Sentiment Analysis on Twitter Data

Submitted By

Anika Rahman

ID: 2013-1-60-054

Ahamad Ali

ID: 2013-1-60-016

Department of Computer Science and Engineering

East West University

Supervised By

Dr. Mohammad Rezwanul Huq

Assistant Professor

Department of Computer Science and Engineering

East West University

Dhaka, Bangladesh

December, 2016

Sentiment Analysis on Twitter Data

Submitted By

Anika Rahman
ID: 2013-1-60-054
Ahamad Ali
ID: 2013-1-60-016

Supervised By

Dr. Mohammad Rezwanul Huq
Assistant Professor,
Department of CSE, EWU.

A project

Submitted in partial fulfillment of the requirements

for the degree of Bachelor of Science to

Computer Science and Engineering

Dhaka, Bangladesh

December, 2016

Declaration

This thesis has been submitted to the department of Computer Science and Engineering, East West University in the partial fulfillment of the requirement for the degree of Bachelor of Science in Computer Science and Engineering by us under the supervision of Dr. Mohammad Rezwanul Huq, Assistant Professor at Department of CSE at East West University under the course 'CSE 497' . We also declaring that this thesis has not been submitted elsewhere for the requirement of any degree or any other purposes. This thesis complies with the regulations of this University and meets the accepted standards with respect to originality and quality. We hereby release this thesis to the public. We also authorize the University or other individuals to make copies of this thesis as needed for scholarly research.

_____

Anika Rahman

Id: 2013-1-60-054

Department of Computer Science and Engineering

East West University.

_____

Ahmad Ali

Id: 2013-1-60-016

Department of Computer Science and Engineering

East West University.

Letter of Acceptance

The thesis entitled “Sentiment Analysis on Twitter Data” submitted by Anika Rahman, ID 2013-1-60-054 & Ahamad Ali, ID 2013-1-60-016 to the department of Computer Science & Engineering, East West University, Dhaka 1212, Bangladesh is accepted as satisfactory for partial fulfillments for the degree of Bachelor of Science in Computer Science & Engineering on December 2016.

Board of Examiners

1_____

Dr. Mohammad Rezwanul Huq

Assistant Professor — Supervisor

Department of Computer Science and Engineering

East West University, Dhaka, Bangladesh

2_____

Dr. Md. Mozammel Huq Azad Khan

Professor and Chairperson — Chairperson

Department of Computer Science and Engineering

East West University, Dhaka, Bangladesh

Acknowledgements

First of all, we are grateful to the Almighty God for establishing us to complete this research. We wish to express our sincere thanks and gratitude to my advisor Dr. Mohammad Rezwanul Huq, Assistant Professor at Dept. of CSE for the continuous support during our thesis study and related research, for his patience, motivation and immense knowledge. His guidance helped us in all the time of research and writing of the thesis. We will always be grateful for having the opportunity to study under him.

We are thankful to all of our teachers, Department of CSE, East West University. We also grateful to all of our primary and secondary school teachers who are our first teachers in our life and initiator of our basic knowledge.

We would like to express our thankful to our parents and siblings for supporting us spiritually throughout writing this thesis. And we are thankful to all our friends. And at last we again thanks to the creator Allah for everything.

Abstract

Every day using micro blogging millions of users share opinions on various topics. Twitter is a very popular micro blogging site where users are allows a limit of 140 characters; this kind of restriction makes the users to be concise as well as expressive at the same moment. For that reason it's become a rich source for sentiment analysis and belief mining. For the same reason we become interested to work with twitter data. The aim of this project is to develop such a functional classifier which can accurately and automatically classify the sentiment of an unknown tweet.

In this thesis, we propose techniques to classify the sentiment label accurately. Therefore, we introduce two techniques: one of the technique is sentiment classification algorithm (SCA) and the other one is a machine learning algorithm SVM. For both SCA and SVM we calculate weights based on different features. Then in SCA, build feature vector we build pair of tweets by using different features. From those pair we measure the Euclidian distance for every tweet with its pairs. From those distance we only consider nearest 8 tweets label to classify that tweet. On the other hand in SVM, build a matrix from the calculated weights based on different features and by applying PCA (principle component analysis) we try to find k eigenvector with the largest Eigen values. From this transformed sample dataset we try to find the best c and best gamma by using grid search technique to use in SVM. Finally, we apply SVM to assign the sentiment label of each tweet in the test dataset. In both algorithms we use confusion matrix to calculate the accuracy.

In out we have found that SCA always perform better than the SVM. We also evaluate the performance of these two techniques for different size of datasets.

Contents

List of Figures

List of Tables

Chapter 1

1 Introduction

Microblogging websites have become a source of various kinds of information. In micro blogs, people post real time messages about their opinions on various topics such as discuss current issues, complain and express positive sentiment for products they use in daily life. Twitter is such a platform where people expressing and sharing their thoughts and opinions on all kinds of information or opinions about products, politicians, events etc. Sentiment detection of tweet is one of the basic research topics over twitter data and it is very useful because it allows feedback to aggregate without manual intervention. From the sentiment analysis many people like consumer, businessman or may be typical person get benefit and many companies study user reactions and reply to users on micro blogs. One challenge is to build technology to detect and summarize an overall sentiment. In this paper, we look at Twitter and build models for classifying "tweets" into positive and negative sentiment.

1.1 Tweets reflect sentiment

A huge amount of social media including news, forums, product reviews and blogs contain numerous sentiment-based sentences. Sentiment is defined as a personal belief or judgment that is not founded on proof or certainty and sentiment expressions may describe the mood of the writer (happy/anger/sad/bored/grateful/...) or the opinion of the writer towards some specific entity (A is great/I hate A, etc.).

Twitter micro blogging service, a huge amount of frequently self-standing short textual tweets became openly available for the research community and many of these tweets contain a wide variety of user-defined hash tags. Some of these tags are sentiment tags which assign one or more sentiment values to a tweet. Twitter data for classification of a wide variety of sentiment types from text.

Emotions can be expressed by tweets. There are six basic human emotions such as anger, disgust, fear, joy, sadness and surprise. Tweets are reflected those emotion very clearly. Suppose someone is very well rested, their tweet will be positive, outgoing and engaging. Thus, their tweets may have positive sentiment and emotions (e.g., joy, happiness, love). These users may also tweet during regular working hours, produce original content, and engage with other users. By Evaluating content one tweet can easily detect if someone is depressed. For example, a study found that increased negative sentiment and greater expression of religious involvement may suggest someone is depressed. Some users may even tweet about their depression-related thoughts (e.g., thoughts of hurting themselves). Positive tweets inspire users to tweet more positively. On average who tweeted negatively had been exposed to about 4.34 about 4.34 percent more negative tweets and those who tweeted positively had seen about 4.5 percent more positive content and this is found by the researchers.

1.2 Motivation

We have chosen to work with twitter since we feel it is a better approximation of public sentiment as opposed to conventional internet articles and web blogs. It includes rich structured information about the individuals involved in the communications. For example it maintains information of who follows whom and re-tweets and tags inside of tweet provide discourse

information. Another reason is that the amount of relevant data is much larger for twitter, as compared to traditional blogging sites. The response of twitter is more prompt and more general. Sentiment analysis of public is highly critical in macro-scale socio economic phenomena like predicting the stock market rate of a particular firm and this could be done by analyzing overall public sentiment towards that firm with respect to time and using economics tools for finding the correlation between public sentiment and the firm's stock market value. At the same time firms can also estimate how well their product is responding in the market, which areas of the market is it having a favorable response and in which a negative response as twitter allows us to download stream of geo-tagged tweets for particular locations. By getting such information firms can find the reason behind geographical different response and they can try to make solution for negative response. As a result they can market their product in a more optimized manner. Predicting the results of popular political elections and polls is also an emerging application to sentiment analysis. One such study was conducted by Tumasjan et al. in Germany for predicting the outcome of federal elections in which concluded that twitter is a good reflection of offline sentiment.

1.3 Research Questions

We propose two techniques for sentiment analysis. One of the technique is KNN and another technique is SVM. Both techniques work with same dataset and same features.

1.3.1 Why do we need Sentiment Analysis?

Everyday social networks, blogs and other media produced huge amount of data in the World Wide Web. This huge amount of data contain very crucial opinion related information that can be used to benefit for businesses and other aspects of commercial and scientific industries. If

we try to manually track and extract this useful information which is impossible. For that reason sentiment analysis is required. Sentiment analysis is the process which extract sentiments or opinions from reviews which are given by users over a particular subject, area or product in online. We can be categories the sentiment as positive or negative. Which determines the general attitude of the people to a particular topic.

1.3.2 Why do we use KNN?

The KNN is an easy algorithm to understand and implement. We think that it will be a powerful tool for our proposed technique of sentiment analysis. In our technique for each tweet we calculate weight based on some features and by using KNN a distance measure can be calculated consistently between a tweets of test set with a tweets of training set. Consider the K nearest neighbor assign the sentiment of unknown sample.

1.3.3 Why do we use SVM?

From the feature weight we get some value. We consider 5 features in our experiment and we get appoint in 5 dimensional space for each tweet. Then we can easily apply SVM. Another reason behind use SVM is that we classify the tweets into two category positive and negative and for this type of binary classification SVM may be a good choice.

1.4 Thesis Overview

Our main goal is to correctly detect sentiment of tweets as more as possible. Our thesis has two main parts; first one is to detect sentiment of tweets by using some feature and in the second one we use machine learning algorithm SVM. In both cases we use 5 fold cross validation. We propose two techniques to detecting sentiment analysis:

Sentiment Classification Algorithm: In this technique from the input dataset we make feature vectors by using different features like as word feature, n-gram feature, punctuation feature, pattern feature and keyword based feature. After build feature vector we build pair of tweets by using different features. From those pair we measure the Euclidian distance for every tweet with its pairs. From those distance we only consider nearest 8 tweets label to classify that tweet. After completing this classifying step we measure the accuracy by using confusion matrix.

Support Vector Machine (SVM): In this technique we use those feature weight which we use in sentiment classification technique. From that weight which we are calculate based on different features we build a matrix and by applying PCA (principle component analysis). We try to find k eigenvector with the largest Eigen values. From this transformed sample dataset we try to find the best c and best gamma by using grid search technique to use in SVM. Finally, we apply SVM to assign the sentiment label of each tweet in the test dataset.

Later on we compare our two techniques in respect to accuracy level of detecting the sentiment accurately. We found that Sentiment Classifier Algorithm performs better than SVM.

Chapter 2

2 Background

Twitter data has an impressive predictive power. Using this power everything from the stock market, elections to movie box performance, through pandemics are amenable to such forecasting. Twitter is such a platform where people express and discuss their opinions on current issues As a result it is helpful for variety of people. In this paper we analysis the sentiment over twitter data.

