CUSTOMER SENTIMENT ANALYSIS BY TWEET MINING: UNIGRAM DEPENDENCY APPROACH

KIRAN DEY

Department of Computer Science & Engineering,
Maulana Abul Kalam Azad University of Technology, Salt Lake City
Kolkata, West Bengal 700064, India
kirandey93@gmail.com

SAHELI MAJUMDAR

Department of Computer Science & Engineering,
Maulana Abul Kalam Azad University of Technology, Salt Lake City
Kolkata, West Bengal 700064, India
sahelimajumdar1993@gmail.com

Abstract

The contribution of the following report is a classification algorithm developed, in order to analyze unstructured customer sentiment for any product or service. Unstructured data targeted in this work are the public tweets in twitter. The companies can judge their respective products by analyzing the sentiments of the customers, for any negative sentiment they can improve their product or service so that the customers get satisfied. The algorithm considered for the above mentioned purpose includes the concept of classification algorithm that maps the input data to specific categories on the basis of training data, and a brief explanation on unigram and bigram approaches along with their concerned limitations. Lastly, a histogram (graphical representation) is plotted accounting positive and negative sentiments. The technology used here is Python 3.4.

Keywords: Sentiment Analysis unigram dependency; training data; machine learning; tweets.

1. Introduction

The focus of development in the field of Information Technology has, in the recent years, been largely concentrated to producing, maintaining, manipulating and accessing data on a global scale retrieved from microblogging websites which are evolving nowadays to become a source of varied kind of information. This is due to nature of microblogs on which people post real time messages about their opinions on a variety of topics, discuss current issues, complain, and express positive or negative sentiment for products they use in daily life. Thereby using sentiment analysis, we can determine the attitude of a speaker or writer. The rise of social media such as blogs and social networks has fueled interest in sentiment analysis. In this paper, we look at one such popular microblog called Twitter and build a model for classifying “tweets” into positive, negative and neutral sentiment. In this project of **Customer Sentiment Analysis**, we utilize the methodologies of Sentiment Analysis, which includes collecting datasets of sentiments from social networking websites and blogs (however, initially we are starting with twitter), training the classifier with a list of positive and negative sentiments and then running the classifier algorithm on our collected dataset (tweets) and then displaying the resultant output in a graphical manner (in this case, it is done using a histogram).

The rest of the paper is organized as follows. In section 2, we give details about the data. In section 3, we discuss classification of data by machine learning. In section 4 we present original Naive-Bayes' Classifier. In section 6 we give details of our unigram dependency classification based approach. In section 7 we present implementation of our unigram dependency approach along with outputs. We conclude in section 8.

2. Data Description

Unstructured data (or **unstructured information**) refers to information that either does not have a pre-defined data model or is not organized in a pre-defined manner. Unstructured information is typically text-heavy, but may contain data such as dates, numbers, and facts as well. This results in irregularities and ambiguities that make it difficult to understand using traditional programs as compared to data stored in fielded form in databases or annotated (semantically tagged) in documents.

By the term ‘**unstructured**’, it is meant:

- Structure, while not formally defined, can still be implied.

- Data with some form of structure may still be characterized as unstructured if its structure is not helpful for the processing task at hand.
- Unstructured information might have some structure (semi-structured) or even be highly structured but in ways that are unanticipated or unannounced.

In this paper, we restrict our discussion to only one type of unstructured data *i.e.* tweets. Twitter is a social networking and microblogging service that allows users to post real time messages, called tweets. Due to the nature of this microblogging service (quick and short messages), people use acronyms, make spelling mistakes, use emoticons and other characters that express special meanings.

3. Data Classification by Machine Learning

Machine Learning is a branch of Computer Science that is concerned with designing systems that can learn from the provided input. Usually the systems are designed to use this learned knowledge to better process similar input in the future. Machine learning can be considered as a subfield of Artificial Intelligence. The machine learning approach applicable to sentiment analysis mostly belongs to supervised classification in general and text classification techniques in particular. Thus, it is also called supervised learning.

