The View through MetaLens: Usage Patterns for a Meta-Recommendation System

AUTHORS
J. Ben Schafer
Department of Computer Science
University of Northern Iowa
Cedar Falls, IA 50614-0507 USA
+1 319 273 2187
schafer@cs.uni.edu

Joseph A. Konstan
Computer Science and Engineering
University of Minnesota
Minneapolis, MN 55455USA
+1 612 625 4002
konstan@cs.umn.edu

John Riedl
Computer Science and Engineering
University of Minnesota
Minneapolis, MN 55455USA
+1 612 625 4002
riedl@cs.umn.edu

ABSTRACT

In a world where a person's 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. Such systems have become powerful tools in domains such as electronic commerce, digital libraries, and knowledge management. In this paper, we address such systems, as well as a relatively new class of recommender system called meta-recommenders. Meta-recommenders provide users with personalized control over the generation of a single recommendation list formed from a combination of rich data using multiple information sources and recommendation techniques. We discuss observations made from the public trial of a meta-recommender system in the domain of movies, and lessons learned from the incorporation of features that allow persistent personalization of the system. Finally, we consider the challenges of building real-world, usable meta-recommenders across a variety of domains.

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 [1]

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 whole host of journal articles, conference proceedings, textbooks, and web sites. During our personal time we must choose which television show to watch, which movie to see, which CD to play, or which book to read. The number of options from which to choose in each of these categories is often more than we can possibly process. While the Internet has been touted as "the great equalizer"[2], it has also 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. 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, ranging from ephemeral attribute matching to persistent personalized suggestions. Regardless of technique, these systems attempt to help users identify the items that best fit their needs, their tastes, or even both.

In this paper, we discuss a new class of recommendation interface known as *meta-recommendation systems*. These systems present recommendations fused from "recommendation data" from multiple information sources. Meta-recommendations systems encourage users to provide both ephemeral and persistent information requirements, so the resulting recommendations blend query-fit with long-term personalization. Furthermore, these systems provide a high level of user control over the combination of recommendation data, providing users with more unified and meaningful recommendations. In presenting meta-recommenders, we discuss data and observations obtained from the public release of a meta-recommendation system for the domain of movies. We consider the usefulness of the introduction of persistent personalization for individual users. Finally, we consider the challenges of building real-world, usable meta-recommenders across a variety of domains.

2. RELATED WORK

2.1 Recommender System Actions

According to Resnick and Varian [3], "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 provide a list of candidate items or recommendations. Estimation systems provide an estimate of user preference on specific items or predictions. Comment systems provide access to textual recommendations of members of a community.

In our work [4], we have extended this 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 systems include feature-search systems, which provide users with the ability to express explicitly an interest in items with a particular set of features. While the line between feature-search systems and keyword retrieval systems is a fine one, the distinction lies in the overall feel of the system. These "recommendations" serve as an important first step in the user's decision-making process. In this paper, we will use the term "recommender system" to refer to any system that provides a recommendation, prediction, opinion, or user-configured list of items that assists the user in evaluating items.

2.2 Recommender System Algorithms

Although the algorithms used within these systems vary, most are based on one or more of three classes of technology: data mining, information filtering and retrieval, and collaborative filtering.

The term data mining refers to a broad spectrum of mathematical modeling techniques and software tools that are used to find patterns in data. Recommender systems that incorporate data mining techniques make their recommendations using knowledge learned from the actions and attributes of users. These systems are often based on the development of user profiles that can be persistent (based on demographic or item "consumption" history data), ephemeral (based on the actions during the current session), or both. While recommender systems using data mining techniques are common in the domain of e-commerce [4], these techniques are not used in the present research, and no exemplars are discussed.

The earliest "recommender systems" were information filtering and retrieval systems designed to fight information overload in textual domains. Recommender systems that incorporate information retrieval methods are frequently used to satisfy ephemeral information needs from relatively static databases. Conversely, recommender systems that incorporate information filtering (IF) methods are frequently used to identify items that match relatively stable and specific information needs in domains with a rapid turnover or frequent additions. Although information retrieval and information filtering are considered fundamentally different tasks [5], they are based on similar techniques. In this paper we will consider both under the singular term "information filtering."

Without computers, we often receive recommendations by listening to what those around us have to say. If many people in my office state that they enjoyed a particular movie, or if a member of my book club who I tend to agree with suggest a given book, then I treat these as recommendations. Collaborative filtering (CF) is an attempt to facilitate the process of "word of mouth." Users provide the system with evaluations of items that may be used to make "recommendations" to other users. The simplest of CF systems provides generalized recommendations by aggregating the evaluations of the community at large. More advanced systems personalize the process by forming an individualized neighborhood for each user consisting of a subset of users whose opinions are highly correlated with those of the original user.

2.3 Recommender System Applications

The algorithms discussed in the previous section have been employed in the creation of numerous research and commercial recommender systems that demonstrate that such systems are powerful tools for helping connect users with items of interest. In this section we will attempt to demonstrate the scope of these techniques through a brief review of several of these applications. A complete review of the broad range of recommender system applications is beyond the scope of this article, and several articles and books have already been written on this topic [4] [6].

Recommender systems employing information filtering techniques often do so through the use of IF agents. Operating in the domain of Usenet news, NewT [7] employs a vector-space-based genetic algorithm to learn which articles should be selected and which should not. Ripper [8] and RE:Agent [9] use learning techniques to classify e-mail based on a user's prior actions. Amalthaea [10] is a multi-agent system for recommending information sources on the Internet. Information filtering agents keep track of a user's interests while information discovery agents search and retrieve documents matching the user's interest profile.

Commercial applications of IF-based recommender systems include library and clipping services, such as webclipping.com, which use keyword searches of online newspaper, magazines, Usenet groups, and web pages to deliver recommended information to customers.

Recommender systems based on collaborative filtering have produced recommendations in a variety of domains. Operating on email and Usenet news postings, Tapestry [11] allows users to identify other users whose knowledge should be trusted (e.g. "show all books on 'agents' in which Nathan's evaluation contains 'outstanding'"). These rules actively establish a neighborhood for recommendations. The original GroupLens project [12] provides automated neighborhoods for recommendations in Usenet news. Users rate articles, and GroupLens automatically recommends other articles to them. This work was expanded to include the release of MovieLens (movielens.umn.edu), a CF system for the domain of movies [13]. Ringo [14] uses CF techniques to provide users with recommendations about audio CDs. In addition, Ringo has support for message boards – independent of the recommender system – on which users can discuss their music tastes. Finally, while the previous examples rely on explicit ratings, PHOAKS [15] uses implicit ratings to create a recommender system by examining Usenet news postings to find "endorsements" of web sites and creating a listing of the top web sites endorsed in each newsgroup.

Commercial applications of CF-based recommender systems include e-commerce sites, such as Amazon.com, which use implicit recommendations via purchase history and/or explicit recommendations via "rate it" features to generate recommendations of products to purchase.

2.4 Hybrid Recommender Systems

As researchers have studied different recommender system technologies, many have suggested that no single technology works for all situations. Thus, hybrid systems have been built in an attempt to use the strengths of one technology to offset the weaknesses of another. Burke [16] discusses several different hybridization methods, but points out that most hybrid systems involve the combination of collaborative filtering with either a content-based (IF) or data mining technique. While a thorough discussion of such systems is beyond the scope of this article, and has already been completed [16], we will present a simple introduction to hybrid systems in this section.

