A Comparative Analysis of Machine Learning Classifiers for Twitter Sentiment Analysis

Heba M. Ismail[1], Saad Harous[1], Boumediene Belkhouche[1]
[1] College of Information Technology,
United Arab Emirates University, Al Ain, UAE
{hebaismail20@gmail.com},{harous, b.belkhouche}@uaeu.ac.ae

Abstract. Twitter popularity has increasingly grown in the last few years making influence on the social, political and business aspects of life. Therefore, sentiment analysis research has put special focus on Twitter. Tweet data have many peculiarities relevant to the use of informal language, slogans, and special characters. Furthermore, training machine learning classifiers from tweets data often faces the data sparsity problem primarily due to the large variety of Tweets expressed in only 140-character. In this work, we evaluate the performance of various classifiers commonly used in sentiment analysis to show their effectiveness in sentiment mining of Twitter data under different experimental setups. For the purpose of the study the Stanford Testing Sentiment dataset STS is used. Results of our analysis show that multinomial Naïve Bayes outperforms other classifiers in Twitter sentiment analysis and is less affected by data sparsity.

1. Introduction

The use of social networking websites such as Twitter and Facebook has been witnessing a rapid growth in the last few years. Probably the reason behind this increase is that people feel comfortable expressing their views and opinions casually on a wide array of topics via such websites. On the other hand, our decision-making process is oftentimes influenced by other people's opinions. Most of us would seek our friends', family members', or co-workers' recommendations before making important purchase decisions, before eating at a specific restaurant, or watching a new movie. Sometimes we even base our decision solely on those opinions. To this end, sentiment analysis has attracted a huge research interest especially in recent years. Researchers analyzed sentiment in many domains: movie reviews, news articles, blogs, forums, product reviews, and more recently social media data. Sentiment analysis of data available on the social networks which comprises of people's views is becoming very important in order to gauge public opinion on a particular topic of interest. It can help evaluate consumer satisfaction about some products, customers' interests and preferences, political viewpoints and many others. Indeed, number of surveys shows that:

- 91% of people visited a store because of an online experience. Among which 22% were influenced by Twitter and Facebook experiences[1]
- 72% of consumers trust online reviews as much as personal recommendations[2]
- 78% of consumers state that posts made by companies on social media influence their purchases[3]

Twitter amongst other social networks is becoming the most popular and influential social network. Every month, millions of people tweet about what they love: products they buy, places they visit, books they are reading, vacations they are planning, and public figures or politicians they like or dislike. Such an enormous amount of public opinions can be of great value. As well as, it can be challenging to identify and engage with the most relevant Tweets about specific topic of interest at the time they are needed. In order to reveal the overall sentiment of the population, retrieval of data from such sources and subsequent sentiment analysis becomes vital.

Sentiment analysis on text is a very difficult task by itself, given the unstructured or in the best cases ill-structured nature of text along with the context complexity [4], let alone extracting sentiment from a text as noisy as social media text. There are some difficulties inherent in analyzing sentiment from social media [5]. One example is "False negatives" where words such as "crying" and "crap" generally suggest negativity, yet they imply positive sentiment when used in a sentence such as "I was crying with joy" or "Holy crap! This is great". Another example is "Conditional sentiment," such as "If someone doesn't call me back, I will never do business with them again." These examples show how sentiment analysis of social media text can be hard. Moreover, the process gets even harder with the use of emoticons such as "." ("smiley) and hash-tags such as "#happy" to express feelings ironically or sarcastically. In addition to the previous and in particular to Twitter, text is usually very short, whereby a maximum Tweet size is 140 characters, and as a consequence, the generated dataset for a specific Twitter corpus may have very large feature space with few values for each Tweet, resulting in a highly sparse dataset that negatively influences the accuracy of the sentiment analysis. These inherent problems in social media text in general and in Twitter in particular impose significant challenges on the sentiment analysis process.

