Tweet Classification Sentiment Analysis and Opinion Mining

Shanmathi Mayuram Krithivasan
Stuti Ohri

1. Abstract

We have examined the sentiment analysis on Twitter data. Sentiments analysis focuses on people's opinions, attitude, emotions. Different people have different say on the same topic. With the growing availability and popularity of opinion-rich resources such as online review sites and personal blogs, new opportunities and challenges arise as people now can, and do, actively use information technologies to seek out and understand the opinions of others. Thus we need to make a decision we often seek out the opinions of others. Sentiment analysis is system are required in almost all the sectors these days, as opinions play very vital role in human activities.

2.Introduction

The process of computationally identifying and categorizing opinions expressed in a piece of text, especially in order to determine whether the writer's attitude towards a particular topic, product, etc., is positive, negative, or neutral. The rise of social media such as blogs and social network has fueled interest in sentiment analysis. Generally speaking, sentiment analysis aims to determine the attitude of a speaker or a writer with respect to some topic or the overall contextual polarity of a document. Existing approaches to sentiment analysis can be grouped into four main categories: keyword spotting, lexical affinity, statistical methods, and concept-level techniques. Keyword spotting classifies text by affect categories based on the presence of unambiguous affect words such as happy, sad, afraid, and bored. Lexical affinity not only detects obvious affect words, it also assigns arbitrary words a probable affinity to particular emotions.

The project involves the implementation of sentiment and opinion mining

techniques of the tweets provided. The goal of the project is to build a classifier based on the training data provided, and thus predict the nature of the testing data.

The tweets are from the twitter pages of Barack Obama and Mitt Romney posted during the election. The tweets are classified as:

1.Positive : Tweets in the favor of the candidate.

2.Negative : Tweets against the candidate.

3.Neutral : Tweets unrelated to the candidates.

4.Mixed : Tweets used for the comparison. (Not included in the implementation).

We divided the project into the following stages:

1. Reading the data

2. Pre-processing

3. Stop words removal

4. Classification

5. Testing

The tweets of candidates is provided in the excel format, which is read from from the code. Once input is read from the file, the tweets needs to be preprocessed and then feature extraction is performed. The file was then converted to ARFF format, which is compatible with WEKA. This is then used to generate the Naive Bayes classifier, which is then used to predict the sentiments of the test data provided.

3.Technique

We used WEKA library for implementation of the project and the techniques used can be further categorized into the following

3.1 Pre-processing

The following pre-processing were performed :

1. Remove links
2. Remove unnecessary symbols and numbers
3. Name of candidates – Mitt Romney, Mitt, Romney –converted to Romney. Barak Obama, Barak, Obama –converted to Obama
4. Remove # tags that were irrelevant
5. Remove @ tags
6. Remove stop words
7. Emoticons as positive and negative
8. Identify informal identifiers such as all-caps and character-epetitions(“I LOVE this show”, “happyyyyy”)

9. Repeat characters replaced with single character
10. Convert to lowercase
11. Remove the stop-words, like is, are , beyond, etc.
12. Count the frequency of the words, and remove the words which occur to frequently or too rarely.

3.2 Feature Extraction

For tweet preprocessing, a POS Tagger is used. Feature Words is an array list that consists of the feature words. If the feature word is present in the tweet, the index corresponding to that feature word is given a value of 1. All other feature words that are not in the tweet have a value 0. There are totally 6800 feature words. This means that the feature vector is a point in which each axis represents the presence or the absence of the feature word corresponding to that axis. Hence Feature vector takes a sparse form. A Sparse Instance is used to represent a feature vector. Once feature extraction is done, training and testing can be performed.

3.3 Classifier

3.3.1 Classifier

A classifier was build with Naive Bayes algorithm , which used the training data. The training data was used to build this classifier.

After pre-processing, a .csv file is created for the tweets. A .ARFF file is created from the .CSV file. The .ARFF file is given as input to the classifier algorithm. Each individual tweet in the . ARFF file is split into n-grams tokens. We used 10 fold cross validation, to generate this model. The classifier is built using the training data. This classifier is used for classification of test data.

3.2.1 n-Grams

We tried to include n-grams in the project. This was done by using NGramTokenizer from the Weka library. We used max size of 2 and min size 1for the n-grams. The tweet string was converted to word vectors using StringToWordVector class and the n-gram tokenizer was implemented using setTokenizer() function of the StringToWordVector class.

3.2.2 Stemming

Stemming is a method for collapsing distinct word forms. This could help reduce the vocabulary size, thereby sharpening one's results, especially for small data sets.

We split each tweet into n-gram tokens. These tokens are processed and stemming is done. Porter stemming from the Snowball stemmer is used for stemming. Weka contains a wrapper class for the Snowball stemmer that contains the Porter stemmer and several other stemmers for different languages. This was implemented by using the setStemmer() method.

3.2.3 Positive and negative words

We used a predefined set of word list with positive and negative words classify the tweets. Each token was compared to the words, and the polarity of the words is then set correspondingly. The words if not present were marked as “No Word”

3.Evaluation

For Obama

Accuracy 51.6032 %

Positive class Precision: 0.457

Positive class recall: 0.446

Positive class F-score: 0.561

Negative class Precision: 0.443

Negative class recall: 0.491

Negative class F-score : 0.382

For Romney:

Accuracy: 60.1202 %

Positive class Precision: 0.523

Positive class recall: 0.472

Positive class F-score: 0.425

Negative class Precision: 0.328

Negative class recall: 0.557

Negative class F-score: 0.367

4.Conclusion

In our process of trying to classify the tweets, we added more features to pre-processing, feature extraction and implemented n-grams, stemmers and addition of positive and negative wordlist.

We generated the Naive Bayes model using the training data, for both Obama and Romney separately and the test data was then given to these models to predict the class labels and calculate the accuracy, F-score, precision and the recall . The feature extraction process, improved the values a lot and we added many features to improve the algorithm.