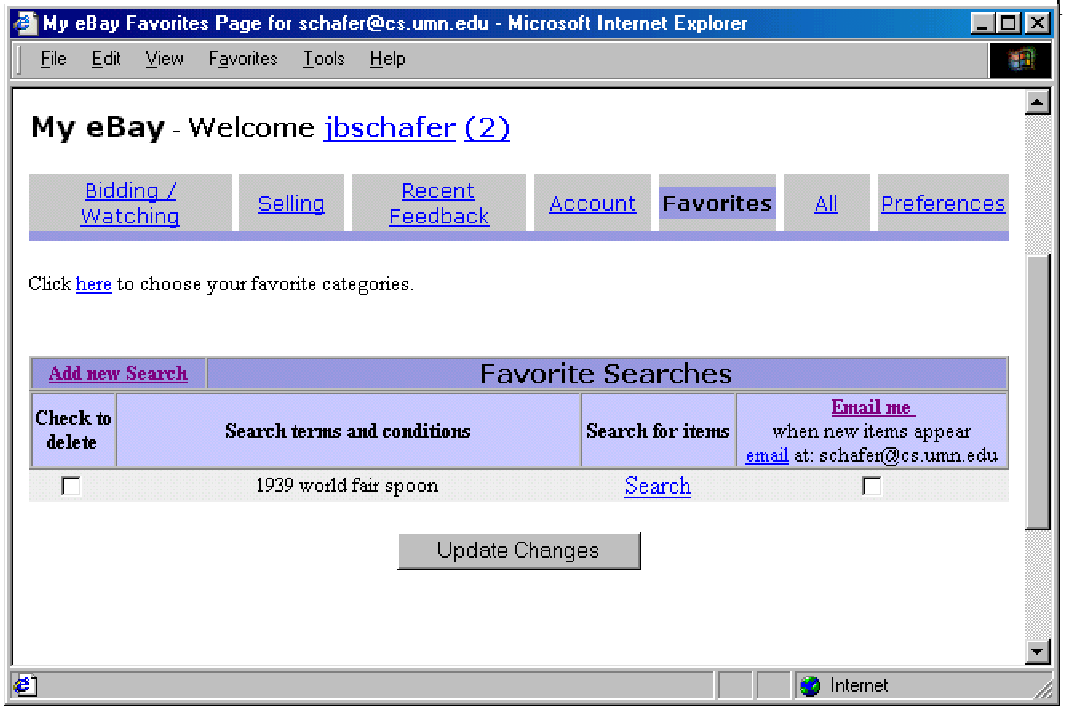


Figure I.16: eBay Favorite Searches (formerly Personal Shopper)

