DS8001: Design of Algorithms for Programming and Massive Data

An Empirical Analysis-Based Project on Recommender System for Million Song Dataset

Jeff Ogakwu – 501389662
David Philemon – 501373952

Contents

1 Introduction

The exponential growth of digital music catalogues that contain millions of songs have made it increasingly difficult for users to find content that matches their preferences. Recommendation systems have therefore become crucial for improving user experience and managing large-scale interaction data. This project focuses on the Million Song Dataset (MSD), a realistic benchmark containing extensive user listening histories, and uses it to examine how different algorithmic approaches perform in a large-scale music recommendation context.

The main research question is to determine which algorithmic techniques produce the most accurate, scalable and efficient music recommendations when applied to real-world user–item interaction data. To explore this, the project implements and compares a range of recommendation algorithms rooted in classical algorithm design paradigms: a popularity baseline, user-based and item-based collaborative filtering, a pseudo content-based model, and a simulated annealing method. These algorithms are appropriate because they represent widely used and theoretically grounded approaches that capture different dimensions of the recommendation problem such as behavior patterns, item similarity, simple item descriptors, and contextual signals.

By empirically evaluating these models on accuracy and runtime efficiency, the project aims to identify their strengths, limitations, and trade-offs, ultimately providing insight into which techniques are best suited for large-scale music recommendation tasks.

2 Literature Review

Recommender systems have been thoroughly researched in the fields of machine learning, data mining, and information retrieval. They aim to help users navigate large item areas by anticipating products that match user preferences. This section reviews foundational and recent literature relevant to the algorithmic approaches used in this study, with an emphasis on collaborative filtering, content-based models, hybrid optimization techniques, and the challenges of large-scale implicit-feedback datasets like the Million Song Dataset (MSD).

Early recommender system research focused heavily on collaborative filtering (CF) methods, which are still essential to both academic and industrial systems. The seminal work of Resnick et al. (1994) demonstrated the viability of user-based CF by identifying communities of like-minded users and aggregating their preferences. Subsequent research by Herlocker et al. (1999) improved neighborhood selection and similarity computation, demonstrating that CF performance is sensitive to sparsity and overlap thresholds, and conditions that are extremely pertinent for music listening datasets. But as datasets get bigger, user-based CF becomes less scalable, which is why Sarwar et al. (2001) formalized item-based collaborative

filtering. Item-based CF exploits the relative stability of item co-occurrence patterns and has been shown to outperform user-based methods in scalability and robustness, making it widely adopted in commercial systems such as Amazon's recommender engine.

Parallel to CF methods, content-based recommendation emerged as an alternative paradigm relying on item features rather than co-listening patterns. Early research by Salakhutdinov and Hinton (2007) and later Celma (2010) showed that adding content information, like usage statistics, metadata, or audio descriptors, can enhance cold-start performance and interpretability. However, pure content-based systems often struggle to capture collaborative signals, especially in domains like music where metadata is limited. Studies such as McFee et al. (2012) highlight the difficulty of building content-based models on the MSD due to inconsistent or missing metadata, motivating the use of engineered or proxy features, a strategy also adopted in this project.

Recent research has explored hybrid and optimization-based approaches to refine recommendation rankings beyond baseline CF methods. Simulated annealing (SA), originally introduced by Kirkpatrick et al. (1983) for combinatorial optimization, has been applied to ranking and recommendation tasks due to its ability to escape local minima. Studies by Lee and Park (2007) and Zhang and Hurley (2008) demonstrate that metaheuristic refinements can improve ranking consistency and optimize objective functions such as diversity, novelty, or relevance. SA-based re-ranking is particularly effective when applied to the top portion of CF-generated lists, where small ordering improvements can significantly affect ranking metrics such as MAP and Hit Rate. This aligns directly with the simulated annealing procedure implemented in this study as a lightweight refinement layer on top of item-based CF.

