Sentiment Analysis of Restaurant Reviews

Sarah-Jane Clarke and **Richard Zhu** and **Alex Teng**
New York University
Natural Language Processing
Fall 2021

Abstract

The goal of this paper is to demonstrate how we have trained multiple classifiers to be able to perform a sentiment analysis of restaurant reviews. This paper will delve into how we have built and trained a Naive Bayes Multinomial classifier, Unigram and Bigram SVM classifier, and a Logistic Regression classifier. We divided a Kaggle dataset of 1,000 restaurant reviews tagged as positive (1) or negative (0) into 80% training data, 10% development data, and 10% test data [3]. The data has 500 positive reviews and 500 negative reviews. We found that the highest scoring algorithm was Unigram TF IDF SVM with 85% accuracy, precision, recall, and f-measure score. We saw improvement of scores through pre-processing such as stemming and stop word removal. These classifiers provide a potential system for customer review web pages to be able to provide a summary of thousands of reviews for other customers or restaurant owners to see the general sentiment towards an establishment.

1 Introduction

Reviews are an excellent way for restaurant owners to receive feedback from customers and for other potential customers to read when deciding whether or not to go to a restaurant. However, it can be tedious to read through each and every review to find the general consensus of whether people have positive or negative opinions of a given restaurant. The goal of this project is to produce a sentiment classifier that is able to analyze restaurant reviews and thus be able to provide valuable feedback and statistics to customers and restaurant owners interested in the reviews. The sentiment analysis contains two main stages: pre-processing/vectorization and sentiment classification. Figure 1 shows the general workflow that we followed to train the classifiers.

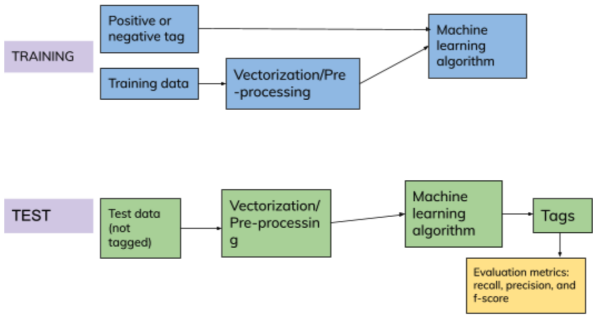

Figure 1: Project Workflow

2 Related Work

Balahur [2] presents a method designed specifically to work with Twitter tweets by addressing the syntactical structure and specificity of the language within the data. The following steps were taken to normalize the language contained: replacement of repeated punctuation with labels, replacement of emoticons with their polarity, lower casing and tokenization, replacement of slang, normalization and marking of "stressed" words like "perrrrrfeeeect", and modifier word matching. Balahur employed supervised learning using SVM SMO trained based on n-grams determined from previously pre-processed tweets. Balahur's algorithm achieved a unigram score of 74.90% and bigram score of 63.27%. A combined unigram and bigram feature with additional word replacements and restrictions was employed to produce Balahur's highest score of 79.97%. Our group's SVM algorithm achieved a greater unigram TF IDF score of 85%, but a lower bigram TF IDF score of 48%. Balahur's results encourage our group to explore new features and feature combinations.

Agarwal, Xie, Vovsha, Rambow, and Passoneau [1] examine sentiment analysis of Twitter data through POS-specific prior polarity features and the use of a tree kernel for feature engineering. This group introduced an emoticon dictionary and acronym dictionary. Each emoticon was labeled with their emotional state from extremely-positive, extremely-negative, positive, negative, to neutral. Acronyms such as "lol" were translated to "laughing out loud". They used prior polarity scoring by assigning words a pleasantness score between 1 and 3 acquired through the Dictionary of Affect in Language. Based on the prior polarity of words, features were added. The group then used a tree representation to combine many features into a succinct representation. Their SVM algorithm achieved a unigram baseline score of 71.35% and they were able to report a 4.04% gain from the baseline by combining unigrams with 100 additional features that fall into polar, non-polar, POS, and non-POS categories. Comparatively, our SVM algorithm obtained a TF IDF unigram score of 85%, though it is not clear how this group vectorized their data. Our group is further motivated by this group's results on top of Balahur's results to train our SVM with additional features and reference the use of tree kernel representation to organize those features.

Further, Hyun Jung Kang, Iris Eshkol-Taravella [5] studied the sentiment analysis of restaurant reviews. This group not only classified the sentiment/ general opinion of the review, but also the reviewer's input/feedback to potential customers and restaurant owners, whether the reviewer wants to return to the restaurant, and the factual statement about the experience [5]. While our group only found the sentiment from each review, we were motivated by this group's high results of 88% using a SVM model to use this model in our project as well.

