Anime Recommendation System

Wan Qi Chen
School of Computing and Information
University of Pittsburgh
Pittsburgh, Pennsylvania
United States of America
wac45@pitt.edu

Victoria Clarchick
School of Computing and Information
University of Pittsburgh
Pittsburgh, Pennsylvania
United States of America
vlc24@pitt.edu

Abhibha Gupta
School of Computing and Information
University of Pittsburgh
Pittsburgh, Pennsylvania
United States of America
abg96@pitt.edu

ABSTRACT

This is a reflection on the creation of a recommendation system using R. The recommendation system was for the purpose of being able to suggest an anime to different users. The dataset was first explored through a various plots (Eg: cor plot) and the most important features were taken into consideration for the recommendation system. The different variables were evaluated as well to determine the influence between the variables. Each variable was run through an extraction and frequency function and then analyzed. After the data exploration the dataset was run through the recommendation system. We used the ‘hybrid’ approach that weighs different methods based on popularity, randomness and recommendation. The results for our method are provided in the subsequent sections.

KEYWORDS

Data Mining, Anime, Recommendation System, RMSE, MSE, MAE, Genre, Score, Rank, Popularity, Distribution

1 Introduction

The goal of the recommendation system is to spread the fun of anime to more people. We want to encourage the competition of the animation industry, by using a ranking system throughout different anime. The way that we wish to address creating a suggestion or ranking system is by use of categorization and familiarity between data that we have found based on viewer surveys. Finally, after the analysis we could base on the result to recommend to people which anime they might like or fit most depending on their preference of genre of anime.

The motivation behind the dive into this particular topic is based on our group's personal interests in the different anime genres. this system can reach audiences that are quite familiar with a variety of different anime series or those looking into what different series would align with their interests. Members may be able to obtain a recommendation from friends but we thought that it would be an interesting project to create a recommendation system through data mining. This way we can personally compare what part of our group would have recommended to the results of a recommendation from an analysis of the view surveys.

We have information related to the genre, ratings, synopsis, episodes, production studio, etc at our disposal. We aim to design a recommendation system that recommends the next anime show that a user should watch. Some tasks that we would like to explore would be classification. As well as recognize some recommendations for different users based on different inputs that a user may choose. To analyze these tasks we want to perform RMSE, MSE, and MAE techniques on the data. These techniques should allow us to process the data and report results that will help in evaluating the classifications and recommendations.

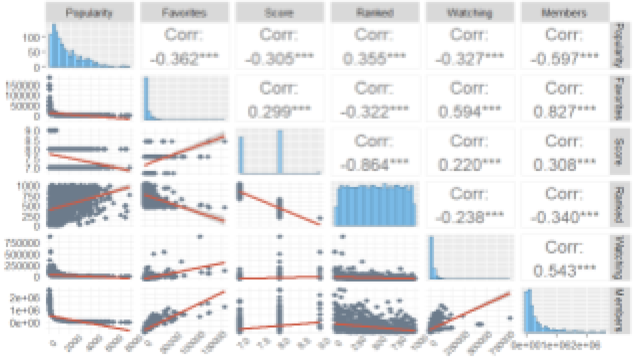
2 Dataset

Our dataset consisted of 57633278 instances. We experimented with a subset of instances because our personal computers couldn’t handle the whole dataset. The datasets that we used were a combination of two different data sources. The first source was titled amine.csv. This source of data includes the name of the Anime, an anime ID, a rank, score, popularity score, number of members, how many have watched the anime, air dates, how many members have favorited the anime, genre, source, type, and maturity level. The second source rating_complete.csv is comprised of a user id, anime id, and the rating that the user gave to the anime. Using these two datasets we were able to explore the different relationships between the variables in an attempt to create a recommendation system.