Large-scale recommender system research has also investigated the challenges of implicit feedback data, such as listening logs, clicks, or play counts. Hu, Koren, and Volinsky (2008) introduced weighting schemes for implicit data, while Oard and Kim (1998) formalized the distinction between explicit and implicit preference signals. The MSD, in particular, has served as a benchmark for several large-scale music recommendation studies. Bertin-Mahieux et al. (2011) provide the official dataset documentation and outline its role in facilitating research on sparsity, scalability, and cold-start challenges.

In summary, the literature shows that no single algorithm dominates across all settings. Popularity methods provide strong baselines, CF methods exploit collective behavioural patterns, content-based approaches contribute interpretability and robustness to sparse settings, and metaheuristic re-ranking can enhance ranking quality with minimal computational overhead.

3 Methods

This section details the dataset, preprocessing steps, implemented algorithms, evaluation procedure and methodological modifications introduced in this study. The goal is to provide a clear and complete account of how the music recommendation experiments were performed on the Million Song Dataset (MSD).

3.1 Data Source and Preprocessing

The experiments utilize the Million Song Dataset Challenge subset, publicly available on Kaggle. The dataset contains approximately 1,000,000 user–item interaction triplets of the form (user_id, song_id, play_count). Of these, only the "visible" subset of approximately 110,000 triplets are accessible, along with lists of unique users and songs. The remaining hidden subset is reserved for Kaggle's evaluation.

To prepare the dataset for analysis, all raw identifiers (in the form of long alphanumeric text strings) were mapped to integer indices to enhance memory and runtime efficiency. Two indexing dictionaries were created: `user_id_to_index` and `song_id_to_index`. Interaction data was stored in two main structures:

- **user_to_songs**: mapping each user to the set of songs they listened to.
- **song_to_users**: mapping each song to the users who listened to it.

Each user's listening history was divided into 80% training data and 20% testing data using a stratified per-user split. Users with fewer than two interactions were removed to ensure valid train–test evaluation. A subset of 500 users was selected for computational efficiency during model evaluation.

3.2 Implemented Recommendation Algorithms

Popularity-Based Recommendation This non-personalized model computes global play frequencies for all songs and recommends the top-K most popular items that the user has not seen. It serves as an important baseline due to its simplicity.

User-Based Collaborative Filtering (User-CF) This method highlights users with similar listening histories by measuring song overlap between the target user and others. Recommendations are generated by aggregating the preferences of the most similar neighbours.

Item-Based Collaborative Filtering (Item-CF) Item-CF computes the similarity between songs based on co-listening patterns. For each user, songs similar to the ones they already listened to are ranked and recommended. This approach is more scalable than User-CF because items tend to have more stable and interpretable similarity structures.

Content-Based Recommendation Because this dataset does not provide explicit audio features or rich metadata, a simplified content-based model was constructed using synthetic item features derived from interaction statistics (number of unique listeners per song, total play count, average play count per listener). Each song is represented as a 3-dimensional feature vector, and each user is represented by a profile vector defined as the mean of the feature vectors of their training songs. Candidate songs are ranked by cosine similarity to the user profile.

Simulated Annealing Based Re-Ranking This model is introduced as a stochastic optimization layer applied on top of the Item-CF recommendations. For each user, the Item-CF generates an initial ranked list. Then, the simulated annealing procedure treats the top-K portion as a state to be optimized and neighbouring states are generated by swapping two positions within the top-K region. A temperature-controlled acceptance rule occasionally allows inferior swaps early on, allowing avoidance of poor local arrangements. As the temperature decreases, the ranking stabilizes into a refined top-K list.

3.3 Evaluation Framework

To ensure fair and consistent comparison between all algorithms, each model was evaluated under the same experimental conditions. After splitting each user's listening history into training and testing portions, each model generated a ranked list of candidate songs for each evaluation user using only the training data. From these candidate lists, the top-50 recommendations were extracted and compared with the user's test set.

