CS 5830 - Project 5

Chris Brown, AJ Walters

INTRODUCTION

In this analysis, we investigated Naive Bayesian algorithms for use in predicting the rating of an Amazon review from its title and the text of the review. This tool could be useful for the Amazon data engineering team as they seek to identify reviews that were written by bots or by vendors trying to boost their reviews. This project assumes that all the reviews analyzed were written by verified customers. In this project, we hoped to identify patterns among these verified customers that could be compared to a dataset written by bots or by vendors. The comparison to bots or vendors was not attempted, and represent future applications of this analysis. This tool could also be used to improve rating and recommendation algorithms for Amazon data scientists.

DATASET

A dataset of Amazon reviews was created by Xiang Zhang of NYU, which was used in the following paper: Character level Convolutional Networks for Text Classification. Advances in Neural Information Processing Systems Xiang Zhang, Junbo Zhao, Yann LeCun. Previous researchers scraped about 35 million Amazon reviews spanning 18 years ending in March 2013. Xiang et al. then randomly sampled 600,000 training and 130,000 testing samples for each review score 1 to 5, including the title and text of the review. This resulted in a dataset of 3 million training samples and 650,000 testing samples. They then used one-hot encoding and a convolutional network to build a model from the dataset and compared their results to traditional methods (handcrafted features such as bag of words) and deep learning methods (such as word2vec).

This dataset was still massive for our computers to run. As such, we reduced the size of the dataset by further randomly sampling 10% of the data. As such, our training data encompassed 300,000 samples and our testing data encompassed 65,000 samples. In this process we dropped the nan values using dropna(). The training data was reduced by 6 samples, and the testing data was unaffected.

The text data was cleaned by removing characters that were not in the traditional alphabet and making everything lowercase. Next, the words were stemmed using the natural language toolkit (nltk) stopwords “english” library. The title and review were then merged and output as a CSV. This CSV was then used for downstream analysis, and the cleanup code was commented out. This cleanup process was also attempted using the nltk.sent_tokenize and word_tokenize. However, this did not resolve punctuation and numeric characters.

ANALYSIS TECHNIQUE

To analyze the data, the CountVectorizer package was used from sklearn.feature_extraction.text. This also used the english stop_words package. This package was then used to create a vectorized version of the merged title and review text into a training set. The review rating was then separated as a class label for training and confirmation. This set was split using the train_test_split method.

Next, a Bernoulli Naive Bayes model was trained using the sklearn.naive_bayes BernoulliNB() method. The method was verified using the sklearn.metrics classification_report method. Seaborn was then used to visualize the distribution of the classification.

A Multinomial Naive Bayes model was trained and verified using the same method.

RESULTS

Figure 1 presents the distribution of ratings from the source data. As previously described, a balanced sample set was generated randomly before applying the naive bayes models. This avoided bias from oversampling one single rating category.

