## **Emotion Analysis using Opinion Mining**

## **B.Tech. Minor Project Report**

**SUBMITTED TO: Mr. Rahul Gupta**

## **BY:**

**VIKRANT DABAS | SHUBHAM VARSHNEY | PRAKHAR DOGRA | VIKAS**

**(2K12/CO/140) | (2K12/CO/126) | (2K12/CO/086) | (2K12/CO/139)**



## **DEPARTMENT OF COMPUTER ENGINEERING**

## **DELHI TECHNOLOGICAL UNIVERSITY**

**SHAHBAD DAULATPUR, MAIN BAWANA ROAD, DELHI – 110042, INDIA**

**DECEMBER, 2014**

# http://dce.edu/Web/images/ImageLogo.png **CERTIFICATE**

We hereby certify that the work which is being presented in this B.Tech. Minor Project Report entitled **“ANALYZING AND COMPARING MACHINE LEARNING ALGORITHMS TO PREDICT WINE QUALITY”,** in partial fulfilment of the requirements for the award of the **Bachelor of Technology in Computer Engineering** and submitted to the Department of Computer Engineering of Delhi Technological University is an authentic record of our own work carried out during a period from August 2014 to November 2014 under the supervision of **Mr. Manoj Sethi,** **Computer Engineering Department**.

The matter presented in this report has not been submitted by us for the award of any other degree elsewhere.

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

#### Date: Mr. Rahul Gupta

#### (Project Supervisor)

**I.) Our Goal**

The goal of our project is to do emotion analysis on tweets using opinion mining. The main idea of the project is to create a model that is able to predict the type of emotion of a tweet using web mining to initially extract tweets using the Twitter API and then further using sentiment analysis to analyse the emotions. The model is made using two modules that are named as follows:

1. Data cleaning module

2. Scoring (emotion value) module

This model is followed by another module that is used for creation and further expansion of an independent dictionary that stores the emotion values of adjectives, adverbs and verbs that are not identified by the original dictionary.

**II.) Introduction**

**Opinion mining** (also known as **Sentiment analysis**) refers to the use of [natural language processing](http://en.wikipedia.org/wiki/Natural_language_processing), [text analysis](http://en.wikipedia.org/wiki/Text_analytics) and [computational linguistics](http://en.wikipedia.org/wiki/Computational_linguistics) to identify and extract subjective information in source materials.

Generally speaking, sentiment analysis aims to determine the attitude of a speaker or a writer with respect to some topic or the overall contextual polarity of a document. The attitude may be his or her judgment or evaluation (see [appraisal theory](http://en.wikipedia.org/wiki/Appraisal_theory)), affective state (that is to say, the emotional state of the author when writing), or the intended emotional communication (that is to say, the emotional effect the author wishes to have on the reader).

Existing approaches to sentiment analysis can be grouped into four main categories: keyword spotting, lexical affinity, statistical methods, and concept-level techniques.[[15]](http://en.wikipedia.org/wiki/Sentiment_analysis#cite_note-IEEEIS2013-15) Keyword spotting classifies text by affect categories based on the presence of unambiguous affect words such as happy, sad, afraid, and bored.Lexical affinity not only detects obvious affect words, it also assigns arbitrary words a probable “affinity” to particular emotions.[[17]](http://en.wikipedia.org/wiki/Sentiment_analysis#cite_note-Stevenson-17) Statistical methods leverage on elements from [machine learning](http://en.wikipedia.org/wiki/Machine_learning) such as [latent semantic analysis](http://en.wikipedia.org/wiki/Latent_semantic_analysis), [support vector machines](http://en.wikipedia.org/wiki/Support_vector_machines), "[bag of words](http://en.wikipedia.org/wiki/Bag_of_words)" and *Semantic Orientation — Pointwise Mutual Information*(See Peter Turney's [[1]](http://en.wikipedia.org/wiki/Sentiment_analysis#cite_note-Turney02-1) work in this area). More sophisticated methods try to detect the holder of a sentiment (i.e. the person who maintains that affective state) and the target (i.e. the entity about which the affect is felt).[[18]](http://en.wikipedia.org/wiki/Sentiment_analysis#cite_note-Kim.2BHovy06-18) To mine the opinion in context and get the feature which has been opinionated, the grammatical relationships of words are used. Grammatical dependency relations are obtained by deep parsing of the text.[[19]](http://en.wikipedia.org/wiki/Sentiment_analysis#cite_note-Dey.2BHaque08-19) Unlike purely syntactical techniques, concept-level approaches leverage on elements from [knowledge representation](http://en.wikipedia.org/wiki/Knowledge_representation) such as [ontologies](http://en.wikipedia.org/wiki/Ontologies) and [semantic networks](http://en.wikipedia.org/wiki/Semantic_networks) and, hence, are also able to detect semantics that are expressed in a subtle manner, e.g., through the analysis of concepts that do not explicitly convey relevant information, but which are implicitly linked to other concepts that do so.

