**GENERIC SENTIMENT ANALYSIS USING**

**MACHINE LEARNING**

**Submitted in Partial Fulfillment of the Requirements**

**For the Award**

**Of**

**Master of Technology**

**In**

**Electronics and Communication**

**Guide : Submitted By:**

**\*Member1\* \*Member2\***

**\*Member3\* \*Member4\***



**Electronics and Communication Department**

**\*College Name\***

****

**CERTIFICATE**

We hereby certify that the work which is being presented in the project report entitled

**“GENERIC SENTIMENT ANALYSIS USING MACHINE LEARNING”** done by Ms. \*\*, Roll No - \*\* is an authentic work carried out by her at \*College Name\* under my guidance. The matter embodied in this project work has not been submitted earlier for the award of diploma or degree to the best of my knowledge and belief.

Date: Signature of the Guide

Name of the guide:

Designation:

**ACKNOWLEDGEMENT**

We would like to acknowledge the contributions of the following people, without whose help and guidance this project would not have been completed.

We acknowledge the counsel and support of our project mentor \*Coordinator Nmae\*, Electronics and Communication Engineering Department, with respect and gratitude whose expertise, guidance, support, encouragement, and enthusiasm has made this project possible. His feedback vastly improved the quality of this report and provided an enthralling experience. I am indeed proud and fortunate to be supervised by her.

We are also thankful to **H.O.D.** Electronics and Communication Engineering Department, \*College Name\***,** New Delhi for his constant encouragement, valuable suggestions and moral support and blessings.

We shall ever remain indebted to the faculty members of **\*College Name\*, New Delhi.**

Finally, yet importantly, we would like to express our heartfelt thanks to our friends/classmates for their help and wishes for successful completion of this project. This acknowledgement will remain incomplete if we fail to express our deep sense of obligation to our parents and god for their consistent blessings and encouragement.

**\*Member4\***

**TABLE OF CONTENTS**

certificatei

acknowledgementii

abstract2

tABLE OF CONTENTSIv

LISt OF TABLES1

LISt OF FIGURES1

INTRODUCTION3

MOTIVATION3

OBJECTIVE3

BACKGROUND4

NATURAL LANGUAGE PROCESSING4

COMPUTATIONAL LOGISTICS5

APPLICATIONS6

ANALYSIS OF SENTIMENT8

DIFFERENCE BETWEEN TEXT AND SENTIMENT8

MACHINE LEARNING9

LEXICON BASED MACHINE LEARNING9

EXPERIMENTS10

PROPOSED FRAMEWORK10

TWITTER API AND WORKING11

PHP BASED API FILTER11

FRONTEND11

HANDSHAKING BETWEEN MATLAB AND TWITTER12

ALGORITHM12

conclusion AND FUTURE DIRECTIONS13

appendix a14

appendix b14

appendix C14

references15

**Table of Figures**

Figure 1 -Sentiment analysis flow 8

Figure 2 -Graph 12

**List of Tables**

1. **ABSTRACT**

Twitter is a popular microblogging service where users create status messages (called "tweets"). These tweets sometimes express opinions about different topics.

Twitter has been a great source for social media information and depending upon that information, the mode of sentiment can be analyzed.

The sentiments play a major role in defining the consistency or launch of a new product or service in a provided duration. For instance, a movie review after its release can be accurately measured by the analysis of tweets that the users are making about it. Similarly, at the time of a launch of a new IPO, its valuation and popularity can be measured by confining the reaction of masses over the offering. It can be used by political parties to gain accurate responses before elections. A lot of things can be done by twitter sentiment analysis.

Keywords – Sentiment Analysis, MATALB, Twitter, API, PHP

**INTRODUCTION**

**MOTIVE:**

Tweets have grown as a major source of information and people use it as a means to present their views on certain issues and topics. If these reviews can be analyzed with precision it can predict the flow and can provide a near to true picture to fast forward future calls.

Also it can be a great source of information based on the sentiments, the positive mindset promotion via simple tweets can trend a lot of happy vibes and can be vital while creating a new product or services.

**OBJECTIVE:**

The project presents an implementation on sentiment analysis using MATLAB. The analysis has various applications ranging from market data research to movie reviews. Several companies are using sentiment analysis for prediction.

**APPLICATIONS**

* It can be used to predict the trend sentiments.
* The trend sentiments can be used to review various events.
* Financially, it can help review stocks and analyze the flow.
* Reviews can be used to improve your product and services.

**BACKGROUND**

**Natural Language Processing**

NLP is a way for computers to analyze, understand, and derive meaning from human language in a smart and useful way. By utilizing NLP, developers can organize and structure knowledge to perform tasks such as automatic summarization, translation, named entity recognition, relationship extraction, sentiment analysis, speech recognition, and topic segmentation.

“Apart from common word processor operations that treat text like a mere sequence of symbols, NLP considers the hierarchical structure of language: several words make a phrase, several phrases make a sentence and, ultimately, sentences convey ideas,” John Rehling, an NLP expert at Meltwater Group, said in How Natural Language Processing Helps Uncover Social Media Sentiment. “By analyzing language for its meaning, NLP systems have long filled useful roles, such as correcting grammar, converting speech to text and automatically translating between languages.”

NLP is used to analyze text, allowing machines to understand how human’s speak. This human-computer interaction enables real-world applications like automatic text summarization, sentiment analysis, topic extraction, named entity recognition, parts-of-speech tagging, relationship extraction, stemming, and more. NLP is commonly used for text mining, machine translation, and automated question answering.

NLP is characterized as a hard problem in computer science. Human language is rarely precise, or plainly spoken. To understand human language is to understand not only the words, but the concepts and how they’re linked together to create meaning. Despite language being one of the easiest things for humans to learn, the ambiguity of language is what makes natural language processing a difficult problem for computers to master.

**Computational Logistics**

NLP algorithms are typically based on machine learning algorithms. Instead of hand-coding large sets of rules, NLP can rely on machine learning to automatically learn these rules by analyzing a set of examples (i.e. a large corpus, like a book, down to a collection of sentences), and making a statically inference. In general, the more data analyzed, the more accurate the model will be.

