Amazon Alexa Reviews Classification

Capstone Project 2
Milestone Report 2

Catherine Somers

Problem Statement

The purpose of this project is to create and train a model using Amazon Alexa reviews. As a main focus we will be using sentiment analysis on the data to help determine how Amazon Alexa products can be improved upon. By doing so, we will need to determine and predict the significance to the overall reception to all Amazon Alexa related products. We can to some extent grasp and have insight on positive and negative reflecting reviews. From the business perspective this is useful to either the product team or marketing team when trying to position future versions in the entire product line. Based on the insights they inquire from the reviews, a more informed decision can be made for a new version of an existing product. A different but also related use would be to detect which features within the data determine what is classified as positive or negative feedback. Product managers usually use reviews as a way to gauge how they could further improve future product versions down the line. With each new product iteration, an existing functionality is almost always made better to improve the user's overall experience using the product. In addition, while this report may focus directly on Amazon Alexa reviews, the techniques used here could also be applied to any product review dataset with a rating system and text reviews.

Data Acquisition and Wrangling

The first step in preparing for the analysis of Amazon Alexa reviews was to search for an appropriate dataset. From a business standpoint, the overall sentiment of a product in a product line is important. Given the nature of how voice assistants have shaped the scope of how we live our lives, I wanted to explore the ways how these products have improved the quality of life. What made me decide upon specifically Alexa enabled devices was more to do with the interest in seeing what people had to say about Amazon's Alexa over the other competing voice assistants that exist.

I was able to find data for Amazon Alexa products that provides input data, ratings, review date, product variant, and feedback for a number of various Alexa products. I looked on Kaggle as it is one of the largest platforms to find public data and an overall great place to find data to solve a problem using classification. The dataset I chose to work with was last updated 2 years ago, but we can still get a snapshot of the impressions customers got within the timeframe of the reviews.

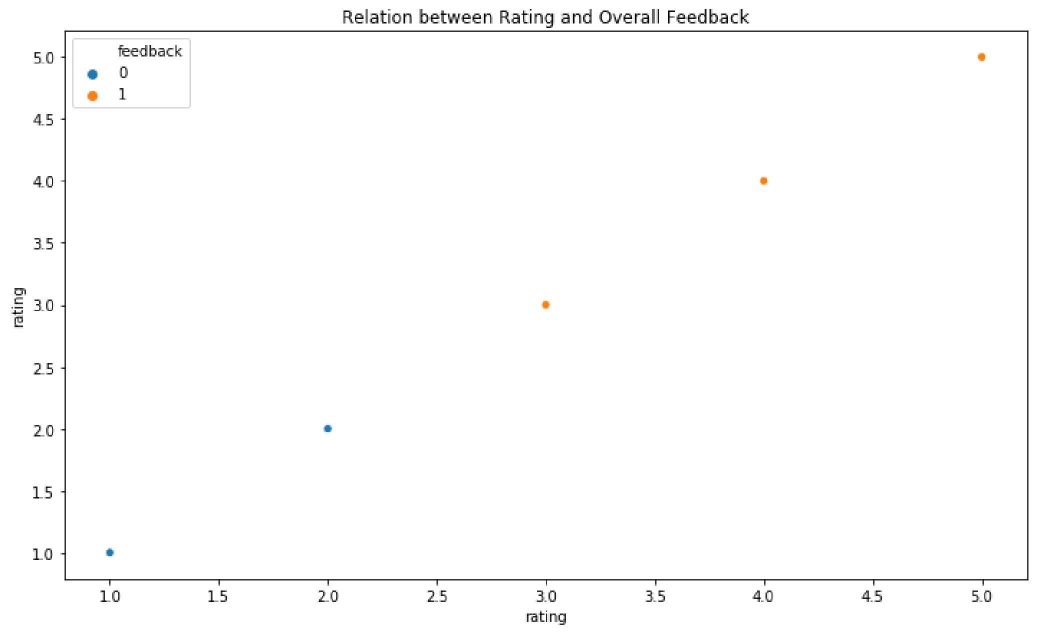
The first step in working with this data from Kaggle was downloading the data file. It is a tsv file that contains 3150 values for ratings on Amazon Alexa products from the last two years. Once I had the file downloaded, I pulled it into a notebook the same way you would read in a csv file. After I converted the tsv into a pandas dataframe, I did some initial examining to see if there were any null values. From an initial observation there was next to none null values within the data, so the data I would be working with would be fairly clean. The average rating for reviews across every product variation was 4.46. For the most part, it seems that most users have been happy with their devices.

