

# User-based collaborative filtering for movie recommendation: implementation and evaluation on the MovieLens dataset

Nur Fulin<sup>1</sup>, Tarik Tinjak<sup>1</sup>, Emine Yaman<sup>1</sup>

<sup>1</sup> Department of Computer Science and Engineering, International University of Sarajevo, Sarajevo, Bosnia and Herzegovina

\*Corresponding author E-mail: [eyaman@ius.edu.ba](mailto:eyaman@ius.edu.ba)

## ABSTRACT

Recommendation systems play an important role in modern data-mining applications by supporting personalized discovery within large item collections. This paper implements an end-to-end user-based collaborative filtering pipeline using the MovieLens dataset. The pipeline includes schema validation, preprocessing, sparsity analysis, positive-only leave-one-out splitting, user-item matrix construction, user-user similarity computation, rating prediction, Top-K recommendation generation, and ranking evaluation. Rather than introducing a new recommendation algorithm, the study presents a reproducible empirical analysis of how evaluation protocol, sparsity handling, neighborhood size, rating normalization, and similarity filtering influence recommendation performance. Under the comparable full-user protocol, the strongest raw-rating configuration achieved HR/Recall@10 of 0.355 and NDCG@10 of 0.216, outperforming the mean-centered reference configuration and the expected random-ranking baseline. The results emphasize that transparent experimental design and controlled comparison are essential for interpreting collaborative-filtering performance.

**Keywords:** recommendation systems; user-based collaborative filtering; MovieLens; cosine similarity; Top-K evaluation

## 1. Introduction

Recommendation systems help users discover relevant items within large digital catalogs, such as media-streaming services, e-commerce platforms, social networks, and educational systems. Rather than requiring the users to perform extensive searches in order to find items matching their preferences among thousands of possible choices, recommender systems learn the preferences and return ranked items based on how likely it is that a user will like the item. In data mining, this problem is crucial because it unites all the essential parts – large-scale data processing, learning of similarities, prediction, ranking, and evaluation.

One of the most popular recommendation algorithms is collaborative filtering, which utilizes mainly user-item interaction data. For example, in the movie domain, collaborative filtering takes into account user ratings in order to predict the user's likes. This method is appealing since it provides recommendations irrespective of any detailed item information. At the same time, some of the critical issues faced by collaborative filtering are sparsity, cold-start problem, noise in ratings, and sensitivity to evaluation protocols.

Recent advances in recommendation systems have demonstrated that sophisticated neural methods do not necessarily outperform carefully tuned classical methods [1]. Reproducible evaluation frameworks have also become important because differences in dataset splitting, negative sampling, and metric calculation can change conclusions [2]. Several studies further warn that reported progress in recommender systems can be difficult to reproduce when baselines are weak or experimental settings are inconsistent [3].

Therefore, evaluation cannot be considered an extra step but a key element of recommender system research. Literature survey highlights the importance of evaluating recommender systems on the basis of their intended tasks and explains the need for careful interpretation of the evaluation metrics used in the offline stage [4]. Classical baselines such as implicit matrix factorization can be competitive when well-tuned [5], whereas misuse of the offline protocol can cause leakage and performance inflation [6]. This calls for designing a simple yet transparent pipeline.

The principal contribution of this study is a reproducible and controlled empirical analysis of design decisions in user-based collaborative filtering. Rather than proposing a new recommendation algorithm, the study integrates preprocessing, sparsity handling, positive-only leave-one-out splitting, user-user similarity computation, prediction, Top-K ranking, and evaluation within a single transparent pipeline. This unified setup makes it possible to isolate the effects of individual methodological choices under clearly defined evaluation conditions.

The study makes four specific contributions. First, it provides an end-to-end implementation that can be reproduced using the MovieLens dataset and standard Python libraries. Second, it compares raw-rating and mean-centered similarity variants, together with positive-only and unrestricted similarity filtering, under the same full-user evaluation protocol. Third, it quantifies the effect of minimum user/item activity thresholds and demonstrates the resulting accuracy-versus-coverage trade-off. Fourth, it distinguishes exploratory hyperparameter analysis from comparable final evaluation and contextualizes the results using a random-ranking baseline.

Accordingly, the study addresses the following research questions:

**RQ1:** How do raw and mean-centered ratings affect Top-K recommendation performance under a fixed evaluation protocol?

**RQ2:** How does positive-similarity filtering affect user-based collaborative filtering?

**RQ3:** How do neighborhood size and minimum activity thresholds influence ranking quality and data coverage?

The rest of the paper is structured as follows. Section II discusses related work. Section III introduces the methodology. Section IV deals with the experimental framework. Section V covers the results of the study. Section VI highlights the limitations of the research and threats to validity. Finally, Section VII summarizes the findings.

## 2. Related work

### 2.1. Reproducibility and evaluation in recommender systems

Increasingly, the field of recommendation system research has started to focus on reproducibility and fair evaluation. For example, a comparison between neural collaborative filtering and matrix factorization reveals that well-tuned classical techniques might be better than the new techniques [1]. Elliot is an approach for achieving reproducibility in recommenders through algorithm standardization, data split, parameter tuning, and evaluation metrics [2]. The reproducibility study showed that much of the reported progress is difficult to reproduce and may disappear when stronger baselines are used [3].