Mayur Wankhade, Chandra Sekhara Rao, Suresh, Dara, and Baijnath Kaushik [8] conduct an opinion analysis on food reviews using logistic regression. This group started with preprocessing that included removing stop words, tokenizing, counting the number of tokens, and then normalizing the data by assigning weights to the most frequent tokens. This group utilized principal component analysis (PCA) to reduce the feature size which gives the dataset linearity by transforming it to a lower dimension. While our group decided not to use any feature selection nor reduction for our logistic regression model, we saw similar results with an average score of 81% compared to this group's 89%. What this group did differently was split the dataset into unigram, bigram, and trigram classification and analysis. While this yielded different results, our group ultimately decided that the results were similar enough to where only a single performance analysis was required.

3 Restaurant Review Pre-processing

The language employed in restaurant reviews is not largely different from the one used in mainstream media. As such, before applying sentiment-classifying algorithms on the reviews, we aim to stress the presence of sentiment- bearing words by normalizing and disentangling the language contained. The pre-processing stage contains the following steps:

- Removing special characters

  In the first step of pre-processing, we detect special characters ("&", "!", "*", etc.). including special characters in words (2 in "good2eat") and replace the character with a space.

- Lower casing and tokenization

  The restaurant reviews are then lower cased and split into tokens based on spaces.

- Filtering stop words

Subsequently, we filter out common words in the English language (pronouns, conjunctions, propositions, etc.) to remove low-level information so as to give more focus to important information. Examples of stop words include “a”, “the”, “so”, and “what”. We were careful to not filter out common stop words that may bear a sentiment. We identified 44 such words including “minus”, “especially”, “pretty”, and “rather”.

- Stemming

  Finally, we used NLTK’s PorterStemmer [7] to reduce words to a root by removing inflection through removing irrelevant characters. The result is helpful in identifying commonalities and relationships across the dataset.

4 Methods

4.1 Naive Bayes

We implemented a Multinomial Naive classifier imported from the sklearn library. This is a probabilistic model based off of Bayes formula displayed below.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Figure 2: Bayes Formula

Ultimately, this algorithm calculates the probability of a review being positive and negative and takes the highest score as the result. This classifier is trained by consuming pre-processed and vectorized restaurant reviews. The feature vectors were formed using sklearn’s CountVectorizer(). This vectorizer is a bag of words model and creates a matrix where the y-axis holds each document and the x-axis holds all of the words that we see in all of the reviews. Thus, the location of (word 2, Review A) would hold the number of times we see word 2 in Review A. This classifier then uses likelihood and prior probabilities to calculate the probability that a review is positive or negative. While calculating these probabilities, the Multinomial Naive Bayes classifier treats each word equally. It does not take into consideration if a word occurs more in a review than another one, and treats every feature in the dataset as mutually exclusive. This is what makes this method ‘naive’. Regardless, it is able to classify the texts and we see solid results when running with the test data.

4.2 SVM

Our SVM implementation contains Unigram TF IDF, Bigram TF IDF, and Unigram Counts features. We utilized sklearn’s TfidfVectorizer() and CountVectorizer() modules to convert our text data into numeric feature vector forms that our model will be able to understand. TfidfVectorizer() was used to create a matrix of TF IDF features. TF IDF means Term Frequency - Inverse Document Frequency and is a statistic based on the frequency of a n-gram in a corpus that numerically represents the importance of the n-gram for statistical analysis.

$$w_{i,j} = tf_{i,j} \times log(\frac{N}{df_i})$$

$$tf_{i,j} = number\ of\ occurrences\ of\ i\ in\ j$$

$$df_{i,j} = number\ of\ documents\ containing\ i$$

$$N = total\ number\ of\ documents$$

Figure 3: TF-IDF Formula

We computed TF IDF feature vectors for unigrams which is a one-word sequence of words and bigrams which is a two-word sequence of words, like “I love” and “love food”. CountVectorizer() was used to convert the data into a matrix of unigram token counts.

$$Data = ['It', 'is', 'true', ',', 'the', 'risotto', 'is', 'phenomenal']$$

$$\downarrow$$

It	1
is	2
true	1
,	1
the	1
risotto	1
phenomenal	1

Figure 4: Transforming text to token counts

The mean and variance of the feature vectors belonging to the training set were calculated and used to transform the train data. The test data was transformed using the same mean and variance from the training set. The model was then fit using the train data and used to predict the test data. We used sklearn's SVC implementation, which is an SVM module based on libsvm that supports different kernels. It most optimally segregates our dataset into positive and negative reviews by selecting a hyperplane with the greatest margin between support vectors in the dataset.

