Beer Rating Prediction Based on Text

December 7, 2020

Abstract

This paper's goal is to predict beer's rating given a review text, using three different methods: Bernoulli Naive Bayes model on word count, Linear Regression and k-NN on term frequency–inverse document frequency. The result of each model will be compared and discussed by their abilities to predict whether a review receives high or low overall rating.

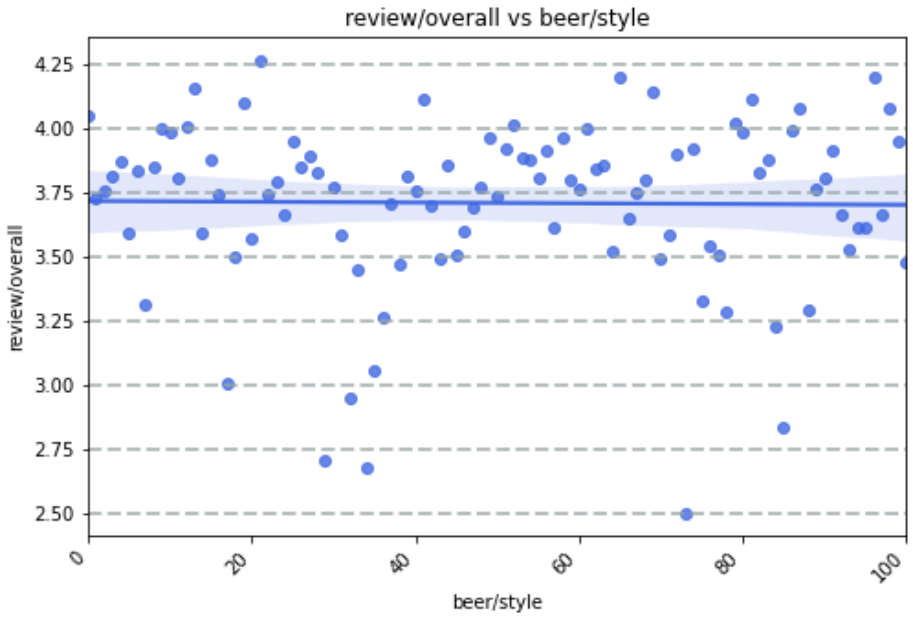
1. Dataset Description

The data set introduced in this project is the BeerAdvocate data set [2]. There are a total of 100000 reviews in the final data set, and each review has the following attributes:

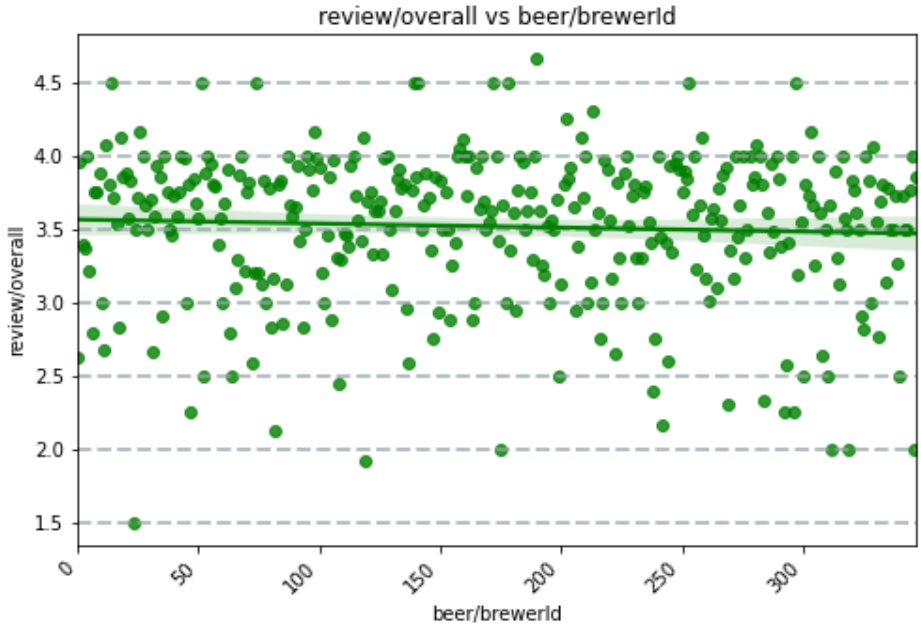
- beer/name: Name of the beer
- beer/beerId: Beer identification
- beer/brewerId: Brewer identification
- beer/style: Style of the beer
- beer/ABV: Alcohol By Volume of beer
- review/appearance [1-5]: Rating based on the look of beer
- review/aroma [1-5]: Rating based on the smell of beer
- review/palate [1-5]: Rating based on the interaction with beer
- review/taste [1-5]: Rating based on the taste of beer
- review/overall [1-5]: Rating based on the cumulative experience.
- review/time: The time when the review was submitted
- review/profileName: Reviewer's profile name
- review/text: Reviewer's opinion on the beer in text

