Sentiment Analysis

Experimented with NLTK and multiple pre-processing algorithms

Mahrad Pisheh Var   
Computer Science DepartmentUniversity of Essex, Colchester  
mpishe@essex.ac.uk

*Abstract*—Sentiment analysis is one of the common methods used in monitoring social media and process human’s opinions online. In this project four different pre-processing methods will be experimented to observe their effects on sentiment analysis, the classifier used is by NLTK Naïve Bayes, there are four different pre-processing methods used such as Sentiment lexicons, TF-IDF, Negation words and Bigrams, each were compared by testing their accuracy in classifying the sentiments.

Keywords—NLTK, Sentiment Analysis, Naïve Bayes, Kaggle, Sentiment Lexicons, Bigrams.

# Introduction

## Sentiment Analysis

Sentiment analysis is a perusal method where it allows the system to gain a survey of the public opinion behind the focused topics. One of the main effective use of sentiment analysis is on social media where it is used in monitoring tools like “*brandwatch analytics*” [1].

The human language is quite complex, and it can contain many different complexities such as, idioms or informal grammatical structures. Sentiment analysis has an ability to be deployed to analyse these elements. Cultural variations, slang, mistypes or grammatical distinctions can occur in an online platform where the users are not assigning an adequate amount of attention into the structure of their comments. Therefore, they deliver what they think in an immediate burst. Moreover, humans are quite clever in interpreting the tone, the use of sarcasm is commonly seen on online platforms. Consider the following sentence:

“I love data science module, especially when the lecturer puts in no effort when helping their students”.

A healthy human being can interpret that sentence to be sarcastic, but to structurally analyse the sentence, the comment uses “love” as an expression that they are liking the module. From the example given, the contextual understanding can help the sentiment analysis system to highlight the sentence to be negative.

## Purpose

The project is purposed to train a corpus which was taken from the “The Rotten Tomatoes” movie reviews and it is described to be a commonly used resource in topic of sentiment analysis, each train dataset entry contains:

1. An identifier
2. Sentence itself
3. Sentiment value of the sentence which varies from 0 to 4, labelling the data as negative, somewhat negative, neutral, somewhat positive and positive accordingly.

The trained classifier then classifies the test corpus. The test corpus will only have the identifier and the sentence. In the project, four different pre-processing approaches were experimented on the train dataset and each were trained by NLTK classifier [2]. These four experiments involved using:

1. Normal phrase pre-processing.
2. Using Sentiment lexicons to boost the classifier.
3. Using Bigrams.
4. Using Negation words list.

# Literature review

## Common features used in sentiment analysis

In classification of sentiments there are number of commonly used algorithms to help the classifier to have a unique approach in classifying the sentences.

These features are:

1. Term Presence Vs Frequency TF-IDF, this feature is mainly used to model documents. In sentiment analysis aspect of analysing the documents the most unique words are preferred to be found [3].

The [3] mentioned, finding term presence in the document increases the overall performance of the system. The action of finding the term presence originates from an algorithm that goes through each phrase in the document and validates the phrases presence in the document. The validation can be done by a simple True or False (1 or 0).

1. “The n-grams are a word-stem, part-of-speech pair ” [3]. [3] suggests that position of the phrases or terms are important to consider in a sentence, the polarity of the phrase must be considered where it can have positive or negative meaning when positioned in a sentence.
2. Part-of-speech, [3] mentions adjectives are beneficial to be detected in a sentence where they are an adequate sign of presence of a sentiment in a text.
3. Syntax features, these features can contain text features such as negation, intensifiers and diminishers. Kudo et al. [4] performed an experiment where a subtree-based boosting algorithm was used with dependency-tree-based features where it was mainly for classification of polarities. The result of [4] experiment showed that the boosting algorithm had significant improvements in results on the *bag-of-words* baseline.
4. Negations has an important role in sentiment analysis, negation words are list of words that forms the polarity of the sentence [3]. Consider the following, “I do like data-science” and “I don’t like data science”, the negation word “don’t” changes the polarity of the sentence. In the process the “don’t like” term is syntax into “like-NOT” where it represents its polarity.

## Overview on Sentiment lexicons

Sentiment lexicons are a directory used for matching the words with its data to decide on the word’s polarity. In the lexicon-based method, the polarity of each words is calculated with a scoring system where it describes the intensity of the polarity of the word or the objectivity element of the word in the dictionary [5]. One of the downsides of this method is the requirement to human-labelled document.

