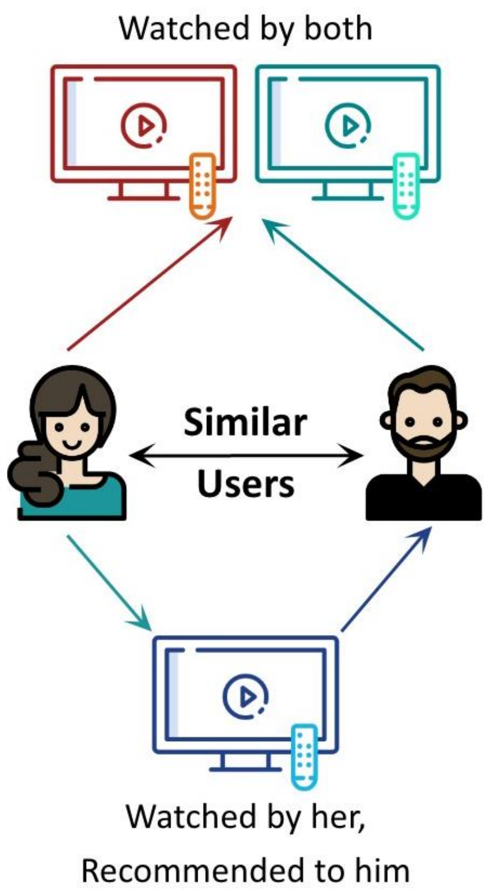
Task 1 - User-based Collaborative Filtering

Develop, implement, and evaluate user-based KNN collaborative filtering system

Collaborative Filtering



This system seeks out users with similar preferences and recommends products to one user based on the product preferences of others who are similar.

Cosine similarity is used to measure how similar users' preferences are to each other. Between two users u and v, the cosine similarity is calculated as:

$$\text{cosine\_sim}(u, v) = \frac{\sum_{i \in I_{uv}} r_{ui} \cdot r_{vi}}{\sqrt{\sum_{i \in I_{uv}} r_{ui}^2} \cdot \sqrt{\sum_{i \in I_{uv}} r_{vi}^2}}$$

Then top k users who are most similar to the target user are selected using k-Nearest Neighbor algorithm.

The k-NN algorithm and cosine similarity are fundamental components of recommendation systems, helping to identify similar users or items based on their historical behavior and preferences, ultimately enabling personalized recommendations.

The code performs collaborative filtering using the Surprise library to recommend movies for a randomly selected user.

In the Surprise library, KNNBasic with Cosine similarity identifies similar users by constructing a user-item matrix, followed by the computation and comparison of cosine similarities between their rating patterns. Subsequently, it recommends items that have received high ratings from these similar users to the target user.

Value of k	RMSE
2	**0.6359**
5	0.7938
10	0.8846
20	0.9552
50	1.0155

The table demonstrates how the Root Mean Squared Error (RMSE) changes as the number of neighbors (k) in a user-based collaborative filtering model varies, with the lowest RMSE occurring at k=2 (0.6359), indicating better prediction accuracy.

Steps:

1. Import Libraries

Import KNNBasic from Surprise library.

2. User Selection:

Randomly selects a user for testing.

3. Data Preparation:

Extracts movies rated by the target user.

4. Surprise Setup:

Configures Surprise with a rating scale (1 to 5).

5. Model Configuration:

Use KNNBasic algorithm for user-based collaborative filtering and Cosine similarity metric for recommendation.

6. Parameter Testing:

Iterates over different k (number of neighbors) values.

7. Recommendations and RMSE:

Generates movie recommendations and calculates RMSE to evaluate recommendation accuracy.

8. Results:

Displays k values and respective RMSE.

Task 2 - Item-based Collaborative Filtering

Develop, implement, and evaluate item-based KNN collaborative filtering system

Unlike user-based collaborative filtering, item-based filtering assesses the similarity between different items, such as movies. It does this by analyzing how often two movies are watched by the same users. If there's a significant correlation, it suggests that the two movies are similar, and they are recommended to viewers who watched one of them. For instance, if a user watched Movie X and there's a high correlation with Movie Y, Movie Y will be suggested to them, and vice versa.

Following the same steps as in Task 1, we start with Item Selection - Randomly selects an item for testing and set the value for k = 5, we then generate two models:

1. KNNBasic with Mean Squared Difference (MSD) similarity
   - Compute the Mean Squared Difference similarity between all pairs of items and select k-Nearest Neighbors.
2. KNNBasic with Pearson Similarity
   - Compute the Pearson correlation coefficient between all pairs of items and select k-Nearest Neighbors.

