# 1 System Details

## 1.1 System Owner

*This may be the designer deploying the system, a larger agency or body, or some combination of the two. The entity completing the report should also be indicated.*

Movielens is maintained by researchers at the University of Minnesota in the Grouplens research group (`https://grouplens.org/`).

## 1.2 Dates

*The known or intended timespan over which this reward function & optimization is active.*

The system has been active since it was first released in August 1997. This reward report (v4.1) was last updated March 2015.

## 1.3 Feedback & Communication

*Contact information for the designer, team, or larger agency responsible for system deployment.*

Information on contact emails for account problems, website problems, movie content issues, and general comments can be found at `https://movielens.org/info/contact`. General comments and ideas for improving MovieLens can be discussed on the UserVoice forum at `https://movielens.uservoice.com`.

## 1.4 Other Resources

*Where can users or stakeholders find more information about this system? Is this system based on one or more research papers?*

A history of the MovieLens system and datasets is presented in [1], and additional research papers are cited therein.

# 2 Optimization Intent

## 2.1 Goal of Reinforcement

*A statement of system scope and purpose, including the planning horizon and justification of a data-driven approach to policy design (e.g. the use of reinforcement learning or repeated retraining). This justification should contrast with alternative approaches, like static models and hand-designed policies. What is there to gain with the chosen approach?*

The system is a website designed to display personalized movie recommendations on the basis of user entered ratings. As a user browses the site, potentially filtering with search terms, the system displays movies in an order determined by predictions of how the user will rate them. When users rate movies, the predictions are updated, altering the ordering on subsequent page views.

The ranking policy effectively considers a one-step time horizon, directly using predictions for ranking. It does not consider the effect of multiple sequential interactions.

This system is best characterized as a "repeated retraining" of a preference model generated by supervised learning (SL). This model is then used to rank movies for display. Using SL allows for preference models which capture highly personal tastes, something that would be difficult to hand design. Repeated retraining allows the preference model to adapt to a changing environment, including shifts in user tastes and the release of new movies.

In addition to the primary goal of movie recommendation, this system supports academic research on human-computer interaction and general recommender system design.

## 2.2 Defined Performance Metrics

*A list of "performance metrics" included explicitly in the reward signal, the criteria for why these metrics were chosen, and from where these criteria were drawn (e.g. government agencies, domain precedent, GitHub repositories, toy environments). Performance metrics that are used by the designer to tune the system, but not explicitly included in the reward signal should also be reported here.*

The ranking policy orders movies by a weighted sum of predicted rating and popularity, so we can view the combination of these quantities as making up the reward signal. Prior to version 4.0, the reward only depended on rating and did not incorporate popularity.

Additionally, recommender models are evaluated offline using prediction accuracy (RMSE), top-N accuracy (recall), diversity (intra-list similarity), and popularity (details in [2]). Prior to v4.0, models were evaluated primarily for accuracy, including MAE, RMSE, and nDCG (details in [3]).

### 2.3 Oversight Metrics

*Are there any additional metrics not included in the reward signal but relevant for vendor or system oversight (e.g. performance differences across demographic groups)? Why aren't they part of the reward signal, and why must they be monitored?*

Metrics which are monitored but not incorporated into the policy or model include the number of users, number of movies, number of entered ratings, monthly active users, and the number of logins for each user. These indicators of overall system operation are not targets for optimization.

### 2.4 Known Failure Modes

*A description of any prior known instances of "reward hacking" or model misalignment in the domain at stake, and description of how the current system avoids this.*

No instances of reward hacking or misalignment have been observed. Because the system allows for explicit user input (search terms, model selection), errors in rating predictions do not prevent users from finding and rating movies.

## 3 Institutional Interface

### 3.1 Deployment Agency

*What other agency or controlling entity roles, if any, are intended to be subsumed by the system? How may these roles change following system deployment?*

MovieLens was released due to the shuttering of EachMovie in 1997, a movie recommendation site hosted by DEC. It was developed and is maintained by Grouplens, a research group at University of Minnesota.

