International Journal of Scientific Engineering and Technology (ISSN : 2277-1581)
Volume No.2, Issue No.6, pp : 589-594 1 June 2013

Opinion Mining For Text Classification

Anand Mahendran, Anjali Duraiswamy, Amulya Reddy, Clayton Gonsalves
School of Computing Science & Engineering, VIT University, Vellore, Tamilnadu, India.
manand@vit.ac.in, anjalidurai.13@gmail.com, amul531@gmail.com, clay_ton1990@hotmail.com

Abstract- There are several micro-blogging services available today such as Twitter, Tumblr, Jisko, Thimbl, Wordpress, blog, etc. These micro-bloggers communicate in the form of posts containing videos, images, text, links, etc. This large amount of raw data may overwhelm the users. One solution to this problem is the classification of this raw data. As these raw data such as "tweets" do not account for sufficient word occurrences, we approach traditional methods of classification such as- Bag of Words, Naïve Bayes classifier, frequency distribution. However, to address this problem, we propose to use a small set of domain-specific features extracted from the author's profile and text. The proposed approach effectively classifies the text to a predefined set of generic classes such as Positive, Negative, Neutral.

Keywords: internet, mining, **blogging**

I. INTRODUCTION

In order for users to browse through tremendous amounts of raw data, text-classification provides an optimal solution. Text-classification is a supervised data mining technique that assigns a label to a set of unlabelled input objects [7]. Data mining is the process of skimming through large data sets and databases to generate patterns among these sets of pre-processed data [6]. Once the data has been mined and the required labelled patterns generated, we use supervised learning, in order to extract functions from the available sets of labelled patterns. The algorithms that have been generated in order to perform text-classification used are called classification algorithms. For example, Bayesian classifier, neural networks classifier, etc. The classification algorithm that we have implemented, uses Naïve Bayes algorithm, classifies the text into two categories- positive text and negative text. Thus, when a text is entered as input to the classification algorithm, the output generated would indicate whether the input text is a positive or negative review. This is done using python and NLTK software [1].

Before the commercialization of the internet, the medium relied upon word of mouth with regard to suggestions about movies, books electronics etc.

Now-a-days, it is possible for us to access reviews and opinions of a large number of people online. Through the web, we are able to access the reviews about various products, books, electronics, etc. from a vast pool of people. However, these people have first-hand knowledge or experience regarding that product and wish to share these reviews. Today, more and more strangers are making their opinion count, by sharing their reviews available to strangers via the Internet through forums like twitter, Face Book, Wordpress.com, Blogspot.com, etc.

II. LITERATURE SURVEY

In contrast to recent Internet applications, which have focused on detecting the polarity of the text, our text classifier helps users distinguish between positive and negative reviews thus assisting the user with opinion mining. This could be very useful for web applications like twitter, where the user has to face large chunks of raw data. Upon being classified, the user can skim through the processed data at ease.

A. *EXISTING TECHNIQUES*

Opinion mining and sentiment analysis are a very active area of research and currently a lot of work is under way in this domain of natural language processing. In the same field, we have applications such as sentiment140, Twitratr that classifies tweets from twitter into positive, negative and neutral tweets.Lexalytics.com is also a classification forum where the input is fed by the user and not derived from twitter. These are some of the active areas of research in classification of text [2][3].

The challenges to be faced during the building of a complete trained classifier have been listed as follows:

- First and foremost, it must be confirmed whether the material is subjective in case of a general search engine.

- The next task would be to partition the reviews needed from the unnecessary text. This is much easier for us to perform when the reviews are presented in a direct manner. For example- flipkart.com, android play store, etc. However, when reviews are in the form of blogs, it could provide to be a tricky task to break down the review.

- Once we have extracted the review part of the text, we are faced with the task of sentiment analysis of the text and understanding the sentiment of the review as positive, negative or neutral. This is where the classifiers are used and different techniques are incorporated. The motive is to attain accuracy of classification which could prove to be difficult.

