SEMANTIC ANALYSIS FOR TEXT DATASET FROM AMAZON REVIEWS

Smita Suresh Daniel[1] , Dr.Ani Thomas[2],Riya Sharma[3],,Niyati Motghare[3]'Ashutosh Gupta[3]
[1] Associate Professor, Department of Comp. Sc. , St. Thomas College, Bhilai
[2] Prof &HOD, Department of Information Technology, BIT- Durg, C.G, India
[3] Student, B.E. (I.T), Bhilai Institute of Technology, Durg, C.G, India
ani.thomas@bitdurg.ac.in

ABSTRACT :

Sentiment analysis is a study of human sentiments over certain entities. The customer often uses reviews of e-commerce sites to analyse positive and negative sentiments regarding any product. It helps in decision making criteria's to evaluate the effectiveness and utility of any product. There are several features of a product but, if reviews of about important features are to be analysed, then it becomes easier for the owner to take a decision. Henceforth, a system is necessary that extracts featured reviews using sentimental analysis.

The overall analysis is based on the weights of positive and negative remarks found in more than one review about a common product. The reviews are obtained in unstructured text formats from the comments of the common man, which requires a lot of noise detection and pre-processing activities. To analyse text, the semantic processing of text is done with the help of natural language processing facilities available in python. Sentiment Analysis is the task of analyzing all this data, retrieving opinions about these products and services and classifying them as positive or negative, in other words good or bad. The key parts of any review of any product are the numeric rating and the textual description provided along with this product. In our project we will take into consideration both these vectors for product reviews to conclusively decide on a classifier that is best suited to analysis of product reviews. We have gathered reviews and based on the features that best describe the sentiment for each review, we have created a feature set of 1000 features, and with this limited set we will determine the accuracy of the classifier to give the best result on review type.

Keywords: Sentiment Analysis, Amazon Reviews, Text Classification, Feature based, Bag of words approach

1. INTRODUCTION

Sentiment analysis is a process in which we study the sentiments of a human towards certain entities. It aims to obtain the feelings of writer which is expressed in positive or negative comments, questions, etc. by analysing large number of documents. We computationally identify, categorize and quantify the opinion expressed in given text, in order to determine the measure, writers view towards a particular product.

The feedback is very important for a business, because it will help to plan marketing strategies based on customer's reactions. So it is very important to understand their sentiments about specific feature, as overall sentiments of a product is a collective one and does not provide information about the goodness or badness of a particular feature. Our system aims to analyse the sentiments of the text about a particular product's feature. For example, if the manufacturer wants to analyse the feature of Kindle store , then all the reviews related to feature camera is extracted into a file and the sentiments are classified as positive or negative at sentence level first and then at document level .

2. THE PROPOSED SYSTEM

In sentimental analysis, quantification plays a major role, since we are interested in estimating sentiment not at the individual level, but at the aggregate level. It helps to make sentiments definable and measurable. In the given textual dataset of reviews, the problem is to categories the text into one particular sentiment polarity, positive, negative (or neutral). Here we can calculate the score of the sentiment at three levels, first sentence level, then document level and finally entity level.

There are various steps involved in the proposed system are pre-processing database , extract specific reviews , POS tagging , sentiment sentence extraction, calculate total number of positives and negatives in each sentence and then result interpretation.

