TWITTER SENTIMENT ANALYSIS

Project report in partial fufillment of the requirement for the award of the degree of

Bachelor of Technology

In

COMPUTER SCIENCE & ENGINEERING

Submitted By

AKASH KUMAR JAI
University Roll No. 12016009001336
Department of Computer Science & Engineering
UEM, Kolkata

AJAY CHAKRABORTY
University Roll No. 12016009001134
Department of Computer Science & Engineering
UEM, Kolkata

PRAKSHA JAIN
University Roll No. 12016009001122
Department of Computer Science & Engineering
UEM, Kolkata

RITIKA SAHA
University Roll No. 12016009001314
Department of Computer Science & Engineering
UEM, Kolkata

RAJNEEK SINGH TOOR
University Roll No. 12016009001133
Department of Computer Science & Engineering
UEM, Kolkata

SIDDHARTH SINGH
University Roll No. 12016009001120
Department of Computer Science & Engineering
UEM, Kolkata

Under the guidance of

PROF. POOJARINI MITRA

Department of Computer Science & Engineering

UEM, Kolkata

UNIVERSITY OF ENGINEERING & MANAGEMENT, KOLKATA

University Area, Plot No. III – B/5, New Town, Action Area – III, Kolkata – 700 156

CERTIFICATE

This is to certify that the project titled TWITTER SENTIMENT ANALYSIS submitted by **Akash Kumar Jai (University Roll No. 12016009001336), Ajay Chakraborty (University Roll No. 12016009001134)**, **Praksha Jain (University Roll No. 12016009001122), Ritika Saha (University Roll No. 12016009001314)**, **Rajneek Singh Toor (University Roll No. 12016009001133)**, and **Siddharth Singh (University Roll No. 12016009001120).** Students of UNIVERSITY OF ENGINEERING & MANAGEMENT, KOLKATA, in fulfilment of requirement for the degree of Bachelor of Computer Science & Engineering is a bona fide work carried out by them under the supervision and guidance of **Prof. Poojarini Mitra** during 8th Semester of academic session of 2019-2020. The content of this report has not been submitted to any other university or institute for the award of any other degree.

I am glad to inform that the work is entirely original and its performance is found to be quite satisfactory.

Prof. Poojarini Mitra	Prof. Sukalyan Goswami
Assistant Professor	Head of the Department
Department of CSE	Department of CSE
UEM, Kolkata	UEM, Kolkata

ACKNOWLEDGEMENT

We would like to take this opportunity to thank everyone whose cooperation and encouragement throughout the ongoing course of this project remains invaluable to us.

We are sincerely grateful to our guide **Prof. Poojarini Mitra** of the Department of Computer Science & Engineering, UEM, Kolkata, for her wisdom, guidance and inspiration that helped us to go through with this project and take it to where it stands now.

We would also like to express our sincere gratitude to **Prof. Sukalyan Goswami**, HOD, Department of Computer Science & Engineering, UEM, Kolkata and all other departmental faculties for their ever-present assistant and encouragement.

Last but not the least, we would like to extend our warm regards to our families and peers who have kept supporting us and always had faith in our work.

AKASH KUMAR JAI

AJAY CHAKRABORTY

PRAKSHA JAIN

RITIKA SAHA

RAJNEEK SINGH TOOR

SIDDHARTH SINGH

TABLE OF CONTENTS

ABSTRACT

Sentiment analysis, also refers as opinion mining, is a sub machine learning task where we want to determine which is the general sentiment of a given document. Using mach ine learning techniques and natural language processing we can extract the subjective information of a document and try to classify it according to its polarity such as positi ve, neutral or negative. It is a really useful analysis since we could possibly determine the overall opinion about a selling objects, or predict stock markets for a given compa ny like, if most people think positive about it, possibly its stock markets will increase, and so on. Sentiment analysis is actually far from to be solved since the language is ve ry complex (objectivity/subjectivity, negation, vocabulary, grammar,...) but it is also why it is very interesting to working on.

In this project we have chosen to classify tweets from Twitter into "positive" or "negative" sentiment by building a model based on probabilities. Twitter is a microblogging website where people can share their feelings quickly and spontane ously by sending a tweets limitted by 140 characters. You can directly address a tweet to someone by adding the target sign "@" or participate to a topic by adding an hasta g "#" to your tweet. Because of the usage of Twitter, it is a perfect source of data to det ermine the current overall opinion ab -out anything.

CHAPTER – 1 : Introduction

In the past few years, there has been a huge growth in the use of microblogging platforms such as Twitter. Spurred by that growth, companies and media organizations are increasingly seeking ways to mine Twitter for information about what people think and feel about their products and services. Companies such as Twitter (twitter.com), tweetfeel (www.tweetfeel.com), and Social Mention (www.socialmention.com) are just a few who advertise Twitter sentiment analysis as one of their services. While there has been a fair amount of research on how sentiments are expressed in genres such as online reviews and news articles, how sentiments are expressed given the informal language and message-length constraints of microblogging has been much less studied. Features such as automatic part-of-speech tags and resources such as sentiment lexicons have proved useful for sentiment analysis in other domains, but will they also prove useful for sentiment analysis in Twitter? In this paper, we begin to investigate this question. Another challenge of microblogging is the incredible breadth of topic that is covered. It is not an exaggeration to say that people tweet about anything and everything. Therefore, to be able to build systems to mine Twitter sentiment about any given topic, we need a method for quickly identifying data that can be used for training. In this paper, we explore one method for building such data: using Twitter hashtags (e.g., #bestfeeling, #epicfail, #news) to identify positive, negative, and neutral tweets to use for training threeway sentiment classifiers. The online medium has become a significant way for people to express their opinions and with social media, there is an abundance of opinion information available. Using sentiment analysis, the polarity of opinions can be found, such as positive, negative, or neutral by analyzing the text of the opinion. Sentiment analysis has been useful for companies to get their customer's opinions on their products predicting outcomes of elections , and getting opinions from movie reviews. The information gained from sentiment analysis is useful for companies making future decisions. Many traditional approaches in sentiment analysis uses the bag of words method. The bag of words technique does not consider language morphology, and it could incorrectly classify two phrases of having the same meaning because it could have the same bag of words . The relationship between the collection of words is considered instead of the relationship between individual words . When determining the overall sentiment, the sentiment of each word is determined and combined using a function . Bag of words also ignores word order, which leads to phrases with negation in them to be incorrectly classified. Other techniques discussed in sentiment analysis include Naive Bayes, Maximum Entropy, and Support Vector Machines. In the Literature Survey section, approaches used for sentiment analysis and text classification are summarized.

