STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **15**, February 2026, pp 1381–1397.
Published online in International Academic Press (www.IAPress.org)

Enhancing Recommender Systems through Active Learning Strategies with Matrix Factorization

Bachir Asri*, Iliass Igmoullan, Sara Qassimi

L2IS Laboratory, Faculty of Science and Technology, Cadi Ayyad University, Marrakesh, Morocco

Abstract With the rise of information overload and a multitude of options, the significance of recommender systems as indispensable tools remains paramount. These systems offer personalized suggestions, greatly improving user experiences in navigating the vast array of available options. This study explores the application of Matrix Factorization combined with active learning techniques to enhance the accuracy of recommender systems and tackle frequent challenges like sparse data and the cold start issue. The active learning strategies employed encompass both personalized approaches, like similarity to profile, highest predicted, and binary predicted, as well as non-personalized methods including random, popularity, Gini, error, popgini, and poperror. By applying these strategies to Matrix Factorization using the MovieLens and GoodBooks datasets, the study demonstrates significant improvements over conventional approaches, highlighting the critical role of active learning in enhancing recommender systems to better capture diverse user preferences.

Keywords Active Learning, Matrix factorization, Collaborative Filtering, Recommender Systems, Recommendations

DOI: 10.19139/soic-2310-5070-2772

1. Introduction

Recommender systems employing *collaborative filtering* (CF) [17] are commonly employed for suggesting products [16]. Yet, these systems struggle to perform at their best, constantly hindered by the ongoing issue of limited data availability [5]. This sparsity arises from limited user interactions with items [19], hindering the precision of traditional recommendation methods. This is not just a technical inconvenience—it directly limits the system's capacity to adapt to user needs, reducing personalization quality and overall engagement. This limitation highlights a critical gap in traditional CF methods, where most approaches still depend on passively collected interaction data without proactively seeking information that could reduce uncertainty.

This paper addresses this gap by promoting the incorporation of active learning strategies [13] as proactive steps to improve the accuracy of recommender systems. Our contribution lies in applying and evaluating active learning methods within CF-based recommendation, with a focus on strategically selecting items for user feedback in sparse environments. Our approach emphasizes deliberate selection of queries to users, surpassing mere reliance on historical interactions to improve system effectiveness.

The core strategy involves actively engaging users to gather feedback on items they haven't interacted with previously. This adaptive strategy seeks to enhance the learning process, particularly in areas of high uncertainty inherent to sparse datasets [1]. By focusing feedback collection on such uncertain regions, our method aims to maximize the information gained from each interaction.

Our approach employs active learning algorithms that selectively choose items for user feedback, aiming to capture meaningful insights and enhance the precision of the CF-based recommender system. The central

*Correspondence to: Bachir Asri (Email: b.asri.ced@uca.ac.ma). L2IS Laboratory, Faculty of Science and Technology, Cadi Ayyad University, Marrakesh, Morocco.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

idea revolves around actively involving users in providing feedback on previously unencountered items, thereby augmenting the recommendation system's efficacy.

The core strategy involves actively engaging users to gather feedback on items they haven't interacted with previously, aiming to optimize the learning process in areas of high uncertainty inherent to sparse datasets [1]. By deploying active learning algorithms that intelligently select items for user feedback, our methodology seeks to refine the accuracy of the CF-based recommender system. In doing so, we demonstrate that even with limited user interactions, a carefully targeted feedback strategy can yield measurable improvements in recommendation quality.

This study investigates the application of active learning techniques as a targeted approach to address data sparsity and improve recommendation quality, ultimately enhancing the user experience. The contributions of this work are: (1) demonstrating the integration of active learning with CF in a sparse data setting, and (2) experimentally showing its potential to improve recommendation accuracy through targeted user feedback.

The following sections are arranged as follows: section 2 provides an in-depth literature review and background; section 3 outlines the methodology, explaining the recommender systems based on active learning strategies; section 4 presents the experimental results, section 5 discussion, and section 6 concludes with insights and future directions.

2. Background and Literature Review

2.1. Collaborative Filtering

The concept of collaborative filtering (CF) emerged with Tapestry [6], a pioneering email filtering system. Initially designed for emails, CF became the cornerstone of recommender systems.

There are two main CF approaches:

- **Memory-based methods:** These methods examine the complete user-item interaction matrix to identify users (user-based) or items (item-based) which resemble the target user or item, as illustrated in Figure 1. This approach is simple to implement; however, it can become computationally intensive when handling larger datasets.
- **Model-based methods:** These methods build a model from the interaction matrix to predict future interactions. This approach often leads to better recommendations, especially with large datasets. Matrix factorization is a commonly adopted model-based technique that helps mitigate issues stemming from sparse interaction data and the cold start problem, where newly added users or items lack sufficient historical engagement.

The next section delve deeper into matrix factorization, recommender system challenges and active learning.

2.2. Matrix Factorization

Matrix factorization is an effective technique in recommender systems for identifying hidden patterns within user-item interactions. This process involves decomposing the user-item interaction matrix into two reduced-dimensional matrices, each corresponding to users and items, respectively (refer to Figure 2). These smaller matrices, often referred to as latent factor matrices, capture the fundamental elements influencing user preferences and item characteristics. This decomposition allows for a more compact and efficient representation of the interaction data, making it easier to analyze and generate recommendations.

This method gained widespread popularity following the Netflix Prize competition, where contestants aimed to improve the accuracy of Netflix's recommendation algorithm. Simon Funk's influential blog post in 2006 introduced a practical and effective approach to matrix factorization using stochastic gradient descent, which marked a significant advancement in collaborative filtering methodologies [20].

2.3. Sparsity

Sparsity presents a significant obstacle in the design of recommender systems. This occurs when the user-item interaction matrix contains many empty entries, see Figure 3. Imagine a vast library where users only check out a