For the purpose of this paper, only the 'review/text' attribute is used to predict the rating of the beer – the 'review/overall' attribute. However, it is informative to analyze if any other attributes will affect the overall rating, such as some other ratings: 'review/appearance', 'review/aroma', 'review/palate', 'review/taste' and the length of 'review/text', or some other beer-categorized attributes like 'beer/brewerId', 'beer/style', 'beer/ABV'.

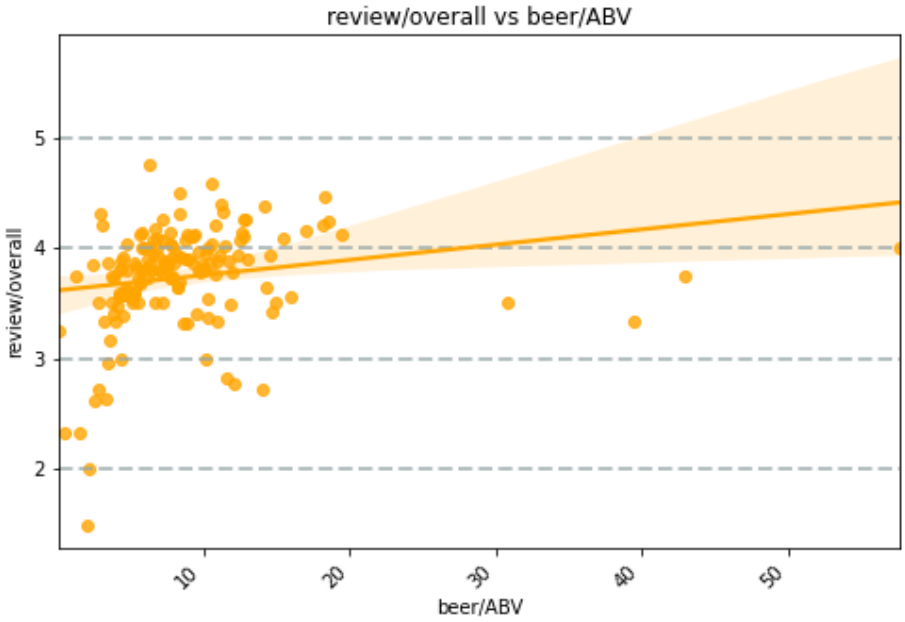
For attribute such as style, brewerId, each integer will represent a category.