3 Data Analysis

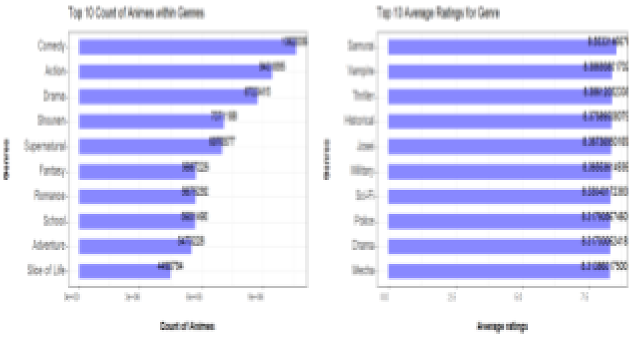
To do some data exploration we started with evaluating if there was any correlation between any of the variables. First we plotted a cor plot (Fig 1).

Fig. 1 Cor Plot



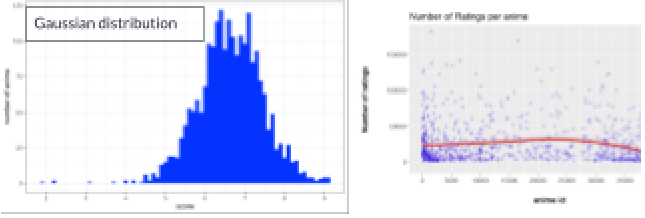
The cor plot was constructed out of the six numerical variables within the top one thousand ranked anime. The cor plot shows that as the score increases so do the members and watching. From this plot we determined that rank as well as score were two important variables. After understanding these important variables we wanted to look closer at different string variables and how they relate to the rank and score variables of the dataset. The first variable that was analyzed was Gere. For each anime there was a ranging number of genres, anywhere from one to over five. To understand better what genres were within the top ranked we created a function to separate the genres in a single line and then count them to give them a frequency. We also wanted to view the average score among the genres so in a similar matter we extracted the genres while also associating a rating for each pulled out genre. From evaluating genre it can be surmised that the genres that are most watched are not as highly rated. This indicates that genres may be popular but are rated lower when they are more popular. This relationship can be seen in Fig 2.

Fig. 2 Genre Plots



There were other variables that we performed the same tasks of extracting singular strings from a string that contains multiple strings as well as associating those strings with a rating. This process performed on Studios, Producers, Source, and Type gave small insite. In the case of Studios it was determined that the Studios that mass produced were not highly ranked. So when associating what studios produce better anime it is noted that the studios that do not produce in mass quantity and are more selective on what they produce will result in better rated anime. Following suit Producers that mass produce do not produce well quality anime. The lack of crossover between popularly watched producers and highly rated producers had such little crossover that we had to expand the number of producers to view twenty instead of just ten like the studios or genres. In evaluating sources there were more commonalities. To look further outside just the top ranked variables we expanded the dataset to the whole dataset and found that the results were similar. It resulted in sources like Manga and Original as similarly well viewed and well ranked. Indicating that these sources are good commonalities between the viewers and scores. In a similar fashion the type of anime resulted in highly watched as highly rated. Although not exact in rating the types like Movie and TV were switched in the larger exploration of the dataset, indicating that people will rate movies higher and they will be harsher to rating TV. In our last dive into exploring the data we checked if the score was well distributed across watchers. We graphed the amount of watchers vs the score as well as the users vs the amount of times they have rated to get a better understanding of the possible bias in the data.

Fig 3. Rating Distribution



From Fig 3 we can see that in the left graph the scores are skewed to the right with the highest concentration within a score of eight. From the right graph we can see that there are a number of users that rate often therefore each anime has a potential to have a well distributed rating. So this indicates that there overall users rate highly but the ratings are well distributed.

4 Method

Since our end goal was to create an anime recommendation system we implemented it using User-based collaborative filtering (UBCF) scheme. In the UBCF scheme, we try to predict user preferences for new items (called filtering) by collecting taste information from many users (collaborative). The UBCF algorithm works as follows. First it identifies a set of items rated by the target user. Second, it identifies which other users rated 1+ items in this set (neighborhood formation). Third we compute how similar each neighbor is to the target user using a similarity function. Fourth, we select the k most similar neighbors. Fifth, we predict ratings for the target user's unrated items using a prediction function. And finally we recommend to the target user the top N items based on the predicted ratings.

For the UBCF algorithm, we need a user to anime mapping which is defined by the rating provided by the user for each anime they have watched. Then based on the ratings provided by similar users we predict the top animes for the target user.