We have used two techniques out of the above 4 stated, namely Keyword spotting and Lexical Affinity.

The problem is that most sentiment analysis algorithms use simple terms to express sentiment about a product or service. However, cultural factors, linguistic nuances and differing contexts make it extremely difficult to turn a string of written text into a simple pro or con sentiment.[[31]](http://en.wikipedia.org/wiki/Sentiment_analysis#cite_note-Mining_the_Web_for_Feelings.2C_Not_Facts-31) The fact that humans often disagree on the sentiment of text illustrates how big a task it is for computers to get this right. The shorter the string of text, the harder it becomes.

Even though short text strings might be a problem, sentiment analysis within [microblogging](http://en.wikipedia.org/wiki/Microblogging) has shown that [Twitter](http://en.wikipedia.org/wiki/Twitter) can be seen as a valid offline indicator of political sentiment. Tweets’ political sentiment demonstrates close correspondence to parties’ and politicians’ political positions, indicating that the content of Twitter messages plausibly reflects the offline political landscape.

**III.) Why Twitter?**

How would you define Twitter?

There are many ways to answer this question, but let’s consider it from an overarching

angle that addresses some fundamental aspects of our shared humanity that any technology

needs to account for in order to be useful and successful. After all, the purpose

of technology is to enhance our human experience.

As humans, what are some things that we want that technology might help us to get?

• We want to be heard.

• We want to satisfy our curiosity.

• We want it easy.

• We want it now.

In the context of the current discussion, these are just a few observations that are generally

true of humanity. We have a deeply rooted need to share our ideas and experiences,

which gives us the ability to connect with other people, to be heard, and to feel a sense

of worth and importance. We are curious about the world around us and how to organize

and manipulate it, and we use communication to share our observations, ask questions,

and engage with other people in meaningful dialogues about our quandaries.

The last two bullet points highlight our inherent intolerance to friction. Ideally, we don’t

want to have to work any harder than is absolutely necessary to satisfy our curiosity or

get any particular job done; we’d rather be doing “something else” or moving on to the

next thing because our time on this planet is so precious and short. Along similar lines,

we want things now and tend to be impatient when actual progress doesn’t happen at

the speed of our own thought.

One way to describe Twitter is as a microblogging service that allows people to communicate

with short, 140-character messages that roughly correspond to thoughts or

ideas. In that regard, you could think of Twitter as being akin to a free, high-speed,

global text-messaging service. In other words, it’s a glorified piece of valuable infrastructure

that enables rapid and easy communication. However, that’s not all of the story.

It doesn’t adequately address our inherent curiosity and the value proposition that

emerges when you have over 500 million curious people registered, with over 100 million

of them actively engaging their curiosity on a regular monthly basis.

Besides the macro-level possibilities for marketing and advertising—which are always

lucrative with a user base of that size—it’s the underlying network dynamics that created

the gravity for such a user base to emerge that are truly interesting, and that’s why Twitter

is all the rage. While the communication bus that enables users to share short quips at

the speed of thought may be a necessary condition for viral adoption and sustained

engagement on the Twitter platform, it’s not a sufficient condition. The extra ingredient

that makes it sufficient is that Twitter’s asymmetric following model satisfies our curiosity.

