Amazon rating predication based on the review text

CSE 190: Data Mining and Predictive Analytics

Bijan Vossoughian
UCSD
A12441880
bvossoug@ucsd.edu

Nikan Aminian
UCSD
A10039139
naminian@ucsd.edu

Abstract

In this paper, we formulate a model that attempts to predict user's rating (number of stars) of a specific book based on the review they submitted for that book. We used the Amazon book reviews dataset [1]. As mentioned, the main feature in this study is the review text. However, as sub-features inside the texts, we are separately using unigrams, bigrams, and a combination of both in order to calculate the frequency of words. The main reason of using both unigram and bigrams and their combination is to achieve a better accuracy with our predictions. When it comes to text mining a dataset such as review texts written by a random user, the sparsity of words and their randomness is a main problem. Which is why we did not only use the unigram model. Using bigrams and their combination results in a better build in our language model and word sequences, resulting in better predictions.

Dataset

The data set we used for this paper is the same JSON file provided for assignment1 (train.json), where we predicted the helpfulness of the reviews. The dataset includes 1,000,000 reviews of various books, with details on different features of each review. The main features used in the data set are as follow:

Rating:	Reviewer's actual rating of item
reviewText:	Review that user submitted

The actual rating that the user gives an item is an integer between 1 to 5, where 1 is the lowest possible rating and 5 is the highest.

Predictive Task

Our project's predictive task is to predict users' rating of a particular book solely based on their review text of that item. To predict the user's rating, we used their

review text as the feature within a linear regression model:

$$X\theta = y$$

Where X is the feature matrix, y is a vector of outputs, and θ is the weight of each feature with an offset of 1 at the end. The size of the training set is 100,000 and the testing set is 20,000.

The reason we used linear regression as our model to predict user's ratings is because the predicted score is a real rating number which is calculated by the weighted sum of the features. So since the ratings are full integers the calculated prediction can easily be casted to the nearest integer. The measurement figure that was used to evaluate our model is the Mean Absolute Error:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - X_i \cdot \theta|$$

Where y_i is the true value and $X_i \cdot \theta$ is the prediction. The reason we chose MAE is that it seems to be the most fit and accurate metric for our model, especially since we are calculating the errors of one specific variable (ratings). Also we wanted to avoid giving more weight to data points that are further away from the mean, since in case of MSE each point gets penalized more as it gets further from the mean.

Model

We decided to use bigrams and a combination of both bigrams and unigrams as our model because it made sense that some combinations may have more weight on the rating than just single words, such as: 'loved this', 'highly recommended', 'great read', and 'wait for'.

In order to optimize the results, we used different numbers of words as features and different sizes of training sets. At first our model consisted of trying the top 2000, 4000, and 6000 words in each of the categories of unigrams, bigrams, and combined words; but the results were not satisfying because taking this approach led to over fitting of our prediction, where the training data were linearly fit with good errors but the test data's error would be extremely large. This outcome made it clear that we had to decrease the size of our features for each category thus we used features of size 500, 1000, and 1500 for each category to optimize our model. We also considered to use trigrams or maybe even higher n-grams but our current model seemed sufficient as the results were not drastically different. Given more processing power we can use larger training sets and testing sets for better models and possibly use more n-grams for better error rates. Though using more n-grams as features can improve the prediction but the cost of its computation compared to the improvement is too high.

For our **baseline** we started by simply using unigrams only as to determine our feature vector. But we limited our unigram library only to the top 500 most frequently used words in by reviewers. To obtain the feature vector for said category, a vector of

size 501 filled with 0's is created. Each word in individual reviews are checked against the words in the top 500 unigrams and an integer value in the feature vector is incremented for each occurrence of the word. Each feature vector is ended with a 1 for an offset term. Once the feature vector is created it is used along with the ratings given to each book by the reviewer to obtain a θ vector. This vector is used to predict an outcome using the test data's features. And thus the performance of our model is captured.

