SENTIMENT ANALYSIS ON DEMONETIZATION TWEETS

Progress Report

BACHELOR OF TECHNOLOGY
In
Computer Science and Engineering

By

Abhishek Singh Adhikari (Roll no. 140180101001)

Ajay Rawat (Roll no. 140180101004)

Manoj Kumar (Roll no.140180101027)

Under the supervision of

Dr. Ajit Singh

Head Of the Department

Department of Computer Science and Engineering

Acknowledgement

First of all, I would like to express my thanks to my guide **Mr Amit Paul** Assistant Professor, Computer Science and Engineering Department, B.T.K.I.T, Dwarahat for being an excellent mentor for me during my whole course of project. His encouragement and valuable advice during the entire period has made it possible for me to complete my work.

I am also thankful to our superviser **“Mr Vishal Kumar”** Assistant Professor Computer Science Engineering Department, B.T.K.I.T Dwarahat for setting high standards for his students and encouraging them time to time so that they can achieve them as well.

Lastly, I would like to thank my parents for their years of unyielding love and encourage. They have wanted the best for me and I admire their sacrifice and determination.

Abhishek Adhikari

Ajay Rawat

Manoj Chaku

Chapter:1 Introduction

In this chapter we are going to give the introductions on Sentiment Analysis, Python and Natural Language Toolkit (NLTK). Then we are explaining the objective of our project. After this we will discuss why there is a need of sentiment analysis and some of the applications of Sentiment Analysis which are used in our dailylife.

1.1 Introduction to SentimentAnalysis

Sentiment Analysis is process of collecting and analyzing data based upon the person feelings, reviews and thoughts. Sentimental analysis often called as opinion mining as it mines the important feature from people opinions. Sentimental Analysis is done by using various machine learning techniques, statistical models and Natural Language Processing (NLP) for the extraction of feature from a large data.

Sentiment Analysis can be done at document, phrase and sentence level. In document level, summary of the entire document is taken first and then it is analyze whether the sentiment is positive, negative or neutral. In phrase level, analysis of phrases in a sentence is taken in account to check the polarity. In Sentence level, each sentence is classified in a particularclass to provide the sentiment.

Sentimental Analysis has various applications. It is used to generate opinions for people of social media by analyzing their feelings or thoughts which they provide in form of text. Sentimental Analysis is used in many real life scenarios, to get reviews about any product or movies, to get the financial report of any company, for predictions or marketing.

Twitter is a micro blogging platform where anyone can read or write short form of message which is called tweets. The amount of data accumulated on twitter is very huge. This data is unstructured and written in natural language. Twitter Sentimental Analysis is the process of accessing tweets for a particular topic and predicts the sentiment of these tweets as positive, negative or neutral with the help of different machine learning algorithm.

1.1 Introduction to Python

Python is a high level, dynamic programming language which is used for this project. Python3.4 version was used as it is a mature, versatile and robust programming language. It is an interpreted language which makes the testing and debugging extremely quickly as there is no compilation step. There are extensive open source libraries available for this version of python and a large community of users.

Python is simple yet powerful, interpreted and dynamic programming language, which is well known for its functionality of processing natural language data, i.e. spoken English using NLTK. Other high level programming languages such as 'R' and 'Matlab' were considered because they have many benefits such as ease of use but they do not offer the same flexibility and freedom that Python candeliver.

1.2 Introduction to NLTK

Natural Language Toolkit (NLTK) is library in Python, which provides a base for building programs and classification of data. NLTK is a collection of resources for Python that can be used for text processing, classification, tagging and tokenization. This toolbox plays a key role in transforming the text data in the tweets into a format that can be used to extract sentiment from them.

NLTK provides various functions which are used in pre-processing of data so that data available from twitter become fit for mining and extracting features. NLTK support various machine learning algorithms which are used for training classifier and to calculate the accuracy of differentclassifier.

In our project we use Python as our base programming language which is used for writing code snippets. NLTK is a library of Python which plays a very important role in converting natural language text to a sentiment either positive or negative. NLTK also provides different sets of data which are used for training classifiers. These datasets are structured and stored in library of NLTK, which can be accessed easily with the help of Python.

