TWITTER SENTIMENT ANALYSIS

Chirag Soni and Muttavarapu Sreeharsha

Introduction

Abstract:

Data Mining and Text mining has become popular since the early 2000's in the field of computer science and the concepts and algorithms attributed to this domain have been extensively used in various industries. This project deals with application of various data mining algorithms and text processing techniques on the data collected during 2012 US Presidential election.

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.

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 and SCIKIT - LEARN libraries of Python provide a rich source of classification techniques which simplifies the task.

We used the following modules of NLTK:
i. nltk.stem for Stemming

We used the following modules of SCIKIT-LEARN:
i. sklearn.metrics for generating confusion matrix
ii. sklearn.model_selection for K-Fold cross validation
iii. sklearn.feature_extraction for TF- IDF feature extractor
iv. sklearn.pipeline for list of transformations and applying final estimator
v. sklearn.svm for applying SVM classifier
vi. sklearn.linear_model for applying SGD classifier
vii. sklearn.ensemble for applying Bagging, Boosting and VotingClassifier

Technique

Data Preprocessing: Preprocessing of tweets involved following steps:

A. **Removal of Hyperlinks**: Used regular expressions to remove links as they do not provide essential information for classification.

B. **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. *@FunnyJokeBook*.

C. **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.

D. **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'.

E. **Removal of annotations**: They do not contribute towards classification and are hence removed. E.g. '<a>' and '<e>'.

F. **Removal of other special characters**: All punctuations are removed. E.g. punctuations like '',"",!,?,;,% etc.

G. **Removal of RT(ReTweet) Tag**: RT tags at the start of the tweets do not contribute towards classification.

H. **Stripping off white spaces**: Additional white spaces are stripped off from the tweet.

I. **Fixing repeated characters**: E.g. "Goooood" becomes "Good".

J. **Conversion to lower case**: This helps in maintaining uniformity.

K. **Tokenizing the tweets**: The space separated tweets are tokenized to obtain the tweets as list of words.

L. **Expanding abbreviated words**: Abbreviations are expanded to make them more meaningful. E.g. "lol" becomes "negative" and "pts" becomes "points".

M. **Lemmatizing the tweet**: Each word undergoes lemmatization in order to obtain the root word. E.g. "telling" becomes "tell".

N. **Removal of stopwords**: Maintained a separate file of all possible stop words of English language. These do not contribute towards the classification.

O. **Removal of digits**: Digits are unimportant during classification and are removed. E.g. Obama2012.

P. Other Preprocessing steps included removing white spaces, Removing words that start with a special character, etc.

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 3 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 through the model created by the pipeline 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". Also, in pipeline the other feature that is used is TF-IDF i.e.TermFrequency-InverseDocumentFrequency. TfidfTransfomer of the library **sklearn.feature_extraction.text** is used which uses CountVectorizer and then Tfidf-Transformer to store the count of each word in the separate vector.

Classification Methods:

A. Multinomial Naive Bayes Classifier
B. Support Vector Machines(SVM) with RBF/Gaussian Kernel
C. Stochastic Gradient Descent(SGD)
D. Logistic Regression
E. Ensemble Methods:
  i. Random Forest
  ii. Bagging
  iii. Boosting(XG-Boost)
  iv. Voting using SVM with Gaussian kernel, Logistic Regression, Random Forest and Stochastic Gradient Descent.
F. Neural Network: Convolutional Neural Network

Evaluation

A. **Experimental Results**

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.

OBAMA:

Average%	Precision_Negative	Precision_Positive	Recall_Negative	Recall_Positive	Fscore_Negative	Fscore_Positive	Overall Accuracy
Multinomial Naive Bayes Classifier	**52.3**	**66.0**	**77.6**	**51.99**	**60.87**	**57.07**	**55.74**
SVM(rbf)	**56.52**	**65.73**	**56.77**	**68.00**	**60.320**	**60.59**	**58.72**
Stochastic Gradient Descent	**56.54**	**59.11**	**62.5**	**60.24**	**58.21**	**59.22**	**56.2**
Random Forest	**57.14**	**58.30**	**63.76**	**57.85**	**58.34**	**57.48**	**56.22**
Logistic Regression	**57.60**	**61.49**	**64.4**	**60.62**	**59.75**	**60.71**	**58.78**
Majority Voting	**57.36**	**62.4**	**65.3**	**60.03**	**59.88**	**60.79**	**58.58**
XG-Boosting	**56.55**	**55.7**	**58.54**	**54.15**	**55.37**	**56.59**	**54.88**

ROMNEY:

Average%	Precision_Negative	Precision_Positive	Recall_Negative	Recall_Positive	Fscore_Negative	Fscore_Positive	Overall Accuracy
Multinomial Naive Bayes Classifier	**53.6**	**71.15**	**98.32**	**7.65**	**68.11**	**13.43**	**53.57**
SVM(rbf)	**59.26**	**64.67**	**88.26**	**29.66**	**69.5**	**39.98**	**57.33**
Stochastic Gradient Descent	**63.15**	**53.31**	**77.61**	**41.65**	**68.37**	**45.83**	**57.50**
Random Forest	**55.76**	**71.11**	**94.56**	**18.7**	**68.71**	**28.46**	**55.61**
Logistic Regression	**63.89**	**54.16**	**74.99**	**47.23**	**67.79**	**49.70**	**57.68**
Majority Voting	**61.57**	**57.65**	**81.37**	**38.00**	**68.82**	**45.13**	**57.57**
XG-Boosting	**55.33**	**59.15**	**90.15**	**16.59**	**67.13**	**25.53**	**55.90**

Conclusion

Several models have been built and then trained and verified using k-fold cross validation (k=10). Stochastic Gradient descent, Random Forest, SVM and logistic regression gave reasonable average f-scores but an ensemble voting classifier of the above mentioned classifiers in the ratio of 1:1:1:2 performed slightly better than the rest. The corresponding classes of the test data have been determined using ensemble voting classifier.

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. **Bigram or n-gram model**: Bigrams and n-grams collocations are important while classifying tweets as bigrams or n-grams make more sense and provide more information about the sentiment of the tweet.
C. **Spellchecker**: correcting the spellings of words would help to a great extent during classification because misspelled words are meaningless to the classifier.
D. **Neural Network Models** : Recurrence Neural Network with LSTM can be used in order to improve the F1-scores of the test data.

References

1. http://nltk.org/ - for documentation about NLTK libraries
2. Bing Liu. “Sentiment Analysis and Opinion Mining” , May 2012. eBook: ISBN 9781608458851
3. http://streamhacker.com/2010/10/25/training-binary-text-classifiers-nltk-trainer/ - for info on text classification methods
4. http://www.ravikiranj.net/drupal/201205/code/machine-learning/how-build-twitter-sentiment-analyzer - for information on tweet classification
5. http://scikit-learn.org/stable/supervised_learning.html#supervised-learning
6. http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-tensorflow/
7. https://keras.io/
8. https://bbengfort.github.io/tutorials/2016/05/19/text-classification-nltk-sckit-learn.html
9. https://blog.statsbot.co/text-classifier-algorithms-in-machine-learning-acc115293278
10. https://machinelearnings.co/text-classification-using-neural-networks-f5cd7b8765c6
11. https://nlp.stanford.edu/IR-book/html/htmledition/text-classification-and-naive-bayes-1.html