"I stand with you: using Emojis to study Solidarity in Crisis Events"

A PROJECT REPORT

Submitted by

PRIYAM MUKHERJEE

SHAWAN BASU

SUVAM DAS

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

Year: 2019-2020

GOVERNMENT COLLEGE OF ENGINEERING AND CERAMIC TECHNOLOGY

73, Abinash Chandra Banerjee Lane, Kolkata-700 010

BONAFIDE CERTIFICATE

Certified that this project report **“I STAND WITH YOU: USING EMOJIS TO STUDY SOLIDARITY IN CRISIS EVENTS”** is the bonafide work of “**PRIYAM MUKHERJEE, SHAWAN BASU AND SUVAM DAS”** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr. Mousumi Maitra	Mrs. Paramita Dey
HEAD OF THE DEPARTMENT	**SUPERVISOR**
	Asst. Professor
Information Technology	Information Technology
Government College Of Engineering	Government College Of Engineering
And Ceramic Technology	And Ceramic Technology
73, Abinash Chandra Banerjee Lane,	73, Abinash Chandra Banerjee Lane,
Kolkata-700 010	Kolkata-700 010

SIGNATURE

External Examiner:

TABLE OF CONTENTS

ACKNOWLEDGEMENT

We take this opportunity to express our profound gratitude and deep regards to our project guide **Mrs. Paramita Dey** for her exemplary guidance, monitoring and constant encouragement throughout the course of the project. The help and guidance given by her time to time will carry us a long journey of life on which we are about to embark.

Last, but not the least we would like to thank our parents and our friends who are involved directly and indirectly in successful completion of the project work.

PRIYAM MUKHERJEE (GCECTB-R16-2019)

SHAWAN BASU (GCECTB-R16-2026)

SUVAM DAS (GCECTB-R16-2034)

ABSTRACT

Emojis are very common in social media and understanding their underlying semantics is of great interest from machine learning and natural language processing point of view. In this project, emojis have been used for sentiment analysis in order to discern the subjective opinion of a crisis event through a social media platform, that is, Twitter.
The crisis event that have been taken into account is #AyodhyaVerdict.
For collecting the data, a simple Twitter Application was build using twitter api. With the help of the credentials(consumer key, consumer secret key, access token, access token secret) the tweets were streamed and stored in a .csv file.
Recognition of the emojis and setting Sentiment rankings were done accordingly.
With the help of data preprocessing, the raw data was transformed into a useful and efficient format for cluster building and thus calculating Final Centroids, Accuracy for indiviadual Emojis, Average Accuracy score .

Chapter 1

INTRODUCTION

1.1 Defining Solidarity:

We start by defining what we mean by solidarity. The concept of solidarity has been defined by scholars in relation to complementary terms such as “community spirit or mutual attachment, social cooperation or charity”. In our work, we use the definition of expressional solidarity, characterized as individuals expressing empathy and support for a group they are not directly involved in (for example, expressing solidarity for victims of natural disasters or terrorist attacks).

1.2 Need for Solidarity Analysis:

The collective enactment of online behaviors, including prosocial behaviors such as solidarity, has been known to directly affect natural disasters and social movements. Social media, due to its increasingly pervasive nature, permits a sense of immediacy **[a notion that produces high degree of identification among politicized citizens of the web, especially in response to crisis events].**

> We study how emojis are used to express solidarity in social media in the context of two major crisis events - a natural disaster, Cyclone Fani Date: 26 April 2019 – 5 May 2019.
>
> Next, we use these expressions of solidarity to characterize human behavior in online social networks, through the temporal and geospatial diffusion of emojis, exemplified by movements related to #AyodhyaVerdict, #BlackLivesMatter, #MeToo.

Chapter 2

PROJECT PLAN

2.1 Feasibility Study

A feasibility analysis involves a detailed assessment of the need, value and practicality of a proposed enterprise, such as systems development. The process of designing and implementing record keeping systems has significant accountability and resource implications for an organization. Feasibility analysis will help you make informed and transparent decisions at crucial points during the developmental process to determine whether it is operationally, economically and technically realistic to proceed with a particular course of action.

Most feasibility studies are distinguished for both users and analysts. First, the study often presupposes that when the feasibility document is being prepared, the analyst is in a position to evaluate solutions. Second, most studies tend to overlook the confusion inherent in system development – the constraints and the assumed attitudes.

2.2 System Requirements

The system services and goals are established by consultation with system user. They are then defined in details and serve as a system specification. System requirement are those on which the system runs.

Hardware Requirements:

- Computer with either Intel Pentium processor or AMD processor.
- 8GB DDR RAM
- 1TB hard disk drive

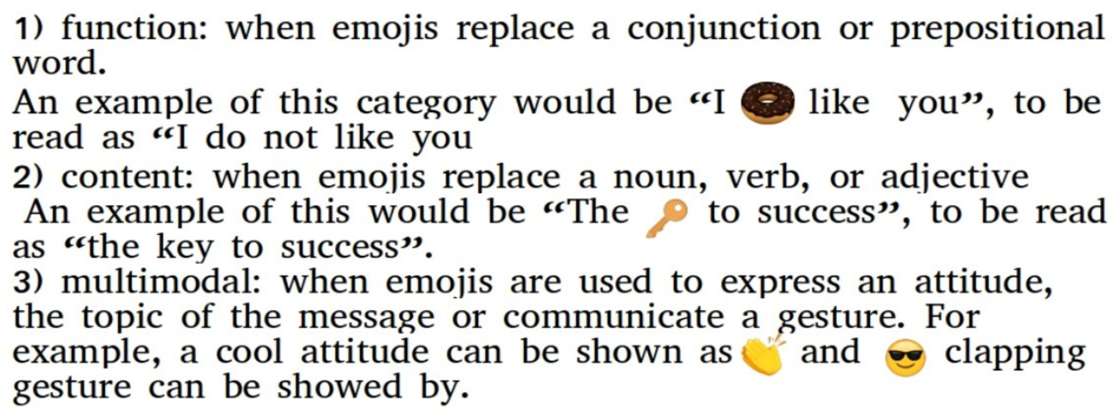
Software Requirements:

- UBUNTU operating system.
- Preferable language Python
- Data Analysis: Machine Learning (Preferable Algorithm SVM)

Chapter 3

USING EMOJIS TO UNDERSTAND HUMAN BEHAVIOR

With respect to research on expressional solidarity, it is found that individuals were more outspoken on social media after a tragic event. They studied solidarity in tweets spanning geographical areas and several languages related to a crisis event. Extant research on emojis usage has designated three categories, that are



1) function: when emojis replace a conjunction or prepositional word.
An example of this category would be “I 🍩 like you”, to be read as “I do not like you
2) content: when emojis replace a noun, verb, or adjective
An example of this would be “The 🔑 to success”, to be read as “the key to success”.
3) multimodal: when emojis are used to express an attitude, the topic of the message or communicate a gesture. For example, a cool attitude can be shown as 👏 and 😎 clapping gesture can be showed by.

Sentiment Analysis

Sentiment analysis uses data mining processes and techniques to extract and capture data for analysis in order to discern the subjective opinion of a document or collection of documents, like blog posts, reviews, news articles and social media feeds like tweets and status updates. Sentiment analysis allows organizations to track the following:

- Brand reception and popularity
- New product perception and anticipation
- Company reputation

RC1: How useful are emojis as features in classifying expressions of solidarity?

Standard pre-processing techniques are used for removing stopwords and lowercasing the tweets. The hashtags that were annotated from the tweets are removed and this raw cleaned data is used for processing of the result.

RC2: Which emojis are used in expressions of solidarity during crisis events and how do they compare to emojis used in other tweets?

The emojis used during crisis events are very much used in large number compared to other emojis which are not much linked with crisis event.

For eg: Terrorist attack at a certain place calls for the emojis 🙏 and 😢 over other images.

So, the emojis used in expressions of solidarity is very much dependent on the crisis events.

RC3: Which emojis occur in tweets that are posted within areas directly affected by crisis events as compared to those tweets that are posted from other areas?

This research question puts forth that solidarity would be expressed differently by people that are directly affected by the crisis than those who are not.
For eg: The recent terror attacks in Sri Lanka would call for 😔 for grief, 😠 for anger towards the terrorists, 🇱🇰 Flag for supporting the country (among the people of the country) and people who did not face the situation would call for 🙏 for praying for the victims and 🇱🇰 flag by supporting the country in tough times.

RC4: How can emojis be used to study the temporal and geographical difussion of solidarity expressions during crisis events?

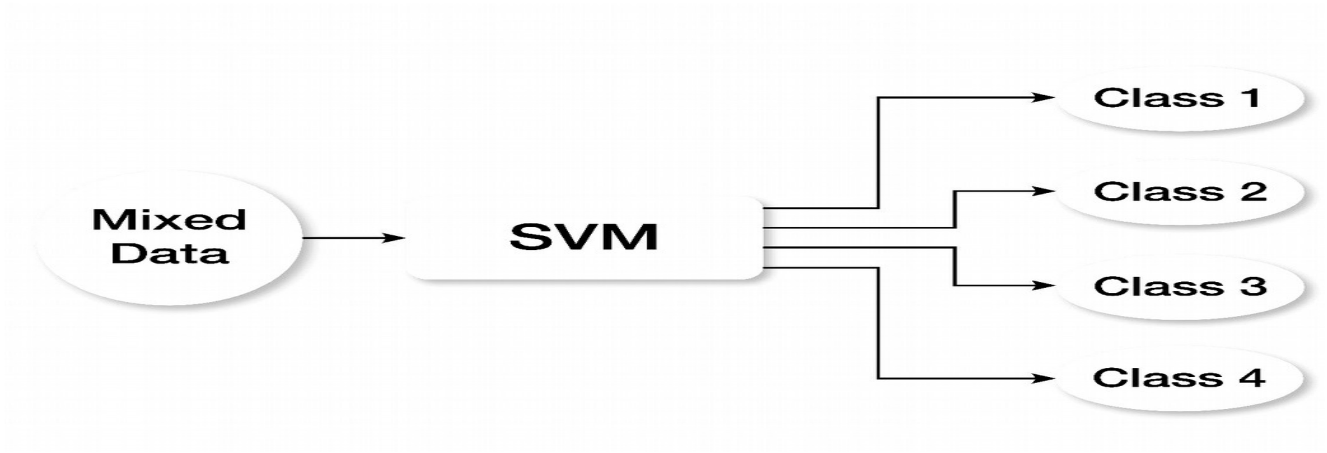
Our primary aim with this research question was to look at how the emojis of solidarity diffuse over time within the affected community and compare communities not affected by the same event.
(different portrayals of emojis from different communities [one community affected and other community portraying feels regarding the other one]).
For example, after an earthquake at a certain place, 😢 emojis depicts the tension among the people, 🙏 praying to God at the time of crisis and ☢️ emojis depicts radioactivity if happens and depicting danger at that certain place and 🆘 depicting help.

