Complex Event Processing

Twitter Sentiment and Emotion Analysis using Python

Project By: Indraneel Pise Instructor: Ankit Jain

Acknowledgements: Ankit Jain, Tushar Vetal, Priyadarshi Sharma, Rijo Thomas

**Abstract:**

The project aims at identifying a complex event based on the data collected from twitter with a keyword search and analysing the data to display results. The results include the sentiment polarity ie. positive, negative or neutral. The sentiment will also be rated on a scale of five. It also shows the emotion levels of the event classifying it based on six basic emotions viz. joy, sadness, anger, fear, surprise and disgust.

**Project Requirements:**

Python 2.7 with packages numpy, pandas, scipy, matplotlib, csv, json, twitter, math, re, nltk and os preinstalled.

Working twitter account with API access codes.

Preferable OS: Windows 7 or higher.

**Contents:**

1. Setting up the live stream from twitter.
2. Identify and display the real-time trends(Worldwide and India).
3. Filtering the live stream based on a keyword.
4. Store the data temporarily to analyze.
5. Implementing the bag of words text mining technique on the temporarily stored data.

Bag of words includes :

* Preprocessing.
* Creating a feature vector.
* Applying term frequency to the feature vector.

1. Analysis:

* Using the Natural Language Processing Toolkit(NLTK) to attach sentiment and emotional quotient. Emotional quotient includes 6 emotions viz. Joy, Sadness, Fear, Anger, Surprise and Disgust.
* Calculating the sentiment of each tweet based on the feature vector of that tweet.
* Calculating the overall emotion level of the dataset.

1. Displaying the results:

* Percentage of tweets having positive, negative and neutral sentiment.
* Dividing the sentiment on a scale of 5.
* Levels of all the six emotions in the dataset.
* The respective graphs of the above three measures.

1. Future Work
2. Bibliography
3. Python Code

**1. Setting up the Twitter Live Stream**

Twitter provides its API to all its users.

We need to create a twitter app that allocates the codes required to access the API.

Python has a library called “twitter” to access the API.

The data returned is in the json format.

Sample Tweet Data(Single Tweet) looks like this:



**2. Displaying the World and India Trends**

Twitter allows us to access the trends based on the geographical location.

Every location has a particular WOEID (where on earth id).

The WOEID for the world is 1 and for India is 2295420.

**3. Filtering the Stream Based on Keyword**

The keyword based search helps to fetch the data related to the event we want to analyse.

It returns the data of tweets having the keyword in the json format.

We only extract the actual content of the tweet and ignore the metadata.

Store the data temporarily in a text file.

**4. Store the Data Temporarily to Analyze**

The data extracted from twitter will be stored in a temporary text file ‘output.txt’.

The number of tweets extracted depends on the number requested and current traffic for the keyword.

Generally analyzing 100-500 tweets will give correct results. Less than 100 tweets cause overfitting and more than 500 tweets are not necessary. Also twitter allows 1000 API calls fifteen minutes so we need to avoid requesting of large number of tweets at a time.

**5. Bag Of Words**

* Preprocessing:

It includes removing the contents that are not useful from the tweet such as URL, retweet identification etc.

All the text is converted to lowercase using regular expression.

Regular expression also helps to remove symbols and numbers which do not help in analysis.

* Creating Feature Vector:

Feature vector is splitting the sentence into words and appending each word into an array. Each word can be viewed as a feature.

Stopwords are also eliminated in this step.

(Stopwords are words that do not help in analysis and occur very frequently such as ‘and’, ‘or’, ‘be’, ‘as’ etc.)

Another important step is stemming of the words. Stemming converts the various forms of words to the word itself.

Eg. ‘Running’, ‘runner’, ‘ran’, ‘runnnnnn’ are all converted to ‘run’.

* Applying the term frequency:

The words which occur more than once are stored only once with a term frequency of number of times they occur.

Term frequency will be used while assigning the sentiments.

**6. Analysis**

* Attaching Sentiment to every Tweet:

We have the tweet as a feature vector with words as features.

Using the NLTK database assign a sentiment to every feature and multiply it by term frequency . Now add the sentiment of every feature to get the overall sentiment of tweet.

The sentiments in the NLTK database range from -4 to +4.

* Counting emotion words:

From the feature vector count the number of words depicting each emotion viz. joy, sadness, fear, anger, surprise and disgust.

These six emotions are widely used to analyse the psychology of a person. It is called “Profile Of Mood States”.

