# Amazon Fine Food Reviews Analysis

CSYE 7245 Project Report

Taotao Jing
Student
Northeastern University
Massachusetts, U.S.

Prof. Nik Bear Brown
Academic Advisor
Northeastern University
Massachusetts, U.S.

***Abstract*—The purpose of this project is to provide business insights of "Review" based data set. We will adopt multiple learning methods to do binary classification (Positive or Negative). "Amazon Fine Food" dataset has a common format like other product reviews in different companies and different kinds of products. In other words, if we can find a good prediction model that fit well with this dataset, it can also be applied to other similar datasets that are popular in sales industry.**

**In the project, we use popular machine learning based classifier (Naïve Bayes / Regression /SVM) and neural network based method (LSTM). Through exploration and preprocess our dataset, transform our dataset into a proper input matrix for those models and tuned the parameters by using cross validation. Finally, we will give out the best model based on our analysis and explain the reason.**

***Keywords—AI; Machine Learning; Naïve Bayes; Regression; SVM; Neural Network; Reviews***

## I. INTRODUCTION

This project will use the text information as feature and score as target (see data exploration for more information about the dataset). Classify the reviews by mining the text to see if we can infer the score through the customer reviews and measure whether the score customer gives to the product is reasonable based on their reviews. Reviews will be divided it into two partitions according to the "Score" and retrieve "Text" feature as training data. Several traditional machine learning methods such as: naive bayes, logistic regression, SVM will be applied. The results of those models will be compared with baseline model which is dummy classifier. By comparison, the team analysis the performance of each method.

Although the traditional machine learning methods train relatively quick, the overall accuracy of those models on balanced dataset are around 0.8. It looks good, but not good enough. To improve the accuracy, the team use neural network. Recurrent Neural Network (RNN) is used in this paper. RNN is a specialized kind of neural network for processing data that has a known grid-like topology (Deeplearningbook.org, 2017). By using RNN, the accuracy on train set could be improved significantly. However, overfitting problem cannot be ignored when use RNN. The team tuned the different parameters and use filters to avoid this problem.

This paper is organized as follows: Background gives the general area and specific problem we addressed. Data exploration and preprocessing introduces the basic distribution of our dataset and the basic preprocessing we did on our dataset. In Data Analysis, we discussed how do we vectorize our dataset and the method we use. In Result, we compared the result of different method and the different parameter of same method. Some graphs included in this part which visualizes the performance of model. Conclusion part gives the conclusion of this paper.

## II. BACKGROUND

Nowadays, most of the company will formulate their plan, e.g. type of product, quantity of item, price of item... etc., based on the feedback of their customers. Among those feedbacks, the most directly and valuable data that need to be analyzed is 'reviews'. Customer reviews are more and more important in spreading information and contributing e-commerce (Ngo-Ye and Sinha, 2012). Generally, it will greatly improve the cost and user experience by optimizing the plan to meet the customer's' preference. Several researchers have examined the role of expert reviews (Chen and Xie 2005), and the role of online recommendation systems (Bakos 1997; Chen et al. 2004; Gretzel and Fesenmaier 2006). More recently, research has examined the

role of online customer product reviews, specifically looking at the characteristics of the reviewers (Forman et al. 2008, Smith et al. 2005). Also researches shows that customer reviews can have a positive influence on sales (Chen et al. 2008; Chevalier and Mayzlin 2006; Clemons et al. 2006)

Analyzing reviews nowadays are more focusing on products itself or just filtering 'nonqualified' (defined by naive filtering models or just if-clause) reviews out. A technique to motivate users to give better quality reviews, for example, giving logical scores on their reviews by AI process, is needed.

Letting the customer clear about what score of their reviews is can give them clear idea of how great their reviews are and may lead them to know how to make reviews more valuable to the company and the products. So that, if the company give the customer accordingly rewards systems for giving greater quantity and score reviews, the quality of rewards will benefit from the model a lot.

## III. Data Exploration and Preprocessing

### A. *Data Exploration*

The dataset for this project is: Amazon Fine Food Reviews dataset. This dataset was originally published in SNAP. The Link is: http://snap.stanford.edu/data/web-FineFoods.html. In this dataset, there are 568,454 reviews in total which comes from 256,059 users for 74,258 products.

This dataset consists of 10 features which are: ID, ProductId(unique identifier for the product), UserId (unique identifier for the user), Profile Name, Helpfulness Numerator (number of users who found the review helpful), Helpfulness Denominator (number of users who indicated whether they found the review helpful), Rating(name as Score in database, but it is rather rating between 1 and 5), Time(timestamp for the review),Summary(brief summary of the review) and Text.