For our model, we presume that using combinations of words will improve upon the baseline but we also experimented by trying different sized word libraries (500, 1000, and 1500). We created word libraries of unigrams, bigrams, and combination of both in three different sizes based on the most frequently used words. The approach to creating the feature vector is similar to the baseline as each review text is iterated through word by word (non-punctuation words) and if they are within the proper library their count is incremented, and ended by a 1 for the offset term.

For our training data and test data we selected json objects from the mentioned file in a linear manner and selected 100,000 reviews for training and 20,000 for testing. We limited these numbers to help reduce computation time.

Literature

The 1 million reviews of amazon books are part of a larger dataset of amazon products and their reviews. This particular dataset was originally provided to research and calculate the helpfulness of each amazon review. It appears that various Amazon metadata are heavily used in different studies to predict different values and metrics. The main reason of this being is mostly because for being considered one of the biggest retailers in the world, Amazon's data is very accessible and well organized. So different studies can rely on their quality in attempt to calculate their own models. A lot of outside retailers who use amazon as a main outlet to make their products available to customers have been studying the helpfulness feature of amazon reviews. This feature on reviews which works with a simple up/down vote system has proven to be more effective than any advertisement that the seller could have used. However, at the end of the day reviews are just plain opinion someone has towards a specific product and could be either fairly balanced or very much biased based on their particular experience. This specific phenomena is what let us to instead predict their ratings based on the review text they left for the product.

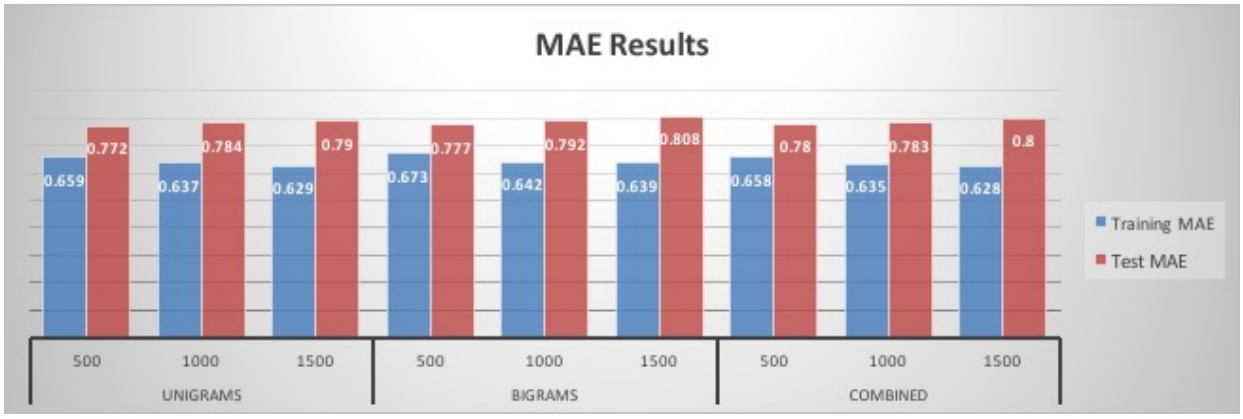
In a more similar study to this paper, Gerhard Weikum from Max-Planck institute of Informatics wrote the paper *The Bag-of-Opinions Method for Review Rating Prediction from Sparse Text Patterns* [2]. In which they used the "bag-of-opinions" method in order to predict review ratings based on review texts. Their approach is similar in terms of building their model based upon the finding unigrams texts for common choices. However, they realize the same major problem with only using unigrams which is their limitation to capturer important expressions mentioned in the review with longer lengths. As mentioned in the abstract

