RECOMMENDER SYSTEMS

By: Pranav Dev Kottimukkalur Ramasubramanian

Basic Outline of the data

There are totally 6040 users and 3706 unique movies.

The data is available in .dat format and is imported to 'Jupyter Notebook' using read_csv from Pandas library and with encoding='ISO-8859-1' for better readability of text data.

Based on these data, we will be looking into different recommendation systems including User-based Collaborative Filtering, Item-based filtering with two methods in that. Then a better recommendation system is proposed and followed by MovieAvg based recommendation system.

Metrics used:

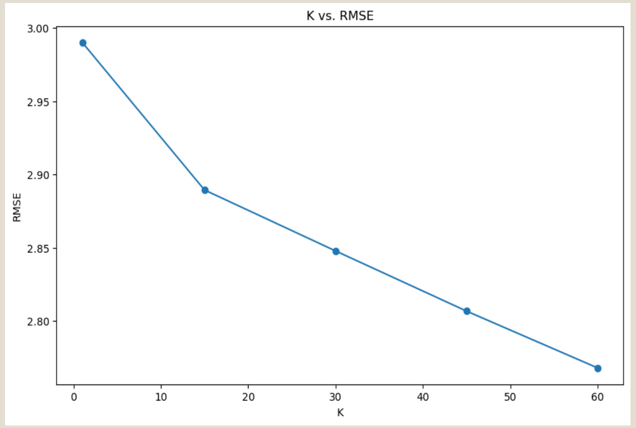
Average Precision (AP): AP is used to assess the quality of ranked recommendation lists. It quantifies how well relevant items are positioned in the list, providing insights into the precision of recommendations and their order of presentation.

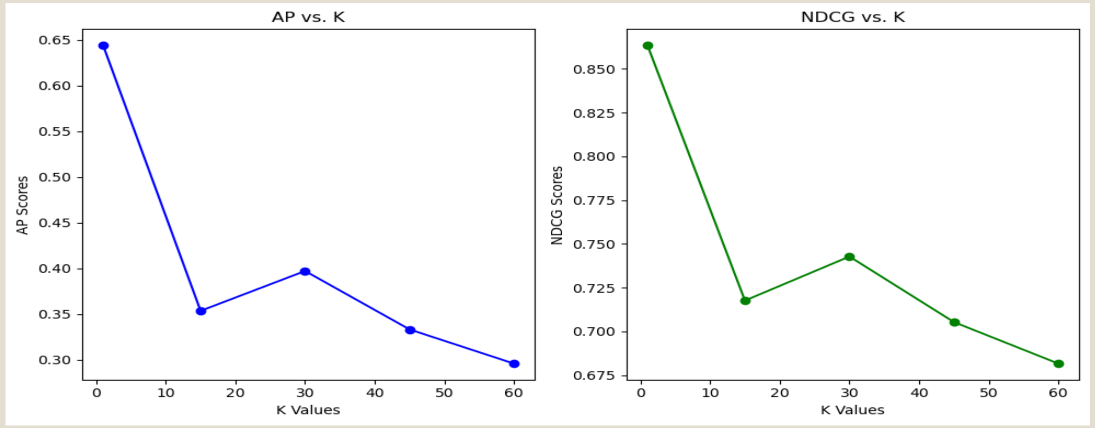
Normalized Discounted Cumulative Gain (nDCG): nDCG evaluates the effectiveness of ranked recommendations by considering both relevance and ranking position. It provides a normalized score to measure the quality of recommendations and their alignment with user preferences.

Root Mean Square Error (RMSE): RMSE is commonly used to evaluate the accuracy of predicted ratings in recommendation systems. It quantifies the difference between predicted and actual ratings, helping to assess the system's predictive performance.

Finally, all these models are compared using suitable parameters and the best model is chosen.