- Once the text has been classified, the system now has to be able to produce the result of the classification to the users in a well-reasoned manner.

There are some learning techniques which are widely used for the training purpose of the classifier. The classifier is the heart and soul of the opinion mining system. It is the module of the system which filters and analyzes text and groups them according to polarity.

B. PROPOSED ALGORITHM-NAÏVE BAYES CLASSIFIER

The Naive Bayes Classifier is a very popular algorithm due to its simplicity, computational efficiency and its surprisingly good performance for real-world problems. For instance most email clients such as Mozilla Thunderbird or Microsoft Outlook use naive bayes classifiers for filtering out spam emails

The "Naive" attribute comes from the fact that the model assumes that all features are fully independent, which in real problems they almost never are. In spite of serious violations of the basic assumptions and the simplistic design of the classifier it turns out that they are very well suited for problems involving normal distributions, which are very common in real-world problems [10].

The bayes classification is a supervised learning technique as well as a statistical technique for classification. This method assumes an underlying probabilistic model and it allows for the capture of uncertainty about the model, in a principled way by determining probabilities of the outcomes.

Naïve Bayes classifier formula [9]:

The Bayes Naïve Classifier selects the most likely classification Vnb given the attribute values $a_1, a_2, \ldots, a_n$.

This results in:

$$Vnb=\mathrm{argmax}_{v_j \in V} P(v_j) \prod P(a_i|v_j)$$

We generally estimate $P(a_i \mid v_j) = \dfrac{n_e + mp}{n + m}$

Where:

n= the number of training examples for which $v = v_j$

$n_{e\,=}$ number of examples for which $v = v_j$ and $a=a_i$

p = a priori estimate for $P(a_i \mid v_j)$

m = the equivalent sample size

In simple terms, a naive Bayes classifier assumes that the presence of a particular feature of a class is unrelated to the presence of any other feature. For example, a person may be considered to be a male if he is tall, has short hair and a strong build. Even if these features depend on each other or upon the existence of the other features, a naive Bayes classifier considers all of these properties to independently contribute to the probability that the person is a male.

This classifier has the following advantages [10]:-

- It is easy to train and implement. Versatile in nature. Process of learning by classifier is fast compared to other learning methods such as logistic regression and support vector machines.
- Suppose, if the naïve bayes conditional independence holds then the classifier will converge much quicker than the other training methods.
- Even if the conditional independence does not hold, the naïve bayes classifier still performs well.
- The classifier has the option of probability distribution, thus giving us the accuracy of the prediction. If the accuracy is not up to scratch then

subsequently the prediction can be discarded or ignored.

- We can compensate for class imbalance wherein one or more than one instances occur very rarely (1 in 1000). This class imbalance can lead to wrong and incorrect predictions. This is called degenerate solution. This problem can be overcome by using a balanced training set.
- Works with exceptional accuracy when the training data set is large.
- The training time complexity of this classifier is linear to the number of training data and the space complexity is also linear to the number of features, thus it makes this learning technique both time and storage efficient.

Uses of Naive Bayes classification [11]:

- Naive Bayes text classification
  The Bayesian classification is used as a probabilistic learning method (Naive Bayes text classification). Naive Bayes classifiers are among the most successful known algorithms for learning to classify text documents.
- Spam filtering
  Spam filtering is the best known use of Naive Bayesian text classification. It makes use of a Naive Bayes classifier to identify spam e-mail. Bayesian spam filtering has become a popular mechanism to distinguish illegitimate spam email from legitimate email. Many modern mail clients implement Bayesian spam filtering.
- Hybrid Recommender System Using Naive Bayes Classifier and Collaborative Filtering
  Recommender Systems apply machine learning and data mining techniques for filtering unseen information and can predict whether a user could like a given resource.

C. ALGORITTHM

Assume that we have two classes c1 = male, and c2 = female.

We have a person whose sex we do not know, say "Surya". Classifying Surya as male or female is equivalent to asking is it more probable that Surya is male or female, i.e which is greater p(male| Surya) or p(female| Surya).

