Visual Tweets Sentimental Analysis

Lilong Jiang (jiang.573)

April 24, 2014

1 Introduction

In this project, I build a visual online tweets sentimental analysis tool, which allows users to do the sentimental analysis for a certain topic. I also implement two classifiers and compare both classifiers over the performance and accuracy.

2 Dataset

We get our datset from the Sentiment140[1]. The training dataset in this dataset is collected by the Twitter API, whose content contains the emoticons :) and :(. The test dataset in this dataset is manually annotated.

3 Preprocessing

There are several challenges for the tweets processing: firstly, the tweets are very short, limited as 140 characters. Secondly, the tweets are very sloppy and it contains a lot of misspellings, slangs, abbreviations, emoticons, urls, etc. So the preprocessing step is very important for the later classification. We preprocess all the tweets as follows:
(1) Remove usernames and urls.
(2) I use an emoticon dictionary [1], in which each emoticon is labeled as emotion(positive or negative). The emoticons in the tweets are replaced by happy or sad.
(3) Replace all sequences of repeated characters by three characters, for example, convert goooood to goood.
(4) Expand abbreviations with a dictionary[2]. For example, lol is converted into laughing out loud.
(5) NLTK[3] is used to tokenize the tweet, remove the stop words, punctuations, and do the lemmatization.
(6) PyEnchant[4] is used to check whether the token is a correct English word.

[1] `http://www.sentiment140.com/`
[2] `http://www.noslang.com/`
[3] `http://www.nltk.org/`
[4] `http://pythonhosted.org/pyenchant/`

Only correct English words are retained.

4 Classifiers

Two classifiers are implemented in this project. One is the naive Bayes classifier and the other is the SVM.

4.1 Naive Bayes Classifier

The Naive Bayes classifier is a simple probalistic classifier which is based on Bayes theorem. Each tweet will be classified into the class with the highest posterior probability.

$$C_{map} = argmax_{c \in C}(logP(c) + \sum_{1 \leq k \leq n_d} logP(t_k|c))$$

The conditional probability of a particular word given a class as the relative frequency of term t in documents belongs to class c in the multinomial naive bayes model while in the binarized multinomial naive bayes model, each term has the same frequency which is 1:

$$P(t|c) = \frac{T_{ct}+1}{\sum_{t \in V}(T_{ct}+1)} = \frac{T_{ct}+1}{\sum_{t \in V}(T_{ct})+B}$$

where B is the number of documents.
I use add-one smoothing to address the problem that a particular word doesn't appear in a particular class.

4.2 Feature Selection

For the naive bayes classifier, I implement two feature selection methods. One is the frequency-based feature selection. The other is based on the mutual information.

4.2.1 Frequency-based Feature Selection

All the adverb and adjective tokens are retained since it is more related to the polarity and for the verb and noun tokens, only tokens whose frequency is greater than 4 are retained.

4.2.2 Mutual Information

Mutual Information is used to measure how much a term t contributes to the class c.

$$I(U;C) = \frac{N_{11}}{N}\log_2\frac{NN_{11}}{N_1.N_{.1}} + \frac{N_{01}}{N}\log_2\frac{NN_{01}}{N_0.N_{.1}} + \frac{N_{10}}{N}\log_2\frac{NN_{10}}{N_1.N_{.0}} + \frac{N_{00}}{N}\log_2\frac{NN_{00}}{N_0.N_{.0}}$$

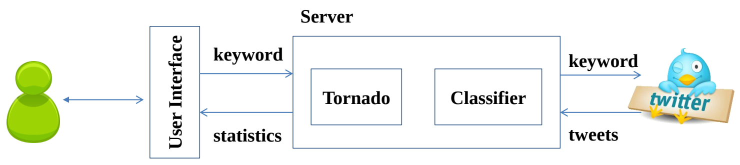


Figure 1: Architecture

where N_{10} is the number of documents that contain t and are not in class c. N_{11} is the number of documents that contain t and are in class c. N_{01} is the number of documents that don't contain t and are in class c. N_{00} is the number of documents that don't contain t and are not in class c. $N_{1.}$ is the number of documents that contain t. $N_{0.}$ is the number of documents that don't contain t. $N_{.0}$ is the number of documents which are not in class c. $N_{.1}$ is the number of documents in class c.
We use add-1 smoothing for each count to void the divided by 0.