* Summarize blocks of text using Summarizer to extract the most important and central ideas while ignoring irrelevant information.
* Create a chat bot using Parsey McParseface, a language parsing deep learning model made by Google that uses Point-of-Speech tagging.
* Automatically generate keyword tags from content using AutoTag, which leverages LDA, a technique that discovers topics contained within a body of text.
* Identify the type of entity extracted, such as it being a person, place, or organization using Named Entity Recognition.
* Use Sentiment Analysis to identify the sentiment of a string of text, from very negative to neutral to very positive.
* Reduce words to their root, or stem, using PorterStemmer, or break up text into tokens using Tokenizer.

**Applications**

NLP algorithms are typically based on machine learning algorithms. Instead of hand-coding large sets of rules, NLP can rely on machine learning to automatically learn these rules by analyzing a set of examples (i.e. a large corpus, like a book, down to a collection of sentences), and making a statical inference. In general, the more data analyzed, the more accurate the model will be.

Social media analysis is a great example of NLP use. Brands track conversations online to understand what customers are saying, and glean insight into user behavior.

“One of the most compelling ways NLP offers valuable intelligence is by tracking sentiment — the tone of a written message (tweet, Facebook update, etc.) — and tag that text as positive, negative or neutral,” Rehling said.

Build your own social media monitoring tool

Start by using the algorithm Retrieve Tweets With Keyword to capture all mentions of your brand name on Twitter. In our case, we search for mentions of Algorithmia.

Then, pipe the results into the Sentiment Analysis algorithm, which will assign a sentiment rating from 0-4 for each string (Tweet).

Similarly, Facebook uses NLP to track trending topics and popular hashtags.

“Hashtags and topics are two different ways of grouping and participating in conversations,” Chris Struhar, a software engineer on News Feed, said in How Facebook Built Trending Topics With Natural Language Processing. “So don’t think Facebook won’t recognize a string as a topic without a hashtag in front of it. Rather, it’s all about NLP: natural language processing. Ain’t nothing natural about a hashtag, so Facebook instead parses strings and figures out which strings are referring to nodes — objects in the network. We look at the text, and we try to understand what that was about.”

It’s not just social media that can use NLP to it’s benefit. Publishers are hoping to use NLP to improve the quality of their online communities by leveraging technology to “auto-filter the offensive comments on news sites to save moderators from what can be an ‘exhausting process’,” Francis Tseng said in Prototype winner using ‘natural language processing’ to solve journalism’s commenting problem.

Other practical uses of NLP include monitoring for malicious digital attacks, such as phishing, or detecting when somebody is lying.

Use NLP to build your own RSS reader. You can build a machine learning RSS reader in less than 30-minutes using the follow algorithms:

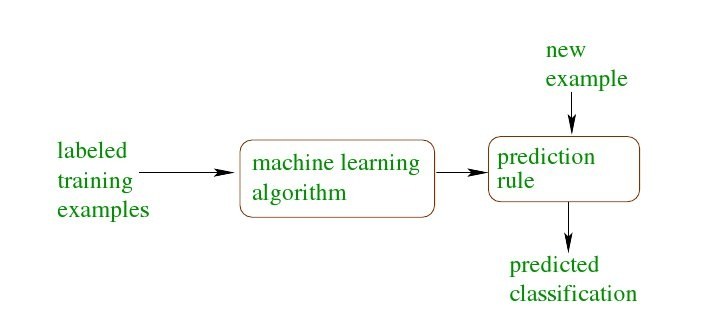
* ScrapeRSS to grab the title and content from an RSS feed.
* Html2Text to keep the important text, but strip all the HTML from the document.
* AutoTag uses Latent Dirichlet Allocation to identify relevant keywords from the text.
* Sentiment Analysis is then used to identify if the article is positive, negative, or neutral.
* Summarizer is finally used to identify the key sentences.

**ANALYSIS OF SENTIMENTS**

**Difference between text and sentiment**

Classification - You have set of predefined classes, and you want to classify new incoming item in one of those classes. There you would use any of classification technique from machine learning. Basically what you do here: You make your system learn using existing data for all classes you have. Actually it will consider the basic features for each class, matches those classes with incoming item. The class, for which maximum features get matched, new item get classified in that class.

Sentiment analysis: It is the technique for measuring the polarity of input text. How much positive or negative content the text has.



Oh! Iphone 6 is so decent mobile. That shows positive sentiment. Your idea is fine, but we don’t feel it is going to create any positive intact. That shows negative sentiment. Nowadays, machine learning techniques also being used for sentiment analysis.

**MACHINE LEARNING**

**Lexicon based Machine learning**

There are broadly two categories of sentiment analysis:

Lexical Methods: These techniques employ dictionaries of words annotated with their semantic polarity and sentiment strength. This is then used to calculate a score for the polarity and/or sentiment of the document. Usually this method gives high precision but low recall.

Machine Learning Methods: Such techniques require creating a model by training the classifier with labeled examples. This means that you must first gather a dataset with examples for positive, negative and neutral classes, extract the features from the examples and then train the algorithm based on the examples. These methods are used mainly for computing the polarity of the document.

Choice of the method heavily depends on the application, domain and language. Using lexicon based techniques with large dictionaries enables us to achieve very good results. Nevertheless they require using a lexicon, something which is not always available in all languages.

On the other hand Machine Learning based techniques deliver good results but they require obtaining training on labeled data.

AFINN Model: In the AFINN model, the authors have computed sentiment scores for a list of words. The sentiment of a tweet is computed based on the sentiment scores of the terms in the tweet. The sentiment of the tweet is defined to be equal to the sum of the sentiment scores for each term in the tweet. The AFINN-111 dictionary contains 2477 English words rated for valence with an integer value between -5 and 5. The words have been manually labelled by Finn Arup Neilsen in 2009-2010. Some of the words are the grammatically different versions of the same stem e.g. ‘favorite’ and ‘favorites’ are listed as two different words with different valence scores.

Naive Bayes Classifier: The Naive Bayes classifier can be trained on a corpus of labeled (+ve, -ve, neutral) tweets and then employed to assign polarity to a new tweet. The features used in this model are the words or bi-grams with their frequencies in the tweet strings. You may want to keep or remove URLs, emoticons and short tokens depending on the application.

Using the pre-estimated values of these probabilities, one can compute the probability of a tweet to be positive, negative and neutral. Whenever a new tweet is fed to the classifier, it will predict the polarity of the tweet based on the probability of its having that polarity. An implementation of Naive Bayes classifier for classifying spam and non-spam messages can be found here.

Several other methods in both the categories are prevalent today. Lots of companies using sentiment analysis employ lexical methods where they create dictionaries based on their trade algorithms and the domain of the application.