KNNCF:





Here user based collaborative filtering is used to predict the rating of the movies for a user. Here, user with id 1 has been chosen. The values of K have been tuned as 1,15,30,45 and 60. RMSE for all these have been found and is displayed in the first graph. From the graph, the model becomes effective (reduced RMSE) with increase in K.
K = 60 is best with RMSE 2.77 .
AP and NDCG for each value of K for user 1 has been displayed in the second graph.

Overall AP , NDCG and RMSE for the model has been displayed below:

AP	NDCG	RMSE
0.405	0.742	1.02

Item Based CF:

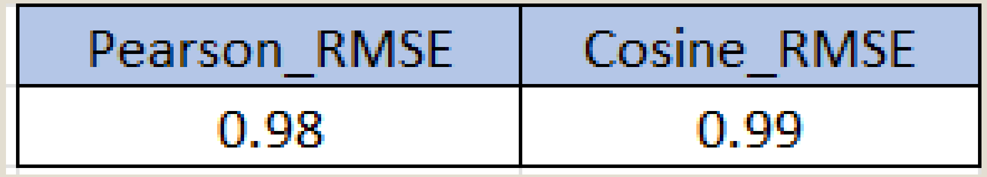
For item(movie) based collaborative filtering, two approaches were chosen. First is the Pearson Correlation and second is the Cosine Similarity.

Pearson correlation system calculates the correlation between a target movie and other movies then selects similar movies and predicts a user's ratings on the selected movie. This is a also shown with the help of a correlation matrix.

Cosine similarity calculates the movie-movie cosine similarity to find the top K most similar movies to a chosen movie, it then calculates weighted sum and differences from mean and then normalises it to give predicted ratings.

Here, a random movie is chosen for Pearson and same movie is chosen for Cosine for effective comparison.

The RMSEs have been displayed below.



Pearson_RMSE	Cosine_RMSE
0.98	0.99

On comparing, Peasron Correlation seems to perform better in predicting ratings of a movie.

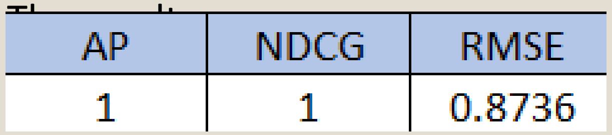
Option1RecSys:

Of the many recommendation systems, matrix factorization has been chosen as the better one here. It can also handle sparse and incomplete data. This can fill in the missed ratings or unrated items by users. It reduces the dimensionality of the data for efficiency.

This model can give personalised recommendations and sparsity handling.

The model recommends top 10 movies to the selected user and also displays their predcited ratings.

Single Value Decomposition is a specific type of matrix factorization that breaks down a matrix into components known as ‘singular values – S ’ and ‘vectors - V’. SVD decomposes the matrix into U (first matrix), S and V and predicts the values. The functionality is inbuilt in the SVD() from surprise library in Jupyter Notebook and does all these internally.



AP	NDCG	RMSE
1	1	0.8736

Single Value Decomposition is a specific type of matrix factorization that breaks down a matrix into components known as ‘singular values – S ’ and ‘vectors - V’. SVD decomposes the matrix into U (first matrix), S and V and predicts the values. The functionality is inbuilt in the SVD() from surprise library in Jupyter Notebook and does all these internally.

These finding have been referred from:

- M. Sunitha Reddy and T. Adilakshmi, ‘Music recommendation system based on matrix factorization technique -SVD’, in *2014 International Conference on Computer Communication and Informatics*, Jan. 2014, pp. 1–6. doi: 10.1109/ICCCI.2014.6921744.

