INSY 5378

PROJECT 1:

POST-ELECTION SOCIAL MEDIA ANALYTICS

Team 02:

Dhanashri Ostwal (dhanashrivilas.ostwal@mavs.uta.edu) – 1001277328

Priyanka Sekhar (priyanka.sekhar@mavs.uta.edu) - 1001215254

Saranya Ravichandran (saranya.ravichandran@mavs.uta.edu) -1001160582

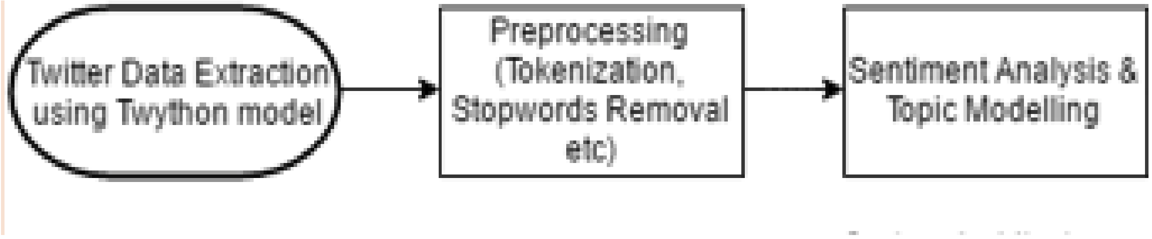
AIM OF THE PROJECT

This project was targeted towards the analysis of public response for 3 most popular people around 2016 U.S. elections (Barack Obama, Hillary Clinton and Donald Trump), as expressed on Twitter, which is a micro-blogging service. After which we calculated the sentiment score for each person to answer the following questions: 1) Which person has the most positive tweets 2) Which person has the most negative tweets.

Using the tweets that we got for each person, we created a word cloud (a visual representation of the text data) which gave us a clear picture about which words have been used the most for which person. Additionally, we performed Topic Modelling on these tweets by using a python sentiment analysis module.

This process was repeated by take location (Dallas TX, New York NY, San Francisco CA, Los Angeles CA, Chicago IL, Washington DC, Atlanta GA) as a criteria of search for each of the three important people. Following which we found out which person has the most positive/negative tweets and in which city. Also, we created word clouds for each city and person.

MODEL


Twitter Data Extraction using Twython model
Preprocessing (Tokenization, Stopwords Removal etc)
Sentiment Analysis & Topic Modelling

Our system analyzes the twitter data and gives a new and meaningful perspective on the dynamics of the electoral process and public opinion.

1. **Tweet extraction from Twitter:**

   We extracted about 10,000 tweets from twitter for Hilary, Trump and Obama, using the Twitter Streaming API and Twython, by getting 500 tweets every 15 minutes.

   We used **track = KEYWORD** to get keyword filtered tweets for each person.

2. **Sentiment Analysis:**

   In this step, we calculated the sentiment score for each person using the tweets we extracted. We used Naive Bayes Classifier for serving this purpose.

   The Naïve Bayes Classifier is based on the bag-of-words model. With this model we check which word of the tweet belongs to the positive-word-list and which to the negative-word-list. If the word belongs to the positive word list, the total score of the tweets (for that person) is updated to +1 and vice versa. If at the end the total score is positive, the tweets for that person is classified as positive, and if not, then negative.

   At the end of this classification, we use this sentiment score for each person to answer the question: Which person has the most positive/most negative tweets?

3. **Data Pre-Processing:**

   Pre-processing is a very important step in the data collection process. Data gathering such as tweet extraction usually contains many garbage values. It's necessary to remove such characters and values, otherwise it may lead to misleading results.

i. Stop word removal:

Stop words are common words that carry less importance than the keywords. Words like “is”, “a”, “the” etc. are usually removed in this step before creating the word cloud.

NLTK corpus (a list of 2,400 stop words) which is part of Python was used by us to remove these stop words from our twitter extracted data.

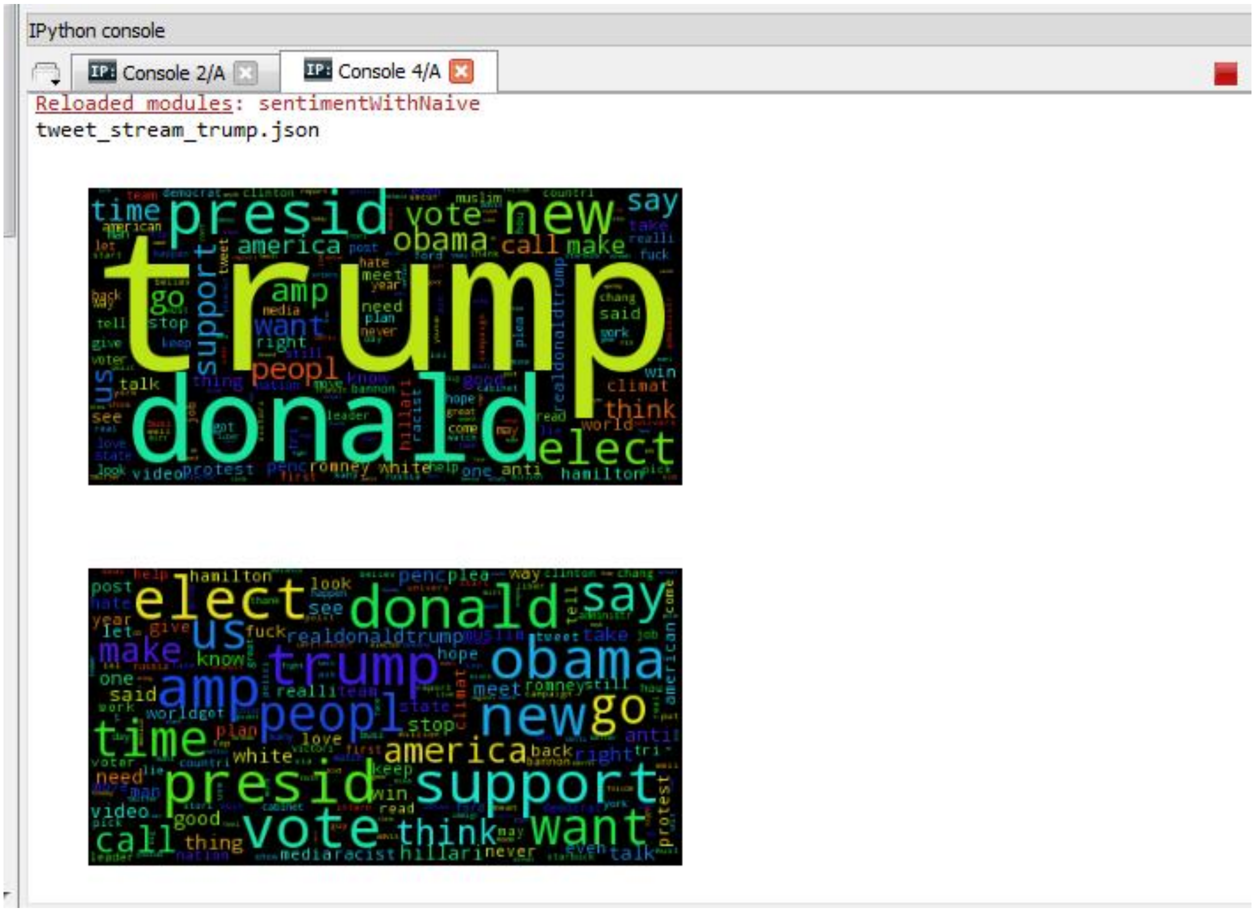
ii. Stemming:

This method is used to identify the root/stem of a word. For example, the words “connect”, “connected”, “connecting”, “connections” can all be stemmed to the word “Connect”. The purpose of this step is to remove various suffixes, to reduce the number of words, to have matching stems. This helps us save time and memory space too.

4. Word Cloud Generation

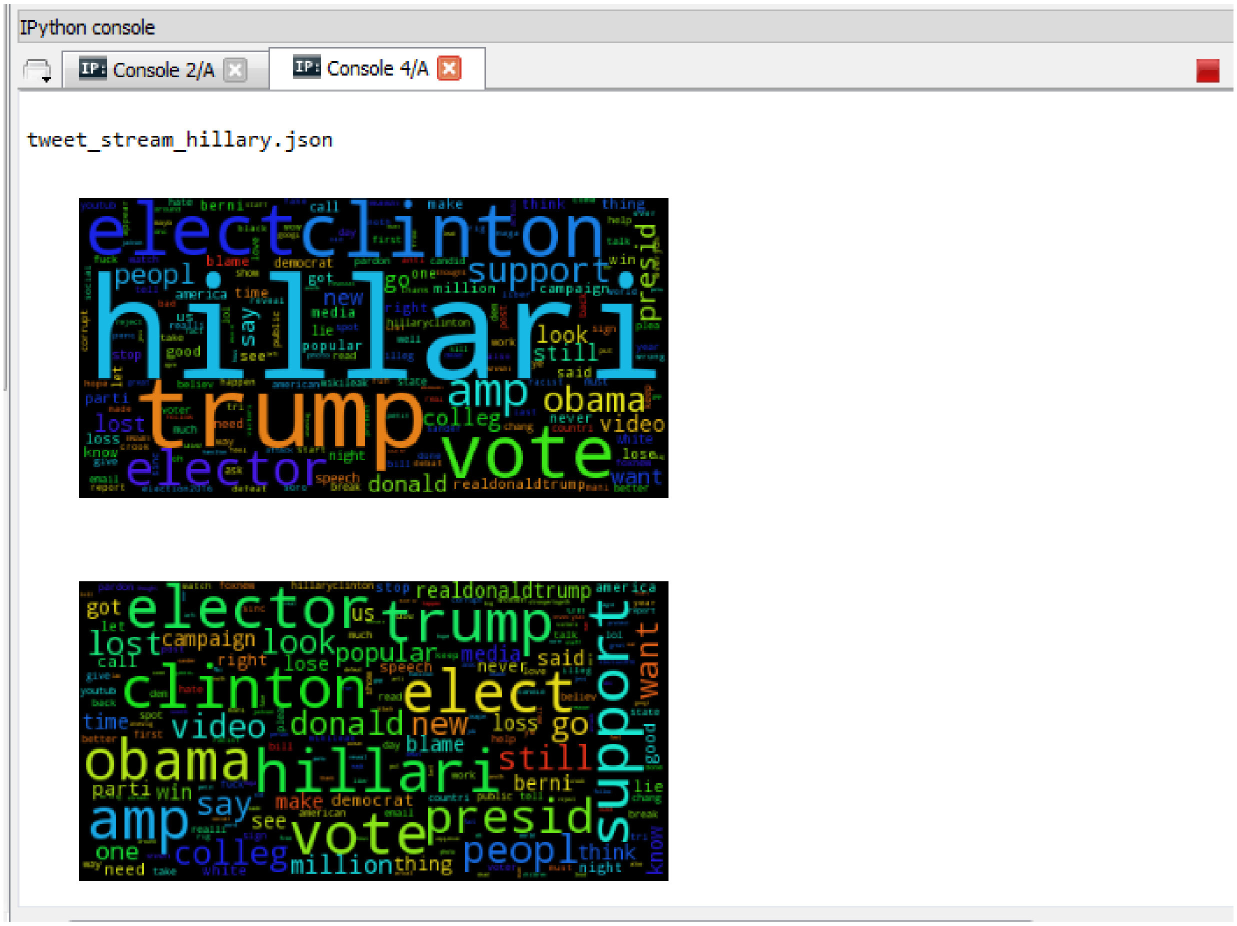
Word Cloud is one of the simplest and most intuitive form of visualizing data. These clouds give greater importance to words which occur more often than others in the tweets.

We formed these clouds for each person using the python “WordCloud” module in Python.



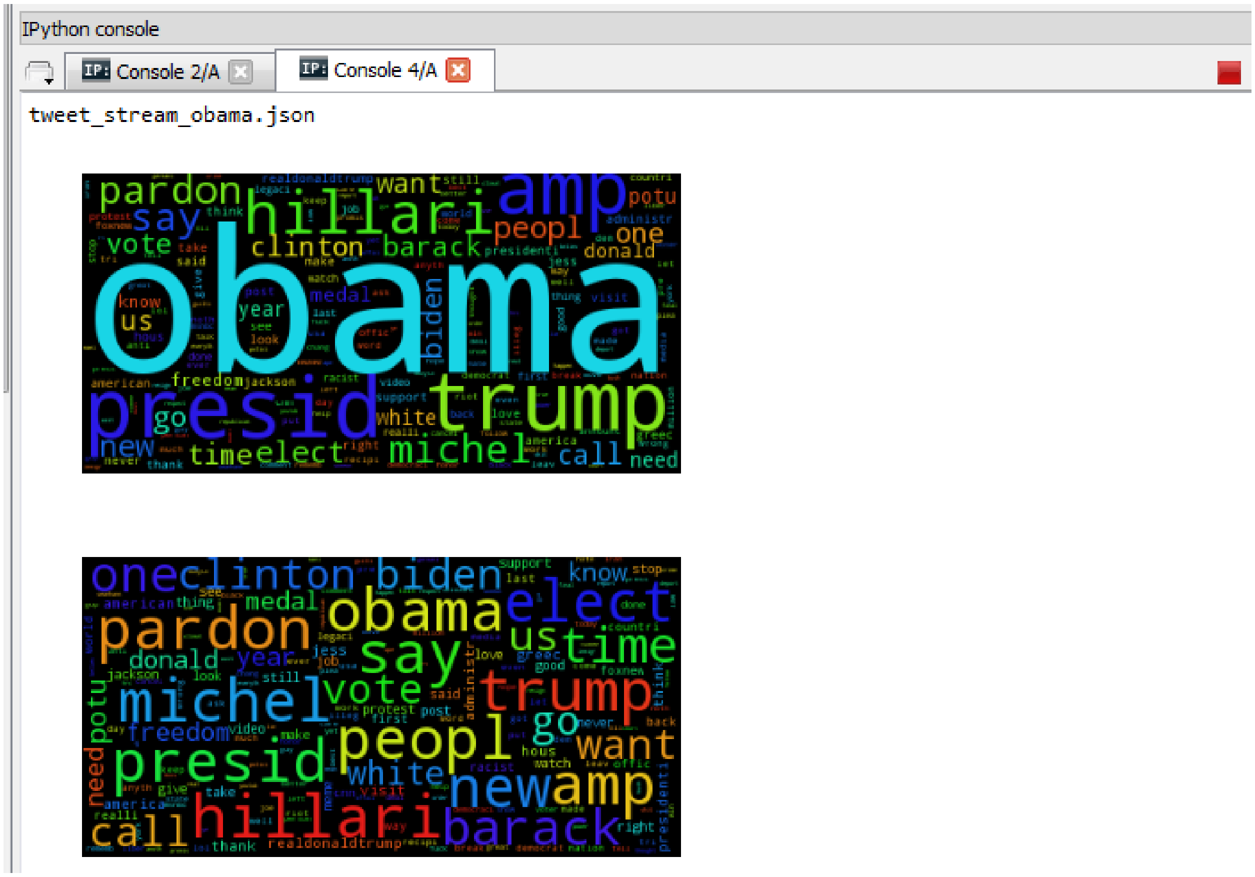
```
Reloaded modules: sentimentWithNaive
tweet_stream_trump.json
```

Word Cloud for Trump using 10k tweets



```
tweet_stream_hillary.json
```

Word Cloud for Hillary using 10k tweets.



```
tweet_stream_obama.json
```

Word Cloud for Obama using 10k tweets.

5. Location specific Tweets

Next, we performed all the above steps (Collection, Sentiment Analysis, Word Cloud generation) by adding the locations filter to the Streaming API. We used **"locations = lat1, lon1, lat2, lon2"** to collect tweets from a specific region.

We collected tweets from the following 7 cities: Dallas TX, New York NY, San Francisco CA, Los Angeles CA, Chicago IL, Washington DC and Atlanta GA.

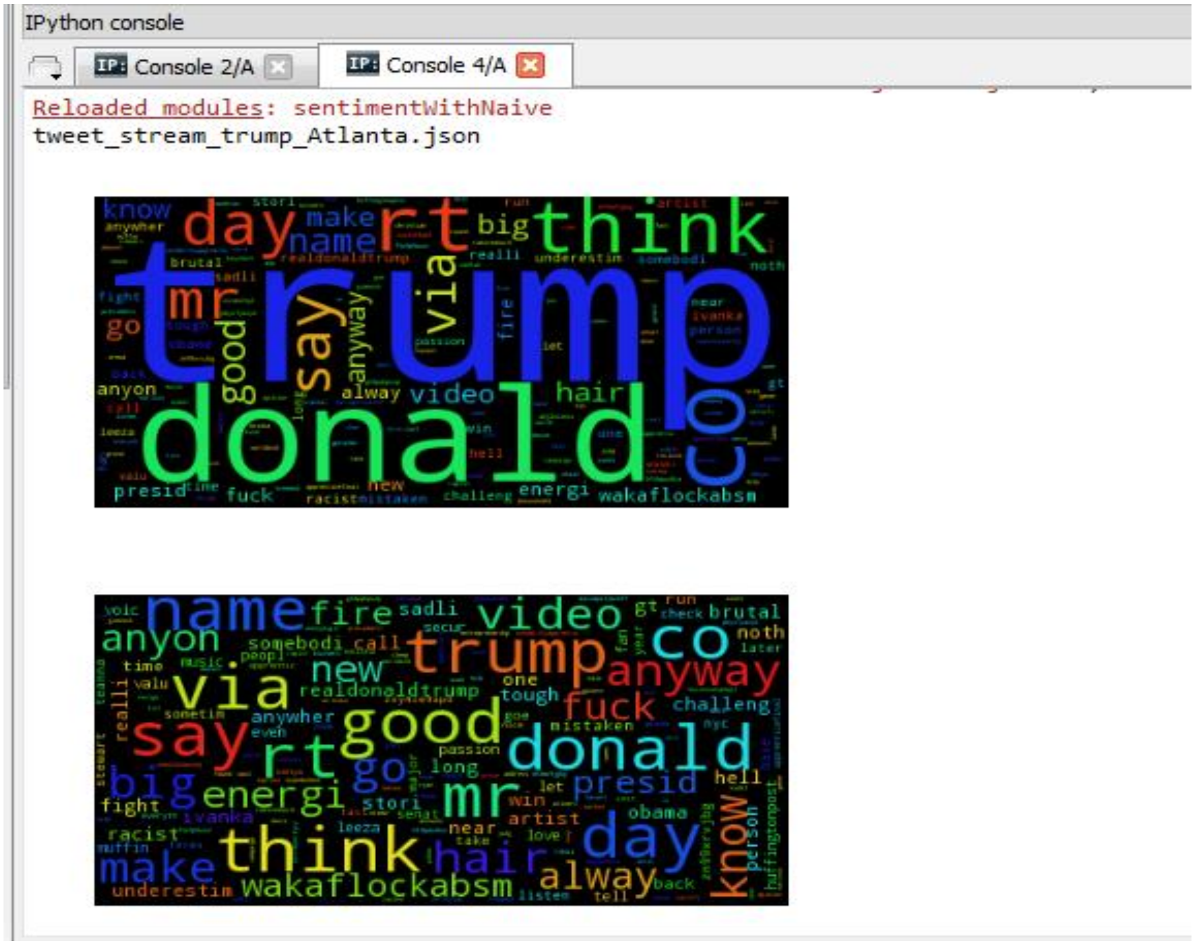
Then we used this information to determine **who out of the three people have the most positive/negative tweets.**

We also created word clouds for each city and person to visualize which words were used most often in the case of each person for each city.