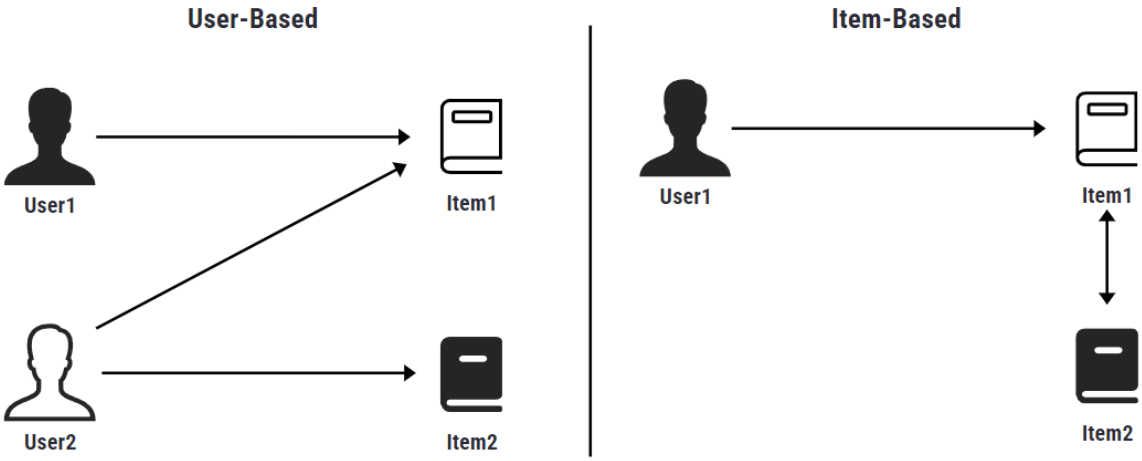

Figure 1. User-Based and Item-Based

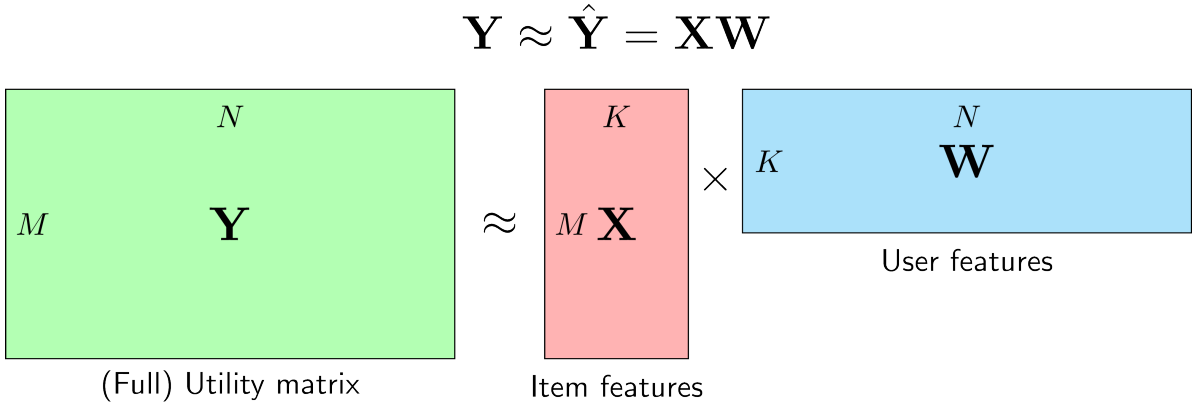

Figure 2. Matrix Factorization

tiny fraction of the books. Sparsity arises given that users usually engage with only a small portion of all available items.This lack of data makes it difficult for the system to accurately capture user preferences. It's like trying to understand someone's taste in music with only a few songs in their playlist.In essence, sparsity creates a blind spot for recommender systems, limiting the system's capacity to deliver genuinely personalized recommendations. Sparsity in recommender systems arises when the user-item interaction matrix contains numerous missing values or limited user-item interactions, typically due to users interacting with only a fraction of the total items. This incomplete representation of user preferences poses challenges in accurately capturing and understanding preferences across the entire item spectrum within the system.

2.4. Cold Start

Recommender systems may experience considerable difficulties when dealing with new users or items, a challenge commonly referred to as the cold start problem. This issue arises because these entities lack a history of interactions, leaving the system with insufficient data to understand and predict their preferences accurately, see Figure 4. As a result, the recommendations generated for these new users or items are often less precise and personalized.

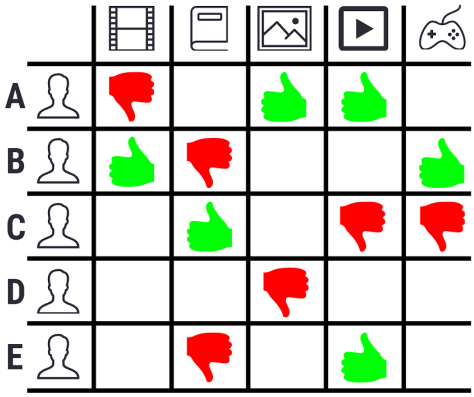

Figure 3. Sparsity

Traditional recommendation methods heavily depend on past behavior and interaction data to generate suggestions. In the absence of such data, the system struggles to provide meaningful recommendations, making it difficult to engage and retain new users or promote new items effectively.

To mitigate the cold start problem, augmenting user profiles with additional information, such as demographic details or social connections has been proposed. This supplementary data can help the system make more informed predictions about user preferences. However, incorporating such information raises privacy concerns and can lead to user resistance, as individuals may be reluctant to share sensitive personal details [15].

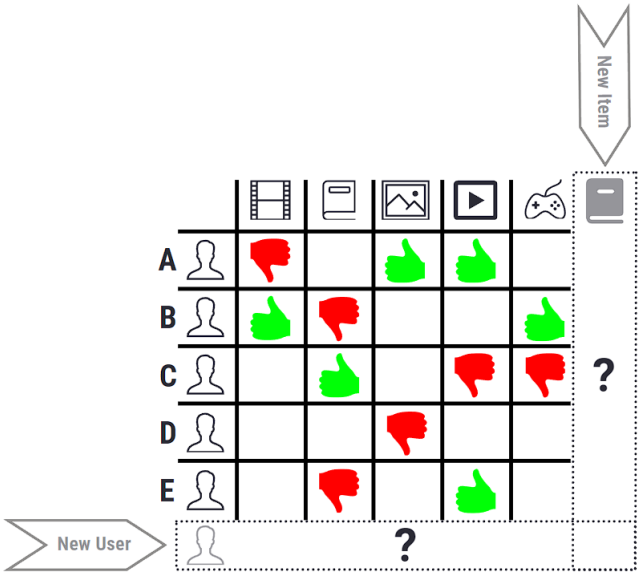

Figure 4. Cold Start

2.5. Active Learning

Traditional recommender systems primarily rely on passive learning, where a model is trained on a pre-existing dataset without actively seeking additional data, see Figure 5. Nevertheless, this method may face challenges due to sparse user-item interactions and the cold start problem, which complicates delivering tailored recommendations for new users or items. Active learning presents a promising approach to overcoming these challenges.

Active learning strategically acquires data points based on the system's current needs. In recommender systems, this translates to soliciting user feedback on specific items. This targeted approach offers several advantages:

- **Addressing the Cold Start Problem:** When a user first interacts with the system, active learning algorithms identify and suggest a targeted subset of items according to defined criteria. User feedback on these recommendations helps the system rapidly learn user preferences, overcoming the initial data scarcity characteristic of cold starts. This accelerates user adaptation and facilitates earlier delivery of personalized recommendations.

- **Mitigating Sparsity:** Sparse user-item interaction matrices hinder accurate recommendation generation. Active learning identifies these data gaps and prompts users for feedback on specific items. Through this iterative approach, sparsity is gradually reduced, fostering a more refined comprehension of item properties and user preferences. By actively exploring the item space, the system becomes less reliant on sparse data, ultimately improving recommendation quality.

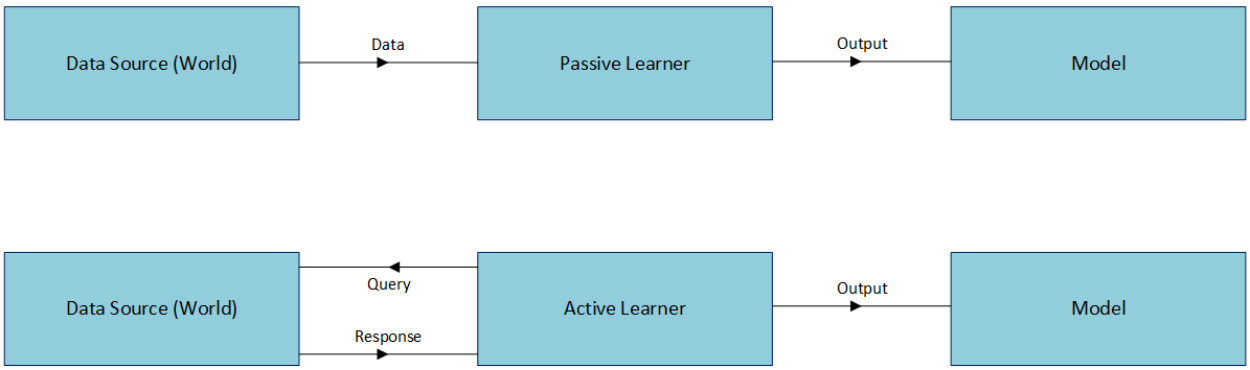

Figure 5. Active Learning vs Passive Learning

Active learning provides a valuable strategy for improving recommender systems by tackling data sparsity and the cold start issue. This results in better user adaptation and ultimately yields more precise and personalized recommendations.