Sentiment analysis refers to the broad area of natural language processing which deals with the computational study of opinions, sentiments and emotions expressed in text. Sentiment Analysis (SA) or Opinion Mining (OM) aims at learning people's opinions, attitudes and emotions towards an entity. The entity can represent individuals, events or topics. An immense amount of research has been performed in the area of sentiment analysis. But most of them focused on classifying formal and larger pieces of text data like reviews. With the wide popularity of social networkingandmicroblogging websites and an immense amount of data available from these resources, research projects on sentiment analysis have witnessed a gradual domain shift. The past few years have witnessed a huge growth in the use of microblogging platforms. Popular microblogging websites like Twitter have evolved to become a source of varied information. This diversity in the information owes to such microblogs being elevated as platforms where people post real time messages about their opinions on a wide variety of topics, discuss current affairs and share their experience on products and services they use in daily life. Stimulated by the growth of microblogging platforms, organizations are exploring ways to mine Twitter for information about how people are responding to their products and services. A fair amount of research has been carried out on how sentiments are expressed in formal text patterns such as product or movie reviews and news articles, but how sentiments are expressed given the informal language and message-length constraints of microblogging has been less explored.

A careful investigation of tweets reveals that the 140 character length text restricts the vocabulary which imparts the sentiment. The hyperlinks often present in these tweets in turn restrict the vocabulary size. The varied domains discussed would surely impose hurdles for training. The frequency of misspellings and slang words in tweets (microblogs in general) is much higher than in other language resources which is another hurdle that needs to be overcome. On the other way around the tremendous volume of data available from microblogging websites on varied domains are incomparable with other data resources available. Microblogging language is characterized by expressive punctuations which convey a lot of sentiments. Bold lettered phrases, exclamations, question marks, quoted text etc. leave scope for sentiment extraction. The proposed work attempts a novel approach on twitter data by aggregating an adapted polarity lexicon which has learnt from product reviews of the domains under consideration, the tweet specific features and unigrams to build a classifier model using machine learning techniques.

Machine Learning

Once we have applied the different steps of the preprocessing part, we can now focus on the machine learning part. There are three major models used in sentiment analysis toclassify a sentence into positive or negative: SVM, Naive Bayes and Language Models (NGram).SVM is known to be the model giving the best results but in this project we focus onlyon probabilistic model that are Naive Bayes and Language Models that have been widely usedin this field. Let's first introduce the Naive Bayes model which is well-known for its simplicity and efficiency for text classification.

1.1 Naive Bayes

In machine learning, Naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive)independence assumptionsbetween the features. Naive Bayes classifiers are highly scalable, requiring a number ofparameters linear in the number of variables (features/predictors) in a learning problem.Maximumlikelihoodtraining can be done by evaluating a closedformexpression (mathematicalexpression that can be evaluated in a finite number of operations), which takes linear time.It is based on the application of the Baye's rule given by the following formula:

$$P(C = c|D = d) = \frac{P(D = d|C = c)P(C = c)}{P(D = d)}$$

Formula 1.1.1: Bayes' rule

where denotes the document and the category (label), and are *D C d c* instances of *D* and *C.* We can simplify this expression by,

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$

Formula 1.1.2: Bayes' rule simplified

In our case, a tweet d is represented by a vector of K attributes such as d = (w1, w2, ..., wk) .Computing P(d|c) is not trivial and that's why the Naive Bayes introduces the assumption thatall of the feature values wjare independent given the category label c . That is, for i =/ j , wiandwjare conditionally independent given the category label c . So the Baye's rule can be rewrittenas,

$$P(c|d) = P(c) \times \frac{\prod_{j=1}^{K} P(w_j|c)}{P(d)}$$

Formula 1.1.3: Baye's rule rewritten

Based on this equation, maximum a posterior (MAP) classifier can be constructing by seekingthe optimal category which maximizes the posterior P(c|d) :

$$c* = arg\max_{c \in C} P(c|d)$$

$$c* = arg\max_{c \in C} \left\{ P(c) \times \frac{\prod_{j=1}^{K} P(w_j|c)}{P(d)} \right\}$$

$$c* = arg\max_{c \in C} \left\{ P(c) \times \prod_{j=1}^{K} P(w_j|c) \right\}$$

Formula 1.1.4: Classifier maximizing the posterior probability P(c|d)

Note that $P(d)$ is removed since it is a constant for every category c .
There are several variants of Naive Bayes classifiers that are:

● The **MultivariateBernoulli Model** : Also called binomial model, useful if our feature
vectors are binary (e.g 0s and 1s). An application can be text classification with bag of words model where the 0s 1s are "word does not occur in the document" and "word occurs in the document" respectively.

● The **Multinomial Model** : Typically used for discrete counts. In text classification, weextend the Bernoulli model further by counting the number of times a word w_i appears over the number of words rather than saying 0 or 1 if word occurs or not.

● The **Gaussian Model** : We assume that features follow a normal distribution. Instead ofdiscrete counts, we have continuous features.For text classification, the most used considered as the best choice is the Multinomial Naïve Bayes.

The prior distribution $P(c)$ can be used to incorporate additional assumptions about the relative frequencies of classes. It is computed by:

$$P(c) = \frac{N_i}{N}$$

Formula 1.1.5: Prior distribution P(c)

where N is the total number of training tweets and Ni is the number of training tweets in class$c$.

The likelihood is usually computed P(wj|c) using the formula:

$$P(w_j|c) = \frac{1 + count(w_j, c)}{|V| + N_i}$$

Formula 1.1.6: Likelihood P(wj|c)

where count(wj, c) is the number of times that word wjoccurs within the training tweets of classc. This estimation uses the simplest smoothing method to solve the **zeroprobabilityproblem** , that arises when our model encounters a wordseen in the test set but not in the training set, **Laplace** or addonesince we use 1 as constant.We will see that Laplace smoothing method is not really effective compared to other smoothing
methods used in language models.

