SOLVING THE COLD START PROBLEM IN RECOMMENDATION SYSTEMS

FINAL PROJECT REPORT

Gorden Jin
Carnegie Mellon University
Pittsburgh, PA 15213
yijin2@andrew.cmu.edu

Tom Gao
Carnegie Mellon University
Pittsburgh, PA 15213
tomgao@andrew.cmu.edu

Keerthi Kunnumkai
Carnegie Mellon University
Pittsburgh, PA 15213
kkunnumk@andrew.cmu.edu

Leo Zhuang
Carnegie Mellon University
Pittsburgh, PA 15213
muyangz@andrew.cmu.edu

May 8, 2026

1 Introduction

In a world where the Internet and social media have become such an integral part of our lives, the sheer volume of information we encounter every day can be overwhelming. From streaming platforms suggesting our next favourite movie, to online stores curating products tailored to our tastes, and to social media platforms, recommendation systems play a crucial role in helping us navigate this digital abundance. Recommendation systems provide items or experiences that may be of interest to the users based on their previous interaction data saved in the system.

One approach to designing a recommendation system is the collaborative filtering approach. Collaborative filtering generates recommendations for users by analysing past user data, and comparing it with similar users. However, one problem with this approach is that new users cannot be suggested accurate recommendation due to lack of detailed preference information. This is the user cold start problem, which creates a problem of information overload for the user. Currently, the two major types of approaches used to solve cold start problem are data-driven techniques and approach-driven techniques. [Yuan and Hernandez, 2023] Data driven techniques focus on collecting more data about a new user in order to make more accurate decisions, while approach driven techniques attempt to solve the problem by using another algorithm. Much research has been done on both types of techniques.

This project will focus on approach-driven techniques in addressing the user cold start problem. This report will analyse and synthesise key contributions and methodologies, and implement some of the results. Our implementation will attempt to combine the different methods and evaluate the performance. In particular, this report reviews existing literature on the user-cold start problem in recommender systems and verifies several findings reported in works such as Duricic et al. [2018] and Panteli and Boutsinas [2023]. We reproduce and evaluate key techniques from the literature, and compare them against our own implementations to assess their effectiveness in cold-start scenarios.

2 Relevant Works

The user cold-start problem is a well-studied field. In this section, we review several popular approach-driven solutions to the cold start problem. Some methods extend matrix factorization by predicting user latent vectors from auxiliary signals [Sedhain et al., 2017], while others rely on relational information, such as trust propagation networks, to infer user similarity under sparse ratings [Duricic et al., 2018]. A third popular category of the works builds clustering-based models that group users by behavioral or attribute patterns to provide recommendations in pure cold-start scenarios

[AL-Bakri and Hassan, 2019] [Panteli and Boutsinas, 2023] [Panteli and Boutsinas, 2023]. More recent approaches also introduce deep architectures that combine embeddings with additional side information [Mondal and Bhowmik, 2022].

The following subsections summarize representative papers from each of these categories. Together, they give a picture of the current popular methods to solve the cold start problem, either by estimating latent preferences, spreading information through networks, or grouping users into clusters. These prior works motivate the set of techniques we later experiment.

One of the earliest influential works on the cold-start problem is Schein et al. [2002]. This work investigated combining content-based and collaborative filtering models to overcome the lack of initial user ratings. This study showed that pure collaborative filtering performs poorly in sparse settings and that integrating item attributes can significantly improve predictions.

2.1 Low-rank linear cold-start from Social data

Sedhain et al. [2017] implements Low rank linear Cold start (LoCo) model to tackle the cold start problem with a matrix factorization frame work. Instead of estimating user embeddings purely from ratings, LoCo turns the cold start problem into a linear prediction problem in latent space using side information such as social connections. Conventional matrix factorizations require full user-item interaction matrix to infer latent vectors. Instead LoCo learns a transformation that maps user social features to item latent space. This allows the system to generate meaningful recommendation even for users with no historical ratings.

Our project uses singular value decomposition based matrix factorization for the baseline model. While we do not incorporate social features like LoCo does, we use the idea that latent representations are key to improving recommendations.

