Predicting Sentiment of Steam Game Reviews Using Machine Learning

Winifred Nada wnauda@umass.edu

Miles Swank mswank@umass.edu

Abstract

Our project started with the question: Can machine learning models predict characteristics of video game reviews left by users? Game reviews have the ability to lend valuable insight to game developers about how users experienced their product, which makes exploring this topic equally as valuable. To accomplish the task, we trained different flavors of machine learning models on a large dataset of game reviews from the Steam gaming platform and evaluated the efficacy of each. In summary, machine learning models that we employed were surprisingly effective at classifying reviews.

Introduction

It is the third decade of the twenty-first century and video games are a dominant choice of consumers in the market for leisure activities. As is the case with other forms of entertainment media, user reviews of video games provide game distributors with insight towards what the market's interests are. As such, large scale analysis of user reviews is a helpful tool for those looking to compete in this market.

We formulated our initial research question as such: Using words as features, can a machine learning model predict the sentiment of a steam user's game review and with what accuracy? Following from this, we also wondered if using additional metrics as features would help the model's performance. Some examples of additional features could include user playtime or a "helpfulness rating" of the review.

The goal of asking these questions is to learn how accurately machine learning can be used to analyze game reviews. Testing the ability of various machine learning algorithms to classify the sentiment of reviews is a rudimentary first step in exploring this area. If the results proved sufficient, more in-depth questions could be answered using machine learning. For example, the question "What portion of reviews that mention a specific game aspect are positive and what portion are negative?" This type of analysis would help game developers learn what users liked and disliked about their product on a large scale.

Related Work

The study What Causes Wrong Sentiment Classifications of Game Reviews? by M. Viggiato et al. conducted an analysis on sentiment classification of video game reviews. They found that existing sentiment classification models underperform on video game reviews, with accuracy ranging from 0.53 to 0.70 among the models tested in the study. This approach used pre-existing models trained for more broad sentiment classification tasks, and observed how they would extrapolate to video game reviews.

During this study, the authors were able to identify examples of game review archetypes that were problematic for the classifiers. One specific example is reviews that mention both positive and negative things about a game, or rather reviews that aren't entirely positive or negative throughout.

Another study, Length and Sentiment Analysis of Reviews, focused their efforts on observing relationships between features such as word count and the user's emotional experience with the game. It was found that generally, word counts were a good indicator of whether a review was positive or negative.

These studies show that the sentiment classification of game reviews has been studied before in moderation. Our project aims to conduct observations with our own custom hyperparameters and model settings which we can compare with the findings of other studies in a similar area. Using this as a baseline for comparison, we then aim to extend the training features to include things such as playtime that were not discussed in these previous works.

Data

Our dataset comes from an open source Kaggle dataset of over 100 million steam reviews. The set we used for the majority of our testing was a subset containing only those reviews with a "helpfulness" rating above 0.8.

Dataset: Kaggle - 100 Mil+ Steam Reviews
License: MIT

This is a large collection of video game reviews submitted by users of the Steam gaming platform. They were collected by the author using Steam's User Reviews - Get List API. The set contains two files. One is a set that contains all reviews in the dataset, numbering more than 100,000,000 samples. The other is a subset of this set, containing only reviews marked with a "helpfulness rating" above 0.8.

The “helpfulness rating” is a metric specific to the Steam platform. It is generated by users who read reviews. When browsing reviews, users can mark other people’s submissions as “helpful” or “not helpful” which contributes to the calculation of the helpfulness rating for that review. The rating is expressed as a percentage, or in our case a float value from 0 - 1.0.

Both files contain the same column headers. These column headers include “recommendationid”, “appid”, “game”, “author_steamid”, “author_num_games_owned”, “author_num_reviews”, “author_playtime_forever”, “author_playtime_last_two_weeks”, “author_playtime_at_review”, “author_last_played”, “language”, “review”, “timestamp_created”, “timestamp_updated”, “voted_up”, “votes_up”, “votes_funny”, “weighted_vote_score”, “comment_count”, “steam_purchase”, “received_for_free”, “written_during_early_access”, “hidden_in_steam_china”, and “steam_china_location”. The following headers are the ones we deemed relevant for our project.