The problem of evaluating recommender systems is challenging. There have been attempts to organize evaluation based on its goals, stakeholders, and metric types [4]. Some benchmarks suggest that good results in MovieLens do not mean generalization [5]. Moreover, temporal leakage can occur if a model utilizes future knowledge while being evaluated offline and produces overly optimistic outcomes [6]. Modern evaluation review suggests that offline evaluation, online experiments, user studies, fairness, novelty, and diversity reflect various aspects of recommender quality [7], [8].

## 2.2. Collaborative filtering, sparsity, and model improvements

Collaborative filtering continues to be very relevant as it captures preferences from the interaction dataset. Collaborative filtering with consideration for fairness is an area of research that has been explored even without sensitive user data [9]. There have been collaborative filtering models which try to leverage interaction data along with semantic information to enhance the recommendation system [10]. The embedding technique has been widely used in collaborative filtering models, such as UI2vec and VUI2vec. These models represent the users and items in their respective latent spaces to capture non-linear preference patterns [11]. Matrix factorization techniques have also played a vital role in collaborative filtering research [12].

The main issue faced in collaborative filtering is sparsity in the data set. To combat the sparsity problem, there have been approaches like metadata-embedded deep collaborative filtering to incorporate the item metadata into user-item interaction [13]. Variational models can be used for probabilistic learning through latent variable analysis in collaborative filtering [14]. Hybrid models try to solve the problem associated with rating alone by integrating both collaborative and content-based approaches [15]. Some of the other collaborative filtering models use genetic algorithms, reliability deep learning, user experience signals, fuzzy neural networks, and survey synthesis [16]-[20].

## 2.3. Broader extensions and contextual positioning

Modern research even takes recommender systems from user-item recommendations toward multi-criteria, item-based, privacy-preserving, graph-based, causal, context-aware, and tourism-oriented recommendations [21]-[29]. While they cannot serve as baselines directly for the proposed user-based MovieLens experiment, these articles illustrate ways in which the approach might evolve and give context to the contribution presented in the current study. Modern literature that discusses surveys and tutorials helps understand the place of classical collaborative filtering techniques in the overall context.

Articles related to the Netflix Prize give reasons why rating recommendation should still be taken into account [26], whereas modern surveys discuss various aspects of recommender systems and their relation to artificial intelligence [27], [30]. For this reason, the related literature here is used to explain why reproducibility of recommendation evaluation is important and how the approach proposed in this paper differs from existing approaches.

Table 1. Summary of related-work themes used to position this paper

| Theme                                | Representative references | Relevance to this paper                                                                    |
|--------------------------------------|---------------------------|--------------------------------------------------------------------------------------------|
| Reproducibility and evaluation       | [1]-[8]                   | Motivates transparent splitting, sampled evaluation, and standard ranking metrics.         |
| Collaborative filtering and sparsity | [9]-[20]                  | Motivates user-based CF, sparsity analysis, filtering, and simple baseline interpretation. |
| Broader extensions and context       | [21]-[30]                 | Provides context for future extensions; not treated as direct baseline comparisons.        |

**Table 1** organizes the reviewed literature into three groups that directly support the study design. The first group motivates reproducible evaluation and standardized ranking metrics; the second supports the use of collaborative filtering and sparsity handling; and the third positions the study within broader recommender-system developments without treating those methods as directly comparable baselines.

## 3. Methodology

The methodology follows the complete data-mining workflow shown in **Figure 1**. The pipeline begins with loading MovieLens files, validating the schema, cleaning ratings, analyzing sparsity, filtering sparse users and items, creating a positive-only train-test split, constructing the user-item matrix, computing user-user cosine similarity, predicting scores for unseen movies, generating Top-K recommendations, and evaluating the ranking quality.

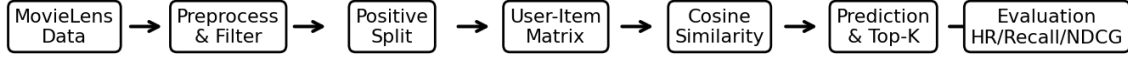


Figure 1. Implemented recommendation pipeline

As illustrated in **Figure 1**, each stage produces the input required by the next stage. Preprocessing and filtering determine the interaction data used to construct the user-item matrix, while similarity computation supports prediction and Top-K ranking. The final evaluation stage measures whether the held-out positive item is recovered and how highly it is ranked.

### 3.1. Dataset loading and validation

The MovieLens dataset contains four files: ratings, movies, links, and tags. The ratings file contains explicit user ratings for movies and provides the primary interaction data used by the collaborative-filtering model. The movies file maps movie IDs to titles and genres. The links and tags files provide supplementary metadata and are not used in the present user-based collaborative-filtering implementation. Before preprocessing, the pipeline verifies that the ratings file contains `userId`, `movieId`, and `rating`, and that the movies file contains `movieId` and `title`.

### 3.2. Preprocessing and sparsity handling

The preprocessing stage retains the columns required for collaborative filtering, removes missing values, and eliminates duplicate user-movie pairs. Because recommender-system datasets are inherently sparse, sparsity is measured before filtering. It is calculated as one minus the ratio of observed ratings to all possible user-item pairs. Filtering is done to reduce noise in the dataset by eliminating those users and items having less than a certain number of ratings.