Tango [17] recommends articles in the domain of an online newspaper. It does so by creating separate recommendations from CF and IF algorithms and merging these using a separate combination filter. Tango's collaborative filter provides 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.

Torres et al. [18] present the results of several experiments involving TechLens. Similar to Tango, TechLens combines both a collaborative filter and a content-based filter to recommend research papers. In both offline and online studies they consider five different algorithms for combining the recommendations from these filters, including sequential algorithms. These techniques take the recommendations from one filter as a seed to the second filter. They conclude that different algorithms should be used for recommending different kinds of papers, although they discovered that sequential algorithms tend to produce poor results under most circumstances.

The SmartPad supermarket product recommender system [19] 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.

Nakamura and Abe [20] 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 that 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 user takes the final action. With a personalized video system, the video recorder takes the action. 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?).

Commercial applications of hybrid-based recommender systems include search tools such as Google (www.google.com) that combines results of both content searches (keyword analysis) and collaborative recommendations (authoritativeness). Brin and Page [21] calculate the "authoritativeness" of page p_t by summing the authoritativeness of those pages in set P which link to $p_{t.}$ The more pages that link to p_t, and in turn, the higher the authoritativeness of these pages, the higher the authoritativeness score for p_t.

Our work builds on the hybrid systems by developing a recommender framework that is able to incorporate inputs from multiple sources simultaneously. The sources we consider are covered by the hybrid systems, but since each hybrid system is developed to work with a closed set of source, no single system incorporates as much data on the items being recommended. Further, the meta-recommender framework we propose is flexible to the emergence of future data sources.

2.5 Recommending in Other Domains

While a traditional CF-based recommender requires users to provide explicit feedback, a social navigation system attempts to mine the social activity records of a community of users to implicitly extract the importance of individuals and documents [22]. Such activity may include Usenet messages, system usage history, citations, or hyperlinks. TopicShop [23] is an information workspace which allows groups of common websites to be explored, organized into user defined collections, manipulated to extract and order common features, and annotated by one or more users. These actions on their own may not be of large interest, but the collection of these actions can be mined by TopicShop and redistributed to other users to suggest sites of general and personal interest. Agrawal et. al [24] explored the threads of newsgroups to identify the relationships between community members. Interestingly, they concluded that due to the nature of newsgroup postings – users are more likely to respond to those with whom they disagree – "links" between users are more likely to suggest that users should be placed in differing partitions rather than the same partition. Although this technique has not been directly applied to the construction of recommendations, such an application seems a logical field of future study.

Although traditional recommenders suggest what item a user should consume they have tended to ignore changes over time. Temporal recommenders suggest when a recommendation should be made or when a user should consume an item. Adomavicius et. al [25] suggest the construction of a recommendation warehouse which stores ratings in a hypercube. This multidimensional structure can store data on not only the traditional user and item axes, but also for additional profile dimensions such as time. Through this approach, queries can be expanded from the traditional "what items should we suggest to user X" to "at what times would user X be most receptive to recommendations for product Y." Hamlet [26] is a system designed to minimize the purchase price of airplane tickets. Hamlet combines the results from time series analysis, Q-learning, and the Ripper algorithm to create a multi-strategy data-mining algorithm. By watching for trends in airline pricing and suggesting when a ticket should be purchased, Hamlet was able to save the average user 23.8% when savings was possible.

Finally, Miller et. al. consider the challenges inherent when converting systems which are normally tied down to the desktop to systems for our increasingly mobile society. They discuss an interface that helps users of MovieLens take their recommendations with them through the use of a recommender system for the occasionally connected PDA [27].

The work in this paper is focused on recommending in traditional domains, not the emerging domains described in this subsection. However, we see the long-term impact of meta-recommenders as extending beyond traditional domains to include many of these emerging domains. For instance, meta-recommenders will have to be developed to incorporate temporal information, or to operate in the limited environment of a PDA.

3. METALENS

3.1 THE NEED FOR META-RECOMMENDERS

Consider the following scenario. Mary's 8-year-old nephew is visiting for the weekend, and she would like to take him to the movies. Mary has several criteria for the movie that she will select. She would like a comedy or family movie rated no "higher" than PG-13. She would prefer that the movie contain no sex, violence or offensive language, last less than two hours and, if possible, show at a theater in her neighborhood. Finally, she would like to select a movie that she herself might enjoy.

Traditionally, Mary might decide which movie to see by checking the theater listings in the newspaper and asking friends for recommendations. More recently, her quest might include the use of the Internet to access online theater listings and search databases of movie reviews. Additionally, she might be able to obtain personalized, CF-based recommendations from a web site such as MovieLens. Producing her final selection, however, requires a significant amount of manual intervention; Mary must visit each source to gather the data and then decide how to apply this data in making her final decision.

The hybrid systems mentioned in the previous section are a significant step toward solving problems like Mary's. A hybrid movie recommendation system would provide Mary with lists of movies blended from her long-standing collaborative filtering and content-interest profiles. It is likely, however, that such a system would not offer her the ability to provide information that might improve the recommendations produced by the combination algorithm. For example, if given access to the combination algorithm, Mary could indicate that predictions should be biased less towards the British art films she frequently likes and more toward the family movies appropriate for her nephew, or that the movie should be relatively free of offensive language and last less than two hours.

Similar situations can be found across a variety of domains. A consumer can use dozens of sources to gather a variety of attribute data and opinions regarding a product. Internet users browsing for information on a given topic can try any number of search engines, each of which uses a slightly different mechanism for determining the "top recommendations." Knowledge workers would like to combine a variety of techniques including keyword analysis, citation analysis, and the recommendations of other users to select appropriate documents. Even the addition of hybrid systems does not completely solve the problem. A user can't tell Google to weigh the currency of the web pages more highly in a search for "world cup results" even if currency may be part of the underlying algorithm. Thus, the user may be forced to process the recommendations manually in order to weed out the results from previous years. Similarly, a user may find it difficult to tell his hybrid recommendation newspaper that, while he was interested in the first dozen articles about the latest political scandal, he isn't really interested in reading any more.

In prior work [28] we have defined a new form of hybrid system with the level of user control needed to allow for the meaningful blending of recommendations from multiple techniques and sources. These systems, known as meta-recommenders, provide users with personalized control over the generation of a single recommendation list formed from a combination of rich data using multiple information sources and recommendation techniques. Based on the lessons we learned from existing hybrid systems, we built the MetaLens Recommendation Framework (MLRF), a general architecture for the construction of meta-recommenders. Using this framework, we implemented MetaLens, a meta-recommender for the domain of movies. Much like Mary, who makes her final choice by examining several movie data sources, MetaLens uses IF and CF technologies to generate recommendation scores from several Internet film sites. In the remainder of this section, we will briefly explain MLRF and discuss the personalization features added prior to the first public trial of MetaLens.

3.2 The MetaLens Recommendation Framework

The MetaLens Recommendation Framework (MLRF) (Figure 1) is an architectural framework within which multiple meta-recommenders can be constructed. It does so through a three-layer process. We will explain this process through the context of MetaLens for movies.

The user interface for MetaLens centers on two screens. On the preferences screen, users indicate their ephemeral requirements for their movie search. They do this by providing information concerning nineteen features of movies and theaters including genre, MPAA rating, critical reviews, and distance to the theater (Table 1). For each feature the user may indicate the specific factors he considers important (e.g., "I want to see a film from the ‘comedy’ or ‘family’ genre"), a weight that indicates how important it is that the recommended movie matches these factors (e.g., "It is very important that the movie I see be one of the genres I selected") and a “Display Info?” selection which indicates that data related to the specific feature should be included with the recommendations. As an example, Figure 2 might represent a portion of Mary's requirements for the movie that she views with her nephew. When Mary submits her preferences, the interface layer validates the information provided, formats it, and transfers control to the computation layer.