To perform feature engineering later in the model, the year, the month, and day of the week were extracted into separate columns. Estimating the review length is also an important feature for classifying text in Natural Language Processing (NLP). As a result, a new column has been added to the data called review_length.

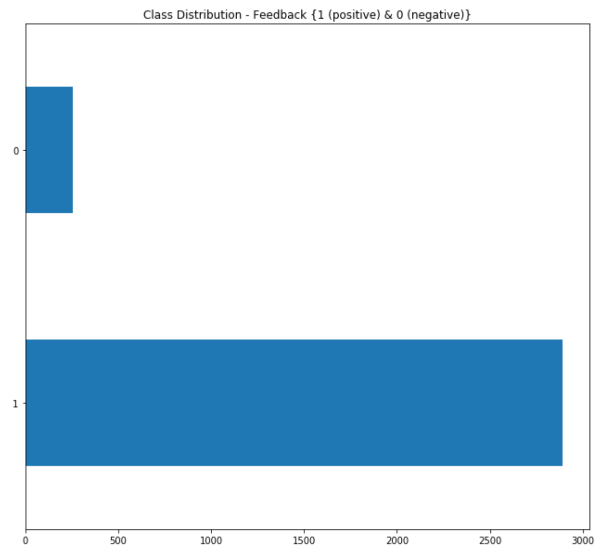
Data Exploration and Storytelling

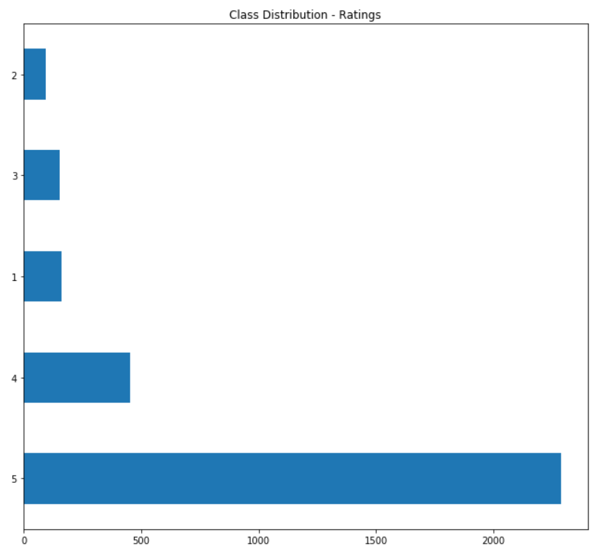
Upon completing the previous step, it was time next to explore the dataset through visualizations. I was hoping to find some patterns among the product variation and overall feedback. I also wanted to find out more about the relationship between the rating and overall feedback.

I began looking into the relationship between rating and feedback by plotting in a scatterplot comparing both. In order for feedback to be considered positive, its rating needs to be greater than 3.

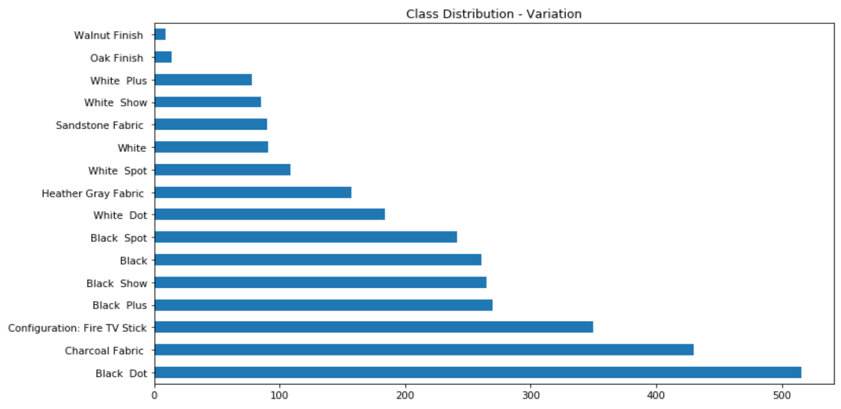


This gives more of a general overview on the correlation between rating and feedback. As we see here, the rating value range for negative feedback is 2 and below. Rating value range for positive feedback is 3 and above. I felt like this didn't give me as much insight as I would like so I wanted to look at feedback and ratings in their own respective bar plots to see how feedback and ratings are distributed in the dataset.