### 3.2 Stakeholders

*What other interests are implicated in the design specification or system deployment, beyond the designer? What role will these interests play in subsequent report documentation? What other entities, if any, does the deployed system interface with whose interests are not intended to be in scope?*

One interface of interest is the technology that powers the recommendation engine. Currently, it is powered by Lenskit, an open source framework developed to promote reproducability and openness in the recommendation systems community [3]. Previously in v3.0-v3.4, the recommendations were powered by MultiLens, another open source recommendation engine. MultiLens replaced Net Perceptions (v1.1-v2.0), a recommendations systems company cofounded in 1996 by GroupLens faculty and students and sold in 2004 [4]. The recommendation model in v0.0-v1.0 was originally developed by GroupLens for personalized Usenet news recommendation [5].

Another relevant interface is with The Movie Database, a free and open source user editable movie database for plot summaries, movie artwork, and trailers. Previously, from in v3.4-v4.0, MovieLens integrated with the Netflix API to display movie posters and plot synopsis on the movie details page. However, Netflix eventually discontinued its API support.

An important stakeholder is the Movielens users. Soliciting user judgements and opinions is often a key element in determining if an experimental change is successful. Additionally, one-off user studies (with participants recruited from email) are used to test features that are not ready to scale or integrate into the main user interface.

Finally, a key stakeholder is the researchers: both in Grouplens and the in the community more broadly. The openness of users to experiments on a broad range of features has enabled GroupLens research in many different areas on the Movielens platform. The regular release of anonymized datasets of movie ratings is important to the broader machine learning, data science, and information retrieval communities.

A potentially relevant group of stakeholders is movie producers. However, because Movielens is relatively small and isolated from larger commercial endeavors, it has limited impact on movie studios and production, so their interests are not in scope.

### 3.3 Explainability & Transparency

*Does the system offer explanations of its decisions or actions? What is the purpose of these explanations? To what extent is the policy transparent, i.e. can decisions or actions be understood in terms of meaningful intermediate quantities?*

The system displays predicted ratings alongside movies, explaining the movies position within a list, and suggesting to the user whether or not they will like the movie. The ranking policy is easily understood as a weighted combination of predicted rat-

ing and popularity. However, the computation of predicted ratings is more complex. Some available models are more easily explained to users than others (e.g. nearest neighbors vs. matrix factorization). However, the details are well documented in publicly available research papers [2], and researchers respond to user requests for explanation on the UserVoice discussion board [6].

### 3.4 Recourse

*Can stakeholders or users contest the decisions or actions of the system? What processes, technical or otherwise, are in place to handle this?*

By entering ratings, users are able to affect their preference models to hopefully become more accurate. Additionally, the movies displayed by the system are sourced from The Movie Database, which is user-editable. (Previously in v3.2-v3.5, users could add and edit movies to MovieLens directly.) Furthermore, the current version of the system allows users to choose between three recommender models. Finally, users can make suggestions and requests directly to designers on the UserVoice forum.

## 4 Implementation

### 4.1 Reward Details

*How was the reward function engineered? Is it based on a well-defined metric? Is it tuned to represent a specific behavior? Are multiple terms scaled to make one central loss, and how was the scaling decided?*

The reward is a weighted sum:

$$0.9 \cdot \text{rank}(\widehat{r}_{ui}) + 0.1 \cdot \text{rank}(p_i)$$

where $\widehat{r}_{ui}$ is the predicted rating of movie $i$ by user $u$, $p_i$ is the number of ratings movie $i$ has recieved in the past 10 days, and rank normalizes input, returning 1 for the largest (across all movies) and 0 for the smallest. This blending is the result of empirical evidence that it improves user satisfaction.

### 4.2 Environment Details

*Description of states, observations, and actions with reference to planning horizon and hypothesized dynamics/impact. What dynamics are brought into the scope of the optimization via feedback? Which dynamics are left external to the system, as drift? Have there been any observed gaps between conceptualization and resultant dynamics?*

The system handles approximately 250k users and 30k movies. These numbers have grown over the years. In 1999 (v1.1), MovieLens received attention from the mass media, causing an increase in user signups. Since then, the user growth has been stable (20-30 signups per day), largely the result of word-of-mouth or unsolicited press. Early on, the movie database was hand-curated and primarily contained movies with wide theatrical release in the United States. In v3.2-v3.5, MovieLens added the ability for users to edit and add movies. Since v4.0, MovieLens uses The Movie Database, a free and open source user editable movie database.

The actions taken by the system are page displays of 10 movies in a ordered list, where pages can be perused by arrows. The views can be explicitly filtered with search terms like year and genre; these explicit inputs this make up a component of the observation. The second component is the entered ratings in the form `<user_id, movie_id, rating, timestamp>`.