Prior to making any computation, the computation layer requests that the data layer produce the appropriate information concerning the theater/movie/show time triples for the user's location. The data layer gathers the information either from a local cache or through runtime data acquisition using three sources – Yahoo Movies, Rotten Tomatoes, and MovieLens. Yahoo Movies (movies.yahoo.com) provides information concerning movies and theaters including genre, MPAA rating, content, show times, and theater location. This data is relatively static; for instance, a movie’s MPAA rating should never change, and show times at a particular theater tend to be modified on a regular schedule typically centered on the Friday release of new movies. Because of this, data from Yahoo Movies is gathered is gathered offline by several cached data modules which collect information on regular, but potentially independent, schedules to account for newly added entities and allow for potential changes in known entities. Similarly, a cached data module gathers data from Rotten Tomatoes (www.rottentomatoes.com) regarding critical review information, including the number of critics rating the movie and the percentage of favorable reviews. Finally, MovieLens provides personalized prediction information on a user/movie basis. Since these predictions may change at any time based on new or modified data by other users of the system, a runtime acquisition module acquires this data. The Rotten Tomatoes and MovieLens modules must also negotiate a data fusion process to coordinate their data with that extracted from Yahoo Movies. While each of these three sites lists the title of each movie, we must resolve variations in title format ('The In-Laws' vs. 'In-Laws, The') and different releases of movies with the same name (Is that the 2003 or the 1979 version of 'The In-Laws'?).

Once all of the data is gathered, it is returned to the computation layer. The algorithm employed by the computation layer is based on the extended Boolean information retrieval algorithm proposed by Salton et al [29] 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, 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. In essence, Mary 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 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.

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

The value of d_a is calculated as follows:

- For features that match a requirement on a single option (i.e. “the movie theater should be offering discounted tickets for the particular showtime”), we chose to represent each item by a binary score of 1 or 0. Realistically, this choice may not be the best option for all features. 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. We could imagine the use of a “decay function” which reduces the score from 1 to 0 as the runtime increases above 130 minutes. However, this involves an analysis of user expectations we have not yet explored.
- 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.

MetaLens judges overall query fit based on recommendation scores from these multiple data sources. No attempt is made to resolve potential information conflicts. Instead, each piece of data is converted as-is, and the item match scores combined to calculate a query-fit score for each triple. These are returned to the interface layer where the recommendations are sorted to contain only the highest-rated triple for each movie – each movie is recommended once in conjunction with the theater and show time that best fits the user's requirements – and the final recommendations displayed. Thus, according to

Figure 3, MetaLens recommends that Mary should take her nephew to see the 4:50 showing of “Kangaroo Jack” at the CEC-Crossroads 12 Theater complex.

Users may obtain additional information about any of the recommended movies or theaters by selecting the hyperlink of the item in question. This spawns a separate browser window containing information about the item. Furthermore, results may be "tweaked" by the user who may modify the requirements or weights for each feature, thus modifying which subset of features is considered optimal.

3.3 Persistent Personalization Features

In addition to the base design of MLRF, which allows for the ephemeral personalization of the recommendation request, the publicly tested version of MetaLens contains several features allowing users to configure the system for persistent personalization. These consist of membership-level personalization and query-level personalization.

Membership-level personalization focuses on data which is expected to remain relatively constant from session to session. Although users may modify this data at any time, doing so involves an interface separate from the base meta-recommender. Data gathered for membership-level personalization include the US ZIP code for which the user wishes to receive recommendations, theaters in that ZIP Code that he wishes to exclude from recommendations, and the number of recommendations to display (currently “top-10” or “all”).

Query-level personalization focuses on data that may change from session to session. When a user first uses MetaLens, each of the features on the preferences screen is set to its “MetaLens Default” value. This default setting consists of “all off” or “all on” when a selection list is needed or an “average” value when numerical input is needed, has the importance of every feature set to a mid-range “sort of important” value, and has the “Display Info?” selection turned off for all features. Users may easily modify these settings to represent their requirements for the recommendations. As previously demonstrated, Mary can configure the preferences screen to indicate the ephemeral requirements she has when taking her nephew to the movies. But what if this situation is a common occurrence? Mary doesn’t want to adjust repeatedly this default query into one that actually represents her needs. Upon submitting any query, Mary has the option of saving the preference values as a query profile under a name of her choice. Profiles can be selected on future visits and submitted as-is or tweaked to fit the specific situation. Profiles can be adjusted and saved again, or can be saved as new variations. Finally, Mary may even indicate which query profile she wishes to use as her personal default (Figure 4).

3.4 The Accuracy of Meta-recommenders

While this paper discusses several issues concerning the use of MetaLens, it will not discuss the “accuracy” of the system nor compare it to any of the benchmarks set forth by other recommendation algorithms. This choice was made as a conscious decision. It is unclear how you would evaluate the "accuracy" of MetaLens given existing data. For example, a traditional recommender system in the domain of movies needs to compare whether the movies that it is recommending to a user are ones the user would agree are "good" movies. Since these user opinions are frequently inputs to traditional recommenders, this data exists, and traditional "train/test" techniques can be applied. However, the recommendations from MetaLens predicts how well a movie, theater, and specific showtime will fit a user's overall profile for what it is they are looking for. Existing data sets do not provide user ratings in a manner consistent with this recommendation approach. We are in the process of gathering data that would allow us to perform various accuracy analyses.

4. Analysis of Public Usage of MetaLens

Our previous work was designed to consider the interfaces for meta-recommenders and controlled user studies suggested that users found these systems more helpful than "traditional" systems. With this work we wanted to consider what happened when users were given free reign with a public meta-recommender. To address this interest, we conducted an initial public trial of MetaLens for movies. During the first eight weeks in which MetaLens was offered as a part of the MovieLens website[1], 838 users "registered" and submitted 1668 queries to MetaLens. The majority of these 838 users visited MetaLens only once during the 8-week period (Table 2). 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[2]. Of the 1668 total queries submitted to MetaLens, 603 (36.2%) were submitted by active users. In considering how users interacted with MetaLens, we are particularly interested in which movie/theater features users consider important, and if users would take advantage of personalization features. In the remainder of this section we will focus on data and observations related to these issues.

4.1 Movie/Theater Features

Assume that all users have approximately the same requirements for selecting a movie. If this were the case, there would be no need for the level of personalization provided by meta-recommenders. Designers could simply build a system to look for the exact set of features that all users want. Prior work, however, has indicated that users prefer personalized control of the combination of

[1] Since much of the power of MetaLens derives from its inclusion of personalized movie predictions from MovieLens, MetaLens has been initially released as a feature of the MovieLens website.

recommendation data [28]. We have proposed that there are two differing but complementary reasons for this. First, different users have very different interests and requirements. While theater location may be a deciding factor for one user it may have little to no impact for another user. Second, what is important to a user today may not be important tomorrow. That is, while a user may be unwilling to drive 30 minutes with his children, he may be willing to do so when it is just him and his spouse.

Because of this, we designed MetaLens with a large and diverse set of movie and theater features. We were interested in which of these features users consider important. It was our belief that this large set of data provided users with a better system and that each of the pieces of recommendation data we made available to users would be considered important at some point to some user.