Chapter 4

ALGORITHMS

4.1 What is Support Vector Machine (SVM)?

“Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well.



4.2 Working of SVM

Raw unclean data is bought into the SVM and with a hyperplane the data is divided into four classes. For example,if we take Terrorist attack into account, then,

😔 - will be kept in Class 1 depicting grief.

🙏 - will be kept in Class 2 depicting support.

😡 - will be kept in Class 3 depicting anger.

🇱🇰 - will be kept in Class 4 depicting nationality.

Chapter 5

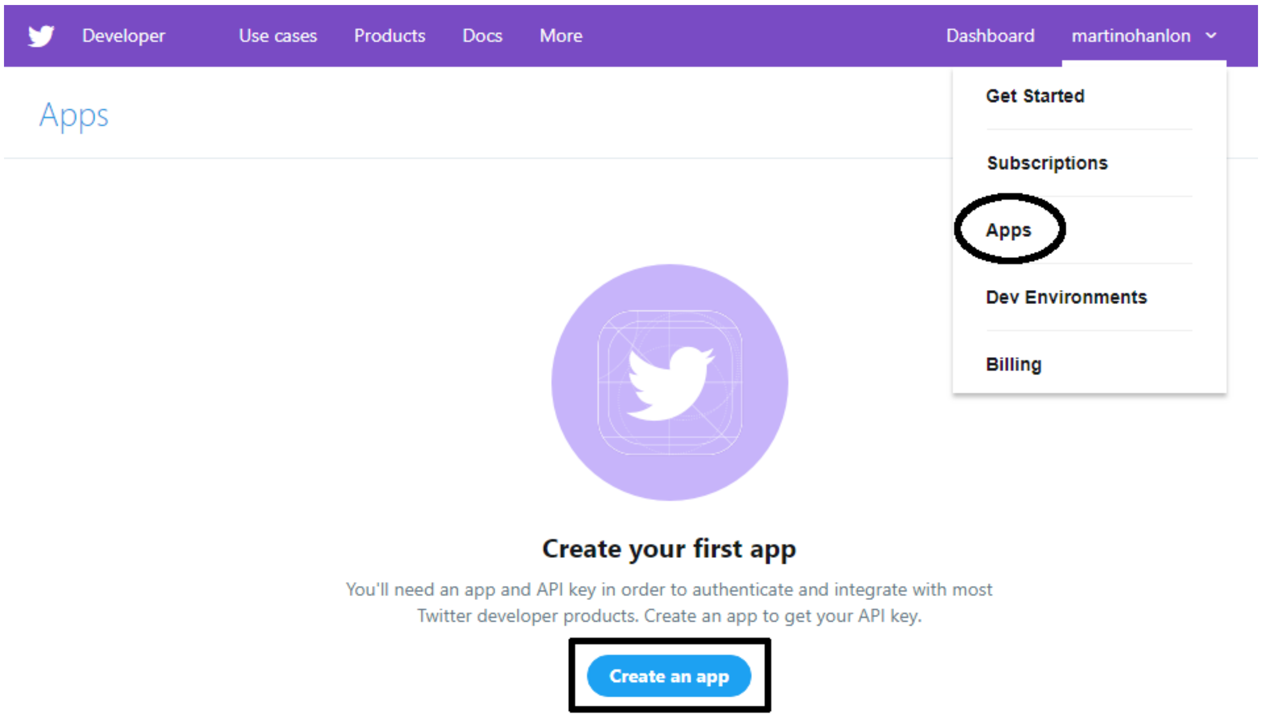
DATA COLLECTION

5.1 Building Twitter Application

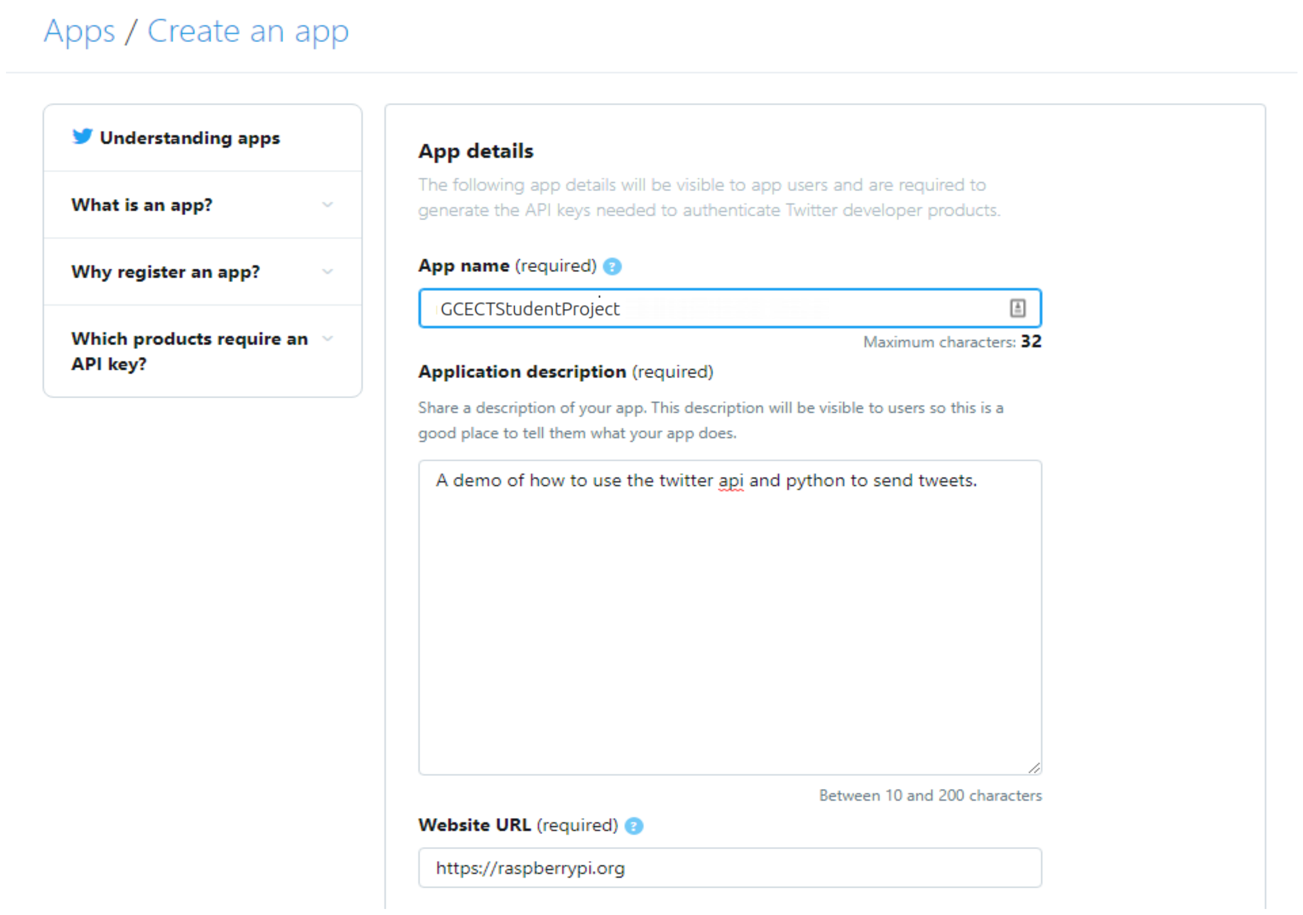
To create a twitter application first we need a twitter account. After creating an account we have to apply for twitter developer account.

After our developer account has been approved, we need to register our application with Twitter to get keys. These keys allow us to access our Twitter account from our Python program using the Twitter API (Application Programming Interface).

Go to developer.twitter.com, select Apps from the menu, and click on the Create an app button.



Complete the application details form. We must enter an app name, description, website and some information about how the app will be used. You can leave the other fields blank. Then click Create.


Apps / Create an app

Understanding apps

What is an app?

Why register an app?

Which products require an API key?

App details

The following app details will be visible to app users and are required to generate the API keys needed to authenticate Twitter developer products.

App name (required)

GCECTStudentProject

Maximum characters: **32**

Application description (required)

Share a description of your app. This description will be visible to users so this is a good place to tell them what your app does.

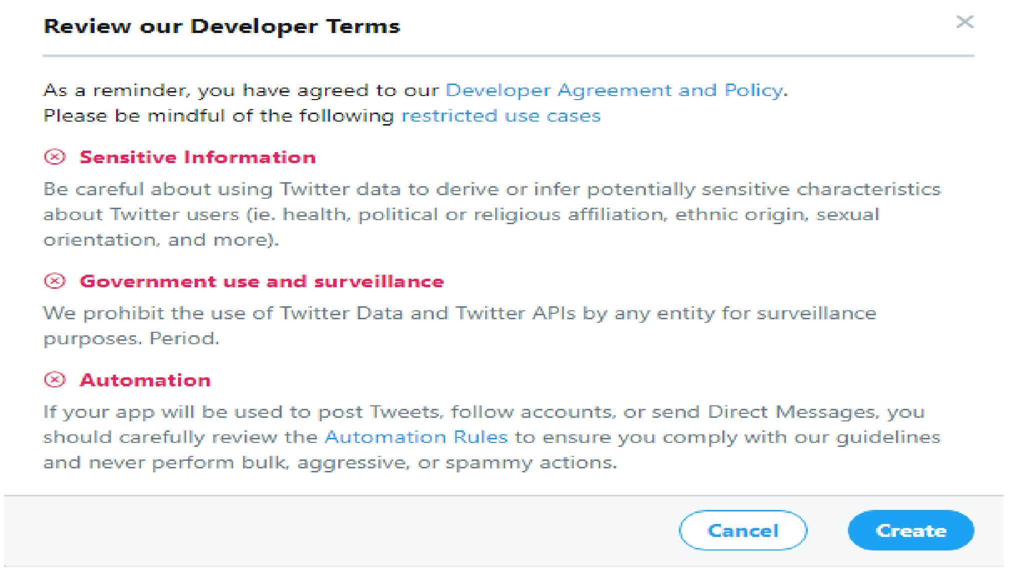
A demo of how to use the twitter api and python to send tweets.