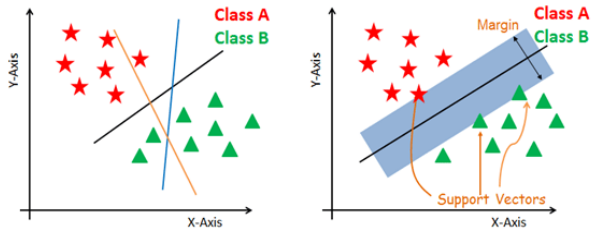

Figure 5: The hyperplane with the largest margins is chosen to separate the dataset into two classes that represent positive and negative reviews.

4.3 Logistic regression

We utilized a Logistic Regression model from the sklearn library. This is an appropriate regression analysis that should be used when the dependent variable is dichotomous which means that there are only two possible classes. In our case, those classes are positive and negative restaurant reviews. Logistic regression uses the sigmoid function:

$$sigmoid(x) = \frac{1}{1 + e^{-x}}$$

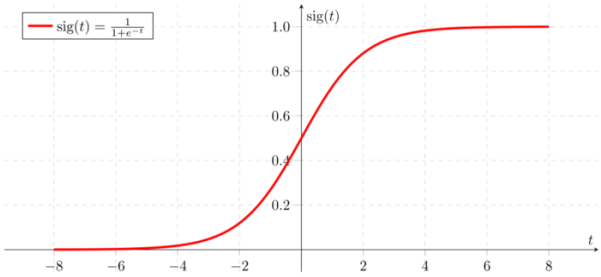

Figure 5: Sigmoid Function

The sigmoid function takes any real value and maps it to a value between 0 and 1 as seen in figure 5. If the curve goes to positive infinity, y predicted becomes 1. If the curve goes to negative infinity, y predicted becomes 0. Therefore, if the output of the sigmoid function is more than 0.5, we classify the outcome as positive, and if it is less than 0.5, we classify as negative. The sklearn classifier takes training (X) and target (Y) vectors as parameters. These vectors were created using sklearn's CountVectorizer() which uses a bag of words model. The goal of this supervised machine learning technique is to learn and approximate a mapping function (X) from the input parameters to output variable (Y). The model predictions are evaluated iteratively and corrected against the output values until, ultimately, an acceptable performance is achieved and it is able to predict whether a given data entry is above or below the 0.5 sigmoid threshold.

5 Evaluation

The table displayed below is a summary of the results for each classifier we produced.

	Accuracy	Precision	Recall	F1-score
Naive Bayes Multinomial	76.0 %	82.6%	70.4 %	76.0 %
Unigram TF IDF SVM	85%	85%	85%	85%
Bigram TF IDF SVM	55%	60%	55%	48%
Unigram Counts SVM	83%	83%	83%	83%
Logistic Regression	81%	83%	80%	81%

Figure 6: Evaluation results

From the results obtained, we can conclude that the best algorithm to be employed in our sentiment analysis of restaurant reviews is a simple Support Vector Machine classifier with a Unigram TF IDF feature. It achieved an average accuracy of 85%, marginally improving over Unigram Counts, which scored an average accuracy of 83%. The TF IDF vectorizer builds a more accurate vocabulary than the Count Vectorizer because it not only considers the frequency of words present in the corpus but also the overall document weightage of a word. Both methods provide a good understanding about the importance of words, but fail to provide linguistic information about the words including similarity with other words, the meaning of the word, etc. Higher levels of accuracy may be obtained with word embedding techniques that use neural network models to learn word associations. Like Balahur [2], we observed that the Bigram TF IDF feature performed the worst, with an average accuracy of 55%. We noticed that a reason for error persisted in the little power the method had
to spot modifiers and negations. Therefore, we plan to study a way in which bigrams of non-consecutive tokens can be implemented to counter this issue. We also plan to evaluate use of higher order n-grams like trigrams in identifying negations and extracting deeper patterns of sentiment.

When running the Naive Bayes Multinomial Classifier with the test data, we saw a decrease in scores from 85% to around 76% accuracy. However, because the pre-processing was closely tailored to the development data, it is not surprising to see the decrease in scores. While training the classifier, we removed different stop words and experimented with different combinations of pre-processing to help achieve higher results. More specifically, the Naive Bayes predicted 38 out of 56 positive reviews and 38 out of 46 negative reviews which we can see in figure 7 below.

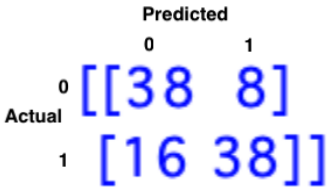

Figure 7: Naive Bayes Confusion Matrix

Moving forward, it would be interesting to look further into any trends there were in the development data when the Naive Bayes classifier incorrectly predicted the positive development to try and increase the number of correctly predicted positive reviews.