Table 3 summarizes the weight assigned to each of the nineteen feature across 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. Among active users we get fairly similar results (Table 4). Seventeen of the nineteen features received the highest weight available in at least one query, while fifteen received this weight in at least 1% of the queries.

On the surface, these results would appear to validate our belief that providing a large set of data is advantageous since every piece of data is considered important to some segment of users. At the same time, we note that, over all users, the most commonly provided weight for eighteen of the nineteen features was the default value set by the system. Furthermore, seventeen of the nineteen features had the zero weight as the second most common weight. Among active users, only eight of the features have the default weight as the most commonly-provided value, but nine of the features were most commonly set to zero weight. In other words, all users, and particularly active users, found many of the features to be irrelevant to their query.

To consider this from a different direction we 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)? Over all users, fifteen of nineteen features were more frequently lowered than raised (each of these is statistically significant with z scores ranging from –10 to –41). Among active users, sixteen of nineteen features were more frequently lowered than raised (z scores ranging from –4 to –34). An

[2] For comparison, during the three calendar-month period that encompassed this study, 4724 users visited

active user is nearly three times as likely to downgrade a features weight from the default weight as she is to raise it (statistically significant difference, $p<.05$). This may suggest that users are mostly happy with the relative weights we assign to the features they care to use, but want to eliminate many of the features from the ranking.

A third way of analyzing "importance" is to consider which features users select to display with their recommendations. The assumption that we can make is that if a user selects the “Display Info?” option for a particular feature, they consider that piece of data to be an important part of their decision making process. The results of this analysis are in Table 5. Observe that for all but the “special” feature, at least 10% of the queries submitted by both all users and active users asked to have the feature included in the final recommendation table.

Regardless of which “metric” we use to analyze the data, it would appear that our initial beliefs were justified. Although many of these movie and theater features were considered unimportant by a large segment of the user base, each item was considered important, even a "must," by a different segment. Potential implications of this are explored more in sections 5 and 6.

4.2 Query Profiles

We previously described MetaLens’ ability for users to create and save queries for later retrieval. These "query profiles" become an important piece of the personalization 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."[3] While the weights and selections within that request may change slightly over time, the "base" of the request is the same (a movie rated no higher than PG-13 and over by 9:30). Query profiles allow users to configure the system to save these individual base preferences and even set up profiles for different daily moods. In doing so, they improve the end system by reducing the amount of future effort required to get recommendations.

Of the 838 users who submitted at least one query to MetaLens, 278 (33%) of them established 355 profiles. For future reference, we will refer to these users as "Power Users." Table 6 lists the distribution of number of profiles created by power users. On initial inspection, this seems to be a

MovieLens. Of these, 724 visited the site during three or more sessions (15.2%).

[3] Indeed, we subsequently added the ability to save named queries to the basic features in MovieLens. This feature is quite popular.

particularly poor result - only one third of all users created a query profile. However, further analysis reveals some interesting observations.

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. The increased likelihood of more frequent users becoming power users is statistically significant ($p < .05$). 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 are able to conclude that active users are more likely to be power users.

Similarly, we are able to conclude that power users are more likely to be active users. Of the users who created one or more query profiles on their initial visit, 24.8% of these power users came back to use the site on at least one other visit. This is nearly twice the return rate of those who do not become power users (12.8%) with statistical significance. While 16% of power users become active users, this rate is only 7% among users in general.

5. CAN YOU HAVE ACCESS TO TOO MUCH DATA?

While we consider our hypothesis from section 4.1 to be valid, 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 3). In light of this observation, it seems relevant to ask, can you have access to too much data?

In an effort to address this issue a sampling of MetaLens "users" were sent email invitations to participate in an online survey. Of those sent invitations, 26 subjects accepted and completed the two-part survey. These users consisted of 10 active users and 16 non-active users.

In part one, subjects were asked several questions concerning their MetaLens usage. Users were asked 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.

Another question 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. User responses are summarized in Table 8. Interestingly, of the eleven users who provided this response, six also indicated that the data used by MetaLens was a strength.

Part two of this survey provided users with access to demonstration versions of two new MetaRecommenders built within the MetaLens Recommendation Framework – MetaLite and MetaClick. **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. MovieLens, Genre, Cream %, Not Seen, and Distance were the five features receiving the highest average weight. If we consider the features receiving the most “Must” votes, 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. 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 (e.g. “chick flicks,” “date night,” “movie with the kids”). Each scenario is connected with a system-defined “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. 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.

Users were given a brief description of each system, were allowed to interact with each at will, and were asked several questions concerning their perceived use of the systems. For each system, subjects were asked to indicate if they would be likely to use the system. While 23 of the 26 (88.4 %) indicated that they would use MetaLite, only 18 (69.2%) indicated they felt they would use MetaClick. Subjects were then presented with three system-vs-system comparisons, and asked to estimate which system they would be most likely to use. The results of this comparison are presented in Table 9. Finally, users were given the opportunity to comment on each system. Although few uses took advantage of this opportunity, comments concerning MetaLite were relatively evenly split between those who felt that it took away what they loved about MetaLens (access to all that data), and those who felt MetaLite improved on MetaLens by simplifying the process.

In an effort to better analyze the results of the pairwise comparisons summarized in Table 9, 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 10.

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.

6. DESIGN IMPACT OF LESSONS LEARNED

When considering the observations discussed in sections 4 and 5, we have identified several lessons learned that have a variety of implications for designers of future meta-recommenders.

When first designing MetaLens, we proposed that such a system is helpful because it assists users in consolidating and evaluating a large quantity of recommendation data. As such, we argued that it seems relevant to provide users with access to as much recommendation data as possible and let them configure the system to best fit their needs. Sections 4.1 and 5 showed us that while this belief is likely correct, designers must tread lightly around the belief that “more is better.” While it is true that users have a variety of needs, in which case building a system with access to a large variety of data may be beneficial, a significant segment of users seemed more interested in a system with a limited amount of data. While the data isn’t conclusive, it could be speculated that those who didn’t return to use the system did so because they found it too overwhelming. Yet, those who were frequent users of the system seemed to do so specifically because of the large amount of recommendation data available. These results suggest that future systems should consider a design that allows users to “start small.” That is, systems should provide access to the core set of data that is deemed to be the bare bones data required to make a useful recommendation. However, such systems must be easily modified to increase the amount of data used in their recommendations to allow active users access to the diverse data they seem to desire. Such systems may take the format of common search engines – providing a simple search mechanism with “advanced features” hidden one layer down from the main search – or may take the format of a “plug and play” system such as the modular systems described in the next section.

Statistical analysis suggests both that power users are more likely to become active users and that active users are more likely to become power users. It is difficult to conclude which is the cause and which is the effect without specifically studying the return rates of those without access to query profiles. However, it seems apparent that there is a correlation between the tendency to personalize a system and the frequency with which one uses the system. One reason electronic commerce sites implement recommender applications is in an effort to increase consumer loyalty to the site [4]. This being the case, we may conclude that if personalization capabilities increase the likelihood of a user being a repeat user, they may also increase the user's loyalty to the site.

7. FUTURE WORK

We are interested in several areas of future work concerning meta-recommenders. These include the transfer of meta-recommenders to other domains, the future role of personalization, and additional interfaces for meta-recommendation systems.