### 3.3. Positive-only leave-one-out split

For Top-K evaluation, not every observed rating should be interpreted as a liked item. This study therefore defines positive interactions as ratings of 4.0 or higher. For each eligible user, one positive interaction is held out as the test item, while the remaining interactions are retained for training. The evaluation then measures whether the model can recover a movie that the user positively rated.

### 3.4. User-Item matrix and cosine similarity

The training data is encoded in terms of a user-movie matrix where each row is associated with the users and each column corresponds to the movies, while the value is the rating received for a movie by a user. The missing cells correspond to movies that have not been rated by a particular user. User-based Collaborative Filtering algorithm uses this matrix to find similarities between the rows. Cosine Similarity is adopted since it computes the orientation of the preference vector and is widely used in memory-based Collaborative Filtering.

For two users  $u$  and  $v$ , raw cosine similarity is written as:

$$\text{sim}(u, v) = \frac{(r_u \cdot r_v)}{(\|r_u\|^2 \|r_v\|^2)} \quad (1)$$

where  $r_u$  and  $r_v$  are the rating vectors of users  $u$  and  $v$ , and  $\|r_u\|_2$  is the Euclidean length of user  $u$ 's rating vector. The system also evaluates a mean-centered variant:

$$\text{sim}_{c(u,v)} = \frac{((r_u - \mu_u) \cdot (r_v - \mu_v))}{(\|r_u - \mu_u\|^2 \|r_v - \mu_v\|^2)} \quad (2)$$

where  $\mu_u$  and  $\mu_v$  are the mean ratings of users  $u$  and  $v$ , respectively. This mean-centered cosine formulation is closely related to Pearson correlation because both methods compare users after subtracting their respective mean ratings. However, the two measures are not treated as interchangeable in this study. Missing ratings represent unknown preferences rather than negative feedback. Therefore, mean-centering is performed before

missing entries are filled with zero. After centering, zero represents the absence of observed centered information rather than a zero-star rating.

### 3.5. Rating prediction and Top-K recommendation

The prediction function estimates a target user's preference score for an unseen movie. First, it identifies users who have rated the target movie. It then retrieves the similarities between these users and the target user, keeps the positive similarities only, takes the top-N similar users and computes a similarity-weighted prediction. In case of mean-centering, target user's mean score is added to the weighted average of neighbors' scores after being adjusted by their mean. If it is not mean-centered, a straight weighted average of neighbors' rating scores is calculated. The two prediction variants can produce different rankings and are therefore evaluated experimentally. The last step includes ranking all unknown movies according to their score predictions and taking the Top-K movies to recommend.

This part connects similarity and recommendation processes because after predicting score values, it ranks all candidate movies using these values. This ranking is done among only unknown movies, thus excluding movies with known scores.

### 3.6. Evaluation metrics

Evaluation metrics considered for the model include HR@10, Recall@10, and NDCG@10. The HR@10 metric calculates if the withheld positive test instance can be seen in the Top-10 recommendations list. The Recall@10 evaluates the proportion of relevant items that appear in the Top-10 list. In this leave-one-out experiment design with precisely one relevant test instance for each user, the value of HR@10 is identical to Recall@10. Both metrics are reported for completeness and consistency with common recommender-system evaluation practice. For a ranked list,

$$DCG@K = \sum_{i=1}^K \frac{rel_i}{\log_2(i+1)} \quad (3)$$

, and

$$NDCG@K = \frac{DCG@K}{IDCG@K} \quad (4)$$

, where  $rel_i$  equals 1 if the item at rank  $i$  is relevant and 0 otherwise. In the single-positive setup, NDCG rewards models that place the held-out item near the top of the recommendation list.

## 4. Experimental setup

The code was written in Python, using the Pandas, NumPy, and scikit-learn libraries. The experiments were run on the MovieLens dataset. For each user, the held-out positive item had a rating of at least 4.0. Ninety-nine negative items were sampled uniformly from movies that the user had not rated in the training set, excluding the held-out positive item. The positive item was then added to the sampled negatives to form a candidate set of 100 movies.

Table 2. Dataset statistics and preprocessing output

| Stage                 | Users | Movies | Ratings | Notes                                                       |
|-----------------------|-------|--------|---------|-------------------------------------------------------------|
| Original dataset      | 610   | 9,724  | 100,836 | Raw MovieLens ratings after validation and duplicate check. |
| After 20/20 filtering | 566   | 1,297  | 67,225  | Users/items with insufficient support removed.              |
| Training set          | 566   | 1,297  | 66,659  | One positive item held out per user.                        |
| Test set              | 566   | --     | 566     | Minimum test rating is 4.0.                                 |

As shown in

**Table 2**, filtering reduces the dataset from 610 users, 9,724 movies, and 100,836 ratings to 566 users, 1,297 movies, and 67,225 ratings. Although this removes a substantial number of low-support movies, it produces a denser interaction space with more reliable overlap for user-similarity estimation.

The original user-item space is highly sparse. With 610 users and 9,724 movies, the total possible number of ratings is much larger than the observed 100,836 ratings. The measured sparsity before filtering is 0.9830, meaning that approximately 98.3% of the matrix is missing. This motivates filtering and careful evaluation.

