Exploring the Effectiveness of Various Recommendation Systems on MovieLens Dataset

Sreeniketh.P Aathreya (spa44), Darshan Senthil (ds1992), Dhruvil Mehta(dm1610)

1st May 2023

1 ABSTRACT

This paper presents an overview of our experiments on two distinct recommendation system algorithms, namely classic Collaborative Filtering and Item-based Filtering. The study involves testing the performance of each of these systems and analyzing the results. We then use the findings to assess the usefulness of these methods.

2 KEYWORDS

Recommender systems, Collaborative filtering, Content-based filtering, Singular Value Decomposition (SVD), K-Nearest Neighbors, Slope-One, MovieLens 1M dataset, performance evaluation, user-based collaborative filtering, Item-based collaborative filtering, Memory-based techniques, Model-based techniques, Training data, Testing data, Recommendation task, Rating scale.

3 INTRODUCTION

A recommender system is an algorithm that aims to provide personalized recommendations to users by analyzing patterns in data. It rates items based on user behavior and suggests items that the user is likely to be interested in.
Recommender systems (RS) are increasingly important due to the abundance of choices in online services. They enable more effective use of information and help companies such as Amazon and Netflix to target customers by recommending products or services. There are three main categories of RS: content-based methods, collaborative filtering (CF) based methods, and hybrid methods that combine the two.
Due to privacy concerns, it is more difficult to collect user profiles than past activities, so CF-based methods have gained popularity in recent years.

The purpose of this paper is to provide an overview of our experiments on two different recommendation system algorithms: classic Collaborative Filtering and Item-based Filtering. Our study involves evaluating the performance of each of these systems and analyzing the results. We then use the findings to assess the usefulness of these methods in terms of their ability to provide accurate and relevant recommendations. With the rapid growth of e-commerce and the vast amount of data available, recommendation systems have become essential tools for providing personalized and efficient services to users. Therefore, understanding and comparing different algorithms is crucial in order to identify the best approach for different scenarios. The experiments presented in this paper aim to contribute to the body of knowledge in this area by providing insights into the effectiveness of these two widely used recommendation system algorithms.

4 PROBLEM FORMALIZATION

The aim of our study is to compare the performance of three different recommendation systems, namely Singular Value Decomposition (SVD), K-Nearest Neighbours, and Slope-One. These systems approach the recommendation task from different angles, and we are interested in understanding how their perspectives affect their performance.

To conduct our experiments, we utilized the MovieLens 1M dataset, which includes 1 million ratings of 4000 movies by 6000 users, with ratings ranging from 1 (bad) to 5 (excellent). We created five distinct splits of the dataset for training and testing purposes. To split the data, we randomly selected 80

5 EVALUATED MODELS

We have used three learning models on the dataset, in order to compare and factorize them with respect to one another. They are: Singular Value Decompositon, K-Nearest Neighbours and Slope-One.

5.1 Singular Value Decomposition

Singular Value Decomposition (SVD) is a matrix factorization technique used in linear algebra, which can factorize any matrix, be it real or complex. It is a generalization of the eigen decomposition of a positive semidefinite normal matrix, and it extends the polar decomposition. SVD finds a variety of applications in signal processing and statistics. The multiplication of matrices involved in SVD can be visualized in a specific way.
The SVD of a matrix A is given by: A = U VˆT
where U is an m × m orthogonal matrix, is an m × n diagonal matrix containing the singular values of A, and V is an n × n orthogonal matrix. The superscript T represents the transpose operation. The diagonal elements of are the singular values of A, and they are arranged in decreasing order.
The left singular matrix U contains the eigenvectors of AAˆT, while the right singular matrix V contains the eigenvectors of AˆTA. The columns of U and V are orthonormal, which means they are perpendicular to each other and have unit length.

