An Enhanced Ensemble Classifier Framework for Sentiment Analysis of Social Media Issues

A Thesis

Submitted to the Faculty of Graduate Studies and Research

In Partial Fulfillment of the Requirements

For the Degree of

Master of Applied Science

In

Software Systems Engineering

University of Regina

by

Talha Ahmed Khan

Regina, Saskatchewan

July, 2015

UNIVERSITY OF REGINA

FACULTY OF GRADUATE STUDIES AND RESEARCH

SUPERVISORY AND EXAMINING COMMITTEE

Talha Ahmed Khan, candidate for the degree of Master of Applied Science in Software Systems Engineering**,** has presented a thesis titled, ***An Enhanced Ensemble Classifier Framework for Sentiment Analysis of Social Media Issues,*** in an oral examination held on July 10, 2015. The following committee members have found the thesis acceptable in form and content, and that the candidate demonstrated satisfactory knowledge of the subject material.

External Examiner:	Dr. JingTao Yao, Department of Computer Science
Co-Supervisor:	Dr. Craig Gelowitz, , Software Systems Engineering
Co-Supervisor:	Dr. Luigi Benedicenti, Software Systems Engineering
Committee Member:	*Dr. Mohamed El-Darieby, Software Systems Engineering
Committee Member:	Dr. Raman Paranjape, Electronic Systems Engineering
Chair of Defense:	Dr. Mohamed Ismail, Industrial Systems Engineering

*Not present at defense

Abstract

Sentiment Analysis is the study of determining an author's opinion from written text using artificial intelligence and data mining techniques. In this thesis, three different sentiment analysis techniques; Naïve Bayes Classification, Support Vector Machine and Ensemble Classification are studied and applied to social media datasets for extracting opinions.

One of the uses of sentiment analysis is to act as a feedback mechanism to aid in decision making. In this thesis a Probabilistic Feature Weighting (PFW) technique is proposed using the principle of the Naïve Bayes Classifier and Bayesian Probability. The PFW helps in ranking the documents into further sub-categories and is useful to compare features and their importance in sentiment classification.

An Enhanced Ensemble Classifier Framework (EECF) is also developed based on the PFW technique. The Enhanced Ensemble Classifier increases the accuracy of the system compared to the existing techniques. Social media documents consist of a smaller number of words and often lack formal use of language. As such, social media requires more sophisticated techniques to establish sentiment. EECF helps in classifying shorter documents that have a smaller number of features such as Twitter posts. The development of the Enhanced Ensemble Classifier is a contribution in the sentiment analysis domain.

The proposed PFW technique provides an alternative method to investigate features and classify sentiment into sub-categories beyond positive and negative

sentiment. The Enhanced Ensemble Classifier that utilizes the PFW is shown to improve the determination of sentiment.

Acknowledgements

I would like to express my sincere gratitude to my co-supervisor, Dr. Craig M. Gelowitz, for providing me an excellent opportunity and atmosphere to conduct this research. This would have not been possible without his support, care and guidance. I would like to thank my supervisor, Dr. Luigi Benedicenti, for teaching me the methods of conducting scientific research and providing technical reviews during the research. I would like to thank Petroleum Technology Research Centre for funding parts of this research. I would like to thank Faculty of Graduate Studies & Research for providing me scholarship and funding during my studies at the University of Regina. I would like to thank my teachers for transferring their knowledge due to which I was able to conduct this research. I would also like to thank my friends and family members, especially my mother and my wife who supported me emotionally through every phase of life.

Table of Contents

List of Figures

1 Introduction

Sentiment Analysis is the study of determining an author's opinion from written text using artificial intelligence and data mining techniques. For social networks, messages posted can be about virtually anything including products, celebrities or any other topic. There are a variety of ways to classify documents ranging from supervised learning to unsupervised learning [18]. Supervised algorithms make use of pre-classified data to extract mathematical functions that can predict class of new documents whereas, unsupervised algorithms does not use labeled data. They make use of clusters and patterns in the data to predict new documents.

Unsupervised learning is difficult and time consuming because it requires one to carefully design and implement a variety of rules in order to achieve desired results. Supervised learning, on the other hand, use labeled training data to infer a function that can determine sentiment and has been used successfully for text classification and opinion mining.

Figure 1 shows the general representation of a Class and its features. C denotes the class and A1, A2, A3 & A4 are the features in the Class C.

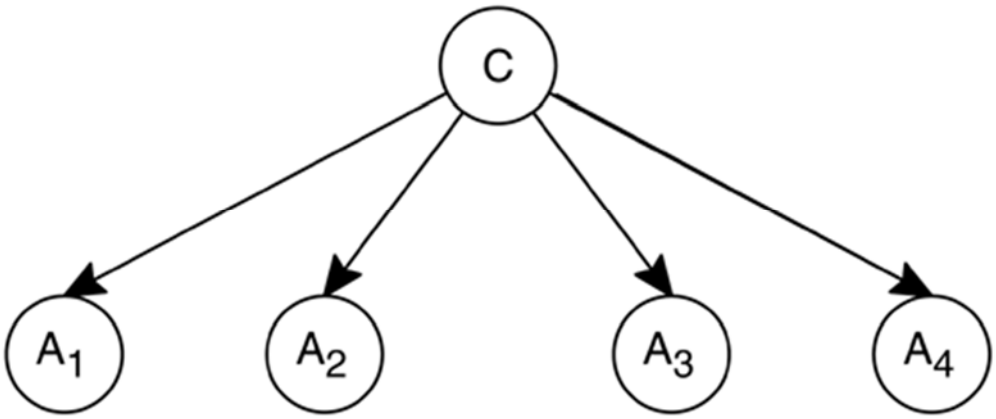

Figure 1: General Representation of Class and Features [70]

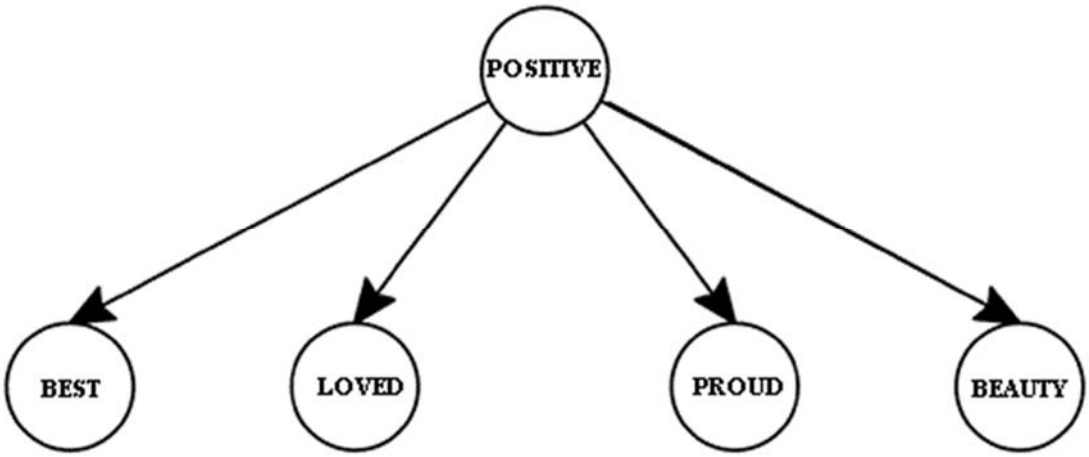

Figure 2: Example of Sentiment Analysis & Features

In order to classify text, feature selection methods, i.e. statistical measures to select best features out of data, is often used to select a subset of features from a dataset that can be used for training purposes. An example of features in a Positive class in Sentiment Analysis is shown in Figure 2. Features are then quantized to construct a model for machine learning [3]. The model varies with the type of classification that is being performed and may have a significant impact on the accuracy of the classifier. Feature extraction techniques may also help to determine the accuracy of the analysis [4].

1.1 Motivation

Social networking has become a primary mechanism for Internet users to share their opinions, feelings and ideas. Individuals often utilize social media for responding to events or providing opinions on particular topics. This public discourse can be useful when utilized as a feedback mechanism for industry or governments that wish to track the evolution of the public conscience on a given topic [1].

Many communities have been formed over social networks throughout the world. The interest in social media has increased dramatically over the past ten years. For example, Twitter currently has more than 550 million users and over 1 billion users utilize Facebook each month [2]. Social media networks are not only popular among individuals; they have also been used extensively for business related purposes as millions of websites have integrated social networking. This has been accomplished through APIs that have been made available to both publish and gather social network data by popular social media platforms such as Facebook and Twitter.

1.2 Research Objectives

The objectives of this research are to:

- Study Support Vector Machines, Naïve Bayes Classifier and Ensemble Classification techniques for sentiment analysis and apply them to social media datasets.
- Develop a classification model that can extract the sentiment of users from their posts on social media.

- Investigate the kind of noise found in social media posts and removing them from the dataset.
- Develop a Feature Weighting technique for sentiment analysis.
- Develop an Ensemble Classifier for classification of short documents such as social media posts which have a small number of features.
- Develop a ranking algorithm for ranking the documents from most positive to most negative sentiment.
- Test the proposed model on social media datasets and compare it with existing techniques.

1.3 Research Contributions

The research contributions in the area of sentiment analysis for social media are:

- The development of a Probabilistic Feature Weighting (PFW) methodology for sentiment analysis.
- The development of an Enhanced Ensemble Classification Model that helps in classifying documents with fewer features such as social media posts.
- The development of a document ranking algorithm that utilizes normalized probability scores to rank a document's degree of sentiment.

1.4 Thesis Organization

Chapter 1 contains the introduction of Sentiment Analysis, the motivation and the objectives of this research. Chapter 2 provides a literature review of sentiment analysis and some of the work done in this area. It also describes the working principle of Naïve Bayes Classifiers, Support Vector Machines and Ensemble Classifiers. Chapter 3

describes the proposed Probabilistic Feature Weighing (PFW) Model for feature weighting, the development process of Enhanced Ensemble Classifier Framework (EECF) and the algorithm for document ranking. Chapter 4 provides test results and a comparative analysis of the EECF. Chapter 5 provides a conclusion and the potential future work for this research.

2 Literature Review

Sentiment Analysis has been extensively studied by researchers in the past decade due to the evolution of social media platforms and the ever growing number of users of these platforms on the Internet. Researchers have used various machine learning techniques to make it possible for a computer to identify good, bad, positive or negative sentiment from any given electronic documents. Pang & Lee provided a broad overview of the problem and have worked on various aspects of sentiment analysis techniques [3].

Machine learning techniques have been used for the purpose of classifying text documents according to their sentiment [8]. The most widely used technique for sentiment analysis is supervised machine learning. Pak & Paroubek studied the behaviour of the Naïve Bayes classifier using a Twitter corpus and reported that Bayesian analysis works well when compared to Support Vector Machine analysis [7]. Frank & Bouckacrt studied the Naïve Bayes classifier by varying the number of words present in each class and concluded that unbalanced classes affect the performance of the classifier in a negative way. They also proposed a centroid based class normalization technique to counter the problem. The altered classifier works on two out of four datasets used in the experiment but they also state that by altering the values of the parameters in the algorithm , the desired results can be observed [9].

Kamaruzzaman & Rahman conducted a study to find the performance of the Naïve Bayes classifier using an association concept between features which is opposite to the feature independence assumption. The result of the study shows that association

mining reduces the number of features significantly which affects the performance of the classifier and decreases its accuracy [10]. M. Karamibekr, Ali A. Ghorbani, proposed a sentiment analysis model and carried out tests by classifying documents into three different levels (document level, sentence level and phrase level). They used verbs and nouns to identify the polarity of a document using parts of speech tagging in the sentences. The accuracy reported based on this model was 65% [11]. A similar approach was also utilized by Neri, Aliprandi, Capeci & Cuadros, where the researchers used "parts of speech" tagging to find the polarity of a sentence by analyzing the complete architecture of the sentence rather than just the overall polarity. The precision and the recall of the system were found to be 83% and 87% respectively [12].

Istvan Pilaszy concluded that a text categorization system may have many parameters and it is often not clear how to set up those parameters. It significantly depends on the type of the text categorization problem and the datasets available to train the system. According to Pilaszy, each and every problem is unique and needs to be handled differently. Some datasets have only a few words in the vocabulary where some datasets have only a few classes but a large number of instances. Different strategies are required to pre-process the data in order to achieve the desired results [13]. Machine Learning techniques are simpler and more efficient than symbolic techniques; however, there are certain issues that are difficult to handle such as misspellings and slang words [14].

Ensemble Techniques have sometimes shown improvement in the task of sentiment analysis where two or more algorithms are used to obtain more reliable results. This does not always work well as shown in the research carried out by Neethu &

Rajasree where the accuracy decreased when compared to the accuracy of individual algorithms [14]. However Hassan, Abbasi, & Zeng developed a bootstrap ensemble framework that utilizes an iterative approach to classify the documents. The results of their research showed that the bootstrap ensemble framework is much more efficient with higher accuracy. The performance was also balanced throughout the classes [15].

2.1 Naïve Bayes

A Naïve Bayes classifier is a probabilistic classifier which applies the Bayesian rule with a strong (naïve) assumption [16]. The Naïve Bayes Classifier assumes that a feature from the data set is independent of the other features in a given class. For example, a fruit which is green in color, has a spherical shape, and is 10 to 12 inches in diameter maybe a watermelon. Although these features depend on each other, the Naïve Bayes classifier will consider each feature as independent of the other features that contribute to the probability that the fruit is watermelon.

The Naïve Bayes classifier uses Bayesian statistics to find out the class of an instance. At first, the posterior probability of each class is calculated given the feature values present in the instance. The instance is then assigned to the class that has the highest probability [17]. Equation 1 is the Naïve Bayes formula:

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

Equation 1 indicates that the features are independent of each other within each class. This equation is used to find the posterior probability of the class given the feature values. The term P(c) on the right side of the equation denotes the prior probability of the

class. This probability depends on the number of training instances in a class. For example, if there is one class out of two possible classes to which a new document is to be classified and the number of training instances are 3 and 6 in class A and B respectively, the probability of the class A will be 1/3, whereas the probability of class B will be 2/3 [18].

Similarly, term P (t_k|c) in Equation 1 provides the evidence of how much the term t_k contributes to class C being the correct class of the document. If the document's terms do not provide clear evidence of the class, we choose the one with the highest prior probability [18].

2.1.1 Multinomial Naïve Bayes Model

The Multinomial Naïve Bayes Model is based on Naïve Bayes Classifier framework with an assumption that the frequency of the features follows a multinomial distribution. According to Murphy, this event model is mostly used for text classification purposes. In text classification, the task is to find the best class for the document. The class with the maximum posterior probability is the best class for the document being tested [19]. Equation 2 is the equation that finds the maximum a posteriori (MAP) class C_{map}. According to the equation, the conditional probabilities are based on the features present in a document. The multiplication of the conditional probabilities is due to the assumption that the features are independent of each other.

$$C_{map} = arg \max_{c \in C} P(c|d) = \arg\max P(c) \prod_{1 \le k \le n_d} P(t_k|c) \quad (2)$$

The conditional probabilities in text classification may be relatively small and as such, multiplication may result in very small numbers that could be insignificant and can create floating point underflow errors [18]. In order to avoid these conditions, instead of multiplying the probabilities, the logarithms of the probabilities are added together. This avoids the floating point error and the results remain unchanged. The class with the highest logarithmic probability is still the most probable. Equation 3 shows the true implementation of the Naïve Bayes classifier with the logarithmic function included.

$$C_{map} = arg \max_{c \epsilon C} [\log P(c) + \log P(t_k|c)] \quad (3)$$

The equation can be interpreted as follows. The term log P(c) is the weight that indicates the relative frequency of the class C whereas, log [$P(t_k|c)$] indicates how effective the term t is to classify the document to class C [18]. The sum of the log prior and the log of term weights then indicates the most evident class for the document.

We first find the maximum likelihood estimate which simply corresponds to the most likely value for every feature in the dataset. The prior P(c) is calculated by dividing the number of instances of a given class present in the dataset with the total number of classes in the dataset.

The conditional probability P ($t_k|c$) is estimated as the relative frequency of term t in the documents belonging to class C [18]. Equation 4 is the formula to find the value of $P(t_k|c)$;

$$P(t_k|c) = \frac{T_{ct}}{\sum_{t' \epsilon V} T_{ct'}} \quad (4)$$

T_{ct} represents the number of times a term has occurred in the documents belonging to class C including the multiple occurrences of the term in a single document. We also make a positional independence assumption which means that the occurrence of a particular term does not relate to the occurrence of any other term [18].

2.1.2 Laplace Smoothing

One problem with the maximum likelihood estimate is when a feature is not present in the dataset. This makes the probability of the feature zero and hence results in the posterior probability also to be zero [20]. This may make the term significant enough to avoid the document to be classified in the class where the conditional probability was zero. If this issue is not resolved, the results cannot be trusted. As a result, a single term could decide the fate of the document where it would fall into a class where the term is present regardless of the evidence provided by the other terms.

To remove the effect of zeros, Laplace smoothing is utilized and adds one to the term frequency [20]. This does not cause the probability of the missing term to change significantly, but makes it very small yet significant enough for calculation. Equation 5 is the equation with Laplace smoothing;

$$P(t_k|c) = \frac{T_{ct}+1}{\sum_{t' \epsilon V} T_{ct'}} \quad (5)$$

TRAINMULTINOMIALNB($\mathbb{C}, \mathbb{D}$)

1 $V \leftarrow$ EXTRACTVOCABULARY($\mathbb{D}$)
2 $N \leftarrow$ COUNTDOCS($\mathbb{D}$)
3 **for each** $c \in \mathbb{C}$
4 **do** $N_c \leftarrow$ COUNTDOCSINCLASS($\mathbb{D}, c$)
5 $prior[c] \leftarrow N_c/N$
6 $text_c \leftarrow$ CONCATENATETEXTOFALLDOCSINCLASS($\mathbb{D}, c$)
7 **for each** $t \in V$
8 **do** $T_{ct} \leftarrow$ COUNTTOKENSOFTERM($text_c, t$)
9 **for each** $t \in V$
10 **do** $condprob[t][c] \leftarrow \frac{T_{ct}+1}{\sum_{t'}(T_{ct'}+1)}$
11 **return** $V, prior, condprob$

APPLYMULTINOMIALNB($\mathbb{C}, V, prior, condprob, d$)

1 $W \leftarrow$ EXTRACTTOKENSFROMDOC(V, d)
2 **for each** $c \in \mathbb{C}$
3 **do** $score[c] \leftarrow \log prior[c]$
4 **for each** $t \in W$
5 **do** $score[c] \mathrel{+}= \log condprob[t][c]$
6 **return** $\arg\max_{c \in \mathbb{C}} score[c]$

Figure 3: Algorithm for training and testing Naïve Bayes classifier [18]

Figure 3 shows the Algorithm for training and testing the Naïve Bayes Classifier. D denotes the documents/dataset for training while V is the extracted vocabulary for the dataset.

