**1.0 Introduction**

Users can utilize recommender systems as guides to help them determine what to do next. In the situation of anime series, users are sometimes bewildered by the ever-increasing number of projects available and end up only touching the surface of what the animation world has to offer (Vie *et al*., 2017). Collaborative filtering is a well-known technique for predicting unseen ratings based on known rating data from users. A large amount of data is often required for collaborative filtering (Aggarwal, 2016). This necessitates the use of PySpark for this research. PySPark allows for the use of apache Spark using python. Spark data frame distributes operations over clusters in a parallel form across available processors; as such, it scales to the processing of big data (Drabas and Lee. 2017).

There have been numerous projects on the recommender system. Kuchaiev and Ginsburg (2017) adopted a deep auto-encoder of 6 levels to create a recommendation system called ‘DeepRec.’ Ramashini et al. (2018) overcame the cold start problem on building a recommender system for leisure and tourism. The system implement collaborative filtering alongside content-based filtering, thus forming a hybrid system to make a recommendation system. ‘AniReco: Japanese Anime Recommendation System’ was proposed by Ota *et al.* (2017), which used a content-based filtering approach. This research presents a collaborative filtering system of recommending anime to users.

**2.0 Approach**

The method used in this research are described under the following headings

* 1. **Pre-processing**:

The datasets were provided as two ‘comma separated values’ (CSV) datasets – ‘anime.csv’ and ‘rating.csv.’ Each dataset contains a rating column; to avoid ambiguity, these columns were renamed. The datasets were merged using a similar column in the datasets with an inner join. An inner join ensures an intersection of the two datasets (Brumm, 2019). This ensures that the final output contains information from both datasets.

The combined dataset was grouped for subsequent analysis, and operations such as average, maximum, and sum were performed to obtain the final outputs.

* 1. **Algorithm:**

The Alternating Least Squares (ALS) matrix factorization recommender algorithm was implemented from the PySPark machine learning library. The rating matrix R is estimated by ALS by multiplying the input matrix (X) with the output matrix (Y). These two matrices have low ranks. (Spark user guide, no date). These estimates are commonly referred to as 'factor' matrices. The method is recursive. One of the factor matrices is kept unchanged after each cycle, while least squares are used to compute the other. While solving for the other factor matrix, the factor matrix already obtained remains unchanged (Hu *et al.*, 2008). Rather than finding low-rank estimates to R, this method finds solutions for a preferred matrix P, with entries 1 and 0. Instead of explicit ratings provided to things, the ratings are provided as 'confidence' values tied to the strength of expressed user-preferred item (Spark user guide, no date).

**3.0 Results**

1. The anime dataset has 7 columns which consists of 'anime\_id', 'name', 'genre', 'type', 'episodes', 'rating', and 'members' while the rating dataset has 3 columns which includes 'user\_id', 'anime\_id', and 'rating'.
2. An inner join was performed on the datasets using the common column - 'anime\_id.'
3. Table of results

Table 1: Top ten anime based on user rating

|  |  |
| --- | --- |
| **Name** | **avg(user\_rating\_)** |
| Warui no wo Taose!! Salaryman Man | 10.0 |
| Shiroi Zou | 10.0 |
| Choegang Top Plate | 10.0 |
| STAR BEAT!: Hoshi no Kodou | 10.0 |
| Shiranpuri | 9.0 |
| Yakushiji Ryouko no Kaiki Jikenbo: Hamachou, Voice &amp; Fiction | 9.0 |
| Tang Lang Bu Chan | 9.0 |
| Doukyuusei | 9.0 |
| Steins;Gate 0 | 8.5 |
| Kimi no Na wa. | 8.3 |

Source: Author’s Computation (2021)

