Predicting a User's Mean Rating and Anime Consumption Using Review Data from MyAnimeList

Joseph Hsieh

1. Dataset

MyAnimeList is a website where users can track and rate the anime that they have watched, similar to the website IMDb. Users can add anime to their anime list and rate each anime on a scale from 1 to 10. Users can also write reviews for anime that they have watched. For this project, I scraped 50000 reviews and the user data for every unique reviewer.

In each review, the user provides six ratings—a "story" rating, an "animation" rating, a "sound" rating, a "character" rating, an "enjoyment" rating, and an "overall" rating. The user must write a minimum of 251 words in order to publish their review.

I scraped the top 40 reviews (sorted by the most helpful first) of the top 1250 most popular anime on MyAnimeList. Within the top 1250 most popular anime, there were some anime that had less than 40 reviews, which were omitted in the dataset. I scraped the 41st to 80th reviews of the top anime ranked by popularity until I received 50000 entries. I also scraped the user data for each user in my review dataset for the predictive tasks. 1282 out of 24526 unique users had deleted their profiles, so their data was not available.

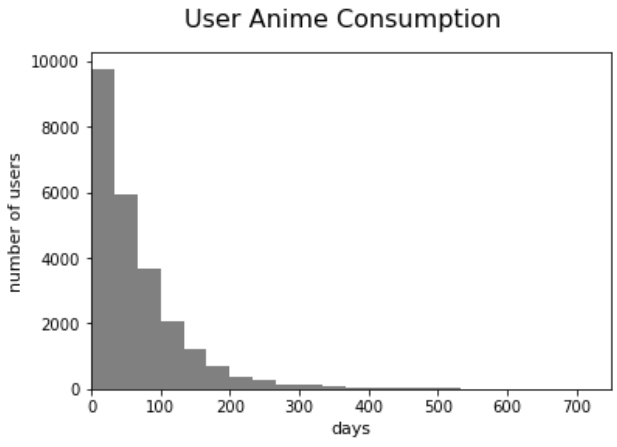

Figure 1. This plot shows the distribution of time watched in days among the reviewers. Data points above 700 days were omitted from the graph.

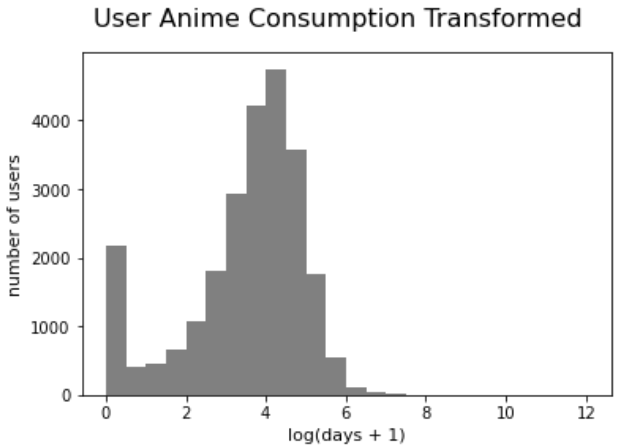

Figure 2. This plot shows the distribution of the log of time watched in days among the reviewers.

In **Figure 1**, we can observe that the distribution of anime consumption in days heavily skews right. This task would be difficult to predict for, and measuring our prediction accuracy could be problematic due to extreme outliers. However, in **Figure 2**, we can observe that the plot has a somewhat normal distribution if we exclude the "0" data points (which are a result of the lack of available data), though there are still some extreme outliers (which are likely false data).

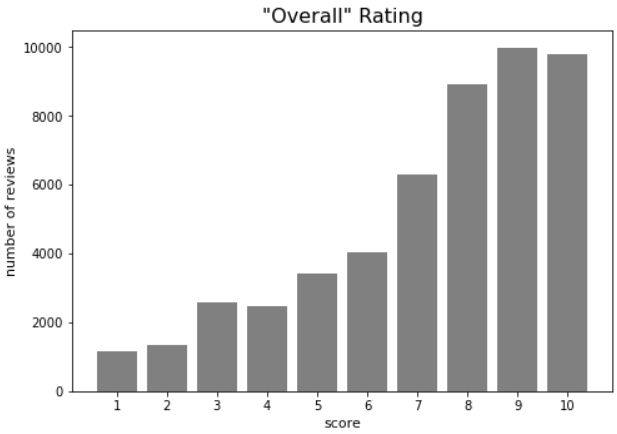

Figure 3. This plot shows the distribution of "overall" rating scores among the reviews.