2.2 Support Vector Machine

The Support Vector Machine (SVM) is a supervised machine learning algorithm that learns from a set of labeled data and produces a function that outputs labels for new documents. The output from the algorithm is a mathematical function that is inferred from a set of labeled examples. The space takes on one of two values at all points in the space, corresponding to the two class labels and is considered a binary classification [21]. The derivation of the SVM is based upon a linear decision function. Figure 4 shows an example of a linear function in a two dimensional vector space where the x's denote positive data points while the o's represent negative data points.

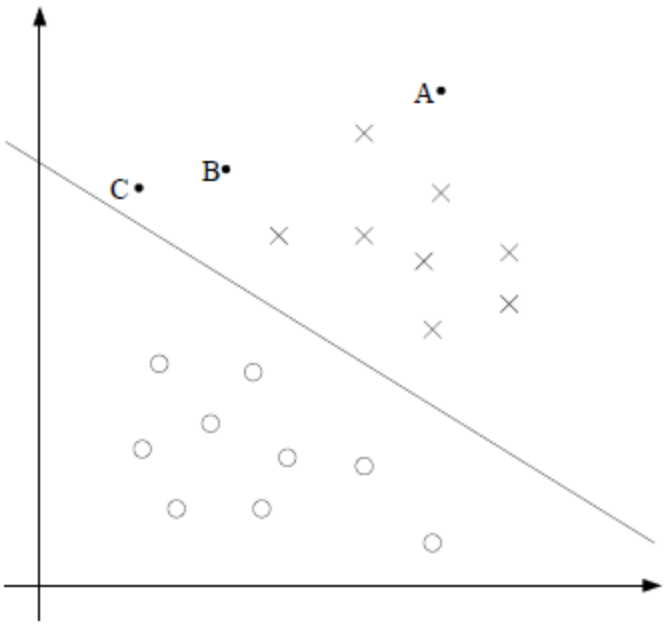

Figure 4: Negative & Positive training examples with separating hyperplane [22]

The centre line indicates the decision function. Point A lies too far from this boundary and hence we can say with certainty that it is a positive point. However, In the

case of point C, although it is on the positive side of the boundary, even making slight change to the decision function could bring it to the negative side and change its classification.

2.2.1 Mathematical Notation

These decision functions can be mathematically represented as a function of input x and the output is the predicted label y for the vector x i.e. {-1, 1}. The linear classifier can be expressed as equation 6 [21];

$$f(x) = < w, x > + b \tag{6}$$

The parameter w is the weight vector while b is the scalar. Notation <w,x> is the scalar product of w and x and is defined by equation 7 [21];

$$< w, x > = \sum_{i=1}^{d} w_i x_i \tag{7}$$

Where *d* refers to the dimensionality and *i* is the i^{th} component of *w* and *x* in the form of (w_1, w_2, w_3, w_4, w_d) and (x_1, x_2, x_3, x_4, x_d). Figure 5 shows the w and scalar b in the vector space;

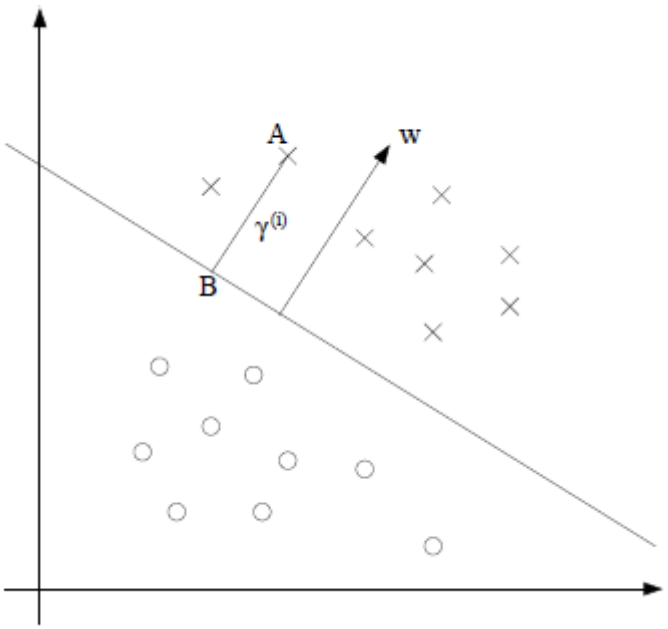

Figure 5: Geometric margins with decision boundaries [22]

In figure 5, ***W*** is orthogonal to the separating hyper-plane. Point ***A*** represents some input from the dataset which has a label $y_i = 1$. Line AB denotes the shortest distance from input A to the distance boundary denoted by $\gamma^{(i)}$ [22].

2.2.2 Finding Optimal Margin

For a Linear SVM, "*Given a training set of vectors x1, x2, … xn with corresponding class membership labels y1, y2, … yn that take on the values +1 or -1, choose parameters w and b of the linear decision function that generalizes well to unseen examples.*" [21]. Consider the example below in Figure 6,

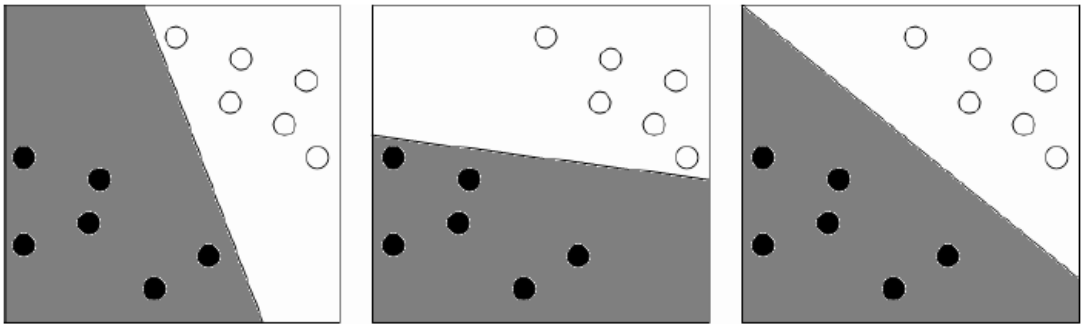

Figure 6: A simple 2D classification task with multiple linear decision functions [21]

In this problem, there are multiple decision boundaries that are drawn for a single problem. All of them provide a feasible solution to the problem. "*Given a training set, the natural need is to try to find a decision boundary that maximizes the margin, since this would reflect a very confident set of predictions on the training set and a good "fit" to the training data. Specifically, this will result in a classifier that separates the positive and the negative training examples with a 'gap'*." [22]. The idea is to select decision boundaries which do not only correctly distinguish between two classes but also lie as far from the training examples as possible [21]. This approach leads to the Linear SVM, which choses the hyper-plane with a maximum margin.

Finding the maximum margin decision boundary is an optimization problem. In general, optimization problems have decision variables, a set of constraints and an objective function. The task is to find the maximum or minimum value for the objective function.

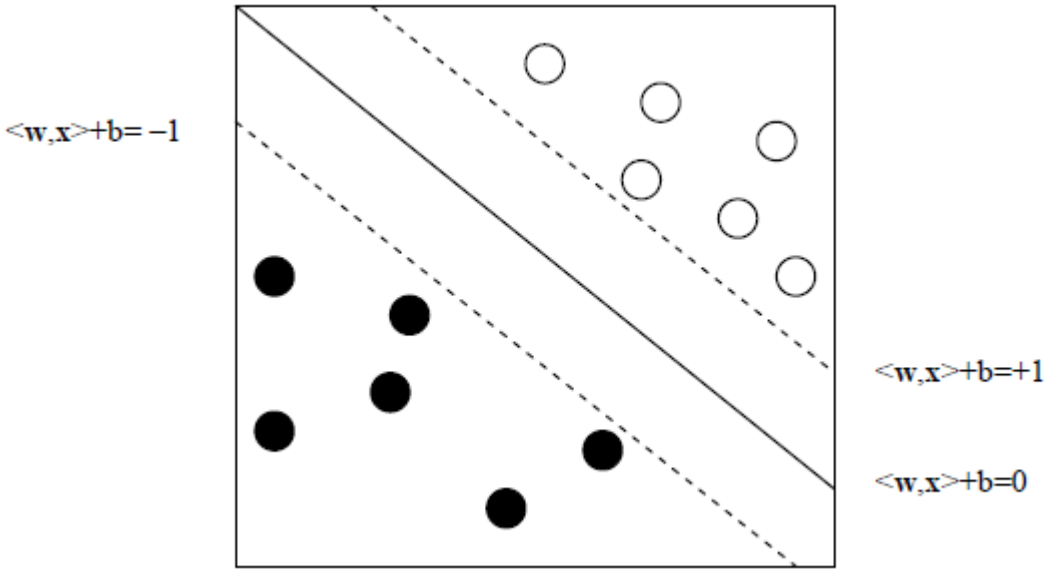

Figure 7: A linear separable classification problem with additional constraints [21]

To formulate the maximum margin hyperplane as an optimization problem, consider Figure 7. The figure shows some data plotted in two dimensional space having labels 1 & -1 respectively. The decision function is defined by the parameters w and b. For our hyper-plane to correctly distinguish between the two classes, the following constraints in equation 8 & 9 should be considered;

$$< w, x_i > +b > 0, \ for\ all\ y_i = 1 \tag{8}$$

$$< w, x_i > +b < 0, \ for\ all\ y_i = -1 \tag{9}$$

These constraints can be combined in a single set of constraints as equation 10;

$$(< w, x_i > +b) y_i > 0, \ i = 1 \ldots n \tag{10}$$

However, this is not enough to separate the two classes optimally and require a maximum margin. The solid line in the figure is the decision surface which is denoted by equation 11 [21];

$$< w, x_i > + b = 0 \tag{11}$$

The dotted lines are the additional hyper-planes where the function <w,x> + b is equal to 1 (at the upper right) and -1 (at the lower right). To find the maximum margin, the dotted lines should be such that the dotted lines are equidistant and parallel to the decision hyper-plane and maximize their distance from one another. This can be written mathematically as equation 12 [21];

$$y_i(< w, x_i > + b) > 1, \ i = 1 \dots n \tag{12}$$

The distance between the dotted lines is found to be [21];

$$\frac{2}{\sqrt{<w,w>}} \tag{13}$$

Finally we end up with an optimization problem which helps to find the parameters of the maximum margin hyper-plane. Equation 14 shows the finalized optimization problem [21];

$$\min_{w,b} \frac{1}{2} w.w \tag{14}$$

$$such\ that\ y_i(w.x_i + b) \geq 1$$

$$for\ all\ i = 1,2, \dots, m$$

Note that the problem is written as a minimization problem rather than maximization because they are both the same [22].

2.3 Ensemble Classifier Framework

An ensemble of classifiers is a set of classifiers where individual decisions are combined in some way (typically by weighted or un-weighted voting) to classify new examples [23]. An ensemble classifier is constructed from a set of base classifiers that have individual learning over a training dataset [15]. A general representation of Ensemble Classifier is shows in Figure 8. A necessary condition and assumption for an ensemble classifier is that the accuracy must be greater than the individual baseline classifiers [23]. This is only possible if the errors of the base classifiers are uncorrelated [15].

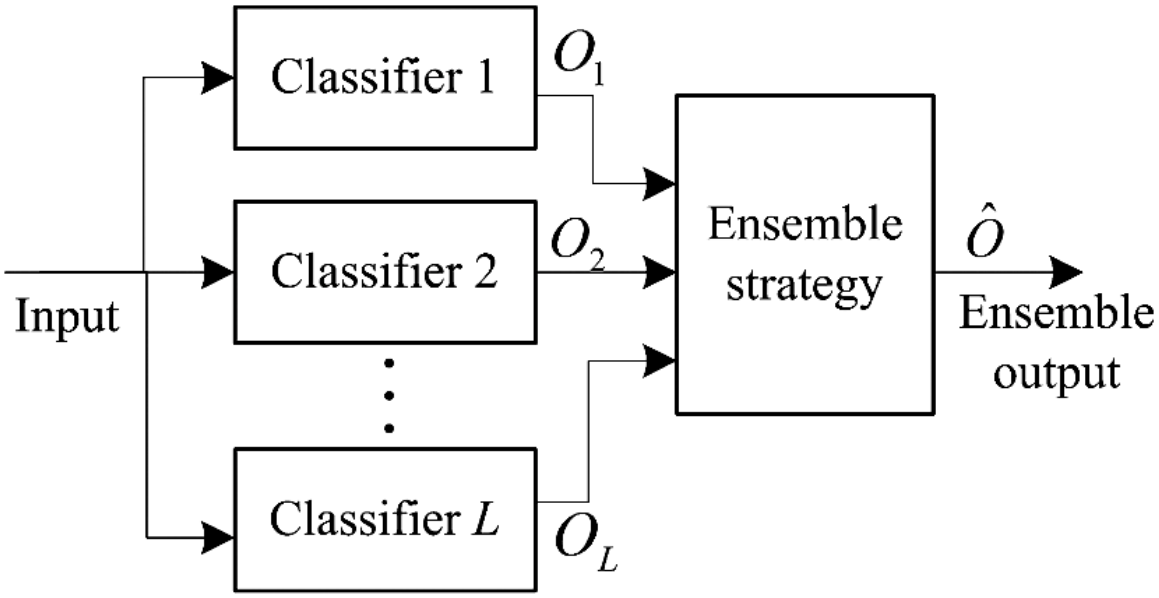

Figure 8: General representation of Ensemble Classifier [70]

Base classifier refers to the conventional classifiers used in a classification task. For example a Naïve Bayes Classifier, a Support Vector Machine, k-NN, Artificial Neural Networks and the like. The choice of an individual classifier may depend upon its

individual performance in a certain classification task. The Ensemble Classifier may enhance the results of the classification system when two or more of the best performing classifiers are selected [24].

2.3.1 Methods for Constructing Ensemble Classifiers

An Ensemble Classifier can be constructed in several ways. This idea has been studied extensively by several researchers and there are multiple ensemble techniques for ensemble classifiers. Some of these techniques are:

2.3.1.1 Generation by manipulating training data

In this method, the training algorithm is trained multiple times and each time with different subsets of dataset [23, 24]. This technique is useful with unstable classifiers whose output varies with the slight change in the input data. The easiest way to manipulate the dataset is to select a substantial amount of training examples randomly from the dataset and then replacing them with an equal amount of randomly drawn training examples in each iteration. This method is known as bagging [23].

2.3.1.2 Generation by Manipulating Input Features

In this method, different subsets of features are extracted from the dataset and the base classifiers are trained on one of each dataset. The selection of the subsets is random. This type of subspace ensemble classifier does not work as effectively as other techniques [24]. This technique is found to work best when the features are highly redundant. Removing some of the features from the subsets affects the individual

classifiers negatively and ultimately affects the overall performance of the ensemble classifier [23].

2.3.1.3 Generation by Manipulating Weights

In this method, each classifier is trained utilizing different sets of weighing techniques depending upon its individual performance on a particular weighing method [24]. For example, SVM might work well with Term Frequency - Inverse Document Frequency (Tf-IDF) but not with simple Term Frequencies (TF). Conversely, Naïve Bayes classifier requires term frequencies to calculate the probabilities of the features. In this case, two separate weighting methods can be used for two different classifiers [25]. Once the classifiers are trained each of the classifier categorizes the documents independently and the final result can be obtained by using a logical "AND" operation.

Another variant for this method utilizes a third classifier with another weighting method. In this case, the best two out of three are used to find the class of new documents. This is an effective way to reduce the margin of error [25].

2.3.2 Ensemble Selection

An important aspect of the ensemble classification method is the selection of the best classifiers for the task and whether the individual classifiers would yield better results. Several ways have been adopted by researchers like the bagging algorithm, where the determination of the number of iterations is done in advance and acts as a controlling parameter. This parameter can be set while training the classifiers. Individual classifiers can be added until the desired output has been achieved.

One possible method of doing this would be to test the ensemble classifier after each individual classifier is added to the framework. The accuracy of the resulting classifier is observed and the addition of more classifiers is ceased when the maximum accuracy value has been achieved [26].

2.4 Data Pre-processing

Data pre-processing is an important step in the task of text classification. Data is not always clean and is comprised of many unwanted impurities that can alter the results of a classification task in a negative way. The classification results can be unrealistic, sometimes biased and unacceptable until the training data is cleaned. This in turn makes the classification systems unstable and unreliable. Low quality data can lead to low quality data mining results [27]. Figure 9 shows the general and overall process of data mining and the stages of data pre-processing are marked with arrows. Data pre-processing depends upon the type of data mining task to be performed. Researchers have used many different approaches to clean the data but the goal remains the same, to make classification more effective and efficient.

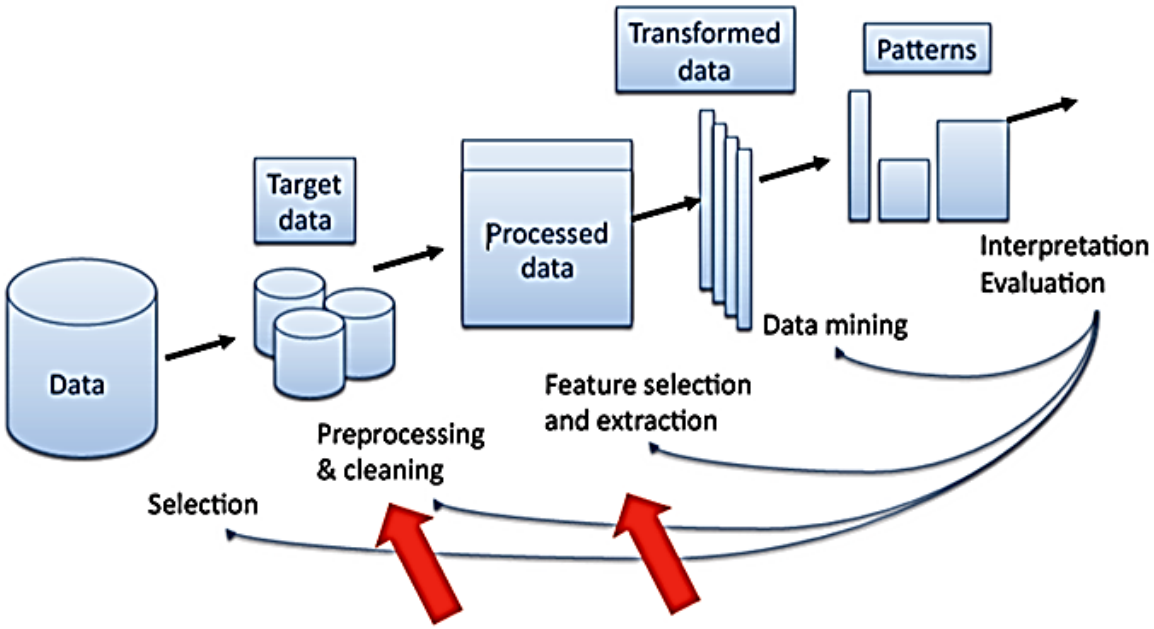

Figure 9: Stages of Data Mining

There are numerous data pre-processing techniques that are available to remove noisy data. It is important to know the attributes of the data that is being used for the data mining task [27]. The task where numerical values are involved requires a statistical branch of mathematics to clean the data. Sometimes, the data source is a key factor in deciding on the approach should be followed in order to clean the data. There are numerous quality attributes of the data such as; data consistency, completeness, accuracy and precision levels to be achieved before using it for data mining purposes.