2.2 Tackling the Cold Start Problem in Collaborative Filtering using Regular Equivalence

While LoCo extended matrix factorization using side information, use of trust networks to compensate for missing data emerged in parallel. In Duricic et al. [2018], to tackle the cold start problem in collaborative filtering, a trust network is used to generate similarity matrix, which is then used to select k-nearest neighbours to recommend items. Trust statements from users are first gathered from platforms such as Eopinions Massa and Avesani [2007] and used to form trust network. But trust networks are often sparse, so transitivity property of trust network is used to establish connections between users who are not directly connected. Pairwise similarities between users are then calculated using an iterative approach, which allows us to decide how far we want to propagate trust in the network. In the end, various normalization techniques are applied to get a better distribution of similarity values and for better evaluation of results.

Our project does not implement methods that are trust propagation based, since they require explicit social network information between users. Datasets like MovieLens do not contain such relational data. Thus, constructing a trust-graph would introduce some modelling challenges and addressing it would go beyond the project timeline. Thus, we instead focus on methods that can be implemented from ratings and user metadata.

2.3 Differentiating Regularization Weights – A Simple Mechanism to Alleviate Cold Start in Recommender Systems

Chen and Chen [2019] argue that standard matrix factorization (MF) approaches for recommendation with uniform regularization weights oversimplify the diversity among users and items, especially under cold-start or long-tail conditions. Instead, they propose differentiating the regularization strength: imposing weaker regularization (smaller penalty) on latent factors of well-revealed or well-studied users/items, including active users or popular items with may interactions, while applying stronger regularization on sparsely observed users/items. By integrating this weighting scheme into common MF-based models such as SVD, SVD++, or NMF, they demonstrate improved rating prediction accuracy across several public datasets. Specifically, their method better predicts ratings on long-tail items, suggesting a partial solution to the item cold-start problem. However, their method has mixed improvements on different datasets compared to the baseline methods, and the model does not perform very well against user-cold-start situations. Overall, the paper contributes a general, easily implementable regularization strategy that improves MF's robustness and accuracy to sparsity and long-tail distributions, which offers an alternative to more complex side-information or cross-domain methods.

2.4 A Proposed Model to Solve Cold Start Problem using Fuzzy User-Based Clustering

While the matrix factorization extensions and trust based propagation methods are two prominent ways to address cold start problem, clustering methods groups users according to behaviour or profile based similarities. Clustering allows new users to inherit preference information from an entire group rather than from individually matched neighbours.

A representative example of this is the work by AL-Bakri and Hassan [2019], where they propose an enhanced collaborative filtering recommender system that addresses the cold-start and data sparsity problems by integrating fuzzy clustering into the user similarity computation. Traditional collaborative filtering depends on users' rating histories, which limits its effectiveness for new users with few ratings. To overcome this, the authors introduce a fuzzy C-means (FCM) clustering approach that groups users based on "truthfulness information," including activity level, rating consistency, and friend-based reliability. A new fuzzy user similarity formula combines traditional Pearson correlation with fuzzy truthfulness similarity through a weighted coefficient. Experiments using the MovieLens 100K dataset show that this fuzzy-based model significantly improves recommendation accuracy and robustness under cold-start and sparse data conditions, outperforming conventional methods.

While LoCo handles cold-start users by predicting their latent factors from side information, fuzzy clustering groups users based on simple behavioural patterns. Both models try to compensate the lack of ratings, but LoCo leans a predictive mappers whereas fuzzy clusturing creates structural groupings.

2.5 A DNN-based cross-domain recommender system for alleviating cold-start problem in e-commerce

Wang et al. [2020] proposes a deep-learning framework that uses knowledge from an auxiliary Ads domain to improve product recommendations for cold-start users in an e-commerce field. Prior work on cold-start recommendation has centered on hybrid models that combine collaborative filtering with content-based features, which is addressed in most of the above literature reviews, but these often struggle to capture both semantic and structural information or rely on limited user overlap. This paper proposes the method of integrating Word2Vec-based textual embeddings with R-metapath2Vec graph-structured embeddings and feeding them into a DNN, then using a stacking mechanism to enhance user and item representations across domains. Empirical results show that this hybrid embedding approach improves recommendation accuracy for latent users compared with traditional methods such as collaborative filtering and earlier cross-domain methods.