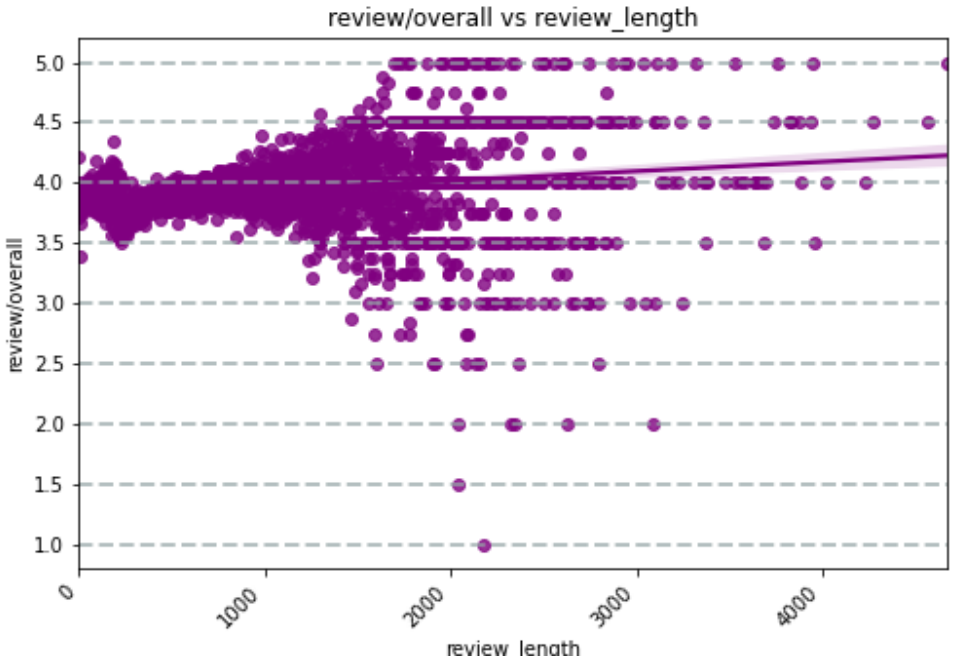
There are clearly differences in rating regarding to beer style but many ratings stick around 3.75 and most of them are > 3.5



There are less variations in ratings of different brewer, compared to the style category. However, it is not a strong indicator since there are brewers with less reviews.