Word clouds for each person and city:

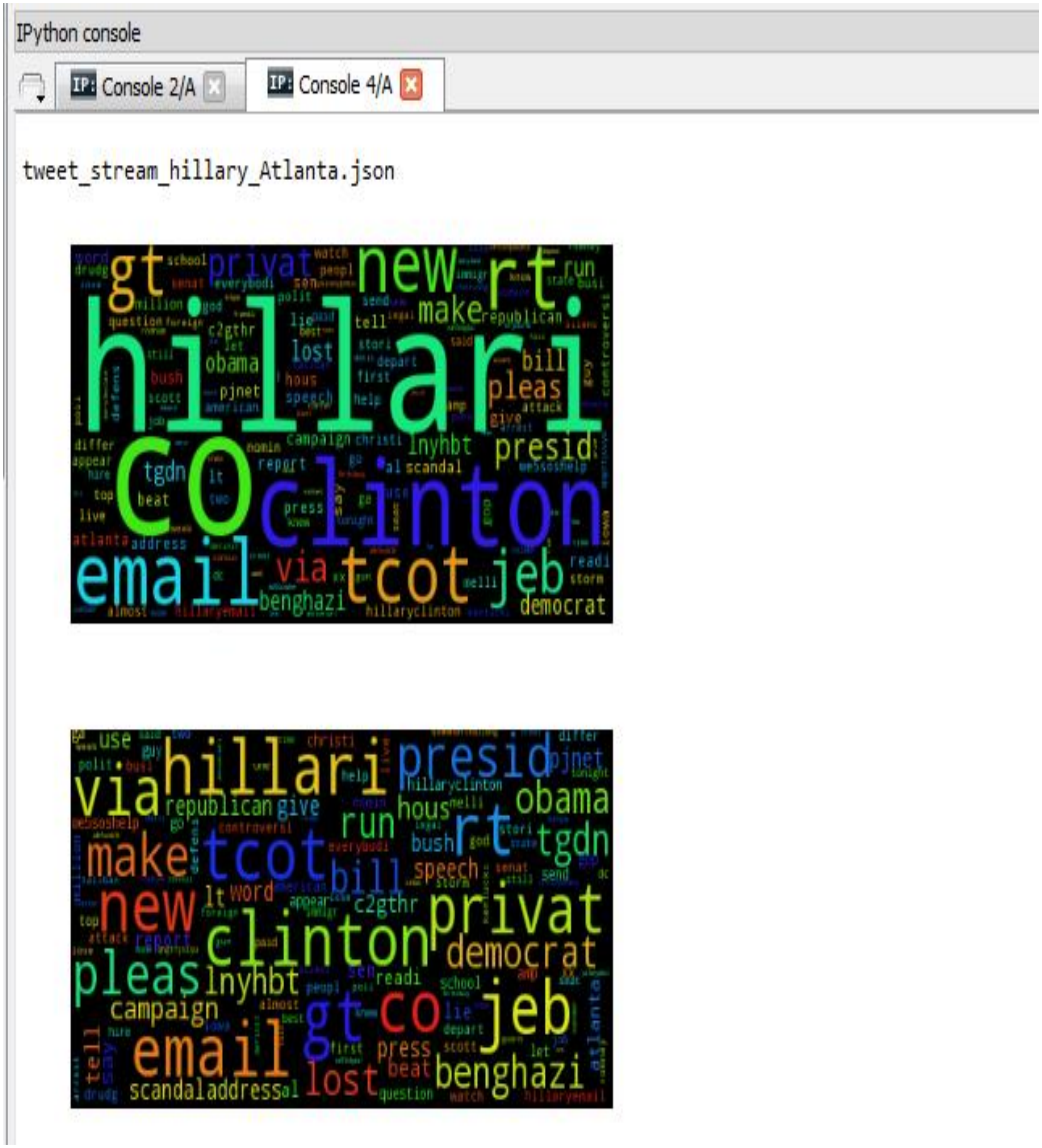
1. **Atlanta, GA**

    a. Donald Trump



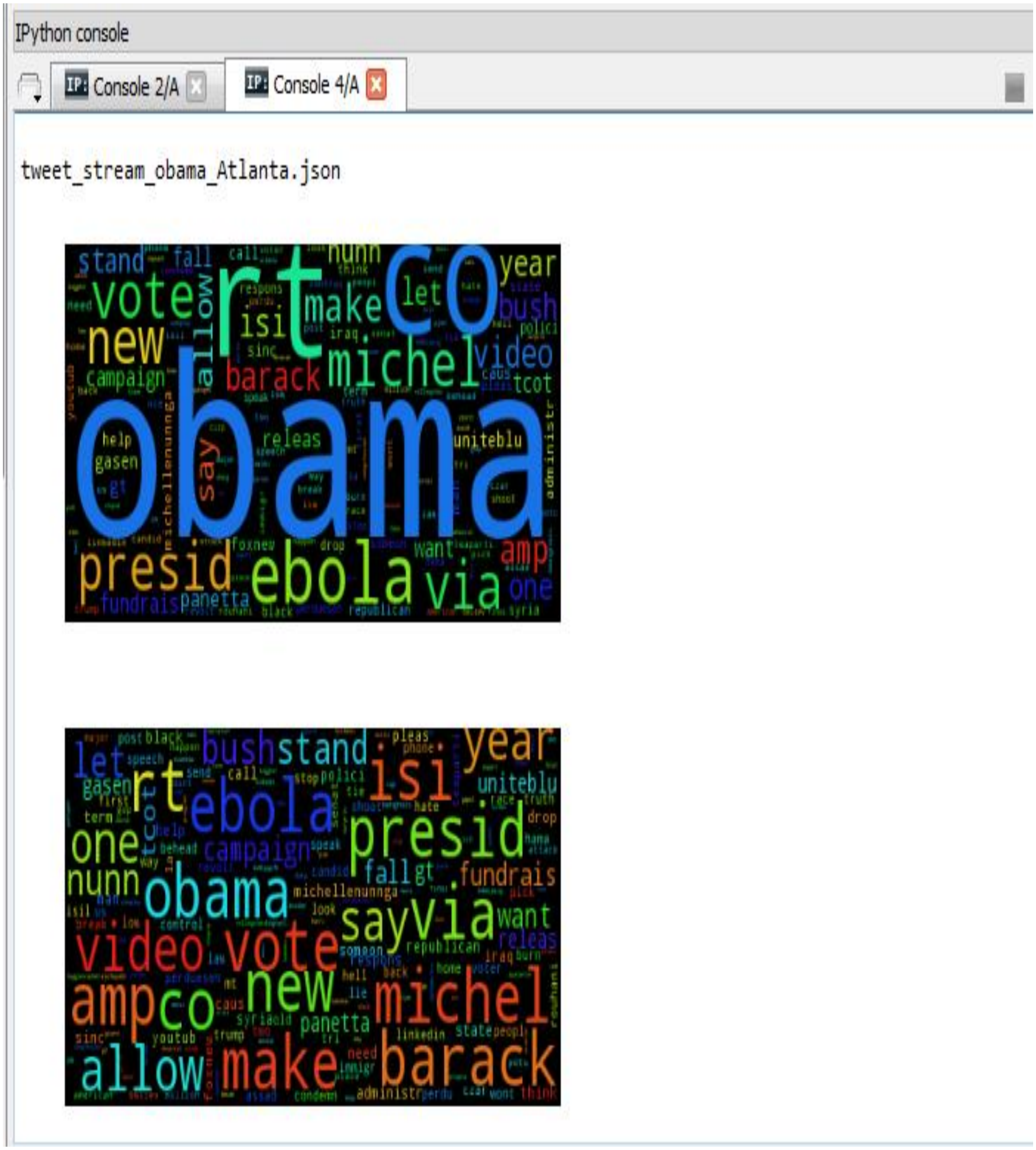
```
Reloaded modules: sentimentWithNaive
tweet_stream_trump_Atlanta.json
```

b. Hillary Clinton

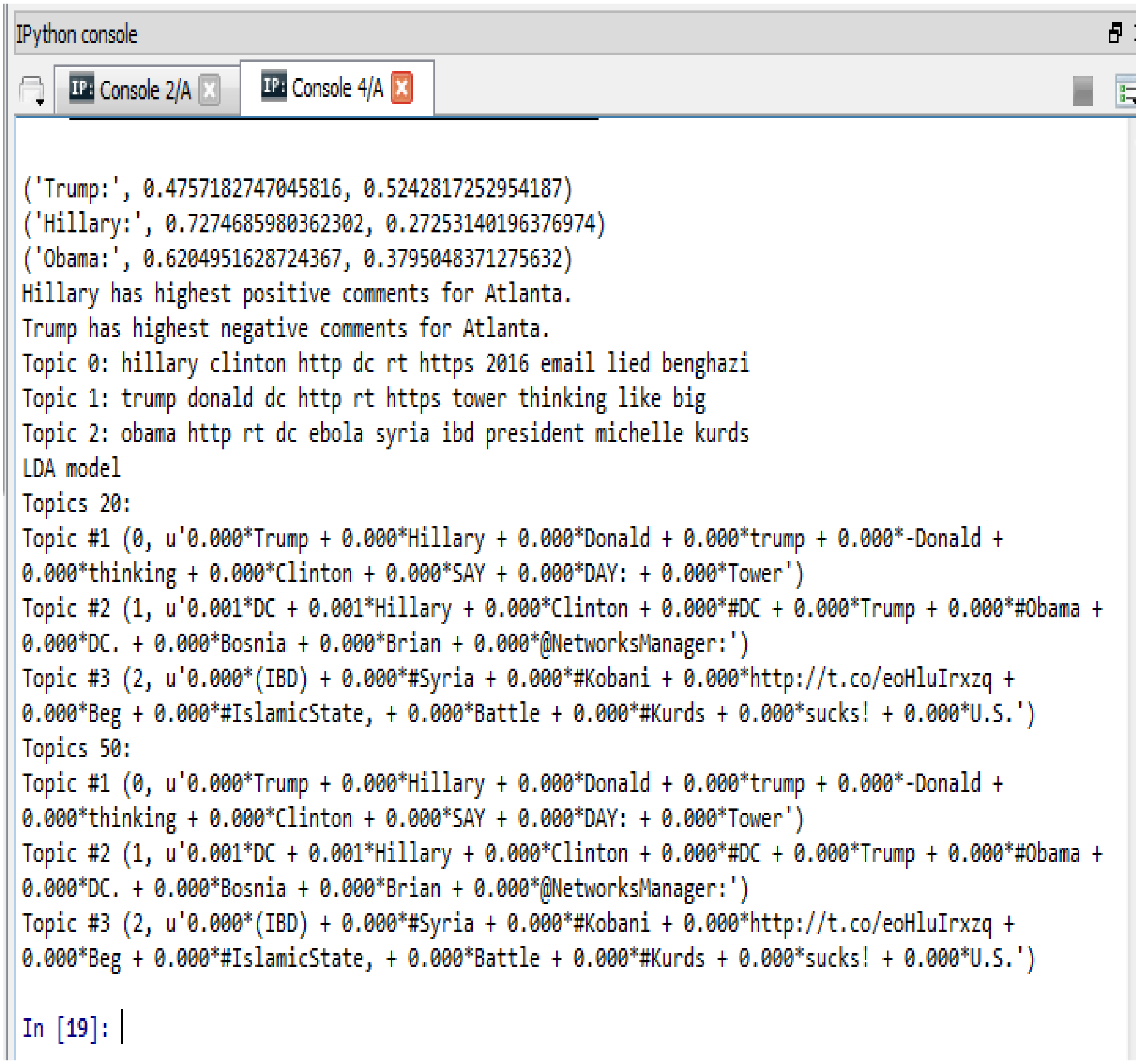


```
tweet_stream_hillary_Atlanta.json
```

c. Barack Obama



```
tweet_stream_obama_Atlanta.json
```



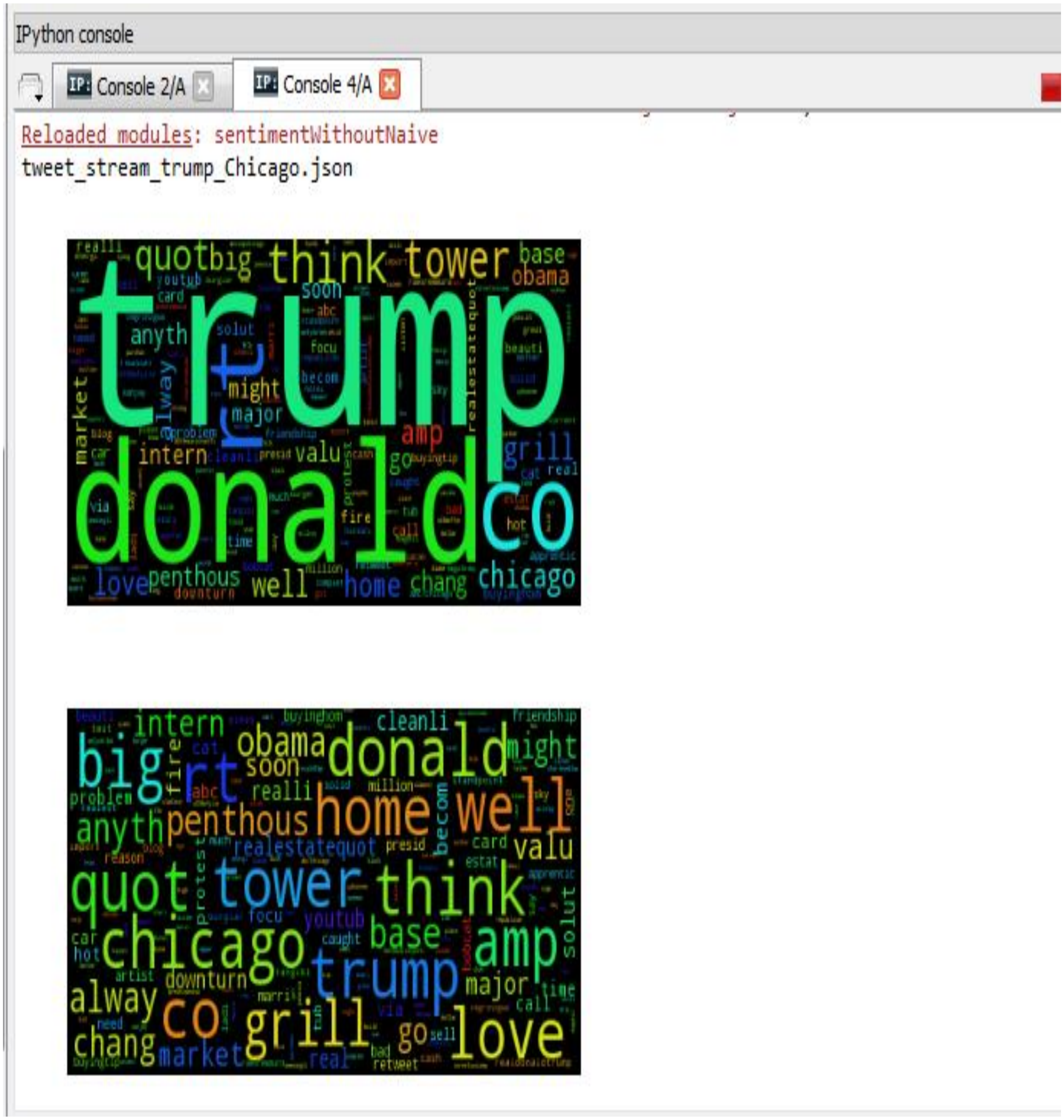
```
('Trump:', 0.4757182747045816, 0.5242817252954187)
('Hillary:', 0.7274685980362302, 0.27253140196376974)
('Obama:', 0.6204951628724367, 0.3795048371275632)
Hillary has highest positive comments for Atlanta.
Trump has highest negative comments for Atlanta.
Topic 0: hillary clinton http dc rt https 2016 email lied benghazi
Topic 1: trump donald dc http rt https tower thinking like big
Topic 2: obama http rt dc ebola syria ibd president michelle kurds
LDA model
Topics 20:
Topic #1 (0, u'0.000*Trump + 0.000*Hillary + 0.000*Donald + 0.000*trump + 0.000*-Donald +
0.000*thinking + 0.000*Clinton + 0.000*SAY + 0.000*DAY: + 0.000*Tower')
Topic #2 (1, u'0.001*DC + 0.001*Hillary + 0.000*Clinton + 0.000*#DC + 0.000*Trump + 0.000*#Obama +
0.000*DC. + 0.000*Bosnia + 0.000*Brian + 0.000*@NetworksManager:')
Topic #3 (2, u'0.000*(IBD) + 0.000*#Syria + 0.000*#Kobani + 0.000*http://t.co/eoHluIrxzq +
0.000*Beg + 0.000*#IslamicState, + 0.000*Battle + 0.000*#Kurds + 0.000*sucks! + 0.000*U.S.')
Topics 50:
Topic #1 (0, u'0.000*Trump + 0.000*Hillary + 0.000*Donald + 0.000*trump + 0.000*-Donald +
0.000*thinking + 0.000*Clinton + 0.000*SAY + 0.000*DAY: + 0.000*Tower')
Topic #2 (1, u'0.001*DC + 0.001*Hillary + 0.000*Clinton + 0.000*#DC + 0.000*Trump + 0.000*#Obama +
0.000*DC. + 0.000*Bosnia + 0.000*Brian + 0.000*@NetworksManager:')
Topic #3 (2, u'0.000*(IBD) + 0.000*#Syria + 0.000*#Kobani + 0.000*http://t.co/eoHluIrxzq +
0.000*Beg + 0.000*#IslamicState, + 0.000*Battle + 0.000*#Kurds + 0.000*sucks! + 0.000*U.S.')

In [19]:
```

Most positive/negative tweets in Atlanta, GA.