One of the most common machine learning techniques is **classification** (actually, **statistical classification**). More precisely it is a supervised statistical classification. In a machine learning based classification, two sets of documents are required: training and a test set. A training set is used by an automatic classifier to learn the differentiating characteristics of documents, and a test set is used to validate the performance of the automatic classifier. Supervised because the system needs to be first trained using already classified training data as opposed to an unsupervised system where such training is not done. In machine learning, **classification** is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known. An algorithm that implements classification is known as a **classifier**.

4. Naive-Bayes' Classifier

Naive Bayes' classifier is a simple supervised model for classification. It is a probabilistic classifier based on applying Bayes' theorem with strong independence assumptions. This is the simplest form of Bayesian Network, in which all attributes are independent given the value of the class variable. This is called conditional independence. It assumes each feature is conditional independent to other features given the class. A Naive Bayes' classifier is a technique that applies to a certain class of problems, namely those that phrased as associating an object with a discrete category. The Bayes' rule is described as

$$\gamma(\alpha \mid \beta) = \frac{\gamma(\alpha)*\gamma(\beta \mid \alpha)}{\gamma(\beta)} \qquad (1)$$

α: Specific class

β: Document wants to classify

$\gamma(\alpha)$ and $\gamma(\beta)$: Prior probabilities

$\gamma(\alpha \mid \beta)$ and $\gamma(\beta \mid \alpha)$: Posterior probabilities

The value of class α might be positive or negative. Document may be a review of particular movie, product or any other type of service. The multinomial model of Naive Bayes captures word frequency information in documents. The Maximum Likelihood Estimate (MLE) is simply the relative frequency and corresponds to the most likely value of each parameter given the training data. For the prior probability this estimate is shown in Eq. (2).

$$\gamma(\alpha) = \frac{N_c}{N} \qquad (2)$$

N_c: The number of documents in class α

N: Total number of documents

In Multinomial model, assumes attribute values are independent of each other given for the particular class $\gamma(\beta \mid \alpha) = \gamma(\omega_1 \ldots \omega_{nd} \mid \alpha)$. In the multinomial model, a document is an ordered sequence of word events, drawn from the same vocabulary V. Assume that the lengths of documents are independent of class. Thus, each document β_i is drawn from a multinomial distribution of words with as many independent trials as the length of β_i . This yields the familiar bag-of-words (BOW) representation for documents. The BOW model is commonly used in methods of document classification, where the (frequency of) occurrence of each word is used as a feature for training a classifier. From given documents, a predefined list of words appearing in the documents (the dictionary) and computes the vectors of frequencies of the words as appear in the documents. The angle

between the two vectors is a widely used measure of closeness between documents. Let W be the dictionary - the set of all terms (words) that occur at least once in a collection of documents D. The BOW representation of document d_n is a vector of weights (ω_{1n}...ω | W | n). In the simplest case, the weights ω_{in} € 1,-1and denote the presence or absence of a particular term in a document. More commonly, ω_{in} represent the frequency of the i_{th} term in the nth document, resulting in the term frequency representation. The transformation of a document set D into the BOW representation enables the transformed set to be viewed as a matrix, where rows represent document vectors, and columns are terms. A unigram feature marks the presence or absence of a single word within a text. Estimate the conditional probability γ (ω | α) as the relative frequency of term ω in documents belonging to class α including multiple occurrences of a term in a document.

$$\gamma(\omega|\alpha) = \frac{\text{count}(\omega,\alpha) + 1}{\text{count}(\alpha) + |V|} \quad (3)$$

The problem with the MLE estimate is that it is zero for a term-class combination that did not occur in the training data. To eliminate zero probability problem add-one or Laplace smoothing is used, this simply adds one to each count. Add-one smoothing can be interpreted as a uniform prior (each term occurs once for each class) that is then updated as evidence from the training data comes in. Then, the probability of a document given its class is simply the multinomial distribution presents in Eq. (3).

