Twitter Sentiment Analysis

Ajay Singh 12056

CS365A : Artificial Intellegence

Abstract

Twitter is a micro-blogging website where people are allowed to write status updates limited by 140 characters. This project deals with analyzing sentiments behind the tweets, whether they are about a person, product, movie, organization, or people's everyday lives. Classifiers based on supervised machine learning algorithms are used to classify the sentiment present in a tweet. Naive Bayes Classifier and Maximum Entropy Classifier were able to classify the tweets with an accuracy of around 75 %. Focus is not only on classifying the tweets, but also on making this task faster and more accurate by removing the parts of the tweets not contributing to the sentiment analysis, as described later in the preprocessing and filtering of tweets.

1 Motivation

Twitter Sentiment Analysis was thoroughly dealt by Alec Go, Richa Bhayani and Lei Huang, Computer Science graduate students of Stanford University. They used various classifiers, including Naive Bayes, Maximum Entropy as well as Support Vector Machines to classify the tweets. The feature extractors used by them were both unigrams and bigrams combined. Parts of speech tag was used because same word may have different meaning depending on its usage. The data-set used by them was huge, comprising 1.6 million tweets divided equally into positive and negative classes.[1] I have used much a smaller dataset, consisting of 40,000 tweets with 20,000 positive and 20,000 negative tweets. This was done to cut down the running time of the program. Also, only unigrams have been adopted as feature extractors to reduce the size of the feature vector. Bigrams along with unigrams make the feature vector too complicated and bigrams alone make the feature space very sparse. The classifiers used by me are Naive Bayes and Maximum Entropy, which are a lot similar in implementation (not so much in theory) and will be described later.

2 Approach

2.1 Feature Reduction[2]

Usernames are quite often present in tweets, starting with symbol. All such instances of usernames are replaced by generic word AT_USER

Links and URLs are not needed in feature vectors. Hence, all links are replaced by generic word URL. These links are identified by regex "http://" and "www".
Hashtags are used to denote a trending topic and can be used to view tweets with a common hashtag. Every occurence of #word is replaced by word.
Additional blank spaces are removed from the original tweet.

2.2 Filtering tweet words[2]

Two or more repetitions of a letter are removed and replaced by two same letters. E.g. hunnngry, huuungry and hungggry are changed to hunngry, huungry and hunggry respectively.
Stop Words are removed from the tweets as they do not contribute to the sentiment analysis in any way. The list of stop words used by me can be found here.[2]
Punctuations like comma, full stop, exclamation are removed.
Words starting with digit(s) are removed.
Repeated words are not used as they will affect the classifier by biasing it towards repeated instances.

2.3 Feature Vector

For every tweet after feature reduction and filtering, we obtain words from the original tweet and this is called the feature vector for the concerned tweet.

2.4 Extract Features Method[2]

This method takes the feature vector of a tweet as an input and checks **if the words present in the feature vector of the given tweet** are present in the features list made by joining all the feature vectors of the tweets.

2.5 Training Set

It comprises the output of *Extract Features Method* for all the feature vectors of the tweets as well as the sentiment in the tweet

Example to illustrate the approach

Original Tweet: @kenburbary You'll love your Kindle2. I've had mine for a few months and never looked back. The new big one is huge! No need for remorse! :)
After Feature Reduction: AT_USER you'll love your kindle2. i've had mine for a few months and never looked back. the new big one is huge! no need for remorse! :)
After Filtering: love kindle2 mine months looked
Feature Vector: ['love', 'kindle2', 'mine', 'months', 'looked']
After Extract Features Method:
{
.....
'contains(kindle2)': True,

```
.....
'contains(looked)': True,
.....
'contains(love)': True,
.....
'contains(mine)': True,
.....
'contains(months)': True,
.....
}
```

3 Classifiers

3.1 Naive Bayes

In machine learning, Naive Bayes Classifier is based on applying Bayes' Theorem with strong (naive) independence assumptions among the features.[3]
Given a vector of features $\mathbf{x} = (x_1, x_2, ..., x_n)$,
$p(C_k|x_1, x_2, ..., x_n)$ where there are k possible classes.
Applying Bayes Theorem we get:

$$p(C_k|\mathbf{x}) = \frac{p(\mathbf{x}|C_k)p(C_k)}{p(\mathbf{x})} = \frac{p(x_1.x_2, ..., x_n|C_k)p(C_k)}{p(x_1, x_2, ..., x_n)}$$