4.3 SVM Classifier

Libsvm[5] is a library which implemented SVM algorithm. Libsvm is used to classify the tweets. I use C-SVM and set the parameter of the cost to 4.

4.4 Experiment

I compare these two classifier on the performance and accuracy. 359 tweets are used to test the classifier.

4.4.1 Parameter Sensitivity

Shrinking Size
I plot the accuracy of Bayes classifier with the changing of the shrinking ratio in the mutual information. The result is shown in Figure. As we can see

4.4.2 Performance

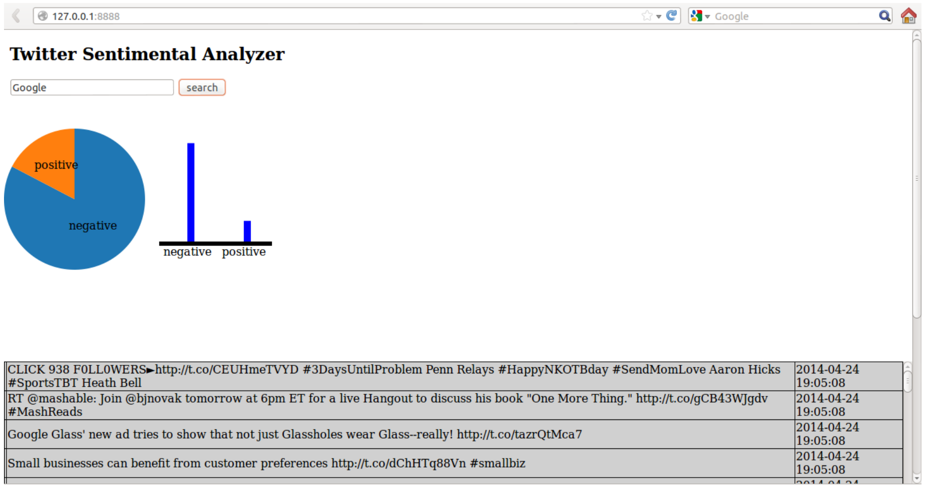
4.4.3 Accuracy

5 Online Sentimental Analysis

5.1 Architecture

In the front end, D3[6] is used to visualize the final result and Tornado[7] works as a web server. When users submit a topic, the server gets 500 tweets from

[5]`http://www.csie.ntu.edu.tw/ cjlin/libsvm/`
[6]`http://d3js.org/`
[7]`http://www.tornadoweb.org/`



CLICK 938 F0LL0WERS►http://t.co/CEUHmeTVYD #3DaysUntilProblem Penn Relays #HappyNKOTBday #SendMomLove Aaron Hicks #SportsTBT Heath Bell	2014-04-24 19:05:08
RT @mashable: Join @bjnovak tomorrow at 6pm ET for a live Hangout to discuss his book "One More Thing." http://t.co/gCB43WJgdv #MashReads	2014-04-24 19:05:08
Google Glass' new ad tries to show that not just Glassholes wear Glass--really! http://t.co/tazrQtMca7	2014-04-24 19:05:08
Small businesses can benefit from customer preferences http://t.co/dChHTq88Vn #smallbiz	2014-04-24 19:05:08

Figure 2: User Interface

twitter through twitter api and analyzes each tweet and returns the statistics information and tweets to the client. The user interface of this system is shown in Figure 2.

5.2 Data Collection

TwitterSearch[8] is a python-based interface which implements the Twitter Search API.

5.3 Online Analysis

We analyzes each tweet using the naive bayes classifier and show the final statistics information in the pie chart and bar chart. The tweets will be shown in the table, where the negative tweet will shown in light gray while the positive tweet will be shown in light blue.

[1]

References

[1] Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. Sentiment analysis of twitter data. In *Proceedings of the Workshop on Languages in Social Media*, pages 30–38. Association for Computational Linguistics, 2011.

[8] `http://twittersearch.readthedocs.org/en/latest//`