Sentiment Analysis
CSCI 5525: Machine Learning
Final Report : Group 8

Group Members - Saloni Srivastava, Shivangi Singh, Shneka Swamy

Problem Statement:

Sentiment is a word usually associated with feelings, such as attitudes, emotions and opinions which are usually subjective impressions but not facts. In today's world, majority of the population has an online presence. People often take to online platforms to express their feelings like through microblogging, giving reviews about products they bought or conveying an opinion in the form of comments. Generally one assumes a binary opposition in terms of opinions, like for/against, like/dislike or good/bad.

Sentiment Analysis constitutes of the use of Natural Language Processing, statistics or machine learning to extract to identify or otherwise characterize the sentiment content of a given unit, such as a sentence or a document. This has many applications in the modern day and age. Is this product review good or bad? Based on customer's response email, is he satisfied or dissatisfied? Based on a sample of tweets how are people responding to an advertisement campaign, product release of a news item? How have bloggers attitudes changed regarding the president since the elections?. Other uses can be in the form of bias identification in news sources, identifying children suitable content, analysing trends and evaluation of public voter's opinions.

There are many challenges one faces when they try to evaluate a person's opinion based on what they wrote since people express their opinions in complex ways in which the lexical content alone can be misleading. Intra-textual reversals, negation and topic changes are some of the common problems one comes across. Further every language has its own set of slang and when using informal platforms like social media, people tend to use shortened forms of each word which is difficult to interpret. Usage of rhetorical modes such as sarcasm, irony and implications create more hurdles. Also, some problems arise when we have few categories to classify text into (positive/negative, 1-3 stars) as compared to text categorization, non independent categories, crosses in terms of domains and topics, difference in characteristics of answers to opinion based interview questions and fact based interview questions.

Related Work:

Expressing opinions via an online portal has been around for quite some time now and many people and organizations have attempted and succeeded till an extent in the field of sentiment analysis for this type of user generated content. The basic approach usually involves the extraction of features using text mining and then applying an algorithm for final categorisation using those features. This interest among the research community in the analysis of opinions began in 2002 with the publication of two reference studies which represent the two main approaches to tackling the problem of Sentiment Analysis.

These two strategies are the ones based on the application of linguistic analysis represented by the work of Turney [1] and that the others that use machine learning techniques which were epitomized by Pang, Lee and Vaithyanathan [2]. Other authors prefer to refer to these two approaches as unsupervised learning and supervised learning. The current body of work attempts to exploit the advantages of both approaches and hybrid systems are proposed, these being represented by the work of Prabowo and Thelwall [3]. The first approach is lexicon-based and second is machine learning-based. The lexicon-based approach determines the sentiment or polarity of opinion via some function of opinion words in the document or the sentence. The machine learning-based approach typically trains sentiment classifiers using features such as unigrams or bigrams. Most techniques use some form of supervised learning by applying different learning techniques such as Naıve Bayes, Random Forests and Support Vector. Much of the research has also gone in identifying and extracting the right features for this. Considering the challenges described above and the research experience by various researchers, achieving a 70-80% accuracy is considered good enough and anything beyond that can lead to overfitting.

Dataset Used:

Twitter: This consists of two columns, One with the labels and the other with the text of the tweets people posted. The labelling consists of 0 for positive, 2 for neutral and 4 for negative. In this we have 1.6 million data points. We are not considering the neutral ones and only running a two-class classification on the dataset. We took a subset of these tweets for training and testing considering the compute limitations on the available systems for running our experiments. http://cs.stanford.edu/people/alecmgo/trainingandtestdata.zip:

Data Pre-processing:

The twitter data available is cleaned and preprocessed to make it suitable for analysis. The following steps are followed.