While the approach is innovative in bridging Ads and shopping domains and demonstrates strong performance, it depends on the availability and quality of shared-user data and does not fully address issues like temporal preference drift or fine-grained aspect-level modeling. This paper serves as an important early contribution to deep cross-domain cold-start recommendation, and later research can focus on developing more flexible and generalizable transfer-learning techniques.

2.6 A Deep Neural Network Framework for Cold Start Problem in Recommender systems

Mondal and Bhowmik [2022] proposes a framework called "DeCS" to address cold start problem in collaborative filtering based recommender systems. The architecture first takes in the matrix representation of user ratings of items as an input. DeCS architecture reduces the dimension of the rating matrix using embedding. The model also uses side information such as age, demographics, gender and genres of movies to predict the rating of an item, especially for cold start users. Side information (SI) is transformed into a vector using one-hot embedding. SI vectors and embeddings are then separately passed into a fully connected neural network to extract features from users and items.

In our project, we follow a simplified version of this idea by separately learning user and item embeddings through SVD and then combining them with user metadata or clustering information to approximate the training process described in DeCS. While DeCS fuses user-item embeddings and side-information vectors through multiple dense layers for rating prediction, our baseline model explores these components independently to explore how each (collaborative vs demographic) contributes to improving recommendations for cold-start systems.

2.7 Addressing the Cold-Start Problem in Recommender Systems Based on Frequent Patterns

Panteli and Boutsinas [2023] proposes a clustering-based approach to address the cold-start problem by grouping users according to their profile attributes and then applying discriminant frequent pattern mining within each cluster to discover itemsets that are highly favored by that specific group of users. These "discriminant" patterns are then used to infer missing preferences for new users, enabling recommendations even when no prior ratings exist. *Panteli and Boutsinas* suggest that, compared to collaborative filtering, which relies heavily on historical user–item interactions and struggles when such data is unavailable, this method performs well for true cold-starts and maintains high precision by

avoiding irrelevant recommendations. However, because recommendations are made at the cluster level rather than tailored to individual behavior, personalization is weaker than CF once sufficient interaction data is available, and the approach relies on forming high-quality clusters to work effectively.

Compared to the Fuzzy user-based clustering above (See section 2.4), which builds clusters based on profile attributes, Panteli and Boutsinas [2023] clusters based on truthfulness features derived from their actual rating behaviour.

2.8 User Cold Start Problem in Recommendation Systems: A Systematic Review

Yuan and Hernandez [2023] divides the solutions to the user cold-start problem into two main categories: data-driven techniques and approach-driven techniques. The data-driven category focuses on the use of external information such as cross-domain data, social network data, and other auxiliary user attributes to improve recommendations for new users. The approach-driven category develops or enhances algorithms to better handle cold-start, and it is further broken down into five sub-categories: meta-learning, deep learning, matrix factorization, improved collaborative filtering, and improved content-based methods. This review provides a good overview of the popular methods we could use to tackle the cold start problem.

3 Data and Setup

In this section, we pick a few recommendation models from the articles that we reviewed, and evaluate their performance.

3.1 Dataset

The datasets we used for training and testing the models are the MovieLens-100k and MovieLens-1M datasets from Harper and Konstan [2015]. The first one consists of 100,000 ratings on around 1700 movies from 1000 users while the second one contains 1000000 ratings on around 4000 movies from 6000 users. The dataset consists of 3 files:

- ratings.dat: This file contains the UserID, MovieIDs, Ratings on a 5 star scale and Timestamps. Each user gives at least 20 ratings.
- user.dat file contains all the demographic information that are provided by the users like UserID, Gender, Age, Occupation and Zip-code. Age is grouped into 7 categories and occupation is grouped into 21 categories.
- movies.dat contains the MovieID, Title and Genres.