Let us consider the following example:

S. No	Hair	Height	Eye Colour	Sex
1	Long	Short	Black	F
2	Short	Tall	Brown	M
3	Medium	Medium	Blue	F
4	Long	Tall	Black	F
5	Short	Short	Black	M
6	Medium	Tall	Black	M

i. Learning phase

Eye colour	M	F
Black	2\3	2\3
Brown	1\3	0\3
Blue	0\3	1\3

Hair	M	F
Long	0\3	2\3
Short	2\3	0\3
Medium	1\3	1\3

Height	M	F
Tall	2\3	1\3
Medium	0\3	1\3
Short	1\3	1\3

P(Sex=M)=1\2

P(Sex=F)=1\2

Test Phase:

Given a new instance x,

x=(hair=medium, height=tall, eye colour=black)

The Look up table value is given as follows:

Male:

P(hair=medium|Sex=M)=1\3

P(height=tall|Sex=M)=2\3

P(eye colour=black|Sex=M)=2/3

P(Sex=M)=1\2

Female:

P(hair=medium|Sex=F)=1\3

P(height=tall|Sex=F)=1\3

P(eye colour=black|Sex=F)=2\3

P(Sex=F)=1\2

P(Male|x): [P(hair|medium)P(height|tall)P(eye colour|black)]P(sex=male)=0.0740

P(Female|x): [P(hair|medium)P(height|tall)P(eye

colourblack)]P(sex=female)

=0.0370

Since P(Sex=M) > P(Sex=F),

Hence our prediction that instance x is a male.

Computational complexity of the above mentioned method is given below [12]:

- One of the fastest learning methods
- O(ndc)= time complexity
- O(dc) =space complexity
- where c=number of classes, n=number of instances and d=number of attributes.
- no hidden constants (number of iterations, etc.)

D. *BAG OF WORDS MODEL*

The **bag-of-words(BoW) model** is a representation method used in natural language processing and information retrieval where, a text is represented as an unordered collection of words, disregarding grammar and even word order. The bag-of-words 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 [8].

As mentioned above, the purpose of using bag-of-words in our implementation is for two reasons. First being, to extract words easily from a text/document/sentence. And second, to get the weightage of the words/features in the document, which in turn would help us to identify the text category-aiding in classification. This also enhances the performance of the classifier when combined with the naïve-bayes classification, as now the text is first filtered for word/feature weightage using BoW and then trains classifier using these words.

IMPLEMENTATION STEPS

The two main modules that we are dealing with are-

- Word/feature extraction
- Classification

Process 1

This the basic and the first step for any text classification. This can be either static data-fed in manually by the user or dynamic-collected from the web in real time. For simplicity and explanation, we will be considering the static data. The dynamic data poses the problem of dealing with information coming in continuously within very short periods of time and in very high quantity. Handling and classifying the information instantaneously is being researched [4].

There are three kinds of data needed. First the input data for training the classifier. Second, input data for testing the classifier. And the final is the user input data which is to be classified.

The text is manually classified with corresponding sentiment as either positive or negative. This can also be a file containing documents of text labelled under each sentiment.

TASKS

1.1) Enter positive text.

a) [text]

b) [Sentiment]

c) e.g.- [I love my car][positive]

1.2) Enter negative text.

a) [Text]

b) [Sentiment]

c) e.g.-[I don’t like rain] [negative]

Process 2

Once the positive and negative text has been entered, the next is the extraction of words. The document is filtered to remove all the stop words such as is, and, the, etc. With respect to our implementation, the BoW is used here to extract words/features. The words extracted are weighed and their order in the text does not matter at all. Once the feature set is got, the frequency distribution function is used to create the training set.

ACTIVITIES:

1) Combine text into one file.

TASKS

1.1) Remove words with length >3.

1.2) Combine all words into one file.

1.3) Get word features.

2) Get frequency distribution for words.

TASKS

2.1) Count each word occurrence with corresponding sentiment.

2.2) Create Frequency distribution.

2.3) Using frequency distribution create training set.

