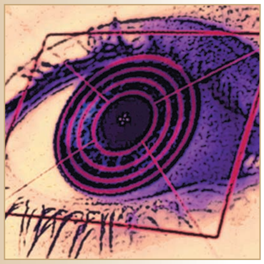
Content-Independent Task-Focused Recommendation.

Article *in* IEEE Internet Computing · December 2001

DOI: 10.1109/4236.968830 · Source: IEEE Xplore

Jon Herlocker
VMware

Joseph A. Konstan
University of Minnesota Twin Cities



Content-Independent Task-Focused Recommendation

A technique that correlates database items to a task adds content-independent context to a recommender system based solely on user interest ratings.

Jonathan L. Herlocker
Oregon State University

Joseph A. Konstan
University of Minnesota

Recommender systems that use collaborative filtering are an important component of personalization solutions on the Internet. These systems predict items of interest to one person according to the recommendations of other people who are also using the system. A user is typically asked to rate items, such as movies, and the system then matches the user to *neighbors* who have rated the items similarly. These neighbors' recommendations of other items tend to be more valuable to the user than recommendations based on overall popularity.

In the past six years, recommender systems have moved from research concepts to commercially marketed systems and high-profile e-commerce sites like Amazon.com and CDNow.com (see the sidebar, "Related Work in Recommender Systems," p. 42). These systems are effective because they predict preferences based on human evaluations and so inherently capture complexities that account for qualities like taste. Furthermore, they do not require either metadata about the content or machine indexing and analysis of it, so they are relatively simple to implement and maintain.

However, a weakness in most current implementations is their basis solely on historical ratings data. This approach assumes that a user's interest is independent of the task at hand. In reality, the task or context greatly affects the value of a recommendation. To support task-specific recommendations, current systems must rely on content-based query engines that use either metadata or full-text indexing and analysis.

In this article, we present a task-focused approach to recommendation that is entirely independent of the type of content involved. The approach leverages robust, high-performance, commercial software. We have implemented it in a live movie recommendation site and validated it with empirical results from user studies.

 1089-7801/01/$10.00

Motivation

There are two content-based approaches commonly used by recommender systems to focus query results:

- *Relational databases*. This approach requires a set of metadata for the items to be recommended. For example, a movie recommender system might include metadata for the director, genre, stars, and release date of each movie. Users specify a set of attributes, which the system translates into a relational database query. The recommender then ranks the items that match the query.
- *Information-retrieval methods*. This approach applies to textual data that has been indexed according to keywords extracted automatically from the text. Users specify a keyword set that describes the task of interest or the information needed — either directly in a query or indirectly through behaviors observed by the system and translated into a query.

Both these approaches are good solutions when appropriate metadata or indexes exist, but they involve significant costs to implement and maintain. They have other limitations, too, that indicate a need for a more general, content-independent solution. For example, relational database solutions require users to specify constraints only through existing metadata attributes. Thus, the success of the recommendation service depends on the creators' foresight in identifying which attributes users will choose to describe their tasks. Likewise, the IR approach is limited by its support for only textual documents.

Task-Focused Approach

To develop a generally applicable, content-independent recommender, we wanted a task-focused approach based primarily on the ratings data that already exists for most current systems, with a minimal requirement for additional data collection from users and no requirement for metadata. Wherever possible, we also wanted to utilize existing recommender system technology.

Figure 1 depicts the architecture of our approach. Its key components include the following:

- *Interest ratings*. A collection of numeric ratings where each rating indicates a user's interest in a specific item. Ratings can be any scale, but are generally discrete. Common ranges are 1 to 10, 1 to 7, 1 to 5, and 0 to 1. The system collects these ratings either explicitly, by asking the user to provide them, or implicitly, by observing what the user does (for example, purchasing an item implies a strong interest, whereas visiting a Web page implies a weak one).
- *Interest recommender*. A traditional recommender system component that predicts ratings from the historical ratings database. It provides an external interface that allows other processes to request an interest prediction for any (user, item) pair. It cannot produce task-focused recommendations by itself.
- *Task specification*. An input that indicates what kind of task the user has in mind to complete. In our approach, this task specification consists of a list of example items, known as a *task profile*.
- *Item association data*. Descriptions of associations between items available for recommendation. In our approach, this data is automatically generated by the task-focused recommender via analysis of the interest ratings data.
- *Task-focused recommender*. A component that collects a task description from the user and returns a set of appropriate recommendations. In our approach, the task-focused recommender operates on ratings data, predictions from the interest recommender, and a task profile.
- *Task-focused recommendations*. A ranked list of items most likely to be associated with the user's current task while also being recommended by other users with similar interests.