Most positive tweets: **Hillary Clinton**
Most negative tweets: **Donald Trump**

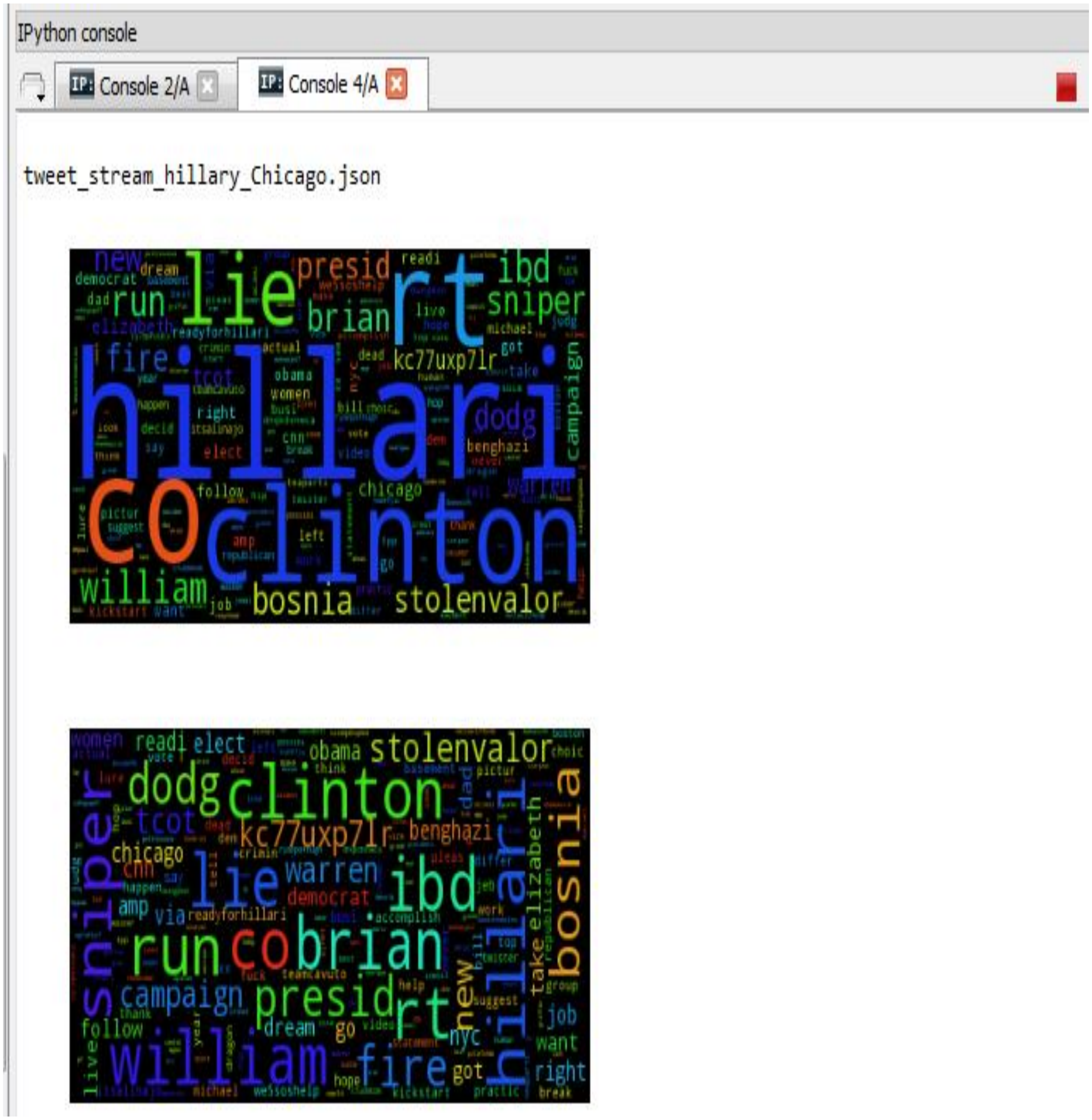
2. Chicago, IL

a. Donald Trump



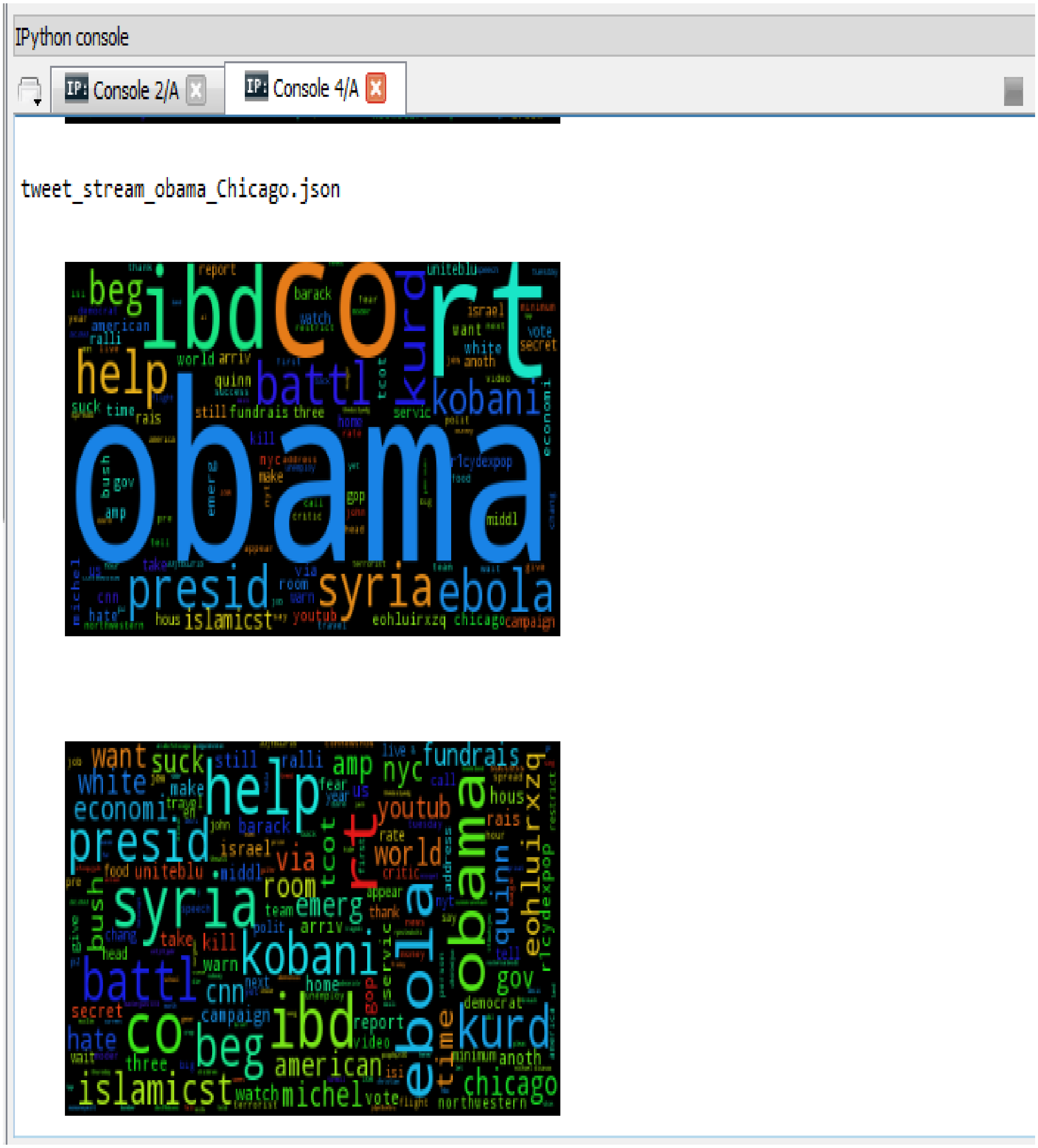
```
Reloaded modules: sentimentWithoutNaive
tweet_stream_trump_Chicago.json
```

b. Hillary Clinton

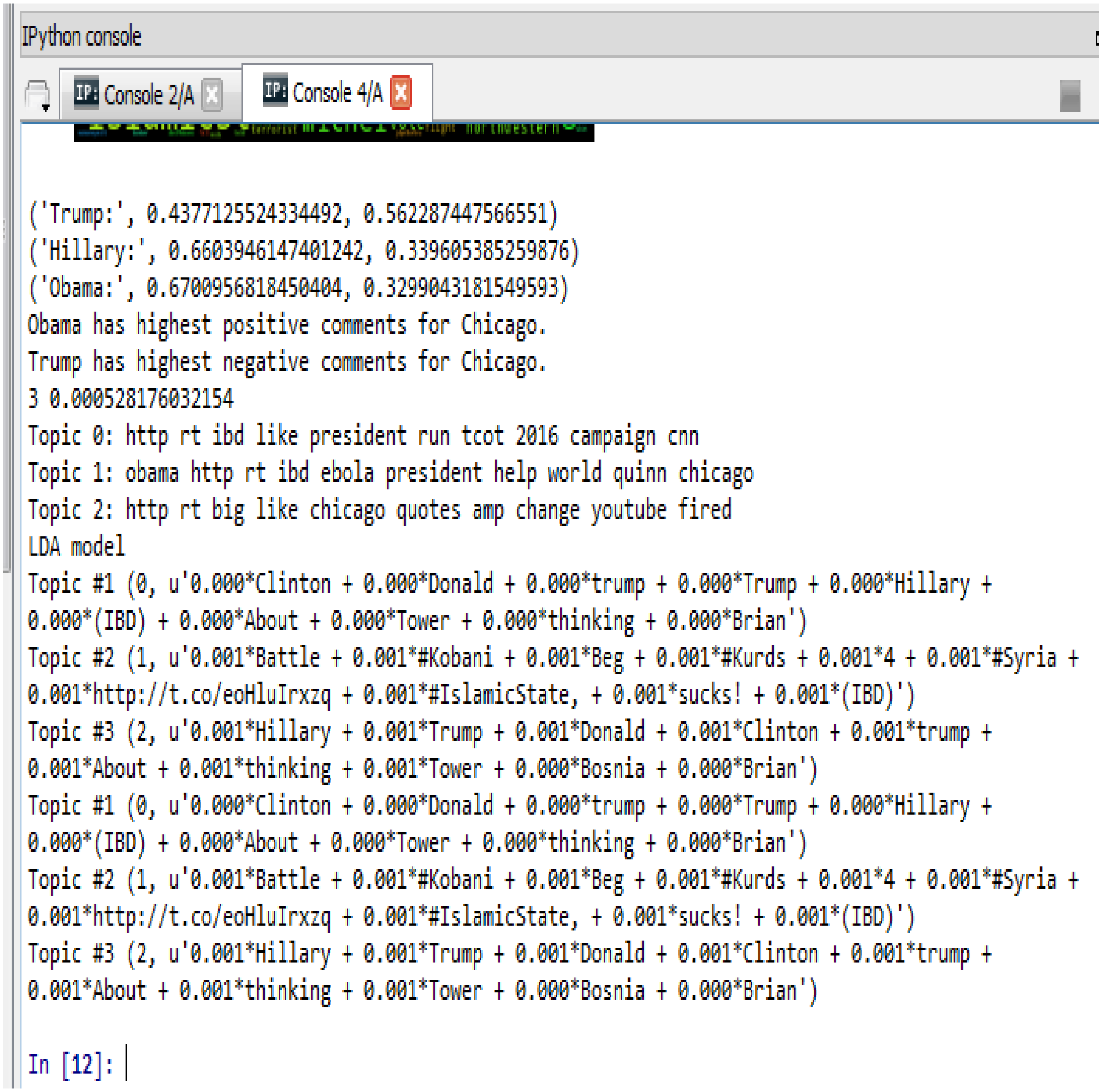


```
tweet_stream_hillary_Chicago.json
```

c. Barack Obama



```
tweet_stream_obama_Chicago.json
```



```
('Trump:', 0.4377125524334492, 0.562287447566551)
('Hillary:', 0.6603946147401242, 0.339605385259876)
('Obama:', 0.6700956818450404, 0.3299043181549593)
Obama has highest positive comments for Chicago.
Trump has highest negative comments for Chicago.
3 0.000528176032154
Topic 0: http rt ibd like president run tcot 2016 campaign cnn
Topic 1: obama http rt ibd ebola president help world quinn chicago
Topic 2: http rt big like chicago quotes amp change youtube fired
LDA model
Topic #1 (0, u'0.000*Clinton + 0.000*Donald + 0.000*trump + 0.000*Trump + 0.000*Hillary +
0.000*(IBD) + 0.000*About + 0.000*Tower + 0.000*thinking + 0.000*Brian')
Topic #2 (1, u'0.001*Battle + 0.001*#Kobani + 0.001*Beg + 0.001*#Kurds + 0.001*4 + 0.001*#Syria +
0.001*http://t.co/eoHluIrxzq + 0.001*#IslamicState, + 0.001*sucks! + 0.001*(IBD)')
Topic #3 (2, u'0.001*Hillary + 0.001*Trump + 0.001*Donald + 0.001*Clinton + 0.001*trump +
0.001*About + 0.001*thinking + 0.001*Tower + 0.000*Bosnia + 0.000*Brian')
Topic #1 (0, u'0.000*Clinton + 0.000*Donald + 0.000*trump + 0.000*Trump + 0.000*Hillary +
0.000*(IBD) + 0.000*About + 0.000*Tower + 0.000*thinking + 0.000*Brian')
Topic #2 (1, u'0.001*Battle + 0.001*#Kobani + 0.001*Beg + 0.001*#Kurds + 0.001*4 + 0.001*#Syria +
0.001*http://t.co/eoHluIrxzq + 0.001*#IslamicState, + 0.001*sucks! + 0.001*(IBD)')
Topic #3 (2, u'0.001*Hillary + 0.001*Trump + 0.001*Donald + 0.001*Clinton + 0.001*trump +
0.001*About + 0.001*thinking + 0.001*Tower + 0.000*Bosnia + 0.000*Brian')

In [12]:
```

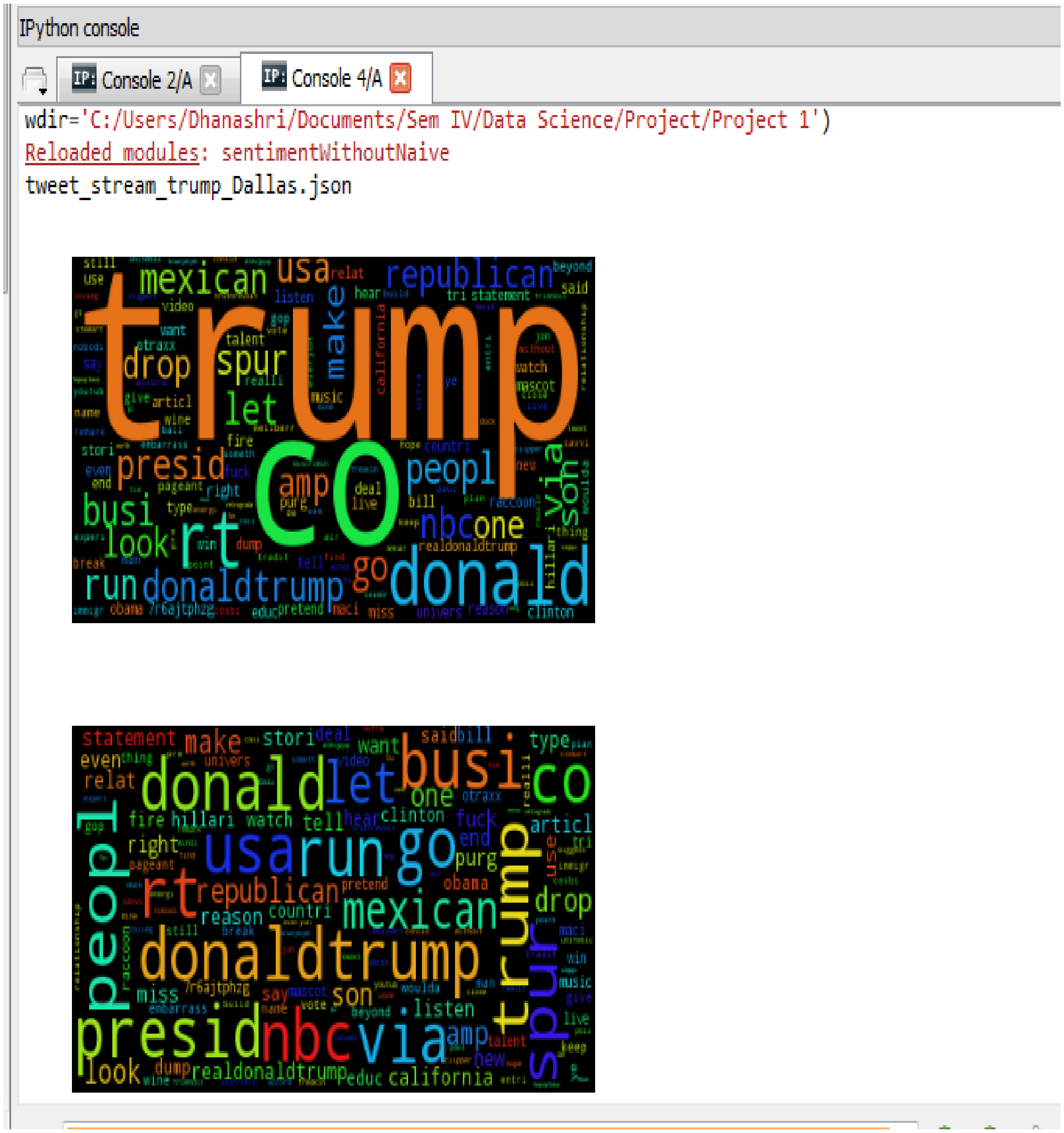
Most positive/negative tweets in Chicago, IL.

Most positive tweets: **Obama**

Most negative tweets: **Donald Trump**

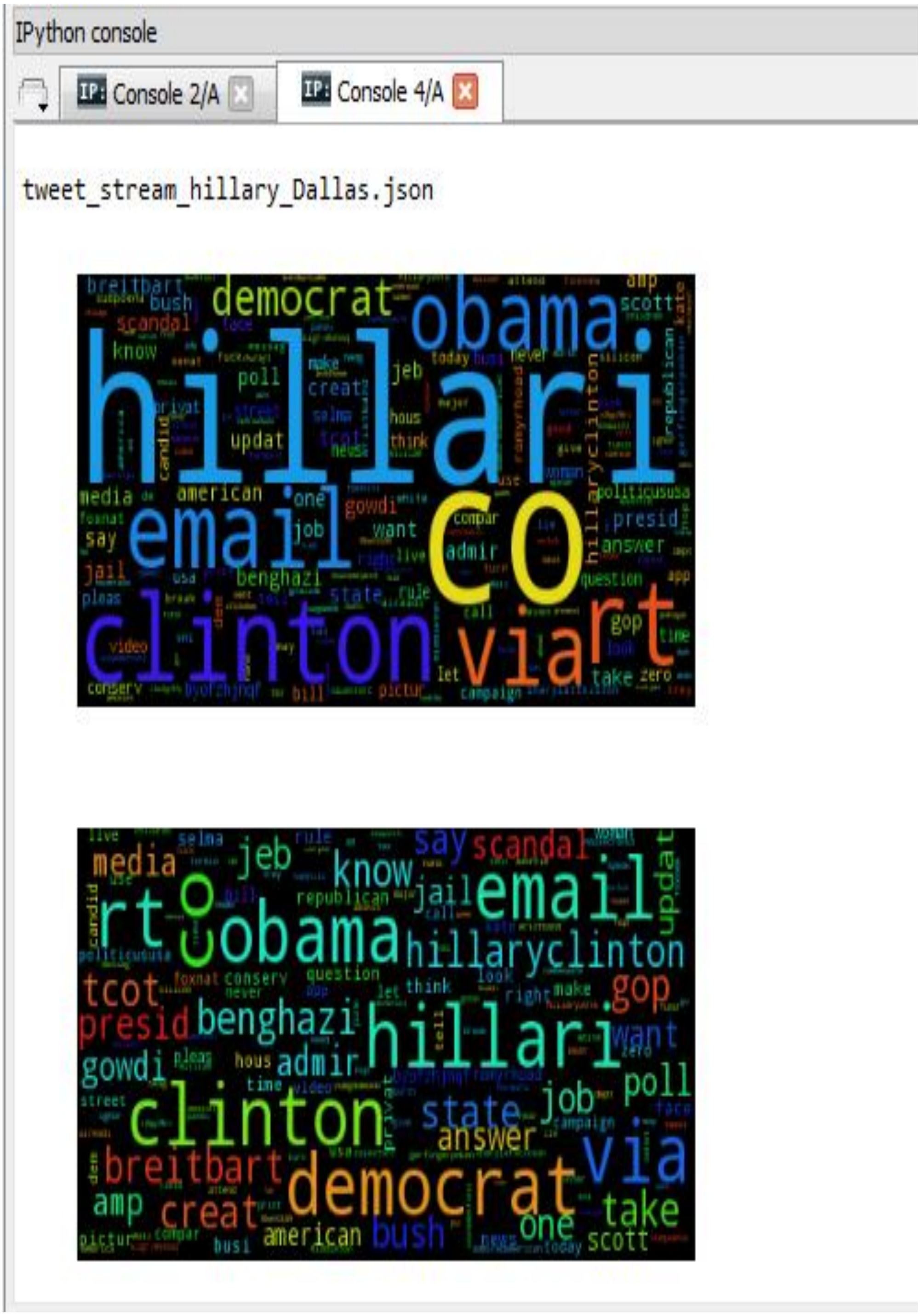
3. Dallas, TX

a. Donald Trump



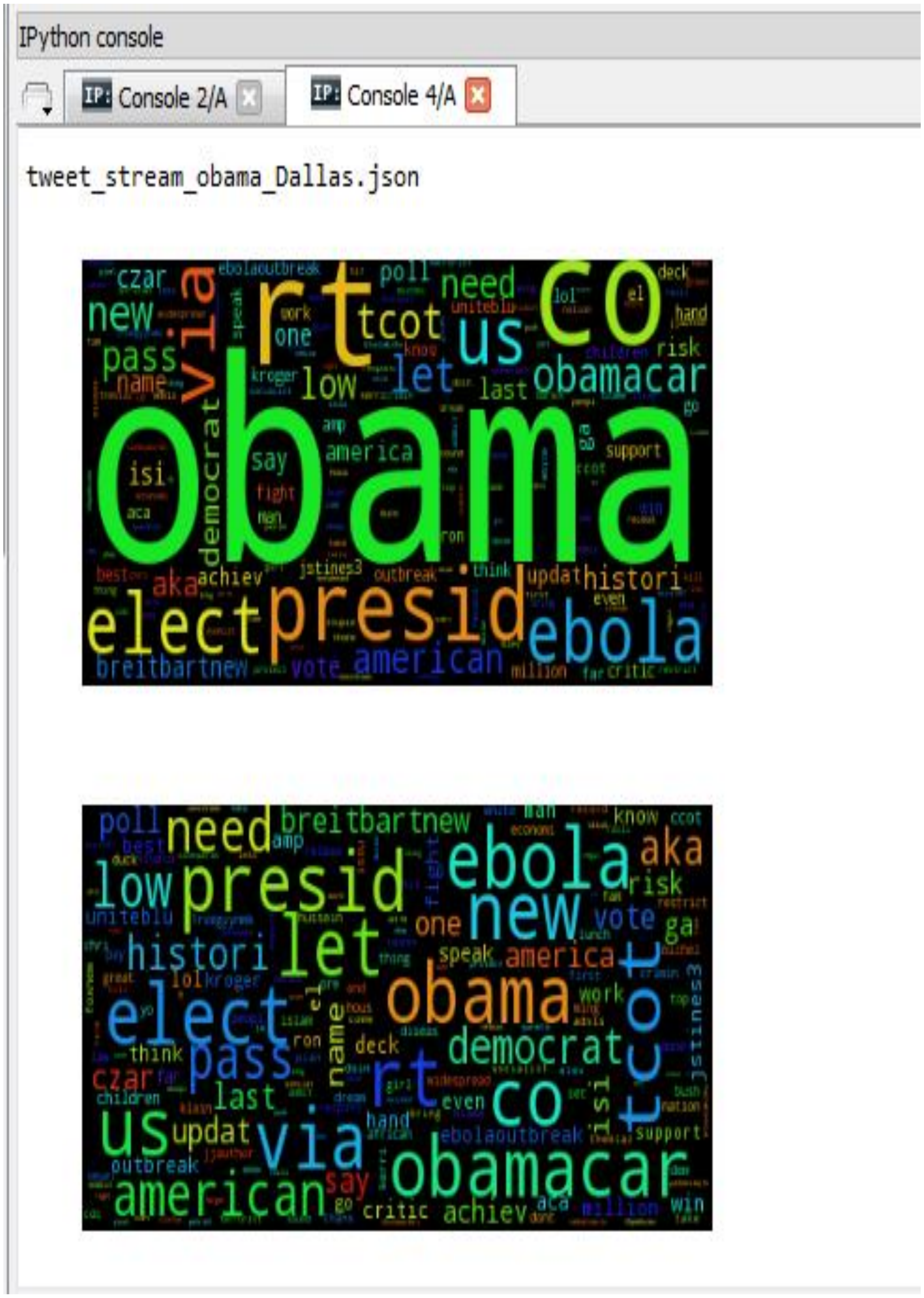
```
wdir='C:/Users/Dhanashri/Documents/Sem IV/Data Science/Project/Project 1')
Reloaded modules: sentimentWithoutNaive
tweet_stream_trump_Dallas.json
```

