Adithyan Subramanian
Capstone Project 2 Final report
Mentors: Mukesh Mithrakumar

Predicting Amazon Ratings Based on Review Text

Problem Statement

Amazon, the Seattle based E-commerce company has earned its title as a “tech giant.” As of 2019, the company sells over 100 million products on its website[1], and over 4000 orders are placed every minute[2]. Many consumers who have ordered a product from Amazon will also write a review of these products, which is used by other consumers to decide if they should buy the product. In this project, we aim to use the reviews of Amazon’s inhouse brand, AmazonBasics, to perform a sentiment analysis of the reviews and write an algorithm that will predict review ratings based on the text of the reviews.

Data Set

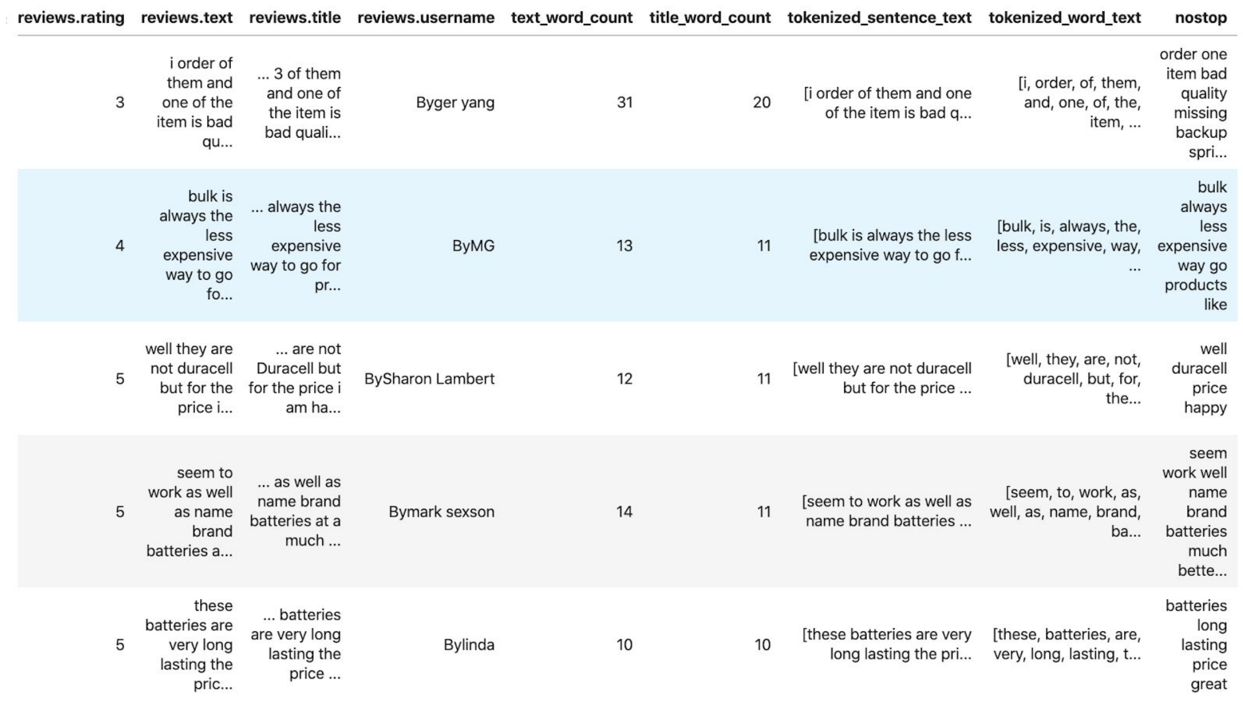
This project will utilize one dataset from data.world (https://data.world/datafiniti/consumer-reviews-of-amazon-products). This data set has approximately 28,332 rows, each corresponding to a single review. The dataset houses the product that is being reviewed, the date it was reviewed, the review title, the text of the review, as well as the review rating. All data is available as CSV files.

Data Wrangling

The data cleaning and wrangling was relatively straightforward. We started by dropping any irrelevant columns to our analysis, such as URLs. When we looked at only the relevant columns, there were no nan values to alter. This made the cleaning process easier. Once the dataset was clean, we started to process the text data.