This project mainly focuses on the Score and Text. The Score is in range of 1 to 5. The higher the score is, the more positive of the attitude which customer towards the product. The text column contains the review of the customer to the product. The distribution of Score is shown in Figure 1.

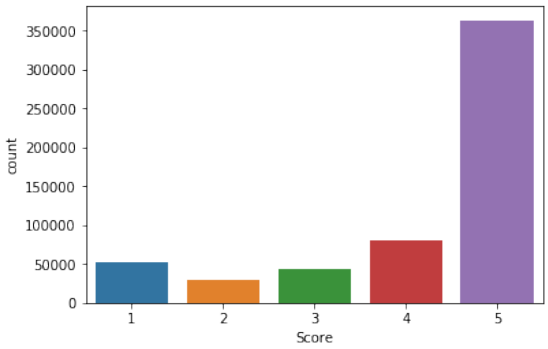


Fig. 1. Distribution of Score

We explore the relationship between the average length of Text and Score. Highest score has shorter average length of Text. The score 3 has longest average length of Text. Figure 2 is the figure which shows the distribution.

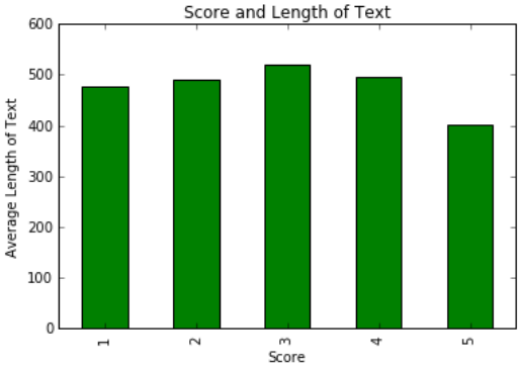


Fig. 2. Average Length of Text and Score

### B. *Data Preprocessing*

The dataset has no missing value. So, we do not need to care about the missing value issue. We assign the Score 1 and 2 into negative group and Score 3,4,5 into positive group. After assignment, the distribution of our dataset is Figure 3.

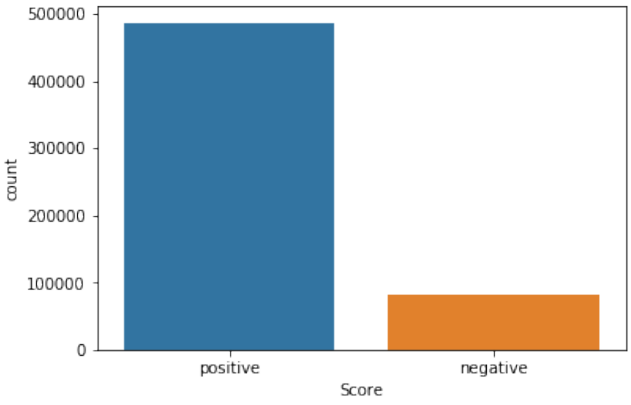


Fig. 3. Unbalanced Label Distribution

From Figure 3, it is not hard for us to find that the distribution of target label is quite imbalanced. There are two common method used to deal with this problem: oversampling and undersampling. For oversampling method, the observations from the minority class should be duplicated in order to obtain the balanced dataset. For undersampling method, the observations from the majority class should be dropped in order to get the balanced dataset. Here, our team uses undersampling method. We resample our data set to use the balanced dataset to train the model. After we resample, the distribution of target label is as Figure 4.

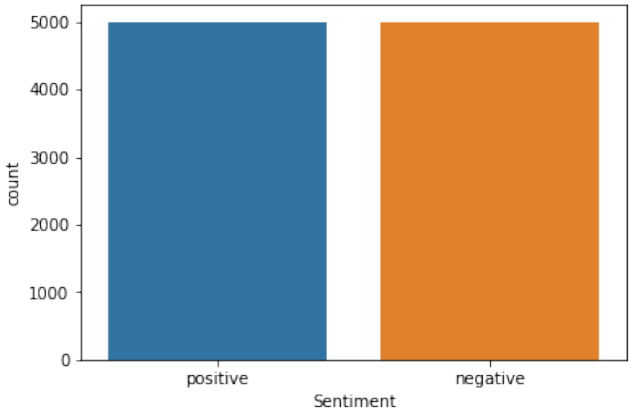


Fig. 4. Balanced Label Distribution

## IV. DATA ANALYSIS

### A. Word Vectorization

To analyze Text, machine learning and neural network are the major methods we used. However, the text is a sequence of symbols and the length of the text is variable. We can not use this kind of data to fit the machine learning and neural network model directly which need to use numerical data with fixed size.

In our project, we use three different methods (word count, TFIDF, TFIDF with Ngram) to vectorize the text and make a comparison among those methods to find a better one.