Machine learning classifiers have been widely used for the purpose of sentiment mining providing good accuracy results. Different research studies, reported different accuracy results for unigrams (i.e. distinct words in the corpus) vs bigrams (i.e. combination of every two consecutive words in the text). As well as, different accuracy results were reported for using term frequency vs term presence in the document.

Yet, there is no formal empirical study evaluating the effect of different input representation on the performance of the classifiers. Hence, our study analyzes formally the performance of sentiment classification methods based on fair experimental setups. We analyze unigrams, as well as, bigrams as features spaces. For example for a tweet "I Love Kindle, It's Amazing", unigrams = {I, Love, Kindle, Its, Amazing}, bigrams = {I Love, Love Kindle, Kindle Its, Its Amazing}. Moreover, we analyze term frequency representation of dataset (i.e. the number of occurrences of a term in a document), as well as, term presence representation (i.e. the occurrence or absence of a term in a document regardless of how many times it occurred). For training and testing we are using Stanford Testing Dataset. Details about the experimental setups are provided in section VI. In the following section we review some related works to ours then we present a brief overview of sentiment analysis and highlight the major areas of research in sentiment analysis. Section IV presents some commonly used classifiers in sentiment analysis. Finally we present our experiment setup and results.

2. Related Work

In the literature there are few studies that attempted to empirically evaluate the performance of classification algorithms in sentiment mining. Vinodhini and Chandrasekaran [6] conducted a comparative study on four classifiers: K-Nearest Neighbors, Decision Trees, Naïve Bayes and Support Vector Machines, to evaluate their performance in sentiment mining of online product reviews. They used different sampling methods (e.g. linear sampling, bootstrap sampling and random sampling) to create training examples from the product reviews dataset. Their results show that support vector machine with bootstrap sampling method outperforms other classifiers and sampling methods in terms of misclassification rate. They used unigrams for feature space and terms occurrences to populate the classification input. They did not provide any information about the influence of input format on the classification results.

On the other hand, Hang et al. [7] evaluated the performance of three classifiers: Passive-Aggressive (PA) Algorithm Based Classifier, Language Modeling (LM) Based Classifier and Winnow, using 100K online product reviews with focus on the impact of higher order n-grams ($n > 3$). They found that discriminating classifier (i.e. Passive-Aggressive Based) combined with high order n-grams as features can achieve comparable, or better performance than that reported in academic papers. Hang et al study analyzed up to 6-grams feature length. However, they did not show the impact of data representation (e.g. frequency, occurrences) on the performance of the classifiers.

Furthermore, Vinodhini's and Hang's studies were conducted on product reviews which may have length up to 800 characters or more. However, a Tweet is limited to 140 character which adds another challenge to sentiment mining of Twitter. Normally datasets generated from Twitter suffer from large sparsity. Higher order n-grams may not be suitable to use as a Tweet may have 6, 5 or 4 words only which are used by Hang's as n-grams features. Given the special peculiarities of twitter text and length, a twitter-specific comparative study is needed to evaluate the performance of popular classification algorithms in the area of sentiment mining using different input formats which actually have the direct impact on the classification accuracy.

3. Sentiment Analysis

Sentiment mining, polarity mining, opinion mining or sentiment analysis is concerned with analysis of direction-based text, i.e. text containing opinions and emotions. Sentiment analysis involves many tasks. Four of the important tasks of sentiment analysis where most of the research effort is focused are: data preprocessing, class labeling, annotation granularity, and target identification [8]. Data preprocessing is vital especially for the text collected from social media websites because it is unstructured and full of spelling mistakes and peculiarities. All researchers in the area of sentiment analysis perform some or all of the natural language preprocessing tasks including: spellchecking, and stop words removal such as punctuation marks. In addition, some researchers perform stemming before classification [9] [10]. In class labeling process (i.e. the process of annotating text into labels or classes) some

research focuses on categorizing text as subjective or objective. In sentiment analysis, this task is usually carried out first, because it was proven that performing it prior to polarity classification improves the latter [4]. In other words, if a text is identified as subjective then we can perform polarity classification to determine whether this subjective text is carrying positive sentiment or negative sentiment. On the other hand, a large body of research focuses on automating the process of class labeling through distant supervision using noisy labels.