Using the NLTK library, we deleted any numerical values, as well as periods and commas. We chose to keep question marks and exclamation points because we believe these might show doubt and excitement, respectively. These emotions may be correlated with ratings.

We then tokenized the reviews by sentence and words. For the purpose of this project, we will be primarily using the tokenized word column. We then removed any and all stop words that were in the reviews. We then created another column, with word counts of each review to aid our analysis. Below is a screenshot of part of the final dataframe.



reviews.rating	reviews.text	reviews.title	reviews.username	text_word_count	title_word_count	tokenized_sentence_text	tokenized_word_text	nostop
3	i order of them and one of the item is bad qu...	... 3 of them and one of the item is bad quali...	Byger yang	31	20	[i order of them and one of the item is bad q...	[i, order, of, them, and, one, of, the, item, ...	order one item bad quality missing backup spri...
4	bulk is always the less expensive way to go fo...	... always the less expensive way to go for pr...	ByMG	13	11	[bulk is always the less expensive way to go f...	[bulk, is, always, the, less, expensive, way, ...	bulk always less expensive way go products like
5	well they are not duracell but for the price i...	... are not Duracell but for the price i am ha...	BySharon Lambert	12	11	[well they are not duracell but for the price ...	[well, they, are, not, duracell, but, for, the...	well duracell price happy
5	seem to work as well as name brand batteries a...	... as well as name brand batteries at a much ...	Bymark sexson	14	11	[seem to work as well as name brand batteries ...	[seem, to, work, as, well, as, name, brand, ba...	seem work well name brand batteries much bette...
5	these batteries are very long lasting the pric...	... batteries are very long lasting the price ...	Bylinda	10	10	[these batteries are very long lasting the pri...	[these, batteries, are, very, long, lasting, t...	batteries long lasting price great

Fig. 1

Exploratory Data Analysis

To fully understand the dataset, it is important to understand the distribution of the ratings. As the histogram on the right shows, the distribution has a very strong left skew. What this means is that the vast majority (almost 20,000 of the 28,332) are 5-star reviews. 4-star reviews are the next highest distribution with roughly 6,000 reviews. The dataset is largely filled with positive reviews. Since this sample is not normal, we cannot definitively state that this sample is representative of the population of

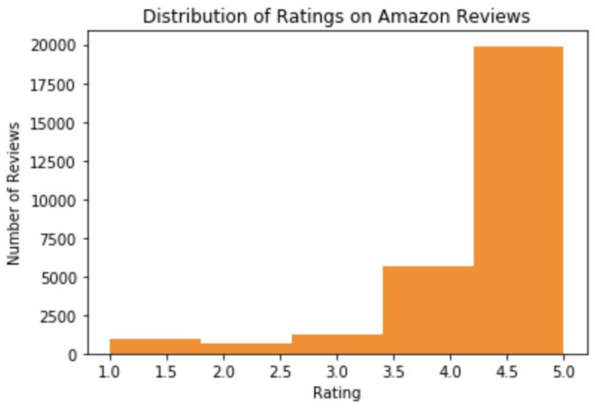


Fig. 2

Amazon reviews. For the purposes of this project, we will proceed as if this distribution is indicative of the entire population. Another aspect to note with regards to distribution is that we may run into challenges in creating our model since we have so few instances of low ratings. The class imbalance is a significant issue and future work may focus on solving the class imbalance, either through oversampling methods or with new data. For now, we will proceed as normal, and if the model signals any need to address distribution issues, we will do so as required.

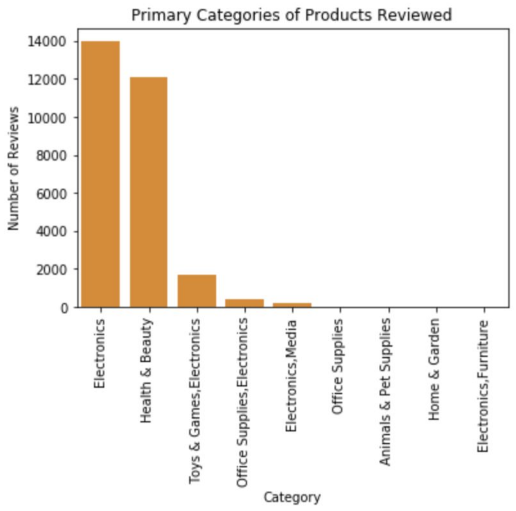


Fig. 3

Next, we wanted to see which category of products these reviews were for. The bar chart above shows that close to half of all reviews in the dataset are of electronics. While the next most populous category is health and beauty.