For machine learning based analysis, instead of Naive Bayes, one can use more sophisticated algorithms like SVMs.

**EXPERIMENTS**

**Framework**

The analysis has three parts –

1. The fetching of tweets using Twitter API based on PHP.

2. The operation on data using MATLAB with Lexicon Algorithm.

3. Addition of Machine learning in order to train the system.

* Twitter API based on PHP

API is application program interface used to bring data in the raw form from the database. The flexibility of API enables you to project data in whichever form you want depending upon its usage.

The content can modified with an altered frontend according to the required process.

* Data manipulation using MATLAB

The analysis is completely based on Lexicon Method of Natural language processing. The method scans the set of words that are taken from a standard data library. The MATLAB program enables the searching of the matching words using the word matching function seqwordcount. After creating a composite review the reports are generated on the basis of positive and negative sentiment in the form of pie charts.

* Machine learning using Training Set

The analyzed data is processed with multiple filters and the matching words from the library are being sent to training file dependent on the relevant topic. The training files are kept in order to provide a relevant data set instead of just generic data set. This filtering enables a much faster result for the reports rather than scanning the whole library.

**TWITTER API AND WORKING**

1. Calculation of Lexicon Theorem with Machine learning

The calculation of the theorem is based on the time required by the Lexicon theorem to analyze the data and provide a graph to present the result. The required result has to be then stored for machine learning so that the process can used for faster implementation.

For example, we can store the stock market data in a provided file and movie review data in a provided separate file. The changes in the file are updated once the program is run again.

Given below are the steps required to process the file -

STEPS FOR TWITTER SENTMENT ANALYSIS

1. Start xampp which is installed in your PC and start the services APACHE and MYSQL.

Xampp - XAMPP is a free and open source cross-platform web server solution stack package developed by Apache Friends,consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. XAMPP stands for Cross-Platform (X), Apache (A), MariaDB (M), PHP (P) and Perl (P). It is a simple, lightweight Apache distribution that makes it extremely easy for developers to create a local web server for testing and deployment purposes.

2. Open the link http://localhost/twitter\_api/

3. Complete the form with tweet requirements and submit.

We have developed a frontend for the twitter API , so that we can filter data accordingly and provide a .txt file to the MATLAB (Data mining software) as a source.

4. You’ll see the collected tweets.

The collected tweets are filtered from retweets in order to focus completely on the required data. The provided data can provide enough accuracy to store and compare the sentiments.

5. Run MATLAB and open the files training.m, run this file. This file will fetch the data from tweet.txt which is created earlier by Twitter API.

6. It will show the training graph when comparing with standard data libs.

7. It will take around 2-3 mins to run.

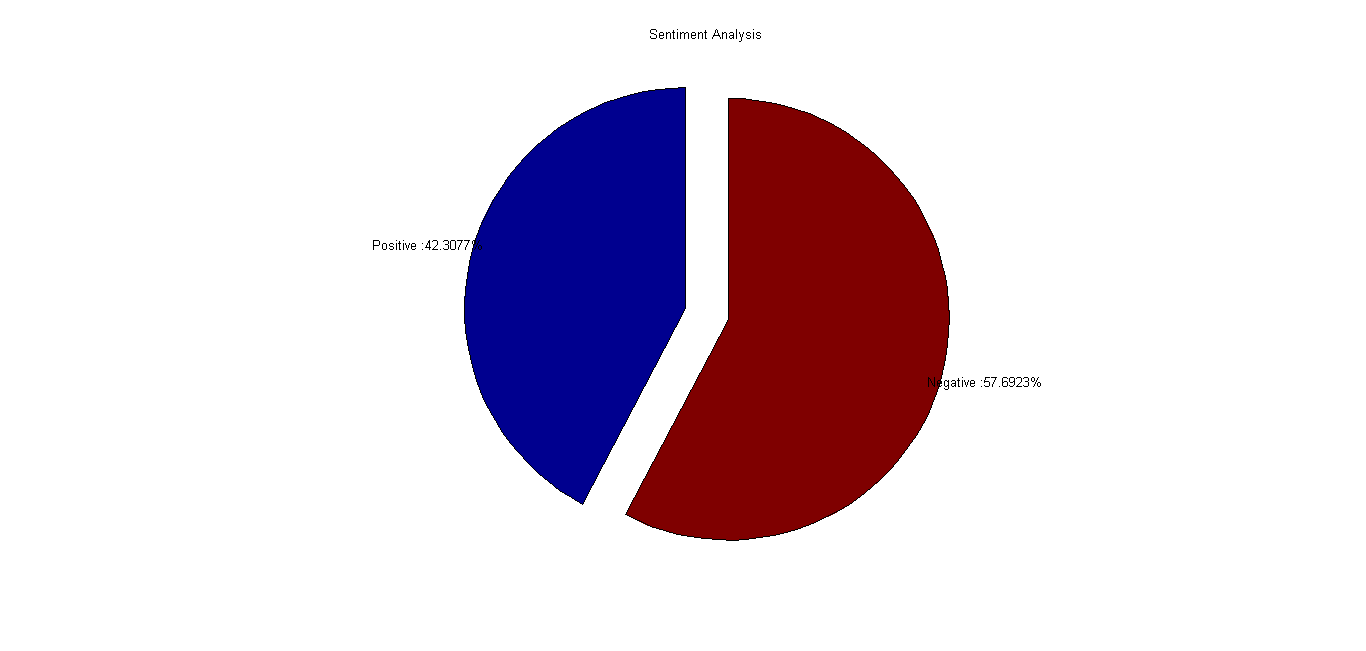
The training file will collect the tweets and provide an option to make the machine learn the sentiments comparing it from the data library provided as positive and negative words.

Once completed, the training file will store the text files and provide you a much more descriptive graph having a detailed analysis.

8. Open the file twitter\_apis.m in MATLAB and run it.

9. It will show the final graph.

The twitter\_apis file will finally be configured on the basis of subject of data and the time take for analysis will be much lesser as the comparison of words are much lesser.



Let us assume we have 10 tweets of each 10 words -

10 X 10 = 100 words to compare with 4782 negative words and 2005 positive words.

This comparison is done by lexicon method. But to decrease the repetition, we will add the repetitive words in the training set for a particular subject.

When training set is executed the execution is much faster as the comparison is only with less than 100 words combining both negative and positive words.