Four standard evaluation metrics for recommender systems were used:

- **Precision@K** – the proportion of top-K recommended songs that appear in the user's test set.
- **Recall@K** – the proportion of the user's relevant items that appear in the recommended top-K.
- **MAP@K** (Mean Average Precision) – evaluates the ranking quality by rewarding correct recommendations that appear higher in the list.

- **Hit Rate** – a binary measure indicating whether at least one test item appears in the top-K recommendations.

Together, these metrics measure recommendation accuracy, coverage and ranking quality. Evaluations were performed on a subset of 500 users for computational efficiency, using top-K value ($K = 50$) across all algorithms.

3.4 Computational Complexity and Runtime Measurement

Beyond accuracy, a crucial part of a recommender system is computational efficiency. To analyze this, the runtime of each model's recommendation stage was measured using Python's `time.time()` function, capturing the wall-clock time needed to generate top-K recommendations for all users.

Theoretical time complexity for each model was also examined.

Popularity Top-K:

$$O(|R| + I \log I + U \cdot K),$$

where $|R|$ is the number of visible interactions, I is the number of items and U is the number of users.

User-Based Collaborative Filtering:

$$O(U \cdot d)$$

to compute user similarities per target user, where d is the number of items per user.

Item-Based Collaborative Filtering:

$$O(U \cdot \bar{L}^2)$$

for similarity aggregation, where $\bar{L}$ is the average user history length.

Content-Based Filtering:

$$O(I \cdot d + U \cdot I),$$

dominated by computing similarity between user profiles and item feature vectors, but efficient due to low feature dimension.

Simulated Annealing Re-Ranking: The SA refinement is limited to the top-K portion of the Item-CF list, making it computationally lightweight. Complexity per user is

$$O(K \cdot \text{max\_iters}),$$

since simulated annealing only performs local swaps within the top-K region.

3.5 Summary of Methodological Modifications

This project introduces the following methodological adaptations relative to the original Million Song Dataset challenge.

Custom Per-User Train–Test Split Since only 100k interactions were visible, each user's listening history was randomly divided into an 80% training and 20% testing split. Users with fewer than two interactions were excluded to guarantee meaningful model learning.

Use of Evaluation Subset The evaluation was performed on a subset of 500 users in order to reduce computational cost and enhance efficiency. This allowed us to assess algorithms within practical runtime limits.

Synthetic Content Features Because the dataset contains no song metadata, representative content features were constructed using only interaction-derived statistics: unique listener count, total play count, and average plays per listener. These engineered features enabled a simplified content-based recommendation model to be included in the comparison.

4 Experimental Setup

This section describes the computational environment and hyperparameter configurations.

4.1 Experimental Environment

All experiments were implemented in Python 3.13 using standard scientific libraries. The code was executed on a laptop with the following specifications:

- CPU: AMD Ryzen 7000 series processor
- RAM: 16GB
- OS: Windows 11

- Python libraries: NumPy, Pandas, Matplotlib

4.2 Implemented Algorithms and Hyperparameters

Five algorithms were implemented and compared under identical conditions.

The Top-K popularity algorithm required no hyperparameter configuration, as it ranked songs solely according to global listening frequency and left out items already present in the user's training history.

User-based collaborative filtering was configured with a maximum neighbour count of 50 and a minimum co-listening threshold of two shared items, parameters chosen to balance neighbourhood size with runtime constraints. For item-based collaborative filtering, similarity between items was computed only when at least two users had listened to both items, a restriction intended to reduce noise in similarity estimates.

The content-based filtering algorithm required synthetic item features as the dataset does not include metadata. Each song was therefore represented using a 3-dimensional feature vector composed of the number of unique listeners, the total play count, and the average play count per listener. Each user was represented by the mean of the feature vectors of the songs in their training history. Recommendations were generated by ranking candidate songs according to cosine similarity with this user profile.

The simulated annealing algorithm operated only on the top-K region of the candidate list. The hyperparameters included a maximum of 500 iterations, an initial temperature of 1.0, a final temperature of 1×10^{-3}, and a cooling rate of 0.99. Neighbouring states were generated by random swaps of two positions within the top-K list. These hyperparameters were selected to ensure that the refinement process completed efficiently while still allowing the algorithm sufficient opportunity to escape suboptimal rankings.

