Sentiment Analysis Focused on Emoticons to Interpret the 2014 Midterm Election

Doga Tuncay
Cong Li

Table of Contents

Preface

This project explains how emoticons can contribute to the accuracy of Machine Learning algorithms. We focused on political data that is collected from the social media platform Twitter and more specifically on the data collected before, during and after the 2014 U.S. Midterm Elections. We will run several classification algorithms to do a Sentiment Analysis on the collected tweets using the data model we generated by using the emoticons as the classes' main attributes.

Abstract

Twitter has become the main social media platform to discuss the current events all around the world and the elections is a great example for it. Social media plays an important part on the elections and there are various papers on how the people signalize their emotions and inclinations using Twitter. Twitter users which include the politicians, citizens and partisans show their strong opinions on the political party and these opinions are very important to strategize the topics to win the elections. There are several researches on predicting the political alignment of Twitter users based on the Twitter content. There are several approaches and various accuracies achieved by the research community ranged between %59 and %95 of accuracy. It is proven that the training set is an important piece for the higher accuracy rather than the Machine Learning algorithm itself and we'll focus on the data model to build our classifiers by using emoticons. In our research, we will prove that the emoticons play a crucial role on sentiment analysis and the usage of emoticons to model the training data will contribute to the Machine Learning algorithms' accuracy. Our other objective is to predict the political inclination using the emoticon based classification and compare the results with other sentiment analysis projects and with the 2014 U.S. Midterm Election's results. We will choose the best machine learning algorithm among Naive Bayes, Logistic Regression, SVM, Maximum Entropy that are available on the Mahout platform, to support our hypothesis.

Introduction

1. Objective

Our objective is to prove the emoticons play a crucial role on sentiment analysis and we believe that the usage of emoticons to model the training data contributes to the Machine Learning algorithms' accuracy. Our other objective is to predict the political inclination using the emoticon based classification and compare the results with other sentiment analysis projects and with the 2014 U.S. Midterm Election's results

2. What is the problem and the statement of the problem

There is no common "problem" per se, but we think that emoticons were not studied in depth and we believe that they have a great impact on the success of the sentiment analysis.

3. Why this is a project related the this class

Sentiment Analysis (Opinion Mining) concerns Natural Language Processing, Text Analysis as well as Machine Learning. Sentiment analysis of a text is mostly done using machine learning algorithms.

4. Why other approach is no good and why you think your approach is better

Other approaches don't take emoticons into consideration as the way we did. We are going to build our model based on the emoticons and we believe that our way is going to have better classifier accuracy. We are also going to examine emoticons closely. Other approaches didn't study emoticons widely.

5. Area or scope of investigation

The area that we are scoping is sentiment analysis on political data to predict the political alignment during recent elections with selected classifiers using emoticons.

Theoretical bases and literature review

1. Definition of the problem

Based on the researches we have done so far, we saw that the usage of hash tags reveals a strong inclination towards to a certain political party and these projects were supported with usage of mentions, Twitter user data and as well as emoticons. However, emoticons weren't taken into consideration as we propose.

2. Theoretical background of the problem

Social media plays an important part on the elections and there are various papers on how the people signalize their emotions and inclinations using Twitter. In theory, using Machine Learning algorithms, we can detect a twitter user's political alignment. There are several approaches and various accuracies achieved by the research community ranged between %59 and %95 of accuracy. It is proven that the training set is an important piece to the accuracy rather than the Machine Learning algorithm itself.

3. Related research to solve the problem, advantages and disadvantages of those research

We have done a research on the published papers on Sentiment Analysis based on Twitter Data with a focus of political data.

One research was to build a system to predict sentiments real-time. The advantage of the project is the ability to combine the historical twitter data to real-time data and run the sentiment analysis on the merged data set and while the Twitter sentiment projects take a long time, their system was producing the results continuously and instantly. However, the disadvantage of this project, in our opinion, is the usage of Amazon's Mechanical Turk and the user interface that asks about the sentiment on the given tweet didn't contribute much on the accuracy, even though the purpose was the otherwise. The data model was based on the 200 rules that they came up with and we believe the data model play an important role on the success of the classifier and they achieved the %59 accuracy with this system, which is not a very successful result. So, we believe that the data model wasn't well built as well. Another disadvantage is the classes they were not picked wisely, the research proposed four classes which are positive, negative, neutral and unsure. However the members of the class unsure could be a result of misclassification.

