Lisa Hu
17:610:564 Machine Learning
Fall 2021

Predicting Genres for Asian Dramas using User Reviews

1. Mission Statement

1.1. Introduction and Goals

Genres are categories or labels often used to indicate the form or content of an artistic work, such as a television show. Many websites that list and display such works include genre labels as content indicators to allow users to filter their searches. One such website, MyDramaList, compiles information on Asian dramatic television series, or dramas. On this website, volunteers update Asian drama information, and users can rate and review dramas.

The goal of this project is to use the text of user reviews to predict the genres of a drama. Through natural language processing (NLP) the corpus of the reviews will be transformed into numerical features, which will then be fed into machine learning models to predict whether a drama is or is not a genre. A battery of such models will be run, resulting in boolean predictions for each genre for each Asian drama.

1.2. Data

The dataset containing Asian drama information was collected from the MyDramaList website in November 2021, using a web scraper I wrote in Python. The dataset consists of over 12,000 Asian dramas that aired between the years of 2000 and 2020, inclusive. Much of the drama information on MyDramaList is from user contributions, where volunteers often collect the information from official websites and add or edit them on the website. Features in the total dataset include episode duration, number of episodes, main cast, crew (directors and screenwriters), audience rating, number of watchers, synopsis, reviews text, tags, and country of origin.

Although I had originally planned to look at more features than only user reviews, time restraints prevented me from incorporating drama synopses and cast as additional features. However, my intentions are reflected in the null-value filtered dataset, which eliminated objects without genre information, reviews, a synopsis, and a cast list.

Target genres were chosen considering their prevalence in the dataset. Genres that were too few in number were either dropped or combined with similar genres. Genres dropped were low in number, specific to a certain country of origin, or not similar enough in content to be combined with another genre. Of the 35 genres in the filtered dataset, 16 such genres were completely dropped. Three genres were combined with existing genres with larger numbers, where the original larger genre labels were retained: political with drama, war with historical, and sitcom with comedy. Low-numbered genres school and youth were combined, and supernatural, fantasy, and horror were combined. The final list consists of 16 target genres.

The size of the dataset after filtering for non-null values in the above features and genres was roughly 4,700 objects.

1.3. Expectations and Assumptions

Based on previous studies using text analysis to predict genre information, I expected to achieve an average accuracy rate of between 60% and 75%, and an average Jaccard similarity score of about 50%. These predictions are based on two studies. Kumar et al. attempted to classify songs into exclusive genres using their lyrics, achieving accuracy rates of between 60% and 75%.[1] A student paper trying to predict movie genres using their synopses, where the genre labels were not exclusive, achieved a Jaccard similarity score of approximately 50%.[2] Although my project differs from those two in the type of text analyzed and the size of the dataset, I expect the results will be roughly similar.

The assumption is that a model trained on numerical features extracted from the review texts are actually representative of genre labels. This assumes that there are relationships between actual drama content and genre labels, where genres do in some way describe drama content; drama content and reviews, where reviews in some way reflect drama content; and therefore reviews and genres, such that the genre of a drama can be determined from its reviews.

1.4. Concerns and Potential Consequences

Web scraping, the method by which I obtained this dataset, can be ethically questionable. Although the scraping in question was not explicitly banned by MyDramaList, through the process of scraping, I could have quite easily collected user data, such as their comments and reviews. However, since the website was no longer providing tokens for API access, there was no other way for me to collect the data. In the dataset itself, I only saved information I deemed necessary and left out all usernames.

There is a possibility of bias in the project outcomes, as the genre labels are determined by human contributors on MyDramaList. However, I do not expect this project to have wide-ranging social consequences. It will likely not perform well enough to provide substantial support in an argument to replace human genre-labelers. In addition, since the genres on MyDramaList are mostly added by unpaid volunteers, even if they are replaced by some machine learning algorithm, the volunteers would not face negative economic impacts.

2. Methods

2.1. Features and Natural Language Processing

My project will be attempting to predict Asian drama genres based on user reviews. Since my project will be looking at text data, natural language processing (NLP) is required. The user reviews were processed using both term frequency-inverse document frequency (TF-IDF) and word2vec word embeddings.

