KnowledgeBase.py:

import nltk

import re

import time

try:

import urllib.request as urllib2

except ImportError:

import urllib2

try:

from urllib.request import urlopen

except ImportError:

from urllib2 import urlopen

import http.cookiejar as cookielib

from http.cookiejar import CookieJar

import datetime

import sqlite3

import yaml

cj = CookieJar()

opener = urllib2.build\_opener(urllib2.HTTPCookieProcessor(cj))

opener.addheaders = [('User-agent', 'Chrome/61.0')]

conn = sqlite3.connect('knowledgeBase.db')

c = conn.cursor()

site = 'http://www.imdb.com'

contentArray = [ ]

ratingArray = []

def loadDicts(dictionary\_paths)

def IMDBcontent()

def buildWordBase(dictionary)

def buildMultiplierBase(dictionary)

dict\_paths = [ 'dicts/pos\_dict.yml', 'dicts/newpos.yml', 'dicts/pos.yml', 'dicts/pos2.yml',

'dicts/neg\_dict.yml', 'dicts/newneg.yml']

multiplier\_dict = ['dicts/inc.yml', 'dicts/dec.yml', 'dicts/inv.yml']

#dictionary = loadDicts(multiplier\_dict)

#buildMultiplierBase(dictionary)

IMDBcontent()

site = 'https://www.amazon.in'

aSearch = 'https://www.amazon.in/s/ref=nb\_sb\_noss\_2?url=search-alias%3Daps&field-keywords='

keyword = 'phones'

def amazonContent()

#amazonContent()

def buildBase(polarity, content)

def processContent():

processContent()

features.py:

import re

import json

class Emoticons:

def analyse(self, string)

def repairString(string)

sentiment\_analyzer.py:

import nltk,yaml

from emo import split\_emo

import datetime

import sqlite3

import features

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)

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\_analyser import Splitter, POSTagger, DictionaryTagger, value\_of

from sentiment\_analyser import sentence\_score, sentiment\_score

import features

ckey = ‘xxxxxxxxxxxxxxkxxxxxcccccccc’

csecret = ‘ccccccccccccccccaaaaaaaaaassssssssssssss’

atoken = ‘aaaaaaaaaaaaaa-ttttttttttttttttttttttttttttttttttt’

asecret = ‘aaaaaaaaaaaaaaaaaaaasssssssssssssssssssssssssxx’

def get\_sentiment(tweet)

class listener(StreamListener):

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

def on\_data(self, data):

print\_final\_result\_and\_quit\_if\_limit\_reached()

text = str(tweet)

splitter = Splitter()

postagger = POSTagger()

emoji = features.Emoticons()

emos = emoji.analyse(text)

text = features.repairString(text)

splitted\_sentences = splitter.split(text)

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

dicttagger = DictionaryTagger()

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

senti = sentiment\_score(dict\_tagged\_sentences) + emos['positive'] - emos['negative']

analyse\_other\_data()

try:

storing\_and\_printing\_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'])