1.2 Baseline

In every machine learning task, it is always good to have what we called a baseline. It often a "quick and dirty" implementation of a basic model for doing the first classification andbased on its accuracy, try to improve it.We use the Multinomial Naive Bayes as learning algorithm with the Laplace smoothingrepresenting the classic way of doing text classification. Since we need to extract features fromour data set of tweets, we use the **bag of words model** to represent it.The bag of words model is a simplifying representation of a document where it isrepresented as a bag of its words without taking consideration of the grammar or word order. Intext classification, the count (number of time) of each word appears is a document is used as afeature for training the classifier.Firstly, we divide the data set into two parts, the training set and the test set. To do this,we first shuffle the data set to get rid of any order applied to the data, then we from the set ofpositive tweets and the set of negative tweets, we take 3/4 of tweets from each set and merge them together to make the training set. The rest is used to make the test set. Finally the size ofthe training set is 1183958 tweets and the test set is 394654 tweets. Notice that they arebalanced and follow the same distribution of the initial data set.Once the training set and the test set are created we actually need a third set of datacalled the **validation set** . It is really useful because it will be used to **validate our modelagainst unseen data and tune the possible parameters** of the learning algorithm to avoidunderfitting and overfitting for example. We need this validation set because our test set shouldbe used only to verify how well the model will **generalize** . If we use the test set rather than thevalidation set, our model could be **overly optimistic and twist the results**.

1.3 Language Models

Let's introduce language models to see if we can have better results than those for our baseline. Language models are models assigning **probabilities to sequence of words** . Initially, they are extensively used in speech recognition and spelling correction but it turns outthat they give good results in text classification.
The quality of a language model can be measured by the empirical perplexity (or entropy) using:

$$Perplexity = T\sqrt{\frac{1}{P(w_1,\ldots,w_T)}}$$
$$Entropy = log_2 Perplexity$$

Formula 1.3.1: Perplexity and Entropy to evaluate language models.

The goal is to **minimize the perplexity which is the same as maximizing probability**.

An **N-Grammodel** is a type of probabilistic language model for predicting the next itemin such a sequence in the form of (n 1)
order Markov Model. The Markov assumption is the
probability of a word depends only on the probability of a limited history (previous words).

$$P(w_i|w_1,\ldots,w_{i-1}) = P(w_i|w_{i-n+1},\ldots,w_{i-1})$$

Formula 1.3.2: General form of N-grams.

A straightforward maximum likelihood estimate of n-gram
probabilities from a corpus is given by
the observed frequency,

$$P(w_i|w_{i-n+1},\ldots,w_{i-1}) = \frac{count(w_{i-n+1},\ldots,w_i)}{count(w_{i-n+1},\ldots,w_{i-1})}$$

Formula 1.3.3: MLE of N-grams.

There are several kind of n-gramsbut the most common are the unigram, bigram and trigram.The **unigram model** make the assumption that every word is independent and so we computethe probability of a sequence using the following formula,

$$P(w_1, w_2, \ldots, w_n) = \prod_i P(w_i)$$

Formula 1.3.4: Unigram.

In the case of the **bigram model** we make the assumption that **a word is dependent of itsprevious word** ,

$$P(w_i|w_1, w_2, \ldots, w_{i-1}) \approx P(w_i|w_{i-1})$$

Formula 1.3.5: Bigram.

To estimate the n-gramprobabilities, we need to compute the **Maximum Likelihood Estimates** .For Unigram:

$$P(w_i) = \frac{C(w_i)}{N}$$

Formula 1.3.6: MLE for unigram.

For Bigram:

$$P(w_i, w_j) = \frac{count(w_i, w_j)}{N}$$

$$P(w_j|w_i) = \frac{P(w_i, w_j)}{P(w_i)} = \frac{count(w_i, w_j)}{\sum_w count(w_i, w)} = \frac{count(w_i, w_j)}{count(w_i)}$$

Formula 1.3.7: MLE for bigram.

Where is the number of words, means count, N C wi and wj are words.

There are two main practical issues:
● We compute everything in log space (log probabilities) to avoid underflow (multiplying so
many probabilities can lead to too small number) and because adding is faster than multiplying (p1 × p2 × p3) = (logp1 log) + p2 + logp3

● We use smoothing techniques such as Laplace, Witten-Bell Discounting, Good-TuringDiscounting to deal with unseen words in the training occurring in the test set.

An N-gramlanguage model can be applied to text classification like Naive Bayes model does. Atweet is categorized according to,

$$c* = arg\max_{c \in C} P(c|d)$$

Formula 1.3.8: Objective function of n-gram.

and using Baye's rule, this can be rewritten as,

$$c* = arg\max_{c \in C} \{P(c)P(d|c)\}$$

$$c* = arg\max_{c \in C} \left\{ P(c) \times \prod_{i=1}^{T} P(w_i|w_{i-n+1}, \ldots, w_{i-1}, c) \right\}$$

$$c* = arg\max_{c \in C} \left\{ P(c) \times \prod_{i=1}^{T} P_c(w_i|w_{i-n+1}, \ldots, w_{i-1}) \right\}$$

Formula 1.3.9: Objective function rewritten using baye's rule of n-gram.

P(d|c)is the likelihood of under category which can be computed by P(d|c) d c an n-gramlanguagemodel.

CHAPTER – 2: LITERATURE SURVEY

Sentiment analysis is a growing area of Natural Language Processing with research ranging from document level classification (Pang and Lee 2008) to learning the polarity of words and phrases (e.g., (Hatzivassiloglou and McKeown 1997; Esuli and Sebastiani 2006)). Given the character limitations on tweets, classifying the sentiment of Twitter messages is most similar to sentencelevel sentiment analysis (e.g., (Yu and Hatzivassiloglou 2003; Kim and Hovy 2004)); however, the informal and specialized language used in tweets, as well as the very nature of the microblogging domain make Twitter sentiment analysis a very different task. It's an open question how well the features and techniques used on more well-formed data will transfer to the microblogging domain. Just in the past year there have been a number of papers looking at Twitter sentiment and buzz (Jansen et al. 2009 ; Pak and Paroubek 2010; O'Connor et al. 2010; Tumasjan et al. 2010; Bifet and Frank 2010; Barbosa and Feng 2010 ; Davidov, Tsur, and Rappoport 2010). Other researchers have begun to explore the use of part-of-speech features but results remain mixed. Features common to microblogging (e.g., emoticons) are also common, but there has been little investigation into the usefulness of existing sentiment resources developed on non-microblogging data. Researchers have also begun to investigate various ways of automatically collecting training data. Several researchers rely on emoticons for defining their training data (Pak and Paroubek 2010; Bifet and Frank 2010). (Barbosa and Feng 2010) exploit existing Twitter sentiment sites for collecting training data. (Davidov, Tsur, and Rappoport 2010) also use hashtags for creating training data, but they limit their experiments to sentiment/non-sentiment classification, rather than 3-way polarity classification, as we do. We use WEKA and apply the following Machine Learning algorithms for this second classification to arrive at the best result:

- Naïve baise
- N-Gram

CHAPTER – 3 : PROBLEM STATEMENT & DISCUSSION

3.1 Problem Definition:

Sentiment analysis of in the domain of micro-blogging is a relatively new research topic so there is still a lot of room for further research in this area. Decent amount of related prior work has been done on sentiment analysis of user reviews , documents, web blogs/articles and general phrase level sentiment analysis . These differ from twitter mainly because of the limit of 140 characters per tweet which forces the user to express opinion compressed in very short text. The best results reached in sentiment classification use supervised learning techniques such as Naive Bayes, but the manual labelling required for the supervised approach is very expensive. Some work has been done on unsupervised and semi-supervised approaches, and there is a lot of room of improvement. Various researchers testing new features and classification techniques often just compare their results to base-line performance. There is a need of proper and formal comparisons between these results arrived through different features and classification techniques in order to select the best features and most efficient classification techniques for particular applications.