b. Hillary Clinton

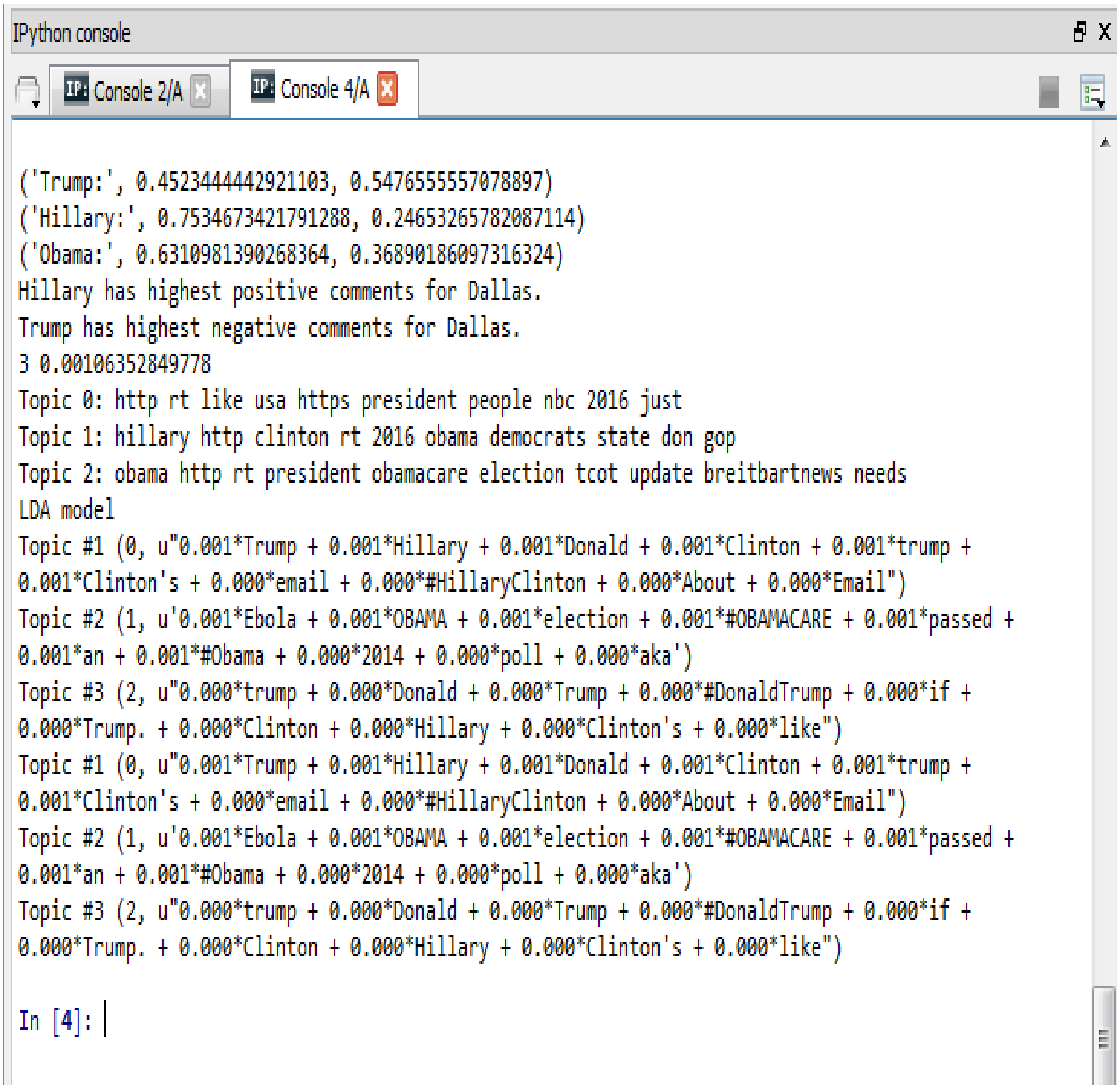


```
tweet_stream_hillary_Dallas.json
```

c. Barack Obama



```
tweet_stream_obama_Dallas.json
```



```
('Trump:', 0.4523444442921103, 0.5476555557078897)
('Hillary:', 0.7534673421791288, 0.24653265782087114)
('Obama:', 0.6310981390268364, 0.36890186097316324)
Hillary has highest positive comments for Dallas.
Trump has highest negative comments for Dallas.
3 0.00106352849778
Topic 0: http rt like usa https president people nbc 2016 just
Topic 1: hillary http clinton rt 2016 obama democrats state don gop
Topic 2: obama http rt president obamacare election tcot update breitbartnews needs
LDA model
Topic #1 (0, u"0.001*Trump + 0.001*Hillary + 0.001*Donald + 0.001*Clinton + 0.001*trump +
0.001*Clinton's + 0.000*email + 0.000*#HillaryClinton + 0.000*About + 0.000*Email")
Topic #2 (1, u'0.001*Ebola + 0.001*OBAMA + 0.001*election + 0.001*#OBAMACARE + 0.001*passed +
0.001*an + 0.001*#Obama + 0.000*2014 + 0.000*poll + 0.000*aka')
Topic #3 (2, u"0.000*trump + 0.000*Donald + 0.000*Trump + 0.000*#DonaldTrump + 0.000*if +
0.000*Trump. + 0.000*Clinton + 0.000*Hillary + 0.000*Clinton's + 0.000*like")
Topic #1 (0, u"0.001*Trump + 0.001*Hillary + 0.001*Donald + 0.001*Clinton + 0.001*trump +
0.001*Clinton's + 0.000*email + 0.000*#HillaryClinton + 0.000*About + 0.000*Email")
Topic #2 (1, u'0.001*Ebola + 0.001*OBAMA + 0.001*election + 0.001*#OBAMACARE + 0.001*passed +
0.001*an + 0.001*#Obama + 0.000*2014 + 0.000*poll + 0.000*aka')
Topic #3 (2, u"0.000*trump + 0.000*Donald + 0.000*Trump + 0.000*#DonaldTrump + 0.000*if +
0.000*Trump. + 0.000*Clinton + 0.000*Hillary + 0.000*Clinton's + 0.000*like")

In [4]:
```

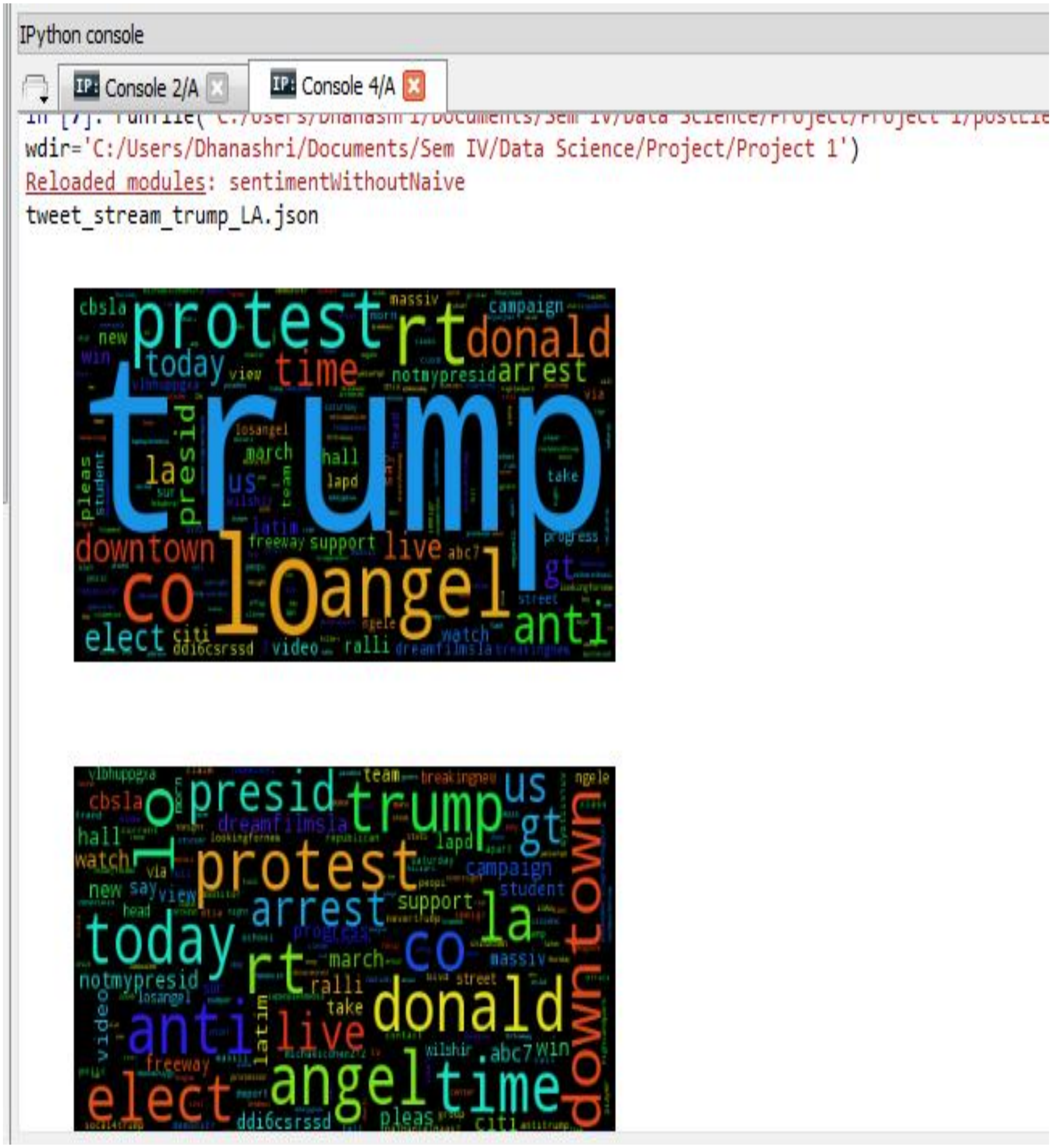
Most positive/negative tweets in Dallas, TX.

Most positive tweets: **Hillary Clinton**

Most negative tweets: **Donald Trump**

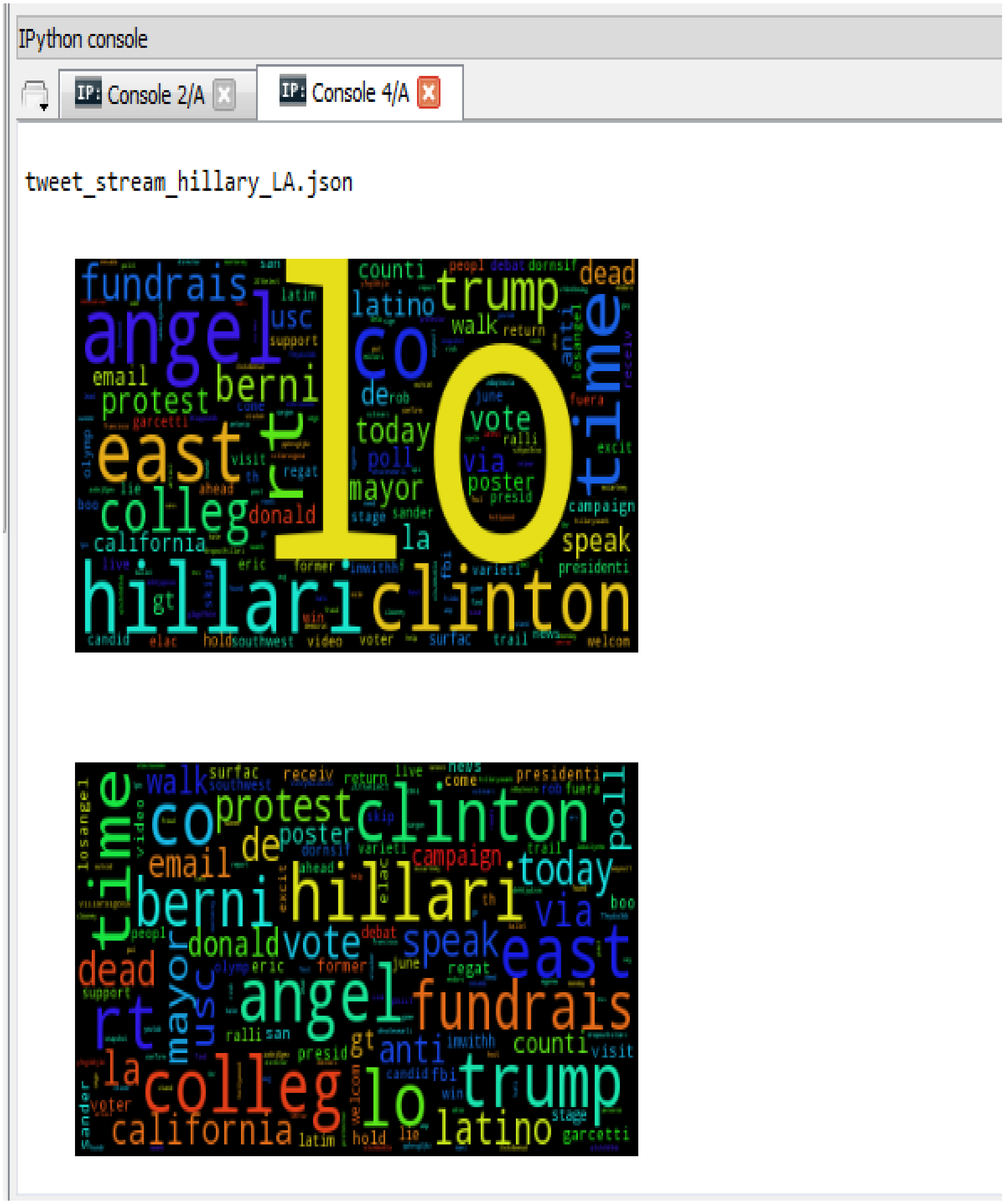
4. Los Angeles, California

a. Donald Trump



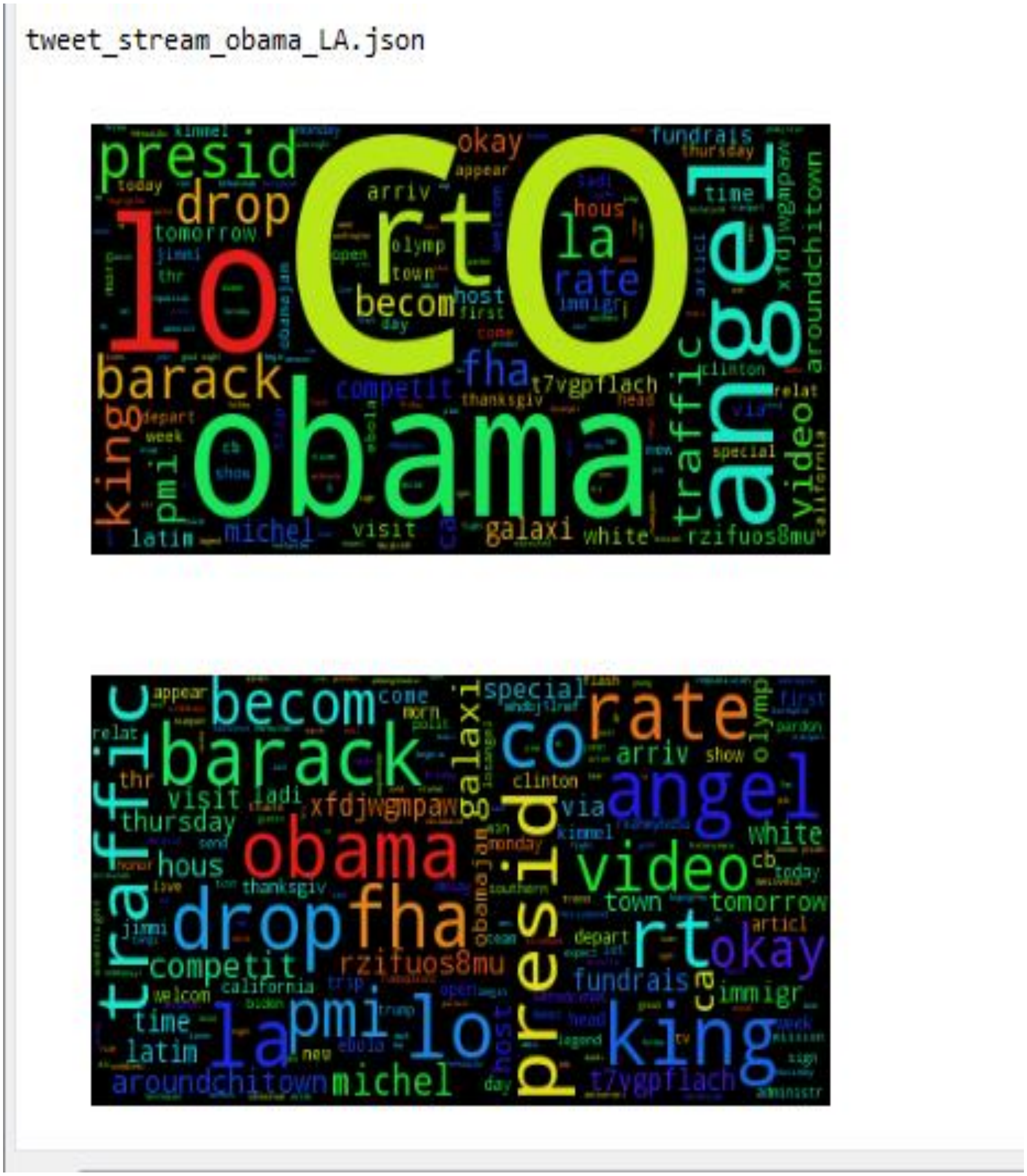
```
In [7]: runfile('C:/Users/Dhanashri/Documents/Sem IV/Data Science/Project/Project 1/postLie
wdir='C:/Users/Dhanashri/Documents/Sem IV/Data Science/Project/Project 1')
Reloaded modules: sentimentWithoutNaive
tweet_stream_trump_LA.json
```