Table 3. Evaluation configurations used in the experiments

| Experiment                   | Main setting                                                               | Purpose                                                                                          |
|------------------------------|----------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------|
| Reference sampled evaluation | 20 neighbors, mean-centering, positive similarities, 99 negatives          | Provide a reproducible reference configuration, not necessarily the best final model.            |
| Neighbor tuning              | 10, 20, and 30 neighbors on a random 100-user sample with 30 negatives     | Measure neighborhood sensitivity under a faster exploratory setting.                             |
| Model variations             | Raw vs. mean-centered; positive-only vs. all similarities on all 566 users | Provide the comparable full-user model comparison used to identify the strongest tested setting. |
| Filtering thresholds         | 10/10, 20/20, and 30/30 min ratings                                        | Study data quality vs. data quantity.                                                            |

**Table 3** distinguishes the four experimental components used in the study. The reference evaluation and model-variation experiments use the full set of 566 eligible users, whereas neighborhood tuning uses a smaller sample for computational efficiency. The filtering-threshold experiment separately examines the trade-off between retaining more data and obtaining more reliable similarity estimates.

## 5. Results and discussion

### 5.1. Reference sampled evaluation and random baseline

The first full-user evaluation uses a reference configuration with positive-only leave-one-out evaluation, 99 sampled negatives, 20 neighbors, mean-centered ratings, and positive similarities only. This configuration evaluated 566 users and achieved  $HR/Recall@10 = 0.300$  and  $NDCG@10 = 0.165$ .  $HR@10$  and  $Recall@10$  are equal because each user has exactly one held-out positive item. A random ranker over 100 candidate items would be expected to achieve  $HR/Recall@10 = 0.100$  and  $NDCG@10 \approx 0.045$ , so the collaborative-filtering model is meaningfully above chance.

Nonetheless, it should be noted that this particular configuration is not claimed to be the optimal one. According to the comparable full-user model variation experiment, the raw rating configurations yielded  $HR/Recall@10 = 0.355$  and  $NDCG@10 = 0.216$ . Consequently, the best full-user configuration is the raw-rating user-based CF variant, whereas the mean-centered configuration is still applicable as a theoretical reference point.

Table 4. Random baseline, reference configuration, and strongest full-user result

| Metric          | Expected random baseline                     | Observed full-user results                                                                            |
|-----------------|----------------------------------------------|-------------------------------------------------------------------------------------------------------|
| $HR/Recall@10$  | 0.100                                        | 0.300 reference configuration; 0.355 strongest raw-rating configuration                               |
| $NDCG@10$       | 0.045                                        | 0.165 reference configuration; 0.216 strongest raw-rating configuration                               |
| Evaluated users | 566 candidate evaluations assumed            | 566 users in the full-user experiments                                                                |
| Interpretation  | Random chance with 1 positive + 99 negatives | Both observed models are above chance; raw ratings were strongest among comparable full-user variants |

**Table 4** shows that both collaborative-filtering configurations outperform the expected random-ranking baseline. The strongest raw-rating configuration improves HR/Recall@10 from the random expectation of 0.100 to 0.355 and NDCG@10 from approximately 0.045 to 0.216, indicating that the model learns meaningful user-preference structure rather than ranking candidates randomly.

## 5.2. Hyperparameter tuning

**Table 5** and **Figure 2** show a monotonic increase in all three ranking metrics as the neighborhood size increases from 10 to 30. On this 100-user exploratory sample, HR/Recall@10 increases from 0.540 to 0.620, while NDCG@10 increases from 0.280 to 0.316. This suggests that additional neighbors provided useful collaborative evidence in this sample. However, because the experiment used fewer users and fewer negative candidates than the full evaluation, the values should be interpreted as sensitivity results rather than final model-selection evidence.

Therefore, the tuning results can only be viewed as a sensitivity analysis and not as an evaluation method for model selection. While they indicate that larger neighborhoods could yield better results, further experiments would have to be done for the full-user protocol. Each neighbor value would need to be repeated for different seeds, and confidence intervals could also be calculated. For this reason, the paper makes no claims about the global optimum being 30 neighbors.

| Table 5. Exploratory neighbor sensitivity analysis on a 100-user sample |          |           |          |                 |
|-------------------------------------------------------------------------|----------|-----------|----------|-----------------|
| Neighbors                                                               | HR@10    | Recall@10 | NDCG@10  | Evaluated users |
| 10                                                                      | 0.540000 | 0.540000  | 0.279939 | 100             |
| 20                                                                      | 0.570000 | 0.570000  | 0.297414 | 100             |
| 30                                                                      | 0.620000 | 0.620000  | 0.316220 | 100             |

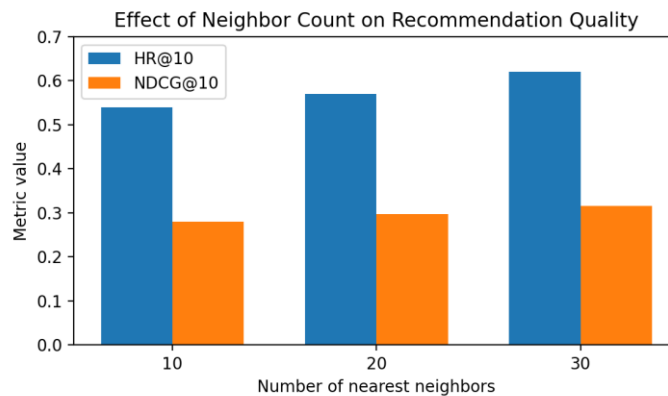


Figure 2. HR@10 and NDCG@10 under different nearest-neighbor values

## 5.3. Comparable full-user model variation comparison

**Table 6** and **Figure 3** compare the four model variants under the same full-user sampled-evaluation protocol. Raw-rating configurations achieve the highest HR/Recall@10 and NDCG@10. Mean-centering therefore does not improve ranking evaluation in this filtered dataset, although it remains theoretically useful for controlling differences in users' rating scales.