Another research was on the sentiment classification using distant supervision and we believe that we need to use the learning curve to know whether Distant Supervision is needed for the emotion classification using emoticons. Even if the paper discusses the emoticons, they didn't mention it. The paper only proposed an idea to get more data but not the necessity of the data, if they believe that a learning curve shows that more data means more accurate classification.

4. Your solution to solve this problem/ where your solution different from others and why your solution is better

We propose a classifier model based on using emoticons as labels and generate our training data. Our goal is to see how the accuracy changes based on the labels we define and test the classifier on a new data set where the emoticons are filtered out and see how well our machine learning performs. This way, we can build a better system to classify the tweets into sentiments and have a better accuracy than the just text based classifiers.

Hypothesis

Positive/negative hypothesis, multiple hypothesis

- Larger training and validation data can be generated by tweets with emoticons.
- Larger training data can be used to generate more accurate classification algorithm.
- The learning curve of any machine-learning algorithm can be used to show that;
    - Larger data set means better algorithm
    - The method that can produce larger data set may result in more accurate classification algorithm

Methodology

1. How to generate/collect input data
    - Collect data from Twitter API using predefined mentions, emoticons and hash tags
    - Generate training and validation data labeled with the emoticon
    - Use tweets of top 10 politicians on each party to know the positive or negative emotions from our people before and after midterm election
2. How to solve the problem
    - Choose the best ML algorithm among Naive Bayes, Logistic Regression, SVM, Maximum Entropy to support our hypothesis
    - Generate learning curve to know the data needed for better classification method
3. Algorithm design
    - Generate tweets to data point by standard sentiment analysis methods
    - Classify data by applying ML classification methods; find the best algorithm with highest accuracy
    - Draw learning curve with training and validation accuracy with limited data
4. Languages used
    - Apache Mahout libraries in Java platform
    - Python
    - C#
5. Tools used
    - Twitter API
    - AWS CloudFormation
    - Amazon S3

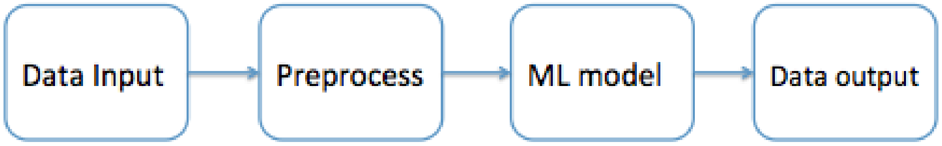
6. *a prototype (optional if time permit)*
7. How to generate output
    - Apply different ML classification among Naive Bayes, Logistic Regression, SVM, Maximum Entropy algorithms directly and compare accuracy with other existing sentiment analysis platform
    - Minor adjustment such as data preprocessing might be needed for high accuracy
    - Apply the most accurate algorithm to those tweets to top 10 politicians and know people's emotions and preference
8. How to test against hypothesis

- Pick a high accuracy classification algorithm
- Train the algorithm with limited data and draw the learning curve for training and validation set
- 3. If the gap between learning curves of training and validation data is large, larger data is needed for better classification
- Then the distant supervision with any weak label, such as emoticon here, can significantly improve the
- If the gap between learning curves of training and validation data is not large, larger data is not needed for better classification. Then small data set with accurate label should be used to generate good sentiment analysis model

9. *how to proof correctness (required by dissertation only)*

Implementation

1. Code
    - Our code base is composed of two major parts, data preprocessing part and model training and validation part, as figure below shows.



