



DEPARTMENT OF INFORMATION & COMMUNICATION TECHNOLOGY  
**MANIPAL INSTITUTE OF TECHNOLOGY**  
(A Constituent College of Manipal University)  
MANIPAL – 576104, KARNATAKA, INDIA



Data Warehousing & Data Mining Lab Project  
**B.TECH (IT) SIXTH SEMESTER**

Buzz Miner

Submitted by

Amit Banerjee (Roll-36) 130911290  
Abhinav Agrawal (Roll-37) 130911304

# Index

<b>1. Abstract.....</b>	<b>3</b>
<b>2. Problem Statement.....</b>	<b>4</b>
<b>3. Objectives.....</b>	<b>5</b>
<b>4. Literature Survey.....</b>	<b>6</b>
<b>5. Design.....</b>	<b>7</b>
<b>6. Methodology.....</b>	<b>8</b>
<b>7. Implementation.....</b>	<b>10</b>
<b>8. Conclusion.....</b>	<b>11</b>
<b>9. References.....</b>	<b>12</b>
<b>10.Future Work.....</b>	<b>13</b>
<b>11.Appendix.....</b>	<b>14</b>

# ABSTRACT

*Social media can be an incredibly important tool for any business. But at the same time it can also be very overwhelming. The team's purpose is to produce and share great content, while interacting with customers and finding other prospects. The potential of social media in this field isn't always harnessed to its full capacity. As soon as a product is unveiled, people go berserk with their reviews online. If these reviews were to be consolidated in one place and analysed, this would provide an extremely efficient way for companies to decide on a plethora of decisions like Where to launch their product first? What is the product's primary consumer base? etc.*

Our primary motivation for this project is to get an idea of sentiments of people across the world for a product prior to its debut in the market so that the target location for launching the product can be decided according to the sentiments in different areas. Furthermore, we can also get an idea of sentiments around a product after it has been released for more advancements. This business logic can help any company to launch the product as per demand.

## Problem statement

Users of this application will provide it with a buzzword (keyword or phrase) along with the number of tweets to be analyzed from the micro blogging social networking site Twitter.

The user has two choices for the classifier for sentiment analysis :

1. Our own custom implementation of Naive Bayes Classifier
2. The deep learning Alchemy API Classifier.

Given this input, Buzzminer should fetch tweets as data in the specified range from Twitter based upon location, language etc and perform sentiment analysis on the set of data streamed either using Naive Bayes Classifier or deep learning Alchemy API to classify the data set into Positive, Negative and Neutral classes.

The end result should be a summary of the data collected along with proper visualization to let the users know about the opinion based statistics for the particular buzzword they entered.

# Objectives

Motive of this project is to stream/find the data on a social media platform using specific set of keywords related to the “Product to be launched” or any buzzword that the user has in mind and then segregate the data according to continents. Then further the with the help of Naive Bayes Classifier the data obtained according to continents is classified into three kinds of sentiments : **Positive**, **Negative** and **Neutral**. The end result will be a visualization using charts to summarize our findings from the mined data.

The main Objectives are:

- To stream proper data from the Social Networking channel which will be our test data.
- To find a proper training data set for our **Naive Bayesian classifier** i.e. a data set of positive and negative phrases from a well qualified data source.
- To apply the algorithm for classification on the test data using adequate knowledge about of training data.
- Calculating the sentiment scores for each tweet from different geographical locations.
- Summarizing the sentiment scores of the classes (**positive** | **negative** | **neutral**) by making use of proper data visualization tools.
- Determining the geographical areas where the product got the most amount of positive opinions and also the area with the most amount of negative opinions.

# Literature survey

- **Twitter Sentiment Classification using Distant Supervision by Alec Go , Richa Bhayani , Lei Huang , Stanford University Stanford, CA 94305 [1]**

We introduce a novel approach for automatically classifying the sentiment of Twitter messages. These messages are classified as either positive or negative with respect to a query term. This is useful for consumers who want to research the sentiment of products before purchase, or companies that want to monitor the public sentiment of their brands. There is no previous research on classifying sentiment of messages on micro blogging services like Twitter.