Between 10 and 200 characters

Website URL (required)

https://raspberrypi.org

Review the Developer Terms and click on Create.


Review our Developer Terms

As a reminder, you have agreed to our Developer Agreement and Policy.
Please be mindful of the following restricted use cases

Sensitive Information

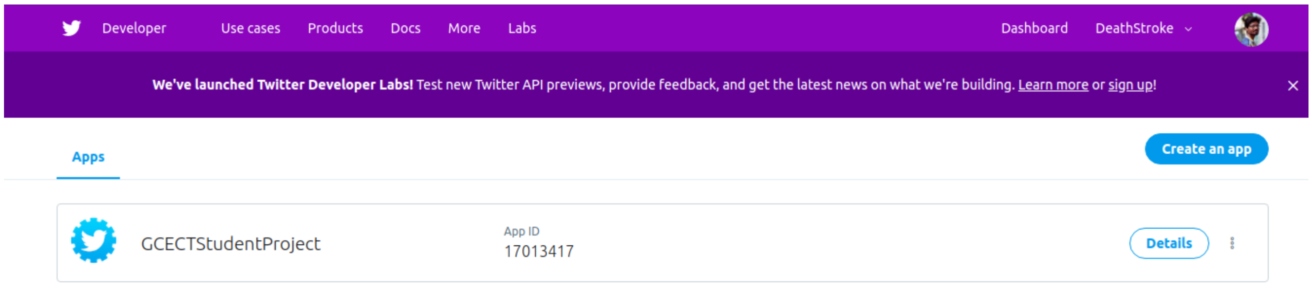
Be careful about using Twitter data to derive or infer potentially sensitive characteristics about Twitter users (ie. health, political or religious affiliation, ethnic origin, sexual orientation, and more).

Government use and surveillance

We prohibit the use of Twitter Data and Twitter APIs by any entity for surveillance purposes. Period.

Automation

If your app will be used to post Tweets, follow accounts, or send Direct Messages, you should carefully review the Automation Rules to ensure you comply with our guidelines and never perform bulk, aggressive, or spammy actions.


Developer Use cases Products Docs More Labs Dashboard DeathStroke

We've launched Twitter Developer Labs! Test new Twitter API previews, provide feedback, and get the latest news on what we're building. Learn more or sign up!

Apps

Create an app

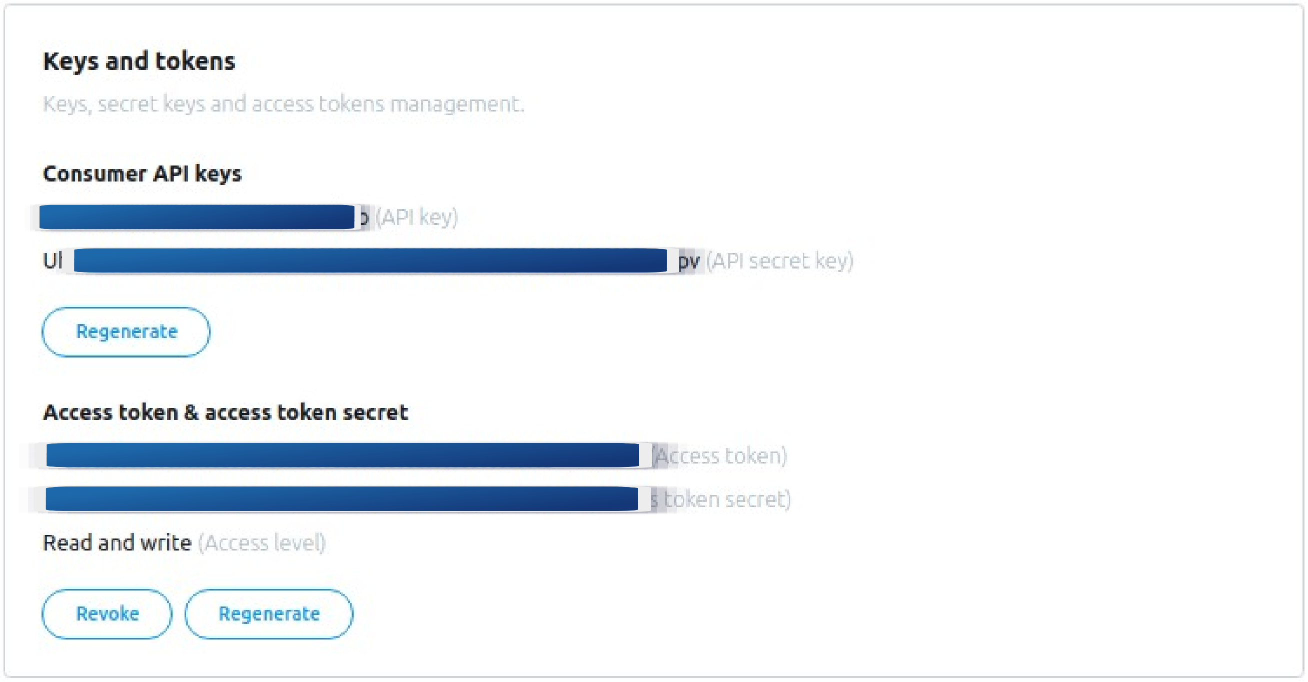
GCECTStudentProject

App ID
17013417

Details

Click on the Keys and tokens tab to view your **keys** and **access tokens**.

Click on the Create button under **Access token** & A**ccess token secret.**


Keys and tokens

Keys, secret keys and access tokens management.

Consumer API keys

[redacted] (API key)

Ul [redacted] (API secret key)

Regenerate

Access token & access token secret

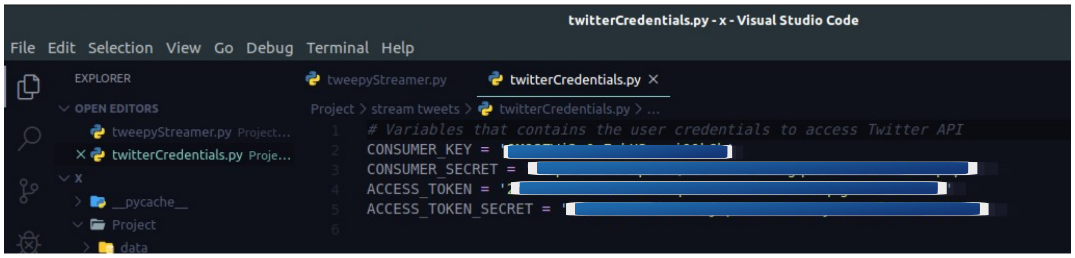
[redacted] (Access token)

[redacted] token secret)

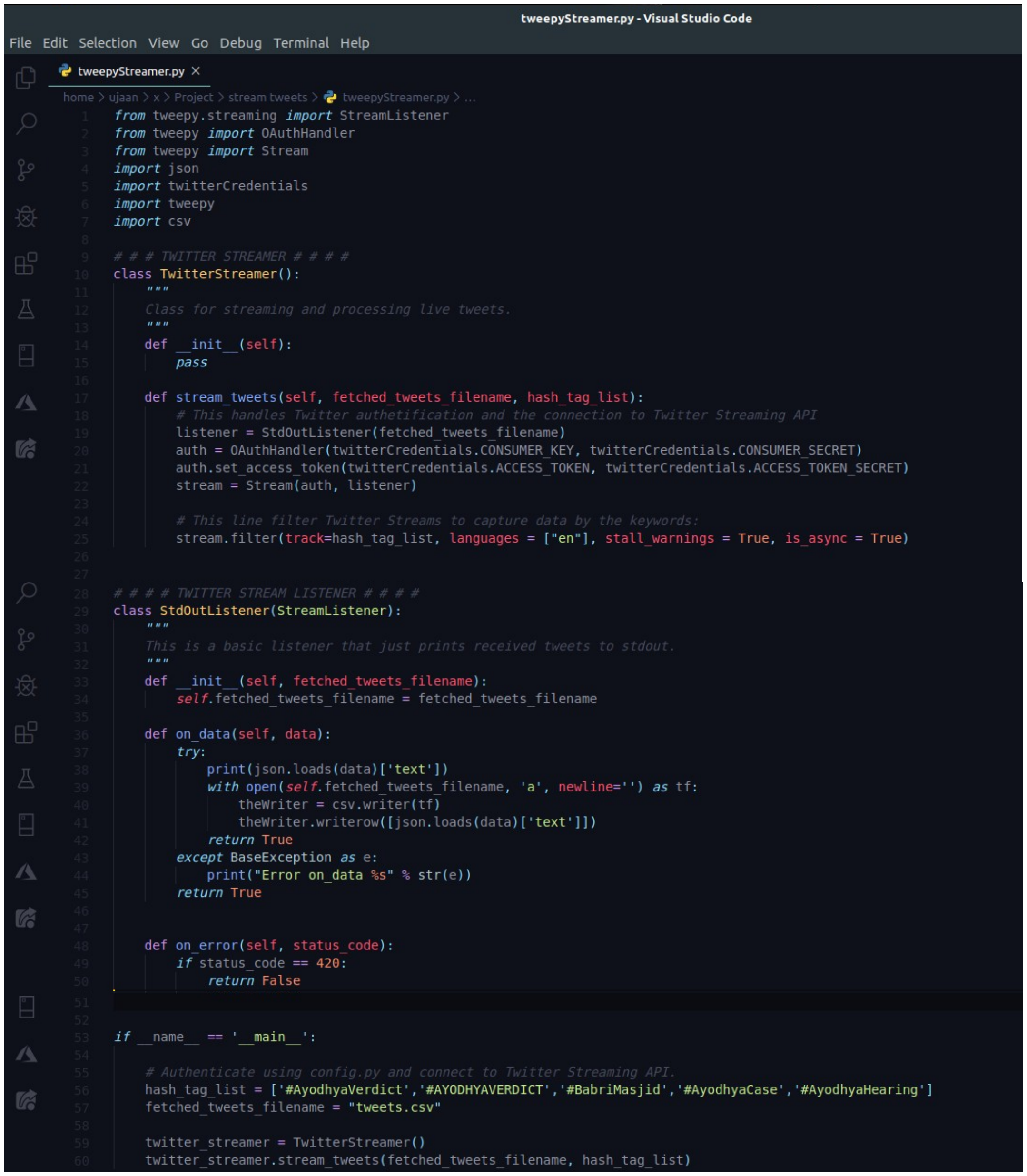
Read and write (Access level)