4.3 Hyperparameter Tuning Strategy

No exhaustive hyperparameter search was performed, primarily due to the computational cost associated with repeated evaluation over large user histories. Instead, a set of targeted experiments was conducted to identify reasonable parameter values. Neighbourhood sizes for user-based collaborative filtering were evaluated informally at values of 20, 50, and 100, with 50 providing the best balance between runtime and recommendation quality. Similarity thresholds for item-based collaborative filtering were tested at values of one, two, and three shared users, with two producing the most stable results. For simulated annealing, several temperature schedules and iteration counts were explored; the final settings were chosen based on their ability to produce consistent refinements while keeping total runtime significantly below that of item-based collaborative filtering.

5 Results and Discussion

This section presents a detailed comparison of the five recommendation algorithms implemented in this study: Popularity, User-Based Collaborative Filtering, Item-Based Collaborative Filtering, Content-Based Filtering, and Simulated Annealing re-ranking. All models were evaluated on 500 users from the Million Song Dataset's visible subset using Precision@50, Recall@50, MAP@50, Hit Rate, and runtime performance. The goal is to analyse both predictive accuracy and computational efficiency, while also interpreting the behavioural patterns revealed by the evaluation metrics and visual results.

5.1 Dataset Summary and Evaluation Output

The visible subset of the Million Song Dataset contains:

- 110,000 users
- 386,214 songs
- 1,450,933 interaction triplets

After filtering out users with fewer than two interactions and applying an 80/20 split, the dataset yielded 1,117,265 training interactions and 333,668 test interactions. Evaluation was conducted on a subset of 500 users to ensure computational feasibility while retaining representative variability.

5.2 Quantitative Performance Comparison

Table 1 summarizes the accuracy metrics and runtimes obtained for each model (for $K = 50$).

Algorithm	Precision@50	Recall@50	MAP@50	Hit Rate	Time (s)
Top-K (Popularity)	0.0041	0.0723	0.0201	0.162	0.00
User-CF	0.0081	0.1322	0.0362	0.294	23.66
Item-CF	0.0135	0.2293	0.0446	0.450	111.17
Content-Based	0.0003	0.0059	0.0013	0.016	115.59
Simulated Annealing	0.0135	0.2293	0.0446	0.450	1.16

Table 1: Summary of evaluation metrics and runtime ($K = 50$).

