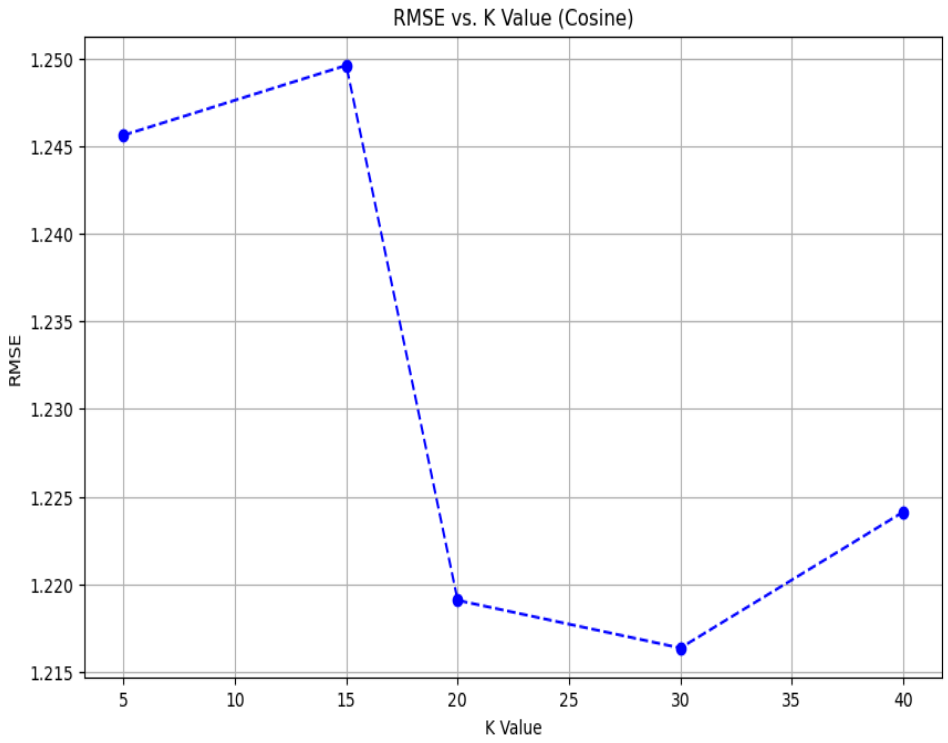
Assignment 3: Recommender Systems (Virtual Presentation)

Bharath Narayanan Venkatesh

S4033348

Task 1: User-based Collaborative Filtering



I am importing the necessary libraries
I am reading movies and ratings data file using pd.read_csv
user_item_matrix = ratings.pivot(index='UserID', columns='MovieID', values='Rating') - I am creating a user-item matrix with
index='UserID', columns='MovieID', values='Rating'. We are creating a matrix-like structure and then placing the values into the
user_item_matrix

Calculating user similarities using Cosine Similarity

user_item_matrix_filled = user_item_matrix.fillna(0)
user_similarity_cosine = cosine_similarity(user_item_matrix_filled)

The user_item_matrix likely contains NaN (empty or missing) values where users haven't rated certain movies.
In this line of code, these NaN values are replaced with zeros using the fillna method.
After filling in the missing values, the cosine_similarity function is applied to the filled user-item matrix

Displaying Actual vs Predicted ratings for the movies the user has rated:-
Comparison between actual ratings and predicted ratings for movies that a specific user has already rated.
We are using the previously defined predict_ratings_mean_centered_cosine function, which predicts ratings for a user's unrated movies based on cosine similarity among users.

We are then calculating the Impact of different K values on RMSE and we are ploting it.

I have set k_values = [5, 15, 20, 30, 40]