Then both the models are trained for each model we predict the item’s rating by different users and calculate RMSE using actual and predicted ratings.

Mean Squared Difference Similarities between items i and j are calculated using the below formula:

$$\text{msd}(i, j) = \frac{1}{|U_{ij}|} \cdot \sum_{u \in U_{ij}} (r_{ui} - r_{uj})^2$$

Pearson Similarities between items i and j are calculated using the below formula:

$$\text{pearson\_sim}(i, j) = \frac{\sum\limits_{u \in U_{ij}} (r_{ui} - \mu_i) \cdot (r_{uj} - \mu_j)}{\sqrt{\sum\limits_{u \in U_{ij}} (r_{ui} - \mu_i)^2} \cdot \sqrt{\sum\limits_{u \in U_{ij}} (r_{uj} - \mu_j)^2}}$$

Similarity metric	RMSE
MSD	0.4336
Pearson	0.5485

The Root Mean Squared Error (RMSE) for MSD similarity is 0.4336, indicating better prediction accuracy compared to a lower RMSE. In contrast, the RMSE for Pearson similarity is 0.5485, suggesting slightly less accurate predictions.

Task 3: A Better Recommender System

To make a better recommendation system, I first concentrated on the advantages and disadvantages of previously used algorithm which is KNNBasic. K-NN mainly focuses on the similarity, closeness, and distance of the data points [2]. The main advantage of K-NN is the effectiveness in training larger data as the cost of the learning process is zero[1]. However, a limitation of KNNBasic is its lack of explicit consideration for user and item biases, as it assumes a uniform distribution of ratings. Consequently, it may not offer highly accurate recommendations when working with sparse datasets or datasets characterized by diverse user-item interactions.

To enhance the performance of KNN, we used the KNNBaseline method. KNNBaseline incorporates baseline estimations to correct for user and item biases. The KNNBaseline algorithm, unlike conventional KNN, computes the baseline, in other words it balances the scores when there are very high or low ratings [3].

It adjusts recommendations based on the historical behaviour of users and items, leading to more accurate predictions. KNNBaseline uses the baseline factor. The baseline technique helps the learning models to identify the functional relationships between the input (feature) and the desired output (label). This approach predicts the rating by using a baseline rating[1].

This choice aimed to capture more nuanced patterns in user behaviour and also allowed us to make use of **the Pearson baseline similarity metric**, which is a robust method for measuring similarity between items.

In optimizing our recommendation system, we conducted experiments to find the ideal 'k' parameter for neighbor selection. After testing different values, we determined that setting 'k' to 10 resulted in the most accurate recommendations. Additionally, for our KNNBaseline model, we configured key hyperparameters: 'method: sgd' for effective optimization, 'n_epochs: 5' to balance learning and overfitting, and 'learning_rate: 0.00005' for stable, gradual model adjustments during training.

In assessing our recommendation system's performance, we utilized the Root Mean Squared Error (RMSE) as a reference metric. Lower RMSE values signify more accurate recommendations. Our method enabled a systematic refinement of the recommendation system, ultimately identifying 'k’ = 2 as the optimal choice for delivering precise movie recommendations.