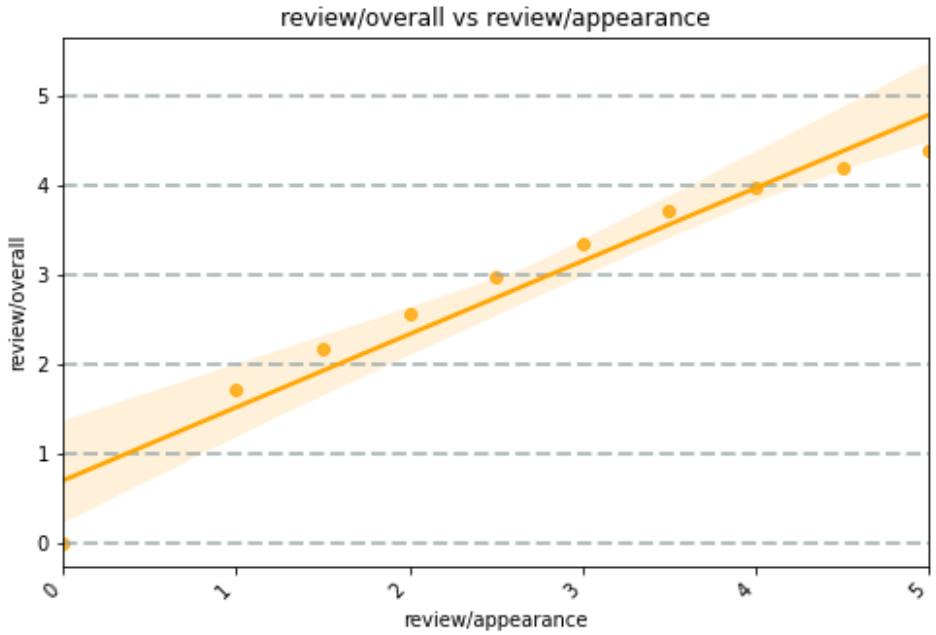


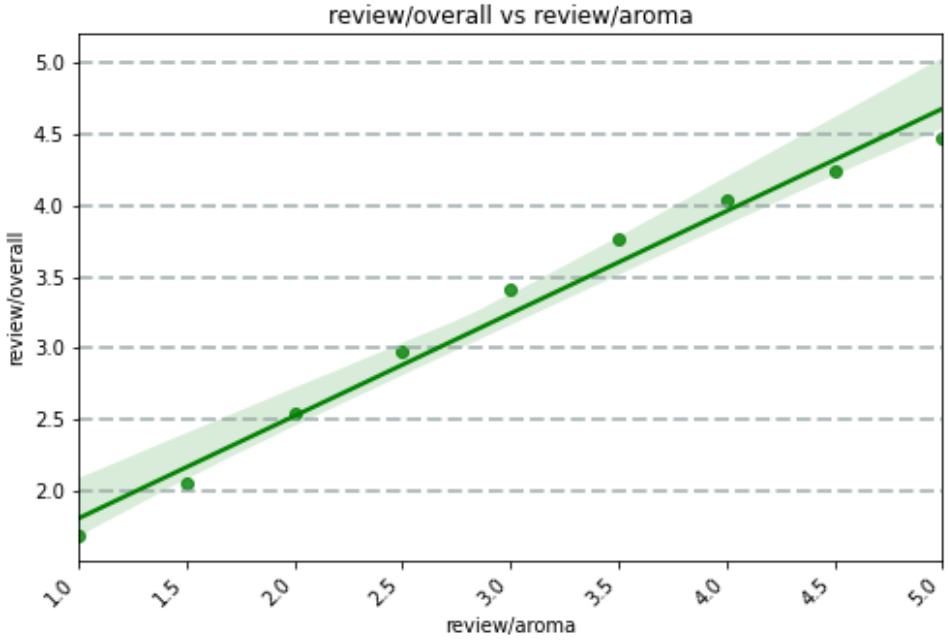
The graph indicates that the majority of reviewers like beer with ABV < 20, where most of them will rate >= 3.5 if ABV is in between 0 and 20.



The bad ratings are from the lengthy reviews, which are mostly > 1700 characters. Moreover, all the 5.0 ratings are from the review with length > 1700 characters.

In the following four graphs, we could see that the overall rating is positively correlate with the appearance, aroma, palate, taste ratings, which is reasonable since the overall rating considers these elements as judging criteria.