For this reason we use some machine learning algorithm. They are Support Vector Machines (SVM), Principal Component Analysis (PCA), Grid search and Confusion Matrix. Using such algorithms we calculate the accuracy of our experiment. The basic information of such topics is given below:

2.1 Support Vector Machines (SVM)

A Support Vector Machine (SVM) is a supervised machine learning algorithm[1] . It can be used for both classification and regression purposes. In classification problems SVMs are most commonly used and SVMs are based on the idea of finding a hyper plane that best divides a dataset into two classes, as shown in the image below.

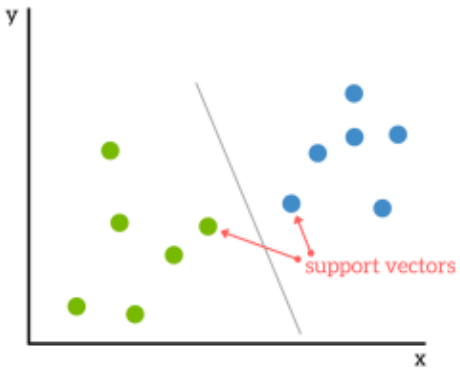

Figure 2.1 : Support Vectors

Support Vectors

The data points nearest that are nearest to the hyper plane is called support vectors and the points of a data set that, if removed, would alter the position of the dividing hyper plane. This is because, they can be considered the critical elements of a data set.

Hyper plane

If any classification task with only two features. A hyper plane for this (above the image) as a line that linearly separates and classifies a set of data. The further from the hyper plane our data points lie, the more confident we are that they have been correctly classified and the distance between the hyper plane and the nearest data point from either set is known as the margin. So we can say the goal is to choose a hyper plane with the greatest possible margin between the hyper plane and any point within the training set, giving a greater chance of new data being classified correctly.

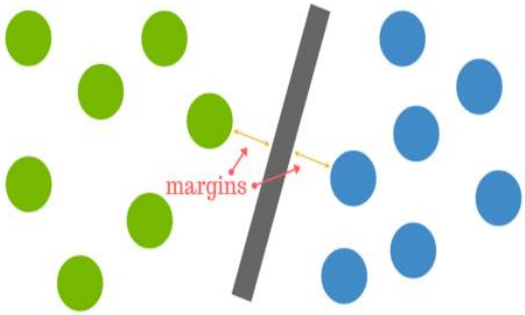

Figure 2.2 Hyper plane

2.2 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is mainly used for the analysis of data to identify patterns and finding patterns to reduce the dimensions of the dataset with minimal loss of information[2] . A possible application would be a pattern classification task. Here we try to reduce the computational costs and the error of parameter estimation. It can be possible by reducing the number of dimensions of our feature space by extracting a subspace that describes our data "best".

We consider that our goal is to reduce the dimensions of a d-dimensional dataset by projecting it onto a k-dimensional subspace (where k <d) and for this we have to compute eigenvectors which are the components from our data set. We also collect them in a so-called scatter-matrix. We can alternatively calculate them from the covariance matrix. All the eigenvectors is associated with an Eigen value. These Eigen Value tells us about the "length" or "magnitude" of the eigenvectors. If all the Eigen values are of very similar magnitude, this is a good indicator that our data is already in a good subspace. Some of the Eigen values are much higher than others. As the higher Eigen Values contain more information about our data distribution, so we might be interested in keeping only those eigenvectors with the much larger Eigen values. On the other hand, Eigen values that

are close to 0 are less informative. So when we construct the new feature subspace, we might consider in dropping those.

2.3 Confusion matrix

Confusion matrix mainly describe the performance of a classifier model and it contains information about actual and predicted classifications done by a classification system[3]. For evaluating the performance of such systems we have to use the data in the matrix. The following table shows the confusion matrix for a two class classifier.

- **TN** is the number of **correct** predictions that an instance is **negative**.
- **FP** is the number of **incorrect** predictions that an instance is **positive**.
- **FN** is the number of **incorrect** of predictions that an instance **negative**. and
- **TP** is the number of **correct** predictions that an instance is **positive**.

		Predicted	
		Negative	Positive
Actual	Negative	**TN**	**FP**
	Positive	**FN**	**TP**

Table 2.1 :Confusion Matrix

Several standard terms have been defined for the 2 class matrix

$$\text{fp rate} = \frac{FP}{N} \qquad \text{tp rate} = \frac{TP}{P}$$

$$\text{precision} = \frac{TP}{TP+FP} \qquad \text{recall} = \frac{TP}{P}$$

$$\text{accuracy} = \frac{TP+TN}{P+N} \qquad \text{F-measure} = \frac{2}{1/\text{precision}+1/\text{recall}}$$

- **Accuracy**: The accuracy means the proportion of the total number of predictions that is correct.
- **TP (true positive) rate:** The TP means the proportion of positives that are correctly identified.
- **FP (false positive) rate:** The FP means the proportions of negatives that are incorrectly identified.
- **Precision**: The precision means the proportion of positively identified that are correct.
- **F-measure:** A measure that combines precision and recall. It is the harmonic mean of precision and recall.

2.4 Grid search

The usual way of performing hyper parameter optimization is grid search, or a parameter sweep[4]. It is an exhaustive searching through a manually specified subset of the hyper parameter

space of a learning algorithm and this algorithm must be guided by some performance metric. These performance metric typically measured by cross-validation on the training set or evaluation on a held-out validation set. As the parameter space of a machine learner may include real-valued or unbounded value spaces for certain parameters, manually set bounds and discretization may be necessary before applying grid search.

As an example, a typical soft-margin SVM classifier equipped with an RBF kernel has at least two hyper parameters. These parameters need to be tuned for good performance on unseen data: a regularization constant *C* and a kernel hyper parameter γ. Performing grid search, we have to selects a finite set of reasonable values for each as both parameters are continuous. We can say,

$$C \in \{10, 100, 1000\}$$

$$\gamma \in \{0.1, 0.2, 0.5, 1.0\}$$

Grid search trains an SVM with each pair (C, γ) in the Cartesian product of these two sets and evaluates their performance on a held-out validation set (or by internal cross-validation on the training set, in which case multiple SVMs are trained per pair). Finally, the grid search algorithm outputs the settings that achieved the highest score in the validation procedure. Grid search suffers from the curse of dimensionality, but is often embarrassingly parallel because typically the hyper parameter settings it evaluates are independent of each other.

Chapter 3

Related work

The contributions of this paper[5] are to introduction POS-specific prior polarity features and to explore the use of a tree kernel to obviate the need for tedious feature engineering. Looking at popular micro blog Twitter, here the authors build models for two classifying tasks. These are a binary task of classifying sentiment into positive, negative classes and 3 way tasks means to classifying sentiment into positive, negative and neutral classes. They also experiment with unigram model, a feature based model and a tree kernel based model. In the tree kernel based model they design a new tree representation for tweets. Their tree kernel based model out performs both these models by a significant margin. In the unigram based model, their experiment shows that a hard baseline is indeed for achieving over 20% over the chance baseline for both classification tasks. Their feature based model that uses only 100 features achieves similar accuracy as the unigram model that uses over 10,000 features. They also combining unigrams with their features and features with the tree kernel. Both these combinations outperform the unigram baseline by over 4% for both classification tasks. In this paper, they present extensive feature analysis of the 100 features they propose. For both the classifying tasks their parts-of-speech tags are most important. Standard natural language processing tools are useful even in a genre which is quite different from the genre on which they were trained. Both the tree kernel model and best feature based models perform roughly and it does not require detailed feature engineering. For this experiment they use manually annotated Twitter data and one of the advantages of this data over the previous used dataset is that tweets are collected in a streaming fashion. These can represent a true sample of actual tweets in terms of language and content.

In this paper[6] by proposing an approach to automatically detect sentiment on Twitter messages (Tweets) and also propose two step sentiment analysis classification method for Twitter. First classifies messages as subjective and objective and further distinguishes the subjective tweets as positive or negative. For creating these classifiers, instead of using manually annotated data to compose the training data as regular supervised approaches, they leverage sources of noisy labels as their training data. These noisy labels were provided by a few sentiment detection websites over twitter data. For better utilizing these sources it is important to verify the potential value of using and combining them. Providing an analysis of the provided labels and examining different strategies of combining these source in order to find the best result. A more robust feature set that captures a more abstract representation of tweets and it is composed by meta-information associated to words and specific characteristics of how tweets are written is also proposed by the authors. In Meta-features, they map a given word in a tweet to its part-of-speech using a part-of-speech dictionary as POS tags are good indicators for sentiment tagging. . In POS tags, they map the word to its prior subjectivity (weak and strong subjectivity), and polarity (positive, negative and neutral). The prior polarity is switched from positive to negative or vice-versa when a negative expression (as, e.g., "don't", "never") precedes the word. They obtained the prior subjectivity and polarity information from subjectivity lexicon of about 8,000 words used in (Riloff and Wiebe, 2003). Although this is a very comprehensive list, slang and specific Web vocabulary are not present on it, e.g., words as "yummy" or "ftw". Popular words used on online discussions from many online sources were collected by the authors and added them to the list. For syntax feature they exploited the syntax of the tweets and the frequency of each feature in a tweet is divided by the number of words in the tweet. An effective and robust sentiment detection approach for Twitter

messages is presented by this paper which uses biased and noisy labels as input to build its models. The main limitation of our approach is the cases of sentences that contain antagonistic sentiments.

By investigating the utility of linguistic features for detecting the sentiment of Twitter messages, the author of this paper[7] evaluate the usefulness of existing lexical resources as well as features to capture information about the informal and creative language used in microblogging. For building training data they take a supervised approach but leverage existing hashtags in Twitter data. In their experiment they use three different corpora of Twitter messages. For development and training they use hash tagged dataset (HASH) and the emoticon dataset (EMOT). There are three steps for preprocessing the dataset. They are 1) tokenization, 2) normalization and 3) parts-of-speech (POS) tagging and they also use a variety of features for their classification and experiment. They use unigrams and bigrams for the baseline and they also include features typically used in sentiment analysis such as sentiment lexicon and POS features.

In n-gram features for identifying a set of useful n-grams, at first they remove stop words and then they perform rudimentary negation detection by attaching the word not to a word that follows a negation term. Finally all the unigram and bigrams are identified by the training data according the information they gained.

In Lexicon features, subjectivity lexicon are tagged with their prior polarity: positive, negative, or neutral. They create three features based on the presence of any words from the lexicon.

In Part-of-speech features, for each tweet they build features to count the number of verbs, adverbs, adjectives, nouns and parts of speech.

In Micro-blogging features, to capture the presence of positive, negative, and neutral emoticons and abbreviations and the presence of intensifiers they create binary features.

Their experiments[8] show that part-of-speech features may not be useful for sentiment analysis in the micro blogging domain. Features from an existing sentiment lexicon were somewhat useful in conjunction with micro blogging features. For automatically classifying the sentiment of Twitter messages, in this paper the authors introduce a novel approach and they classified these messages as either positive or negative with respect to a query term. For this reason they build a framework that treats classifiers and feature extractors as two distinct component. This framework allows them to easily try out different combinations of classifiers and features extractors. Normalizing the effect of query terms along with corresponding tweets. Their assumption is that users prefer to perform sentiment analysis about a product and not of a product. When a user enters a query 'ABC', they normalize the sentiment carried by 'ABC' itself. For example, the tweet ABC is hardly interesting should be classified as negative. If the word "ABC" by itself has a positive sentiment, it would bias the results. As a result each query term is presented as a QUERY TERM equivalence class, and allows them to normalize the effect it has on classification.