For example, [11] used emoticons such as ":-)" and ":(" to label tweets as positive or negative. However, [12] argued that using noisy sentiment labels may hinder the performance of sentiment classifiers. They proposed exploiting the Twitter follower graph to improve sentiment classification and constructed a graph that has users, tweets, word unigrams, word bigrams, hashtags, and emoticons as its nodes which are connected based on the link existence among them (e.g., users are connected to tweets they created; tweets are connected to word unigrams that they contain etc.). Then they applied a label propagation method where sentiment labels were propagated from a small set of nodes seeded with some initial label information throughout the graph. Having a pre-processed subjective text with class labels, sentiment classification can be conducted at the document [13], sentence [14] or phrase levels [15] (where a phrase is part of a sentence) which we refer to as the granularity of the classification. Finally, knowing the source and the target of a sentiment is considered as one of the challenges of sentiment analysis that was addressed by number of researchers [16].

4. Machine Learning Classifiers For Sentiment Analysis

The two most commonly used approaches in sentiment analysis techniques are: the lexicon-based approach and the learning approach [17]. Lexicon based approaches are used widely to classify unsupervised text sentiment. Such classifiers attempt to classify data on the number of positive and negative words present in the text, and do not need any training dataset. These words which express opinion are known as "opinion words" and the lexicon is known as "opinion lexicon". Basically in the lexicon based approaches we rely on external lexical resources that associate polarity score to each term. Sentiment of text depends on the sentiment of the terms that compose it. Examples of lexical dictionaries are: (i) SentiWordNet, (ii) WordNet Affect, (iii) Sentic Net and (iv) MPQA. The major problem with this approach is that there is no mechanism to deal with context dependent words. For example, the word," Long" can be used to convey a positive as well as a negative opinion both depending upon the context in which it is used. For example, we can think of two sentences as "This mobile takes long time to charge" which is a negative opinion, whereas saying "This mobile phone has long battery life" is a positive opinion. On the other hand, classification approaches involve building classifiers from labeled instances of texts or sentences, essentially a supervised classification task. In our research we focus on classification learning approaches for sentiment analysis. In the following sections we explore some of the most commonly used machine learning classifiers for sentiment analysis

4.1. Naïve Bayes Classifiers

Naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem. The descriptive attributes/features are assumed to be conditionally independent from each other, which makes a naïve assumption [18]. Typically, due to the independence assumption, the class-conditional probability for an object X (i.e. which is a record or a row in the dataset), is estimated as the product of all independent events' (i.e. Features' Values, X1, X2, X3 …. Xd) conditional probabilities for a given class Y, such that:

$$P(X \mid Y = y) = \prod_{i=1}^{d} P(Xi|Y=y)$$

Therefore, for predicting a class Y:

$$P(Y = y \mid X) = P(Y = y) \left(\prod_{i=1}^{d} P(Xi|Y=y) \right) / P(X)$$

Since P(X) is a common denominator for all class prediction calculations for a single record (X), it does not affect the choice of the class; therefore we can replace the previous formula with the following:

$$P(Y = y \mid X) = P(Y = y) \left(\prod_{i=1}^{d} P(Xi|Y=y) \right)$$

Major strengths of naïve Bayes classifier are: handling noisy data since it is averaged out in the estimation of conditional probability, null values are ignored and irrelevant features are uniformly distributed so they do not have significant influence on the classification result. Weaknesses are mainly attributed to the assumption of complete independence amongst attributes. If there are no occurrences of a class label and a certain attribute value together (e.g. class="nice", shape="sphere") then the frequency-based probability estimate will be zero. Given Naive-Bayes' conditional independence assumption, when all the probabilities are multiplied we will get zero and this will affect the posterior probability estimate. This problem happens when we are drawing samples from a population and the drawn vectors are not fully representative of the population. Lagrange correction and other schemes have been proposed to avoid this undesirable situation. There are several Naive Bayes variations. Here we will consider two of them: the Multinomial Naive Bayes, and the Bernoulli Naïve Bayes