The interest ratings and recommender components of this architecture exist in any current collaborative filtering system for predicting interests; the other four components are unique to our solution.

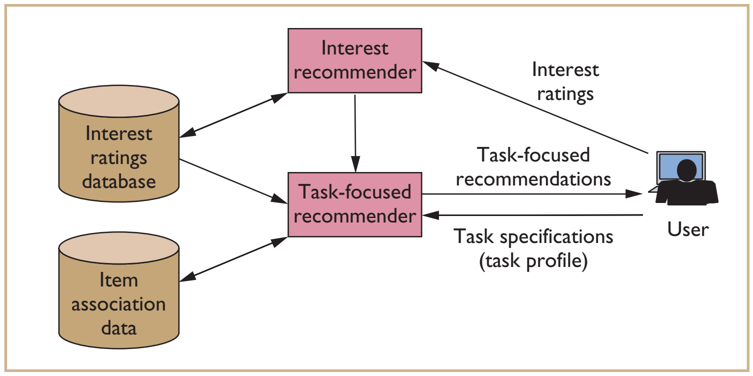


Figure 1. Task-focused recommender system architecture. Traditional interest recommenders predict ratings based on history. The task-focused architecture adds task specification and a task-focused recommender that uses both interest ratings and item associations.

Related Work in Recommender Systems

The recommender system architecture presented in this article represents a combination of ideas from collaborative filtering and data mining. For recent work in these areas, see Sarwar et al.,[1] the special issue of *Communications of the ACM* on personalization,[2] and the papers from this year's workshop on Web Knowledge Discovery in Databases (WebKDD).[1]

Early research on recommender systems was reported in 1994 and 1995 at the ACM Computer-Supported Collaborative Work (CSCW) conference[4] and the Special Interest Group on Computer-Human Interaction (SIGCHI) conference.[5,6] GroupLens was documented in 1997.[7]

In addition to high profile e-commerce sites such as Amazon.com and CDNow.com, there are now commercially marketed systems like Net Perceptions (www.netperceptions.com), LikeMinds (www.macromedia.com/software/likeminds/), and TripleHop (www.triplehop.com).

There has been considerable research in many areas of recommender systems, including the following:

- Quality-of-prediction algorithms,[8,9]
- Effectiveness of user interfaces,[10]
- Use of content-based agents,[11,12] and
- Applications in e-commerce.[13,14]

References

1. Special issue on personalization, M.D. Mulvenna, SS. Anand, A.G. Büchner; and D. Riecken, eds., *Comm. ACM,* vol. 43, no. 8, Aug. 2000.
2. B.M. Sarwar et al., "Analysis of Recommendation Algorithms for E-Commerce," *Proc. 2nd ACM Conf. Electronic Commerce (EC2000),* ACM Press, New York, 2000, pp. 158-167.
3. WebKDD 2001 Workshop, held in conjunction with ACM Int'l Conf. on Knowledge Discovery and Data Mining, 2001; accepted workshop papers available online at http://robotics.stanford.edu/~ronnyk/WEBKDD2001/; to be published in Lecture Notes on Artificial Intelligence, Springer-Verlag.
4. P. Resnick et al. "GroupLens: An open architecture for Collaborative Filtering of *netnews," Proc. 1994 Conference on Computer Supported Cooperative Work* (CSCW), ACM Press, New York, 1994, pp. 175-186.
5. W. Hill, et al., "Recommending and Evaluating Choices in a Virtual Community of Use," *Proc. ACM CHI95 Conf. Human Factors in Computing Systems,* ACM Press, New York, 1995, pp. 194-201.
6. U. Shardanand and P. Maes, "Social Information Filtering: Algorithms for Automating 'Word of Mouth'," *Proc. ACM CHI95 Conf. Human Factors in Computing Systems,* ACM Press, New York, 1995, pp. 210-217.
7. J.A. Konstan et al., "GroupLens: Applying Collaborative Filtering to Usenet News," *Comm. ACM,* vol. 40, no. 3, Mar. 1997, pp. 77-87.
8. C.C. Aggarwal et al., "Horting Hatches an Egg: A New Graph-Theoretic Approach to Collaborative Filtering," *Proc. 5th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining,* 1999, ACM Press, New York, 1999, pp. 201-212.
9. J.S. Breese, D. Heckerman, and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," *Proc. 14th Conf. Uncertainty in Artificial Intelligence (UAI-98),* Morgan Kaufmann, San Francisco, 1998, pp. 43-52.
10. J. L. Herlocker, J.A. Konstan, and J. Riedl, "Explaining Collaborative Filtering Recommendations," *Proc. ACM 2000 Conf. Computer Supported Cooperative Work,* ACM Press, New York, 2000, pp. 241-250.
11. N. Good et al., "Combining Collaborative Filtering with Personal Agents for Better Recommendations," *Proc. 1999 Conf. Am. Assoc. of Artificial Intelligence (AAAI-99),* AAAI Press, Menlo Park, Calif., pp. 439-446.
12. B. M. Sarwar, et al., "Using Filtering Agents to Improve Prediction Quality in the Grouplens Research Collaborative Filtering System," *Proc. ACM 1998 Conf. Computer Supported Cooperative Work,* ACM Press, New York, 1998, p. 345.
13. J.L. Herlocker et al., "An Algorithmic Framework for Performing Collaborative Filtering," *Proc. 22nd Ann. Int'l. ACM SIGIR Conf.,* ACM Press, New York, 2001, pp. 230-237.
14. J.B. Schafer, J.A. Konstan, and J. Riedl, "Recommender Systems in E-Commerce," *Proc. First ACM Conf. Electronic Commerce (EC1999),* ACM Press, New York, 1999, pp. 158-166.