1. ***Tokenization:*** Each tweet obtained was split into words based on whitespace for analysis. Apart from that, the emotion icons and the URL links are also identified and separated out.This was done using the TweetTokenizer function present in the NLTK Toolkit.
2. ***Reduce Lengthening and Tolower:*** Twitter data is highly personalized and hence the user might use "I looooove kindle"instead of "I love Kindle". The reduce_lengthening function is used to remove such duplicates. All the words are uniformly kept in lowercase for ease of computation.
3. ***Emoji Extraction:*** The emotion icons play a very important role in Sentiment Analysis and hence they were extracted separately. This was done by replacing happy and sad smileys with EMO_HAPPY and EMO_SAD.
4. ***Punctuation Extraction:*** Usually punctuations do not have an impact on Sentiment Analysis but if a particular punctuations like "?" and "!" occurs more than once in a sentence then it might indicate a person's excitement regarding a particular topic. Hence these punctuations were replaced by "QUES" and "EXPL" respectively.
5. ***Unnecessary Substring Removal:*** As a part of this step URLs, user tags and hashtags were removed since these are irrelevant when it comes to analysis.This was done by identifying words that contain "http","#" or "@" and removing them.
6. ***Punctuation Removal:*** Apart from the Punctuations mentioned in point 4 all other punctuations do not impart any information.Hence they are not considered as a feature and are removed from the maintained tokenized data.
7. ***Negation Marking:*** The presence of words like "not", "no"and "never" influences the analysis by negating the meaning of a given sentence. Therefore all such words are extracted by replacing it by a simple "NOT".
8. ***POS Feature Extraction:*** Each word is tagged by using the pos function defined in the NLTK Toolkit.This was used on the data that is generated after all the preceding data pre processing steps are done.Since this takes a lot of time to run and it was therefore executed only for the number of words that were used in the analysis rather than the entire sentence.
9. ***Stop word removal:*** All stopwords such as 'a', 'an', 'the', 'and' and 'but' are removed. This was achieved by using the stopword function in the NLTK Toolkit.

Feature Extraction:

1) Bag of words: This was done using the scikit library. We first fed it all the tweets from our data to identify a set of unique words. Each of these words was then considered as a potential feature for our analysis. This set of unique words was really large. Hence we identified a subset of most frequent words as our feature set. Hence for each tweet, we generated an array of numbers corresponding to each of these words indicating their frequency in the tweet.
2) Punctuations: For each tweet,based on the information obtained from the preprocessing step a feature was defined to indicate the presence or absence of the same.
3) Negation:Count of number of "NOT" in the preprocessed text is used as a feature.
4) Sentiment Lexicon: Each word we got after preprocessing had an emotion attached with it. The Multi-Perspective Question Answering (MPQA) [11] database from the University of pittsburgh provided this for 20,000+ words in the english language. For each token from the tweet, we queried this database and identified the nature of the word. Based on this, we added two features, one corresponding to the count of positive token encounters, and the other one for the negative tokens.
5) Emojis: We added the count of emotions preprocessed earlier as a feature in a way that is similar to the Sentiment Lexicon.

6) POS features: During pre-processing, tag for each word was obtained which was used to count the number of nouns, verbs ,adjective and adjective present in the sentence and each count was considered as a feature.

The final set of features used:

1) Count of n most frequent words (Unigrams) and phrases (Bigrams and Trigrams)
2) Punctuations: Boolean indicating presence of Question mark and Exclamation.
3) Emoticons: Count of positive emoticons and negative emoticons
4) Negation - Count of any word used in the tweet that negates the meaning.
5) Part of speech: For each tweet, count of Nouns, Verbs, Adverbs and Adjectives
6) Emotions attached with words: For each tweet, count of Positive and Negative words used

Methods Used:

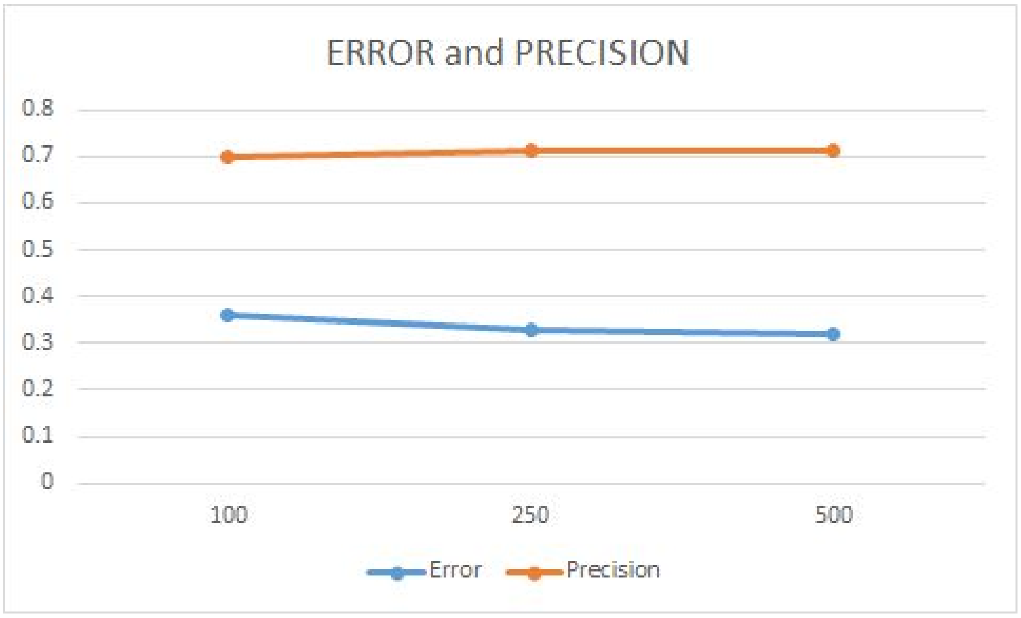
Text Classification is a well-known problem and the features chosen generally are the frequency of certain characteristics in each text (Tweets in our case). Further bag of words is the main feature used for our classification. Considering the fact that tweets are limited to 140 character, limits the count of words that can be present in a tweet, and thus results is a very sparse feature set for bag of words. A set of algorithms have shown to work better with such frequency data which is also sparse in nature. We used a subset of such algorithms, listed below, for our study:

1) Naive Bayes (Multinomial)
2) Maximum Entropy (LR)
3) Linear SVM (SVM)
4) Ridge Classifier (RC)
5) Stochastic Gradient Descent (SGD)
6) Ensemble methods - AdaBoost Algorithm (AB) and Random Forest (RF)

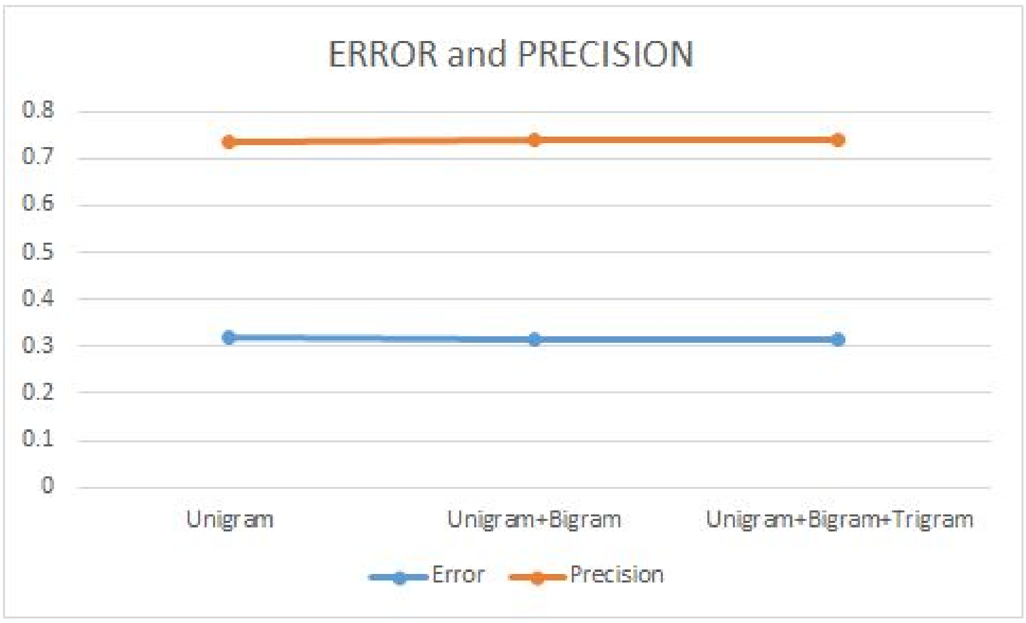
Result and Analysis:

We performed 2 set of experiments:

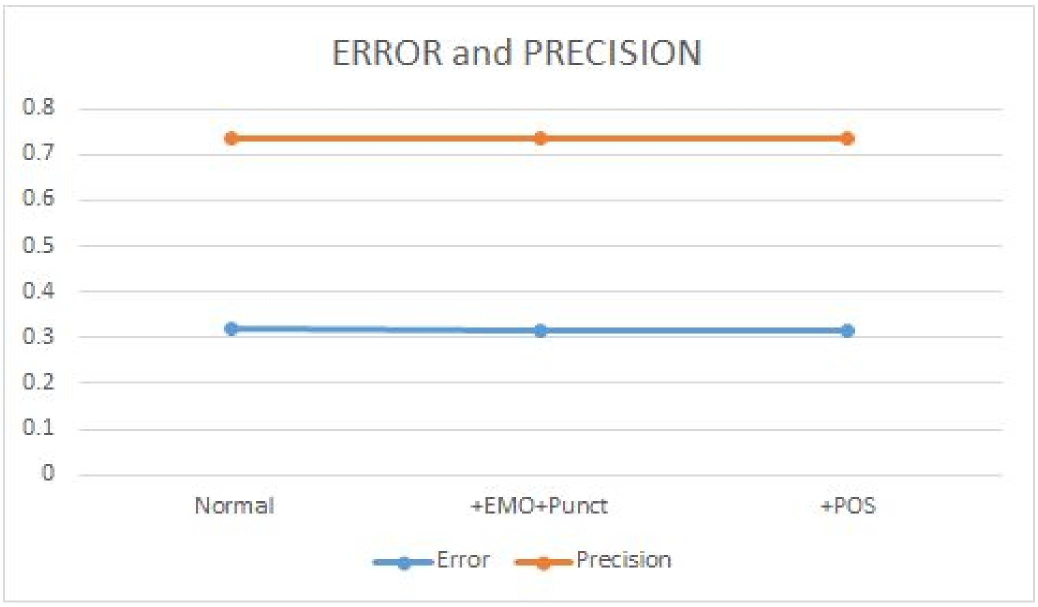
1) Study the impact of feature variation: These sets of experiments were performed on our local systems, hence were limited in scale. Hence we used 50,000 tweets for training and testing. With 10-fold cross-validation following results were observed:
    a) *Bag of words*: With varying count of most frequent words following variations in accuracy was observed: **[Please refer to Table 1 for details]**



b) From including just the Unigram, to Unigram+bigram and finally Unigram+bigram+trigrams, we measure the variation in accuracy by adding following features: **[Please refer to Table 2 for details]**