2.6. Active Learning Strategies in Matrix Factorization Recommender Systems

Active learning has become an important technique in recommender systems to efficiently acquire user preferences and improve recommendation quality, especially in matrix factorization (MF) frameworks. Recent studies have explored various active learning strategies to enhance cold-start performance and model accuracy.

The paper by Geurts et al. (2020) evaluated several active learning methods within matrix factorization recommenders but focused exclusively on RMSE as the evaluation metric [2]. The study by De Pessemier et al. (2025) proposed a decision tree-based active learning strategy and compared it primarily to a most popular baseline, examining only this single approach [3].

In contrast, this work systematically tests nine different active learning strategies within an MF framework. The evaluation is conducted on three benchmark datasets: MovieLens 100K, MovieLens 1M, and Goodbooks 10K. Performance is assessed using multiple metrics including precision, recall, F1-score, and mean squared error (MSE), providing a comprehensive understanding of how different active learning approaches perform across various recommendation quality dimensions.

3. Methodology

This section outlines the model we propose to tackle the cold user problem in RS through the use of active learning strategies.

3.1. Notation

An RS typically involves N users denoted by the set $U = u1, u2, ..., uN$. Similarly, there are M items represented by the set $I = i1, i2, ..., iM$. User u rates item i with a numerical value rui. Ratings usually fall within a scale of 1 to 5 for explicit feedback (e.g., movie ratings). The user-item interaction matrix, $R \in \mathbb{R}^{N \times M}$, captures all interactions. Here, r_{ui} represents the explicit rating user u gives item i. The range of r_{ui} is determined by the rating scale $r_{ui} \in \{1, 2, 3, 4, 5\}$.

Our model focuses on explicit feedback, allowing for a wider range of values to capture nuanced user preferences compared to binary implicit feedback (0 or 1).

3.2. Matrix Factorization

Unlike traditional approaches using Singular Value Decomposition (SVD) [7], our model utilizes embeddings to represent the latent factors of users and items. Specifically, the user latent factor matrix $P \in \mathbb{R}^{F \times N}$ and the item latent factor matrix $Q \in \mathbb{R}^{F \times M}$ are not obtained directly from singular value decomposition (SVD).

Embeddings enable the dynamic learning of user and item representations within a common latent space. Each user and item has learned embedding vectors, facilitating the model's ability to identify complex interdependencies in the data. The model's prediction for the user-item interaction, denoted as $\hat{a}_{ui}$, is a function of these embeddings.

This embedding-based approach replaces conventional SVD-based matrix factorization. It offers a more flexible framework for handling latent factors in collaborative filtering-based RS. Embeddings enable the model to adjust to non-linear relationships and uncover complex patterns, thereby enhancing the recommendation process.

3.3. Active Learning Strategies

The user vector u_k for cold user k is empty at first due to the absence of any item interactions, posing a challenge for personalized recommendations. To overcome this, it's crucial to gather preferences for a specific set of items, aiming to determine the true values $a_{k1}, a_{k2}, \ldots, a_{kj}$, where j represents the number of candidate items presented to gather user preferences. Active learning techniques play a crucial role in reducing data sparsity and addressing the cold start challenge by proactively soliciting user feedback. Each a_{kj} reflects the preference or rating assigned by user k to the j-th item, providing insight into their opinion or preference for each item and maximizing recommendation quality.

3.3.1. ***Random Strategy*** The Rand strategy randomly assigns scores between 0 and 1 to items for an active user. The method proceeds by selecting the top nq items to be presented for user feedback. This random approach serves as a benchmark to compare more complex recommendation strategies that consider user preferences or item similarities. The goal is to show that more advanced strategies can outperform pure randomness.

3.3.2. ***Popularity Strategy*** The Pop strategy picks the most popular items (based on past ratings) for all active users. It simply chooses the nq items with the highest total rating counts. This approach isn't personalized because everyone gets the same recommendations for the same item pool, regardless of their individual tastes.

3.3.3. ***Similarity-To-Profile Strategy*** The Similarity-To-Profile (S2P) method [8] customizes recommendations by suggesting items that resemble those the user has previously rated. It assumes that users are more likely to engage with items sharing similarities with their past preferences. The S2P formula:

$$f_S(u, i, G = D_{\text{before}}) = \sum_{j \in I_u} \mathbb{1}(i, j)$$

Here, $I_u = I^{\text{before}}_{D_u}$ denotes the items rated by user u. The indicator function $\mathbb{1}(i,j)$ returns 1 if item i is among the $NS2P$ nearest neighbors of item j, determined by cosine similarity of their ratings. Following the scoring process, the top nq items with the highest scores are selected.

3.3.4. ***Highest Predicted Strategy*** The HP approach, described in [9], operates by predicting ratings for each user-item pair using a Matrix Factorization recommender. For active users in set U_{Active}, and their candidate items C_u, the recommender M predicts ratings $\hat{r}_{u,i}$. The score assigned to each item is its predicted rating. This personalized method chooses the top nq items with the highest predicted ratings, focusing on prioritizing those most likely to appeal to the user.

3.3.5. ***Binary Predicted Strategy*** The Binary-Predicted (BP) approach, as described in [9], involves several steps. Initially, the matrix D_{before} is transformed into a binary matrix B, where each element $b_{u,i}$ is set to 1 if a rating exists in D_{before}, and 0 otherwise. Then, an implicit MF model M is built based on B. For each active user u in U_{Active}, and each item in their candidate set C_u, the model predicts a score $\hat{r}_{u,i}$. This prediction serves as the item's score, denoted by $f_S(u,i,G=M) = \hat{r}_{u,i}$. Finally, the top nq items with the highest scores are chosen. This approach personalizes recommendations by prioritizing items likely to be familiar to the user.

3.3.6. ***Gini Strategy*** Introduced in [2], Gini Strategy ranks items according to the Gini impurity measure, making it an uncertainty-based active learning (AL) method. This approach calculates the Gini impurity for each item by analyzing the distribution of user ratings, with a focus on items that generate varied opinions. Items with higher Gini impurity scores represent more balanced ratings, signifying uncertainty in user preferences. The Gini impurity of item i is given by:

$$\text{Gini}(i) = 1 - \sum_{j \in \{1,2,3,4,5\}} p(j|i)^2$$

The term $p(j|i)$ represents the frequency of each rating from 1 to 5 for item i. The item ranking is performed based on Gini impurity in descending order, where elevated impurity values imply a balanced distribution of rating levels, suggesting diverse user perceptions.

