DSA4212 Assignment 2 - Report
Group 39 (Chang An Le Harry Jr, A0201825N)

Introduction
Anime has become a popular form of entertainment worldwide, with a diverse range of genres and styles. Building an effective recommendation system can help users discover new anime based on their preferences, enhancing their overall experience. In this overview, we explore several approaches to building an anime recommendation system with the datasets provided, including content-based filtering, clustering, collaborative filtering, and matrix factorization.

Content-based filtering
Content-based filtering recommends items to users based on their past preferences and the features of the items. In this approach, we first preprocess the anime dataset by filling missing values for the 'genre' and 'type' columns. We then create a TF-IDF matrix for the combined features (genre and type) and compute the cosine similarity between anime. The predicted rating for an item is calculated as a weighted sum of the user's ratings for similar items, where the weights are based on the similarity scores.

Clustering
Clustering involves grouping similar items together based on their features. In this approach, we preprocess the anime dataset in the same way as for content-based filtering. We then create a TF-IDF matrix for the combined features and perform k-means clustering using the matrix. The predicted rating for an item is calculated as the user's mean rating for the anime in the same cluster. If the user has not rated any anime in the same cluster, we use the user's overall mean rating as the prediction.

Collaborative filtering
Collaborative filtering uses the preferences of other users to recommend items to the target user. We can implement user-based and item-based collaborative filtering using the KNNWithMeans algorithm with Pearson correlation.

- User-based Collaborative Filtering: Finds users with similar preferences to the target user and recommends items those similar users liked. The predicted rating for an item is calculated as a weighted average of the ratings given by similar users.
- Item-based Collaborative Filtering: Finds items similar to the ones the user liked and recommends them. The predicted rating for an item is calculated as a weighted average of the user's ratings for similar items.

Matrix factorization
Matrix factorization decomposes the user-item preference matrix into lower-dimensional user and item feature matrices, allowing for the identification of latent factors. In this report, we implement Singular Value Decomposition (SVD) to achieve this.

- Singular Value Decomposition (SVD): SVD decomposes the user-item preference matrix into three matrices: user feature matrix, singular value matrix, and item feature matrix. SVD can handle missing values and is usually used with gradient descent optimization to minimize the error between the original and reconstructed preference matrix.

Evaluation
To evaluate the performance of these approaches, we calculate evaluation metrics such as mean squared error (MSE), root mean squared error (RMSE), or mean absolute error (MAE) on a test dataset. The test dataset contains user preferences data that was not used during the training process. By comparing the performance of content-based filtering, clustering, collaborative filtering, and matrix factorization, we can determine which method or combination of methods is most effective for recommending anime based on user preferences.

Tabulating some of the test MSE values below, listed are the main methods that were tested:

Method	**Test MSE**
Content-based filtering	2.7956033897683574
Clustering	2.054134889019177
Item-based collaborative filtering	1.3823935232597724
Matrix factorization (SVD)	1.316212903953369

Optional Extensions
In a third and optional stage, we can improve the recommendation system further by incorporating additional data and techniques. One possible approach is to scrape additional data online, such as user reviews, staff information, or anime production details, to enrich the dataset and potentially enhance the system's performance. We also explore hybrid approaches, combining multiple recommendation techniques or incorporating deep learning methods.

Web Scraping for Additional Data
Scraping additional data can provide more information to support the recommendation process. For example, we can scrape user reviews from websites like MyAnimeList or IMDb to collect user sentiments and opinions about different anime. We can also scrape information about anime staff (directors, writers, composers, etc.) or production details (studios, airing dates, etc.) to enrich the content-based features.

Hybrid Approaches
Hybrid approaches combine different recommendation techniques to exploit the strengths of each method. For example, we can combine content-based filtering with collaborative filtering or matrix factorization. Additionally, we can use ensemble techniques, such as stacking or weighted averaging, to combine the predictions from multiple models.

Deep Learning Methods
Deep learning methods can help capture complex patterns in the data and improve the recommendation system's performance. We can implement neural networks, such as

autoencoders, to perform matrix factorization, or use sequence models like LSTM (Long Short-Term Memory) or GRU (Gated Recurrent Unit) networks to capture temporal patterns in user preferences.

Conclusion

This report has presented various approaches for building an anime recommendation system, including content-based filtering, clustering, collaborative filtering, and matrix factorization using Singular Value Decomposition (SVD). Additionally, it explores optional extensions such as web scraping for additional data, hybrid approaches, deep learning methods, and incorporating new data into the recommendation system. By evaluating their performance using metrics such as MSE on a test dataset, we can determine the most suitable approach or combination of approaches for recommending anime based on user preferences. Further improvements can be made by experimenting with different feature combinations, similarity metrics, clustering algorithms, parameter tuning techniques, and optional extensions.

References

Hug, N. (n.d.). Surprise: A Python scikit for recommender systems. Retrieved from https://surprise.readthedocs.io/en/stable/

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. Knowledge-based systems, 46, 109-132. https://doi.org/10.1016/j.knosys.2013.03.012

Lu, J., Wu, D., Mao, M., Wang, W., & Zhang, G. (2015). Recommender system application developments: A survey. Decision Support Systems, 74, 12-32. https://doi.org/10.1016/j.dss.2015.03.008

Zhang, S., Yao, L., Sun, A., & Tay, Y. (2017). Deep learning based recommender system: A survey and new perspectives. ACM Computing Surveys (CSUR), 52(1), 1-38. https://doi.org/10.1145/3285029

Lops, P., de Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In Recommender systems handbook (pp. 73-105). Springer. https://doi.org/10.1007/978-0-387-85820-3_3