Figure I.17: MovieFinder.com Our grade/User's grade

Figure I.18: Moviefinder.com Top 10

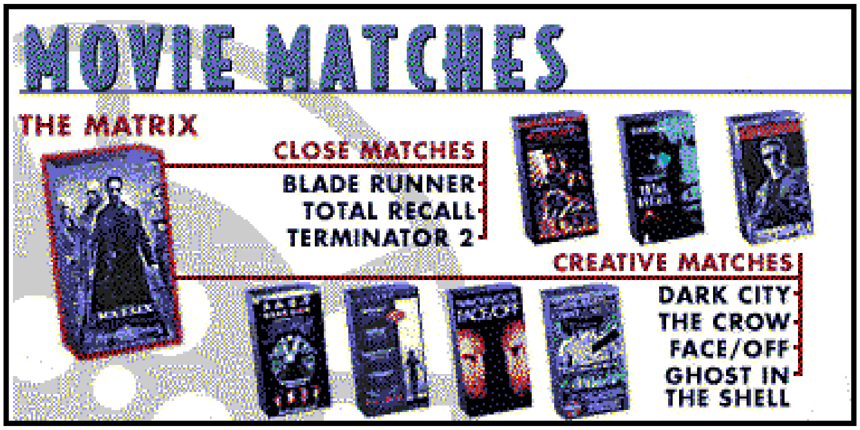


Figure I.19: reel.com Movie Matches

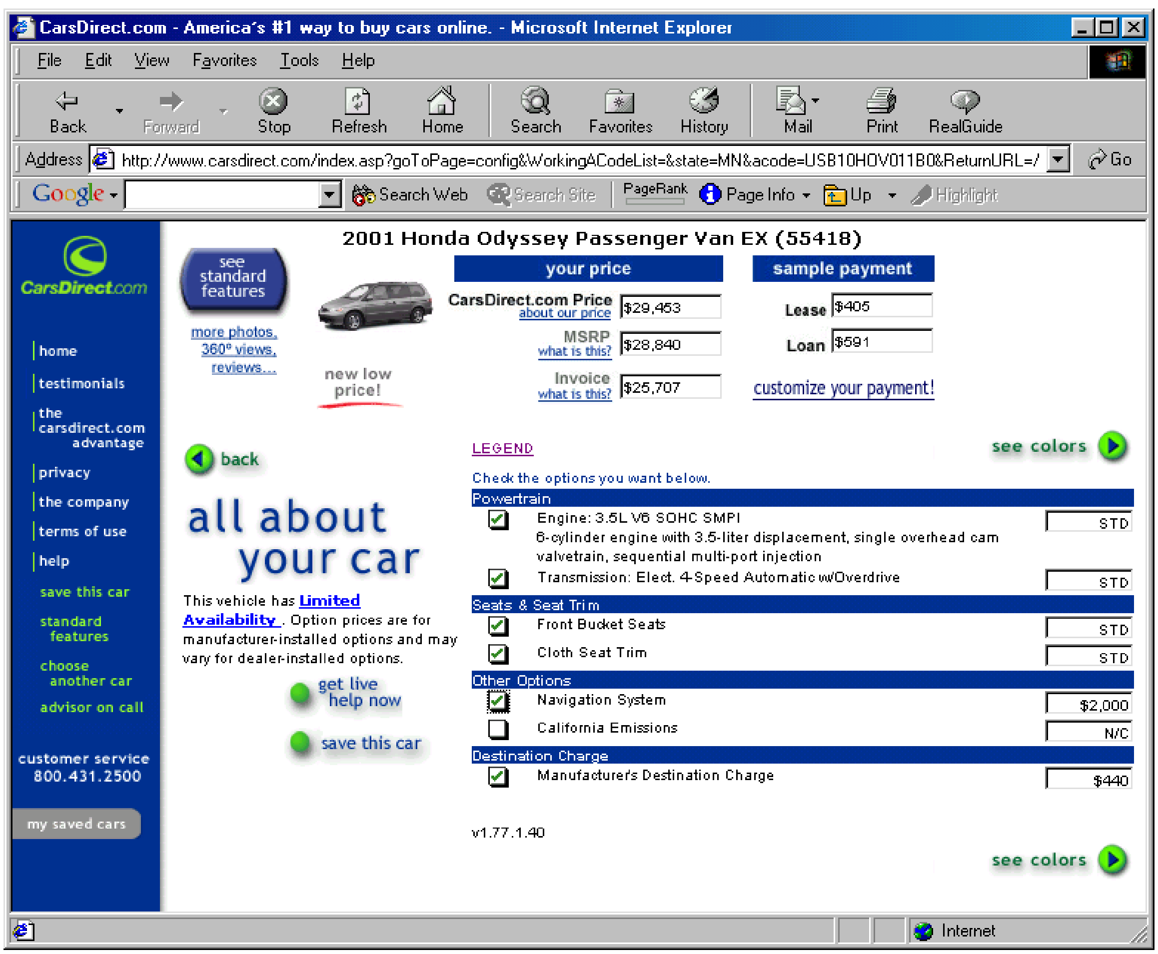


Figure I.20: carsdirect.com Shopper

Bibliography

1. Adomavicius, G., and Tuzhilin, A. (1999). User Profiling in Personalization Applications through Rule Discovery and Validation. *1999 Conference on Knowledge Discovery in Data (KDD99)*, pp.377-38.

