Final Report for Movie Recommender System
PMLDL course Innopolis University Fall2023

by Said Kamalov BS21 DS01
s.kamalov@innopolis.university

Introduction

In this project I tried to develop a recommendation system to solve movie recommendation problem based on *MovieLens 100k dataset* [1].

First, I started to explore the information about recommendation systems, their types and ideas behind them. This habr article [2] provides a good definition of recommendation systems and clearly describes their goals. Moreover, this article lists main types of such systems and briefly describes them.

Second, I examined and preprocessed the data to deeply understand the structure of the provided dataset. Then I conducted Exploratory Data Analysis to examine some trends, flows and patterns in the data, especially in ratings distribution. For this stage of the project I referenced these sources: [3], [4].

Third, I chose and implemented the matrix factorization approach based on SVD. I made this decision based on performance evaluation of this approach provided in [5] and also because it is relatively simple to implement with use of the surprise [6] python library. As a benchmark I chose the following metrics: Root Mean Square Error, Precision@k, Recall@k. This decision was based on [4] and [5].

Finally, I evaluated the fitted model and implemented an algorithm for movie suggestions based on the model results.

Data analysis

Let us dive deeper into the data analysis part of my project. At first I preprocessed the given data, for example I removed movies without title and genres, and processed null values.
Then I tried to visualize the information about ratings and its distribution among users and movies. I have concentrated on this specific part to explore, because this particular information is the most important for the approach I have chosen. As I decided to implement the Collaborative Filtering approach, I was only interested in the history of ratings.

Model Implementation

For the implementation of Collaborative Filtering system based on SVD decomposition I used the surprise [6] python library as it provides simple to use and optimized implementations.
The idea behind this model is briefly described in [2]. As a result, the model provides a matrix where rows represent users id and columns represent movies id, and [i,j] element stands for estimated rating from user_i for movie_j. And roughly speaking, the idea is to minimize the difference between matrix with real ratings and matrix with estimated ones.
To get movies recommendations for a specific user, my implemented algorithm chooses unseen movies with the highest estimated ratings for this user.

Model Advantages and Disadvantages

Advantages:

- relatively efficient with respect to chosen metrics
- captures underlying patterns in user-to-item relations
- can be optimized to work with highly sparse data

Disadvantages:

- suffers from cold start problem
- does not consider information about items and users themselves

Training Process

The model takes as input the table which should have user ids, movie ids and a rating for every pair of user and movie.
To find best parameters for fitting the SVD the grid search can be used.
The training is done for n epochs and uses gradient descent as an optimizer.