In Text Classification, the data is comprised solely of the String data type which includes the alphabet and sometimes is also comprised of numerical values [28]. Although punctuation is usually neglected, it can be used when rule based classification techniques are applied. Text classification requires additional pre-processing techniques

when compared to the conventional numeric based data mining [28]. It not only requires statistical pre-processing techniques but also lexical rules to eliminate the undesired impurities out of the data set to ensure better results. The selection of lexical rules depends on the nature of text classification because text classification is also further divided into various sub-categories.

In sentiment analysis for social media analytics, the training data consists of comments and posts shared by users on their social media profiles. Therefore, the kinds of impurities found in this data are different from the data obtained from other sources. The data contains a lessor amount of information that is useful for sentiment analysis. Words may not have any polarity information or they are insignificant enough to be dropped from the training data set. Social media data also has a different form of vocabulary because of the use of casual language when writing posts and comments. The nature of user posts may vary from highly formal to informal language with the use of slang words to describe their feelings about a certain issue. This variation in written language makes it difficult to develop efficient systems [27].

Another problem that arises is human nature with respect to opinion. Every human being is different with different thought processes and interpretations [29]. Human nature plays a strong role in selecting the appropriate data for the sentiment analysis task. Due to difference of opinions, sentiment is not always as expected. For example, “A bright sunny day” could have a positive polarity to one person but could be negative to another. This difference of opinion sometimes results in ambiguous data which hinders the development of accurate Sentiment Analysis systems [28]. This problem is difficult to handle given that the assumptions are that any data coming from

one source is classified by either single person or a group of people with the same thought process.

There are various kinds of impurities that can be removed during the task of sentiment analysis. These impurities contain standard words or features which are global and do not represent sentiment. They can be removed regardless of the type of text classification being done. There are some special features that are encountered while working with social media. In general, there are six categories that can be removed from social media datasets for sentiment analysis:

- Stop Words
- Numbers
- Hashtags
- Links
- Tags
- Stemming

These categories are explained in the next section.

2.4.1 Stop Words

In text classification, stop words are considered to be words that do not have significant importance in classifying documents. It is common practice to eliminate them before training the classifier. Stop words is a general term used for any set of words that may affect the results of the classification. Although they may be found in every class with a similar ratio, they still affect the results negatively. They also increase the training

time of the system because they increase the size of the feature set. Search engines also use a stop words filter to eliminate the articles and short function words [30]. Figure 10 shows some basic stop words used by search engines.

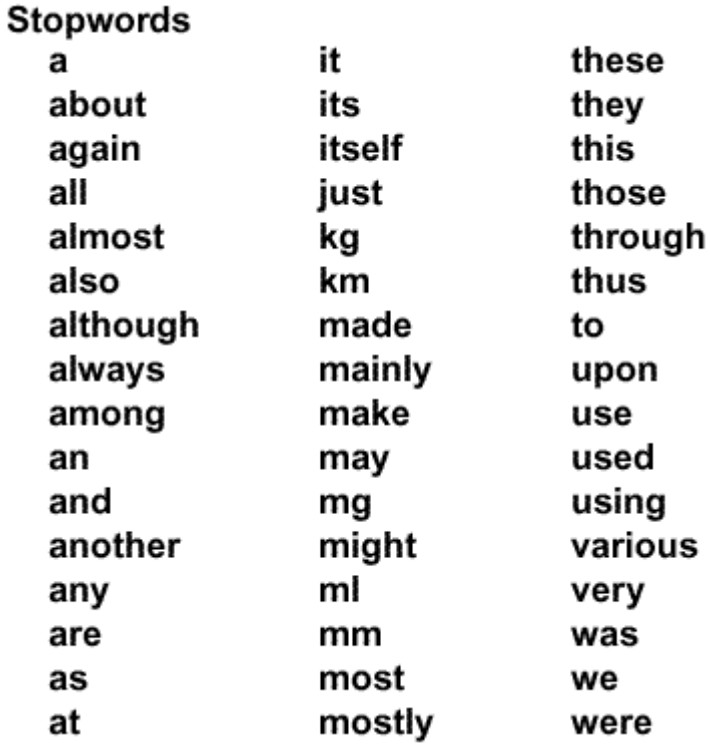

Figure 10: Common Stop words [75]

Stop words are determined by the nature of the text classification. For example, if a system is being designed to identify the subject and object in a sentence, the researcher would be more interested in nouns and would eliminate the adjectives and articles from the sentence because of their insignificance. Similarly, in sentiment analysis, there are only a few words in a sentence that help in determining the polarity of the sentence. Theoretically, anything other than the adjectives can be eliminated from a sentence in order to get the features that are significant for the task. However, this is not always the

case. Since the dataset from social media is not formally written, it is difficult to determine polarity by using the standard adjectives from a dictionary [32]. If a classifier does not find any features from its vocabulary in a test document, it may produce random results which can decrease the accuracy of the classifier.

For sentiment analysis, there is a list of words categorized as stop words and available on the Internet for free. This is a standard list but it can also be customized by adding stop words depending upon the task. This standard list is available at http://www.ranks.nl/stopwords and is comprised of 174 words which can be edited according to the need.

2.4.2 Numbers

While working with social media datasets, another kind of impurity encountered is numeric data. Sometimes people use numbers for short notations of words. There is typically no standard pattern followed and is therefore difficult to use them as training data. The numbers themselves do not represent any polarity so it is assumed that the numbers are neutral in nature. For this reason they should be removed from the dataset.

People sometimes use numbers to express sentiment by defining a scale in their posts. For example "The movie was great. It's a master class by Steven Speilberg. However I would rate it 8 on a scale of 10 because of the benchmark he set with his movie Titanic." In this post the user rated the movie on a scale which shows that the movie was above average. But for a computer, 8 and 10 are just numbers and considered two distinct features. Due to the assumption of feature independency, there is no connection between them theoretically. The numbers can only be used if rule based

classification is applied where the positioning of the words and numbers would be taken into account by analyzing the structure of the sentence [32].

In sentiment analysis using supervised machine learning techniques, the numbers are considered to be neutral in polarity because as a single feature they are of no value to the system. In fact, increasing the feature size increases the training time of the system. In addition, the time to classify the documents also increases, so it is a good practice to eliminate them from the dataset.

2.4.3 Hashtags

Hashtags are words or phrases prefixed with the # sign without a space. This is a popular way on social media and micro-blogging websites to highlight an important event name or express feeling [33]. Hashtags are also used by social network websites to indicate trending topics. The hashtag is often used in information technology to define or highlight important items. This idea was used by Twitter between 2009 & 2010 and is now used by most of the social network sites including Facebook. The hashtags are used anywhere freely whether in comments or user status.

Figure 11: Hashtags [71]

Hashtags can be anything from a name of an event to an expression of one's feelings [35]. Users can post anything under hashtags which they feel is important to them or to society. Although they are used in social media posts, not all of them represent sentiment [34]. For example, a person who posts his concerns about the Iraq war on his social media website may post something like "1000's of lives lost in #Iraq and #Afghanistan but all went in vein". The hashtags here are pointing to the locations. These can be useful for subjective classification or in regression analysis but are of no use in sentiment analysis [33].

2.4.4 Links

The use of hyperlinks on social media is common. It has been observed that community pages on social media often post links on their profiles. The main reason for this is that they want more visitors to their websites. This does not only increase traffic

on their webpages but is also a way to increase their revenue from advertisements. Organizations make their community pages on social media and instead of posting information directly, they post links to their webpages. Users who click on the posts are automatically redirected to the specific link where the real information is shared.

Often, links are just webpage addresses that direct users to the real information. The text of the link itself does not contain any information so it does not have any polarity to it [35]. However, links are also posted between texts.

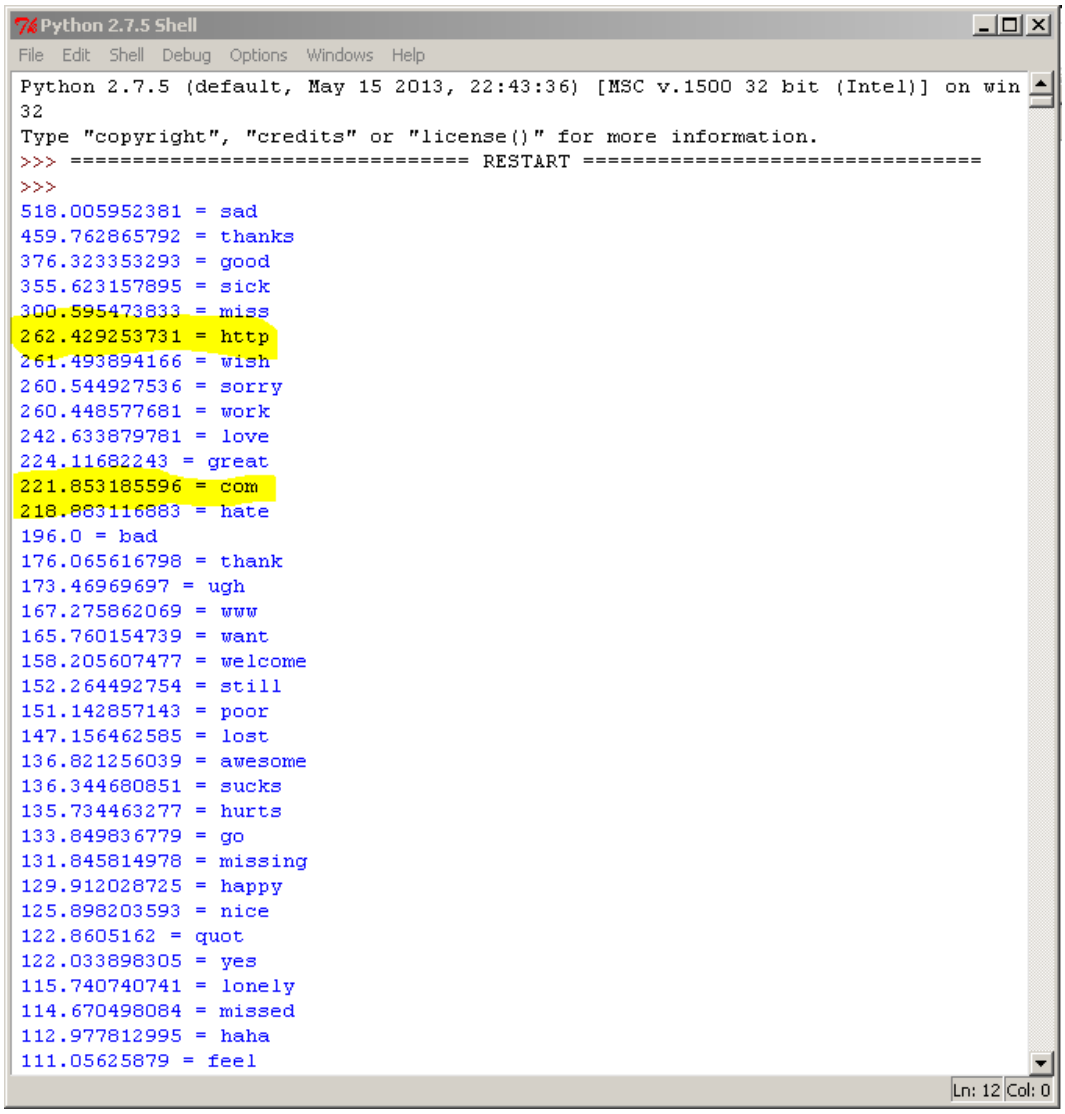

Figure 12: Chi-square test scores without removing links from dataset

The links increase the feature size if not removed from the text. Figure 12 shows the highest ranking features without the removal of links from the dataset while http and .com is amongst the top ranked features. An increase in the feature size makes the classifier less effective and affects the performance of the classifier [35]. It can make it slower and decrease the accuracy of the classifier. Since the links themselves do not contain any information that represents sentiment, it is a good idea to remove them from

the training dataset [35]. Another option is to disregard the instances that contain links but there is a chance that useful information will be lost.

2.4.5 Tags

Tags are similar to hashtags but the difference is that they are used to indicate a person. This notifies the person through a notification that they are being addressed somewhere in a comment or a user post. Since names are nouns and do not have any sentiment attached to them, they can be removed from the dataset. If they are not removed, the feature size would increase and hence will affect the accuracy of the system as discussed in previous sections.

Tags are difficult to remove from the dataset. No standard procedure or programming function is available to remove tags. A name like "Andrew" and a word like "Beautiful" are same for a machine and count them as two distinct features. Regular Expression functions are not useful in this situation. One way to handle this would be to make a global dictionary that contains names and remove those names from the dataset. Another way might be to make a dictionary of all the names that are in the dataset and remove them from the data set one by one. However, both of these methods would be difficult and inefficient. Another alternate method is to use statistical techniques to remove them. These are called data selection techniques. One of them is discussed in detail later in this thesis.

2.4.6 Stemming

People tend to make mistakes while writing their views and opinions and sometimes they do it on purpose. This has become a norm on social media to write in an informal style that indicates the degree of their emotion. In contrast, there are professional bloggers and writers who might be excited, over joyed or saddened about some issue but tend to write in a formal manner.

Due to the differences in the style of writing and the informal norms on social media, it becomes difficult to handle information that is important but maybe full of errors. One example is the use of casual writing with spelling mistakes that are deliberate [37]. This is a common problem found within social media datasets. People use short notations or short forms for words that represents sentiment. The following post in Figure 13 is an example of a tweet that indicates the need for the stemming algorithm.

Figure 13: Post with deliberate spelling mistakes

The feelings and concerns expressed in tweets are genuine but the way of writing them is not formal or with correct spelling. The word "Happy" is a word that has a positive sentiment attached to it is written in different ways. The user wrote "Happpppyyyyy" in a way that looks like it is the superlative degree of happiness. However, for a computer system, the two words are two distinct features. As a result, there is always a chance of losing data which could have created an impact on the learning process and ultimately affects the results of the classifier.

Genuine writing can also create similar problems. This is because of the grammatical rules of the English language. A verb can be used in many different forms. For example, the word "kill", "killing" and "killed" despite being the same are treated as three separate features [37]. As a result the system measures their probabilities separately. This can affect the importance of the feature in a class or in the dataset.

"*In linguistic morphology and information retrieval, stemming is the process for reducing inflected words to their stem, base or root from.*" [67]. The stemming algorithm provides a way to convert the extracted features into a standard form so that the words that are similar but are written differently could be counted as a single feature [38].

Stemming is a general term and there are no documented rules for designing a stemming algorithm. However the idea dates back to late 60's and the early work was presented by Julie Beth Lovins which was published in 1968 [65]. Later in the 80's, "*A stemmer was written by Martin Porter and was published in the July 1980 issue of the journal Program. This stemmer was very widely used and became the de facto standard algorithm used for English stemming. Dr. Porter received the Tony Kent Strix award in*

2000 for his work on stemming and information retrieval." [66]. Many other researchers have also worked on stemming and produced algorithms that differ in the aspects of performance, accuracy and the type of task to be performed. Some of these algorithms are:

- Lookup algorithms
- Suffix stripping algorithms
- Lemmatisation algorithms
- Stochastic algorithms

Each of the above mentioned algorithms work differently. Lookup algorithms and suffix stripping algorithms are simpler and rule based. In contrast, lemmatization algorithms are more complex and involve the determination of parts of speech and applying different normalization rules for different parts of speech. Stochastic algorithms involve the use of probability to determine the root form of a word [38].

2.5 Feature Extraction & Transformation

Feature extraction is a process in information retrieval and data mining which involves the extraction of meaningful features that defines a large set of data. For datasets that are very large and contain redundant data, the feature extraction process transforms the data into set of features which defines the whole dataset [39]. When dealing with large datasets, it becomes more difficult to process the information as they require more memory for processing. The larger the dataset, the larger the number of variables involved. Feature extraction is a way to reduce the data set into a smaller number of variables [39].

In Text Classification, the datasets can be very large containing millions of words. Each character in a word requires a byte of memory to be stored. Processing trillions of bytes at once is difficult for normal systems. Also, raw text values in the form of alphabets and symbols cannot be fed into the algorithms as they require numeric values to perform calculations. Due to these factors, feature extraction techniques are applied to the dataset and are discussed in the following subsections.

2.5.1 Bag of Words Model

The bag of words model is an important step in feature extraction of text data. It is a simplified representation of text data used in Natural Language Processing (NLP) and Information Retrieval (IR) [40]. In this model, the dataset is considered to be a bag of words. The presence and occurrence of words are considered to be independent of each other. The strings in the text documents are tokenized to separate words. Since the ordering of the words is not taken into account, the document "Australia defeated Srilanka" and "Srilanka defeated Australia" are both identical [40].

2.5.2 N-gram Model

For Information retrieval and text classification, the N-gram model is a continuous stream of text or speech of an 'n' number of elements. These elements can be numbers, words or characters in a text or speech document. In text classification, The N-gram model is a way of retrieving features from text documents. The value of 'n' depends on the classification task. It is not necessary that all the text classification tasks would use a similar value of 'n'. However '1' and '2' are commonly used in sentiment

analysis. When n=1, the n-grams are known as unigrams and when n=2, the n-grams are called bigrams [41].

2.5.2.1 Unigram

The Unigram is the basic N-gram model but is the most effective of all other n-gram models for sentiment analysis. The Unigram is the implementation of the Bag of Words model [41]. For Unigram, the words are assumed to be independent of their occurrence in the document. The Unigram is the most widely used model for sentiment analysis. This is because it yields the best results when the algorithms are probabilistic in nature and it takes conditional independence of the features into account. The use of the unigram also helps in deep analysis of the documents. Every word is counted as a feature and thus makes it easier to analyze what kind of data is present in the dataset. It also makes it easier to remove impurities from the dataset. Statistical tests can also be conducted to check the quality of the data. It also provides the freedom to analyze the impact of a feature on the classification. Any other n-gram besides unigram does not provide the freedom to conduct a deep analysis of the data [41]. A comparison of the Unigram model with other N-gram models is discussed in the results section.