2. Aggarwal, C.C., Sun, Z., and Yu, P.S. (1998). Online generation of Profile Association Rules. *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD98),* pp. 129-133.

3. 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. *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 201-212.

4. Agrawala, A.A., Larsen, R.L., and Szajda, D. (2000). "Information Dynamics: An Information-Centric Approach to System Design". University of Maryland Tech Report CS-TR-4108.

5. Agrawal, R.C., and Srikant, R. (1994). Fast Algorithms for Mining Association Rules. *Proceedings of the 20th VLDB Conference,* pp. 487-499.

6. Balabanovíc, M., Shoham, Y. (1997). Fab: Content-Based, Collaborative Recommendation. *Communications of the ACM* 40 (3), pp. 66-72.

7. Belkin, N.J. and Croft, W.B. (1992). Information Filtering and Information Retrieval: Two Sides of the Same Coin. *Communications of the ACM* 35(12), pp. 29-38.

8. Berleant, D. and Berghel, H. (1994). Customizing Information. *IEEE Computer* 279(9 and 10): pp. 96-98, 76-78.

9. Billsus, D., Pazzani, M.J. (1998). Learning collaborative information filters. *Machine Learning Proceedings of the Fifteenth International Conference (ICML'98)*.

10. Billsus, D., Pazzani, M.J. (1998). Learning collaborative information filters. *Proceedings of the 1998 Workshop on Recommender Systems*.

11. Bohte, S.M., Langdon, W.B., Poutré, H.L. (2000). On Current Technology for Information Filtering and User Profiling in Agent-Based Systems, Part I: A Perspective. (http://www.cwi.nl/projects/TA/reports/profiling.ps).

12. Boone, G. (1998). Concept Features in Re:Agent, an Intelligent Email Agent. *Proceedings of the ACM Conference on Autonomous Agents (AA-98)*, pp. 141-148.

13. Brachman, R.J., Khabaza, T., Kloesgen, W., Piatesky-Shapiro. G., and Simoudis, E. (1996). Mining Business Databases. *Communications of the ACM, 39(11*), pp. 42-48.

14. Bradley, P.S., Fayyad, U.M., and Reina, C. (1998). Scaling Clustering Algorithms to Large Databases. *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD98)*, pp. 9-15.

15. Breese, J.S., Heckerman, D., Kadie, C. (1998). Empirical Analysis of Predictive Algorighms for Collaborative Filtering. *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pp. 43-52.

16. Buckley, C., Salton, G., Allan, J., Singhal, A. (1994). Automatic query expansion using SMART: TREC 3. In *D. K. Harman, editor, Overview of the Third Text Retrieval Conference (TREC-3),* pp. 69-80.

17. Bush, V. (1945). As We May Think. *The Atlantic Monthly.* July, 1945.

18. Chien, Y.-H., and George, E. I. (1999). A Bayesian Model for Collaborative Filtering. *Proceedings of the Seventh International Workshop on Artificial Intelligence and Statistics.*

19. Chung, Y.-M., Pottenger, W.M., and Schatz, B.R. (1998). Automatic Subject Indexing Using an Associative Neural Network. *Proceedings of the ACM Conference on Digital Libraries (DL-99),* pp. 59-68.

20. Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D. and Sartin, M. (1999). Combining Content-Based and collaborative Filters in an Online Newspaper. *ACM SIGIR Workshop on Recommender Systems*, Berkeley, CA, August 19, 1999.

21. Cohen, W.W. (1996). Learning Rules that Classify E-mail. *Proceedings of the AAAI Spring Symposium on Machine Learning in Information Access.* AAAI Press.

22. Cohen,W.W., Basu,C., Hirsh,H. (1998). Using Social and Content-Based Information in Recommendation. *Proceedings of the AAAI-98*.