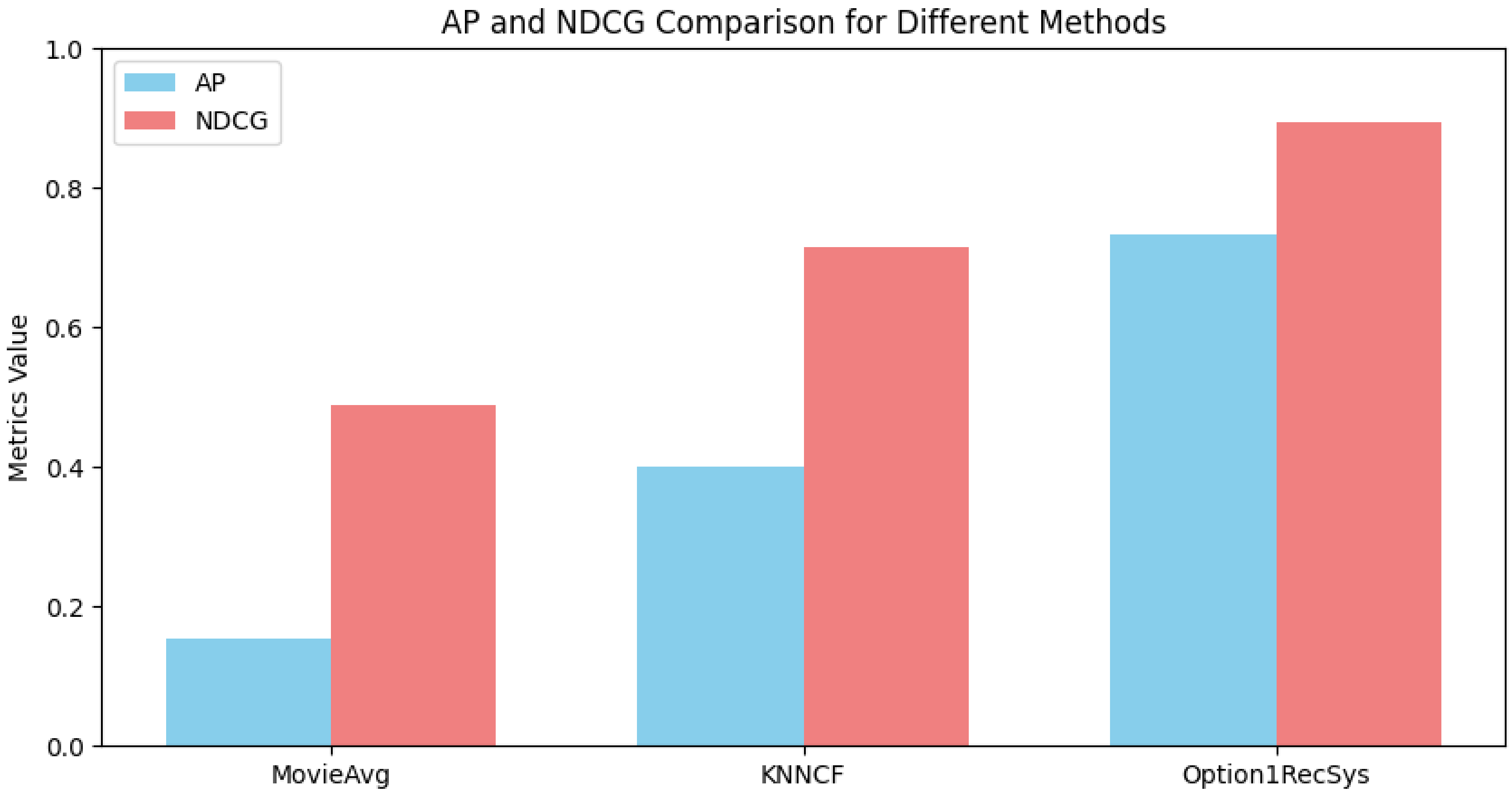
To enhance our recommendation system, we transitioned from KNNBasic, which lacks bias handling, to KNNBaseline. This switch led to more accurate predictions by considering user and item biases. With 'k' set to 10 and well-configured hyperparameters, our refined system achieved improved accuracy, as reflected in lower RMSE values.

Number of Neighbors (k): 2

Root Mean Squared Error (RMSE): 0.3561

Task 3.2 – Evaluate models using AP and NDCG

Model	AP	NDCG
MovieAvg	0.15	0.49
KNNCF	0.39	0.71
Option1RecSys	0.73	0.89



References

[1] Anwar, T., Uma, V., Hussain, M.I. *et al*. Collaborative filtering and kNN based recommendation to overcome cold start and sparsity issues: A comparative analysis. *Multimed Tools Appl* 81, 35693–35711 (2022). https://doi.org/10.1007/s11042-021-11883-z

[2] Subramaniyaswamy V, Logesh R (2017) Adaptive knn based recommender system through mining of user preferences. Wirel Pers Commun 97(2):2229–2247

[3] Yehuda Koren. Factor in the neighbors. ACM Transactions on Knowledge Discovery from Data, 4(1):1–24, January 2010. doi: 10.1145/1644873.1644874. URL https: //doi.org/10.1145/1644873.1644874.

[4] Ricci, F., Rokach, L., Shapira, B. (2011). Introduction to Recommender Systems Handbook. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P. (eds) Recommender Systems Handbook. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-85820-3_1

[5] Walek, Bogdan & Fojtik, Vladimir. (2020). A hybrid recommender system for recommending relevant movies using an expert system. Expert Systems with Applications. 158. 113452. 10.1016/j.eswa.2020.113452.

[6] Surprise, “Surprise’ documentation” 2023. [Online]. Available: https://surprise.readthedocs.io/en/stable/index.html#