We present the results of machine learning algorithms for classifying the sentiment of Twitter messages using distant supervision. Our training data consists of Twitter messages with emoticons, which are used as noisy labels. This type of training data is abundantly available and can be obtained through automated means. We show that machine learning algorithms (Naive Bayes, Maximum Entropy, and SVM) have accuracy above 80% when trained with emoticon data.

This paper also describes the preprocessing steps needed in order to achieve high accuracy. The main contribution of this paper is the idea of using tweets with emoticons for distant supervised learning.

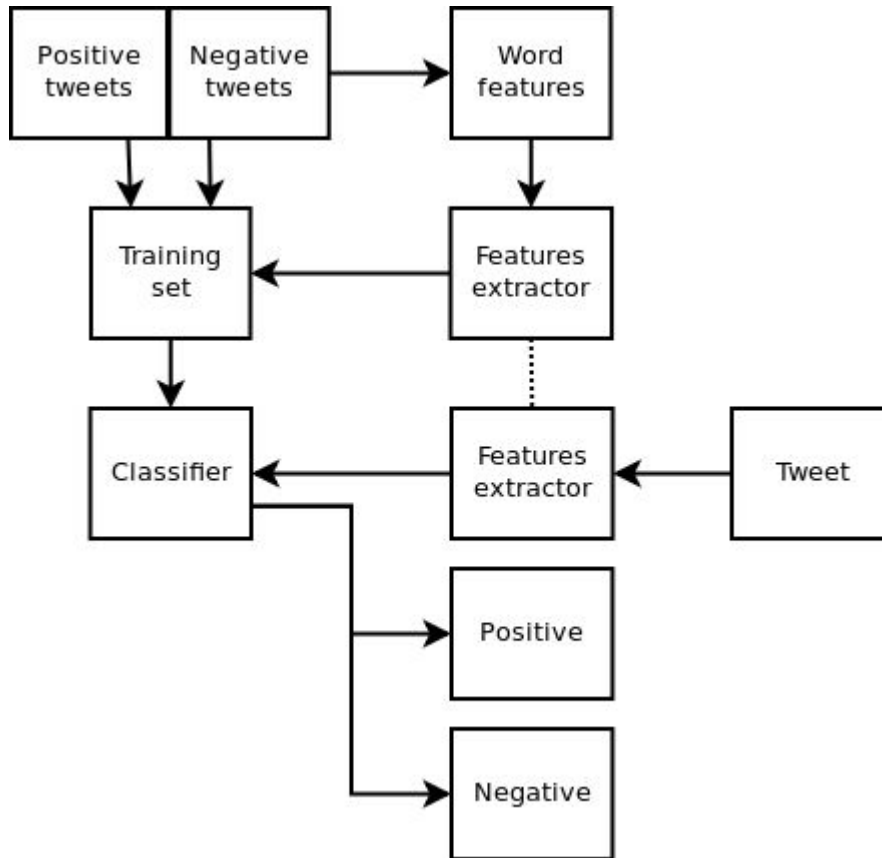
- **Opinion Mining and Sentiment Analysis by Bo Pang, Yahoo! Research and Lillian Lee, Computer Science Department, Cornell University [2]**

An important part of our information-gathering behavior has always been to find out what other people think. With the growing availability and popularity of opinion-rich resources such as online review sites and personal blogs, new opportunities and challenges arise as people now can, and do, actively use information technologies to seek out and understand the opinions of others. The sudden eruption of activity in the area of opinion mining and sentiment analysis, which deals with the computational treatment of opinion, sentiment, and subjectivity in text, has thus occurred at least in part as a direct response to the surge of interest in new systems that deal directly with opinions as a first-class object.