It is the asymmetric following model that casts Twitter as more of an interest graph

than a social network, and the APIs that provide just enough of a framework for structure

and self-organizing behavior to emerge from the chaos.

In other words, whereas some social websites like Facebook and LinkedIn require the

mutual acceptance of a connection between users (which usually implies a real-world

connection of some kind), Twitter’s relationship model allows you to keep up with the

latest happenings of any other user, even though that other user may not choose to

follow you back or even know that you exist. Twitter’s following model is simple but

exploits a fundamental aspect of what makes us human: our curiosity. Whether it be an

infatuation with celebrity gossip, an urge to keep up with a favorite sports team, a keen

interest in a particular political topic, or a desire to connect with someone new, Twitter

provides you with boundless opportunities to satisfy your curiosity.

For example, the @HomerJSimpson account is the official account for Homer Simpson,

a popular character from The Simpsons television show. Although Homer Simpson isn’t

a real person, he’s a well-known personality throughout the world, and the @Homer‐

JSimpson Twitter persona acts as an conduit for him (or his creators, actually) to engage

his fans. Likewise, although this book will probably never reach the popularity of Homer

Simpson, @SocialWebMining is its official Twitter account and provides a means for a

community that’s interested in its content to connect and engage on various levels. When

you realize that Twitter enables you to create, connect, and explore a community of

interest for an arbitrary topic of interest, the power of Twitter and the insights you can

gain from mining its data become much more obvious.

There is very little governance of what a Twitter account can be aside from the badges

on some accounts that identify celebrities and public figures as “verified accounts” and

basic restrictions in Twitter’s Terms of Service agreement, which is required for using

the service. It may seem very subtle, but it’s an important distinction from some social

websites in which accounts must correspond to real, living people, businesses, or entities

of a similar nature that fit into a particular taxonomy. Twitter places no particular restrictions

on the persona of an account and relies on self-organizing behavior such as

following relationships and folksonomies that emerge from the use of hashtags to create

a certain kind of order within the system.

**IV.) Architecture**

We now present the architecture of the system proposed for our emotion analysis.It mainly involves:

○Tweet retrieval Module

○Preprocessing (cleaning) Module

○POS Tagging

○Emotion Scoring Module

○Dictionary Expansion Module



**V.) Twitter Retrieval Module**

Twitter has taken great care to craft an elegantly simple RESTful API that is intuitive

and easy to use. Even so, there are great libraries available to further mitigate the work

involved in making API requests. A particularly beautiful Python package that wraps

the Twitter API and mimics the public API semantics almost one-to-one is twitter.

Like most other Python packages, you can install it with pip by typing pip install

twitter in a terminal.

Before you can make any API requests to Twitter, you’ll need to create an application

at https://dev.twitter.com/apps. Creating an application is the standard way for developers

to gain API access and for Twitter to monitor and interact with third-party platform

developers as needed. The process for creating an application is pretty standard,

and all that’s needed is read-only access to the API.

In the present context, you are creating an app that you are going to authorize to access

your account data, so this might seem a bit roundabout; why not just plug in your

username and password to access the API? While that approach might work fine for

you, a third party such as a friend or colleague probably wouldn’t feel comfortable forking

over a username/password combination in order to enjoy the same insights from

your app. Giving up credentials is never a sound practice. Fortunately, some smart people

recognized this problem years ago, and now there’s a standardized protocol called

OAuth (short for Open Authorization) that works for these kinds of situations in a

generalized way for the broader social web. The protocol is a social web standard at this

point.