“author_playtime_forever”: How many hours of playtime the author of the review has at the time of formulation of the dataset. This could be used as an additional feature.

“language”: What language the review is written in. This is Important because our project only deals with English reviews.

“review”: The textual content of the review. Used as the feature for our main research question experiments.

“voted_up”: Whether a review is marked as positive or negative, AKA the sentiment. The value of this column is what our classifiers aim to predict.

“votes_up”: A count of the number of upvotes, AKA likes, the review has. Could be used as an additional feature.

“votes_funny”: A count of the number of users that marked the review as funny. Could be used as an additional feature.

“weighted_vote_score”: The helpfulness rating of the review. Could be used as an additional feature. All reviews in the second file of the dataset have this value greater than 0.8.

“comment_count”: A count of the number of users who left comments on the review. Could be used as an additional feature.

Method

Our preliminary testing stage involved training a Logistic Regression model on the dataset containing only reviews with helpfulness ratings above 0.8. For this, Scikit-learn's LogisticRegression model for Python was used. The solver in this instance was Scikit-learn's 'lbfgs' solver. The textual content of each review was converted into a bag-of-words representation using Scikit-learn's CountVectorizer class. These were the only features used.

The model was only trained on English reviews, meaning reviews whose value under the 'language' column was not set as 'english' were disregarded. Stopwords were also removed from the reviews. The list of words stopped contained all common English stopwords. Specifically, the list used was the one defined in the CountVectorizer class. The model was trained on 80% of the dataset and asked to predict the value of 'voted_up' on the remaining 20%. Possible values were 0 to indicate a negative review and 1 to indicate a positive review. We evaluated the model's results in terms of accuracy, precision, recall,and the f1-score.

For further experiments, we tried the 'liblinear' and 'sag' solvers, as well as experimenting with different regularization algorithms. This was done by modifying the parameters of Scikit-learn's logistic regression model to alter the solver and regularization used.

When introducing playtime into the model, an extra token was inserted into the bag of words model to represent the playtime. Two methods of modeling playtime were used. The first model had two playtime categories, one for less than 1000 minutes and one for 1000 minutes or more. The second model had 5 categories, corresponding to 0-99 minutes, 100-999 minutes, 1,000-9,999 minutes, 10,000 - 99,999 minutes, and 100,000+ minutes.

Other methodologies in the later testing stage involved experimentation with various different hyperparameters. For example, different Normalization settings were tried. Additionally, different Logistic Regression solving algorithms provided by Scikit-learn were tested.

Results

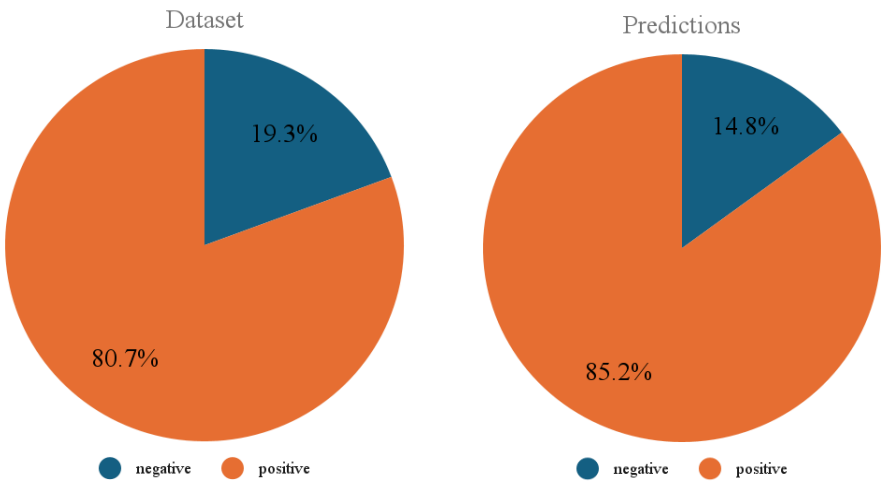
Our preliminary logistic regression model predicting only based on the text of the reviews resulted in an accuracy of 89.6%. For the positive class, it had a precision of 91% and a recall of 96%, while for the negative class it had a precision of 80% and a recall of 62%. Other metrics are shown, along with the full confusion matrix, in the following figure:

```
Accuracy:  0.8964410508606186
Confusion Matrix:
 [[ 4601  2871]
 [ 1130 30033]]
Classification Report:
              precision    recall  f1-score   support

           0       0.80      0.62      0.70      7472
           1       0.91      0.96      0.94     31163

    accuracy                           0.90     38635
   macro avg       0.86      0.79      0.82     38635
weighted avg       0.89      0.90      0.89     38635
```