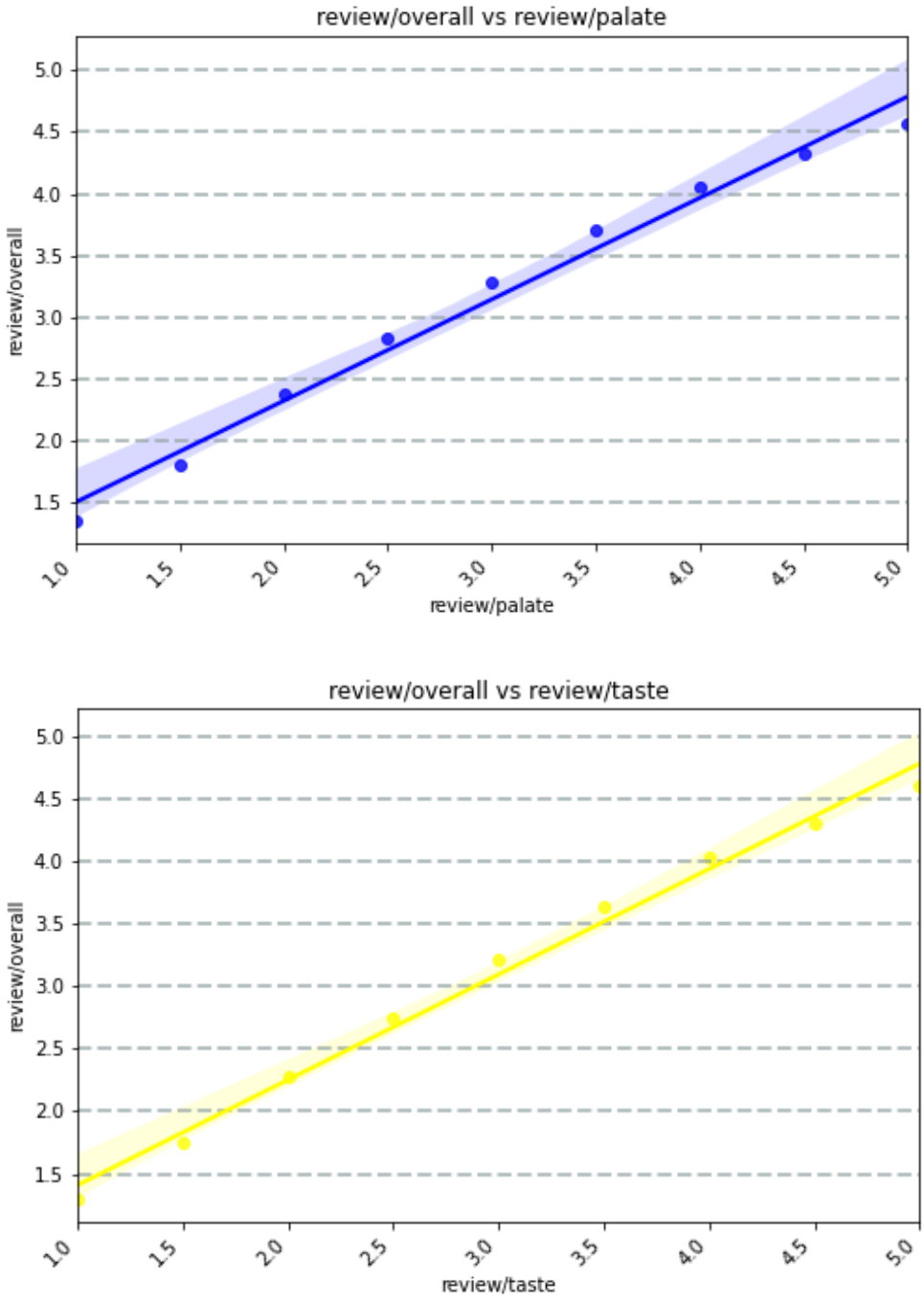


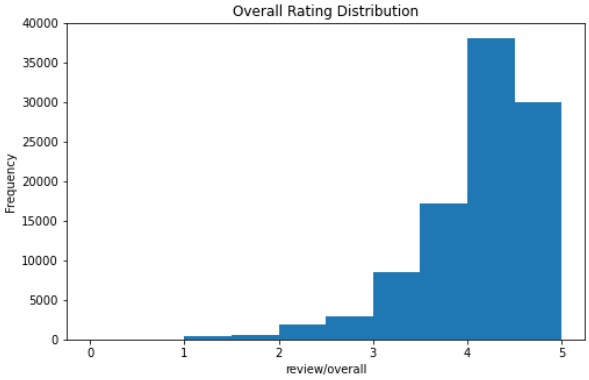
Figure 1: Distribution of Overall Rating

Table 1: Correlation between numerical features

	review/overall	review/appearance	review/palate	review/taste	review/aroma	beer/ABV	review_length
review/overall	1.000000	0.489678	0.686402	0.775298	0.596313	0.131734	0.051897
review/appearance	0.489678	1.000000	0.549310	0.526898	0.528390	0.240494	0.081490
review/palate	0.686402	0.549310	1.000000	0.716624	0.592861	0.287835	0.077584
review/taste	0.775298	0.526898	0.716624	1.000000	0.699170	0.296684	0.070301
review/aroma	0.596313	0.528390	0.592861	0.699170	1.000000	0.348777	0.090255
beer/ABV	0.131734	0.240494	0.287835	0.296684	0.348777	1.000000	0.120151
review_length	0.051897	0.081490	0.077584	0.070301	0.090255	0.120151	1.000000