1. Table of result

Table 2: Top ten anime based on number episodes

|  |  |
| --- | --- |
| **Name** | **max(episodes\_)** |
| Oyako Club | 1818 |
| Doraemon (1979) | 1787 |
| Kirin Monoshiri Yakata | 1565 |
| Manga Nippon Mukashibanashi (1976) | 1471 |
| Hoka Hoka Kazok | 1428 |
| Monoshiri Daigaku: Ashita no Calendar | 1274 |
| Sekai Monoshiri Ryoko | 1006 |
| Kotowaza House | 773 |
| Shima Shima Tora no Shimajirou | 726 |
| Ninja Hattori-kun | 694 |

Source: Author’s Computation (2021)

Table 3: Top ten genres based on user rating

|  |  |
| --- | --- |
| **Genre** | **avg(user\_rating\_)** |
| Action, Historical, Kids | 10.0 |
| Action, Adventure, Drama, Fantasy, Magic, Military, Shounen | 8.0 |
| Action, Comedy, Historical, Samurai| | 8.0 |
| Drama, Fantasy, Romance, Slice of Life, Supernatural | 7.84 |
| Drama, Horror, Mystery, Police, Psychological, Seinen, Thriller | 7.81 |
| Action, Drama, Mecha, Military, Sci-Fi, Super Power | 7.77 |
| Drama, Music, Romance, School, Shounen | 7.74 |
| Drama, Historical, Seinen, Thriller | 7.71 |
| Sci-Fi, Thriller | 7.68 |
| Drama, Romance, School, Supernatural | 7.67 |

Source: Author’s Computation (2021)

1. Table 4: Table of result

|  |  |
| --- | --- |
| **User** | **Recommended anime** |
| 364 | 'Fullmetal Alchemist: Brotherhood,' 'Death Note,' 'Sword Art Online' |
| 648 | 'Shingeki no Kyojin,' 'Death Note,' 'Sword Art Online.' |
| 215 | 'Death Note,' 'Shingeki no Kyojin,' 'Sword Art Online.' |
| 946 | 'Code Geass: Hangyaku no Lelouch,' 'Code Geass: Hangyaku no Lelouch R2', 'Darker than Black: Kuro no Keiyakusha.' |
| 53 | 'Clannad', 'Toradora!', 'Angel Beats!' |
| 976 | 'Angel Beats!', 'Steins;Gate', 'Mirai Nikki (TV)' |
| 947 | 'Naruto', 'Death Note', 'Fullmetal Alchemist' |
| 366 | 'Shingeki no Kyojin', 'Death Note', 'Sword Art Online' |
| 386 | 'Death Note,' 'Naruto,' 'Code Geass: Hangyaku no Lelouch.' |
| 807 | 'Sen to Chihiro no Kamikakushi,' 'Howl no Ugoku Shiro,' 'Death Note.' |

Source: Author’s Computation (2021)

**References**

Aggarwal C.C. (2016) *Recommender systems*. Springer

Brumm B. (2019) Inner Join in *Beginning Oracle SQL for Oracle Database 18c*. Berkeley, CA.: Apress.

Drabas T., Lee D. (2017) *Learning PySpark*. Packt Publishing.

Hu Y., Koren Y. and Volinsky C. (2008) Collaborative Filtering for Implicit Feedback Datasets,’ *Eighth IEEE International Conference on Data Mining*, pp. 263-272.

Kuchaiev O, Ginsburg B (2017) Training deep autoencoders for collaborative filtering’ arXiv preprint, [online]. Available at: https://arxiv.org/abs/1708.01715 (Accessed: 6th November 2021).

Ota S., Kawata H., Muta M., Masuko S., Hoshino J. (2017) Anireco: Japanese anime recommendation system,  *International Conference on Entertainment Computing*, pp 400–403.

Ramashini M, Jayathunga D, Gowthamy A, Rammiya N, Kiruthiga U (2018) Personalized recommendation system for leisure time activity using social media data,’ *Psychology.*

Spark user guide (no date) ‘ALS,’ *Spark Python API Reference* [online]. Available at <https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.recommendation.ALS.html> (Accessed: 6th November 2021).

Vie J. *et al.* (2017) Using Posters to Recommend Anime and Mangas in a Cold-Start Scenario, *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, pp. 21-26.