**CONCLUSION**

It will have a lot to do with social forums/platforms where people express free opinion. Presently tweets are one such open medium, then if Facebook at some point chooses to make the timeline updates/status messages open to search (I think it will someday do that through a minuscule sounding update in "privacy policy") it will be gold mine of real-time sentiments.

Present Sentiments hold a key to the future events. To make it sound a bit technical, you can say that the sentiments represent the "present value of future events". Now this value can have deep social, political and monetary significance. It can be "Expression of opinion about a public figure", "opinions expressed through tweets before elections", or "the buzz before a movie release", all these can be great cues for things to come.

Therefore when people comment about present news stories, the sentiment analysis can actually offer a key to predict the future outcomes or at least anticipate them better!

**APPENDIX A**

clc;

fileID = fopen('Z:/elecbits original/matlabp/tweet.txt');

c = textscan(fileID,'%s','Delimiter','|');

m = strtrim(c)

fclose(fileID);

n = size(m);

out = cellfun('prodofsize', m);

fileID\_neg = fopen('Z:/elecbits original/matlabp/negative-words.txt');

c\_neg = textscan(fileID\_neg,'%s','Delimiter','|');

m\_neg = strtrim(c\_neg)

fclose(fileID\_neg);

n\_neg = size(m\_neg);

out\_neg = cellfun('prodofsize', m\_neg);

fileID\_pos = fopen('Z:/elecbits original/matlabp/positive-words.txt');

c\_pos = textscan(fileID\_pos,'%s','Delimiter','|');

m\_pos = strtrim(c\_pos)

fclose(fileID\_pos);

n\_pos = size(m\_pos);

out\_pos = cellfun('prodofsize', m\_pos);

fidneg = fopen('Z:/elecbits original/matlabp/analysis/negative\_training.txt','wt');

yy = 0;

for i = 1:out

for j = 1 : out\_neg

f = seqwordcount(m{1}{i}, m\_neg{1}{j});

str\_neg=strcat('\n', m\_neg{1}{j});

if(f > 0 )

{

fprintf(fidneg, str\_neg);

}

end

yy = f + yy;

end

end

fclose(fidneg);

Negative\_emotions = yy;

display(Negative\_emotions);

fidpos = fopen('Z:/elecbits original/matlabp/analysis/positive\_training.txt','wt');

xx = 0;

for i = 1:out

for j = 1 : out\_pos

t = seqwordcount(m{1}{i}, m\_pos{1}{j});

str\_pos=strcat('\n', m\_pos{1}{j});

if(t > 0)

{

fprintf(fidpos, str\_pos);

}

end

xx = t + xx;

end

end

fclose(fidpos);

Positive\_emotions = xx;

display(Positive\_emotions);

F = [xx, yy]

total = sum(F);

percentage1= (xx/total)\* 100;

percentage2= (yy/total)\* 100 ;

s1 = strcat('Positive :', num2str(percentage1),'%');

s2 = strcat('Negative :', num2str(percentage2),'%');

labels = {s1,s2};

explode=[1,1];

percentage = [percentage1, percentage2];

pie(percentage,explode, labels);

title('Sentiment Analysis');

%display(m{1}{72});

%display(out);

%display(m\_neg{1}{3});

%display(out\_neg);

%display(m\_pos{1}{3});

%display(out\_pos);

% celldisp(m\_neg);

% celldisp(m\_pos);

clc;

fileID = fopen('Z:/elecbits original/matlabp/tweet.txt');

c = textscan(fileID,'%s','Delimiter','|');

m = strtrim(c)

fclose(fileID);

n = size(m);

out = cellfun('prodofsize', m);

fileID\_neg = fopen('Z:/elecbits original/matlabp/analysis/negative\_training.txt');

c\_neg = textscan(fileID\_neg,'%s','Delimiter','|');

m\_neg = strtrim(c\_neg)

fclose(fileID\_neg);

n\_neg = size(m\_neg);

out\_neg = cellfun('prodofsize', m\_neg);

fileID\_pos = fopen('Z:/elecbits original/matlabp/analysis/positive\_training.txt');

c\_pos = textscan(fileID\_pos,'%s','Delimiter','|');

m\_pos = strtrim(c\_pos)

fclose(fileID\_pos);

n\_pos = size(m\_pos);

out\_pos = cellfun('prodofsize', m\_pos);

yy = 0;

for i = 1:out

for j = 1 : out\_neg

f = seqwordcount(m{1}{i}, m\_neg{1}{j});

yy = f + yy;

end

end

Negative\_emotions = yy;

display(Negative\_emotions);

xx = 0;

for i = 1:out

for j = 1 : out\_pos

t = seqwordcount(m{1}{i}, m\_pos{1}{j});

xx = t + xx;

end

end

Positive\_emotions = xx;

display(Positive\_emotions);

F = [xx, yy]

total = sum(F);

percentage1= (xx/total)\* 100;

percentage2= (yy/total)\* 100 ;

s1 = strcat('Positive :', num2str(percentage1),'%');

s2 = strcat('Negative :', num2str(percentage2),'%');

labels = {s1,s2};

explode=[1,1];

percentage = [percentage1, percentage2];

pie(percentage,explode, labels);

title('Sentiment Analysis');

%display(m{1}{72});

%display(out);

%display(m\_neg{1}{3});

%display(out\_neg);

%display(m\_pos{1}{3});

%display(out\_pos);

% celldisp(m\_neg);

% celldisp(m\_pos);

**APPENDIX B**

**PHP SCRIPT**

<!DOCTYPE html>

<html lang="en">

<head>

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" type="text/css" href="form.css">

<!-- Website Font style -->

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/4.6.1/css/font-awesome.min.css">

<link rel="stylesheet" href="style.css">

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

<!-- Google Fonts -->

<link href='https://fonts.googleapis.com/css?family=Passion+One' rel='stylesheet' type='text/css'>

<link href='https://fonts.googleapis.com/css?family=Oxygen' rel='stylesheet' type='text/css'>

<title>Twitter API</title>

</head>

<body>

<div class="container">

<div class="row main">

<div class="main-login main-center">

<h5>Create Tweets.</h5>

<form class="" method="post" action="">

<div class="form-group">

<label for="name" class="cols-sm-2 control-label">Tag you want to search</label>

<div class="cols-sm-10">

<div class="input-group">

<span class="input-group-addon"><i class="fa fa-user fa" aria-hidden="true"></i></span>