These results illustrate why experimental comparisons of modeling methods are critical. Even though a modeling choice makes theoretical sense, it may not provide the best outcome empirically. In the raw rating example, positive similarity filtering and all similarities evaluation provided the same outcome since the cosine similarity of raw nonnegative ratings was nonnegative, meaning there were no negative similarities to exclude. However, mean-centered vectors might have negative elements; hence, positive-only and all similarities variations would differ.

Table 6. Comparable full-user model variation comparison

| Model variation                       | HR@10    | Recall@10 | NDCG@10  | Evaluated users |
|---------------------------------------|----------|-----------|----------|-----------------|
| Mean-centered + positive similarities | 0.300353 | 0.300353  | 0.165254 | 566             |
| Mean-centered + all similarities      | 0.310954 | 0.310954  | 0.172216 | 566             |
| Raw ratings + positive similarities   | 0.355124 | 0.355124  | 0.215520 | 566             |
| Raw ratings + all similarities        | 0.355124 | 0.355124  | 0.215520 | 566             |

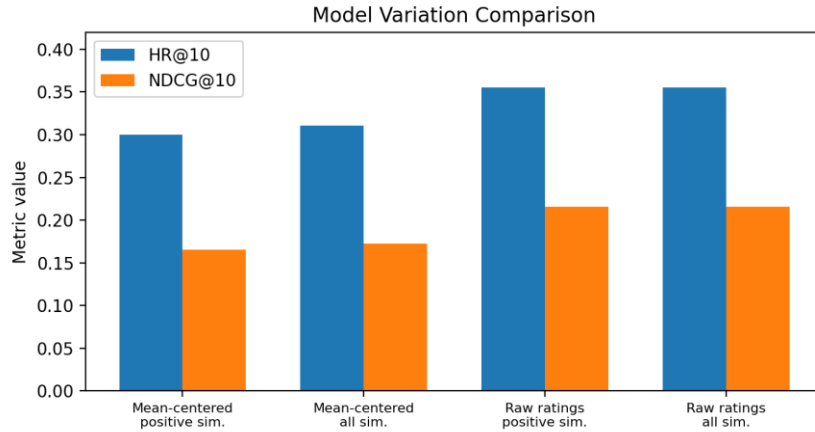


Figure 3. Comparison of model variations using HR@10 and NDCG@10.

#### 5.4. Filtering-threshold analysis

**Table 7** and **Figure 4** illustrate the accuracy-versus-coverage trade-off introduced by filtering. Increasing the threshold from 10/10 to 30/30 reduces the evaluated population from 609 to 443 users and the item set from 2,269 to 860 movies. At the same time, HR/Recall@10 increases from 0.275 to 0.325 and NDCG@10 increases from 0.137 to 0.176. The higher threshold therefore produces stronger ranking results, but only for a smaller and more active subset of users and movies.

The result demonstrates a quality-versus-coverage trade-off. The 30/30 configuration produces the best metric values, but it evaluates fewer users and movies. Therefore, a production recommender system would need to balance accuracy with coverage. For this research paper, the result is valuable because it shows that sparsity handling is not only a preprocessing detail but a direct factor affecting recommendation quality. These differences should still be treated as exploratory because no confidence intervals or multiple random seeds were used.

Table 7. Filtering threshold experiment

| Min user/item ratings | Filtered ratings | Users | Movies | HR@10    | Recall@10 | NDCG@10  |
|-----------------------|------------------|-------|--------|----------|-----------|----------|
| 10 / 10               | 81,109           | 609   | 2,269  | 0.274671 | 0.274671  | 0.137456 |
| 20 / 20               | 67,225           | 566   | 1,297  | 0.300353 | 0.300353  | 0.165254 |
| 30 / 30               | 54,143           | 443   | 860    | 0.325056 | 0.325056  | 0.175711 |

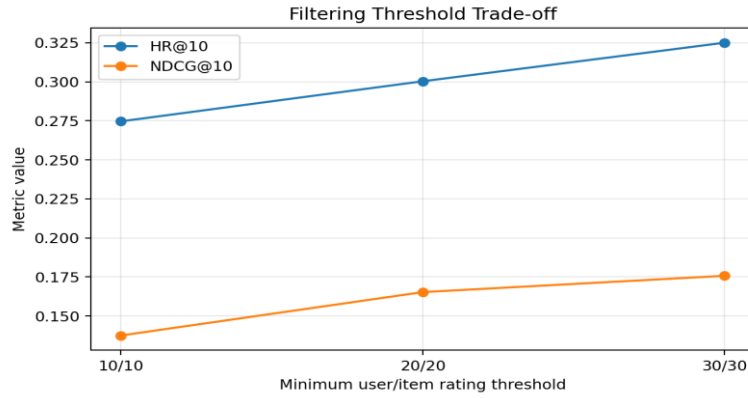


Figure 4. Filtering threshold trade-off between retained data and recommendation quality

### 5.5. Example recommendation list

**Table 8** presents the Top-10 recommendations generated for sample user 1. The list contains several widely rated films, including *The Godfather*, *The Shawshank Redemption*, *Casablanca*, and *Memento*. This indicates that the baseline captures meaningful collaborative signals, although the concentration of well-known titles also suggests a possible popularity bias that is not measured by the current accuracy-oriented metrics.