There are three potential sources of dynamics in this environment: the addition of new movies, the joining and departing of users, and the preferences that users have for movies. Because this system effectively uses a planning horizon of 1, none of these dynamics are explicitly accounted for. This is appropriate, as the goal of MovieLens is not to shift broad patterns of movie consumption. Though the movies, users, and preferences may change over time, these changes are more likely to be due to external factors than feedback with the MovieLens system. Additionally, the data collected by MovieLens is not fine-grained enough to detect such impacts of feedback.

### 4.3 Measurement Details

*How are the components of the reward and observations measured? Are measurement techniques consistent across time and data sources? Under what conditions are measurements valid and correct? What biases might arise during the measurement process?*

Ratings are entered by users via clicks on a star graphic, and can take values 0.5-5 in half integer increments. Prior to v3.0, ratings took values in integer increments. The increased granularity was the most requested feature in a user survey. Prior to v4.0, ratings were entered through a drop-down menu, and the meaning of rating values was de-

scribed in a legend at the top of the page (see Figure 1).

A possible source of bias in the measured ratings is due to anchoring effects, due either to the displayed predicted rating or due to the historically provided movie rating legend. However, broad trends in rating values did not change when the legend was removed in v4.0

Finally, the recorded timestamp represents when a user adds a particular rating rather than when they watched a movie. This limits the ability of the system to detect the impacts of its own recommendations.

### 4.4 Algorithmic Details

*The key points on the specific algorithm(s) used for learning and planning. This includes the form of the policy (e.g. neural network, optimization problem), the class of learning algorithm (e.g. model-based RL, off-policy RL, repeated retraining), the form of any intermediate model (e.g. of the value function, dynamics function, reward function), technical infrastructure, and any other considerations necessary for implementing the system. Is the algorithm publicly documented and is code publicly available? Have different algorithms been used or tried to accomplish the same goal?*

The policy selects a page view to present to the user based on explicitly provided input and rating data. First, explicit input is used to filter the list of movies. Then, the recommender model is used to predict a user's ratings of these movies. Finally, the movies are displayed in order of these predicted ratings, blended with a popularity factor.

The main component of the policy is therefore the recommender model. This model is user-selectable, so that users can choose between a non-personalized baseline, a preference elicitation model intended for new users, an item-item collaborative filtering model, or a matrix factorization model. Further details on how these models are trained is available in [2]. Previously in v3.0-3.5, the recommender was fixed as an item-item collaborative filtering model. Prior to that in v1.0-2.0, the model was a user-user collaborative filtering model.

### 4.5 Data Flow

*How is data collected, stored, and used for (re)training? How frequently are various components of the system retrained, and why was this frequency chosen? Could the data exhibit sampling bias, and is this accounted for in the learning algorithm? Is data reweighted, filtered, or discarded? Have data sources changed over time?*

All user rating data is stored by MovieLens and used by the recommender models to make rating predictions. When a user enters a new rating, it immediately impacts their rating predictions, since the "input" to the recommender changes. Less frequently, the ratings are used to update the parameters of the recommender models. An anonymized subset of this data is also periodically released for use by the wider research community.

The dataset of user ratings is likely biased. There is sampling bias due to the fact that users only rate movies that 1) appear on a page and 2) that they have watched. These factors are directly and indirectly impacted by the MovieLens system itself. The fact that users can explicitly filter pageviews with search terms mitigates these effects, but it is unlikely that it removes them.

The initial Movielens system was trained on a public dataset from EachMovie of approximately 2.8 million ratings from 72k users across 1.6k movies, but this has since been discarded. The dataset was retired by HP in October 2004, and due to privacy concerns, it is no longer available for download.

### 4.6 Limitations

*Discussion and justification of modeling choices arising from computational, statistical, and measurement limitations. How might (or how have) improvements in computational power and data collection change(d) these considerations and impact(ed) system behavior?*

The most prevalent limitation of this system is that it does not plan over a long horizon and therefore does not consider the effects of dynamics. While a more complex policy would allow the system to adapt to ordering effects, the resulting temporal dependence would complicate the ability to users to reliably navigate the movie database. Furthermore, users do not always enter movie ratings immediately after watching a movie, instead sometimes entering batches of ratings for movies that they watched in the past.