4.2 Multinomial Naïve Bayes Text Classifiers

Using the Multinomial Naïve Bayes Text Classifier, the probability of a document d being in class c is computed as [19]:

$$P(c|d) \propto P(c) \prod_{1 \le k \le n_d} P(t_k|c)$$

where P(tk|c) is the conditional probability of term tk occurring in a document of class c. We interpret P(tk|c) as a measure of how much evidence tk contributes that c is the correct class. P(c) is the prior probability of a document occurring in class c. If a

document's term does not provide clear evidence for one class versus another, we choose the one that has a higher prior probability. (t1, t2, . . . , tnd) are the tokens in d that are part of the vocabulary we use for classification and nd is the number of such tokens in d. For example, (t1, t2, . . . , tnd) for the one-sentence document "Beijing and Taipei join the WTO" might be (Beijing, Taipei, join, WTO), with nd = 4, if we treat the term "and" as a stop word. In text classification, our goal is to find the best class for the document. The best class in Naïve Bayes classification is the most likely or maximum posteriori (MAP) class cmap:

$$c_{\text{map}} = \arg\max_{c \in \mathbf{C}} \hat{P}(c|d) = \arg\max_{c \in \mathbf{C}} \hat{P}(c) \prod_{1 \le k \le n_d} \hat{P}(t_k|c).$$

P^(c) is calculated by finding the frequency of class c relative to the total size of the given training data such that:

$$\hat{P}(c) = \frac{N_c}{N},$$

where Nc is the number of documents in class c and N is the total number of documents. P^(tk|c) is calculated by finding the number of occurrences of t in training documents from class c, including multiple occurrences of a term in a document such that:

$$\hat{P}(t|c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}},$$

where Tct is the number of occurrences of t in training documents from class c, including multiple occurrences of a term in a document. In the implementation of Multinomial Naïve Bayes (MNB) we need to add a smoothing one to the conditional probability so as to avoid zero probability of new terms in the testing set that were not available in the training set:

From the pseudocode in figure 1, we see that the complexity of the training process is Θ (|C||V|) because the set of parameters consists of |C||V| conditional probabilities and |C| priors. The time complexity for text pre-processing will be Θ (|D| * Lavg) (i.e. number of documents times the average length of documents) [19]. In this study we do not consider the pre-processing time since we are using the classifier as an independent program not including text pre-processing.

4.3 Bernoulli Naïve Bayes Text Classifiers

An alternative to the multinomial model is the multivariate Bernoulli model or Bernoulli model, which generates an indicator for each term of the vocabulary, either 1 indicating presence of the term in the document or 0 indicating absence. Figure 2 shows the training and testing algorithms for the Bernoulli model. This model estimates P^(t|c) as the fraction of documents of class c that contain term t. The Bernoulli model has the same time complexity as the multinomial model.

```
TRAINMULTINOMIALNB(C, D)
 1  V ← EXTRACTVOCABULARY(D)
 2  N ← COUNTDOCS(D)
 3  for each c ∈ C
 4  do N_c ← COUNTDOCSINCLASS(D, c)
 5     prior[c] ← N_c / N
 6     text_c ← CONCATENATETEXTOFALLDOCSINCLASS(D, c)
 7     for each t ∈ V
 8     do T_ct ← COUNTTOKENSOFTERM(text_c, t)
 9     for each t ∈ V
10     do condprob[t][c] ← (T_ct + 1) / Σ_t'(T_ct' + 1)
11  return V, prior, condprob

APPLYMULTINOMIALNB(C, V, prior, condprob, d)
1  W ← EXTRACTTOKENSFROMDOC(V, d)
2  for each c ∈ C
3  do score[c] ← log prior[c]
4     for each t ∈ W
5     do score[c] += log condprob[t][c]
6  return arg max_c∈C score[c]
```