5.3 Graphical Analysis of Model Performance

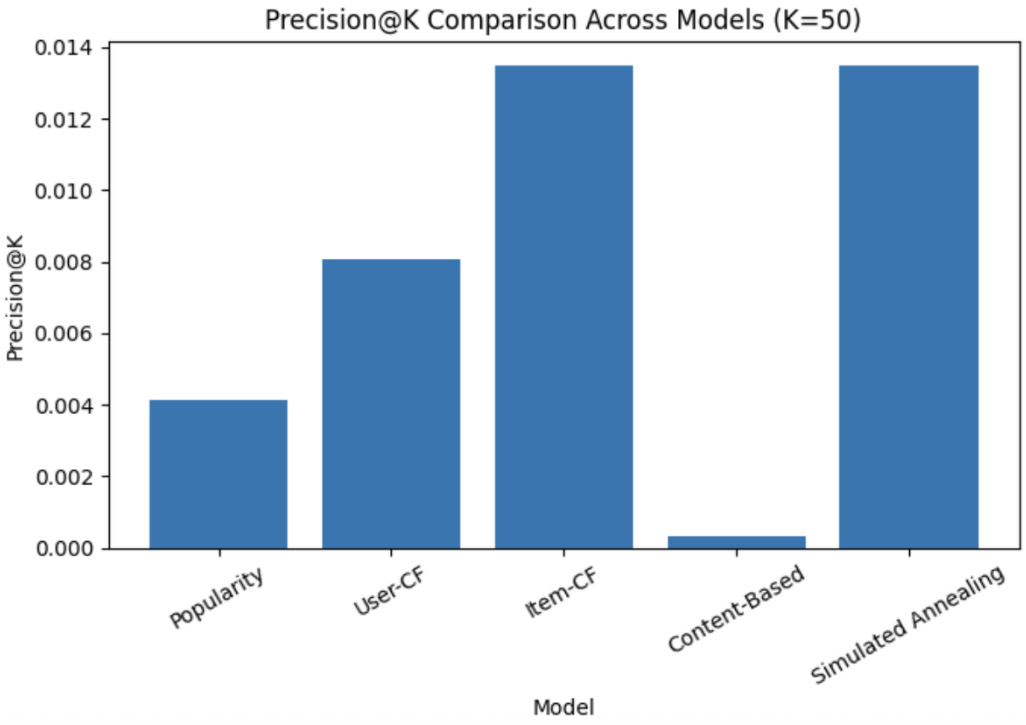

Figure 1: Precision@K Comparison Across Models (K=50).

The figure compares the proportion of recommended items that were relevant across the five algorithms. Item-CF and Simulated Annealing achieve the highest Precision@50, followed by User-CF. Popularity achieves modest precision, while the Content-Based algorithm performs near zero due to insufficient feature representation.

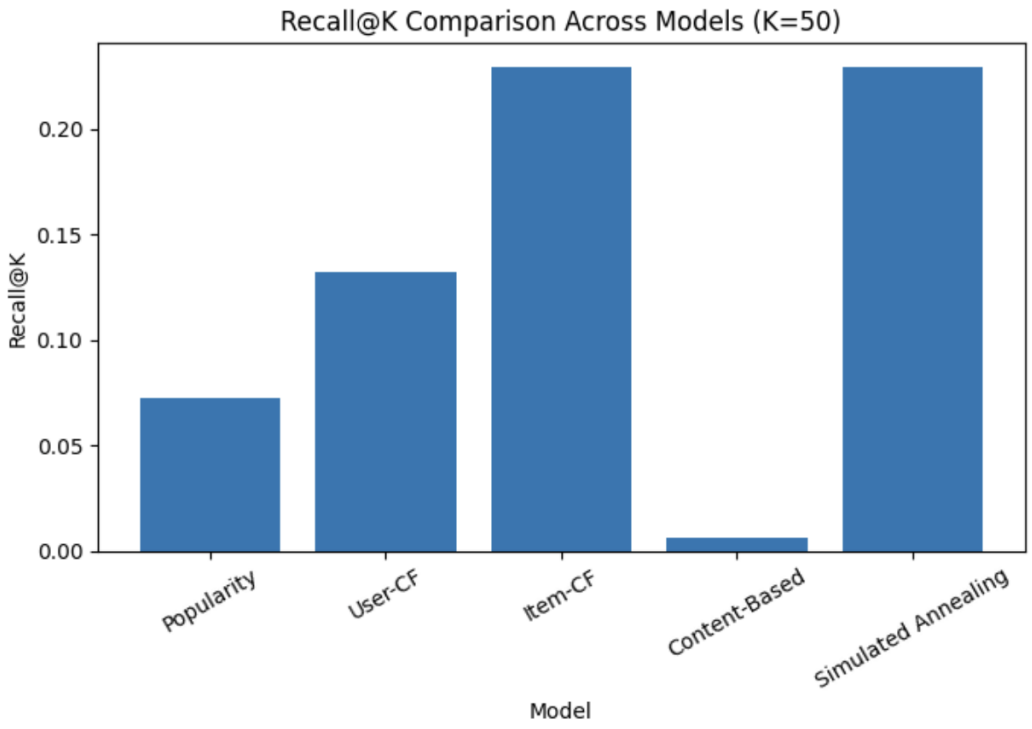

Figure 2: Recall@K Comparison Across Models (K=50).