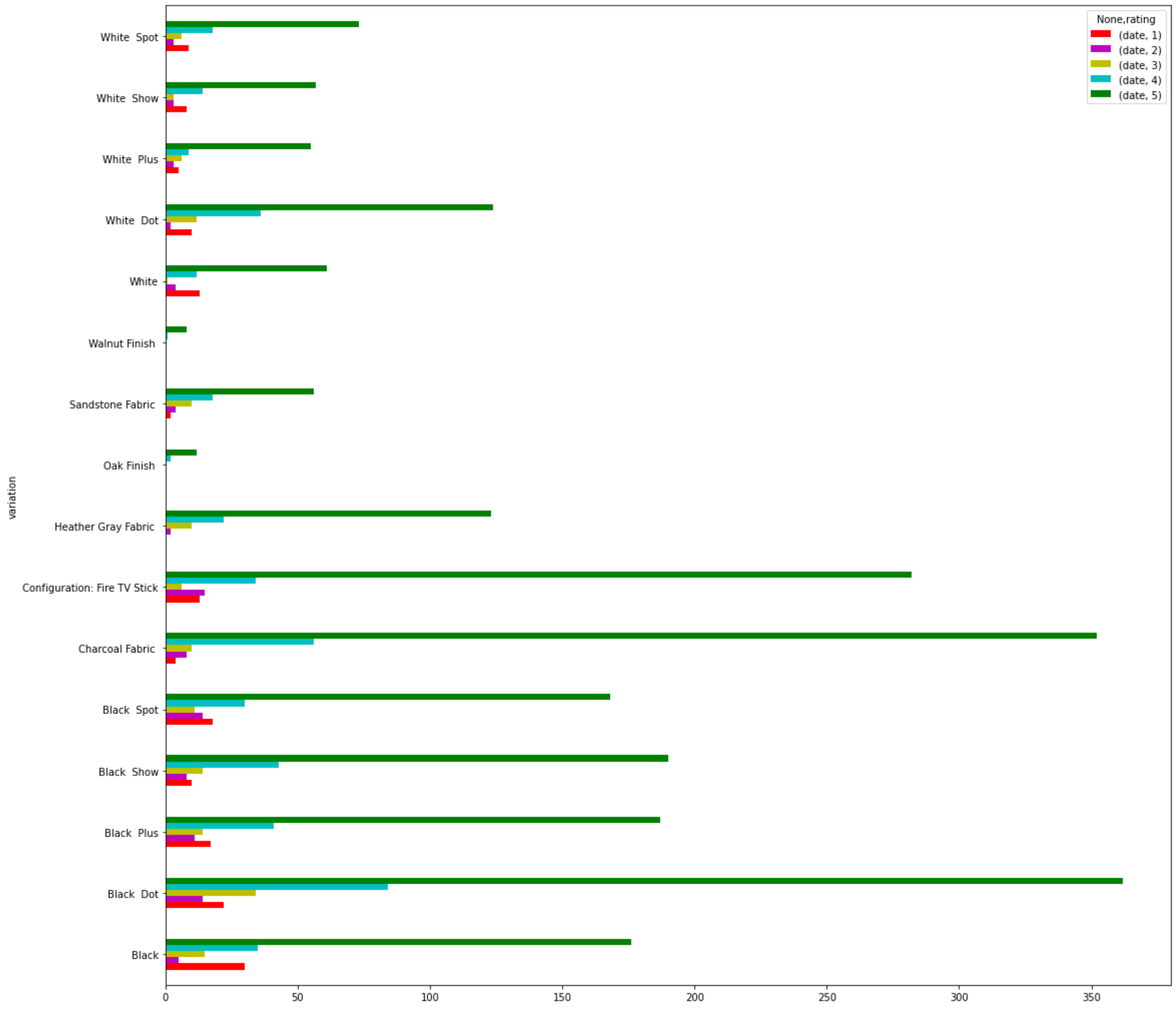




Looking at both cases here we see that the distribution among feedback and ratings is highly skewed on the positive side. For the most part, products have been well received by the customers. What may help in the future when it comes to reviewing models is stratifying the data to avoid a class imbalance. This is something to keep in mind when working with this kind of data and in future cases.