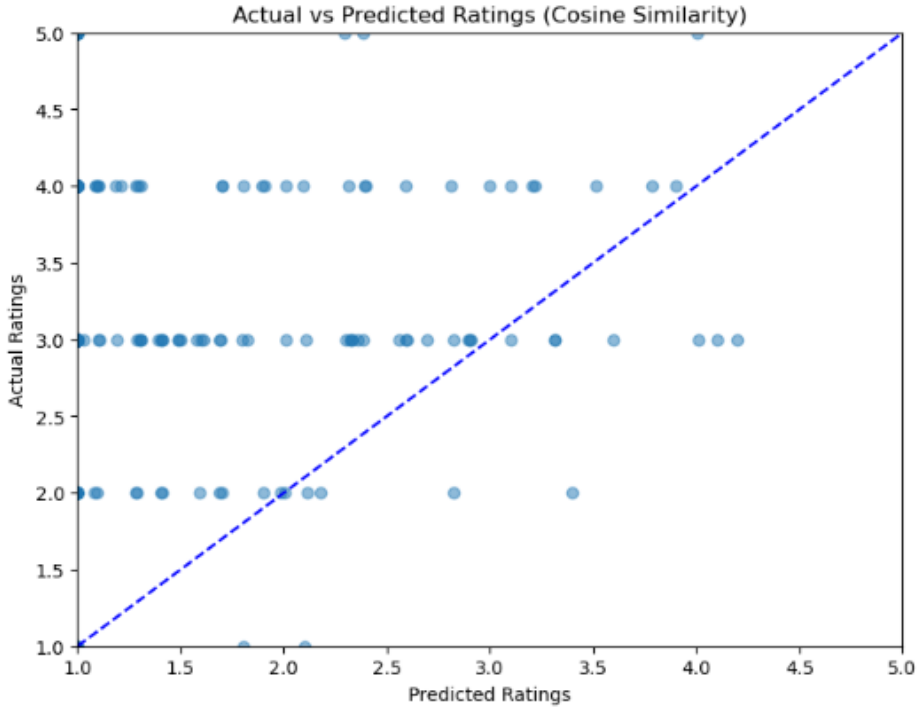
Why we have values close to 1?

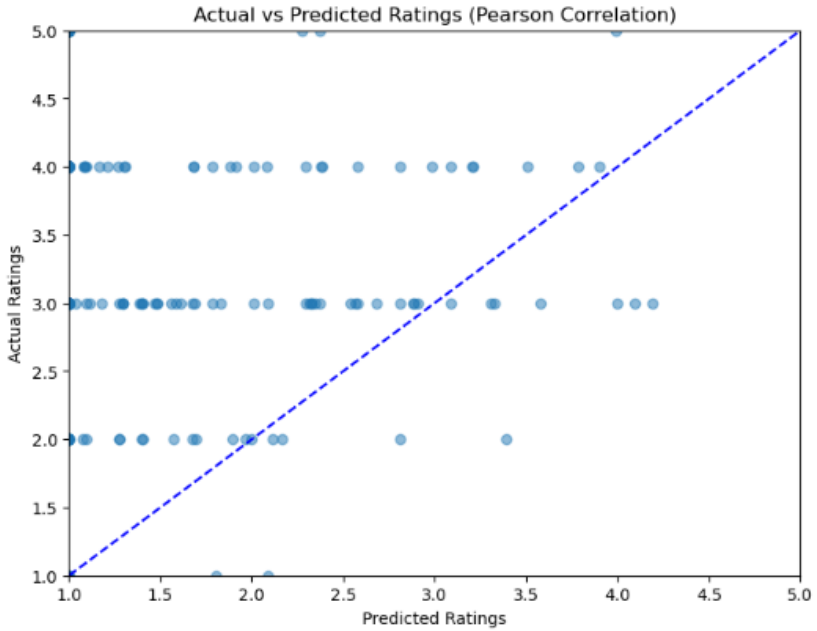
This is because, we have

1.Less number of movies rated by users,
2.Similarities is less when compared with other users,
3.The users who have rated those movies have rated less for these movies.

```
I've randomly selected the movie with MovieID: 1892
I've randomly selected the movie name: Perfect Murder, A (1998)
RMSE for cosine similarity: 2.385
RMSE for Pearson correlation: 2.386
   UserID  Actual Ratings  Predicted Ratings (Cosine)  \
0      15             4.0                    1.000000
1      33             3.0                    1.000000
2      48             3.0                    1.407862
3      60             3.0                    1.000000
4      82             4.0                    1.000000
5      90             4.0                    1.000000
6     123             3.0                    1.109604
7     127             3.0                    1.000000
8     133             4.0                    1.000000
9     156             5.0                    1.000000

   Predicted Ratings (Pearson)
0                     1.000000
1                     1.000000
2                     1.400425
3                     1.000000
4                     1.000000
5                     1.000000
6                     1.114595
7                     1.000000
8                     1.000000
9                     1.000000
```





Task 2:- Item-based Filtering

I am reading files from ratings dataset

We are randomly selecting a movie from the 'user_item_matrix' columns.

The predicted ratings are calculated using the similar movies' similarity values and their ratings to compute a weighted average as predictions for the movie.
We are calculating Root Mean Squared Error (RMSE) between the actual ratings of the randomly chosen movie and the predicted ratings obtained using both cosine similarity and Pearson correlation.
predicted_ratings_df :- Collects the user IDs, actual ratings, and predicted ratings for the randomly chosen movie and displays the first 10 entries.