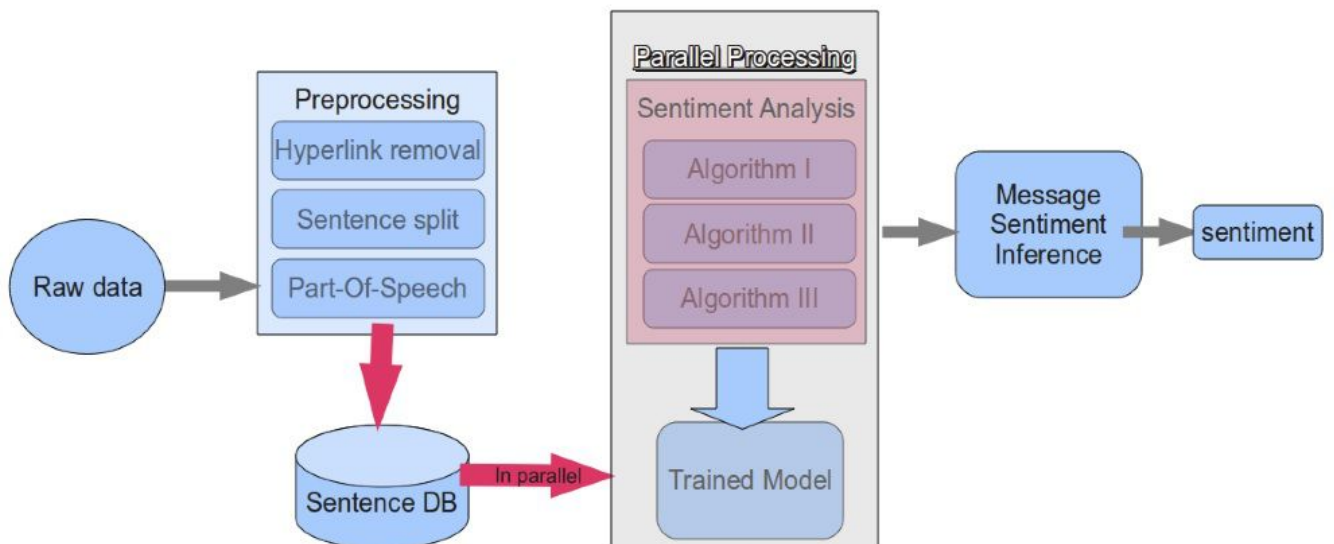
This survey covers techniques and approaches that promise to directly enable opinion-oriented information-seeking systems. Our focus is on methods that seek to address the new challenges raised by sentiment-aware applications, as compared to those that are already present in more traditional fact-based analysis. We include material on summarization of evaluative text and on broader issues regarding privacy, manipulation, and economic impact that the development of opinion-oriented information-access services gives rise to. To facilitate future work, a discussion of available resources, benchmark datasets, and evaluation campaigns is also provided.

# Design

- Custom Naive Bays Classifier



- Alchemy API



# Methodology

## ● NAIVE BAYES FOR SENTIMENT ANALYSIS

Sentiment analysis aims to detect the attitude of a text. A simple subtask of sentiment analysis is to determine the polarity of the text: positive, negative or neutral. In this tutorial we concentrate on detecting if a short text like a Twitter message is positive or negative. For example: for the tweet “Have a nice day!” the algorithm should tell us that this is a positive message. For the tweet “I had a bad day” the algorithm should tell us that this is a negative message.

From the machine learning domain point of view this can be seen as a classification task and naive Bayes is an algorithm which suits well this kind of task.

The naive Bayes algorithm uses probabilities to decide which class best matches for a given input text. The classification decision is based on a model obtained after the training process. Model training is done by analysing the relationship between the words in the training text and their classification categories. The algorithm is considered naive because it assumes that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 3" in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness and diameter features.

Each text we will classify contains words noted with  $W_i$  ( $i=1..n$ ). For each word  $W_i$  from the training data set we can extract the following probabilities (noted with  $P$ ):

***$P(W_i \text{ given Positive}) = (\text{The number of positive Texts with the } W_i) / \text{The number of positive Texts}$***

***$P(W_i \text{ given Negative}) = (\text{The number of negative Texts with the } W_i) / \text{The number of negative Texts}$***

For the entire test set we will have:



$$P(\text{Positive}) = (\text{The number of positive Texts}) / \text{The total number of Texts}$$

$$P(\text{Negative}) = (\text{The number of negative Texts}) / \text{The number of Texts}$$

For calculating the probability of a Text being positive or negative, given the containing words we will use the following theorem:

$$P(\text{Positive given Text}) = P(\text{Text given Positive}) \times P(\text{Positive}) / P(\text{Text})$$

$$P(\text{Negative given Text}) = P(\text{Text given Negative}) \times P(\text{Negative}) / P(\text{Text})$$