Analyzing the results more deeply, our test set contained a total of 38,635 reviews. Of those reviews, 80.7% were positive and 19.3% were negative. Our model predicted 85.2% positive and 14.8% negative, slightly over predicting the positive class. These results are visualized in the following figures.



If we look closer at the classes, 96% of ground truth positive reviews were predicted correctly, compared to only 62% of negative reviews. The following figure demonstrates this discrepancy.

Adjusting the logistic regression algorithm and the normalization technique did little to change the results. Removing regularization from the lbfgs algorithm decreased accuracy to 86.7%, and precision for the negative class to 66%.

The liblinear algorithm performed similarly to lbfgs, with an accuracy of 89.8% using L2 normalization and 89.4% using L1 normalization. However, liblinear failed to converge, indicating it may not be a good choice for such a large dataset.

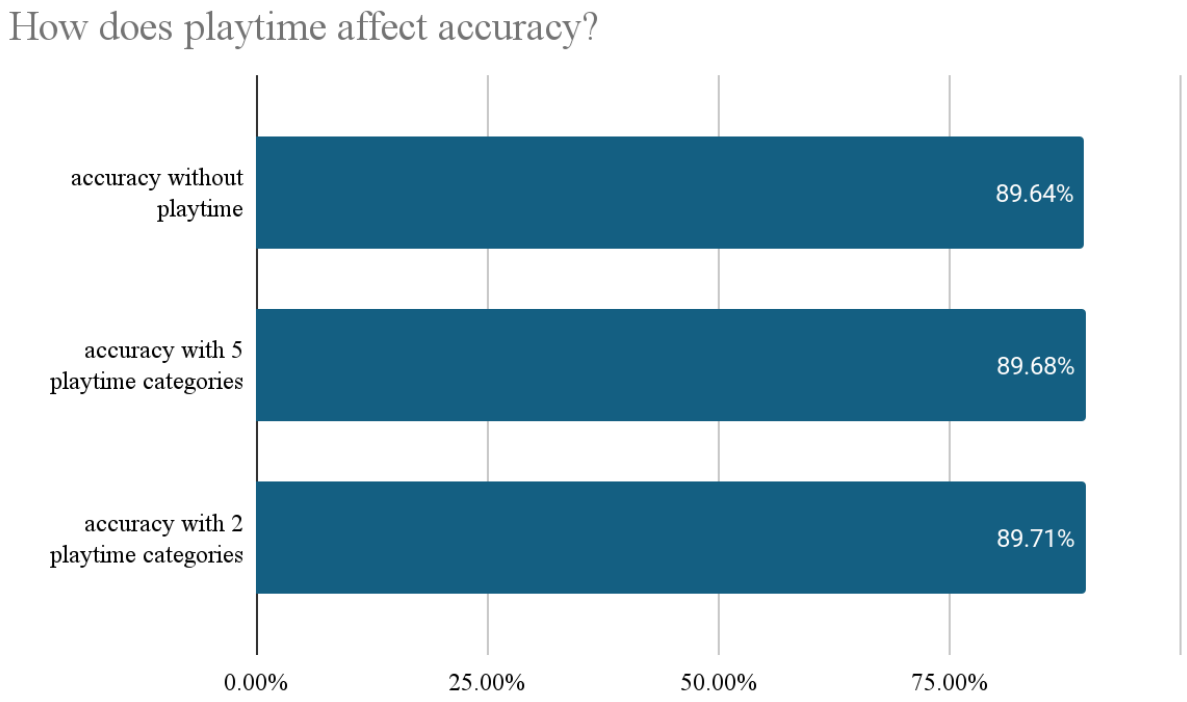
L2 showed to be the best penalty overall, with better performance no matter which model was used. Despite the intuition that a higher level of regularization would hurt outcomes for the negative class, it actually improved them, with precision and recall both being higher with L2 than with L1 or no regularization.