We need to use only the numerator as the denominator remains constant for all the classes. Assumming that each feature is conditionally independent of every other feature (hence the name *Naive Bayes*),

$$\begin{aligned} p(C_k|\mathbf{x}) &\propto p(C_k)p(x_1|C_k)p(x_2|C_k)....p(x_n|C_k) \\ &\propto p(C_k)\prod_{i=1}^{n} p(x_i|C_k) \end{aligned}$$

This is a naive Bayes Probability model. The naive Bayes Classifier combines this model with a decision rule, which picks up the hypothesis with maximum probability. In simple words, we pick the class which has maximum value for $p(C_k)\prod_{i=1}^{n} p(x_i|C_k)$

3.2 Maximum Entropy

It is a probabilistic classifier which belongs to the class of exponential models. It does not assume that the features are conditionally independent of each other. It is based on the Principle of Maximum Entropy and from all the models that fit our training data, selects the one which has the largest entropy.[4] Summarizing the training data in terms of its empirical probability distribution.
$\tilde{p}(x, y) \equiv \frac{1}{N} \times$number of times (x,y) occurs in the sample

$$f_j(x, y) = \begin{cases} 1 & \text{if } y = c_i \text{and x contains } w_k \\ 0 & otherwise \end{cases}$$

The above indicator function is called as feature. The expected value of feature f_j with respect to the empirical distribution:

$$\tilde{p}(f_j) \equiv \sum_{x,y} \tilde{p}(x,y) f_j(x,y)$$

The expected value of feature f_j with respect to the model $p(y|x)$ is equal to:

$$p(f_j) \equiv \sum_{x,y} \tilde{p}(x) p(y|x) f_j(x,y)$$

According to the principle of Maximum Entropy, we should select the model that is as close as possible to uniform.

$$p^* = \arg\max_{p \in C} \left(-\tilde{p}(x) p(y|x) \log p(y|x)\right)$$

Using Lagrange multipliers, the probability given a document x to be classified as y is equal to:

$$p^*(y|x) = \frac{\exp\left(\sum_i \lambda_i f_i(x,y)\right)}{\sum_y \exp\left(\sum_i \lambda_i f_i(x,y)\right)}$$

In order to classify a new document we use the “maximum a posteriori” decision rule and select the category with the highest probability. MaxEnt handles conditional overlap better than naive Bayes classifier.[5]

