Group: DSA4212 Assignment 2 Report

MAH SHIAN YEW BRENDAN, A0216349B
TEO MING JUN, A0216305R
SANDRA WONG JIA YI, A0222021H

Building models without factor models:

Building recommendation models without factor models involves using algorithms that do not rely on matrix factorization or latent features. A few of the non-factor methods we researched include Content-based filtering, Collaborative filtering and Hybrid systems.

Firstly, content-based filtering is an approach that recommends items to users based on the similarity of the items' attributes to the user's preferences. It analyzes the content of the films and recommends films that are similar to films the user has liked in the past. Secondly, collaborative filtering recommends films to users based on the similarity of their preferences to those of other users. It finds users with similar preferences to the user and recommends films that those users have liked. Thirdly, hybrid systems combine multiple approaches to provide more accurate and diverse recommendations to users. This leverages the strengths of each approach and overcomes its limitations to provide personalized and relevant recommendations to users.

The main difference between content-based and collaborative filtering is the source of the recommendations. Content-based filtering uses the attributes of the items, while collaborative filtering uses the behaviour and preferences of other users. Hybrid systems combine both approaches to improve the accuracy and coverage of recommendations. Later on, we will delve deeper into these three categories of systems.

We decided to use Collaborative Filtering as our recommendation system for the initial stage of the assignment. This is because it has some advantages such as not needing any prior knowledge of the domain since all the characteristics are learned automatically. Additionally, it assists users in discovering new interests.

In this first implementation, we use Pandas to load three datasets: 'assignment_2_anime.csv', 'assignment_2_ratings_test.csv', and 'assignment_2_ratings_train.csv'. We then calculate the mean rating for each anime in the training set and fill in the missing rating values in the test set with the corresponding mean rating using apply() and lambda functions. Next, we calculate the mean rating for each user in the training set and predict the ratings for each user and anime in the test set using a for loop and if-else statements. Finally, it calculates the mean squared error (MSE) between the predicted ratings and actual ratings in the test set using the mean() function and prints it. We obtained a mean squared error (MSE) of 1.86 which indicates that the

recommendations are relatively accurate. However, there is always scope for enhancement by tackling some of Collaborative Filtering's drawbacks, which include its suboptimal performance in situations where data is scarce or where popularity bias is present. This is because the system is built on past ratings, which means that it won't suggest items with little or limited data. As a result, well-known products are likely to remain popular over time, while novel and varied choices may be neglected.

Building models with factor models:

Factor models are a specific type of collaborative filtering that use matrix factorization to extract latent factors representing user preferences and item attributes. These factors can be used to predict missing ratings in the ratings_matrix and generate personalized recommendations.

Singular Value Decomposition (SVD) is a type of factor model that is commonly used in recommender systems. SVD is a matrix factorization technique that reduces the dimensionality of a user-item interaction matrix, where each row represents a user and each column represents an item, by decomposing it into a product of three matrices: a user-factor matrix, a factor-item matrix, and a diagonal matrix of singular values. U, V^T, and Σ respectively. The U and V matrices contain the left and right singular vectors of the original matrix, and the Σ matrix contains the singular values of the original matrix. The factor matrices represent latent factors that capture the underlying preferences of users and features of items, while the singular values indicate the relative importance of these factors. The SVD algorithm works by finding the best approximation of the original matrix using a lower-rank approximation. The lower-rank approximation is achieved by setting some of the singular values to zero, effectively removing some of the dimensions of the original matrix. The resulting approximation is then used to make predictions about user-item ratings.

In the context of recommendation systems, the SVD algorithm is used to approximate the user-item rating matrix. The user-item rating matrix is typically sparse, meaning that most entries are missing. The SVD algorithm fills in the missing entries by approximating the original matrix using the lower-rank approximation described above.

For this project, we implemented SVD using the surprise library in Python to predict what users would rate an anime.

Firstly, we load the anime, training and test data into pandas data frames. Next, we define the rating scale(1-10) using the Reader class from Surprise. We then split the training set into a smaller training set and a validation set. We train our collaborative filtering model using the SVD algorithm from Surprise on the smaller training set and evaluate its performance on the validation set using the MSE metric.