* The databases required for analysis are as follows:

Sentiment – ‘AFINN-111.txt’

Joy – ‘emo\_joy\_s.txt’

Sadness – ‘emo\_sadness\_s.txt’

Fear – ‘emo\_fear\_s.txt’

Anger – ‘emo\_anger\_s.txt’

Surprise – ‘emo\_surprise\_s.txt’

Disgust – ‘emo\_surprise\_s.txt’

All these databases have been processed and stemmed so that they are similar to our tweet corpus.

* Overall Sentiment and Emotion of the Dataset:

Since we have sentiment for every tweet, the overall sentiment will be the average.

To get the rating on a scale of five:

If sentiment<-2\*s.d , then rating=0

If -2\*s.d<=sentiment<-1\*s.d , then rating=1

If -1\*s.d<=sentiment<0 , then rating=2

If 0<=sentiment<s.d , then rating=3

If s.d<=sentiment<2\*s.d , then rating=4

If 2\*s.d<=sentiment, then rating=5

For emotion we calculate :

No. of words of one emotion/Total no. of emotion words.

It will give the percentage of every emotion.

**7. Sample Results**

When the simulation begins:

World Trends are

set([u'#BoringRealityShows', u'Forfun', u'Chris Heston', u'The Warriors', u'#HappyBDayNathanBarone', u'#MitadDel2015YTodavia', u'Burlando', u'#MasterChefBR', u'#TeLoDigoEnUnaCancionDeRock', u'Curry'])

India Trends are

set([u'Essar', u'#IndvsBan', u'Reliance', u'Top Stories', u'#LagGayiVaat', u'#INDIANARMYROCKS', u'Curry', u'#NBAFinals', u'#ModiScaredOfArvind', u'Audi'])

(The ‘u’ before every trend is to be ignored. It only represents that the encoding is UTF8)

Enter a keyword for analysis

The keyword entered is “#IndvsBan”.

Following are the results displayed:

No. of tweets analysed 60

The percentage of tweets that are joyful is 16.1290322581

The percentage of tweets that are sad is 12.9032258065

The percentage of tweets that are angry is 0.0

The percentage of tweets that show surprise is 3.22580645161

The percentage of tweets that show disgust is 0.0

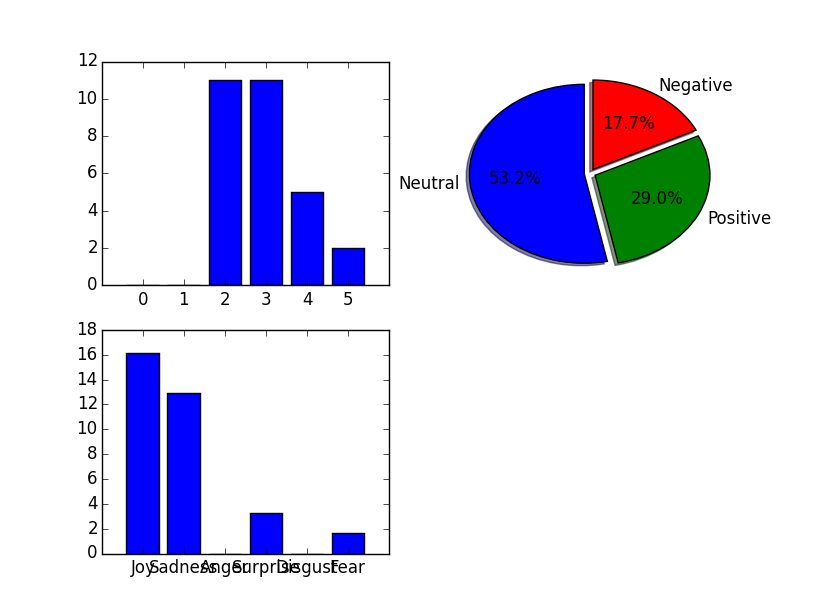
The percentage of tweets that are fearful is 1.61290322581

Percentage of tweets having positive sentiment is 29.0322580645

Percentage of tweets having negative sentiment is 17.7419354839

Percentage of tweets having neutral sentiment is 53.2258064516

The average rating of people's sentiment out of five is 2.93103448276



The 1st graph is a bar-chart that shows the rating on the scale of five.

The 2nd graph is a pie-chart that shows the distribution of tweets based on sentiment positive, negative or neutral.

The 3rd graph is a bar-chart showing the proportions of the six emotions.

**8. Future Work**

Integrate with other social media like facebook, instagram etc.

Using better learning algorithms which keep learning over time such as deep learning or neural networks.

Such algorithms require huge amount of data so use big data tools to implement such algorithms.

