**Sentimental Analysis of Restaurant Reviews**

Farhan Azhar, Wajid Bin Aqeel

National University of Computer and Emerging Sciences

**Abstract**

Sentimental Analysis is a part of the Natural Language Processing which deals with extracting the hidden sentiment (opinion and subjectivity) lying in the text. For computing machines, it’s quite difficult to find the hidden sentiment, so Machine Learning algorithms play a role in extracting the sentiment from the text. In this paper, we will be discussing the different classifying algorithms for sentimental analysis while using the restaurant reviews as our primary data. We had used Naïve Bayes as base line model. We also compared our results with the other classifying algorithm. The goal of our project is to build a classifier that classifies the review of customer into positive and negative classes based on the text.

**Related Work**

Text Analysis is not a new problem, researchers have been trying to solve this problem since decades. Sentimental analysis is kind of text analysis and natural language processing on which research has progressing since many years. Since yelp publicly introduced the data set challenge, many have tried to find the insights from the given data. Many researchers had performed sentimental analysis on their dataset. In their research Yun Xu, Xinhui Wu, Qinxia Wang [1] predicted the ratings of text reviews using the sentimental analysis of text review. They had used Neural Networks and Naïve Bayes as their main algorithm for classifying the reviews. They got approx. 70% precision for negative reviews while 76% precision for positive reviews using the perceptron learning algorithm. While using the Naïve Bayes their accuracy dropped to 38% for negative reviews and 58% for the positive reviews. Kent Lee and James Ross [2] predicted the ratings based on reviewer’s comments. The main algorithm used by them were Multinomial Naïve Bayes. They got an accuracy of 0.5 to 0.6 using different experimental setup.

**Background**

Different types of reviews are available online for different products. Restaurant likewise also get reviews from customers in the form of textual review and numeric ratings. Yelp is an online platform where customers can go and giving rating to the business. The reviews give us an idea about whether the customer liked the business or not. There should be a mechanism to predict the numerical ratings from the textual reviews, since it is quite difficult for the computer to predict the actual feedback of customers from textual reviews. This would be quite helpful for business owners as they could get the customer likeness or disliking in the form of ratings from the free text review. This could also help many websites offering products as they could rate the product or service based on the customer comments. This mechanism is known as opinion mining or sentimental analysis to be precise.

**Goal**

The main goal of our project is to use the textual review given by the customer to predict the numerical ratings (Positive or Negative). We would be using the supervised learning algorithms for prediction of ratings/sentiment (Positive or Negative) using free text of reviews. The algorithms we would be using include Multiclass Naïve Bayes, SVM. We had used classification report of scikit to evaluate our results. For feature selection we defined our own feature set by removing the stop words, punctuation marks and tokenizing the text. We will be discussing the algorithm used in detail further in the report. We would also be comparing our results with non-machine learning methods.

**Experimental Settings**

All the classifiers used for our experiment were implemented using the algorithm implementation provided in the scikit library. While performing the experiment with Naïve Bayes, we performed the experiment with multiple analyzer which is the text tokenizer i.e. we performed the experiment with and without removing stop words, with and without performing the stemming. Moreover, to select the best value for top max features in the feature selection, we performed the experiment with different values ranging from 5000 to 13000. However, we obtained the best results at the value of 9000. So, we selected 9000 top features for our Uni-gram representation. Moreover, for bi-gram and tri-gram experiment was performed with default settings and with the removal of stop words. For SVM, we provided the feature vector in the form of binary format using binarizer provided by skLearn Preprocessing Library. For evaluation of our results, we had used classification report provided by the sklearn metrics library.