By Stripping out the emoticons causes the classifier to learn from other features such as unigrams and bigrams present in the tweet. An interesting side effect of their feature is that they use non-emoticon to determine the sentiment. They consider emoticons as noisy labels because they are not perfect at defining the correct sentiment of a tweet. For example: @BATMANNN :(i love chutney..... Without the emoticon, most people would probably consider this tweet to be positive. Tweets with these types of mismatched emoticons are used to train our classifiers because they are difficult to filter out from our training data.

The Twitter language model has many unique properties such as **usernames** and **usage of links** and taking advantage of the following properties to reduce the feature space.

Tweets can contain repeated letters. For example by searching “hungry” with an arbitrary number of u’s nonempty result set. For this reason they use preprocessing so that any letter occurring more than two times in a row is replaced with two occurrences. They also use multinomial Naïve Bayes model.

In Maximum Entropy models one should prefer the most uniform models that satisfy a given constraint. It is like feature-based models. In a two class scenario, it is the same as using logistic regression to find a distribution over the classes and it makes no dependence assumptions for its features.

They also use support vector machine classification technique. Here their output data are two sets of vectors of size m. Each entry in the vector corresponds to the presence a feature.

In this paper they show that machine learning algorithms (Naive Bayes, maximum entropy classification, and support vector machines) can achieve high accuracy for classifying sentiment when using this method.

Chapter 4

4 Working Procedure

We use a variety of features for our classification experiments. For the baseline, we use word feature, n-gram feature, pattern feature and punctuation feature [9]. Finally, we also include another feature key based feature. Calculated weight based on features for each tweet. Then we use our techniques.

4.1 Features

In this section, we present detail about our features which we use in our techniques. For each tweet we calculated weight based on features. We apply the features proposed in [9] with some necessary modification and we added an extra feature.

4.1.1 Word Feature

Each word feature in tweet considered as binary feature. For counting the word feature of a tweet it compare with a dictionary which is used to detect which words are stop word and which are not. Stop words are out of consideration. Otherwise every word is considered for word feature. Moreover, if we encounter sequences of two or more punctuation symbols inside a tweet, we consider them as word features. Additionally, the common word RT, which means "retweet", does not constitute a feature because it may be appear in the majority of tweets inside the dataset. So, It may not make a significant effect in the result rather it lead to burden in classification task. When we calculate word feature weight we calculate is as $ws = \frac{Nf}{coun(f)}$ where Nf represent the number

of feature present in the tweet and count (f) represent the number of total feature present in the whole dataset. We use this formula for all the features of our experiment.

4.1.2 N-Gram Feature

A sequence of 2-5 consecutive words in a sentence is regarded as a binary n-gram feature. The tweet which contains more rare words that has higher weight than which contain common words and it has made a greater effect on the classification task.

4.1.3 Pattern Feature

In Pattern Features, the words are divided into three categories. They are high-frequency words (HFWs), content words (CWs) and regular words (RWs). When a word frequency is considered as f which frequency in the dataset is represent as frf and it will be considered as high frequency word if frf>FH. On the other hand, if frf<FC, then f is considered to be content word and the rest of the words are considered as regular words. The word frequency is calculated from all the words of the dataset and it is estimated from the training set rather than from an external corpus. We treat as HFWs all consecutive sequences of punctuation characters as well as URL, REF, TAG and RT meta-words for pattern extraction as they play an important role in pattern detection. Pattern as an ordered sequence of HFWs and slots for content words. The upper bound for FC is set to 1000 words per million and the lower bound for FH is set to 10 words per million. FH is set to 100 words per million and we provide a smaller lower bound as the experimental evaluation produced better results. Observing that the FH and FC bounds allow overlap between some HFWs and CWs. By addressing this issue, we follow a simple strategy as described next.iff$_{rf}$€$(FH, FH + \frac{FC}{2})$the word is classified as HFW, elseif f$_{rf}$€$[FH + \frac{FC}{2}, FC)$the word is

classifed as CW. We seek for patterns containing 2-6 HFWs and 1-5 slots for CWs. Moreover, we require patterns to start and to end with a HFW, thus a minimal pattern is of the form [HFW][CW slot][HFW].

4.1.4 Punctuation Feature

The last feature type is divided into five generic features as follows: 1) tweet length in words, 2) number of exclamation mark characters in the tweet, 3) number of question mark characters in the tweet, 4) number of quotes in the tweet and 5) number of capital/capitalized words in the tweet. The weight wp of a punctuation feature p is defined as $wp = (3 * Np)/(Mp * (Mw + Mng + Mpa))$, where Mw; Mng; Mpa declare the maximal values for word, n-gram and pattern feature groups, respectively. So, wp is normalized by averaging the maximal weights of the other feature types.

4.1.5 Key Word Based Feature

In key based feature we use a list[13] where there are 18000 words with its sentiment strength whose range are 1 to -1.based on those words strength we calculate the key based feature weight.

Source Code Available here: “https://github.com/aababu/sentiment-analysis”

4.2 Pseudo code of our experiment

```
1. declear a user define class type array d
2. declear double type array wf,ngf,puf,kf,pf
3. declear two dimensional array pair,dcm
4. declear a integertype variable i,ktp,kfp,ktn,kfn,stp,sfp,stn,sfn
```

```
5. declear kaccuracy,kprecision,krecall,kfscor,ktpr,kfpr
6. declear a string td
7. declear double type variable c,gamma
8. declear saccuracy,sprecision,srecall,sfscor,stpr,sfpr
9. d←GetContent(input_file);  //input_file is a text file where
                                Dataset is stored
//Get all feature weight
10. wf ← Getwordfeatureweight(d);  //store weight of tweets based on
                                Weight feature
11. ngf← Getngramfeatureweight(d);     //store weight of tweets
                                based on n-gram feature
12. puf← Getpunchuationfeatureweight(d); // store weight of
                                tweets based on punchuation
                                feature
13. pf←Getpatternfeatureweight(d);  // store weight of tweets
                                based on pattern feature

14. kf← Getkeywordbaseweight(d); // store weight of tweets based
                                on keywordbase feature
15. d.assign(wf,ngf,puf,pf,kf); //store those value to the appropriate
                                member variables of d using function

//Sentiment Classification Algorithm
16. pair← Getmatchingvectors(d);     // store each tweets in d`s
                                pair information
```

```
//apply 5 fold
17. set i ←— 0;
18.  while i is less than 5
19.     ttd ←—— d/5            //test data subset of d
20.     td←—— (d-(ttd))          //training data subset of d
//calculate eucledian distance with its pair
21.   dcm←— caldisatnce(td)    // dcm is array contains distance
                                between a tweet and its pairs
22.  (ktp,kfp,ktn,kfn) <- applyknn(dcm);    //predict based on
                                              the sentiment
23.  set i ←—— i+1
24. (kaccuracy,kprecesion,krecall,kfscore,ktpr,kfpr) ←—
    calculaterperformance(ktp,kfp,ktn,kfn)
//support vector machine
25. td ←— applypca(d);  //td is an array which contain
                          transformed dataset of d

26. (c,gamma) ←— applygridsearch(td);// return best pair of c
                                       and gamma
27. (ktp,kfp,ktn,kfn)←— applysvm(td,cgamma);//predict is array
                                              of prediction
                                              result From svm
28. (saccuracy,sprecesion,srecall,sfscore,stpr,sfpr) ←——
     calculaterperformance(stp,sfp,stn,sfn)
```

4.3 Experiment Evaluation

Our objective is to detect sentiment correctly as more as possible. For doing that we propose two technique sentiment classification by using some feature and another is machine learning algorithm SVM. From the very beginning we describe our working procedure below: first we collect tweet from the site[10] which are not labeled with sentiment label. when we collect tweet form the site we get the tweet as like below:

"60730027 6320951896 @thediscovietnam coo. thanks. just dropped you a line. 2009-12-03 18:41:07"

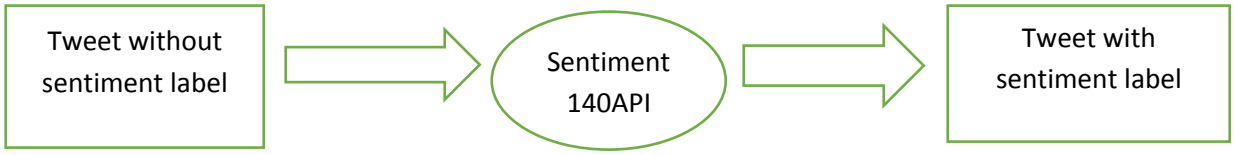

Figure 4.1: preparing dataset

here 60730027 represent the user id ,6320951896 represent tweet id , @thediscovietnam coo. Represent the user name. " thanks. just dropped you a line. " this is actually the tweet and the last part is the date and time when the user tweet .

After getting this kind of tweet we send it to the [11] the sentiment140 api . For find the sentiment label of tweets we create a file where we store the tweets whose we want to labeled. After that open the terminal and goto the directory where the file location is.going to that location following one line command need to execute

curl –data-binary @file.txt "http://www.sentiment140.com/api/bulkClassify?

After executing this command we get the output of above tweet is

"2",""6073002763209518 96 @thediscovietnam coo. thanks. just dropped you a line.2009-12-03 18:41:07"""

here 2 represent the sentiment label of the tweet.t here may be three type of sentiment label return given tweet whose are indicate by some integer number and those number and their label is given below:

The polarity values are:

- 0: negative
- 2: neutral
- 4: positive

We get a dataset where the tweets contain the sentiment label. We also calculate the length of the tweet for punctuation feature , count the number of words for word feature , number of n gram for n-gram feature and then we find the high frequency word, low frequency word and content word for calculating the pattern weight feature. We also try to detect if there are stop word present in the tweets based on a list [12] which contain around 500 stop words. We use a list [13] which contain around 20000 word with its sentiment label strength -1 to 1 for calculate the key word base features. After building all the requirements for calculating feature weight we calculate the feature Weight for the each tweets in the dataset based on these features. This feature weighted dataset is used to both techniques to classify an unknown tweet.