2.5.2.2 Bigram

The Bigram is another variant of N-gram model where the value of N is 2. This means that two consecutive words are considered as a single feature [40]. For example, in the phrase “fault in our stars”, the pair “fault, in” will be counted as a single feature. This can be useful in the kind of classification where conditional probabilities of occurrence of a word is taken into account. For example to check the conditional

probability between a negation and an adjective would be very helpful in analyzing the sentiment of a sentence. The main idea around the feature independence makes bigrams ineffective for classification. Since most social media posts and especially Twitter, are comprised of only a few words, the limitation of the number of characters makes it rare to find a pair of words being used back to back in the dataset. This ultimately leads to decreased accuracy because there is a higher chance of not finding features in the test documents.

2.5.3 Feature Weights

Classification algorithms require numeric data to perform calculations in order to produce results. Raw data in the form of text cannot be fed into the algorithms. It is important to covert text data into numerical values. The process of converting them into numerical values is known as feature weighting. "*Feature weighting is a technique used to approximate the optimal degree of influence of individual feature*." [42]. Feature weights are a measure of the importance of a feature in a dataset, class or document. Over the years, researchers in the field of data mining have applied various feature weighting techniques for obtaining optimal results. Many of them have concluded that feature weighting techniques helps to increase efficiency of classification systems [43].

There are many feature weighting techniques that have been developed and have been used in the field of text classification. Choosing a weighting technique depends upon the method used for classification. The most common weighting methods are:

- Term Frequency (TF)
- Term Frequency – Inverse Document Frequency (TF-IDF)

2.5.3.1 Term Frequency

Term Frequency is the measure of number of times a word has occurred in a document [43]. The frequency of the occurrence of a word is helpful in determining many aspects of the dataset. Probabilistic algorithms such as the Naïve Bayes classifier require frequency of a feature to calculate the conditional, the prior and the posterior probabilities which helps it to classify the documents into a class. Similarly, other algorithms such as support vector machine also use word counts as a weight for the feature.

In term frequency, we assign a weight to the term ***t*** which is equal to the number of its occurrence in a document ***d.*** The term frequency is denoted as **tf(t,d)** with the subscripts denoting the term and the document respectively [69]. The ordering of a term in a document is ignored and the document is considered to be a bag of words model. The Term frequency weights are also helpful in applying various statistical methods for feature selection and reduction [43].

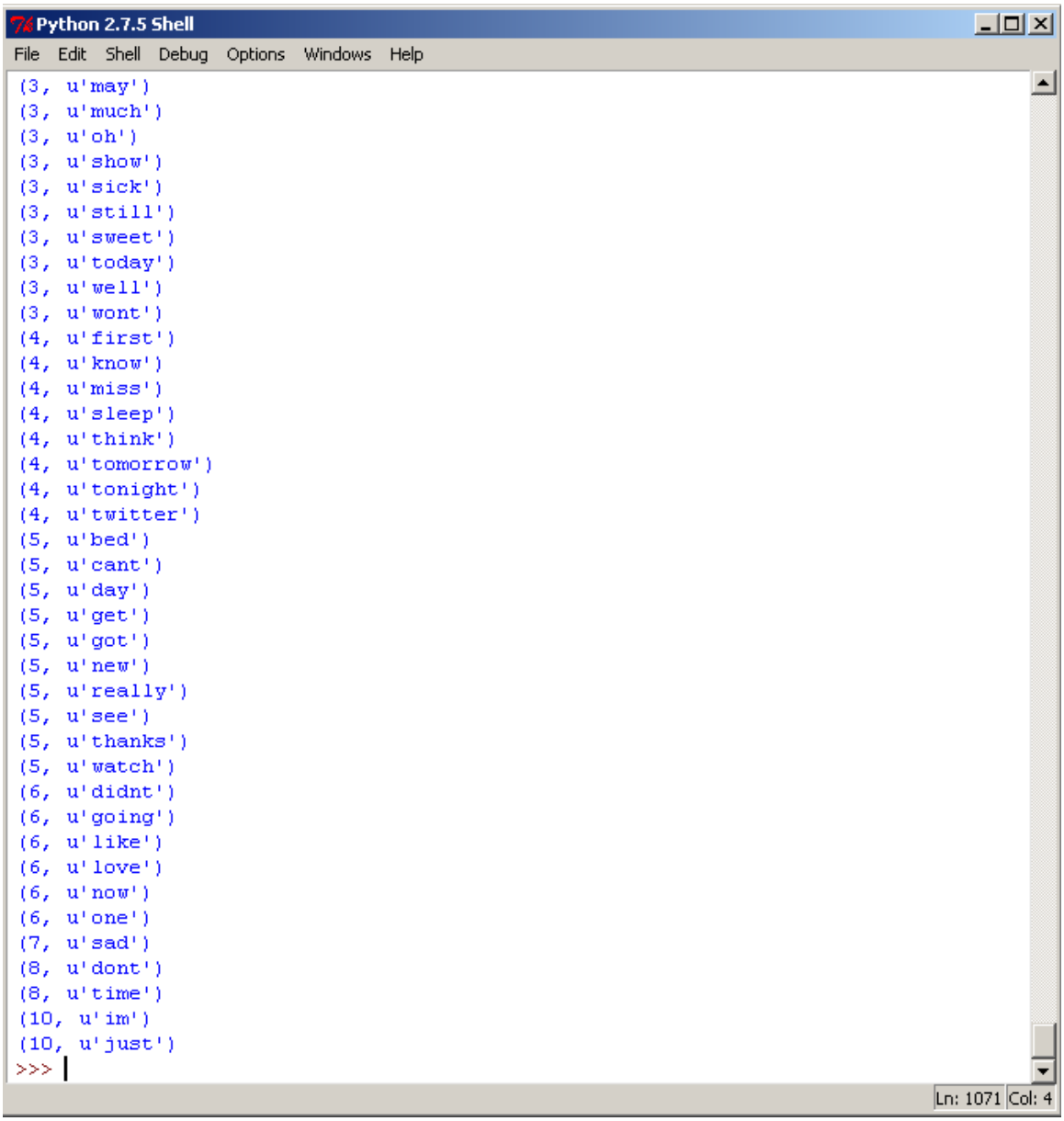

Figure 14: Term Frequencies of features in the dataset

2.5.3.2 Term Frequency – Inverse Document Frequency (TF-IDF)

Term Frequency on its own can be ineffective for classification tasks. The issue that arises with the raw term frequency is that all of the terms are considered equally

important [44]. In some information retrieval cases, frequently used words do not help in determining the class of a document [45]. For example, every document contains articles such as "a", "an" and "the" but they do not help in determining the polarity of the document. To reduce the effect of recurring words and to increase the impact of less frequently used words, the Inverse Document Frequency (IDF) is introduced.

The term document frequency which is denoted by ***df*** is the measure of the number of documents in which term ***t*** has occurred [45] [46]. Table 1 below shows an example of words and their respective term counts and the document frequency.

Table 1: Term Frequency versus Document Frequency [69]

WORD	**TERM FREQUENCY**	**DOCUMENT FREQUENCY**
TRY	10422	8760
INSURANCE	10440	3997

Table 1 shows that although the words have similar term frequency but the word "***insurance"*** has occurred in less number of documents. This example indicates that the term frequency and the document frequency exhibit different behaviours. In search engines, a similar technique is applied to retrieve a smaller number of documents that are more informative rather than more documents which are less informative. Therefore, inverse document frequency is used to scale the weight of a term. Equation 15 is the formula to calculate the Inverse Document Frequency:

$$idf_t = \frac{\log N}{df_t} \tag{15}$$

N stands for total number of documents in a corpus. The log of the ratio between ***N*** and document frequency helps to boost the score of a term which is frequently used in a smaller number of documents. Table 2 below shows the ***df*** versus ***idf*** of different terms from the Reuter's dataset [69].

Table 2: Document Frequency versus Inverse Document Frequency [69]

Term	**Document Freq.**	**Inverse Document Freq.**
Car	18,165	1.65
Auto	6723	2.08
Insurance	19,241	1.62
Best	25,235	1.5

Table shows the lower the document frequency, the higher idf. This is how rarely occurring terms obtain a higher ranking.

The term frequency and inverse document frequency can be combined by multiplying them together to find the weight of a feature. Equation 16 shows the tf-idf formula for a weight calculation,

$$tf\_idf_{t,d} = tf_{t,d} * idf_t \quad (16)$$

Tf-idf assigns a weight to term **t** that is:

- Higher when occurs many times within a small number of documents.
- Lower when the term occurs fewer times in a document, or occurs in many documents.

- Lower when the term occurs in virtually all documents.

2.5.4 Feature Matrix

A Feature Matrix is a two dimensional representation of a dataset. The rows are the instances and the columns are the features in the dataset. Each element in the matrix is the weight of a feature in an instance. Figure 15 shows an example of a feature matrix.

	F1	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**	**F9**
D0	0	2	1	0	0	0	0	1	0
D1	1	1	0	0	0	2	0	0	3
D2	0	0	0	1	1	1	0	2	0
D3	0	0	1	2	0	1	0	1	2
D4	0	1	0	0	0	3	2	0	0
D5	0	0	1	1	0	0	2	0	1
D6	2	0	0	0	3	1	0	1	0

Figure 15: Feature Matrix

Each row represents a document while each element in a row is the calculated weight of the features present in the document. The purpose of converting the documents into matrix form is to aid the calculation process of the algorithms. These matrices can also be referred to as a bag of words matrix because they store information about frequency and disregard the placement and occurrence of features in the documents.

2.5.4.1 Data Sparsity

Data sparsity refers to the empty spaces in a dataset [47]. Figure 16 shows an example of Sparse Matrix. When a dataset is transformed into a matrix, there may be a large number of zeros in the matrix [48]. These zeros are due to the small number of words in a document. For example in Twitter posts, tweets have very few words in them. This is due to the 140 character limit allowed by Twitter for a tweet. For example, consider a unique set of words of around 50 thousand. Each document would represent a 1 X 50,000 matrix, where only 20 to 25 elements would be non-zero. Figure 17 shows the ratio of zeros to non-zero numbers in the dataset.

0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Figure 16: Example of Sparse Matrix with high density of zeros

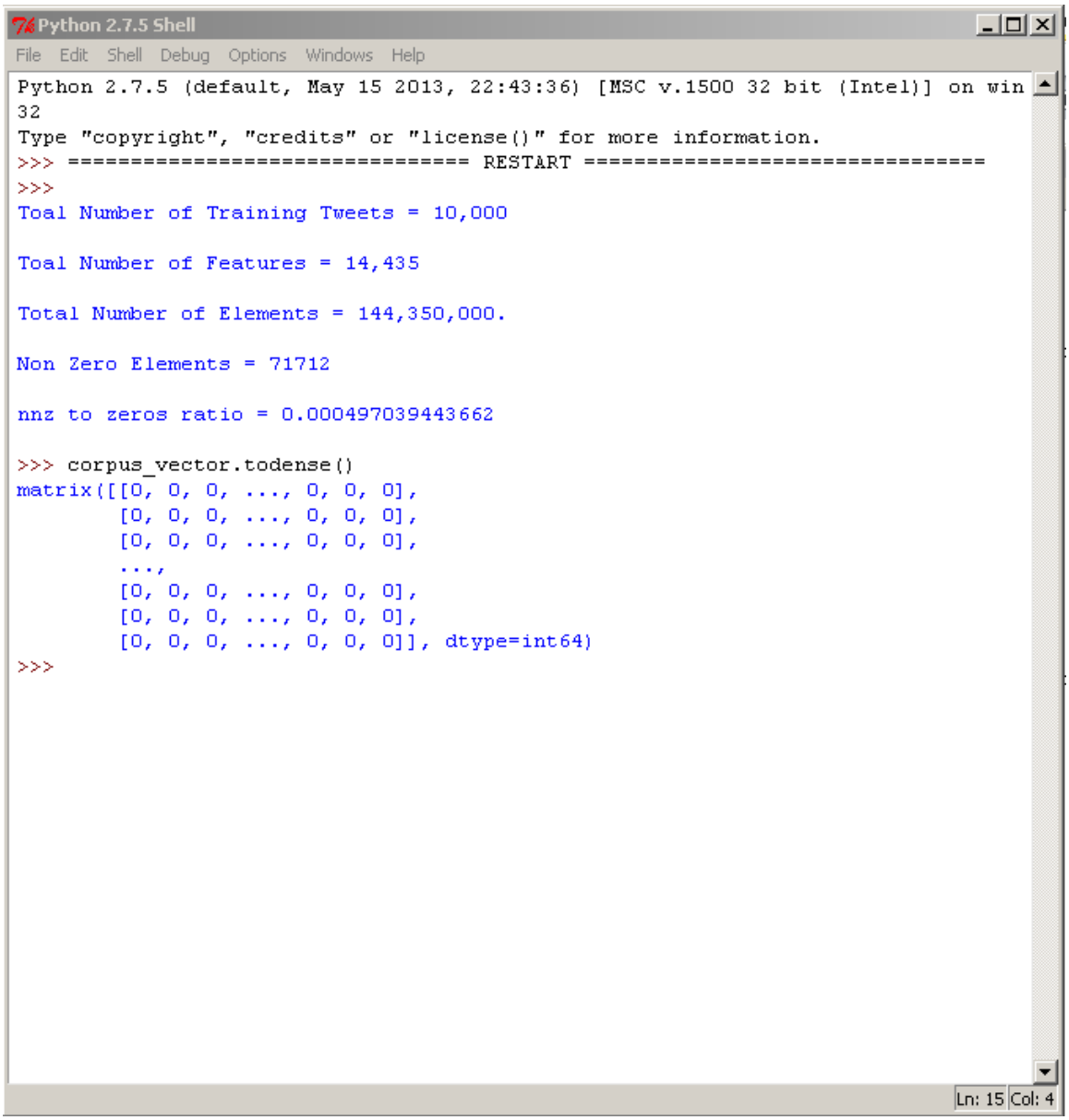

Figure 17: Non-zero and zero ratio in the dataset

To store these matrices and to speed up algebraic operations, sparse representation is used [49]. The main technique to store a sparse matrix and to reduce the memory consumption is to store non-zero elements only in either a coordinate list or compressed sparse column format.

2.5.5 Feature Selection

Feature Selection is referred to the techniques for selecting a subset of relevant features to be utilized in classification [68]. The primary assumption for feature selection techniques is that the dataset contains many irrelevant features and impurities. Feature selection is different than feature extraction. Feature extraction only provides a way to extract the features from a dataset while feature selection techniques help to select appropriate features that can aid in constructing efficient classifiers. Feature selection techniques help in three ways [50]:

- Dimensionality reduction
- Training time reduction
- Generalization by reducing overfitting

There are various feature selection techniques that are used extensively in the field of data mining. One such method is the Chi-square feature selection method.

2.5.5.1 Chi-Square Feature Selection

The Chi-square method is a statistical test that is used to test the independence of two events. In feature selection it is used to test the independence of occurrence of a specific feature and a class [51]. The overall feature selection procedure is to score each potential feature according to a particular feature selection metric, and then take the best features [52]. Equation 17 shows the general formula for calculating score of a term [69].

$$X^2(D,t,c) = \sum_{e_t \epsilon \{1,0\}} \sum_{e_c \epsilon \{1,0\}} \frac{(N_{e_t e_c} - E_{e_t e_c})^2}{E_{e_t e_c}} \quad (17)$$

Higher values of x^2 shows that the null hypothesis (H_0) should be rejected and the feature and the class are dependent. Feature with high dependence are then selected for the model. Equation 17 can be simplified as Equation 18 below,

$$X^2(D,t,c) = \frac{N(N_{11}N_{00}-N_{10}N_{01})^2}{(N_{11}+N_{01})(N_{11}+N_{10})(N_{10}+N_{00})(N_{01}+N_{00})} \quad (18)$$

3 Proposed Model

This section discusses the proposed model for sentiment analysis in social media. The model is comprised of three parts; the proposed term weighing scheme, the ensemble classifier framework and the document ranking using the probabilistic weights of the features. The first section explains the proposed weighting scheme for the features extracted from the dataset. The second section explains the classification based on the proposed ensemble classification technique. The third section explains the document ranking algorithm.

3.1 Probabilistic Feature Weighing

The probabilistic feature weighing technique is derived from the multinomial Naïve Bayes classifier. The feature independence assumption of the Naïve Bayes classifier helps to calculate the conditional probability of features in the dataset. The amount of conditional probabilities of a term is proportional to the number of classes in the dataset. For the sentiment analysis problem, the number of classes is two (positive and negative). Equation 19 is the equation that finds the conditional probability of a feature from the dataset [11].

$$P(t|C_n) = \frac{t_f+1}{\sum w_{C_n}+|V|} \tag{19}$$

After the pre-processing techniques are applied to the dataset, the conditional probabilities are calculated for every word in the vocabulary. Every feature has two distinct conditional probabilities so there are two distinct weights of a feature. Each

weight indicates the importance of a feature in a particular class. For example, the word 'beautiful' is more likely to have a higher probability in the positive class as compared to the negative class. The Naïve Bayes classifier predicts the class of a new document based on the maximum likelihood calculation. It finds the posterior probability and compares the two calculated posterior probabilities [10]. Other algorithms such as the Support Vector Machine and Maximum Entropy takes a single weight such as term frequency as input for every feature [12]. Although the Naïve Bayes classifier the takes frequency of a term as input in its initial stage, it still requires multiple posterior probabilities for its prediction stage. Therefore, the conditional probabilities cannot be used as a weight in their purest form.

3.1.1 Probability Unification Method

To use the conditional probability as a single weight, this thesis proposes a probability unification method that utilizes vector resultants for unification. The number of conditional probabilities for every feature is two because there are only two classes for sentiment analysis (positive and negative). As shown in Figure 18, the horizontal scale represents the conditional probability of the feature in the positive class, whereas the vertical scale represents the conditional probability of the feature in the negative class. The resultant probability vectors of every feature are found by using Equation 20,

$$|R| = \sqrt[2]{[P(t|pos)]^2 + [P(t|neg)]^2} \tag{20}$$

Fig 18 represents the resultant vector of word ‘beautiful’ from the dataset. The resultant probability vector is the unified probability of the feature.

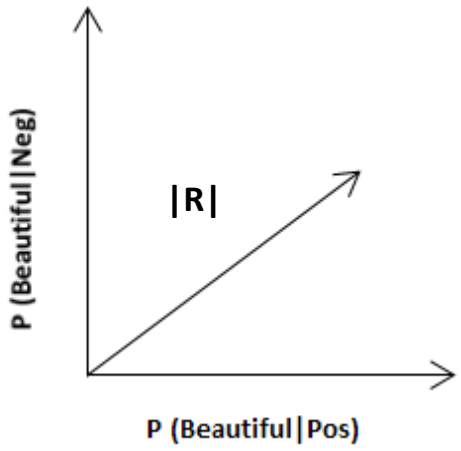

Figure 18: Probability Resultant Vector

The resultant unified probability can be used as a weight for algorithms such as a support vector machine and maximum entropy. The experiment results of this probability unification method are discussed in section 4.