Revoke Regenerate

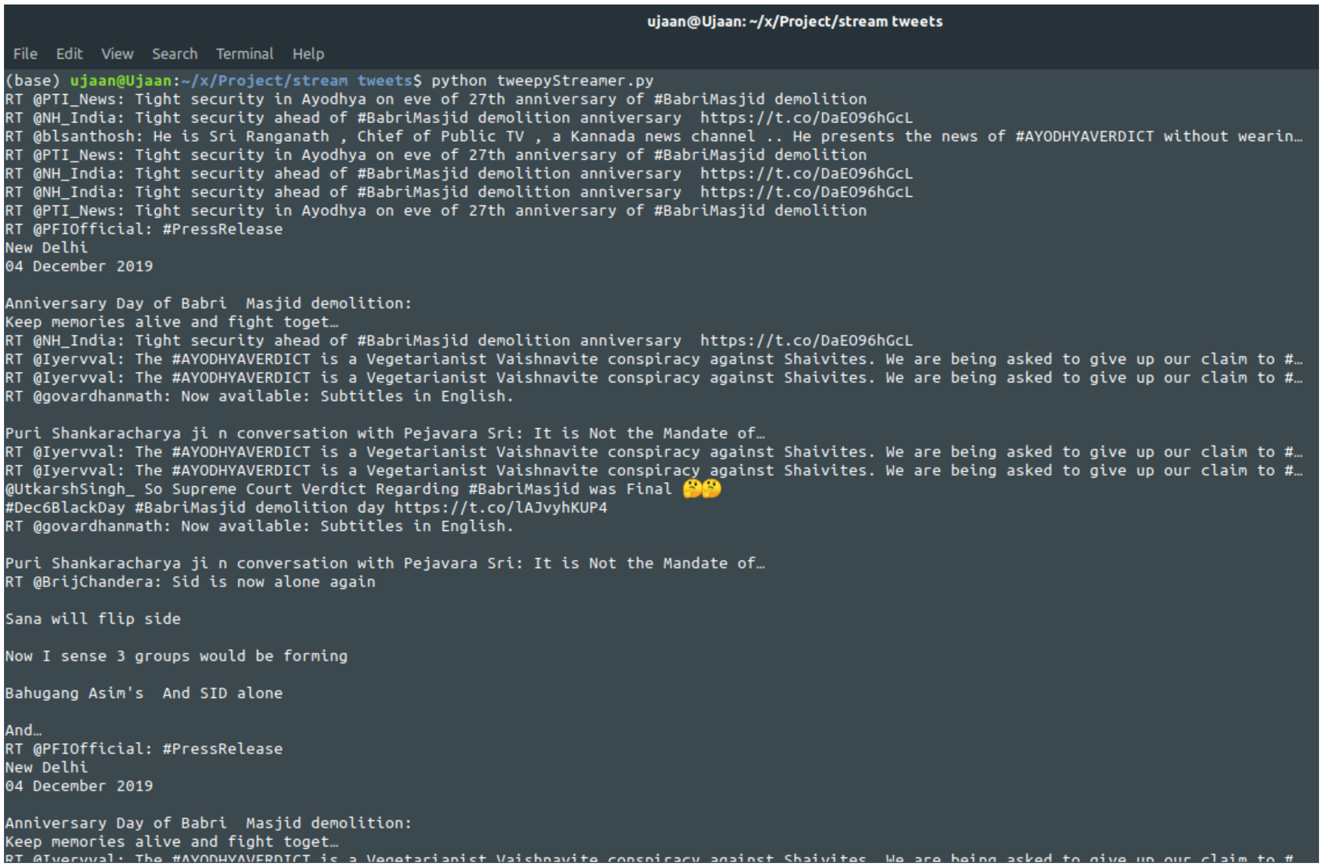
5.2 Streaming Tweets



```python
# Variables that contains the user credentials to access Twitter API
CONSUMER_KEY = [redacted]
CONSUMER_SECRET = [redacted]
ACCESS_TOKEN = [redacted]
ACCESS_TOKEN_SECRET = [redacted]
```

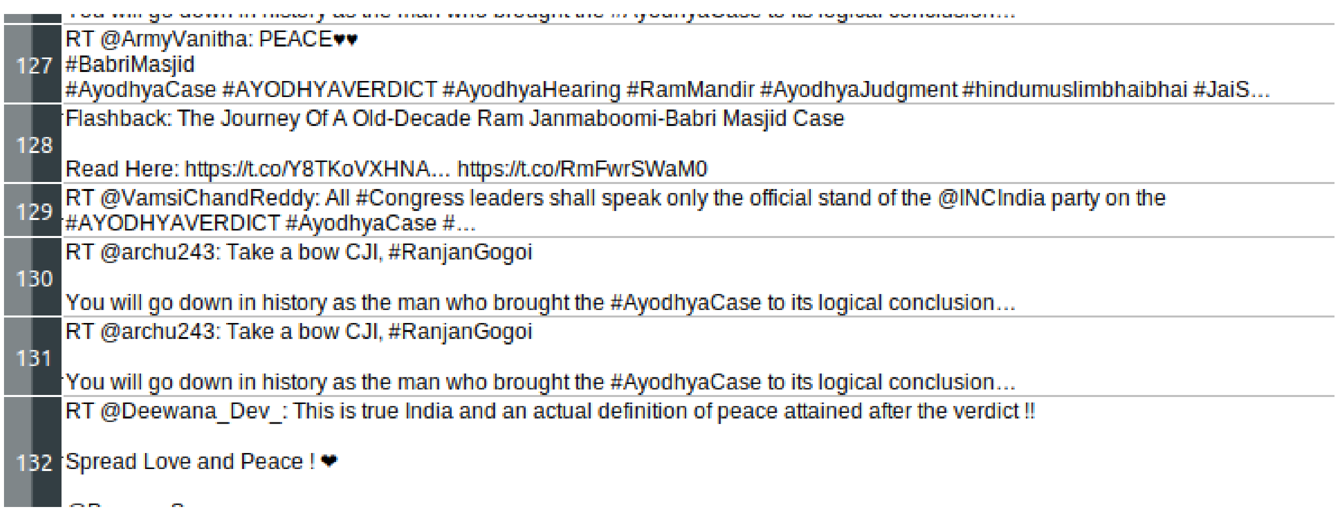


```python
from tweepy.streaming import StreamListener
from tweepy import OAuthHandler
from tweepy import Stream
import json
import twitterCredentials
import tweepy
import csv

# # # TWITTER STREAMER # # # #
class TwitterStreamer():
    """
    Class for streaming and processing live tweets.
    """
    def __init__(self):
        pass

    def stream_tweets(self, fetched_tweets_filename, hash_tag_list):
        # This handles Twitter authetification and the connection to Twitter Streaming API
        listener = StdOutListener(fetched_tweets_filename)
        auth = OAuthHandler(twitterCredentials.CONSUMER_KEY, twitterCredentials.CONSUMER_SECRET)
        auth.set_access_token(twitterCredentials.ACCESS_TOKEN, twitterCredentials.ACCESS_TOKEN_SECRET)
        stream = Stream(auth, listener)

        # This line filter Twitter Streams to capture data by the keywords:
        stream.filter(track=hash_tag_list, languages = ["en"], stall_warnings = True, is_async = True)


# # # # TWITTER STREAM LISTENER # # # #
class StdOutListener(StreamListener):
    """
    This is a basic listener that just prints received tweets to stdout.
    """
    def __init__(self, fetched_tweets_filename):
        self.fetched_tweets_filename = fetched_tweets_filename

    def on_data(self, data):
        try:
            print(json.loads(data)['text'])
            with open(self.fetched_tweets_filename, 'a', newline='') as tf:
                theWriter = csv.writer(tf)
                theWriter.writerow([json.loads(data)['text']])
            return True
        except BaseException as e:
            print("Error on_data %s" % str(e))
        return True


    def on_error(self, status_code):
        if status_code == 420:
            return False


if __name__ == '__main__':

    # Authenticate using config.py and connect to Twitter Streaming API.
    hash_tag_list = ['#AyodhyaVerdict','#AYODHYAVERDICT','#BabriMasjid','#AyodhyaCase','#AyodhyaHearing']
    fetched_tweets_filename = "tweets.csv"

    twitter_streamer = TwitterStreamer()
    twitter_streamer.stream_tweets(fetched_tweets_filename, hash_tag_list)
```



```
(base) ujaan@Ujaan:~/x/Project/stream tweets$ python tweepyStreamer.py
RT @PTI_News: Tight security in Ayodhya on eve of 27th anniversary of #BabriMasjid demolition
RT @NH_India: Tight security ahead of #BabriMasjid demolition anniversary  https://t.co/DaE096hGcL
RT @blsanthosh: He is Sri Ranganath , Chief of Public TV , a Kannada news channel .. He presents the news of #AYODHYAVERDICT without wearin…
RT @PTI_News: Tight security in Ayodhya on eve of 27th anniversary of #BabriMasjid demolition
RT @NH_India: Tight security ahead of #BabriMasjid demolition anniversary  https://t.co/DaE096hGcL
RT @NH_India: Tight security ahead of #BabriMasjid demolition anniversary  https://t.co/DaE096hGcL
RT @PTI_News: Tight security in Ayodhya on eve of 27th anniversary of #BabriMasjid demolition
RT @PFIOfficial: #PressRelease
New Delhi
04 December 2019

Anniversary Day of Babri  Masjid demolition:
Keep memories alive and fight toget…
RT @NH_India: Tight security ahead of #BabriMasjid demolition anniversary  https://t.co/DaE096hGcL
RT @Iyervval: The #AYODHYAVERDICT is a Vegetarianist Vaishnavite conspiracy against Shaivites. We are being asked to give up our claim to #…
RT @Iyervval: The #AYODHYAVERDICT is a Vegetarianist Vaishnavite conspiracy against Shaivites. We are being asked to give up our claim to #…
RT @govardhanmath: Now available: Subtitles in English.

Puri Shankaracharya ji n conversation with Pejavara Sri: It is Not the Mandate of…
RT @Iyervval: The #AYODHYAVERDICT is a Vegetarianist Vaishnavite conspiracy against Shaivites. We are being asked to give up our claim to #…
RT @Iyervval: The #AYODHYAVERDICT is a Vegetarianist Vaishnavite conspiracy against Shaivites. We are being asked to give up our claim to #…
@UtkarshSingh_ So Supreme Court Verdict Regarding #BabriMasjid was Final 🤔🤔
#Dec6BlackDay #BabriMasjid demolition day https://t.co/lAJvyhKUP4
RT @govardhanmath: Now available: Subtitles in English.

Puri Shankaracharya ji n conversation with Pejavara Sri: It is Not the Mandate of…
RT @BrijChandera: Sid is now alone again

Sana will flip side

Now I sense 3 groups would be forming

Bahugang Asim's  And SID alone

And…
RT @PFIOfficial: #PressRelease
New Delhi
04 December 2019

Anniversary Day of Babri  Masjid demolition:
Keep memories alive and fight toget…
```