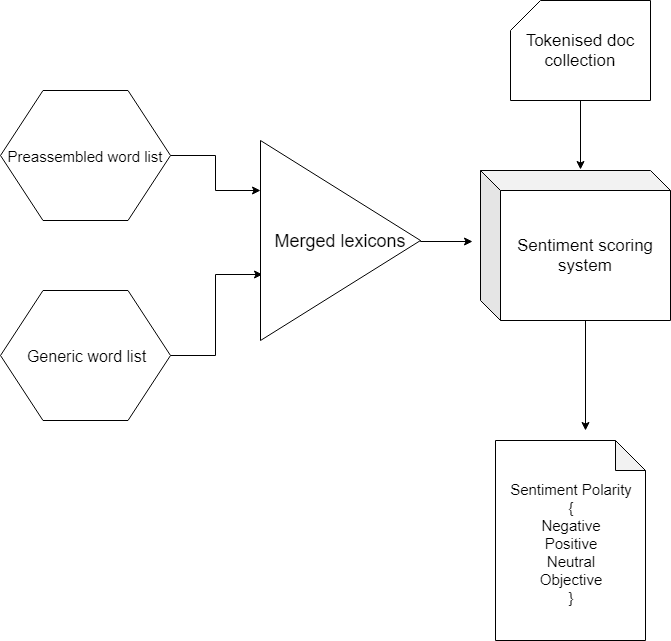


Figure 1 Sentiment Lexicon method visualized inspired by [5].

Figure 1 visualises the sentiment lexicon method where before the tokenised document words are sent to the sentiment scoring system, the preassembled word-list and generic word-list are processed to merge their lexicons. The merged lexicons list is matched with tokenised doc words and their polarity is calculated. As it can be understood there are two sub classification before the main classification process which helps the and boosts the classifier’s accuracy.

## Effects of Bigram on classifiers

Bigram are mainly used to capture features that are in combination of two words. For example, “not bad”, is a term with combination of two words “not” and “bad”, the unigrams only capture one word at the time and fails to capture these combinations. [6] suggests that bigrams can help the accuracy, the comparison between unigrams and bigrams performance in sentiment analysis using Naïve Bayes algorithm is shown in Table 1. The content of Table 1 is taken from [6].

|  |  |
| --- | --- |
| Unigram accuracy | Bigram accuracy |
| 74.56% | 76.44 |

*Table 1. Comparison between Unigram and Bigram accuracy in sentiment analysis using Naïve Bayes algorithm.*

The Table 1 shows that the Bigram model has almost 2 percent more accuracy than Unigram model.

## Previous experiments

[7] performed sentiment analysis on Twitter dataset. In the pre-processing they have used “POS-specific prior polarity features” [7]. The new features proposed by [7] are used in the tree kernel. In the experiment, [7] shown that they have used different models such as:

1. Unigram
2. Senti-features
3. Kernel
4. Unigram senti-features
5. Kernel senti-features

Sent-features are claimed by [7] that involves their method, “prior polarity of part-of-speech”.

In the Table 2 it is shown that [7] method has achieved the highest accuracy of 75.39 percent using Unigram senti-features that only uses prior polarity POS feature.

|  |  |
| --- | --- |
| Model | Avg. Accuracy(%) |
| Unigram Senti-features | 75.39 |
| Kernel Senti-feature | 74.61 |
| Kernel | 73.93 |
| Unigram | 71.35 |
| Senti-features | 71.27 |

*Table 2.Experiments results made by [7], the content of the table is taken from [7].*

Comparatively, [8] experimented sentiment analysis on French movie reviews. the analysis was based on:

1. Shallow POS tagging.
2. Chunking
3. Simple negation forms.

The [8] suggests that the method was not enough and to improve the classification, they have extracted from SentiWordNet [8], the semantic orientation required to improve the algorithm.

The experiment described by [8], shows that the process includes extracting 2000 French movie reviews where half of the reviews are considered positive and the other half are negative reviews. The reviews are first pre-processed with the TreeTagger[Sch94] [8].

SVM (Support Vector Machine) was mainly used to classify the polarities [8].

Table 3 shows the performance of most relevant feature list in the experiment. Each entry in the table 3 is a combination of features with unigrams and lemmatization and their performance is calculated in percentage. these combined features are:

1. Negation features.
2. POS features
3. Polarity features

|  |  |  |
| --- | --- | --- |
| Feature combination | Pos. Acc[%] | Neg, Acc[%] |
| Unigrams + lemmatization + Negation | 92.50 | 94 |
| Unigrams + lemmatization + POS | 93 | 92.50 |
| Unigrams + lemmatization + polarity | 93 | 93.50 |

*Table 3.Experiments results made by [8], the content of the table is taken from [8].*

In Table 3. It can be understood that negation features have helped the negative classification accuracy but comparatively, the positive accuracy was lower than the other two experiment. Using POS in the experiment has shown that positive and negative classification was balanced to be around 93.25 percent.

