CptS 315 Course Project: My Anime List Recommended System

Denise Tanumihardja (11698042)

Introduction

The purpose of the task is to create a recommendation system for anime (Japanese animation). A motivation for this task is to help a user to pick shows that may be outside of their preferences or close to it, because in today's world full of endless forms of entertainment, one could get lost or even discouraged to try something new. Another motivation is to highlight shows that don't get much attention because of other massively popular shows. My personal motivation, however, is to simply let people who are new into anime, experience other great shows outside of the popular show(s) that brought them into this entertainment medium.
A challenge I faced was interpreting the database and what it represented. This includes figuring out whether a value under a certain class represented a score or a rank. Another challenge was to determine which classes would be the best to take into account when calculating recommendation scores. Another challenge was to determine how to proceed when I figured out that the class value of a certain class was not a score, but instead a rank. This is because the normalization function required score values.

The output of the task resulted in six csv files of 50 recommended animes and an unedited results file. Each of the six files are split depending on what type of anime is it (TV show, Movie, Original Video Animation, Special, Music Video, Original Net Animation), sorted from most recommended to least. The unedited file contains the full ranking of all the shows. Moreover, scatterplots of the results were also generated.

Data Mining Task

The task I chose is to create a recommendation system that utilizes both content-based filtering as well as collaborative filtering; in other words, a hybrid collaborative filtering recommendation system. However, instead of the classic movie recommendation system, I decided on an anime recommendation system. As such, a dataset is required, which I utilized form Kaggle. The dataset was mined from an anime review site (similar to IMDB), called MyAnimeList. It contained three files, only one of which I used for the input of the recommendation system. As such, the output of this would generate a list of recommended shows based off of the collective and individual sentiments of each show, represented by the scores, popularity ranking and number of people who favourited the show.

Technical Approach

The process is divided into four parts: Data cleaning, Calculating weights, Normalization, and Output.

Data Cleaning

Using the 'pandas' library, the dataset was imported into a dataframe and was cleaned. This was done by filtering out shows that has less than 100 votes/ratings, as anything less would imply obscure or unknown shows which those new to anime would likely avoid watching.

Calculating Weights

The dataset has two classes called 'score' and 'scored_by', which represents the average rating score of a show and the number of people who rated, respectively. These two were used to calculate the weighted average score for each show by using the following formula:

$$W = \frac{Rv + Cm}{v + m}$$

where:

W = Weighted Rating
R = average for the movie as a number from 0 to 10 (mean) = (Rating)
v = number of votes for the movie = (votes)
m = minimum votes required to be listed in the Top 250 (currently 3000)
C = the mean vote across the whole report (currently 6.9)

Where W represents the weighted average.

R represents the score of a show, using the 'score' class.

v represents the number of people who rated a show, using the 'scored_by' class.

m represents the minimum number of votes to be calculated, using the 0.01% quartile of the 'scored_by' class.

C represents the average score of all shows, using the mean of the 'score' class values.

Normalization

The system was designed to use the weighted averages, popularity score for each show, and the number of users who favourited the show. However, as the 'popularity' class was not based on scores, and instead a rank, I decided to reverse the order of popularity ranks to signify popularity score; the higher the popularity score, the more popular it is.

The values in each class specified ('weighted_avg', 'popularity_score', and ''favorites) were normalized using a 'sklearn.preprocessing' library function called 'MinMaxScaler', to normalize the values in each class specified. In this case, the weighted average scores, the number popularity score, as well as the number of favourites, were normalized before being appended to the cleaned dataframe.

Output

The dataframe with the four appended classes was then split into each ‘type’ class (category), before exporting it as csv files containing recommended shows from a certain category in each. A seventh file was also exported which contains the unsplit dataframe containing the full list of animes from most to least recommended.

Additionally, by using the ‘matplotlib’ library, a scatter plot of the results was generated to find correlations between classes.

Evaluation Methodology

Utilizing the mined dataset from Kaggle, from MyAnimeList, the database held three files, a dataset containing the animes, a dataset containing the user profiles, and a dataset containing users’ bookmarked animes. Out of the three, only the first I utilized, which contained all the information regarding each show such as its name, score, runtime, date of aired, etc. Once processed in my system, it would output six csv files corresponding to each anime type: TV, Movie, OVA, Special, Music, and ONA. With each file containing the top 100 anime recommendations for that category, complete with all the necessary information.