Chapter 2: Problem Statement

In this fast growing world, Social Networking website like Facebook, Twitter, etc. plays a very vital role. Twitter is a micro-blogging platform which provides a tremendous amount of data which can be used for various applications of Sentiment Analysis like predictions, reviews, elections, marketing, etc. Sentiment Analysis is a process of extracting useful information from large amount of data, and classifies them into different classes called sentiments.

Basically these sentiments can be:

- Negative
- Positive
- Neutral

This is basically an unsupervised classification of data, as no initial classes are present to distinguish the data or tweets.

This project is being developed with the aim of finding the reviews or emotions regarding the Policy of Demonetization that occurred under the rule of Prime Minister Narendra Modi on NOV 2016. So that by finding the sentiments of people regarding the policies can help prevent/encourage such policies from happening in future for the betterment of country and for the people of country. The main objective of this project work is to perform the sentiment analysis on Demonetization so that people opinions about this policy could be predicted through the tweets which are extracted from Twitter.

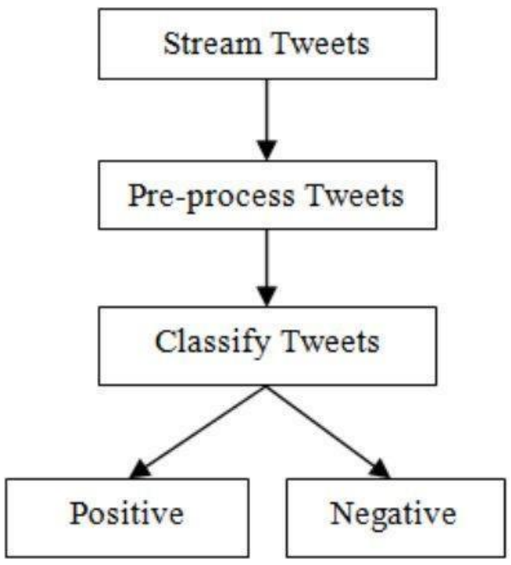
Thus to achieve this objective we build a classifier based on supervised learning and perform live sentiment analysis on the tweets collected.

Chapter 3:Implementation

Data collection is not a simple task, as it may seem. Various decisions have to be made for collecting data. For our project we maintain dataset for training, testing and for twitter sentiment analysis. In this chapter we are going to study how data is collected, stored, processed and classified. Before discussing these process and different dataset, let us discuss our proposed architecture.

3.1ProposedArchitecture

As our goal is to achieve sentiment analysis for data provided from Twitter. We are going to build a classifier which consists of different machine learning classifiers. Once our classifier is ready and trained we are going to follow the steps shown in the figure below.



.

Step-1 First we are going to stream tweets in our build classifier with the help of Tweepy library in python

Step-2 Then we pre-process these tweets, so that they can be fit for mining.

Step-3 After pre-processing we pass this data in our trained classifier, which then classify them into positive or negative class based on trained results.

Since, Twitter is our source of data for analysis. We are going to stream the tweets from twitter in our database. For this we are going to use Twitter Application