Sentiment analysis is a growing area of Natural Language Processing with research ranging from document level classification (Pang and Lee 2008) to learning the polarity of words and phrases (e.g., (Hatzivassiloglou and McKeown 1997; Esuli and Sebastiani 2006)). Given the character limitations on tweets, classifying the sentiment of Twitter messages is most similar to sentencelevel sentiment analysis (e.g., (Yu and Hatzivassiloglou 2003; Kim and Hovy 2004)); however, the informal and specialized language used in tweets, as well as the very nature of the microblogging domain make Twitter sentiment analysis a very different task. It's an open question how well the features and techniques used on more well-formed data will transfer to the microblogging domain. Just in the past year there have been a number of papers looking at Twitter sentiment and buzz (Jansen et al. 2009 ; Pak and Paroubek 2010; O'Connor et al. 2010; Tumasjan et al. 2010; Bifet and Frank 2010; Barbosa and Feng 2010 ; Davidov, Tsur, and Rappoport 2010). Other researchers have begun to explore the use of part-of-speech features but results remain mixed. Features common to microblogging (e.g., emoticons) are also common, but there has been little investigation into the usefulness of existing sentiment resources developed on non-microblogging data. Researchers have also begun to investigate various ways of automatically collecting training data. Several researchers rely on emoticons for

defining their training data (Pak and Paroubek 2010; Bifet and Frank 2010). (Barbosa and Feng 2010) exploit existing Twitter sentiment sites for collecting training data. (Davidov, Tsur, and Rappoport 2010) also use hashtags for creating training data, but they limit their experiments to sentiment/non-sentiment classification, rather than 3-way polarity classification, as we do.

In here we use Naïve Bayes and n-gramClassifiersto retrieve the Tweets from Twitter Data.

3.2 Discussion

1 Data

To gather the data many options are possible. In some previous paper researches, they built a program to collect automatically a corpus of tweets based on two classes, “posit ive” and “negative”, by querying Twitter with two type of emoticons:

- Happy emoticons, such as “:)”, “:P”, “:-)” etc.
- Sad emoticons, such as “:(“, “:’(”, “=(“.

Others make their own dataset of tweets my collecting and annotating them manually which very long and fastidious.

Additionally to find a way of getting a corpus of tweets, we need to take of having a balanced data set, meaning we should have an equal number of positive and negative t weets, but it needs also to be large enough. Indeed, more the data we have, more we c an train our classifier and more the accuracy will be.

After many researches, I found a dataset of 1578612 tweets in english coming from t wo sources: Kaggle and Sentiment140.

It is composed of four columns that are ItemID , Sentiment , SentimentSource and SentimentText. We are only interested by the Sentiment column corresponding to our label class taking a binary value, 0 if the tweet is negativ e, 1 if the tweet is positive and the SentimentText columns containing the tweets in a raw format.

	ItemID	Sentiment	SentimentSource	SentimentText
0	1	0	Sentiment140	is so sad for my APL friend.....
1	2	0	Sentiment140	I missed the New Moon trailer...
2	3	1	Sentiment140	omg its already 7:30 :O
3	4	0	Sentiment140	.. Omgaga. Im sooo im gunna CRy. I've been at this dentist since 11.. I was suposed 2 just get a crown put on (30mins)...
4	5	0	Sentiment140	i think mi bf is cheating on me!!! T_T
5	6	0	Sentiment140	or i just worry too much?
6	7	1	Sentiment140	Juuuuuuuuuuuuuuuussssst Chillin!!
7	8	0	Sentiment140	Sunny Again Work Tomorrow :-| TV Tonight
8	9	1	Sentiment140	handed in my uniform today . i miss you already
9	10	1	Sentiment140	hmmmm.... i wonder how she my number @-)

Table1 Example of twitter posts annotated with their corresponding sentiment, 0 if it is negati ve, 1 if it is positive.

Table 1 showing the first ten twitter posts we can already notice some particularities a nd difficulties that we are going to encounter during the preprocessing steps.

● The presence of acronyms
"bf" or more complicated "APL". Does it means apple ? Apple (the company) ? In this context we have "friend" after so we could think that he refers to his smartphone and so Apple, but what about if the word "friend" was not here ?

● The presence of sequences of repeated characters
such as "Juuuuuuuuuuuuuuuussssst", "hmmmm". In general when we repeat several characters in a word, it is to emphasize it, to increase its impact.

● The presence of emoticons
, ":O", "T_T", ":-|" and much more, give insights about user's moods.

● Spelling mistakes and " urban grammar " like "im gunna" or "mi".

● The presence of nouns such as "TV", "New Moon".

Furthermore, we can also add,

● People also indicate their moods, emotions, states, between two such as, *\ cries*, *hummin*, *sigh*.

●The negation, "can't", "cannot", "don't", "haven't" that we need to handle like: "I don 't like chocolate", "like" in this case is negative. We could also be interested by the

grammar structure of the tweets, or if a tweet is subjective/objective and so on. As you can see, it is extremely complex to deal with languages and even more when we want to analyse text typed by users on the Internet because people don't take care of making sentences that are grammatically correct and use a ton of acronyms and words that are more or less english in our case. We can visualize a bit more the dataset by making a chart of how many positive and negative tweets does it contains,

