CSE 158 Assignment 2
Fall 2024

Introduction: MyAnimeList.net is a popular website for fans of Japanese animation, where users can maintain information about anime they have watched, browse recommendations and reviews, and see historical and current popularity rankings for shows. Since MyAnimeList's foundation in 2006, its number of users has grown steadily, with over 12 million visitors and over 13 million forum users as of 2022 (according to the site). The most basic user interactions involve giving ratings from 0-10 to anime (which contributes to an anime's overall ranked popularity within the site), and denoting watch status for a show (watching, completed, plan-to-watch, dropped). With this data and techniques from this class, we can build models to predict how users may interact with anime.

Dataset:
Although MyAnimeList.com has millions of user interactions stored, the dataset used for this analysis was mined 6 years ago (found on Kaggle), and may not be entirely reflective of trends that may have emerged since 2018, when this data was collected.

Within the dataset I used, there were subsets containing anime information (title, genre, duration, score from 0-10, number of episodes, popularity within the site, etc.), user information (i.e. username, registration date, birth date, gender, location, etc.), and user anime lists (i.e. user interactions with an anime, most importantly, their score and watch status). The dataset included information for 108,711 users and 31,284,030 user interactions. To reduce the size of the user interactions dataset, and to hopefully gain more useful information about user ratings, I filtered the data to only include interactions where users had completed the anime, which reduced the number of datapoints to 19,946,276. In order to decrease the sparsity of the dataset further, I removed users who had fewer than 10 ratings and anime with less than 50 ratings. My training and testing subsets were drawn from this filtered dataset, which hopefully helped improve the models.

I. Exploratory Analysis
To get an idea of how ratings are broadly distributed across the entire dataset, this is a plot of the ratings distribution. Ratings seem to generally trend more positive than negative, with an average rating of about 6.8489, and most ratings falling roughly in the 6-8 range.

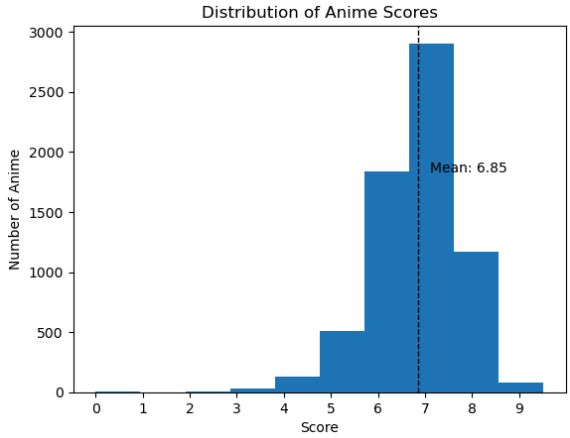

Figure: Distribution of anime scores across all anime, mean score is about 6.85.

I wanted to examine whether any user or item traits could serve as predictors for the scores an anime received. For example, would a user's age or gender, or the year of an anime's release determine what score an anime was given? The user dataset contained an average score for each user, which I used to perform the following analyses.

While around 500 users did not specify their gender (and the only options available to users were the binary male or female), the number of female users is approximately 37330, accounting for about 34.5% of users who reported a gender, while the number of male users is around 70880, around 65.5% of users. For female users, the

average score was about 7.8481, while for male users, the average score was 7.6964. Based on these averages, it seems that female users tend to rate slightly higher on average than male users, but since there were fewer female users in the dataset than male users, this may not be an entirely accurate representation.

I then grouped users by their ages (integer values between 16 and 55, inclusive), and plotted the average scores (y-axis) given by each age (x-axis).

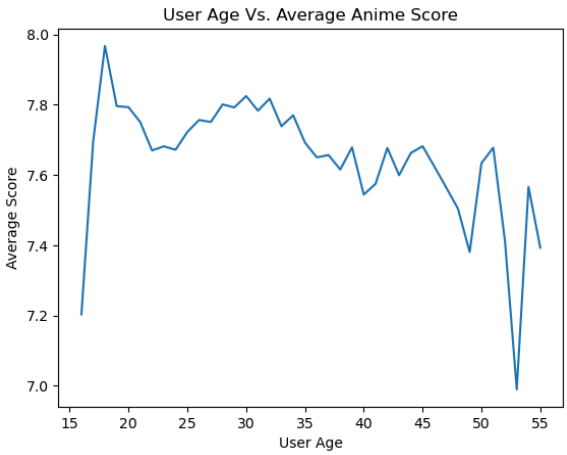

Figure: Average anime score distribution across user ages

When looking at the average score given by users of the same age, the average anime score fell between about 7.0 and 8.0 across all ages. There appears to be a generally declining trend for average score as age increases, although interestingly, 16 year-old users (the youngest age allowed on MyAnimeList) had an average score of about 7.2033, after which the average score jumped to around 7.6957 for 17 year-old users. The lowest average score of about 6.989722 was given by 53 year-old users.