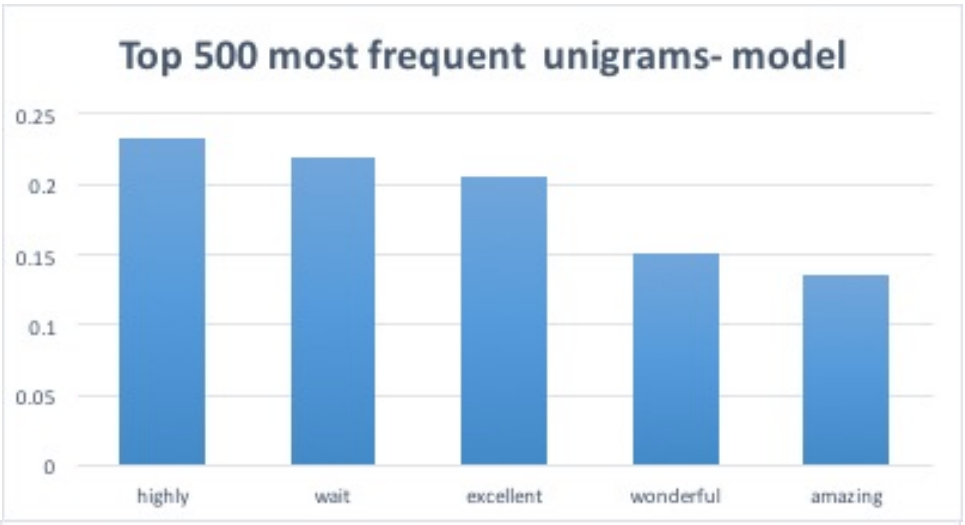
section of this paper, we tackled this problem by implementing bigrams and a combination of unigrams and bigrams, alongside only unigrams to build a better language model. Wiekum and his colleagues approached this problem with implementing n-grams alongside their unigram models in order to gather better review expressions. This is where they introduce the bag-of-opinions method where an opinion within a review consists of three parts: “a root word, modifier words from the same sentence, and one or more negation words”. As a result of their work they believed that they overcame the word sparsity bottleneck of item reviews.

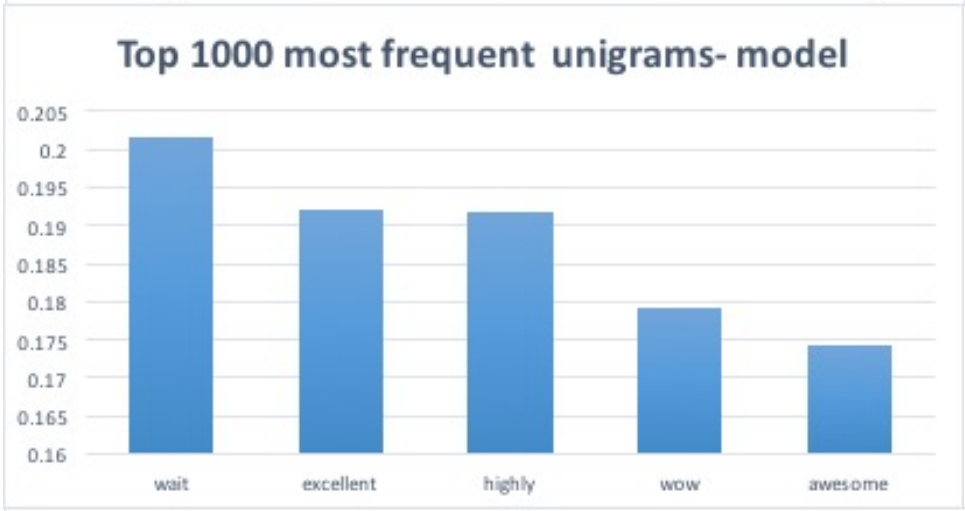
Amazon has become the leading B2C (business to consumer) commerce portal and has redefined products and service criticism by implementing major features such as helpfulness and star ratings. Therefore, this feature is being adopted by many other online websites in order to optimize customers’ experience on their site and save them more time by showing them the reviews they want to read, so they can spend more time shopping instead [3].