<input type="text" class="form-control" name="tag" id="name" placeholder="#hashtag" required/>

</div>

</div>

</div>

<div class="form-group">

<label for="name" class="cols-sm-2 control-label">Tweet Count (Not more than 500)</label>

<div class="cols-sm-10">

<div class="input-group">

<span class="input-group-addon"><i class="fa fa-envelope fa" aria-hidden="true"></i></span>

<input type="number" class="form-control" name="tw\_count" id="email" placeholder="Count" required/>

</div>

</div>

</div>

<!-- <div class="form-group">

<label for="username" class="cols-sm-2 control-label">Username</label>

<div class="cols-sm-10">

<div class="input-group">

<span class="input-group-addon"><i class="fa fa-users fa" aria-hidden="true"></i></span>

<input type="text" class="form-control" name="username" id="username" placeholder="Enter your Username"/>

</div>

</div>

</div>

<div class="form-group">

<label for="password" class="cols-sm-2 control-label">Password</label>

<div class="cols-sm-10">

<div class="input-group">

<span class="input-group-addon"><i class="fa fa-lock fa-lg" aria-hidden="true"></i></span>

<input type="password" class="form-control" name="password" id="password" placeholder="Enter your Password"/>

</div>

</div>

</div>

<div class="form-group">

<label for="confirm" class="cols-sm-2 control-label">Confirm Password</label>

<div class="cols-sm-10">

<div class="input-group">

<span class="input-group-addon"><i class="fa fa-lock fa-lg" aria-hidden="true"></i></span>

<input type="password" class="form-control" name="confirm" id="confirm" placeholder="Confirm your Password"/>

</div>

</div>

</div> -->

<div class="form-group ">

<button type="submit" id="button" class="btn btn-info btn-lg" name="update">Submit</button>

</div>

</div>

</form>

</div>

</div>

</div>

<?php

require"twitteroauth/autoload.php";

use Abraham\TwitterOAuth\TwitterOAuth;

$show\_modal = false;

if (isset($\_POST['update'])) {

$hashtag = $\_POST['tag'];

$t\_count = $\_POST['tw\_count'];

$consumer\_key = 'vEQwA3v0giwOk1yHQ1K0te9Ky';

$consumer\_secret= 'ZTJa8KM5SlWqkMCiLokyhL2qaAckvqISBfBWqAsPOLeC6tRe4e';

$access\_token = '767306187511586816-u4zetTbkUHxKodveA4QApmeGK5JPeqL';

$access\_token\_secret = 'QyNSrqKpB9hte7x6TSDik6VE9PaPl40fCfbrIepf2HD7r';

//include library

//Connect to API

$connection = new TwitterOAuth($consumer\_key, $consumer\_secret, $access\_token, $access\_token\_secret);

$content = $connection -> get("account/verify\_credentials");

//$new\_status = $connection -> post("statuses/update", ["status"=>"chill"]);

$statuses = $connection -> get("statuses/user\_timeline", ["count"=>0,"exclude\_replies"=>true]);

$statuses1 = $connection->get("search/tweets", ["q" => $hashtag, "count"=> $t\_count]);

$var = print\_r($statuses1, true);

//echo substr\_count($var, '[text] => RT');

$text\_count = substr\_count($var, '[text] => RT');

//echo "<br>";

//echo substr\_count($var, '[truncated] =>');

$trun\_count = substr\_count($var, '[truncated] =>');

$original\_count = $trun\_count - $text\_count;

//echo "<br>";

//echo $original\_count;

$myfile = fopen("D:/saurav/elecbits original/matlabp/tweet.txt", "w") or die("Unable to open file!");

$myfile1 = fopen("tweet.txt", "w") or die("Unable to open file!");

for ($i=0; $i < $trun\_count ; $i++) {

$trun\_dist = explode("[truncated] =>", $var);

$text\_info\_count = substr\_count($trun\_dist[$i], '[text] =>');

// echo "<br>";

//echo $text\_info\_count;

// echo "<br>";

$tweets = explode("[text] => ", $trun\_dist[$i]);

// echo "<br>";

if (substr\_count($trun\_dist[$i], 'RT @') == 0) {

$tweets\_write = trim($tweets[$text\_info\_count]);

//echo $real\_value;

$delimiter = "|";

fwrite($myfile, ($tweets\_write.$delimiter));

fwrite($myfile1, ($tweets\_write.$delimiter));

}

}

if (fclose($myfile)) {

echo "<script>alert(' $original\_count original Tweets are collected , $text\_count Retweets are denied.')</script>";

echo "<script>window.open('tweet.txt', '\_blank', 'toolbar=yes, location=yes, status=yes, menubar=yes, scrollbars=yes')</script>";

}

/\*{

for ($i=0; $i < $trun\_count ; $i++) {

$text\_info\_count = substr\_count($trun\_dist[$i], '[text] => RT');

echo $text\_info\_count;

/\*for ($j=0; $j < $text\_info\_count ; $j++) {

$tweets = explode("[text] => ", $trun\_dist[$i]);

echo $tweets[$text\_info\_count];

}

}

}

\*/

}

?>

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->

</body>

</html>

**CSS**

/\*

/\*

\* Specific styles of signin, register, component

\*/

/\*

\* General styles

\*/

#playground-container {

height: 500px;

overflow: hidden !important;

-webkit-overflow-scrolling: touch;

}

body, html{

height: 100%;

background-repeat: no-repeat;

background:url(https://i.ytimg.com/vi/4kfXjatgeEU/maxresdefault.jpg);

font-family: 'Oxygen', sans-serif;

background-size: cover;

}

.main{

margin:50px 15px;

}

h1.title {

font-size: 50px;

font-family: 'Passion One', cursive;

font-weight: 400;

}

hr{

width: 10%;

color: #fff;

}

.form-group{

margin-bottom: 15px;

}

label{

margin-bottom: 15px;

}

input,

input::-webkit-input-placeholder {

font-size: 11px;

padding-top: 3px;

}

.main-login{

background-color: #fff;

/\* shadows and rounded borders \*/

-moz-border-radius: 2px;

-webkit-border-radius: 2px;

border-radius: 2px;

-moz-box-shadow: 0px 2px 2px rgba(0, 0, 0, 0.3);

-webkit-box-shadow: 0px 2px 2px rgba(0, 0, 0, 0.3);

box-shadow: 0px 2px 2px rgba(0, 0, 0, 0.3);