Process 3

The most essential part of the sentiment classification tool- the classifier. After the removal of stop words, application of BoW and the creation of the training set, the classifier is the last step. Using the nltk library, we get the Naïve Bayes classifier. This classifier is trained by giving it the training set that was created. Once the classifier has been trained, it is tested using the test data set and once the needed accuracy is reached, it can be implemented via an interface [5].

ACTIVITY

Train Bayes Naive Classifier.

TASKS:

1) Using achieved frequency distribution, train classifier.

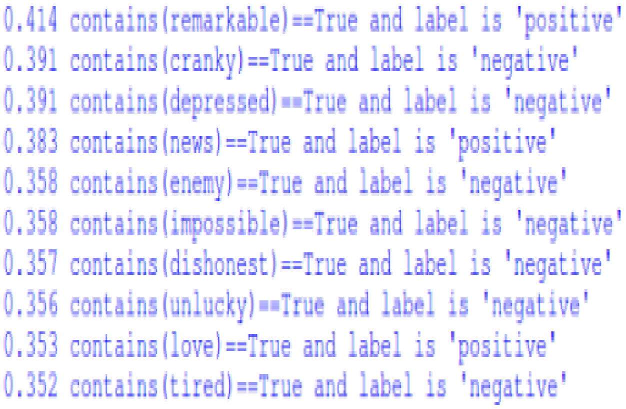
2) Test the classifier for accuracy using the test data.

III. RESULTS

In this section, we explain about the various results obtained.

A. *SCREEN SHOTS*

Features from maxent classifier

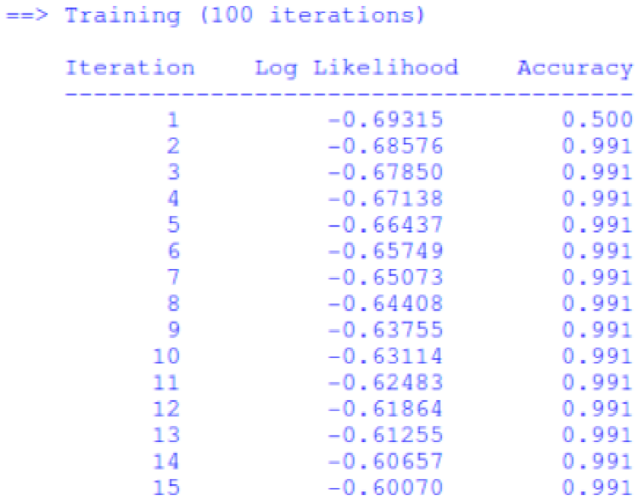


```
0.414 contains(remarkable)==True and label is 'positive'
0.391 contains(cranky)==True and label is 'negative'
0.391 contains(depressed)==True and label is 'negative'
0.383 contains(news)==True and label is 'positive'
0.358 contains(enemy)==True and label is 'negative'
0.358 contains(impossible)==True and label is 'negative'
0.357 contains(dishonest)==True and label is 'negative'
0.356 contains(unlucky)==True and label is 'negative'
0.353 contains(love)==True and label is 'positive'
0.352 contains(tired)==True and label is 'negative'
```

Fig.1. Features from maxent classifier

The features and their labels as output after the classifier has been trained using the created training set. For every feature, its frequency distribution is calculated and the higher appearance is given as the labelled sentiment.