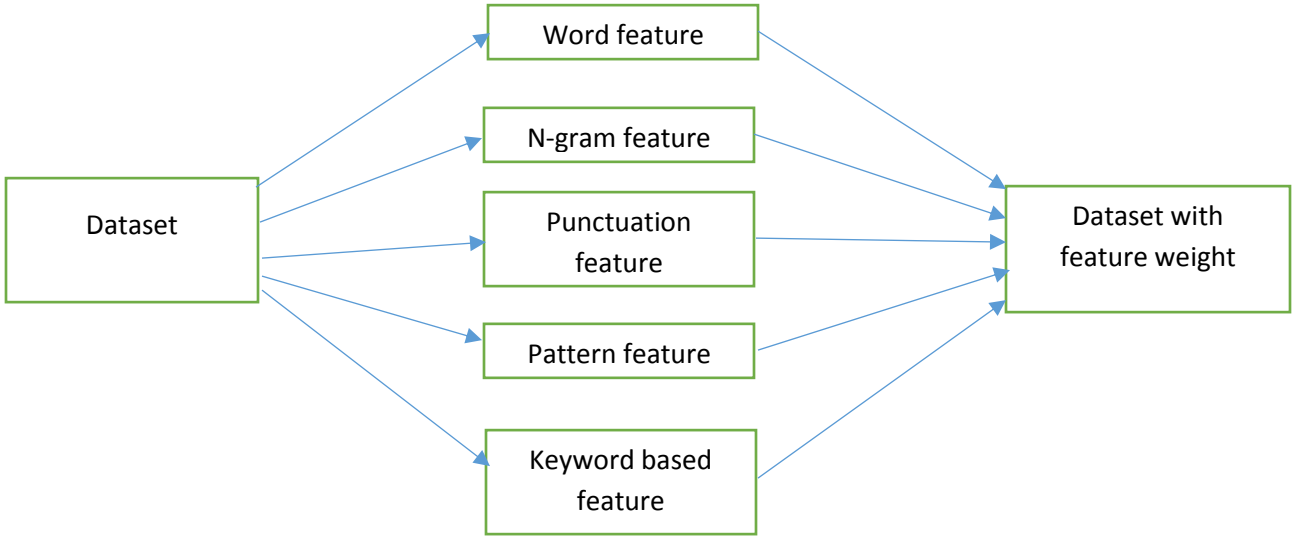

Figure 4.2: calculating weight

A. Sentiment Classification Algorithm

For all the tweets which we use in our dataset make pair with others tweets in the dataset based on the features. For example, when we calculate pair based on the word feature we consider two tweet will be pair if there is a common word among them

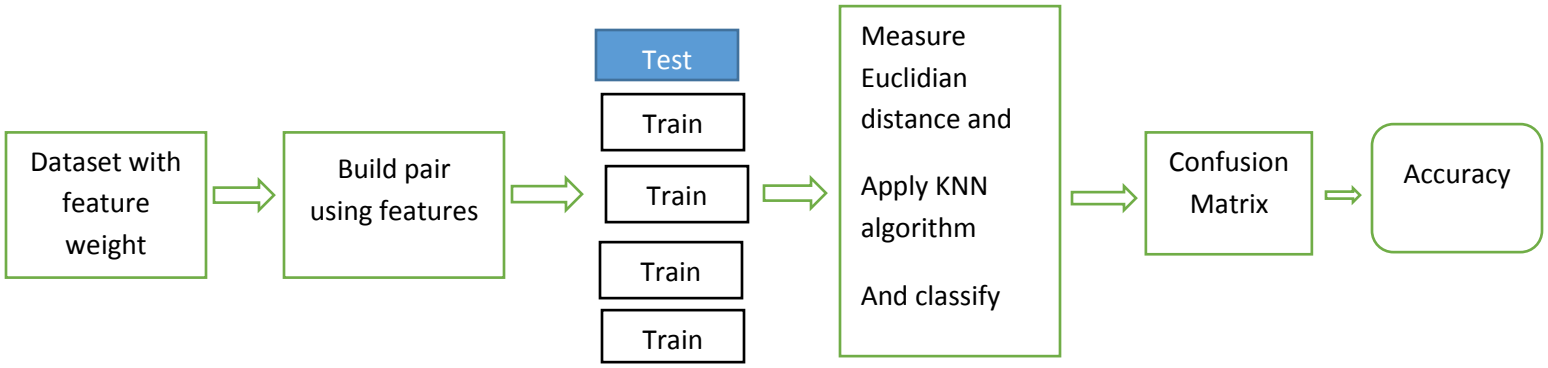

Figure 4.3: working flow of SCA

Suppose a word “happy” we found in the tweet x and the same word present in the tweet y then we consider them as pair. In the same way we try to find the possible pair for all tweets in the dataset based on different features. After finding the all pairs of the tweets of the dataset we apply K-fold-cross validation. Where we use k=5 that means we split the dataset into 5 portion where one portion use as test data and rest of the four part use as training data. For each tweet in the test data we calculate the Euclidian distance with their pair. For calculating Euclidian distance we consider the all features whose are describe above. If p = (p_1, p_2,..., p_n) where p represent a tweet from the test dataset and p1,p2,……..,p_n which are represented the weights of different features for the particular tweet and q = (q_1, q_2,..., q_n) where q represent a tweet from the pair set of that particular tweet p and q1,q2…….qn same as the p1,p2……..pn. if we consider those two points p and q in Euclidean *n*-space, then the distance (d) from p to q, or from q to p is given by the Pythagorean formula:

$$d(p,q) = \sqrt{(p1-q1)^2 + (p2-q2)^2 + \cdots + (pn-qn)^2}$$

After calculating the distance from p to all other tweets of its pair set we need to measure the 8 nearest neighbor based on distance. For finding the nearest neighbor we simply use bubble sort based on distance among them. When we finding our nearest neighbors we consider their classification label. Those neighbors sentiment label either positive or negative. For eight neighbors we calculate the majority that means if the number of positive sentiment holders tweets is greater than or equal to negative sentiment holders tweets than we assign sentiment of that tweet as positive otherwise negative. For classifying all the tweets in the test dataset we follow the same procedure.

B. Support vector machine

On the dataset with feature weight we apply PCA (principle component analysis). We ignoring the class labels for the PCA analysis because we don't need class labels for the PCA analysis[16]. In the first step of the PCA analysis we compute the d-dimensional mean vector. In the second step of PCA analysis there are two option are possible one of the option is to calculate scatter matrix and the one is covariance matrix. We like to calculate covariance matrix. The equations which is used to calculate the covariance and the scatter matrix are very similar and their Eigen spaces will be identical that means the eigenvectors found by calculation in both cases are same but they scaled differently by a constant factor. In this step we computing eigenvectors for the corresponding Eigen values and sorting them by decreasing the Eigen values. And finally we choose k eigenvectors with the largest Eigen values. By the above way we get a transformed dataset. We used grid search algorithm for find the best c and gamma for the transformed dataset which will be used in the SVM algorithm. We measure the best c and gamma from the range of $\{2^{-5},2^{-4},\ldots\ldots\ldots,2^{15}\}$ and $\{2^{-15},2^{-14},\ldots..,2^{3}\}$ respectively. Computing c and gamma we use the jar file [3]. After finding the best c and gamma for particular dataset we use SVM for classifying our test data based on the training data. For assign the sentiment label of the tweets of the test dataset we use the jar file [15][16].

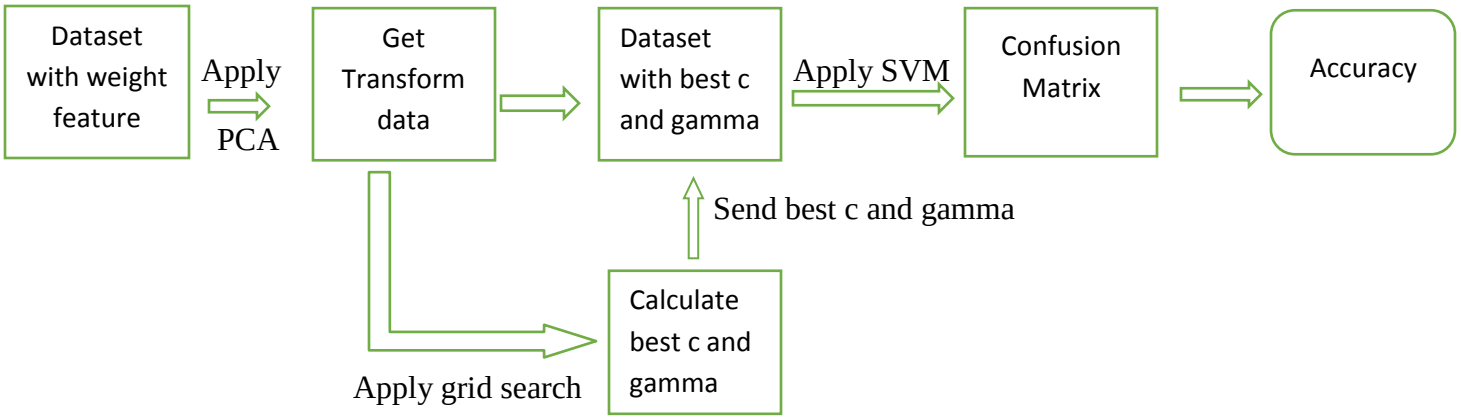

Figure 4.4: working flow of SVM

The technique is almost same for the all the versions of the SVM. In the first version of SVM we only use the 4 features and there is no grid search. We use the c and gamma value are 1 and 0.5 respectively. In the second version we follow the same procedure but before apply the SVM we normalize the dataset and the c and gamma values are same as before. We added another feature name is keyword base feature and of course its dataset is normalized. For the final version of SVM we apply the grid search along with the all previous things.

In both techniques we calculate the accuracy by using confusing matrix. Besides calculating the accuracy we also calculate the precession, recall and f-score. We apply the same process for different size of the dataset.

Chapter 5

5.1 Experimental Result:

In this chapter the result of Sentiment Classification Algorithm and Support Vector Machine are presented and later on we try to evaluate the performance of different versions of the Sentiment Classification Algorithm and Support Vector Machine based on Accuracy, Recall and Precision etc.

Algorithm	Number of tweets	Precision	Recall	FScore	TPR	FPR	Accuracy (100%)
Algorithm[9]	100	0.79	0.72	0.75	0.68	0.19	76.0
	500	0.75	0.60	0.67	0.72	0.20	71.31
	1000	0.81	0.76	0.78	0.79	0.13	79.99
	1500	0.79	0.75	0.77	0.81	0.16	78.19
	2000	0.69	0.85	0.76	0.88	0.33	74.56
	2500	0.68	0.85	0.76	0.87	0.34	73.58
	3000	0.69	0.87	0.77	0.88	0.36	74.56

Table 5.1: Experimental result of Algorithm[9]

Table 5.1 shows the experiment result based on the algorithms used in[9]. Based on 4 features we calculate those result. We use the same procedure for the different size of data.

Algorithm's	Number of tweets	Precision	Recall	FScore	TPR	FPR	Accuracy (100%)
KNN with normalization (4 features)	100	0.88	0.84	0.84	0.76	0.19	84.0
	500	0.74	0.63	0.67	0.73	0.20	71.11
	1000	0.83	0.75	0.79	0.81	0.14	80.80
	1500	0.78	0.78	0.78	0.82	0.17	78.80
	2000	0.71	0.88	0.78	0.89	0.32	76.45
	2500	0.70	0.88	0.78	0.89	0.33	75.45
	3000	0.69	0.88	0.77	0.89	0.33	74.79

Table 5.2: Experimental result of KNN with normalization (4 features)

Table 5.2 represent the result of algorithm KNN with normalization (4 features). In this algorithm we use the same features whose are used in table 5.1. But before use those 4 features we normalize the weights which we get from the four features.