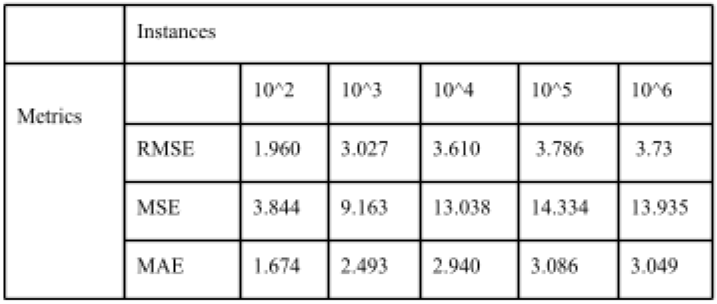
We try two approaches for the UBCF algorithm. The first approach involves recommending anime to the user using the mapping described before. The second approach involves addition of extra features to the user - anime mapping. We use data exploration to decide these extra features.

The pipeline for our approach is as follows. First we start by doing a data exploration of the dataset to identify the important features for a particular anime. Through analysis we find that 'Score' and 'Ranking' are the features of interest. Second, we preprocess the data by converting the categorical features into the numerical format. We then standardize and scale the numeric values so that every feature has equal weightage. Third, we create sparse matrices for both the approaches. Fourth, we use cross validation to train the recommender model.We experiment with different dataset sizes namely, 10^2, 10^3, 10^4, 10^5 and 10^6 instances. The algorithm uses cosine similarity as its default similarity metric. We use a 'hybrid method' that gives weightage to 3 methods, that are 'popularity' based, 'Random' based and 'Re-recommendation'. Description about the 3 methods are as follows,Popularity-based model recommends items that are popular among users, regardless of the user's preferences or past behavior. It is useful for new or inactive users who do not have enough data for personalized recommendations. Random-based model recommends items randomly, without considering any user or item information. It is useful for introducing diversity in the recommendations and avoiding over-reliance on popular items. Re-recommendation based model re-ranks the items recommended by other techniques based on additional information such as user feedback, social network, or item attributes. It is useful for incorporating user feedback and improving the relevance of the recommendations.

5 Evaluation

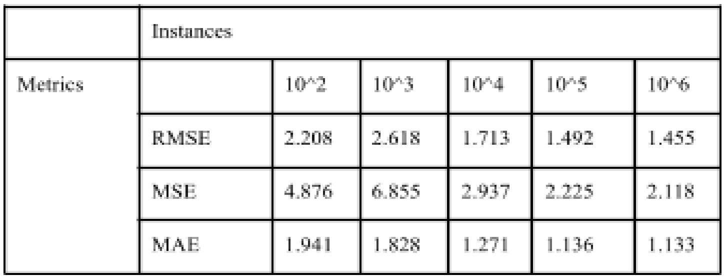
We use Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and Mean Squared Error (MSE) to evaluate the performance of our model. We use these metrics because under the hood the model outputs a rating for all the candidate animes for the target user. We can calculate the error by comparing the predicted rating to the actual rating. The baseline method was implemented by setting the model to 'Random' mode. For the hybrid approach we assign the following weights, 'popular': 0.6, 'random': 0.1 and 'Re Recommend': 0.4. The results can be summarized as below.

Table 1: Baseline results



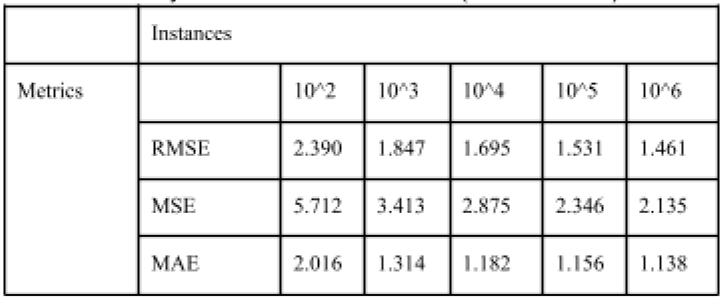
	Instances					
Metrics		10^2	10^3	10^4	10^5	10^6
	RMSE	1.960	3.027	3.610	3.786	3.73
	MSE	3.844	9.163	13.038	14.334	13.935
	MAE	1.674	2.493	2.940	3.086	3.049