Figure 1. Naïve Bayes Multinomial Algorithm [19]

```
TRAINBERNOULLINB(C, D)
1  V ← EXTRACTVOCABULARY(D)
2  N ← COUNTDOCS(D)
3  for each c ∈ C
4  do N_c ← COUNTDOCSINCLASS(D, c)
5     prior[c] ← N_c / N
6     for each t ∈ V
7     do N_ct ← COUNTDOCSINCLASSCONTAININGTERM(D, c, t)
8        condprob[t][c] ← (N_ct + 1)/(N_c + 2)
9  return V, prior, condprob

APPLYBERNOULLINB(C, V, prior, condprob, d)
1  V_d ← EXTRACTTERMSFROMDOC(V, d)
2  for each c ∈ C
3  do score[c] ← log prior[c]
4     for each t ∈ V
5     do if t ∈ V_d
6           then score[c] += log condprob[t][c]
7           else score[c] += log(1 − condprob[t][c])
8  return arg max_c∈C score[c]
```

Figure 2. Bernoulli Naive Bayes Algorithm [19]

4.4. Support Vector Machines Classifiers

Support Vector Machine (SVM) is a non-probabilistic binary linear classifier that constructs a hyperplane or set of hyperplanes in a high or infinite dimensional space, which can be used for classification, regression, or other tasks. The main idea underlying SVM for sentiment classification is to find a hyper plane which divides the documents, or in our case, tweets as per the sentiment, and the margin between the classes being as high as possible [17]. For example, if we have a training set expressed mathematically as follows:

$$D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_m, y_m)\}$$

where xi is an n-dimensional real vector (i.e. document or a tweet in our case), yi is either 1 or -1 denoting the class to which the point xi belongs. First, The SVM classification function F(x), must return positive numbers for positive data points and negative numbers otherwise, that is, for every point xi in D. Second, F(x) (or the hyperplane) needs to maximize the margin. The margin is the distance from the hyperplane to the closest data points or vector (i.e. which will be called the support vector). This turns the SVM classifier into an optimization constraint problem. Solving this problem using Lagrange multipliers, the solution can be written as [20]:

$$F(\mathbf{x}) = \sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} - b$$

where the auxiliary nonnegative variable α is called Lagrange multipliers, and b is the bias, which will be computed by the SVM in the training process. Note that according to the property of Kuhn–Tucker conditions of optimization theory, the solution of the dual problem α must satisfy the following condition:

$$\alpha_i^* \{y_i(\mathbf{w}^* \cdot \mathbf{x}_i - b) - 1\} = 0 \quad \text{for } i = 1, 2, \ldots, m$$

and either α or its corresponding constraint {yi (w.xi – b) -1} must be nonzero. This condition implies that only when xi is a support vector or {yi (w.xi – b) -1} = 1, its corresponding coefficient αi will be nonzero (or nonnegative). After exploring the theoretical background of SVM, we understand that it is not an algorithm, but rather a mathematical relationship which leads to an optimization problem. This problem obviously requires an optimization algorithm to solve.

4.5 Sequential Minimal Optimization

Sequential minimal optimization (SMO) is an algorithm for solving the quadratic programming (QP) problem that arises during the training of support vector machines. SMO is widely used for training support vector machines and is implemented by popular data mining tools such as Weka.

SMO breaks the SVM optimization problem into a series of smallest possible sub-problems, which are then solved analytically. Because of the linear equality constraint involving the Lagrange multipliers αi, the smallest possible problem involves two such multipliers. Then, for any two multipliers α_1 and α_2, the constraints are reduced t_0:

$$0 \leq \alpha_1, \alpha_2 \leq C,$$

$$y_1\alpha_1 + y_2\alpha_2 = k,$$

where C is an SVM hyper-parameter, and k is the negative of the sum over the rest of terms in the equality constraint, which is fixed in each iteration. This reduced problem can be solved analytically: one needs to find a minimum of a one-dimensional quadratic function. The training algorithm proceeds as follows:

- Find a Lagrange multiplier α1 that violates the Karush–Kuhn–Tucker (KKT) conditions for the optimization problem
- Pick a second multiplier α2 and optimize the pair (α1, α2),
- Repeat steps 1 and 2 until convergence

When all the Lagrange multipliers satisfy the KKT conditions (within a user-defined tolerance), the problem is then solved. Although this algorithm is guaranteed to converge, heuristics are used to choose the pair of multipliers so as to accelerate the rate of convergence. This is critical for large data sets since there are n*(n − 1) possible choices for αi and αj. In the worst case the algorithm has complexity of Θ (n3), where n is the number of vectors.