Algorithm's	Number of tweets	Precision	Recall	FScore	TPR	FPR	Accuracy (100%)
KNN with normalization and keyword base(5 features)	100	0.904	0.77	0.77	0.66	0.21	78.0
	500	0.78	0.75	0.77	0.68	0.17	77.0
	1000	0.85	0.81	0.83	0.68	0.17	84.32
	1500	0.82	0.82	0.82	0.72	0.18	82.6
	2000	0.73	0.89	0.80	0.88	0.43	78.89
	2500	0.72	0.89	0.80	0.89	0.44	78.20
	3000	0.72	0.89	0.80	0.89	0.44	77.97

Table 5.3: Experimental result of KNN with normalization and keyword base (5 features)

Table 5.3 shows the result found by using KNN with normalization and keyword base (5 features). In this algorithm we use 5 features. The new feature is keyword base feature which is work based on a list which contain a word list with its sentiment strength.

Algorithm's	Number of tweets	Precision	Recall	FScore	TPR	FPR	Accuracy (100%)
SVM with (4 features)	100	0.60	0.70	0.63	0.72	0.33	61.0
	500	0.53	0.63	0.57	0.82	0.51	53.93
	1000	0.56	0.69	0.61	0.78	0.49	58.79
	1500	0.58	0.61	0.59	0.84	0.52	59.6
	2000	0.61	0.54	0.54	0.77	0.45	58.35
	2500	0.58	0.65	0.61	0.78	0.46	59.40
	3000	0.60	0.48	0.51	0.79	0.46	58.24

Table 5.4: Experimental result of SVM with (4 features)

Table 5.4 shows the experiment result based on the algorithm Based on SVM with (4 features).4 features we calculate the those result. So every data point present in 4 dimensional space. Where the each feature represent one dimension. We use the same procedure for the different size of data.

Algorithm's	Number of tweets	Precision	Recall	FScore	TPR	FPR	Accuracy (100%)
SVM with normalization (4 features)	100	0.62	0.65	0.63	0.72	0.39	63.0
	500	0.62	0.57	0.55	0.77	0.47	60.0
	1000	0.55	0.73	0.62	0.80	0.52	58.39
	1500	0.57	0.63	0.59	0.84	0.51	59.6
	2000	0.58	0.69	0.62	0.76	0.44	59.80
	2500	0.61	0.54	0.51	0.78	0.47	57.17
	3000	0.58	0.67	0.62	0.79	0.46	59.23

Table 5.5: Experimental result of SVM with normalization (4 features)

Table 5.5 shows the experimental result based on the algorithm Based on SVM with normalization (4 features). So every data point present in 4 dimensional space. Before applying SVM we normalize the weights which we get from the features.

Algorithm's	Number of tweets	Precision	Recall	FScore	TPR	FPR	Accuracy (100%)
SVM with normalization and keyword base (5 features)	100	0.69	0.78	0.73	0.72	0.33	71.0
	500	0.62	0.79	0.69	0.75	0.41	67.27
	1000	0.62	0.79	0.70	0.79	0.50	67.03
	1500	0.63	0.82	0.71	0.82	0.50	67.66
	2000	0.63	0.79	0.70	0.74	0.43	67.33
	2500	0.62	0.80	0.70	0.80	0.49	66.57
	3000	0.62	0.80	0.70	0.78	0.47	66.23

Table 5.6: Experimental result of SVM with normalization and keyword base (5 features)

Table 5.6 shows the experimental result based on the algorithm Based on SVM with normalization and keyword base (5 features).So every data point present in 5 dimensional space and of course the weights are normalized.

Algorithm's	Number of tweets	Precision	Recall	FScore	TPR	FPR	Accuracy (100%)
SVM with normalization and keyword base (5 features) with grid search	100	0.71	0.80	0.75	0.74	0.33	73.0
	500	0.64	0.81	0.71	0.73	0.40	69.80
	1000	0.72	0.89	0.80	0.70	0.30	77.97
	1500	0.65	0.81	0.72	0.75	0.36	69.88
	2000	0.64	0.78	0.68	0.67	0.32	71.23
	2500	0.63	0.82	0.73	0.74	0.39	66.57
	3000	0.64	0.77	0.73	0.74	0.42	69.58

Table 5.7: Experimental result of SVM with normalization and keyword base (5 features) with grid search

Table 5.6 shows the experimental result based on the algorithm Based on SVM with normalization and keyword base (5 features) with grid search. We use grid search before applying SVM.

From the table we can see that there are four attributes whose are precession, recall, F-Score and Accuracy. Precision (also called positive predictive value) is the fraction of retrieved instances that are relevant while recall (also known as sensitivity) is the fraction of relevant instances that are retrieved. Both precision and recall are therefore based on an understanding and measure of relevance. F-score is calculated from precision and recall.

5.2 Performance Evaluation

From the Experiment Result we can get an overview about the performance of the algorithms for different size of the dataset but for better analysis we should compare the results based on different parameter which is easily represented via graphs.

5.2.1 Performance Evaluation Based on Accuracy.

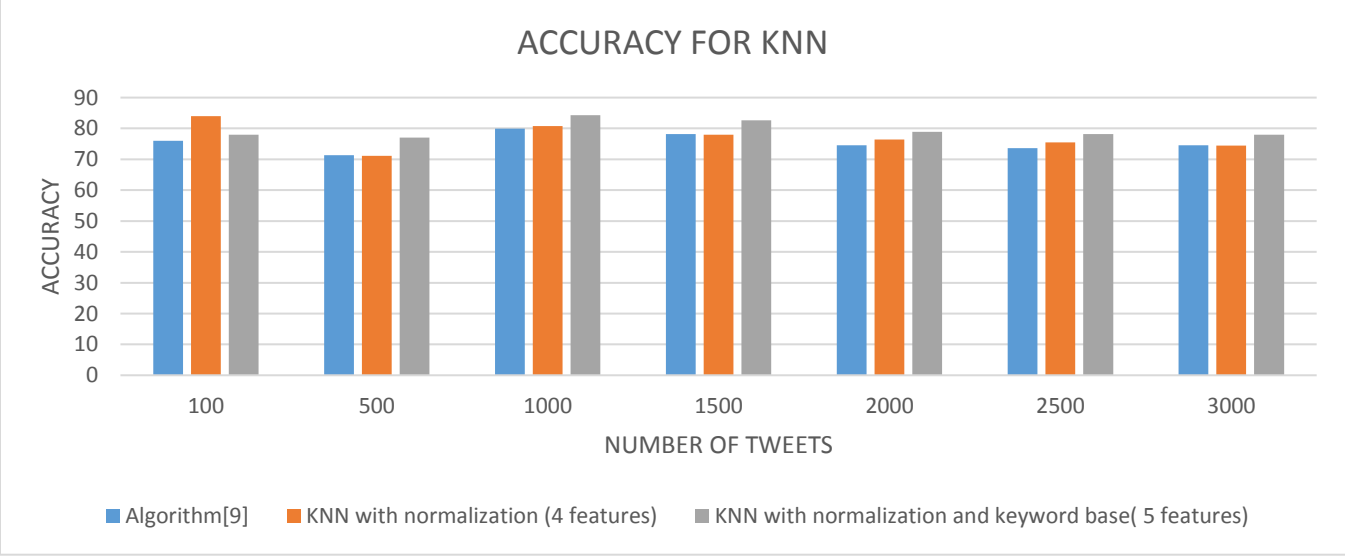

Figure 5.1: Accuracy of KNN

From the figure 6.1 we can see that this graph show the accuracy of different versions of KNN for different size of the dataset. From the graph we can see that when we use the original version of KNN with 4 features we get an accuracy which is increasing for all the dataset when we normalize the datasets. It is much more increasing when we add an additional feature name keyword based feature.

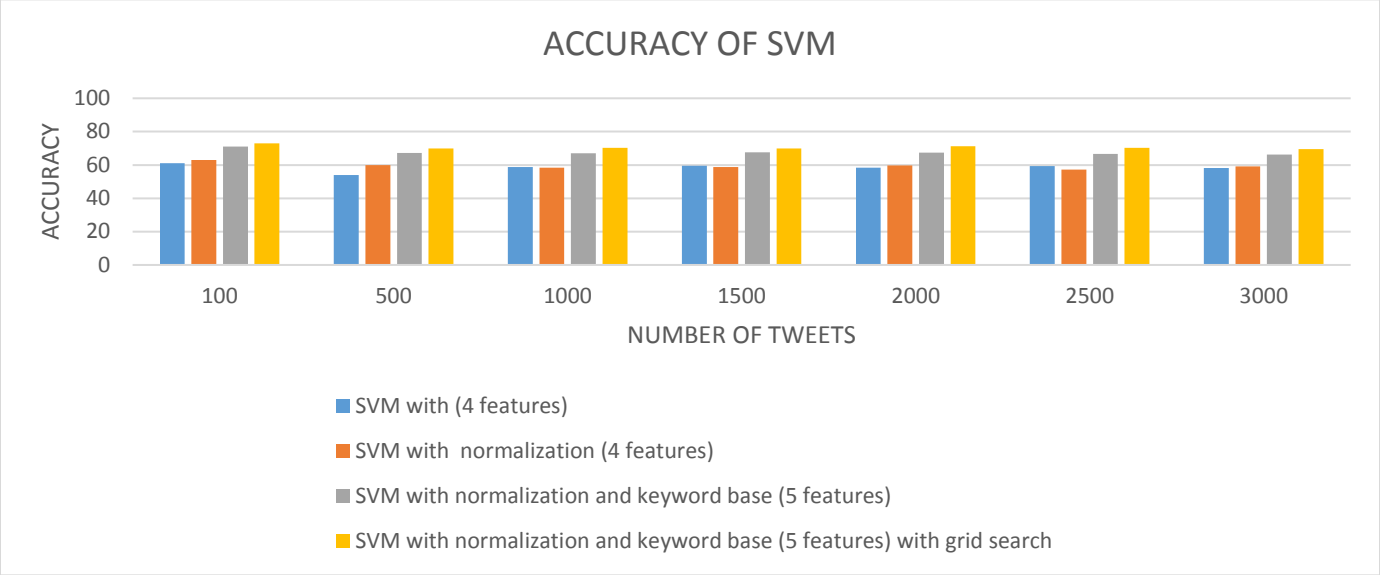

Figure 5.2: Accuracy of SVM

This graph is show the accuracy of the SVM. From the graph we see that the accuracy of SVM with 4 features is around 60 percent for different size of the dataset. This is increased by normalization of the dataset. When we add another feature keyword base feature its accuracy much more increasing and finally we get around 70 percent accuracy by using grid search for every dataset.