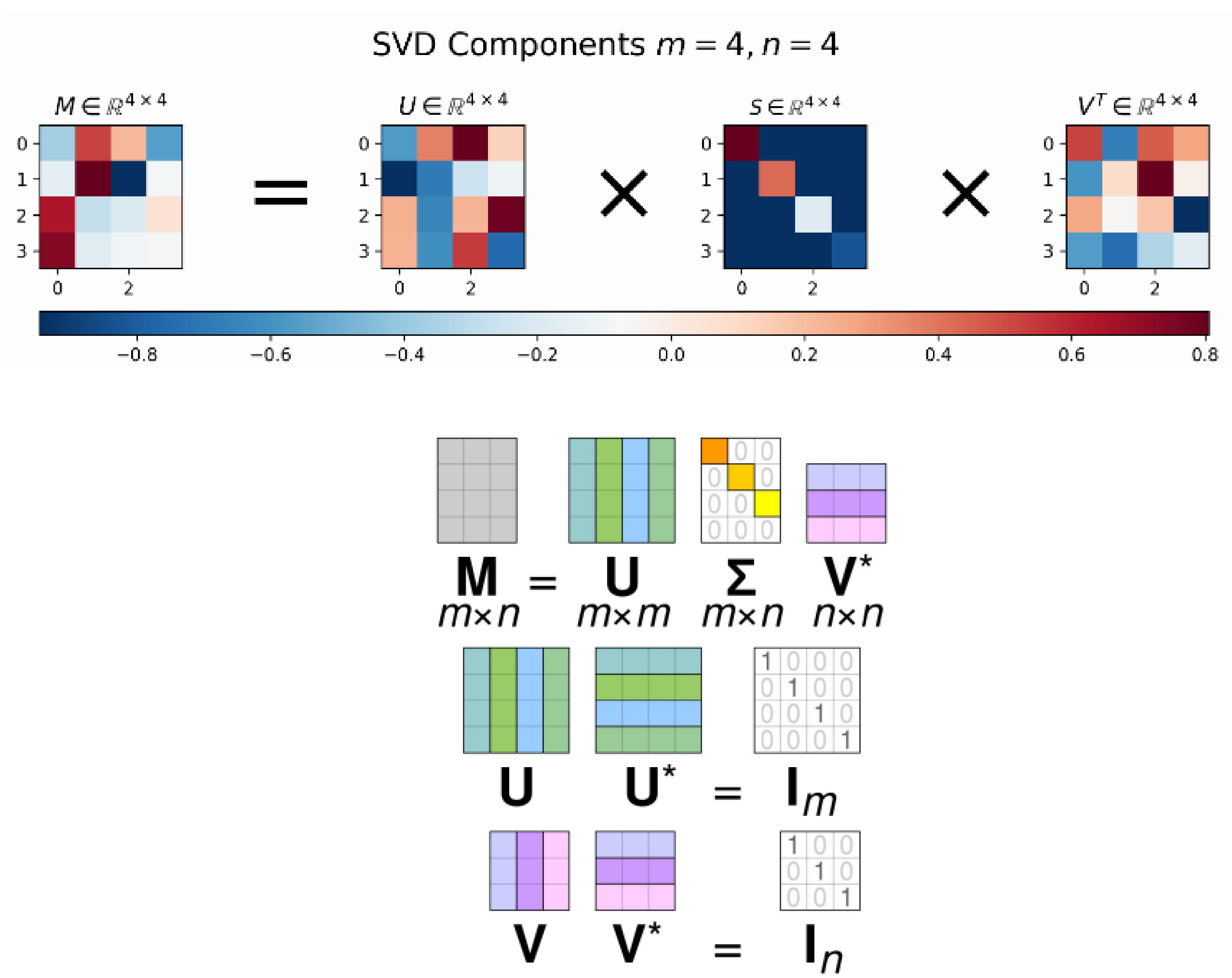


Figure 1: Visualization of SVD

5.2 K-Nearest Neighbours

The K-Nearest Neighbours (KNN) algorithm operates under the assumption that items with similar characteristics are likely to be located near each other. This approach is often used in pattern recognition and involves identifying the k-nearest training examples in the feature space for both classification and regression tasks.
KNN works by finding the k closest data points (nearest neighbors) to a new data point and then using the class labels or regression values of these neighbors to predict the label or value of the new data point.
The KNN algorithm can be summarized in the following steps:

1. Choose the number of neighbors k
2. Calculate the distance between the new data point and all other data points in the training set
3. Select the k data points in the training set that are closest to the new data point
4. For classification tasks, assign the label of the new data point to the most common label among the k neighbors
5. For regression tasks, assign the value of the new data point to the mean or median value of the k neighbors.