Finally, we wanted to see if word count had any correlation with ratings. Using the word count column we created in the data wrangling section, we created the following boxplot.

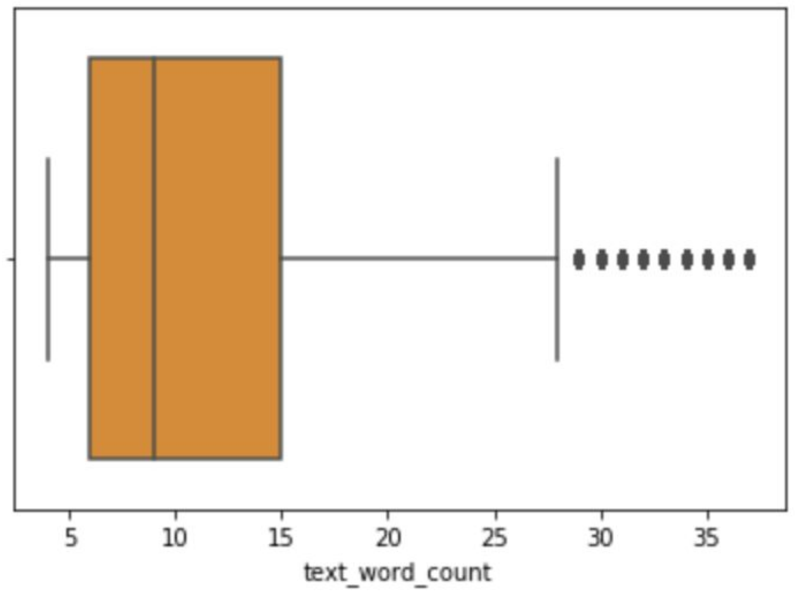
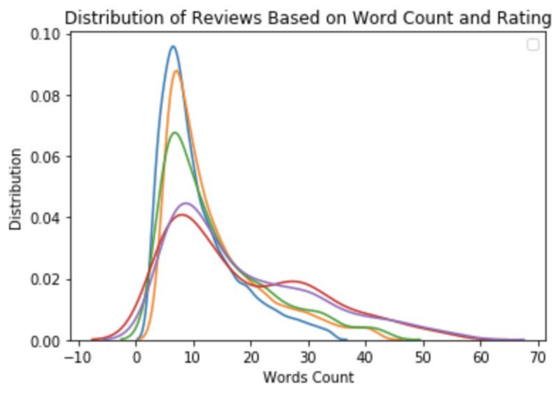


Fig. 4

It is important to note that in order to create a legible boxplot we had to remove outliers. The maximum word count was 719. While the boxplot gives a good visualization of the distribution, it does not give any information on how word count affects ratings. In order to understand this, we used distribution plots.



As seen on the right, all 5 distributions follow the same general trend, with a peak at around 10 words per review. There seems to be no correlation with word count and rating.

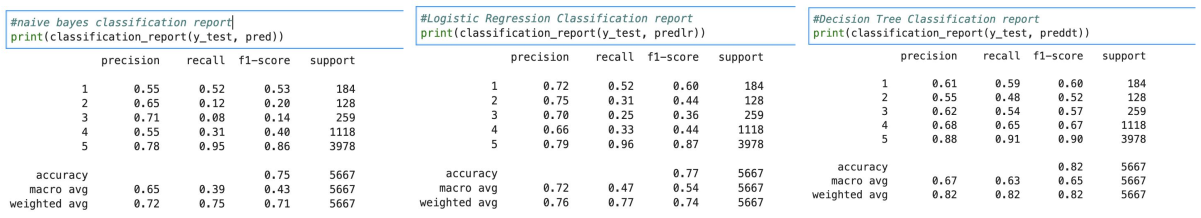
Fig. 5

Initial Processing

To start we split the dataset into train and test sets, with an 80/20 split. When we had the train and test sets, we then initialized a countervectorizer and tfidfvectorizer to be used for machine learning. We used the vectorizers on the train and test set and created dataframes to be used for machine learning.