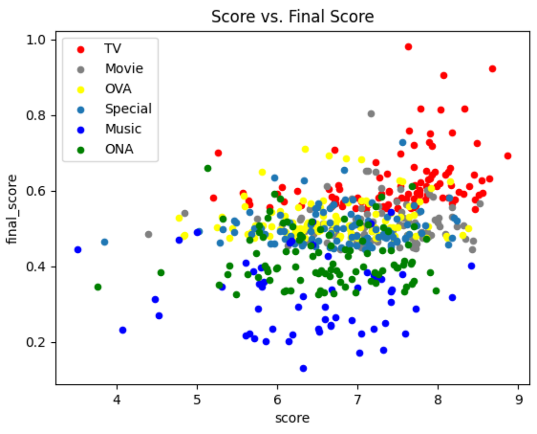
To evaluate the data, the three normalized classes (‘normalized_weight’, ‘normalized_popularity’, and ‘normalized_fav’) are used to evaluate a final score. This is done by having each instance in each class summed with 33% priority, before the results are appended into the same dataframe. Afterwards, it was sorted in descending order of the new class ‘final_score’ and filtered for any show that had ‘NaN’ characters in either of the four classes.

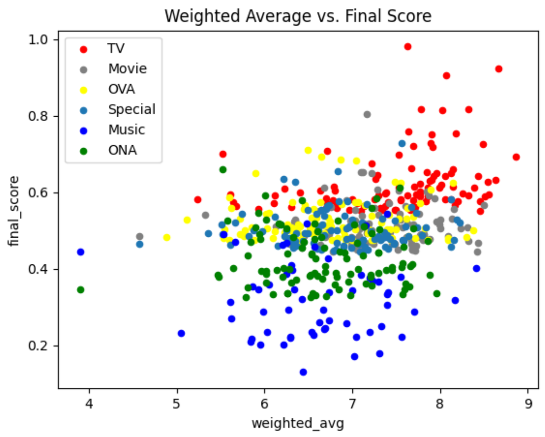
Results and Discussion

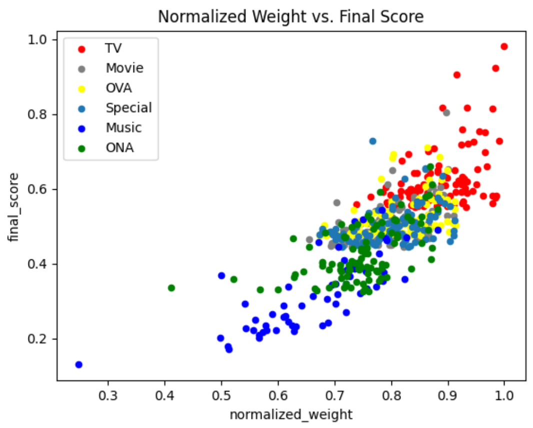
In the process of doing the project, I have noticed how the final scores of each show seemed almost nonsensical. As such, with a large dataset, exploratory data analysis is a given, and I decided to plot two classes in a scatterplot to reveal, if any, relationships between the classes. Moreover, I decided to combine the plots for each anime type/category in one plot.

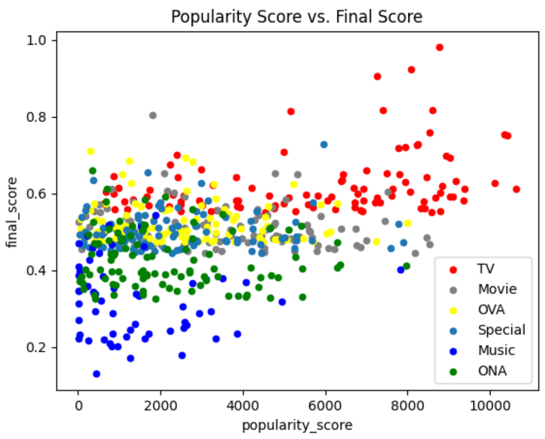
When the classes were plotted, it was evident that the final scores had overall higher value for TV animes, and lower values with Music animes. In between, ONAs came after Music animes, with anime Movies and Specials overlapping next, and OVAs between Movies and Specials and TV animes. In the case of the plots with classes in their base form and normalized forms, it is easier to see this relationship in the plots with the normalized values.