b. Hillary Clinton

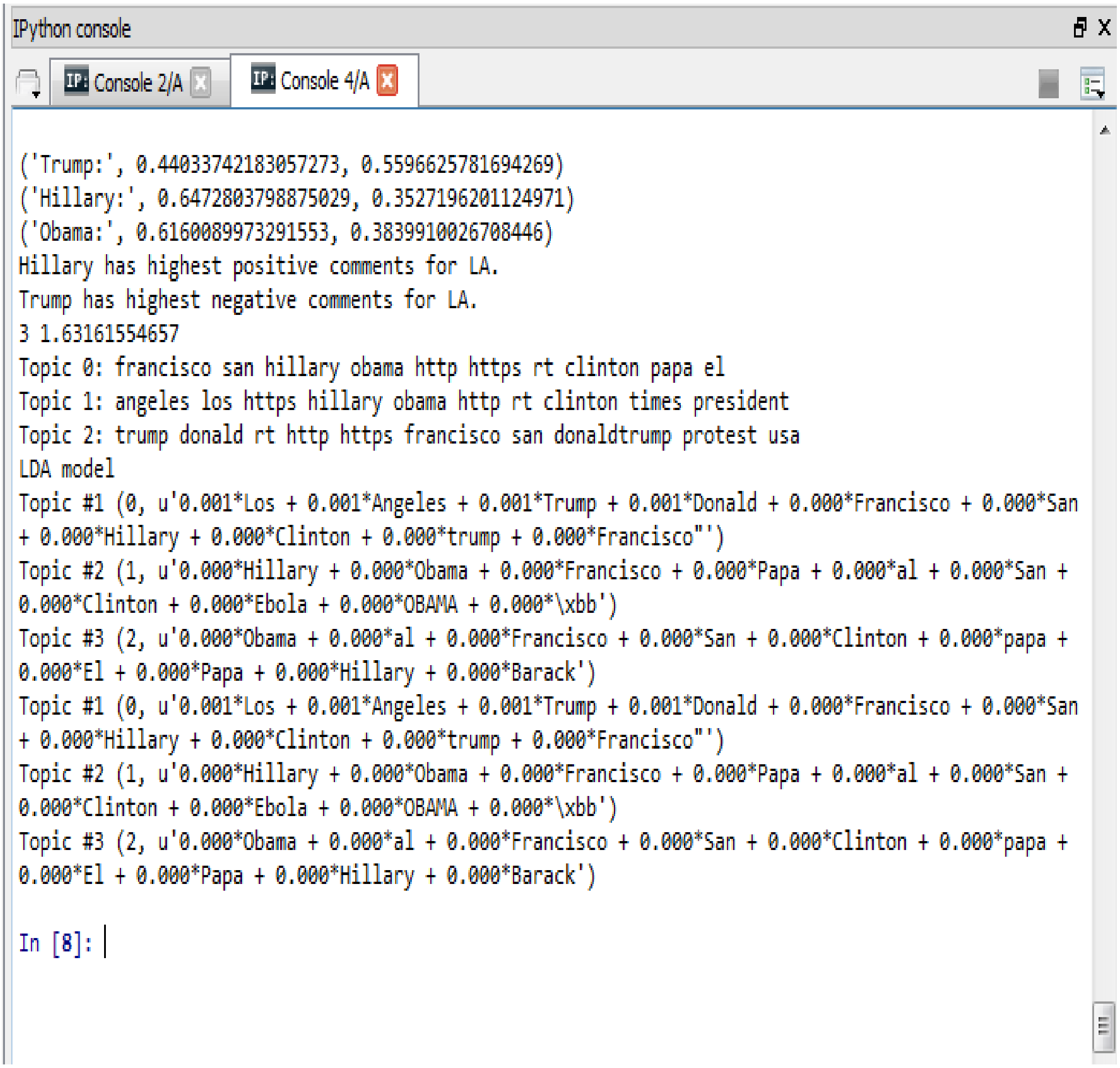


```
tweet_stream_hillary_LA.json
```

c. Barack Obama



tweet_stream_obama_LA.json



```
('Trump:', 0.44033742183057273, 0.5596625781694269)
('Hillary:', 0.6472803798875029, 0.352719620112497l)
('Obama:', 0.6160089973291553, 0.3839910026708446)
Hillary has highest positive comments for LA.
Trump has highest negative comments for LA.
3 1.63161554657
Topic 0: francisco san hillary obama http https rt clinton papa el
Topic 1: angeles los https hillary obama http rt clinton times president
Topic 2: trump donald rt http https francisco san donaldtrump protest usa
LDA model
Topic #1 (0, u'0.001*Los + 0.001*Angeles + 0.001*Trump + 0.001*Donald + 0.000*Francisco + 0.000*San
+ 0.000*Hillary + 0.000*Clinton + 0.000*trump + 0.000*Francisco"')
Topic #2 (1, u'0.000*Hillary + 0.000*Obama + 0.000*Francisco + 0.000*Papa + 0.000*al + 0.000*San +
0.000*Clinton + 0.000*Ebola + 0.000*OBAMA + 0.000*\xbb')
Topic #3 (2, u'0.000*Obama + 0.000*al + 0.000*Francisco + 0.000*San + 0.000*Clinton + 0.000*papa +
0.000*El + 0.000*Papa + 0.000*Hillary + 0.000*Barack')
Topic #1 (0, u'0.001*Los + 0.001*Angeles + 0.001*Trump + 0.001*Donald + 0.000*Francisco + 0.000*San
+ 0.000*Hillary + 0.000*Clinton + 0.000*trump + 0.000*Francisco"')
Topic #2 (1, u'0.000*Hillary + 0.000*Obama + 0.000*Francisco + 0.000*Papa + 0.000*al + 0.000*San +
0.000*Clinton + 0.000*Ebola + 0.000*OBAMA + 0.000*\xbb')
Topic #3 (2, u'0.000*Obama + 0.000*al + 0.000*Francisco + 0.000*San + 0.000*Clinton + 0.000*papa +
0.000*El + 0.000*Papa + 0.000*Hillary + 0.000*Barack')

In [8]:
```

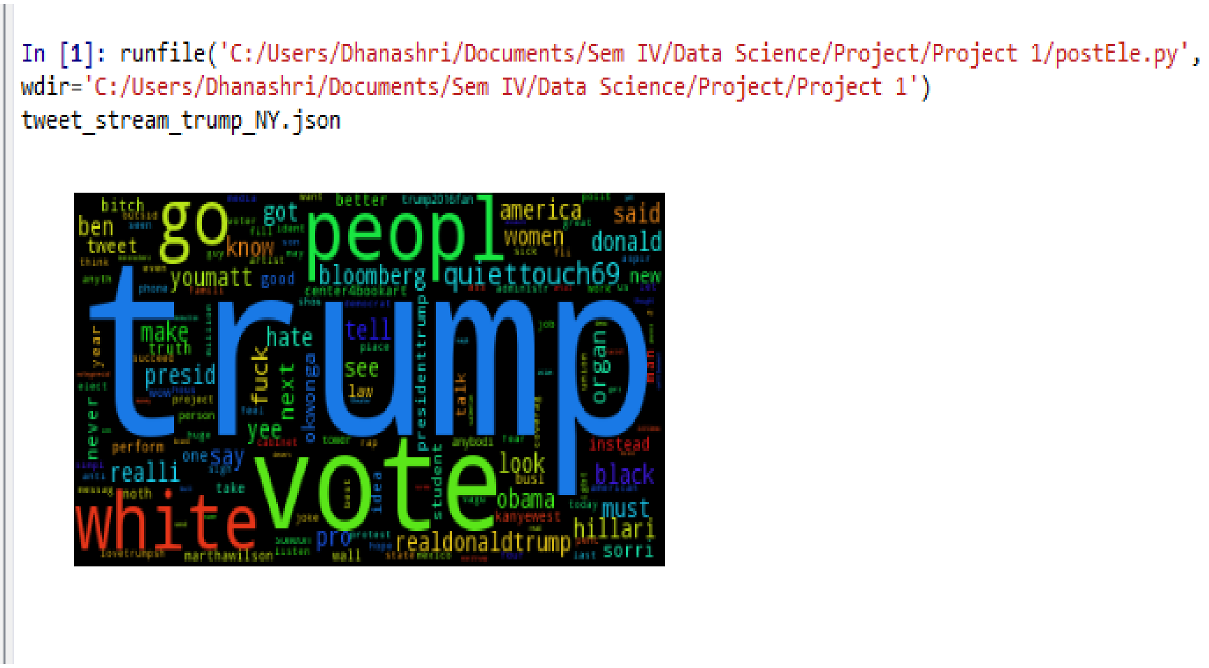
Most positive/negative tweets in Los Angeles, CA.

Most positive tweets: **Hillary Clinton**

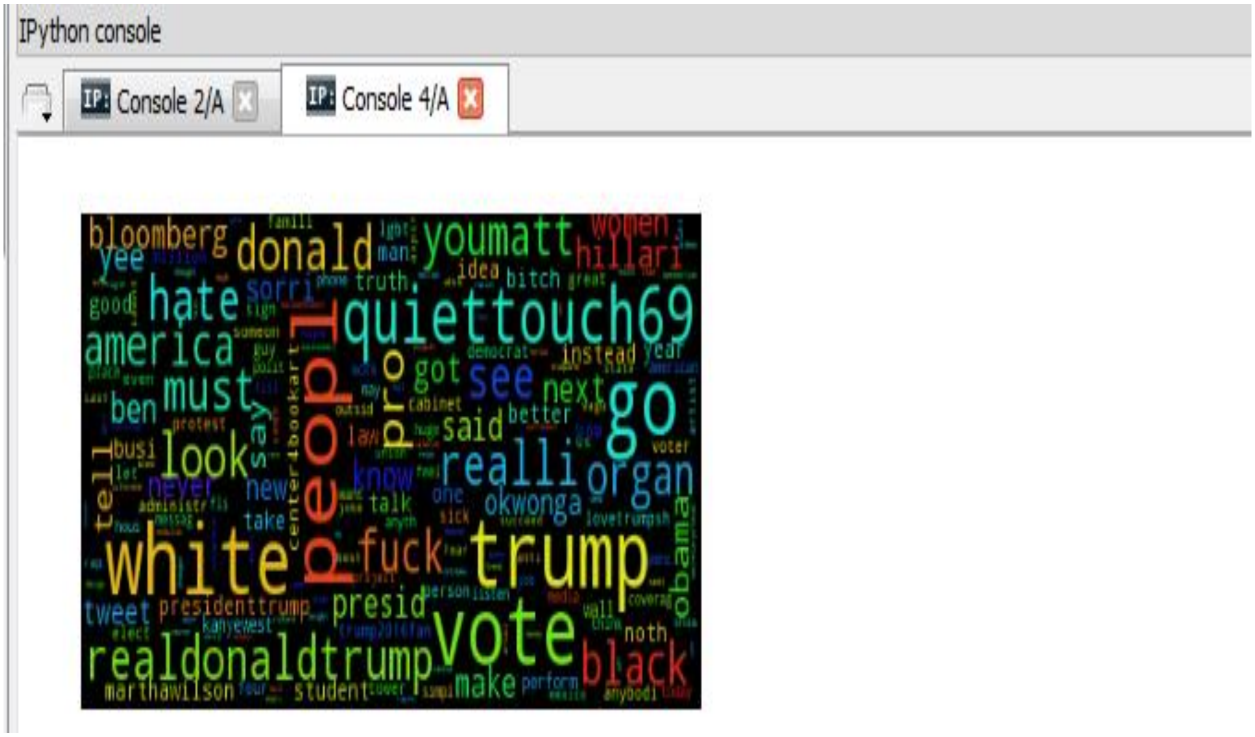
Most negative tweets: **Donald Trump**

5. New York, NY

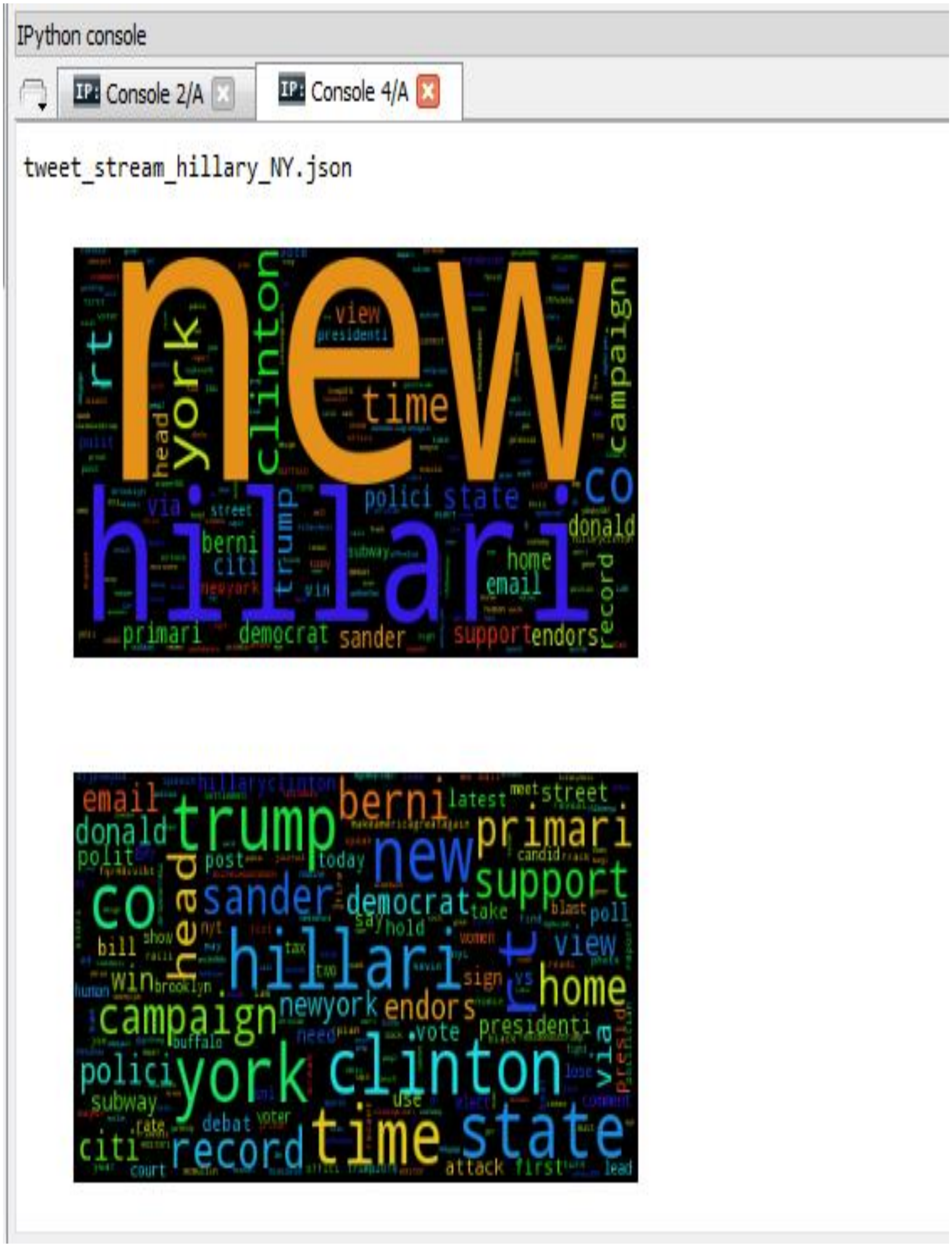
a. Donald Trump



```
In [1]: runfile('C:/Users/Dhanashri/Documents/Sem IV/Data Science/Project/Project 1/postEle.py',
wdir='C:/Users/Dhanashri/Documents/Sem IV/Data Science/Project/Project 1')
tweet_stream_trump_NY.json
```

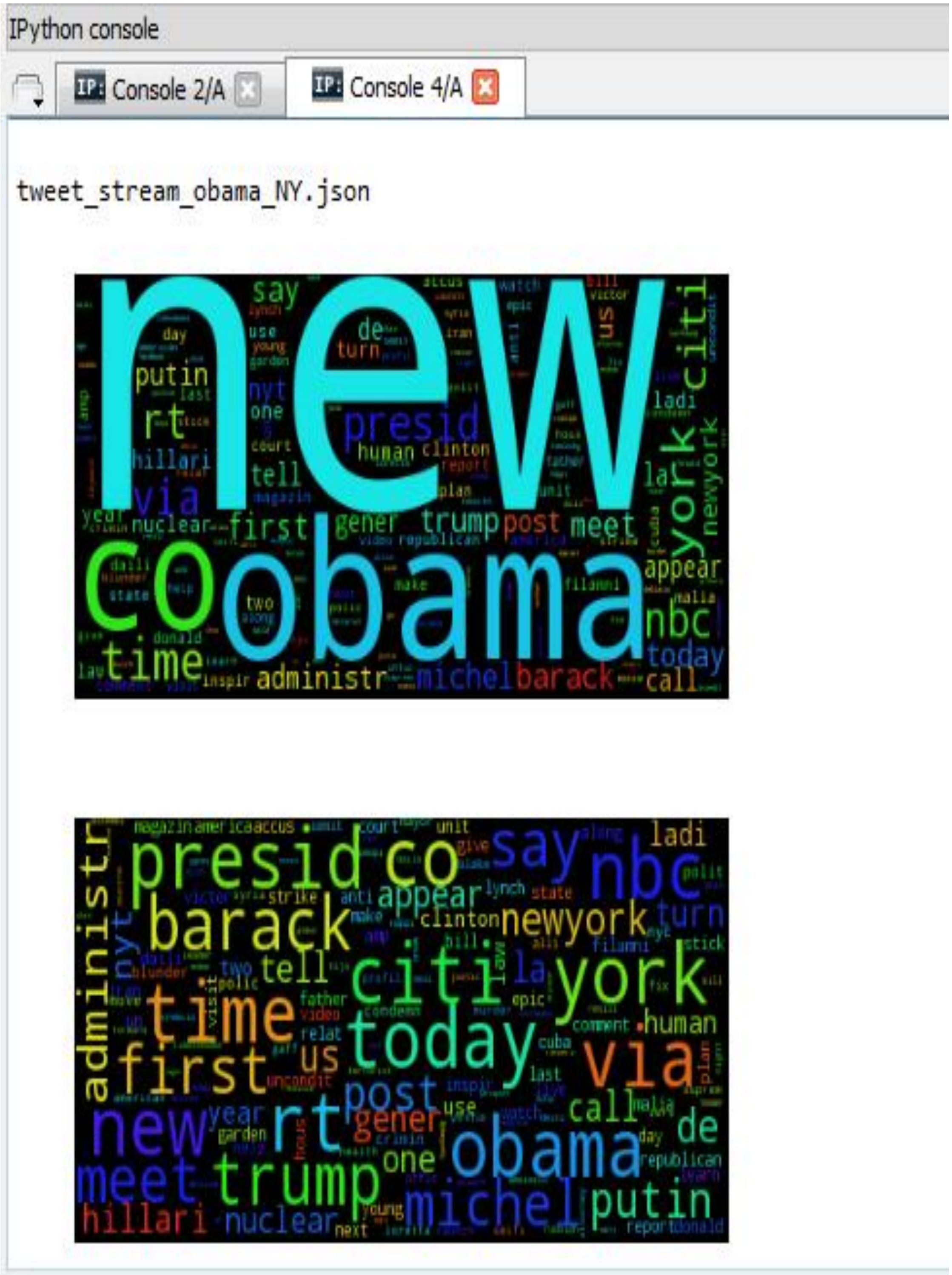


b. Hillary Clinton

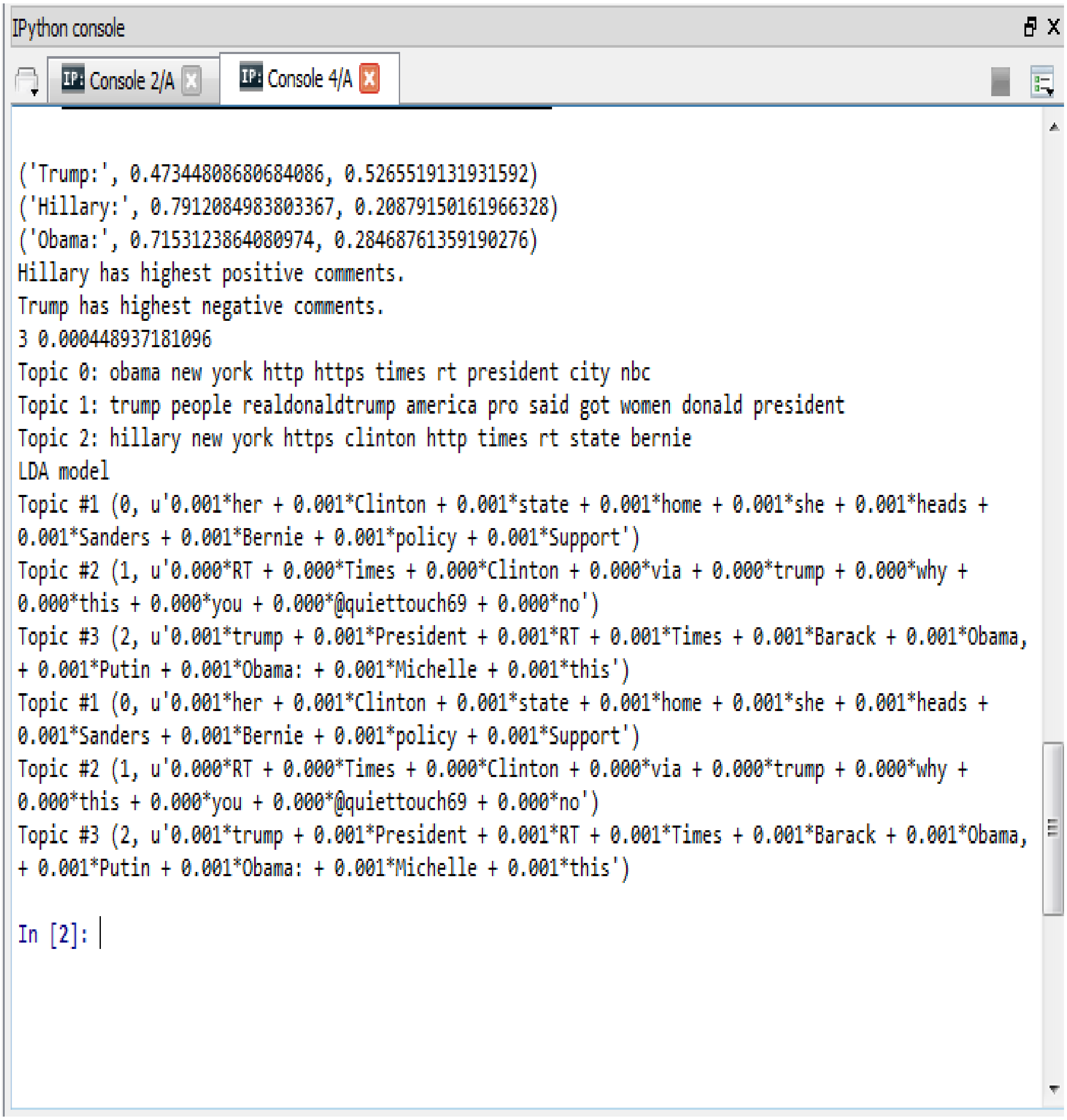


```
tweet_stream_hillary_NY.json
```

c. Barack Obama



```
tweet_stream_obama_NY.json
```



```
('Trump:', 0.47344808680684086, 0.5265519131931592)
('Hillary:', 0.7912084983803367, 0.20879150161966328)
('Obama:', 0.7153123864080974, 0.28468761359190276)
Hillary has highest positive comments.
Trump has highest negative comments.
3 0.000448937181096
Topic 0: obama new york http https times rt president city nbc
Topic 1: trump people realdonaldtrump america pro said got women donald president
Topic 2: hillary new york https clinton http times rt state bernie
LDA model
Topic #1 (0, u'0.001*her + 0.001*Clinton + 0.001*state + 0.001*home + 0.001*she + 0.001*heads +
0.001*Sanders + 0.001*Bernie + 0.001*policy + 0.001*Support')
Topic #2 (1, u'0.000*RT + 0.000*Times + 0.000*Clinton + 0.000*via + 0.000*trump + 0.000*why +
0.000*this + 0.000*you + 0.000*@quiettouch69 + 0.000*no')
Topic #3 (2, u'0.001*trump + 0.001*President + 0.001*RT + 0.001*Times + 0.001*Barack + 0.001*Obama,
+ 0.001*Putin + 0.001*Obama: + 0.001*Michelle + 0.001*this')
Topic #1 (0, u'0.001*her + 0.001*Clinton + 0.001*state + 0.001*home + 0.001*she + 0.001*heads +
0.001*Sanders + 0.001*Bernie + 0.001*policy + 0.001*Support')
Topic #2 (1, u'0.000*RT + 0.000*Times + 0.000*Clinton + 0.000*via + 0.000*trump + 0.000*why +
0.000*this + 0.000*you + 0.000*@quiettouch69 + 0.000*no')
Topic #3 (2, u'0.001*trump + 0.001*President + 0.001*RT + 0.001*Times + 0.001*Barack + 0.001*Obama,
+ 0.001*Putin + 0.001*Obama: + 0.001*Michelle + 0.001*this')

In [2]:
```