From *Table 1*, we observe that the overall rating correlates the most with taste and correlates the least with review length, which are also visualized in the graphs above. The review length attribute is not a strong indicator since longer text could hold more positive or negative meaning. In fact, the rating is more extreme when the reviewer is more invested, which explains the lengthy review. Taste has the highest correlation since it is easier to judge and more appealing to the average reviewer, compared to the other categories. Moreover, since each person has their own preference on the alcohol percentage, the ABV score would vary largely from personal experience, which can be seen in the low ratings from across all criteria.

Table 2: Rating for different words in review



	Mean Rating	Median Rating
awful	2.634	2.5
bad	3.529	3.5
ok	3.861	4.0
beer	3.903	4.0
fine	4.006	4.0
amazing	4.301	4.5

Figure 1: Rating Distribution

From *Figure 1,* we could see that the reviews from this dataset are overwhelmingly positive as the distribution is right skewed with the peak near 4.0. The mean and the median are found to be 3.897 and 4.0, respectively. Under this distribution, one could expect the predicted rating to be high. However, using different keywords, we were able to find reviews with a variation of scores. To be precise, reviews that have the words

associate with negative meaning such as “awful” and “bad” yield a lower rating than the reviews containing “fine” and “amazing”. These values are presented in *Table 2.*

Under this impression, we were motivated to find how meaningful the words behind text is in review, such that one could measure the reviewer emotion toward the beer – that is the overall rating.

2. Predictive Tasks

For this paper, our task is to predict how high a rating is compared to other reviews, given the user’s opinion by text. Although it is tempting to use other attributes maybe with higher correlation to predict the rating, we are more interested in how the composition and selection of words, which is more likely to be included in reviews, could yield a result in categorization.

Given the *Figure 1*, we could see the distribution of the ratings is more right skewed. This is important, as we need to be careful in selecting a threshold θ to separate reviews with high rating and reviews with low rating. In this data set, the mean and median are 3.897 and 4.0, respectively. We could see that the mean and median are very similar, and 4.0 being the most popular rating.

In addition, the ratings are in the form of [0, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5]. Taking those into consideration, the ratings will not be evenly separated either mean or median are chosen, meaning there will not be 50% of reviews labeled as “high rating” and 50% of reviews labeled as “low rating”. Therefore, even if the mean (3.897) is not statistically interesting, it is worthy of being the threshold for high rating, on the scale of 5. As a result, there will be more reviews considered as “high rating” (larger than 3.897) than “low rating”.

3. Literature

Beeradvocate [2] is the data set chosen for our analysis. This was selected among some other similar data sets such as Ratebeer [6] and CellarTracker [4], as it is enough for our bag-of-word selection and has a decent size for testing.

As the need for online services grows, our need for information-extracting will increase over time. There are already many resourceful but unstructured data available. Most of them are emails, responses, web pages, reviews, and more in the form of text. To aid with the ability to understand these data, Natural Language Processing (NLP) is introduced. NLP is a specific field in artificial intelligent (AI) and machine learning (ML) that collaboratively using computer science and linguistic skills to filter and analyze language [3].

Perhaps text classification is one of the most important and typical tasks in NLP. For most of the time, there are three approaches to text classification: rule-based system, machine learning-based system, and hybrid system [5].

Rule-based system will follow the rule made by the user that works with relevant elements in text and categorize them [5]. One example of using rule-based system is to differentiate between spam and non-spam in emails. User will need to define two lists of words that identify spam and non-spam. Email having more words of one list will then be separated into that one category. Machine learning-based system makes classification based on past behaviors using a trained data set [5]. Finally, the hybrid system, as the name suggests, combines the previous two systems to fine-tune the base classifier [5].