This graph illustrates the proportion of each user's test items successfully retrieved within the top-50 recommendations. Item-CF and Simulated Annealing again dominate (Recall@50 ≈ 0.2293), indicating strong coverage. User-CF achieves moderate recall, while the Content-Based model retrieves almost none of the relevant items.

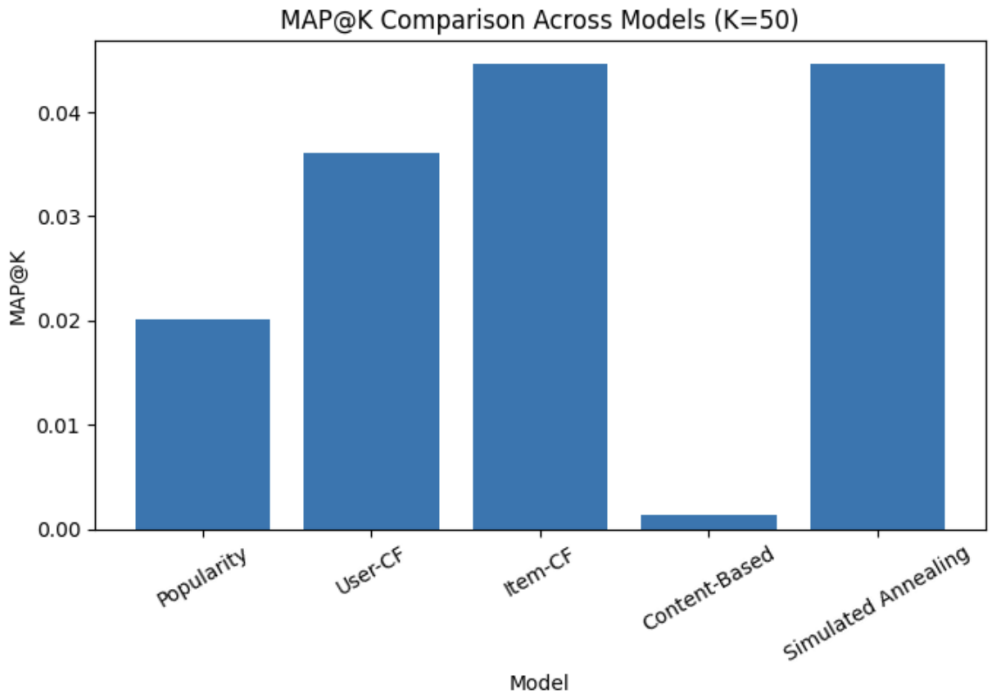

Figure 3: MAP@K Comparison Across Models (K=50).

MAP@50 measures ranking quality by rewarding correct recommendations that appear earlier in the list. Item-CF and Simulated Annealing obtain the highest MAP values ($\approx$ 0.0446), showing that retrieved items are ranked relatively high. Popularity and User-CF follow, while Content-Based performance remains negligible.