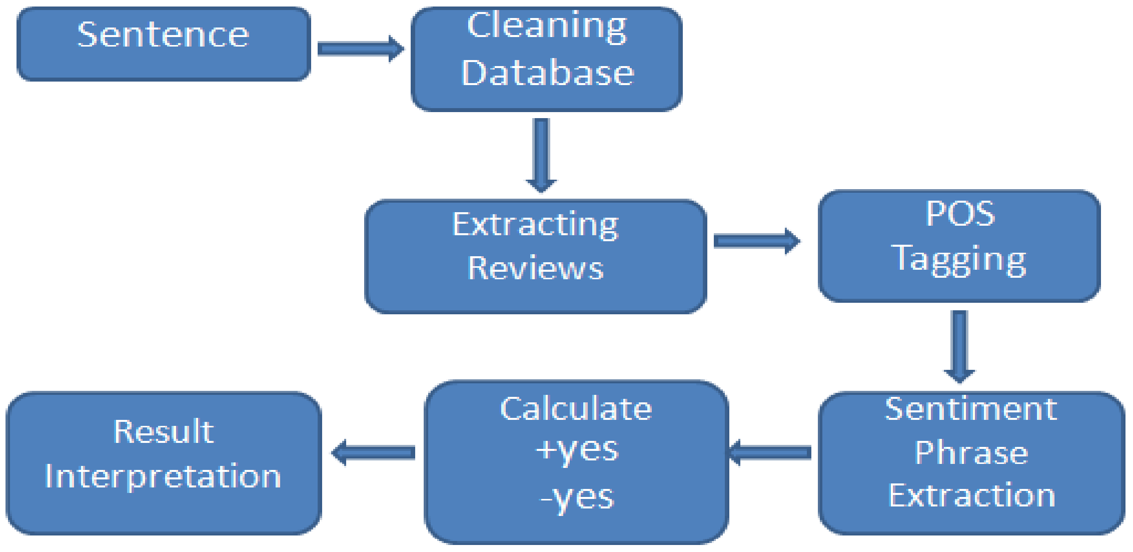

Fig-3.1 Activity diagram

3. APPROACH

We followed a simple approach to train the Naive Bayes classifier and run the tests on it.

The steps we followed are as discussed below,

1. **Gather data:** In this step, gathering all the data from amazon's product review page. The reviews we collected are stored along with a class label for each review, all the reviews, which get a rating of 3 or more stars were classified as positive and all ^{the} ratings which got a review score below 3 were classified as negative. By doing this we were able to create a training set, then, we have used this metric to evaluate our classifiers accuracy. In this way we collected over 500 reviews of a product and applied the class labels to the reviews based on the star rating provided by the reviewing users.

2. **Pre-process data:** In this step, we decided to process the data before we are able to extract the features. The various pre-processing steps we applied are,
   - **2.1. Tokenization:** We tokenized the reviews using tokenizer. The reason for this approach as opposed to splitting the sentences to tokenize them is clear when we compared the content of the reviews and the twitter. We found striking similarity.
   - **2.2. Remove Punctuations:** We remove the punctuations from the reviews, this is done so that we do not have any unwanted punctuations in our resultant feature-set.
   - **2.3. Remove Stop-words:** This is a basic requirement, so that we can focus on words that are actually relevant to the document instead of, determiners, prepositions and coordinating conjunctions, which can appear a number of times in any given training set, if not removed. Removal of stopwords is crucial to supervised learning. While words like “the” and “to” occur quite frequently, almost all words (>99%) occur less than four times per 1,000 words:
   - **2.4. Spell-check:** We perform spell-check on the documents(reviews) to ensure that the feature-set being generated has relevant words and not commonly misspelled words, apart from this, spell-check allows for accurate frequency calculation, which is crucial when the basis of the feature-set generation is frequency distribution over the set of processed documents.

3. **Feature-set generation (Using Bag-of-words method):**

   In NLP and information retrieval, bag-of-words is used as a simplified representation. Here, a text is represented as a bag of its words, disregarding the word order and the grammar associated with the text. To generate the feature-set we have considered using, the term frequency technique which uses frequency distribution method for generating the feature-set. After removal of stop-words, this method gives a feature set that appears to be very similar to a good feature set.

4. Text Classification using Naive Bayes Classifier

It is a classification technique based on Bayes' theorem with an assumption of independence among predictors. It is a simple technique for constructing classifiers using statistics and probability: It assigns class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. Naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable.

Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:

$$P(c|x) = P(c|x)\,P(c) \;/\; P(x)$$

- $P(c|x)$ is the posterior probability of *class* (c, *target*) given *predictor* (x, *attributes*).
- $P(c)$ is the prior probability of *class*.
- $P(x|c)$ is the likelihood which is the probability of *predictor* given *class*.
- $P(x)$ is the prior probability of *predictor*.

Each class contains distinctive words required to make a decision. and these words are found exclusively in texts associated with the given class. But it is challenging about how to treat words that occur in both groups of texts but they do so with different rates.

We focused on 2 possible sentiment classifications: *positive* and *negative*.

Our dataset with reviews were categorized into *pos* and *neg* categories, and a trainable classifiers. We used the NaiveBayes Classifier as a baseline.

5. Counting Word Frequencies

Word counts help us to know about which feature is more talked about by the reviewers .One issue with simple counts is that some words like "*the*" will appear many times and their large counts will not be very meaningful in the encoded vectors.A simple way of identifying words unique to a given class is to calculate the average rate of word used across a texts for of a group

An alternative is to calculate word frequencies, and by far the most popular method is called TF-IDF. This is an acronym than stands for "*Term Frequency – Inverse Document*" Frequency which are the components of the resulting scores assigned to each word.