3.1.2 Gradient Method with Tangent Normalization

Another method proposed by this thesis is the Gradient method. This method is also based on a two-dimensional vector but with a slightly different approach. In this method the gradient of the vector is used for the weights of the features. Equation 21 is the formula for calculating the slope or gradient of the vector [14]. The steeper the slope, the more negative the feature becomes. The negativity of a feature is proportional to the steepness of the vector.

$$Grad = \frac{P(t|neg)}{P(t|pos)} \tag{21}$$

The conditional probability of a feature depends on the number of times it has occurred in a class, the number of total words in that class and the total vocabulary size. Therefore, it may vary between 0 and 1. Zero probability is not possible because of the use of Laplace smoothing in the conditional probability formula. However, probabilities can be very low and sometimes raise floating point errors. The probability of a '1' is impossible in the presence of other features but may be close to 1. Suppose the negative probability of a feature is very high in one class and very low in the other, this could make the value of the slope very large.

Another issue when using the raw values of the gradients is the linearity problem. Figure 19 shows the plot of slope values of the resultant vector. The graph rises in a non-linear fashion. As the denominator becomes smaller and approaches to zero, the ratio becomes very high and the graph approaches infinity.

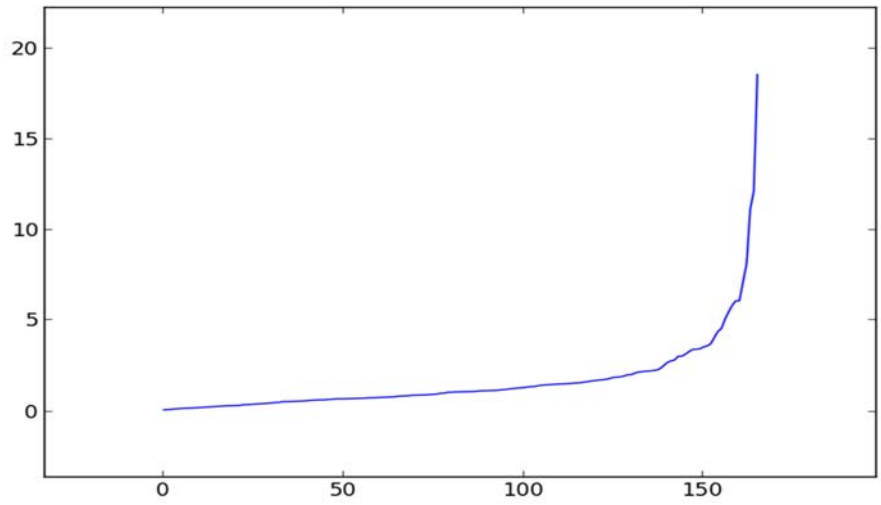

Figure 19: Plot of Gradient Values

Because of these issues, the method is modified to utilize tangent normalization. Instead of using a slope value, the angle becomes the scoring function. Figure 20 shows the graph of the tangent normalized values. It can be seen from the graph that the graph rises in a linear fashion. This also helps to rank the features between the scales of best to worst. This is discussed in detail in section 3.3.

Equation 22 is the tangent normalization equation:

$$\alpha = \tan^{-1} \frac{P(t|neg)}{P(t|pos)} \tag{22}$$

This modification has three benefits;

1. The score remains normalized between 0 and 90 degrees
2. Features can be compared on the basis of their sweetness or bitterness
3. The features can be ranked with respect to the degree of positivity or negativity

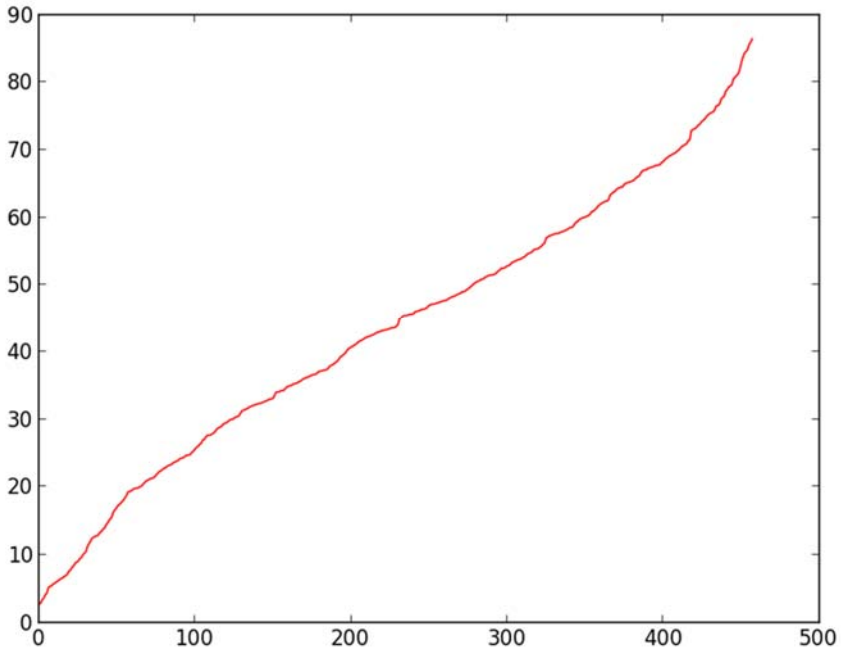

Figure 20: Normalized Values

3.2 Enhanced Ensemble Classifier Framework

Ensemble Classification is an alternative classification technique where multiple classifiers are used to predict a document's class [57]. This technique has been proved to be effective with a noisy dataset [58, 59]. An ensemble classifier is constructed using base classifiers [57]. The approach used for the ensemble classifier is a common technique where multiple classification algorithms are used in parallel and then a voting method is used to choose the result based on the highest number of votes [57, 58, 59].

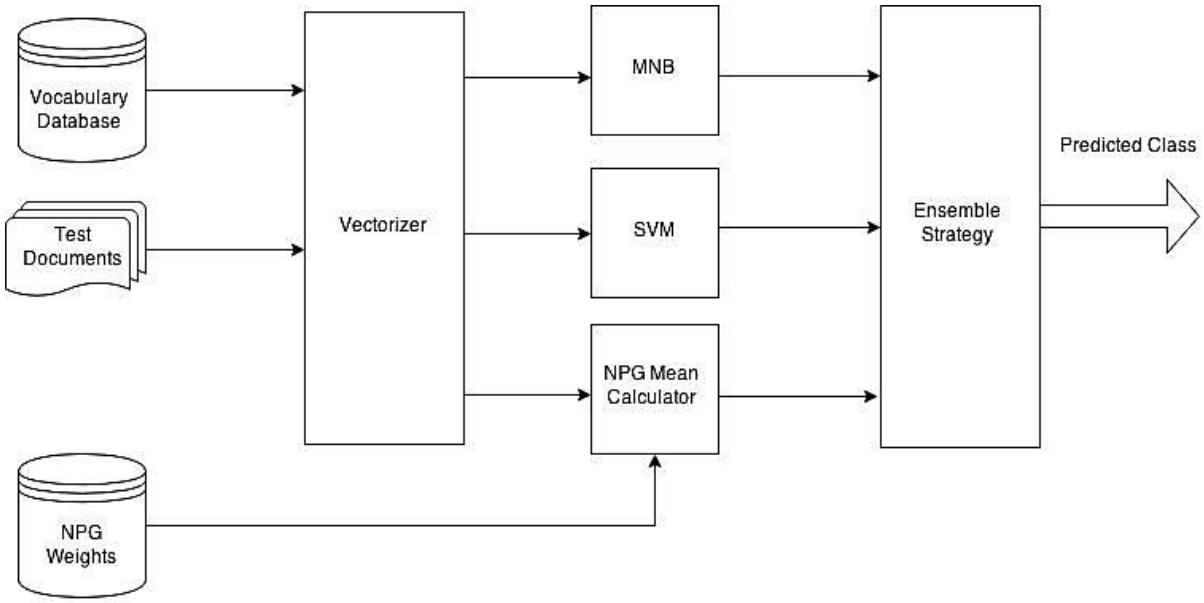

Figure 21: Enhanced Ensemble Classifier Architecture

The proposed ensemble classification technique utilizes the gradient method in parallel with the Naïve Bayes and SVM classifier to predict the class of new documents. Figure 21 shows the Architecture of the proposed ensemble classifier framework.

The working principle is a modified version of the ensemble method. Naïve Bayes and the Support Vector Machine are the base classifiers. When a disagreement occurs between two of the baseline algorithms, the normalized weighing method is used to break the tie. Algorithm 1 shows the working principle of the Enhanced Ensemble Classifier framework.

Algorithm 1: Ensemble Classification

Input: Test Documents

Output: Class

Method:

1. Extract features from test document
2. Create feature vector space
3. Calculate Mean NPFW-G
4. Classify document with SVM and NB
5. Check output from SVM and NB
6. **If** SVM ≠ NB **then**
7. Check Mean NPFW-G value
8. **If** Mean > Threshold **then**
   Output = Positive
9. **else**
   Output = Negative
10. **end if**
11. **end if**

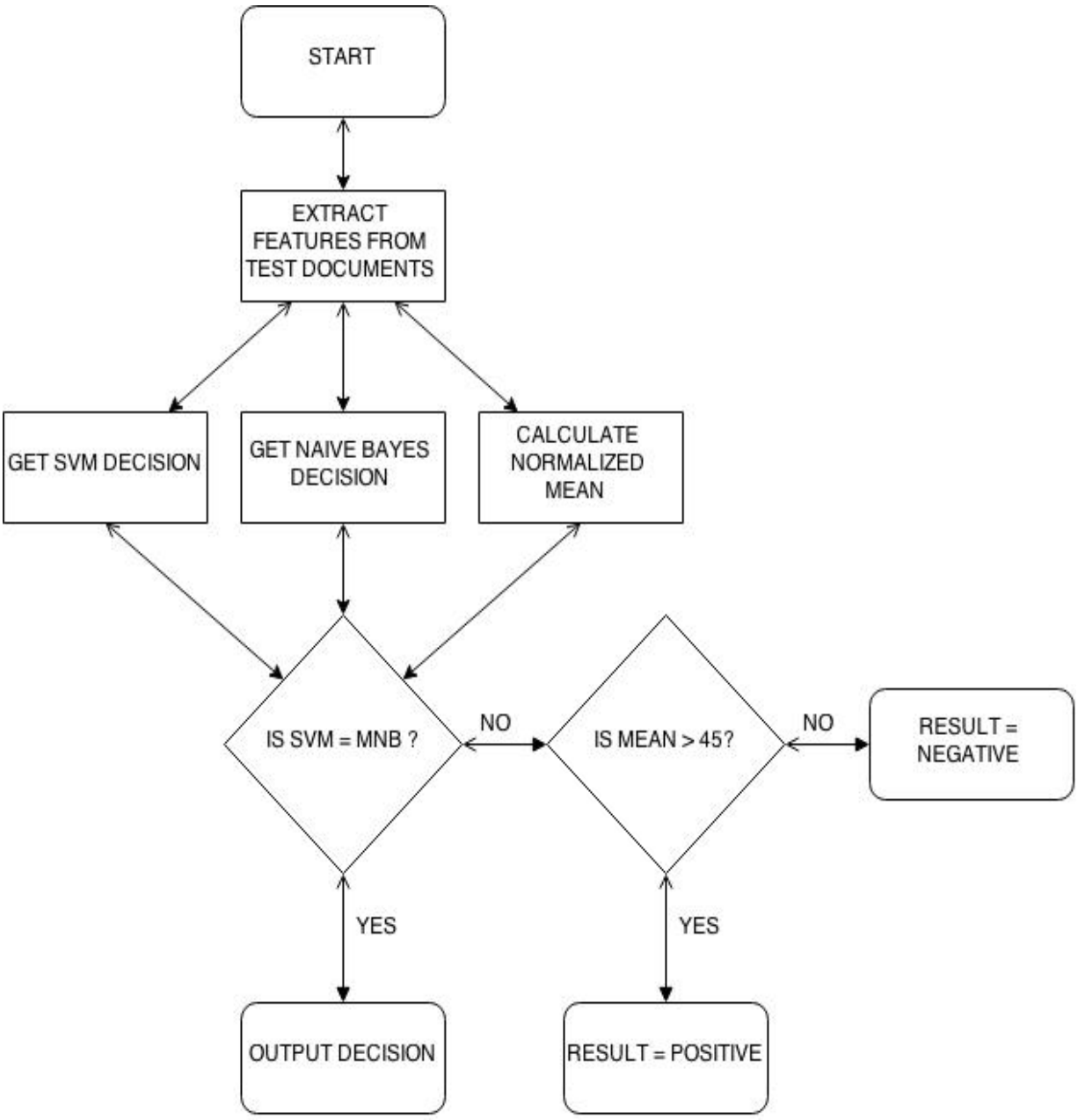

Figure 22: Flow chart of Enhanced Ensemble Classifier

The algorithms are trained with the same datasets and an equal feature set. The Mean Normalized weight of the features in the document is also calculated using Equation 23.

$$\mu = \frac{\sum_{i=1}^{n} tf_n * \alpha_n}{\sum tf_n} \tag{23}$$

Figure 22 shows the flowchart of Enhanced Ensemble Classifier. When a new document arrives, the features are extracted based on the features in the library. Each classifier classifies the document. If both of the classifiers agree, the decision is considered final and the output is considered to the predicted class of the document. If disagreement occurs in between the baseline classifiers, the mean normalized value of the features is checked. If the value is greater than or equal to a threshold angle, it is considered to be a positive document. If the mean value is less than the threshold angle, the document is considered to be a negative document.

3.2.1 Optimal Threshold Estimation

The threshold value should be 45 degrees as the probability ratio is equal to 1 when both positive and negative conditional probabilities are equal. In practice, this does not work all of the time because of the noise in the dataset. Because it is impossible to get an ideal set of features and since the threshold is a mathematical mean of the angles of the features in a document, the threshold may be above or below 45 degrees. The figure below shows the histogram plots for an ideal dataset. The mean values are normally distributed within each class. The red histogram shows the negative dataset and

the green shows positive data instances. The horizontal axis is the mean angle from 0 to 90 degrees where the vertical axis is the probability of the mean angles in a class. For an ideal dataset, the mean of the distribution should be equal to 22.5 for the positive dataset and 67.5 for the negative dataset and the threshold angle is 45 degrees.

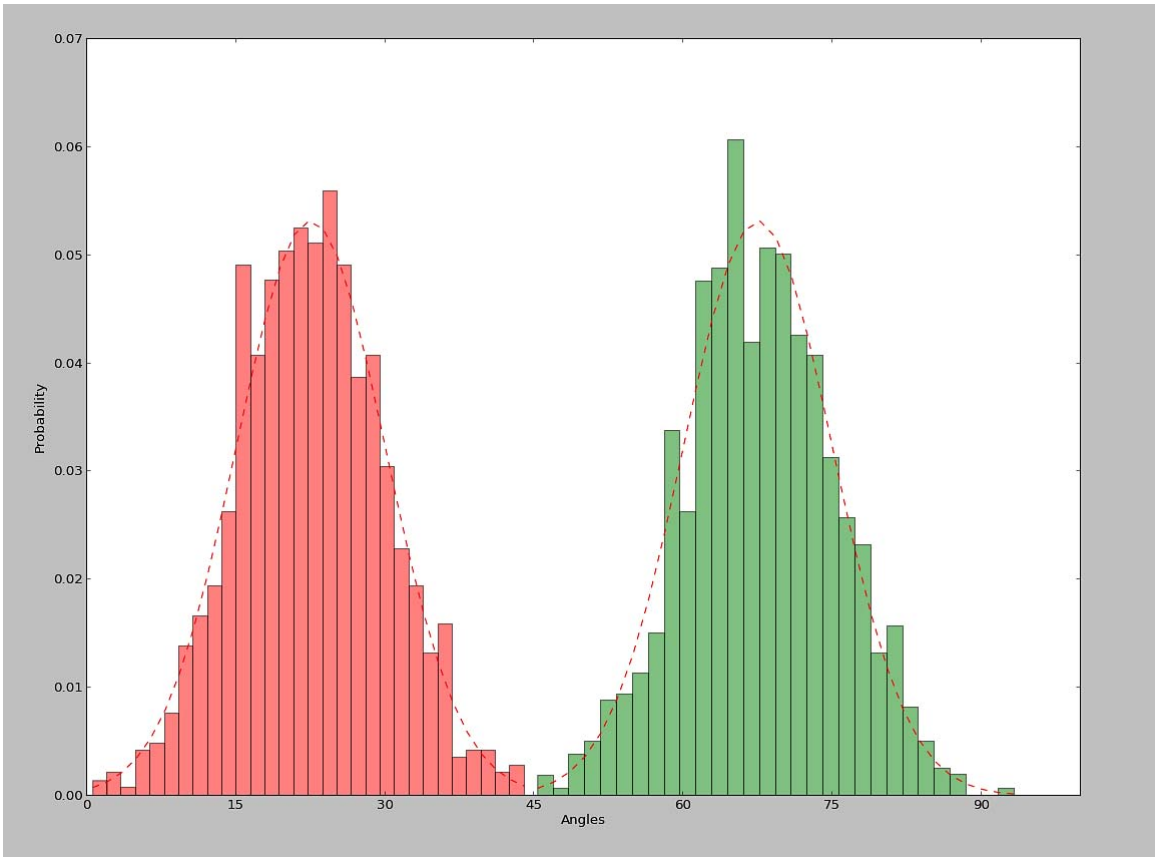

Figure 23: Ideal Histogram plots of Positive & Negative training data

Figure 23 shows the threshold value is 45 degrees in an ideal case. However, the threshold value is not always 45 degrees in real dataset. The value depends on the training dataset and affects the performance of the enhanced ensemble classifier. Figure below shows an example of a more realistic plot.