With the trained model, we use it to make predictions on the test set and evaluate its performance using the MSE metric. The lower the MSE score, the more accurate the model is. From the model, we obtained an MSE of 1.37, which can be considered a good result. This indicates that the SVD algorithm can predict the ratings of anime titles with a reasonable level of accuracy. Hence, the recommended anime titles are likely to be of interest to the users as it indicates that the recommended anime titles are more likely to align with the users' preferences.

Hybrid Model:

When it comes to recommending anime shows, there are different approaches that can be used, such as Content-Based Filtering and Collaborative Filtering. Content-Based Filtering analyzes various factors like genres, themes, and production studios of the animes that the user has rated highly to recommend similar shows with the same attributes. One significant advantage of this approach is that it does not depend on any latent factors, but instead uses the attributes or features of the items to create a user profile and make recommendations based on similar features.

In contrast, Collaborative Filtering only uses user-item ratings to make recommendations. Content-Based Filtering, on the other hand, uses features or attributes of the items. The goal of Content-Based Filtering is to learn a model that maps user preferences to item features, which can then be used to recommend items that share similar features with the ones the user already likes. This approach is especially useful when there is limited user-item interaction data available or when the user's preferences are very specific.

In our case, we encoded the genres of anime into a 43-column feature data frame, where each column would be a binary indication of whether the anime falls under a certain genre. However, the limitations of these models can be overcome by combining them in a hybrid recommendation system. This approach can effectively leverage the strengths of both methods to provide better recommendations to users.

Hybrid approaches use a combination of content-based and collaborative filtering techniques to generate recommendations. In the case of anime, this approach could entail analyzing the attributes and features of highly-rated anime by similar users and utilizing this information to generate recommendations for the target user. By combining the strengths of both approaches, hybrid techniques can provide more accurate and diverse recommendations that cater to the unique preferences of each user. This can result in a more satisfying and engaging experience for users who are seeking tailored recommendations that align with their interests and tastes. Therefore, by utilizing a hybrid recommendation system, we can provide a more personalized recommendation service to our users.

Our research group endeavoured to develop a hybrid model inspired by Denise Chen's work on Matrix Factorization in Recommender Systems, as cited below. However, we encountered challenges during the model building and testing phase as the model's training process was significantly time-consuming. As a solution, we attempted to reduce the training dataset by sampling it, but the model still required an excessive amount of time to run. Consequently, we reached a consensus as a group that the model was not feasible for use in our final predictions.

Conclusion:

In conclusion, this project aimed to explore different approaches to building recommendation systems that do not rely solely on factor models. Various models, both simple and complex, were experimented with, and new techniques were explored beyond those discussed in class. Additionally, the project studied whether variations and extensions of factor model ideas could improve recommendation system performance.

Overall, out of all the methods, the SVD model performed the best. The model did not have a long run time, possibly due to it using pre-built Python functions and packages. Also, compared to the non-factor Collaborative Filtering model, the SVD performed better in predicting the ratings of the test dataset provided. Both the MSE and RMSE are lower for the SVD compared to the non-factor Collaborative Filtering model.

Works Cited:

Chen, Denise. "Recommender System — Matrix Factorization." Towards Data Science, 9 July 2020, https://towardsdatascience.com/recommendation-system-matrix-factorization-d61978660b4b.

gspmoreira. "Recommender Systems in Python 101." Kaggle, 8 Dec. 2019, https://www.kaggle.com/code/gspmoreira/recommender-systems-in-python-101.

jieyima. "Netflix Recommendation: Collaborative Filtering." Kaggle, 23 Feb. 2018, https://www.kaggle.com/code/jieyima/netflix-recommendation-collaborative-filtering.

Turing. "How Does Collaborative Filtering Work in Recommender Systems?" Turing Enterprises Inc, 29 July 2022, https://www.turing.com/kb/collaborative-filtering-in-recommender-system#advantages.