- Data inputs are Tweets captured by Tweets fetching application in Python code
- Preprocessing are done by C# to transform any tweets with given significant positive and negative word list
- ML model are based on Weka free software in Java API
- We mainly use Weka as our machine learning tool to run most of classification algorithms including Naïve Bayes, Decision Tree, Logistic Regression, and Support Vector Machine.
- By Wikipedia, Weka (Waikato Environment for Knowledge Analysis) is a popular suite of machine learning software written in Java, developed at the University of Waikato, New Zealand. Weka is free software available under the GNU General Public License.
- Data output contains error rate and any related statistic results

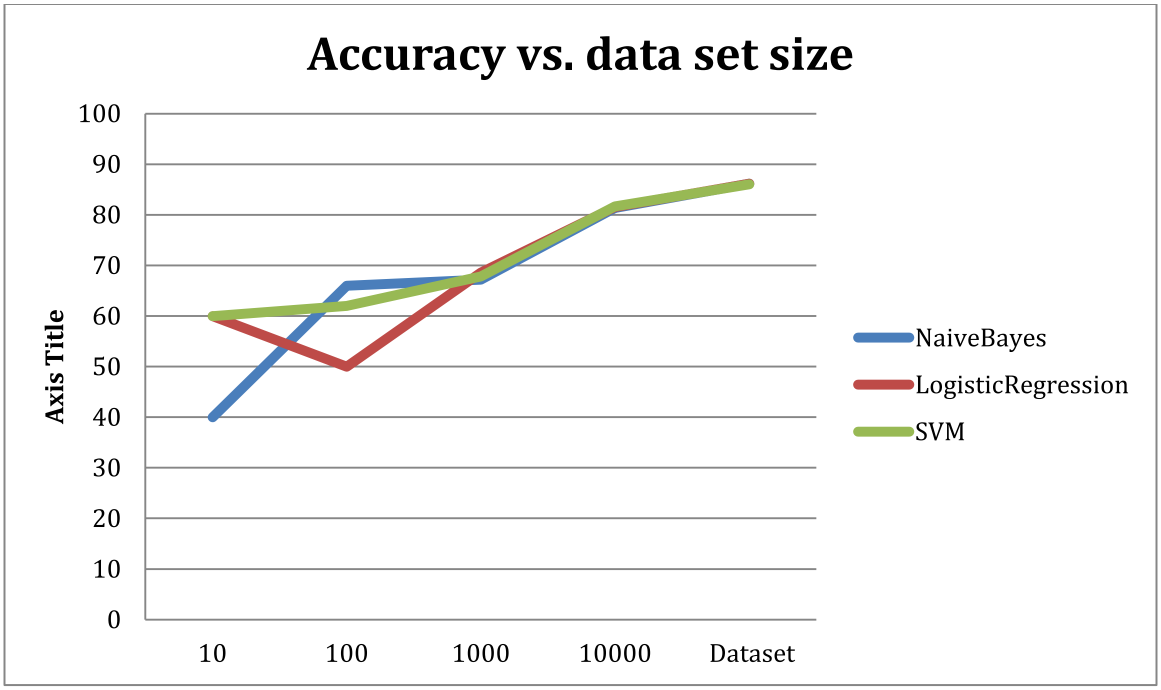
2. Design document and flowchart
    - Flow chat is shown above
    - Data input is to capture tweets by Python app
    - Data preprocessing is to generate any tweets to data point by match each keywords in positive and negative keyword list from Twitrratr.com. Input is any tweet. Output is a binary line like 1, 0, 1, 0, … as the first digit shows whether the tweet itself is positive or negative. And the rest of digits show whether this tweet contains corresponding keywords. In the keyword list from

Twitrratr.com, it contains to 174 positive words and 185 negative words. We also created positive and negative keyword frequency list by MapReduce. It gives us some sense for how to generate our own keyword list