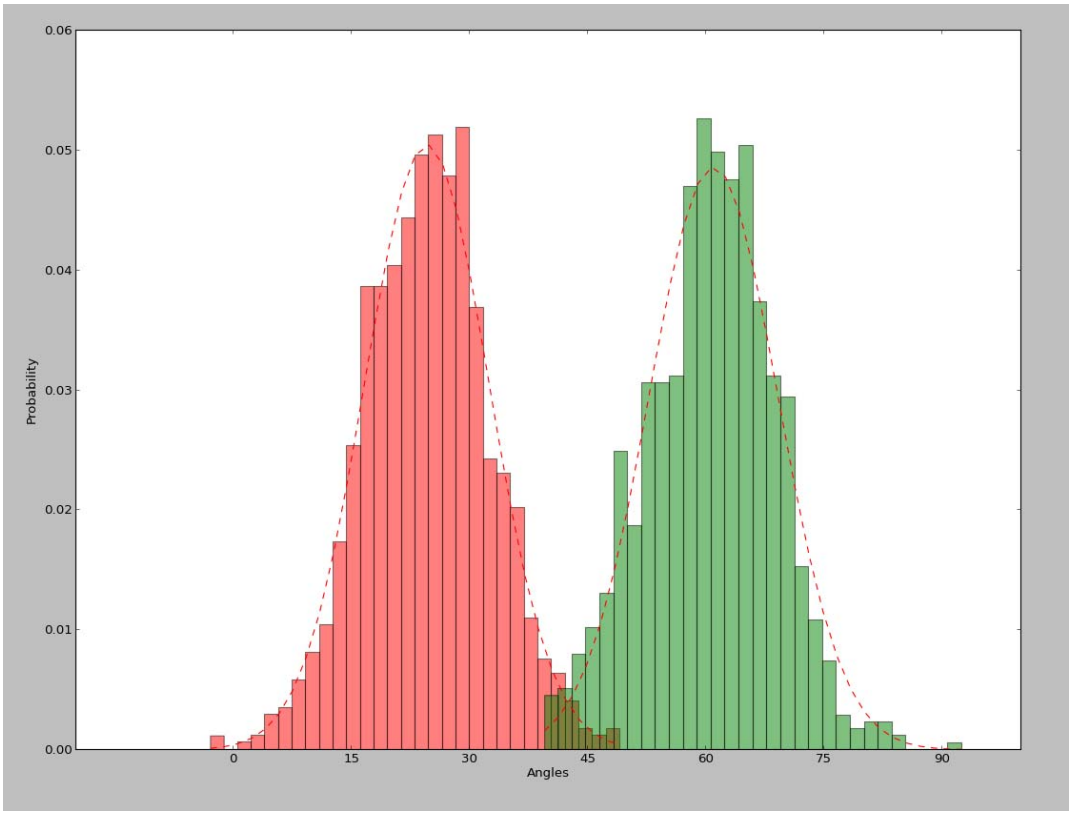

Figure 24: Histogram plots of social media training dataset

Figure 24 shows that the mean values overlaps and the mean values exceeds 45 degrees in negative instances and are below 45 degrees in positive dataset. This can be caused by irrelevant features in the dataset. Since it is impossible to clean the data 100%, this phenomenon cannot be avoided. However, the plots can be helpful in determining the best threshold value for the Ensemble Classifier. The following indicates how to determine better threshold values:

- Calculate mean values of angles for all training documents.
- Plot histograms of positive and negative instances separately.

- Plot Probability Density Function (PDF) of the mean values.
- Look for point of intersection between both PDF plots.
- Select the intersection point as the Threshold value.

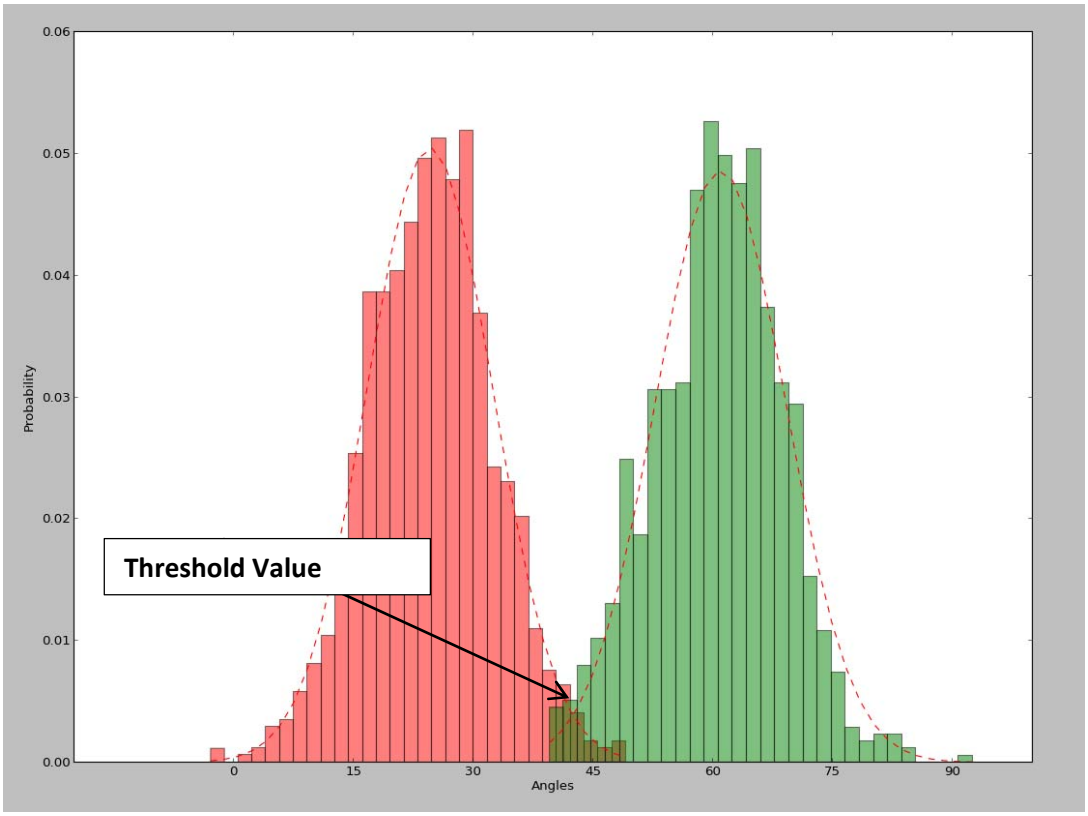

Figure 25: Threshold value to use

Figure 25 shows a threshold value example. The determined value in this case is approximately 42.5 degrees.

3.3 Document Ranking

Document ranking is a more advanced level of sentiment analysis than simply determining positive or negative classes. Ranking documents from best to worst can be useful in some applications. In sentiment analysis, the training data consists of thousands

of documents comprised of both positive and negative sentiments. Since the documents are pre-classified by humans, their polarity differs from one another. For example movie reviews posted on the Internet by users vary in sentiment. Some people may like a movie so much that they post their reviews full of praising words. On the other hand, movie critics may give average comments with both positive and negative sentiments. This can imply a review is average or neutral. Machine learning algorithms help to find the overall sentiment of the documents. Alternately, they can be trained with more precise datasets to differentiate between excellent, good and average documents.

There are various techniques to rank documents according to the degree of their polarity. One technique is the clustering technique. This technique uses the K-means algorithm to find similar features in the dataset and uses them to further divide the documents in sub-categories [61]. Another popular technique is to use WordNet which can be used to find the score of a document based on the features in it [63]. WordNet is a lexical database that was created by Princeton University in 1985. They developed a scoring function to find distance between two words that indicates semantic similarity between them. The nearer their distance is, the closely they are related to each other [63]. Another technique was developed by Singh & Kumar in which they used domain-ontology and statistical method to find semantic similarities between the words [60]. However, they also used WordNet as the weight of their features. Farhadloo & Roland used the approach of nouns instead of adjectives to create their Bag of Nouns representation and applied clustering method to rank documents [62].

This thesis proposes a normalized gradient weighting technique discussed in section 3.1.2 to score documents and rank them in three sub-categories within two super-

classes. The positive class is assigned three categories; definitely positive, positive and marginally positive. The negative class is also assigned three categories; marginally negative, negative and definitely negative. Figure 26 shows the hierarchy of the classes in defined for the test.

Algorithm 2 is the steps for proposed document ranking task.

Algorithm 2: Document Ranking Model Training

Input: Training Documents

Output: NPFW-G Values

Method:

1. Extract features from training documents
2. Create feature vector space
3. Calculate Conditional Probabilities
4. Calculate Gradients for each probability pair
5. Calculate Normalized gradient value for each calculated gradient
6. Store NPFW-G values into feature library

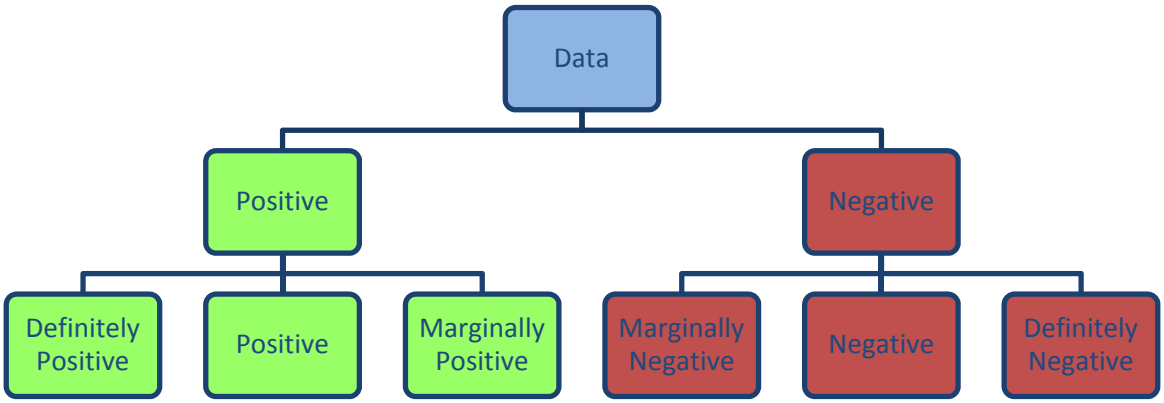

Figure 26: Hierarchy of Classes

Algorithm 3 is the steps for assigning a rank to the new documents based on NPFW-G values.

Algorithm 3: Document Ranking

Input: Test Documents
Output: Rank
Method:

1. Extract features from test document
2. Extract NPFW-G from library
3. Calculate Mean NPFW-G for the document
4. **If** Mean ≥ 45º **then**
   Check Upper Regions
5. **If** Region = 6 **then**
   Output = Definitely Negative
6. **end if**
7. **If** Region = 5 **then**
   Output = Negative
8. **end if**
9. **If** Region = 4 **then**
   Output = Marginally Negative
10. **end if**
11. **else**
    Check Lower Regions
12. **If** Region = 1 **then**
    Output = Definitely Positive
13. **end if**
14. **If** Region = 2 **then**
    Output = Positive
15. **end if**
16. **If** Region = 3 **then**
    Output = Marginally Positive
17. **end if**
18. **end if**

During the training process, the boundaries for the normalized gradient are found. This is done by finding the mean of the training documents and sorting them in ascending

order. The first and the last element in the array represent the boundary of the normalized means. The boundary for the negative documents is between 0 and 45 degrees. Similarly the boundary for the positive documents is between the 45 and 90 degrees. The mean normalized gradient of the document is calculated using Equation 23. After assigning each super-class its initial set of boundaries, we further divide each super-class in three distinct intervals to get 6 sub-categories. These categories are used to rank new documents from most positive to most negative. Fig 27 shows the subcategories on the two dimensional vector spaces.

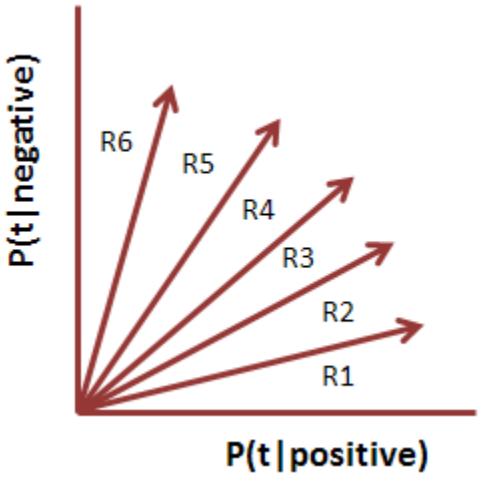

Figure 27: Sub-Classes Regions

4 Experiment & Results

4.1 Datasets

This research in sentiment analysis was conducted on social media. However, other sentiment analysis datasets have also been used for comparative analysis to measure the performance of the classifier outside the scope of social media.

4.1.1 Twitter Dataset

There is a dataset for social media that is available at http://cs.stanford.edu/people/alecmgo/trainingandtestdata.zip. The dataset was collected by researchers from Stanford University and is openly available on the Internet for educational and research purposes. The dataset contains 1.6 million Tweets which contains positive, negative and neutral tweets on multiple topics. The data is the same dataset that is used to create www.sentiment140.com, which is an online testing tool for sentiment analysis. Alec Go, Richa Bhayani, and Lei Huang created this website and were involved in collecting Stanford dataset in 2009.

4.1.2 IMDB Movie Review Dataset

The IMDB movie review dataset contains movie reviews with hand coded polarity for a binary classification task. The core dataset contains 50,000 reviews split evenly into 25000 training sets and 25000 test sets. The overall distribution of labels is balanced (25000 positive and 25000 negative). In the entire collection, no more than 30 reviews are allowed for any given movie because reviews for the same movie tend to

have correlated ratings. The training and test sets contain a disjointed set of movies, so no significant performance is obtained by memorizing movie-unique terms and their associated labels [64].

4.2 Testing Tools

Tools used for development, training and testing the classifiers are described below:

4.2.1 Python

Python is a high-level programming language which is designed with the intention to be highly readable language. It was authored in 1989 by Guido van Rossum, a Dutch computer programmer as a hobby programming project. It can perform same tasks with fewer lines of codes than in any other mainstream programming languages. Python supports multiple programming paradigms from object-oriented to functional or procedural programming. It poses a dynamic type system, an automatic memory management system and has a large and comprehensive library. An important feature is the dynamic name resolution which binds variables and methods during program execution. Python's interpreter and the extensive standard library are freely available for all major platforms from the Python web site [72].

4.2.2 NumPy

NumPy also known as Numeric Python is an extension to Python programming language which adds support for large, multi-dimensional arrays and matrices with an extensive library of high-level mathematical functions to perform on these arrays. It was developed in 2005 by Travis Oliphant as a successor of Numeric which was originally

authored by Jim Hugunin. The core functionality of python is its n-dimensional array. Python was not originally designed for numeric computing which left a vacuum for the development of a module that can do powerful computing such as matlab and similar products [73].

4.2.3 Scikit-learn

Scikit-learn is an open source machine learning library for Python. It comprises of various classification, regression and clustering algorithms including many of the mainstream supervised machine learning algorithms. It is designed to interoperate with python modules like NumPy and SciPy [74].

4.3 Pre-Processing Tests and Classifier Performance Analysis

This section provides the results of the tests performed on the datasets. The tests were conducted in four stages. The first stage tested the individual classifiers and their performance on the datasets as mentioned in the previous section. The performance of the classifiers in the presence and absence of impurities were compared. The data selection technique's effect on the performance of the classifier was also analyzed. The proposed probability feature weighting method was tested. The ensemble classifier framework with normalized feature weights as a scoring function was also tested. Finally, the ranking technique was also tested. The results of these tests are shown in the following sections.

4.3.1 Stop words

This section provides the analysis of the classifiers and the effect of stop words on the feature size, training time and accuracy of the classifiers.

The Table 3 and Table 4 shows the comparison of various attributes of the classifier and the impact of stop words on the efficiency of the algorithms for Twitter dataset and Movie Reviews dataset respectively.

Table 3: Stop Words test on Twitter dataset

Tweets	Before Filtering	After Filtering	Change %
Number of Documents	20,000		
Feature Size	29,108	28,988	-0.412
Total Number of Words	252,941	161,055	-36.33
MNB Accuracy	73.45	74.25	0.8
SVM Accuracy	71.70	73.30	1.60

Table 4: Stop Words test on Movie Reviews

Movie Reviews	Before Filtering	After Filtering	Change %
Number of Documents	2000		
Feature Size	22,822	22,700	-0.54
Total Number of Words	431,217	240,757	-44.17
MNB Accuracy	82.90	84.05	1.15
SVM Accuracy	80.75	80.35	0.4

The results from the tables indicate that removing stop words reduce the number of total words in the datasets up to 45%. This also increases relative weights and conditional probability of individual terms which also increases the accuracy of the classifiers.

4.3.2 Classification using Adjectives as Features

Table 5 and 6 below shows the results of training the classifier on standard adjectives for Twitter and Movie reviews respectively.

Table 5: Results of using standard adjectives as features on Twitter dataset

Number of Adjectives as Features	1328
Accuracy on Naïve Bayes	41.35 %
Accuracy on Support Vector Machine	37.65%

Table 6: Results of using standard adjectives on Movie Reviews

Number of Adjectives as Features	1328
Accuracy on Naïve Bayes	61.60 %
Accuracy on Support Vector Machine	59.95%

The result obtained from the test indicates that the classifier based on adjectives as features is not effective for social media. In the case of the movie reviews, the classifiers still managed to classify the documents around 60% correctly. This is likely due to movie reviews having more features on the order of 500 to 1000 words. A possible reason for low accuracy on the social media dataset is because the words in the

social media dataset lack the use of formal language. Also the number of features is low in comparison for any individual document.

4.3.3 Stemming Results

Table 7 and 8 below compares the feature size in the presence and absence of stemming algorithm applied on the dataset for Twitter and Movie Reviews.

Table 7: Stemming Results on Twitter Dataset

Tweets Dataset	Before Stemming	After Stemming	Change %
Feature Size	18,063	17,045	-5.64
SVM Accuracy	73.45	72.8	-0.65
NB Accuracy	71.7	71.5	-0.20

Table 8: Stemming Results on Movie Reviews

Movie Reviews Dataset	Before Stemming	After Stemming	Change %
Feature Size	22,822	20,540	-9.99
SVM Accuracy	80.35	76.75	-3.60
NB Accuracy	84.05	80.85	-3.20

It can be seen from the above tables that there is a significant difference between the feature sizes in both the tests. The number of features is reduced when the stemming algorithm is applied to the dataset. However, the results obtained from the classifier shows that stemming decreases the performance of the classifier. For the movie reviews dataset, that is formally written, it experiences significant decreased accuracy on both

classifiers. However, tweet data is not significantly affected by stemming. The likely cause of the failure of the stemming algorithm is that, in the process of correcting the data, it also affects the data which does not require any changes to it.

4.3.4 Bigram versus Unigram Analysis

The test were performed using same dataset with equal number of instances. Table 9 and 10 shows the comparison of unigrams to bigrams on the Twitter dataset and Movie Reviews respectively.