*1) Word Count: We usually use three steps to vectorize the text: tokenizing, counting, normalizing and weighting. In tokenizing step, the documents are tokenized and an integer ID is given for each token. In counting step, the occurrences of tokens in each document are counting. In last step, the tokens are normalizing and weighting. Word count contain the first two step of this process. By doing the word count, a set of documents can be represented by a matrix in which each document is a row and each token is a column. Each individual token occurrence frequency is a feature. The vector of all the token frequencies for a document is a sample.*

*2) TFIDF (Term Frequency–Inverse Document Frequency): In a large text corpus, some words have high frequency but very little meaningful information. If we use the count data directly to train a model, the very frequent tokens with little meaning will have a significant influence over the low frequency but far more interesting terms. So it is necessary to normalize and re-weight the word count result. One of the most popular method is TF-IDF which means term frequency times inverse document frequency. TF (Term Frequency), which measures how frequently a term occurs in a document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. Thus, the term frequency is often divided by the document length (aka. the total number of terms in the document) as a way of normalization: TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document). IDF (Inverse Document Frequency), which measures how important a term is. While computing TF, all terms are considered equally important. However it is known that certain terms, such as "is", "of", and "that", may appear a lot of times but have little importance. Thus we need to weigh down the frequent terms while scale up the rare ones, by computing the following: IDF(t) = log_e(Total number of documents / Number of documents with term t in it).*

*3) TFIDF With Ngram: The lower and upper boundary of the range of n-values for different n-grams to be extracted. All values of n such that min_n <= n <= max_n will be used. If ngram_range has larger region, it will have much more combinations from features.*

### B. Baseline Model

This project use dummy classifier as a simple baseline model to compare with other classifiers. We use dummy classifier on both balanced data and imbalanced data. The result is in following Table 1.

| Imbalanced data | Word Count | 0.75 |
|---|---|---|
| Balanced data | Word Count | 0.50 |

TABLE 1. BASELINE MODEL

## V. MACHINE LEARNING MODELS

### A. *Naïve Bayes (Multinomial / Bernoulli)*

Naive Bayes is a conditional probability model, given a problem instance to be classified, represented by a vector $X = (X_1, X_2, .... X_n)$ representing some n features (independent variables). It assigns to this instance probabilities $p (C_k | X_1, X_2, .... X_n)$ for each of K possible outcomes or classes Ck, using Bayes' theorem, $p (C_k | X) = (p(C_k)*p (X | C_k)) / p(X)$. Because p(X) doesn't depend on C and the values of Xi are given, we can treat p(X) as a constant. We only need to consider $p(C_k)*p (X | C_k) = p (C_k, X_1, X_2, .... X_n)$. With chain rule $p (C_k, X_1, X_2, .... X_n)$, it will be converted into $p(C_k)*p (X_1 | C_k) *p(X_2 | C_k)* .... = p(C_k)*\prod p (X_i | C_k)$.

In the next step, we can construct a classifier with the probabilistic model. The simplest decision rule is to pick up the most probable class. The corresponding classifier is this function:

$C_{max} = \text{argmax } k \in (1 \text{ to } K)\ p(C_k)*\prod p(X_i | C_k)$.

Multinomial Naive Bayes: With a multinomial model, samples represent the frequencies with which certain words have been generated by a multinomial (p1, p2, .... pn) where is the probability that $word_i$ occurs (or K such multinomial in the multiclass case). A sample $(X_1, X_2, .... X_n)$ is then a histogram, with $X_i$ counting the number of times wordi was observed in an instance.

However, this approach has a drawback when the (word, class) pair never occurred. It will make the product of probability to be zero and get rid of information generated by other features. For solving this issue, we will apply tf-idf weighted value to replace the count of words.

Bernoulli Naive Bayes: In the multivariate Bernoulli model, features are independent binary variables describing inputs. Like the multinomial model, this model is popular for document classification task, where binary term occurrence features are used rather than term frequencies. The binary terms express the occurrence or absence of the i'th term from the vocabulary.

After pre-processing of dataset, we fit the X_train_tfidf and y_train into Multinomial Naive Bayes classifier. Finally, we will use the prediction model to predict X_test_tfidf. Based on test result and X_train_tfidf, we give out the accuracy of prediction, confusion matrix, and learning curve.