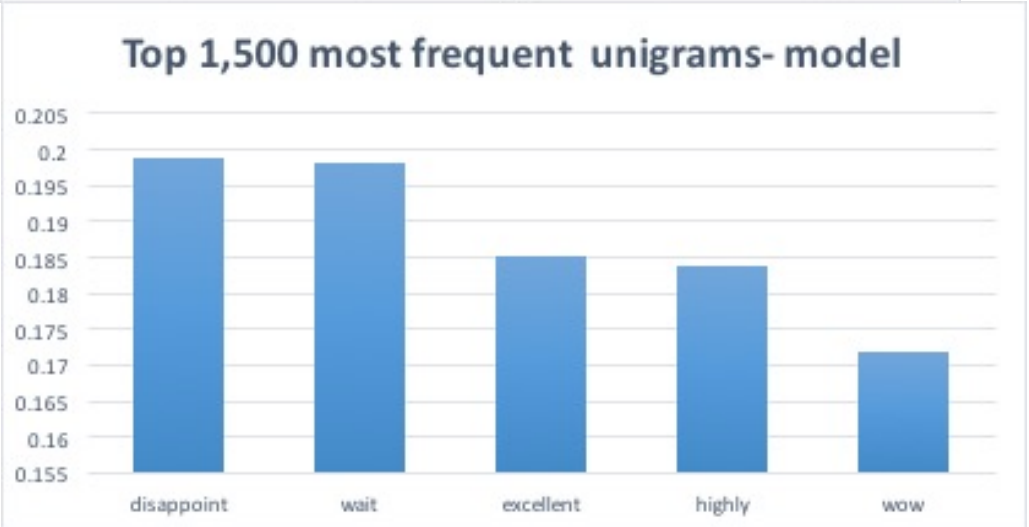
Results and Conclusions

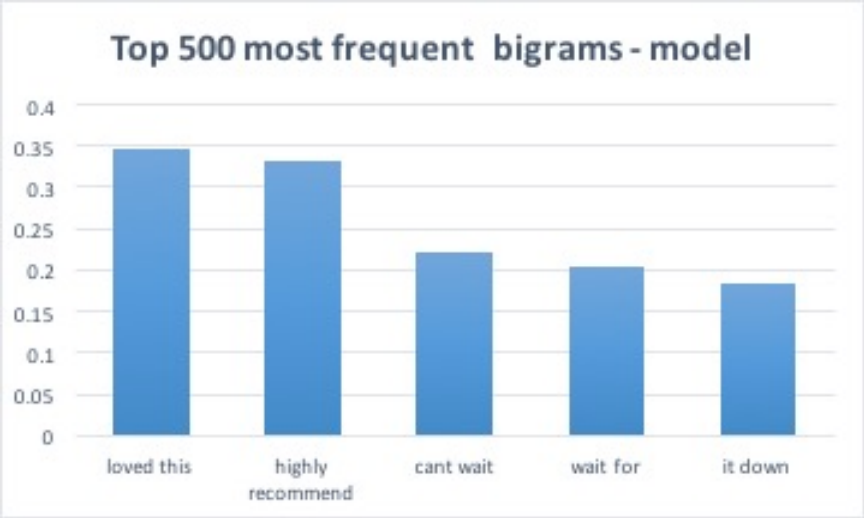
After testing different combinations of sizes of training set and different sizes of feature vectors, we settled with a training data set of 100,000, test set of 20,000 and feature vectors look through top 500, 1000, and 1500 most frequently words for each category of unigrams, bigrams and combination of both. Based on the results obtained from our many predictors (some of the results are provided as charts at the end of report) we came to the conclusion that using a combination of both unigrams and bigrams yields marginally better error rates for training and test sets. We expected the results be significantly better with the higher n-grams but it appears that the decrease in error rates are very minimal, this can be due to the approach that was used to collect the data set or simply their sizes. Maybe a more sparse and random collection of data would yield more noticeable results. As part of the results we wanted to include the top weighted words in the categories that we calculated, to show which words indicated that the reviewers would give a higher rating.

