**Opinion Mining**: A Technique to find opinion in the tweets.

Data Mining Project Report – Spring 2011

**1.Introduction:**

Growth in Innovations and inventions in technology are introducing many ways of communications between individuals. Social Networking sites play an important role in every individual schedule. Data mining techniques used in such sites are introducing many new interesting patterns to the world. In this Project I would like to introduce an idea of an interesting pattern which can be used in social networking sites.

My main aim in the project would be evaluating the conversations (Tweets) in the Twitter and determining the opinion of the tweets.

**Problem Definition:**

In this project I would like to generate a cluster of positive opinion tweets and negative opinion tweets from the given twitter dataset.

For example let us consider a tweet:” *I love apple*”.

The computer may not understand and determine the opinion of the tweet, but I would like to implement an algorithm to find the opinion of each tweet. In the above tweet the keyword ‘love’ is a positive keyword based on the analysis. So my project will determine that the above tweet has a positive opinion in it.

Initially in the given dataset, the tweets consist of many missing values and unwanted symbols. Removing all this missing values and unwanted symbols is my preprocessing step (Data cleaning). In social networking sites Users communicates using different languages, different words such as “hmmmmm , gd mng, etc.”. Finding the opinion of such tweets is really a challenging task. Finding the opinion of each tweet and dividing them into positive opinion and negative opinion tweets is my clustering process.

I have used twitter dataset to train and test my algorithm. I have taken this dataset from DMML lab.

The URL of twitter dataset that I have used is <http://dmml.asu.edu/datasets/>

**2. Literature Review:**

I have gone through the paper **AI and Opinion Mining.** This Paper address the issue of opinion mining. The URL to this page is stated below.

http://doi.ieeecomputersociety.org/10.1109/MIS.2010.75

**3. Approach:**

The main goal of my project is to get meaningful groups, for this the clusters should capture the natural structure of the data. The step by step methodology to solve this project is given as:

**Training Algorithm:**

**Step 1**: Initially I retrieve the records of twitter dataset and store each record into an array.

**Step 2:** In the twitter dataset there are many columns which are not necessary for finding the opinion of the tweets such as (date and time, transaction number, etc.).

So In this step I have applied the **preprocessing step** (Data cleaning) to remove unwanted columns and unwanted symbols in each record of an array and stored the final set of tweets in a new array.

**Step 3:** In this step I have divided each tweet or record of an array into set of keywords by using “ “ delimiter and stored each record keywords into an multidimensional array into a specified row.