### B. *Logistic Regression*

Logistic regression is a kind of regression method used to solve classification problem. To explain logistic regression, it is necessary to know sigmoid function which is defined as follows:

$$g(z) = \frac{1}{1+e^{-z}}$$

In sigmoid function, no matter what input value is, the output value should be in range 0 to 1 and can be interpretable as a probability. The cost function of is:

$$\frac{1}{m}\left[\sum_{i=1}^{m} -y\log(h_w(x)) - (1-y)\log(1-h_w(x))\right]$$

However, to avoid overfitting, we penalize the parameters. We get the cost function below:

$$J(\theta) = -1/m\sum_{i=1}^{m}\left[y^{(i)}\log h_\theta\left(x^{(i)}\right) + (1-y^{(i)})\log\left(1-h_\theta(x^{(i)})\right)\right] + lamda/2m\sum_{j=1}^{n}\theta_j^2$$

$$J(\theta) = -1/m\sum_{i=1}^{m}\left[y^{(i)}\log h_\theta\left(x^{(i)}\right) + (1-y^{(i)})\log\left(1-h_\theta(x^{(i)})\right)\right] + lamda/2m\sum_{j=1}^{n}\theta_j^2$$

where:

$h_w(x)=g(w^T x)$

By using gradient descent, the weight can get update iteratively:

$$\theta_0 = \theta_0 - \alpha/m\sum_{i=1}^{m}(h_0(x^{(i)}) - y^{(i)})x_0^{(i)}$$

$$\theta_j = \theta_j - \alpha/m\left[\sum_{i=1}^{m}(h_0(x^{(i)}) - y^{(i)})x_j^{(i)} + lamda * \theta_j\right]$$

### C. *SVM (Linear / RBF)*

In the class, we have learned that SVM is a supervised machine learning. Its goal is to find the maximal margin between different classes. As we know, SVM has multiple types of kernel, e.g. linear, rbf, poly, sigmoid. The most common used kernel is 'linear' and 'rbf'.

Linear:

$$\left[\frac{1}{n}\sum_{i=1}^{n}\max\left(0,1-y_i(\vec{w}\cdot\vec{x}_i-b)\right)\right]+\lambda\|\vec{w}\|^2$$

Gaussian radial basis function:

$$k(\overrightarrow{x_i},\overrightarrow{x_j})=\exp(-\gamma\|\overrightarrow{x_i}-\overrightarrow{x_j}\|^2),\text{ for }\gamma>0.$$

When the training data are linearly separable, we will use Linear SVM to train the dataset. It cost less time than the Gaussian SVM and we don't have to set too many parameters for tuning the performance. In our project, we will apply the SVM on imbalanced dataset because it will cost less time for training. (Original dataset is too large, and we cannot afford the training time). Simultaneously, the result will show the relationship between the number of features and the number of samples. When the number of features is large or equal to the number of samples, Linear SVM will theoretically perform well.

### D. *RNN (Recurrent Neural Network)*

In order to predict if a review is positive or negative we will use a recurrent neural network. Most of the methods for sentiment analysis look at each word individually, attribute positive points for positive words and negative points for negative words, and then total the points. This is the lexicon approach. However, the problem with this method is that it ignores the sequence of the words, which can lead to the loss of important information. The RNN approach can understand subtleties because it doesn't analyze text at face value. It creates abstract representations of what it learned.

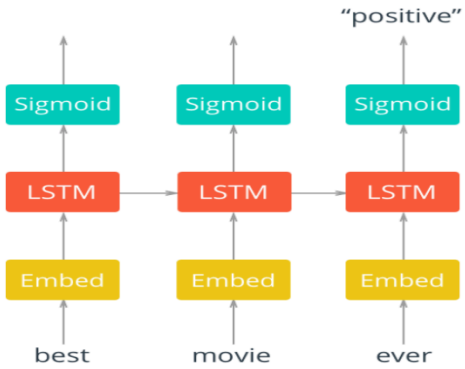


Fig. 5. Model architecture with two channels for an example sentence(WildML, 2017)

The neural network we build in this part looks roughly as the architecture shown in Fig.5. We will use Long Short Term Memory Networks (LSTM), which is a special case of RNN. The main advantage of LSTM is that it can memorize information. When there's too large of a gap between two pieces of information, RNN is unable to learn to connect information. To learn more about LSTM, you can read this excellent article written by Christopher Olah.

First we will use an embedding layer in order to represent words by a vectors. This representation is more efficient than one-hot encoding, especially in the actual context where the size of the vocabulary is more than 240,000 words.

The next step is to pass the embedded words to the LSTM cells. In the graph, you can observe that there are connections between the first, second and third LSTM cells. In contrast to other models that assume that all the inputs are independent of each other, these connections here allow us to pass information contained in the sequence of words. One of the strengths of the LSTM cell is the ability to add or remove information to the cell state.

Finally, we will predict if the review is positive or negative using the sigmoid function. The advantage of the function is that it's bound between 0 and 1 and can be interpreted as a probability of success. For example, we can estimate the probability that a review is positive. At each step, we have an output. However, we only care about the final output that predicts the sentiment at the end of the sentence.