5.3 Collected Data

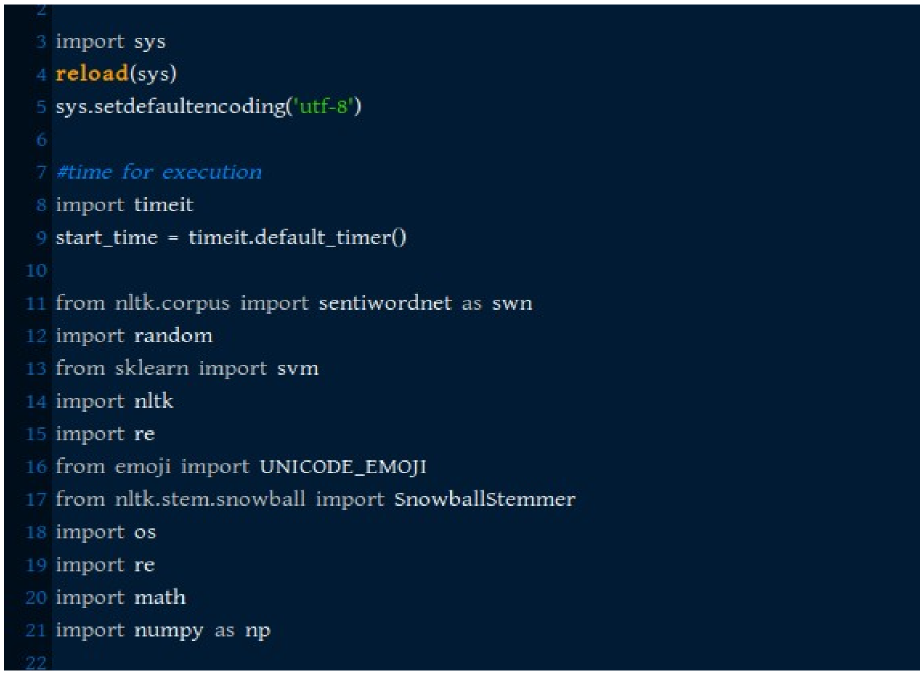


	You will go down in history as the man who brought the #AyodhyaCase to its logical conclusion…
127	RT @ArmyVanitha: PEACE♥♥ #BabriMasjid #AyodhyaCase #AYODHYAVERDICT #AyodhyaHearing #RamMandir #AyodhyaJudgment #hindumuslimbhaibhai #JaiS…
128	Flashback: The Journey Of A Old-Decade Ram Janmaboomi-Babri Masjid Case Read Here: https://t.co/Y8TKoVXHNA… https://t.co/RmFwrSWaM0
129	RT @VamsiChandReddy: All #Congress leaders shall speak only the official stand of the @INCIndia party on the #AYODHYAVERDICT #AyodhyaCase #…
130	RT @archu243: Take a bow CJI, #RanjanGogoi You will go down in history as the man who brought the #AyodhyaCase to its logical conclusion…
131	RT @archu243: Take a bow CJI, #RanjanGogoi You will go down in history as the man who brought the #AyodhyaCase to its logical conclusion…
132	RT @Deewana_Dev_: This is true India and an actual definition of peace attained after the verdict !! Spread Love and Peace ! ❤

Chapter 6

MODEL BUILDING

6.1 Importing Libraries



```
import sys
reload(sys)
sys.setdefaultencoding('utf-8')

#time for execution
import timeit
start_time = timeit.default_timer()

from nltk.corpus import sentiwordnet as swn
import random
from sklearn import svm
import nltk
import re
from emoji import UNICODE_EMOJI
from nltk.stem.snowball import SnowballStemmer
import os
import re
import math
import numpy as np
```

sys — System-specific parameters and functions. This module provides access to some variables used or maintained by the interpreter and to functions that interact strongly with the interpreter. It is always available. The list of command line arguments passed to a Python script.

Timeit() — method is available with python library timeit. It is used to get the execution time taken for the small code given. The library runs the code statement 1 million times and provides the minimum time taken from the set. It is a useful method that helps in checking the performance of the code.

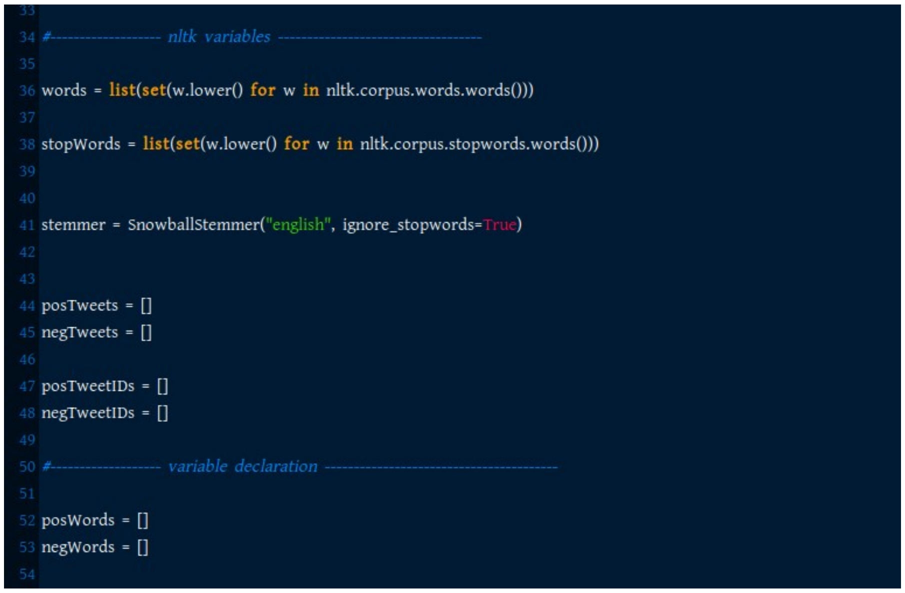
nltk — Natural Language Processing with Python provides a practical introduction to programming for language processing. Written by the creators of **NLTK**, it guides the reader through the fundamentals of writing Python programs, working with corpora, categorizing text, analyzing linguistic structure.

random — You can generate random numbers in Python by using random module. Python offers random module that can generate random numbers. These are pseudo-random number as the sequence of number generated depends on the seed. If the seeding value is same, the sequence will be the same.

re — A regular expression (or RE) specifies a set of strings that matches it; the functions in this module let you check if a particular string matches a given regular expression (or if a given regular expression matches a particular string, which comes down to the same thing).

Sklearn — Scikit-learn is an open source Python library that has powerful tools for data analysis and data mining. It's available under the BSD license and is built on the following machine learning libraries: NumPy, a library for manipulating multi-dimensional arrays and matrices.
os — The OS module in python provides functions for interacting with the operating system. OS, comes under Python's standard utility modules. This module provides a portable way of using operating system dependent functionality.
math — The Python Math Library provides us access to some common math functions and constants in Python, which we can use throughout our code for more complex mathematical computations. The library is a built-in Python module, therefore you don't have to do any installation to use it.
numpy — Numpy. Numpy is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays.

6.2 Variable Declaration



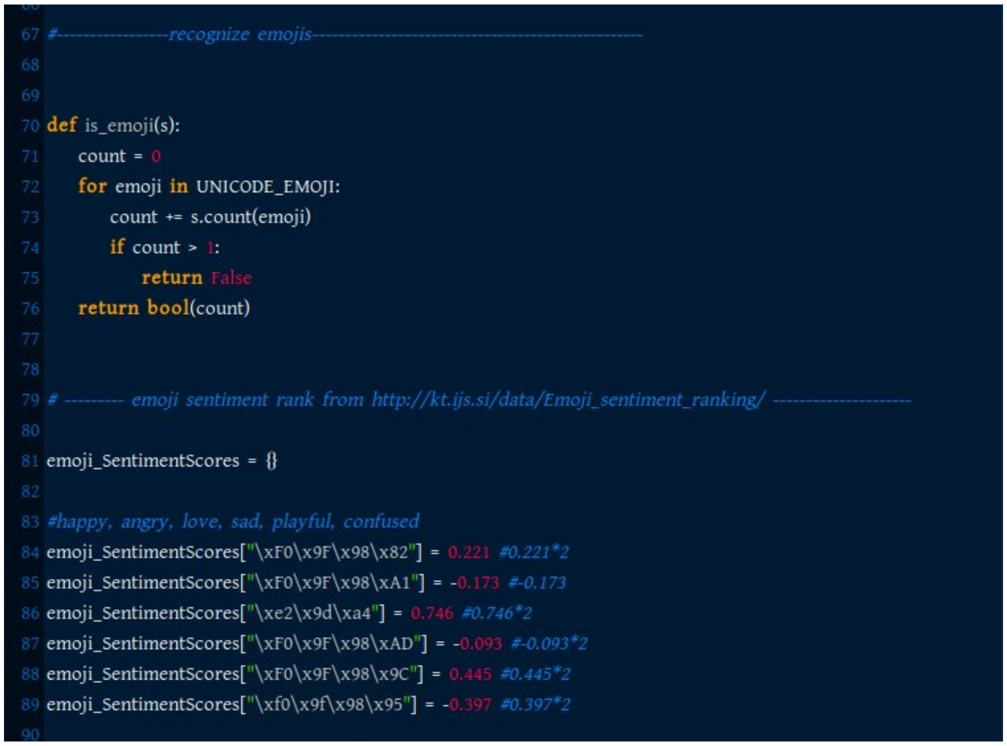
```
#------------------- nltk variables -----------------------------------

words = list(set(w.lower() for w in nltk.corpus.words.words()))

stopWords = list(set(w.lower() for w in nltk.corpus.stopwords.words()))


stemmer = SnowballStemmer("english", ignore_stopwords=True)


posTweets = []
negTweets = []

posTweetIDs = []
negTweetIDs = []

#------------------- variable declaration ----------------------------------------

posWords = []
negWords = []

```

words and stopwords - Stopwords are the most common words in any natural language. For the purpose of analyzing text data and building NLP models, these stopwords might not add much value to the meaning of the document. Generally, the most common words used in a text are “the”, “is”, “in”, “for”, “where”, “when”, “to”, “at” etc.

posWords – This is to store the words which are responcible for positive solidarity.

negWords - This is to store the words which are responcible for negative solidarity.