Some of the more popular machine learning algorithms consist of Naive Bayes, Support Vector Machines (SVM) and deep learning [5]. Given our relatively small data set size, we will be using the Naive Bayes algorithm. The Bayes theorem will be applied to each review to compute the conditional probabilities of the words' occurrence in two scenarios (high or low rating).

For a successful classification, one should follow specific steps to prepare for the framework [1]:

1. Dataset preparation: this is where we will decide the train and test sets.
2. Feature Engineering: the text data will be transformed into feature vectors using word count, tf-idf, or word embedding, etc.
3. Model building: this is the step to train our classifier using a machine learning model.

4. Features

For our best interest, the most important feature in the data set is the 'review/overall', which will be decided by the 'review/text' feature. In the pre-processing step, we will need to extract the information from 'review/text' across all reviews to use on the model. Each unique word in the dictionary will correspond to a feature.

There are four vocabularies built: U_1 is a unigram dictionary which holds all the words from the low rating reviews, U_2 is a unigram dictionary from the words of high rating reviews, B_1 is a bigram dictionary from the words of low rating reviews, B_2 is a bigram dictionary from the words of high ratings.

A unigram consists of single words whereas a bigram consists of two consecutive words in the sentence. All texts are lowercase. There will not be punctuations, and stop words such as "in", "the", "she" for the fact that they are not constructive toward the beer rating. For example, the sentence "The beer has good taste!" will be represented as {"beer", "good", "taste"} in the unigram dictionary and {"beer good", "good taste"} in the bigram dictionary.

5. Models

In this project, a Bernoulli Naïve bag-of-words model [7] was used to predict the beer rating. The text will be chosen from the review and treated as a list of strings. These

strings will then be evaluated based on U_1 U_2 or B_1, B_2 depends on which type of word selection (unigram, bigram) we interested in testing. It is assumed that the words in the review are independent from each other. Given a review *r*, we want to predict if its rating is high or low by the following rule for *unigram*: if $P(U_2 \mid r) > P(U_1 \mid r)$, then we predict that the rating is high (recall, U_1 and U_2 are defined based on threshold θ = 3.897 with U_2 be a unigram dictionary from reviews rated > θ and the opposite for U_1). For *bigram*, if $P(B_2|r) > P(B_1|r)$, then we predict high. The formula for unigram is calculated as such

$$P(U_1 \mid r) = \frac{P(r \mid U_1) \cdot P(U_1)}{P(r)} \quad (Bayes'theorem)$$
$$= \frac{\prod_{i=1}^{n} P(w_i \mid U_1) \cdot P(U_1)}{P(r)} \quad (w_i \; is \; the \; i^{th} \; word \; in \; r)$$

$P(U_2 \mid r)$ will also be calculated in a similar fashion. As Bernoulli Naive Bayes theorem is being used, if a word *w* appears more than once in a review, its probability will only be accounted for once.

The other models introduced in this paper are Linear Regression and k-NN. For these models, we will calculate the term-frequency–inverse-document-frequency (*tfidf*) instead of word count. The formulas are

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D)$$

$$tf(t, d) = \#\; term\; t\; appear\; in\; review\; d$$

$$idf(t, D) = \log_{10} \frac{D}{df(t)} \quad (D\; is\; number\; of\; reviews)$$

$$df(t) = \#\; of\; document\; contains\; term\; t$$

We will be using a unigram dictionary (T) to store the *tfidf* of the word. The top 1000 words with high *tfidf* are selected and used to create a sparse matrix, such that each cell holds the *tfidf* for one of the 1000 words and each row is the review in question. The Linear Regression and k-NN model will be fit using this matrix. We are testing the Linear Regression model with different inverse of regularization strength (C), such as 0.01, 0.1, 1, 10, 100. The k-NN model will be tested using uniform weight, and Minkowski distance with different number of nearest neighbor (N), such as 1, 2, and 3.