We will mainly use the first file in our experiment, because we are mostly evaluating approach-driven models, which would not rely on demographics data. They are only used as side information when appropriate.

To simulate the cold-start problem, we randomly select 20% of the users to be cold-start users. For each cold-start user, we reveal 5 ratings in the training data, and treat the rest as test data, representing a partial cold start.

3.2 Baseline Models

We selected 3 models as our baseline model:

1. BaselineOnly from the scikit-surprise library, introduced by Koren [2010]. This algorithm works by predicting
$$\hat{r}_{ij} = \mu + b_i + b_j \tag{1}$$
where μ is the global mean rating and b_i, b_j represents the biases, or deviation from the average, of user i and item j, respectively.
2. SVD algorithm, from the scikit-surprise library. This algorithm is a famous matrix factorisation algorithm and improves on the baseline model by incorporating latent factors $\mathbf{q}_j$ and $\mathbf{p}_i$.
$$\hat{r}_{ij} = \mu + b_i + b_j + \mathbf{q}_j^\top \mathbf{p}_i \tag{2}$$
The algorithm minimizes the following regularized squared error
$$\sum_{r_{ij} \in R_{train}} (r_{ij} - \hat{r}_{ij})^2 + \lambda(b_i^2 + b_j^2 + \|\mathbf{q}_j\|^2 + \|\mathbf{p}_i\|^2) \tag{3}$$
where the parameters are found via gradient descent.
3. k-Nearest Neighbors (kNN) with $k = 40$ using user-based cosine similarity. This is a very basic collaborative filtering algorithm.

3.3 Evaluation Metrics

We use the following 4 evaluation metrics:

1. Root Mean Square Error (RMSE). The formula is as follows:

$$RMSE = \sqrt{\frac{1}{|\hat{R}|} \sum_{\hat{r}_{ij} \in \hat{R}} (r_{ij} - \hat{r}_{ij})^2} \tag{4}$$

   Where $\hat{R}$ is the set of predictions.

2. Mean absolute error (MAE). The formula is

$$MAE = \frac{1}{|\hat{R}|} \sum_{\hat{r}_{ij} \in \hat{R}} |r_{ij} - \hat{r}_{ij}| \tag{5}$$

3. Precision at 10. The formula is

$$\text{Precision at 10 for user } i = \frac{|\text{Items recommended to user } i \text{ that are relevant}|}{|\text{Recommended Items}|} \tag{6}$$

   Where an item j is recommended if the predicted value $\hat{r}_{ij}$ is greater than some threshold and is among the top 10 highest predicted ratings for this user, and an item j is relevant if the actual rating r_{ij} is greater than some threshold. In our experiment, the threshold is 3.5. The final precision at 10 is the mean precision among all users.

4. Recall at 10. The formula is

$$\text{Recall at 10 for user } i = \frac{|\text{Items recommended to user } i \text{ that are relevant}|}{|\text{Relevant Items}|} \tag{7}$$

   The threshold of relevance is 3.5. The final recall at 10 is the mean recall among all users.

All these metrics are widely used in research papers around the topic of cold start problems.

3.4 Libraries

Our code is written entirely in python. We primarily make use of the following libraries

- scikit-surprise: for baselines and dataset utilities
- scikit-learn: for K-means clustering, preprocessing, and evaluation metrics.
- scikit-fuzzy: for implementing the fuzzy algorithm.
- NumPy/Pandas for data handling and metric computation.

4 Methodology

4.1 Popularity-Based Approach

To begin with, we implement a simple non-personalized model that scores items solely based on their overall popularity in the training dataset. The model make use of Surprise framework to convert popularity counts into predicted rating values, with more frequently rated items receiving higher predicted ratings. These predicted ratings are then used to rank items for each user. We refer to this as the global popularity model. This model provide a solid baseline for evaluating whether more sophisticated models meaningfully improve the recommendations.