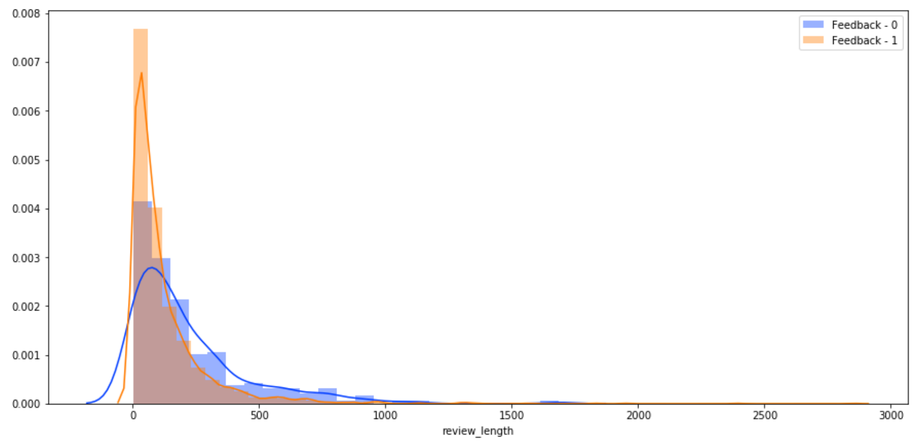
There are distinct bins here with each product type in the Alexa enabled line of devices show a pattern for the model preference. The Black Dot model is the most popular one out of all the different variations.



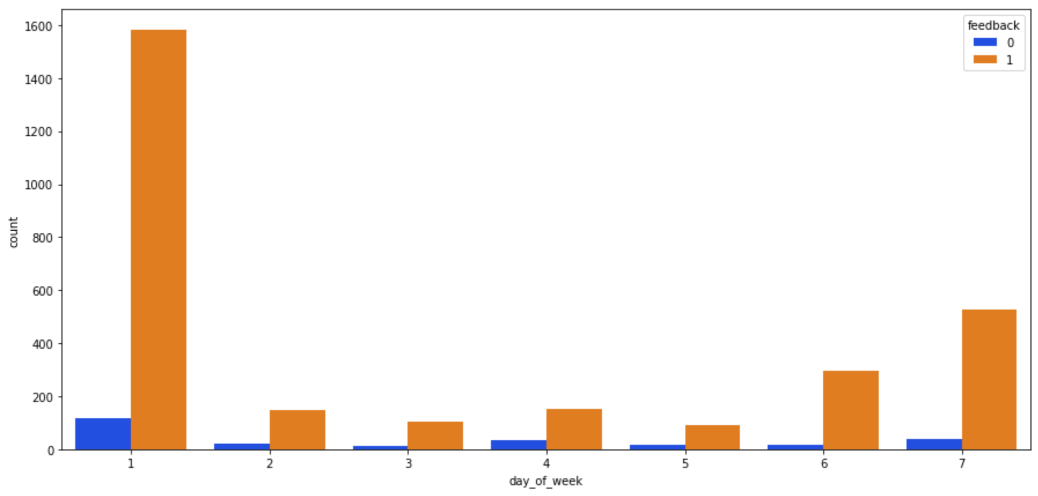
From the above plot, the Black Dot is the Amazon Alexa product with the most ratings of 5. This shows that more people may want to spend less than the normal version to experience using Alexa as a voice assistant in their home as the central control hub for a person's entertainment system, kitchen, or a children's room.

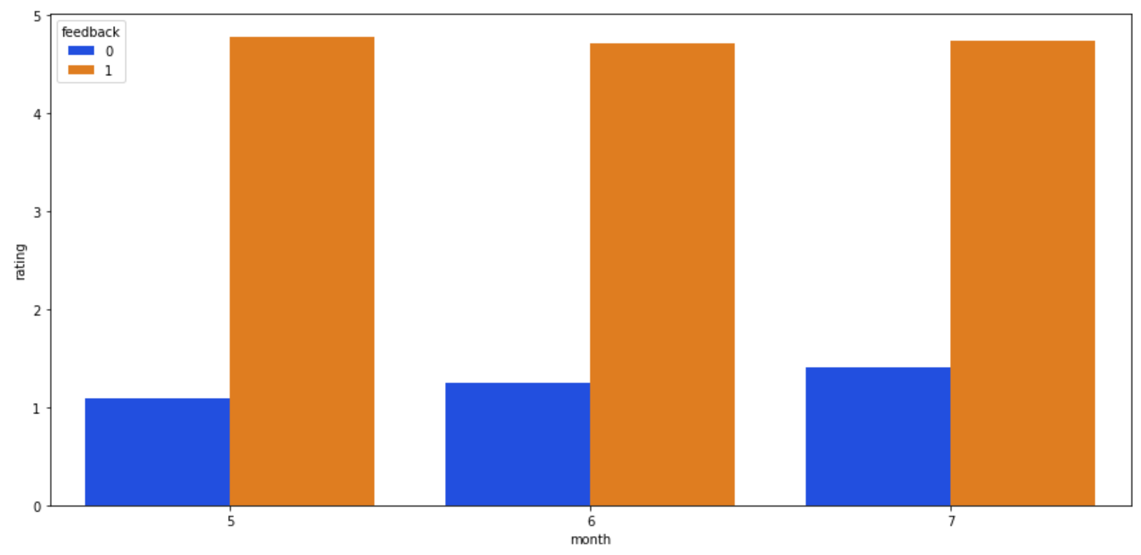
Application of Inferential Statistics

Once I explored the data visually, I began looking to perform statistical analysis on the review data. Exploring the relationship between review length and type of feedback. Looking at this plot, we see that customers with negative reviews write longer reviews.



