TWITTER SENTIMENT ANALYSIS

Rajath Ramakrishna and Varun Gokulnath

1. Introduction

a. Problem Statement

Given a collection of tweets, classify them into four classes namely: positive, negative, neutral and mixed. The tweets used here are pertaining to two US presidential candidates namely: Barack Obama and Mitt Romney. By classifying the tweets into the mentioned classes we would be capable of predicting the opinion of the public and get a sense of the outcome of the election.

b. Approach

Initially we have a number of tweets (approximately 7000 for each Obama and Romney) and their classes, in an Excel spreadsheet.
The steps we take for classifying the tweets are as follows:

i. Data Preprocessing
ii. Feature extraction
iii. Training
iv. Classification

Programming language used for implementation is **Python**. The **NLTK** libraries of Python provide a rich source of classification techniques which simplifies the task. We used the following modules of NLTK:

- `nltk.stem` for Lemmatization
- `nltk.tokenize` for tokenization
- `nltk.classify` for Naïve Bayes Classifier
- `nltk.metrics` for generating confusion matrix
- `nltk.corpus` for stopwords

2. Algorithm Description

a. Preprocessing

Preprocessing of tweets involves the following steps:

i. **Removal of hyperlinks**: Used regular expressions to remove links as they do not provide essential information for classification.
ii. **Removal of usernames**: These are the items in the tweet that begin with '@' character. They are removed because we do not consider opinion holders important for the classification of tweets. E.g. *@NickiMinaj*.

iii. **Removal of hash character in hashtags**: These are the items in the tweet that begin with '#' character. They represent the topic of the tweet which in our case tends to be important. Hence we drop the '#' character and preserve rest of the word. E.g. #voteForObama becomes voteForObama.
iv. **Split camel case words**: Camel casing means two or more words merged into one such that the first word starts with a lowercase and subsequent words start with a capital letter. E.g.' voteForObama' becomes 'vote For Obama'.
v. **Removal of annotations**: They do not contribute towards classification and are hence removed. E.g. '<a>' and '<e>'.
vi. **Removal of other special characters**: All punctuations are removed except the single quote. The single quote is utilized later on while expanding words like "don't" to "do not".
E.g. punctuations like !,?,;,% etc.
vii. **Removal of digits**: Digits are unimportant during classification and are removed. E.g. Ryan2012.
viii. **Stripping off white spaces**: Additional white spaces are stripped off from the tweet
ix. **Fixing repeated characters**: E.g. "Goooood" becomes "Good".
x. **Conversion to lower case**: This helps in maintaining uniformity.
xi. **Tokenizing the tweets**: The space separated tweets are tokenized to obtain the tweets as list of words.
xii. **Expanding abbreviated words**: Abbreviations are expanded to make them more meaningful. E.g. "lol" becomes "laugh out loud" and "ppl" becomes "people".
xiii. **Lemmatizing the tweet**: Each word undergoes lemmatization in order to obtain the root word. E.g. "Winning" becomes "win".
xiv. **Removal of stopwords**: These do not contribute towards the classification.
xv. **Removing duplicate words**: As the frequency of a word in a tweet does not affect the class of the tweet it shall be removed.

b. Features

The individual features extracted from each tweet are tuples of the form: (tweet, sentiment). For each tweet extracted from the training set, the words in the tweet were assigned to the corresponding sentiment, thus resulting in 4 lists of words – one for each sentiment. These features are fed to the `train` function to prepare the training model. The test set is later fed to the `classify` function to classify the tweets and generate the *Accuracy*, *Precision*, *Recall* and

F-score for each class/sentiment. The features used are ***unigrams*** thus ignoring the position of the words in the tweet. The tweet is instead considered as a "**bag of words**".

c. **Learning Algorithm**: Naïve Bayes Classifier.

3. Experimental Results

a. **Description of data**: The data considered for this project is a set of tweets about Obama and Romney and their corresponding sentiments. The set of tweets and their sentiments is provided in an excel file.

b. **Precision, Recall and F-Score for the positive and negative classes**:

OBAMA:

ACCURACY: **40.39%**

	Positive	**Negative**	**Neutral**	**Mixed**
Precision	55.47%	46.61%	48.52%	28.4%
Recall	26.97%	48.11%	28.92%	63.39%
F-Score	36.30%	47.35%	36.24%	39.22%

CONFUSION MATRIX

	Neg	**Neu**	**Pos**	**Mix**
Neg	**331**	93	38	226
Neu	173	**197**	67	244
Pos	105	74	**157**	246
Mix	101	42	21	**284**

ROMNEY:

ACCURACY: **47.66%**

	Positive	**Negative**	**Neutral**	**Mixed**
Precision	50.00%	56.18%	43.26%	37.94%
Recall	23.63%	63.43%	21.98%	63.88%
F-Score	32.09%	59.58%	29.15%	47.61%

CONFUSION MATRIX

	Neg	**Neu**	**Mix**	**Pos**
Neg	<609>	92	235	24
Neu	235	<122>	164	34
Mix	113	40	<329>	33
Pos	127	28	139	<91>

4. Conclusion

The classification was done on the provided test set with reasonable accuracy. We found that Naïve Bayes Classifier works well with the feature set that we generated from the tweets. In order to improve the results, we could have experimented with bigrams and collocations in feature sets.

5. Future Work

More preprocessing can be done on the given data like:

a. **POS-tagging**: in this technique, the parts of speech of each word in the tweet will be identified. One can decide which parts of speech are relevant for training/classification.
b. **Synsets**: Wordnet provides the facility to identify synonyms, hyponyms, hypernyms, etc. This would help in enriching the feature thereby improving the classification model.
c. **Bigram model**: Bigrams and collocations are important while classifying tweets as bigrams make more sense and provide more information about the sentiment of the tweet.
d. **Spellchecker**: correcting the spellings of words would help to a great extent during classification because misspelled words are meaningless to the classifier.

6. References

- http://nltk.org/ - for documentation about NLTK libraries
- Bing Liu. "Sentiment Analysis and pinion Mining" ,May 2012. eBook: ISBN 9781608458851
- http://streamhacker.com/2010/10/25/training-binary-text-classifiers-nltk-trainer/ - for info on text classification methods
- http://www.ravikiranj.net/drupal/201205/code/machine-learning/how-build-twitter-sentiment-analyzer - for info on tweet classification