4.2 RDFSVD

This method is adopted from Chen and Chen [2019]. As explained above in section 2.3, this method is basically the same as the famous SVD algorithm, which is one of our baseline algorithms, with the exception that the objective function is replaced to give more weights to users who rate more items and items that have more ratings. The new objective function looks like

Metric	Baseline	SVD	kNN
RMSE	1.0170	1.0136	1.0370
MAE	0.8073	0.8062	0.8192
Precision@10	0.7730	0.7582	0.7755
Recall@10	0.2783	0.2689	0.2765

Table 1: Performance Comparison of Baseline Models

$$\begin{aligned}\mathcal{L}(\theta) = \frac{1}{2}\sum_{(i,j)\in\mathcal{K}}(r_{ij}-\hat{r}_{ij})^2 + \frac{\lambda_p}{2}\sum_{i\in\mathcal{K}_U}\frac{1}{f(R^{(U)}(i))}\|\mathbf{p}_i\|_2^2 + \frac{\lambda_q}{2}\sum_{j\in\mathcal{K}_I}\frac{1}{f(R^{(I)}(j))}\|\mathbf{q}_j\|_2^2 \\ + \frac{\lambda_U}{2}\sum_{i\in\mathcal{K}_U}\frac{1}{f(R^{(U)}(i))}\|b_i^{(U)}\|_2^2 + \frac{\lambda_I}{2}\sum_{j\in\mathcal{K}_I}\frac{1}{f(R^{(I)}(j))}\|b_j^{(I)}\|_2^2\end{aligned} \tag{8}$$

where $R^{(U)}(i)$ is the set of items rated by user i, $R_j^{(I)}$ is the set of users who rated the item j, $\mathcal{K}$ is the set of all user item index pairs, $\mathcal{K}_U$ and $\mathcal{K}_I$ are the set of all known users and items, respectively. θ represents all parameters to learn, which includes all the biases and latent factors, namely, $\mathbf{p}, \mathbf{q}, b^{(U)}, b^{(I)}$. The λ's are regularisation factors. The algorithm gets its name from the function f, which is called the RDF, because f differentiates the regularisation weights. The authors proposed 3 different possibilities for f, and we adopted $f(x) = \log(x + c)$ for $c \geq e$. For all other parameters, we adopted the authors' default configurations.

4.3 K-Means Clustering Method

We used K-means clustering to cluster users in the SVD latent factor space to generate group-level recommendations based on shared preferences. After training SVD on the available ratings (including only a few revealed ratings for cold-start users), each user's latent vector is extracted and clustered with k-means, grouping users who have similar inferred taste profiles. For each cluster, the algorithm aggregates ratings from users in that group, computes the average rating per movie, and selects the highest-rated items to form a cluster-specific Top-N list. Any user can then be assigned to the nearest cluster using their SVD embedding, and the system recommends the movies most preferred by that cluster. This wass a simple method we came up to attempt blend matrix factorization with unsupervised clustering to solve the cold-start problem where user ratings are sparse.

4.4 Fuzzy User-Based Clustering Approach

This method is inspired by the paper by AL-Bakri and Hassan [2019], where the authors described an algorithm to predict users' ratings to movies and generate recommendations using the Fuzzy C-Means (FCM) approach.

The algorithm consists of a training stage and a testing stage. For the training stage, we first create a user-movie rating matrix, where each entry is the rating for a movie from a user (empty cells are filled with 0.0). Then we use Pearson correlation coefficient to create a similarity matrix among users, which we label as Similarity Matrix 1 (SM1). Then for each user, we compute their number of ratings, the variance in their ratings, and the max average rating of their closest k neighbors, which are determined by SM1. The FCM approach is then applied to the users with these information, and the fuzzy similarity matrix is generated (SM2). The combined similarity is a weighted average of SM1 and SM2.

The testing stage is highly similar to the training stage. In order to predict the rating for a movie (i) from a user (u), we use the exact process from the training stage, and refit the similarity matrix so that it contains (u). Finally, we use the formula

$$Predict(u,i) = \bar{r_u} + \frac{\sum_{v\in U} sim(u,v) * (r_{v,i} - \bar{r_v})}{\sum_{v\in U}|sim(u,v)|} \tag{9}$$

where $\bar{r_u}$ is the mean rating of the user u, U are neighbors of u, $r_{v,i}$ is user v's rating to movie i, and $sim(u,v)$ is the combined similarity between user u and user v.

