Predictive analysis based on Amazon Game Reviews

1. Introduction

The goal of this project was to use review text from the reviews provided by the Amazon video games dataset to predict the ratings of Amazon game reviews. We also want to find and categorize the ratings into sub-categories such as density of ratings (5 star, 4 star , 3 star ETC.), number of reviews per reviewer, verified reviewers, number of words per review.

The dataset used for this task is the Amazon Review Data. This dataset includes the rating of the product, whether the user who made the review purchased the product, the text of the review, the summary of the review provided by the reviewer, and the time of the review provided in a *%m %d, %Y* format and as a Unix timestamp, and columns regarding the product and reviewer details. We specifically explored data pertaining to video games from the Amazon dataset. For our predictive task, we are using a k-core subset of the data (more specifically 5-core), where all the remaining users and items have k-reviews each. Our predictive task aims to predict the rating of a particular game based on the review that a user has given. We believe that the review text is key to estimating the rating of a product. By analyzing the sentiment of the given review, we are able to predict what the user feels about the product. This has an important function in creating recommender systems, where the scope of this project can be expanded to predict what rating a user would give a new product.

2. Exploratory Data Analysis

Relevant Features of the Dataset:

Feature	Type	Description
overall	Integer	User Rating, ranging between values of 1 and 5
verified	Boolean	True or False, indicating if user purchased the game
reviewText	String	User review of the product
summary	String	Summary of the review

Basic Statistics:

- **Dataset Length Following Preprocessing**: 497240
- **Number of Columns Following Preprocessing**: 9
- **Number of Relevant Columns**: 4
- **Mean Word Count per Review**: 122.71413200868795
- **Median Word Count per Review**: 40
- **Standard Deviation of Word Count**: 227.01922456991764

Our analysis of the entire dataset revealed a number of interesting correlations. In our EDA, we modeled the distribution of ratings, the number of reviews given per reviewer, the number of verified vs. unverified reviews.

Our findings in Figure 1 showed that ratings 4 and 5 are the most common. Getting a low rating from a user is fairly unlikely, with 2 being the least probable rating to be given by a user. From this, we can assume that this dataset is pretty heavily skewed as the user is far more likely to give a high rating than to purposely give a game a new rating. One reason for that could be that a user is far more likely to leave a rating if he thinks favorably of the product.

Predictive analysis based on Amazon Game Reviews

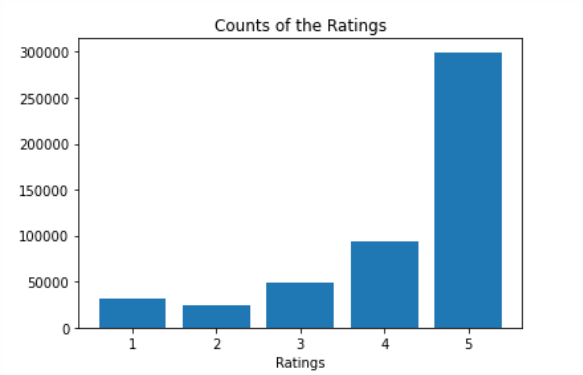


Figure 1: Counts of different Ratings given in Dataset

In Figure 2, we plotted the number of reviews that each reviewer gave. From this plot, we can infer that the number of reviews that a user is most likely to give are 5s are primarily located within bins 0 to 6, with the probability decreasing as the number of reviews going down. However, there are still people who give more than 16 - 20 reviews on a number of different games - these are most likely customers that are more passionate about the video who will likely give an in-depth analysis into the game.