Table 2: Hybrid recommender model (Without features)



	Instances					
Metrics		10^2	10^3	10^4	10^5	10^6
	RMSE	2.208	2.618	1.713	1.492	1.455
	MSE	4.876	6.855	2.937	2.225	2.118
	MAE	1.941	1.828	1.271	1.136	1.133

Table 3: Hybrid recommender model (With features)



	Instances					
Metrics		10^2	10^3	10^4	10^5	10^6
	RMSE	2.390	1.847	1.695	1.531	1.461
	MSE	5.712	3.413	2.875	2.346	2.135
	MAE	2.016	1.314	1.182	1.156	1.138

The observations are as follows. First, the Hybrid model performed significantly better than the baseline. We report the best RMSE scores of 1.455, MSE of 2.118 and MAE of 1.133. Second, It can be observed that increasing the number of instances reduces the RMSE but only to a certain extent. After 10^5 instances the RMSE doesn't change much in both cases. After this point probably feature addition is required. Third, the very low RMSE for baseline results when the number of instances are 10^2 is due to overfitting. Fourth, from Table 1, 2, and 3 it can be observed that adding extra features is not helpful as the RMSE value is the same in most of the cases.

Fig 4: RMSE vs Dataset Size

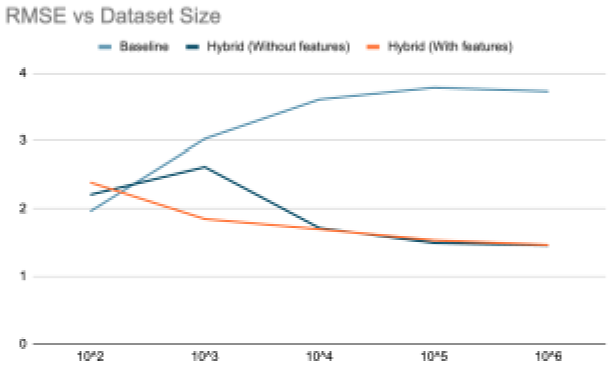


Fig 5: MSE vs Dataset Size

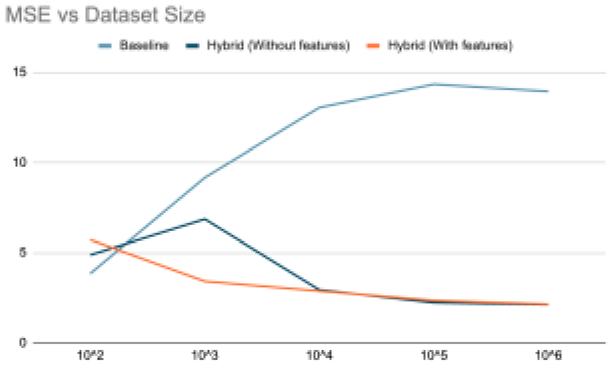
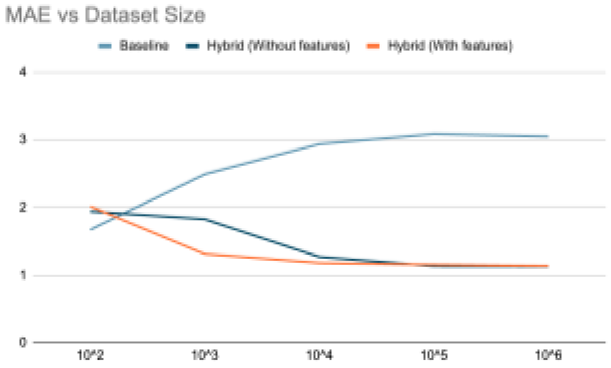


Fig 6: MAE vs Dataset Size



6 Discussion

On looking for related code implementations for the UBCF algorithm using additional features in R, we weren't able to find any code examples. Hence we incorporated the features to the best of our knowledge. We have used only one set of weights to train our model. We could experiment with different weight settings (or define a grid of weights) and test which is the best combination. We could also experiment with other similarity metrics such as 'Pearson correlation coefficient' or 'Euclidean distance'.