The predictions include ratings higher than those of the original rating scale of 0.5-5.0 since the similarity-weighted model was optimized with respect to ranking, and thus did not go through clipping after being predicted. This is not an issue for our Top-K ranking task evaluation, but may be one when rating accuracy is directly assessed. In any case, we make this point explicit in this paper since we focus solely on ranking evaluation.

Table 8. Top-10 recommendations for sample user 1

| Rank | Movie                                                                       | Genre                    | Predicted score |
|------|-----------------------------------------------------------------------------|--------------------------|-----------------|
| 1    | The Godfather (1972)                                                        | Crime   Drama            | 5.597079        |
| 2    | Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb (1964) | Comedy   War             | 5.507953        |
| 3    | The Shawshank Redemption (1994)                                             | Crime   Drama            | 5.466995        |
| 4    | The Godfather: Part II (1974)                                               | Crime   Drama            | 5.431162        |
| 5    | Casablanca (1942)                                                           | Drama   Romance          | 5.352412        |
| 6    | Harold and Maude (1971)                                                     | Comedy   Drama   Romance | 5.318593        |
| 7    | Brazil (1985)                                                               | Fantasy   Sci-Fi         | 5.304822        |
| 8    | The Bridge on the River Kwai (1957)                                         | Adventure   Drama   War  | 5.273957        |
| 9    | Eternal Sunshine of the Spotless Mind (2004)                                | Drama   Romance   Sci-Fi | 5.269131        |
| 10   | Memento (2000)                                                              | Mystery   Thriller       | 5.268974        |

### 5.6. Discussion

The findings indicate that recommendation quality is strongly affected by evaluation design and preprocessing. Positive-only evaluation ensures that the target item represents a movie the user liked. Sampled evaluation makes the ranking task computationally feasible, although its difficulty depends on the negative-sampling strategy. The random-baseline comparison shows that both collaborative-filtering configurations perform above chance, while the model-variation experiment indicates that raw ratings outperform mean-centered ratings under the full-user protocol.

Sparsity remains a central limitation of memory-based collaborative filtering because reliable user similarity requires sufficient overlap in rated items. Filtering improved ranking performance by retaining users and items

with stronger interaction support and denser overlap, even though it reduced the total number of ratings and limited coverage.

Finally, the experimental results support the central argument of this research: a clear pipeline helps identify effects of different design decisions on recommendation quality. The paper shows that even the classic CF algorithm can give insights when used in an appropriate way, evaluated against a random baseline, and compared under comparable conditions. The conclusions therefore remain specific to the MovieLens dataset and the offline experimental protocols used in this study.

## 6. Limitations and threats to validity

This study uses the MovieLens dataset, a widely used benchmark that may not fully represent production recommendation environments. User interactions in practice involve implicit feedback, time-based effects, exposure biases, multiple interactions, and evolving user interests. Hence, results obtained with MovieLens can be seen as an outcome of offline testing and not as direct indications of practical performance in production.

Another limitation is the fact that the system assesses ranking metrics but does not estimate novelty, diversity, fairness, or user satisfaction. These factors matter for real-life recommendation engines. For instance, recommending only popular items would lead to higher accuracy yet low diversity. As such, one must understand the presented metrics as the ranking quality metrics and not comprehensive measures of recommendation utility.

Finally, the system uses sampled negatives for evaluation purposes. Although this method saves computation time and is often employed in recommender-systems research, it depends on the chosen sampling technique. In our case, negatives are sampled uniformly from movies in the filtered item catalog that the user had not rated in the training set. A different random seed, a larger candidate set, or popularity-biased negatives may change the measured HR@10 and NDCG@10. In addition, the leave-one-out strategy provides one positive test item per user, which makes HR@10 and Recall@10 identical and does not capture full user preference diversity.

An additional problem with validity is the lack of statistical significance testing. While the difference results are helpful in analyzing the data, the paper does not give confidence intervals, significance tests, or several repeated experiments with different seeds. Accordingly, small differences, such as 0.300 compared with 0.311, should be interpreted cautiously. Larger differences, such as the improvement over the random baseline and the advantage of raw ratings over mean-centered ratings, provide stronger preliminary evidence but still require repeated experiments and formal statistical testing.

The algorithm used in the current experiment is a user-based collaborative filtering approach that is easily understandable and works well as a baseline, although it is probably not the most efficient one for very large datasets. Further work can test the implementation against other collaborative filtering techniques like an item-based approach, matrix factorization, implicit ALS, neural network-based approach, and hybrids of these models.

## 7. Conclusion

In this paper, a complete recommendation system pipeline based on user-based collaborative filtering was presented for the MovieLens dataset. This included the following steps: data loading, validation, preprocessing, sparsity analysis, filtering, positive only train test split, user-item matrix construction, computing raw and mean-centered cosine similarity, prediction of ratings, generation of Top-K recommendations, sampled ranking evaluation, comparison to a random baseline, and analysis via HR@10, Recall@10, and NDCG@10.