Task Specification

One way to specify content type without metadata is to supply examples of items related to the task at hand. We call this set of items a *task profile.* There are two distinct approaches to collecting task profiles.

In the first approach, the user must explicitly specify items associated with the task. For example, if we want to find gift recommendations for children from a book recommender like Amazon.com, we might select two or three items that our children already own and like. The benefit of this approach is that the items in the task profile are guaranteed to be associated with the user's task. A downside is that the user must do the work. The task-focused recommendations must therefore justify the additional work by offering significantly more value than general-interest recommendations.

The second approach is automatic. The system observes user behaviors, such as items purchased or Web pages visited, and infers a task profile. For example, if a user places a hammer into a shopping basket, the system could use it as a single-item task profile. From this profile, it might recommend nails. Task profiles created this way do not require additional work from the user, but they can be misleading and thus require recommendation algorithms that find task-associated items even when errors exist in the task profile.

Item Association Data

Given a task profile, the problem of task-focused recommendation reduces to one of identifying associations between items. If we can identify a network of associations among items in our database, then we can use it to identify items associated with items in the task profile. Any item that is associated with all items in a task profile is likely to be associated to the user's task.

Our primary mechanism for discovering item associations automatically, independent of content, uses the existing user interest-ratings data. We can represent this data as a matrix, where each row represents a user and each column represents an item. Typically, interest recommenders determine the similarity between users by correlating the rows to find rows of ratings that agree. For task-focused retrieval, however, we want to identify associations between items, not users. Thus, the task-focused recommender computes correlations between columns.

In Table 1, for example, *Star Wars* and *Return of the Jedi* are strongly correlated, as people tend to give them the same rating. Joe rated both movies high on a scale of 1 to 5, and John and Al rated them mediocre. On the other hand, *Star Wars* and *Hoop Dreams* are negatively correlated, since people who liked one apparently disliked the other. In our approach, we consider only positive correlations, using the standard Pearson correlation formula described in any introductory statistics textbook.

A strong positive correlation between items A and B means that, in general, the higher the rating a user gives to A, the higher the rating given to B. Our hypothesis is that this correlation will capture associations between items. For example, if a single user gives the same rating to two items, then there is a probability that those items are associated or contain common elements. As the number of users who give those two items an identical or similar rating increases, the probability that those items have something in common increases. If we compute correlations over the entire user base, a high positive correlation could represent significant common elements.

Although the work reported here does not infer item associations from observed user behaviors, it is possible to do so. For example, if a user places two items in a task profile, there is a probability that the two items are related. If a user purchases two items at the same time, the probability increases. We plan to investigate this line of association in the future.

Task-Focused Recommender

The task-focused recommender (see Figure 1, page 41) computes both the item associations and the task-focused recommendations. The item associations are computed in a nightly batch process. To reduce the amount of storage, we keep only the top 100 most-correlated associations for each item.

Table 1. Example collaborative-filtering data space for movies, represented as a matrix of users and their ratings of movies.