This strategy aims to present items that effectively split user opinions, viewing such items as valuable for gathering information. By ranking items according to their Gini impurity, the strategy highlights those likely to provide valuable insights, particularly from cold users, who offer new perspectives on items that generate diverse reactions.

This strategy is particularly effective for identifying ambiguous items where user opinions are split. Such items are ideal targets in active learning because they offer high informational gain. In the context of matrix factorization, refining predictions for these uncertain items can improve the overall latent space structure and model generalization.

3.3.7. ***Error Strategy*** As presented in [2], Error Strategy constitutes a constant, non-personalized method that leverages uncertainty in item ranking. Item i's error score is determined using the expression below:

$$\text{Error}(i) = 1 - \max_{j \in \{1,2,3,4,5\}} p(j \mid i)$$

This computed value represents subtracting the maximum relative frequency from one among all rating levels (1 to 5) assigned to item i, capturing the degree of uncertainty in user evaluations.

This score serves as a simple yet effective measure of entropy or confusion in item ratings. By prioritizing items with low agreement, the Error strategy focuses user feedback on areas where additional data can most improve model certainty.

3.3.8. ***PopError Strategy*** PopError, as introduced in [2], offers a novel approach to tackling the cold-user challenge by combining key aspects of both the Popularity and Error strategies. It combines these strategies using a linear combination, assigning weights to strike a balance between popularity and error considerations. To address

the challenges linked to the popularity strategy, PopError strategy applies a base-10 logarithmic transformation to adjust its scoring mechanism.

3.3.9. ***PopGini Strategy*** In [2], an active learning (AL) strategy combines popularity and Gini scores using a linear blend, creating a static heuristic. To reduce the impact of extreme values in the popularity metric, a logarithmic transformation is employed, promoting a more balanced item representation—particularly useful for engaging users with limited interaction history. The respective weights of popularity and diversity are tuned on training data, enabling the system to recommend both widely favored items and those that elicit a range of user opinions.

3.3.10. ***Rationale Behind Uncertainty-Based Strategies*** Uncertainty-based strategies in active learning aim to prioritize items for which the recommender system is least confident in its predictions. In the context of recommender systems, this uncertainty can be interpreted through the diversity and inconsistency of user feedback. The Gini and Error strategies are grounded in this principle.

Gini Impurity quantifies rating disagreement. A high Gini score implies that user ratings for an item are well distributed across multiple values (e.g., 1 to 5), indicating conflicting user opinions and hence, high uncertainty. Items with high Gini impurity are likely to provide informative feedback from new users because their preferences can help resolve these uncertainties.

Error Strategy estimates uncertainty by calculating how much the ratings for an item diverge from consensus. It subtracts the maximum observed rating probability from 1, capturing how evenly the ratings are spread. A higher score indicates less consensus and more uncertainty, making the item more valuable for active feedback.

Hybrid strategies like PopGini and PopError combine popularity and uncertainty through a weighted sum. This reflects a trade-off between exploring informative but less-rated items and exploiting known popular ones. While the weights were fixed in this study, they can be optimized or adapted dynamically in future work.

4. Experimentation

4.1. Datasets Description

The paper utilizes three datasets: MovieLens 100k, MovieLens 1M [22], and Goodbooks-10k [21]. The MovieLens datasets, sourced from the MovieLens platform, contain movie ratings provided by users, including information about users, movies, and their ratings. The Goodbooks-10k dataset includes book ratings, providing a different context for analyzing user-item interactions. Refer to Table 1 for a detailed comparison.

For research purposes, the dataset D^{before} is used by various active learning algorithms to gather feedback, while the dataset D^{after} represents the data after incorporating the feedback. Five-fold cross-validation was conducted on D^{after} to train the matrix factorization model, with the evaluation metric averaged across the folds to provide a more reliable estimate of performance.

In this simulated environment, we selected different proportions of active users for each dataset to reflect practical constraints and dataset characteristics. For MovieLens datasets (ML-100K and ML-1M), 25% of users were designated as active, aligning with common practice in experimental AL simulations.

In contrast, the GoodBooks-10K dataset contains over 53,000 users, most of whom have sparse interaction histories. To prevent excessive computational load and to simulate a realistic feedback scenario, we selected approximately 4% (2,000 users) as active users. This smaller sample ensures that the added feedback remains meaningful without overwhelming the dataset or inflating precision artificially. It's important to note that these ratings are random (from 1 to 5) and don't reflect real user preferences.

4.2. Evaluation Metrics

- **MSE** (Mean Squared Error) evaluates the average squared difference between observed values and their predicted counterparts. By penalizing larger errors more than smaller ones, MSE provides a comprehensive

Table 1. Comparison of Datasets

Aspect	MovieLens 100K	MovieLens 1M	Goodbooks-10k
Number of Users	943	6,040	53,424
Number of Active Users	235	1,510	2000
Number of Items	1,682 (Movies)	approx 3,900 (Movies)	10,000 (Books)
Number of Ratings	100,000	1,000,209	981,756
Rating Scale	1-5 stars	1-5 stars	1-5 stars

measure of prediction accuracy. Formally, it is expressed as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Where:

y_i : Real values
$\hat{y}_i$: Predicted values
n : Number of observations

- **Precision** quantifies the proportion of correctly predicted positive cases out of all positive predictions made by the model. A higher precision score indicates fewer false positive errors, reflecting the reliability of positive predictions. It is calculated as:

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **Recall** assesses the model's ability to detect all actual positive instances. It is defined as the ratio of true positive predictions to the total number of true positives and false negatives, thus reflecting completeness in identifying positives:

$$\text{Recall} = \frac{TP}{TP + FN}$$

Where:

TP : True Positives (Correct positive predictions)
FP : False Positives (Incorrect positive predictions)
FN : False Negatives (Incorrect negative predictions)

- **F1 Score** serves as a harmonic mean of precision and recall, providing a single performance metric that balances the trade-off between false positives and false negatives. This is especially valuable when dealing with uneven class distributions:

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

4.3. Training Process

We trained an initial Matrix Factorization (MF) model using an 80-20 split of the dataset, designating D_{train} for training and D_{test} for testing. After collecting feedback from active users, we formed a new dataset, $D_{\text{after}} = D_{\text{train}} +$ acquired feedback, which combined this feedback with D_{train}. Subsequently, we utilized 5-fold cross-validation to assess the performance of the matrix factorization model trained on the newly collected

feedback, as depicted in Figure 6.