If you remember nothing else from this tangent, just remember that OAuth is a means

of allowing users to authorize third-party applications to access their account data

without needing to share sensitive information like a password. Appendix B provides

a slightly broader overview of how OAuth works if you’re interested, and Twitter’s

OAuth documentation offers specific details about its particular implementation.1

For simplicity of development, the key pieces of information that you’ll need to take

away from your newly created application’s settings are its consumer key, consumer

secret, access token, and access token secret. In tandem, these four credentials provide

everything that an application would ultimately be getting to authorize itself through a

series of redirects involving the user granting authorization, so treat them with the same

sensitivity that you would a password.

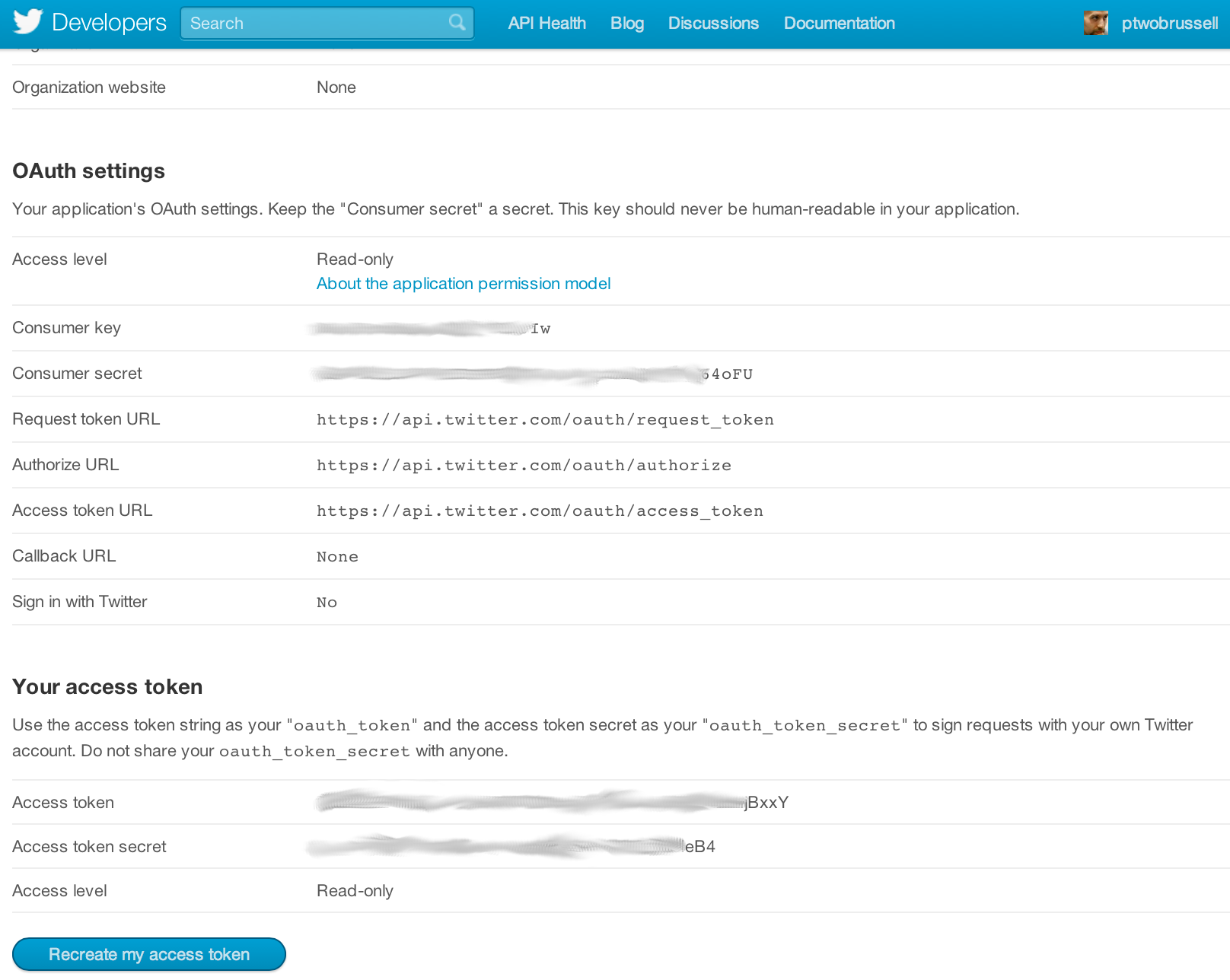


Figure 1-2. Create a new Twitter application to get OAuth credentials and API access

at https://dev.twitter.com/apps; the four (blurred) OAuth fields are what you’ll use to

make API calls to Twitter’s API

Without further ado, let’s create an authenticated connection to Twitter’s API and find

out what people are talking about by inspecting the trends available to us through the