3.2 Data Collection

3.2.1 TwitterData

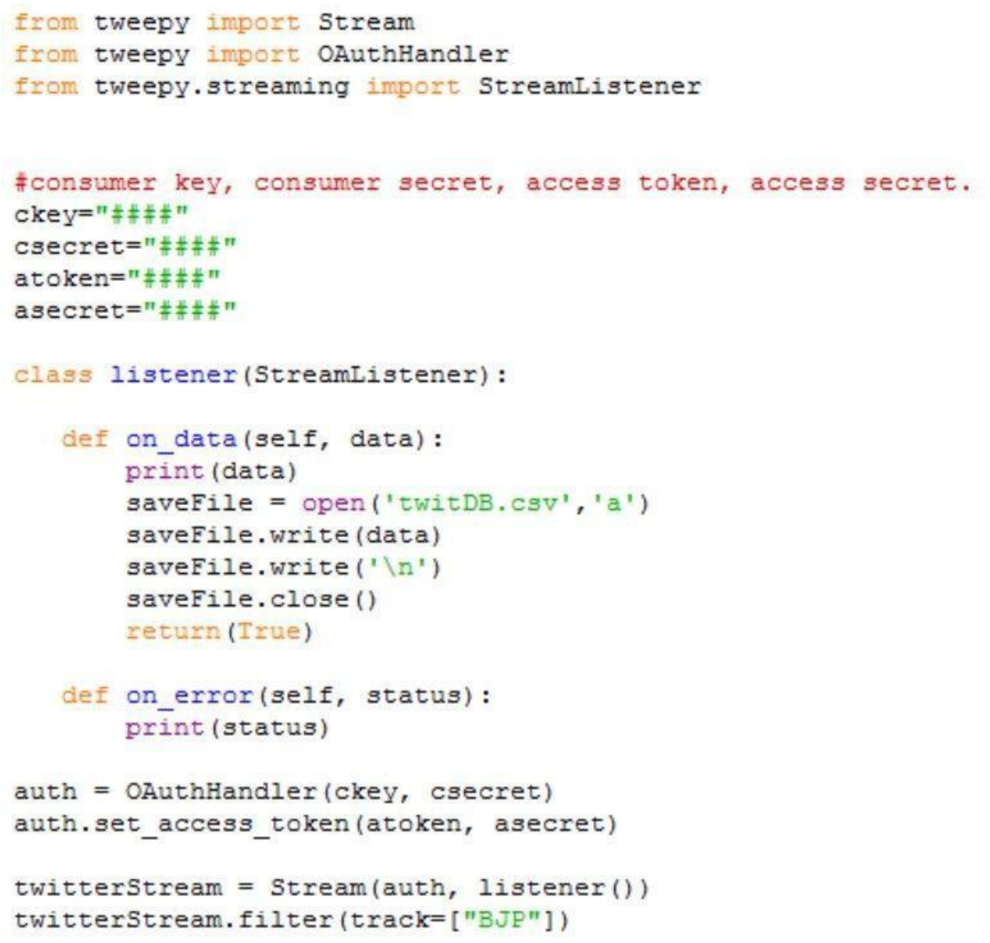
To use Twitter API we must first have a twitter account. It can be easily created by filling the sign up details in twitter.com website. After this you will be provided with a username and password which is use for login purpose. Once your account is created, you can now read and send tweets on any topic you want toexplore.

Twitter provider a platform from which we can access data from twitter account and can use it for our own purpose. For this we have to login with our twitter credentials in dev.twitter.com website. In this website, we first create an application which will be used for streaming tweets by providing necessary details. Once our API is created we can get to know customer key, customer secret key, access token key and access secret key. These keys are used to authenticate user when user want to access twitter data.

As the objective of this project is to analyze the sentiment of Tweets posted on Demonetization, only tweets about related to this should be collected. Hence for this we create a Python script which will be used to fetch tweets from twitter. Before creating this script we first install a library in Python called **tweepy.**

Python is a very powerful language which provides many services with the help of many Python libraries. Tweepy is one of the open source Python library which enables Python to communicate with twitter and use its API to collect data so that we can use it in our program. To install tweepy, just provide a command ‘pip install tweepy’ in command prompt or bash and we ready to go with ourscript.

In this script we use all the keys and secrets which we got in API, we first create listener class which is used to load the data from the twitter. Now to gather data we first set up ‘OAuth’ protocol. OAuth is a standard protocol which is used for authorization. It allow user to log in any third party websites by using any social network website account without exposing passwords. OAuth provides security and authorization to user. The script which we use to access data with the help of twitter is shown in figure below.



```
from tweepy import Stream
from tweepy import OAuthHandler
from tweepy.streaming import StreamListener


#consumer key, consumer secret, access token, access secret.
ckey="####"
csecret="####"
atoken="####"
asecret="####"

class listener(StreamListener):

    def on_data(self, data):
        print(data)
        saveFile = open('twitDB.csv','a')
        saveFile.write(data)
        saveFile.write('\n')
        saveFile.close()
        return(True)

    def on_error(self, status):
        print(status)

auth = OAuthHandler(ckey, csecret)
auth.set_access_token(atoken, asecret)

twitterStream = Stream(auth, listener())
twitterStream.filter(track=["BJP"])
```

Figure 4.2 Code for getting tweets using Twitter API

In this script we have to provide all the keys which are given by Twitter API. To get the tweet for a particular topic we import ‘Stream’ library from tweepy. In this we pass the authorization detail and the class in which we import tweets. We also apply a filter in the stream which will help us to provide the tweets for the particular topic by providing a keyword related to that topic in filter. Once we run our script, we see tweets are imported from Twitter and we can then use them for our purpose.