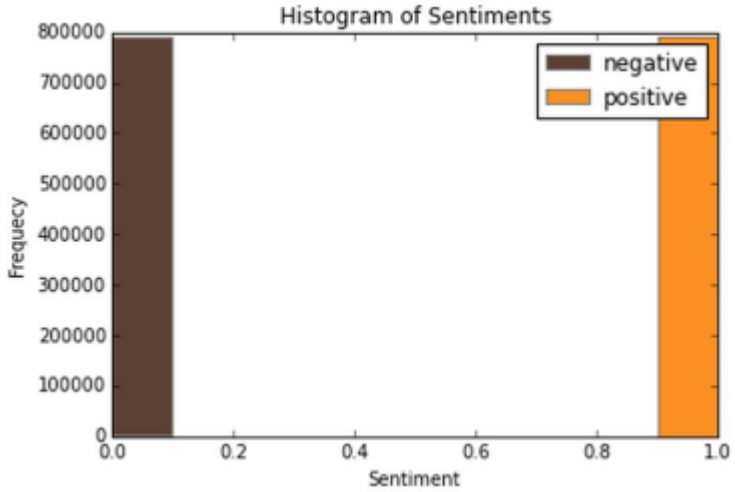

Figure 1.2: Histogram of the tweets according to their sentiment

We have exactly 790177 positive tweets and 788435 negative tweets which signify that the dataset is well-balanced. There is also no duplicates. Finally, let's recall the Twitter terminology since we are going to have to deal with in the tweets:

●Hashtag: A hashtag is any word or phrase immediately preceded by the # symbol. When you click on a hashtag, you'll see other Tweets containing the same keyword or topic.

●@username: A username is how you're identified on Twitter, and is always preceded immediately by the @ symbol. For instance, Katy Perry is @katyperry.

●MT: Similar to RT (Retweet), an abbreviation for "Modified Tweet." Placed before the Retweeted text when users manually retweet a message with modifications, for example shortening a Tweet.

●Retweet: RT, A Tweet that you forward to your followers is known as a Retweet. Often used to pass along news or other valuable discoveries on Twitter, Retweets always retain original attribution.

●Emoticons: Composed using punctuation and letters, they are used to express emotions concisely, ";) :) ...".

Now we have the corpus of tweets, we need to use other resources to make easier the pre-processing step.

3.3 SRS (Software Requirement Specification):

• Internal Interface requirement:

Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem. Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority. Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type. Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here. The recent explosion in data pertaining to users on social media has created a great interest in performing sentiment analysis on this data using Big Data and Machine Learning principles to understand people's interests. This project intends to perform the same tasks. The difference between this project and other sentimnt analysis tools is that, it will perform real time analysis of tweets based on hashtags and not on a stored archive. ;Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a ;follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this ;software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, ;subsystem interconnections, and external interfaces can be helpful. The Product functions are: • Collect tweets in a real time fashion i.e. , from the twitter live stream based on specified hashtags

• Remove redundant information from these collected tweets.

• Store the formatted tweets in MongoDB database

• Perform Sentiment Analysis on the tweets stored in the database to classify their nature viz. positive, negative and so on.

• Use a machine learning algorithm which will predict the ‘mood’ of the people with respect ot that topic. ;Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, ;so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to ;any reader of the SRS. A picture of themajor groups of related requirements and how they relate, such as a top level data ;flow diagram or object class diagram, is often effective.

Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy. Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.

• External Interface Requirement:

We classify External Interface in 4 types, those are:

User Interface:

Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.

Hardware interface:

Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.

Software Interface:

Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (forexample, use of a global data area in a multitasking operating system), specify this as an implementation constraint.

Communication Interface:

Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.

Other Requirements:

- Linux Operating System/Windows
- Python Platform(Anaconda2,Spyder,Jupyter)
- Modern Web Browser
- Twitter API, Google API

CHAPTER – 4: PROPOSED SOLUTION & RESULT ANALYSIS

4.1 Pre-processing

Now that we have the corpus of tweets and all the resources that could be useful, we can pre-process the tweets. It is a very important since all the modifications that we are going to during this process will directly impact the classifier's performance. The pre-processing includes cleaning, normalization, transformation, feature extraction and selection, etc. The result of pre-processing will be consistent and uniform data that are workable to maximize the classifier's performance.

All of the tweets are pre-processed by passing through the following steps in the same order.

4.1.1 Emoticons

We replace all emoticons by their sentiment polarity ||pos|| and ||neg|| using the emoticon dictionary. To do the replacement, we pass through each tweet and by using a regex we find out if it contains emoticons, if yes they are replaced by their corresponding polarity.

	ItemID	Sentiment	SentimentSource	SentimentText
0	1	0	Sentiment140	is so sad for my APL friend.....
1	2	0	Sentiment140	I missed the New Moon trailer...
2	3	1	Sentiment140	omg its already 7:30 :O
3	4	0	Sentiment140	.. Omgaga. Im sooo im gunna CRy. I've been at this dentist since 11.. I was suposed 2 just get a crown put on (30mins)...
4	5	0	Sentiment140	i think mi bf is cheating on me!!! T_T
5	6	0	Sentiment140	or i just worry too much?
6	7	1	Sentiment140	Juuuuuuuuuuuuuuuussssst Chillin!!
7	8	0	Sentiment140	Sunny Again Work Tomorrow :-| TV Tonight
8	9	1	Sentiment140	handed in my uniform today . i miss you already
9	10	1	Sentiment140	hmmmm.... i wonder how she my number @-)

Table 4.1.1.1Before processing emoticons, list of tweets where some of them contain emoticons.

	ItemID	Sentiment	SentimentSource	SentimentText
0	1	0	Sentiment140	is so sad for my APL friend.....
1	2	0	Sentiment140	I missed the New Moon trailer...
2	3	1	Sentiment140	omg its already 7:30 ||pos||
3	4	0	Sentiment140	.. Omgaga. Im sooo im gunna CRy. I've been at this dentist since 11.. I was suposed 2 just get a crown put on (30mins)...
4	5	0	Sentiment140	i think mi bf is cheating on me!!! ||neg||
5	6	0	Sentiment140	or i just worry too much?
6	7	1	Sentiment140	Juuuuuuuuuuuuuuuussssst Chillin!!
7	8	0	Sentiment140	Sunny Again Work Tomorrow ||neg|| TV Tonight
8	9	1	Sentiment140	handed in my uniform today . i miss you already
9	10	1	Sentiment140	hmmmm.... i wonder how she my number ||pos||

Table 4.1.1.1: After processing emoticons, they have been replaced by their corresponding tag

The data set contains 19469 positive emoticons and 11025 negative emoticons.

4.1.2 URLs

We replace all URLs with the tag "url". There is about 73824 urls in the data set and we proceed as the same way we did for the emoticons.