The advantages of logistic regression are that it's efficient and straightforward. This makes it require low computation, easy to implement, and not require scaling of features. However, a downside to its simplicity, Logistic Regression is unable to handle a larger number of categorical features/variables and suffers when presented with non-linear problems and as such, will not perform as well with independent variables that are not correlated with each other or with the target variable. Thankfully, with our relatively compact and pre-processed data set and linear variables, we were able to get a high score of 81%. Wankhade, Rao, Suresh, and Kaushik [8] were able to see improvements to their logistic regression results by implementing unigram, bigram, and trigram models. This could have a potential improvement to our model as well.

6 Future work

In Jung Kang, Hyun & Eshkol-Taravella's [5] sentiment analysis of restaurant reviews, they classify not only positive and negative sentiment, but also mixed opinions. A potential expansion of this study is to incorporate neutral reviews into our dataset and see how our evaluation results change. It would be interesting to see if it aids the classifiers ability to distinguish between positive and negative reviews with that added sentiment. An issue we plan on examining is negation, e.g., differentiating the sentiment associated with "I liked that pizza" and "I never liked that pizza". A possible approach is utilizing known words databases. In our testing, we did not use previously developed dictionaries containing phrases and words with positive and negative connotations. As part of their feature-based approach, Agarwal, Xie, Vovsha, Rambow, and Passoneau [1] used an emoticon dictionary and a Dictionary of Affect to assign emotional state labels to emoticons and pleasantness scores to words respectively. Similarly, our group plans to test features developed using dictionaries denoting the sentiment of word patterns. For example, the WordStat Sentiment Dictionary [6] includes 9164 negative and 4847 positive word patterns. Negative sentiment is denoted when a negative word is not preceded by a negation within three words of the same sentence or when a positive word is preceded by a negation within three words of the same sentence and the rules for positive sentiment are the same vice-versa. Potential implementations include identifying and removing non sentiment bearing words, assessing the correlation and indicators between frequent negative and positive words, and correlating specific sentiment-predictive recurring words with pleasantness/satisfaction scores. Additionally, we plan to evaluate the impact of more extensive linguistic processing by performing POS-tagging and sequence labeling. Finally, we would like to study the performance of our approach in the context of restaurant reviews that are specific to a specific cuisine, in which reviews can be further contextualized using other cultural and culinary based information sources.

7 Conclusion

We presented results for sentiment analysis on restaurant reviews. We investigated 3 different models and found the best results with Unigram TF IDF SVM classification. However, the results could have varied by a large percentage had we conducted a non-linear analysis such as including neutral reviews along with the positive and negative. Ultimately, we conclude that these models are sufficient in being able to analyze restaurant reviews and provide feedback and statistics to customers and restaurant owners.

8 Work Division

8.1 Technical Work

- Pre-processing (Sarah-Jane, Alex)
- Naive Bayes (Sarah-Jane)
- SVM (Richard)
- Logistic Regression (Alex)

8.2 Written Work

- Abstract (Sarah-Jane)
- 1 Introduction (Sarah-Jane)
- 2 Related Work (Sarah-Jane, Richard, Alex)
- 3 Restaurant Review Pre-Processing (Richard)
- 4.1 Naive Bayes (Sarah-Jane)
- 4.2 SVM (Richard)
- 4.3 Logistic Regression (Alex)
- 5 Evaluation (Sarah-Jane, Richard, Alex)
- 6 Future Work (Richard)
- 7 Conclusion (Alex)

References

[1] A. Aggarwal, B. Xie, I. Vovsha, O. Rambow, R. Passonneau. Sentiment Analysis of Twitter Data. Proceedings of the Department of Computer Science Columbia University. Retrieved from https://aclanthology.org/.

[2] A. Balahur. Sentiment Analysis in Social Media Texts. *Proceedings of the European Commission Joint Research Centre*. Retrieved from https://aclanthology.org/.

[3] "Restaurant Reviews." *Kaggle*, 7 June 2021, https://www.kaggle.com/d4rklucif3r/restaurant-reviews/code.

[4] "API Reference." *Scikit*, https://scikit-learn.org/stable/modules/classes.html .

[5] Jung Kang, Hyun & Eshkol-Taravella, Iris (2020). An Empirical Examination of Online Restaurant Reviews. *Proceedings of the 12th Language Resources and Evaluation Conference*, 4942–4947. Retrieved from https://aclanthology.org/.

[6] "Sentiment Dictionaries." Provalis Research, 26 January 2018, https://provalisresearch.com/products/content-analysis-software/wordstat-dictionary/sentiment-dictionaries/.

[7] "Sample usage for Stem." NLTK Documentation, 19 October 2021, https://www.nltk.org/howto/stem.html.

[8] *A Sentiment Analysis of Food Review Using Logistic Regression*. International Conference on Machine Learning and Computational Intelligence-2017, https://www.researchgate.net/publication/334654833_A_Sentiment_Analysis_of_Food_Review_using_Logistic_Regression