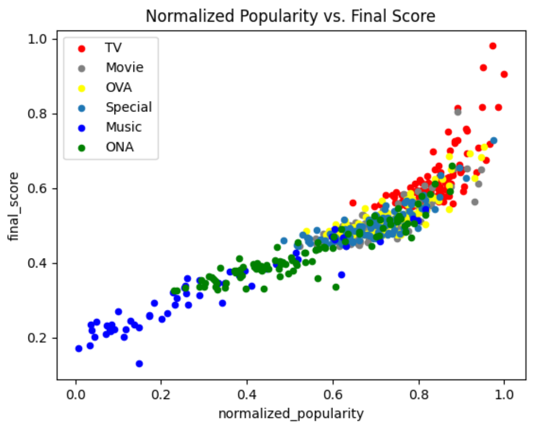
The plots are as given below:

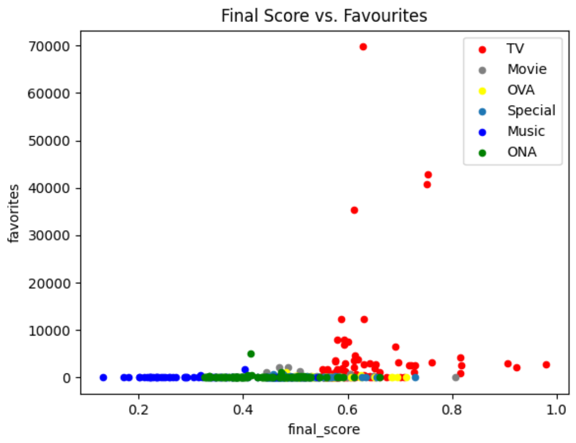


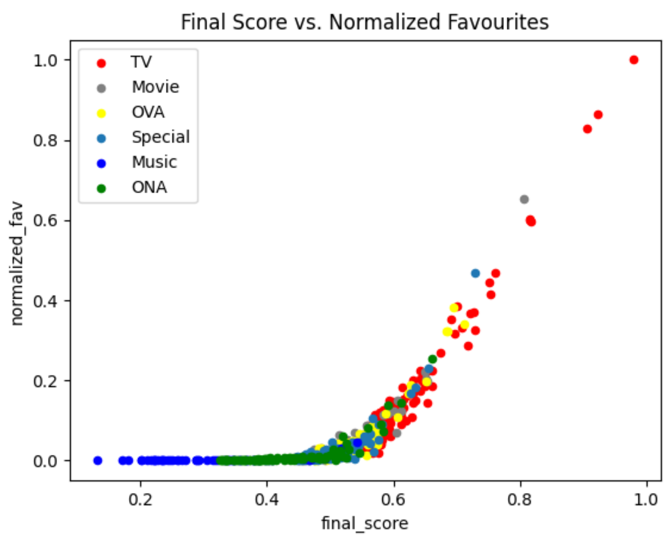












Plotting the final scores against the scores and weighted averages did not seem to show much difference. This was because the values in both classes were about equal. Moreover, it seems that plotting the final scores against the number of favourites did not work as the few instances where the number of favourites is significantly higher than the majority, resulted in a distorted graph. However, in both cases, the plot with final scores against their normalized values seemed to result in a clearer graph where it is clearly shown a proportional relationship between the two axes. This is also seen for the plot with final score against popularity scores and normalized popularity.

Lessons Learned

In hindsight, perhaps if I had used a different method of recommended system, the results would have been somewhat improved. Moreover, I should have found a way to present the results (recommendations) in a cleaner and convenient fashion rather than a csv file. In addition to this, I should have though of a way to present which piece of information (classes) of a datapoint is necessary and which is unnecessary information.

Perhaps, I should have also looked into other libraries or functions in the current libraries I used, to further simplify the algorithm and/or simplify the process. However, as evident in the plots, it was a good decision to split each recommendation file by the category, else the top recommendations would preferentially be TV animes, rather than the others.

Acknowledgements

- Mukherjee, Aritro. “Building a Recommendation System Using Weighted-Average Score.” *Medium*, Medium, 16 Sept. 2019, medium.com/@developeraritro/building-a-recommendation-system-using-weighted-hybrid-technique-75598b6be8ed.

- Azathoth. “Myanimelist Dataset.” *Kaggle*, www.kaggle.com/datasets/azathoth42/myanimelist.