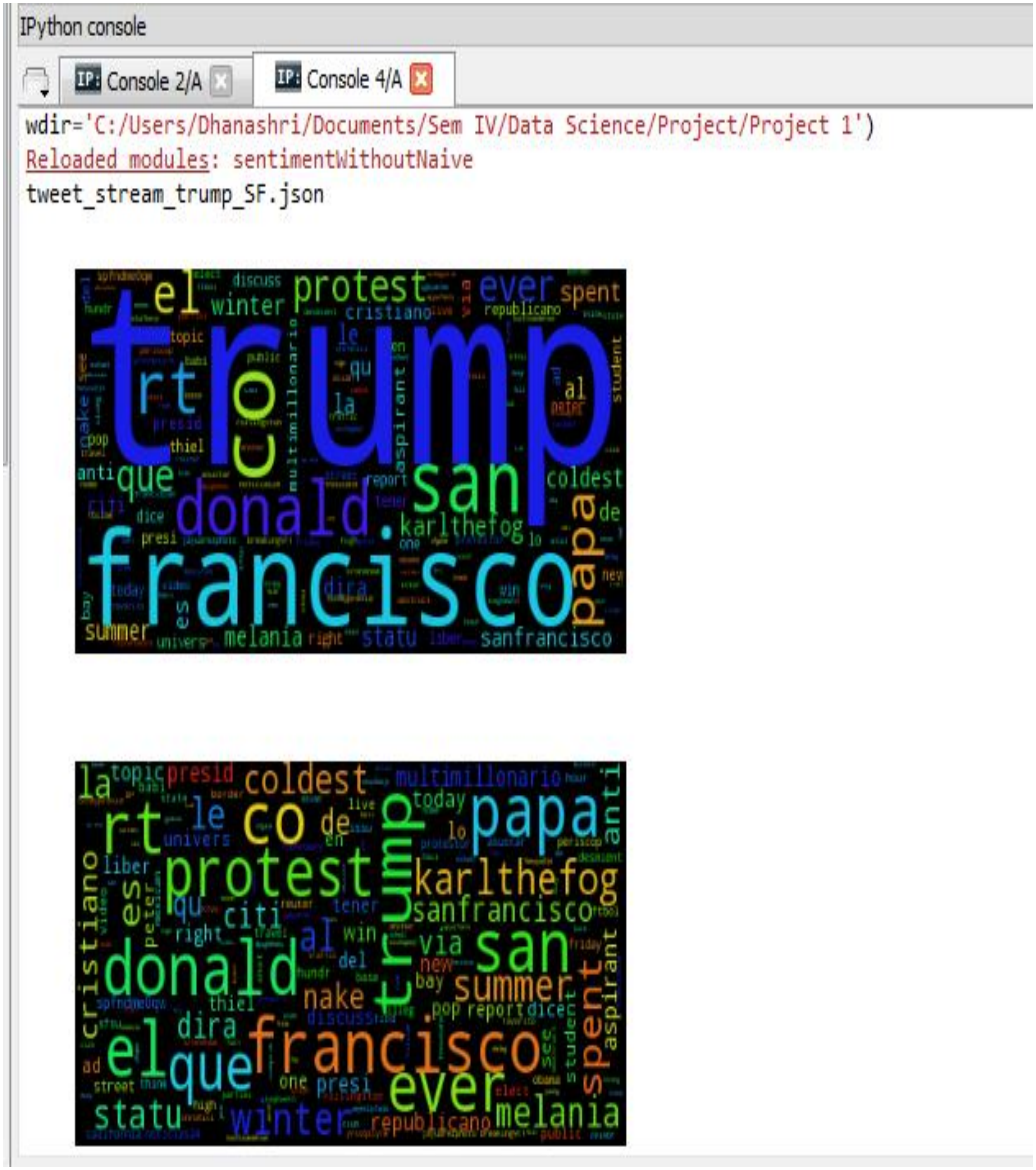
Most positive/negative tweets in New York, NY.

Most positive tweets: **Hillary Clinton**

Most negative tweets: **Donald Trump**

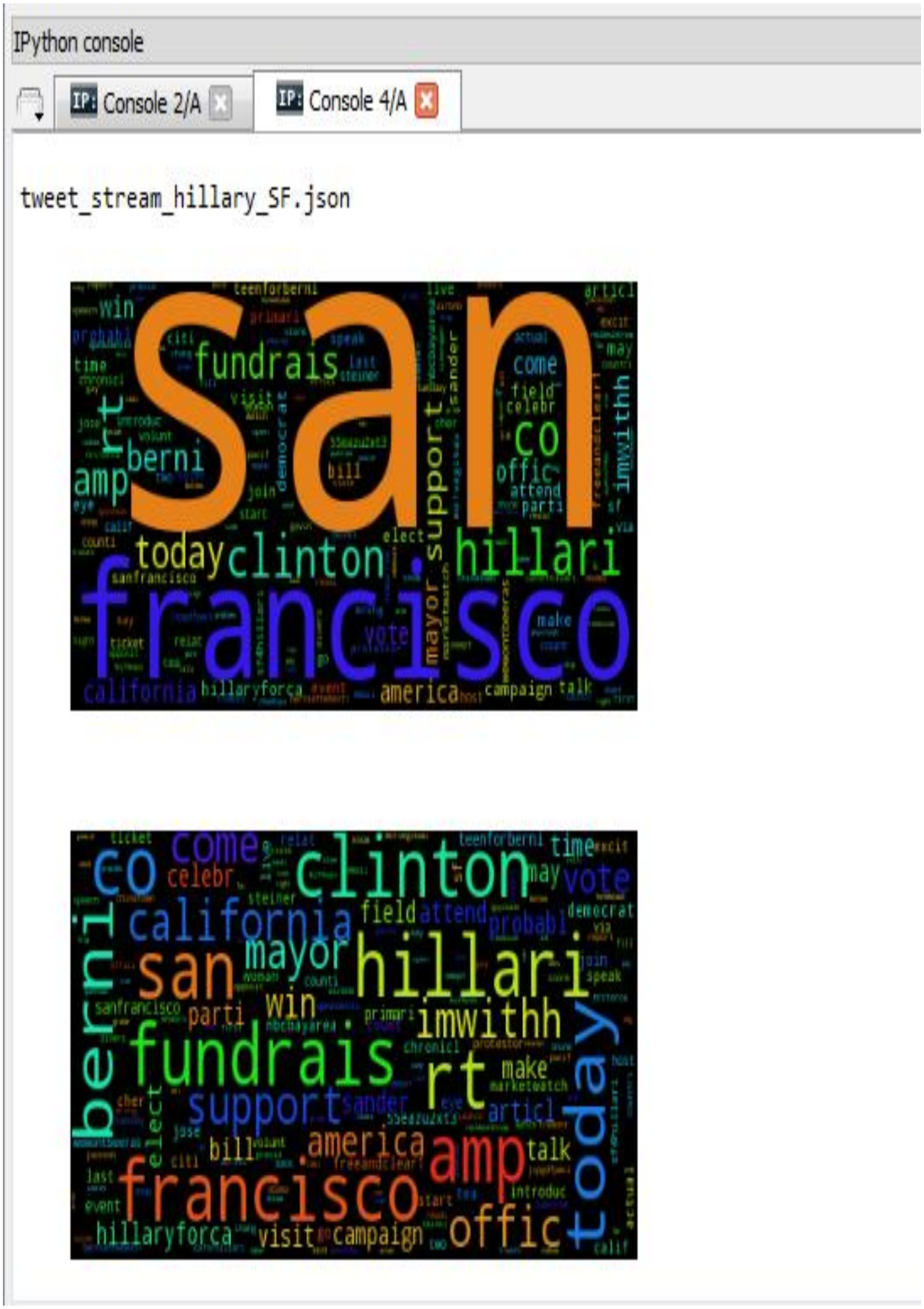
6. San Francisco, California

a. Donald Trump



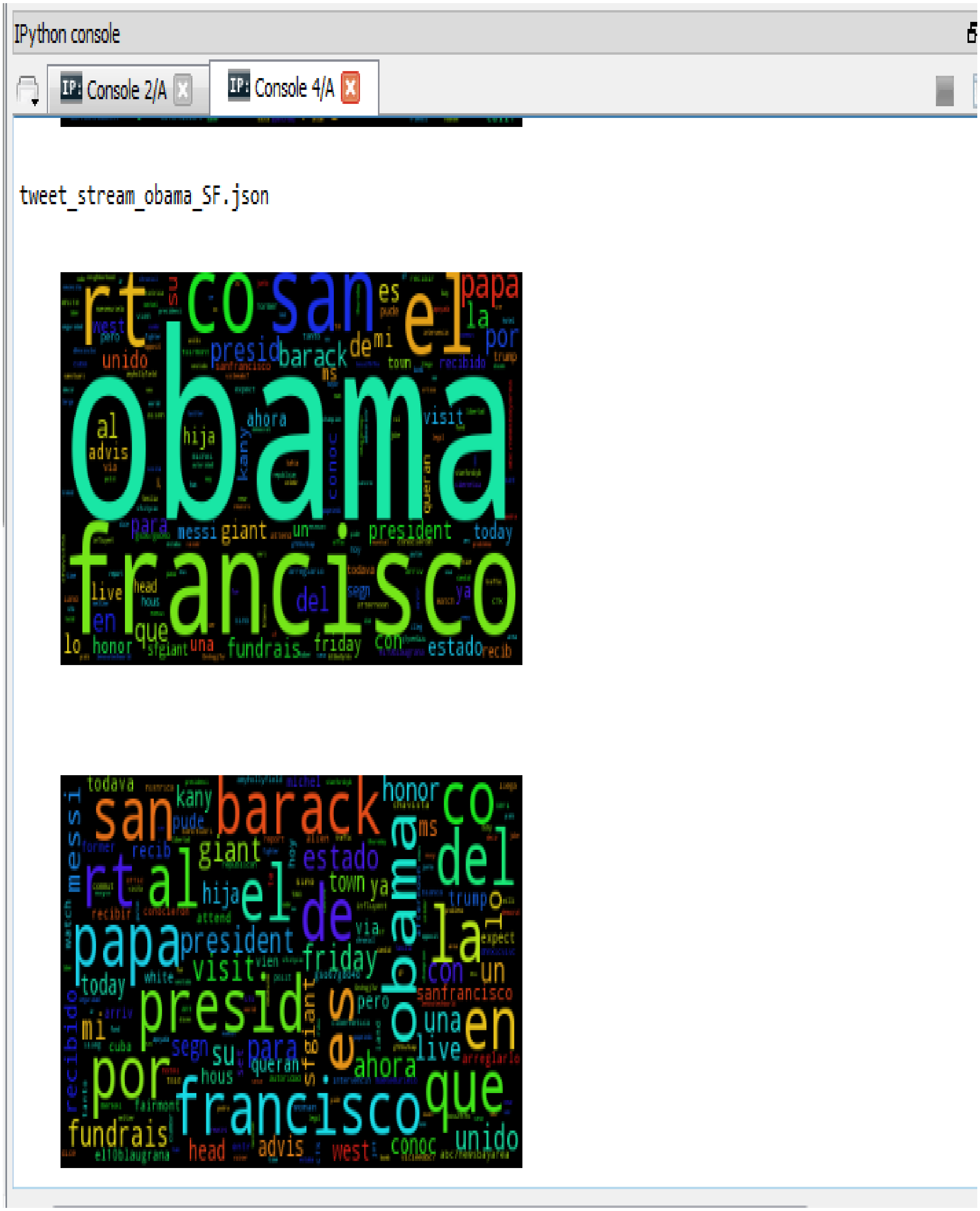
```
wdir='C:/Users/Dhanashri/Documents/Sem IV/Data Science/Project/Project 1')
Reloaded modules: sentimentWithoutNaive
tweet_stream_trump_SF.json
```

b. Hillary Clinton

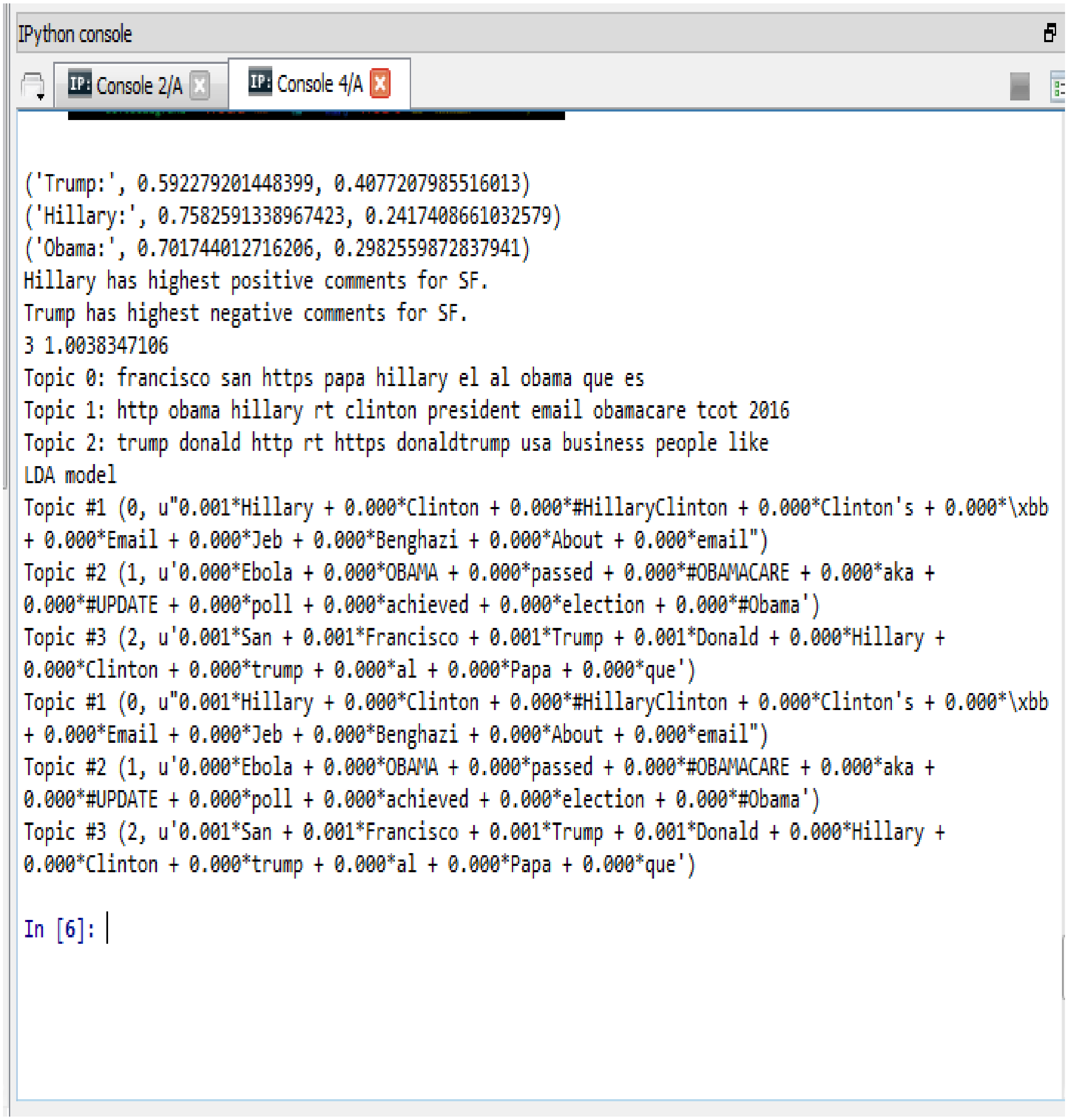


```
tweet_stream_hillary_SF.json
```

c. Barack Obama



tweet_stream_obama_SF.json



```
('Trump:', 0.592279201448399, 0.4077207985516013)
('Hillary:', 0.7582591338967423, 0.2417408661032579)
('Obama:', 0.701744012716206, 0.2982559872837941)
Hillary has highest positive comments for SF.
Trump has highest negative comments for SF.
3 1.0038347106
Topic 0: francisco san https papa hillary el al obama que es
Topic 1: http obama hillary rt clinton president email obamacare tcot 2016
Topic 2: trump donald http rt https donaldtrump usa business people like
LDA model
Topic #1 (0, u"0.001*Hillary + 0.000*Clinton + 0.000*#HillaryClinton + 0.000*Clinton's + 0.000*\xbb
+ 0.000*Email + 0.000*Jeb + 0.000*Benghazi + 0.000*About + 0.000*email")
Topic #2 (1, u'0.000*Ebola + 0.000*OBAMA + 0.000*passed + 0.000*#OBAMACARE + 0.000*aka +
0.000*#UPDATE + 0.000*poll + 0.000*achieved + 0.000*election + 0.000*#Obama')
Topic #3 (2, u'0.001*San + 0.001*Francisco + 0.001*Trump + 0.001*Donald + 0.000*Hillary +
0.000*Clinton + 0.000*trump + 0.000*al + 0.000*Papa + 0.000*que')
Topic #1 (0, u"0.001*Hillary + 0.000*Clinton + 0.000*#HillaryClinton + 0.000*Clinton's + 0.000*\xbb
+ 0.000*Email + 0.000*Jeb + 0.000*Benghazi + 0.000*About + 0.000*email")
Topic #2 (1, u'0.000*Ebola + 0.000*OBAMA + 0.000*passed + 0.000*#OBAMACARE + 0.000*aka +
0.000*#UPDATE + 0.000*poll + 0.000*achieved + 0.000*election + 0.000*#Obama')
Topic #3 (2, u'0.001*San + 0.001*Francisco + 0.001*Trump + 0.001*Donald + 0.000*Hillary +
0.000*Clinton + 0.000*trump + 0.000*al + 0.000*Papa + 0.000*que')

In [6]:
```

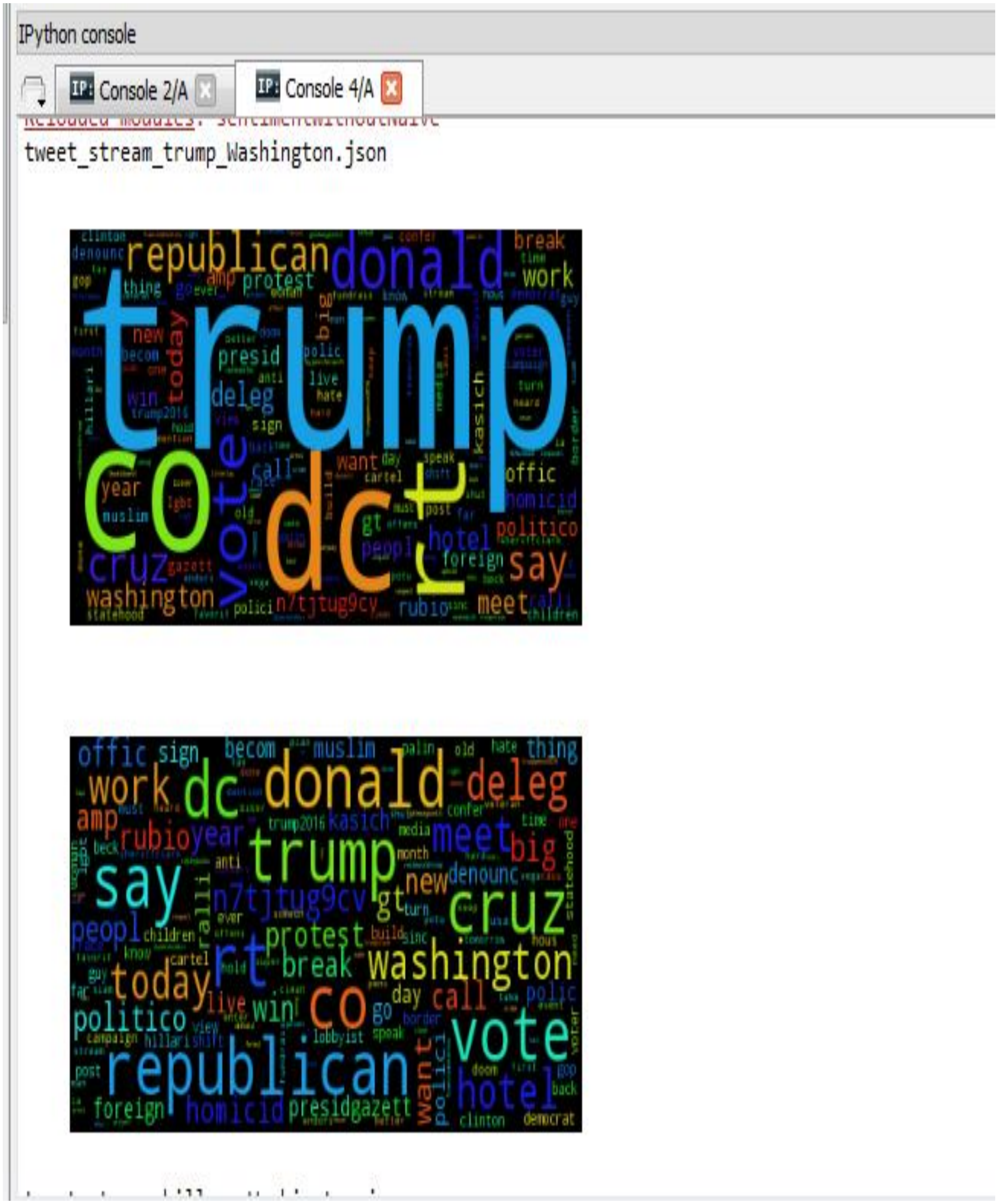
Most positive/negative tweets in San Francisco, CA.