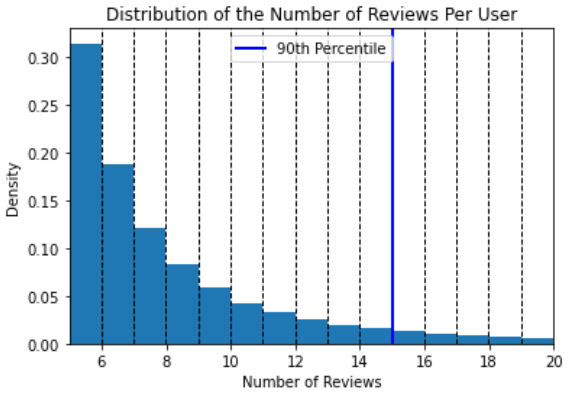


Figure 2: Density Histogram of the Distribution of Reviews per User

In Figure 3, we created a bar plot to compare the number of verified reviews to reviews which are unverified. A 'verified' user is one who is confirmed to have purchased the game. The point of verified users and reviews is to validate the actual claims made in the review to protect people from assuming things due to trolls, frauds and other claims from people who have not purchased the game. In this dataset, a small number of unverified users would indicate that the reviews are indicative of how liked the game is among users. Conversely, a large number of unverified users makes it hard to infer whether the game is generally liked or disliked. In Figure 3.1, we can see that the ratio of verified users to unverified is ~ 2:1. From 3.2, we can see that the percentage of verified users is roughly the same for ratings 1-4. However, the odds of a 5 star rating being written by a verified user is higher than other ratings. This indicates that the verification status is a useful feature to include in our model.

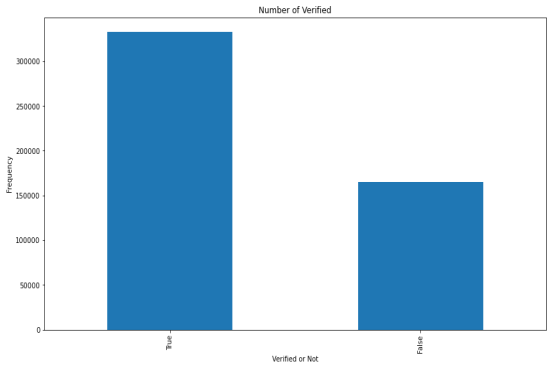


Figure 3.1: Bar Plot of number of Verified vs Unverified users

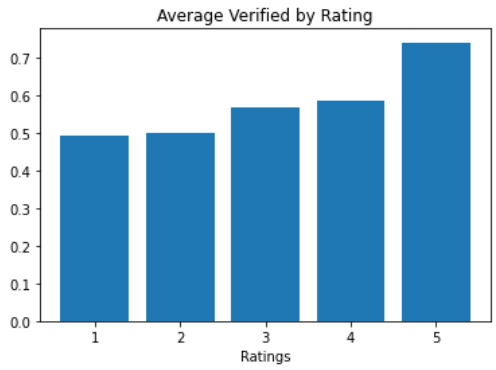


Figure 3.2: Percentage of verified users per rating

In Figure 4, we compared the length of the review vs. the average rating given. The size of the marker corresponded to the number of reviews that. We found that sers who gave reviews which were shorter in length were far more likely to give higher ratings, as a majority of the high ratings are concentrated in that area.

Predictive analysis based on Amazon Game Reviews

The longer the review, the more likely it is negative and correlates with a lower rating.

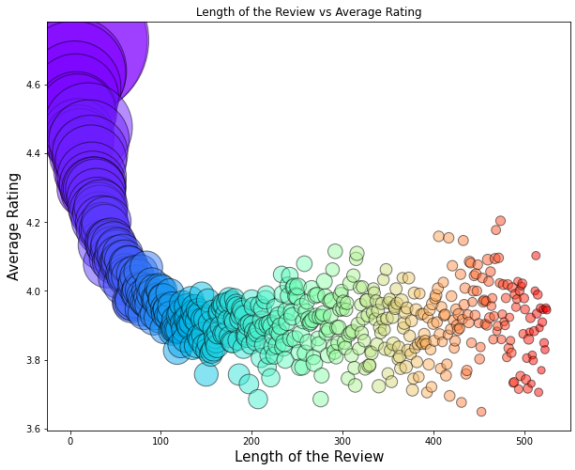


Figure 4: Scatter Plot of Length of Review vs. Average Rating

Our final plot finds an important trend in the average length of reviews over time. In Figure 5, we can see that from roughly twenty years ago to now, the length of reviews that people post online has drastically decreased from 500-600 words to around 300 words. This is important as we can infer that people are putting less effort into reviews in general, making it harder to predict ratings from them.