The primary significance of this work lies in its reproducibility and interpretability through an experimentation process instead of introducing a novel recommendation algorithm. The experiments carried out indicated that evaluation protocol, neighborhood size, rating normalizations, similarity filtering, negative sampling and sparsity thresholds all impact recommendation performance metrics. Among all comparable full-user setups, the setup employing raw ratings achieved the best performance with HR@10/Recall@10 = 0.355 and

NDCG@10 = 0.216, while the reference setup with mean-centered ratings scored HR@10/Recall@10 = 0.300 and NDCG@10 = 0.165.

Future directions should expand the comparison to item-based collaborative filtering, matrix factorization, implicit ALS, neural collaborative filtering, and popularity baselines. Some possible further enhancements would be full-protocol neighborhood tuning, temporal splitting, multiple random seeds, confidence intervals, paired statistical testing, cold-start experiments, score calibration/clipping, and measures of diversity/novelty.

### **Declaration of competing interest**

The authors declare that they have no known financial or non-financial competing interests in any material discussed in this paper.

### **Funding information**

No funding was received from any financial organization to conduct this research.

### **Author contribution**

The contribution to the paper is as follows: Nur Fulin: data collection, dataset preparation, preprocessing, implementation support, result organization, and manuscript drafting. Tarik Tinjak: study design, recommendation-system pipeline implementation, experimental evaluation, result analysis, visualization, and manuscript drafting. Emine Yaman: research supervision, conceptual guidance, methodology validation, critical review, academic feedback, and final manuscript revision. All authors reviewed and approved the final version of the manuscript.

### **References**

- [1] S. Rendle, W. Krichene, L. Zhang, and J. Anderson, “Neural collaborative filtering vs. matrix factorization revisited,” in *Proceedings of the 14th ACM Conference on Recommender Systems (RecSys '20)*, 2020, pp. 240–248, <https://doi.org/10.1145/3383313.3412488>.
- [2] V. W. Anelli, A. Bellogín, A. Ferrara, D. Malitesta, F. A. Merra, C. Pomo, F. M. Donini, and T. Di Noia, “Elliot: A comprehensive and rigorous framework for reproducible recommender systems evaluation,” in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*, 2021, pp. 2405–2414, <https://doi.org/10.1145/3404835.3463245>.
- [3] M. F. Dacrema, S. Boglio, P. Cremonesi, and D. Jannach, “A troubling analysis of reproducibility and progress in recommender systems research,” *ACM Transactions on Information Systems*, vol. 39, no. 2, Art. no. 20, pp. 1–49, Jan. 2021, <https://doi.org/10.1145/3434185>.
- [4] E. Zangerle and C. Bauer, “Evaluating recommender systems: Survey and framework,” *ACM Computing Surveys*, vol. 55, no. 8, Art. no. 170, pp. 1–38, Dec. 2022, <https://doi.org/10.1145/3556536>.
- [5] S. Rendle, W. Krichene, L. Zhang, and Y. Koren, “Revisiting the performance of iALS on item recommendation benchmarks,” in *Proceedings of the 16th ACM Conference on Recommender Systems (RecSys '22)*, 2022, pp. 427–435, <https://doi.org/10.1145/3523227.3548486>.
- [6] Y. Ji, A. Sun, J. Zhang, and C. Li, “A critical study on data leakage in recommender system offline evaluation,” *ACM Transactions on Information Systems*, vol. 41, no. 3, Art. no. 75, pp. 1–27, Feb. 2023, <https://doi.org/10.1145/3569930>.
- [7] Y.-C. Fan, Y. Ji, J. Zhang, and A. Sun, “Our model achieves excellent performance on MovieLens: What does it mean?,” *ACM Transactions on Information Systems*, vol. 42, no. 6, Art. no. 159, pp. 1–25, Oct. 2024, <https://doi.org/10.1145/3675163>.