**Online Process:**

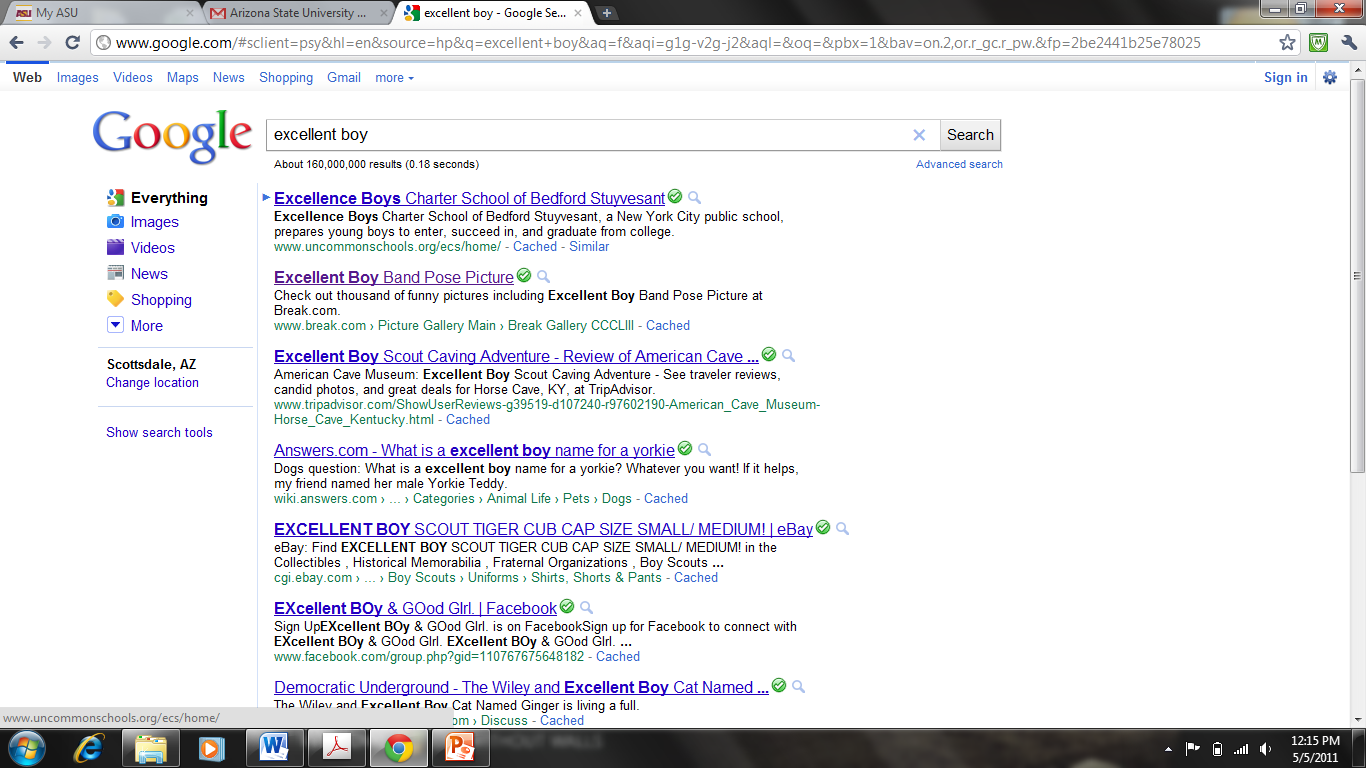
**Step 4:** The main idea of this project resides in this step. I will use each keyword of a record or tweet and query **Google** with two combinations (Excellent and poor).

Excellent Tweet[recordi][keywordj]

For example: **Google search**

**Google search**

Poor Tweet[recordi][keywordj]



Now In the above example the tweet[record][keyword] is “boy” and we have got 160,000,000 results that are found for the query “excellent boy”.

So for each keyword pair (i.e. Excellent Tweet[recordi][keywordj], Poor Tweet[recordi][keywordj] ) I will store the results count in an array record in two different columns and subtract the results count by using mathematical functions and store the result in next column in the same array record.

The results table can be shown diagrammatically as:

|  |  |  |  |
| --- | --- | --- | --- |
| **keyword** | **Excellent(a)** | **Poor(b)** | **Result(a-b)** |
| boy | 160,000,000 | 2,720,000 | 157280000 |
| girl | 179,000,000 | 5,860,000 | 173140000 |
| love | 331,000,000 | 25,700,000 | 305300000 |

**Step 5:** Now we have an array of keywords and its corresponding value. I will store the contents of array into an External file. This file can be used for testing purpose.

**Testing algorithm:**

In the above steps I have described about training the algorithm and now I would like to inform about testing the algorithm

**Step 6:** Now I will select the testing data set, retrieve the tweets from the file by applying step 1, step 2 and step 3 as above.

**Step 7:** Now I will retrieve the file generated by the step 5 and store the contents of file in the table format in an array as shown above.

**Step 8:** Now I consider each tweet generated from step 6, compare each keyword in a tweet with the keywords in the table generated by step 7. If I come across any keyword match I will assign the corresponding keyword value in table to the keyword in tweet.

**Step 9:** Now I sum the value of all keywords in tweet, if the value of tweet is greater than 0 then I send the tweet to **positive cluster** else I will send the tweet to **negative cluster**.

In this way I can get the positive and negative cluster tweets.

**Algorithm used:**

I have taken the concept of **K-means and** **Agglomerative Clustering Algorithm** in order to divide the tweets into clusters. In the initial step I have considered two clusters (positive and negative) where the value of **k=2**. Now I considered all the tweets as **individual** clusters. Now I have evaluated each cluster and assigned to the positive or negative cluster until only one cluster (or k clusters) left. In this way I have obtained the desired set of results.

**Preprocessing Step:**

As mentioned in the step 2, I have used Preprocessing steps like data cleaning. I have used this step to remove many unwanted symbols, missing values, redundant information in the tweets.