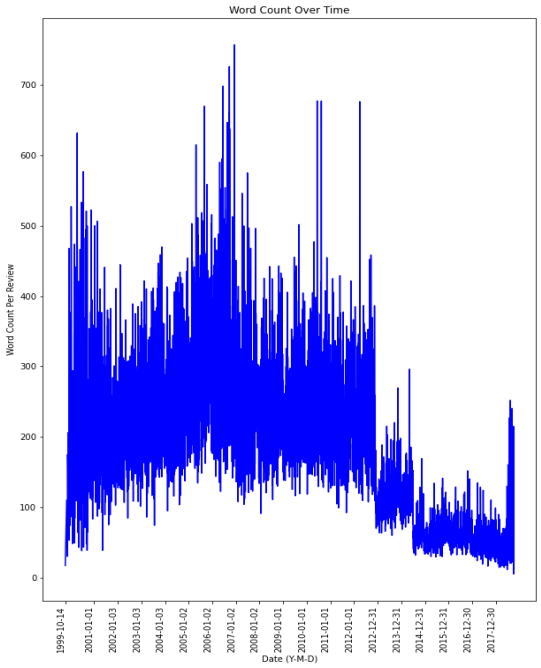


Figure 5: Time Series plot of Word Count Per Review

To visualize what words may be most prominent in each rating class, we created a couple of word clouds. For example, figure 6 displays what the word clouds look like for reviews given a 5 star rating. From these word clouds, we noticed that game was often the most prominent word in each class. This indicates that the word game will not provide any extra information in a review, motivating us to add it as a stopword for text processing.

Figure 6: Word cloud of 5 star ratings

2. Predictive Task and Models

As discussed previously, the predictive task in this project is to predict the user rating of a video game through the reviews given. After dealing with the null values, we are left with 497K user interactions through which we can make our predictions. However, due to the dataset having a large class imbalance, any model we train on this full dataset will be far more biased to predict 5 star ratings. In order to resolve this problem, we took a smaller sample of 100k interactions, with 20k interactions for each class. We split these interactions into training and testing data with a 80-20 split. As a baseline from randomly choosing a rating, we would expect a 20% accuracy. While we initially considered incorporating text length and verification status due to the results of our EDA, we found that including these features reduced the performance of the model significantly. We chose only to consider review text, due to summary text often including the rating given, defeating the purpose of the model.

There are a few different ways to approach this problem. In our project, we

Predictive analysis based on Amazon Game Reviews

implemented several embedding and classification strategies. Our first baseline embedding is the Bag of Words approach. This involves converting reviews into numerical vectors based on word presence. A machine learning model can then be used to fit on these vectors to predict ratings from text. This approach is a relatively simple one as it focuses solely on presence rather than the context and order. Therefore, while this method is a good baseline, it fails to capture linguistic nuances.

The second baseline embedding approach is to use TF-IDF text analysis, which evaluates word importance by considering frequency and rarity of words - it gives the highest weight to impactful words. Like Bag of Words, it generates a vector from the review that can be used to make predictions. Once again however, while it performs better than Bag of Words, it lacks contextual understanding.

Our final embedding method is by using a language representation model called BERT, which stands for Bidirectional Encoder Representations from Transformers. This embedding method takes into account both position and uses methods to create a bidirectional representation, making it far stronger than the bag of words and TF-IDF embeddings. To generate embeddings, we imported a pre-trained BERT model and tokenized text using it.

To preprocess the text for the Bag of Words and TF-IDF representations we removed traditional stop words and made each word lowercase. In addition, we removed extra stop words that appeared commonly in each rating, such as “game”. For the BERT embedding, we did not remove stop words, as they can provide important positional context that the other representations do not consider.

For the Bag of Words and TF-IDF representations, we compared several different classification models. We tested a Naive Bayes classifier, a logistic regression classifier and a linear support vector classifier.

While the BERT model can be used to generate text representations, it can also be trained to use these representations for various predictive tasks. We fine-tuned the pre-trained BERT model to use its embeddings to classify text into ratings.

To evaluate our final model, we chose to take into account both Mean Square Error (MSE) and accuracy.