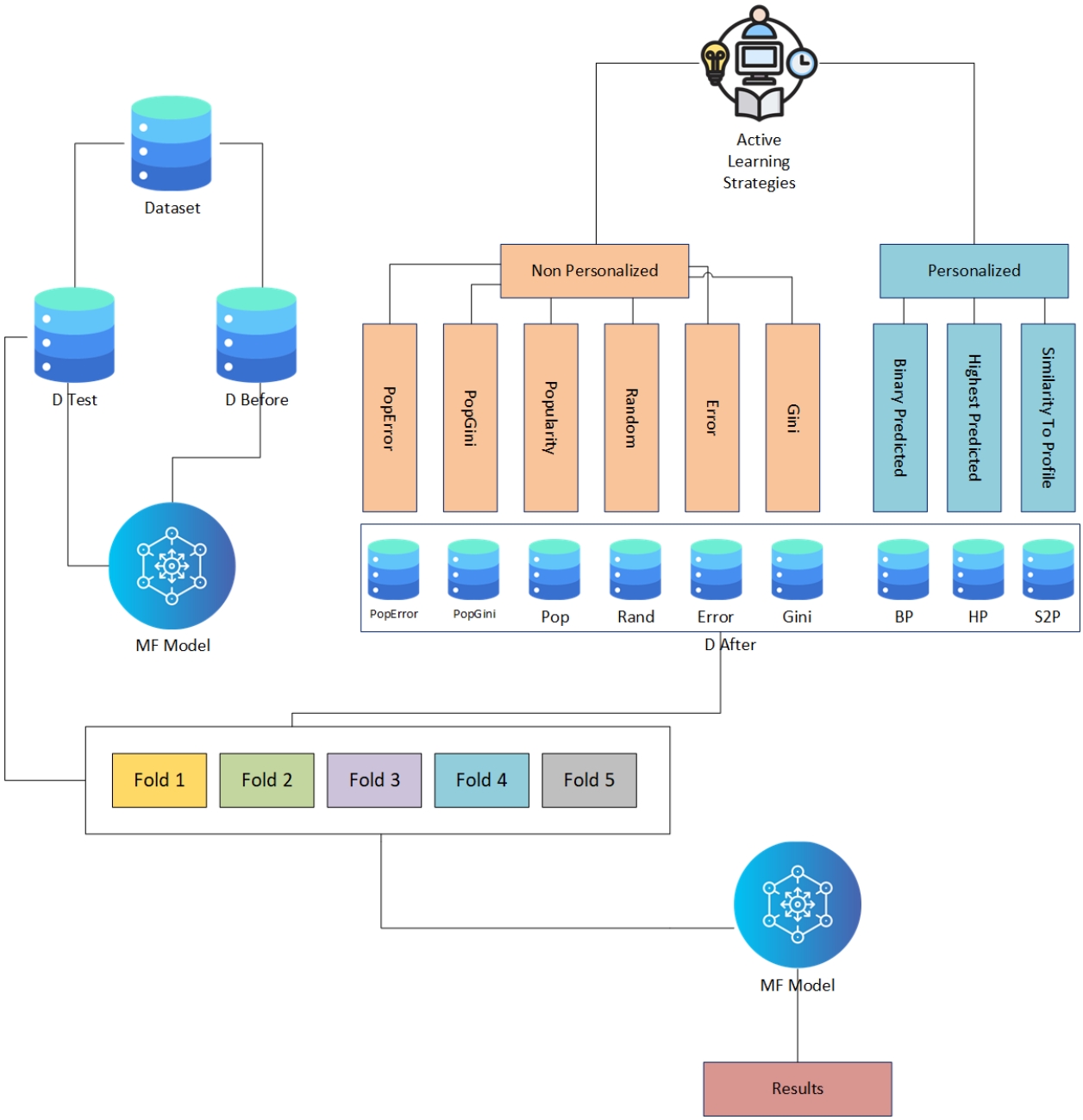

Figure 6. Methodology

Before:

- Create a recommender model using D_{before}.

AL Interaction:

- Select a strategy from the available set of strategies.
- For each active user $u \in U_{\text{Active}}$:
  - Gather the ratings for the displayed items E_u.
  - Update the dataset with the obtained ratings: $D_{\text{after}} = D_{\text{before}} \cup \bigcup_{u \in U_{\text{Active}}} E_u$.

After:

- Construct a new recommendation model using K_{after}.
- Evaluate the performance of the newly developed model.

4.4. Results

4.4.1. ML-1M The results presented in Tables 2, 3, 4, and 5 indicate that the Binary Predicted (BP) strategy consistently outperformed other strategies across various item counts. For 5 items, BP achieved the lowest Mean Squared Error (MSE) of 0.8329, with a precision of 0.8545, recall of 0.3992, and an F1 score of 0.5441. As the number of recommended items increased to 10, the MSE slightly rose to 0.8331, while precision remained strong at 0.8541, and recall stayed the same at 0.3992, resulting in an F1 score of 0.5440. With 20 items, the MSE remained nearly unchanged at 0.8329; precision showed a slight improvement, reaching 0.8542, with recall rising to 0.3995, yielding an F1 score of 0.5443. The Error and Gini strategies closely followed BP with competitive results, particularly excelling in precision. This suggests that, although the Mean Squared Error (MSE) is not optimal, better precision can be achieved, as demonstrated by the Error strategy for 10 and 20 items and the Gini strategy for 5 and 10 items.

4.4.2. ML-100K As illustrated in Tables 2, 3, 4, and 5 , the BP strategy continued to be the best performer. For 5 items, BP recorded an MSE of 0.9186, a precision of 0.8558, a recall of 0.2749, and an F1 score of 0.4138. When increasing the recommendations to 10 items, the MSE rose slightly to 0.9183, with precision dropping to 0.8554 and recall decreasing to 0.2745, resulting in an F1 score of 0.4143. With 20 items, the MSE further decreased to 0.9182, precision improved to 0.8569, accompanied by a slight enhancement in recall, which reached 0.2747, yielding an F1 score of 0.4140. The Error, Gini, PopGini, and Popularity strategies performed particularly well, improving precision compared to BP.

4.4.3. GoodBooks-10K The BP strategy continued to excel across different item counts, as evidenced in Tables 2, 3, 4, and 5 . For 5 items, BP achieved the best MSE of 0.7490, a precision of 0.8918, a recall of 0.4102, and an F1 score of 0.5596. For 10 items, the MSE was 0.7491, with precision slightly improving to 0.8922, while recall slightly decreased to 0.4099, resulting in an F1 score of 0.5592. With 20 items, BP maintained solid performance with an MSE of 0.7493, precision of 0.8920, and recall of 0.4100, yielding an F1 score of 0.5593. However, the PopError and Gini strategies demonstrated competitive performance, particularly in precision, where they outperformed BP. This highlights the strengths of alternative strategies, especially in precision-based tasks, demonstrating that while BP is balanced, it is not always the top choice when precision is prioritized in larger item sets within the GoodBooks-10K dataset.

Table 2. MSE values for ML-1M, ML100K, and GoodBooks-10K

	ML1M			ML100K			GoodBooks-10K		
	5 items	**10 items**	**20 items**	**5 items**	**10 items**	**20 items**	**5 items**	**10 items**	**20 items**
BP	**0.8329**	**0.8331**	**0.8329**	**0.9186**	**0.9183**	**0.9182**	**0.7490**	**0.7491**	**0.7493**
Err	0.8375	0.8417	0.8503	0.9394	0.9589	0.9915	0.7559	0.7619	0.7743
PopErr	0.8383	0.8419	0.8512	0.9234	0.9279	0.9362	0.7556	0.7620	0.7740
Gini	0.8375	0.8417	0.8497	0.9244	0.9283	0.9390	0.7558	0.7624	0.7743
PopGini	0.8366	0.8396	0.8470	0.9228	0.9279	0.9342	0.7558	0.7613	0.7745
HP	0.8438	0.8530	0.8697	0.9309	0.9421	0.9629	0.7571	0.7649	0.7777
Pop	0.8369	0.8414	0.8503	0.9246	0.9302	0.9417	0.7505	0.7506	0.7506
Rand	0.8393	0.8456	0.8581	0.9265	0.9338	0.9464	0.7619	0.7761	0.8017
S2P	0.8398	0.8459	0.8582	0.9255	0.9340	0.9459	0.7624	0.7742	0.7941