Initial modeling

In order to determine which model would be best for this dataset, we initially created three models: a Naive Bayes model, a logistic regression model, and a decision tree model. The classification report for all three models can be seen below



```
#naive bayes classification report
print(classification_report(y_test, pred))

              precision    recall  f1-score   support

           1       0.55      0.52      0.53       184
           2       0.65      0.12      0.20       128
           3       0.71      0.08      0.14       259
           4       0.55      0.31      0.40      1118
           5       0.78      0.95      0.86      3978

    accuracy                           0.75      5667
   macro avg       0.65      0.39      0.43      5667
weighted avg       0.72      0.75      0.71      5667
```

```
#Logistic Regression Classification report
print(classification_report(y_test, predlr))

              precision    recall  f1-score   support

           1       0.72      0.52      0.60       184
           2       0.75      0.31      0.44       128
           3       0.70      0.25      0.36       259
           4       0.66      0.33      0.44      1118
           5       0.79      0.96      0.87      3978

    accuracy                           0.77      5667
   macro avg       0.72      0.47      0.54      5667
weighted avg       0.76      0.77      0.74      5667
```

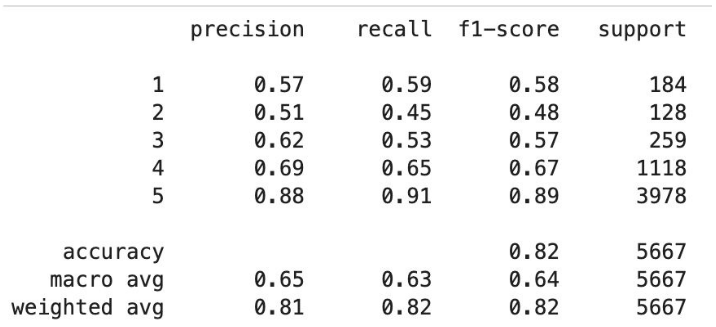
```
#Decision Tree Classification report
print(classification_report(y_test, preddt))

              precision    recall  f1-score   support

           1       0.61      0.59      0.60       184
           2       0.55      0.48      0.52       128
           3       0.62      0.54      0.57       259
           4       0.68      0.65      0.67      1118
           5       0.88      0.91      0.90      3978

    accuracy                           0.82      5667
   macro avg       0.67      0.63      0.65      5667
weighted avg       0.82      0.82      0.82      5667
```

Fig. 7

As can be seen, the decision tree model had the most accuracy, so we chose to focus our efforts on optimizing that model.

Model optimization

Using a GridSearchCV to find the best parameters, we found that the most accurate model, would be one with a max_depth of 5000, and a minimum samples split of 2. This model gave us a training set accuracy of 0.984, and a test set accuracy of 0.818. The classification report can be seen to the left

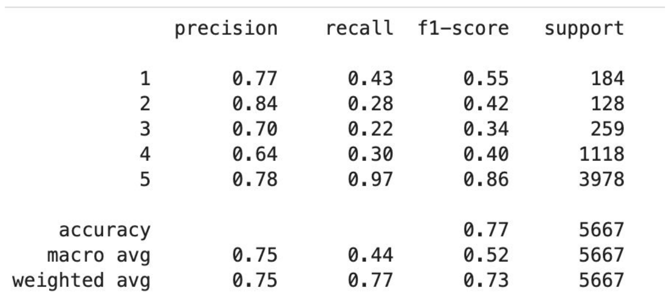


```
              precision    recall  f1-score   support

           1       0.57      0.59      0.58       184
           2       0.51      0.45      0.48       128
           3       0.62      0.53      0.57       259
           4       0.69      0.65      0.67      1118
           5       0.88      0.91      0.89      3978

    accuracy                           0.82      5667
   macro avg       0.65      0.63      0.64      5667
weighted avg       0.81      0.82      0.82      5667
```

Although noting a high level of accuracy, the difference between the train test accuracy and test set accuracy was concerning for overfitting. We then decided to alter the parameters to decrease the difference between training set accuracy and the test set accuracy in order to lower overfitting.