	Star Wars	Hoop Dreams	Contact	Return of the Jedi
Joe	5	2	5	4
John	2	5		3
Al	2	2	3	2
Nathan	5	1	5	

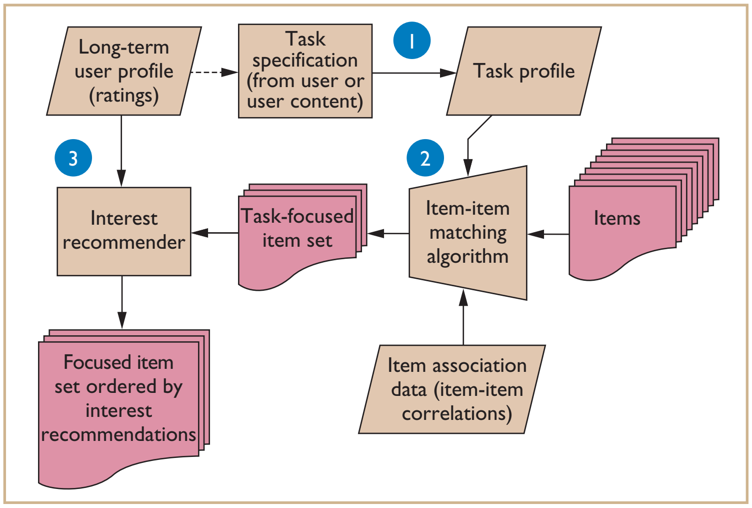


Figure 2. Data flow for proposed task-focused recommendation engine. (1) The user specifies a task profile to the task-focused recommender (the dotted line indicates that users often specify previously rated items). (2) The recommender uses item associations to identify a task-focused item set. (3) The system requests an interest prediction for each item in the focused item set, re-ranks the items accordingly, and returns the resulting recommendation to the user.

Figure 2 depicts the online process of computing task-focused recommendations. First, the user specifies a task profile to the task-recommender algorithm (the dotted line from the long-term user profile to the task specification indicates that users will most commonly specify items they have already rated). Second, the task recommender uses the precomputed item associations to identify those items most likely to be associated with the items in the task profile. This is known as the *task-focused item set*. Finally, the interest recommender re-ranks the set based on the interest predictions and returns the resulting ordered recommendation list to the user.

Matching a task profile with associated items. We call the items in a task profile *query items*. To generate the task-focused item set, we select only database items that occur in the top-correlates list for

Figure 3. Main screen of MovieLens Matcher. Recommendations are filtered through theme profiles that users create.

Figure 4. Theme profile screen. Users create a profile by naming it and adding previously rated movies to it; they can edit their profiles by adding or removing movies to fine-tune their preferences.

each query item. Items that occur close to the top of the lists of all query items are more likely to be selected. An item that occurs close to the bottom of any of the lists will probably not be selected. We do not consider the actual value of the correlation used to identify the association, only the ranking.

If the ratings database is sparse, that is, if each user rates only a small percentage of all items, then certain (item, item) pairs will have very little overlap. That is to say, for two movies A and B, there may be only a small number of users who have rated both A and B. Correlations based on small numbers of data points may be misleadingly high. To address this problem, we use a technique common in data mining, namely, *support*. The support for an association is the number of data points used in the computation of that correlation. In other words, support is the number of users who have rated both A and B. In our approach, we consider only those correlations with a support above a specified *support threshold*. In this manner, we ensure that we recommend items based on sufficient evidence.

For a more detailed description and analysis of the matching algorithm, see Herlocker.[1]

Re-ranking the task-focused item set. The focused item set will contain items that are likely to be associated with the items in the task profile. However, the items selected may or may not meet a user's standards of quality or taste. We want to rank the items such that the first items returned to the user are the most likely to meet the user's standards.

To achieve this, we request interest predictions from the interest recommender for each item in the focused set and then order the items solely according to the interest predictions. We return these ordered results to the user, highest predictions first so that the top-rated items will be those that are highly recommended by the user's neighbors.

Experimental Implementation

We implemented our approach as an experimental new feature on MovieLens, a movie recommendation Web site. MovieLens was already using an interest recommender for new movies, and had 5.3 million ratings, 81,000 users, and 3,800 movies in its database.

We linked the task-focused recommendation interface to the main site and called it "MovieLens Matcher." The interface lets users specify a profile by giving examples of movies that they identify with a certain theme. For this implementation, we refer to the task profile as the *theme profile*. The task-focused recommender matches the user's theme with movies in the database that the user has not yet rated, and then recommends movies that match the theme and have been recommended by the user's neighbors.