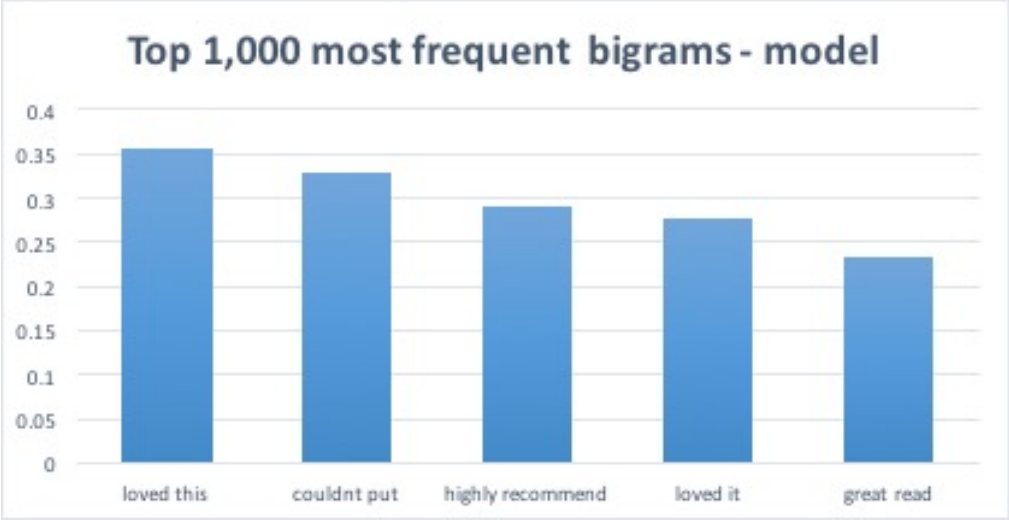

MAE Results
0.659
0.772
0.637
0.784
0.629
0.79
0.673
0.777
0.642
0.792
0.639
0.808
0.658
0.78
0.635
0.783
0.628
0.8
500
1000
1500
500
1000
1500
500
1000
1500
UNIGRAMS
BIGRAMS
COMBINED
Training MAE
Test MAE

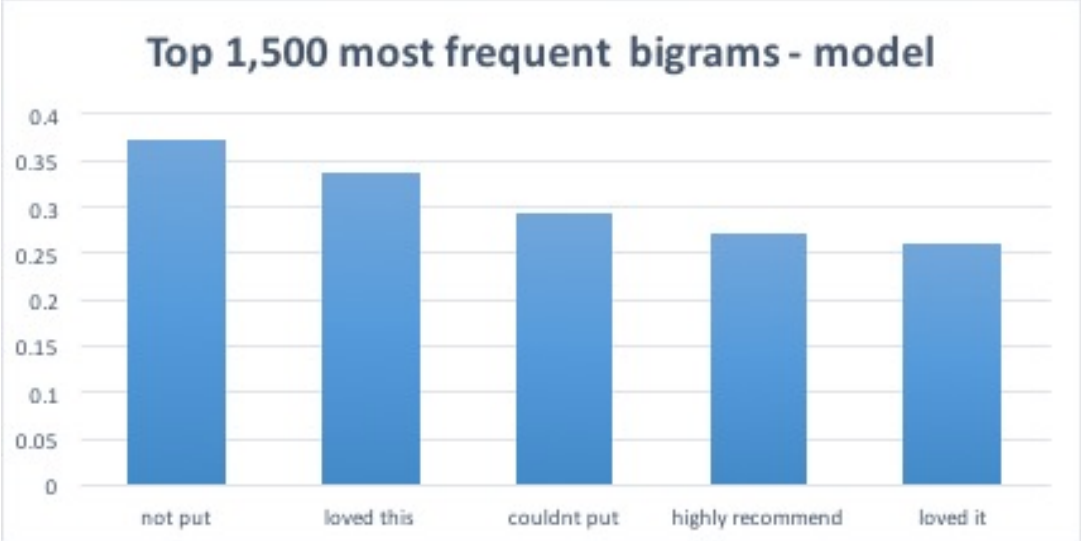

Top 500 most frequent unigrams- model
0.25
0.2
0.15
0.1
0.05
0
highly
wait
excellent
wonderful
amazing