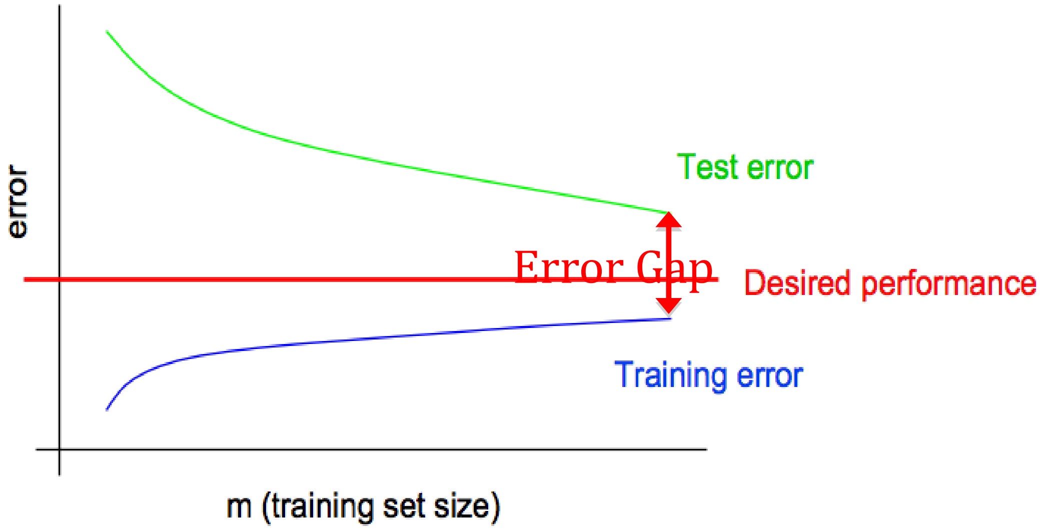
- ML model contains 4 major ML algorithms, Naive Bayes, Logistic Regression, SVM, Decision Tree. I implement them all with Weka MLlib and compare their performance. By changing the data set size, we can also get some ideas about ML algorithm performance with data set size increasing.
- Data output contains accuracy with different data set size and different ML models. After proper visualization, we created figures to show results for or against our hypothesis.

Data analysis and discussion

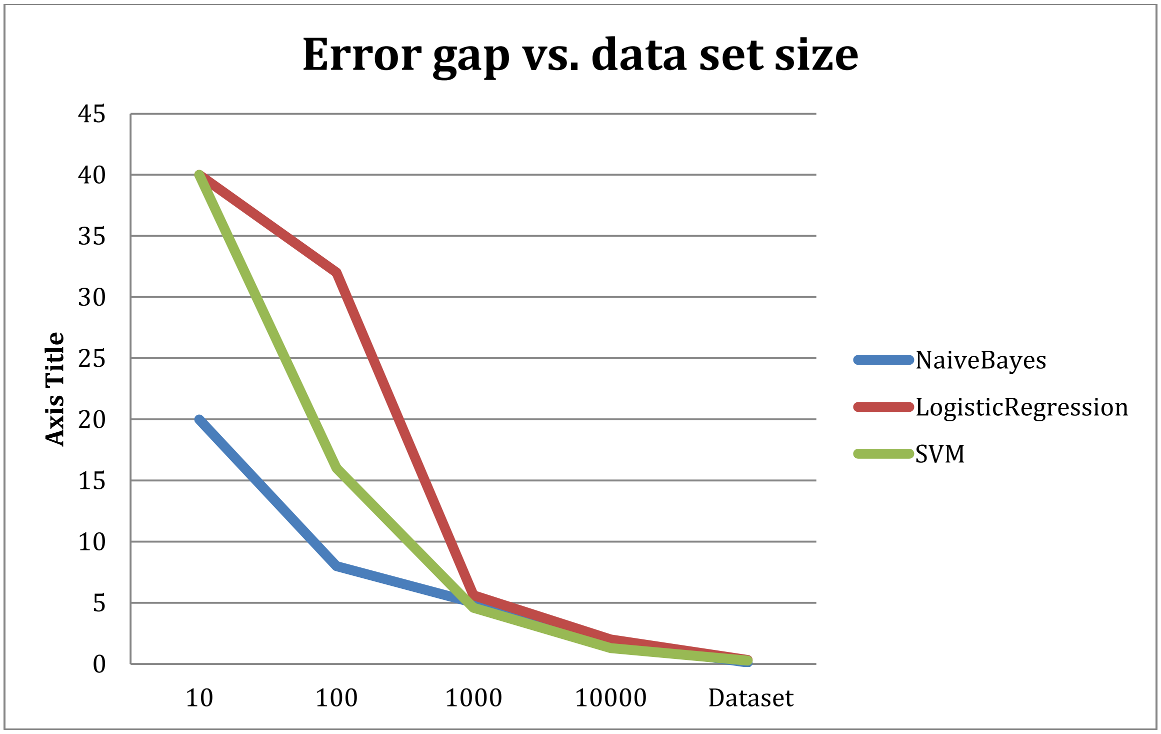
1. Output generation
    - We randomly pick half of data to be training data and the other half to be validation data
    - We save error rates for both training data and validation data
    - When training data has much lower error rate, it means that sample size is not big enough to avoid bias and over fitting problem; we need to increase the sample size
    - We increase data set continuously to draw learning curve to determine the best size of data set
    - Learning curve output also helps us understand the proper data size needed for any given ML model on Twitter sentimental analysis
2. Output analysis
    - As data set increases, different ML algorithms show very promising and similar accuracy increase