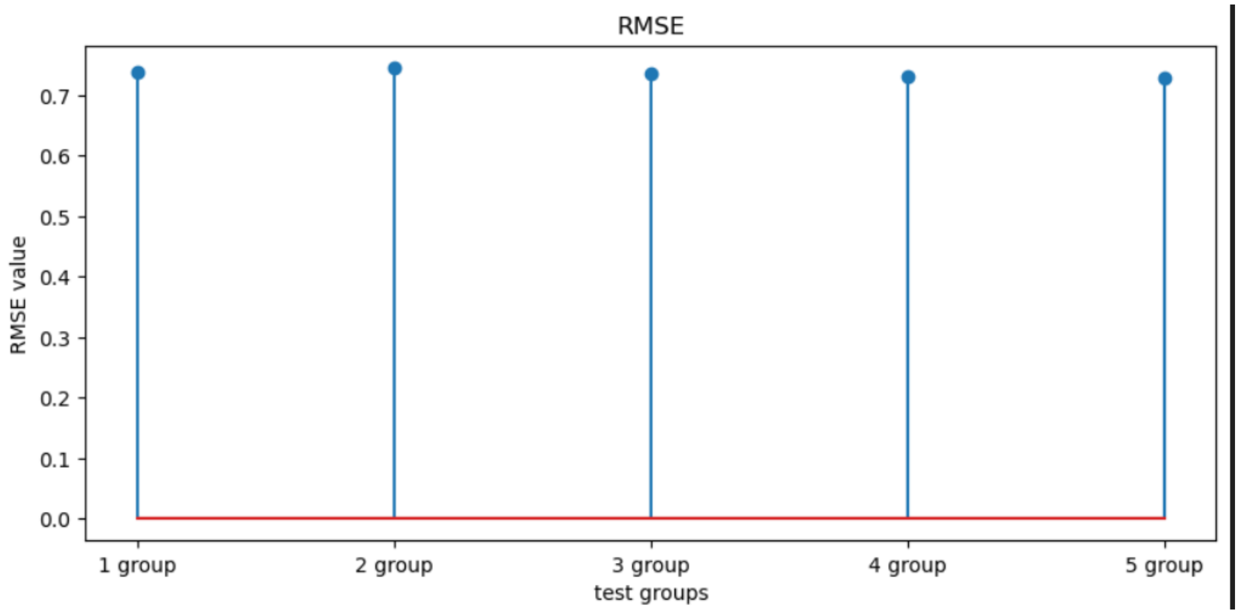
Evaluation

As a benchmark I have chosen 3 metrics:

- Root Mean Square Error
- Precision@k
- Recall@k.

At the evaluation stage I use 5 disjoint test datasets to get results of chosen metrics. Then I use stem plots to visualize and compare obtained results

RMSE plot for 5 test groups with k = num_of_dimensions = 100:



It is interesting, that I managed to get better results on the same dataset that was shown in [5]:

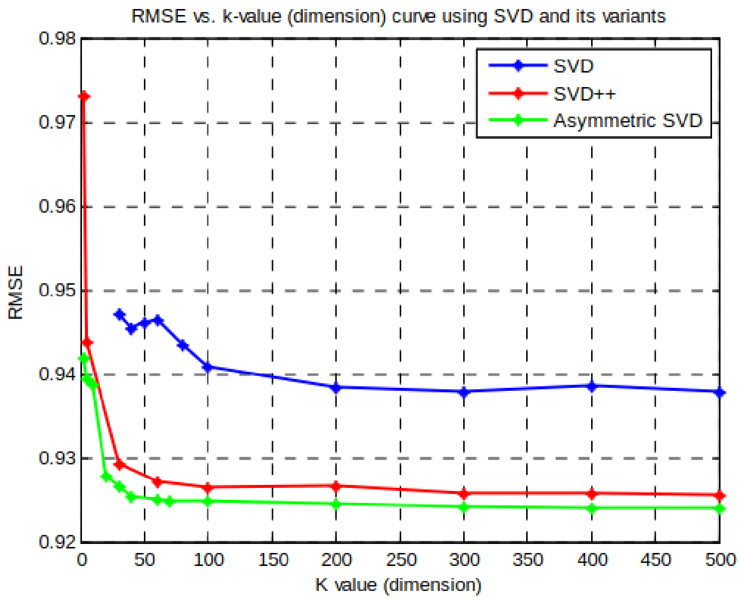


Fig. 4 RMSE curve based on different k-values(dimension) using SVD and its variants

Results

As a result, I provided data exploration of famous benchmark dataset MovieLens 100k dataset. Especially, I tried to focus on user-to-movie relations and rating distributions. Also I provided my implementation of Collaborative Filtering approach based on matrix factorization, strictly speaking, on SVD. Finally I evaluated the results with the use of three common metrics for such types of problems. And implemented the movie suggestions system based on the results of the user-to-item matrix obtained from SVD.

And last, but not least, I have significantly expanded my knowledge in the domain of recommendation systems and corresponding problems.

References:

[1] MovieLens 100k dataset
(https://grouplens.org/datasets/movielens/100k/)
[2] “People meet recommender systems. Factorization”
(https://habr.com/ru/articles/486802/)
[3] movielens-100k-data-analysis notebook
(https://www.kaggle.com/code/yoghurtpatil/movielens-100k-data-analysis)
[4] Recommendation Systems with TensorFlow notebook
(https://colab.research.google.com/github/google/eng-edu/blob/main/ml/recommendation-systems/recommendation-systems.ipynb?utm_source=ss-recommendation-systems&utm_campaign=colab-external&utm_medium=referral&utm_content=recommendation-systems#scrollTo=k0IFBGCx8_im)
[5] MOVIE RATING ESTIMATION AND RECOMMENDATION
(https://cs229.stanford.edu/proj2012/BaoXia-MovieRatingEstimationAndRecommendation_FinalWriteup.pdf)
[6] Surprise python library
(https://surpriselib.com/)