I also plotted an anime's average rating versus the anime's release year to see if there were any long-term trends. I was curious whether the nostalgia effect, when older movies tend to have higher rankings because they are sought out by users rather than being recommended, might appear, but it seems like this may not be the case. Based on this plot, it seems like more recently released anime tend to have higher average scores than older anime, though with a fair amount of fluctuation.

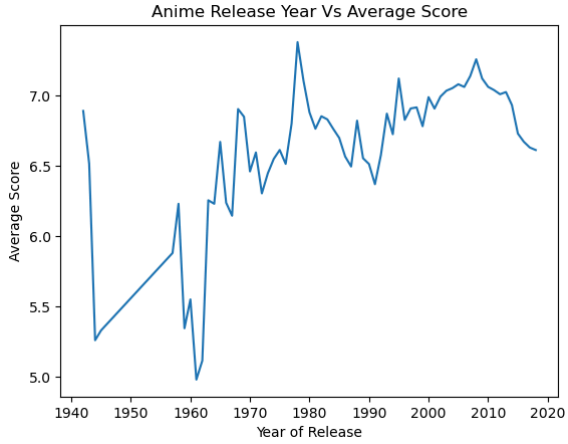

Figure: Average anime score distribution across year of anime release

II. Identifying a Predictive Task

One task I wanted to study with this dataset was predicting whether or not a user will have watched a given anime. As baselines for this task, I tested two naive models. The first model randomly predicted 0 (unwatched) or 1 (watched) for a given (user, anime) pair. When comparing these predictions to the actual watched/unwatched values, this model had an accuracy of 0.49765. The second model predicted that a user had watched an anime if it was in the set of animes accounting for 50% of all interactions in the training set (same as the baseline model for Assignment 1), and this resulted in an accuracy of 0.73305.

For my models beyond the baselines, I wanted to test a variety of features, including the sets of all animes watched by a user, the number of anime completed by each user, the user's age, the anime's popularity within the site, the anime's average rating within the site, and the time since the anime's release. All of these features were extractable from the user information or anime information dataset respectively.

Additionally, given that a major aspect of this dataset is the score a user gives to a particular show, I also wanted to predict what score a user will give to an anime. As baselines for this task, I tested a few naive approaches. First, I generated random predictions from 0-10 for each (user, item) pair, which resulted in an MSE of 21.70265. I then tried simply predicting the user's average rating (based on all ratings that user had in the training set) for all (user, anime) pairs, which resulted in an MSE of around 6.3138. Lastly, I tried predicting the anime's average rating (based on all ratings it had in the training set) for all (user, anime) pairs, which resulted in an MSE of about 8.323.

With these baselines and features in mind, I wanted my models for watched prediction and rating prediction to outperform these accuracy and MSE values.

III. Model Development

Watched Prediction

To build a model to predict whether an item was watched or not, I used a similar approach of generating negative samples for each user in the training and validation sets by randomly sampling one anime they had not watched from the pool of all anime present in the training set. I used a training set of 200,000 total observations (100,000 positive and 100,000 negative), and tested the model using a validation set of 20,000 total observations (10,000 positive and 10,000 negative).

Building off my approach to read prediction from Assignment 1, I first implemented a model that generated predictions based off a user's Jaccard similarity to other users who had watched a given show, and the popularity of that show (i.e. the number of total users who had watched that show in the training set). Using this model to generate predictions for the validation set, by experimenting with different thresholds for maximum Jaccard similarity and popularity, the model had an accuracy of about 0.71775 (14355/20000 correct predictions).

I then implemented a model that used cosine similarity (between sets) instead of Jaccard similarity, which did not significantly improve the accuracy. Using the same thresholds for maximum similarity (but using cosine similarity instead of Jaccard) and popularity, the model had an accuracy of about 0.7162, similar to the previous model.