3.2.2 Training Data:

Other data which we collected for this project is training data. This data is used to train the classifier which we are going to build. To collect this data we use NLTK library of Python. NLTK consists of corpora, which is very large and consists of structured set of text files which are used to perform analysis. In these corpora there are various types of text files like quotes, reviews, chat, history, etc. From these corpora we will select files of movie reviews for our training purpose. Sample of these reviews is shown in Table4.1

Table 4.1 Sample movie reviews in NLTK Corpus

Movie Reviews	**CLASS**
foolish, idiotic and boring it's so lad dish and youngish , only teenagers could find it funny	NEGATIVE
the rock is destined to be the 21st century's new conan and that he's going to make a splash even greater than arnoldschwarzenegger	POSITIVE
Barry Sonnenfeld owes frank the pug big time the biggest problem with roger avary's uproar against the map	NEGATIVE
the seaside splendor and shallow , beautiful people are nice to look at while you wait for the story to get going	POSITIVE

In movie reviews corpus there are around 5000 reviews each for positive and negative feedback. These reviews are short and arranged in text files which are easy to access. We train our classifier from around 80% of the data and then we test it with remaining 20% to check that trained classifier is working properly or not.

3.3 Data Storage

Once, we start getting our data from Twitter API our next step is to store that data so that we can use it for sentiment analysis. We ran our scripts for period of month and collect the tweets on our topic of interest ie Demonetization.Every time we ran the script described in figure a .csv (comma separated values) file is generated which consists of tweets that are extracted from Twitter API. We use .csv format for our collected data files because data consists of many fields. CSV separate each field with a comma, thus make it very easier to access the particular field which consists of text. CSV files also provide faster read/write time as compared to others.

We make separate directories to store tweets of for respective month. We store them in our hard drive from where these can be easily imported to our snippet and further proceed for analysis. Once we stored our tweet we have to pre process the data stored before applying it to classifier because the data we collect from API is not fit for mining. Therefore pre-processing the data is our next step. Figure shows glimpse of different files stored in hard drive.

3.4 Data Preprocessing

Data obtained from twitter is not fit for extracting features. Mostly tweets consists of message along with usernames, empty spaces, special characters, stop words, emoticons, abbreviations, hash tags, time stamps, URL’s ,etc. Thus to make this data fit for mining we pre-process this data by using various function of NLTK. In pre- processing we first extract our main message from the tweet, then we remove all empty spaces, stop words (like is, a, the, he, them, etc.), hash tags, repeating words, URL’s, etc. We then replace all emoticons and abbreviations with their corresponding meanings like :-), =D, =), LOL, Rolf, etc. are replaced with happy or laugh. Once we are done with it, we are ready with processed tweet which is provided to classifier for required results. A sample processed tweet is shown in Table 4.2

Tweet Type	**Result**
Original tweet	RT @rssurjewala: Critical question: Was PayTM informed about #Demonetization edict by PM? It's clearly fishy and requires full disclosure &amp;...
Processed tweet	critical question paytm informed demonetization edict pm clearly fishy requires full disclosure

Cleaning of Twitter data is necessary, since tweets contain several syntactic features that may not be useful for analysis. The pre-processing is done in such a way that data represented only in terms of words that can easily classify the class.

We create a code in Python in which we define a function which will be used to obtain processed tweet. This code is used to achieve the following functions:

- ∑ remove quotes - provides the user to remove quotes from the text
- ∑ remove @ - provides choice of removing the @ symbol, removing the @ along with the user name, or replace the @ and the user name with a word 'AT_USER' and add it to stopwords
- ∑ remove URL (Uniform resource locator) - provides choices of removing URLs or replacing them with 'URL' word and add it to stop words
- ∑ removeRT (Re-Tweet) - removes the word RT from tweets
- ∑ remove Emoticons - remove emoticons from tweets and replace them with their specificmeaning
- ∑ removeduplicates – remove all repeating words from text so that there will be no duplicates
- ∑ remove # - removes the hash tag class

Table 4.3 Removed and modified content

CONTENT	**ACTION**
Punctuation (! ? , . ” : ;)	Removed
#word	Removed #word
@any_user	Remove @any_user or replaced with “AT_USER” and then added in stop words.
Uppercase characters	Lowercase all content
URLs and web links	Remove URLs or replaced with “URL” and then added in stop words
Number	Removed
Word not starting with alphabets	Removed
All Word	Stemmed all word (Converted into simple form)
Stop words	Removed
Emoticons	Replaced with respective meaning
White spaces	Removed

