CSE 158 - Assignment 2 Report

Danh Nguyen Eric Salguero Richard Tran

1. Dataset

1.1. Description

The dataset that we will be using for this assignment was taken from https://cseweb.ucsd.edu/~jmcauley/datasets.html. More specifically, we used *"Steam Video Game and Bundle Data" Version 1: Review Data*. The data was scrapped from the site https://store.steampowered.com/ and consists of game reviews.

Figure 1. Example of steam review

Figure 1 is an example of a game review that, for example, consists of the following metadata:

- Recommended / Not Recommended
- The date on which the review was posted
- The review text
- How many people found the review helpful
- How many people found the review funny

1.2. Processing

Upon processing the dataset, we found that it contained 59305 reviews. Each entry of the data set is of the following form: [userId, userUrl, reviews].

The items are self-explanatory. The userId denotes a unique integer specified to each user, the userUrl denotes the link to the user profile, and reviews is the set of game reviews the user posted. For example, the data consists of the (userId, userUrl) pair (76561197970982479, https://steamcommunity.com/profiles/76561197970982479).

1.3. Analysis

The first thing we did was look at how many positive (recommended) or negative (not recommended) reviews there were. It came to be

- Positive: 52473
- Negative: 6832

meaning that about 88% of the reviews were positive. This makes sense since people are more prone to avoid saying negative things about something, but this suggests that our data is highly imbalanced. Next, we also looked at the top 35 occurring uni/bi/tri-grams across the reviews, depending on how we parsed the text:

1. No Punctuation, Keep Stopwords, No Stemming
2. No Punctuation, No Stopwords, No Stemming
3. No Punctuation, No Stopwords, Stemming

So, in total, we generated 9 different bar charts. An example of such a chart is given by Figure 2, which depicts the top 35 unigrams appearing using text parsing process (1).

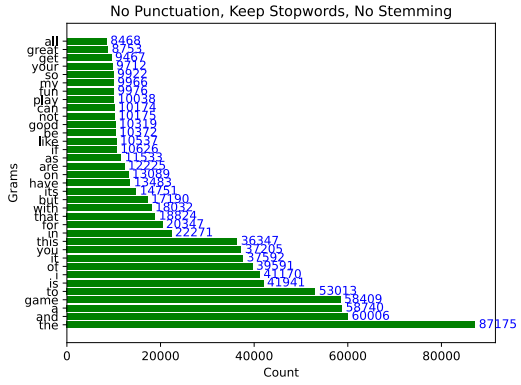

Figure 2. 1-grams (1)

As can be seen, the top 5 unigrams are "the", "and", "a", "game", "to"– all of which are mostly stopwords. Upon removal of stopwords, the top 5 unigrams become "game", "like", "good", "play" and "fun". This is depicted in Figure 3.

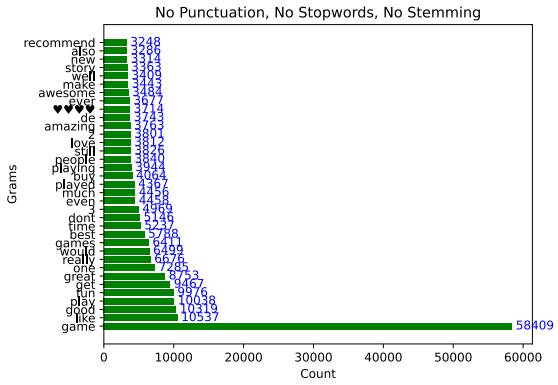

Figure 3. 1-grams (2)

2. Task

2.1. Background

When examining our data set, we determined that the review texts will be the data that is used for our prediction since it is the most diverse and complicated data compared to the others. After carefully examining other data, we set our goal to be predicting whether a user would recommend a game or not depending on the popular words among all the reviews using the review texts.

In a broader sense, this classification task can be categorized under sentiment analysis, which is the following problem: given a text, analyze it and determine how "positive" or "negative" it is. Then we can think of classifying as a review as a "recommendation"/"non-recommendation" is the same as classifying it as "positive"/"negative."