5. Unigram Dependency Classification Algorithm

5.1 Modifying naive-bayes'

Uniform Dependency Classification Algorithm is obtained by performing several modifications in the original naive bayes' algorithm. “Naive Bayes' Classification” is termed such because it makes the assumption that each word is statistically independent from each other word. In language, however, that assumption doesn’t hold true. For instance, “Bull's” word is followed by "eye" word commonly than most other words (“Bull's horn” just is not anything we see every day), so it’s clear that in reality, words are somehow dependent of one another. Still, it’s been shown that the “naive” assumption is pretty good for the purposes of document classification. As a result, we tried to shift from the basic assumption that words are independent. Moreover, this Bayes' classifier can be easily extended to use bigrams just by tokenizing a document two words at a time instead of one for the purpose of considering dependency between consecutive words. For instance, a sentence "This book is interesting" can be tokenized as:

“This book” “book is” “is interesting”. Now the classifier knows the difference between “is interesting” and “not interesting”!

5.2 Entropy

Entropy is a measure of the number of possible “states” a system can be in. However using bigrams, it is found the entropy gets increased. If for instance, 5,000 different words in conversational English are used and a unigram-based Bayes' classifier is built for them, then only 5,000 words will be needed to get stored. But if instead bigrams are used, it is potentially needed to store data for up to 25,000,000 different word pairs (that’s maximum possible value; though in practice around 13,000 unique word pairs will be encountered).But storage space and computational complexity is not the major concern. The real problem is the lack of training data. Using bigrams and introducing more entropy into our system means that each word pair is going to be relatively rare.The bigram approach is useful only when there is a huge training corpus. So, in order to decrease the entropy, we decided to use a unigram-based classifier.

5.3 Unigram dependency considering negations

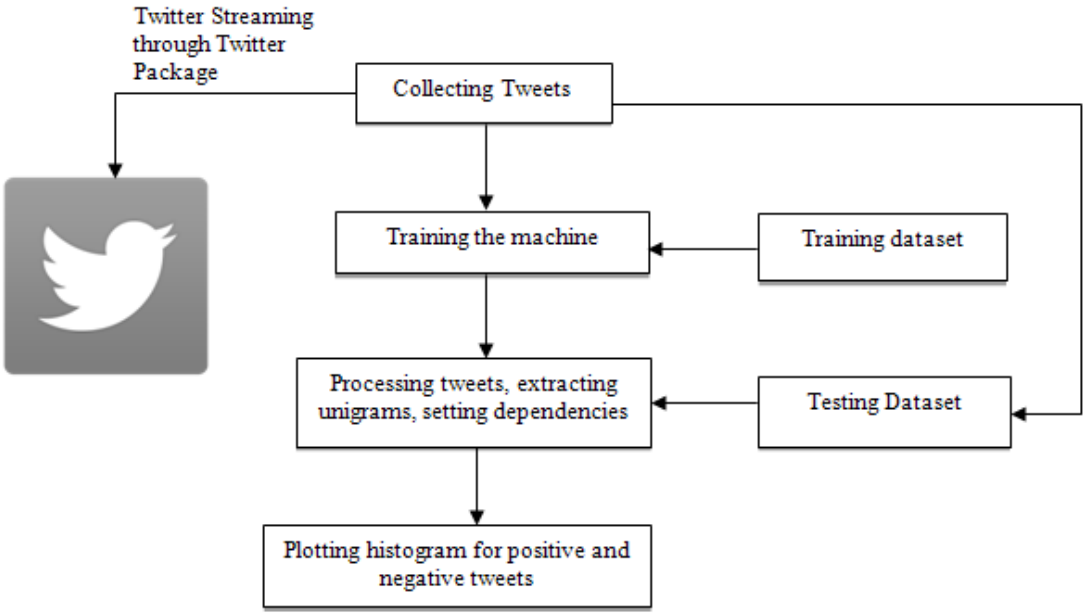
Thereafter another way is found out to combat entropy, and that too by using unigrams. Words are almost always stemmed as part of the tokenization process. Stemming puts word variations like “great”, “greatly”, “greatest”, and “greater” all into one bucket, therefore effectively decreasing the entropy and giving more data around the concept of “great”.