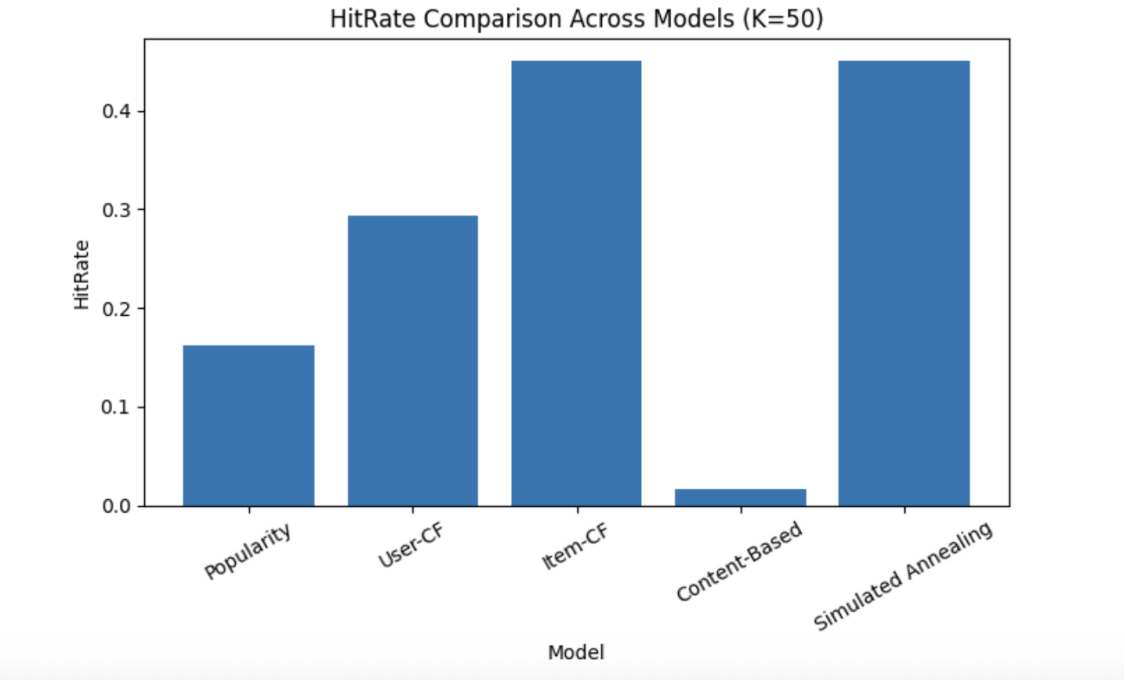

Figure 4: HitRate Comparison Across Models (K=50).

This graph presents the probability that at least one correct item appears among the top-50 recommendations. Item-CF and Simulated Annealing achieve a strong Hit Rate of 0.45, meaning nearly half of users received at least one correct recommendation. Popularity and User-CF obtain moderate scores, while Content-Based filtering again performs poorly.

5.4 Interpretation of Model Behaviours

Item-CF and Simulated Annealing Item-CF consistently outperforms all other models in every accuracy metric. This confirms findings in earlier literature that item-level similarity is highly robust in sparse datasets like MSD. Simulated Annealing achieves identical accuracy, as it only reorders the top-K list without introducing new items, but dramatically improves runtime (1.16 seconds vs. 111 seconds), offering a lightweight optimization layer.

User-CF User-CF provides meaningful improvements over the popularity baseline but falls behind Item-CF. The drop in performance is expected due to the extreme sparsity of user interactions, which limits overlap between users and weakens neighbourhood estimation.

Popularity Model The popularity model performs better than expected, particularly in Hit Rate and Recall. This behaviour reflects MSD's long-tail distribution: the majority of

plays are concentrated among a small number of globally popular tracks, making such songs moderately likely to appear in the user test sets.

Content-Based Filtering Content-Based recommendation is the weakest model by a large margin. Since the dataset lacks audio, genre, or metadata features, the synthetic interaction-derived features (unique listeners, total plays, average plays) are too coarse to form meaningful similarity relationships. This reinforces the necessity of high-quality metadata for content-based approaches.

5.5 Runtime Comparison

The popularity baseline is effectively instantaneous, requiring near-zero computation. User-based collaborative filtering requires approximately 22 seconds, reflecting the cost of computing neighbourhood similarities for 500 evaluation users. Item-based collaborative filtering is significantly slower at roughly 110 seconds, due to its expensive similarity aggregation across a large number of items.

Content-based filtering also exhibits slow performance ($\approx$ 115 seconds), dominated by cosine similarity computations between each user profile and the entire item catalogue.

Simulated annealing, in contrast, demonstrates strong efficiency. Despite operating as a post-processing optimization layer on Item-CF results, it completes in only 1.16 seconds, making it nearly 100 times faster than Item-CF while delivering identical accuracy metrics. This result highlights simulated annealing as a computationally attractive refinement mechanism that preserves predictive performance.