### 4.7 Engineering Tricks

*RL systems are known to be sensitive to implementation tricks that are key to performance. Are*

*there any design elements that have a surprisingly strong impact on performance? For example, state-action normalization, hard-coded curricula, model-initialization, loss bounds, or more?*

The system cannot provide reliable recommendations until users provide a minimum number of ratings. This problem is avoided by the interface design: when a user joins the site, they express their preferences over several displayed clusters of movies. These preferences are used, in combination with the rating profiles of other users, to generate a psuedo-rating profile for the new user. Further description is available in [7].

This preference elicitation process replaced a minimum movie requirement. Previously, until a user rated a minimum number of movies, the front page would display 10 movies at a time. From v0-v3, the minimum number was 5, and of the 10 movies per page, nine were randomly selected from the database and one from a hand-designed list of recognizable titles. In v3, the minimum number was 15, and the 10 movies were selected for their popularity, excluding the top 50-150 movies. This increased requirement was due to the needs of an item-item (rather than user-user) collaborative filtering algorithm. The switch to a preference elicitation process was motivated by the observation that the 15 rating requirement was too arduous, taking users an average of 6.8 minutes to complete and 12.6% of users failing to complete it.

# 5 Evaluation

## 5.1 Evaluation Environment

*How is the system evaluated (and if applicable, trained) prior to deployment (e.g. using simulation, static datasets, etc.)? Exhaustive details of the offline evaluation environment should be provided. For simulation, details should include description or external reference to the underlying model, ranges of parameters, etc. For evaluation on static datasets, considering referring to associated documentation (e.g.* Datasheets *[8]).*

The primary evaluation is to consider various properties of recommender models on offline datasets. This includes many of the publicly released MovieLens datasets, which are described in detail in [1].

## 5.2 Offline Evaluations

*Present and discuss the results of offline evaluation. For static evaluation, consider referring to associated documentation (e.g.* Model Cards *[9]). If applicable, compare the behaviors arising from counterfactual specifications (e.g. of states, observations, actions).*

This offline evaluation includes prediction accuracy (RMSE), top-N accuracy (recall), diversity (intra-list similarity), and popularity. Detailed evaluations are available in [2], and key quantities are displayed in (Figure 2).

## 5.3 Evaluation Validity

*To what extent is it reasonable to draw conclusions about the behavior of the deployed system based on presented offline evaluations? What is the current state of understanding of the online performance of the system? If the system has been deployed, were any unexpected behaviors observed?*

Offline evaluation metrics (like top-N accuracy) were chosen to align with the ranking setting. While the offline evaluations are imperfect (due to dataset biases), the system appears to work well ad no unexpected behaviors have been observed.

## 5.4 Performance standards

*What standards of performance and safety is the system required to meet? Where do these standards come from? How is the system verified to meet these standards?*

N/A

# 6 System Maintenance

## 6.1 Reporting Cadence

*The intended timeframe for revisiting the reward report. How was this decision reached and motivated?*

This report is updated whenever there is a major system update, either to the user interface or the backend. Such updates will occur periodically, coinciding with research initiatives.

## 6.2 Update Triggers

*Specific events (projected or historic) significant enough to warrant revisiting this report, beyond the cadence outlined above. Example triggers include*

*a defined stakeholder group empowered to demand a system audit, or a specific metric (either of performance or oversight) that falls outside a defined threshold of critical safety.*

If a large change is observed in oversight metrics, or if many users express dissatisfaction on the UserVoice forum, the system design will be revisited by the researchers who maintain it. If an update is deemed necessary, this report will be updated.

### 6.3 Changelog

*Descriptions of updates and lessons learned from observing and maintaining the deployed system. This includes when the updates were made and what motivated them in light of previous reports. The changelog comprises the central difference between reward reports and other forms of machine learning documentation, as it directly reflects their intrinsically dynamic nature.*

The versions of this report are enumerated as vX.Y where X corresponds to the user interface version and Y corresponds to major changes within interfaces.