2.2. Metrics

When testing our models we created a function that tested the Accuracy (ACC), Balanced Error Rate (BER), True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) of the model.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$
$$BER = \frac{1}{2}(\frac{TP}{TP + FP} + \frac{TN}{TN + FN})$$

This allowed for us to see how specifically our model was preforming. We were able to examine whether our model was producing more False Positives or False Negatives and adjust our model accordingly. Since our data set was imbalanced and had more positive reviews than negative reviews the accuracy of our models would not be credible. The models would be more fit to reading positive reviews and not have been trained much on negative reviews. With this in mind, we instead focused on reducing the Balanced Error Rate of our models. This would allow for us to gauge the proportion of wrong classifications for each model.

2.3. Features

The features that we will consider for our models are entirely derived from the review texts:

- unigrams
- bigrams

These are natural features to consider because we are trying to measure sentiment. Moreover, besides of just looking at unigrams/bigrams directly, we can analyze the structure of such grams. Other features that we tested involved stemming our bag-of-words as well as only including particular part-of-speech words. For example, we can look at whether an n-gram contains an adjective or follows a particular structure.

2.4. Baseline model

For the baseline model, we use a simple Naive Bayes classifier based on a bag-of-words model, where the features were the top 1000 unigrams. This was modified so that if a test sample contained no features, we automatically guess "positive." This results in the following statistics:

- (ACC, BER) = (0.91274, 0.45637)
- (TP, TN, FP, FN) = (10775, 0, 1030, 0)

As can be seen, the Naive Bayes classifier simply guessed that every review was positive. This is not surprising since the dataset was imbalanced with positive reviews. The model thus over-fitted for the imbalance of positive reviews and thus had a low Balance Error Rate. Since the model over fitted we disregard the credibility of this model as test other models.

3. Models

3.1. Overview of Models

Unless mentioned otherwise, we processed as follows:

- Remove all punctuation
- Remove all stopwords

We did not stem because the top occurring grams were similar to when we did not stem. Moreover, stemming might remove subtle but important context such as "remove" vs "removed", which is important in sentiment analysis. We used a train/test data set split of 80/20. And we used Logistic Regression.

3.2. Model 1: Unigrams

This model did not remove stopwords. It was a bag-of-words model trained on the top 1000 unigrams occurring across the train set. We then record the number of occurrence of those unigrams in each review and store it in a list.

3.3. Model 2: Unigrams Filtered by POS

This is model 1 but we filter the unigrams according to whether it was an adjective, adverb or verb. Tagging each word with its part of speech was done using the NLTK package. Any unigram that did not match one of the above parts of speech was thrown out. From the remaining unigrams, we considered the top 1000 occurring.

3.4. Model 3: Bigrams

This model had the exact layout as model 1, but we used bigrams instead. After recording the new features, we have the following bag of words:

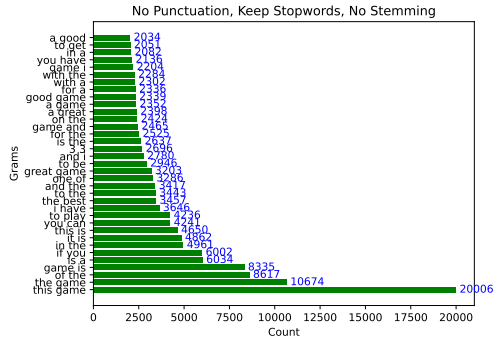

Figure 4. Top bigrams, with stopwords

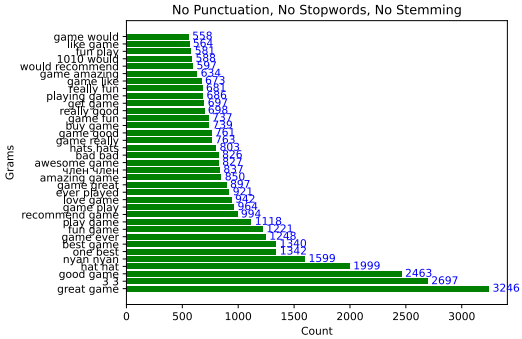

Figure 5. Top bigrams, no stopwords

When using bigrams, we were able to extract more information about each term comparing to unigrams. This provide us a more context and a better analysis of the dataset. Moreover, we decided to remove stopwords in order to further remove unrelated bigrams from our research analysis. When we did not filter by stopwords, the 5 top occurring bigrams were “this game”, “the game”, “of the”, “is a”, “if you”, which do not particularly provide any indication of sentiment. So, these bigrams are likely to be redundant features. When we removed the stopwords, our unique bigrams would contain more positive words; as a result, the top 5 bigrams become “great game”, “3 3”, “good game”, “hat hat”, “nyan nyan” which are more indicative of sentiment.