## VI. Result and Conclusion Exploration

### A. *Model Result*

The dataset for this project is: Amazon Fine Food Reviews dataset. This dataset was originally published in SNAP. The Link is: http://snap.stanford.edu/data/web-FineFoods.html. In this dataset, there are 568,454 reviews in total which comes from 256,059 users for 74,258 products.

a) Unbalanced Dataset

1. Accuracy Comparison

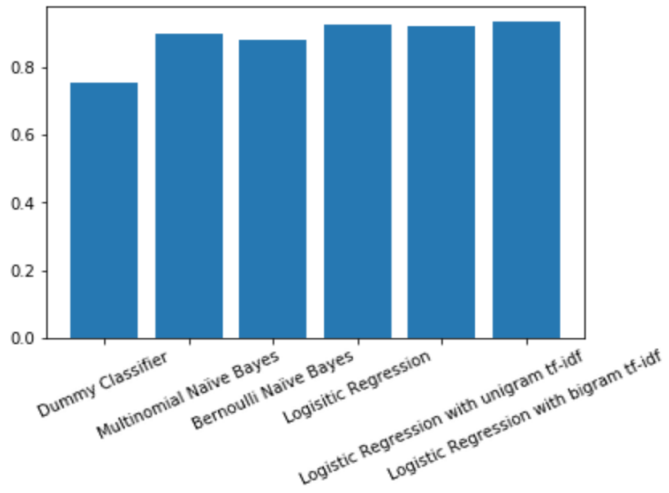


Fig. 6. Accuracy of prediction in different models with unbalanced dataset

Logistic Regression has the best accuracy and the difference between positive samples and negative samples is small. However, for Naïve Bayes, the performance on positive samples is worse than negative samples.

2. Classification Report

**CountVectorizer**

| Features: 119939, Train Records: 426340, Test Records: 142113, Negative Samples: 82037, Positive Samples: 486417 | | | | | |
|---|---|---|---|---|---|
| | Precision/ Positive | Precision/ Negative | Avg/ Total | Recall | F1-Score |
| Dummy Classifier | 0.15 | 0.86 | 0.75 | 0.75 | 0.75 |
| Logistic Classifier | 0.81 | 0.94 | 0.92 | 0.93 | 0.92 |
| Multinomial Naïve Bayes | 0.65 | 0.94 | 0.90 | 0.90 | 0.90 |
| Bernoulli Naïve Bayes | 0.60 | 0.93 | 0.88 | 0.88 | 0.88 |

TABLE 2. Comparison of classification report in balanced dataset with CountVectoizer.

We didn't run SVC in unbalanced dataset because of time limitation. The probabilistic classifier did worse in positive samples. It may be caused by the number of positive samples which are greatly larger than negative samples.

**TF-IDF Vectorizer**

| Features: 20386(unigram)/262300(bigram), Train Records: 7500, Test Records: 2500, Negative Samples: 1237, Positive Samples: 1263 | | | | | |
|---|---|---|---|---|---|
| | Precision/ Positive | Precision/ Negative | Avg/ Total | Recall | F1-Score |
| Logistic Classifier (Unigram) | 0.82 | 0.94 | 0.92 | 0.92 | 0.92 |
| Logistic Classifier (Bigram) | 0.89 | 0.94 | 0.93 | 0.93 | 0.93 |

TABLE 3. Comparison of classification report in balanced dataset with TF-IDFVectorizer.

It seems that Logistic Regression can be improved from unigram to bigram (add more features)