While results indicate that meta-recommenders are a promising class of recommender systems, these results are based on a system limited to the domain of movies and whose users are experienced MovieLens users and thus may not represent users at large. The MLRF was designed to be domainneutral, but it remains to be seen whether the benefit and acceptance of such systems will transfer to other domains. While we expect our results to generalize to other domains, including e-commerce, web search engines, and knowledge management, we must consider how the design of meta-recommenders may change based on the domain. For example, what adaptations have to be made in domains where item-features are less objective? For example, in the domain of books would features such as Library of Congress classification, genre, keyword, and author be helpful? Our initial expectation is that meta- recommenders will be most useful in domains with a wide variety of objective information about the content, and in which that objective information is important to users' decisions. Thus, while we would expect a meta-recommender in the domain of books to be valuable, we suspect that users may find meta-recommenders even more valuable in domains such as automobiles or technical reports.

How do such systems handle access to privileged data? 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. An additional protocol would need to be added to the MetaLens framework allowing for the acquisition of data by a registered user. For example, a meta-recommender in the domain of automobiles might allow for the integration of recommendations from Consumer Reports only for those users with memberships at consumerreports.com.[4]

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. The creation of "modular" systems would allow users to build custom recommenders based on the features each user feels is important. The creation of such a system, however, is not expected to be a trivial matter. The recommendation framework will need to be modified to accommodate easy and quick changes to the overlying system. Similarly, it is not uncommon for users of recommender systems to request that "Feature X" be added into the system. A recommender system using the MLRF should be able to handle inputs from any recommendation data source as long as the input is provided in a format that can be easily fused with the base data. The

[4] We should make it clear that current US copyright law and terms of use for web sites would prevent non-research systems from directly "scraping" content from movie information sites as we've done. Some public sites do provide interfaces for such queries, and other may be willing to do so in exchange for a link back to the site, but the challenge of supporting access to individually subscribed or licensed data is likely to be an important one.

framework would need to be modified, however, to allow for the dynamic input of data and the recommendation algorithm adjusted to handle a varying number of recommendation features.

Finally, we are interested in how the recommendation process might change as we make modifications to the various interfaces with which users interact. One of the advantages of meta-recommenders is that they involve such a rich assortment of recommendation data. While the present interface matches other similar interfaces, user interface design experts like Shneiderman [30] would argue that the current interface does not allow users to interact properly with the data since users cannot directly see a query's affect on the recommendation. We are very interested in the affect that "dynamic query" interfaces may have on the way users interact with a meta-recommender.

8. CONCLUSIONS

In this paper we have discussed meta-recommenders – a new way to help users find recommendations that are understandable, usable, and helpful. We have discussed observations made from the public release of a meta-recommender system in the domain of movies, and lessons learned from the incorporation of features that allow persistent personalization of the system. These include the observation that giving users access and personalized control over a wide variety of data may make for more meaningful recommendations, and that features which allow for persistent personalization of meta-recommenders may generate more loyal users. All told, we feel these results provide a meaningful foundation for the design of future meta-recommenders.

9. REFERENCES

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

[2] 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)

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

[4] Schafer, J.B., Konstan, J.A., and Riedl, J. (2001). E-Commerce Recommendation Applications. *Data Mining and Knowledge Discovery* 5(1/2) pp.115-153.

[5] Belkin, N.J. and Croft, W.B. (1992). Information Filtering and Information Retrieval: Two Sides of the Same Coin? *CACM* 35(12) pp.29-38.

[6] Riedl, J., Konstan, J.A, and Vrooman, E. (2002). Word of Mouse: The Marketing Power of Collaborative Filtering. Warner Business Books.

[7] Maes, P. (1994). Agents that Reduce Work and Information Overload. *CACM* 37(7) pp.31-40.

[8] Cohen, W.W. (1996). Learning Rules that classify E-mail. *Proceedings of the AAAI Spring Symposium on Machine Learning on Information Access*.

[9] Boone, G. (1998). Concept Features in RE:Agent, an Intelligent Email Agent. *Proceedings of Autonomous Agents 98*. pp.141-148.

[10] Moukas, A. and Zacharia, G. (1997). Evolving a Multi-agent Information Filtering solution in Amalthaea. *Proceedings of Autonomous Agents 97* pp.394-403.

[11] Goldberg, D., Nichols, D., Oki, B.M., and Terry, D. (1992). Using Collaborative Filtering to Weave an Information Tapestry. *CACM* 35(12) pp.31-70.

[12] 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.

[13] 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.

[14] Shardanand, U. and Maes, P. (1995). Social Information Filtering: Algorithms for Automating Word of Mouth. *Proceedings of CHI-95* pp.210-217.

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

[16] Burke, R. (2002). Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*, v. 12, pp 331-370.

[17] Claypool, M., Gokhale, A., and Miranda, T. (1999). Combining Content-Based and Collaborative Filters in an Online Newspaper. *ACM SIGIR Workshop on Recommender Systems*.

[18] Torres, R., McNee, S., Abel, M., Konstan, J.A., and Riedl, R. (2004). Enhancing Digital Libraries with TechLens. *Proceedings of JCDL'04.* pp 228-236.

[19] Lawrence, R.D., Almasi, G.S., Kotlyar, 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.

[20] Nakamura, A. and Abe, N. (2000). Automatic Recording Agent for Digital Video Server. *Proceedings of MM-00* pp.57-66.

[21] Brin, S. and Page, L. (1998). The Anatomy of a Large-scale Hypertextual Search Engine. *Computer Networks and ISDN Systems*, 30(1-7), pp 107-117.

[22] Munro, A., Hook, K., and Benyon, D. (editors) (1999), Social Navigation of Information Space, Springer Verlag.

[23] Amento, B., Terveen, L., Hill, W., Hix, D., & Schulman, R (2003). Experiments in Social Data Mining: The TopicShop System, *ACM Transactions on Computer-Human Interaction*, 10, 1, pp 54-85.

[24] Agrawal R., Rajagopalan, S., Srikant, R., & Xu, Y. (2003). Mining newsgroups using networks arising from social behavior. *Proceedings of the Twelfth World Wide Web Conference*. pp. 529-535

[25] Adomavicius, G., & Tushilin, A. (2001). Extending Recommender Systems: A Multidimensional Approach. *IJCAI-01 Workshop on Intelligent Techniques for Web Personalization* (ITWP'2001).

[26] Etzioni, O., Knoblock, C.A., Tuchinda, R., & Yates, A. (2003). To Buy or Not to Buy: Mining Airfare Data to Minimize Ticket Purchase Price, *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp 119-128.

[27] Bradley N. Miller, Istvan Albert, Shyong K. Lam, Joseph A. Konstan, John Riedl. (2003). MovieLens Unplugged: Experiences with an Occasionally Connected Recommender System. *Proceedings of ACM 2003 International Conference on Intelligent User Interfaces (IUI'03)*

[28] Schafer, J.B., Konstan, J.A., and Reidl, J. (2002). Meta-recommendation Systems: User-controlled Integration of Diverse Recommendations. *Proceedings of CIKM-02* pp. 196-204.

[29] Salton, G., Fox, E., and Wu, H. (1983). Extended Boolean Information Retrieval. *CACM* 26(11) pp.1022-1036.