From the main screen of MovieLens Matcher, shown in Figure 3, users can create and store any number of theme profiles. To get recommendations, they simply select a profile and click on Go.

Figure 4 shows the create/edit screen for theme profiles. Users specify a profile explicitly by adding movies to it from a list of all movies they have rated in the past. They can edit their profiles by adding or removing movies at any time.

Finally, when users select a profile from the main screen and request recommendations, they get a list of potentially similar movies, ordered by predicted rating as shown in Figure 5. This screen also lets users rate movies that they have already seen but not yet rated.

Experiments

To evaluate the task-focused recommendation system, we performed controlled empirical experiments with volunteer MovieLens users. We placed an invitation on the MovieLens home page. Users could join the experiment by clicking on the invitation link and agreeing to a consent statement. We closed the invitation after a few days so that participants would have about the same time to use the system.

We had 90 participants, whom we assigned randomly to one of three different groups based on a support threshold: 30 (low), 50 (midrange), or 75 (high). (Preliminary studies indicated that a support threshold less than 30 generated highly unpredictable and unacceptable recommendations.)

After a week, we sent an e-mail to all users, inviting them to fill out a survey. The user survey contained 11 questions covering various aspects of the task-focused recommender system and its user interface. In this article, we examine the results from only four of the most important questions (for a more detailed discussion of the results, see Herlocker[1]).

Table 2 reports the level of participation in the experiment, and Table 3 (next page) summarizes the responses to the following four questions according to the category descriptions:

- *Relevance*. On a scale of 1 to 5, please rate how much you agree with the following statement: "MovieLens Matcher returned movies that were relevant to my selected profiles."
- *Discovery*. On a scale of 1 to 5, please rate how much you agree with the following statement: "I discovered interesting new movies using MovieLens Matcher."

Figure 5. MovieLens Matcher search results. Users can further refine the recommender system by ranking movies that they have already seen but not yet ranked in the database.

Table 2. User participation in the experiment.

Category	Number
Opted in to the experiment	90
Created at least one profile	73
Average profiles/user	1.8
Average profile size	10.6
Users who responded to survey	52

- *Value*. On a scale of 1 to 5, please rate how much you agree with the following statement: "I would find valuable a service that accurately recommends movies that are similar to ones I present."
- *Add*. Would you like to see this feature added? (yes/no response).

Empirical Results

Groups 1, 2, and 3 had support thresholds of 30, 50, and 75, respectively.

As Table 3 shows, users would value a task-focused recommendation system such as the one provided by the MovieLens Matcher interface if it were accurate (average 4.3 out of 5). The low standard deviation shows that there was considerable consensus on this response. This leads us

Table 3. Group responses to four survey questions.

Group		Relevance	Discovery	Value	Add
1 (30)	Mean	2.4000	2.4667	4.3333	0.8667
	No answer	15	15	15	15
	Std. Dev.	0.9103	0.9904	0.9759	0.3519
2 (50)	Mean	3.6364	3.4545	4.1818	0.9091
	No answer	11	11	11	11
	Std. Dev.	1.2863	1.2136	0.8739	0.3015
3 (75)	Mean	3.2667	3.4000	4.4000	0.8667
	No answer	15	15	15	15
	Std. Dev.	1.2228	1.2421	0.7368	0.3519
Total	Mean	3.0488	3.0732	4.3171	0.8780
	No answer	41	41	41	41
	Std. Dev.	1.2237	1.2122	0.8497	0.3313

to believe that the user interface – task specification by example – has potential to be accepted and valued by users. This belief is strengthened by the fact that 88 percent of all surveyed users indicated they would like to add the task-focused recommender interface to the MovieLens Web site permanently.

The questions of relevance and discovery were designed to measure the accuracy and quality of recommendations generated by our task-focused recommendation algorithm. As shown in Table 3, the overall responses to the questions were midrange (near three) on the average. A midrange response of three indicates neither a negative nor a positive belief. However, comparing the mean responses per experimental group, it becomes clear that group 1, which received recommendations from an algorithm with a low support threshold (30), had a significantly lower perception of the relevance of the task-focused recommendations. Nor did they agree that the task-focused recommender helped them to discover new items. In contrast, users from groups 2 and 3 on average had positive perceptions of the relevance of the recommended items and the system's helpfulness in discovering new items of interest. This data indicates two important points:

- the support threshold is an important parameter to which accuracy is highly sensitive; and
- given a good value for support threshold, our task-focused recommendation approach can generate valuable results.