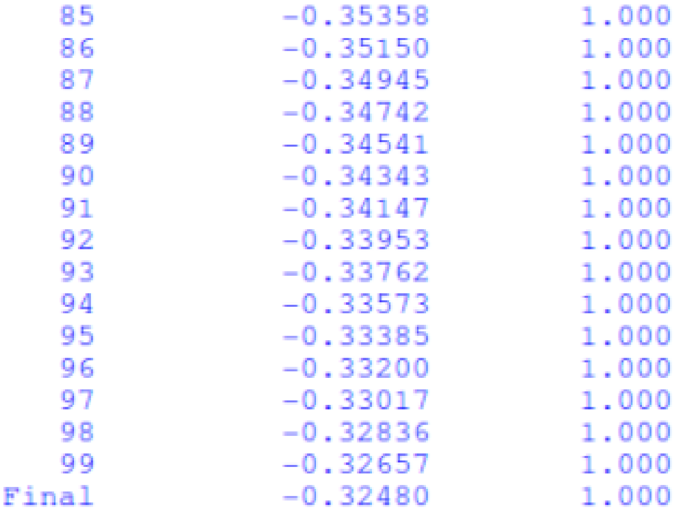
Maxent training sets



```
==> Training (100 iterations)

      Iteration    Log Likelihood    Accuracy
      ---------------------------------------
             1          -0.69315        0.500
             2          -0.68576        0.991
             3          -0.67850        0.991
             4          -0.67138        0.991
             5          -0.66437        0.991
             6          -0.65749        0.991
             7          -0.65073        0.991
             8          -0.64408        0.991
             9          -0.63755        0.991
            10          -0.63114        0.991
            11          -0.62483        0.991
            12          -0.61864        0.991
            13          -0.61255        0.991
            14          -0.60657        0.991
            15          -0.60070        0.991
```

Fig.2. Maxent training sets

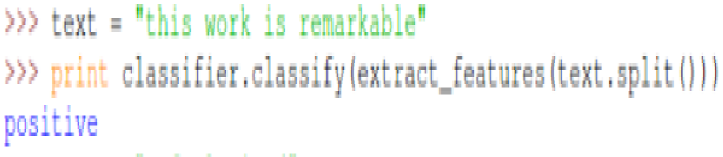


```
    85          -0.35358          1.000
    86          -0.35150          1.000
    87          -0.34945          1.000
    88          -0.34742          1.000
    89          -0.34541          1.000
    90          -0.34343          1.000
    91          -0.34147          1.000
    92          -0.33953          1.000
    93          -0.33762          1.000
    94          -0.33573          1.000
    95          -0.33385          1.000
    96          -0.33200          1.000
    97          -0.33017          1.000
    98          -0.32836          1.000
    99          -0.32657          1.000
 Final          -0.32480          1.000
```

Fig.3. Maxent training sets

The above two images show the iterations and the details of each run while the classifier is being trained. Once this step is over, it displays the list of features/words with the sentiment given to them.

Maxent-postive input

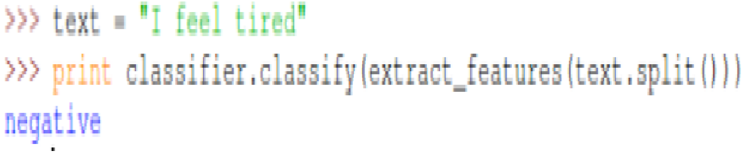


```
>>> text = "this work is remarkable"
>>> print classifier.classify(extract_features(text.split()))
positive
```

Fig.4. Maxent-positive input

After the classifier has been trained and tested, the user gives in an input text. Here, we have provided a text to be classified and the classifier has classified it as positive.

Maxent-negative input

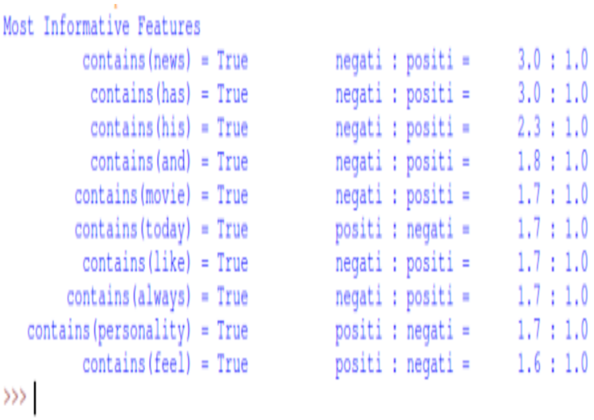


```
>>> text = "I feel tired"
>>> print classifier.classify(extract_features(text.split()))
negative
```

Fig.5. Maxent-negative input

Similarly, if a negative sentence is given, the classifier accurately classifies the text as a negative sentiment.