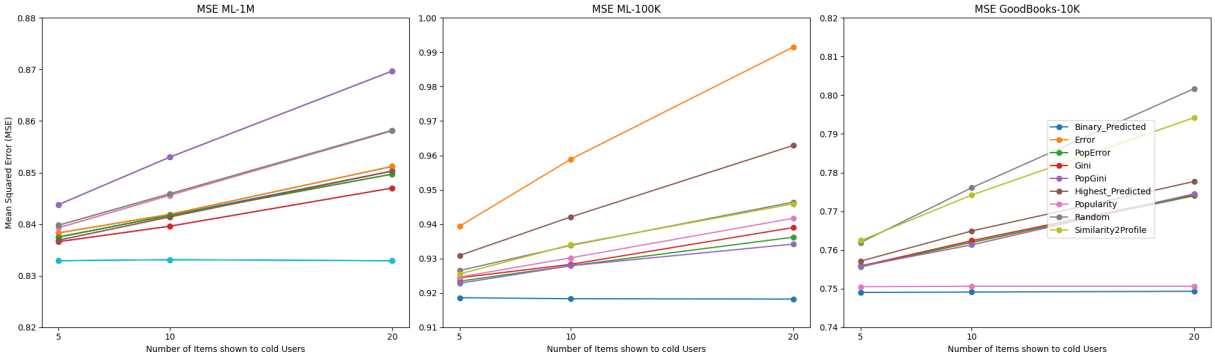

Figure 7. MSE Results

Table 3. Precision values for ML-1M, ML100K, and GoodBooks-10K

	ML1M			ML100K			GoodBooks-10K		
	5 items	**10 items**	**20 items**	**5 items**	**10 items**	**20 items**	**5 items**	**10 items**	**20 items**
BP	**0.8545**	0.8541	0.8542	0.8558	0.8554	0.8569	0.8918	0.8922	0.8920
Err	0.8540	**0.8547**	**0.8547**	0.8564	0.8581	**0.8583**	0.8926	0.8937	0.8960
PopErr	0.8535	0.8522	0.8495	0.8557	0.8548	0.8528	0.8927	0.8943	**0.8962**
Gini	**0.8545**	**0.8547**	0.8545	**0.8571**	0.8566	0.8574	**0.8931**	**0.8947**	**0.8962**
PopGini	0.8541	0.8542	0.8536	**0.8571**	0.8561	0.8581	0.8930	0.8945	0.8960
HP	0.8470	0.8449	0.8402	0.8455	0.8415	0.8388	0.8929	0.8941	0.8953
Pop	0.8543	0.8544	0.8543	0.8558	**0.8585**	0.8581	0.8921	0.8922	0.8919
Rand	0.8531	0.8518	0.8502	0.8564	0.8547	0.8545	0.8900	0.8873	0.8853
S2P	0.8527	0.8513	0.8491	0.8550	0.8537	0.8536	0.8903	0.8910	0.8941

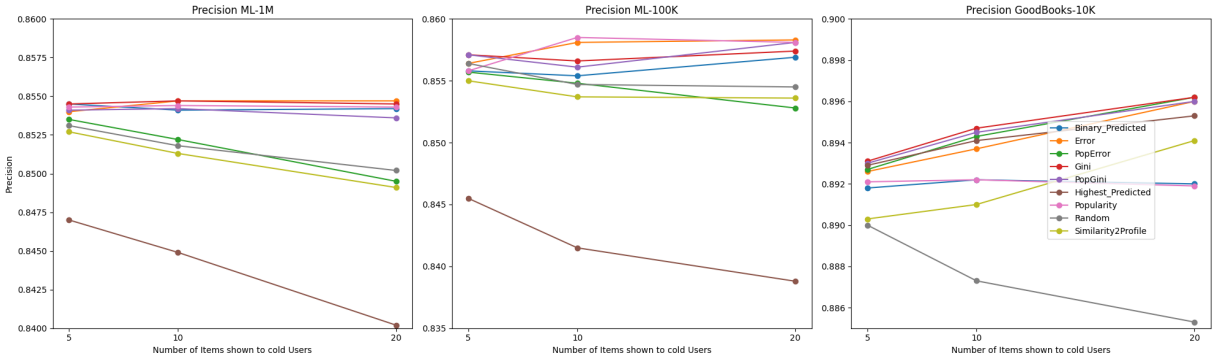

Figure 8. Precision Results

Table 4. Recall values for ML-1M, ML100K, and GoodBooks-10K

	ML1M			**ML100K**			**GoodBooks-10K**		
	5 items	**10 items**	**20 items**	**5 items**	**10 items**	**20 items**	**5 items**	**10 items**	**20 items**
BP	**0.3992**	**0.3992**	**0.3995**	**0.2749**	**0.2745**	**0.2747**	**0.4102**	**0.4099**	**0.4100**
Err	0.3974	0.3954	0.3917	0.2630	0.2537	0.2372	0.4030	0.3969	0.3838
PopErr	0.3941	0.3902	0.3799	0.2688	0.2648	0.2602	0.4026	0.3965	0.3843
Gini	0.3973	0.3955	0.3916	0.2711	0.2684	0.2624	0.4037	0.3955	0.3850
PopGini	0.3968	0.3940	0.3879	0.2713	0.2687	0.2628	0.4027	0.3965	0.3844
HP	0.3938	0.3878	0.3741	0.2638	0.2518	0.2286	0.4024	0.3950	0.3813
Pop	0.3978	0.3954	0.3915	0.2708	0.2685	0.2607	0.4085	0.4084	0.4087
Rand	0.3960	0.3923	0.3853	0.2698	0.2651	0.2579	0.3977	0.3834	0.3587
S2P	0.3953	0.3914	0.3839	0.2707	0.2637	0.2572	0.3969	0.3836	0.3593