- v0.0 (August 1997) Initial release.
- v0.1 (April 1998) The ML 100K dataset is released, covering 9/1997–4/1998.
- v1.0 (September 1999) Update to v1 interface.
- v1.1 (November 1999) Media exposure causes an increased number of users. Switch from GroupLens to Net Perceptions recommender model.
- v2.0 (February 2000) Update to v2 interface. Additional movie metadata and reviews added to movie details pages.
- v3.0 (February 2003) Update to v3 interface. Switch from Net Perceptions user-user recommender to MultiLens item-item recommender. Ratings now in half-star (rather than full) increments. Require that users rate at least 15 movies before receiving recommendations. The ML 1M dataset is released, covering 4/2000–2/2003.
- v3.1 (June 2005) Added discussion forums to site.
- v3.2 (September 2008) Added feature so that users can add movies to database.
- v3.3 (January 2009) The ML 10M dataset is released, covering 1/1995–1/2009.
- v3.4 (Spring 2009) Netflix API integration for poster art and synopsis.
- v3.5 (January 2012) Switch from Multilens to Lenskit recommender (still item-item).
- v4.0 (November 2014) Update to v4 interface. Rating interface combined with "predicted rating" star graphic to accept click events. Switch to user-selectable recommender model. Legend describing the meanings of ratings and dropdown menu removed. Drop minimum rating requirement in favor of group-based preference elicitation. Integration with The Movie Database for plot summaries, movie artwork, and trailers.
- v4.1 (March 2015) The ML 20M dataset is released, covering 1/1995–3/2015. Moving forward, MovieLens will make public additional nonarchival datasets: `latest` which is unabridged for completeness and `latest-small` for educational use.

## References

[1] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *Acm transactions on interactive intelligent systems (tiis)*, vol. 5, no. 4, pp. 1–19, 2015.

[2] M. D. Ekstrand, D. Kluver, F. M. Harper, and J. A. Konstan, "Letting users choose recommender algorithms: An experimental study," in *Proceedings of the 9th ACM Conference on Recommender Systems*, 2015, pp. 11–18.

[3] M. D. Ekstrand, M. Ludwig, J. A. Konstan, and J. T. Riedl, "Rethinking the recommender research ecosystem: reproducibility, openness, and lenskit," in *Proceedings of the fifth ACM conference on Recommender systems*, 2011, pp. 133–140.

[4] A. Press, "Net perceptions returns cash to shareholders," *USA Today*, 08 2003. [Online]. Available: http://usatoday30.usatoday.com/tech/techinvestor/techcorporatenews/2003-08-07-net-perceptions_x.htm

[5] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl, "Grouplens: Applying collaborative filtering to usenet news," *Communications of the ACM*, vol. 40, no. 3, pp. 77–87, 1997.

[6] Anonymous, "Explain what the recommendation options mean." [Online]. Available: https://movielens.uservoice.com/forums/238501-general/suggestions/7006672-explain-what-the-recommendation-options-mean

[7] S. Chang, F. M. Harper, and L. Terveen, "Using groups of items for preference elicitation in recommender systems," in *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, 2015, pp. 1258–1269.

[8] T. Gebru, J. Morgenstern, B. Vecchione, J. W. Vaughan, H. Wallach, H. D. Iii, and K. Crawford, "Datasheets for datasets," *Communications of the ACM*, vol. 64, no. 12, pp. 86–92, 2021.

[9] M. Mitchell, S. Wu, A. Zaldivar, P. Barnes, L. Vasserman, B. Hutchinson, E. Spitzer, I. D. Raji, and T. Gebru, "Model cards for model reporting," in *Proceedings of the conference on fairness, accountability, and transparency*, 2019, pp. 220–229.