**Data**

The data we used in this experiment was provided by Yelp which is available at <https://www.yelp.com/dataset/challenge>. However, this dataset was available in JSON files, so we opted an online data source which was a portion of the yelp data available in the form of CSV files. The link of the online data source is <https://www.dropbox.com/s/wc6rzl1a2os721d/yelp.csv?dl=0>. The yelp dataset contains approximately 1,569,264 business reviews. The most prominent category was restaurant containing 990,627 restaurant reviews. The dataset used by us for this experiment consisted of Restaurant reviews. Since they are too large in number too, so we ran our Algorithms on the sample size of 10,000.

The data consisted of 10 columns, however we only considered review text and stars column, since they only were only useful for textual analysis. We split the data using train\_test\_split of sklearn. The ratio in which we split data was 70% for training and 30% for testing. Further we also added a column text length of review for better understanding of data.

The stars (ratings) in data were from 1 to 5, with 1 being the negative and 5 being the positive. Since our goal was to predict that the text of the review was positive or negative, so we only selected the reviews with stars either 1 (negative) and 5 (positive), This also helped us in getting a better accuracy. We didn’t consider the neutral sentiment since our goal was to predict that whether the text was positive or negative.

The first five rows of our dataset are



**Data Split**

For splitting our data into test and train portions, we had used the train\_test\_split from the sklearn model selection library. Data was divided into 0.7 and 0.3 ratio for training and testing data respectively for Naïve Bayes. However, for the SVM, we had split the data in the ratio of 0.75 and 0.25 for training and testing data respectively using the same library of the sklearn.

**Preprocessing**

For our data preprocessing part, we removed the punctuation marks, stop words and performed stemming on the data using Porter’s Algorithm.

**Feature Selection**

We have performed several feature selection algorithms on our data which includes n-grams (Uni-gram, bi-gram and tri-gram) and binarize the data. For feature selection we removed stop words (most common words in a language) from our data as they do not play any role in our analysis, stemming (to remove affixes from our features) and removing punctuation marks. We also performed an analysis using the top max features instead of using the entire features in case of n-grams.

For naïve Bayes the n-gram feature vector was used. This was implemented using count vectorizer and our own analyzer which removed the stop words, punctuation and performed stemming. Secondly, we also made our feature vector as the default parameter to the Count Vectorizer (which converts a collection of text documents to a matrix of token counts).

**Unigram**

In case of Unigram the text is divided into tokens of single words. For Uni gram we used our own customized analyzer (which decides whether text should be divided into Uni, bi or tri gram representation) for tokenization. Our own defined function for analyzer used Porter’s Algorithm defined in NLTK library. Moreover, for punctuation marks and stop words removal NLTK library was used. On the contrary we also used the default function available in count vectorizer. In this case we analyzed the results with and without removing the stop words. Results were also analyzed with using the entire feature vector and using the top max features (most occurring feature) in the feature vectors.

**Bigram and Trigram**

In case of bigram the text is divided into tokens of two words and for trigram it is divided into tokens of three words. We used the default function available in count vectorizer for the analyzer. We analyzed the results with and without removing the stop words. Results were also analyzed with using the entire feature vector and using the top max features (most occurring feature) in the feature vectors. Since the count vectorizer doesn’t support stop words removal above Uni-gram, so we used the function to remove to the features above a certain threshold. This can be used in this scenario as stop words are the most occurring words in any text, so they could have a document frequency of more than 70 percent.

**Effect of removing Stop Words**

The removal of stop words from the text had different effect on the results of the classifier for different representation of the count vector. In case of unigram upon removing the stop words, the precision and recall of the negative reviews dropped, however the precision and recall for the positive reviews showed a slight change with a change of unit difference. If we selected the top features (most occurring) along with removal of stop words then in that case, recall improved for both positive and negative reviews. However, the precision improved for positive reviews and on the other hand dropped for negative reviews. In case of bi and tri gram, since the stop word removal were not available in case of count vectorizer, so we removed the stop words via removing the words with document frequency above then 0.7 (methodology described in the documentation of count vectorizer). For bigram the precision and recall improved for negative reviews upon removal of stop words, however remained same for the positive reviews. For trigram the precision and recall dropped for negative reviews upon removal of stop words, however almost remained same for the positive reviews.