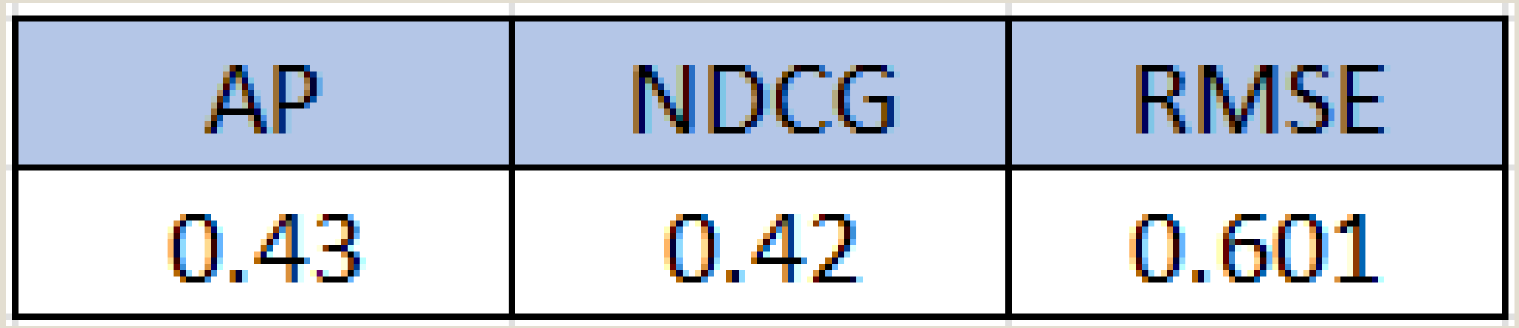
- R. Barathy and P. Chitra, ‘Applying Matrix Factorization In Collaborative Filtering Recommender Systems’, in *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Mar. 2020, pp. 635–639. doi: 10.1109/ICACCS48705.2020.9074227.

- A. Mnih and R. R. Salakhutdinov, ‘Probabilistic Matrix Factorization’, in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2007. Accessed: Oct. 28, 2023. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2007/hash/d7322ed717dedf1eb4e6e52a37ea7bcd-Abstract.html

MovieAvg:

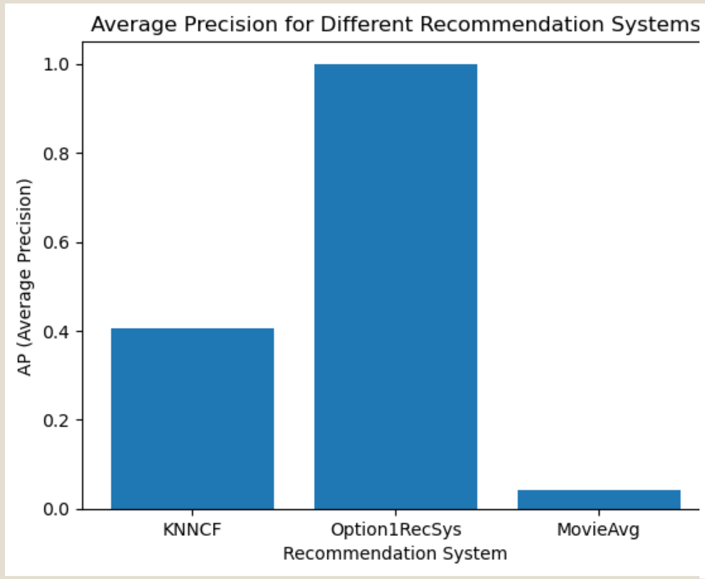
5 random users are selected who have rated more than 100 movies and top 30 movies for each user was predicted using Movie Average based on the mean rating of the movies.
Pretty simple and straight forward approach and does not require any complex algorithm
This successfully recommended movies to the users based on the top average ratings.