- Error gap in the learning curve decreases as data set increases
  Error gap is defined as the difference between accuracy rate of training data and validation data, as shown in the figure below.

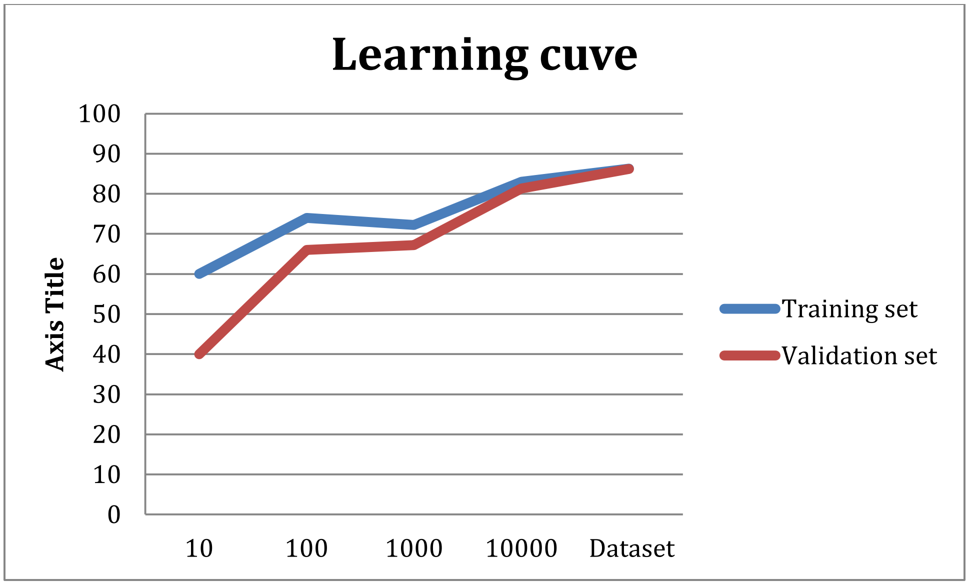


Here is the figure to show that error gap diminishes as data set size increases.



- When data set is about 12,000, error rate of different algorithms all diminishes to almost 0. And it means that we do not need to increase training data set any more to have a better ML model to predict Twitter sentiment analysis

3. Compare output against hypothesis
    - By distant supervision, large training and validation data can be easily generated by tweets with weakly and noisily labeled by emoticons, and without any human efforts
    - Large data set does provide better ML models. All different models has increasing accuracy when data set increases
    - However, data set with 12,000 tweets seems good enough, with about 86% accuracy for basic Twitter sentiment analysis. We may need to have a more accurate input data set rather than having a weakly labeled tweets
4. Abnormal case explanation
    - Learning curve is not like normal in same shape as expected



- It is probably because starting data set is too small as 10. We need to have minimum data set for about 10,000 and start to draw learning curve, like the figure here referenced from [13]. They start to draw the figure there with 0.1 million data set.