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

10. TABLES

	Meaning
Avg User	An aggregate score from user comments at Yahoo Movies
Content	Decreases the rating of movies containing selected objectionable content
CreamMin	The number of reviews from "cream of the crop" critics at RottenTomatoes
Cream %	The percent of "cream of the crop" critics liking the movie
CriticMin	The # of total reviews at RottenTomatoes
Critic %	The percent of the critics liking the movie
Discount	Gives preference to showtimes where the tickets are at a discount
Distance	The distance from a user's ZIP code to the theater
Distributor	The distributor of the film
End Time	What time the movie ends
Genre	Genre of the film
MaxLength	Movies should not exceed this amount of time
MinLength	Movies should be at least this amount of time
MovieLens	Gives preferences to movies that the user is expected to like
MPAA	What the MPAA rates the movie
Not Seen	Will not recommend movies the user has seen (rated in MovieLens)
Release	How recently the movie was released
Special	Is the theater equipped to handle handicapped or hearing impaired patrons
Start Time	What time does the movie start

Table 1: Movie/Theater features and their meaning

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 2: Number of sessions per user.

Movie/ Theater Feature	User Selection of Feature Weight						Avg. Weight	Direction when default weight is modified	
	0	0.25	0.5 default	0.75	1	Must (1)		Lowered	Raised
MovieLens	11	9	513	387	748	NA	0.78	1.7%	98.3%
Not Seen [5]	101	17	622	133	273	439	0.70	12.3%	87.7%
Genre	227	119	839	201	158	124	0.53	41.7%	58.3%
Cream %	274	126	827	181	142	118	0.50	47.6%	52.4%
Distance	319	175	932	123	79	40	0.43	66.5%	33.5%
Critic %	368	136	910	152	63	39	0.42	67.1%	32.9%
End Time	406	42	1033	76	85	24	0.42	69.3%	30.7%
Discount	480	30	932	115	49	62	0.40	70.8%	29.2%
Start Time	448	63	987	72	78	20	0.40	75.8%	24.2%
Avg User	355	278	833	168	34	NA	0.39	74.5%	25.5%
MPAA	500	108	852	67	56	85	0.39	75.0%	25.0%
CreamMin	496	94	933	66	55	24	0.37	80.3%	19.7%
Release	548	61	960	78	8	13	0.34	84.3%	15.7%
MinLength	576	88	880	54	52	18	0.34	86.0%	14.0%
CriticMin	536	101	955	30	30	16	0.34	89.3%	10.7%
Content	634	43	923	24	26	18	0.32	90.9%	9.1%
MaxLength	664	106	822	30	30	16	0.30	91.0%	9.0%
Distributor	782	47	783	5	5	44	0.27	93.9%	6.1%
Special	841	9	816	1	0	1	0.25	99.8%	0.2%
All Features	8566	1652	16352	1963	1971	1101	0.42	67.0%	33.0%

Table 3: MetaLens distribution of feature weights – all users.

[5] "Not Seen" was added shortly after the original deployment of MetaLens and was not on option for 83 of the 1668 queries.

Movie/ Theater Feature	**User Selection of Feature Weight**						**Avg. Weight**	**Direction when default weight is modified**	
	0	**0.25**	**0.5 default**	**0.75**	**1**	**Must (1)**		**Lowered**	**Raised**
MovieLens	4	1	104	75	419	NA	0.87	1.0%	99.0%
Not Seen [6]	61	2	152	11	114	235	0.75	14.9%	85.1%
Cream %	127	67	201	69	56	83	0.51	48.3%	51.7%
Genre	154	85	225	34	36	69	0.44	63.2%	36.8%
End Time	205	15	292	40	49	2	0.38	70.7%	29.3%
Critic %	199	52	267	56	14	15	0.36	74.7%	25.3%
MPAA	243	46	237	3	15	59	0.34	79.0%	21.0%
Discount	248	3	278	37	5	32	0.34	77.2%	22.8%
Start Time	244	26	251	38	44	0	0.34	76.7%	23.3%
Avg User	181	135	202	66	19	NA	0.34	78.8%	21.2%
Distance	193	115	251	21	17	6	0.32	87.5%	12.5%
CreamMin	292	31	209	24	41	6	0.29	82.0%	18.0%
Release	289	19	256	31	1	7	0.27	88.8%	11.2%
Distributor	310	17	234	0	0	42	0.27	88.6%	11.4%
CriticMin	299	38	242	0	21	3	0.26	93.4%	6.6%
MinLength	311	26	249	1	10	6	0.25	95.2%	4.8%
Content	328	13	254	0	2	6	0.23	97.7%	2.3%
MaxLength	326	39	219	6	2	11	0.23	95.1%	4.9%
Special	397	1	205	0	0	0	0.17	100.0%	0.0%
All Features	4411	731	4328	512	865	582	0.37	72.4%	27.6%

Table 4: MetaLens distribution of feature weights – active users.

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

	All Users	Active Users
Avg User	31.1%	44.9%
Content	17.8%	26.2%
CreamMin	13.7%	17.9%
Cream %	29.0%	43.3%
CriticMin	13.4%	18.9%
Critic %	25.2%	40.3%
Discount	13.7%	20.4%
Distance	17.4%	20.6%
Distributor	9.8%	17.1%
End Time	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 Time	20.0%	30.5%

Table 5: Feature inclusion in the recommendation table.

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

Table 6: Number of profiles per power user.

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: 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 8: User reported weaknesses of MetaLens

	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 9: User reported system preferences

	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 10: Projected “rankings” of the three systems. [Mean (Std. Dev.)]

11. CAPTIONS FOR ILLUSTRATIONS

Figure 1: MetaLens Recommendation Framework – The data layer gathers data from three internet sources. Data from Yahoo Movies is the reference data and data from the other sources must undergo data fusion to ensure all data contains the same keys. The computation layer receives query requirements from the interface layer, applies an extended Boolean retrieval algorithm to the data from the data layer, and returns recommendations to the interface layer which formats the recommendations for human consumption.

Figure 2: MetaLens Preferences Screen (partial) – The preferences screen allows users to provide information on 19 different query attributes. For each attribute, this involves either selecting which subset of features are desired or which limits will be allowed. It also includes indicating the degree to which movies should be rewarded for fitting the requirements (or penalized for not fitting the requirements as the case may be) and whether information about the attribute should be displayed with recommendations.

Figure 3: MetaLens Recommendation Screen – The recommendation screen displays the movie/theater/showtime triple for each movie with the highest query fit score, as well as movie/theater attribute data selected on the preferences screen.

Figure 4: Options for saving Query Profiles – Each query submitted from the preferences screen, may be saved as a new or existing query for future use.