Table 9: Unigrams versus Bigrams on Twitter Dataset

Twitter	Unigrams	Bigrams	Change %
Feature Size	5,476	14,187	+61.4 %
NB Accuracy	72.65 %	52.6 %	-17.3 %
SVM Accuracy	71.75 %	57 %	-14.75 %

Table 10: Unigrams versus Bigrams on Movie Reviews

Movie Reviews	Unigrams	Bigrams	Change %
Feature Size	21,387	55,598	+61.53 %
NB Accuracy	83.60 %	61.24 %	-17.91 %
SVM Accuracy	81.01 %	59.98 %	-16.36 %

The result shows that Unigrams outperforms Bigrams in all aspects. The Overall accuracy of the classifiers is approximately 20% higher when Unigrams are used while

the feature size is also reduced up to 61.5%. This means that Unigrams does not only increase accuracy but also reduce training and prediction time.

One of the main reasons of Bigrams to be less effective is that, bigram feature extraction considers two consecutive words as a single feature. This reduces the probability of finding a feature in a test document sharply. Whereas in Unigrams, the chances of getting the appropriate features out of the test documents are higher.

4.3.5 Chi-squared Feature Selection Performance

Table 11 and 12 shows the comparison of the amount of features before and after applying the Chi-square method on the Twitter dataset and Movie Reviews.

Table 11: Chi-square Feature Selection on Twitter Dataset

Tweets	Without Chi-square	With Chi-square	Change %
Feature Size	49,056	8,709	-82.5 %
SVM Accuracy	71.05 %	74.65 %	3.60 %
Naïve Bayes Accuracy	72.65 %	73.89 %	0.24 %

Table 12: Chi-square Feature Selection on Movie Reviews

Movie Reviews	Without Chi-square	With Chi-square	Change %
Feature Size	22,163	18,315	-17.36 %
SVM Accuracy	82.55%	82.95%	0.40 %
Naïve Bayes Accuracy	81.01 %	82.35 %	1.34 %

The Chi-square selection method is very useful in selecting most significant features for classification. It decreased the feature size up to 82.5% and 17.36% in Twitter and Movie Review datasets respectively. It also helped in increasing the performance of the classifiers. The most evident case is the test on Twitter dataset with SVM Classifier where the accuracy was increased up to 3.60%. However, it may not always be a good approach to apply feature selection methods on social media datasets. Since there is a 140 character limit in Twitter posts, the number of words in a post averages between 10 and 12. After filtering stop words, only a handful of them are useful in classification. If features are removed excessively, the classifier is unable to find words in a document and will not classify it. Conversely, movie reviews average around 250-445 words per document and have a higher margin for feature reduction. However, in social media, each and every feature takes on higher importance.

4.4 Comparison of PFW with existing weighing methods

Tests were performed with all three variants of the PFW methods proposed in this thesis. The Support Vector Machine Algorithm was trained using PFW-U (Probability Feature Weights Unified), PFW-G (Probability Feature Weights Gradient) and the Normalized PFW-G. The training utilizes 20,000 Tweets and 10,000 movie reviews to train the classifiers. Table 13 and 14 shows the accuracy of the classifiers with different feature weighing techniques.

Table 13: Accuracy of Individual Weighting Techniques on Movie Reviews

	SVM	NB
TF	73.05	73.15
TF-IDF	72.05	-
PFW-U	62.35	-
Gradients	69.25	-
N-Gradients	64.50	-

Table 14: Accuracy of Individual Weighting Techniques on Movie Reviews

	SVM	NB
TF	80.35	84.05
TF-IDF	60.55	-
PFW-U	56.75	-
Gradients	68.20	-
N-Gradients	74.75	-

This test was conducted to measure the individual performance of the probabilistic weighting methods with the existing techniques. The result from the tests shows that TFs outperforms all the other weighting methods including our developed probabilistic weighting methods. However, they may be helpful to increase accuracy in other classification models but does not work well with Support Vector Machine. Naïve Bayes Classifier on the other hand, is already a probabilistic classifier; therefore it only takes Term Frequencies as weights of the features.

4.5 Enhanced Ensemble Classifier Test

The Enhanced Ensemble Classifier was tested with the baseline algorithms independently. Each classifier was trained on the same dataset with an equal number of features. The classifiers have different accuracy when tested using both datasets. Table 15 shows the individual accuracy of the classifiers on both the datasets. The algorithm's results are uncorrelated which is a prerequisite condition for designing ensemble classifiers.

Table 15 Accuracy of individual Algorithms

	SVM		**MNB**	
	Reviews	**Tweets**	**Reviews**	**Tweets**
Tf	81.01	71.23	84.05	73.15

4.5.1 Optimal Threshold Value Test

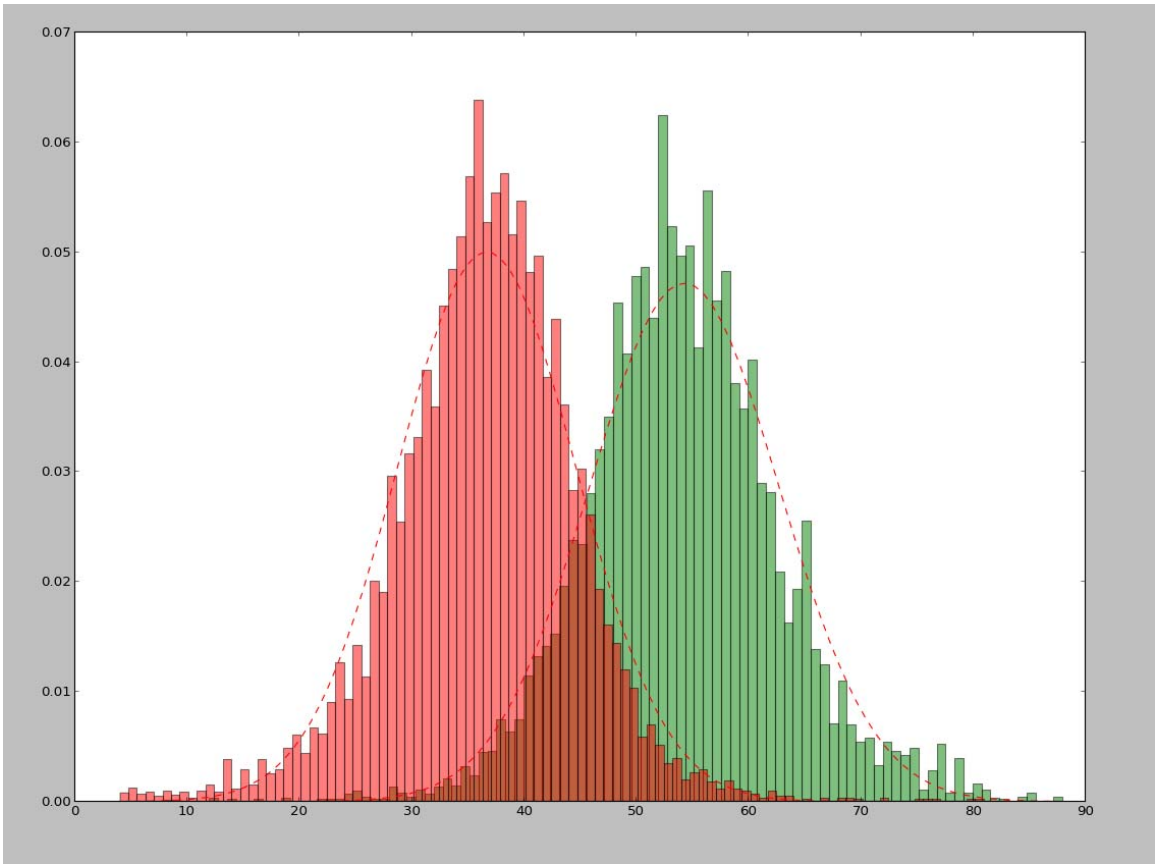

Figure 28: Histogram plots of actual Twitter training dataset used

Figure 28 shows the actual Probability density function graph for the Twitter data set. The value that produced optimal result was found by taking the intersection point of the PDF curves. The Mean values of the dataset also follow a normal distribution. Table 16 shows the selected threshold values for the Twitter dataset. This value produced the optimal result and accuracy of the Enhanced Ensemble Classifier.

Table 16: Standard Deviation (SD), Mean & Threshold value for the Twitter dataset

Twitter	Negative Tweets	Positive Tweets
Standard Deviation (Sigma)	7.69	7.97
Mean	23.47	65.25
Threshold Value	45.28	

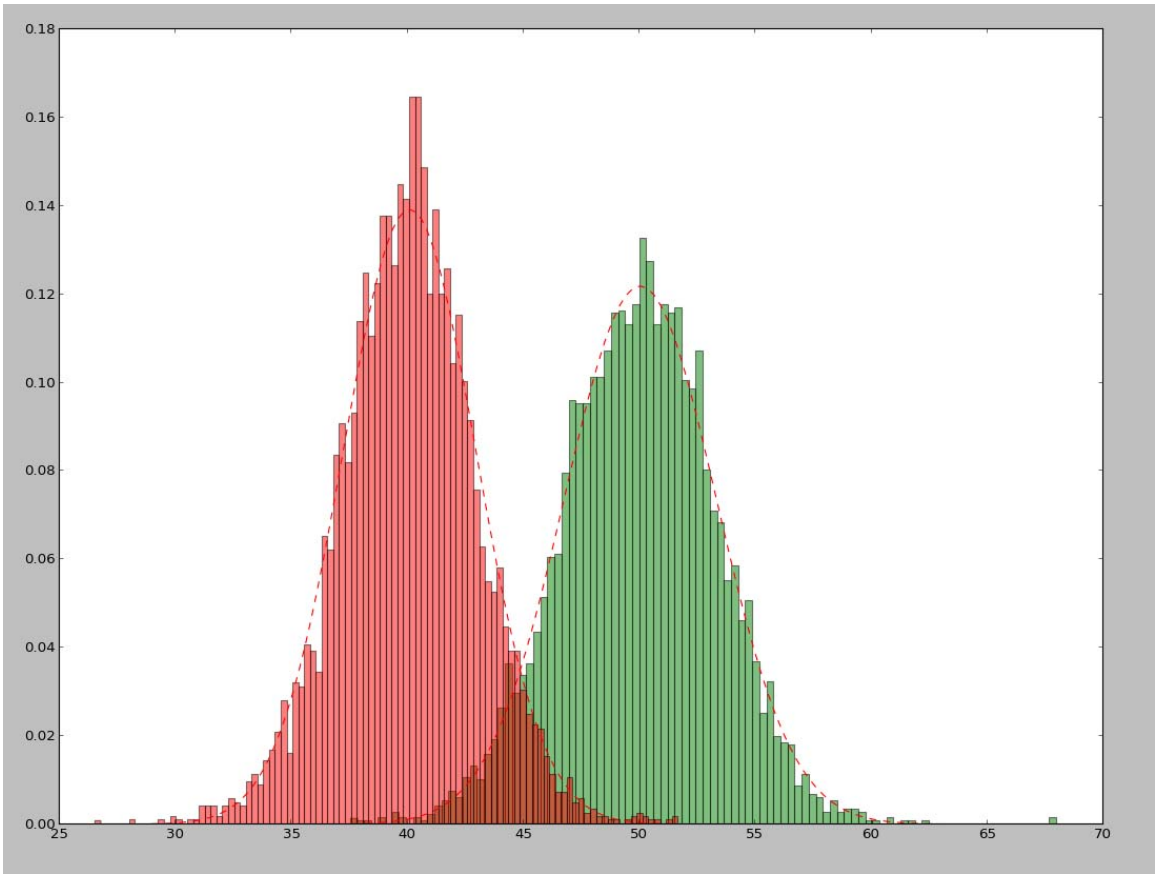

Figure 29: Histogram plot of Movie Review Dataset

Figure 29 shows the probability density function plots of the actual movie review dataset used for the experiment. The PDF curves are intersecting below 45 degrees. Table 17 shows the selected value for the Enhanced Ensemble classifier for movie reviews.

Table 17: Standard Deviation (SD), Mean & Threshold Value for Movie Reviews

Movie Reviews	Negative Reviews	Positive Reviews
Standard Deviation (σ)	7.89	7.95
Mean	22.69	67.58
Threshold Value	43.50	

4.5.2 Enhanced Ensemble Classifier Results

Table 18: Comparison of Enhanced Ensemble Classifier Accuracy on Movie Reviews

	Mean Accuracy	**SD σ**
NB	83.26	1.24
SVM	82.54	0.88
EECF	86.67	0.61

Table 19: Comparison of Enhanced Ensemble Classifier Accuracy on Tweets

	Mean Accuracy	SD σ
NB	72.94	1.6
SVM	71.37	1.13
EECF	76.84	1.18

Table 18 and 19 shows the comparison of the Enhanced Ensemble Classifier with conventional classifying techniques on movie reviews and tweets. Based on the tests shown in Table 18 and 19, the Enhanced Ensemble Classifier is more successful than the classifiers working independently. The NPFW-G contributed in improving the classification when compared to existing methods. The accuracy was increased from 82.54% to 86.67% on Movie Reviews and 71.37% to 76.84% on Twitter Dataset when compared with SVM working individually. Figure 30 shows the overall comparison of the individual classifiers with EEC.

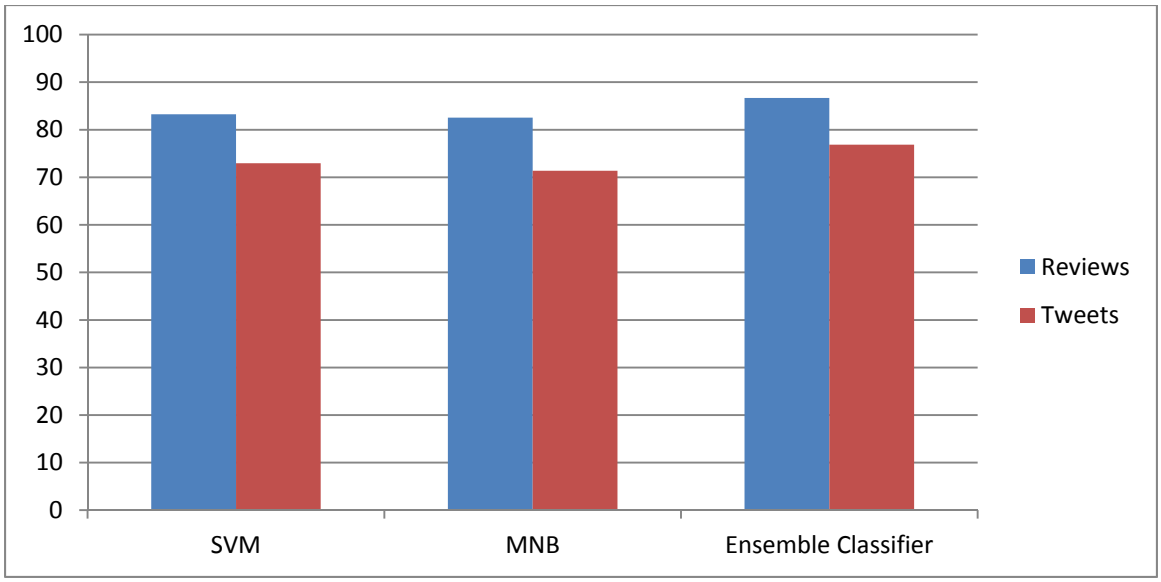

Figure 30: Accuracy Comparison Graph

4.6 Document Ranking with Normalized PFW-G

To perform this test, 100 positive documents and 100 negative documents were manually classified into 6 sub-categories as discussed in section 3.3. The ensemble classifier was used to classify these documents into the two major categories (positive and negative). Out of 200 documents, the classifier predicted 91 and 93 of the documents correctly into their respective classes. In addition, out of the correctly classified documents, the method was able to rank the documents with an accuracy of 82.5%. After further investigation, the performance of the ranking method by varying the size of the features and number of instances used to extract the weights of the features was performed. Figure 31 illustrates the accuracy with respect to the training data provided.

The graph shows that the performance increases by increasing the number of instances. It is because the number of words and vocabulary size also increases with

increasing the number of instances provided for training. More training documents ensure that more samples are provided for every feature that is important for the task of sentiment analysis. Because the weights of the features depend on the conditional probability of the features, it is important to provide a significant amount of training documents.

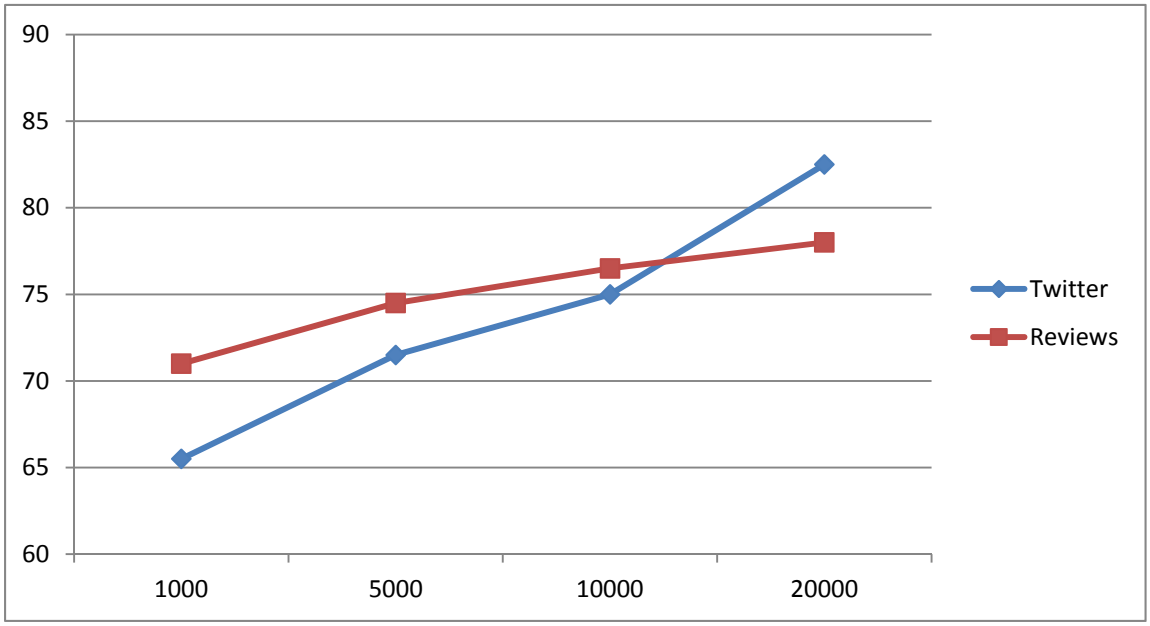

Figure 31: Performance of Document Ranking with number of instances

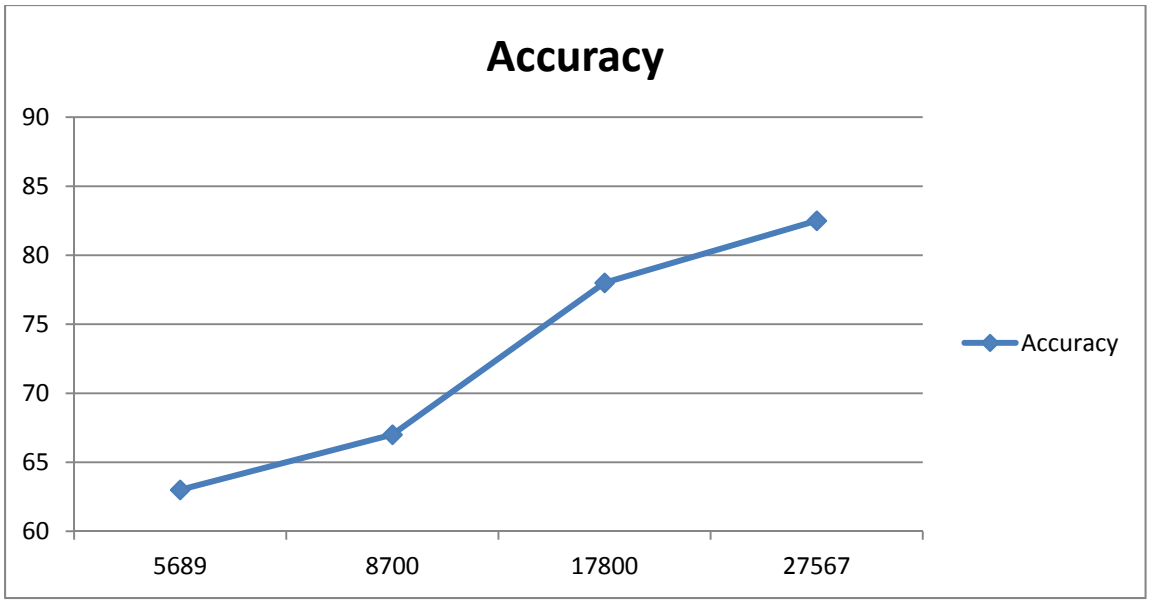

Figure 32: Comparison of Performance of Document Ranking with Vocabulary Size

Figure 32 shows the accuracy of the ranking method with varying vocabulary size. It indicates that the vocabulary size is also important for accurately ranking the documents. With a larger vocabulary size, there is a corresponding increase in correctly ranking the documents. The algorithm increases accuracy in the presence of more available feature size. Since social media documents are smaller it is difficult to find relevant features in the posts. However, the ranking algorithm is very helpful in sub-classifying larger documents.

