THE UNIVERSITY OF NEW SOUTH WALES

The School of Civil and Environmental Engineering

Movie Recommender System:
Designing and Evaluating Personalized Movie Recommendations

z5522700
5th July 2025

1.Scope

1.1. Domain & Intended Users

This project proposes a recommender system designed for website users to suggest movies they are likely to enjoy. The system has two primary goals: to deliver daily movie recommendations tailored to each user's interests and viewing history, and to help users discover new creators or topics through dynamic suggestions. The intended users are movie enthusiasts seeking personalized suggestions based on their past ratings and preferences.

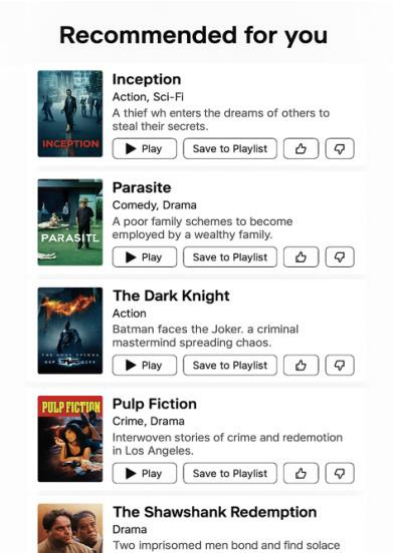
1.2. Items Presented & User Interface

The recommender system will present five personalized movie recommendations at a time through a simple web interface. Each movie will be displayed with its poster, title, genre tags, and a brief description. Users will be able to interact with the interface by rating watched movies (1–5 stars), giving thumbs up/down feedback, saving items to a watchlist, skipping items, and searching for movies by title or genre.

Although no actual interface will be implemented at this stage, the interaction can be simulated in a user study. Example components of the interface include:

- Movie title and release year
- Duration
- "Watch now" or "Download" button
- Thumbs up / down feedback
- "Save for later" and "Skip" options

User interface Mockup：



1.3. User Feedback & Interaction

Users provide explicit feedback via star ratings and thumbs up/down interactions. This data is used to update their user profiles and refine future recommendations. Implicit feedback such as viewing duration, skip rate, and play frequency may also be logged for additional refinement. These interactions are stored in a user-item matrix and used in collaborative filtering to re-rank future content. The project will also explore sequential models to predict the “next movie to watch” based on recent user activity within a session.

1.4. Dynamic Scenario & Cold Start

The recommendation model will be updated periodically (e.g., daily) as new user ratings are collected.

For new users**,** the system includes an onboarding quiz (e.g., “Select three favorite genres”) to build an initial profile. If the user skips the quiz, popular or highly-rated movies will be recommended by default until enough interaction data is gathered for personalization.

For new items**,** where no user ratings exist, the system rely on their text metadata (such as title and description) and apply SBERT to generate semantic embeddings. These embeddings allow the system to match new movies with the preferences of existing users based on content similarity, effectively addressing the cold-start problem (Zhou et al., 2019).

1.5. Business & Revenue Model

The system can generate revenue through several channels. A freemium subscription model allows users to choose between an ad-supported version and a premium ad-free experience. Creators may promote movies or trailers through sponsored placements. Additionally, aggregated audience insights may be offered to content providers via opt-in data analytics services. Another worth to mention business is the affiliate links to streaming platforms or ticket sales, and premium features like advanced filtering and early access to recommendations.

2. Datasets

2.1. Dataset Description

The project will use the MovieLens 1M dataset from Hugging Face, which contains 1 million ratings of 3,900 movies by 6,040 users. This dataset provides tremendous user–item interactions suitable for collaborative filtering, hybrid models, and sequential recommendation approaches.