In my data visualization notebook, I looked at the frequency of which day and month customers were more likely to write reviews on Monday and gave feedback for their devices mostly in the month of July.

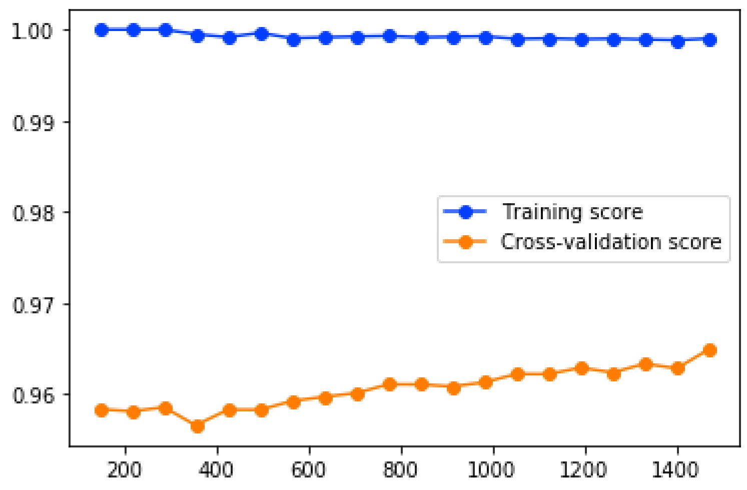




Predictive Modeling

For variation, one hot encoding was also used in my analysis. One hot encoding is a type of vector representation where all the vectors in a vector are 0, except for one, which has a 1 in its value. One important thing to note is that no matter how many dummy variables you end up is to make sure not to drop any one variable. Doing so may end up with a dummy variable trap. With that said, it means that there is the possibility of their being a case for perfect multicollinearity. This multicollinearity may occur when the independent variables in a regression model are correlated. The correlation becomes a problem due to independent variables needing to be independent. Coefficient estimates can swing based on which other independent variables are in the model. These coefficients can become very sensitive to small changes made to the model. Multicollinearity reduces the precision of the estimated coefficients. Due to this, I might not be able to trust any of the p-values to identify independent variables that are considered statistically significant.

I decided to look at using both random forest classifier and gradient boosting classifier in our analysis here. The data was split up into a training set and a testing set first before getting started with random forest. One of the most important methods of random forest classifier in sci-kit learn is feature_importances. Using Random Forest Classifier, I looked at the top 10 features (not pictured) and created our y_pred. From there, the learning curve was looked at between the cross validation score and training score for the model.



As a result from running the model with random forest, the following was the output.

```
Random Forest Classifier:
Accuracy Score:  0.9216931216931217
Precision Score:  0.9284940411700975
Recall Score:  0.9907514450867052
F1 Score:  0.9586129753914988
Confusion Matrix:
 [[ 14  66]
 [  8 857]]
```

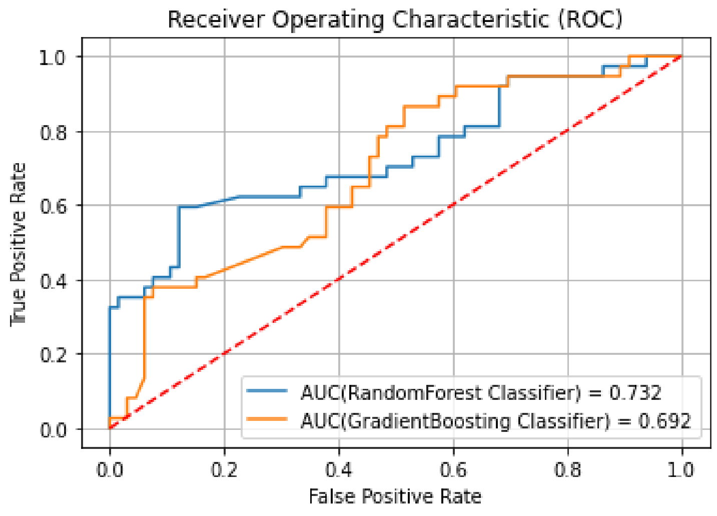
As a result from running the model with gradient boosting, the following was the output.