**Effect of stemming**

We performed stemming on the unigram representation. Stemming basically chops off the words, so that words can be in a common base. This is required as in English language a word can have multiple forms such as verb, noun, adjective but they may refer to the same meaning in the text. The effect of stemming was that the precision decreased slightly for negative reviews and increased slightly for the positive reviews. However, there was an increment in the recall of negative reviews and slight decrement in the recall of the positive ones.

**Binarize**

The representation used to feed input to SVM was binarized form. The features were basically a dictionary of words found in the input. However instead of using the count for each input, one or zero was used to represent whether the word is present or not. Binarized form has no threshold for including a word while we can also define a threshold which will assure that only words with occurrence of greater than or equal to that threshold will be mapped to one. We used the variant with no threshold in our experiment.

**Result and Discussion**

**Evaluation Metric**

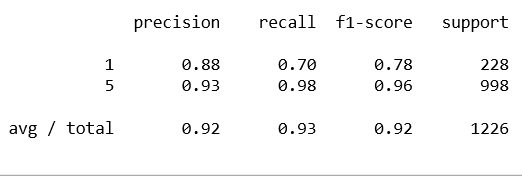
Accuracy is one of the most common measure to analysis the performance of a model and it works pretty in the case if the data is not inclined to one end. As in our case the data is a bit more inclined towards the positive reviews as compared to negative reviews, so in our analysis we have used other measures as well to accurately predict the performance of our model which includes Precision, Recall and F1-score. Precision is the ratio of correctly predicted positive result over the total predicted positive result. Recall is the ratio of correctly predicted positive results over all the positive result in the actual class. F1-Score is the average of precision and recall. Precision is generally given as Precision = tp/tp+fp. Recall is given as Recall=tp/tp+fn. Where tp=true positive (the number of correctly predicted positive results), tn=true negative (the number of correctly predicted negative results), fp =false positive (when the actual class say no, and our model say yes) and fn = false negative (when the actual class say yes, and our model say no)

**Naïve Bayes**

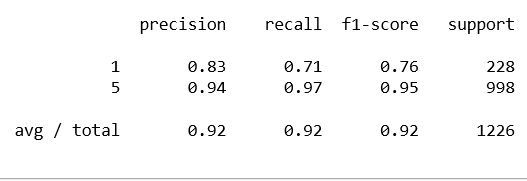
Naïve Bayes is a simple classification model, which helps you to classify by simply multiplying some of the probabilities. Depending on the feature set provided to the classifier, it could give you good accuracy. In this case of sentimental analysis, the feature vector is bag of words. Although Naïve Bayes has a basic assumption that, all features i.e. words are independent of each other, we could use different representations for feature vector i.e. n-gram. Depending on our requirement, we could test different n-grams (Uni, bi, tri) and see which provide us with better results. Basically, English language consists of different parts of speech, if we only consider the single words, many factors that affect sentiment would be ignored.

We used naïve bayes with different preprocessing techniques and feature vector. We started with the default analyzer provided by count vectorizer and using Uni-gram as our representation. The results we obtained after different pre-processing techniques are shown below

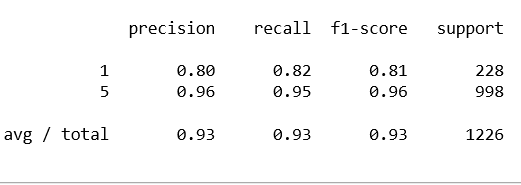
**Results obtained using Uni-gram and without removing stop words**