**9. Bibliography**

* Modeling Public Mood and Emotion:Twitter Sentiment and Socio-Economic Phenomena
* Bollen, J.; Mao, H.; and Zeng, X.-J. 2010. Twitter mood
* predicts the stock market. *Journal of Computational Science*
* Kim, E.; Gilbert, S.; Edwards, M.; and Graeff, E. 2009. Detecting
* Sadness in 140 Characters: Sentiment Analysis of Mourning Michael Jackson on Twitter. Technical report, Web Ecology Project, Boston, MA.
* Sentiment Classication in Twitter: A Comparison between Domain Adaptation and Distant Supervision
* Language-independent Bayesian sentiment mining of Twitter. Alex Davis and Zoubin Ghahramani.
* S. Asur and B. Huberman. Predicting the future with social media.
* L. Barbosa and J. Feng. Robust sentiment detection on Twitter from biased and noisy data.
* M. M. Bradley and P. J. Lang. Aective norms for English words (ANEW): Stimuli, instruction manual and aective ratings.
* T. Lake. Twitter Sentiment Analysis. Technical report, Western Michigan University
* Characterizing Debate Performance via Aggregated Twitter Sentiment.Nicholas A. Diakopoulos and David A. Shamma.
* <http://adilmoujahid.com/posts/2014/07/twitter-analytics/>
* <http://www.laurentluce.com/posts/twitter-sentiment-analysis-using-python-and-nltk/>
* [http://chimera.labs.oreilly.com/books/1234000001583/ch01.html#fundamental\_twitter\_terminology](http://chimera.labs.oreilly.com/books/1234000001583/ch01.html)
* <http://sachithdhanushka.blogspot.in/2014/02/mining-twitter-data-using-python.html>
* <http://badhessian.org/2012/10/collecting-real-time-twitter-data-with-the-streaming-api/>

**10. Code in Python:**

import twitter

import json

import pandas as pd

import numpy as np

import csv

import re

import scipy

import math

import matplotlib

#matplotlib.use("qt4agg")

from matplotlib import pylab

from pylab import \*

from matplotlib import pyplot as plt

import os

os.environ['NLTK\_DATA']='C:\Users\Indraneel\AppData\Roaming'

from nltk import stem

def processTweet(tweet):

tweet = tweet.lower()

tweet = re.sub('((www\.[^\s]+)|(https?://[^\s]+))','URL',tweet)

tweet = re.sub('@[^\s]+','AT\_USER',tweet)

tweet = re.sub('[\s]+', ' ', tweet)

tweet = re.sub(r'#([^\s]+)', r'\1', tweet)

tweet = tweet.strip('\'"')

return tweet

def replaceTwoOrMore(s):

pattern = re.compile(r"(.)\1{1,}", re.DOTALL)

return pattern.sub(r"\1\1", s)

stopWords=[]

def getStopWordList(stopWordListFileName):

stopWords = []

stopWords.append('AT\_USER')

stopWords.append('URL')

fp = open(stopWordListFileName, 'r')

line = fp.readline()

while line:

word = line.strip()

stopWords.append(word)

line = fp.readline()

fp.close()

return stopWords

stemmer=stem.SnowballStemmer("english")

def getFeatureVector(tweet):

featureVector = []

words = tweet.split()

for w in words:

w = stemmer.stem(w)

w = replaceTwoOrMore(w)

w = w.strip('\'"?,.')

val = re.search(r"^[a-zA-Z][a-zA-Z0-9]\*$", w)

if(w in stopWords or val is None):

continue

else:

featureVector.append(w.lower())

return featureVector

def sentiment(text):

words = pattern\_split.split(text.lower())

sentiments = map(lambda word: afinn.get(word, 0), words)

if sentiments:

sentiment = float(sum(sentiments))/math.sqrt(len(sentiments))

else:

sentiment = 0

return sentiment

A=[]

F=[]

J=[]

Sa=[]

Su=[]

D=[]

angerwords=[]

filename='emo\_anger\_s.txt'

fp=open(filename,'r')

line=fp.readline()

while line:

word=line.strip()

angerwords.append(word)

line=fp.readline()

#print angerwords

fp.close()

def anger(tweet) :

X=[]

for w in tweet :

if w in angerwords:

X.append(w)

A.append(len(X))

joywords=[]

filename='emo\_joy\_s.txt'

fp=open(filename,'r')

line=fp.readline()

while line:

word=line.strip()

joywords.append(word)

line=fp.readline()

fp.close()

def joy(tweet) :