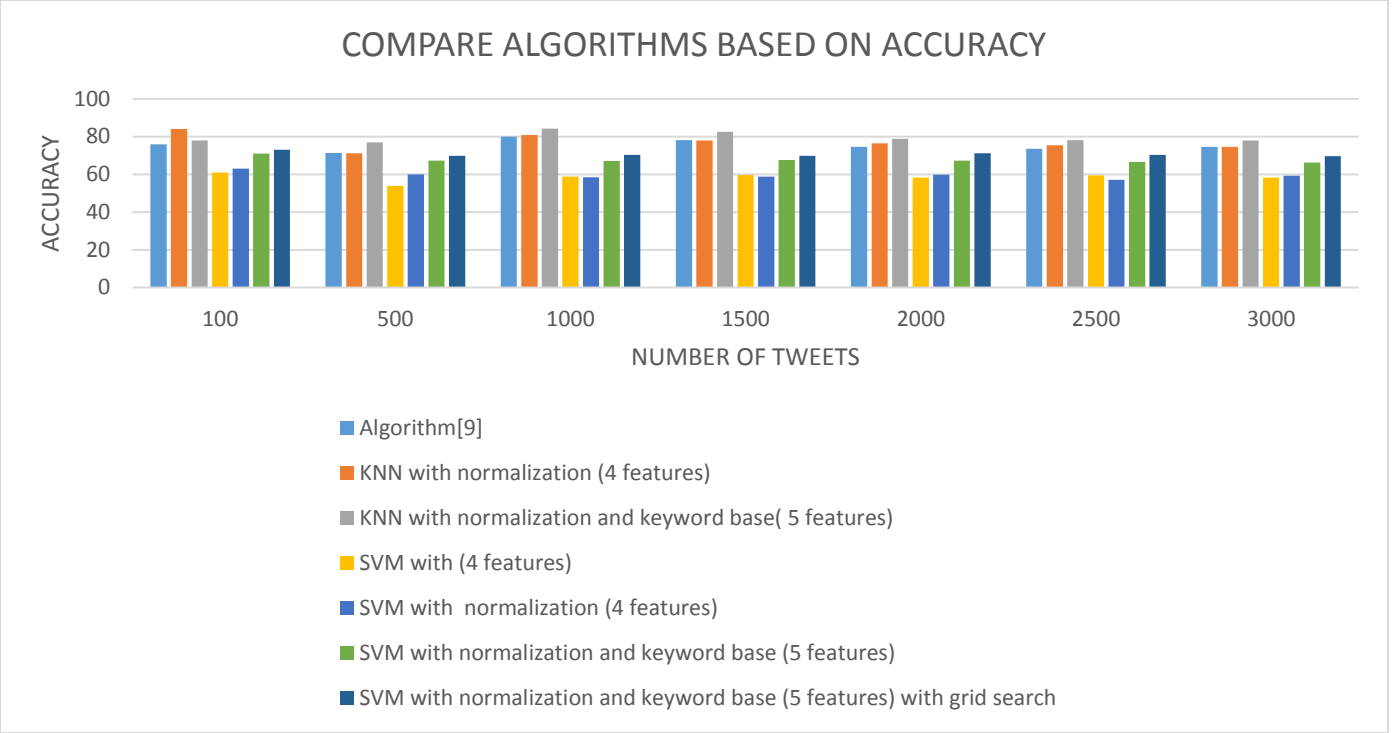

Figure 5.3: Comparing Algorithms based on accuracy

From figure 6.3 we see that the accuracy of different versions of KNN is always higher than the accuracy of the different versions of SVM .From the point of view of accuracy we see that KNN perform better than SVM for all the dataset.

if we only consider the accuracy only then it may be misleading us. Sometime a model has greater predictive power on the problem with lower accuracy may be desirable to select.

If we consider a problem where there exist a large class imbalance and there may be a model which can predict the value of the majority and high classification accuracy achieved by it but this kind of model is not useful in the problem domain.

For this reason there need to consider some additional measures like as precession, recall for evaluate a classifier.

5.2.2 Performance Evaluation Based on Precision

When evaluating classifiers it also necessary to consider the precession because precession can be thought as a measure of classifier exactness. A low precession indicates a large number of false positive. Sometime it may occur that that a classifier has low accuracy but it give high precession. In a problem where exactness is more important than the high accuracy than the precession evaluating is very important.

Precision is calculated from the number of true positives divided by the number of true positives and false positives.

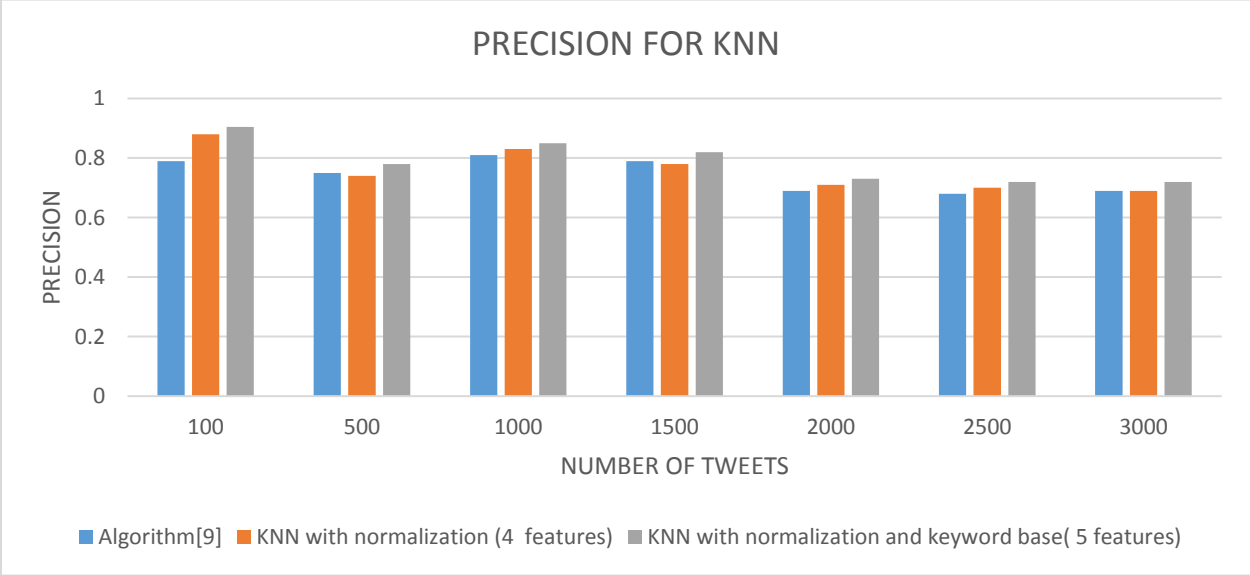

Figure 5.4: Precision of KNN

From the figure 6.4 we see that the precision of the original KNN (4 features) is low than the updated version of KNN which is also contains 4 features but the dataset is normalized. The

precision is also much more increases by adding another feature keyword base feature with the previous version of KNN.

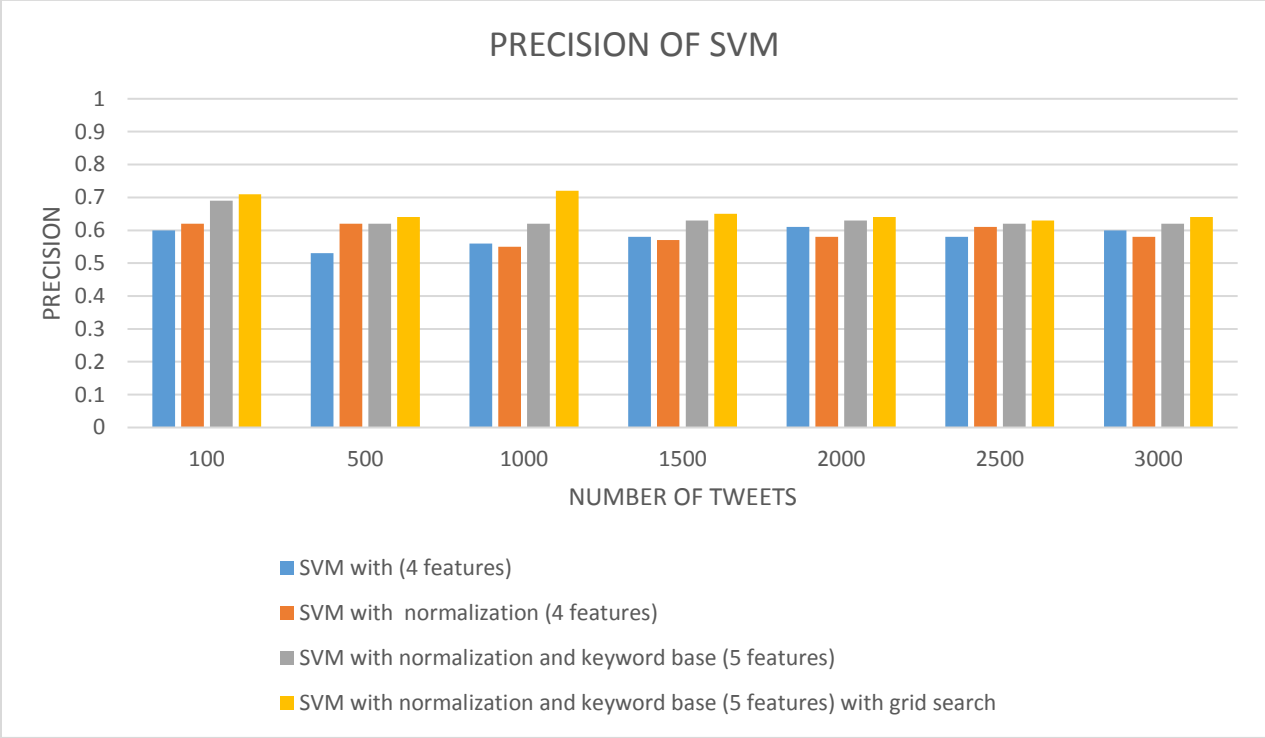

Figure 5.5: Precision of SVM

Figure 6.5 contains the information about the precision of different version of SVM. When we measure the precision of SVM with 4 features we get precision which can be increase by normalize it. It goes much higher when we adding another feature and using grid search.