12. FIGURES/ILLUSTRATIONS

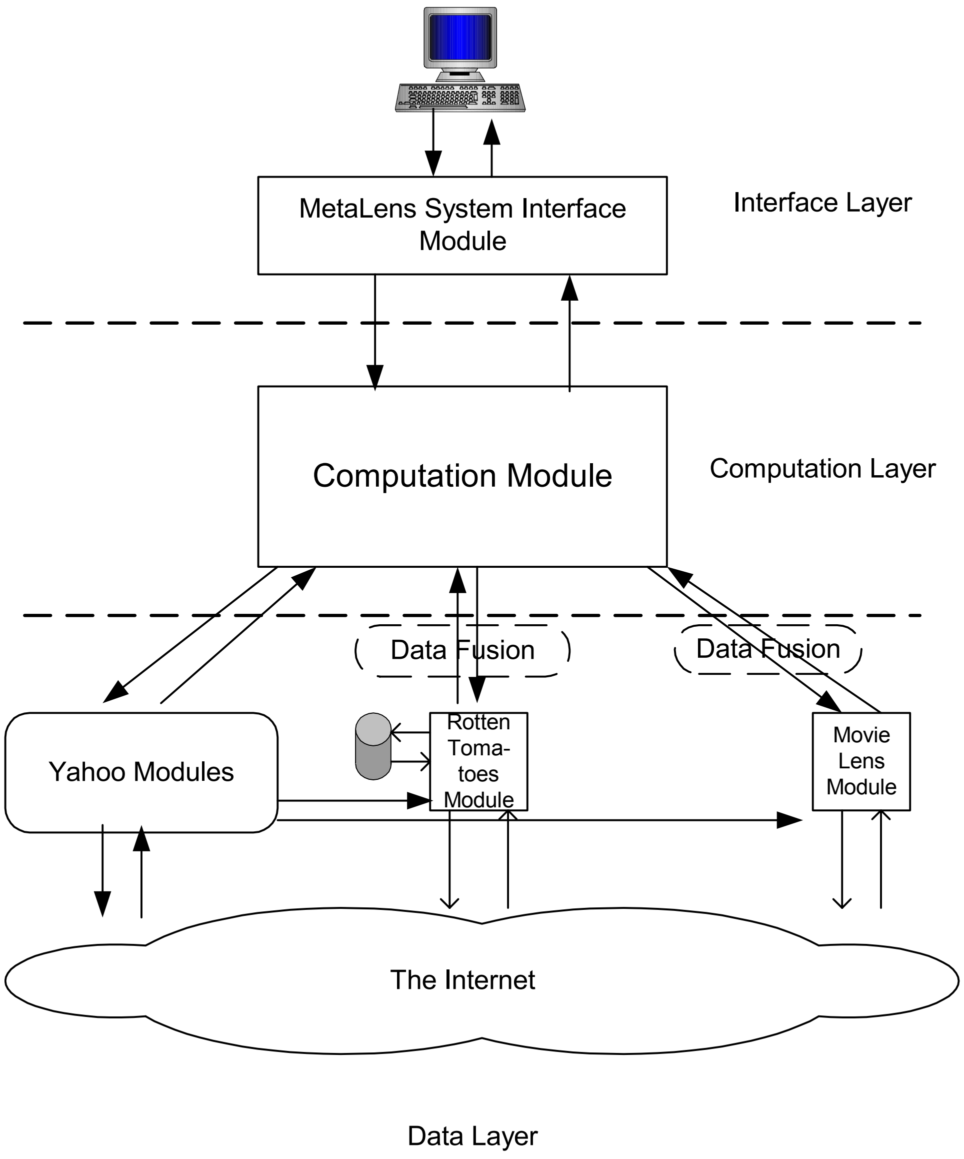
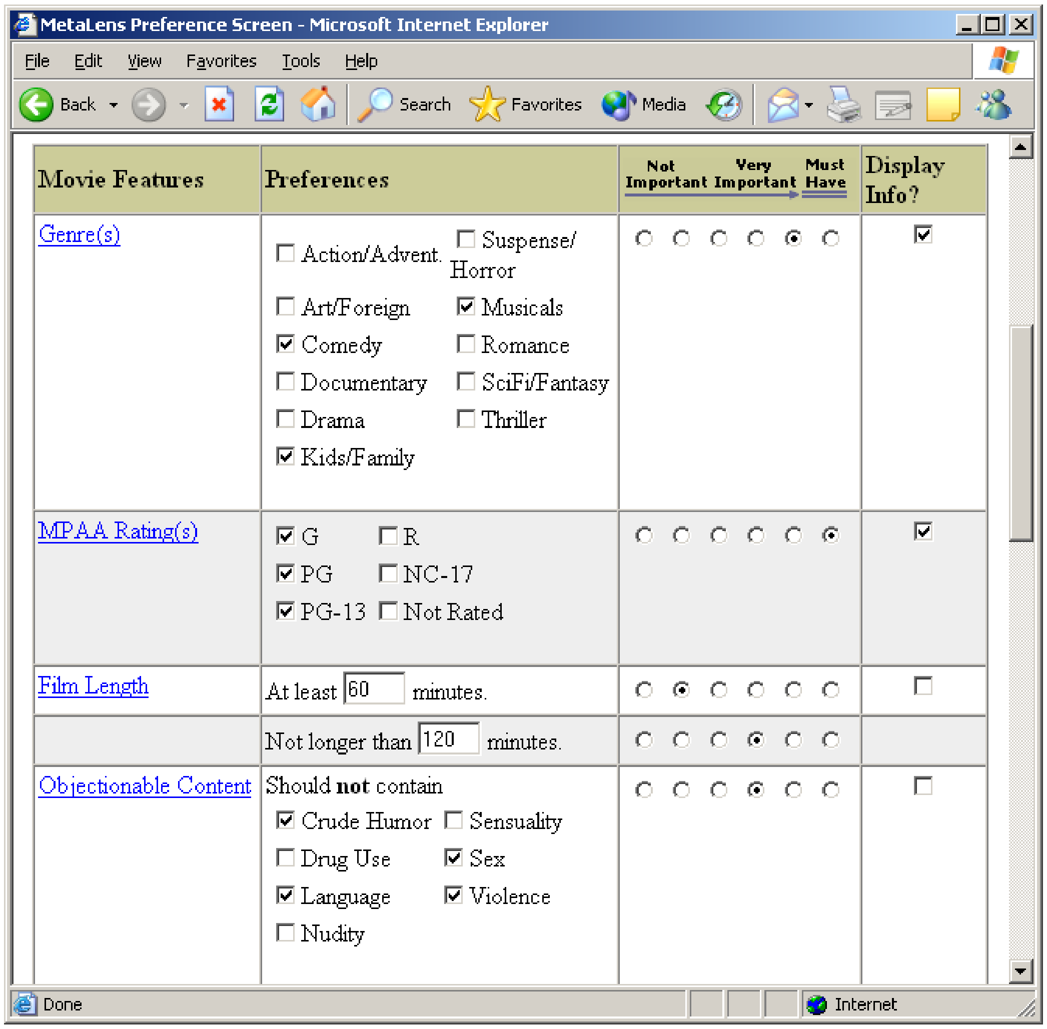
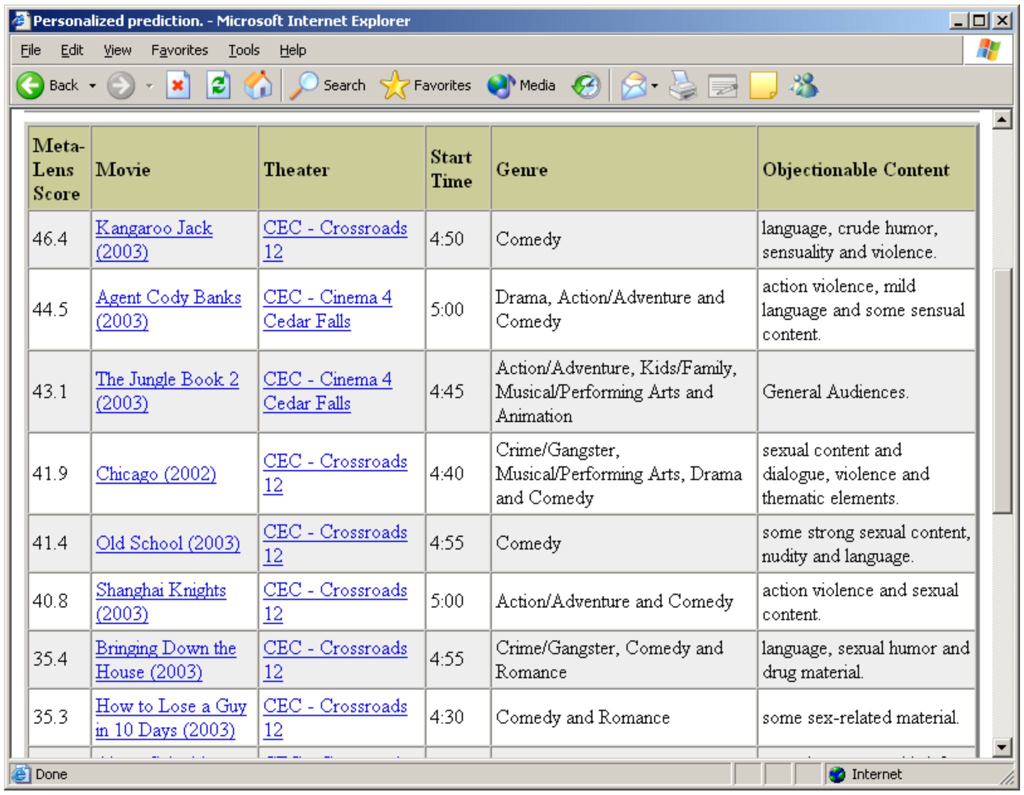