As  $P(\text{Text})$  is 1, as each Text will be present once in the training set, we will have:

$$P(\text{Positive given Text}) = P(\text{Text given Positive}) \times P(\text{Positive}) = P(W1 \text{ give Positive}) \times P(W2 \text{ given Positive}) \times \dots \times P(Wn \text{ given Positive}) \times P(\text{Positive})$$

$$P(\text{Negative given Text}) = P(\text{Text given Negative}) \times P(\text{Negative}) = P(W1 \text{ give Negative}) \times P(W2 \text{ given Negative}) \times \dots \times P(Wn \text{ given Negative}) \times P(\text{Negative})$$

At the end one will compare  $P(\text{Positive given Text})$  and  $P(\text{Negative given Text})$  and the term with the higher probability will decide if the text is positive or negative.

## ● ALCHEMY API FOR SENTIMENT ANALYSIS

AlchemyAPI's sentiment analysis API provides easy-to-use mechanisms to identify the positive or negative sentiment within any document or webpage. The sentiment analysis API is capable of computing document-level sentiment, sentiment for user-specified targets, entity-level sentiment, quotation-level sentiment, directional-sentiment and keyword-level sentiment. These multiple modes of sentiment analysis provide for a variety of use cases ranging from social media monitoring to trend analysis.

AlchemyAPI's sentiment analysis algorithm looks for words that carry a positive or negative connotation then figures out which person, place or thing they are referring to. It also understands negations (e.g.: "this car is good" vs. "this car is not good") and modifiers (i.e. "this car is good" vs. "this car is really good"). The sentiment analysis API works on documents large and small, including news articles, blog posts, product reviews, comments and Tweets.

# Implementation

The runnable instance of Buzz Miner at <http://buzz.mitportals.in> allows a user to input a search term and choose a number of tweets to fetch/stream from the Twitter search API queried with geolocation data. Two classifiers are available, our own Naive-Bayes implementation and the Alchemy API. The custom classifier is very basic. It is trained off a corpus of pre-classified tweets obtained from Sentiment140-- the classification of these tweets is again very basic (positive emoticons imply positive, negative emoticons imply negative), but it certainly outperforms classifying 1.6m by hand. The training involves building a dictionary of tokens that appear in the corpus and assigning each a probability of being positive/negative based on its frequency in each class. AlchemyAPI, uses deep learning for the process of sentiment classification.

- Platform: Ubuntu/Linux based Apache Server , with a minimum of 512 MB of RAM for hosting Buzzminer as a Web App.
- Programming languages and Software packages:

## **Front End:**

HTML5, CSS3, JavaScript(Visualization using Nick Downie's Chart.js)

## **BackEnd:**

MySQL Database, Alchemy API, Twitter 1.1 API (/tweets.json)

## **Server Side Scripting:**

PHP ( the naive bayes classification algorithm as well as api calls)

## **Wrapper classes for PHP:**

J7mbo's PHP wrapper for Twitter API calls.

Achemy\_api PHP by Alchemy, an IBM Company

## Conclusion

The task of sentiment analysis, especially in the domain of micro-blogging, is still in the developing stage and far from complete. We have analyzed the test data set with a very small upper bound imposed on our training data sets which include roughly 8000-11800 tweets gathered based upon the emoticon classification hence the training model can be fairly improved by gathering more data to improve accuracy.

The web app for BUZZ Miner can be used to get a fair idea of sentiments from all over the world for a varied number of search queries thus helping in product placement, launch location decision purposes etc. We have implemented the main algorithm keeping in mind that uni-grams are used for dictionary wise comparison while calculating the probabilistic sentiment score, which will be used to display the data visualization in the form of a doughnut chart.

The machine learning and data mining algorithms taught to us find a very important place in the implementation of this project right from collection of data set using proper parameters for data cleaning to optimizing the code for sentiment score calculation and classification.