Metric	Global Popularity	RDFSVD	Cluster	Fuzzy	Ensemble
RMSE	1.8538	1.3434	1.1805	1.1480	1.0491
MAE	1.5763	1.0437	0.9264	0.8941	0.8359
Precision@10	0.6450	0.5931	0.7251	0.6920	0.7576
Recall@10	0.1240	0.2403	0.2703	0.2388	0.2639

Table 2: Performance of different recommendation models

4.5 Ensemble

We have also implemented an ensemble model. The ensemble model contains 3 copies of each of the 3 methods above (RDFSVD, K-Means Clustering, Fuzzy), and employs the concept of bootstrap aggregation. That is, each child model is trained with a resampled version of the training data with replacement, of the same size. The final prediction will be the average of the prediction of the 9 child models.

One major reason of implementing an ensemble method is because of the poor performance of the basic models. Table 2 lists the performance of the 4 methods above, while table 1 lists the performance of the 3 baseline models. We can see that the performance of the models is not as good as the baseline models. Because of this, we decided to combine the 3 methods in an ensemble method, aiming to improve the performance. Another reason is that, as can be seen in section 2, researchers in machine learning have developed many different recommendation systems to solve the cold start problem, ranging from data driven to approach driven, from matrix factorisation to deep learning methods. Even the 3 basic methods above are different in their own ways. We want to experiment with possible ways to incorporate the various types of methods into one model and see how it would perform.

5 Results

As mentioned above, the 4 methods that we have implemented all performed worse than the baseline models. The ensemble method is no exception. Across all four figures in Figure 1, we see that the baseline models (Baseline CF, SVD, and kNN) consistently outperform the implemented method at all cold-start levels. In particular, the baseline methods maintain high precision and recall throughout, while all four implemented methods stay noticeably below those curves. Even as more ratings are revealed, the gap between the baselines and the implemented models remains significant. We notice that the Root Mean Square Error and Mean Absolute Error of the Ensemble method also remain higher than the baseline methods, while the recall and precision of the ensemble method are barely reaching the minimum of the 3 baselines. This means that the ensemble method is less accurate than baseline models, and is less likely to recommend relevant items. The performance of the individual base models inside the ensemble is even weaker, which explains why the ensemble also struggles to match baseline performance.

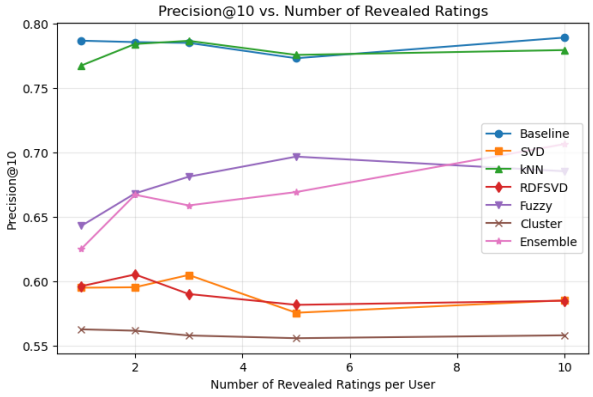
However, on the bright side, we can notice that the ensemble method performed better than the 3 methods that were incorporated. In Figure 1, the ensemble curves consistently sit above RDFSVD, Fuzzy, and Cluster across nearly all metrics. This shows that the ensemble method has the potential to improve the recommendation accuracy beyond the basic models, even if the basic models often disagree, as in the case of our experiments.

There are certainly many ways to improve the results. One possible improvement is to adjust the hyperparameters of the models, possibly using cross-validation. By fine-tuning hyperparameters, we can potentially improve both the basic and ensemble models. Another possible improvement is to include more copies of basic models in the ensemble model. This could lower the variance and increase the correctness of predictions.