**Coding strategy:**

I have used java programming language to design the entire algorithm.

**4. Results:**

The results of this project is 2 clusters (positive and negative opinion clusters) which consist of set of positive and negative tweets respectively.

This project evaluation mainly consists of 2 parts online and offline process.

**Online process:**

The accuracy of online process is **100%** as I retrieve the results from Google.

The main **limitation** of online process is its time complexity:

Because for each keyword in a tweet, the algorithm will query Google for 2 times (excellent and poor), If we have ‘n’ keywords algorithm will have 2n searches. This takes a drastic amount of time for processing. For example if we consider 1000 tweets with 10 keywords in each tweet. The algorithm has to hit the Google for 20,000 times. Which is usually a large count.

The online process time complexity also depends on internet speed.

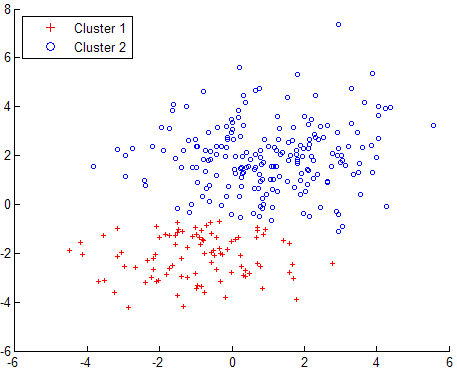
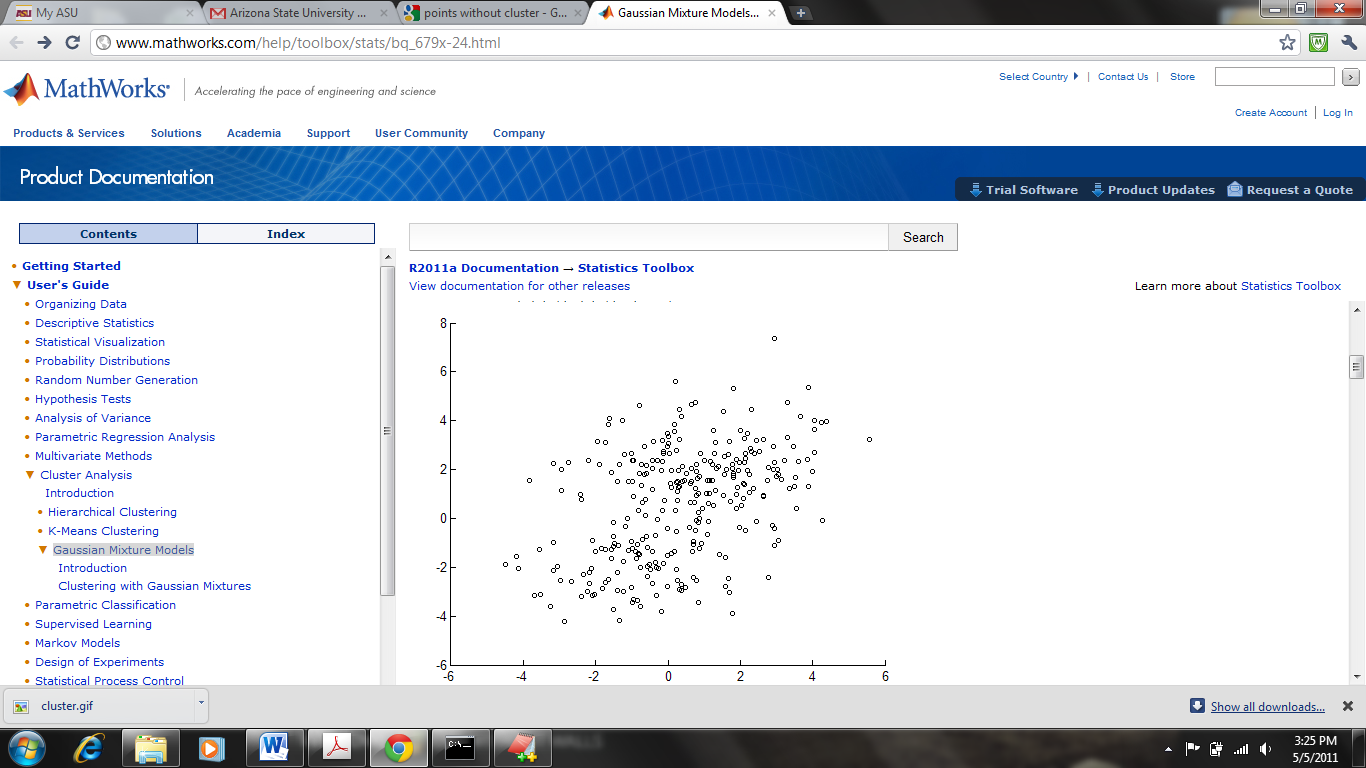
Though the time complexity of algorithm is high, It gives us an idea about how to handle opinion mining in tweets.