In **Figure 3**, we can observe that the distribution of review ratings skews to the left. In general, most

rating distributions on MyAnimeList skew to the left, especially so in this dataset since I pulled reviews from the top 1250 anime by popularity. The average review rating was 7.31988.

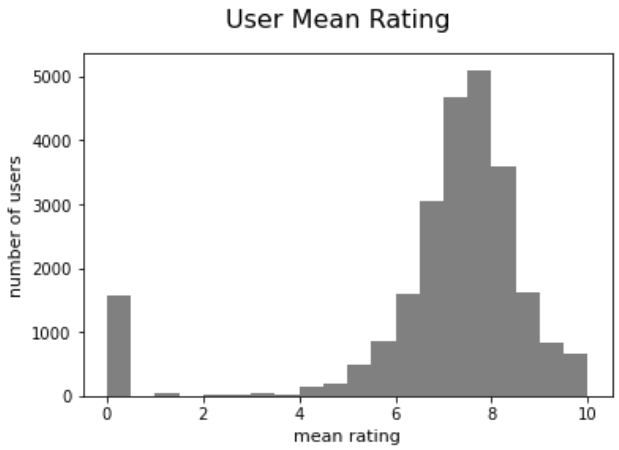

Figure 4. The plot shows the distribution of mean ratings among the reviewers.

In **Figure 4**, we can observe that the reviewers' mean ratings follow a fairly normal distribution if we exclude the "0" data points, The average mean rating among reviewers with accessible data was approximately 7.45185.

2. Predictive Tasks

There are several predictive tasks that can be achieved using the review data since I am scraping the data myself. Given each review entry, a few predictive tasks I considered included:

- the user's anime consumption (i.e. the number of days that the user has spent watching anime)
- the user's average rating across their entire anime list
- the overall rating of the review (without using ratings as a feature in the predictor)
- the genre categories of the anime
- the runtime (episode length multiplied by the number of episodes) of the anime
- the year or season when the anime was released
- the number of friends that the user has

As the title of this paper indicates, I chose two predictive tasks: to predict the user's mean rating and the user's anime consumption (log transformed).

I chose these predictive tasks because, anecdotally, the way a review is written can be indicative of the type of anime watcher a user is. Some users rate anime generously, whereas others are extremely harsh with their ratings. Some users watch anime casually, while others treat it as an extreme sport. Through the data visualization, we can observe these phenomena: some users have hundreds of days watched, and there is even a visible amount of users on the histogram that have a mean rating between 1 and 1.5. (1 is the lowest possible rating).

The advantage of scraping the top 40 (or 80) reviews of the most popular anime is that though anime "elitists" are a small minority, they are also a vocal minority. A great example of this is the second season of Code Geass (anime_id: 2904), where the mean overall rating of the top 40 reviews is 6.85 and the mean rating across all users is 8.91. The dataset contains a wider range of "opinions" (in review text) rather than being centered around the global mean. We can also observe this phenomenon through the distribution of ratings, as we see review ratings are more heavily skewed.

I will use features based on text data written by the user to predict the user's mean rating and their anime consumption (log transformed). For both models, I will preprocess the text data by removing punctuation and removing stopwords. Stopwords are common words that provide little predictive value due to the lack of connotation and frequent usage of the word.[1] I will use a matrix of Term Frequency - Inverse Document Frequency (TF-IDF) scores. TF-IDF is defined as:

$$tfidf_{t,d,D} = n_{t,d} \cdot log\frac{N_D}{N_{D,t}}$$

where $n_{t,d}$ is the number of times term t appears in document d, N_D is the total number of documents, and $N_{D,t}$ is the number of documents that contain the term t.[1]

3. Models

3.1 Predicting Mean Rating

For this prediction, I removed all user data points that did not have a mean rating. Since the possible user mean ratings are non-discrete (continuous), I chose to use linear regression on the matrix of TF-IDF scores and MSE (mean squared error) for assessing the accuracy of the model.