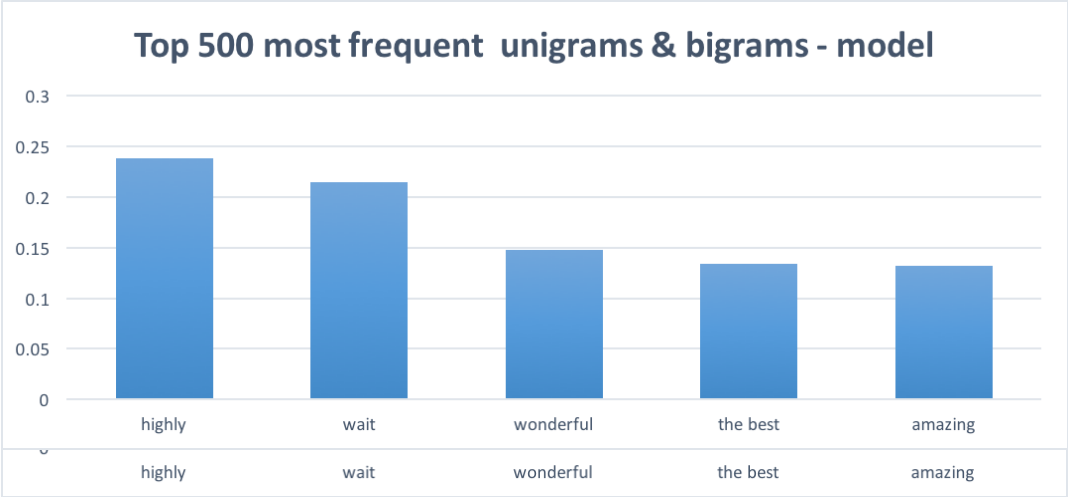

Top 1000 most frequent unigrams- model
0.205
0.2
0.195
0.19
0.185
0.18
0.175
0.17
0.165
0.16
wait
excellent
highly
wow
awesome

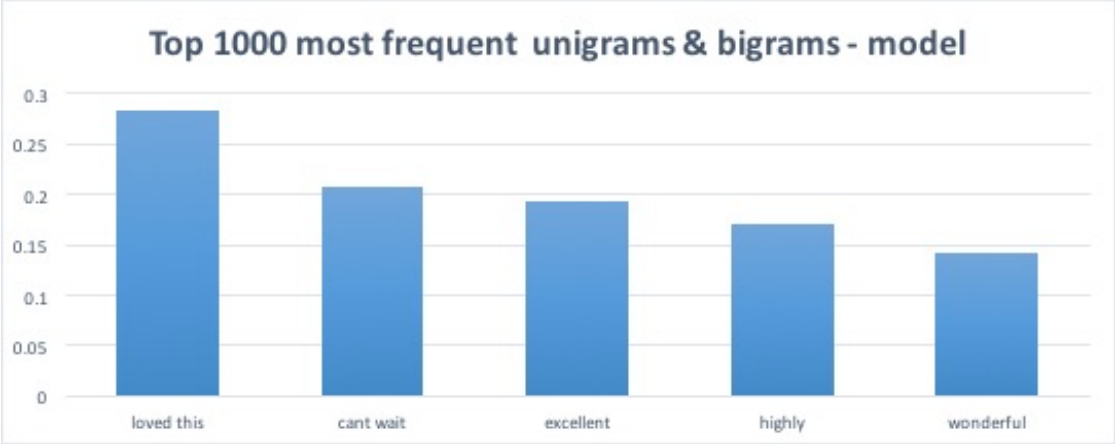

Top 1,500 most frequent unigrams- model
0.205
0.2
0.195
0.19
0.185
0.18
0.175
0.17
0.165
0.16
0.155
disappoint
wait
excellent
highly
wow