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

MSE is crucial for assessing the model's prediction accuracy. Though our task involves predicting ratings—a classification problem—simply taking the accuracy of predicting the exact rating penalizes predicting a 5 star rating as a 1 star rating as much as predicting it as a 4 star rating. In this context, MSE is more apt than accuracy as it gauges the magnitude of prediction error, pivotal for assessing the model's performance in a continuous rating scale. Despite this, our overall interest is in predicting specific classes, so we still wanted a final model that performed well in this regard as well.

3. Final Model

After testing our methods, we found that the BERT method had by far the best performance, across both accuracy and MSE. While the other models all performed far better then a random baseline of 20% accuracy, BERT was the clear stand out. The BERT method had a MSE of .7014, which corresponds to a .84 average difference between the prediction and the true label.

Embedding Method and Classification Model	Test Accuracy	Test MSE
BoW, Logistic Regression	**48.72%**	**1.3969**
TF-IDF, Logistic Regression	**50.41%**	**1.2451**
TF-IDF, LinearSVC	**47.83%**	**1.4203**
TF-IDF, MultinomialNB	**45.63%**	**1.3664**
BERT, BERT	**60.11%**	**0.7014**

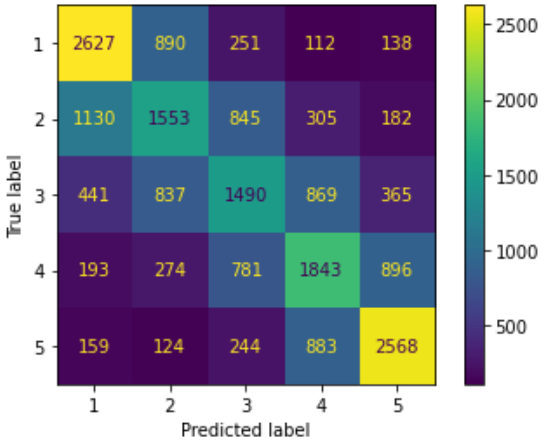
Predictive analysis based on Amazon Game Reviews

We ended up training the model for 2 epochs, as the model converged extremely fast and overfit with more epochs. This was because we were using a pre-trained model, so only a small amount of fine-tuning was needed to adapt it to our use. We used the AdamW optimizer with a learning rate of .00002.

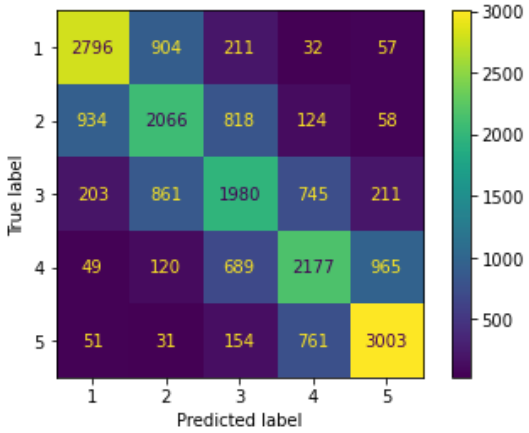
While it was fairly easy to do basic hyperparameter optimization for the TF-IDF and BoW classification models, we ran into issues when attempting to optimize the BERT model. Primarily, training the model took an extremely long time, due to its size (45 min+). However, even without intensive optimization, it was still able to outperform the other models by a wide margin.

We also ran into scalability issues with the BERT model. Due to its size, training it on all the text quickly led to out of memory errors. To resolve this, we used a data loader to batch our data and keep it under our maximum memory.

Comparing the different models using confusion matrices, we found that all the models generally had problems with differentiating ratings close to each other, such as predicting whether a 4 star review should be 3,4, or 5 stars.



This confusion matrix compares the true ratings to predicted ratings for the TF-IDF, logistic regression method, the best performing baseline approach. The similar rating ambiguity issue is illustrated by the counts for the incorrect labels around the diagonal.



For comparison, this is the confusion matrix for the BERT predictions, which also suffers from the same issue. This model, however, predicts very wrong ratings far less commonly.

This issue makes sense, as the exact sentiment corresponding to what someone might write varies from person to person, causing some ambiguity in similar ratings.

While the BERT model clearly had the strongest performance, it came with numerous downsides relating to its size. In comparison to other models, which all took under a minute to train, the BERT model took an enormous amount of time. It also took up significantly more memory.

