**EmoTweets**

A Project Presented to

The Faculty of the School of IT and Computing

De La Salle University – Science & Technology Complex

In Partial Fulfillment

of the Requirements for the Degree of

Bachelor of Science in Computer Science

by

Blanquera, Kimberly D.

Professor Ma. Christine Gendrano

July <#>, 2015

1. **Company Background - Mike**
   1. **Brief History**
   2. **Products/Services**
   3. **Market**
   4. **Organizational Structure (including IT Department)**
2. **Project Background**
   1. **Current Situation**

Twitter has drawn millions of users who use their services, eventually creating a large pool of data. Users, 134 million and growing, send and post messages, called *tweets*, at a rate of 1382% in the year 2008 alone. Twitter data has been used to gather news, customer views about various brands or products, and public opinion regarding politicians. This shows the tangibility of applying natural language processing to Twitter data to gain insight on human emotions. Increased usage and future technology can render services like Twitter as a source of better understanding of the human condition: opinions, feelings and lives.

* 1. **Project Description**

EmoTweets performs sentiment analysis on Tweets and classifies a particular Tweet as either positive or negative. The program fetches Tweets with the hashtag that a user inputs. The Tweets are then tokenized and processed against an existing data set, or *vocabulary*, using the Naïve Bayes algorithm. Based on calculated probabilities, Tweets are then given a corresponding class.

* 1. **Significance**
     1. **Academe - Mike**
     2. **Student - Mike**
     3. **Society**

Text and sentiment analysis, specifically on data from social media, benefits all aspects of society. It can aid the government gauge what citizens are happy or unhappy about with regard to their programs and way of governance. These concepts may also be applied to searching Twitter, and other social media services online, for alarming news to alert others, crisis preparation and prevention as well as shed light on issues that have roused strong emotions from people.

* 1. **Scope and limitation**

The application can only process a maximum amount of 1,500 Tweets from the past 7 days at most because of the limitation posed by the Twitter Search API. Only Tweets in the English language can be classified by the application.

In classification, emoticons are not used as attributes. Internet jargon and lingo, however, are included in the vocabulary and affect the outcome.

The analysis relies on the lexical qualities of the text as opposed to a contextual understanding of the Tweets as a whole. Because of this, the application may have some difficulties classifying Tweets that contain numerous instances of negation.

* 1. **Architecture/framework - Mike**
  2. **Student’s role - Mike**

1. **Methodology** 
   1. **Phases/Activities**
      1. **Planning**

First, the specifications and requirements of the project were discussed. The expected requirements upon completion were defined. Based on this, a list of tasks to be accomplished was produced. Tasks were prioritized and assigned to each group member. The estimated duration of each task or milestone was also decided.

* + 1. **Research**

The first two weeks were spent conducting research on the concept of Natural Language Processing and sentiment analysis as well as tools and libraries that may be used in implementation.

In researching Natural Language Processing and sentiment analysis, course material available online were used. Online journals and papers were also sought after to further aid with implementation using machine learning algorithms. Existing libraries and API were also investigated to gain knowledge of methods that can be used during development. The Twitter Search API was especially focused on as it is the main source of the data to be processed. By the end of this phase, the machine learning algorithms and tools to use was determined.

* + 1. **Implementation**

Implementation began connecting to the Twitter Search API which involved acquiring proper authorization from Twitter and creating the base for the search and retrieve functions. The user interface was also built. Then, focus shifted to preparing the dataset for processing. First, a balanced sample set was created using RapidMiner. The dataset is composed of 1,600,000 instances of Tweets that have been annotated with a sentiment of *positive* or *negative*. A randomly selected but balanced sample set of 5,000 instances was extracted, wherein 2,500 instances were annotated as *positive* and 2,500 instances were annotated as *negative.*

Cleaning the data involved removing certain elements of Tweets. Twitter allows users to *mention* other users in their Tweets. This is included in the body of the Tweet and is represented by the *@* character followed a username. Hashtags are also embedded in the body text, denoted by the *#* appended before the hashtag text. Both elements were removed from the data along with smileys or emoticons, hyperlinks, special characters and numbers.

The *vocabulary* was created from the cleaned data. First, each Tweet is fed to a tokenizer, which divides the Tweet into the words, or *tokens*, that it is composed of. Second, the tokens are tallied. Each token has a *count* value which signifies the number of times a particular word has appeared in a *positive* Tweet and another *count* value to represent the number of times the word has occurred in a *negative* Tweet. A local database was set up to store the sample set and the vocabulary. Data was then stored and retrieved through a combination of SQL and PHP.

The Naïve-Bayes module was constructed while the training set of Tweets was being processed for the *vocabulary****.*** After the construction of both, they were integrated to form the classification function that outputs the sentiment based on the probability calculated by the Naïve-Bayes module. These functions were then joined with the Twitter Search API and the user interface to facilitate the connection between modules.

Throughout implementation, GitHub was used for version control and collaboration between developers.

* + 1. **Testing - Kim**
  1. **Inputs**

The only input of the application is the hashtag keyword, inclusive of the # character, by which it searches Twitter for Tweets.

* 1. **Tools used**

The Twitter API was used to search Twitter for Tweets by hashtag and to retrieve them. TwitterOUath, a PHP library for Twitter license authentication, connected to the API. Tweets were saved on a database stored locally using XAMPP. To perform Natural Language Processing tasks through PHP, the Nlp-tools API was utilized. Nlp-tools facilitated the use of tokenizers in PHP implementation. The pattern-en API, written in Python and integrated with PHP, allowed the checking of noun singularity or plurality which is a function needed in both the processing of the data set as well as for new instances.

* 1. **Outputs**

The application outputs the resulting Tweets from the search, each labeled with its respective class. A chart is also displayed with statistical information on the results, including the amount of Tweets per class.

* 1. **Problems encountered and solutions undertaken - Mike**

1. **Technical Documentation (includes diagrams – where applicable, such as ERDs, DFDs, architecture, business model, manuals, etc.)**
   1. **Architecture - Mike**
   2. **Database Design – Kim**

tweets

Fig. # tweets table

The *tweets* table, illustrated by Fig. #, stores annotated Tweets from the sample set. *tweetID* is a unique integer assigned to each Tweet. *Tweet* is the text body. *Sentiment* is the annotation from the dataset and may have the value of either *positive* or *negative.*

vocab

Fig. # vocab table

The *vocab* table shown in Fig. # represents the *vocabulary* formed after tokenizing tweets and tallying the occurrences of words in Tweets that were annotated with a particular sentiment. *word* signifies the token generated and is of type varchar. *positiveCount* and *negativeCount* are both of type integer. Their values represent the number of times that a word has appeared in positively annotated Tweets and negatively annotated Tweets respectively.

* 1. **Software Requirements**

The following are required to successfully run the application: Python 2.5 or 2.6, pattern-en, XAMPP, web browser and internet connection.

* 1. **Manual**
     1. **Setup and Installation**

Place the main folder, *emotweets*, in the following location

/XAMPP/htdocs

Open XAMPP and start *Apache* and *SQL.* When both of their statuses are *Running*, open a browser. In the URL bar, type *localhost*

* + 1. **Usage**

Open a new tab or new browser window and enter the following in the URL bar:

localhost/emotweets/index.php

Enter a *hashtag* to search for and press enter or click the *Go* button. The tweets retrieved for the searched hashtag will appear on the right with their corresponding classification while a chart showing some statistics of the classification results will be displayed on the left.