Fig. 8

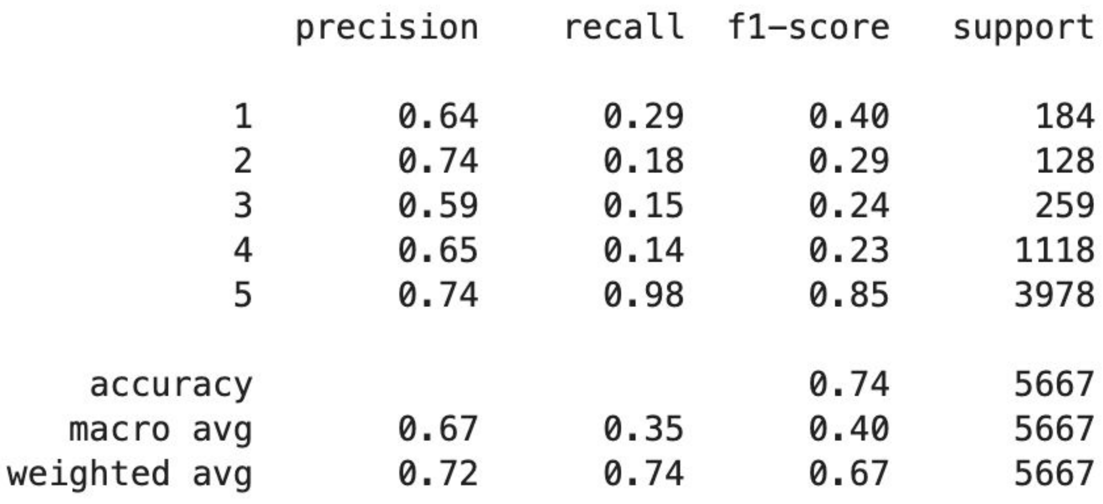
Initially, we attempted to introduce some noise to the model using a ridge classifier. This gave us a training set accuracy of 0.839 and a test set accuracy of 0.766. The classification report can be seen below.



```
              precision    recall  f1-score   support

           1       0.77      0.43      0.55       184
           2       0.84      0.28      0.42       128
           3       0.70      0.22      0.34       259
           4       0.64      0.30      0.40      1118
           5       0.78      0.97      0.86      3978

    accuracy                           0.77      5667
   macro avg       0.75      0.44      0.52      5667
weighted avg       0.75      0.77      0.73      5667
```

Fig. 9

The difference between the training set accuracy and the testing set accuracy is still indicating overfitting, so we will return to the parameters of the decision tree, and see if we can decrease the difference.
After much trial and error, we were able to determine that the parameters with the least likeliness for overfitting were a maximum depth of 20, and a minimal samples split of two. This yielded a training set accuracy of 0.775 and a testing set accuracy of 0.736. The classification report can be seen below.