From the images, it is clear that the Pearson and Cosine ratings do not differ significantly, and the ratings are quite low, indicating excellent prediction accuracy.

Task 3: A Better Recommender System

Option1RecSys: Description of the solution

The ratings data are converted into a user-item matrix, with rows denoting users, columns representing movies, and cells representing user ratings for movies. Null values are used to fill in missing data (unrated movies).

SVD, or singular value decomposition:
The user-item matrix is split up into three matrices using SVD: U, sigma, and Vt. U stands for user features, sigma is a diagonal matrix with singular values, and Vt is for item features.

Rebuilding and Projected Ratings:
To generate predicted ratings for all users and items, the original user-item matrix is reconstructed using the U, sigma, and Vt matrices that were produced by SVD.
Obtain Prognostic Film Suggestions:
The top N movies for a given user are suggested by the get_predicted_recommendations function based on their predicted ratings. It sorts the movies in descending order of predicted ratings and returns the top N movies as recommendations.

Literature Review

Title: Advancements in Recommender Systems: Mitigating Sparsity and Enhancing Recommendations Introduction. The text highlights the growing interest in this field, driven by both industry and academic endeavors, aiming to address challenges in handling data sparsity and the cold start problem.

Main Research Contributions: Utilizing Additional Information for SVD-Based Recommendations: The text emphasizes the importance of enhancing SVD-based recommendations by incorporating additional data to alleviate data sparsity issues. It notes the demonstrated improvement in recommendations due to the incorporation of supplementary information.

Hybrid Recommender Systems Leveraging Auxiliary Information: Addressing data sparsity and the cold start problem, the hybrid system integrates textual descriptions and images in collaborative filtering, proving to be an effective approach for online recommendations.
Incorporating Semantic Attitudes and Temporal Alterations: A three-dimensional matrix factorization method is introduced, incorporating semantic attitudes (sentiment, volume, and objectivity) extracted from user-generated content. Additionally, temporal alterations are considered in the factorization model.

Addressing Cold Start Problem through User and Item Descriptions: Dual Regularized Matrix Factorization (DRMF) with a deep neural network is employed to effectively extract user and item description documents, mitigating the issue of data sparsity.

Data Sparsity in SVD-Based Recommender Systems: The text introduces the problem of data sparsity in Collaborative Filtering (CF) and highlights the use of Singular Value Decomposition (SVD) to improve the quality of predictions.

Techniques involve imputed data creation and incorporation into the SVD framework. Proposed Method and Experimentation: The text elaborates on a proposed method that involves the creation of new data to mitigate sparsity issues in SVD-based recommendations. It details the workflow, including creating rating matrices, similarity calculations, neighbor selection, and data creation techniques.

Experimental Evaluation: Experimental results on the MovieLens 100k dataset demonstrate the effectiveness of the proposed approach, showcasing improved performance in comparison to existing methods.

Conclusion and Future Directions: The study concludes by highlighting the significant advancements made in mitigating data sparsity in recommender systems. It suggests extending imputing data methods to other matrix factorization techniques and exploring more accurate methods for measuring similarity among users or items.

Citation : -

M. Z. Bahar and Z. K. A. Baizal, "Online Course Recommender System using Singular Value Decomposition," 2023 International Conference on Data Science and Its Applications (ICoDSA), Bandung, Indonesia, 2023, pp. 187-190, doi: 10.1109/ICoDSA58501.2023.10276431.X. Zheng, M. Guan, X. Jia, L. Guo and Y. Luo, "A Matrix Factorization Recommendation System-Based Local Differential Privacy for Protecting Users' Sensitive Data," in IEEE Transactions on Computational Social Systems, vol. 10, no. 3, pp. 1189-1198, June 2023, doi: 10.1109/TCSS.2022.3170691.K. Macwan, A. Imine and M. Rusinowitch, "Privacy Preserving Recommendations for Social Networks," 2022 Ninth International Conference on Social Networks Analysis, Management and Security (SNAMS), Milan, Italy, 2022, pp. 1-8, doi: 10.1109/SNAMS58071.2022.10062760.S. G. Aarella, A. K. Tripathy, S. P. Mohanty and E. Kougianos, "iTour2.0: A Smart Tourism Application for Independent Mobility of Tourists," 2021 19th OITS International Conference on Information Technology (OCIT), Bhubaneswar, India, 2021, pp. 472-477, doi: 10.1109/OCIT53463.2021.00097.