Our solution just incorporates 2 features namely 'Score' and 'Rank'. It'll be worthwhile exploring other important features such as reviews given by the user for a particular anime. This can provide more personalized recommendations to the end user.

7 Related Work

We already collected, filtered, and selected the most appropriate data to satisfy our analysis goal. Facing large data with dozens of variables, we have to realize our data, and find out which variables are more significantly influential to others, and would further affect our recommendation results. In this section, we discuss how the related research had been done previously, and figure out how we build our recommendation systems based on their experience.

From the related research, we realize that data visualization is a proper way to understand the structure, distribution and weight of the data. As a result we search various researches to find out how others explored data by visualization. We first want to explore the background of our data. We found that it is simple and easy to realize the ratio of a variable by treemap in [3]. In the research, it presented the wine quantity of all the types by treemap. The result is so clear that we think it is useful to do an initial understanding of the data. We then find out the distribution in genre and source by this method. And then [3, 4] showed that showing the relationships between the change of a variable against time passing by density plot is clear, so we use it to present the top 10 genre's quantity variation in these years. bar charts are common to compare the differences between each type [2, 6]. From [2, 5], we learn to realize the distribution of a feature, making sure whether the score is a normal distribution. And compared to doing correlation one by one, it is more efficient to do a correlation plot including all the important features[6].

To start building a recommendation system, the data should be prepared by selecting useful data, normalizing data, and then binarizing the data. Normalizing data could standardize numerical values and binarizing data could allow the recommendation system to work more efficiently. And then, developing a collaborative filtering system to find similarity in the features. The similarities would be combined and then fed into the recommendation system. Based on it, we used the predict function to identify similar items and rank them properly. Each rating here is used as a weight, and each weight is multiplied with related similarities[5].

8 Conclusion

Recommendation systems is one of the most popular machine learning applications recently. From our research, we find out the relationships between every anime, thereby adding into the recommendation systems and get the result. We report good values of RMSE on the dataset. Apart from accomplishing our goal, spreading the fun of anime to more people, the procedure of building recommendation systems could also be applied to other fields in our daily life. The connection and quick flows of information could not only make more benefits to people, but also make people's work more efficient.

9 Distribution of Work

Throughout this project all three members have done well to contribute to the project. Wan Qi has done research into related work as well as created visualizations. Victoria was able to explore the dataset and create visualizations and determine data needed for the recommendation system. Victoria also organized, formatted and compromised the presentation, reports, and git. Finally Abhibha has worked to create the recommendation system.

REFERENCES

[1] Valdivieso, Hernan. "Anime Recommendation Database 2020." Kaggle, 13 July 2021, https://www.kaggle.com/datasets/hernan4444/anime-recommendation-database-2020/code?select=anime_with_synopsis.csv.

[2] Erik, Bruin."Movie recommendation systems for TMDB." Kaggle, https://www.kaggle.com/code/erikbruin/movie-recommendation-systems-for-tmdb/report

[3] Chaitanya. "Wine Analysis + Recommendation." Kaggle, 7 July 2021, https://www.kaggle.com/code/chaitanya99/wine-analysis-recommendation/report

[4] Anastasiia, Chebotina. "Product Analytics and Recommendation Systems - R." Kaggle, https://www.kaggle.com/code/chebotinaa/product-analytics-and-recommendation-systems-r

[5] Meer, Nagadia. "Movie Recommendation System in R." Kaggle, https://www.kaggle.com/code/meetnagadia/movie-recommendation-system-in-r#Collaborative-Filtering-System

[6] Rodrigo, Serrano. "Movie Content-Based Recommendation System." Kaggle, https://www.kaggle.com/code/rodserr/movie-content-based-recommendation-system#exploring-data

[7]Valdivieso, Hernan. "Anime Recommendation Database 2020." Kaggle, 13 July 2021, https://www.kaggle.com/datasets/hernan4444/anime-recommendation-database-2020/code?select=anime_with_synopsis.csv.