X=[]

for w in tweet :

if w in joywords:

X.append(w)

J.append(len(X))

sadwords=[]

filename='emo\_sadness\_s.txt'

fp=open(filename,'r')

line=fp.readline()

while line:

word=line.strip()

sadwords.append(word)

line=fp.readline()

fp.close()

def sadness(tweet) :

X=[]

for w in tweet :

if w in sadwords:

X.append(w)

Sa.append(len(X))

fearwords=[]

filename='emo\_fear\_s.txt'

fp=open(filename,'r')

line=fp.readline()

while line:

word=line.strip()

fearwords.append(word)

line=fp.readline()

fp.close()

def fear(tweet) :

X=[]

for w in tweet :

if w in fearwords:

X.append(w)

F.append(len(X))

diswords=[]

filename='emo\_disgust\_s.txt'

fp=open(filename,'r')

line=fp.readline()

while line:

word=line.strip()

diswords.append(word)

line=fp.readline()

fp.close()

def disgust(tweet) :

X=[]

for w in tweet :

if w in diswords:

X.append(w)

D.append(len(X))

surwords=[]

filename='emo\_surprise\_s.txt'

fp=open(filename,'r')

line=fp.readline()

while line:

word=line.strip()

surwords.append(word)

line=fp.readline()

fp.close()

def surprise(tweet) :

X=[]

for w in tweet :

if w in surwords:

X.append(w)

Su.append(len(X))

fp = open('dis.txt', 'r')

line = fp.readline()

filenameAFINN = 'AFINN-111.txt'

afinn = dict(map(lambda (w, s): (w, int(s)), [

ws.strip().split('\t') for ws in open(filenameAFINN) ]))

pattern\_split = re.compile(r"\W+")

def main():

CONSUMER\_KEY = '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'

CONSUMER\_SECRET = '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'

OAUTH\_TOKEN = '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'

OAUTH\_TOKEN\_SECRET = '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'

auth = twitter.oauth.OAuth(OAUTH\_TOKEN, OAUTH\_TOKEN\_SECRET,

CONSUMER\_KEY, CONSUMER\_SECRET)

twitter\_api = twitter.Twitter(auth=auth)

#auth = twitter.oauth.OAuth(OAUTH\_TOKEN, OAUTH\_TOKEN\_SECRET,

#CONSUMER\_KEY, CONSUMER\_SECRET)

#twitter\_api = twitter.Twitter(auth=auth)

WORLD\_WOE\_ID = 1

India\_WOE\_ID = 2295420

world\_trends = twitter\_api.trends.place(\_id=WORLD\_WOE\_ID)

India\_trends = twitter\_api.trends.place(\_id=India\_WOE\_ID)

world\_trends\_set = set([trend['name'] for trend in world\_trends[0]['trends']])

India\_trends\_set = set([trend['name'] for trend in India\_trends[0]['trends']])

print "World Trends are"

print world\_trends\_set

print "India Trends are"

print India\_trends\_set

q = raw\_input("Enter a keyword for analysis\n")

count = 10

search\_results = twitter\_api.search.tweets(q=q, count=count)

statuses = search\_results['statuses']

for \_ in range(5):

try:

next\_results = search\_results['search\_metadata']['next\_results']

except KeyError, e: # No more results when next\_results doesn't exist

break

kwargs = dict([ kv.split('=') for kv in next\_results[1:].split("&") ])

search\_results = twitter\_api.search.tweets(\*\*kwargs)

statuses += search\_results['statuses']

print "No. of tweets analysed", len(statuses)

status\_texts = [ status['text']

for status in statuses ]

sta=json.dumps(status\_texts, indent=1)

text\_file = open("Output.txt", "w")

text\_file.write(sta)

text\_file.close()

fp = open('Output.txt', 'r')

line = fp.readline()

st = open('stopwords.txt', 'r')

stopWords = getStopWordList('stopwords.txt')

S=[]

while line:

processedTweet = processTweet(line)

featureVector = getFeatureVector(processedTweet)

anger(featureVector)

joy(featureVector)

fear(featureVector)

sadness(featureVector)

disgust(featureVector)

surprise(featureVector)

sentiments=sentiment(line)