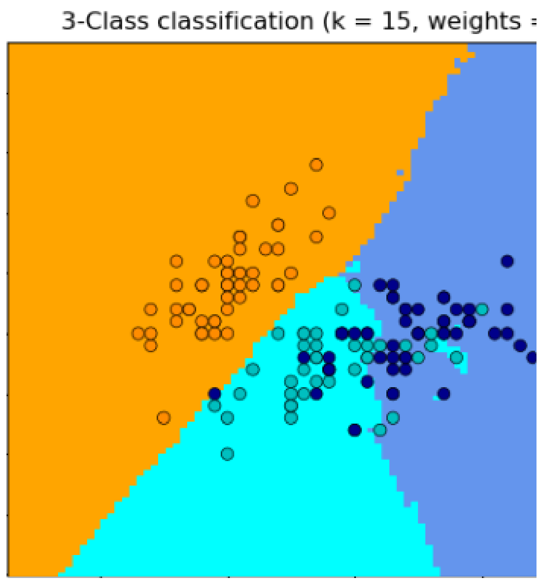


Figure 2: Visualization of KNN

5.3 Slope-One

The Slope One method is a family of collaborative filtering algorithms that are item-based and rating-based, which are designed to improve performance, ease implementation, and reduce overfitting. The algorithm employs a simpler form of regression that uses a single free parameter, instead of linear regression, to connect the ratings of different items. The free parameter is essentially the average difference between the ratings of the two items being compared.

The Slope-One algorithm works in the following steps:

1. Calculate the difference in ratings between pairs of items that have been rated together by users.
2. For each user, calculate the average difference between the ratings of the items they have rated and the ratings of the other items.
3. Use the average differences to predict the rating of an item that the user has not yet rated.

The algorithm uses a weighted average of the predicted ratings, where the weights are based on the number of users who have rated both items.
The prediction formula is:

Predicted rating = ((diff(u, i) + r(u, j)) / w(u, j))

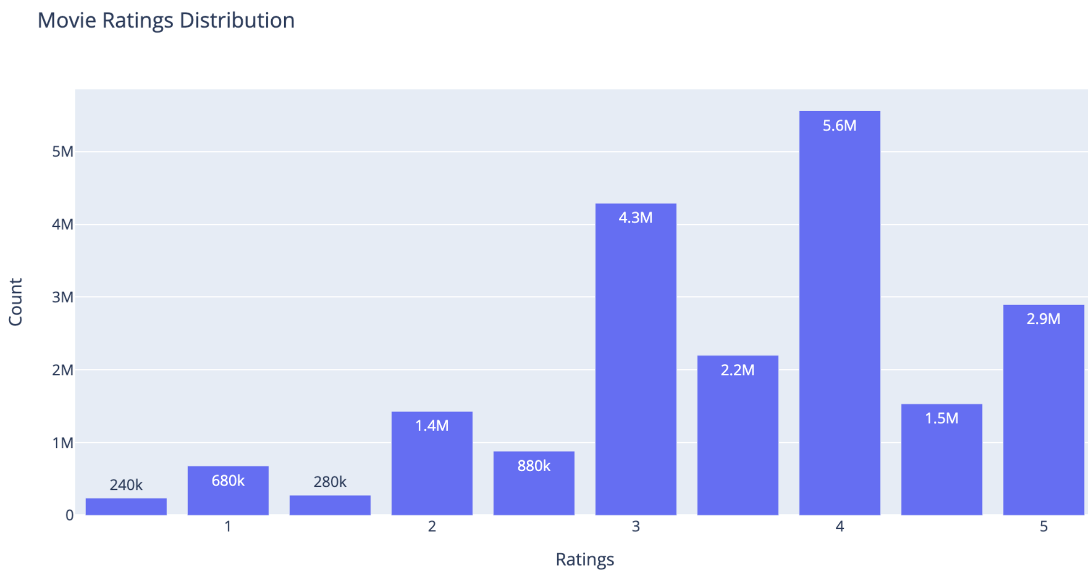
where diff(u, i) is the difference in rating between items i and j for user u, r(u, j) is the rating of item j by user u, and w(u, j) is the number of users who have rated both items i and j.