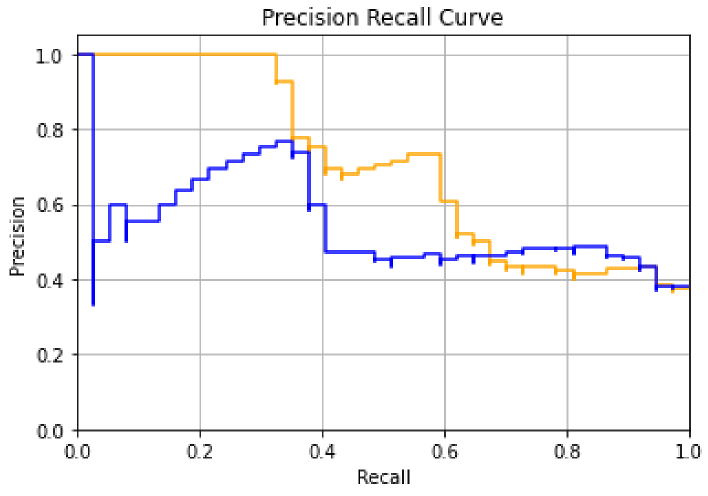
```
Gradient Boosting Classifier:
Accuracy Score:  0.928042328042328
Precision Score:  0.9289558665231432
Recall Score:  0.9976878612716763
F1 Score:  0.9620958751393535
Confusion Matrix:
 [[ 14  66]
 [  2 863]]
```

The models performed well using both methods and scored fairly high in each scoring criteria. From my analysis with the data so far, I can conclude that feature engineering is one of the most crucial steps when it comes to Natural Language Processing. It is now time to further dive into more Natural Language Processing.

Model Performance

I decided to split the dataset again in another notebook. This time specifically looking at just the bad reviews to help aid in how to better improve future product versions. The next step involved testing and training models using both Random Forest Classifier and Gradient Boosting Classifier as well. Our target variable selected was the ‘feedback’ column. When running the model using Random Forest, the accuracy of reviews being classified as negative was around 73%, the precision or the positive predicted value of reviews being negative was 71%, with the recall or sensitivity of relevant reviews being received was around 41%. Under Gradient Boosting, the accuracy of reviews being negative was around 69%, the precision was around 61%, and the recall was about 38%