# References

[1]A. Go, R. Bhayani, and L. Huang, "For academics - sentiment140 - A Twitter sentiment analysis tool,". [Online]. Available: <http://help.sentiment140.com/for-students/>. Accessed: Apr. 20, 2016

[2]Bo Pang , Lillian Lee , "Opinion mining and sentiment analysis" (2008)<sup>1</sup>Yahoo! Research, 701 First Ave. Sunnyvale, CA 94089, U.S.A., [bopang@yahoo-inc.com](mailto:bopang@yahoo-inc.com)<sup>2</sup>Computer Science Department, Cornell University, Ithaca, NY 14853, U.S.A., [llee@cs.cornell.edu](mailto:llee@cs.cornell.edu)

[3]"<http://streamhacker.com/2010/05/10/text-classification-sentiment-analysis-naive-bayes-classifier/>,".

# Future Work

This project covers the very basics of sentiment analysis . There is a lot of potential on this subject and many applications are developed everyday which uses opinion mining in one way or another.

This project can be taken forward by some of the following work -

- **Building a classifier for subjective vs. objective tweets**
- **Handling negation**
- **Handling comparisons**
- **The "aboutness" problem**
- **Determine context switches**
- **Sarcasm detection**
- **Topic classification for tweets**
- **Applying sentiment analysis to Facebook messages**

# APPENDIX

## CODE Snippets:

### Twitter data streaming and sentiment analysis function calls

```

$loc=array(
    'Asia'=>'34.0479,100.6197,10000mi',
    'Europe'=>'54.5260,15.2551,10000mi',
    'North America'=>'54.5260,-105.2551,10000mi',
    'South America'=>'-8.7832,-55.4915,10000mi',
    'Australia'=>'-25.2744,133.7751,10000mi',
    'Africa'=>'-8.7832,34.5085,10000mi',
    'Antarctica'=>'-82.8628,135.0000,10000mi'
);
$locI=array_values($loc);
$locn = array_keys($loc);
$pos_count= [0,0,0,0,0,0,0] ;
$neg_count= [0,0,0,0,0,0,0] ;
$neu_count= [0,0,0,0,0,0,0] ;
$cust_pos=  [0,0,0,0,0,0,0] ;
$cust_neg=  [0,0,0,0,0,0,0] ;

for($i=0;$i<7;$i++)
{
    $url = 'https://api.twitter.com/1.1/search/tweets.json';
    $x1 = " ";
    $y1 = "%20";
    $z1 = $_POST['q'];
    $qry= str_replace($x1,$y1,$z1);
    $getfield ='?lang=en&count=' . $_POST['num_tweets'] . '&geocode=' . $locI[$i].
    '&q=' . $qry . '-filter:retweets';
    //$getfield = '?q=manipal';
    $requestMethod = 'GET';
    $twitter = new TwitterAPIExchange($settings);
    //var_dump($twitter);
    $tweets = json_decode($twitter->setGetfield($getfield)
        ->buildOAuth($url, $requestMethod)
        ->performRequest());
    //var_dump($tweets);
    echo 'Region: <b>'. $locn[$i]. '</b> <br>';
    foreach ($tweets -> statuses as $t) {
        // foreach ($tweet as $t){
        //calculate the sentiment of each tweet
    }
}

```

```

if ($t -> text != ""){
    if($_POST['classifier']=='alchemy'){
        $response = $alchemyapi -> sentiment('text', $t -> text ,
null);
        display_tweet($t, $response);
        // store_tweet_db($_POST['q'], $t->text,
$response['docSentiment']['type'],$response);
        if ($response['docSentiment']['type']=='neutral'){
            $neu_count[$i]++;
        }
        if ($response['docSentiment']['type']=='positive'){
            $pos_count[$i]++;
        }
        if ($response['docSentiment']['type']=='negative'){
            $neg_count[$i]++;
        }
    }

    else {
        $sentiment = $custom_sent ->classify($t->text);
        display_custom($t,$sentiment);
        if ($sentiment=='pos'){
            $cust_pos[$i]++;
        } else {
            $cust_neg[$i]++;
        }
    }

}

}
echo '<br>';
echo "Sentiment Scores Summary for ".$locn[$i].": ". "&nbsp;";
if($_POST['classifier']=='alchemy')
{
echo 'positives: ' . $pos_count[$i] . '&nbsp;';
echo 'negatives: ' . $neg_count[$i] . '&nbsp;';
echo 'neutrals: ' . $neu_count[$i]. '<br>';
}
else if($_POST['classifier']=='custom')
{
echo 'positives: ' . $cust_pos[$i] . '&nbsp;';
echo 'negatives: ' . $cust_neg[$i] . '<br>';
}
echo '<hr>';
}
}