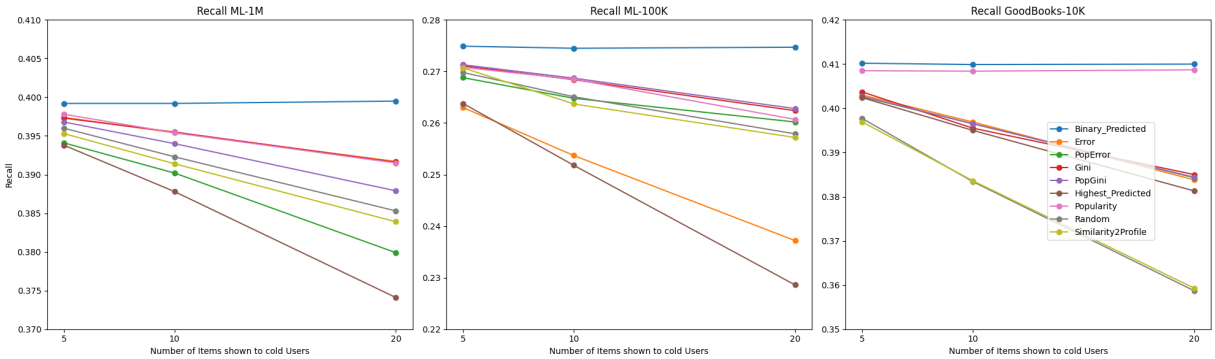

Figure 9. Recall Results

Table 5. F1 Score values for ML-1M, ML100K, and GoodBooks-10K

	ML1M			ML100K			GoodBooks-10K		
	5 items	**10 items**	**20 items**	**5 items**	**10 items**	**20 items**	**5 items**	**10 items**	**20 items**
BP	**0.5441**	**0.5440**	**0.5443**	**0.4138**	**0.4143**	**0.4140**	**0.5596**	**0.5592**	**0.5593**
Err	0.5423	0.5406	0.5372	0.4005	0.3894	0.3695	0.5524	0.5465	0.5333
PopErr	0.5391	0.5353	0.5249	0.4074	0.4023	0.3969	0.5522	0.5462	0.5340
Gini	0.5423	0.5407	0.5370	0.4096	0.4069	0.3994	0.5533	0.5453	0.5348
PopGini	0.5418	0.5392	0.5334	0.4101	0.4072	0.4001	0.5523	0.5462	0.5342
HP	0.5375	0.5316	0.5176	0.4002	0.3858	0.3569	0.5519	0.5447	0.5309
Pop	0.5427	0.5405	0.5368	0.4093	0.4068	0.3978	0.5580	0.5578	0.5581
Rand	0.5409	0.5371	0.5302	0.4083	0.4027	0.3945	0.5471	0.5324	0.5070
S2P	0.5401	0.5362	0.5287	0.4091	0.4006	0.3931	0.5463	0.5333	0.5090

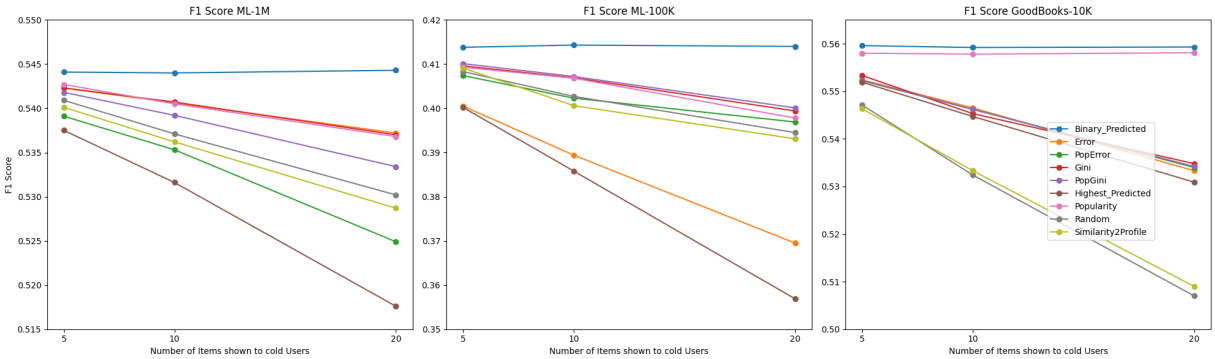

Figure 10. F1 Score Results

4.5. Computational Complexity of Strategies

Active learning strategies differ in their computational demands, which is critical for real-world deployment. Below, we summarize key cost factors:

- Random/Popularity: O(1) per item. These strategies use simple heuristics and are extremely fast.
- S2P (Similarity-to-Profile): O(N_items * K), where K is the number of nearest neighbors. This involves pairwise similarity computations, increasing latency.
- Gini/Error: O(N_items). These strategies require scanning rating distributions for each item.
- BP/HP: O(1) per prediction after model training, assuming precomputed embeddings.

While all strategies were computationally feasible on the tested datasets, scaling them to millions of users or items may require additional optimization or approximation techniques.

5. Discussion

The analysis of diverse evaluation metrics, including Mean Squared Error (MSE), precision, recall, and the F1 score, yields comprehensive insights into the effectiveness of the active learning strategies when applied to the MovieLens 100K, MovieLens 1M, and GoodBooks-10K datasets.

In the ml100k dataset, the Binary Predicted (BP) strategy consistently achieves the lowest MSE for all item counts (5, 10, and 20), demonstrating its robustness in predictive accuracy. However, in terms of precision, other strategies outperform BP, showcasing their ability to recommend highly relevant items. For 10 and 20 items, the Error and Popularity strategies perform better than BP in precision, indicating that they are more effective in

maintaining relevance as the recommendation set expands. For 5 items, the Gini and PopGini strategies showcase better precision than BP. Despite these results, BP maintains a well-balanced trade-off among precision, recall, and F1 score, making it a reliable strategy for larger recommendation sets where capturing user preferences comprehensively is essential.

In the ml1m dataset, BP secures the lowest MSE for all item counts, excelling in recall and F1 scores. However, for precision, the Error and Gini strategies outperform BP for 5, 10, and 20 items. This shows that while BP offers a balanced approach across various metrics, the Error and Gini strategies are particularly strong in maintaining high precision when recommending items.

For the GoodBooks-10K dataset, while BP achieves the lowest MSE across all item counts (5, 10, and 20), it is outperformed in precision by other strategies. The Gini strategy achieves the highest precision across all item sets. Additionally, PopGini and PopError demonstrate precision results that are close to those of Gini, indicating that these strategies effectively capture user preferences when prioritizing relevant recommendations. Nonetheless, BP remains competitive in other evaluation metrics, particularly F1 score and recall, making it a solid choice when considering a balanced approach to recommendation accuracy.

In the GoodBooks-10K dataset, the Popularity strategy occasionally outperformed Binary Predicted (BP) in terms of precision. This may be due to the domain-specific characteristics of book recommendation, where certain titles enjoy universal appeal and high visibility. In such cases, recommending popular books aligns well with user preferences, especially when user histories are limited.

Based on these findings in the realm of recommendation systems, BP, Error, PopError, Gini, PopGini, and Popularity strategies exhibit significant potential across diverse datasets. Non-personalized strategies effectively address the cold start problem, making them suitable for scenarios with limited user data. The effectiveness of these methods depends on the quantity of items presented to users as well as the specific properties of the dataset. The strong results of these strategies across multiple metrics, such as MSE, precision, recall, and F1 score, suggests they are versatile and effective in different contexts, making them valuable approaches for developing robust recommendation systems. Overall, these results emphasize the critical role of tailoring recommendation strategies to the unique attributes of each dataset and the specific context in which they are applied to achieve optimal outcomes.

An important dimension in evaluating active learning strategies is the trade-off between performance gains and user effort. Our experiments suggest that BP and Error strategies achieve robust F1 scores with as few as 5–10 queried items. However, strategies like Gini and PopGini required more queries to reach similar effectiveness. This underscores the need for cost-benefit-aware query planning in real deployments.

However, implementing active learning in recommender systems presents challenges, such as maintaining user engagement and managing feedback fatigue. Continuous querying for feedback can lead to user disengagement if not balanced carefully. Therefore, it is essential to develop strategies that minimize user burden while maximizing the quality of recommendations. Striking this balance is crucial for sustaining long-term user satisfaction and overall system effectiveness.

6. Conclusion and Future Works

This research explored the incorporation of multiple active learning (AL) strategies into matrix factorization techniques to enhance the effectiveness of recommender systems, utilizing the MovieLens ML-1M, ML-100K, and GoodBooks-10K datasets. Following a thorough analysis, we identified the effectiveness of different AL strategies across dynamic scenarios, demonstrating their suitability for diverse item presentation contexts.