6.3 Recognizing Emojis And Setting Sentiment Ranking

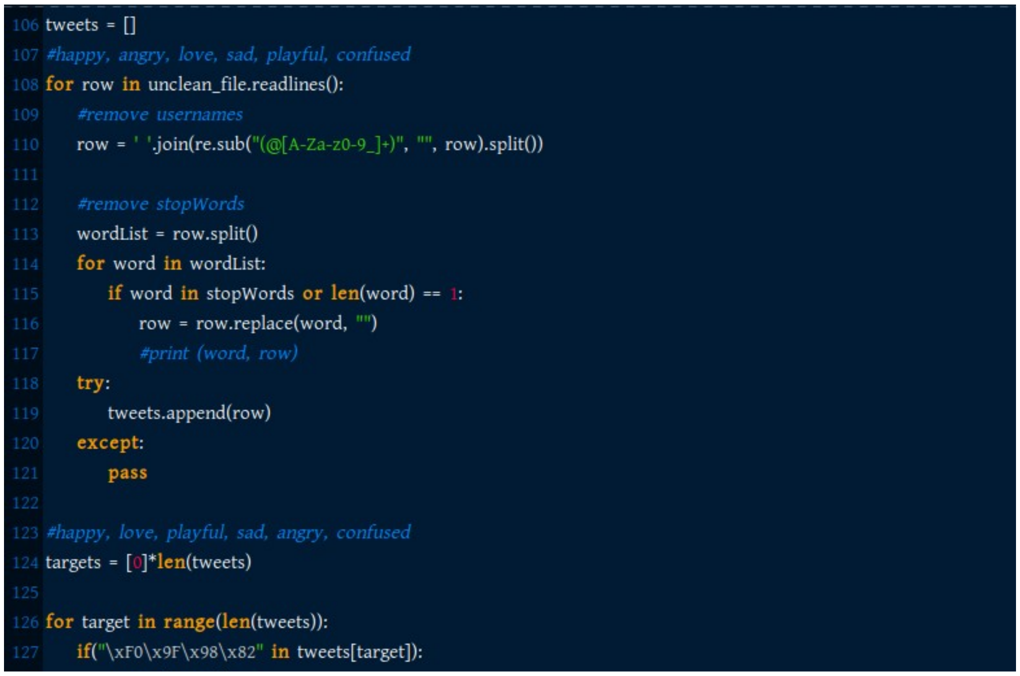


```python
#----------------recognize emojis---------------------------------------------------


def is_emoji(s):
    count = 0
    for emoji in UNICODE_EMOJI:
        count += s.count(emoji)
        if count > 1:
            return False
    return bool(count)


# --------- emoji sentiment rank from http://kt.ijs.si/data/Emoji_sentiment_ranking/ --------------------

emoji_SentimentScores = {}

#happy, angry, love, sad, playful, confused
emoji_SentimentScores["\xF0\x9F\x98\x82"] = 0.221 #0.221*2
emoji_SentimentScores["\xF0\x9F\x98\xA1"] = -0.173 #-0.173
emoji_SentimentScores["\xe2\x9d\xa4"] = 0.746 #0.746*2
emoji_SentimentScores["\xF0\x9F\x98\xAD"] = -0.093 #-0.093*2
emoji_SentimentScores["\xF0\x9F\x98\x9C"] = 0.445 #0.445*2
emoji_SentimentScores["\xf0\x9f\x98\x95"] = -0.397 #0.397*2
```

is_emojis – To recognize the emoticons form the tweets we have used this function.

emoji_SentimentScores – After finding the emojis we have calculated and stored the **Sentiment Score** for each emoticons into this python dictionary. We have taken **Sentiment Ranking** of emoticons from a third party website given below: https://ktijs.si/data/Emoji_sentiment_ranking/

6.4 Data Pre-Processing: Removing stopwords, storing hashtags, emojis and computing sentiment scores



```
tweets = []
#happy, angry, love, sad, playful, confused
for row in unclean_file.readlines():
    #remove usernames
    row = ' '.join(re.sub("(@[A-Za-z0-9_]+)", "", row).split())

    #remove stopWords
    wordList = row.split()
    for word in wordList:
        if word in stopWords or len(word) == 1:
            row = row.replace(word, "")
            #print (word, row)
    try:
        tweets.append(row)
    except:
        pass

#happy, love, playful, sad, angry, confused
targets = [0]*len(tweets)

for target in range(len(tweets)):
    if("\xF0\x9F\x98\x82" in tweets[target]):
```

Data preprocessing is a data mining technique which is used to transform the raw data in a useful and efficient format.

Steps Involved in Data Preprocessing:

1. Data Cleaning:

The data can have many irrelevant and missing parts. To handle this part, data cleaning is done. It involves handling of missing data, noisy data etc.

(a). Missing Data:

This situation arises when some data is missing in the data. It can be handled in various ways.

Some of them are:

Ignore the tuples:

This approach is suitable only when the dataset we have is quite large and multiple values are missing within a tuple.

Fill the Missing values:

There are various ways to do this task. You can choose to fill the missing values manually, by attribute mean or the most probable value.

(b). Noisy Data:

Noisy data is a meaningless data that can’t be interpreted by machines.It can be generated due to faulty data collection, data entry errors etc. It can be handled in following ways :

Binning Method:

This method works on sorted data in order to smooth it. The whole data is divided into segments of equal size and then various methods are performed to complete the task. Each segmented is handled separately. One can replace all data in a segment by its mean or boundary values can be used to complete the task.

Regression:

Here data can be made smooth by fitting it to a regression function.The regression used may be linear (having one independent variable) or multiple (having multiple independent variables).

Clustering:

This approach groups the similar data in a cluster. The outliers may be undetected or it will fall outside the clusters.

2. Data Transformation:

This step is taken in order to transform the data in appropriate forms suitable for mining process. This involves following ways:

Normalization:

It is done in order to scale the data values in a specified range (-1.0 to 1.0 or 0.0 to 1.0)

Attribute Selection:

In this strategy, new attributes are constructed from the given set of attributes to help the mining process.

Discretization:

This is done to replace the raw values of numeric attribute by interval levels or conceptual levels.

Concept Hierarchy Generation:

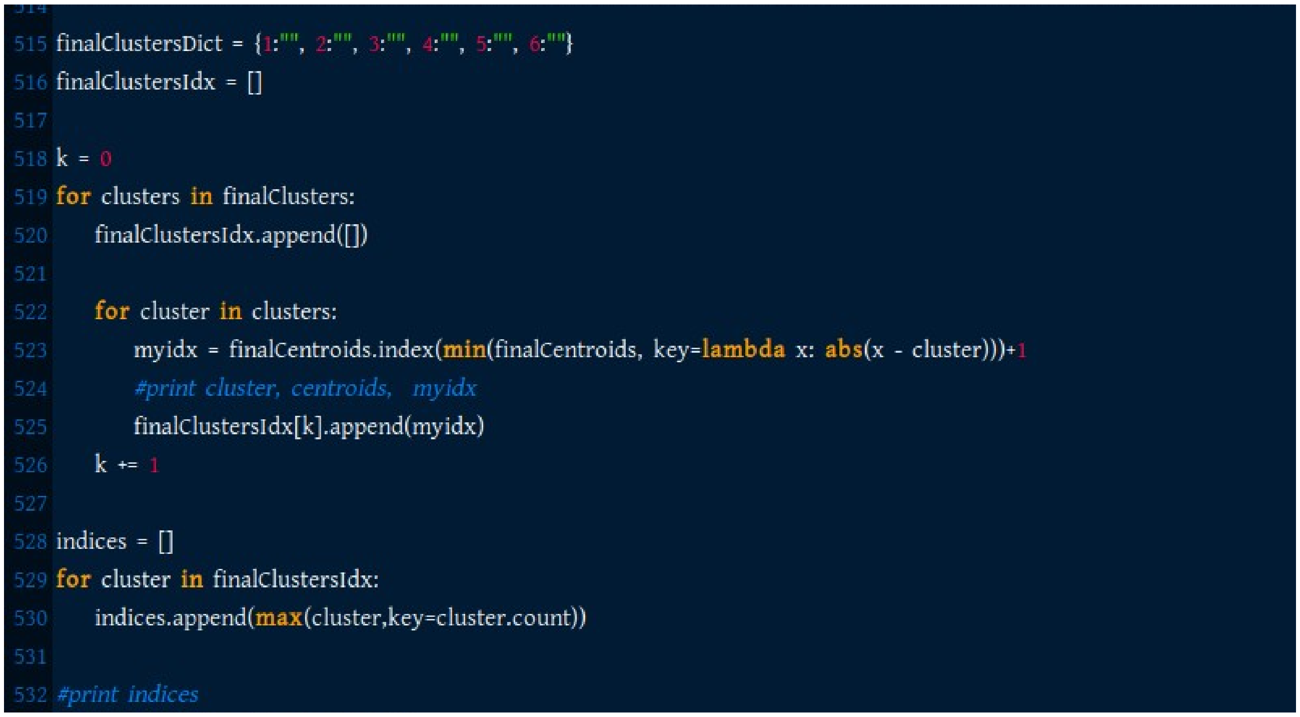
Here attributes are converted from level to higher level in hierarchy. For Example-The attribute “city” can be converted to “country”.