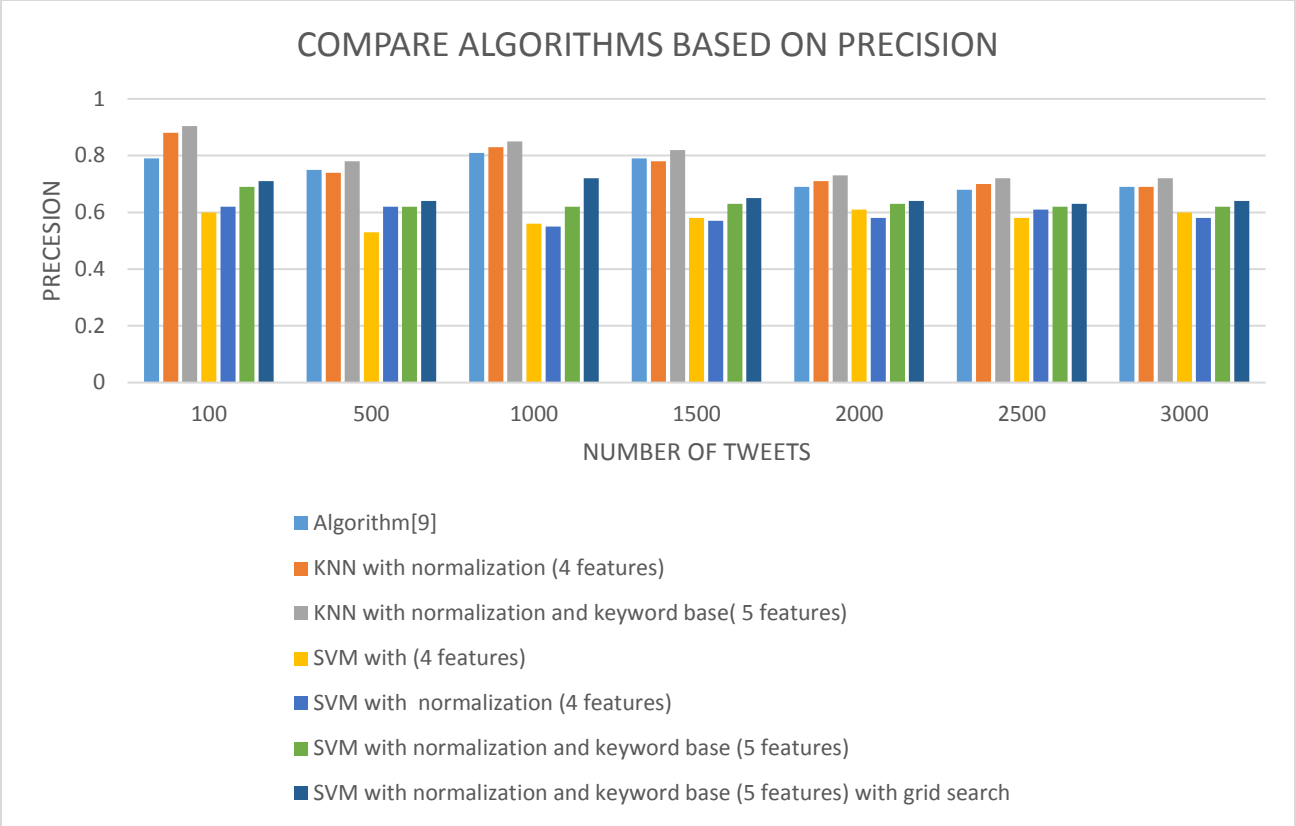

Figure 5.6: Comparing Algorithms based on Precision

As precision of different versions of KNN is higher than the different versions of SVM we consider this that based on precision KNN perform better than SVM.

5.2.3 Performance Evaluation Based on Recall

Recall is also important in classifier performance evaluation. Recall can be consider as a measure of a classifiers completeness and a low recall indicates many false negatives.

Recall is calculated from the number of true positives divided by the number of true positive and the number of false negatives.

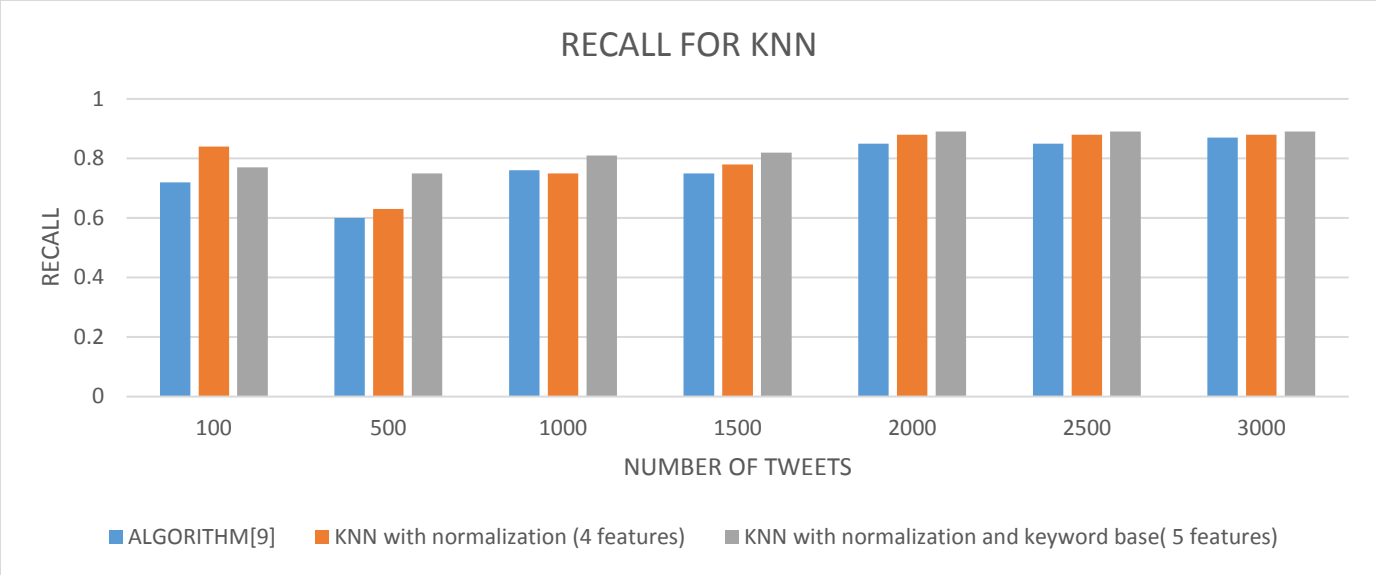

Figure 5.7: Recall for KNN

The different versions of KNN provide good recall for different size of the dataset. As we modify the version of KNN recall increasing than the previous version and it goes up to around 90 percentage which is very good.

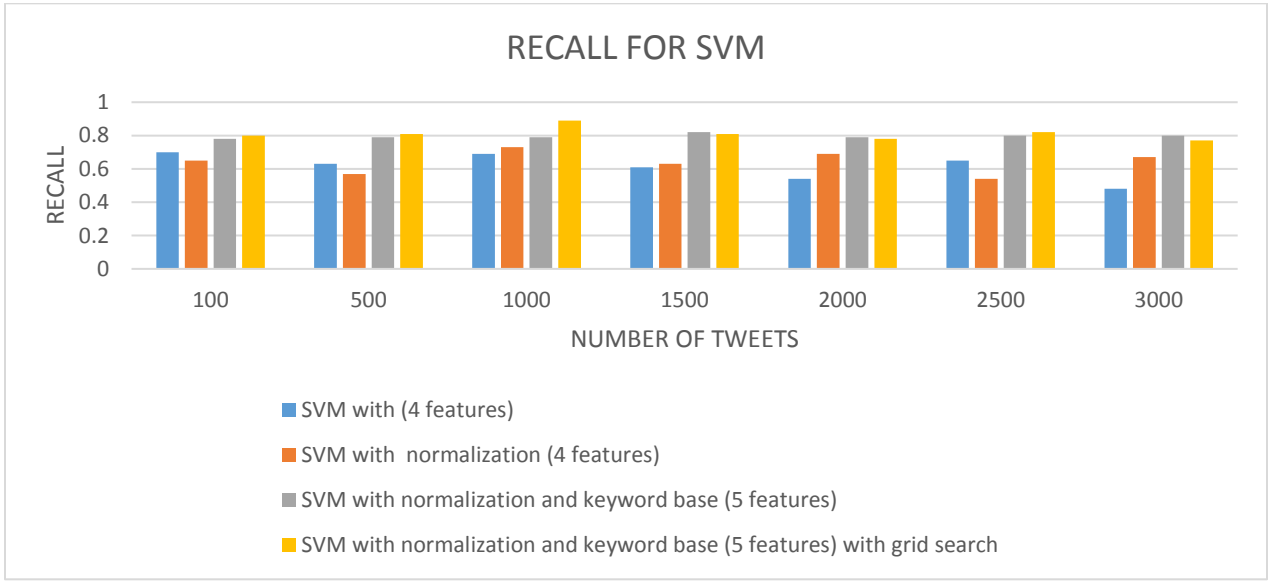

Figure 5.8: Recall for SVM

Recall performance is also good for the different versions of SVM. As version upgraded recall is also increases. SVM with 4 features without normalization and with normalization version recall is around 60 to 70 percent and it is goes to 80 up when we added another feature and using grid search.

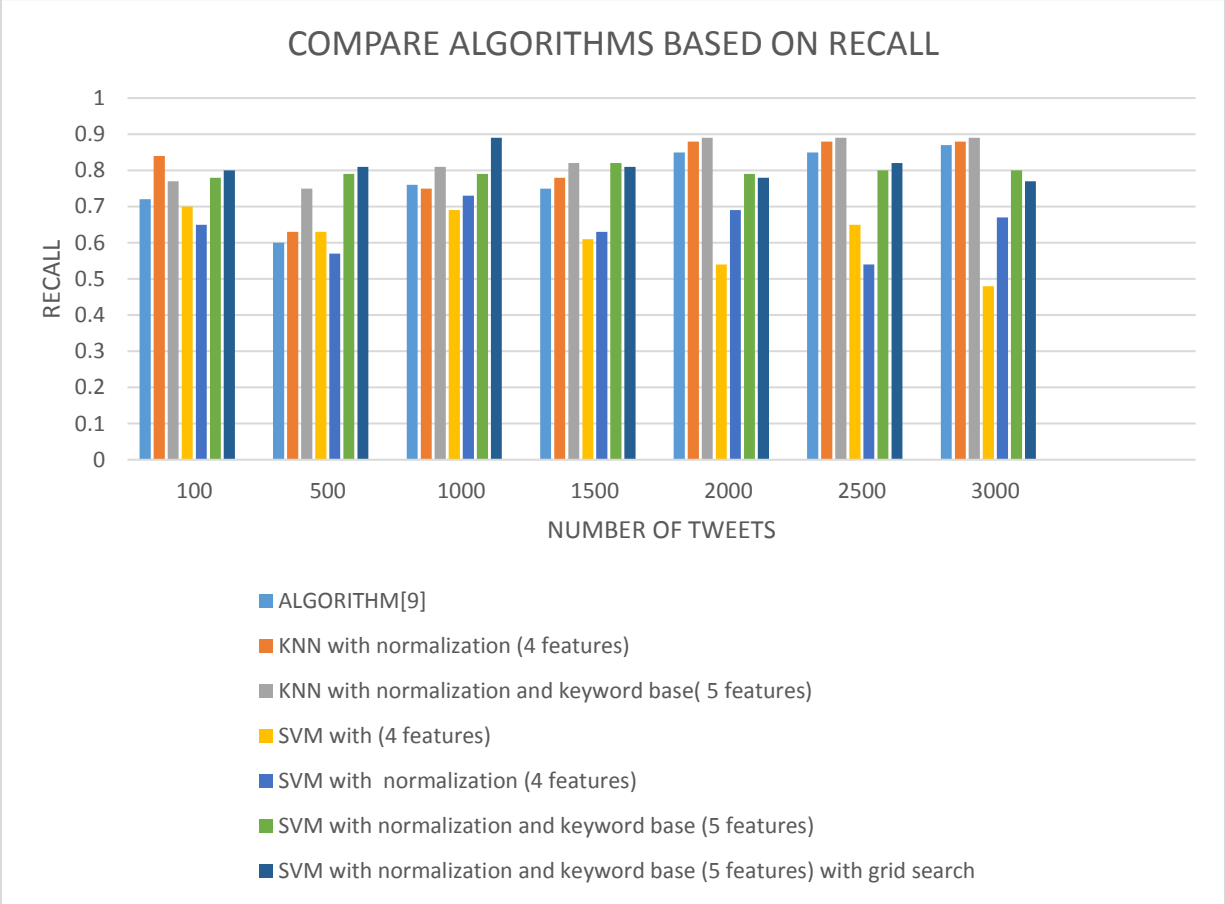

Figure 5.9: Comparing Algorithm based on Recall

Form the figure 6.9 which is indicate the comparison among different versions of KNN and SVM we see that the last two versions of SVM and KNN almost close to each other and the percentage is very good.