4. Literature

The Amazon dataset used for this task is available on Professor McAuley's website of available datasets. The data was extracted using a process to scrape only reviews of items that have "good" justifications [1]. As a result, the dataset may not have been available to the larger public due to their lack of awareness of the dataset repository. However, there are many research papers that utilize similar datasets with vastly different implementations.

In the article, "Sentiment Analysis for Amazon Reviews", the authors compared the performance of multiple algorithms for product rating classification. Multinomial Naive Bayes, a probabilistic machine learning algorithm primarily used for text classification, had a higher accuracy on the test data than the deep learning methods per the author's results. Naive

Predictive analysis based on Amazon Game Reviews

Bayes can be mathematically described as seen below.

$$p(x_1, ..., x_k|y) = \prod_{i=1}^{k} p(x_i|y)$$

Apart from LSTM, Multinomial Naive Bayes received the second highest test accuracy out of all the models. Similarly to our predictive task, the authors of the article used text-based features and encoding in their training of the various models. Because of these similarities, we decided to implement Multinomial Naive Bayes as one of our baseline models, which we would compare against a deep learning approach, such as BERT.

In the paper "Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects" [1], the authors used the BERT model to fine tune the classification task. Later they split the annotated dataset into Train, Dev, and Test sets with a 0.8/0.1/0.1 ratio and with fine tuning, BERT achieved an F1 score of 0.8 on the test set. By doing this they can compare the BERT model with other baseline models, which showed that the BERT model had a higher F1 score and precision when compared to other classifiers. Our findings support this conclusion, with the BERT model massively outperforming the baseline other models we trained set..

In the article "Predicting Amazon customer reviews with deep confidence using deep learning and conformal prediction"[2] A through analysis of deep learning was conducted and the results showed that the combination of deep learning, including natural language processing, and conformal prediction results in highly predictive and efficient temporal test set sentiment estimates the research also showed that the combination of deep learning and conformal prediction successfully handles class imbalances without class balancing measures. Despite this, we still found improved performance on the test set with BERT by balancing the dataset.

5. Results and Conclusion

Overall the BERT language representation model returned the best results on the test dataset relative to the other models. The BERT model outperformed the other models with a test accuracy of 60.11% and test MSE of 0.7014 with very little optimization. This was an increase of just under 10% in accuracy over the next best model, a massive boost. The BERT embedding was the only embedding method that took into account position and context of words. The increased performance corresponding to its use indicates that there is a benefit to considering the position of words in reviews in regards to sentiment analysis. It also shows how improvement on text data can be gained by using larger models. This is in contrast to other forms of data, where complex models can overfit easily.

The small number of epochs and low learning rate we used to train our BERT model were crucial in ensuring that it did not overfit. Since the model was pre-trained, only a small amount of iterations were needed to prepare it for a classification task. High learning rates caused our model's performance on the test data to be quickly thrown off, since the newer data alters the pre-trained knowledge too fast.

Overall, we believe the BERT model succeeded while others failed due to the importance of the position of a word in a review. While the ambiguity problem still could be improved upon, taking into account more than just the frequency of a word in a review was shown to have measurable effects on the performance of a predictor.

6. References

[1] "Justifying recommendations using distantly-labeled reviews and fined-grained aspects." Jianmo Ni, Jiacheng Li, Julian McAuley. *Empirical Methods in Natural Language Processing (EMNLP)*, 2019

[2] Ulf Norinder & Petra Norinder (2022) "Predicting Amazon customer reviews with deep confidence using deep learning and conformal prediction.", *Journal of Management Analytics*, 9:1, 1-16

[3] Tan, Wanliang, et al. "Sentiment Analysis for Amazon Reviews.." *CS229.stanford.edu*,

https://cs229.stanford.edu/proj2018/report/122.pdf.
[4] Balakrishnan, V., Shi, Z., Law, C.L. *et al.* "A deep learning approach in predicting products' sentiment ratings: a comparative analysis." *J Supercomput* 78, 7206–7226 (2022).
[5] Gomez, Aidan N., et al. "Attention Is All You Need." *arXiv*, 12 June 2017, https://doi.org/10.48550/arXiv.1706.03762.