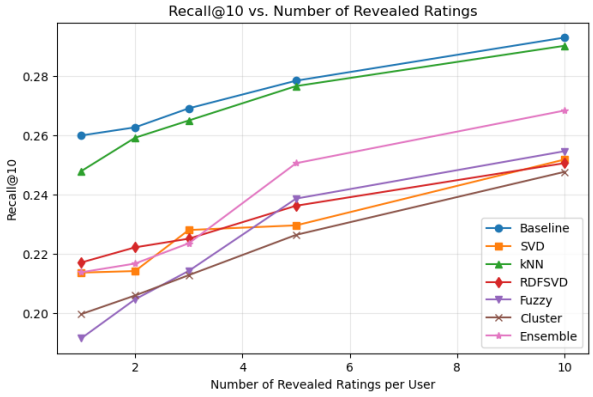
A major problem with the ensemble method is that it is very slow compared to other methods. This is expected because to train an ensemble method, one needs to train multiple models all at once. Our ensemble model takes about 3 times the time to train and test all 3 basic models.

6 Discussion and Conclusion

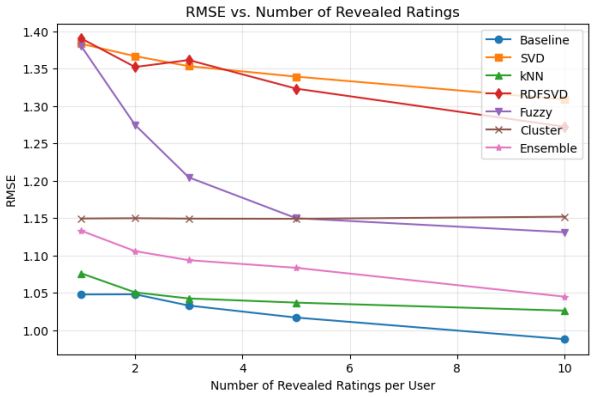
Our finding reinforce a well-known trend in recommender systems: cold-start problem fundamentally limiting the effectiveness of sophisticated collaborative filtering models. Latent factor models like SVD and RDFSVD require more user interaction to meaningfully shape user embeddings. Whereas the cluster methods struggle with personalized ratings in low-data scenario. By contrast, non-personalized baseline models like SVD and kNN performs surprisingly well.



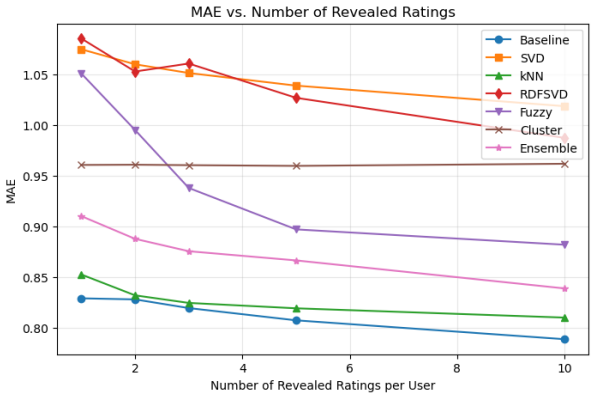
(a) Precision@10 vs. Number of Revealed Ratings



(b) Recall@10 vs. Number of Revealed Ratings



(c) RMSE vs. Number of Revealed Ratings



(d) MAE vs. Number of Revealed Ratings

Figure 1: Cold start evaluation metrics across varying numbers of revealed user ratings

In this project, we have explored many strategies for addressing the cold start problem in recommender systems with collaborative filtering. We have conducted a literature review to understand current approaches, and analysed the current models in research and their advantages as well as limitations. We have also evaluated some models on the datasets provided and attempted to incorporate the different existing approaches by an ensemble method.

6.1 Possible Future Directions

The field of recommender systems in general is grounded upon methods like collaborative filtering and matrix factorization. However, the field is gradually moving towards deep learning models as well as hybrid approaches. This has been suggested in the paper by Wang et al. [2020], and the researchers indeed realized that hybrid methods include both DNNs and traditional methods perform better in general. Therefore, one future direction is to use advanced embedding techniques, such as deep learning or graph-based models, which may capture complex user-item interactions more effectively. One possible direction is to explore different architecture and parameters in the deep learning network to determine which models generate the best predictions. There are many different combinations of methods and parameters, and they may lead to advancements in the field of recommender systems.