	ItemID	Sentiment	SentimentSource	SentimentText
50	51	0	Sentiment140	baddest day eveer.
51	52	1	Sentiment140	bathroom is clean..... now on to more enjoyable tasks.....
52	53	1	Sentiment140	boom boom pow
53	54	0	Sentiment140	but i'm proud.
54	55	0	Sentiment140	congrats to helio though
55	56	0	Sentiment140	David must be hospitalized for five days end of July (palatine tonsils). I will probably never see Katie in concert.
56	57	0	Sentiment140	friends are leaving me 'cause of this stupid love http://bit.ly/ZoxZC
57	58	1	Sentiment140	go give ur mom a hug right now. http://bit.ly/azFwv
58	59	1	Sentiment140	Going To See Harry Sunday Happiness
59	60	0	Sentiment140	Hand quilting it is then...

Table 4.1.2.1: Tweets before processing URLs.

	ItemID	Sentiment	SentimentSource	SentimentText
50	51	0	Sentiment140	baddest day eveer.
51	52	1	Sentiment140	bathroom is clean..... now on to more enjoyable tasks.....
52	53	1	Sentiment140	boom boom pow
53	54	0	Sentiment140	but i'm proud.
54	55	0	Sentiment140	congrats to helio though
55	56	0	Sentiment140	David must be hospitalized for five days end of July (palatine tonsils). I will probably never see Katie in concert.
56	57	0	Sentiment140	friends are leaving me 'cause of this stupid love ‖url‖
57	58	1	Sentiment140	go give ur mom a hug right now. ‖url‖
58	59	1	Sentiment140	Going To See Harry Sunday Happiness
59	60	0	Sentiment140	Hand quilting it is then...

Table 4.1.2.2: Tweets after processing URLs.

4.1.3 Targets:

The target correspond to usernames in twitter preceded by "@" symbol. It is used to a ddress a tweet to someone or just grab the attention. We replace all usernames/targets by thetag "**target**" . Notice that the data set we have 735757 targets.

	ItemID	Sentiment	SentimentSource	SentimentText
45	46	1	Sentiment140	@ginaaa <3 go to the show tonight
46	47	0	Sentiment140	@spiral_galaxy @ymptweet it really makes me sad when i look at muslims reality now
47	48	0	Sentiment140	- all time low shall be my motivation for the rest of the week.
48	49	0	Sentiment140	and the entertainment is over, someone complained properly.. @ruptureapture experimental you say? he should experiment with a me...
49	50	0	Sentiment140	another year of lakers .. that's neither magic nor fun ....
50	51	0	Sentiment140	baddest day eveer.
51	52	1	Sentiment140	bathroom is clean..... now on to more enjoyable tasks.....
52	53	1	Sentiment140	boom boom pow
53	54	0	Sentiment140	but i'm proud.
54	55	0	Sentiment140	congrats to helio though

Table 4.1.3.1: Tweets before processing targets.

	ItemID	Sentiment	SentimentSource	SentimentText
45	46	1	Sentiment140	||target|| <3 go to the show tonight
46	47	0	Sentiment140	||target|| ||target|| it really makes me sad when i look at muslims reality now
47	48	0	Sentiment140	- all time low shall be my motivation for the rest of the week.
48	49	0	Sentiment140	and the entertainment is over, someone complained properly.. ||target|| experimental you say? he should experiment with a melody...
49	50	0	Sentiment140	another year of lakers .. that's neither magic nor fun ....
50	51	0	Sentiment140	baddest day eveer.
51	52	1	Sentiment140	bathroom is clean..... now on to more enjoyable tasks.....
52	53	1	Sentiment140	boom boom pow
53	54	0	Sentiment140	but i'm proud.
54	55	0	Sentiment140	congrats to helio though

Table 4.1.3.2: Tweets after processing targets.

4.1.4 Baseline

In every machine learning task, it is always good to have what we called a baseline. It

To make the validation set, there are two main options:
● Split the training set into two parts (60%, 20%) with a ratio 2:8 where each part containsan equal distribution of example types. We train the classifier with the largest part, andmake prediction with the smaller one to validate the model. This technique works wellbut has the disadvantage of our classifier not getting trained and validated on allexamples in the data set (without counting the test set).

● The **K-fold cross-validation.** We split the data set into k parts, hold out one, combinethe others and train on them, then validate against the heldoutportion. We repeat thatprocess k times (each fold), holding out a different portion each time. Then we averagethe score measured for each fold to get a more accurate estimation of our model'sperformance.

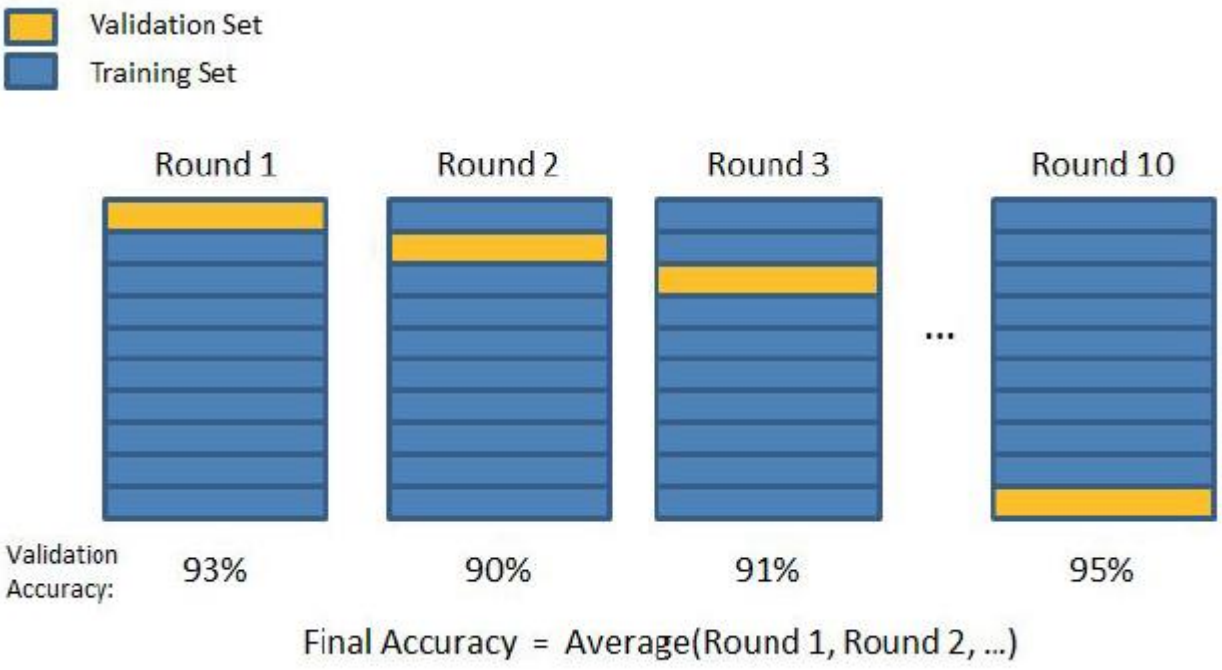

Figure 4.1.4.1: 10-fold cross-validation

We split the training data into 10 folds and cross validate on them using scikitlearn Asshown in the figure 2.4.1.1 above. The number of K-foldsis arbitrary and usually set to 10 it isnot a rule. In fact, determine the best K is still an unsolved problem but with lower K:computationally cheaper, less variance, more bias. With large K: computationally expensive,higher variance, lower bias.