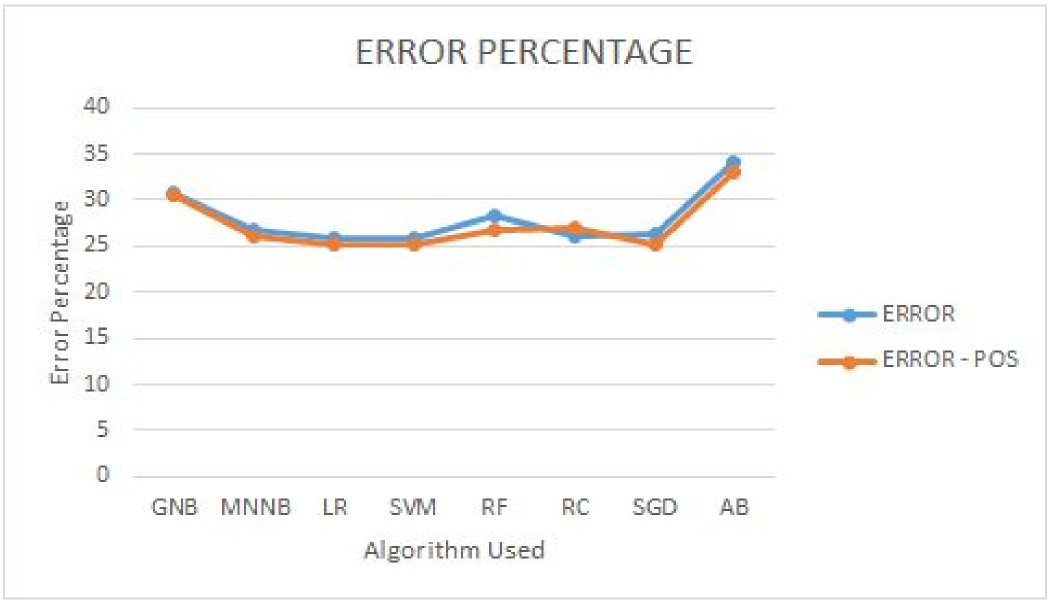


c) Since, we did not observe any significant impact of including bigrams and trigrams on the accuracy, we took unigrams as the base features and added other features one after the other to see their impact. **[Please refer to Table 3 for details]**



2) Study the impact of variation of algorithms: These sets of experiments were performed on cselab machines hence we used 200000 tweets for training and testing. We ran each algorithm with 2 different feature vectors:
    a) Bag of words - 1000 most frequent Unigrams
    b) Bag of words + Part of Speech features + Emoticons + Negation

[Please refer to Table 4 & 5 for details]



Conclusions & limitations:

Based on our experience with the three phases of sentiment analysis - Preprocessing, feature extraction and running the algorithms, we conclude the following:

1) As the related work indicates that for sentiment analysis, an accuracy of 70-80% is good enough, we observed that an accuracy of up to 75% can be achieved with just using the bag of words, if the tokenization and unnecessary substring removal is done appropriately. We also observe improvement in accuracy with increase in the size of bag of words.
2) Our study with varying the feature set indicates that there is not much improvement observed on adding very specific features such as punctuation, POS, emoticons etc. While it can be the case that we are extracting it incorrectly, but considering the fact that they are limited to very few tweet, it seems reasonable to us.
3) Algorithm variation indicates that Multinomial Naive Bayes works the best, which is known to perform well with frequency related applications such as ours. We also observe that algorithms like Maximum Entropy and SGD perform equally well.
4) One surprising result for us was that none of the ensemble methods performed well for our application.
5) Also,more emotions can be included in the analysis using multi-class algorithms.

References:

[1] Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews, Peter D. Turney

[2] Thumbs up? Sentiment Classification using Machine Learning Techniques, Bo Pang, Lillian Lee, Shivakumar Vaithyanathan

[3] Sentiment Analysis: A Combined Approach, Rudy Prabowo, Mike Thelwall

[4] Twitter Sentiment Analysis for Security-Related Information Gathering., Anna Jurek, Yaxin Bi, Maurice Mulvenna. -- For Data preprocessing

[5] Sentiment Analysis in Twitter - Eugineo,Teresa Martin,Alfonso Urena,Arturo Montejo -- For Algo used

[6] Sentiment Analysis in Twitter using Machine Learning Techniques ., Neetu M S,Rajasree R

[7] Twitter Sentiment Analysis: The Good the Bad and the OMG!, Efthymios Kouloumpis, Theresa Wilson, Johanna Moore

[8] Impact of Feature Selection Techniques for Tweet Sentiment Classification, Joseph D. Prusa, Taghi M. Khoshgoftaar, David J. Dittman

[9] Sentiment Analysis using Naïve Bayes with Bigrams, Mohd Abdul Hameed Adnan Rashid Hussain S. Fouzia Sayeedunnissa

[10] Text Mining Facebook Status Updates for Sentiment Classification, Jalel Akaichi, Zeineb Dhouioui, Maria José López-Huertas Pérez

[11] University of Pittsburgh, MPQA library**-** http://mpqa.cs.pitt.edu/lexicons/

[12] Opinion Mining, Sentiment Analysis, and Opinion Spam Detection, https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html#datasets

[13] Sentiment Analysis using nltk - https://github.com/nltk/nltk/wiki/Sentiment-Analysis

[15] Sentiment Symposium Tutorial: Lexicons - http://sentiment.christopherpotts.net/lexicons.html