Fields Used：

- userId: Unique ID for each user
- movieId: Unique ID for each movie
- rating: Integer from 1 to 5
- timestamp: Unix time of interaction
- title: Movie title and release year
- genres: Pipe-separated string of genres (e.g., Action|Adventure|Sci-Fi)

2.2. Use in Recommendation System

We will use the userId, movieId, and rating fields to build a user–item interaction matrix. Ratings of 4–5 are considered as positive signals, rating of 1–2 are treated as negative, and rating of 3 will be marked as neutral or unknown. Genre tags will be converted into one-hot encoded vectors to compute similarity between movies based on genre overlap. Additionally, movie titles and descriptions will be processed using pre-trained NLP models (e.g., BERT) to generate semantic embeddings. These content features help the system to recommend similar movies based on user interests, particularly helpful in cold-start scenarios.

Nevertheless, timestamp is another useful field that allow the system to capture temporal dynamics in user behavior, making recommendations more timely and context-aware. Temporal filtering will be deployed to prioritize recently released movies and re-rank items based on the recency of user interactions. Furthermore, user interactions can be organized chronologically to support session-aware sequential models such as RNNs, GRUs, or Transformer-based recommenders (e.g., SASRec, BERT4Rec). These models learn patterns in the sequence of user actions to predict the user preferences during a viewing session (Kang & McAuley, 2018; Sun et al., 2019).

2.3. Limitations

While the MovieLens dataset from Hugging Face is widely used for academic research, it has several limitations. Firstly, the dataset represents only a subset of the original platform's data, which means it may not fully reflect the diversity of users and their interaction behaviors. It is possible that only the most active users or most-rated movies are presented, creating sampling bias and reducing generalizability to real-world applications.

Secondly, the dataset is often sanitized to remove noise or anomalies. While this improves consistency, it can also eliminate real-world complexities such as inconsistent ratings, cold-start anomalies, or bot activity, making the dataset less realistic for recommendation.

Thirdly, platforms like Hugging Face often structure datasets based on specific prediction tasks with predefined metrics and static train/test splits. This can lead to overfitting problems, as it tends to narrow the tasks and may not be able to evaluate broader system goals such as long-term user engagement, novelty, or diversity, resulting failure to build a truly effective recommender system.

3. Methods

To build an effective movie recommender system using the MovieLens 1M dataset, this project will compare below four stand-alone models and two hybrid approaches that integrates collaborative filtering, content-based filtering, and a sequential variant to address both user personalization and dynamic consumption behaviors.

3.1. Baseline: User-Based Collaborative Filtering (UBCF)

This method recommends movies to a user based on the preferences of similar users, identified using k-nearest neighbors (KNN) on the user–item interaction matrix. UBCF is well-suited to this dataset due to the dense user–item rating data, making it effective at capturing like-minded user groups. It also serves as a strong baseline due to its interpretability and ease of implementation using libraries like surprise or scikit-learn. However, UBCF has limitations in handling cold-start users and performance degradation on sparse interaction data (Ricci et al., 2015).

3.2. Content-Based Filtering (CBF) with Genre and Textual Features

This method recommends movies that are similar to those a user has rated highly in the past, based on genre vectors and optionally movie title/description embeddings. Genres will be converted to one-hot vectors for traditional similarity measures (e.g., cosine similarity), while richer semantic relationships can be captured using pre-trained BERT embeddings applied to textual metadata. This method is especially useful for cold-start users or those with niche interests. However, it does not incorporate the behavior of other users, limiting collaborative diversity (Zhang et al., 2019; Sun et al., 2019).

3.3. Matrix Factorization (SVD/ALS)

Matrix factorization techniques like Singular Value Decomposition (SVD) or Alternating Least Squares (ALS) decompose the user–item interaction matrix into latent user and item factors, capturing implicit dimensions such as genre preference or actor bias. These methods are more scalable and expressive than neighborhood-based CF and can uncover deeper patterns (e.g., a user prefers sci-fi with strong female leads) (Zhang et al., 2019). Implementation will use the surprise SVD model or LightFM.