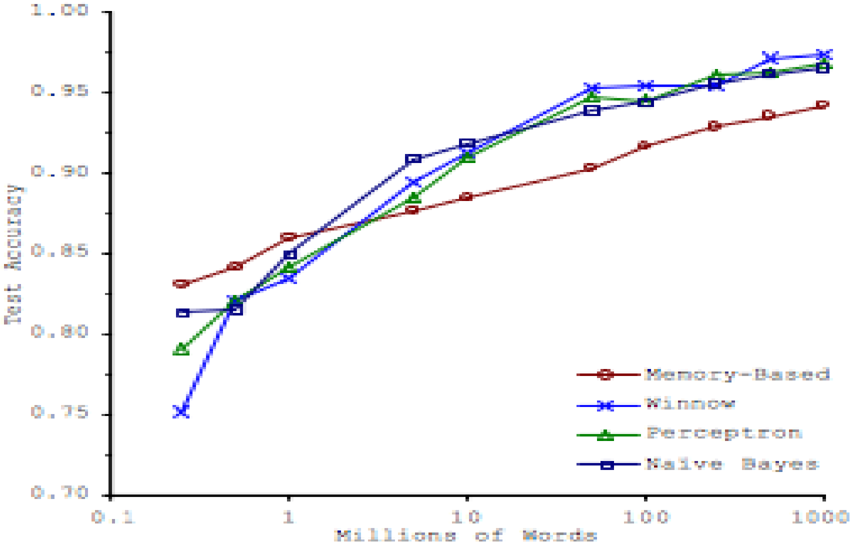


Figure 1. Learning Curves for Confusion Set Disambiguation

5. Statistic regression
    - We run 10-fold cross validation to show that the error rate of entire data with 10-fold is very similar to picking half data as training and validating with the other half.
    - We expect higher accuracy with more data as the figure "accuracy vs. data set size" shows
6. Discussion

- Different ML models have very similar accuracy, when data set size is big enough. So in most of time, finding data has equivalent importance as applying ML models to it.
- Comparing the error rate between training set and validation set can tell us whether we still need more data to improve the classification model. When the error gap decreases to 0, we know that current data size is good enough for ML analysis
- Twitter sentiment analysis needs about 10,000 to have a classification model with good enough accuracy. More data might not be that meaningful.

Conclusions and recommendations

1. Summary and conclusions
   In summary, our research showed us that emoticons can be used to improve the classifiers accuracy rate as well as the data size. By distant supervision large training and validation data can be easily generated by using the tweets even though the data is structured with noise and without any text correction. We proved that as the data size goes up the accuracy of the classified model increases as well. We collected nearly 12.000 tweets and we were able to reach the accuracy of %86 percent. We also proved that different machine learning algorithms' accuracy meet closer as the data size increases, which also causes a decrease in the error gap and convergence. We used two positive, negative words lists; one from is compiled from the frequency of the unique words of the tweets and the other was the list that was available Twitrratr. We found out that using a global list would be fairer and give us more accurate results.
2. Recommendations for future studies
   Our research was run on data size as high as 12k+ tweets. Even though the data size we used was enough to get an accurate result, our future study would be using a bigger dataset, preferably closer to 1 million to support the research that we discussed in the project proposal. Another upgrade would be the spelling and acronym correction. The dataset we used was cleaned from special characters, links, emoticons and extra ordinary spaces. The next step would be using of a better data cleaning method that also corrects the noisy data.

Bibliography

1. Alec Go, Richa Bhayani, Lei Huang, Twitter Sentiment Classification using Distant Supervision.
2. Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, Rebecca Passonneau, Sentiment Analysis of Twitter Data
3. Adam Bermingham and Alan F. Smeaton, On Using Twitter to Monitor Political Sentiment and Predict Election Results

4. Hao Wang, Dogan Can, Abe Kazemzadeh, Francois Bar and Shrikanth Narayanan, A System for Real-time Twitter Sentiment Analysis of 2012 U.S. Presidential Election Cycle

5. Conover, M. D., *Gonçalves, B., Ratkiewicz, J., Flammini, A., & Menczer, F. (2011, October). Predicting the political alignment of twitter users. In Privacy, security, risk and trust (passat), 2011 ieee third international conference on and 2011 ieee third international conference on social computing (socialcom) (pp. 192-199). IEEE.*