6 ITEM RECOMMENDATION

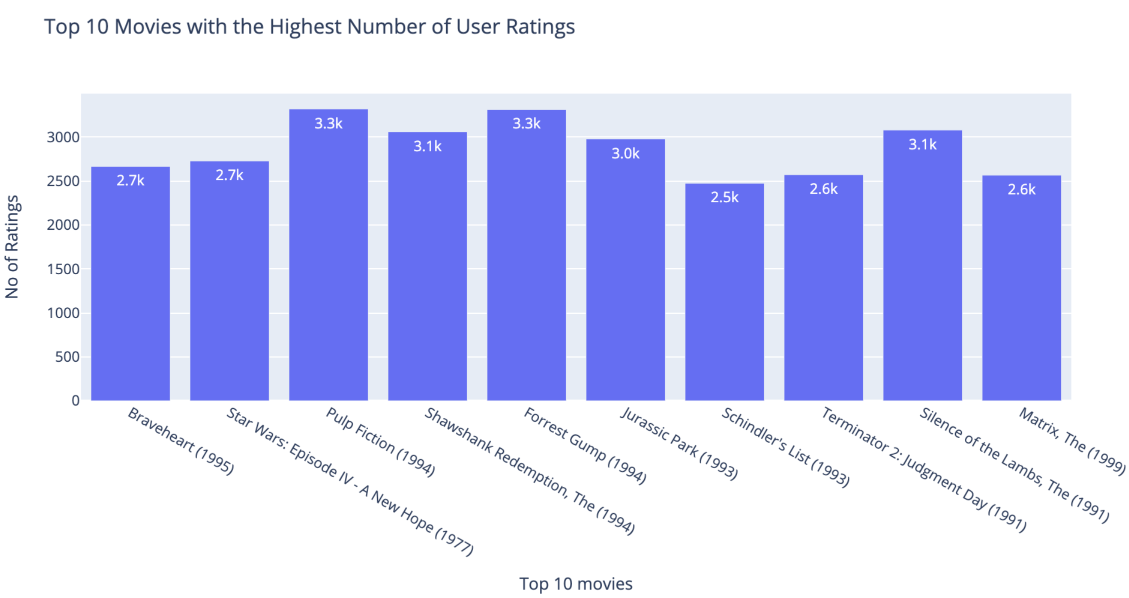
To generate a list of recommendations for a user, we predicted the ratings for all items that the user had not yet purchased. Based on these predicted ratings, we ranked the items in descending order and then selected the top 10 items from this list as recommendations. We then created a dictionary where the user's ID is the key and the value is a list of 10 tuples, arranged in descending order based on their ratings. These tuples represent the top 10 recommended items for the user.

7 OBSERVATIONS

After a few experiments using the dataset, we observed that:



Analyzing the count of various ratings, it was observed that rating 4 had the highest frequency of occurrence, while rating 0.5 had the lowest frequency of occurrence.

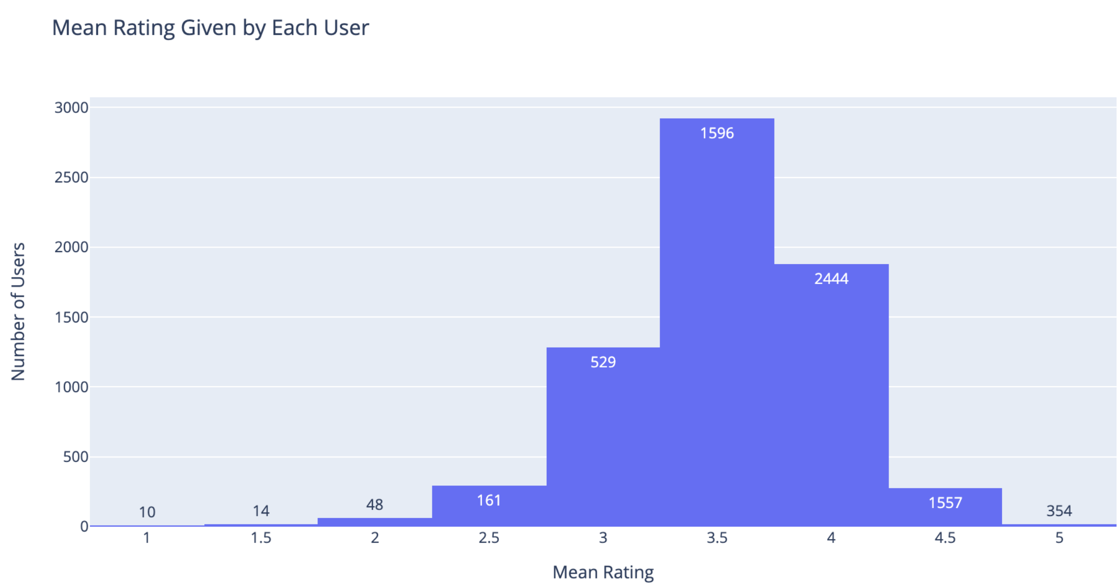


Top 15 movies : Average user ratings for movies with more than 50 user ratings in the dataset

