Sentiment\_analyzer.py:

import nltk,yaml

class Splitter(object):

def \_\_init\_\_(self)

def split(self, text)

class POSTagger(object):

def \_\_init\_\_(self)

def pos\_tag(self, sentences)

class DictionaryTagger(object):

def \_\_init\_\_(self, dictionary\_paths)

def tag(self, postagged\_sentences)

def tag\_sentence(self, sentence, tag\_with\_lemmas=False)

def value\_of(sentiment)

def sentence\_score(sentence\_tokens, previous\_token, acum\_score)

def sentiment\_score(review)

Twitter.py:

from tweepy import Stream, OAuthHandler

from tweepy.streaming import StreamListener

import time, urllib, re

from textblob import TextBlob

from sentiment\_analyzer import Splitter, POSTagger, DictionaryTagger, value\_of

from sentiment\_analyzer import sentence\_score, sentiment\_score

ckey = 'xxxxxxxxxxxxxxxxxxxxxxxx'

csecret = 'xxxxxxxxxxxxxxxxxxxxxxxxxxx'

atoken = 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx’

asecret = 'yyyyyyyyyyyyyyyyyyyyyy'

def get\_sentiment(tweet)

class listener(StreamListener):

def \_\_init\_\_(self)

def on\_data(self, data):

quit\_if\_limit\_reached()

text = extract\_tweet()

splitter = Splitter()

postagger = POSTagger()

splitted\_sentences = splitter.split(text)

pos\_tagged\_sentences = postagger.pos\_tag(splitted\_sentences)

dicttagger = DictionaryTagger(dictionary\_paths)

dict\_tagged\_sentences = dicttagger.tag(pos\_tagged\_sentences)

senti = sentiment\_score(dict\_tagged\_sentences)

analyse\_other\_tweet\_data()

try:

store\_and\_print\_result()

except BaseException:

print('Failed')

time.sleep(5)

return False

return True

def on\_error(self, status):

print(status)

auth = OAuthHandler(ckey, csecret)

auth.set\_access\_token(atoken, asecret)

l = listener()

twitterStream = Stream(auth, l)

twitterStream.filter(track=['Modi'])