3. Learning Curve

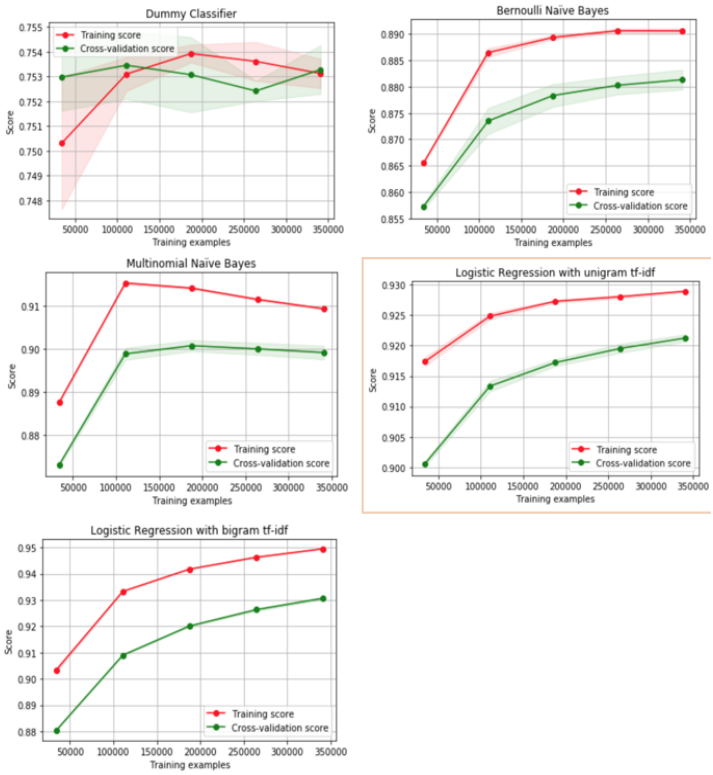


Fig. 7. Learning curve in different models with unbalanced dataset

Only Multinomial Naïve Bayes get very low score at first and the achieve the highest score when number of samples is around 100000. Finally, it will decrease. Other samples will get very low score as well but the scores will increase as number of samples increases.

b) Balanced Dataset

1. Accuracy Comparison

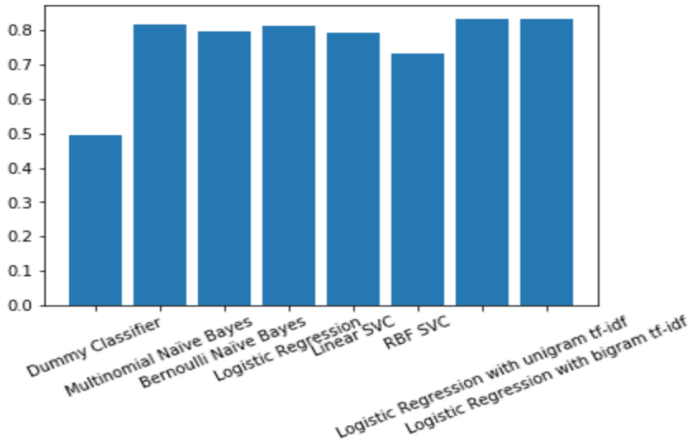


Fig. 8. Accuracy of prediction in different models with balanced dataset

In Fig. 8, it clearly shows Logistic Regression with tf-idf vectorizer achieves the highest accuracy during test. Except for dummy classifier, RBF SVM has lowest accuracy.

2. Classification Report

**CountVectorizer**

| Features: 20386, Train Records: 7500, Test Records: 2500, Negative Samples: 1237, Positive Samples: 1263 | | | | | |
|---|---|---|---|---|---|
| | Precision/ Positive | Precision/ Negative | Avg/ Total | Recall | F1-Score |
| Dummy Classifier | 0.48 | 0.49 | 0.49 | 0.49 | 0.49 |
| Logistic Classifier | 0.80 | 0.82 | 0.81 | 0.81 | 0.81 |
| Multinomial Naïve Bayes | 0.82 | 0.81 | 0.81 | 0.81 | 0.81 |
| Bernoulli Naïve Bayes | 0.85 | 0.76 | 0.80 | 0.80 | 0.80 |
| Linear SVC | 0.79 | 0.80 | 0.79 | 0.79 | 0.79 |
| RBF SVC | 0.70 | 0.78 | 0.74 | 0.73 | 0.73 |

TABLE 4. Comparison of classification report in balanced dataset with Countvectorizer.

We can discovered from TABLE 4. that Logistic Regression and Naïve Bayes has similar accuracy. But Bernoulli Naïve Bayes performs worse in negative samples. In this case, the probabilistic classifier is better than geometric classifier.

**TF-IDFVectorizer**

| Features: 20386(unigram)/262300(bigram), Train Records: 7500, Test Records: 2500, Negative Samples: 1237, Positive Samples: 1263 | | | | | |
|---|---|---|---|---|---|
| | Precision/ Positive | Precision/ Negative | Avg/ Total | Recall | F1-Score |
| Logistic Classifier (Unigram) | 0.81 | 0.85 | 0.83 | 0.83 | 0.83 |
| Logistic Classifier (Bigram) | 0.81 | 0.85 | 0.83 | 0.83 | 0.83 |

TABLE 5. Comparison of classification report in balanced dataset with TF-IDF Vectorizer.

Comparing with unigram and bigram, it doesn't make accuracy gap between them. It means that it may be overfitting when number of features is over a specific upper bound.