Dealing with negations (like “not great”) is another important step in sentiment analysis. A negative word can affect the tone of all the words around it. A better way is found out in order to deal with negations other than using bigrams that may lead to increase in entropy. Whenever a negation (like “not”, “never”, “no”, etc) is encountered, an exclamation point is added to the beginning of the word immediately before and after the negation term. For instance, the tokenizer takes the sentence “This book is not interesting at all” turns into

“This book! is not! interesting at all”, and the token “!interesting” gets stored in our Unigram Dependency classifier as having appeared in a negative review. “interesting” appears in positive reviews, and its negation, “!interesting” appears in negative ones.

6. Implementation

6.1. Flowchart



6.2 Fetching tweets

In order to fetch tweets available in twitter for a particular service provider company, the steps needed to be followed are given below:

Step 1:

At first we log in to apps.twitter.com.

Step 2:

We note the values of different keys such as consumerkey, consumersecret, OAUTH keys etc.

Step 3:

Import the tweepy package in python code. Tweepy package provides necessary methods for fetching the tweets.

Step 4:

Implement a cursor to fetch the tweets from twitter based on some particular topic. For e.g. #hotstar, collects tweets on hotstar app.

6.3 Training machine

In Unigram Dependency classifier, the machine must be initially trained with a sample training data. The steps of training the machine are shown below:

Step 1:

We take a training dataset in which the texts and the polarities (0 for negative, 4 for positive) are already provided.

Step 2:

We define a method for retrieving the texts and polarities of the training dataset which is in CSV (comma separated value) format.

Step 3:

Words with polarity 4 are taken as positive words and added to a list, a list of positive words.

Step 4:

Word with polarity 0 are taken as negative words and added to another list, a list of negative words.

Step 5:

If there are any negative words such as not, never, none etc. then the words before and after those negative words are flagged with an exclamation mark (!) and the flagged words are added to the negative list.

Step 6:

The total length of positive and negative sentiment lists are noted.

6.4 Analyzing testing dataset

After training the machine, our next job is to analyze the tweets representing a brand or a product collected from twitter is shown below:

Step 1:

We open the sample testing data set containing tweets in read mode in python.

Step 2:

Then tweets are extracted one by one from the file and analyzed.

Step 3:

If there are any negative words such as not, never, none etc. in the data set, then the words before and after those negative words are flagged with an exclamation mark and the dataset updated with this new conversion, thereby the independent assumption of Naïve Bayes' get modified.

Step 4:

Now the values of positive and negative lists are calculated after adding their corresponding sentiment values.

Step 5:

Finally the classification of the tweets are done.

6.5 Output

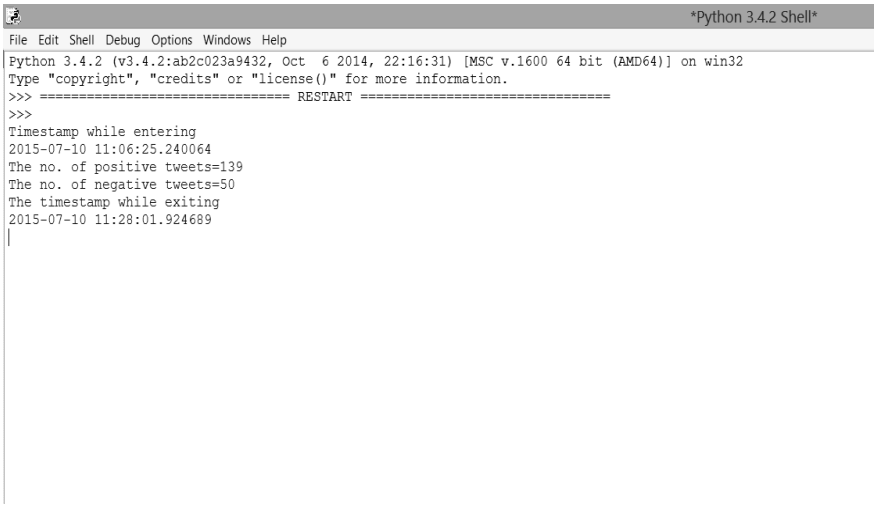


Fig. 1. Screenshot depicting number of positive and negative tweets of #hotstar between 20th May, 2015 and 21st May, 2015