[16] NLTK Library - http://www.nltk.org/

[17] SCIKIT Library - http://scikit-learn.org/stable/documentation.html

Detailed results

1) **Table 1**

	100 words	250 words	500 words
Training Error mean	0.357	0.324	0.309
Training Error STD	0.0009	0.0009	0.0006
Test Error mean	0.359	0.328	0.317
Test Error STD	0.008	0.004	0.006
Training Precision mean	0.702	0.717	0.744
Test Precision mean	0.699	0.712	0.733
Training Recall mean	0.493	0.582	0.581
Test Recall mean	0.491	0.577	0.572

2) Table 2

	Unigram	Uni+Bigrams	Uni+Bi+Trigrams
Training Error mean	0.309	0.308	0.308
Training Error STD	0.0006	0.0006	0.0006
Test Error mean	0.317	0.315	0.314
Test Error STD	0.006	0.005	0.005
Training Precision mean	0.744	0.747	0.747
Test Precision mean	0.733	0.738	0.739
Training Recall mean	0.581	0.578	0.579
Test Recall mean	0.572	0.571	0.572

3) Table 3

	Unigrams	Unigram +Emoticons + Punctuations	Unigram +Emoticons + Punctuations + POS + Emotions
Training Error mean	0.309	0.309	0.309
Training Error STD	0.0006	0.0008	0.001
Test Error mean	0.317	0.316	0.316
Test Error STD	0.006	0.005	0.004
Training Precision mean	0.744	0.745	0.745
Test Precision mean	0.733	0.736	0.735
Training Recall mean	0.581	0.579	0.579
Test Recall mean	0.572	0.572	0.573

4) Table 4

		Gaussian NB	Multinomial NI	Maximum Entropy	Linear SVM
Training error mean	Bag of words	0.302	0.265	0.254	0.255

Training error mean	+ Other Features	0.303	0.259	0.247	0.247
Training error STD	Bag of words	0.0007	0.002	0.0004	0.0002
Training error STD	+ Other Features	0.0008	0.0003	0.0003	0.004
Test error mean	Bag of words	0.306	0.268	0.258	0.258
Test error mean	+ Other Features	0.306	0.261	0.251	0.251
Test error STD	Bag of words	0.004	0.004	0.004	0.002
Test error STD	+ Other Features	0.002	0.003	0.002	0.003
Training Precision	Bag of words	0.751	0.738	0.769	0.772
Training Precision	+ Other Features	0.752	0.742	0.774	0.777
Test Precision	Bag of words	0.746	0.735	0.765	0.769
Test Precision	+ Other Features	0.748	0.740	0.770	0.773
Training Recall	Bag of words	0.588	0.727	0.700	0.693
Training Recall	+ Other Features	0.586	0.736	0.713	0.706
Test Recall	Bag of words	0.584	0.724	0.695	0.689
Test Recall	+ Other Features	0.584	0.733	0.709	0.703

5) Table 5

		Ridge Classifier	SGD	Random Forest	AdaBoost
Training error mean	Bag of words	0.257	0.258	0.0495	0.348
Training error mean	+ Other Features	0.268	0.250	0.027	0.332
Training error STD	Bag of words	0.0005	0.0004	0.0001	0.0007

Training error STD	+ Other Features	0.0003	0.0009	0.0001	0.001
Test error mean	Bag of words	0.260	0.262	0.282	0.349
Test error mean	+ Other Features	0.270	0.254	0.267	0.333
Test error STD	Bag of words	0.003	0.004	0.001	0.004
Test error STD	+ Other Features	0.003	0.002	0.002	0.002
Training Precision	Bag of words	0.771	0.781	0.969	0.783
Training Precision	+ Other Features	0.752	0.785	0.983	0.763
Test Precision	Bag of words	0.767	0.777	0.723	0.782
Test Precision	+ Other Features	0.750	0.780	0.729	0.762
Training Recall	Bag of words	0.690	0.669	0.930	0.417
Training Recall	+ Other Features	0.690	0.687	0.960	0.486
Test Recall	Bag of words	0.686	0.666	0.702	0.416
Test Recall	+ Other Features	0.688	0.682	0.739	0.484