**Offline Process:**

This step includes dividing the tweets into clusters. Which can be handled by keyword matches, so the accuracy and efficiency of this step is **100%** which I have experienced by the testing data.

**Graph:**

Before clustering After Clustering



Negative tweets cluster

Positive tweets cluster

**5. Challenges:**

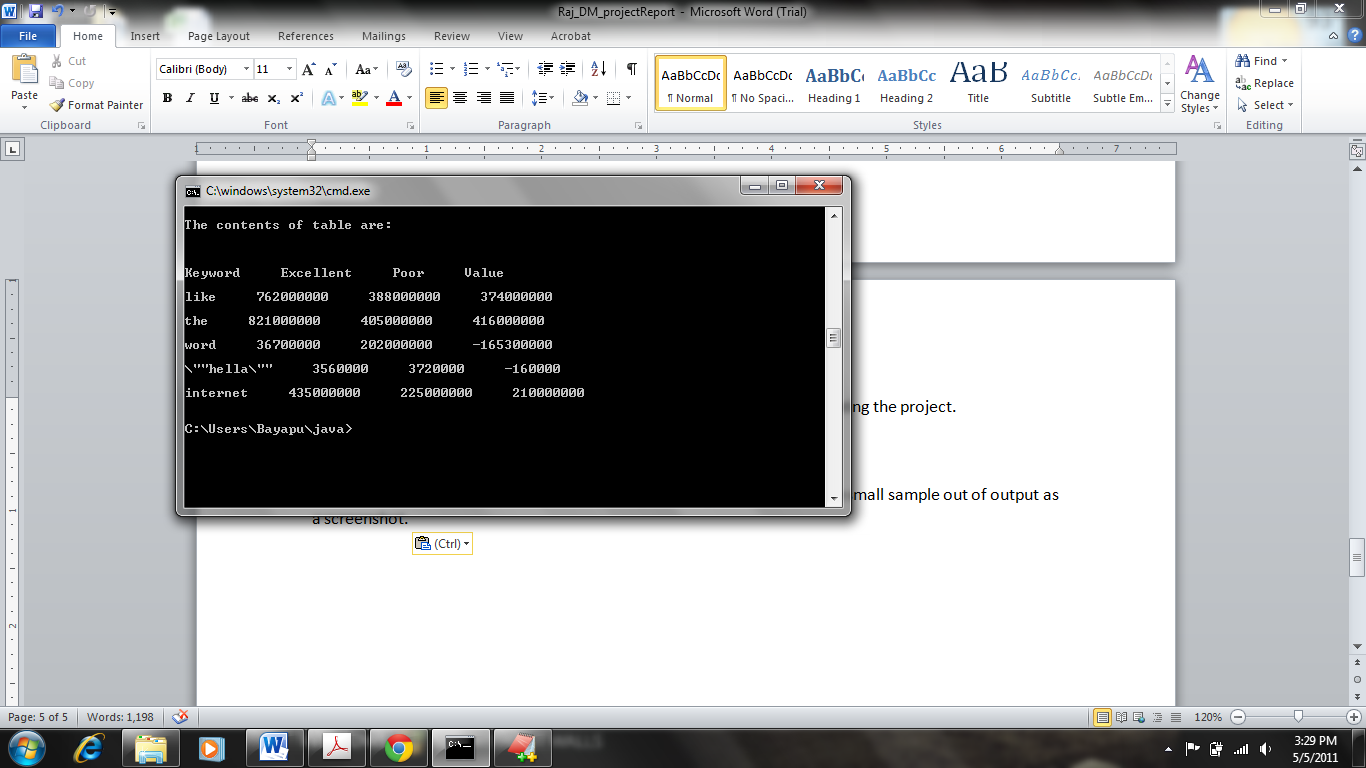
The main challenge that I have encountered during the project development is maintaining the time complexity and accuracy of this algorithm.