5. Proposed Method

In this comparative study we need to evaluate the performance of Multinomial NB, Bernoulli NB and SVM in sentiment mining of Twitter data. The selected classifiers are the most commonly used machine learning classifiers in the literature [11], [12], [17], [21], [22]. For comparison we use a selected Twitter dataset, apply suitable preprocessing steps then produce the dataset with unigrams and bigrams, one time with term frequencies and one time with term presence (i.e. polarity dataset). Afterwards, the three selected classifiers are trained with the four variations of the input dataset and the accuracy results are compared along with training time.

6. Experimental Setup

Our experimental setup is as follows.

6.1. Dataset

In the work conducted in this paper, we use the Stanford Twitter Sentiment Data which was collected between the 6th of April and the 25th of June 2009 [11]. The original test set consists of 177 negative and182 positive manually annotated tweets.

6.2. Pre-Processing and Feature Reduction

Natural language processing of the corpus is performed for stop words removal, bag of words extraction and equivalence classes' replacement such that:

- All Twitter usernames, which start with @ symbol, are replaced with the term "USERNAME".
- All URL links in the corpus are replaced with the term "URL"
- Reduce the number of letters that are repeated more than twice in all words. For example the word "haaaappy" becomes "haappy" after reduction
- Remove all Twitter hashtags which start with the#.
- Remove all emoticons as they add noise during the training of the classifiers

We choose unigrams (i.e. distinct words in the corpus), as well as, bigrams (i.e. combination of every two consecutive words in the text) as features spaces. For example for a tweet "I Love Kindle, It's Amazing", Unigrams would be {I, Love, Kindle, Its, Amazing}, whereas bigrams would be {I Love, Love Kindle, Kindle Its, Its Amazing}. Consequently, bigrams normally produces larger feature space.

6.3. Performance Evaluation Steps

After pre-processing is done, four different variations of the input dataset are produced:

- Unigram with term polarity.
- Unigram with Term Frequency.
- Bigrams with term polarity.
- Bigrams with term Frequency.

We choose Weka for evaluating the performance of the selected classifiers as it has exactly similar implementation to the one discussed in this paper for MNB, BNB and SMO. For one iteration, the dataset inputs are used to train the classifiers and results are verified using 10-fold cross validation. For the second iteration, only 66% of the dataset is used for training whereas the remaining is used for testing.

7. Results and Discussion

Table 1 shows the experimental classification results for Bernoulli Naïve Bayes, Multinomial Naïve Bayes and SOM classifiers with 10-fold cross validation. The results show that overall accuracy for unigrams datasets are higher than the accuracy for bigrams datasets. Furthermore, training time for unigrams dataset is in general less than bigrams. This is expected since bigrams produce larger feature space. Multinomial Naïve Bayes produced the best classification results with frequency, unigrams dataset. SOM requires the longest training time to build the model and does not outperform other classifiers in the context of sentiment analysis of Twitter, which makes it less preferable choice for sentiment analysis compared to multinomial NB that produces good accuracy results at very high training speed.

Table 2 shows the experimental classification results for Bernoulli Naïve Bayes, Multinomial Naïve Bayes and SOM classifiers with STS set divided into training set and testing set. The first observation is that training time did not significantly change. This means that using either method, cross validation or training set would take comparable training time for model building. For MNB and BNB the classification

results of bigrams with training outperformed the classification results with cross validation. For Bernoulli NB the accuracy of unigrams dropped compared to significant increase in the performance of SOM. However, overall accuracy results for unigrams still outperform bigrams. This is expected as well, since bigrams produce datasets that are sparser given the limit of 140 character of Twitter. Moreover, multinomial NB still outperforms other classifiers. We deduce that MNB is less affected by the data sparsity problem inherent in Twitter datasets.