23. Condliff, M.K., Lewis, D.L., Madigan, D., Posse, C. (1999). Bayesian Mixed-Effects Models for Recommender Systems. (URL: www.stat.rutgers.edu/~madigan/PAPERS/collab.ps).

24. Cooley, R., Mobasher, B., and Srivastava, J. (1999). Data Preparation for Mining World Wide Web Browsing Patterns**.** *Knowledge and Information Systems* 1(1).

25. Cooper, W. S., Chen, A., & Gey, F. C. (1994). Full Text Retrieval Based on Probabilistic Equations with Coefficients Fitted by Logistic Regression. *Proceedings of the Second Text Retrieval Conference (TREC-2),* pp. 57-66.

26. Dahlen,B.J., Konstan,J.A., Herlocker,J.L., Good,N., Borchers,A., Riedl,J. (1998). Jump-starting movielens: User benefits of starting a collaborative filtering system with "dead data". University of Minnesota TR 98-017.

27. Decker, K.M., and Focardi, S. (1995). Technology Overview: A Report on Data Mining. *Technical Report*,(CSCS TR-95-02) Swiss Scientific Computing Center.

28. Denning, P.J. (1982). Electronic Junk. *Communications of the ACM.* 25(2), pp. 163-165.

29. Dugan, I.J. (1996). The Internet is the Great Equalizer. *Business Week, Enterprise Edition*, October 21, 1996. (URL: http://www.businessweek.com/1996/43/b349849.htm)

30. Dvorak, J. (1996). Avoiding Information Overload. *PC Magazine*. December 17, 1996. (URL: http://www.zdnet.com/pcmag/issues/1522/pcmg0022.htm)

31. Farber, M. (1994). The quality of information on the internet. *Computer Networks and ISDN Systems* 26, pp. 13-16.

32. Goldberg, D., Nichols, D., Oki, B. M. and Terry, D. (1992). Using Collaborative Filtering to Weave an Information Tapestry. *Communications of the ACM* 35(12), pp. 61-70.

33. Good, N., Schafer, J.B., Konstan, J., Borchers, A., Sarwar, B., Herlocker, J. and Riedl, J. (1999). Combining Collaborative Filtering with Personal Agents for Better Recommendations. *AAAI 99*, Orlando, FL.

34. Gordon, M.D., and Lenk, P. (1991). A Utility Theoretic Examination of the Probability Ranking Principle in Information Retrieval. *Journal of the American Society for Information Science*, 1991(42), pp. 703-714.

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

36. Herlocker, J.L., Konstan, J.A., Riedl, J. (2000). Explaining Collaborative Filtering Recommendations. *Proceedings of the 2000 Conference on Computer Supported Cooperative Work*.

37. Hill, W., Stead, L., Rosenstein, M. and Furnas, G. (1995). Recommending and Evaluating Choices in A Virtual Community of Use. *CHI 95*, Denver, CO, ACM Press, pp. 194-201.

38. Iconocast (2000). Iconocast Newsletter, 27-Jan-00.

39. Jain, A.K., and Dubes, R.C. (1998). Algorithms for Clustering Data. Englewood Cliffs, N.J., Prentice Hall Publishers.

40. Jones, K.S., and Willett, P. (eds) (1997). Readings in Information Retrieval. San Francisco, CA, Morgan Kaufmann Publishers.

41. Khamba, T, Bharat, K., and Albers, M.C. (1993). The Krakatoa Chronicle – An Interactive, Personalized, Newspaper on the Web. *Fourth International World Wide Web Conference*, pp. 159-170.

42. Kohavi, R., and Provost, F. (2001). Applications of Data Mining to Electronic Commerce. *Data Mining and Knowledge Discovery* 5(1/2), pp. 5-10.

43. Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J. L., Gordon, L. R. and Riedl, J. (1997). GroupLens: Applying collaborative filtering to Usenet news. *Communications of the ACM* 40(3), pp. 77-87.

44. Kowalski, G. (1997). Information Retrieval Systems: Theory and Implementation. Boston, MA, Kluwer Academic Publishers.