GET trends/place resource. While you’re at it, go ahead and bookmark the official API

documentation as well as the REST API v1.1 resources, because you’ll be referencing

them regularly as you learn the ropes of the developer-facing side of the Twitterverse.

If we want to get large publically available Twitter datasets is through their API.

●We access the API through the tweepy python package

●The Twitter API has two different flavors: *RESTful and Streaming*.

❖The RESTful API is useful for getting things like lists of followers and those who follow a particular user, and is what most Twitter clients are built off of.

❖The Streaming API works by making a request for a specific type of data — filtered by keyword, user, geographic area, or a random sample — and then keeping the connection open as long as there are no errors in the connection.

**VI.) Cleaning Module**

The following strategy has been followed to make the downloaded twitter data tidy

○For Removal of #tags, a substring removal python function is used.

○Similar to the removal of hashtags, substring removal function is used to remove the links in the tweets.

○Each punctuation is replaced by an empty substring

○Removal of strings generally present in tweets such as RT, @someUser etc. through the substring removal function.

○Removal of non-English tweets by encoding them into ASCII and if encoding produces any exception, then reject them , otherwise, select the tweets.

○Removal of stopping words through nltk library.

●The substring removal function removes the part of the string that contains a substring eg. if substring = 'http' , then http://www.google.com is removed, that means, remove until a space is found.

●Substring removal function has been very useful in removing the links, as well as retweets, hashtags and some other undesired content from the tweets.

●NLTK is a leading platform for building Python programs to work with human language data.

●It provides easy-to-use suite of text processing libraries for tokenization and tagging.

●NLTK has been used to remove the stopping words from all the tweets.

●Words are fed to the Part of Speech tagger after Sentence Tokenizer and Word Tokenizer is applied.

●Only adjectives, adverbs and verbs have been retained.All other words are discarded.

**VII.) Scoring Module**

**Proposed Scoring Modules**

We have proposed several scoring methods/formulas to calculate the average emotion value of tweets of a handle(username) or hashtag:

1.Emotion value = sum of adjective values + sum of adverb and verb values

(no. of adjectives \* 5) + no. of adverb and verb

2. Multiply the value of adverb/verb with the upcoming adjective . If two verb /adverb are next to each other, simply multiply them. And then all these products are added. Further divided by 5\*number of adjectives encountered.

3. If the value of adverb/verb is less than 0 i.e, negative , then for the upcoming adjective, subtract its value from 5 instead of multiplying . And if value of verb/adverb is positive and >= 0.5 then multiply it with the upcoming adjective else multiply 0.5 with upcoming adjective. Later these products are added and the sum is divided by 5\*number of adjectives encountered.

4. Since the emotions are independent of each other, we should treat them independently and use a different formula for each emotion. We have used normalising values of different emotions. These predefined values are used to normalize the emotion values. In this method an assumption is taken that if an emotion is strong it’s respective value is > 3 (out of a scale of 5).