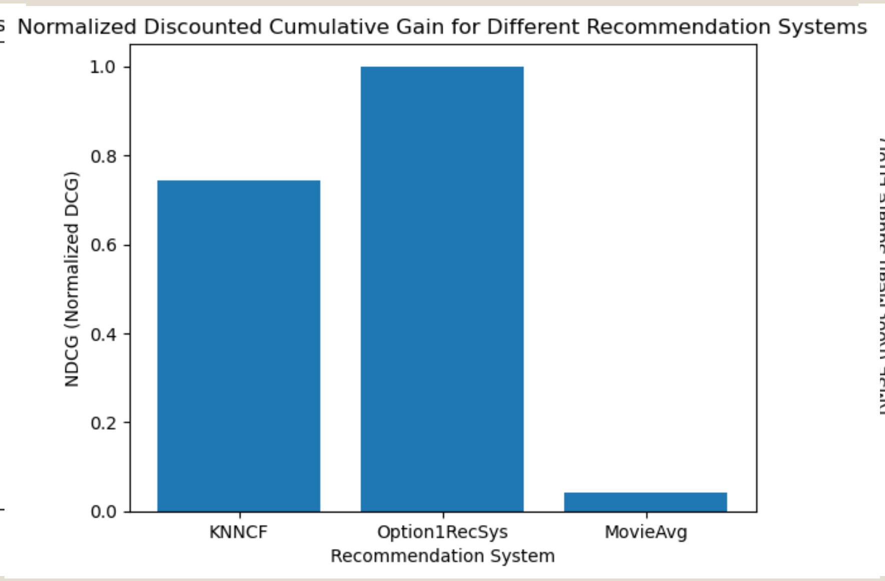
The metrics evaluated are listed below:

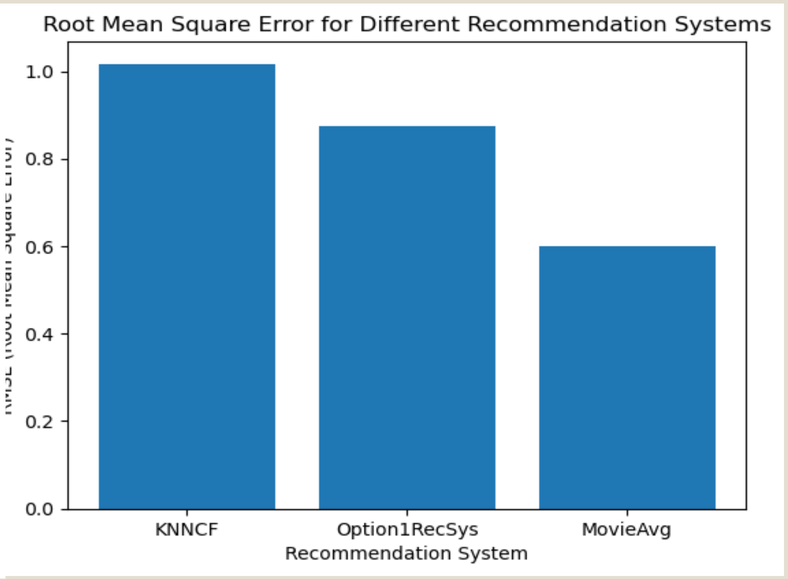


AP	NDCG	RMSE
0.43	0.42	0.601

Comparisons

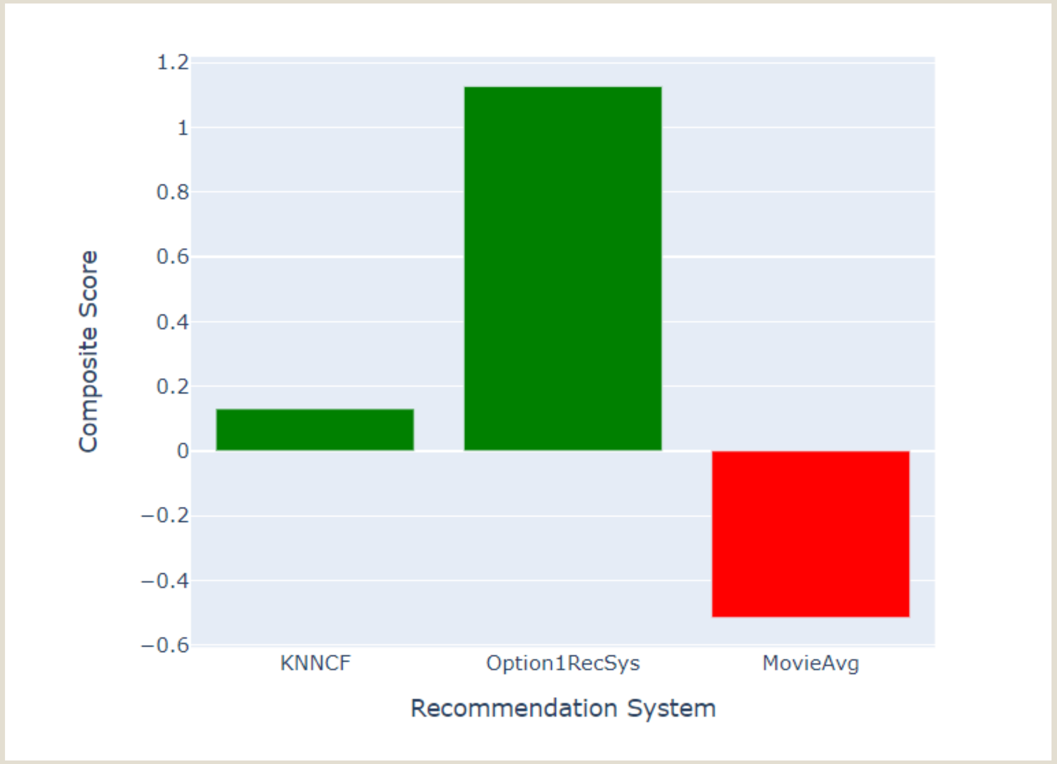






Inference: Regarding AP, MovieAvg is the poorest and SVD performs the best. The same scenario is observed with NDCG also. But in terms of RMSE, MovieAvg performs a bit better than MovieAvg.

The AP and NDCG for Option1RecSys is the highest
So which is the best recoemmender system? Lets see the champion in the next slide.



To check which model is the best to use, a composite value represented by computing all the parameters is defined. It is calculated as [(AP + NDCG) - RMSE]. By comparing the three models. Option1RecSys model (SVD based Matrix factorization model) appears to be the better recommendation system of these all with a score of 1.13 and KNNCF with 0.13 and MovieAvg with -0.5.

Therefore Option1RecSys that is SVD Matrix Factorization works the best out of all these models.

References:

- Task 1 and Task 2 referred from Practical Data Science with Python COSC2670 lab taught and provided materials by tutor.
- All tasks referred from the materials provided in Practical Data Science with Python COSC2670
- uniQin.ai, ‘Cosine Similarity for Item Based Collaborative Filtering’. Available: https://blog.uniqin.ai/p/cosine-similarity-for-item-based. [Accessed: Oct. 28, 2023
- H. Bhowmick, A. Chatterjee, and J. Sen, ‘Comprehensive Movie Recommendation System’. arXiv, Dec. 23, 2021. doi: 10.48550/arXiv.2112.12463. Available: http://arxiv.org/abs/2112.12463. [Accessed: Oct. 28, 2023]
- ‘Getting Started — Surprise 1 documentation’. Available: https://surprise.readthedocs.io/en/stable/getting_started.html. [Accessed: Oct. 28, 2023]
- Y. Koren, R. Bell, and C. Volinsky, ‘Matrix Factorization Techniques for Recommender Systems’, *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009, doi: 10.1109/MC.2009.263. Available: https://ieeexplore.ieee.org/abstract/document/5197422. [Accessed: Oct. 28, 2023]
- A. A. Yeung, ‘Matrix Factorization: A Simple Tutorial and Implementation in Python @ quuxlabs’. Available: http://www.quuxlabs.com/blog/2010/09/matrix-factorization-a-simple-tutorial-and-implementation-in-python/. [Accessed: Oct. 28, 2023]
- F. Lazzeri, ‘How to Auto-Train Your Machine Learning Model’, *Microsoft Azure*, Sep. 03, 2019. Available: https://medium.com/microsoftazure/how-to-auto-train-your-machine-learning-model-d10ab806d3fb. [Accessed: Oct. 28, 2023]
- R. Python, ‘Build a Recommendation Engine With Collaborative Filtering – Real Python’. Available: https://realpython.com/build-recommendation-engine-collaborative-filtering/. [Accessed: Oct. 28, 2023]