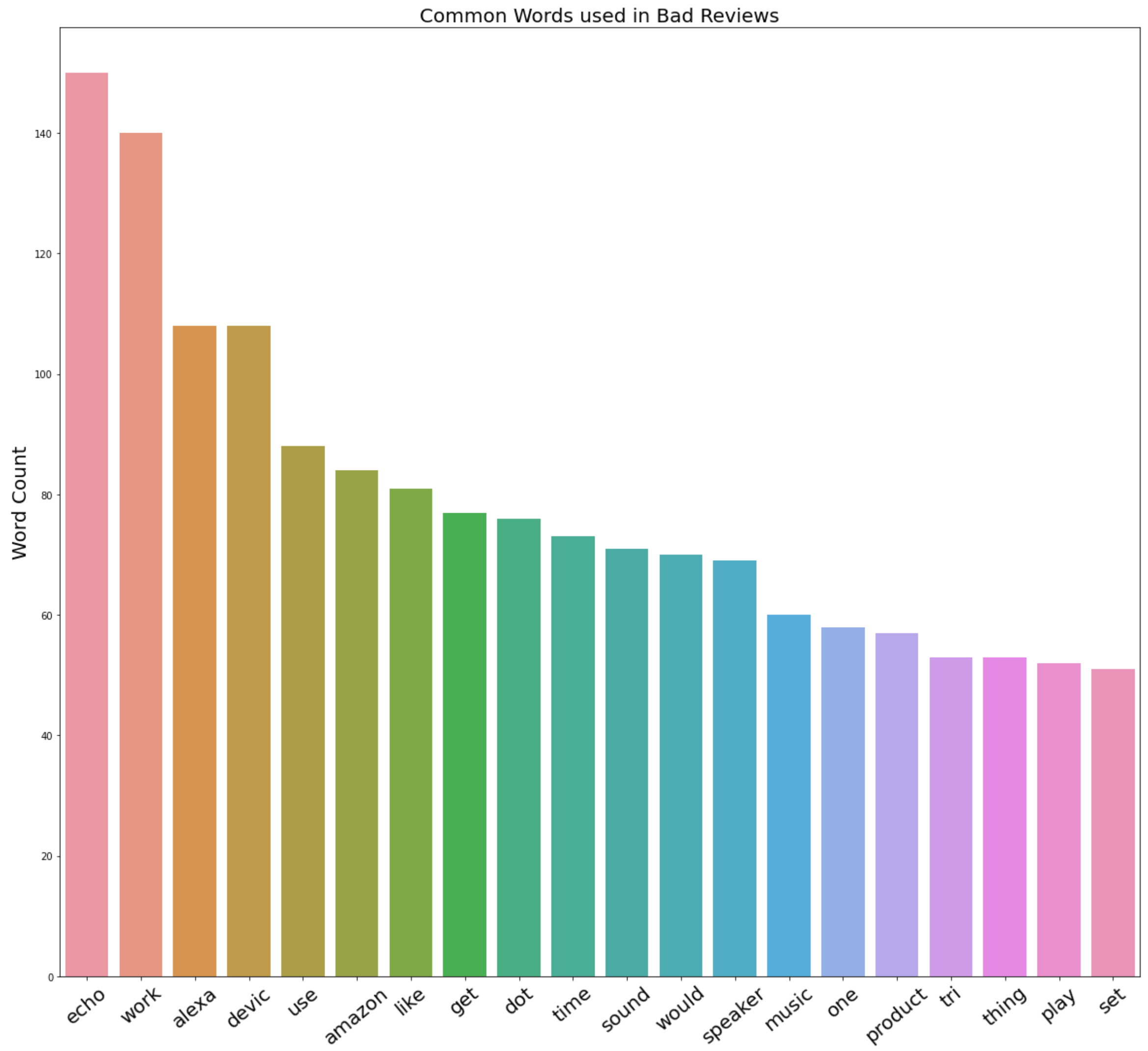




Random Forest: The area under the curve was reported as around 0.73 and the F1 score was around 0.51

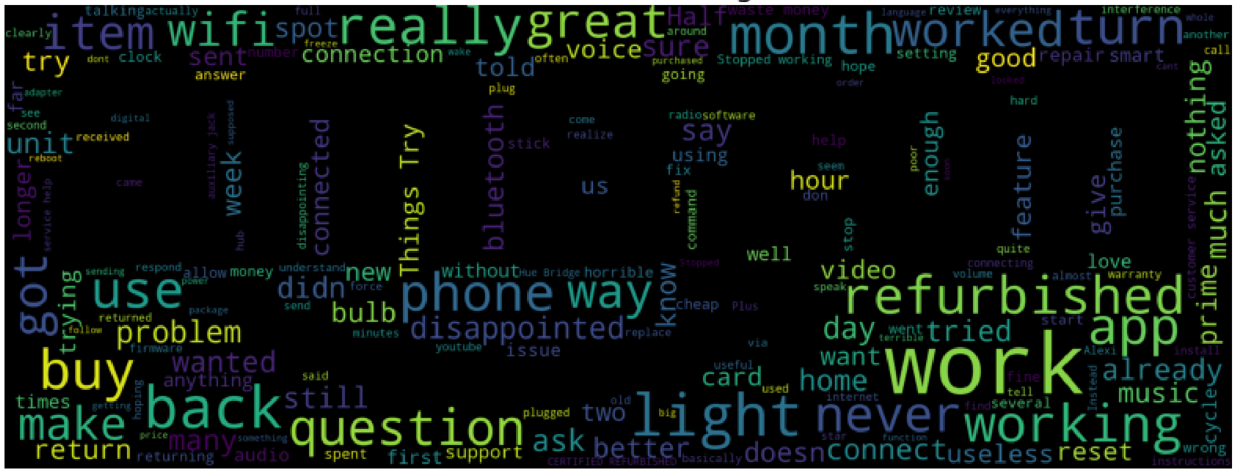
Gradient Boosting: The area under the curve was reported as around 0.69 and the F1 score being about 0.47.

Sentiment analysis with BERT was also used in my assessment of this review dataset. As a result, looking at each feedback class, the accuracy for positive reviews being predicted was 431/434 and negative reviews as 16/39. This sort of makes sense being that there is a larger volume of positive feedback. It does not help us enough though with gaining more insight on negative feedback. Time to go even deeper and look at word frequency with words in ‘verified_reviews.