Figure 1: MetaLens Recommendation Framework


MetaLens Preference Screen - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Media
Done
Internet

Movie Features	Preferences	Not Important — Very Important — Must Have	Display Info?
Genre(s)	☐ Action/Advent. ☐ Suspense/ Horror ☐ Art/Foreign ☑ Musicals ☑ Comedy ☐ Romance ☐ Documentary ☐ SciFi/Fantasy ☐ Drama ☐ Thriller ☑ Kids/Family	○ ○ ○ ○ ● ○	☑
MPAA Rating(s)	☑ G ☐ R ☑ PG ☐ NC-17 ☑ PG-13 ☐ Not Rated	○ ○ ○ ○ ○ ●	☑
Film Length	At least 60 minutes.	○ ● ○ ○ ○ ○	☐
	Not longer than 120 minutes.	○ ○ ○ ● ○ ○	
Objectionable Content	Should **not** contain ☑ Crude Humor ☐ Sensuality ☐ Drug Use ☑ Sex ☑ Language ☑ Violence ☐ Nudity	○ ○ ○ ● ○ ○	☐

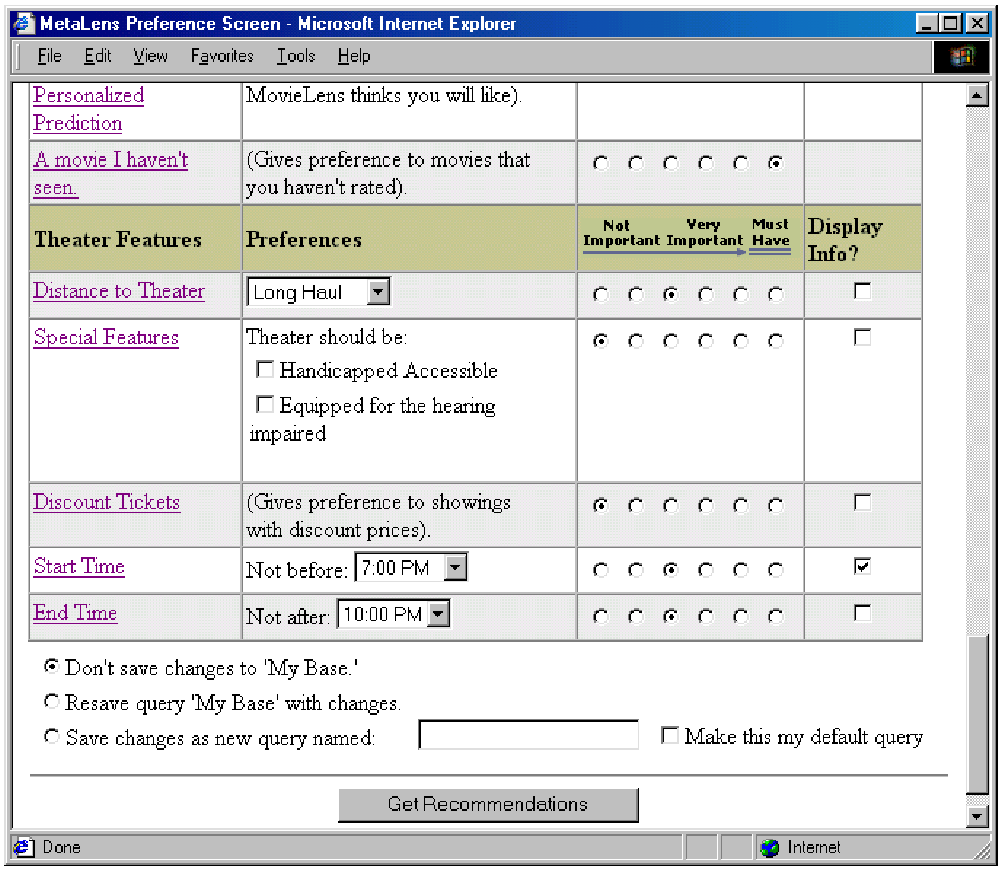
Figure 2: MetaLens Preferences Screen (partial)



Personalized prediction. - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Done Internet

Meta-Lens Score	Movie	Theater	Start Time	Genre	Objectionable Content
46.4	Kangaroo Jack (2003)	CEC - Crossroads 12	4:50	Comedy	language, crude humor, sensuality and violence.
44.5	Agent Cody Banks (2003)	CEC - Cinema 4 Cedar Falls	5:00	Drama, Action/Adventure and Comedy	action violence, mild language and some sensual content.
43.1	The Jungle Book 2 (2003)	CEC - Cinema 4 Cedar Falls	4:45	Action/Adventure, Kids/Family, Musical/Performing Arts and Animation	General Audiences.
41.9	Chicago (2002)	CEC - Crossroads 12	4:40	Crime/Gangster, Musical/Performing Arts, Drama and Comedy	sexual content and dialogue, violence and thematic elements.
41.4	Old School (2003)	CEC - Crossroads 12	4:55	Comedy	some strong sexual content, nudity and language.
40.8	Shanghai Knights (2003)	CEC - Crossroads 12	5:00	Action/Adventure and Comedy	action violence and sexual content.
35.4	Bringing Down the House (2003)	CEC - Crossroads 12	4:55	Crime/Gangster, Comedy and Romance	language, sexual humor and drug material.
35.3	How to Lose a Guy in 10 Days (2003)	CEC - Crossroads 12	4:30	Comedy and Romance	some sex-related material.

Figure 3: MetaLens Recommendation Screen


MetaLens Preference Screen - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
Internet

Personalized Prediction	MovieLens thinks you will like).		
A movie I haven't seen.	(Gives preference to movies that you haven't rated).	○ ○ ○ ○ ○ ●	
Theater Features	**Preferences**	Not Important / Very Important / Must Have	Display Info?
Distance to Theater	Long Haul	○ ○ ● ○ ○ ○	☐
Special Features	Theater should be: ☐ Handicapped Accessible ☐ Equipped for the hearing impaired	● ○ ○ ○ ○ ○	☐
Discount Tickets	(Gives preference to showings with discount prices).	● ○ ○ ○ ○ ○	☐
Start Time	Not before: 7:00 PM	○ ○ ● ○ ○ ○	☑
End Time	Not after: 10:00 PM	○ ○ ● ○ ○ ○	☐

● Don't save changes to 'My Base.'

○ Resave query 'My Base' with changes.

○ Save changes as new query named: ☐ Make this my default query

Get Recommendations

Figure 4: Options for saving Query Profiles