TF-IDF utilizes the bag-of-words logic, where the semantic structure of a document does not matter. Exactly as its name suggests, TF-IDF is the product of the raw frequency count of a term multiplied by the inverse of the document frequency, how many documents contain the term.[3] All terms in a document are assigned a TF-IDF value, a ratio that represents the relative frequency of a unique term in a given document, compared to its frequency in the entire set of documents. The feature

set returned by sklearn's TfidfVectorizer was a matrix containing the TF-IDF values for each term for each document.

Word2vec uses a simple, single-layer neural network to produce word embeddings. Word embeddings are vector representations of words, where words located close together in the vector space are expected to be similar in meaning.[4] Unlike TF-IDF, word2vec considers context, with each term fed into the neural network with a small window containing the words around the term. The result is a vector for each term, with as many dimensions as there are nodes in the hidden layer, with the value for the dimensions being the weights of the hidden layer nodes. From gensim's Word2Vec, I obtained the vector value for each term. I averaged the vector values for each term for each document, again obtaining a matrix.

2.2. Models Considered

For the classification task, I considered and tested logistic regression, random forest, and Gaussian naive Bayes. I did not test k-means or support vector machine classifiers due to the size of my feature data.

Logistic regression is commonly used to predict binary results, whether some event does or does not occur.[3] The features are assigned weights, and the linear combination of the features and weights are inputted into a logistic sigmoid function.[3] The model ultimately outputs the probability of a given outcome. This classifier could work well, given that the targets in this project are binary.

The random forest classifier is an ensemble model of decision tree classifiers. Decision trees classify by subdividing objects at each level, from one initial root node to multiple, multi-level end leaf nodes. Random forests are used to help ameliorate the overfitting issue that decision tree classifiers often have.[5] They are created by training decision trees on overlapping subsets of data, and the final results come from picking the majority from the votes contributed by each decision tree. Random forest classifiers are nonlinear.

The naive Bayes classifier finds the probability that some event occurs given the occurrence of a set of features, under the assumption that all events are independent of each other.[6] This classifier could work well on the TF-IDF features, which removes the context and relationships between terms.

2.3. Model Testing and Selection

Initial tests on a single genre showed that for the TF-IDF features, logistic regression performed the best by far across accuracy, precision, recall, f1 score, and AUC values. However, for all three classifiers, recall was much lower among the minority class, indicating that they were not good at correctly predicting the minority outcome. For the word2vec features, the nonlinear random forest classifier had the best results, but the difference was smaller than for TF-IDF. The minority class recall scores were very low for all three classifiers trained on the word2vec features. Comparing TF-IDF + logistic regression and word2vec + random forest, TF-IDF + logistic regression had better scores across the board. Further testing on a genre target with greater imbalance showed increased disparities between performance for minority and majority classes.

Figure 1: Metrics for initial tests on TF-IDF features. “Romance” is the tested genre.

	test_accuracy		test_precision		test_recall		test_f1		test_roc_auc	
	mean	std	mean	std	mean	std	mean	std	mean	std
model										
gnb	0.653385	0.019428	0.727817	0.015276	0.785233	0.020271	0.755351	0.015366	0.637239	0.018981
logreg	0.686338	0.017608	0.687096	0.016556	0.991643	0.004206	0.811656	0.011957	0.652463	0.014830
rf	0.692629	0.023956	0.727245	0.019564	0.879358	0.028457	0.795836	0.017215	0.693891	0.030787

Figure 2: Metrics for initial tests on word2vec features. “Romance” is the tested genre.

	test_accuracy		test_precision		test_recall		test_f1		test_roc_auc	
	mean	std	mean	std	mean	std	mean	std	mean	std
model										
gnb	0.690231	0.017021	0.727299	0.014993	0.873319	0.009296	0.793582	0.010628	0.584693	0.031364
logreg	0.824744	0.008954	0.820774	0.010036	0.950285	0.007482	0.880775	0.008014	0.905740	0.006586
rf	0.747149	0.020366	0.735005	0.019695	0.984207	0.008986	0.841416	0.013559	0.861579	0.021222

In order to correct the imbalance issue, I used the imblearn module to randomly oversample and undersample the data. The oversampling was performed first, testing minority to majority ratios in intervals of 0.1. Then, I undersampled the data, to a minority to majority ratio of 0.8, which is roughly close to a 45-55 split between the two classes. Again, I ran tests using the three different classifiers, on both the TF-IDF and word2vec features. In order to better determine performance for the minority class, I also looked at the balanced accuracy metric, a combination of the recall, or true positive rate, and selectivity, or true negative rate.