- [8] C. Bauer, E. Zangerle, and A. Said, "Exploring the landscape of recommender systems evaluation: Practices and perspectives," *ACM Transactions on Recommender Systems*, vol. 2, no. 1, Art. no. 11, pp. 1–31, Mar. 2024, <https://doi.org/10.1145/3629170>.
- [9] R. Borges and K. Stefanidis, "Feature-blind fairness in collaborative filtering recommender systems," *Knowledge and Information Systems*, vol. 64, pp. 943–962, 2022, <https://doi.org/10.1007/s10115-022-01656-x>.
- [10] P. M. T. Do and T. T. S. Nguyen, "Semantic-enhanced neural collaborative filtering models in recommender systems," *Knowledge-Based Systems*, vol. 257, p. 109934, Dec. 2022, <https://doi.org/10.1016/j.knsys.2022.109934>.
- [11] N. Alharbe, M. A. Rakrouki, and A. Aljohani, "A collaborative filtering recommendation algorithm based on embedding representation," *Expert Systems with Applications*, vol. 215, p. 119380, Apr. 2023, <https://doi.org/10.1016/j.eswa.2022.119380>.
- [12] F. Horasan, A. H. Yurttakal, and S. Gündüz, "A novel model based collaborative filtering recommender system via truncated ULV decomposition," *Journal of King Saud University - Computer and Information Sciences*, vol. 35, no. 8, p. 101724, Sep. 2023, <https://doi.org/10.1016/j.jksuci.2023.101724>.
- [13] G. Behera and N. Nain, "Handling data sparsity via item metadata embedding into deep collaborative recommender system," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 10, pp. 9953–9963, 2022, <https://doi.org/10.1016/j.jksuci.2021.12.021>.
- [14] J. Bobadilla, F. Ortega, A. Gutiérrez, and Á. González-Prieto, "Deep variational models for collaborative filtering-based recommender systems," *Neural Computing and Applications*, vol. 35, pp. 7817–7831, 2023, <https://doi.org/10.1007/s00521-022-08088-2>.
- [15] Y. Afoudi, M. Lazaar, and M. Al Achhab, "Hybrid recommendation system combined content-based filtering and collaborative prediction using artificial neural network," *Simulation Modelling Practice and Theory*, vol. 113, p. 102375, 2021, <https://doi.org/10.1016/j.simpat.2021.102375>.
- [16] B. Alhijawi and Y. Kilani, "A collaborative filtering recommender system using genetic algorithm," *Information Processing & Management*, vol. 57, no. 6, p. 102310, Nov. 2020, <https://doi.org/10.1016/j.ipm.2020.102310>.
- [17] J. Bobadilla, S. Alonso, and A. Hernando, "Deep learning architecture for collaborative filtering recommender systems," *Applied Sciences*, vol. 10, no. 7, p. 2441, 2020, <https://doi.org/10.3390/app10072441>.
- [18] S. V. Aciar, R. Fabregat, T. Jové, and G. Aciar, "Enhancing recommender system with collaborative filtering and user experiences filtering," *Applied Sciences*, vol. 11, no. 24, p. 11890, 2021, <https://doi.org/10.3390/app112411890>.
- [19] J. Deng, J. Chen, S. Wang, J. Ye, and Y. Wang, "A novel fuzzy neural collaborative filtering for recommender systems," *Expert Systems with Applications*, vol. 258, p. 125153, Dec. 2024, <https://doi.org/10.1016/j.eswa.2024.125153>.
- [20] M. F. Aljunid, M. D. H. Manjaiah, M. K. Hooshmand, W. A. Ali, A. M. Shetty, and S. Q. Alzoubah, "A collaborative filtering recommender systems: Survey," *Neurocomputing*, vol. 617, p. 128718, 2025, <https://doi.org/10.1016/j.neucom.2024.128718>.
- [21] N. Nassar, A. Jafar, and Y. Rahhal, "Multi-criteria collaborative filtering recommender by fusing deep neural network and matrix factorization," *Journal of Big Data*, vol. 7, Art. no. 34, 2020, <https://doi.org/10.1186/s40537-020-00309-6>.

- [22] H. I. Abdalla, A. A. Amer, Y. A. Amer, L. Nguyen, and B. Al-Maqaleh, "Boosting the item-based collaborative filtering model with novel similarity measures," *International Journal of Computational Intelligence Systems*, vol. 16, Art. no. 123, Jul. 2023, <https://doi.org/10.1007/s44196-023-00299-2>.
- [23] E. Yalcin and A. Bilge, "A novel target item-based similarity function in privacy-preserving collaborative filtering," *The Journal of Supercomputing*, vol. 80, pp. 19433–19461, 2024, <https://doi.org/10.1007/s11227-024-06221-7>.
- [24] Y. Duan, Y. Tu, Y. Lu, and X. Wang, "Improving graph collaborative filtering with view explorer for social recommendation," *Journal of Intelligent Information Systems*, vol. 62, pp. 1703–1724, 2024, <https://doi.org/10.1007/s10844-024-00865-w>.
- [25] X. Huang, J. Wang, and J. Cui, "A personalized collaborative filtering recommendation system based on bi-graph embedding and causal reasoning," *Entropy*, vol. 26, no. 5, p. 371, 2024, <https://doi.org/10.3390/e26050371>.
- [26] J. Munson, B. Cummins, and D. Zosso, "An introduction to collaborative filtering through the lens of the Netflix Prize," *Knowledge and Information Systems*, vol. 67, pp. 3049–3098, 2025, <https://doi.org/10.1007/s10115-024-02315-z>.
- [27] O. A. S. Ibrahim, E. M. G. Younis, E. A. Mohamed, and W. N. Ismail, "Revisiting recommender systems: An investigative survey," *Neural Computing and Applications*, vol. 37, pp. 2145–2173, 2025, <https://doi.org/10.1007/s00521-024-10828-5>.
- [28] P. Mateos and A. Bellogín, "A systematic literature review of recent advances on context-aware recommender systems," *Artificial Intelligence Review*, vol. 58, Art. no. 20, 2025, <https://doi.org/10.1007/s10462-024-10939-4>.
- [29] M. Aldayel, A. Al-Nafjan, W. M. Al-Nuwaiser, G. Alrehaili, and G. Alyahya, "Collaborative filtering-based recommendation systems for touristic businesses, attractions, and destinations," *Electronics*, vol. 12, no. 19, p. 4047, 2023, <https://doi.org/10.3390/electronics12194047>.
- [30] D. Pérez-López, J. Dueñas-Lerín, F. Ortega, and Á. González-Prieto, "New trends in artificial intelligence for recommender systems and collaborative filtering," *Applied Sciences*, vol. 13, no. 15, p. 8845, 2023, <https://doi.org/10.3390/app13158845>.