Table 4.4 Sample cleaned data

Raw data	**Clean data**
RT @mrpatel1954: #Demonetization has long term benefits. Pain now is temporary. Let us help #nation to make #India #corruption free.	demonetization long term benefit pain temporary let help nation make india corruption free
RT @Atheist_Krishna: BEFORE and AFTER Gandhi ji heard they are standing there against #Demonetization . https://t.co/9NheK63TPg	gandhi heard standing against demonetization

3.5 Classification:

To classify tweets in different class (positive and negative) we build a classifier which consists of several machine learning classifiers. To build our classifier we used a library of Python called, Scikit-learn. Scikit-learn is a very powerful and most useful library in Python which provides many classification algorithms. Scikit-learn also include tools for classification, clustering, regression and visualization. To install Scikit-learn we simply use on line command in python which is ‘pip install scikit- learn’.

In order to build our classifier, we use seven in-build classifiers which come in Scikit- learn library, which are:

- ∑ Naïve-Bayes Classifier
- ∑ MultinomialNB Classifier

The reason we are using seven classifiers, so that we can get the more reliable output. To use these classifiers, we write a script in Python, in which we first import the classifier and then we pass the training set to each classifier

3.5.1 Feature Extraction

As we already discussed, training and testing data is collected from NLTK corpus. We have round 5000 tweets on demonetization each for positive, negative and neutral class. We take first 4000 tweets as training set and remaining 1000 as testing sets.

Both the training and testing data must be represented in same order for learning. One of the ways that data can be represented is feature-based. By features, it is meant that some attributes that are thought to capture the pattern of the data are first selected and the entire dataset must be represented in terms of them before it is fed to a machine learning algorithm. Different features such as n-gram presence or n-gram frequency, POS (Part of Speech) tags, syntactic features, or semantic features can be used. For example, one can use the keyword lexicons as features. Then the dataset can be represented by these features using either their presence or frequency.

Attribute selection is the process of extracting features by which the data will be represented before any machine learning training takes place. Attribute selection is the first task when one intends to represent instances for machine learning. Once the attributes are selected, the data will be represented using the attributes. So attributes are the features. Although we used the entire data set in our selection of attributes, the representation of the data must be done on a per instance (Twitter post) basis.

Feature vector plays a very important role in classification and helps to determine the working of the build classifier. Feature vector also help in predicting the unknown data sample. There are many types of feature vectors, but in this process we used unigram approach. Each tweet words are added to generate the feature vectors. The presence/absence of sentimental word helps to indicate the polarity of the sentences. We create a python script to extract the features from the training data. Code snippet for extracting features is shown in Figure4.4

