

SVKM’s NMIMS, MPSTME, Shirpur Campus

INTERIM REPORT

INTERIM REPORT

PROJECT: *ALIKE ASIDE*

Faculty mentor: Submitted By:

MR. SURAJ PATIL ANUPREKSHA JAIN –N232

(Assistant Professor) KRATI JAIN – N234

MPSTME, NMIMS SARVI JAIN – N237

Shirpur Campus Course: MBA TECH CSE

Batch: 2017-2022

A report submitted in partial fulfilment of the requirements of 5 Years Integrated MBA (Tech) Program of Mukesh Patel School of Technology Management & Engineering, NMIMS.

ABSTRACT

Sentiment analysis is a type of natural language processing for tracking the mood of the public about a particular product or topic. Sentiment analysis, which is also called opinion mining, involves building a system to collect and examine opinions about the product made in blog posts, comments, reviews or tweets.

This project addresses the problem of sentiment analysis in twitter; that is classifying tweets according to the sentiment expressed in them: positive, negative or neutral. Twitter is an online micro-blogging and social-networking platform which allows users to write short status updates. Project addresses the problem of sentiment analysis in twitter; that is classifying tweets according to the sentiment expressed in them: positive, negative or neutral. Twitter is an online micro-blogging and social-networking platform which allows users to write short status updates of maximum length 140 characters. It is a rapidly expanding service with over 200 million registered users - out of which 100 million are active users and half of them log on twitter on a daily basis - generating nearly 250 million tweets per day . Due to this large amount of usage we hope to achieve a reflection of public sentiment by analysing the sentiments expressed in the tweets. Analysing the public sentiment is important for many applications such as firms trying to find out the response of their products in the market, predicting political elections and predicting socioeconomic phenomena like stock exchange. The aim of this project is to develop a functional classifier for accurate and automatic sentiment classification of an unknown tweet stream.

INTRODUCTION

We have chosen to work with twitter since we feel it is a better approximation of public sentiment as opposed to conventional internet articles and web blogs. The reason is that the amount of relevant data is much larger for twitter, as compared to traditional blogging sites. Moreover the response on twitter is more prompt and also more general (since the number of users who tweet is substantially more than those who write web blogs on a daily basis).

Sentiment analysis of the public is highly critical in macro-scale socioeconomic phenomena like predicting the stock market rate of a particular firm. This could be done by analysing overall public sentiment towards that firm with respect to time and using economic tools for finding the correlation between public sentiment and the firm’s stock market value.

Firms can also estimate how well their product is responding in the market, which areas of the market it is having a favourable response and in which a negative response (since twitter allows us to download a stream of geo-tagged tweets for particular locations. If firms can get this information they can analyze the reasons behind geographically differentiated responses, and so they can market their product in a more optimized manner by looking for appropriate solutions like creating suitable market segments. Predicting the results of popular political elections and polls is also an emerging application to sentiment analysis.

DETAILED OUTLINE

Alike Aside will be developed using the python language, machine learning based. We will implement machine learning using Naive Bayes Algorithm.

The Naive Bayes algorithm is a Machine learning algorithm for classification problems. It is primarily used for text classification, which involves high dimensional training datasets. A few examples are spam filtration, sentiment analysis, and classifying news articles.

We will be using matplotlib and textblob packages. Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications. TextBlob is a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more. The TextBlob package for Python is a convenient way to do a lot of Natural Language Processing (NLP) tasks. For example: From textblob import TextBlob TextBlob(“not a very great calculation”).sentiment This tells us that the English phrase “not a very great calculation” has a polarity of about -0.3, meaning it is slightly negative.

As mentioned above, for twitter sentiment analysis we are using RESTful twitter API which is Tweepy. The API class provides access to the entire twitter RESTful API methods. Each method can accept various parameters and return responses. Tweepy supports accessing Twitter via Basic Authentication and the newer method, OAuth. Twitter has stopped accepting Basic Authentication so OAuth is now the only way to use the Twitter API.