Once more, the overall results were better for the classifiers trained on the TF-IDF features, and among those classifiers, logistic regression performed the best in terms of accuracy and AUC score. In order to determine the oversampling ratio, I looked at the balanced accuracy score, which seemed to peak at a ratio of around 0.5. Thus, for the final sampling strategy and classification, I decided to go with an oversampling ratio of 0.5 (minority sampled to 0.33 of the total dataset), an undersampling ratio of 0.8 (majority sampled to 0.56 of the total dataset), TF-IDF features, and the logistic regression model.

The goal for the final test was to determine the parameters for the logistic regression classifier. Using sklearn’s GridSearchCV, I tried different values for the solver, penalty, and inverse of the regularization strength. The function determined the best respective values to be SAGA, l2, and 10.

3. Results and Conclusion

3.1. Results

Figure 3: Metrics for each genre.

	accuracy	precision	recall	f1	auc
historical	0.946191	0.718593	0.871951	0.787879	0.913876
war_historical	0.944095	0.731343	0.849711	0.786096	0.903393
action	0.899371	0.406452	0.547826	0.466667	0.738959
thriller	0.897275	0.512987	0.523179	0.518033	0.732293
mystery	0.895178	0.619048	0.697561	0.655963	0.812891
fantasy_supernatural_horror	0.891684	0.569378	0.646739	0.605598	0.787283
youth_school	0.889588	0.562212	0.659459	0.606965	0.791608
family	0.879106	0.473430	0.604938	0.531165	0.759522
life	0.876310	0.356250	0.435115	0.391753	0.677942
friendship	0.865828	0.321212	0.398496	0.355705	0.656105
melodrama_romance	0.841370	0.854000	0.913369	0.882687	0.809507
romance	0.835080	0.847011	0.907709	0.876310	0.805815
comedy	0.766597	0.651769	0.704225	0.676983	0.752006
sitcom_comedy	0.759609	0.649351	0.693069	0.670498	0.744483
political_drama	0.603075	0.642857	0.619898	0.631169	0.601294
drama	0.588400	0.598923	0.604620	0.601758	0.587921

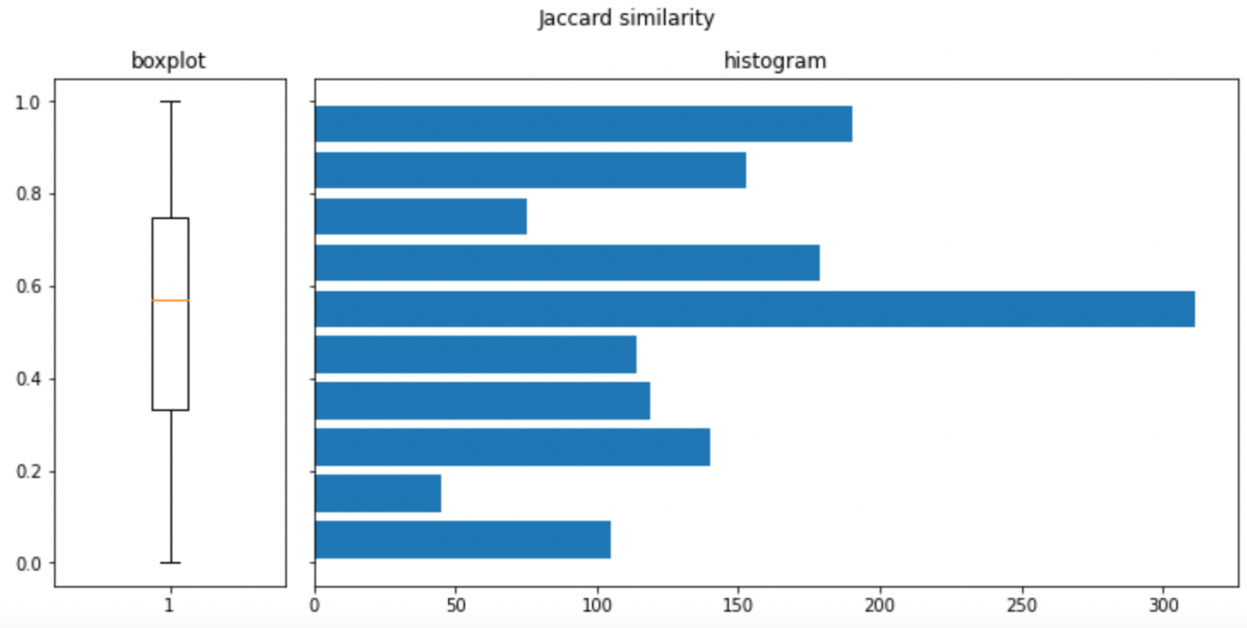
The metrics for the predicted results are shown in Figure 3. The average accuracy score is decent, and higher than expected, at around 83.6%. However, for the genre “drama,” as well as its combination “political_drama,” accuracy is little better than a random guess, something that is also reflected in the respective AUC scores.