I also planned to train a Support Vector Machine to generate read predictions. I attempted to use the SVC class in sklearn to build the model, generating a one-hot encoding to represent the user IDs as numerical values. However, training the model proved very time-consuming due to the high dimensionality of the encoded dataset, so I decided to attempt different models instead.

I then implemented a model using logistic regression in order to generate watched predictions. I used the Logistic Regression class in scikit-learn to train the model. Non-numerical features had to be encoded as numerical values using label encoding, another function performed by scikit-learn. Once encoded, I first used only (user, item) pairs as features, and when testing this model on a validation set of 20,000 pairs, the model had an accuracy of 0.5266.

This accuracy is not much better than the accuracy when randomly predicting watched or unwatched, so I tried incorporating additional features into the model. Each user had a total number of completed anime, and each anime had an overall popularity ranking within the site. Using a scaler to normalize these values, then adding these features to the logistic regression model resulted in an accuracy of 0.6987, which is an improvement over the original logistic

regression model. Finally, I added the anime's average score (across all users of the site) as a feature, which resulted in an accuracy of 0.6998.

Beyond this, adding more features did not seem to improve the model's performance. I tried adding the user's age as a feature to this model, and the accuracy dropped slightly to 0.6965. Likewise, adding the number of years since an anime's release (relative to 2018, the latest release year included in the dataset) resulted in an accuracy of 0.6935. Based on experimenting with different features, the strongest logistic regression model for watched predictions used user, anime, number of anime completed by the user, anime popularity, and anime score as features.

Logistic Regression Classification Report:
Accuracy: 0.6998

	precision	recall	f1-score	support
0 (unwatched)	0.73	0.64	0.68	10000
1 (watched)	0.68	0.76	0.72	10000

Table: Logistic Regression results on validation set using (user, item, num_user_completed, item_popularity, item_score)

b. Rating Prediction

Next, to attempt to improve on the baselines for rating prediction, I trained a latent factor model using the Surprise library's SVD implementation to perform matrix factorization. Using the default parameters, the model initially had an MSE of 6.7559. After using Surprise's GridSearchCV and RandomizedSearchCV, I was able to identify more optimized values for lambda, learning rate, and number of factors, which resulted in an MSE of 5.8237. The process of optimization is summarized in the following table.

Parameter values and resulting MSE

	Default	Trial 1	Trial 2	Final
reg_all (lambda)	0.02	0.07	0.115	0.15
lr_all (learning rate)	0.005	0.008	0.01	0.015
n_factors ()	100	3	1	1
MSE	6.7559	6.1133	5.8845	5.8237

Table: Summary of parameters tested to improve MSE of SVD model for rating prediction

IV. Existing Literature

This dataset was found on Kaggle, where it was uploaded by user Azathoth around 6 years ago. The uploader already performed their own analysis of this dataset, and documented their work on this Github.

Additionally, Kaggle has many similar datasets scraped from MyAnimeList at different points in time, and many recommender system projects have been built off such datasets. Most other projects I saw online focused on rating (score) prediction. One project from 2016 by Aaron Gokaslan, Ahmad Najeeb, Haris Chaudhary, and Nicholas Hartmann tested a variety of approaches, including a factorization machine, MLlib via Spark through AWS, and Simon Funk SVD. Their models used ratings and anime characteristics to predict user scores, and they found that using a model built with the Incremental SVD Julia package had the best results, with a MSE of 7.29 (RMSE of 2.7). They noted that MyAnimeList's 10-point rating scale may make it more difficult to generate predictions, compared to a typical 5-point scale. Whereas in a 5-point scale, it is easier to claim that a user should give a rating of about 3 to something they find 'average,' on a 10-point scale, a rating of anywhere between 4-8 could arguably be average. Having a greater range of

values within which predictions must be made is likely one potential issue for models.

Another project by Xuan Khang Pham used a combination of filtering approaches to generate predictions. He incorporated both content-based filtering and collaborative filtering to build a recommender, documenting a RMSE of about 1.2977 (MSE = 1.68402529) for a model using only content-based filtering and a RMSE of about 1.782 (MSE = 3.175524) for a model using only collaborative filtering. His model used Ridge regression and the Surprise SVD model to perform matrix factorization.