Here is a sample of how to access the Twitter API using tweepy with OAuth:

import tweepy

# Consumer keys and access tokens, used for OAuth

consumer\_key = '7EyzTcAkINVS3T2pb165'

consumer\_secret = 'a44R7WvbMW7L8I656Y4l'

access\_token = 'z00Xy9AkHwp8vSTJ04L0'

access\_token\_secret = 'A1cK98w2NXXaCWMqMW6p'

# OAuth process, using the keys and tokens

auth = tweepy.OAuthHandler(consumer\_key, consumer\_secret)

auth.set\_access\_token(access\_token, access\_token\_secret)

# Creation of the actual interface, using authentication

api = tweepy.API(auth)

Various steps of proposed approach are:

1. Tweet extraction: Simply extracts tweets using twitter API.
2. Tokenizer: A necessary step in many NLP tasks. It is a process of replacing sensitive data with unique identification symbols that retain all the essential information about the data without compromising its security.
3. Pre-processing: Technique used to convert the raw data into a clean data set. In our case pre-processing involves following:

a. Removing English tweets.

b. Replacing emotions by their polarity.

c. Remove URL, target mention, hashtags, numbers.

d. Replace negative mentions.

e. Remove nouns and prepositions.

1. Feature extractor: It analyses the pre-processed data and re-reduce it to more manageable groups.
2. Classification and prediction: Classification predicts categorical class labels and prediction predicts continuous valued functions.

DEVELOPERACCOUNT

We need the developer account of twitter to get access to twitter API which will extract all the tweets that are required in our project. We can get the tweets by generating twitter API key, API secret key, access token and access secret token.

The method to create the developer is given below. By clicking on the given link you will be redirected to the twitter developer account- <https://developer.twitter.com/en/apps> .

1. Log in with your twitter account , click on create an app and select the purpose you are creating the app for.
2. After filling all the information, an application request will be made for accessing the twitter developer account.
3. Once this is done, go to profile and create an app by filling the information.
4. After the creation of the app, we can now have the API key, API secret key, access token , access token secret.

GATHERING DATASET FOR SENTIMENT ANALYSIS

Twitter allows us to extract the tweets of any user using Twitter API or Tweepy. The first thing to do is to get the consumer key, consumer secret, access key and access secret from twitter developer account. These keys will help with the API authentication.

**import sys,tweepy,csv,re**

● sys - The sys module provides information about constants, functions and methods of the Python interpreter.

● csv - The csv module implements classes to read and write tabular data in CSV format. It allows programmers to say, “write this data in the format preferred by Excel,” or “read data from this file which was generated by Excel,” without knowing the precise details of the CSV format used by Excel.

● re - This module provides regular expression matching operations.

**consumer\_key = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"**

**consumer\_secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"**

**access\_key = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"**

**access\_secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"**

Now in order to authorize our app to access Twitter, we need to use the OAuth Interface. Tweepy provides the convenient Cursor interface to iterate through different types of objects.Tweepy supports oauth authentication. Authentication is handled by the tweepy.AuthHandler class.

After having all the keys and secrets from the twitter developer account, you need to create OAuthHandler instance. Into this function, we’ll pass consumer token and secret.

**auth = tweepy.OAuthHandler(consumer\_token, consumer\_secret)**

**auth.set\_access\_token(accessToken, accessTokenSecret)**

**api = tweepy.API(auth)**

Access token is what is issued to the consumer by the service provider once the consumer completes authorization. This token defines the access privileges of the consumer over a particular user's resources. Each time the consumer wants to access the user's data from that service provider, the consumer includes the access token in the API request to the service provider.

Then create API using tweepy.API .

**self.tweets=tweepy.Cursor(api.search,q=searchTerm,lang="en").items(NoOfTerms)**

● **api.search -** Returns a collection of relevant Tweets matching a specified query.

● **q** – the search query string

● **lang** – Restricts tweets to the given language

● **items -** You pass into the items() or pages() methods the limit you want to impose

*# Remove Links, Special Characters etc from tweet*

**def cleanTweet():**

**return ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t]) | (\w +:\ / \ / \S +)", " ", tweet).split())**

A Regular Expression (or Regex) is a pattern (or filter) that describes a set of strings that matches the pattern. In other words, a regex accepts a certain set of strings and rejects the rest.

A regex consists of a sequence of characters, metacharacters (such as ., \d, \D, \s, \S, \w, \W) and operators (such as +, \*, ?, |, ^). They are constructed by combining many smaller sub-expressions.

These characters have special meaning in regex:

● metacharacter: dot (.)