Table 1. Testing Results -10-Fold Cross Validation

Classifier	Dataset Type	Unigrams (1442 Features)	Time (Sec)	Bigrams (4150 Features)	Time (Sec)
BNB	**Polarity**	**76.6%**	**0.22**	**70.75 %**	**0.61**
	Frequency	**75.21 %**	**0.24**	**65.18 %**	**0.61**
MNB	**Polarity**	**79.39 %**	**0.13**	**75.77 %**	**0.13**
	Frequency	**81.34 %**	**0.05**	**72.14 %**	**0.11**
SVM	**Polarity**	**74.37 %**	**4.34**	**74.09 %**	**12.16**
	Frequency	**77.16 %**	**4.22**	**69.95 %**	**12.95**

Table 2. Testing Results - Dataset split into 66% training set Conclusion

Classifier	Dataset Type	Unigrams (1442 Features)	Time (Sec)	Bigrams (4150 Features)	Time (Sec)
BNB	**Polarity**	**73.76%**	**0.22**	**73.77 %**	**0.59**
	Frequency	**73.77 %**	**0.25**	**68.03%**	**0.59**
MNB	**Polarity**	**82.78%**	**0.13**	**80.32 %**	**0.11**
	Frequency	**80.32 %**	**0.05**	**77.86%**	**0.09**
SVM	**Polarity**	**79.50 %**	**4.34**	**72.95 %**	**12.23**
	Frequency	**80.32%**	**4.22**	**66.39%**	**12.42**

8. Future Work

For future work, we would like to conduct our experiment on a larger more representative dataset.

In addition, sentiment cannot be separated from semantic. Counting words or recognizing the polarity of certain terms without making sense of the semantic may hide lots of information. Some positive terms can be used ironically to express negative ideas and some negative terms can be used informally to express extreme positive emotions. In such scenarios semantic means a lot. Some research efforts were made to incorporate the semantic in training the classifiers by means of using sentiment and semantic topics [22]. In the future we will consider incorporating semantic features in our evaluation of classification algorithms for Twitter sentiment analysis.

9. Conclusion

Twitter is one of the most popular social networks where users can tweet about different topics within the 140-character limit. This small size of text imposes a significant challenge to Twitter sentiment analysis since tweets datasets are often too sparse.

In this paper, we have designed an evaluation method for evaluating the effect of different input representations and formats on the performance of the classifiers. Hence, we provided formal performance evaluation of sentiment classification based on fair experimental setups.

The experimental results show that Multinomial Naïve Bayes classifier outperformed other classifiers examined in the study in the context of Twitter sentiment analysis being less affected by the sparsity of Twitter dataset. Unigrams as a form of representing dataset feature proved to be more effective in the context of Twitter sentiment analysis as they produce less sparse datasets. From our experiments, we could not get proof on best choice for frequency vs. polarity representation of data. Finally, despite the strong capabilities of SVM, it generated the least accuracy results taking the longest processing time, it proved to be negatively affected by data sparsity, making it less preferable choice for Twitter sentiment analysis.

For future work, we would like to expand the scope of our experiments and run the classifiers on more than one dataset considering number of different languages in order to have more representative inputs and thus better generalizable results.