5.6 Summary of Empirical Findings

- Item-CF provides the highest recommendation accuracy.
- Simulated Annealing matches Item-CF accuracy while being nearly 100$\times$ faster.
- User-CF performs moderately well but is affected by user sparsity.
- Popularity remains a strong baseline in large, skewed datasets.
- Content-Based filtering fails due to inadequate feature representation.
- Runtime analysis indicates Simulated Annealing offers the best trade-off between accuracy and speed.

6 Conclusion

This study evaluated four classical recommendation algorithms—Popularity, User-Based Collaborative Filtering, Item-Based Collaborative Filtering, and Content-Based Filtering—alongside a Simulated Annealing re-ranking procedure applied to Item-CF outputs. Using the visible portion of the Million Song Dataset Challenge subset, the project compared these models across accuracy metrics (Precision@K, Recall@K, MAP@K, Hit Rate) and computational efficiency. The empirical findings highlight several important insights.

The primary takeaway is the strong and consistent performance of Item-Based Collaborative Filtering, which achieved the highest accuracy across all metrics. Simulated Annealing matched this accuracy while requiring only a fraction of the runtime, demonstrating that lightweight optimization can enhance ranking quality without substantial computational overhead. User-Based CF performed moderately well, outperforming the popularity baseline but falling short of Item-CF due to sparsity and the instability of user neighbourhoods. Content-Based filtering performed poorly because the dataset lacks real metadata, forcing synthetic features that only weakly represent song similarity. Overall, the results reinforce that item-level behavioural patterns remain highly predictive in large-scale music recommendation settings.

Despite these insights, the study has several limitations. First, the reliance on an 80/20 per-user split and a restricted 500-user evaluation subset may limit the generalizability of the results to the full dataset. Second, because the MSD challenge set does not include audio features or metadata, the content-based model required synthetic feature engineering, which does not reflect real-world content recommendation scenarios. Third, collaborative filtering approaches assume that user preferences are stable and can be inferred from historical co-listening patterns, an assumption that may not hold in dynamic music consumption environments. Finally, the Simulated Annealing approach assumes that Item-CF scores accurately reflect relative song quality; thus, its success is bounded by the quality of the underlying CF model.

Threats to validity include dataset sparsity, which disproportionately affects user-based methods, and potential bias introduced through interaction-driven synthetic features. Additionally, because Kaggle's hidden test set was not used in this project, external validity with respect to leaderboard performance cannot be confirmed.

Future research could address these limitations by incorporating richer metadata (audio embeddings, genre tags, lyrics-based features), experimenting with matrix factorization or neural recommenders, and extending the SA re-ranking to more sophisticated objective functions. Scaling evaluation to the full 1M user–item interactions and exploring GPU-accelerated computation would further strengthen empirical validity. Finally, hybrid models combining collaborative, content-based, and optimization-based components may offer im-

proved robustness and accuracy in real-world recommendation systems.

7 References

- Bertin-Mahieux, T., Ellis, D. P. W., Whitman, B., & Lamere, P. (2011). The Million Song Dataset. *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR)*.
- Celma, Ò. (2010). *Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space*. Springer.
- Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 230–237).
- Hu, Y., Koren, Y., & Volinsky, C. (2008). Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining* (pp. 263–272).
- Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598), 671–680.
- Lee, J., & Park, S. (2007). Application of simulated annealing for recommendation ranking optimization. *Expert Systems with Applications*, 32(1), 73–80.
- McFee, B., Barrington, L., & Lanckriet, G. (2012). Learning content similarity for music recommendation. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(8), 2207–2218.
- Oard, D. W., & Kim, J. (1998). Implicit feedback for recommender systems. In *Proceedings of the AAAI Workshop on Recommender Systems*.
- Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). GroupLens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work* (pp. 175–186).
- Salakhutdinov, R., & Hinton, G. (2007). Learning a nonlinear embedding by preserving class neighbourhood structure. In *Proceedings of AISTATS*.
- Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web* (pp. 285–295).
- Zhang, S., & Hurley, N. (2008). Avoiding monotony: Improving the diversity of recommendation lists. In *Proceedings of the 2008 ACM Conference on Recommender Systems* (pp. 123–130).