6. Result and Conclusion

After shuffling, the data set is separated into 70% reviews for training and 30% reviews for testing. The Linear Regression models perform best on the test set with a slight variation in the *accuracy rate* with different *C*, where the *accuracy rate* is calculated

using the formula $\frac{number\ of\ correct\ predictions}{total\ number\ of\ predictions}$. The results of each model are listed in *Table 3*.

Table 3: Accuracy Rate on Bayes Model

	Train	Test
unigram	0.4223	0.599
bigram	0.3485	0.368

Table 4: Accuracy Rate on Linear Regression

C	Train	Test
0.01	0.68201	0.6717
0.1	0.68228	0.6721
1	0.68222	0.6723
10	0.68221	0.6723
100	0.68221	0.6722

Table 5: Accuracy Rate on k-NN Model

N	Train	Test
1	0.9282	0.5919
2	0.7467	0.5236
3	0.7914	0.6388

We could see that the unigram dictionary performs better than the bigram dictionary in the Naive Bayes model, which yield 0.5994 and 0.3681 in the testing set, 0.4224 and 0.3486 in the training set. While expecting the bigram dictionary to have similar or slightly better performance, the outcomes disputed our theory. There could be many speculations behind this result. Perhaps one of them is that unigram is more constructive than bigram as stop words are cut. With stop word being the connection, the combination of the word before and after the stop word could be less meaningful.

Consequently, the Linear Regression and k-NN models are only tested on the unigram dictionary. As tested, the Linear Regression model performs best with C = 0.1 and C = 1 in the training set and C = 1 and C = 10 in the testing set. The k-NN model performs best with N = 1 in the training set and N = 3 in the testing set. Furthermore, the k-NN model seems to be overfitting the trained data since there are some variations between the train and test set.

As the original data set is unbalanced, where most of the reviews have higher rating than the threshold, we need to carefully select a meaningful train and test set for evaluation. Although the accuracy rate is not high in our model, we ensure that the classifier can be used in different data sets, by checking the prediction true negative rate and true positive rate, which are 0.6 and 0.59 respectively. The formulas are as followed

$$True\ positive\ rate = \frac{\#\ correct\ high\ rating\ prediction}{\#\ high\ rating\ prediction}$$

$$True\ negative\ rate = \frac{\#\ correct\ low\ rating\ prediction}{\#\ low\ rating\ prediction}$$

This points out that the classifier does not predict high rating most of the time as our data is right skewed. Moreover, since the accuracy rate of test and train sets are not varied much, the classifier will not be overfitting.

Granted, an accuracy rate of 0.5994 is relatively low compared to other models, the classifier will be more useful once number data is not available, which could happen most of the time. This result does not dispute that text classification is less meaningful. In fact, we believe that there are room for improvement by selecting a different threshold θ, switching to *tfidf* or using a different approach on bag-of-words selection.

Although we might not fully interpret the effective solution to combine linguistics and text classification, we hope that in the near future when more time is allowed, a better classifier could be built.

References

[1] BansalShivam, Shivam. "A Comprehensive Guide to Understand and Implement Text Classification in Python." *Analytics Vidhya*, 26 July 2019, www.analyticsvidhya.com/blog/2018/04/a-comprehensive-guide-to-understand-and-implement-text-classification-in-python/.

[2] BeerAdvocate Dataset: http://snap.stanford.edu/data/Beeradvocate.txt.gz

[3] Brownlee, Jason. "What Is Natural Language Processing?" *Machine Learning Mastery*, 7 Aug. 2019, machinelearningmastery.com/natural-language-processing/.

[4] Cellartracker Dataset: http://snap.stanford.edu/data/cellartracker.txt.gz.

[5] "Guide to Text Classification with Machine Learning." *MonkeyLearn*, monkeylearn.com/text-classification/.

[6] Ratebeer Dataset: http://snap.stanford.edu/data/Ratebeer.txt.gz

[7] Zhang, Angel Iek Hou. *Judging YouTube by Its Covers*. 2015.