Most positive tweets: **Hillary Clinton**

Most negative tweets: **Donald Trump**

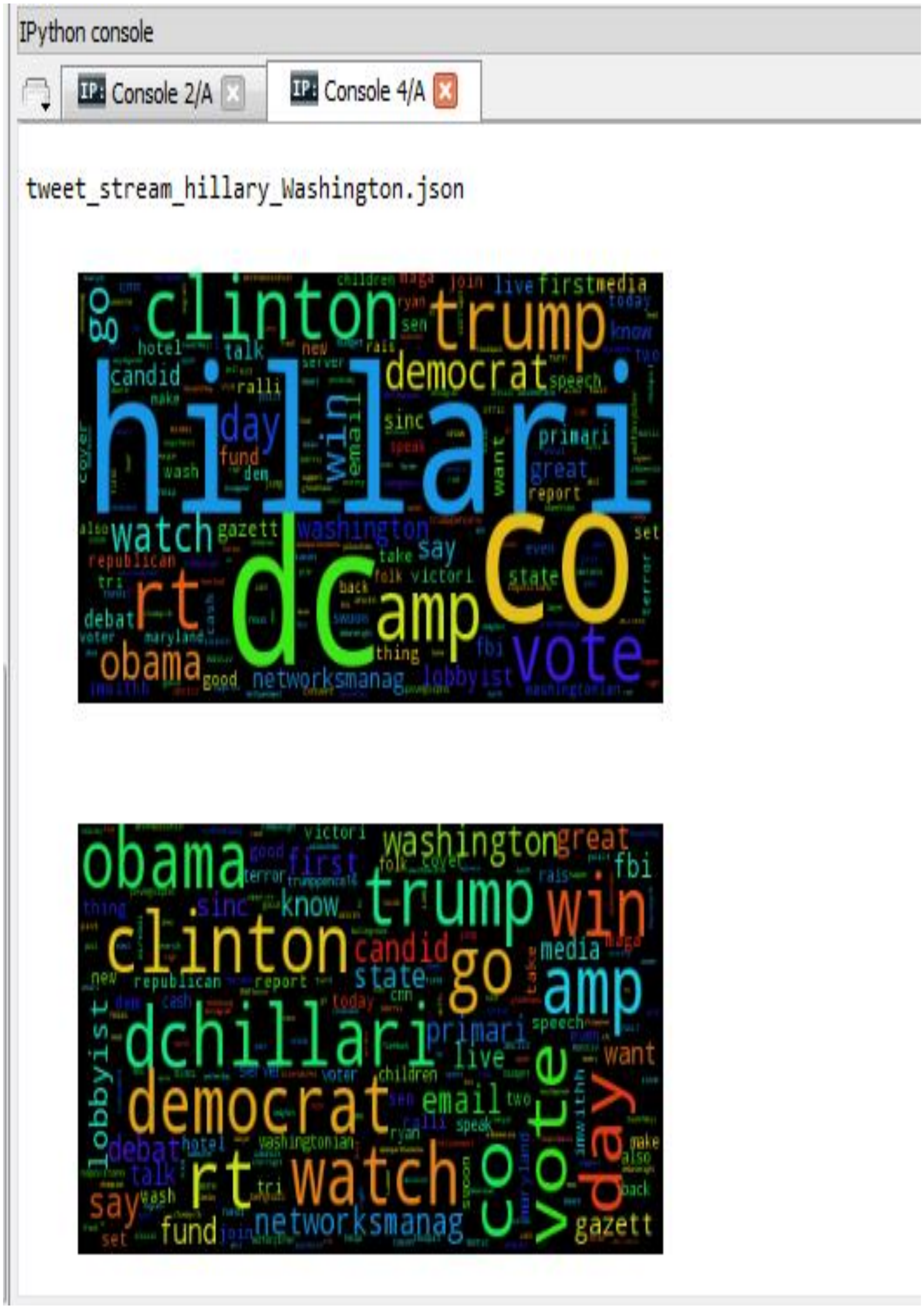
7. Washington DC

a. Donald Trump



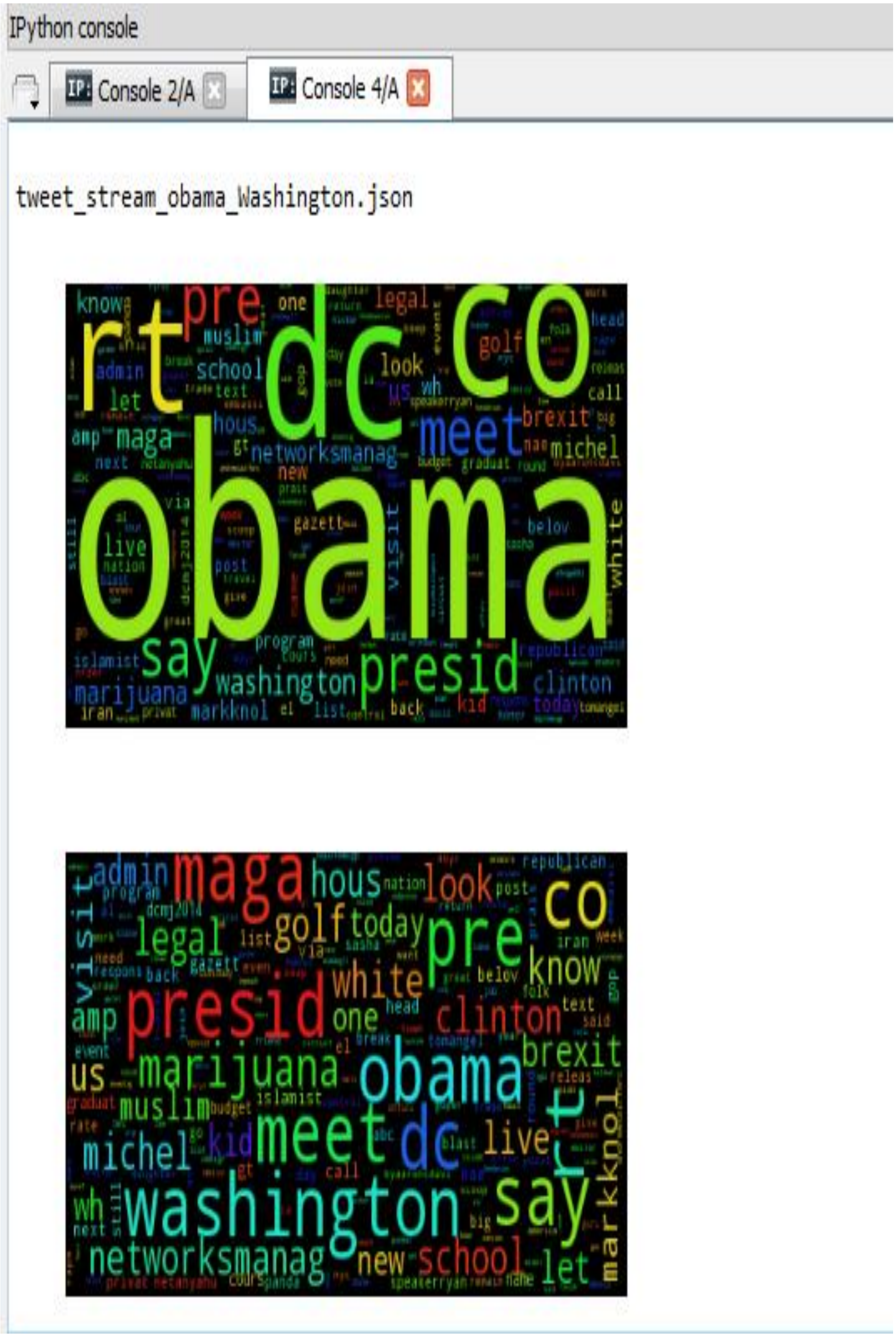
```
Reloaded modules: sentimentwithoutNaive
tweet_stream_trump_Washington.json
```

b. Hillary Clinton

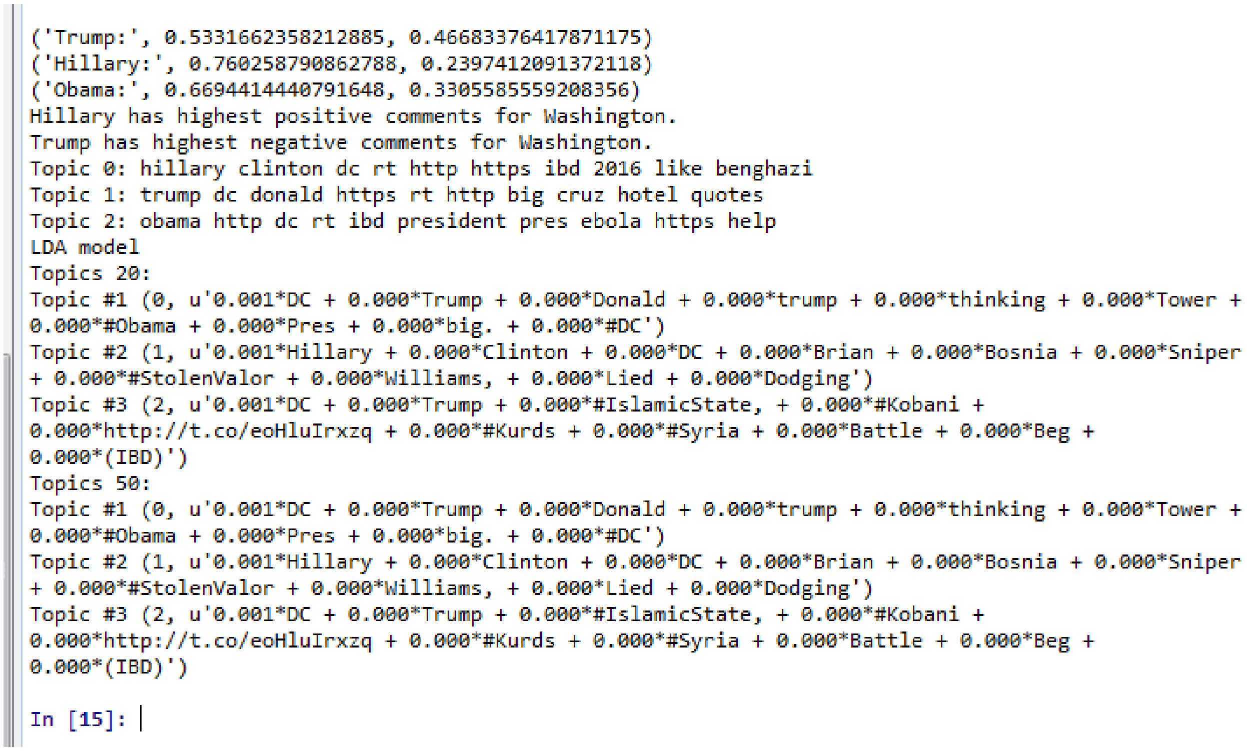


```
tweet_stream_hillary_Washington.json
```

c. Barack Obama



```
tweet_stream_obama_Washington.json
```



```
('Trump:', 0.5331662358212885, 0.46683376417871175)
('Hillary:', 0.760258790862788, 0.2397412091372118)
('Obama:', 0.6694414440791648, 0.3305585559208356)
Hillary has highest positive comments for Washington.
Trump has highest negative comments for Washington.
Topic 0: hillary clinton dc rt http https ibd 2016 like benghazi
Topic 1: trump dc donald https rt http big cruz hotel quotes
Topic 2: obama http dc rt ibd president pres ebola https help
LDA model
Topics 20:
Topic #1 (0, u'0.001*DC + 0.000*Trump + 0.000*Donald + 0.000*trump + 0.000*thinking + 0.000*Tower +
0.000*#Obama + 0.000*Pres + 0.000*big. + 0.000*#DC')
Topic #2 (1, u'0.001*Hillary + 0.000*Clinton + 0.000*DC + 0.000*Brian + 0.000*Bosnia + 0.000*Sniper
+ 0.000*#StolenValor + 0.000*Williams, + 0.000*Lied + 0.000*Dodging')
Topic #3 (2, u'0.001*DC + 0.000*Trump + 0.000*#IslamicState, + 0.000*#Kobani +
0.000*http://t.co/eoHluIrxzq + 0.000*#Kurds + 0.000*#Syria + 0.000*Battle + 0.000*Beg +
0.000*(IBD)')
Topics 50:
Topic #1 (0, u'0.001*DC + 0.000*Trump + 0.000*Donald + 0.000*trump + 0.000*thinking + 0.000*Tower +
0.000*#Obama + 0.000*Pres + 0.000*big. + 0.000*#DC')
Topic #2 (1, u'0.001*Hillary + 0.000*Clinton + 0.000*DC + 0.000*Brian + 0.000*Bosnia + 0.000*Sniper
+ 0.000*#StolenValor + 0.000*Williams, + 0.000*Lied + 0.000*Dodging')
Topic #3 (2, u'0.001*DC + 0.000*Trump + 0.000*#IslamicState, + 0.000*#Kobani +
0.000*http://t.co/eoHluIrxzq + 0.000*#Kurds + 0.000*#Syria + 0.000*Battle + 0.000*Beg +
0.000*(IBD)')

In [15]:
```

Highest positive/negative tweets in Washington DC.

Most positive tweets: **Hillary Clinton**

Most negative tweets: **Donald Trump**

6. Topic Modeling

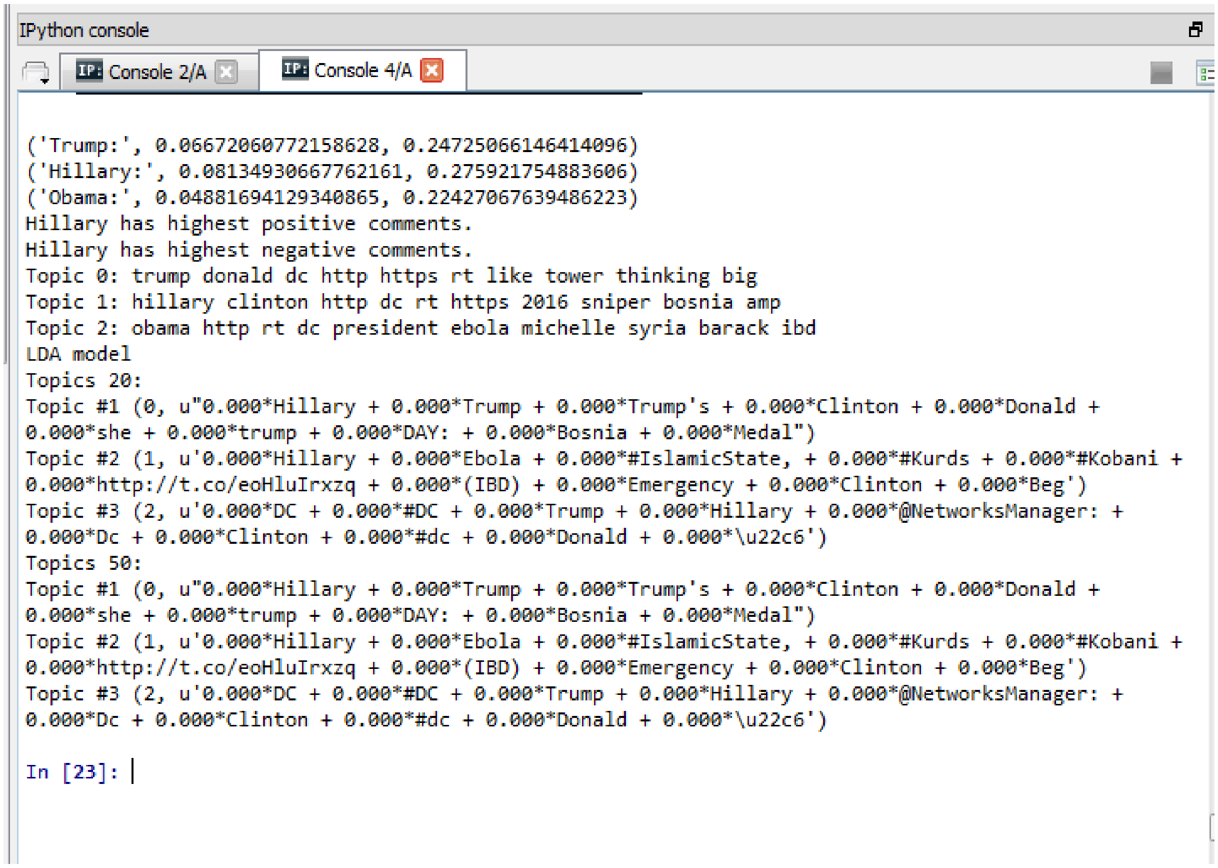
Finally, by using the 30,000 tweet text corpus from Step 1, we ran topic modeling with 20 and 50 topics using NMF and LDA from GENSIM.

i. Non-Negative Matrix Factorization (NMF):

A “non-negative matrix” is a matrix containing non-negative values (here, zero or positive word frequencies). And factorization refers to familiar kind of mathematical factorization. NMF is part of the GENSIM package in Python.
We will use NMF to get a document-topic matrix and a list of top words for each topic.

i. Latent Dirichlet Allocation (LDA):