***Scoring Module used***

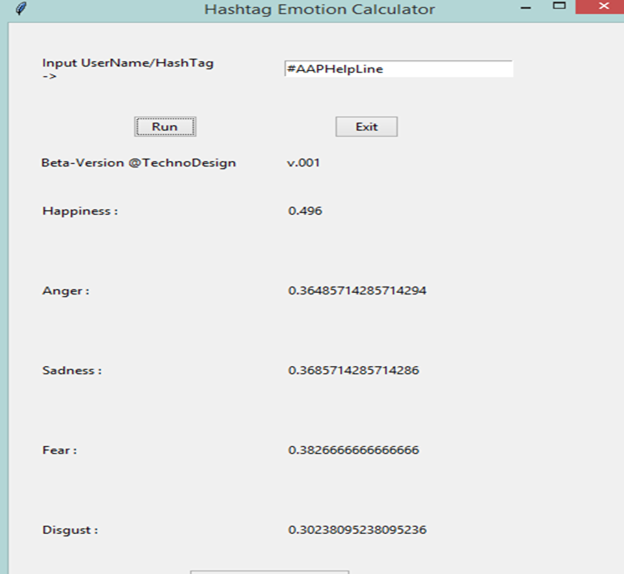
●This algorithm is based on the idea that whenever there is an adverb/verb in front of an adjective there is bound to be an effect on the emotion expressed by the same adjective. It can be either negative or negative. Keeping this in mind we have created method to calculate the emotion values.

●If the value of adverb/verb is less than 0 i.e, negative , then for the upcoming adjective, subtract its value from 5 instead of multiplying . And if value of verb/adverb is positive and >= 0.5 then multiply it with the upcoming adjective else multiply 0.5 with upcoming adjective. Later these products are added and the sum is divided by 5\*number of adjectives encountered.

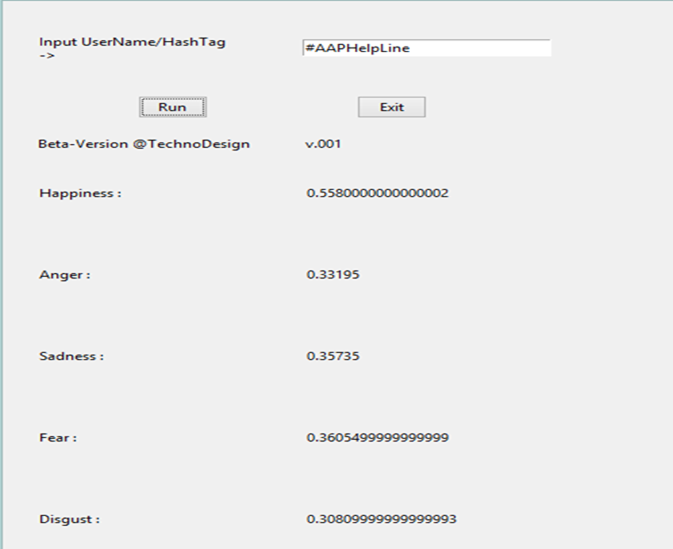
●

●And if there are more than 1 adverbs/verbs before an adjective then they are multiplied together.

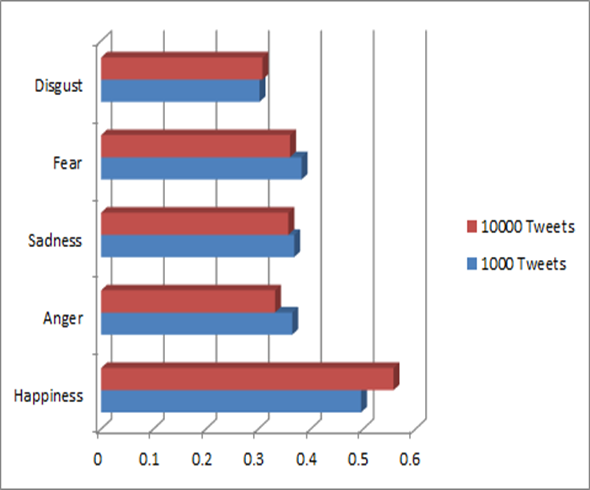
Emotion Values when 1000 tweets are downloaded from #AAPHelpLine