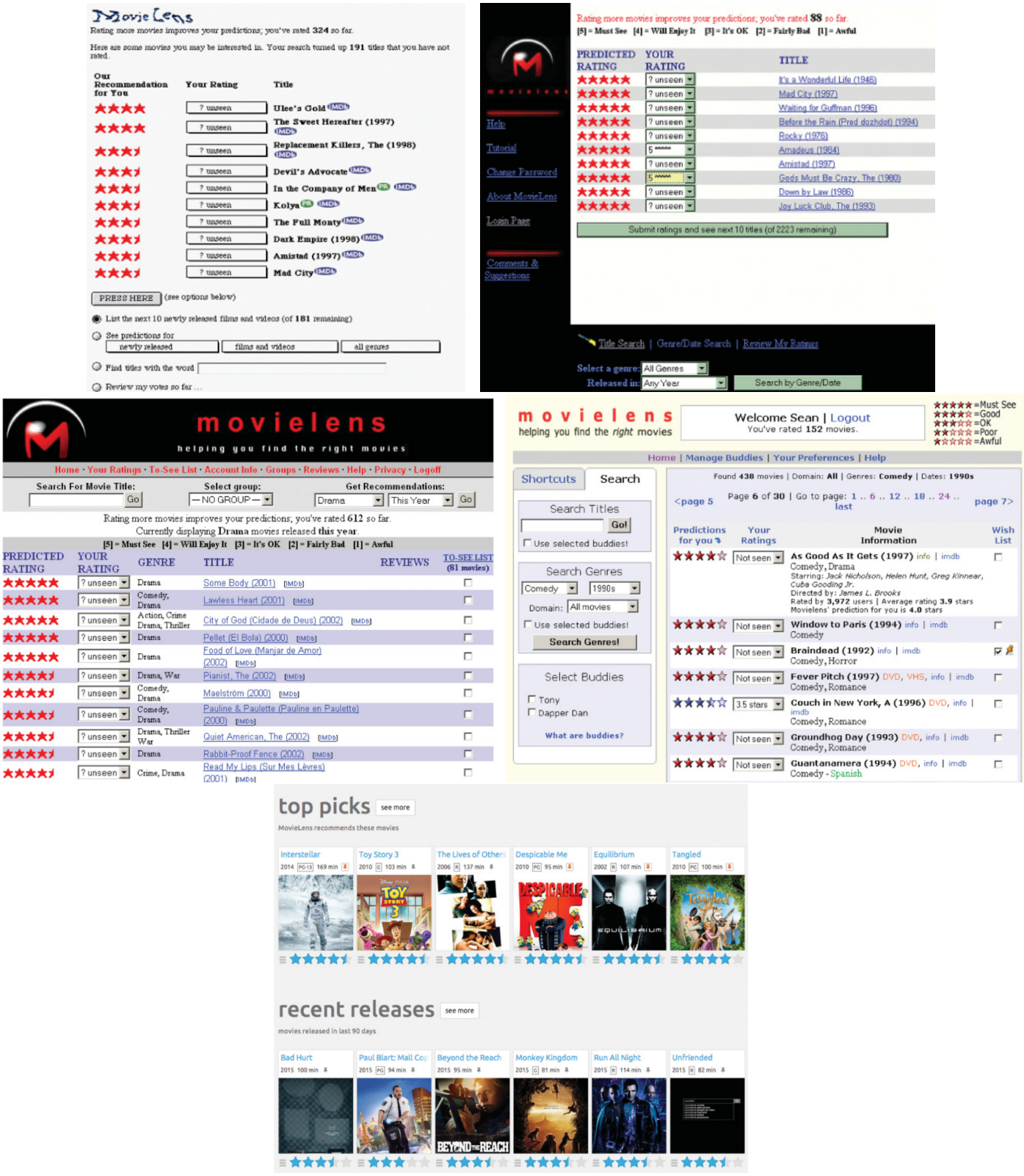


Figure 1: The MovieLens recommender system interface v0-v4.

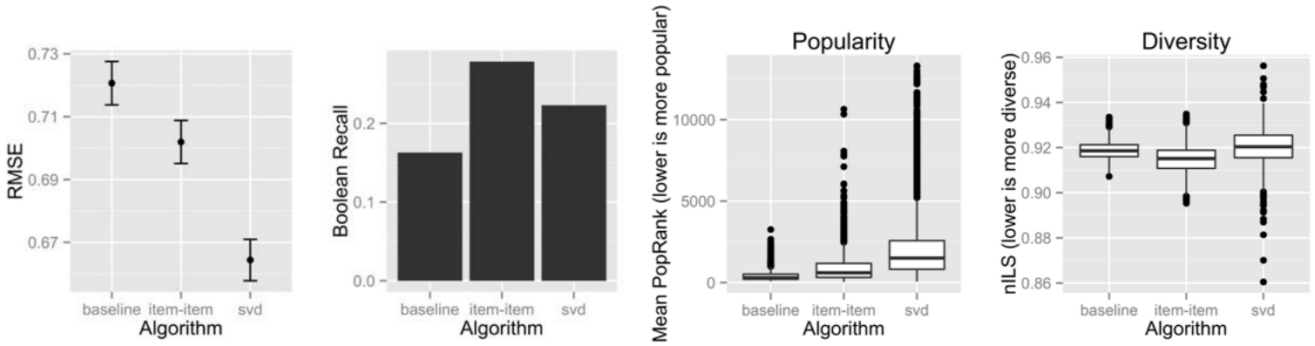


Figure 2: Offline evaluation of recommender models from [2].