Features from naive bayes classifier

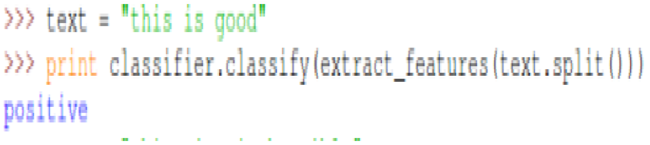


```
Most Informative Features
         contains(news) = True          negati : positi =      3.0 : 1.0
          contains(has) = True          negati : positi =      3.0 : 1.0
          contains(his) = True          negati : positi =      2.3 : 1.0
          contains(and) = True          negati : positi =      1.8 : 1.0
        contains(movie) = True          negati : positi =      1.7 : 1.0
        contains(today) = True          positi : negati =      1.7 : 1.0
         contains(like) = True          negati : positi =      1.7 : 1.0
       contains(always) = True          negati : positi =      1.7 : 1.0
  contains(personality) = True          positi : negati =      1.7 : 1.0
         contains(feel) = True          positi : negati =      1.6 : 1.0
>>> |
```

Fig.6. Features from naive bayes classifier

From the given training set, this is a sample of the few features that have been extracted and displayed.

Bayes-positive input

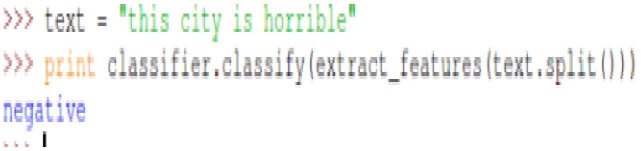


```
>>> text = "this is good"
>>> print classifier.classify(extract_features(text.split()))
positive
```

Fig.7. Bayes-positive input

Once the classifier has been trained, and a sentence is given by the user for identification, it is more or less classified accurately as positive.

Bayes-negative input



```
>>> text = "this city is horrible"
>>> print classifier.classify(extract_features(text.split()))
negative
```

Fig.8. Bayes-negative input

Same as above, once the text has been submitted for identification, the classifier breaks down the sentence and accordingly gives it its sentiment.

IV. CONCLUSION

In this paper, we have classified the text based on opinion mining. With the help of sentiment analysis, we are able to collect features from text, extract them, classify them and provide opinions/sentiment about the text/data/documents to the users with the help of bayes naïve classifier or maximum entropy classifier. Improving the efficient classification of text with other techniques is left as a future work.

V. REFERENCES

[1] B. Sriram, “short text classification in twitter to improve information filtering”, may 2010, Unpublished

[2] B.Pang, L. Lee, “Opinion Mining and sentiment analysis,” now, vol. 1, no.1-2, pp 1-7 march 2008

[3] B.Pang, L. Lee, “Opinion Mining and sentiment analysis,” now, vol. 2, no.1-2, pp 10-11 march 2008

[4] D. Osimo, F. Murredu, “Research Challenge on Opinion Mining and Sentiment Analysis, Unpublished

[5] R. Probowo, M. Thelwall , “ Sentiment Analysis: A Combined Approach,” SCIT, University Of Wolverhamptom

[6] Data Mining [Online]. Available: http://en.wikipedia.org/wiki/Data_mining

[7] Text Classification [Online]. Available: http://en.wikipedia.org/wiki/Text_classification

[8] Bag of words model [Online]. Available: http://en.wikipedia.org/wiki/Bag_of_words

[9] Kevin P. Murphy, “Naive bayes classifier”, October 2006, Unpublished

[10]http://blog.peltarion.com/2006/07/10/classifier-showdown/

[11]http://software.ucv.ro/~cmihaescu/ro/teaching/AIR/docs/Lab4-NaiveBayes.pdf

[12]http://www.inf.ed.ac.uk/teaching/courses/iaml/slides/naive-2x2.pdf