- **Term Frequency**: This summarizes how often a given word appears within a document.

- **Inverse Document Frequency**: This downscales words that appear a lot across documents.

Without going into the maths, TF-IDF are word frequency scores that try to highlight words that are more interesting, e.g. frequent in a document but not across documents. Calculating the inverse document frequencies removes all unimportant words from the list and a more compact vocabulary with relevant words are available which help in improving accuracy,

6 Creating Bag of Words for Feature Extraction

We cannot work with text directly when using machine learning algorithms, so we need to convert the text to numbers. This can be done by assigning each word a unique number. Then any document we see can be encoded as a fixed-length vector with the length of the vocabulary of known words. The value in each position in the vector could be filled with a count or frequency of each word in the encoded document.

In order to perform classification of documents, we access each review as an "*input*" and a class label is the "*output*" for our predictive algorithm. The Algorithm take vectors of numbers as input, therefore we need to convert documents to fixed-length vectors of numbers.Our classifiers are simple dictionaries mapping a *feature name* to a *feature value*. For text, we used a simplified bag of words model which is dictionary in python where every word is feature name with a value. Here's the feature extraction method:

Tokenize the collection of text documents and build a vocabulary of known words, but also to encode new documents using that vocabulary.An encoded vector is returned with a length of the entire vocabulary and an integer count for the number of times each word appeared in the document.Because these vectors will contain a lot of zeros, a sparse vector will be created .. Python provides an efficient way of handling sparse vectors

6. Algorithm to Train a Classifier and Test its Accuracy

1. Accept documents of positive reviews
2. Accept documents of negative reviews
3. Tokenize and build vocabulary list after pre-processing for positive and negative features
4. Transform them to a vector
5. Train on ¾ of positive and negative dataset and use ¼ of them as test set
6. Encode the documents
7. Provide Trained vector as parameter to the NaiveBayes Classifier
8. Classify Test Data and print most frequently used words

9. Find Accuracy of the Classifier based on the above words

Here we see that the encoded vector is a sparse matrix. .On executing this algorithm the array version of the encoded sparse vector shows the occurrence of the each word in the Test data.vocabulary and its classification as positive and negative .

Once a classifier with good accuracy is built then any review documents can be applied for classification ..

8. **Sentiment Quantification**

A Sentiment word or a Phrase consisting of a positive (negative) wordsare collected as many times it occurs in a review. Based on this Sentiment categorization is done. A Positive word gets score of +1 else if found in the list of negative word a score of -1 is provided thus categorizing the text into one specific sentiment polarity, positive or negative Finally, the sentiment score information for positive and negative words can be calculated statistically for are view at the sentence level and then at document level. Finally the quantification targets to find the measure of how much people like or dislike the particular feature from their opinions. Summarization of sentiments is generated as charts finally.

REFERENCES

[1]. Aashutosh Bhatt and, Ankit Patel. and Harsh Cheda ., "Amazon Review Classification and Sentiment Analysis ", IJCSIT , 2015, Vol 6(6).

[2]. Minqing Hu and Bing Liu, “Mining and summarizing customer reviews”, KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, 2004

[3]. Bo Pang1 and Lillian Lee2 “Opinion Mining and Sentiment Analysis

[4]F. Sebastiani, “Machine Learning in Automated Text Categorization”, ACM 2002

[5]A. Dasgupta, P. Drineas, B. Harb, “Feature Selection Methods for Text Classification”, KDD’07, ACM, 2007.

[6]http://en.wikipedia.org/wiki/Naive_Bayes_classifier

[7]Richa, S., N. Shweta and J. Rekha, 2014. Opinion Mining Of Movie Reviews At Document Level.International Journal on Information Theory (IJIT),3(3).

[8]Devang Jhaveri, Aunsh Chaudhari and Lakshmi Kurup, 2015. Twitter Sentiment Analysis on E commerce Websites in India, International Journal of Computer Applications, (0975-8887): 127(18).

[9]P.Kalaivani, “Sentiment Classification of Movie Reviews by supervised machine learning approaches” et.al,Indian Journal of Computer Science and Engineering (IJCSE) ISSN: 09765166 Vol. 4 No.4 Aug-Sep 2013

[10]Lina L. Dhande and Dr. Prof. Girish K. Patnaik,"Analyzing Sentiment of Movie Review Data using Naive Bayes Neural Classifier", IJETTCS, Volume 3, Issue 4 July-August 2014,ISSN 2278-6856.