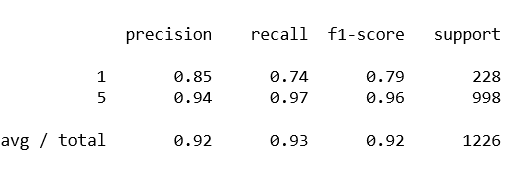


**Results obtained using Uni-gram and after removing stop words**



**Results obtained using Uni-gram and selecting top max features after removing stop words**

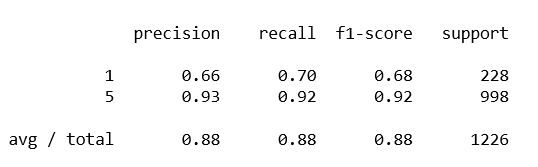


**Results obtained using Uni-gram and after stemming and removing stop words** 

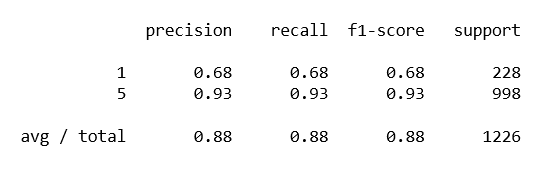
We would be discussing the results in terms of f1-score since it is the average of precision and recall. While using the simple vectorized features, without removing the stop words, our score for the positive reviews was greater as compared to the negative reviews. This might be due to the spread of our test data which would be more inclined towards the positive reviews which means that our data would have a higher ratio of positive reviews. When we removed the stop words, the result of naïve Bayes for the positive reviews improved however, the results were not favorable for the negative reviews. This could be due to the same reason that majority of our training data contained positive reviews. Hence, we had a small number of negative reviews, so upon removal of the stop words, we might have removed some of the words from the dictionary which were common for the negative reviews. Secondly for negative reviews we didn’t append not to negative words, so upon removal of stop words, some words which support negativity such as ‘not’ might have been removed from the dictionary or our feature vector. This resulted in the decrease in the f1-score of the negative reviews.

Now we also performed the experiment of the Uni-gram with selection of the max features and after stemming. In both case, stop words were also removed. For max feature selection we obtained maximum f1-score for the negative reviews. This was since top max features were selected and due to there selection, the data which was inclined towards the positive would have been balanced hence resulting the improvement of the score of the negative reviews. Almost similar results were also obtained using stemming. In this case also the above-mentioned reason of balancing of inclination of data towards positive, improved the results for the negative reviews.

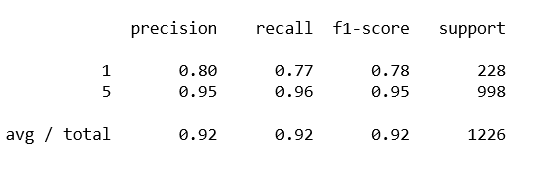
**Results obtained using bi-gram and without removing stop words**



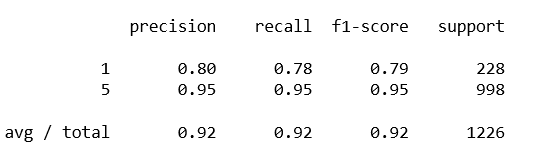
**Results obtained using bi-gram after removing stop words**



**Results obtained using tri-gram and without removing stop words**



**Results obtained using tri-gram after removing stop words**

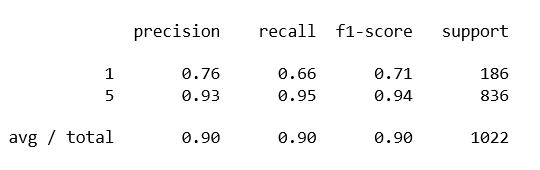


While using the bi gram with stop words, our precision and recall, for both negative and positive and negative dropped. However, there was a slight drop for the tri-gram representation. This drop in f1-score in case of bi gram might be since our data might may be based on the words or phrases that give the entire review its sentiment. However, when we used the removal of stop words by removing the words with document frequency of greater than 0.7, our results showed an improvement. When using the bi-gram, with and without removal of the stop words, the results of the negative reviews had a large error. This error might have been generated due to fact that the negative words used in the review would be either based in single words, or they are based on phrases larger than length of two. The result was such pair of two words would be found in less negative reviews, so the classifier classified it as positive. Moreover, precision also gives an idea us that many of the positive reviews have been classified as negative, resulting in the decrease of the precision. This is also since our dictionary based on pair of words, couldn’t identify the negative correctly, so positive were also misclassified.