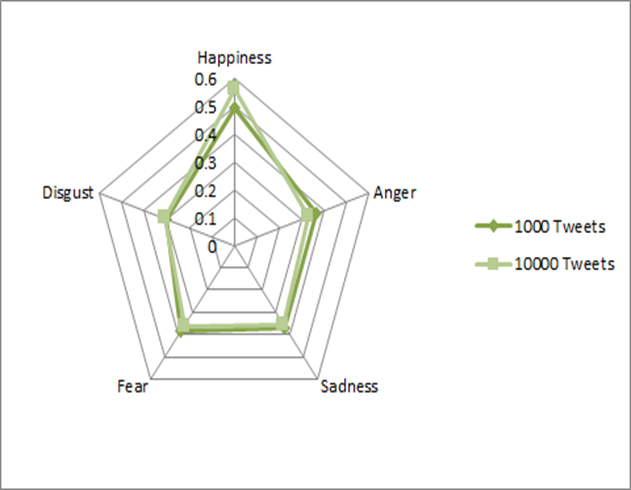


Emotion Values when 10000 tweets are downloaded from #AAPHelpLine



**Bar Graph and spider chart of emotion values**





**Emotion values obtained after running scoring module**

#KKRvMI

[0.556704761904762, 0.3601809523809524, 0.36324761904761915, 0.39180317460317454, 0.28949206349206347]

#HatsOffGeneral

[0.5512065573770494, 0.34857377049180316, 0.3657344262295083, 0.3835344262295082, 0.3112327868852459]

#AAPBreakUp

[0.4824, 0.4449333333333334, 0.44599999999999984, 0.49839999999999995, 0.39799999999999996]

#AAPKaSting

[0.517384, 0.420964, 0.390636, 0.4327319999999999, 0.37956399999999996]

#AAPWAR

[0.7349411764705883, 0.28094117647058825, 0.2952941176470588, 0.30388235294117644, 0.24988235294117644]

**VII.) Dictionary Expansion Module**

●The main idea of this module is to create a dictionary apart from the original dictionary being used.

●This module will be run after the scoring module for a hashtag or user-handle has been run.

●The emotion values from the scoring modules will be fed into this module and those values will be assigned to words (adjective/verb/adverb) that haven’t been identified by the original dictionary. Now this module is separate from the model and uses emotion values of every iteration of the model to expand it’s dictionary and update already used words.

● This module is designed to input emotion values of a hashtag or a user-handle. But it can be modified to process the tweets one by one.

**VIII.) Softwares and Tools used**

Programming Language used: Python 2.7.9 and 3.4.3

Libraries used: csv, nltk, tweepy, twython, Tkinter

Tools used: Twitter API, PyCharm 4.0.4

Operating Systems used: Windows 7/8/10, Ubuntu, Fedora

**IX.) Conclusions**

1. The problem can be classified into two categories
2. Regression Problem: The maximum accuracy achieved was 59.13 using gradient descent (after normalising the features) and quadratic form of the hypothesis function and the mean percentage error was 7.58%.
3. Classification Problem: The maximum accuracy achieved was 70.37% using logistic gradient descent with regularization in the case of binary classification (two classes –bad and good) and the maximum accuracy was 74.61 using gradient descent in the case of multiclass classification (four classes- bad, average, good, excellent).
4. In terms of accuracy, dividing the quality of wine into four groups and then applying logistic gradient descent seems to be the best algorithm for predicting the quality of wine.

**X.) Applications**

**XI.) Future work/additions**

Following is the list of functionalities/modules that haven’t been added in the model:

1.Spell Correction module: This module is required to convert the shorthand notations that usually used in tweets into proper words so as to make them visible to POS\_tag function. This module has to be implemented before the usage of POS\_tagging.

2.Emoticon processor: Emoticons are heavily used in tweets but since they are non-english words so they are removed in the cleaning module. So they need to be identified before that.

3.Apriori Association: We haven’t used any association rule between adverb/verb-adjective pair (or triplet). This method will provide a method that shows how much the adverb/verb affects the following adjective.

4.kNN Classifier: This classifier has to be used after the scoring module. When we encounter a tweet from the same hashtag we will classify using the kNN classification method into one of the pre-defined classes.

**References**