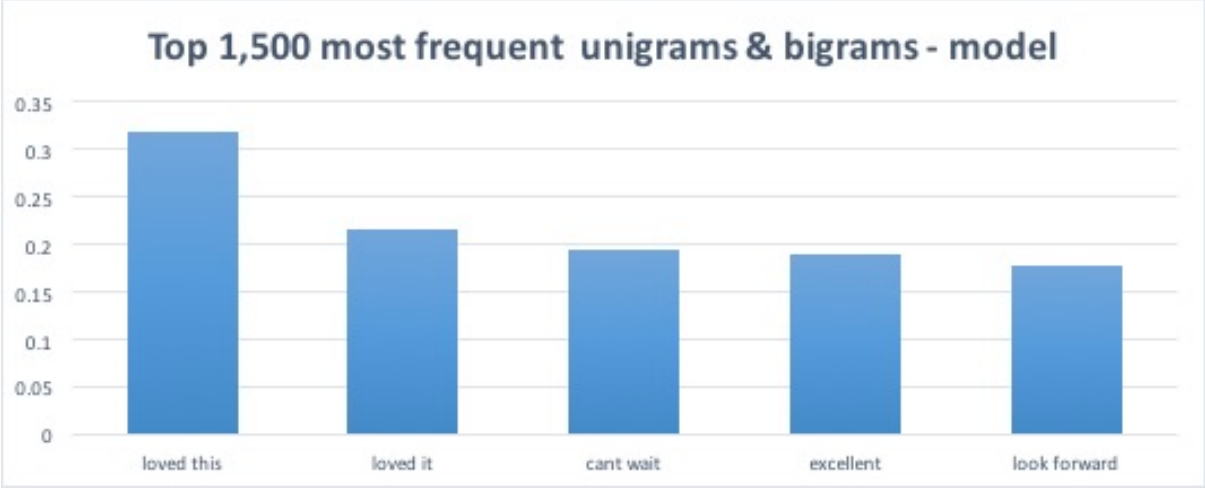

Top 500 most frequent bigrams - model
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
loved this
highly recommend
cant wait
wait for
it down


Top 1,000 most frequent bigrams - model
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
loved this
couldnt put
highly recommend
loved it
great read


Top 1,500 most frequent bigrams - model
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
not put
loved this
couldnt put
highly recommend
loved it


Top 500 most frequent unigrams & bigrams - model
0.3
0.25
0.2
0.15
0.1
0.05
0
highly
wait
wonderful
the best
amazing
highly
wait
wonderful
the best
amazing


Top 1000 most frequent unigrams & bigrams - model
0.3
0.25
0.2
0.15
0.1
0.05
0
loved this
cant wait
excellent
highly
wonderful


Top 1,500 most frequent unigrams & bigrams - model
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
loved this
loved it
cant wait
excellent
look forward

References

1. "Web Data: Amazon Reviews." *SNAP:*. Stanford University, n.d. Web. 02 Dec. 2015.
2. Qu, Lizhen, Georgiana Ifrim, and Gerhard Weikum. "The Bag-of-Opinions Method for Review Rating Prediction from Sparse Text Patterns." *ACL*. Association for Computational Linguistics, n.d. Web.
3. Wan, Yun, and Makoto Nakayama. "Are Amazon.com Online Review Helpfulness Ratings Biased or Not?" Springer. N.p., 2012. Web. 02 Dec. 2015.