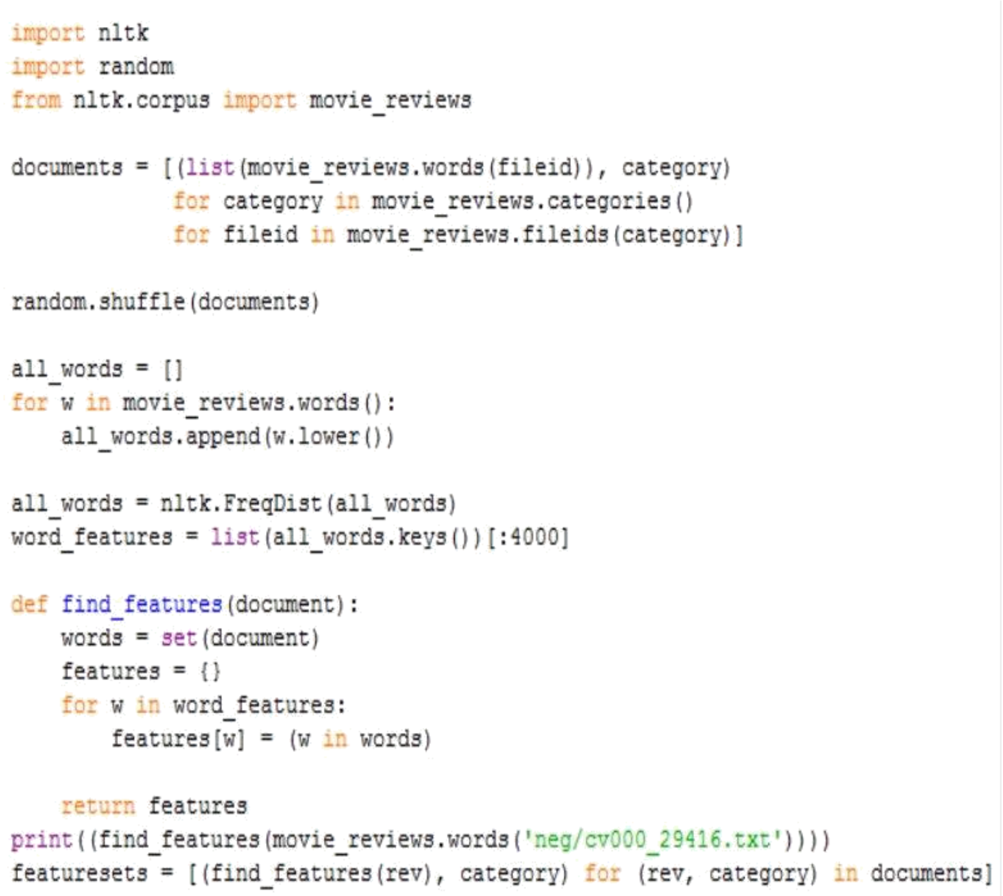
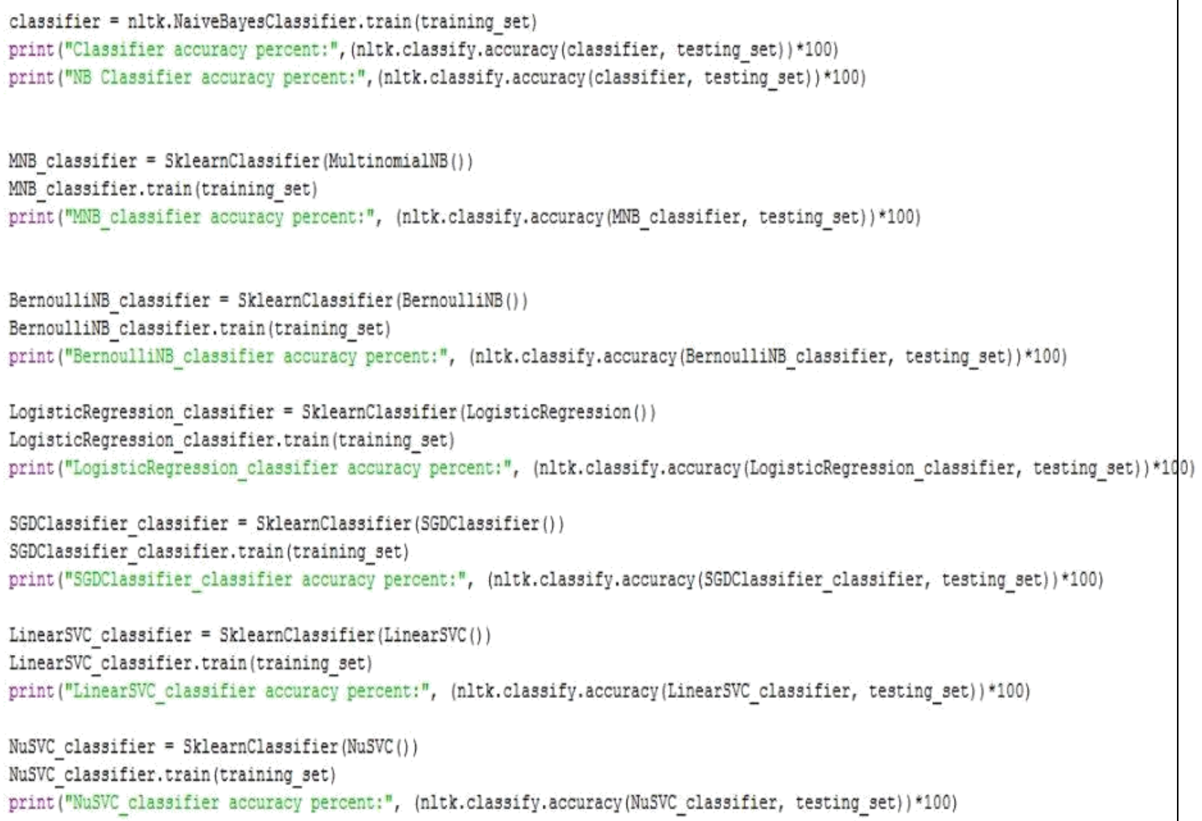

```
import nltk
import random
from nltk.corpus import movie_reviews

documents = [(list(movie_reviews.words(fileid)), category)
             for category in movie_reviews.categories()
             for fileid in movie_reviews.fileids(category)]

random.shuffle(documents)

all_words = []
for w in movie_reviews.words():
    all_words.append(w.lower())

all_words = nltk.FreqDist(all_words)
word_features = list(all_words.keys())[:4000]

def find_features(document):
    words = set(document)
    features = {}
    for w in word_features:
        features[w] = (w in words)

    return features
print((find_features(movie_reviews.words('neg/cv000_29416.txt'))))
featuresets = [(find_features(rev), category) for (rev, category) in documents]
```