3.4. Hybrid Recommender System

3.4.1. Hybrid model 1: Collaborative Filtering + Content-Based Filtering (CF + CBF)

This hybrid model integrates collaborative filtering and content-based filtering to enhance the quality and robustness of recommendations. Collaborative filtering, implemented via SVD, captures patterns in the user–item interaction matrix, learning latent user preferences based on ratings (Zhang et al., 2019). On the other hand, content-based filtering utilizes textual metadata of movies, such as titles and descriptions, and represents them using pre-trained language model embeddings like BERT (Sun et al., 2019).

The model can deliver more accurate and personalized recommendations by combining both approaches. Collaborative filtering performs well when user rating history is available, while the content-based component addresses cold-start issues for new items by leveraging movie content. The final prediction is computed as a weighted combination of both scores, balancing collaborative knowledge and semantic similarity effectively (Burke, 2002).

3.4.2. Hybrid Model 2: Content-Based Filtering + Sequential Model

The second hybrid model blends content-based filtering with a sequential model to support context-aware recommendations. The content-based component constructs user profiles based on previously liked movies, using metadata such as genres and semantic embeddings. In parallel, the sequential model learns from the temporal order of a user's past interactions. The combination allows the system to model short-term user behavior and preferences during specific viewing sessions. It ensures that recommendations reflect both the user's overall interests and their current session context. This is particularly beneficial under the circumstances where the system needs to adapt to recent activities or changing preferences, such as recommending the next movie in a binge-watching session (Quadrana et al., 2018).

The formula for hybrid models is shown as:

$$\text{Final Score} = \alpha \cdot \text{CF score} + (1 - \alpha) \cdot \text{CBF or Seq score}$$

(Burke, 2002)Note: Alpha will be determined using Gridsearch.

3.5. Sequential Recommender Model

For dynamic personalization, the project will explore a sequential model such as a Recurrent Neural Network (RNN) or Transformer-based model like SASRec. These models use temporally ordered interactions to predict the next likely item in a user's session, enabling more contextual, session-aware recommendations. This is feasible because the MovieLens dataset includes timestamped interactions, supporting sequence modeling. While more challenging to implement and evaluate, this approach reflects real-world user behavior such as binge-watching or session-based viewing (Kang & McAuley, 2018).

Recommender Models Pipeline

Model	Fields Used	Stage of Pipeline
User-Based Collaborative Filtering (UBCF)	User–item matrix	Input to Collaborative Filtering
Content-Based Filtering (CBF)	Genre one-hot vectors	Input to Content-Based Filtering
CBF with BERT	Title/description embeddings	Input to Content-Based Filtering
Matrix Factorization (SVD)	User–item matrix	Input to Collaborative Filtering
Sequential Model (e.g., SASRec)	Timestamps (sequential models)	Training input for sequence model
Hybrid Model 1 (CF + CBF)	User–item matrix, Genre vectors, Title embeddings	Scoring stage
Hybrid Model 2 (CBF + Sequential)	Genre vectors, Timestamps	Scoring stage

This project does not incorporate knowledge-based recommender systems, as these are more appropriate for domains that require constraint-based reasoning or explicit knowledge (e.g., travel booking, insurance) (Adomavicius & Tuzhilin, 2005).

4. Evaluation Metrics

To evaluate the effectiveness of the proposed movie recommendation system, we consider two categories of metrics: those that assess the performance of the underlying recommendation model using historical data, and those that assess the quality of the resulting recommender system based on user interactions or simulations.

4.1 Recommendation Model Metrics (Offline Evaluation)

These metrics are computed on the MovieLens dataset using a held-out test set of historical user ratings. They are designed to assess how well the model predicts user preferences.

- Precision@N and Recall@N: These Top-N metrics measure the accuracy of the top-N recommendations. Precision@5 calculates how many of the top 5 recommended movies were actually rated positively by the user (e.g., 4 or 5 stars), while Recall@5 measures how many of all the relevant movies were included in the top 5 recommendations.
- NDCG@N (Normalized Discounted Cumulative Gain): This metric evaluates the quality of the ranking of recommendations by taking the position of correct items into account. Items correctly recommended near the top of the list are weighted more heavily.
- RMSE (Root Mean Square Error): For models that predict explicit ratings (e.g., matrix factorization), RMSE assesses the average deviation between the predicted and actual ratings, reflecting the model's rating prediction accuracy.