```

## Naive Bayes Classifier

```

<?php

class Sentiment1 {

    private $dictionary = array();
    private $classes = array('pos','neg');
    private $classWordCount = array('pos'=>0,'neg'=>0); //w.c. per class
    private $classTweetCount = array('pos'=>0,'neg'=>0);
    private $wordCount = 0;
    private $tweetCount = 0;
    private $priors = array('pos' => 0.5,'neg' => 0.5);

    public function displayFields() {
        //this function is purely for testing purposes.
        //print_r($this->classWordCount);
        //print_r($this->dictionary);
        print_r($this-> classWordCount);
        echo "<br>";
        print_r($this-> wordCount);
        echo "<br>";
        print($this-> tweetCount);
        echo "<br>";
        print_r($this -> classTweetCount);
        echo "<br>";
        print_r($this -> dictionary);
    }

    public function tokenize($tweet){
        $tweet = strtolower($tweet);
        //remove usernames from the tweet
        $tweet = preg_replace("/@([A-Za-z0-9_]{1,15})/", '', $tweet);
        //remove hashtags
        $tweet = preg_replace("/(#\w+)/", '', $tweet);
        $tweet = trim($tweet);
        $tokens = preg_split("/[\s,]*\\\\"([^\s\\\\"]+\\\\"[\s,]*" . "[\s,]*"([^\s]+)"[\s,]*" . "[\s,]+"/,
$tweet);
        //$tokens = explode(" \n\t", $tweet);
        return $tokens;
    }
}

```



```

//takes a tweet as input and returns its classification
public function classify($tweet){
    //create array of words in tweet
    $tokens = $this -> tokenize($tweet);
    $totalCount = 0;
    //create array to store counts for each class (pos,neg,neu)
    $counts = array();

    //loop through each class
    foreach ($this -> classes as $class){

        //set default for class = 1
        $counts[$class] = 1;

        //loop through each word in tweet and check if they're in the dictionary
        foreach ($tokens as $token){
            if (isset($this -> dictionary[$token][$class])){
                $count = $this -> dictionary[$token][$class];
            } else {
                $count = 0; //word is not in dictionary
            }
            //keep multiplicative tally for the counts per class, +1 for Laplace smoothing
            $counts[$class] *= ($count + 1);
        }
        //multiply the count for the class by its prior probability
        $counts[$class] = $counts[$class] * $this -> priors[$class];
    }

    foreach ($this -> classes as $class) {
        $totalCount += $counts[$class];
    }
    //give each class count as a percentage
    foreach ($this -> classes as $class){
        $counts[$class] = $counts[$class]/$totalCount;
    }
    //sort the counts array so the class with the highest count is first
    arsort($counts);
    //return the first key in the sorted counts array (corresponds with the class)
    $tweetClass = key($counts);
    return $tweetClass;
}

```

```

function train ($limit=0) {
    //open database for positive and negative training data
    include 'db_connection.php';
    require_once 'functions.php';

    //get all negative data
    $queryneg = "SELECT * FROM negtrain";
    $resultneg = mysqli_query($connection,$queryneg);
    confirm_query($resultneg);
    //return $resultneg;

    $class = 'neg';
    $i = 0;
    while($negTrainData = mysqli_fetch_assoc($resultneg)) {
        //if limit is set (for testing purposes) break when limit breached
        if ($i > $limit && $limit > 0){
            break;
        }
        //print($negTrainData['content']);
        $i++;
        $this->tweetCount++;
        $this->classTweetCount[$class]++;
        $tokens = $this->tokenize($negTrainData['content']);
        //print_r($tokens);
        foreach ($tokens as $token) {
            if(!isset($this->dictionary[$token][$class])){
                $this->dictionary[$token][$class]=0;
            }
            $this->dictionary[$token][$class]++;
            $this->classWordCount[$class]++;
            $this->wordCount++;
        }
        //print($i . "\n");
    }
    mysqli_free_result($resultneg);

    //get all positive data
    $query = "SELECT * FROM postrain";
    $resultpos = mysqli_query($connection, $query);
    confirm_query($resultpos);
    //return $resultneg;
    $class = 'pos';
    $j=0;