3. Data Reduction:

Since data mining is a technique that is used to handle huge amount of data. While working with huge volume of data, analysis became harder in such cases. In order to get rid of this, we uses data reduction technique. It aims to increase the storage efficiency and reduce data storage and analysis costs.

The various steps to data reduction are:
Data Cube Aggregation:
Aggregation operation is applied to data for the construction of the data cube.
Attribute Subset Selection:
The highly relevant attributes should be used, rest all can be discarded. For performing attribute selection, one can use level of significance and p- value of the attribute.the attribute having p-value greater than significance level can be discarded.
Numerosity Reduction:
This enable to store the model of data instead of whole data, for example: Regression Models.
Dimensionality Reduction:
This reduce the size of data by encoding mechanisms.It can be lossy or lossless. If after reconstruction from compressed data, original data can be retrieved, such reduction are called lossless reduction else it is called lossy reduction. The two effective methods of dimensionality reduction are:Wavelet transforms and PCA (Principal Componenet Analysis).

6.5 Building Clusters



```
finalClustersDict = {1:"", 2:"", 3:"", 4:"", 5:"", 6:""}
finalClustersIdx = []

k = 0
for clusters in finalClusters:
    finalClustersIdx.append([])

    for cluster in clusters:
        myidx = finalCentroids.index(min(finalCentroids, key=lambda x: abs(x - cluster)))+1
        #print cluster, centroids,  myidx
        finalClustersIdx[k].append(myidx)
    k += 1

indices = []
for cluster in finalClustersIdx:
    indices.append(max(cluster,key=cluster.count))

#print indices
```

This module contains a number of basic clustering algorithms. Clustering describes the task of discovering groups of similar items with a large collection. It is also describe as unsupervised machine learning, as the data from which it learns is unannotated with class information, as is the case for supervised learning. Annotated data is difficult and expensive to obtain in the quantities required for the majority of supervised learning algorithms. This problem, the knowledge acquisition bottleneck, is common to most natural language processing tasks, thus fueling the need for quality unsupervised approaches.

This module contains a k-means clusterer, E-M clusterer and a group average agglomerative clusterer (GAAC). All these clusterers involve finding good cluster groupings for a set of vectors in multi-dimensional space.

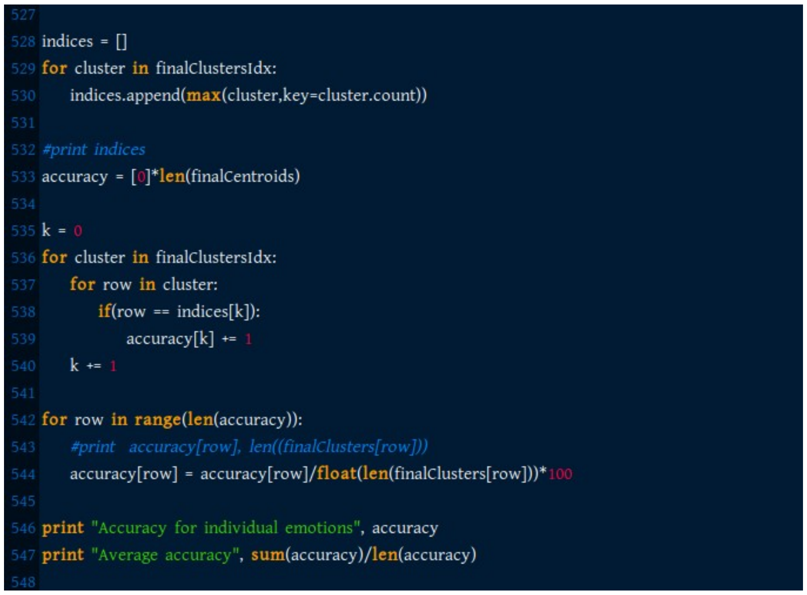
The K-means clusterer starts with k arbitrary chosen means then allocates each vector to the cluster with the closest mean. It then recalculates the means of each cluster as the centroid of the vectors in the cluster. This process repeats until the cluster memberships stabilise. This is a hill-climbing algorithm which may converge to a local maximum. Hence the clustering is often repeated with random initial means and the most commonly occurring output means are chosen.

The GAAC clusterer starts with each of the N vectors as singleton clusters. It then iteratively merges pairs of clusters which have the closest centroids. This continues until there is only one cluster. The order of merges gives rise to a dendrogram - a tree with the earlier merges lower than later merges. The membership of a given number of clusters c, 1 <= c <= N, can be found by cutting the dendrogram at depth c.

The Gaussian EM clusterer models the vectors as being produced by a mixture of k Gaussian sources. The parameters of these sources (prior probability, mean and covariance matrix) are then found to maximise the likelihood of the given data. This is done with the expectation maximisation algorithm. It starts with k arbitrarily chosen means, priors and covariance matrices. It then calculates the membership probabilities for each vector in each of the clusters - this is the 'E' step. The cluster parameters are then updated in the 'M' step using the maximum likelihood estimate from the cluster membership probabilities. This process continues until the likelihood of the data does not significantly increase.

They all extend the ClusterI interface which defines common operations available with each clusterer. These operations include.
cluster: clusters a sequence of vectors
classify: assign a vector to a cluster
classification_probdist: give the probability distribution over cluster memberships
The current existing classifiers also extend cluster.VectorSpace, an abstract class which allows for singular value decomposition (SVD) and vector normalisation. SVD is used to reduce the dimensionality of the vector space in such a manner as to preserve as much of the variation as possible, by reparameterising the axes in order of variability and discarding all bar the first d dimensions. Normalisation ensures that vectors fall in the unit hypersphere.
Usage example (see also demo())::
from nltk import cluster
from nltk.cluster import euclidean_distance
from numpy import array
vectors = [array(f) for f in [[3, 3], [1, 2], [4, 2], [4, 0]]]
# initialise the clusterer (will also assign the vectors to clusters)
clusterer = cluster.KMeansClusterer(2, euclidean_distance)
clusterer.cluster(vectors, True)
# classify a new vector print(clusterer.classify(array([3, 3]))

6.6 Storing Scores: Final Centroids, Accuracy for individual emotions, Average accuracy score



```

indices = []
for cluster in finalClustersIdx:
    indices.append(max(cluster,key=cluster.count))

#print indices
accuracy = [0]*len(finalCentroids)

k = 0
for cluster in finalClustersIdx:
    for row in cluster:
        if(row == indices[k]):
            accuracy[k] += 1
    k += 1

for row in range(len(accuracy)):
    #print  accuracy[row], len((finalClusters[row]))
    accuracy[row] = accuracy[row]/float(len(finalClusters[row]))*100

print "Accuracy for individual emotions", accuracy
print "Average accuracy", sum(accuracy)/len(accuracy)

```

With both forms of clustering, the machine is tasked with receiving a dataset that is just featuresets, and then the machine searches for groups and assigns classes on its own. With Flat clustering, the scientist tells the machine how many classes/clusters to find. With Hierarchical clustering, the machine figures out the groups and how many.
What are some reasons why we might utilize clustering? The objective of clustering is to find relationships and meaning in data. In most cases that I have personally seen, people use clustering mostly as a step in what's known as "semi-supervised" machine learning. The idea here is you might use clustering to define classes, and then use a supervised machine learning algorithm to do further classification. Another use is for feature selection and validation. For example, consider a Breast Cancer dataset that we've been using. We might believe the features we've chosen are indeed descriptive and meaningful. One option we have is to feed this data through a K Means algorithm, and then indeed see whether or not the two groups we thought we were tracking were actually described by the data in a way we expected.
Consider next that you're a data scientist for Amazon. Your CTO has data collected that they believe could be used to predict whether or not customers are buyers or non-buyers. They want you to use K-Means to see if K-Means correctly organizes the customers by the data that the CTO thinks is meaningful.
What about Hierarchical Clustering? Consider you're still the same data scientist for Amazon. This time, you run the seemingly meaningful data through a hierarchical clustering algorithm, such as Mean Shift, and you actually get 5 groups. After further analysis, you realize visitors aren't actually just buyers and non-buyers, they are a spectrum! You really have non-buyers, unlikely-to-buyers, slightly-to-buyers, highly-likely-to-buyers, and certain-to-buyers.
Clustering can also be used on truly unknown data, in attempt to find structure. Consider you're an alien race first discovering human texts in North America. You might take all of the written characters you find and compile them into a big list of features. Then you might feed this list through a Hierarchical Clustering algorithm just to see if you can find specific groups so that you can start to decode the language by characters.
The field of "Big Data Analysis" is generally a prime area for clustering. There's plenty of data, but most companies have absolutely no idea what to do with it or how to actually get meaning from it. Clustering can help data scientists to begin to structure and look into expansive datasets for meaning.

Finally, clustering can also be used for typical classification, you just don't actually need to feed it what the classifications are beforehand, but, if you use clustering on most of the popular classification datasets, you should find that it tends to figure out the groups anyway. This is mainly useful for "testing" the algorithm to make sure it actually works.

Our first algorithm to cover is the K-Means algorithm. The idea of K-Means is to attempt to cluster a given dataset into K clusters. The way it works is actually pretty impressive, and, luckily for us, quite simple. The process goes as follows:

Take entire dataset, and set, randomly, K number of centroids. Centroids are just the "centers" of your clusters. To start, I typically just take the first K values in my dataset and have those as the start, but you could also randomly select them if you wanted. It should not matter, but, if you are not optimizing for some reason, it may make sense to try to shuffle the data and try again.
Calculate distance of each featureset to the centroids, and classify each featureset as the centroid class closest to it. Centroid classes are arbitrary, you will likely just call the first centroid 0, the second centroid 1...and so on.
Once you have classified all data, now you take the "mean" of the groups, and set the new centroids as the mean of their associated groups.
Repeat #2 and #3 until you are optimized. Typically, you measure optimization by movement of the centroid. There are many ways to do this, we're just going to use percent change.