- Per-user averaged metrics: To ensure fairness across users with different interaction volumes, we will report precision, recall, and NDCG averaged per user.

These metrics allow comparison between collaborative filtering, content-based filtering, and hybrid methods under consistent conditions using cross-validation (Adomavicius and Tuzhilin, 2005).

4.2 Recommender System Metrics (Simulated or Online Evaluation)

While offline metrics assess prediction performance, recommender systems in real applications must also be evaluated in terms of user experience and engagement. These metrics reflect how the system performs in real usage scenarios or simulations.

- Click-Through Rate (CTR): Simulated by tracking whether users (or synthetic users in test scripts) interact with the recommended movies. This metric capture user interest in presented recommendations.
- Engagement Metrics: If user feedback is modeled (e.g., thumbs up/down, skip), we will measure acceptance rate, skip rate, and positive feedback rate as indicators of user satisfaction.
- Coverage: Measures the proportion of items or users the system is able to serve effectively. High coverage means the system doesn't just recommend the same items to everyone.
- Novelty and Diversity: These metrics assess how varied and fresh the recommendations are. For example, the average popularity of recommended items can be used to estimate novelty.
- Session Success Rate: For sequential models, we can define a session as "successful" if the model correctly recommends the next movie in the sequence, based on temporal ordering in the dataset(Ricci, Rokach and Shapira, 2015).

By combining these two types of evaluation, the system can be assessed both in terms of predictive accuracy and user-oriented effectiveness, providing a comprehensive view of performance (Zhang et al., 2019).

4.3 Metric Prioritization and Trade-offs

Among the metrics discussed, Precision@N and NDCG@N are important metrics for selecting the best model, as they directly reflect the user's perceived quality of the top-ranked recommendations. These are particularly important in user-facing applications, where users typically see only a limited set of suggestions (Gunawardana & Shani, 2015).

However, trade-offs exist. For instance, optimizing for Precision@N may lead to popularity bias, reducing diversity. In contrast, improving novelty and diversity may come at the cost of precision. RMSE is useful for rating prediction models like SVD, but it may not align with the real-world goal of recommending a short, ranked list of relevant items. To

manage this, per-user averaged NDCG@5 and per-user averaged Precision@5 will be our final metrics, as it balances ranking quality and fairness across users (Gunawardana & Shani, 2015; Zhang et al., 2019).

To prioritize among the system-level metrics, we identify Click-Through Rate (CTR) as the most important indicator of user engagement, since it directly reflects user interest and interaction with the recommended items. In real-world systems, CTR is often used to measure success in driving engagement. While coverage, novelty, and diversity provide complementary insights, they are considered secondary to CTR and engagement, as they trade off with accuracy and are harder to optimize simultaneously. Additionally, session success rate will be emphasized only when sequential recommendation is used, where capturing temporal relevance is essential (Zhang et al., 2019; Gunawardana & Shani, 2015).

4.4 Computational Considerations

Computational requirements also play a key role in evaluation. Lightweight models such as user-based collaborative filtering or genre-based content filtering offer fast predictions suitable for near real-time use. In contrast, matrix factorization models (e.g., SVD) or hybrid systems require periodic offline training but can provide better accuracy once deployed. Sequential models such as RNNs or Transformers (e.g., SASRec) offer contextual personalization but come with significantly higher computational overhead and longer inference time, making them suitable only as optional enhancements or for batch-mode re-ranking (Zhang et al., 2019; Gunawardana & Shani, 2015)..

Therefore, the final model/system selection will consider both quantitative performance (via metrics like per-user NDCG@5) and operational feasibility, ensuring the recommender is both accurate and scalable in realistic use scenarios.

5. User Study Design

To complement offline metrics and assess user satisfaction, we propose an informal user study to evaluate the recommender system through a simulated user interface.