Using polarity features did improve [8] experiment higher than the other features resulting in 93.25 percent in average.

## NLTK in Naïve Bayes classifivation

Naïve Bayes classification is a supervised learning introduced by NLTK [2]. It is called supervised because the classifier requires a training corpus to learn and label or classify correctly for each input. In Naïve Bayes, the process of assigning labels to a given input considers the features as well as the input. In the process of classification, the classifier starts with calculating the prior probability of each label [9]. Each feature will have a contribution in prior probability which changes the value and the decision made by the classifier.

C:\Users\Mahra\Downloads\Untitled Diagram (2).png

Figure 2 process of classification in Naive Bayes (NLTK)

# Methodology

According to literature review, each author has tried to explore before choosing the best method to use in sentiment analysis, these methods either were implemented in the pre-processing or post-processing. The methodology used consists of four different pre-processing methods and one single evaluation method.

The overview of the system is visualised in figure 3.

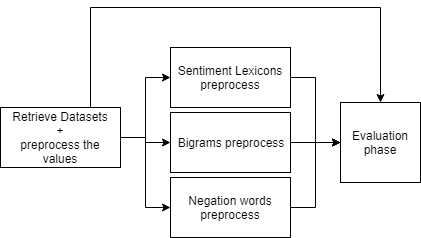


Figure 3 overview of the system

## Preprocessing

[3] term presence method has inspired the project’s pre-processing implementation where each document phrase or term is stored in a directory which holds a Boolean value indicating the presence of the term in the document.

Prior to [3] method, each phrase is processed and converted into lower case and redundant phrases or stop words are removed by checking with NLTK stop words corpus. Other processing algorithms such as regexTokenizer is used to tokenise the phrases.

## Sentiment Lexicons

Sentiment lexicons method from [5] was analysed and a sentiment lexicon corpus was acquired from [10] named “*subjclueslen1-HLTEMNLP05.tff*”. A scoring system was implemented inspired by [5] where it checks the polarity of each term matched and creates a dictionary containing scores for each word’s weak and strong polarity.

## Bigrams

In the Bigrams experimentation, NLTK’s method BigramCollectionFinder was used to add the most frequent significant bigram to the dictionary to be learned by the classifier.

## Negation words

A list of negation words was made in a text-file, each word represents a negation word. These words are:

1. “no”
2. “not”
3. “never”
4. “nowhere”
5. “no one”
6. “rather”
7. “hardly”
8. “scarcely”
9. “rarely”
10. “seldom”
11. “neither”
12. “nor”

The algorithm checks within the document and finds the likelihood of these words and creates a dictionary of the term presence.

## Evaluation on approaches

Each method explained will be evaluated by dividing the dataset into four sub dataset and four step incremental trainings will be performed on each dataset. It has been decided to divide each sub dataset into training set and test set to by one train the classifier and the other to test the classifier. A confusion matrix, average accuracy and most informative features were selected and visualised.

# Experiments

First datasets were divided into four sub datasets, each sub dataset contains around 39014 entries. Each dataset is progressively trained by the classifier chosen the NLTK naïve Bayes. A confusion matrix and their accuracy are compared.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Method | Accuracy of the Classifier (%) | | | | |
|  | 1st test | 2nd test | 3rd test | 4th test | Average |
| raw preprocessed data | 61.3 | 53.6 | 56.3 | 52.5 | 55.92 |
| SL features | 55.3 | 52 | 50.7 | 47.9 | 51.47 |
| Negation words | 46.4 | 44.7 | 45 | 41.1 | 44.3 |
| Bigrams | 60.5 | 51.7 | 53.7 | 50.11 | 54 |

*Table 4.Experiments results on the methods explained.*

Table 4 shows that each method was trained and tested by the classifier in four different steps. Each step the accuracy of the classifier was calculated. According to Table 4, without using other pre-processing methods such as sentiment lexicons, negation words and bigrams, the classifier seems to have lost less accuracy within each test whereas other pre-processing methods have incrementally lost accuracy significantly as shown in the decline for each line as shown in Figure 4.