3. Learning Curve

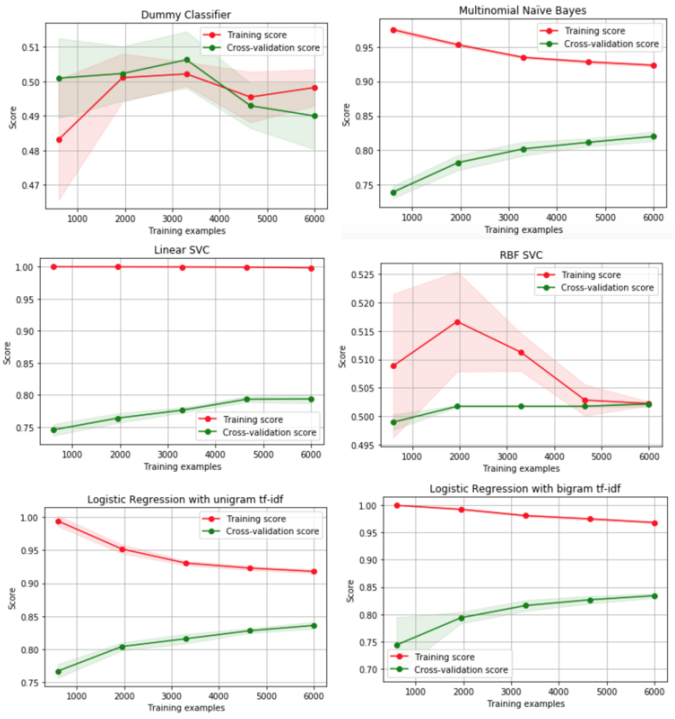


Fig. 9. Learning curve in different models with balanced dataset

These graphs show the relationship between score and size of training sample. Linear SVC is much more stable than other models. Most of models will get lower training score when the training samples become more. But the cross-validation score will be better in the end.

4. Recurrent Neural Network(RNN)

If we don't limit the review to 200 words, it will take too much time for the RNN to train. Here is the distribution of the review length as Fig. 10.

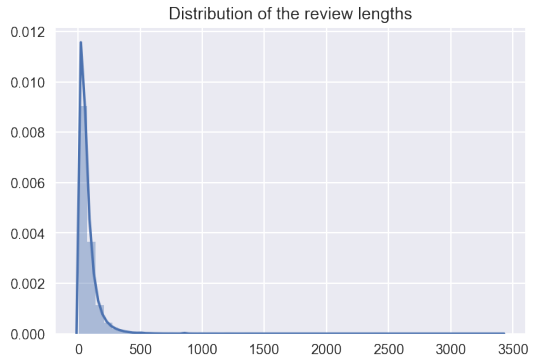


Fig. 10. Review length distribution

The mean and the third quartile of the review length is equal to 79 and 97, respectively. Therefore if we limit the review length to 200 words, we shouldn't lose too much information.

Based on the observation above for our final model, we have decided to include all the vocabulary, have a single LSTM layer, an LSTM size of 256, and a dropout layer with a probability of 0.5.

The best way to see if our model is able to generalize is to test our model on the untouched testing set. In order to measure the performance of the binary classifier, we can plot the ROC curve. The ROC curve consists of plotting the true positive rate against the false positive rate. The best scenario is when the curve matches the left corner. This would mean that we are able to achieve 100% sensitivity (no false negatives) and 100% specificity (no false positives).

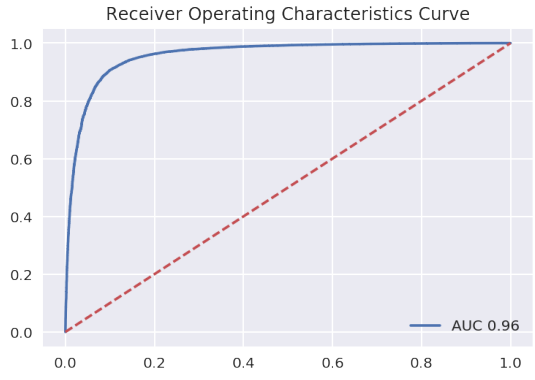


Fig. 11. ROC of RNN (LSTM)

We can observe that the area under the curve (AUC) is equal to 0.96. Knowing that the maximum AUC score is equal to 1, we can conclude that our model does a pretty good job at making accurate predictions on unseen reviews. Moreover our model seems to be very balanced in terms of false positive rate vs false negative rate.

The other important metric is the F1 score, because it also takes into consideration, precision and recall. In our original dataset 78% of the reviews were positive. If we would use a naive algorithm and predict that all the reviews are positive, our accuracy would be 78%. This is why we should use the F1 score in order to measure the performance of our RNN.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0.0 | 0.90 | 0.73 | 0.81 | 12363 |
| 1.0 | 0.93 | 0.98 | 0.95 | 44137 |
| avg / total | 0.92 | 0.92 | 0.92 | 56500 |

Fig. 12. Classification report of RNN

The final F1 score is equal to 0.92. The precision and recall are also equal to 0.92. It means once again that our model has a good balance between sensitivity and specificity. There is room for improvement but our RNN model has learned to distinguish negative fine food reviews from positive reviews.