4 Result

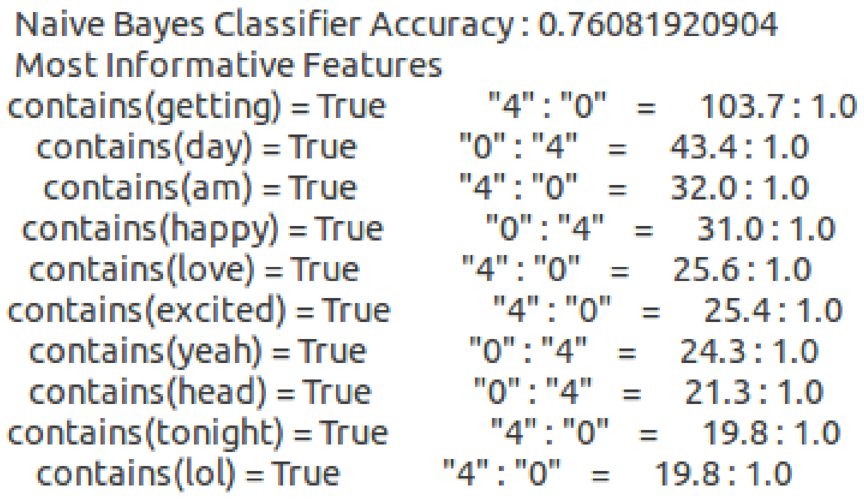
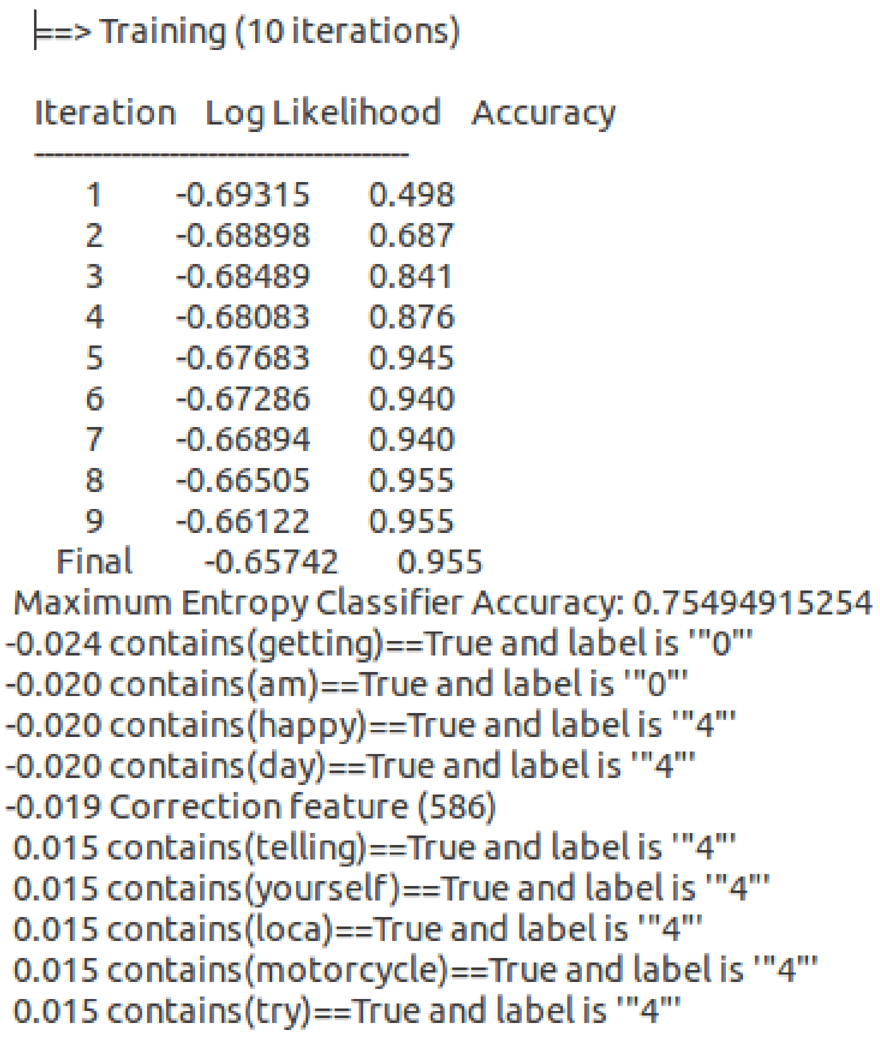

```
Naive Bayes Classifier Accuracy : 0.76081920904
Most Informative Features
contains(getting) = True        "4" : "0"   =    103.7 : 1.0
   contains(day) = True         "0" : "4"   =     43.4 : 1.0
    contains(am) = True         "4" : "0"   =     32.0 : 1.0
 contains(happy) = True         "0" : "4"   =     31.0 : 1.0
  contains(love) = True         "4" : "0"   =     25.6 : 1.0
contains(excited) = True        "4" : "0"   =     25.4 : 1.0
  contains(yeah) = True         "0" : "4"   =     24.3 : 1.0
  contains(head) = True         "0" : "4"   =     21.3 : 1.0
contains(tonight) = True        "4" : "0"   =     19.8 : 1.0
   contains(lol) = True         "4" : "0"   =     19.8 : 1.0
```