Figure 4 progress of classifier for each method

# Evaluation

As it was observed in Table 4, the raw preprocessed data had the highest stability in comparison to other methods , it can be concluded that raw processed data have an average of 55.92 percent which other such as Bigrams had 54 percent and SL- features had almost 51.5 percent accuracy. Moreover, it can be observed that negation words method had 44.3 percent which is the lowest than the other methods. This failure in reaching the 50 percent accuracy can be explained. The negation words as it was observed by [8], it only helped the classifier to find negative sentiments and depending on the variation of the entry’s sentiment, it can be seen, Negation words method is not a suitable method to be used individually in the pre-processing of the data. A slightly preferably option than raw processed data method is Bigrams which achieved an average of 54 percent. It is shown that after each test the classifier had less struggle with positive and negative sentiment classification, but it done poorly in classifying the neutral sentiments.

The raw preprocessed data method was used on the submission test data and the submission csv format file containing phrase ID and sentiment was uploaded to [11].

The score achieved was shown on Figure 5.

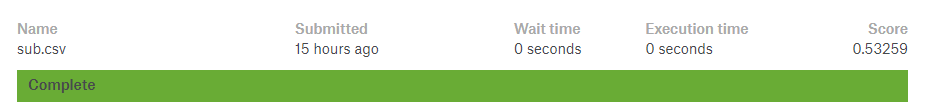


Figure 5 Kaggle sentiment analysis on movie reviews [11].

# Conclusion

From the experiments done with four different methods, raw preprocessed data had more stability through out the tests. But the combination of raw processed data with SL features and negation words had a constant decline of accuracy after every test. It can be concluded that Bigrams had slightly better accuracy than other methods. For the future work, different combination of these method can be tested. The negation words file was quite incomplete, it was made personally due to lack of availability of this kind of corpus in negation words online, therefore, it could be the reason negation words performed poorly. On the other hand, from observing the confusion matrix, the negation words method had done poorly in finding the positive comments as it was expected and learned from [8]. It was quite disappointing to see sentiment lexicons method done poorly in the tests which it was expected to do very well in the tests. It can be suspected that sentiment lexicon corpus achieved might not be very suitable about movie reviews. Moreover, the algorithm written for sentiment lexicon method’s scoring system might be incomplete. A future work can be dedicated in testing and implementing a superior sentiment lexicon algorithm.

##### References

[1]"Sentiment Analysis: How Does It Work? Why Should We Use It?", *Brandwatch*, 2019. [Online]. Available: https://www.brandwatch.com/blog/understanding-sentiment-analysis/. [Accessed: 22- Apr- 2019].

[2]"Natural Language Toolkit — NLTK 3.4.1 documentation", *Nltk.org*, 2019. [Online]. Available: https://www.nltk.org/. [Accessed: 22- Apr- 2019].

[3]Y. Mejova, "Sentiment Analysis: An Overview", 2009. [Accessed 15 April 2019].

[4] Kudo, T. and Matsumoto, Y. A boosting algorithm for classification of semistructured text. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2004.

[5]V. A. and S. Sonawane, "Sentiment Analysis of Twitter Data: A Survey of Techniques", *International Journal of Computer Applications*, vol. 139, no. 11, pp. 5-15, 2016. Available: 10.5120/ijca2016908625.

[6]M. Mäntylä, D. Graziotin and M. Kuutila, "The evolution of sentiment analysis—A review of research topics, venues, and top cited papers", *Computer Science Review*, vol. 27, pp. 16-32, 2018. Available: 10.1016/j.cosrev.2017.10.002.

[7]A. Agarwal, B. Xie, I. Vovsha, O. Rambow and R. Passonneau, "Sentiment Analysis of Twitter Data", 2019. [Accessed 17 April 2019].

[8] Ghorbel, Hatem & Jacot, David.. Further Experiments in Sentiment Analysis of French Movie Reviews, 2011. 86. 19-28. 10.1007/978-3-642-18029-3\_3.

[9]"6. Learning to Classify Text", *Nltk.org*, 2019. [Online]. Available: https://www.nltk.org/book/ch06.html. [Accessed: 22- Apr- 2019].

[10]"View Subjclueslen1-HLTEMNLP05.tff Sentiment-research - Scrapmaker.com", *Scrapmaker.com*, 2019. [Online]. Available: https://www.scrapmaker.com/view/sentiment-research/subjclueslen1-HLTEMNLP05.tff. [Accessed: 22- Apr- 2019].

[11]"Sentiment Analysis on Movie Reviews | Kaggle", *Kaggle.com*, 2019. [Online]. Available: https://www.kaggle.com/c/sentiment-analysis-on-movie-reviews/. [Accessed: 22- Apr- 2019].