5.2.3 Roc Graph for Performance Evaluation

For selecting classifiers based on their performance receiver operating characteristics (ROC) graph is a good technique[17]. By plotting the true positive rate (TPR) against the false positive Rate (FPR) the curve is created. True positive rate indicated the sensitivity or probability of detection and false positive rate indicate the fall-out or probability of false alarm.

A : Original (KNN) (4 features)
B : KNN with normalization (4 features)
C : KNN with normalization and keyword base (5 features)
D : SVM with (4 features)
E : SVM with normalization (4 features)
F : SVM with normalization and keyword base (5 features)
G : SVM with normalization and keyword base (5 features) with grid search

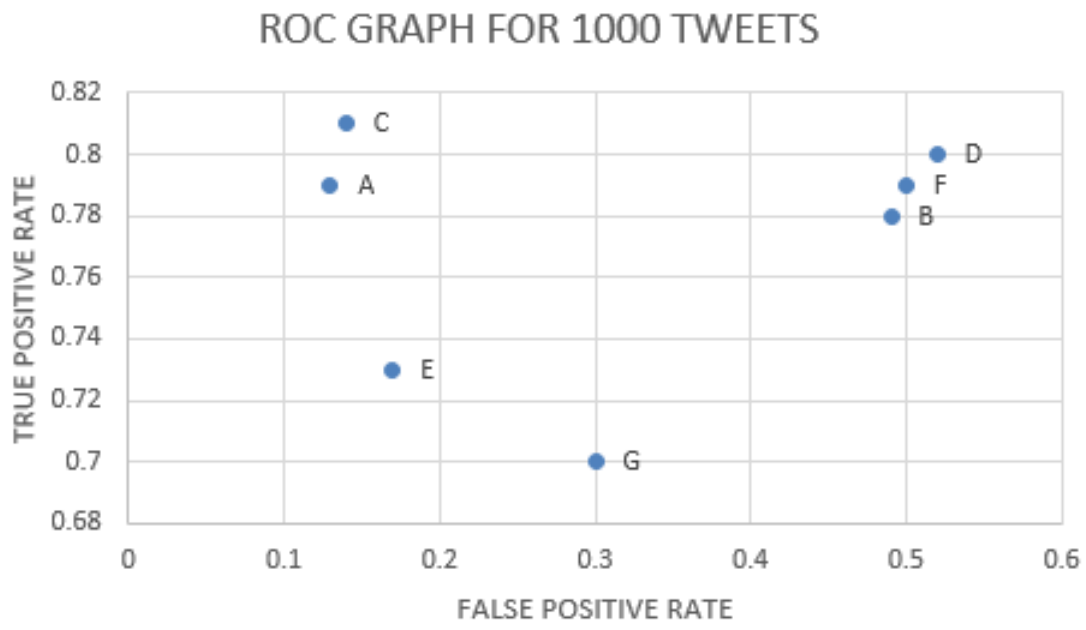

Figure 5.10: ROC graph for 1000 Tweets

Figure 6.11 shows the performance of the algorithms for 1000 tweets. We know that according to the ROC graph the performance of an algorithm is better than the other algorithms if its true positive rate

is high and false positive rate is low. Based on this concept if we plot the true positive rate in Y axis and false positive rate in X axis than we get a point in two dimensional for an algorithm. The more the algorithm is left and upper side the the performance of the algorithms is better than the other algorithms. We see from figure 6.11 A,C and E perform better compare with other algorithms. According to the ROC graph the algorithm is the best for which the data point is northwest corner compare to other algorithms that algorithm is best. But in our figure no data point at this can fulfill the requirement. But we can easily say that B,D and F perform worse than other algorithms. As G has less True positive rate than B,F and D it has also less false positive rate compare to them. Comparing between A and C algorithms A has less true positive rate and false positive rate than C but for the C True positive rate is much more increases than the false positive rate increase.

We can see from all the graphs draw based on different parameters all time KNN always perform better than SVM. There are some reasons behind this kind of performance. Those reasons are given below:-

SVM perform better when the number of dimension is very high. But in our experiment we only use 5 feature and every data point represent in 5 dimension. For a lots of points in few dimension SVM cannot perform better.

As the tweets are collect randomly and it is not guarantee that which dataset we use as training dataset the number of positive and negative tweets are equal. It is also may be produce highly imbalance dataset when we use k-fold-cross validation algorithm. Imbalance dataset means

the difference between the number of positive and negative tweets is very high in the training dataset.

Learning factor c and gamma vary based on dataset. Finding the best pair of c and gamma for a particular dataset is very tough. We try to find the best c and gamma by using grid search algorithm from some particular values of c and gamma. Grid search algorithm return the best pair of c and gamma but there may be exist better c and gamma.

SVM always assume a hyper plane exist between the classes. But sometime it may be very difficult to determine the hyper plane for the position of the data point in the dataset

Chapter 6

Conclusion and Future Work

Social networks have revolutionized the way in which people communicate. So we can it is beneficial to analysis user opinions and twitter is such a platform where people can share their opinions about topics, they can discuss about all current issues. Sentiment analysis on twitter data is a relatively new era, which deals with extracting user opinion automatically.

Sentiment analysis based on micro blogging is still in the developing stage and far from complete. As an example a positive sentiment is “It is a nice Day!” and a negative sentiment is “it is a horrible day!” .In this project we will try to find out positive and negative sentiment on twitter data.

Currently we have worked with a very simple model and in our work we design our classifier with only a very few features like n-gram feature, pattern feature, punctuation feature, keyword based feature and word feature. We also use machine learning algorithm SVM (support Vector Machine). We also use KNN classifier and calculate the accuracy in all the algorithm. In this project we only focus to divide the tweet into positive and negative sentiment. Apart from this here we don’t work with a large amount of tweet. In our project we will see that sentiment classifier algorithm works better than SVM.

In future we would study further many related problems. For this we will try to improve our models by adding some extra features. In this project we work with only the English tweets

and we are not consider the emoticons tweet. So our next plan is to work with other language tweets and add the emoticons tweets. Apart from this, we will also try to detect other sentiment label of human being and at the same time we will work with a big amount of tweets. From this we can say that our future work list may be contains the following things.

1. Add some extra features: Add some features which will helps us to detect sentiment more correctly and provide better result than the present result.
2. Work with others language: in our work we use java language and java jar files . in future we can use different language.
3. Work with emoticons tweets : we only work with text tweets. In future we need to work with text tweets as well as emoticons tweets.
4. Focusing to detect other sentiment label of human being: we only work with positive and negative sentiment label. Other sentiment label is absence in our work. We should also handle this part.
5. Working with large amount of tweets: In future work with dataset which contains large amount of tweets.
6. Accuracy calculate and performance evaluation: in our work we use confusion matrix for calculate accuracy and performance evaluation. In future Apply others machine learning algorithms to calculate accuracy and performance evaluation.
7. Work with real world problems: Given an efficient sentiment label, we will try to see how it can be applied for solving real world problems. (For example predicting presidential election, estimating product reputation etc.)

In this project we are mainly focusing on general sentiment analysis like positive and negative sentiment. There is potential of work in the field of sentiment analysis and we will try to use our knowledge in this field. On the other hand, we would like to compare sentiment analysis with other domains.

.

References

[1] AnalyticsVidhaya: https://www.analyticsvidhya.com/blog/2015/10/understaing-support-vector-machine-example-code/ (accessed on 12/11/16)

[2] Sebastianraschka:http://sebastianraschka.com/Articles/2014_pca_step_by_step.html (accessed on 10/11/16)

[3] Cs.uregenia:http://www2.cs.uregina.ca/~dbd/cs831/notes/confusion_matrix.html (accessed on 10/11/16)

[4] StackOverflow: http://stackoverflow.com/questions/19335165/cross-validation-and-grid-Search (accessed on 10/11/16)

[5] Apoorv Agarwal , Boyi Xie ,Ilia Vovsha, Owen Rambow, Rebecca Passonneau : Sentiment Analysis of Twitter Data. In: Processing LSM '11 Proceedings of the workshop on Language in Social Media, pages 30-38, 2011

[6] Luciano Barbosa, Junlan Feng:Robust Sentiment Detection on Twitter from Biased and Noisy Data. In 23rd International Conference on Computational Linguistics,2010

[7] Efthymios Koulompis, Theresa Wilson, Johanna Moore :Twitter Sentiment Analysis: The Good the Bad and the OMG!.In:Fifth International AAAI Conference on Weblogs and Social Media, 2011

[8] Hassan Saif, Yulan He and Harith Alani:Semantic Sentiment Analysis of Twitter. In: 11th International Semantic Web Conference,2012

[9] Nikolaos Nodarakis,Athanasios Tsakalidis,Spyros Siouts,Giannnis Tzimas: Large Scale Sentiment Analysis on Twitter with Spark. In:1st International Workshop on MultiEngine Data AnaLytics,2016

[10] Archive.org: https://archive.org/details/twitter_cikm_2010 (accessed on 07/07/2016)

[11] Sentiment140: http://help.sentiment140.com/api (accessed on 10/07/2016)

[12] Github: https://github.com/aababu/sentiment-analysis/blob/master/stopword.txt (accessed on (29/11/16)

[13] Github: https://github.com/aababu/sentiment-analysis/blob/master/wordstrength.txt (accessed on 29/11/16)

[14] Sourceforge: https://sourceforge.net/projects/java-ml/files/java-ml/(accessed on 10/08/16)

[15] Java2s: http://www.java2s.com/Code/Jar/l/Downloadlibsvm317sourcesjar.htm (accessed on 11/08/16)

[16] Java2s:http://www.java2s.com/Code/Jar/j/Downloadjama103jar.htm (accessed on 12/08/16)

[17] Tom Fawcett: ROC Graphs: Notes and Practical Considerations for Data Mining Researchers. In: Journal Pattern Recognition Letters – Special issue :Roc analysis in pattern recognition archive, 2006