References

1. Sterling, G., "Survey: 91 Percent Have Gone Into Stores Because Of Online Promotion," Marketing Land, December 2012. [Online]. Available: http://marketingland.com/survey-91-percent-have-gone-into-stores-because-of-online-promotion-28796. [Accessed November 2015].
2. DeMers, J., "How Social Media is Changing the SEO Industry," Search Engine Journal, April 2013. [Online]. Available: http://www.searchenginejournal.com/how-social-media-is-changing-the-seo-industry/63060/. [Accessed November 2015].
3. Olensky, S., "Are Brands Wielding More Influence In Social Media Than We Thought?," Forbes, May 2012. [Online]. Available: http://www.forbes.com/sites/marketshare/2012/05/07/are-brands-wielding-more-influence-in-social-media-than-we-thought/. [Accessed November 2015].
4. Pang, B., & Lee, L, "Opinion Mining and Sentiment Analysis," *Foundations and Trends in Information Retrieval,* vol. 2, no. 2, p. 1–135, 2008.
5. Henschen, D., "Seven Shades of Sentiment," *Information Week 1337,* 2012.
6. Vinodhini, G., & Chandrasekaran, R. M, "Performance Evaluation of Machine Learning Classifiers in Sentiment Mining," *International Journal of Computer Trends and Technology (IJCTT),* vol. 4, no. 6, 2013.
7. Cui, H., Mittal, V., & Datar, M, "Comparative Experiments on Sentiment Classification for Online Product Reviews," *AAAI* , vol. 6, pp. 1265-1270, 2006.

8. Abbasi, A., Chen, H., & Salem, A., "Sentiment Analysis in Multiple Languages: Features selection for Opinion Classification in Web Forums," *ACM Transactions on Information Systems (TOIS)*, vol. 26, no. 3, pp. 1-34, 2008.
9. Bollen, J., Pepe, A., & Mao, H, "Modeling Public Mood and Emotion: Twitter Sentiment and Socio-Economic Phenomena," in *International AAAI Conference on Weblogs and Social Media*, Barcelona, 2009.
10. Jiang, L., Yu, M., Zhou, M., Liu, X., & Zhao, T, "Target-dependent Twitter Sentiment Classification," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, 2011.
11. Go, A., Bhayani, R., & Huang, L, "Twitter Sentiment Classification using Distant Supervision," Stanford Universty Press, Stanford, 2009.
12. Speriosu, M., Sudan, N., Upadhyay, S., & Baldridge, J, "Twitter Polarity Classification with Label Propagation over Lexical Links and the Follower Graph," in *Proceedings of the First workshop on Unsupervised Learning in NLP*, UK, 2011.
13. Chaovalit, P., & Zhou, L, "Movie Review Mining: a Comparison between Supervised and Unsupervised Classification Approaches," in *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, Hawaii, 2005.
14. Elhawary, M., & Elfeky, M, "Mining Arabic Business Reviews," in *Data Mining Workshops (ICDMW)*, Sydney, 2010.
15. Wilson, T., Wiebe, J., & Hoffmann, P, "Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis," in *In Proceedings of the conference on human language technology and empirical methods in natural language processing*, Vancouver, 2005.
16. Perez-Tellez, F., Pinto, D., Cardiff, J., & Rosso, P, "On The Difficulty of Clustering Company tweets," in *Proceedings of the 2nd international workshop on Search and mining user-generated contents*, New York, 2010.
17. Bhuta, S., Doshi, A., Doshi, U., & Narvekar, M, "A Review of Techniques for Sentiment Analysis of Twitter Data," in *Issues and Challenges in Intelligent Computing Techniques (ICICT)*, 2014.
18. Hongbo, D., Data Mining Techniques and Applications An Introduction, C&C Offset China, 2010.
19. Manning, C. D., Raghavan, P., & Schütze, H, Introduction to information retrieval, Cambridge : Cambridge University Press, 2008, pp. 253-287.
20. Yu, H., & Kim, S., "SVM Tutorial—Classification, Regression and Ranking.," Springer Berlin Heidelberg in Handbook of Natural Computing, 2012.
21. Bo Pang, Lillian Lee, Shivakumar Vaithyanathan, "Thumbs up: Sentiment Classification Using Machine Learning Techniques," Association for Computational Linguistics, 2002.
22. Saif, H., He, Y., & Alani, H, "Alleviating Data Sparsity for Twitter Sentiment Analysis," in *CEUR Workshop Proceedings (CEUR-WS. org)*, Lyon, France., 2012.