45. Lawrence, R.D., Almasi, G.S., Kotlyear, V., Viveros, M.S., and Duri, S.S. (2001). Personalization of Supermarket Product Recommendations. *Data Mining and Knowledge Discovery* 5(1/2) pp. 11-32.

46. Lee, W. S. (2000). Online Clustering for Collaborative Filtering. National University of Singapore, School of Computing Technical Report TRA8/00.

47. Lin, W.-Y., Alvarez, S.A., Ruiz, C. (2000). Collaborative Recommendation via Adaptive Association Rule Mining, *WebKDD-2000*, Boston, MA, Aug. 2000

48. Maltz, D., Ehrlich, K. (1995). Pointing the way: Active collaborative filtering. *Proceedings of the 1995 ACM Conference on Human Factors in Computing Systems*. New York.

49. Maes, P. (1994). Agents that Reduce Work and Information Overload. *Communications of the ACM* 37 (7), pp. 31-71.

50. Moukas, A., and Zacharia, G. (1997). Evolving a Multi-agent Information Filtering Solution in Amalthaea. *Proceedings of the First International Conference on Autonomous Agents*, pp. 394-403.

51. Moxy Früvous, (1999). My Poor Generation. Thornhill [Audio Recording]. The Bottom Line Record Company.

52. Nakamura, A., and Abe, N. (2000). Automatic Recording Agent for Digital Video Server. *Proceedings of the ACM Conference on Multi-Media (MM-00)*.

53. Obraczka, K., Danzig, P.B., and Li, S.H. (1993). Internet resource discovery services. *Computer* 26(9), pp. 8-22.

54. Park, J., Chen, M., Yu, P. (1995). An Effective Hash-based Algorithm for Mining Association Rules. *Proceedings of the ACM SIGMOD Conference on Management of Data,* pp. 175-186.

55. Pazzani, M.J. (1999). A Framework for Collaborative, Content-Based and Demographic Filtering. *Artificial Intelligence Review* 13(5/6), pp. 393-408.

56. Peppers, D. and Rogers, M. (1997). The One to One Future : Building Relationships One Customer at a Time, Bantam Doubleday Dell Publishing.

57. Pine II, B.J. (1993). Mass Customization. Boston, MA, Harvard Business School Press.

58. Pine II, B.J., and Gilmore, J.H. (1999). The Experience Economy. Boston, MA, Harvard Business School Press.

59. Pine II, B.J., Peppers, D. and Rogers, M. (1995). Do You Want to Keep Your Customers Forever? *Harvard Business School Review* 1995(2), pp. 103-114.

60. Pirroli, P., Pitkow, J., and Rao, R. (1996). Silk from a Sow's Ear: Extracting Usable Structures from the Web. *Proceedings of the ACM Conference on Human Factors in Computeing Systems (CHI-96).*

61. Reichheld, F.F. (1993). Loyalty-Based Management. *Harvard Business School Review* 1993(2), pp. 64-73.

62. Reichheld, F.F. and Sasser, W.E. (1990). Zero Defections: Quality Comes to Services. *Harvard Business School Review* 1990(5), pp. 105-111.

63. Reisman, S. (1995). I have seen the future: it is flooded with email. *IEEE Software* 12(3), pp. 111-112.

64. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. and Riedl, J. (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews. *CSCW 94*, pp. 175-186.

65. Resnick, P. and Varian, H.R. (1997). Recommender Systems. *CACM* 40(3), pp. 56-58.

66. Salton, G. (1968). Automatic Information Organization and Retrieval, McGraw-Hill Book Company.

67. Salton, G., Fox, E. A. and Wu, H. (1983). Extended Boolean Information Retrieval. *Communications of the ACM* 26(11), pp. 1022-1036.

68. Salton, G. and M.J., M. (1983). Introduction to Modern Information Retrieval, New York: McGraw-Hill Book Company.

69. Sarwar, B. M., Konstan, J. A., Borchers, A., Herlocker, J., Miller, B. and Riedl, J. (1998). Using Filtering Agents to Improve Prediction Quality in the GroupLens Research Collaborative Filtering System. *CSCW 98*.

70. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J. (2000). Analysis of Recommender Algorithms for E-Commerce. *Proceedings of E-Commerce 2000*.

71. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J. (2000). Application of Dimensionality Reduction in Recommender System—A Case Study. *WebKDD workshop* August 20, 2000.

72. Schafer, J.B., Konstan, J.A. and Riedl, J. (1999). Recommender Systems in Electronic Commerce. *ACM Conference on Electronic Commerce (EC-99)*.

73. Schafer, J.B., Konstan, J.A. and Riedl, J. (2001). Electronic Commerce Recommender Applications. *Data Mining and Knowledge Discovery* 5(1/2), pp. 115-153.

74. Schwab I., Pohl W. and Koychev, I. (2000). Learning to Recommend from Positive Evidence. *Proceedings of Intelligent User Interfaces 2000*, pp. 241-247.

75. Shardanand,U., Maes,P. (1995). Social Information Filtering: Algorithms for Automating Word of Mouth. *Proceedings of ACM CHI '95*, pp. 210-217.

76. Shneiderman, B. (1998). Designing the User Interface: Strategies for Effective Human-Computer Interaction. Addison Wesley Longman, Inc.

77. Tatemura, J. (1999). Visual querying and explanation of recommendations from collaborative filtering systems. *Proceedings of the 1999 international conference on Intelligent user interfaces,* p. 189.

78. Tauritz, D.R. and Sprinkhuizen-Kuyper, I.G. (1999). Adaptive Information Filtering: Improvement of the Matching Technique and Derivation of the Evolutionary Algorithm. *Technical Report 99-04, Leiden University*.

79. Terveen, L., Hill, W., Amento, B., McDonald, D. and Creter, J. (1997). PHOAKS: A System for Sharing Recommendations. *CACM* 40(3), pp. 59-62.

80. Ungar, L.H., Foster, D.P. (1998). Clustering Methods for Collaborative Filtering. *Proceedings of the 1998 Workshop on Recommender Systems,* pp. 114-129.

81. Uthurusamy, R. (1996). From Data Mining to Knowledge Discovery: Current Challenges and Future Directions. *Advances in Knowledge Discovery and Data Mining.* AAAI Press/MIT Press.

82. van der Puten, P. Data Fusion: A Way to Provide More Data to Mine in? Available from (http://www.wi.leidenuniv.nl/~putten/library/fusiebnaic00.htm)

83. Van Dyke, N.W., Lieberman, H., and Maes, P. (1999). Butterfly: A Conversation-Finding Agent for Internet Relay Chat. *Proceedings of the 1999 International Conference on Intelligent User Interfaces*, pp. 39-41.

84. Voorhees, E. M. and Tong, R. M. (1997). Multiple Search Engines in Database Merging. *The 2nd ACM International Conference on Digital Libraries,* pp. 93-102.

85. Wald, L. (1998). A European proposal for terms of reference in data fusion. *International Archives of Photogrammetry and Remote Sensing*, Vol. XXXII, Part 7, pp. 651-654.

86. Wasfi, A. (1999). Collecting User Access Patterns for Building user Profiles and Collaborative Filtering. *1999 Conference on Intelligent User Interfaces (IUI99)*, pp. 57-64.

87. Wheatley, M. (2000). Jeff Bezos takes everything personally. *CIO Magazine*. August 1, 2000. (URL: http://www.cio.com/archive/080100_bezos_content.html)

88. http://www.activebuyersguide.com

89. http://www.amazon.com

90. http://www.carsdirect.com

91. http://www.cdnow.com

92. http://www-datafusion.cma.fr/welcome.html

93. http://www.drugstore.com

94. http://www.ebay.com

95. http://www.excite.com

96. http://www.frictionless.com

97. http://www.gomez.com

98. http://movielens.umn.edu

99. http://movies.eonline.com

100. http://www.reel.com