Algorithm:-

Recommender System Evaluation using SVD

Step 1: User-Item Matrix Creation

1.Input: Ratings dataset containing UserID, MovieID, and Ratings.
2.Process: Convert the ratings data into a user-item matrix, where rows represent users, columns represent movies, and the cells represent ratings. If a user hasn't rated a movie, it is filled with 0.

Step 2: Singular Value Decomposition (SVD)

1.Input: User-Item Matrix
2.Process: Perform SVD using numpy's linalg.svd function to decompose the user-item matrix into three matrices: U, sigma, and Vt.

Step 3: Predicted Ratings Reconstruction

1.Input: U, sigma, and Vt matrices obtained from SVD.
2.Process: Reconstruct the matrix by multiplying U, sigma (converted to a diagonal matrix), and Vt to obtain predicted ratings for all users and items.

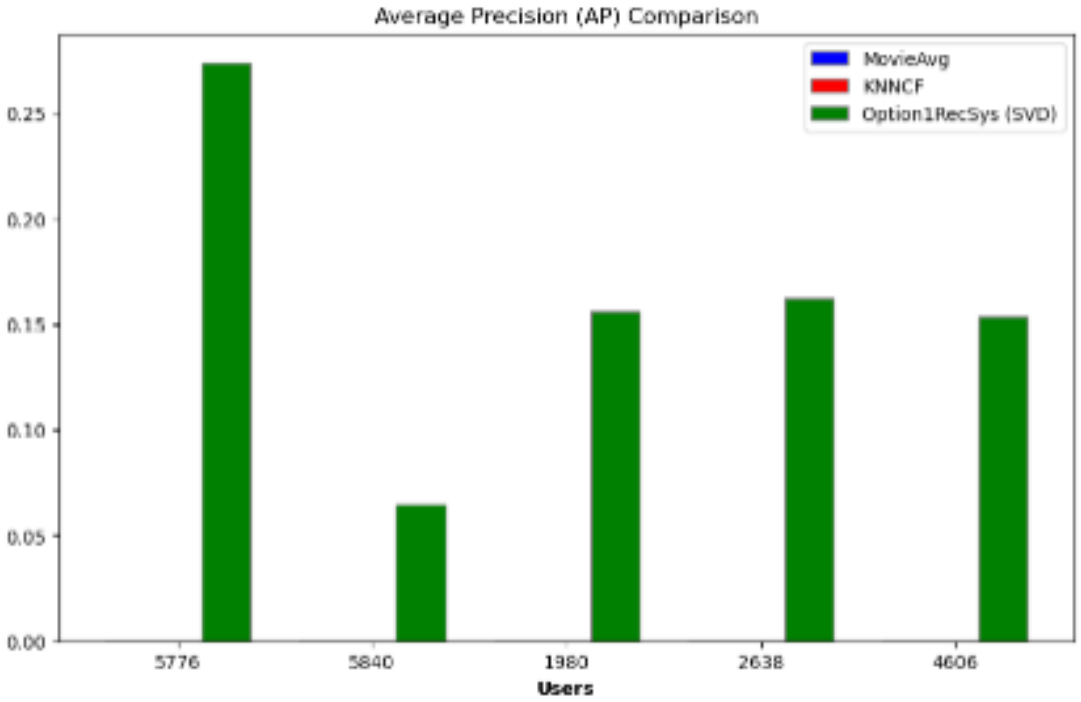
Step 4: Generating Recommendations

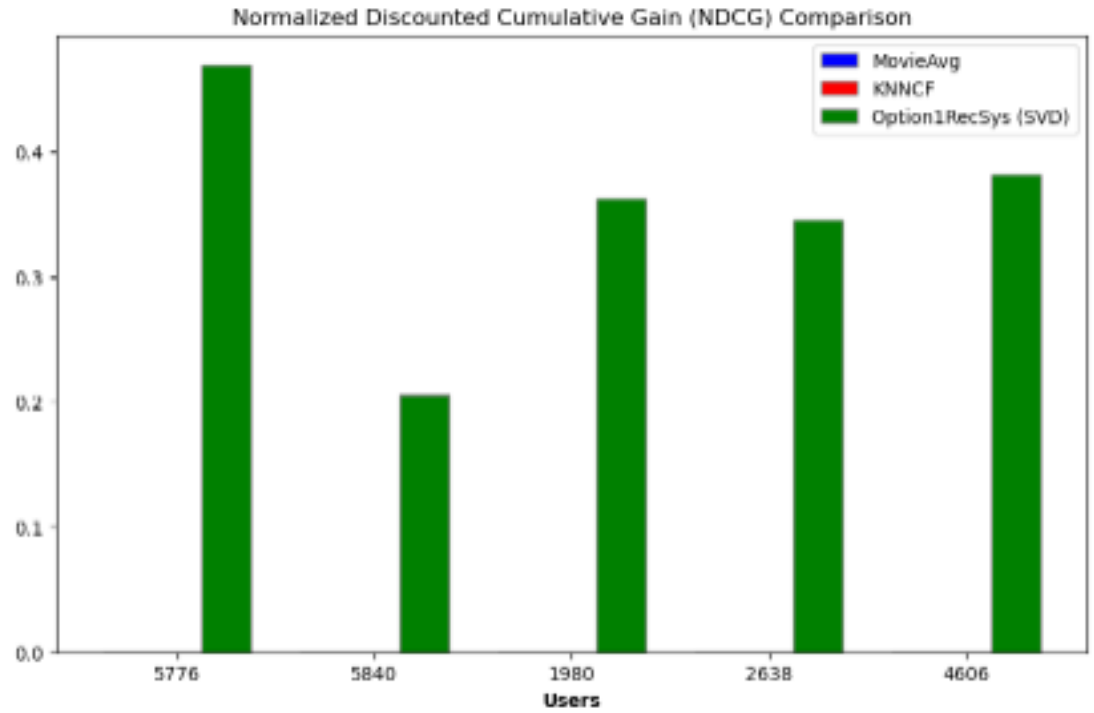
1.Input: User ID, N.
2.Process:
•Extract the predicted ratings for the given user from the reconstructed matrix.
•Sort the predicted ratings in descending order and select the top N movies as recommended items.

Step 5: Evaluating Recommendations

1.Input: User ID and Recommended Movies
2.Process:
•Get the top-rated movies from the original ratings dataset for the specified user . Compute Average Precision (AP) and Normalized Discounted Cumulative Gain (NDCG) to evaluate the quality of recommendations against the user's actual top-rated movies.

```
Randomly selected users: [5776 5840 1980 2638 4606]
MovieAvg Metrics: {5776: (0.0, 0.0), 5840: (0.0, 0.0), 1980: (0.0, 0.0), 2638: (0.0, 0.0), 4606: (0.0, 0.0)}
KNNCF Metrics: {5776: (0.0, 0.0), 5840: (0.0, 0.0), 1980: (0.0, 0.0), 2638: (0.0, 0.0), 4606: (0.0, 0.0)}