6. Felipe Bravo-Marquez, Marcelo Mendoza, and Barbara Poblete. 2013. Combining strengths, emotions and polarities for boosting Twitter sentiment analysis. In *Proceedings of the Second International Workshop on Issues of Sentiment Discovery and Opinion Mining* (WISDOM '13). ACM, New York, NY, USA, , Article 2 , 9 pages. DOI=10.1145/2502069.2502071 http://doi.acm.org/10.1145/2502069.2502071

7. Jose J. Padilla, Saikou Y. Diallo, Hamdi Kavak, Olcay Sahin, and Brit Nicholson. 2014. Leveraging social media data in agent-based simulations. In *Proceedings of the 2014 Annual Simulation Symposium* (ANSS '14). Society for Computer Simulation International, San Diego, CA, USA, , Article 17 , 8 pages. DOI=10.1145/2481244.2481247 http://doi.acm.org/10.1145/2481244.2481247

8. Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. 2011. Sentiment analysis of Twitter data. In *Proceedings of the Workshop on Languages in Social Media* (LSM '11). Association for Computational Linguistics, Stroudsburg, PA, USA, 30-38

9. Vinh Ngoc Khuc, Chaitanya Shivade, Rajiv Ramnath, and Jay Ramanathan. 2012. Towards building large-scale distributed systems for twitter sentiment analysis. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing* (SAC '12). ACM, New York, NY, USA, 459-464. DOI=10.1145/2245276.2245364 http://doi.acm.org/10.1145/2245276.2245364

10. Pak, A. & Paroubek, P. (2010). Twitter as a Corpus for Sentiment Analysis and Opinion Mining. *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, May, Valletta, Malta: European Language Resources Association (ELRA). ISBN: 2-9517408-6-7

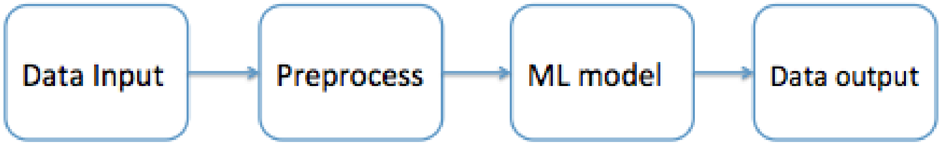
11. Go, Alec, Richa Bhayani, and Lei Huang. 2009. "Twitter Sentiment Classification Using Distant Supervision." *CS224N Project Report, Stanford*: 1–12.http://cs.wmich.edu/~tllake/fileshare/TwitterDistantSupervision09.pdf(May 23, 2013).

12. Brent Hecht, Lichan Hong, Bongwon Suh, and Ed H. Chi. 2011. Tweets from Justin Bieber's heart: the dynamics of the location field in user profiles. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '11). ACM, New York, NY, USA, 237-246. DOI=10.1145/1978942.1978976 http://doi.acm.org/10.1145/1978942.1978976

13. Michele Banko and Eric Brill. 2001. Scaling to Very Very Large Corpora for Natural Language Disam- biguation. In Proc. of ACL-2001

Appendices

1. Program flowchart



2. Program source code with documentation
    - Data input : \MLProject\code\twittercollect27\
    - Source Code:
        - Preprocessing : \MLProject\code\DataCleanup\
        - ML model : \MLProject\code\Weka*
          SourceCode_MLmodels folder
          Each folder contains test code for each model
          Every java file contains processing with different data set size with half data as training and the other half as validation
        - Data output \MLProject\data\
          Outputs_MLmodels
          Each txt file contains output for each model
          Figures.xlsx shows data collected from each txt and corresponding figures drawn based on output data
3. Input/output listing
    - Data input: MLProject\data\crosscheck_data_input\*.txt
    - Data output:
        - MLProject\data\outputs\Logistics.txt
        - MLProject\data\outputs\SVM.txt
        - MLProject\data\outputs\Naive Bayes.txt
    -
4. Other related material
    - Basic example source code for Weka MLlib (http://weka.wikispaces.com/Use+WEKA+in+your+Java+code)
    - Positive and negative keyword list found online from Twitrratr.com (http://www.twtbase.com/twitrratr/)
    - Positive and negative keyword list that we generated based on the frequency.
        - MLProject\data\ frequencies\negative_frequency.txt
        - MLProject\data\frequencies\ positive_frequency.txt