In conclusion, the results highlight the effectiveness of incorporating active learning strategies to substantially enhance the performance of recommender systems. These strategies effectively address dataset sparsity and enhance recommendation quality, particularly for cold users. While the Binary Predicted (BP) strategy showed some effectiveness across various datasets, other approaches like Popularity, Gini, PopGini, Error, and PopError also demonstrated strengths in specific contexts. Overall, leveraging active learning strategies facilitates a more effective and responsive recommendation process, ultimately enhancing user satisfaction and engagement.

Looking ahead, there are significant opportunities to build upon this work. Future research will include ablation studies to better understand the impact of each component within hybrid strategies such as PopGini and PopError. Visual interpretability tools like t-SNE plots of the latent factor space are planned to illustrate how active learning shapes embedding structures. We also intend to extend evaluations to multi-cycle longitudinal active learning simulations to capture temporal dynamics and assess robustness as user preferences evolve. Incorporating implicit feedback signals such as clicks and dwell time alongside explicit ratings will be an important direction, requiring an extension of the matrix factorization framework to hybrid models. We plan to explore matrix factorization-aware active learning strategies, including latent-factor variance sampling and adaptive ensembles combining methods like BP and PopGini, to innovate within this domain. Further enhancements include improving feedback realism using synthetic user behaviors or natural datasets from platforms like Yelp or Netflix, broadening comparisons to modern recommenders such as Neural Collaborative Filtering and transformer-based models, and integrating multi-objective metrics like diversity, novelty, and coverage for a more comprehensive evaluation. Cold-start scenarios will be enriched through auxiliary user information to support hybrid models, and practical deployment considerations—including minimizing user fatigue, safeguarding privacy, and implementing A/B testing—will be key to ensuring real-world applicability and adoption.

These directions aim to enhance both the theoretical depth and practical applicability of active learning in recommender systems.

REFERENCES

1. M. Elahi, F. Ricci, and N. Rubens, *A survey of active learning in collaborative filtering recommender systems*, Computer Science Review, vol. 20, pp. 29–50, 2016. doi:10.1016/j.cosrev.2016.05.002.
2. T. Geurts, S. Giannikis, and F. Frasincar, *Active learning strategies for solving the cold user problem in model-based recommender systems*, Web Intelligence, vol. 18, pp. 269–283, 2020. doi:10.3233/WEB-210448.
3. De Pessemier, T., Willems, B. and Martens, L., *Active learning algorithm for alleviating the user cold start problem of recommender systems*, Scientific Reports, vol. 15, pp. 24493, 2025. doi:10.1038/s41598-025-09708-2.
4. H. Zhang, I. Ganchev, N. Nikolov, Z. Ji, and M. O'Droma, *FeatureMF: An Item Feature Enriched Matrix Factorization Model for Item Recommendation*, IEEE Access, vol. PP, pp. 1–1, 2021. doi:10.1109/ACCESS.2021.3074365.
5. N. Idrissi and A. Zellou, *A systematic literature review of sparsity issues in recommender systems*, Social Network Analysis and Mining, vol. 10, 2020. doi:10.1007/s13278-020-0626-2.
6. D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, *Using Collaborative Filtering to Weave an Information Tapestry*, Commun. ACM, vol. 35, no. 12, pp. 61–70, 1992. doi:10.1145/138859.138867.
7. S. Atif, S. Qazi, and N. Gillis, *Improved SVD-based initialization for nonnegative matrix factorization using low-rank correction*, Pattern Recognition Letters, vol. 122, 2019. doi:10.1016/j.patrec.2019.02.018.
8. A. M. Rashid, I. Albert, D. Cosley, S. K. Lam, S. M. McNee, J. A. Konstan, and J. Riedl, *Getting to know you: learning new user preferences in recommender systems*, Proceedings of the 7th International Conference on Intelligent User Interfaces, pp. 127–134, 2002. doi:10.1145/502716.502737.
9. M. Elahi, F. Ricci, and N. Rubens, *Active learning strategies for rating elicitation in collaborative filtering: A system-wide perspective*, ACM Transactions on Intelligent Systems and Technology, vol. 5, pp. 1–33, 2014.
10. F. M. Harper and J. A. Konstan, *The MovieLens Datasets: History and Context*, ACM Trans. Interact. Intell. Syst., vol. 5, no. 4, pp. 19, 2015. doi:10.1145/2827872.
11. M. Deshpande and G. Karypis, *Item-based top-N recommendation algorithms*, ACM Trans. Inf. Syst., vol. 22, no. 1, pp. 143–177, 2004. doi:10.1145/963770.963776.
12. D. Valcarce, A. Landin, J. Parapar, and A. Barreiro, *Collaborative filtering embeddings for memory-based recommender systems*, Engineering Applications of Artificial Intelligence, vol. 85, pp. 347–356, 2019. doi:10.1016/j.engappai.2019.06.020.
13. N. Rubens, D. Kaplan, and M. Sugiyama, *Active Learning in Recommender Systems*, in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer US, Boston, MA, pp. 735–767, 2011. doi:10.1007/978-0-387-85820-3_23.
14. M. Elahi, F. Ricci, and N. Rubens, *Active Learning in Collaborative Filtering Recommender Systems*, in *E-Commerce and Web Technologies*, M. Hepp and Y. Hoffner, Eds. Springer International Publishing, Cham, pp. 113–124, 2014.
15. X. N. Lam, T. Vu, T. D. Le, and A. D. Duong, *Addressing cold-start problem in recommendation systems*, Proceedings of the 2nd International Conference on Ubiquitous Information Management and Communication, pp. 208–211, 2008. doi:10.1145/1352793.1352837.
16. N. M. S. Iswari, W. Wella, and A. Rusli, *Product Recommendation for e-Commerce System based on Ontology*, Proceedings of the 2019 1st International Conference on Cybernetics and Intelligent System (ICORIS), pp. 105–109, 2019. doi:10.1109/ICORIS.2019.8874916.
17. M. D. Ekstrand, J. T. Riedl, and J. A. Konstan, *Collaborative Filtering Recommender Systems*, Found. Trends Hum.-Comput. Interact., vol. 4, no. 2, pp. 81–173, 2011. doi:10.1561/1100000009.
18. N. Hazrati and F. Ricci, *Simulating Users' Interactions with Recommender Systems*, Adjunct Proceedings of the 30th ACM Conference on User Modeling, Adaptation and Personalization, pp. 95–98, 2022. doi:10.1145/3511047.3536402.

19. R. G. Crespo, O. S. Martínez, J. M. C. Lovelle, B. C. P. García-Bustelo, J. E. L. Gayo, and P. O. de Pablos, *Recommendation System based on user interaction data applied to intelligent electronic books*, Computers in Human Behavior, vol. 27, no. 4, pp. 1445–1449, 2011. doi:10.1016/j.chb.2010.09.012.
20. G. Piatetsky, *Interview with Simon Funk*, SIGKDD Explor. Newsl., vol. 9, no. 1, pp. 38–40, 2007. doi:10.1145/1294301.1294311.
21. Z. Zajac, *Github: Goodbooks-10k* Available online: https://github.com/zygmuntz/goodbooks-10k (accessed on 31 August 2022).
22. F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. Acm transactions on interactive intelligent systems (tiis) 5, 4 (2016), 19.