SVD Metrics: {5776: (0.27391071068475786, 0.4696426003356602), 5840: (0.06444444444444444, 0.205307077467291), 1980: (0.1562649
5726495726, 0.36311533850387234), 2638: (0.16231685497165368, 0.3450996801725811), 4606: (0.15369938107869144, 0.38133098444382
81)}
```





Output from the 2 graphs:-

1st graph shows Average Precision Comparison for all the 3 methods
2nd graph shows NDCG Comparison for all the 3 methods

I am first randomly selecting 5 users who have rated more than 100 movies
each) and recommending Top-30 movies (to each user).

For SVD Metrics, the results seem to be comparatively higher, indicating a better performance in terms of precision and relevance.
This implies that SVD-based recommendations might suit the user's preferences more effectively in comparison with other 2 methods

KNNCF Metrics and MovieAvg Metrics based on AP_and_NDCG doesn't give better results when compared to Collaborative filtering approach using matrix factorization and SVD.

Ref: -

1. R. Barathy; P. Chitra (23 April 2020) ‘Applying Matrix Factorization In Collaborative Filtering Recommender Systems’ Available at : https://ieeexplore-ieee-org.ezproxy.lib.rmit.edu.au/abstract/document/9074227/references#references
2. Xin Guan; Chang-Tsun Li; Yu Guan (24 November 2017) ‘Matrix Factorization With Rating Completion: An Enhanced SVD Model for Collaborative Filtering Recommender Systems’ Available at: https://ieeexplore-ieee-org.ezproxy.lib.rmit.edu.au/abstract/document/8119909
3. Dheeraj Bokde, Sheetal Girase, Debajyoti Mukhopadhyay (17 June 2015) ‘Matrix Factorization Model in Collaborative Filtering Algorithms: A Survey ‘ Available at : https://www-sciencedirect-com.ezproxy.lib.rmit.edu.au/science/article/pii/S1877050915007462?via%3Dihub