3.5. Model 4: Bigrams Filtered by POS

Similar to model 3, we filtered the bigrams by parts of speech. One different modifcation was we filtered by a list of rules taken from (Turney, 2002). First, we have the following classifications of POS tags:

1. JJ – adjectives
2. NN* – nouns
3. RB* – adverbs
4. VB* – verbs

Upon parsing the data, we adhere to the following pattern of tags for extracting bigrams:

First Word	Second Word	Third Word (Omit)
JJ	NN or NNS	anything
RB, RBR, or RBS	JJ	not NN nor NNS
JJ	JJ	not NN nor NNS
NN or NNS	JJ	not NN nor NNS
RB, RBR, or RBS	VB, VBD, VBN or VBG	anything

The idea behind constructing this model was to filter out more words that did not provide help to our analysis. As a result, our remaining words are more indicative of sentiment as well as the structure for the bigram was more complex and accurate. Following this extraction process, the top 35 bigrams are:

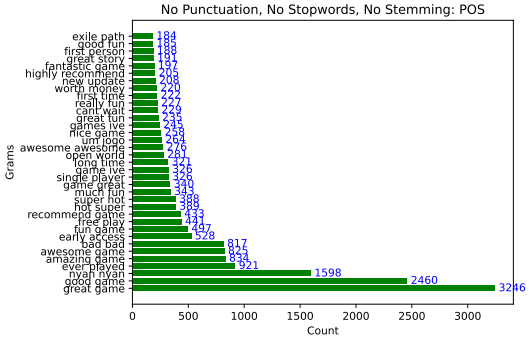

Figure 6. Top bigrams filtered POS

Then the top 5 bigrams are “great game”, “good game”, “nyan nyan”, “ever played”, “amazing game”. This is more

indicative of positive sentiment than the top 5 grams obtained when we only removed punctuation and stopwords.

3.6. Results

Because we had an imbalanced data set, we ran each model above twice: one where the class weight parameter was set to the default – each class was given equal weight – and one with the class weight parameter set to "balanced."

Model 1: Unigrams

	ACC	BER
Unbalanced	0.92570	0.80175
Balanced	0.85203	0.64918

Model 2: Unigrams Filtered by POS

	ACC	BER
Unbalanced	0.91749	0.76166
Balanced	0.83477	0.61306

Model 3: Bigrams

	ACC	BER
Unbalanced	0.91850	0.78646
Balanced	0.53375	0.55469

Model 4: Bigrams Filtered by POS

	ACC	BER
Unbalanced	0.91791	0.79340
Balanced	0.85700	0.59296

3.7. Analysis of Results

As we can see, the all performed similarly well when both classes is assumed to be the same weight. However, when we weigh the two classes differently by setting the class weight parameter to "balanced" (because the dataset leans more heavily into positive reviews), performance dropped considerably.

Of the four, however, we see that taking into consideration the parts of speech makes a difference. For unigrams, considering the parts of speech gives a lower accuracy, but also yields a lower BER. The siginificance of parts of speech is more apparent when considering bigrams. When we changed the class weight parameter to balanced, the model not considering parts of speech structure dropped its accuracy from 91% to 53%. But the model when consider parts of speech only dropped from 91% to 85%. Moreover, it retains a relatively small BER among the four models.

This suggests that when performing sentiment analysis, it is not enough to naively take the top occurring n-grams to train on. This is because such a method does not take into account review texts that employ sarcasm. So, it is necessary to also analyze the structure of a review text.