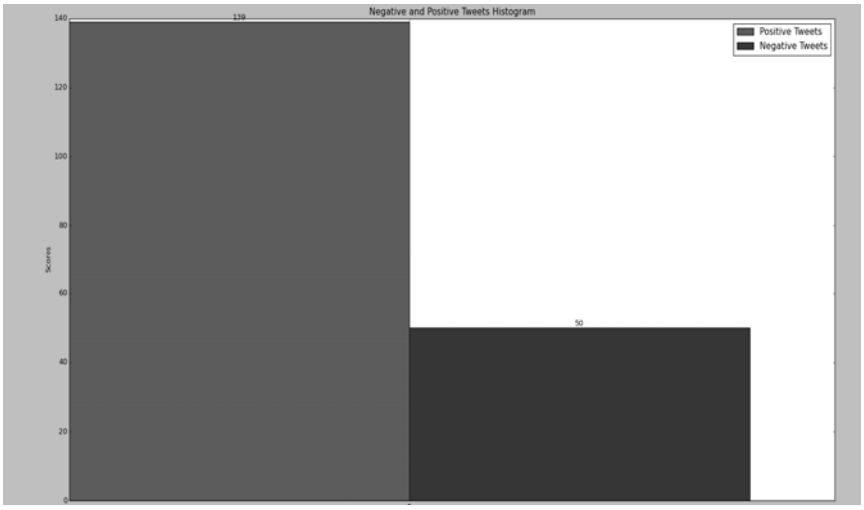


Fig. 2. Histogram with the X-axis showing the number of positive and negative tweets of #Hotstar between 20th May, 2015 and 21st May, 2015 and Y-axis plotting the scores.

7. Conclusion

This paper work aims at improving the experience of the customers for products and services by analyzing their sentiments. The companies can get a decent idea of what the customers want, what they are happy with and what they find lacking, what the customers are enjoying and what not. In this project we have intended to do a bonafide work in this context. This project work completely focuses on building a classification algorithm and obtain better results than those using Naïve Bayes. The effort has been not to treat the words independently as in Naïve Bayes. Although this work aims in customer's sentiment analysis, it can be used in other fields as well. The entire work is based on the training dataset. For instance, the same algorithm can be modified to check for spams in an email. This effort can only be stated as the beginning, this work can be further enhanced and its future scope is large and varied. In this present world of digitalization where every day we produce millions of bytes of data, analyzing them is the challenge which has gained a lot of importance.

8. Acknowledgements

We would sincerely like to thank all the Professors from the Computer Science Department of Institute of Engineering and Management Kolkata, for their continued patience and support during the course of this ongoing research, and also for providing the inspiration to make this paper a reality. We would also like to express our gratitude to our respective family members, families and peers who have kept supporting us and always had faith in our work.

9. References

[1] Gentile, C. "A new approximate maximal margin classification algorithm". Journal of Machine Learning Research, 2:213–242, 2001.
[2] Lewis, D Naive (Bayes) at forty: "The independence assumption in information retrieval". ECML Conference, 1998, 4-15.
[3] Witten, I.H. et al. "Data Mining Tools and Techniques practical Machine Learning".
[4] Domingos, P; Pazzani, M. "On the optimality of the simple Bayesian classifier under zero-one loss, Machine Learning 29 (2–3) (1997) 103–130".
[5] Younis, M.G.E. "Sentiment Analysis and Text Mining for Social Media Microblogs using Open Source Tools: An Empirical Study". International Journal of Computer Applications(0975-8887) Volume 112-No. 5, February 2015.
[6] Sharma, T.C.; Jain, M. "WEKA Approach for Comparative Study of Classification Algorithm". International Journal of Advanced Research in Computer and Communication Engineering. Vol 2, Issue 4, April 2013.
[7] Freund, Y; Schapire, R.E. "Large margin classification using the perceptron algorithm". Machine Learning, 37(3):277–296, 1999.
[8] Kim, Y.H. et al. "Text filtering by boosting naive Bayes classifiers". ACM SIGIR Conference:168-175, 2000.