Linear regression minimizes the objective:

$$||y - A\theta||_2^2$$

and has a solution in the form of:

$$\theta = (A^T A)^{-1} A^T y$$

Mean squared error is defined as:[2]

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Mean squared error heavily penalizes large errors since it takes the square of every error. Since the mean rating data follows a somewhat normal distribution, mean squared error is an appropriate measure for accuracy.

I first chose a baseline model of always predicting the "overall" rating given in the review. This was not a very good model (with an MSE greater than 5), since people tend not to write reviews for things they find average.

I came up with another baseline model of always predicting the average user mean rating. This resulted in a much lower MSE than the previous baseline model and was a reasonable baseline to test against. Since the distribution of ratings is close to a normal distribution, the trivial predictor is actually a decent predictor.

To test for overfitting, I made a train/test split by randomly shuffling the dataset and taking the first half as the training set and the second half as the testing set.

Table 1. Comparison of MSE between baseline and linear regression

	Predict average mean rating	TF-IDF Linear regression
Train	1.39333	1.51281e-11
Test	1.36635	1.99198

The linear regression model greatly overfit the data, resulting in a miniscule MSE for the training data but a MSE worse than the baseline model's. This is likely due to the fact that there are thousands of features in the TF-IDF matrix, allowing the objective function to be easily minimized with parameters that do not necessarily represent the actual correlation between the feature and the output.

In order to address the overfitting issue of linear regression, we can use Tikhonov regularization. Tikhonov regularization minimizes the objective function:

$$||y - A\theta||_2^2 + \lambda||\theta||_2^2$$

where λ can be described as the regularization strength. A larger lambda will penalize the model more for overfitting the training data, as the L2 norm of theta is weighted more heavily. The solution to the optimization problem can be written as the following:

$$\theta = (A^T A + \lambda I)^{-1} A^T y$$

Table 2. Comparison of MSE between baseline and Tikhonov regularization

	Predict average mean rating	TF-IDF Tikhonov regularization $\lambda = 1$
Train	1.39333	0.52252
Test	1.36635	1.05939

The model still slightly overfits the training data, but with the regularization constant $\lambda = 1$, the model performs better than the baseline model on the test set.

Increasing lambda (to a certain point) should even further decrease the MSE in the test set.

Table 3. Comparison of MSE between models using different values of lambda in Tikhonov regularization

	TF-IDF Tikhonov regularization $\lambda = 5$	TF-IDF Tikhonov regularization $\lambda = 10$
Train	0.79809	0.88831
Test	1.00721	1.01297

The Tikhonov regularization model can probably be improved upon even more through spell-checking, using bigrams, et cetera when preprocessing the corpus for the matrix A.

3.2 Predicting Days Transformed

For this prediction task, I removed all outliers where their untransformed time watched was greater than 10000 days. As seen from the previous prediction task, linear regression will greatly overfit to the

training data due to having too many features. Because of this, the first model I considered is Tikhonov regularization, which should take into account the overfitting of the training data.

For the baseline model of this prediction task, I chose to predict the average transformed days among the reviewers. Since the transformed days follow a fairly normal distribution spare a few outliers, this is a decent baseline predictor.

Table 4. Comparison of MSE between baseline and Tikhonov regularization

	Predict average days watched	TF-IDF Tikhonov regularization $\lambda = 1$
Train	1.37914	0.65193
Test	1.35907	1.01296

Though the Tikhonov regularization still slightly overfits the training data, the MSE of Tikhonov regularization on the test data is still significantly better than predicting the average.

I attempted to improve this model by further processing the corpus by allowing only ASCII characters in each feature of the model. This removed any words that contained non-ASCII characters from the feature vector.