}

.form-control {

height: auto!important;

padding: 8px 12px !important;

}

.input-group {

-webkit-box-shadow: 0px 2px 5px 0px rgba(0,0,0,0.21)!important;

-moz-box-shadow: 0px 2px 5px 0px rgba(0,0,0,0.21)!important;

box-shadow: 0px 2px 5px 0px rgba(0,0,0,0.21)!important;

}

#button {

border: 1px solid #ccc;

margin-top: 28px;

padding: 6px 12px;

color: #666;

text-shadow: 0 1px #fff;

cursor: pointer;

-moz-border-radius: 3px 3px;

-webkit-border-radius: 3px 3px;

border-radius: 3px 3px;

-moz-box-shadow: 0 1px #fff inset, 0 1px #ddd;

-webkit-box-shadow: 0 1px #fff inset, 0 1px #ddd;

box-shadow: 0 1px #fff inset, 0 1px #ddd;

background: #f5f5f5;

background: -moz-linear-gradient(top, #f5f5f5 0%, #eeeeee 100%);

background: -webkit-gradient(linear, left top, left bottom, color-stop(0%, #f5f5f5), color-stop(100%, #eeeeee));

background: -webkit-linear-gradient(top, #f5f5f5 0%, #eeeeee 100%);

background: -o-linear-gradient(top, #f5f5f5 0%, #eeeeee 100%);

background: -ms-linear-gradient(top, #f5f5f5 0%, #eeeeee 100%);

background: linear-gradient(top, #f5f5f5 0%, #eeeeee 100%);

filter: progid:DXImageTransform.Microsoft.gradient(startColorstr='#f5f5f5', endColorstr='#eeeeee', GradientType=0);

}

.main-center{

margin-top: 30px;

margin: 0 auto;

max-width: 400px;

padding: 10px 40px;

background:#009edf;

color: #FFF;

text-shadow: none;

-webkit-box-shadow: 0px 3px 5px 0px rgba(0,0,0,0.31);

-moz-box-shadow: 0px 3px 5px 0px rgba(0,0,0,0.31);

box-shadow: 0px 3px 5px 0px rgba(0,0,0,0.31);

}

span.input-group-addon i {

color: #009edf;

font-size: 17px;

}

.login-button{

margin-top: 5px;

}

.login-register{

font-size: 11px;

text-align: center;

}

**TWITTEROAUTH**

<?php

/\*\*

\* The most popular PHP library for use with the Twitter OAuth REST API.

\*

\* @license MIT

\*/

namespace Abraham\TwitterOAuth;

use Abraham\TwitterOAuth\Util\JsonDecoder;

/\*\*

\* TwitterOAuth class for interacting with the Twitter API.

\*

\* @author Abraham Williams <abraham@abrah.am>

\*/

class TwitterOAuth extends Config

{

const API\_VERSION = '1.1';

const API\_HOST = 'https://api.twitter.com';

const UPLOAD\_HOST = 'https://upload.twitter.com';

const UPLOAD\_CHUNK = 40960; // 1024 \* 40

/\*\* @var Response details about the result of the last request \*/

private $response;

/\*\* @var string|null Application bearer token \*/

private $bearer;

/\*\* @var Consumer Twitter application details \*/

private $consumer;

/\*\* @var Token|null User access token details \*/

private $token;

/\*\* @var HmacSha1 OAuth 1 signature type used by Twitter \*/

private $signatureMethod;

/\*\*

\* Constructor

\*

\* @param string $consumerKey The Application Consumer Key

\* @param string $consumerSecret The Application Consumer Secret

\* @param string|null $oauthToken The Client Token (optional)

\* @param string|null $oauthTokenSecret The Client Token Secret (optional)

\*/

public function \_\_construct($consumerKey, $consumerSecret, $oauthToken = null, $oauthTokenSecret = null)

{

$this->resetLastResponse();

$this->signatureMethod = new HmacSha1();

$this->consumer = new Consumer($consumerKey, $consumerSecret);

if (!empty($oauthToken) && !empty($oauthTokenSecret)) {

$this->token = new Token($oauthToken, $oauthTokenSecret);

}

if (empty($oauthToken) && !empty($oauthTokenSecret)) {

$this->bearer = $oauthTokenSecret;

}

}

/\*\*

\* @param string $oauthToken

\* @param string $oauthTokenSecret

\*/

public function setOauthToken($oauthToken, $oauthTokenSecret)

{

$this->token = new Token($oauthToken, $oauthTokenSecret);

}

/\*\*

\* @return string|null

\*/

public function getLastApiPath()

{

return $this->response->getApiPath();

}

/\*\*

\* @return int

\*/

public function getLastHttpCode()

{

return $this->response->getHttpCode();

}

/\*\*

\* @return array

\*/

public function getLastXHeaders()

{

return $this->response->getXHeaders();

}

/\*\*

\* @return array|object|null

\*/

public function getLastBody()

{

return $this->response->getBody();

}

/\*\*

\* Resets the last response cache.

\*/

public function resetLastResponse()

{

$this->response = new Response();

}

/\*\*

\* Make URLs for user browser navigation.

\*

\* @param string $path

\* @param array $parameters

\*

\* @return string

\*/

public function url($path, array $parameters)

{

$this->resetLastResponse();

$this->response->setApiPath($path);

$query = http\_build\_query($parameters);

return sprintf('%s/%s?%s', self::API\_HOST, $path, $query);

}

/\*\*

\* Make /oauth/\* requests to the API.

\*

\* @param string $path

\* @param array $parameters

\*

\* @return array

\* @throws TwitterOAuthException

\*/

public function oauth($path, array $parameters = [])

{

$response = [];

$this->resetLastResponse();

$this->response->setApiPath($path);

$url = sprintf('%s/%s', self::API\_HOST, $path);

$result = $this->oAuthRequest($url, 'POST', $parameters);

if ($this->getLastHttpCode() != 200) {

throw new TwitterOAuthException($result);

}

parse\_str($result, $response);

$this->response->setBody($response);

return $response;

}

/\*\*

\* Make /oauth2/\* requests to the API.