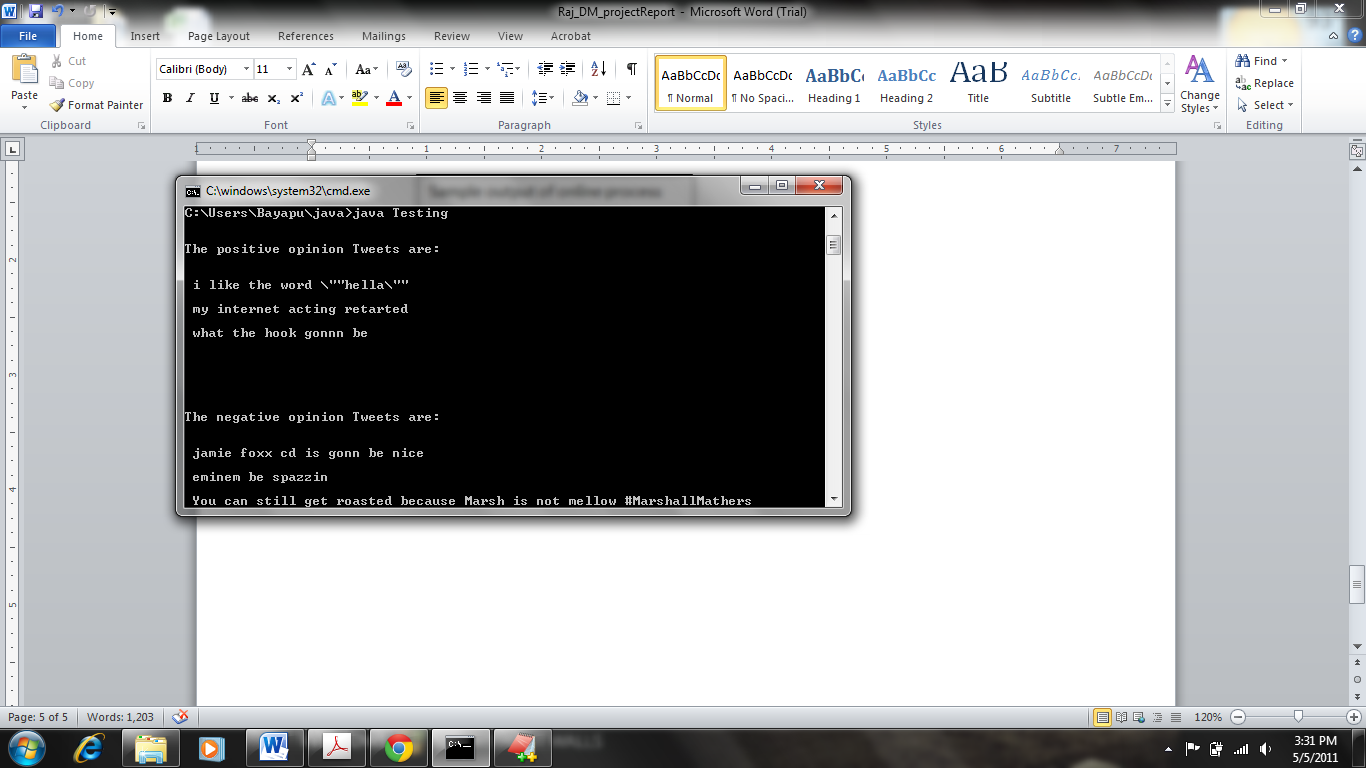
Next challenge would be handling large dataset for developing and evaluating the project.

**6. Screenshots:**

I have designed the project using the java programming. So I can show the small sample out of output as a screenshot.

Sample output of online process





Sample output of offline process

**Conclusion:**

I can conclude by saying that this project will identify the opinion of tweet. I have taken the concept of opinion mining in order to approach this problem. This project divides the list of tweets automatically in to positive and negative opinion tweets. Which is nothing but the concept of clustering.