**SVM**

We had implemented SVM using the sklearn linear SVC. SVM is another algorithm which is quite suitable for text analysis. SVM works on the principle that the dimensionality of the features is independent of the learnability of the algorithm. Since for text analysis we have many features, so SVM becomes a suitable algorithm for this. The implementation of SVM uses over fitting protection, so we need not to worry about the overfitting. The feature vector which was fed to the classifier was a binary vector which used zero or one to describe the presence of a word, rather than considering its count.

**Results obtained using SVM and binarized vector as features**



The results we obtained were quite impressive for the positive reviews however for the negative reviews classified by the classifying model, recall value was quite low which indicated that the svm classifier was unable to detect the negative reviews or a minimum number of reviews. Moreover, classifier has satisfying results regarding correctly classifying the negative reviews a negative. This was inferred from the fact that the precision of the classifier for the negative reviews was around 0.76.

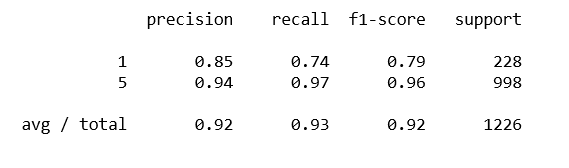
**Non-Machine Learning Method**

**Vader Sentimental Analysis**

Vader Sentimental analysis calculates the sentiment of text based on the sentiment of the individual words. Based on the words in the lexicon, a word is rated as negative or positive. Then using the words in the text and weightage of negative and positive words, it decides the sentiment of the text. Vader returns a score containing neg and positive rating. Based on the score returned, we classify each review as positive and negative which so ever gets a higher score.

When we ran the Vader, sentimental analysis using its implementation provided by vaderSentiment library, we got the following results

**Results obtained using Vader Sentimental Analysis**



**Comparison of Algorithms**

We got the best precision for positive reviews from the Naïve Bayes using Uni-gram and selecting the top max features. For the negative reviews, we got maximum precision using the Uni-gram and without removing the stop words. The recall was maximum for both positive and negative reviews using the Uni-gram representation of the feature vector. Moreover, when bigram and trigram were used as feature vector, we observed a drop in the precision and recall of the algorithms. The results of SVM were not bad, but we got a bit less precision and recall using SVM for both negative and positive reviews. The results obtained using the non-machine learning method i.e. Vader sentimental analysis surprised us with its result. The results were equally good as Naïve Bayes.

**Conclusion and future work**

We had used different Supervised learning algorithms i.e. Naïve Bayes and SVM for classifying the Textual reviews of yelp dataset into positive and negative reviews. We had used precision and recall as our evaluation metrics. We got the highest precision and accuracy for Naïve Bayes after removing stop words and selecting top max features.

Future research could use the sentimental emotions (funny, useful and cool) for more accurate analysis of the textual reviews. This could help us to know how others user feel about the review. Moreover, this experiment was specifically for restaurant, other business reviews sentimental analysis could be done while combining the business specifics for better results.

**References**

[1] Yun Xu, Xinhui Wu, Qinxia Wang,” Sentiment Analysis of Yelp ‘s Ratings Based on Text Reviews”, Stanford University.

[2] Kent Lee and James Ross,” Prediction of Yelp Ratings Based on Reviewer Comments Segmented by Business Type”, Stanford University, Department of Computer Science.

[3] <http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html>

[4] <http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html>

[5] <http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html>

[6] <https://github.com/cjhutto/vaderSentiment>

[7] <http://www.nltk.org/>