We can now train the naive bayes classifier with the training set, validate it using the hold out part of data taken from the training set, the validation set, repeat this 10 times andaverage the results to get the final accuracy which is about **0.77** as shown in the screen resultsbelow,

```
Total tweets classified: 1183958
Score: 0.77653600187
Confusion matrix:
[[465021 126305]
 [136321 456311]]
```

Figure 4.1.4.2: Result of the naive bayes classifier with the score representing the average of the results of each 10-fold cross-validation, and the overall confusion matrix.

Notice that to evaluate our classifier we two methods, the F1 score and a confusion matrix. The **F1 Score** can be interpreted as a weighted average of the precision and recall,where an F1 score reaches its best value at 1 and worst score at 0. It a measure of a**classifier's accuracy** . The F1 score is given by the following formula,

$$F1 = \frac{2 \times (precision \times recall)}{(precision + recall)}$$

Formula 4.1.4.1: F1 score

where the precision is the number of true positives (the number of items correctly labeled asbelonging to the positive class) divided by the total number of elements labeled as belonging tothe positive class,

$$Precision = \frac{TP}{TP + FP}$$

Formula 4.1.4.2: Precision

and the recall is the number of true positives divided by the total number of elements thatactually belong to the positive class,

$$Recall = \frac{TP}{TP + FN}$$

Formula 4.1.4.3: Recall

A precision score of 1.0 means that every result retrieved was relevant (but says nothingabout whether all relevant elements were retrieved) whereas a recall score of 1.0 means thatall relevant documents were retrieved (but says nothing about how many irrelevant documentswere also retrieved).

There is a **tradeoff between precision and recall** where increasing one decrease the other and we usually use measures that combine precision and recall such as F-measureorMCC.A **confusion matrix** helps to visualize how the model did during the classification andevaluate its accuracy. In our case we get about 156715 false positive tweets and 139132 falsenegative tweets. It is "about" because these numbers can vary depending on how we shuffle ourdata for example.

		Predicted Class: Yes	Predicted Class: No
Actual Class	Yes	TP	FN
	No	FP	TN

Figure 4.1.4.3: Example of confusion matrix

Notice that we still didn't use our test set, since we are going to tune our classifier for improving its results.The confusion matrix of the naive bayes classifier can be expressed using a color mapwhere dark colors represent high values and light colors represent lower values as shown in thecorresponding color map of the naive bayes classifier below,

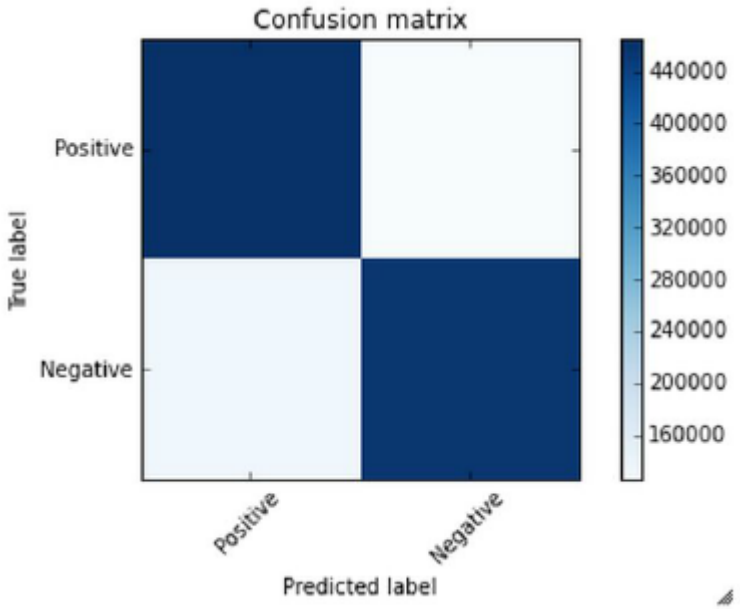

Figure 4.1.4.4: Colormap of the confusion matrix related to the naive bayes classifier used.

Hopefully we can distinguish that the number of true positive and true negative classified tweetsis higher than the number of false and positive and negative tweets. However from this result we try to improve the accuracy of the classifier by experimenting different techniques and we repeatthe same process using the k-fold cross validation to evaluate its averaged accuracy.

4.1.5 Improvements

From the baseline, the goal is to improve the accuracy of the classifier, which is 0.77, inorder to determine better which tweet is positive or negative. There are several ways of doingthis and we present only few possible improvements (or not).First we could try to removed what we called, stop words. Stop words usually refer to the most common words in the English language (in our case) such as: "the", "of", “to” and so on.They do not indicate any valuable information about the sentiment of a sentence and it can benecessary to remove them from the tweets in order to keep only words for

which we areinterested. To do this we use the list of 635 stopwords that we found. In the table below, youcan see the most frequent words in the data set with their counts,

```
[('||target||', 780664),
 ('i', 778070),
 ('to', 614954),
 ('the', 538566),
 ('a', 383910),
 ('you', 341545),
 ('my', 336980),
 ('and', 316853),
 ('is', 236393),
 ('for', 236018),
 ('it', 235435),
 ('in', 217350),
 ('of', 192621),
 ('on', 169466),
 ('me', 163900),
 ('so', 158457),
 ('have', 150041),
 ('that', 146260),
 ('out', 143567),
 ('but', 132969)]
```

Table 4.1.5.1: Most frequent words in the data set with their corresponding count.