```

```

while($posTrainData = mysqli_fetch_assoc($resultpos)) {
    //if limit is set (for testing purposes) break when limit breached
    if ($j > $limit && $limit > 0){
        break;
    }
    $j++;
    $this -> tweetCount++;
    $this -> classTweetCount[$class]++;
    $tokens = $this -> tokenize($posTrainData['content']);
    foreach ($tokens as $token) {
        if(!isset($this -> dictionary[$token][$class])){
            $this -> dictionary[$token][$class]=0;
        }
        $this -> dictionary[$token][$class]++;
        $this -> classWordCount[$class]++;
        $this -> wordCount++;
    }
}
mysqli_free_result($resultpos);

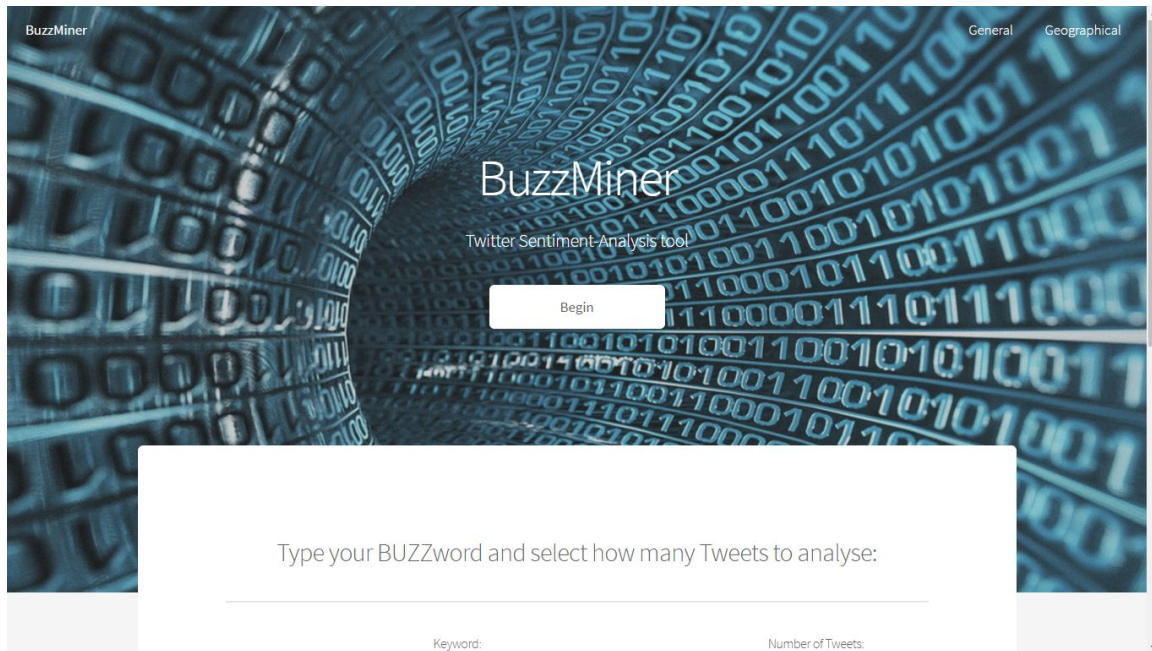
if (isset($connection)) {
    mysqli_close($connection);
}

} //end of train()
}

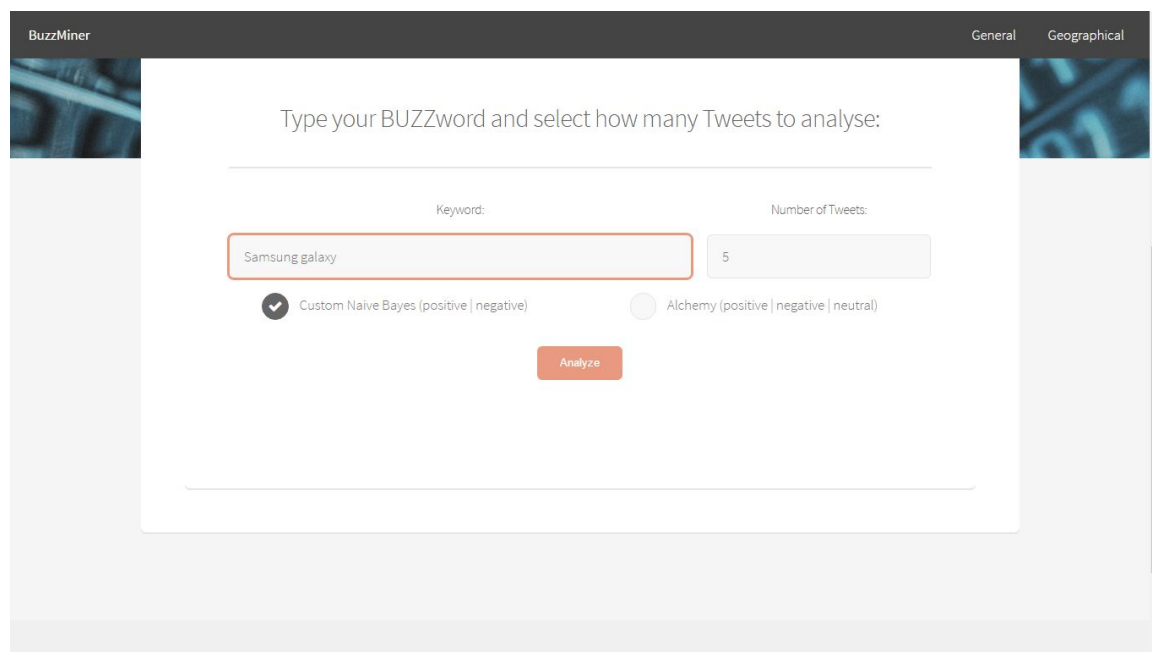
?>