Figure 1: Naive Bayes Classifier Results



```
==> Training (10 iterations)

Iteration   Log Likelihood   Accuracy
---------------------------------------
     1        -0.69315        0.498
     2        -0.68898        0.687
     3        -0.68489        0.841
     4        -0.68083        0.876
     5        -0.67683        0.945
     6        -0.67286        0.940
     7        -0.66894        0.940
     8        -0.66505        0.955
     9        -0.66122        0.955
   Final      -0.65742        0.955
 Maximum Entropy Classifier Accuracy: 0.75494915254
-0.024 contains(getting)==True and label is '"0"'
-0.020 contains(am)==True and label is '"0"'
-0.020 contains(happy)==True and label is '"4"'
-0.020 contains(day)==True and label is '"4"'
-0.019 Correction feature (586)
 0.015 contains(telling)==True and label is '"4"'
 0.015 contains(yourself)==True and label is '"4"'
 0.015 contains(loca)==True and label is '"4"'
 0.015 contains(motorcycle)==True and label is '"4"'
 0.015 contains(try)==True and label is '"4"'
```

Figure 2: Maximum Entropy Classifier Results

Classifier	Naive Bayes	Maximum Entropy
Accuracy	76.0%	75.4%

5 Conclusion

Naive Bayes Classifier gives relatively good results in spite of the strong assumptions made about the independence among the features. Practically, it may outperform Maximum Entropy Classifier (as it does in this case), even though MaxEnt Classifier takes into account the feature overlapping. Increasing the size of the dataset in steps revealed that accuracy increases with the increase in the size of the training dataset. Using unigrams as feature extractor makes the task simpler, however it fails to equip the classifier with the ability to identify negations. Using bigrams alone makes feature space too sparse. A better method would be to include both unigrams as well as bigrams.

6 Future Work

Neutral Tweets need to be classified in order to increase the utility of a classifier. Many tweets lack any particular sentiment and are more focused on direct and unbiased statements about facts or events.
Parts Of Speech tag can be utilized to interpret emotions in a better way. E.g. 'over' as a verb conveys a negative emotion, but as a noun it is neutral (over in cricket).[5]
Large Data Set can be used if better computation facility is available. As it took more than two hours on my system to train the current dataset and obtain the output, bigger dataset could not be used even though it was available.
Usage of bigrams need to be included as classifiers handling negations always produce unexpected results.

7 Dataset and Code

The dataset used by me can be obtained from here[1]. It consists of the training data in regular csv format with 1.6 million tweets with equal number of positive and negative tweets. The fields "tweet_id", "date_time", "query" and "user" have been removed. This leaves "sentiment" (0 for negative and 4 for positive) and the actual "tweet" without any emoticons.
First 300,000 tweets from each category (sentiment) were extracted and placed in 15 different files, each with 20,000 tweets. Thus, we can choose to train our classifier from any two of these 30 files, one with positive sentiments and the other with negative sentiments. If better computing facilities are available 2 or more files can be joined to produce a larger training data set.
The test set used by me was also downloaded from the above link, but the already mentioned four fields were removed, as well the rows with neutral sentiment (marked as 2). This resulted in test set with size 354. The final (after above alterations) training and test data set can be downloaded from here

The base code used by me can be found here. The actual code (written in python using nltk) is here.

References

[1] Twitter Sentiment Analysis, `http://help.sentiment140.com/for-students/`

[2] How To Build Twitter Sentiment Analyzer, `http://ravikiranj.net/posts/2012/code/how-build-twitter-sentiment-analyzer/`

[3] Naive Bayes Classifier, `http://en.wikipedia.org/wiki/Naive_Bayes_classifier`.

[4] Maximum Entropy Classifier, `http://blog.datumbox.com/machine-learning-tutorial-the-max-entropy-text-classifier/`

[5] Twitter Sentiment Classification under Distant Supervision, Alec Go, Richa Bhayani, Lei Huang, Stanford University http://cs.stanford.edu/people/alecmgo/papers/TwitterDistantSupervision09.pdf

[6] witter Sentiment Analysis using Python and NLTK, http://www.laurentluce.com/posts/twitter-sentiment-analysis-using-python-and-nltk/

[7] witter Sentiment Corpus http://www.sananalytics.com/lab/twitter-sentiment/