S.append(sentiments)

line = fp.readline()

j=pd.DataFrame(J)

sa=pd.DataFrame(Sa)

a=pd.DataFrame(A)

su=pd.DataFrame(Su)

d=pd.DataFrame(D)

f=pd.DataFrame(F)

data=pd.concat([j,sa,a,su,d,f],axis=1)

data.columns=['joy','sadness','anger','suprise','disgust','fear']

count=data.anger.size

i=0

jj=0

while i < count:

if data['joy'][i]!=0:

jj=jj+data['joy'][i]

i=i+1

else:

i=i+1

joy\_percent=100.0\*jj/count

k=0

ss=0

while k < count:

if data['sadness'][k]!=0:

ss=ss+data['sadness'][k]

k=k+1

else:

k=k+1

sadness\_percent=100.0\*ss/count

l=0

aa=0

while l < count:

if data['anger'][l]!=0:

aa=aa+data['anger'][l]

l=l+1

else:

l=l+1

anger\_percent=100.0\*aa/count

m=0

uu=0

while m < count:

if data['suprise'][m]!=0:

uu=uu+data['suprise'][m]

m=m+1

else:

m=m+1

surprise\_percent=100.0\*uu/count

n=0

dd=0

while n < count:

if data['disgust'][n]!=0:

dd=dd+data['disgust'][n]

n=n+1

else:

n=n+1

disgust\_percent=100.0\*dd/count

o=0

ff=0

while o < count:

if data['fear'][o]!=0:

ff=ff+data['fear'][o]

o=o+1

else:

o=o+1

fear\_percent=100.0\*ff/count

print "The percentage of tweets that are joyful is",joy\_percent

print "The percentage of tweets that are sad is",sadness\_percent

print "The percentage of tweets that are angry is",anger\_percent

print "The percentage of tweets that show surprise is",surprise\_percent

print "The percentage of tweets that show disgust is",disgust\_percent

print "The percentage of tweets that are fearful is",fear\_percent

s=0

p=0

n=0

nu=0

while s < count:

if S[s] > 0:

p=p+1

s=s+1

elif S[s] < 0:

n=n+1

s=s+1

else:

nu=nu+1

s=s+1

p\_per=100.0\*p/count

n\_per=100.0\*n/count

nu\_per=100.0\*nu/count

print "Percentage of tweets having positive sentiment is",p\_per

print "Percentage of tweets having negative sentiment is",n\_per

print "Percentage of tweets having neutral sentiment is",nu\_per

#print S

#print min(S)

z=0

SS=[]

R=[]

r=[]

T=[]

while z < count :

if S[z]!=0:

T.append(S[z])

SS.append(S[z]-min(S))

z=z+1

else:

z=z+1

count1=len(SS)

sd=np.std(T)

zzz=0

R0=[]

R1=[]

R2=[]

R3=[]

R4=[]

R5=[]

while zzz < count1:

if T[zzz] < (-2\*sd):

R0.append(T[zzz])

zzz=zzz+1

elif (-2\*sd) <= T[zzz] < (-sd):

R1.append(T[zzz])

zzz=zzz+1

elif (-sd) <= T[zzz] < 0:

R2.append(T[zzz])

zzz=zzz+1

elif 0 <= T[zzz] < sd:

R3.append(T[zzz])

zzz=zzz+1

elif sd <= T[zzz] < 2\*sd:

R4.append(T[zzz])

zzz=zzz+1

else:

R5.append(T[zzz])

zzz=zzz+1

r0=len(R0)

r1=len(R1)

r2=len(R2)

r3=len(R3)

r4=len(R4)

r5=len(R5)

avg=(r1 + 2.0\*r2 + 3.0\*r3 + 4.0\*r4 + 5.0\*r5)/count1

print "The average rating of people's sentiment out of five is", avg

fig=plt.figure()

x = scipy.arange(6)

y = scipy.array([joy\_percent,sadness\_percent,anger\_percent,surprise\_percent,disgust\_percent,fear\_percent])

ax=fig.add\_subplot(223)

ax.bar(x, y, align='center')

ax.set\_xticks(x)

ax.set\_xticklabels(['Joy', 'Sadness', 'Anger', 'Surprise','Disgust','Fear'])

ax1=fig.add\_subplot(222)

labels = 'Neutral', 'Positive', 'Negative'

fracs = [nu\_per,p\_per,n\_per]

explode=(0.05, 0.05, 0.05)

ax1.pie(fracs, explode=explode, labels=labels,

autopct='%1.1f%%', shadow=True, startangle=90)

x = scipy.arange(6)

y = scipy.array([r0,r1,r2,r3,r4,r5])

ax2=fig.add\_subplot(221)

ax2.bar(x, y, align='center')

ax2.set\_xticks(x)

ax2.set\_xticklabels(['0', '1', '2', '3','4','5'])

plt.show()

if \_\_name\_\_ == '\_\_main\_\_':

main()