Table 5. Comparison of MSE between ASCII only processing and Unicode using TF-IDF, Tikhonov regularization $\lambda = 1$

	ASCII	Unicode
Train	0.64919	0.65193
Test	1.29121	1.01296

Interestingly, the accuracy drops on the test set for the ASCII model, even if we try to tune the regularization parameter. Using different values of lambda, I could only get the ASCII model to have a testing MSE of around 1.22. We could infer that seemingly unhelpful text data (i.e. text that is not in standard English) can actually be a useful predictor when analyzing online text data.

4. Related Literature

There isn't very much related literature regarding my specific prediction tasks, probably due to the fact that they are somewhat useless predictions; i.e. the prediction tasks are just statistics that can be calculated directly given enough data.

Many other studies used similar preprocessing techniques, such as removing stopwords[2] and using TF-IDF to create their features.[1] They also used other techniques that I didn't implement, such as removing sentences with over 50 words[2] and considering stemming and lemmatization during preprocessing.[1]

Text mining is often used in classification problems. One study used text mining of accident reports to classify the cause of a construction accident, and proposed an ensemble model based on other existing models by optimizing for and weighting their outputs.

Another common prediction task is to predict user ratings on specific items through the use of text mining. One model that has been proposed is the HFT model, which stands for "Hidden Factors as Topics." HFT combines latent factors in item ratings with latent topics in reviews.[3] Essentially, it combines the latent factor model with the Latent Dirichlet Allocation, or LDA.[2] The model is useful for sparse data, as it extracts information from review text.

Another model that has been proposed for the prediction of user ratings on items is RAS (Ratings are Sentiments), which builds upon the HFT model. The authors describe the model as a way of mapping ratings to the "sentiment distribution probability space," connecting review texts to the rating scores of users or items. They also found their model to be helpful in sparse distributions and that text mining can help build a model in these situations.

A similar problem involving text mining and user rating that has been studied is Latent Aspect Rating Analysis, in which the model proposed, the Latent Rating Regression model, decomposes text review data that is provided with an overall rating into latent ratings of some given aspects.[4]

The papers on HFT, RAS, and LRR all found more advanced ways of using text mining in recommender systems. They all found that evaluating review data on each of their models can significantly improve the prediction of product ratings by fitting users and items. From my experiments, I also found that using

review data can be a good predictor for user behavior.

5. Conclusions

The idea of my features and predictive tasks was to experiment with how much information text review data can provide about an individual. There is similar research which suggests that text review data can be a powerful resource for predicting user behavior.[3][4] My models demonstrate that there are trends in review text that can loosely predict a user's behavior regarding their consumption and ratings of anime. Furthermore, I found that traditional natural language processing techniques may not always be applicable in an online context, especially when analyzing text data that is often written informally,

My linear regression model failed due to overfitting the matrix of TF-IDF scores, as there were too many features in the matrix. Tikhonov regularization was able to solve this issue, though despite fine tuning the regularization strength, it still overfit the training data. However, using the TF-IDF matrix with Tikhonov regularization was still significantly better than using the base models, demonstrating that there is a correlation between features found in text and the anime consumption and mean rating of a user. Review text data can be a useful source for creating features when predicting user behavior.

6. References

[1] Zhang, Fan, et al. "Construction Site Accident Analysis Using Text Mining and Natural Language Processing Techniques." *Automation in Construction*, vol. 99, Mar. 2019, pp. 238–48. ScienceDirect, doi:10.1016/j.autcon.2018.12.016.

[2] Yu, Dongjin, et al. "Rating Prediction Using Review Texts with Underlying Sentiments." *Information Processing Letters*, vol. 117, Jan. 2017, pp. 10–18. ScienceDirect, doi:10.1016/j.ipl.2016.08.002.

[3] McAuley, Julian, and Jure Leskovec. "Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text." *Proceedings of the 7th ACM Conference on Recommender Systems*, Association for Computing Machinery, 2013, pp. 165–172. ACM Digital Library, doi:10.1145/2507157.2507163.

[4] Wang, Hongning, et al. "Latent Aspect Rating Analysis on Review Text Data: A Rating Regression Approach." *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Association for Computing Machinery, 2010, pp. 783–792. ACM Digital Library, doi:10.1145/1835804.1835903.