Generally, as discussed in class, there are a wide variety of models that can be used to generate ratings and watched (or equivalent measures of interaction, i.e. read for a book, bought for a product, etc.) predictions. For rating predictions, popular approaches include matrix factorization, or methods that use aspects of both content and collaborative filtering. Deep learning methods are also becoming increasingly popular to perform prediction tasks.

For anime-specific recommender systems, many projects I found online generated predictions using methods very similar to those we have discussed in class, i.e. using cosine similarity or matrix factorization methods. Most approaches to anime recommendation focused on identifying similar shows based on characteristics of the anime, like its genre or title (seeking to identify sequels or spin-off shows). I did not see many projects that generated watch predictions rather than rating predictions for similar datasets, but generally, it seems like Bayesian networks, clustering algorithms, and regression models are all popular choices to model and predict user-item interactions.

V. Results and Conclusion

Since I did not find many existing projects about watched predictions for similar anime datasets, it is difficult to judge how my model compares to others for this task. The best accuracy among the models I implemented was actually the baseline, which predicted true if the anime was in the set of animes accounting for 50% of all interactions in the training set. This seems to suggest that the popularity of an anime among users is a relatively strong metric for whether an anime has been watched in general, but the higher accuracy may also be due to overfitting or bias in the training dataset.

Among the other models I tested, it seems like Jaccard similarity between users who had watched an anime, combined with the relative popularity measure, was a fairly effective heuristic for this task. Logistic regression did not seem to capture any patterns that could predict whether a user had watched an anime or not as effectively. Still, based on the accuracies obtained by including various features from the dataset in logistic regression models, the most informative features seem to be the number of anime a user has completed and the anime's score. As might have been suggested by visualization of anime rating distributions across user ages and time since an anime's release, these features were not particularly helpful for predicting whether or not a user had watched a certain anime.

The logistic regression model might be improved further by exploring different features, such as how many (if any) awards an anime has won, the genres of an anime and all the genres watched by a user, the time since a user was last active, etc. With more feature engineering, a logistic regression model's performance could likely be improved, or different models could also be considered for watched prediction. An SVM or matrix factorization approach would

likely be strong alternatives for this task, although I did not implement them for this assignment.

For rating prediction, the latent factor model only slightly outperformed the baseline of predicting the user's average rating in terms of MSE. This could be due to a number of factors, including the overall sparsity of the dataset, or the broader 0-10 rating scale instead of the more-common 0-5 scale. Within the dataset, many users have relatively few ratings even after filtering, so predicting the average can reduce overfitting, while advanced models like SVD might be more affected by noise or outliers. Since the rating distribution in the overall dataset seemed to be most concentrated in the 6-8 range, it makes sense that predicting user averages would yield fairly strong predictions for ratings. Since MSE also heavily penalizes larger errors, predicting the average will translate to fewer extreme predictions and thus reduced MSE, even if the predictions are less accurate.

In comparison to the other rating prediction models found online, it seems that using a combination of content filtering and collaborative filtering might be more effective for generating the most accurate predictions. Likewise, incorporating more advanced approaches like factorization machines, neural networks, random forests, or SVD++ with implicit feedback could likely yield viable models for rating prediction.

Going forward, it would be interesting to perform further analysis of what features could be useful predictors of scores. Analyzing temporal dynamics or review text was something I considered for this assignment, and although I ultimately did not incorporate these ideas, this could be a future direction for anime rating prediction.

The Jupyter notebooks I used to develop these models and perform exploratory analysis can be found on this Github.

VI. Sources cited

Matěj Račinský, "MyAnimeList Dataset." Kaggle, 2018, doi: 10.34740/KAGGLE/DSV/45582. https://www.kaggle.com/datasets/azathoth42/myanimelist/data

Pham, X. K. (2024, November 2). Hybrid recommendation system. Kaggle. https://www.kaggle.com/code/xuankhangpham/hybrid-recommendation-system/notebook

Gokaslan, A., Najeeb, A., Chaudhary, H., & Hartmann, N. (2016, May 14). Recommendation System for Anime. cs1951arecommender. https://cs1951arecommender.wordpress.com/

CooperUnion. (2016, December 21). Anime recommendations database. Kaggle. https://www.kaggle.com/datasets/CooperUnion/anime-recommendations-database/code