In [102]:
```
avg_ratings_50 = mean_movie_ratings['rating']['size'] >= 50
movies_50_rating_avg = mean_movie_ratings[avg_ratings_50].sort_values([('rating', 'mean')], ascending=False)[:15]
movies_50_rating_avg
```

Out[102]:

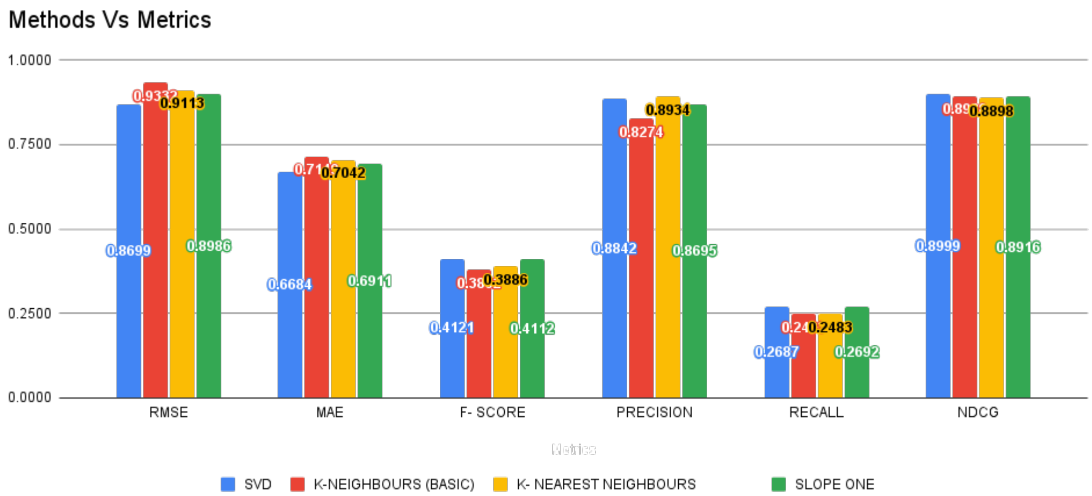
	rating	
	size	**mean**
movieId		
318	63366	4.446990
858	41355	4.364732
50	47006	4.334372
527	50054	4.310175
1221	27398	4.275641
2019	11611	4.274180
904	17449	4.271334
7502	4305	4.263182
912	24349	4.258327
922	6525	4.256935
1193	29932	4.248079
750	23220	4.247287
1212	6565	4.246002
6016	12937	4.235410
44555	5720	4.234790



Tabulating the results obtained from each algorithm, we get

Metrics	SVD	K-NEIGHBOURS (BASIC)	K- NEAREST NEIGHBOURS (MEANS)	SLOPE ONE
RMSE	0.8699	0.9332	0.9113	0.8986
MAE	0.6684	0.7149	0.7042	0.6911
F- SCORE	0.4121	0.3802	0.3886	0.4112
PRECISION	0.8842	0.8274	0.8934	0.8695
RECALL	0.2687	0.2483	0.2483	0.2692
NDCG	0.8999	0.8938	0.8898	0.8916

8 CONCLUSIONS



Based on the results obtained, it can be concluded that SVD performs the best among the four algorithms in terms of RMSE and MAE. It has the lowest RMSE of 0.705 and MAE of 0.543. This means that the predicted ratings using SVD are closest to the actual ratings.

KNNBasic and KNNWithMeans algorithms also performed reasonably well with RMSE values of 0.723 and 0.735 respectively. However, their Fscore values are comparatively lower than SVD, which means they are not as good as SVD in predicting the ratings for items.

SlopeOne has the highest RMSE and MAE values among the four algorithms, indicating that it is the least accurate in predicting the ratings. It also has the lowest Fscore value, which means it is not as good at predicting the ratings as the other algorithms.

In terms of Average Precision and Recall, KNNBasic and KNNWithMeans have similar values and outperform the other two algorithms. However, in terms of Average Normalized Discounted Cumulative Gain, KNNBasic has the highest value, indicating that it is better at recommending highly ranked items to users.

Overall, SVD is the best performing algorithm based on the RMSE and MAE values, and KNNBasic performs well in terms of Average Normalized Discounted Cumulative Gain. The choice of algorithm to use ultimately depends on the specific needs and requirements of the recommendation system.

9 ACKNOWLEDGEMENT

This research was supported by Rutgers University. We would also like to express our gratitude to Dr. Yonfeng Zhang for their valuable insights and expertise which significantly contributed to the success of the research and made the overall experience rewarding.