The sag algorithm reached the maximum number of iterations before converging, showing it may also be unsuited to this size of dataset. When the maximum number of iterations was increased to 5,000, it performed slightly worse than lbfgs, reaching an accuracy of 89.2%.

The following chart shows the performance of the different solvers and penalties:

Solver	Accuracy	Class 1 precision	Class 1 recall	Class 1 F1-score	Class 0 precision	Class 0 recall	Class 0 F1-score
LBFGS L2	89.6%	91%	96%	94%	80%	62%	70%
LBFGS None	86.7%	92%	92%	92%	66%	65%	65%
Liblinear L2	89.8%	91%	97%	94%	81%	61%	70%
Liblinear L1	89.4%	91%	97%	94%	81%	59%	69%
Sag L2 max_iter =1000	87.4%	90%	95%	92%	72%	57%	64%
Sag L2 max_iter = 5000	89.2%	90%	97%	94%	82%	57%	67%

With playtime introduced into the model, the hypothesis was that it would improve the accuracy of predictions. However, accuracy was not largely affected by including playtime, either with 5 categories or with 2 categories. The model with 2 categories was slightly better, but the difference was less than a tenth of a percent.



Discussion and Future Work

This work shows that even very simple sentiment classification models like bag of words logistic regression can be effective for video game reviews when trained on other reviews. The accuracy metrics achieved during our study exceeded what was expected going into the project. The English language is large and full of ambiguity which makes training machines on it a complex task. This is especially true in internet forums where things such as slang, improper grammatical structures, and sarcasm are present.

The results also show that playtime may not be as good of an indicator of a review's sentiment as it seems. One reason for this could be the extremely wide ranges of playtimes, from single digits numbers of minutes to hundreds of thousands. More carefully engineered playtime ranges could have the potential to improve predictions. In future work, these reasons could be more closely studied, with an investigation into in what cases playtime was a better or worse indication of sentiment.

Future extensions of this study could entail looking into game reviews sourced from other platforms or forums. Our project only looks at reviews from Steam. It is not apparent whether the results we found would stay consistent across different platforms. It is possible that small differences in things like the culture of users on the platform would affect the accuracy of a classifier under these conditions.

Further investigation into the reviews for which an incorrect result was predicted could also be a topic for future research. A very high portion of negative reviews were miscategorized as positive, showing a positive bias in the model. Was the reason for this positive bias the overrepresentation of positive reviews in the dataset, or is there a cause specific to this analysis of video game reviews that causes so many such misclassifications? The previous study by M. Viggiato et al. found that video game reviews were more difficult to categorize due to the popular format of including both positive and negative sentiment in the same review. A future experiment could look into whether accounting for that tendency could improve the effectiveness of this model for correctly categorizing negative reviews.

Another further extension could look at how the logistic regression model used in our experiments compares to other NLP models at this task. We only focused on this particular model, but results could differ when taking into account the performance of models such as Naive Bayes or BERT-based models, or a more complex feature-based logistic regression model than bag of words.