Figure 4.4 Code for extracting features from tweets

Once we extract the features from training data, we are going to pass these in our build classifiers. A script in written in python which is used to pass training sets in classifier. Once, the classifier is trained we can also check the accuracy of each classifier by passing the testing set. Sample script of training and testing of classifier is shown in Figure4.5



```
classifier = nltk.NaiveBayesClassifier.train(training_set)
print("Classifier accuracy percent:",(nltk.classify.accuracy(classifier, testing_set))*100)
print("NB Classifier accuracy percent:",(nltk.classify.accuracy(classifier, testing_set))*100)


MNB_classifier = SklearnClassifier(MultinomialNB())
MNB_classifier.train(training_set)
print("MNB_classifier accuracy percent:", (nltk.classify.accuracy(MNB_classifier, testing_set))*100)


BernoulliNB_classifier = SklearnClassifier(BernoulliNB())
BernoulliNB_classifier.train(training_set)
print("BernoulliNB_classifier accuracy percent:", (nltk.classify.accuracy(BernoulliNB_classifier, testing_set))*100)

LogisticRegression_classifier = SklearnClassifier(LogisticRegression())
LogisticRegression_classifier.train(training_set)
print("LogisticRegression_classifier accuracy percent:", (nltk.classify.accuracy(LogisticRegression_classifier, testing_set))*100)

SGDClassifier_classifier = SklearnClassifier(SGDClassifier())
SGDClassifier_classifier.train(training_set)
print("SGDClassifier_classifier accuracy percent:", (nltk.classify.accuracy(SGDClassifier_classifier, testing_set))*100)

LinearSVC_classifier = SklearnClassifier(LinearSVC())
LinearSVC_classifier.train(training_set)
print("LinearSVC_classifier accuracy percent:", (nltk.classify.accuracy(LinearSVC_classifier, testing_set))*100)

NuSVC_classifier = SklearnClassifier(NuSVC())
NuSVC_classifier.train(training_set)
print("NuSVC_classifier accuracy percent:", (nltk.classify.accuracy(NuSVC_classifier, testing_set))*100)
```

Figure 4.5 Sample code for training and testing build classifier.

Current Status of the Project:

1. Till now we have reached the 5th step of the procedure i.e. Building of baseline of 'data_collector' script to collect the data or tweets from Twitter using the Twitter Streaming API and using the Tweepy library.

2. Creation of 'pre_processor' script to clean the data collected through API. In this step we made use of regular expressions to find the pattern that are needed to remove from the dataset

.

3. Creation of *'stemmer'* script to stem the words used in the tweets of dataset.
   - **Stemming –** IT basically refers to change the word into its root form from where it has Been derived Example – working · work worked · work

4. Creation of 'split_data' script to split the data into two portion for the purpose of project:

- Test data

.

- Training data.

The dataset is divided in a ratio of 2:3. And kept in separate files programmatically.

5. Creation of 'feature_list_init' script to create feature vectors of tweets and then forming a global feature vector from these feature vectors. Basically, the concept of bag of words has been applied here.

6. Creation of 'baseline_classifier' script has been written to find the polarity of tweets at a very basic level. It also helps to find the occurrence of words present in feature list in both the –ve and +ve classes, doing this will help us further in our supervised machine learning technics.

Other supporting files/programs are also created that helps all the above main scripts to do their task efficiently.