We can derive from the table, some interesting statistics like the number of times the tags usedin the preprocessingstep appear,

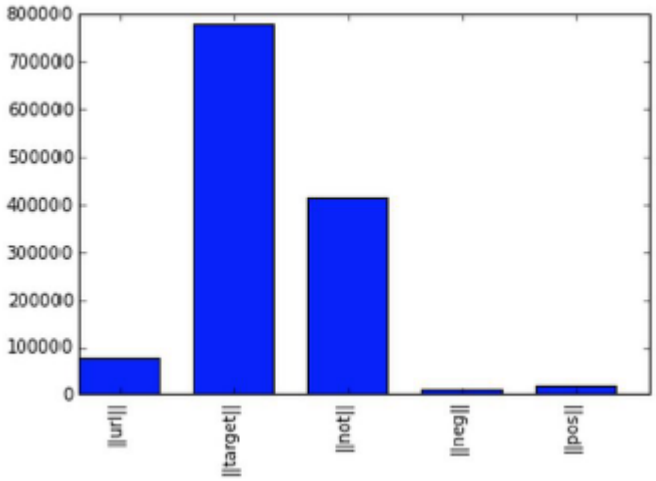

Figure 4.1.5.1: Tags in the data set with their corresponding count.

Recall that ||url|| corresponds to the URLs, ||target|| the twitter usernames with the symbol "@"before, ||not|| replaces the negation words, ||pos|| and ||neg|| replace the positive and negativesmiley respectively. After removing the stop words we get the results below,

```
Total tweets classified: 1183958
Score: 0.758623708326
Confusion matrix:
[[437311 154015]
 [136343 456289]]
```

Figure 4.1.5.2: Result of the naive bayes classifier with stopwords removed.

Compared to the previous result, we lose in accuracy and the number of 0.02 false positive goesfrom 126305 to 154015. We conclude that stop words seem to be useful for our classificationtask and remove them do not represent an improvement.

We could also try to stem the words in the data set. **Stemming** is the process by which endings are removed from words in order to remove things like tense or plurality. The stem formof a word could not exist in a dictionary (different from Lemmatization). This technique allows tounify words and reduce the dimensionality of the dataset. It's not appropriate for all cases butcan make it easier to connect together tenses to see if you're covering the same subject matter.It is faster than **Lemmatization** (remove inflectional endings only and return the base ordictionary form of a word, which is known as the lemma). Using the library NLTK which is alibrary in Python specialized in natural language processing, we get the following results afterstemming the words in the data set,

```
Total tweets classified: 1183958
Score: 0.773106857186
Confusion matrix:
[[462537 128789]
 [138039 454593]]
```

Figure 4.1.5.3: Result of the naive bayes classifier after stemming.

We actually lose 0.002 in accuracy score compared to the results of the baseline. We concludethat stemming words does not improve the classifier’s accuracy and actually do not make any sensible changes.

4.2 N-Gram

An important note is that n-gramclassifiers are in fact a generalization of Naive Bayes. Aunigram classifier with Laplace smoothing corresponds exactly to the traditional naïve Bayes classifier.

Since we use bag of words model, meaning we translate this sentence: "I don't like chocolate"into "I", "don't", "like", "chocolate", we could try to use bigram model to take care of negationwith "don't like" for this example. Using bigrams as feature in the classifier we get the followingresults,

```
Total tweets classified: 1183958
Score: 0.784149223247
Confusion matrix:
[[480120 111206]
 [138700 453932]]
```

Formula 4.2.1: Results of the naive bayes classifier with bigram features

Using only bigram features we have slightly improved our accuracy score about 0.01. Based onthat we can think of adding unigram and bigram could increase the accuracy score more.

```
Total tweets classified: 1183958
Score: 0.795370054626
Confusion matrix:
[[486521 104805]
 [132142 460490]]
```

Formula 4.2.3: Results of the naive bayes classifier with unigram and bigram features

and indeed, we increased slightly the accuracy score about 0.02 compared to the baseline.

All the code datails we have uploaded in our github profile. For more info , Please refer the following link:

- https://github.com/Twitter-Sentiment-Analysis-2020/Twitter-Sentiment-Analysis

CHAPTER – 5: CONCLUSION & FUTURE SCOPE

CONCLUSION

Nowadays, sentiment analysis or opinion mining is a hot topic in machine learning. W e are still far to detect the sentiments of s corpus of texts very accurately because of th e complexity in the English language and even more if we consider other languages su ch as Chinese.

In this project we tried to show the basic way of classifying tweets into positive or neg ative category using Naive Bayes as baseline and how language models are related to t he Naive Bayes and can produce better results. We could further improve our classifie r by trying to extract more features from the tweets, trying different kinds of features, tuning the parameters of the naïve Bayes classifier, or trying another classifier all toge -ther.

FUTURE SCOPE

The task of sentiment analysis, especially in the domain of micro-bloging, is still in the developing stage and far from complete. So we propose a couple of ideas which we feel are worth exploring in the future and may result in further improved performance.

Right now we are exploring Parts of Speech separate from the unigram models, we can try to incorporate POS information within our unigram models in future. So say instead of calculating a single probability for each word like P(word | obj) we could instead have multiple probabilities for each according to the Part of Speech the word belongs to. For example we may have P(word | obj, verb), P(word | obj, noun) and P(word | obj, adjective). Pang et al. used a somewhat similar approach and claims that appending POS information for every unigram results in no significant change in performance (with Naive Bayes performing slightly better and SVM having a slight decrease in performance), while there is a significant decrease in accuracy if only adjective unigrams are used as features. However these results are for classification of reviews and may be verified for sentiment analysis on micro blogging websites like Twitter.

In this project we are focussing on general sentiment analysis. There is potential of work in the field of sentiment analysis with partially known context. For example we

noticed that users generally use our website for specific types of keywords which can divided into a couple of distinct classes, namely: politics/politicians, celebrities, products/brands, sports/sportsmen, media/movies/music. So we can attempt to perform separate sentiment analysis on tweets that only belong to one of these classes (i.e. the training data would not be general but specific to one of these categories) and compare the results we get if we apply general sentiment analysis on it instead.

CHAPTER – 6: BIBLIOGRAPHY

1. Alexander Pak, Patrick Paroubek. 2010, Twitter as a Corpus for Sentiment Anal ysis and Opinion Mining.
2. Alec Go, Richa Bhayani, Lei Huang. Twitter Sentiment Classification using Dist ant Supervision.
3. Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, Rebecca Passonneau. Sentiment Analysis of Twitter Data.
4. Fuchun Peng. 2003, Augmenting Naive Bayes Classifiers with Statistical Langu age Models
5. ManjuVenugopalan and Deepa Gupta ,Exploring Sentiment Analysis on Twitter Data, IEEE 2015
6. Brett Duncan and Yanqing Zhang, Neural Networks for Sentiment Analysis on Twitter.2017
7. EfthymiosKouloumpis and Johanna Moore,IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 4, No 3, July 2012
8. Afroze Ibrahim Baqapuri, Twitter Sentiment Analysis: The Good the Bad and the OMG!, Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media.2011