Qualitative Results

Participants in the experiment were given the opportunity to evaluate the task-focused recommender qualitatively. Multiple users observed that the recommendation quality decreased dramatically when the task profile contained more than three or four items. This observation seems consistent with our approach for matching the task profile to associated items. To be associated with the task profile, an item must occur in the top-correlates list of every item in the profile. As the number of items in the profile increases, the probability decreases for a strong association in every query item's top-correlates list. We did not attempt to measure the optimal theme profile size, but if the optimal size is around four items, then poorer performance is to be expected, given that the average profile size was greater than 10.

The task profiles that the participants created identified themes that were personalized and expressive. They were personal in that they identified themes that were personally important ("my kind of Sci Fi"). They identified attributes of movies that were complex and expressive, beyond those attributes for which metadata is commonly available (genre, actors, director, and so on). Table 4 lists some examples of theme profiles created by users.

Applicability

Our approach uses software components that are simple to implement and that scale easily to performance levels comparable to commercially available recommender systems. The approach has the most potential in situations where automatic content analysis or indexing cannot be applied easily. The ideal target domain is one where users find it hard to describe what they are looking for but easy to produce an example of it. A recommender system for artwork or poems might fall under this category.

We recognize that profiles will not solve every task-focused recommendation problem. For example, they will be of less use if your task is to "find a movie that I will like that starts in the next 30 minutes at the theater closest to my house." For this task and many others, you would need content-based metadata describing the movies and theaters.

However, there are many tasks where they do apply. For example, on a technical support Web site, a task profile could be created from the Web hyperlinks a user clicks on. Since people with similar problems are likely to click on the same links, the site could recommend likely solutions based on session browsing behaviors. These recommendations could also be used to adaptively adjust the content of the Web display.

For another example, online retailers could offer profiles to focus purchasing task recommendations. This would help users who are seeking recommendations for a type of item that differs substantially from their normal buying behaviors (as when purchasing a gift).

Privacy

Interest recommenders rely on interest ratings from users; without historical ratings, they cannot predict future ratings. Users of interest recommenders generally try to protect their privacy by using pseudonyms, or screen names. In our approach, users can choose a higher level of privacy by not providing historical interest ratings. They can still issue queries using task profiles, but the query results will be ranked by overall popularity, not predicted level of interest. Thus, the results will be personalized to user tasks, but not necessarily to taste.

Conclusion

The empirical data shows that associations based on item-to-item correlations in rating space are sufficient to produce useful task-focused recommendations. One step toward an even more robust recommender system would be to collect information from the users regarding associations between items. Such data would most likely produce significantly more accurate task-focused recommendations.

Our future work will look at how we can maximize the effectiveness of the current system by relaxing the content-independence constraint and incorporating some domain-specific content knowledge. For example, in recommending movies we might alter the matching algorithm to use genre, director, and star information in addition to correlations between interest ratings. Such approaches are also likely to increase the recommendation accuracy.

Table 4. Themes for which the system was effective.

British costume dramas made in the '90s
Woody Allen films
Kids movies
"Light" entertainment movies
Action movies — minus sequels
Witty romances
Fun '80s nostalgia flicks
Intelligent romantic comedies
Horror comedies
Raw comedy — usually vulgar
Films the critics panned but I loved

Reference

1. J.L. Herlocker, *Understanding and Improving Automated Collaborative Filtering Systems*, doctoral dissertation, Dept. of Computer Science and Engineering, Univ. of Minnesota, Minneapolis, 2000.

Jonathan L. Herlocker is assistant professor of computer science at Oregon State University. His research interests address technologies that support intelligent information creation and management systems, most recently, recommender system (collaborative filtering) technology. He received a PhD from the University of Minnesota in 2000. He chaired the organizing committee for the 2001 Recommender Systems workshop at the ACM Conference for Research and Development in Information Retrieval (SIGIR) in New Orleans.

Joseph A. Konstan is associate professor of computer science and engineering at the University of Minnesota. His research addresses a variety of human-computer interaction issues related to filtering, comprehending, organizing, and automating large and complex data sets. He is best known for his work in collaborative filtering — the GroupLens recommender system. Konstan received a PhD from the University of California, Berkeley, in 1993; he is an ACM lecturer and editor of the SIGCHI Bulletin.

Readers may contact the authors via e-mail at herlock@cs.orst.edu and konstan@cs.umn.edu.