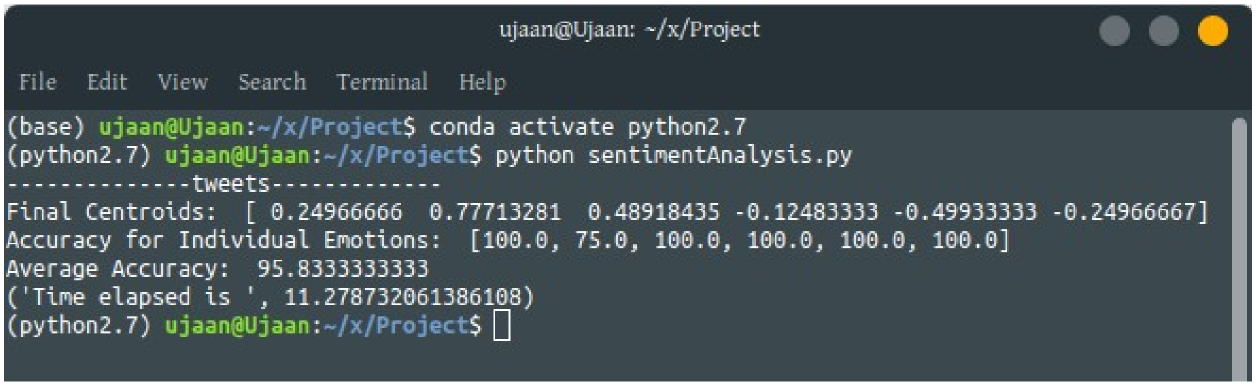
Chapter 7

RESULTS

1. Final Centroids:

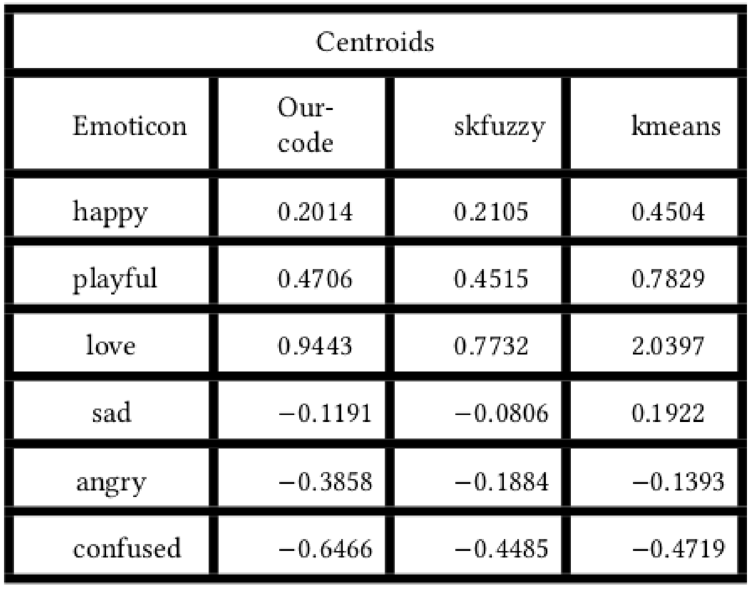
   [0.24966666, 0.77713281, 0.48918435, -0.12483333, -0.49933333, -0.24966667]
2. Accuracy for Individual Emotions:

   [100.0, 75.0, 100.0, 100.0, 100.0, 100.0]
3. Average Accuracy: 95.8333333333



```
(base) ujaan@Ujaan:~/x/Project$ conda activate python2.7
(python2.7) ujaan@Ujaan:~/x/Project$ python sentimentAnalysis.py
--------------tweets-------------
Final Centroids:  [ 0.24966666  0.77713281  0.48918435 -0.12483333 -0.49933333 -0.24966667]
Accuracy for Individual Emotions:  [100.0, 75.0, 100.0, 100.0, 100.0, 100.0]
Average Accuracy:  95.8333333333
('Time elapsed is ', 11.278732061386108)
(python2.7) ujaan@Ujaan:~/x/Project$ 
```

Fuzzy C-Means is also referred to as soft K-Means. In skfuzzy, Fuzzy K-Means is equal to FCM for 1-dimensional data. We take the average of centroids over 10 runs for all the cases to get the following centroids table for our FCM code vs skfuzzy FCM and kmeans.

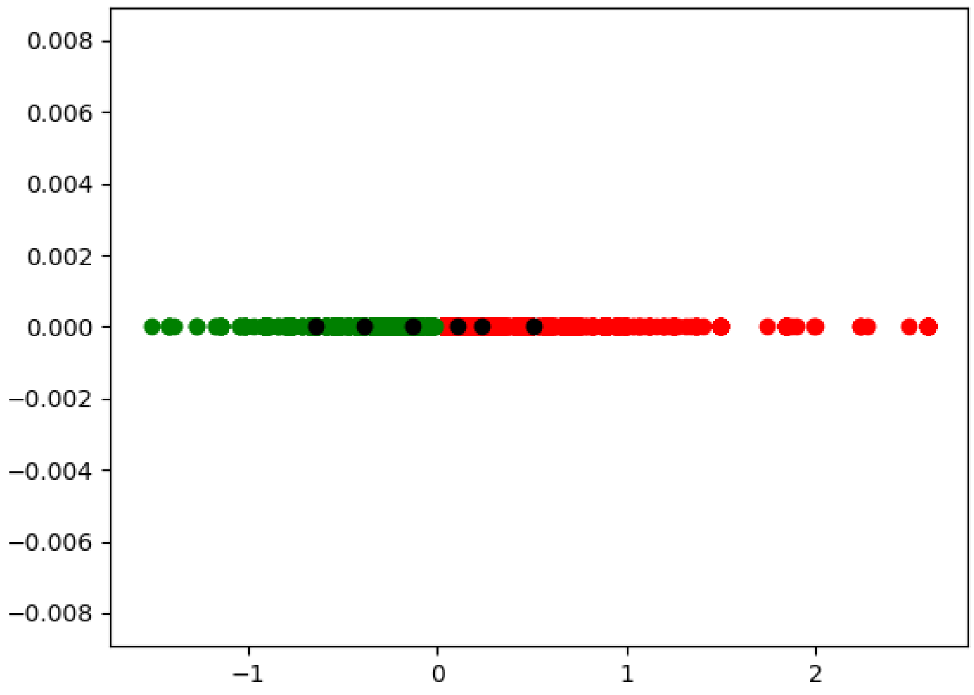


Centroids			
Emoticon	Our-code	skfuzzy	kmeans
happy	0.2014	0.2105	0.4504
playful	0.4706	0.4515	0.7829
love	0.9443	0.7732	2.0397
sad	−0.1191	−0.0806	0.1922
angry	−0.3858	−0.1884	−0.1393
confused	−0.6466	−0.4485	−0.4719

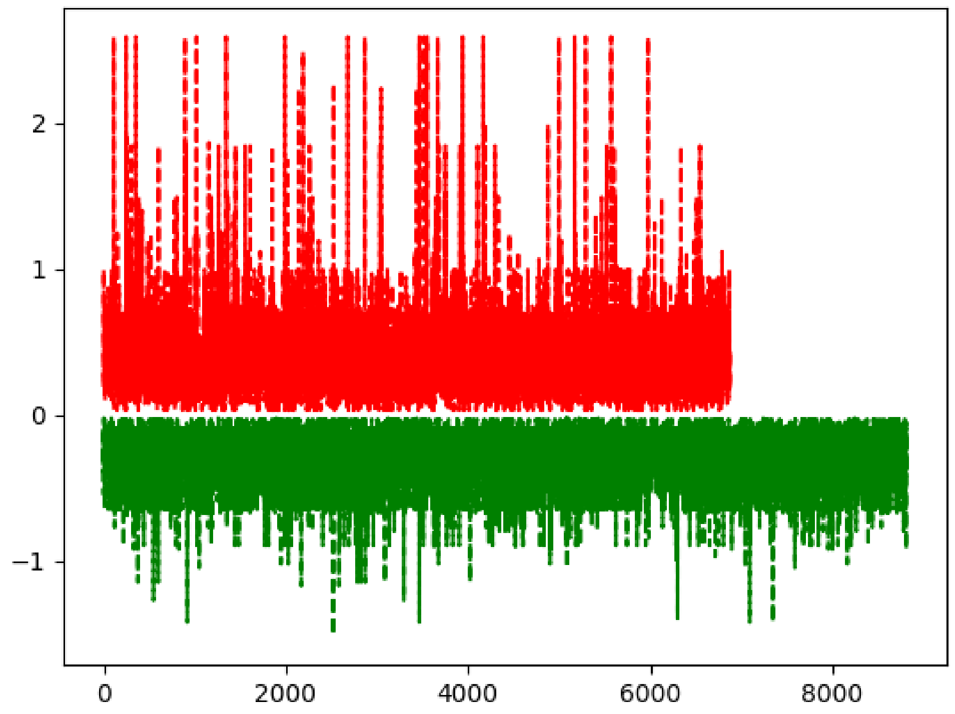
Chapter 8

DATA VISUALIZATION

1. Final Centroids:



2. Positive Tweets vs Negative Tweets:



Above figure shows the distribution of positive and negative tweets, which gives a better idea about the variance between the tweet sentiment scores. For figures 1, 2 the black dots from left to right represent the centroids for each emoticon in the order – con- fused, angry, sad, happy, playful and love. The figures show the variation in the final centroids obtained by using the corre- sponding technique. We see that our SVM works almost as well as skfuzzy FCM, while K-means doesn’t perform as well as the other 2 methods. The accuracy table shows the maximum frequency of accuracy over 50 runs of each code.

Chapter 9

FUTURE SCOPE

Future Scope

1. Ambiguity of text and emojis can be lowered if more possibilities are taken into consideration.
2. We will use the output data in various type of Machine Learning models for various type of analysis like Text Analysis, Sentiment Analysis, Solidarity Analysis.

Chapter 10

CONCLUSION

Conclusion

Our analysis will reveal that emojis are a powerful indicator of sociolinguistic behaviors (solidarity) that are exhibited on social media as the crisis events unfold.

An additional future goal is to analyze the interaction of sentiment of emojis and solidarity as well as the text that ooccurs with these emojis in further detail.

Chapter 11

REFERENCES

References

1. https://www.analyticsvidhya.com/
2. https://www.kaggle.com/
3. http://www.datasciencecentral.com/
4. https://towardsdatascience.com/
5. Bac Le and Huy Nguyen, Twitter Sentiment Analysis Using Machine Learning Techniques, Advances in Intelligent Systems and Computing book series, Volume 358, 2015, 978-3-319-17996-4
6. Ali Hasan, Sana Moin, Ahmad Karim and Shahaboddin Shamshirband, Machine Learning-Based Sentiment Analysis for Twitter Accounts, Mathematical Computational Application, Volume 23, 27 February 2018, **https://doi.org/10.3390/mca23010011**