## VII. CONCLUSION

At first, we can discover that the accuracy of prediction in imbalanced dataset is better than balanced dataset. There are couple of possible reasons, e.g. number of features, size of training data. The main reason should be the number of samples. In the future work, we can try to resample the imbalance data to keep the total size of imbalance data in consistent with the total size with balanced data to make a better comparison.

In terms of current model result, in imbalanced dataset, the Logistic Regression model has the highest accuracy of prediction. In balanced dataset,

the Logistic Regression with bigram tf-idf model has the highest accuracy of prediction.

By comparing LinearSVC and RBF SVM, LinearSVC apparently performs well because of that the number of feature is larger than the of training set. It matches what we learned in the class. (relationship between number of feature and number of training samples). By comparing MultinomialNB and BernoulliNB, MultinomialNB is slightly better than BernoulliNB. We guess the reason is BernoulliNB will do better when feature values are binary. But in this case, terms have multiple types so that we think it will be the root cause.

With different vectorizer, TFIDF and Countvectorizer, it will generate different result. For example, we can improve the model by removing words in very low or high frequency by TFIDF. It will make the prediction model to filter outliers and improve the accuracy of model.

We can see that RNN (LSTM) in this work received a pretty good result -- average precision is 92%, which is higher than most other methods we used in this project. However, it is not much higher than Naive Bayes with TFIDF, which means for some tasks, deep learning is not also the best tool, because classic machine learning models are much faster to fit and predict than deep learning model. And there may also because our model is too simple and hyper parameters haven't been well tuned, but I don't think this model is much better than other methods, because even we make the network more complex, and tune to find the best hyper parameters to make this model perform the best performance, we still cannot ignore the time cost of this work.

In this project, we use multiple training models to make prediction and try to find the most appropriate model on our dataset. These methods can be applied to other dataset and advanced users can implement their specific model by Tensor Flow or other libraries based on the type of dataset. During the process, we learned the analyzing steps and training models in scikit-learn library which are helpful and easy to use.

## REFERENCES


[1] Deeplearningbook.org. (2017). Cite a Website - Cite This For Me. [online] Available at: http://www.deeplearningbook.org/contents/convnets.html [Accessed 11 Dec. 2017].

[2] Bakos, J. 1997. “Reducing Buyer Search Costs: Implications for Electronic Marketplaces,” Management Science (43:12), pp. 1676-1692.

[3] Chen, P., Dhanasobhon, S., and Smith, M. 2008. “All Reviews Are Not Created Equal: The Disaggregate Impact of Reviews on Sales on Amazon.com,” working paper, Carnegie Mellon University (available at SSRN: http://ssrn.com/abstract=918083)

[4] Chen, P., Wu, S., and Yoon, J. 2004. “The Impact of Online Recommendations and Consumer Feedback on Sales,” in Proceedings of the 25th International Conference on Information Systems, R. Agarwal, L. Kirsch, and J. I. DeGross (eds.), Washington, DC, December 12-14, pp. 711-724.

[5] Chen, Y., and Xie, J. 2005. “Third-Party Product Review and Firm Marketing Strategy,” Marketing Science (24:2), pp. 218-240.

[6] Chevalier, J., and Mayzlin, D. 2006. “The Effect of Word of Mouth on Sales: Online Book Reviews,” Journal of Marketing Research (43:3), pp. 345-354.

[7] Clemons, E., Gao, G., and Hitt, L. 2006. “When Online Reviews Meet Hyperdifferentiation: A Study of the Craft Beer Industry,” Journal of Management Information Systems (23:2), pp. 149-171.

[8] Forman, C., Ghose, A., Wiesenfeld, B. 2008. “Examining the Relationship Between Reviews and Sales: The Role of Reviewer Identity Disclosure in Electronic Markets,” Information Systems Research (19:3), pp. 291-313.

[9] Gretzel, U., and Fesenmaier, D. R. 2006. “Persuasion in Recommendation Systems,” International Journal of Electronic Commerce (11:2), pp. 81-100.

[10] Ngo-Ye, T. and Sinha, A. (2012). Analyzing Online Review Helpfulness Using a Regressional ReliefF-Enhanced Text Mining Method. ACM Transactions on Management Information Systems, 3(2), pp.1-20.

[11] Smith, D., Menon, S., and Sivakumar, K. 2005. “Online Peer and Editorial Recommendations, Trust, and Choice in Virtual Markets,” Journal of Interactive Marketing (19:3), pp. 15-37.

[12] https://github.com/yanndupis/RNN-Amazon-Fine-Food-Reviews