**Sentiment analysis of social network project**

This project is based on social network analytics. Mainly this will focus on sentiment analytics, trends and other interesting analytical outcomes of Tweets by Twitter users. The social network being used is Twitter, Tweets will be retrieved from Twitter using an API call. This API will be made using Twitter4J which is Java based code used to make calls to Twitter in order to retrieve the tweets based on some set criteria. The tweets to be retrieved will be based on specific hashtags and/or text present in the tweet. This data will then be stored in a database and used as the basis of the analytics to be carried out. There are three main components to this project these are as follows;

1. Java code using Twitter4J to carry out API calls to Twitter.
2. Database storage and management to store retrieved tweets and a collection of search phrases and sentiment phrases.
3. Data analytics and the production of graphical representation of the outcomes of the analytics.

Twitter4J is a Java based API program. It will be used, developed and written to meet the needs of this project and to carry out the main task of retrieving the data needed. The API results will be processed, formatted, split and copied to a table in a database which will be explained in more detail below.

The database will consist of a minimum of 3 normalised tables (although this is likely to be more in order to store additional information relating to the data); there will be a table which will contain all the search text needed to filter down which tweets to retrieve (these may be hashtags, keywords or phrases). This data could be anything from a topical current affair in the news or a trend that is currently occurring on Twitter/ the internet. Another table will contain all the sentiment phrases that will be used to determine the nature of each tweet, for example this table may contain words such as hate, love, like, dislike, terrible, horrible, brilliant, great etc. along with whether these words are negative, positive or neutral. The final core table will be used to store the tweets that have been retrieved by the API call and this table will be updated each time the API call is used.

The database will be stored locally on the host computer (the computer that is carrying out the API calls and analytics) and in the event that the database gets too big some data will be compressed and archived. If it is found that the database is still unmanageable the database/ data will be stored on an external hard drive with manageable chunks being accessed/ downloaded to the host computer at any one time.

The database, API and Java code will be used together as follows; the table of search text will be looped through with each record in the table altering the API call code to change which tweets will be returned. If this proves to be too intensive or time consuming the work load will be split across multiple threads and the GPU of the computer being used to increase performance with each thread taking one record from the table. The results returned by the API call will be processed by Java code and then written to the respective database table using SQL which will be embedded within a Java class allowing for ease of access and method calling.

The sentiment table will then be looped through, in the same fashion as the retrieval of tweets above, and compared to the results table where the data retrieved by the API will be stored. Each tweet/ record will then be assigned an overwhelming positive, negative or neutral feeling. These will be determined by the word(s) which the data contains in correspondence with the sentiment dictionary table.

This new dataset will then be used to carry out analytics. The analytics may include, but is not limited to, determining the sentiment of each text phrases and how this changed over time, showing the number of tweets over time containing certain text phrases, showing the most used/ unused emotional word in tweets from those present in the table and how these have changed over time, the number of emotional words used in tweets and the trend of this number relating to the overall sentiment of the tweet. All of the analysis, where appropriate, will be represented in both figures and a graphical representation. In order to do this tools such as TweetStats and Rapid Miner may be used. TweetStats can produce histograms displaying information such as number of retweets (and most common retweeting users), number of replies (and most common replying users), tweet density and interfaces used (Instagram, Twitter for iPhone etc.). The data science tool Rapid Miner can be used for providing graphical outputs of the analysed data.

This project will be carried out under an agile workflow with a series of sprints being planned out in advance and progress evaluated a minimum of twice a week. The results of this will cause planned work to be adjusted as necessary for upcoming sprints. All work will be version controlled and stored on GitHub within a repository under a secure SSH key and all sensitive information (such as twitter login information) to be sure to be omitted in the git ignore file. The Git structure will consist of a Master branch, a feature branch and a test branch. Work will be done on feature branches and tested on the test branch. The test branch will contain the current contents of the Master branch. Once testing has passed this test branch will merged to the Master branch and the feature branch will be deleted. This is to maintain the integrity of the work and to allow back outs and code reverts to be easy while also keeping track of progress of work done.

The findings of the analytics will be briefly written up into documentation with some explanation to what was done and background information.

A stretch goal will be to implement a predictive model to predict outcomes of certain events, such political votes, based on the Twitter users feeling toward each party/ candidate. If this is completed an even further goal is to use the Facebook API on Facebook statuses and carry out the same process as Twitter and compare the results.