\*

\* @param string $path

\* @param array $parameters

\*

\* @return array|object

\*/

public function oauth2($path, array $parameters = [])

{

$method = 'POST';

$this->resetLastResponse();

$this->response->setApiPath($path);

$url = sprintf('%s/%s', self::API\_HOST, $path);

$request = Request::fromConsumerAndToken($this->consumer, $this->token, $method, $url, $parameters);

$authorization = 'Authorization: Basic ' . $this->encodeAppAuthorization($this->consumer);

$result = $this->request($request->getNormalizedHttpUrl(), $method, $authorization, $parameters);

$response = JsonDecoder::decode($result, $this->decodeJsonAsArray);

$this->response->setBody($response);

return $response;

}

/\*\*

\* Make GET requests to the API.

\*

\* @param string $path

\* @param array $parameters

\*

\* @return array|object

\*/

public function get($path, array $parameters = [])

{

return $this->http('GET', self::API\_HOST, $path, $parameters);

}

/\*\*

\* Make POST requests to the API.

\*

\* @param string $path

\* @param array $parameters

\*

\* @return array|object

\*/

public function post($path, array $parameters = [])

{

return $this->http('POST', self::API\_HOST, $path, $parameters);

}

/\*\*

\* Make DELETE requests to the API.

\*

\* @param string $path

\* @param array $parameters

\*

\* @return array|object

\*/

public function delete($path, array $parameters = [])

{

return $this->http('DELETE', self::API\_HOST, $path, $parameters);

}

/\*\*

\* Make PUT requests to the API.

\*

\* @param string $path

\* @param array $parameters

\*

\* @return array|object

\*/

public function put($path, array $parameters = [])

{

return $this->http('PUT', self::API\_HOST, $path, $parameters);

}

/\*\*

\* Upload media to upload.twitter.com.

\*

\* @param string $path

\* @param array $parameters

\* @param boolean $chunked

\*

\* @return array|object

\*/

public function upload($path, array $parameters = [], $chunked = false)

{

if ($chunked) {

return $this->uploadMediaChunked($path, $parameters);

} else {

return $this->uploadMediaNotChunked($path, $parameters);

}

}

/\*\*

\* Private method to upload media (not chunked) to upload.twitter.com.

\*

\* @param string $path

\* @param array $parameters

\*

\* @return array|object

\*/

private function uploadMediaNotChunked($path, array $parameters)

{

$file = file\_get\_contents($parameters['media']);

$base = base64\_encode($file);

$parameters['media'] = $base;

return $this->http('POST', self::UPLOAD\_HOST, $path, $parameters);

}

/\*\*

\* Private method to upload media (chunked) to upload.twitter.com.

\*

\* @param string $path

\* @param array $parameters

\*

\* @return array|object

\*/

private function uploadMediaChunked($path, array $parameters)

{

$init = $this->http('POST', self::UPLOAD\_HOST, $path, $this->mediaInitParameters($parameters));

// Append

$segment\_index = 0;

$media = fopen($parameters['media'], 'rb');

while (!feof($media))

{

$this->http('POST', self::UPLOAD\_HOST, 'media/upload', [

'command' => 'APPEND',

'media\_id' => $init->media\_id\_string,

'segment\_index' => $segment\_index++,

'media\_data' => base64\_encode(fread($media, self::UPLOAD\_CHUNK))

]);

}

fclose($media);

// Finalize

$finalize = $this->http('POST', self::UPLOAD\_HOST, 'media/upload', [

'command' => 'FINALIZE',

'media\_id' => $init->media\_id\_string

]);

return $finalize;

}

/\*\*

\* Private method to get params for upload media chunked init.

\* Twitter docs: https://dev.twitter.com/rest/reference/post/media/upload-init.html

\*

\* @param array $parameters

\*

\* @return array

\*/

private function mediaInitParameters(array $parameters)

{

$return = [

'command' => 'INIT',

'media\_type' => $parameters['media\_type'],

'total\_bytes' => filesize($parameters['media'])

];

if (isset($parameters['additional\_owners'])) {

$return['additional\_owners'] = $parameters['additional\_owners'];

}

if (isset($parameters['media\_category'])) {

$return['media\_category'] = $parameters['media\_category'];

}

return $return;

}

/\*\*

\* @param string $method

\* @param string $host

\* @param string $path

\* @param array $parameters

\*

\* @return array|object

\*/

private function http($method, $host, $path, array $parameters)

{

$this->resetLastResponse();

$url = sprintf('%s/%s/%s.json', $host, self::API\_VERSION, $path);

$this->response->setApiPath($path);

$result = $this->oAuthRequest($url, $method, $parameters);

$response = JsonDecoder::decode($result, $this->decodeJsonAsArray);

$this->response->setBody($response);

return $response;

}

/\*\*

\* Format and sign an OAuth / API request

\*

\* @param string $url

\* @param string $method

\* @param array $parameters

\*

\* @return string

\* @throws TwitterOAuthException

\*/

private function oAuthRequest($url, $method, array $parameters)

{

$request = Request::fromConsumerAndToken($this->consumer, $this->token, $method, $url, $parameters);

if (array\_key\_exists('oauth\_callback', $parameters)) {

// Twitter doesn't like oauth\_callback as a parameter.

unset($parameters['oauth\_callback']);

}

if ($this->bearer === null) {

$request->signRequest($this->signatureMethod, $this->consumer, $this->token);

$authorization = $request->toHeader();

if (array\_key\_exists('oauth\_verifier', $parameters)) {

// Twitter doesn't always work with oauth in the body and in the header

// and it's already included in the $authorization header

unset($parameters['oauth\_verifier']);

}

} else {

$authorization = 'Authorization: Bearer ' . $this->bearer;

}

return $this->request($request->getNormalizedHttpUrl(), $method, $authorization, $parameters);

}

/\*\*

\* Make an HTTP request

\*

\* @param string $url

\* @param string $method

\* @param string $authorization

\* @param array $postfields

\*

\* @return string

\* @throws TwitterOAuthException

\*/

private function request($url, $method, $authorization, array $postfields)

{

/\* Curl settings \*/

$options = [

// CURLOPT\_VERBOSE => true,

CURLOPT\_CAINFO => \_\_DIR\_\_ . DIRECTORY\_SEPARATOR . 'cacert.pem',

CURLOPT\_CONNECTTIMEOUT => $this->connectionTimeout,

CURLOPT\_HEADER => true,

CURLOPT\_HTTPHEADER => ['Accept: application/json', $authorization, 'Expect:'],

CURLOPT\_RETURNTRANSFER => true,

CURLOPT\_SSL\_VERIFYHOST => 2,

CURLOPT\_SSL\_VERIFYPEER => true,

CURLOPT\_TIMEOUT => $this->timeout,

CURLOPT\_URL => $url,

CURLOPT\_USERAGENT => $this->userAgent,

];