5.1 Study Setup and Procedure

Participants will be invited to interact with a mock web interface simulating the final recommender system. Each user will be shown a curated list of 5 movie recommendations generated by one of the trained models (e.g., content-based, hybrid, or sequential). For each recommendation, users can:

- View the movie title, poster, genre(s), and a short description.
- Click to simulate "watching" the movie.
- Rate the recommendation (e.g., 1 to 5 stars).
- Provide binary feedback: "Like" or "Skip".

Each participant will interact with at least two different recommendation lists, generated using different underlying methods (e.g., content-based vs. hybrid), allowing us to compare user preference across models.

5.2 Questionnaire and User Feedback

After interacting with the system, each participant will complete a brief questionnaire to provide subjective feedback. The questionnaire includes:

- How relevant were the movie recommendations to your interests? (1–5 scale)
- How likely would you be to watch one of the recommended movies? (1–5 scale)
- Did you feel the recommendations were too repetitive or too similar?
- Did you discover any movies you had not heard of but found interesting?
- Which recommendation list did you prefer, and why?
- Suggestions for improving the recommendation system?

Optionally, we may ask users about perceived novelty, diversity, and satisfaction in free-text responses.

5.3 Evaluation Goals

The study aims to evaluate the recommender system from a user-centered perspective. It focuses on measuring how relevant and engaging users find the recommendations, comparing different models based on users' subjective satisfaction rather than relying solely on quantitative metrics like precision or recall. It also seeks to uncover qualitative gaps in the system, such as a lack of novelty or an overemphasis on popular titles.

6. Timeline & Milestones

Week	Milestone
6-7	Finish preprocessing, implement UBCF and content-based
8	Implement hybrid and matrix factorization models
9	Evaluate models (Precision@N, nDCG, Hit Rate, etc.)
10	Simulated user study and feedback analysis
11	Report writing and polish presentation/demo

7.References

Adomavicius, G. and Tuzhilin, A., 2005. *Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions*. IEEE Transactions on Knowledge and Data Engineering, 17(6), pp.734–749. https://doi.org/10.1109/TKDE.2005.99

Burke, R., 2002. *Hybrid recommender systems: Survey and experiments*. User Modeling and User-Adapted Interaction, 12(4), pp.331–370. https://doi.org/10.1023/A:1021240730564

Harper, F.M. and Konstan, J.A., 2015. *The MovieLens Datasets: History and Context*. ACM Transactions on Interactive Intelligent Systems (TiiS), 5(4), pp.1–19. https://doi.org/10.1145/2827872

Kang, W.-C. and McAuley, J., 2018. *Self-Attentive Sequential Recommendation*. In: Proceedings of the 2018 IEEE International Conference on Data Mining (ICDM). IEEE, pp.197–206. https://doi.org/10.1109/ICDM.2018.00035

Quadrana, M., Karatzoglou, A., Hidasi, B. and Cremonesi, P., 2018. *Personalizing session-based recommendations with hierarchical recurrent neural networks*. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18). ACM, pp.130–138. https://doi.org/10.1145/3159652.3159656

Ricci, F., Rokach, L. and Shapira, B., 2015. *Recommender Systems Handbook*. 2nd ed. Springer.

Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W. and Jiang, P., 2019. *BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer*. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM). ACM, pp.1441–1450. https://doi.org/10.1145/3357384.3357895

Zhang, S., Yao, L., Sun, A. and Tay, Y., 2019. *Deep Learning Based Recommender System: A Survey and New Perspectives*. ACM Computing Surveys, 52(1), pp.1–38. https://doi.org/10.1145/3285029

Zhou, G., Mou, N., Fan, Y., Pi, Q., Bian, W., Zhou, C. and Gai, K., 2019. *Deep interest network for click-through rate prediction*. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, pp.1059–1068. https://doi.org/10.1145/3219819.3220007

Gunawardana, A. and Shani, G., 2015. Evaluating recommender systems. In Ricci, F., Rokach, L. and Shapira, B. (eds.) Recommender Systems Handbook. 2nd ed. Springer, pp. 265–308. https://doi.org/10.1007/978-1-4899-7637-6_8