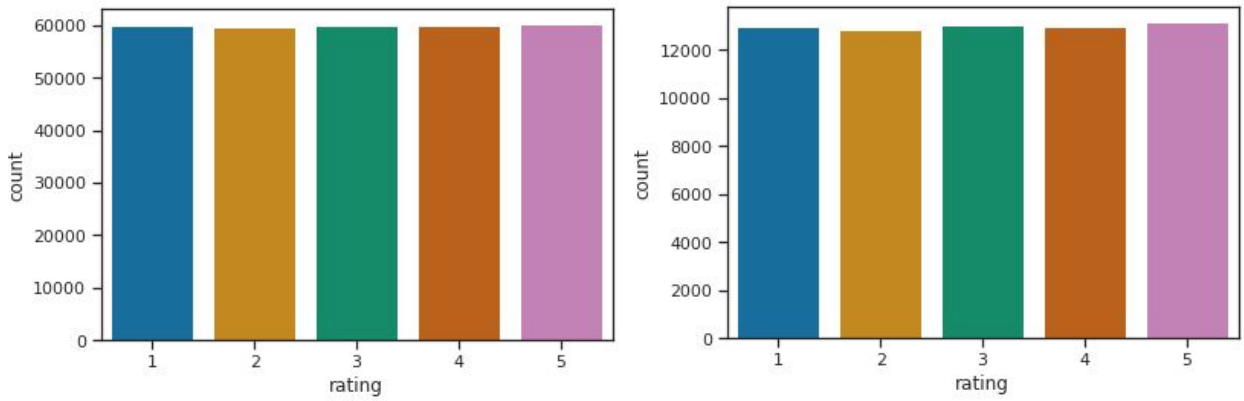

Figure 1. Distribution of each rating in the train (left) and test (right) data

In Figure 2, the precision, recall, F1-score and support for the Bernoulli Naive Bayes model is presented. It was noted that the F1-score improved for reviews rated 1 and 5, but was about 0.36 for the reviews rated 2-4. This is likely because most reviews for 1 star reviews contain strongly negative language, and 5 star reviews contain strongly positive language. The language of reviews 2-4 stars are often more ambiguous in nature. Figure 2 also presents the counts of each review as predicted by the Bernoulli Naive Bayes model.

Figure 3 shows the classification report and distribution of predicted ratings for the Multinomial Naive Bayes Model. This model performed slightly better on the 2 star reviews, but performed similarly to the Bernoulli model for the 1 and 3-5 star reviews.

Bernoulli	Precision	Recall	F1-Score	Support
1	0.52	0.66	0.58	15075
2	0.41	0.33	0.37	14852
3	0.40	0.34	0.37	14789
4	0.43	0.30	0.35	15215
5	0.50	0.67	0.57	15068

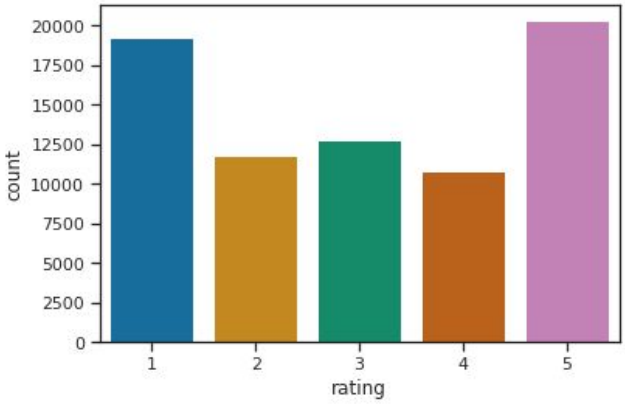

Figure 2: Classification Report and Prediction distribution for Bernoulli Naive Bayes Model

Multinomial	Precision	Recall	F1-Score	Support
1	0.53	0.63	0.57	15075
2	0.40	0.37	0.39	14852
3	0.38	0.35	0.36	14789
4	0.41	0.34	0.37	15215
5	0.55	0.60	0.58	15068

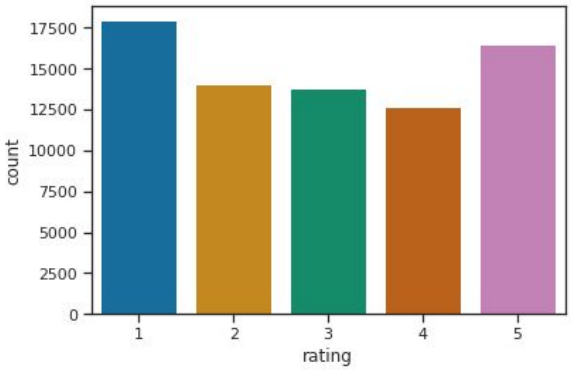

Figure 3: Classification Report and Prediction distribution for Multinomial Naive Bayes Model

CONCLUSIONS

From our analysis, the Amazon data engineering team can find what words cause reviews to have low or high ratings. While we did not get great results, there are other methods that can produce great results. One is using logistic regression with the reviews vectorized. We believe that our model did not do well because Naive Bayes assumes all variables are independent, but words are highly dependent. One approach we could take to improve our Naive Bayes models would be partitioning the space of the word vectors and binning these word vectors into frequencies, but that still does not avoid the issue of dependence.