4. Related Works

From the url `https://cseweb.ucsd.edu/~jmcauley/datasets.html`, there were 2 research papers relating to our research: *Item recommendation on monotonic behavior chains* (Wan & McAuley, 2018) and *Generating and personalizing bundle recommendations on Steam* (Pathak et al., 2017). Both of these research papers and our paper share the same predictive task, which is to predict a user's recommendation. "Generating and personalizing bundle recommendations on Steam" also shared a similar approach with ours in which they used ranking of bundle to create bundle recommendations and use Bayesian Personalized Ranking, while ours used ranking of uni-grams in review text and Naive Bayes to predict whether the user would recommend the game or not.

Another similar research that we found was called *Sentiment Analysis of Steam Review Datasets using Naive Bayes and Decision Tree Classifier* (Zuo, 2018). Similar to ours, the research used review texts as uni-grams, recorded their tf-idf, and applied Gaussian Naive Bayes model to predict a Steam market report. From our research paper and the others, we could imply that this dataset can be used to create many predictions for accuracy as well as recommendation for the user. It also shows that Naive Bayes is commonly used on this dataset. With these articles in mind, we can see that the Naive Bayes model should preform very well in predictive tasks such as sentiment analysis. In our case, we are unable gauge its performance because of the imbalance and thus over-fitting to our dataset.

5. Conclusion and Future Work

5.1. Conclusion

To conclude our result, the Naive Bayes holds the best BER out of all. However, because of the unbalanced in the data, we believe this might an inaccurate representation for predicting data because the Naive Bayes prediction outputs where all true. So, this naive bayes model would not generalize well in practice.

The second best model was model 4, Logistic Regression that uses a set with 1000 most common bigrams, no stopwords, and filtered by POS. This was thanks to the removal of unnecessary words and a better bigram structure.

5.2. Further Expansion

There are many more models that we can create to further expand the sentiment analysis on this dataset. For the sake of time our tested models were prioritized but can be further improved.

5.2.1. Unigrams + Trigrams

One natural extension is to apply the idea of model 4 to trigrams, where we extract trigrams following a certrain structure. In terms of unigrams, we can extend model 1 and model 2 by considering only the context around the word.

5.2.2. SVM

Our models used linear regression but instead could have been tested using Support Vector Machines (SVM). With SVMs our text examples would be used as points in a multidimensional space. For this type of model it would be best to change our features into their tf-idf representations. During our testing this was attempted but would take to long to run on our dataset. In the future we could instead switch to linear Support Vector classification which scales better with bigger datasets.

5.2.3. Ensemble Model

Our models were each individually created but another approach that can be made is creating a model which combines other models. This is known as an ensemble model where the predictions of each model are taken into account and aggregates the predictions to create one final prediction. Since this approach uses multiple model algorithms, it is possible that a better result can be achieved. This may not be the case though as there is the possibility of the models being used over-fitting individual and thus affecting the ensemble model.

5.2.4. Other Features and datasets

For our models we used the review texts of all the different reviews in the dataset. We could further expand this by collecting more data and changing which features we used. One particular feature that may be helpful is hours played. For a specific review if we are able to collect the amount of hours played by the user we can determine the importance of any particular review and include that in our features. Another feature that we could have used is the tag "Helpful" which was included in the dataset but left out in our models. This "Helpful" tag signifies how helpful this review was for other steam users who have read the review. This value could improve our result since a helpful review is more likely to have significant sentiment.

References

Pathak, A., Gupta, K., and McAuley, J. Generating and personalizing bundle recommendations on steam. pp. 1073–1076, 08 2017. ISBN 978-1-4503-5022-8. doi: 10.1145/3077136.3080724.

Turney, P. D. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pp. 417–424, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073153. URL https://doi.org/10.3115/1073083.1073153.

Wan, M. and McAuley, J. Item recommendation on monotonic behavior chains. In *Proceedings of the 12th ACM Conference on Recommender Systems*, RecSys '18, pp. 86–94, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450359016. doi: 10.1145/3240323.3240369. URL https://doi.org/10.1145/3240323.3240369.

Zuo, Z. Sentiment analysis of steam review datasets using naive bayes and decision tree classifier. 2018.