```

# SCREENSHOTS



The screenshot shows the BuzzMiner website with a background of binary code. The header includes the BuzzMiner logo and navigation links for 'General' and 'Geographical'. The main heading is 'BuzzMiner' with the subtitle 'Twitter Sentiment Analysis tool'. A 'Begin' button is centered below the heading. Below this, a white box contains the instruction 'Type your BUZZword and select how many Tweets to analyse:'. At the bottom of the white box, there are two labels: 'Keyword:' and 'Number of Tweets:'.




This screenshot shows the same BuzzMiner interface as the previous one, but with the input fields filled. The 'Keyword' field contains 'Samsung galaxy' and the 'Number of Tweets' field contains '5'. Below these fields, there are two radio button options: 'Custom Naive Bayes (positive | negative)' which is selected, and 'Alchemy (positive | negative | neutral)'. An 'Analyze' button is positioned below the radio buttons. The background and header remain the same.

BuzzMiner

General Geographical


Analyzing Keyword : Samsung galaxy



Sentiment: Positive

Xauqro\_\_Rautio


BRAND NEW SAMSUNG GALAXY S7 32GB BLACK (VERIZON) FACTORY UNLOCKED - Bid Now! Only \$550.0 <https://t.co/yMyxhcmClS> <https://t.co/PuZdgyiu8D>



Sentiment: Negative

Xauqro\_\_Rautio

SAMSUNG GALAXY S5 SM-G900F WHITE UNLOCKED MOBILE CELL PHONE - Bid Now! Only \$102.5 <https://t.co/YRSp3Ri70T> <https://t.co/ZG0y7DSFq3>



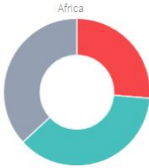
Sentiment: Negative

BuzzMiner

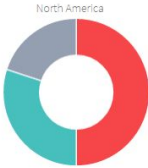
General Geographical

Sentiment Scores Summary for Antarctica: positives: 7 negatives: 7 neutrals: 5

Most Positive Opinions




Most Negative Opinions




BuzzMiner

General Geographical

Xiaomi pushed out of global top 5: IDC <https://t.co/64zjVOniHy>





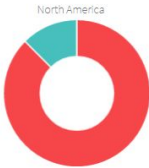
Sentiment: Negative

Dual\_Media

#dualmedia Oppo and Vivo replace Xiaomi and Lenovo on the top smartphone makers list <https://t.co/5viryA0m70>

Sentiment Scores Summary for Antarctica: positives: 0 negatives: 20

Most Positive Opinions



Most Negative Opinions