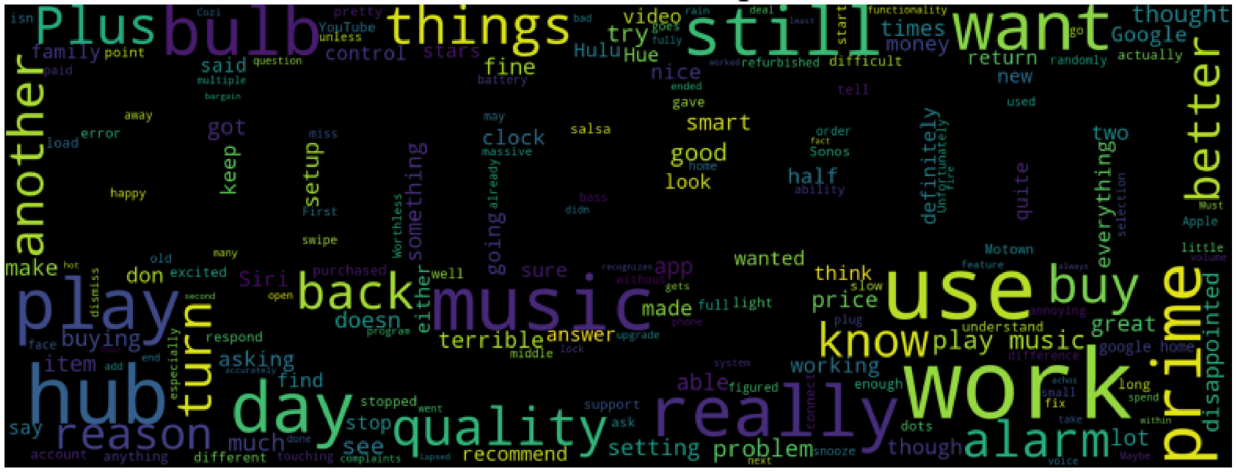


As shown above, the highest count of common words in bad reviews is echo and work. They have been used at least 150 and 140 times in written product reviews across all product variations. A Tfidvectorizer was used to get a general idea of which words show up when using feature extraction from the text. I could also get a look at which words were used the most in the majority of negative feedback using wordclouds. To broaden out the frequency of words, reviews with rating 3 were also included since a rating of 3 is not fully positive and could still be classified as negative feedback.

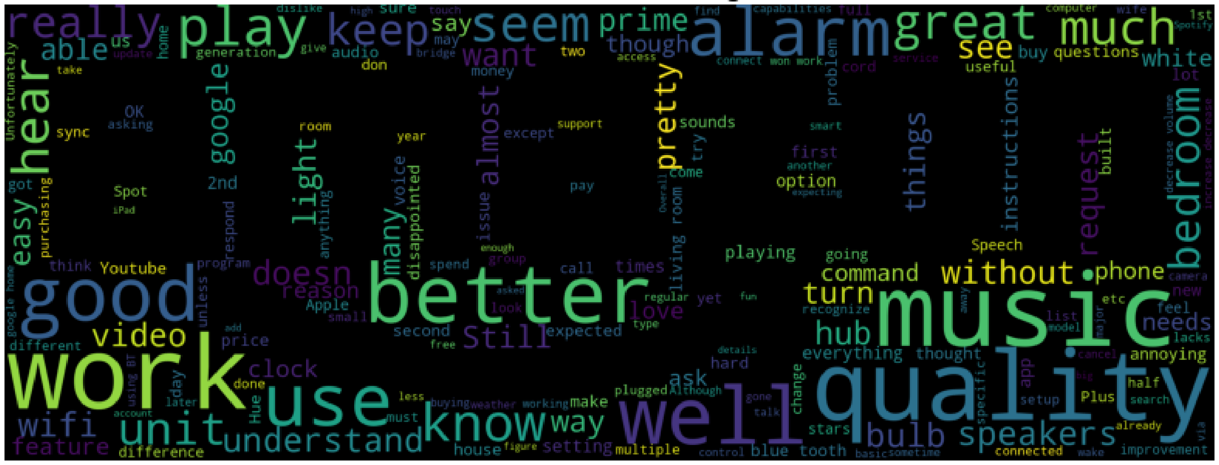
Reviews where rating = 1



Reviews where rating = 2



Reviews where rating = 3



Some of the most frequent common words across ratings 1 through 3 are ‘work’, ‘quality’, and ‘music’.

Conclusion and Recommendations

The takeaway from this exploration is that predicting what determines negative feedback is more than just looking at the rating. Looking at the text used helps get more insight into why a customer chose a certain rating. While the results from the sentiment analysis using BERT was not as helpful as I had thought, there were more improvements I could make in my analysis by focusing more on the negative feedback. The performance of the model seemed less inflated and more realistic.

If I wanted to see if the model could perform better, I could also consider looking at logistic regression in my analysis.. When the dataset is not very large, the scope of review feedback can be limiting. So if I were to look at another classification problem, I would probably consider one larger than around 3000.