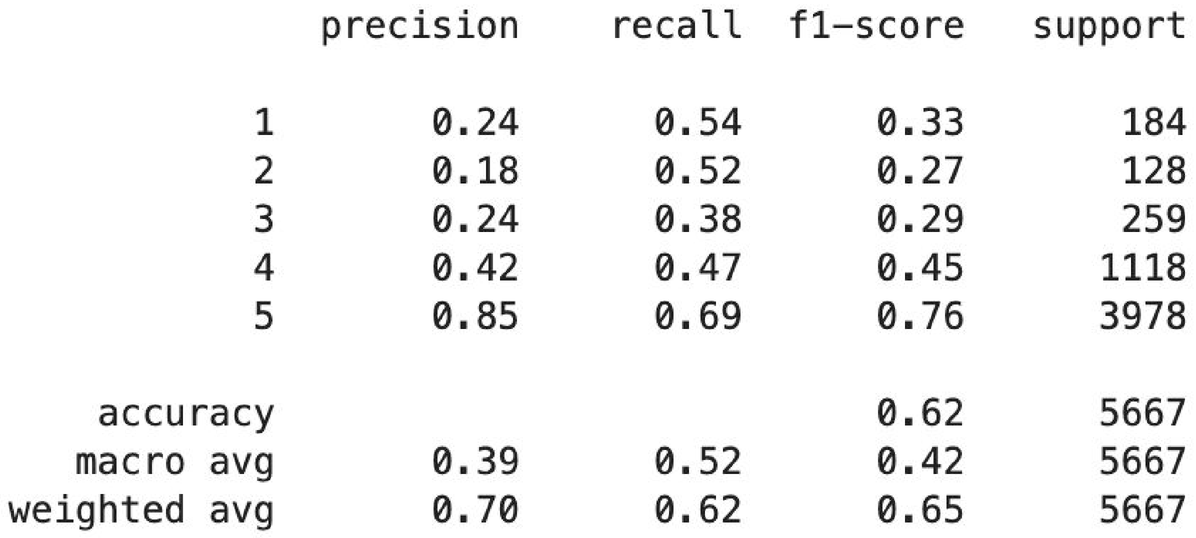
```
              precision    recall  f1-score   support

           1       0.64      0.29      0.40       184
           2       0.74      0.18      0.29       128
           3       0.59      0.15      0.24       259
           4       0.65      0.14      0.23      1118
           5       0.74      0.98      0.85      3978

    accuracy                           0.74      5667
   macro avg       0.67      0.35      0.40      5667
weighted avg       0.72      0.74      0.67      5667
```

Fig. 10

This seems to have solved the overfitting issue, but now the accuracy is low. As mentioned in the previous status report, the dataset is heavily skewed towards 5-star reviews. In order to compensate for this, we decided to employ some oversampling methods.

Oversampling

Given the issues of overfitting and accuracy due to the class imbalances, it may be prudent to do some work with oversampling methods. We used the SMOTE method to equalize all review ratings from 1 to 5 stars. To accomplish this we used the ‘not majority’ setting to resample all reviews with 1-4 star ratings. Pairing this resampling with the ridge classifier, we got a training set accuracy of 0.627 and a testing set accuracy of 0.624. The classification report can be seen below.



```
              precision    recall  f1-score   support

           1       0.24      0.54      0.33       184
           2       0.18      0.52      0.27       128
           3       0.24      0.38      0.29       259
           4       0.42      0.47      0.45      1118
           5       0.85      0.69      0.76      3978

    accuracy                           0.62      5667
   macro avg       0.39      0.52      0.42      5667
weighted avg       0.70      0.62      0.65      5667
```

Fig. 11

This completely solved the overfitting issue, but now the accuracy was very low. This is not an ideal model. Thus, we chose to recommend the decision tree with a maximum depth of 20, and a minimal samples split of two as the best model to predict ratings based on review text. This model has a low indication of overfitting with only a train/test accuracy difference of 0.04. The test accuracy is also at an acceptable level of 0.74.

Future Works

Some next steps for this project would be to get more data on low rated reviews. The lack of 1 and 2-star reviews hamstrung any model that was attempted. It may help to scrape for 1 and 2-star reviews on Amazon in order to get a more diverse dataset. Finally, more time could be spent on both feature and parameter optimization.

Conclusion

The goal of this project was to predict amazon review ratings based on the text of the review. Our chosen model had an accuracy of 0.74. This is significantly better than an expected pure chance accuracy of 0.20. Due to the uneven data set, 5-star reviews had a very high accuracy of 0.85. The rest of the rating classifications were also above chance. We consider this evidence of a successful model. Attempts at solving class imbalance issues were largely fruitless. Using SMOTE as an oversampling method to solve the class imbalance solved overfitting at the cost of accuracy. We believe that our current model with limited overfitting and higher accuracy is a much more usable model. The next steps would be to introduce a more diverse dataset and to further optimize the model.

Works cited:

[1] https://www.scrapehero.com/number-of-products-on-amazon-april-2019/
[2]https://d39w7f4ix9f5s9.cloudfront.net/61/3b/1f0c2cd24f37bd0e3794c284cd2f/2019-amazon-smb-impact-report.pdf