/\* Remove CACert file when in a PHAR file. \*/

if ($this->pharRunning()) {

unset($options[CURLOPT\_CAINFO]);

}

if($this->gzipEncoding) {

$options[CURLOPT\_ENCODING] = 'gzip';

}

if (!empty($this->proxy)) {

$options[CURLOPT\_PROXY] = $this->proxy['CURLOPT\_PROXY'];

$options[CURLOPT\_PROXYUSERPWD] = $this->proxy['CURLOPT\_PROXYUSERPWD'];

$options[CURLOPT\_PROXYPORT] = $this->proxy['CURLOPT\_PROXYPORT'];

$options[CURLOPT\_PROXYAUTH] = CURLAUTH\_BASIC;

$options[CURLOPT\_PROXYTYPE] = CURLPROXY\_HTTP;

}

switch ($method) {

case 'GET':

break;

case 'POST':

$options[CURLOPT\_POST] = true;

$options[CURLOPT\_POSTFIELDS] = Util::buildHttpQuery($postfields);

break;

case 'DELETE':

$options[CURLOPT\_CUSTOMREQUEST] = 'DELETE';

break;

case 'PUT':

$options[CURLOPT\_CUSTOMREQUEST] = 'PUT';

break;

}

if (in\_array($method, ['GET', 'PUT', 'DELETE']) && !empty($postfields)) {

$options[CURLOPT\_URL] .= '?' . Util::buildHttpQuery($postfields);

}

$curlHandle = curl\_init();

curl\_setopt\_array($curlHandle, $options);

$response = curl\_exec($curlHandle);

// Throw exceptions on cURL errors.

if (curl\_errno($curlHandle) > 0) {

throw new TwitterOAuthException(curl\_error($curlHandle), curl\_errno($curlHandle));

}

$this->response->setHttpCode(curl\_getinfo($curlHandle, CURLINFO\_HTTP\_CODE));

$parts = explode("\r\n\r\n", $response);

$responseBody = array\_pop($parts);

$responseHeader = array\_pop($parts);

$this->response->setHeaders($this->parseHeaders($responseHeader));

curl\_close($curlHandle);

return $responseBody;

}

/\*\*

\* Get the header info to store.

\*

\* @param string $header

\*

\* @return array

\*/

private function parseHeaders($header)

{

$headers = [];

foreach (explode("\r\n", $header) as $line) {

if (strpos($line, ':') !== false) {

list ($key, $value) = explode(': ', $line);

$key = str\_replace('-', '\_', strtolower($key));

$headers[$key] = trim($value);

}

}

return $headers;

}

/\*\*

\* Encode application authorization header with base64.

\*

\* @param Consumer $consumer

\*

\* @return string

\*/

private function encodeAppAuthorization(Consumer $consumer)

{

$key = rawurlencode($consumer->key);

$secret = rawurlencode($consumer->secret);

return base64\_encode($key . ':' . $secret);

}

/\*\*

\* Is the code running from a Phar module.

\*

\* @return boolean

\*/

private function pharRunning()

{

return class\_exists('Phar') && \Phar::running(false) !== '';

}

}

**APPENDIX C**

**Test Sample for connectivity**

<?php

session\_start();

require\_once("twitteroauth/autoload.php"); //Path to twitteroauth library

$twitteruser = "elecbits16";

$notweets = 30;

$consumerkey = "vEQwA3v0giwOk1yHQ1K0te9Ky";

$consumersecret = "ZTJa8KM5SlWqkMCiLokyhL2qaAckvqISBfBWqAsPOLeC6tRe4e";

$accesstoken = "767306187511586816-u4zetTbkUHxKodveA4QApmeGK5JPeqL";

$accesstokensecret = "QyNSrqKpB9hte7x6TSDik6VE9PaPl40fCfbrIepf2HD7r";

use Abraham\TwitterOAuth\TwitterOAuth;

//Connect to API

$connection = new TwitterOAuth($consumer\_key, $consumer\_secret, $access\_token, $access\_token\_secret);

$content = $connection -> get("account/verify\_credentials");

//$new\_status = $connection -> post("statuses/update", ["status"=>"chill"]);

$statuses = $connection -> get("statuses/user\_timeline", ["count"=>0,"exclude\_replies"=>true]);

$statuses1 = $connection->get("search/tweets", ["q" => "engvaus", "count"=> 20]);

print\_r($statuses1);

?>

**REFERENCES**

**Web URLs:**

* Sentiment lexicon [(http://www.cs.pitt.edu/mpqa/]((http:/www.cs.pitt.edu/mpqa/)) 8221 words scored for polarity (positive or negative), subjectivity. Distinguishes between POS tags.
* General inquirer (<http://www.wjh.harvard.edu/~inquirer/homecat.htm>) has several dictionaries, e.g., a "positive" list with 1'915 words and one 'negative' list with 2'291 words
* Affective word list <http://sentic.net/senticnet-4.0.zip>
* <http://www.mitpressjournals.org/doi/abs/10.1162/COLI_a_00049>
* <http://www.nowpublishers.com/article/Details/INR-011>
* <https://github.com/jeffreybreen/twitter-sentiment-analysis-tutorial-201107.git>
* <http://www.mitpressjournals.org/doi/pdfplus/10.1162/COLI_a_00049>
* <https://www.researchgate.net/profile/Fabrizio_Sebastiani/publication/220746537_SentiWordNet_30_An_Enhanced_Lexical_Resource_for_Sentiment_Analysis_and_Opinion_Mining/links/545fbcc40cf27487b450aa21.pdf>
* <http://software.ucv.ro/~cmihaescu/ro/teaching/AIR/docs/Lab4-NaiveBayes.pdf>
* <http://ieeexplore.ieee.org/abstract/document/6726818/?reload=true>
* <https://www.mathworks.com/videos/news-sentiment-analysis-using-matlab-and-ravenpack-117953.html>
* <http://tomx.inf.elte.hu/twiki/pub/Tudas_Labor/2012Summer/GoelMittal-StockMarketPredictionUsingTwitterSentimentAnalysis.pdf>
* <http://ieeexplore.ieee.org/abstract/document/6726818/>