● bracket list: [ ]

● position anchors: ^, $

● occurrence indicators: +, \*,?, { }

● parentheses: ()

● or: |

● escape and metacharacter: backslash (\)

*# iterating through tweets fetched*

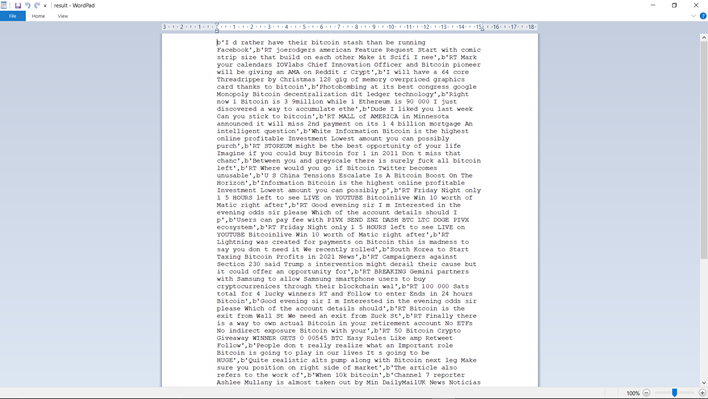
**for tweet in self.tweets:**

*#Append to temp so that we can store in csv later.*

**self.tweetText.append(self.cleanTweet(tweet.text).encode('utf-8'))**

We stored all the tweets which we fetched into tweets [ ]. In this *for* loop, we will iterate through all the tweets which we have fetched from twitter and then we’ll call cleantweet() function which will return processed tweets ( processed tweets are tweets from which hashtags, links, emoticons etc are being removed. ) Now all the processed tweets will be then appended to the tweetText []. So, we have all the clean tweets, which we’ll need for analysis, in the list.

We have stored data in a csv file to display tweets later.



This shows the data which is tweets extracted from twitter using tweepy API. This data refers to the fifty tweets about bitcoin which were recently uploaded on twitter.

ANALYZING TWEETS AND CODING

After collecting tweets on a particular subject or event, we analyze them to find out the polarity of each fetched tweet to generate a general report of sentiment regarding those fetched tweets.

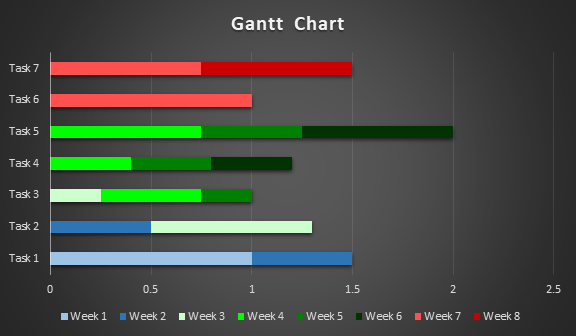
TextBlob is a python library and offers a simple API to access its methods and perform basic NLP tasks.TextBlob aims to provide access to common text-processing operations through a familiar interface. You can treat TextBlob objects as if they were Python strings that learned how to do Natural Language Processing.

The *sentiment* function of textblob returns two properties, polarity, and subjectivity.

Polarity is float which lies in the range of [-1,1] where 1 means positive statement and -1 means a negative statement. Subjective sentences generally refer to personal opinion, emotion or judgment whereas objective refers to factual information. Subjectivity is also a float which lies in the range of [0,1].

Return type: namedtuple of the form Sentiment (polarity, subjectivity)

DETAILED SCHEDULE



Task 1 - Framing the problem

Task 2 - Obtaining Dataset

Task 3 - Cleaning and processing data

Task 4 - Selecting Algorithm

Task 5 - Designing the model

Task 6 - Selecting the model

Task 7 - Testing the model

PROGRESS TILL NOW

Till now we’ve covered the main part of the project. User runs the code and enters the keyword and number of tweets to be extracted from the twitter. Program connects through twitter API tweepy and extracts that particular number of tweets and stores it. After that it processes them to remove the unwanted part from the tweets and finds out the polarity of every tweet which helps to generate the overall polarity regarding that particular subject. Now, we are currently working on the front-end, building a graphical user interface so that a user can connect to the program in a better and more understanding way.

REFRENCES

# <https://stackoverflow.com/questions/34518570/how-are-sentiment-analysis-computed-in-blob>

<https://towardsdatascience.com/creating-the-twitter-sentiment-analysis-program-in-python-with-naive-bayes-classification-672e5589a7ed>

<https://stackoverflow.com/questions/10059594/a-simple-explanation-of-naive-bayes-classification>

https://monkeylearn.com/blog/sentiment-analysis-of-twitter/#What-is-Sentiment-Analysis