Compared to the basic models, the ensemble method has displayed promising results in our study. It could be improved with adaptive weighting, while using more powerful boosting algorithms could enhance recommendation relevance. Additionally, the field is moving towards incorporating richer contextual and temporal information valuable for recommender systems. Future research is likely to focus on integrating these dimensions into models in a systematic way, which balances predictive performance with practical considerations. Finally, in order to improve model performance, future research should also tune hyperparameters and improve scalability. They should also consider metrics that evaluate different aspects of the prediction, leading to more balanced and user-centric recommendations.

References

Hongli Yuan and Alexander A. Hernandez. User cold start problem in recommendation systems: A systematic review. *IEEE Access*, 11:136958–136977, 2023. doi:10.1109/ACCESS.2023.3338705.

Tomislav Duricic, Emanuel Lacic, Dominik Kowald, and Elisabeth Lex. Trust-based collaborative filtering: tackling the cold start problem using regular equivalence. In *Proceedings of the 12th ACM Conference on Recommender Systems*, RecSys '18, page 446–450, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450359016. doi:10.1145/3240323.3240404. URL https://doi.org/10.1145/3240323.3240404.

Antiopi Panteli and Basilis Boutsinas. Addressing the cold-start problem in recommender systems based on frequent patterns. *Algorithms*, 16(4), 2023. ISSN 1999-4893. doi:10.3390/a16040182. URL https://www.mdpi.com/1999-4893/16/4/182.

Suvash Sedhain, Aditya Menon, Scott Sanner, Lexing Xie, and Darius Braziunas. Low-rank linear cold-start recommendation from social data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1), Feb. 2017. doi:10.1609/aaai.v31i1.10758. URL https://ojs.aaai.org/index.php/AAAI/article/view/10758.

Nadia F. AL-Bakri and Sukaina Hassan. A proposed model to solve cold start problem using fuzzy user-based clustering. In *2019 2nd Scientific Conference of Computer Sciences (SCCS)*, pages 121–125, 2019. doi:10.1109/SCCS.2019.8852624.

Rudraprasad Mondal and Biswajit Bhowmik. Decs: A deep neural network framework for cold start problem in recommender systems. In *2022 IEEE Region 10 Symposium (TENSYMP)*, pages 1–6, 2022. doi:10.1109/TENSYMP54529.2022.9864409.

Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, and David M. Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '02, page 253–260, New York, NY, USA, 2002. Association for Computing Machinery. ISBN 1581135610. doi:10.1145/564376.564421. URL https://doi.org/10.1145/564376.564421.

Paolo Massa and Paolo Avesani. Trust-aware recommender systems. In *Proceedings of the 2007 ACM Conference on Recommender Systems*, RecSys '07, page 17–24, New York, NY, USA, 2007. Association for Computing Machinery. ISBN 9781595937308. doi:10.1145/1297231.1297235. URL https://doi.org/10.1145/1297231.1297235.

Hung-Hsuan Chen and Pu Chen. Differentiating regularization weights – a simple mechanism to alleviate cold start in recommender systems. *ACM Trans. Knowl. Discov. Data*, 13(1), January 2019. ISSN 1556-4681. doi:10.1145/3285954. URL https://doi.org/10.1145/3285954.

Hanxin Wang, Daichi Amagata, Takuya Maekawa, Takahiro Hara, Niu Hao, Kei Yonekawa, and Mori Kurokawa. A dnn-based cross-domain recommender system for alleviating cold-start problem in e-commerce. volume 1, pages 1–1, 01 2020. doi:10.1109/OJIES.2020.3012627.

F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 5(4):19:1–19:19, December 2015. doi:10.1145/2827872. URL https://doi.org/10.1145/2827872.

Yehuda Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. *TKDD*, 4, 01 2010. doi:10.1145/1644873.