For many of the genres, even though oversampling was applied to increase the number of objects that are labeled as the genre, recall remains low compared to accuracy. This is true of genres such as “action,” which had an accuracy near 90% and an AUC near 75%, but a recall of only 55%. However, this was not true of the “historical” and “war_historical” genres, which were oversampled and had recall scores of 87.2% and 85.0%.

For the most part, this model does a decent job at predicting genre based on user reviews. The average AUC score is about 75.5%, higher than random. However, for the genres “drama” and “political_drama” the score falls to 60.1% and 58.8%. The model also performs around one standard deviation below average for genres “life” and “friendship.”

I also looked at the Jaccard similarity scores, the number of items in the intersection of two sets divided by the number of items in the union of the two sets. The results were similar to expected, with a mean Jaccard index of 0.548 across the test data.

Figure 4: Boxplot and histogram of the Jaccard similarity scores between the sets of actual genres and predicted genres.



3.2. Conclusion

It is possible to predict Asian drama genres from user reviews, at least for certain genres. Characteristics of the dataset did impact performance, such as imbalanced representation between is genre and is not genre labels. However, overall, accuracy and AUC values were decent, so the assumption that user review text, drama genre, and drama content are related in some way holds true for the most part.

That assumption does not necessarily apply to all genres, as “drama” and “political_drama” have considerably low AUC scores, indicating that the model cannot predict the genre much better than random. Since the accuracy scores were low as well, perhaps the genre labels do not clearly and distinctly indicate a type of content. As such, the user reviews, which are assumed to reflect the type of content, will not contain many clear and distinct indicators of that genre. On the other hand, “friendship” and “life” both have decent accuracy scores but poor precision and recall scores.

Given the user reviews and genre labels on the MyDramaList website, the model using over- and undersampling, TF-IDF text processing, and the logistic regression classifier does for the most part do a decent job of labeling Asian drama genres when compared to other simple machine learning models that try to predict genre from text features. Performance does, however, suffer for some genres with imbalanced classes, and there may be genres that do not clearly indicate Asian drama content at all.

A. Code

All notebooks containing code used in the project, including for scraping, are in the same submitted folder. A text document describing each file is also in the folder. Data files are available for download from my Google Drive at this link.

B. References

1. Kumar, A., Rajpal, A., and Rathore, D. (2018). "Genre Classification using Word Embeddings and Deep Learning," *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2018, pp. 2142-2146. https://doi.org/10.1109/ICACCI.2018.8554816
2. Hoang, Q. (2018). *Predicting Movie Genres Based on Plot Summaries*. https://arxiv.org/pdf/1801.04813.pdf
3. Raschka, S., & Mirjalili, V. (2017). *Python Machine Learning* (2nd ed.). Packt Publishing.
4. Word embedding. *Wikipedia*. https://en.wikipedia.org/wiki/Word_embedding
5. VanderPlas, J. (2016). *Python Data Science Handbook*. O'Reilly Media. https://jakevdp.github.io/PythonDataScienceHandbook/
6. Navlani, A. (2018, April 12). *Sklearn Naive Bayes Classifier Python. DataCamp*. https://www.datacamp.com/community/tutorials/naive-bayes-scikit-learn