5 Conclusion & Future Work

5.1 Conclusion

This thesis presented a different approach for sentiment analysis classification which utilizes the probabilistic feature weighting scheme. The proposed methodology was tested on two widely used datasets publically available on the Internet. The outcome of this research shows a 4% improvement in sentiment analysis using the proposed methodology. The improvement was particularly significant on the Twitter dataset but can also be helpful in improving the accuracy in other Text Classification domains.

The weighting scheme provides a method to investigate the impact of features on classification. By calculating the normalized probabilistic weights, one can identify which feature is more or less significant for sentiment analysis. This helps to reduce the number of features and decreases the training time of the algorithms. In addition, individual features can be compared by their normalized probabilistic weight which allows for ranking the features according to their significance in sentiment analysis classification. The quality of the data available for training plays an important role in the accuracy of the methodology. Performing statistical tests on the datasets and high level of pre-processing on the training dataset makes it possible to select significant features from the dataset.

The second part of the research was to develop a classification model based on Ensemble Classification. Although Ensemble Classification has been used by researchers in past, the Enhanced Ensemble Classification Technique developed in this research

provides an advanced methodology utilizing the weights of the features as a part of Ensemble Classifier Framework. This technique was also found to be very effective in classifying the documents with a low number of features such as Twitter posts. Twitter has a limit of 140 characters and this methodology proved to be more successful in classifying them than the individual algorithms. The methodology improved accuracy up to 3 percent. The Enhanced Ensemble Classifier also proved to be very efficient in analyzing Movie Reviews as it outperformed the baseline algorithms such as Naïve Bayes and SVM performing individually. If data is carefully selected and the noise ratio can be reduced by discovering new ways of cleaning data, it may improve the accuracy of EECF not only in sentiment analysis area but also on other text classification problems.

The weighting scheme in this research also helps in ranking documents using the score achieved by the mean normalized weights. It enables the ability to rank documents in terms of their likely degree of positivity or negativity. The algorithm developed for ranking produced results with an accuracy of approximately 90 percent. However, the results may vary depending upon the individual classifying the training data as sentiment may vary from one person to another.

The research also studied various existing techniques to select and clean the data for sentiment analysis. Although these techniques are able to select appropriate features from the datasets, not always they are successful in producing better results with classifiers. Sentiment analysis of social media is one such issue where the documents are relatively small in terms of features and therefore further reducing the feature size is catastrophic.

When comparing sentiment analysis with other text classification problems, the feature set is found to be the main driver for the quality of the classifiers. Naïve Bayes and Support Vector Machines have a higher accuracy on other text classification problems than sentiment analysis. The reason behind this is that, feature sets in other classification problems are disjointed or have a minimum common set of features while sentiment analysis have a more common set of features. This means that both the positive and negative classes consist of features that are common in both classes. For example, the probability of finding a term like soccer, football and cricket is close to zero in weather news. Similarly finding the terms like monsoon, tsunami and hurricane in sports news is also uncommon. That is why disjoint sets produce better results with probabilistic classifiers like Naïve Bayes. On the other hand, Sentiment Analysis datasets contains a significant amount of common features resulting in lower accuracy of the classifiers.

Polarity Confusion is also problem in sentiment analysis which is difficult to overcome. This happens when a document has more than one perspective. A document can be positive from one person's perspective and negative from another. For example, consider the following sentence, "Saskatchewan Roughriders beats Calgary Stampedes in Grey Cup." This kind of sentence is positive when seen from a Roughrider fan's perspective but negative from a Calgary fan's perspective. These kinds of documents cannot be classified correctly using the baseline algorithms. They require more sophisticated techniques which can analyse the structure of the sentence and also take the position of a feature into account.

5.2 Applications of Sentiment Analysis

Sentiment Analysis is useful for many different reasons. Due to the availability of Internet resources all around the globe and the introduction of intelligent mobile devices, social networking and blogging is becoming an integral part of life. Sentiment Analysis can help consumers and stakeholders to make appropriate decisions and choices based on group sentiment. Sentiment Analysis can be used:

- As a feedback mechanism for organizations to obtain people's opinion on the issues that are important for their organizations.
- As a feedback mechanism for consumers who take other opinions into account before making decisions.
- To forecast stocks and markets based on the news on social media and blogs.
- To track down employees those are not loyal to their organizations and speak ill about them on social media.
- To perform surveys with feedback data that may take a long time to do manually.
- By political parties to understand the effect of their political decisions.
- In psychology to study human behaviour.

5.3 Future Work

Sentiment analysis is not a new area of research but can still be improved. This section highlights areas where more research can be done to increase the effectiveness of sentiment classification.

The Probabilistic Feature Weighting technique provides an alternative way to weight features in sentiment analysis. However, more research can be done on unifying the conditional probabilities of a feature such as, mathematical and statistical functions that take multiple variables as input and provide a single output. The PFW can also be tested on text classification datasets that have more than two classes. For example, a neutral class can be added to the dataset to have three classes instead of two. The determination of a neutral class can be complex and challenging problem.

The Enhanced Ensemble Classifier shows significant improvement in the sentiment analysis of social media datasets. It provides an alternate ensemble strategy using the feature weights as a decision factor. The proposed EECF technique requires further research in this area. For example, more classification algorithms can be applied in parallel to the existing baseline algorithm to ensure better accuracy. More research work can be done on ensemble strategies which may use alternative feature weighting methods than can increase the accuracy of the classification. The EECF could also be tested on other text classification datasets.

One of the areas where the research can be done is to determine sarcasm from ambiguous documents. Sarcasm is one of the things that are difficult to determine. Sarcastic document have a completely opposite polarity to what has been written. Humans have the power to identify sarcasm but current sentiment analysis techniques have failed to do so. Artificial intelligence in future may make it possible to design systems that would also be able to identify sarcasm.

6 References

[1] C. Kingston, Twitter for Beginners, Social Media DIY Workshop for Small Business, Crow Communications, 2011.

[2] N. Taylor, Choosing between social media platforms and understanding the markets they reach, Journal of Digital & Social Media Marketing, Volume 1, Number 3, pp. 283 – 291, 2013.

[3] B. Pang, L. Lee, Sentimental Education: Sentiment Analysis Using Subjectivity Summarization, Proceedings of ACL, pages 271-278, 2004.

[4] A. Jose, Twitter Sentiment Analysis, Seminar Report, National Institute of Technology Calicut, 2010.

[5] B. Pang, L. Lee, Opinion mining and sentiment analysis, Foundations and Trends in Information Retrieval, Volume 2 Issue 1-2, 2008.

[6] S. Wang, C. Manning, Baselines and Bigrams: Simple, Good Sentiment and Topic Classification, Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, Volume 2, pages 90-94, 2012

[7] A. Pak, P. Proubek, Twitter as Corpus for Sentiment Analysis and Opinion Mining, LREC 2010

[8] A. Sharma, S. Dey, Performance Investigation of Feature Selection Methods for Sentiment Analysis, ACCTHPCA June 2012

[9] E. Frank, R. R. Bouckaert, Naïve Bayes for Text Classification with Unbalanced Classes, PKDD 2006 Pages 503-510

[10] S.M. Kamaruzzaman, C.M. Rahman, Text Categorization using Association Rules and Naïve Bayes Classifier, AJIT, Vol. 3, No. 9, pp 657-665, Sep. 2004

[11] M. Karamibekr, Ali A. Ghorbani, Sentiment Analysis of Social Issues, ICSI 2012

[12] F. Neri, C. Aliprandi, F. Capeci, M. Cuadros, Sentiment Analysis on Social Media, IEEE/ACM 2012

[13] Istvan Pilaszy, Text Categorization and Supprt Vector Machine, Proceedings of the 6th International Symposium of Hungarian Researchers on Computational Intelligence, 2005

[14] Neethu M S, Rajasree R, Sentiment Analysis in Twitter using Machine Learning Techniques, IEEE-31661

[15] A Hassan, A Abbasi, D Zeng, "Twitter Sentiment Analysis: A Bootstrap Ensemble Framework, Social Computing (SocialCom), 2013 International Conference on, On page(s): 254 – 263

[16] I. Rish, T.J. Watson Research Center, An empirical study of the naive Bayes classifier

[17] Michael Collins, The Naïve Bayes Model, Maximum-Likelihood Estimation, and the EM Algorithm [Online]

[18] NLP Stanford, Text Classification and Naïve Bayes

[19] Kevin P. Murphy, Binomial and multinomial distributions

[20] Quan Yaun, Gao Cong, Nadia M. Thalmann, Enhancing Naïve Bayes Classifier with Smoothing Methods, In proceedings of the 21st international conference companion on the world wide web, WWW '12 Companion, Pages 645-646, New York, USA, 2012, ACM.

[21] Brian C. Lovell and Christian J Walder, Support Vector Machines for Business Applications

[22] Andrew Ng, Stanford University, Support Vector Machines

[23] Thomas G Dietterich, Ensemble Methods in Machine Learning, Oregon State University, Corvallis, Oregon, USA

[24] Akhlaqur Rahman, Sumaira Tasnim, Ensemble Classifiers and Their Applications: A Review, International Journal of Computer Trends and Technology (IJCTT) – volume 10 number 1 – Apr 2014

[25] Rinat Khoussainov, Andreas Heß, Nicholas Kushmerick, Ensembles of Biased Classifiers, School of Computer Science and Informatics, University College Dublin, Belfield, Dublin 4, Ireland

[26] Lior Rokach, Ensemble-based classifiers, Published online: 19 November 2009, Springer Science+Business Media B.V. 2009

[27] Eleanor Clarka and Kenji Arakia, Text normalization in social media: progress, problems and applications for a pre-processing system of casual English, Pacific Association for Computational Linguistics (PACLING 2011)

[28] C.Ramasubramanian, R.Ramya, Effective Pre-Processing Activities in Text Mining using Improved Porter's Stemming Algorithm, International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 12, December 2013

[29] Fehmi Ben Abdesslem, Iain Parris, and Tristan Henderson, Reliable Online Social Network Data Collection

[30] Alec Go, Richa Bhayani, Lei Huang, Twitter Sentiment Classification using Distant Supervision

[31] Farah Benamara, Carmine Cesarano, Diego Reforgiato, Sentiment Analysis: Adjectives and Adverbs are betterthan Adjectives Alone

[32] Minoru SASAKI & Kenji KITA, Rule-Based Text Categorization Using Hierarchical Categories

[33] Simon Carter, Manos Tsagkias, Wouter Weerkamp, Twitter hashtags: Joint Translation and Clustering

[34] Lisa Posch, Claudia Wanger, Phillip Singer, Markus Strohmaier, Meaning as Collective Use:Predicting Semantic Hashtag Categories on Twitter

[35] Allison Shapp, Variation in the Use of Twitter Hashtags, New York University

[36] I.Hemalatha, Dr. G. P Saradhi Varma, Dr. A.Govardhan, Preprocessing the Informal Text for efficient Sentiment Analysis, International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), Volume 1, Issue 2, July – August 2012, ISSN 2278-6856

[37] N.H.N.D. de Silva, M.K.D.T. Maldeniya & D.N.C. Wijeratne, Subject Specific Stream Classification Preprocessing Algorithm for Twitter Data Stream

[38] Anjali Ganesh Jivani, A Comparative Study of Stemming Algorithms, Anjali Ganesh Jivani et al, Int. J. Comp. Tech. Appl., Vol 2 (6), 1930-1938

[39] J.J. Verbeek, Supervised Feature Extraction for Text Categorization

[40] Xiaojin Zhu, Andrew B. Goldberg, Michael Rabbat & Robert Nowak, Learning Bigrams from Unigrams

[41] William B. Cavnar and John M. Trenkle, N-Gram-Based Text Categorization

[42] Mohammadreza Keyvanpour, Reza Tavoli, Feature Weighting for Improving Document Image Retrieval System Performance

[43] Georgios Paltoglou, Mike Thelwall, A study of Information Retrieval weighting schemes for sentiment analysis

[44] Pascal Soucy & Guy W. Mineau, Beyond TFIDF Weighting for Text Categorization in the Vector Space Model

[45] Juan Ramos, Using TF-IDF to Determine Word Relevance in Document Queries

[46] ZHANG Yun-tao, GONG Ling & WANG Yong-cheng, An improved TF-IDF approach for text classification, Journal of Zhejiang University SCIENCE ISSN 1009-3095

[47] Hassan Saif, Miriam Fernandez, Yulan He, Harith Alani, On Stopwords, Filtering and Data Sparsity for Sentiment Analysis of Twitter

[48] Inderjit S. Dhillon, Dharmendra S. Modha, Concept Decompositions for Large Sparse Text Data using Clustering

[49] John C. Duchi, Michael I. Jordan, H. Brendan McMahan, Estimation, Optimization, and Parallelism when Data is Sparse

[50] Mykola Pechenizkiy, The Impact of Feature Extraction on the Performance of a Classifier: kNN, Naïve Bayes and C4.5

[51] Sanasam Ranbir Singh, Hema A. Murthy, Timothy A. Gonsalves, Feature Selection for Text Classification Based on Gini Coefficient of Inequality, JMLR: Workshop and Conference Proceedings 10: 76-85

[52] Zhaohui Zheng, Xiaoyun Wu, Rohini Srihari, Feature Selection for Text Categorization on Imbalanced Data

[53] Text Classification & Naïve Bayes, April 1, 2009 Cambridge University Press. [Online]

[54] Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features (pp. 137-142)

[55] *Vector Resultants.* Retrieved from physicsclassroom.com/class/vectors/Lesson-1/Resultants [Online]

[56] *Gradient* Retrieved form wikipedia.org/wiki/Gradient [Online]

[57] B. Verna, A. Rahman, Cluster-Oriented Ensemble Classifier: Impact of Multicluster Characterization on Ensemble Classifier Learning, IEEE Transactions On Knowledge And Data Engineering, Vol. 24, No. 4, April 2012

[58] M. R. Kaleem, D. M. Farid, An Adaptive Ensemble Classifier for Mining Complex Noisy Instances in Data Streams, 3rd International Conference On Informatics, Electronics & Vision 2014

[59] Z. Ouyang, M. Zhou, T. Wang, Q. Wu, Mining Concept-Drifting and Noisy Data Streams using Ensemble Classifiers, 2009 International Conference on Artificial Intelligence and Computational Intelligence

[60] P. Chahal, M. Singh, S. Kumar, Ranking of Web Documents using Semantic Similarity, Information Systems and Computer Networks (ISCON), 2013 International Conference

[61] G. Li, F. Liu, A Clustering-based Approach on Sentiment Analysis, Intelligent Systems and Knowledge Engineering (ISKE), 2010 International Conference

[62] M. Farhadloo, E Rolland, Multi-Class Sentiment Analysis with Clustering and Score Representation, 2013 IEEE 13th International Conference on Data Mining.

[63] *WordNet,* Retrieved from wordnet.princeton.edu [Online]

[64] Potts, Christopher. 2011. on the negativity of negation. In Nan Li and David Lutz, eds., Proceedings of Semantics and Linguistic Theory 20, 636-659.

[65] Lovins, Julie Beth (1968). "Development of a Stemming Algorithm". Mechanical Translation and Computational Linguistics 11: 22–31

[66] Porter, Martin F. (1980); An Algorithm for Suffix Stripping, Program, 14(3): 130–137

[67] Maurya, Pandey (2013), Effective Information Retrieval System, IJETAE, ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 3, Issue 4

[68] Forman, George (2003). "An extensive empirical study of feature selection metrics for text classification". Journal of Machine Learning Research 3: 1289–1305

[69] Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze (2009), An Introduction to Information Retrieval, Cambridge University Press, Cambridge, England. Web: http://www.informationretrieval.org

[70] Zheng-Jun Liu ; Qi Li ; Zhi-Wei Xia and Qi Wang, "Target recognition for small samples of ladar range image using classifier ensembles", Opt. Eng. 51(8), 087201 (Aug 06, 2012)

[71] http://blog.touchedeclavier.com/wp-content/uploads/2013/06/comment-utiliser-les-hashtags.gif

[72] http://en.wikipedia.org/wiki/Python_(programming_language)

[73] http://en.wikipedia.org/wiki/NumPy

[74] http://en.wikipedia.org/wiki/Scikit-learn

[75] http://techneerajnandwana.blogspot.ca/2013/04/stop-words-in-full-text-search.html?view=mosaic&m=1