LDA is a topic model that generates topics based on word frequency from a set of documents. LDA is particularly useful for finding reasonably accurate mixture of topics within a given document set. Like NMF, LDA is also part of the topic modeling package called GENSIM.



```
('Trump:', 0.06672060772158628, 0.24725066146414096)
('Hillary:', 0.08134930667762161, 0.27592175488360б)
('Obama:', 0.04881694129340865, 0.22427067639486223)
Hillary has highest positive comments.
Hillary has highest negative comments.
Topic 0: trump donald dc http https rt like tower thinking big
Topic 1: hillary clinton http dc rt https 2016 sniper bosnia amp
Topic 2: obama http rt dc president ebola michelle syria barack ibd
LDA model
Topics 20:
Topic #1 (0, u"0.000*Hillary + 0.000*Trump + 0.000*Trump's + 0.000*Clinton + 0.000*Donald +
0.000*she + 0.000*trump + 0.000*DAY: + 0.000*Bosnia + 0.000*Medal")
Topic #2 (1, u'0.000*Hillary + 0.000*Ebola + 0.000*#IslamicState, + 0.000*#Kurds + 0.000*#Kobani +
0.000*http://t.co/eoHluIrxzq + 0.000*(IBD) + 0.000*Emergency + 0.000*Clinton + 0.000*Beg')
Topic #3 (2, u'0.000*DC + 0.000*#DC + 0.000*Trump + 0.000*Hillary + 0.000*@NetworksManager: +
0.000*Dc + 0.000*Clinton + 0.000*#dc + 0.000*Donald + 0.000*\u22c6')
Topics 50:
Topic #1 (0, u"0.000*Hillary + 0.000*Trump + 0.000*Trump's + 0.000*Clinton + 0.000*Donald +
0.000*she + 0.000*trump + 0.000*DAY: + 0.000*Bosnia + 0.000*Medal")
Topic #2 (1, u'0.000*Hillary + 0.000*Ebola + 0.000*#IslamicState, + 0.000*#Kurds + 0.000*#Kobani +
0.000*http://t.co/eoHluIrxzq + 0.000*(IBD) + 0.000*Emergency + 0.000*Clinton + 0.000*Beg')
Topic #3 (2, u'0.000*DC + 0.000*#DC + 0.000*Trump + 0.000*Hillary + 0.000*@NetworksManager: +
0.000*Dc + 0.000*Clinton + 0.000*#dc + 0.000*Donald + 0.000*\u22c6')

In [23]:
```

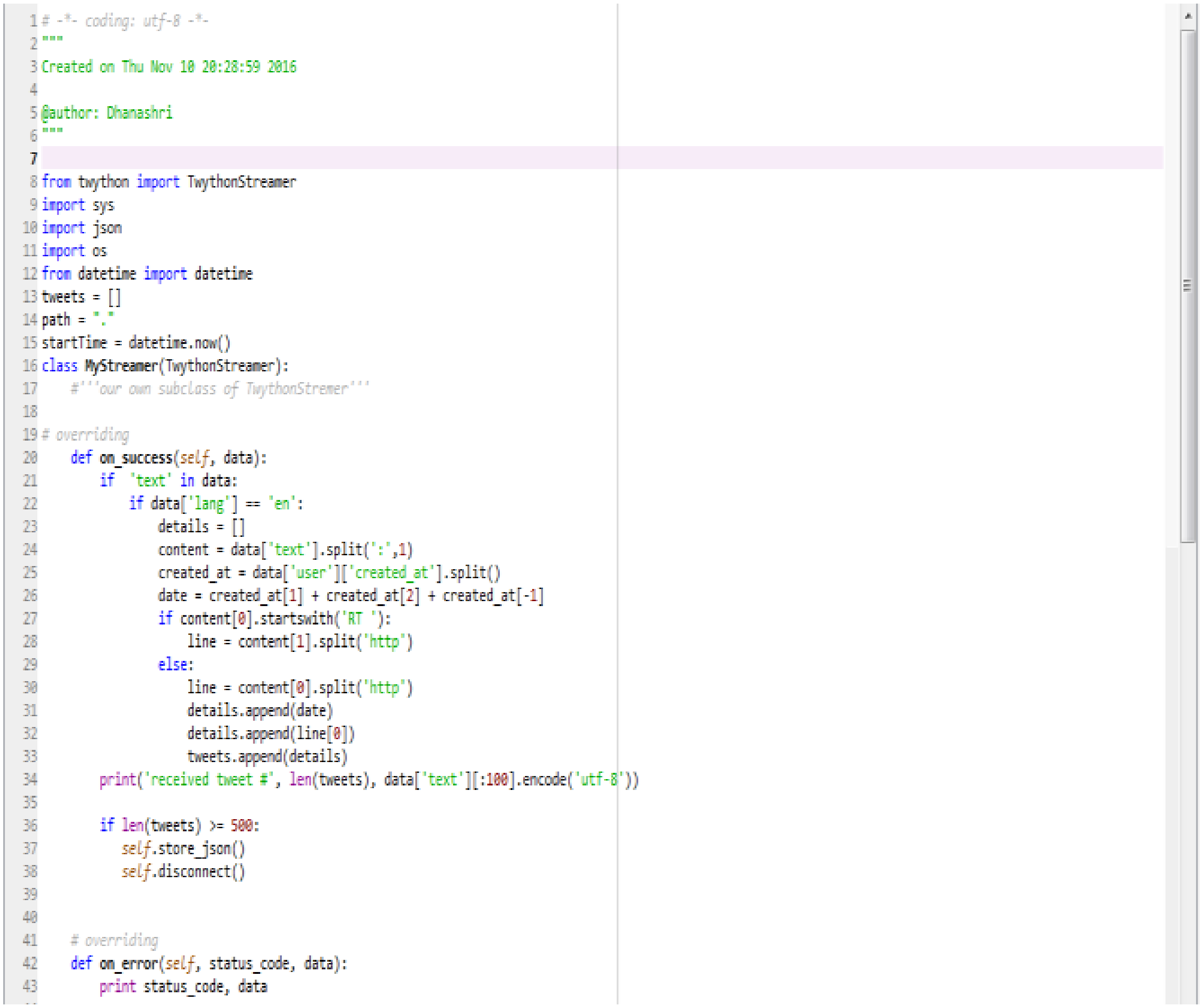
a) Most positive/negative tweets and

b) Topic Modeling with LDA and NMF(20 and 50 topics).

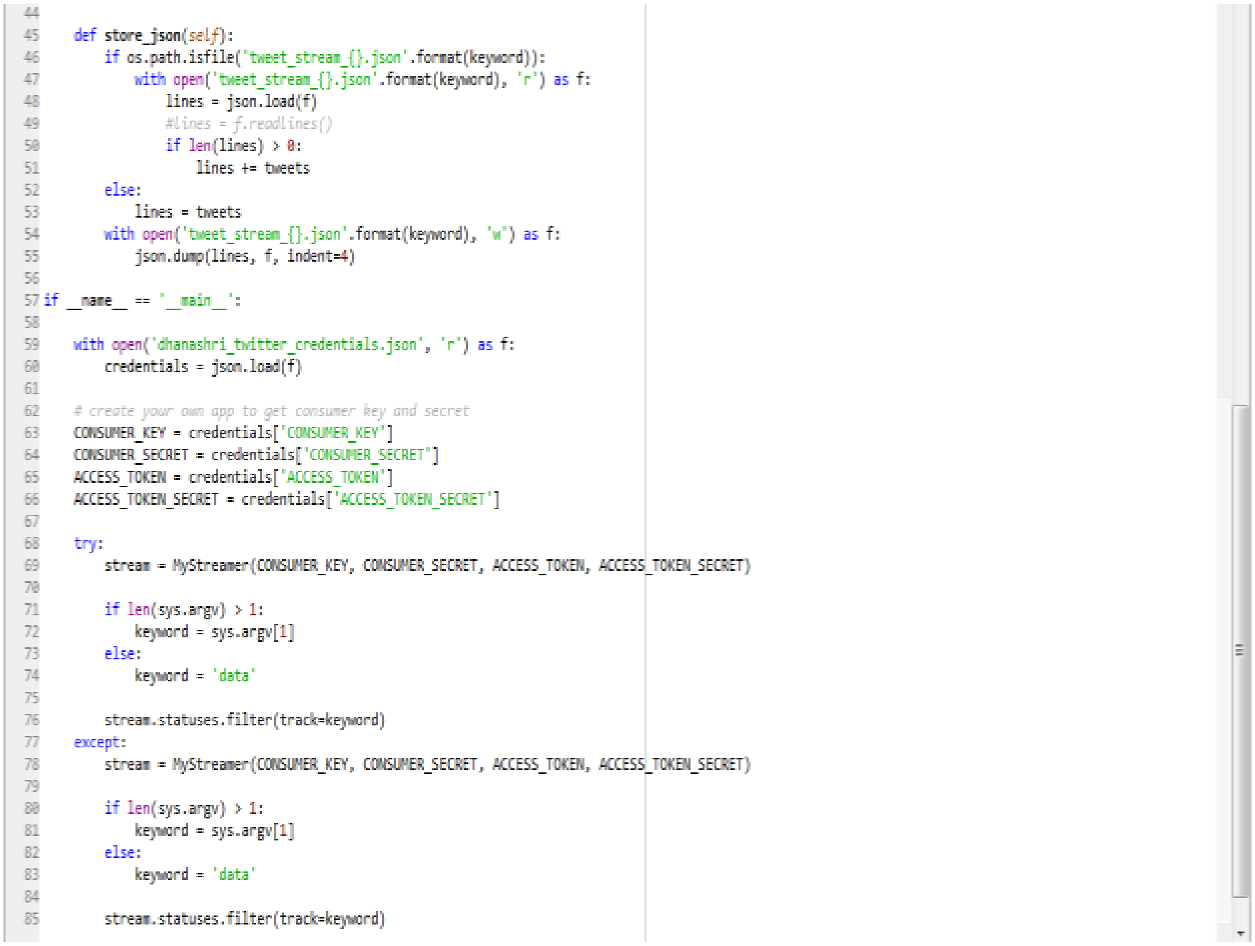
Note: Although we ran the Topic Modeling for 20 and 50 topics we have displayed only 3 topics for each model.

PYTHON SCRIPTS

twitterStream.py

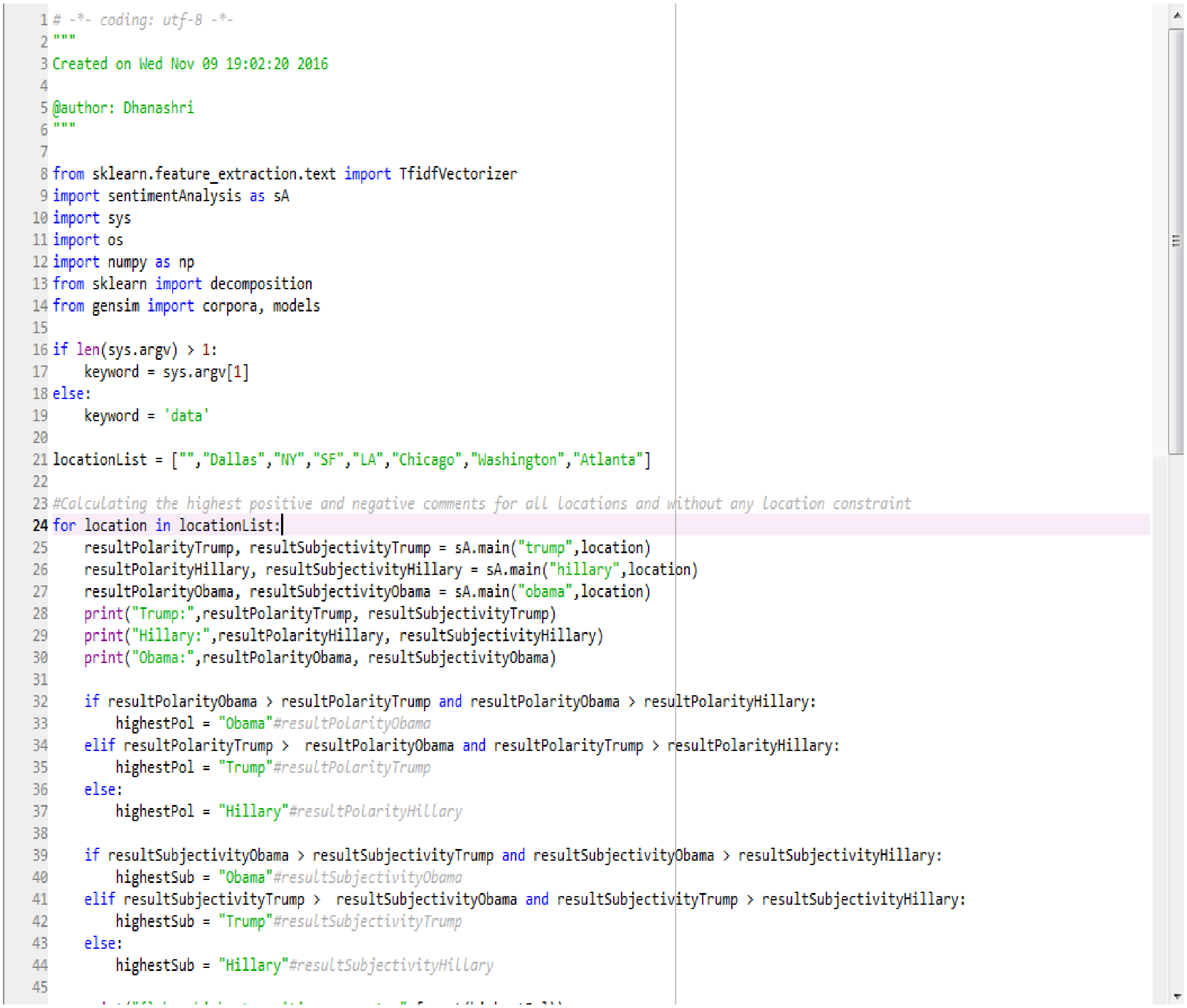


```python
# -*- coding: utf-8 -*-
"""
Created on Thu Nov 10 20:28:59 2016

@author: Dhanashri
"""

from twython import TwythonStreamer
import sys
import json
import os
from datetime import datetime
tweets = []
path = "."
startTime = datetime.now()
class MyStreamer(TwythonStreamer):
    #'''our own subclass of TwythonStremer'''

# overriding
    def on_success(self, data):
        if  'text' in data:
            if data['lang'] == 'en':
                details = []
                content = data['text'].split(':',1)
                created_at = data['user']['created_at'].split()
                date = created_at[1] + created_at[2] + created_at[-1]
                if content[0].startswith('RT '):
                    line = content[1].split('http')
                else:
                    line = content[0].split('http')
                    details.append(date)
                    details.append(line[0])
                    tweets.append(details)
        print('received tweet #', len(tweets), data['text'][:100].encode('utf-8'))

        if len(tweets) >= 500:
           self.store_json()
           self.disconnect()


    # overriding
    def on_error(self, status_code, data):
        print status_code, data
```

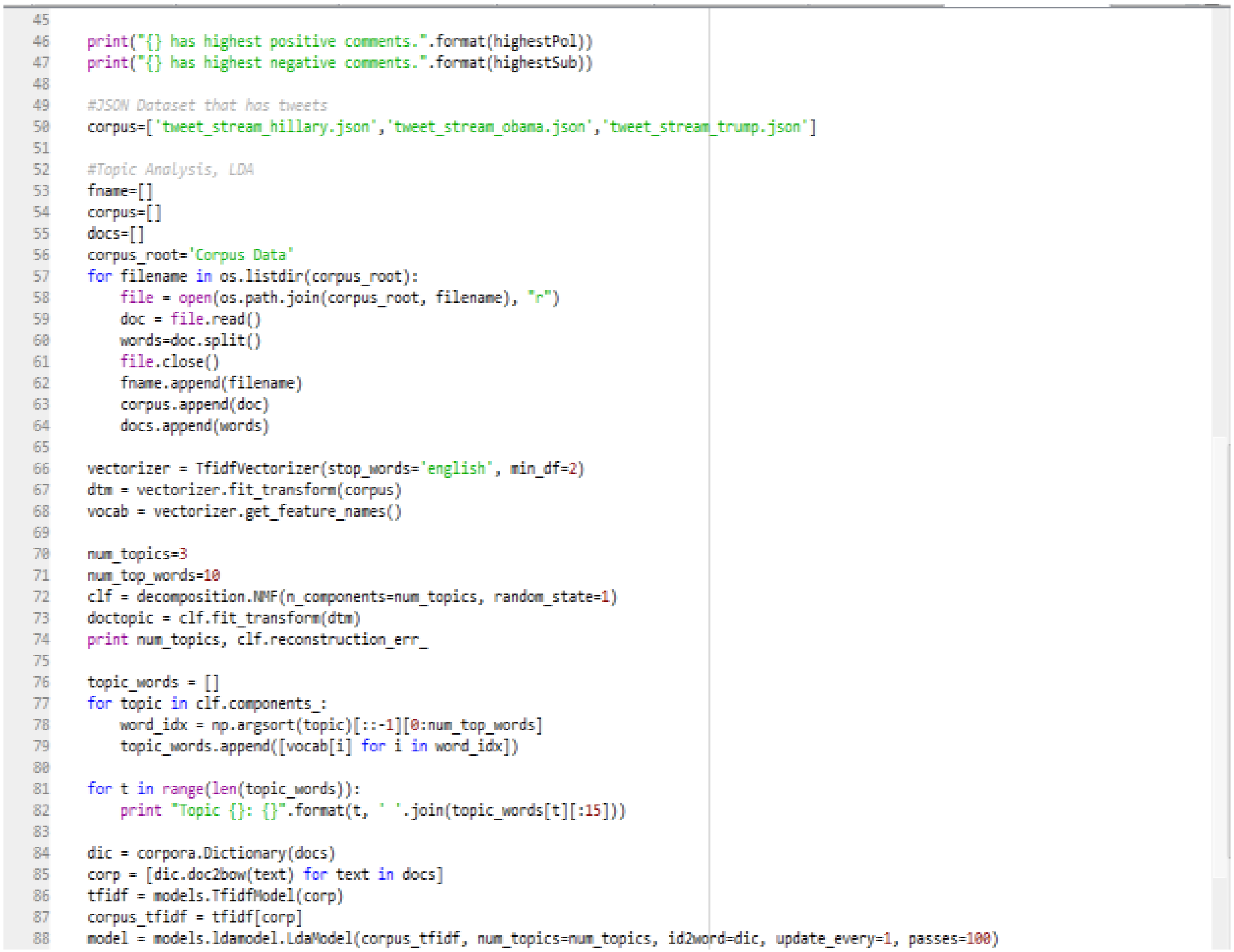


```
    def store_json(self):
        if os.path.isfile('tweet_stream_{}.json'.format(keyword)):
            with open('tweet_stream_{}.json'.format(keyword), 'r') as f:
                lines = json.load(f)
                #lines = f.readlines()
                if len(lines) > 0:
                    lines += tweets
        else:
            lines = tweets
        with open('tweet_stream_{}.json'.format(keyword), 'w') as f:
            json.dump(lines, f, indent=4)

if __name__ == '__main__':

    with open('dhanashri_twitter_credentials.json', 'r') as f:
        credentials = json.load(f)

    # create your own app to get consumer key and secret
    CONSUMER_KEY = credentials['CONSUMER_KEY']
    CONSUMER_SECRET = credentials['CONSUMER_SECRET']
    ACCESS_TOKEN = credentials['ACCESS_TOKEN']
    ACCESS_TOKEN_SECRET = credentials['ACCESS_TOKEN_SECRET']

    try:
        stream = MyStreamer(CONSUMER_KEY, CONSUMER_SECRET, ACCESS_TOKEN, ACCESS_TOKEN_SECRET)

        if len(sys.argv) > 1:
            keyword = sys.argv[1]
        else:
            keyword = 'data'

        stream.statuses.filter(track=keyword)
    except:
        stream = MyStreamer(CONSUMER_KEY, CONSUMER_SECRET, ACCESS_TOKEN, ACCESS_TOKEN_SECRET)

        if len(sys.argv) > 1:
            keyword = sys.argv[1]
        else:
            keyword = 'data'

        stream.statuses.filter(track=keyword)
```

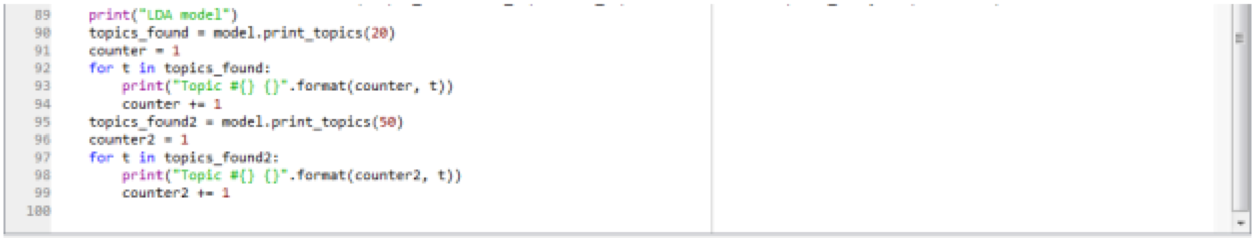
postElection.py



```
# -*- coding: utf-8 -*-
"""
Created on Wed Nov 09 19:02:20 2016

@author: Dhanashri
"""

from sklearn.feature_extraction.text import TfidfVectorizer
import sentimentAnalysis as sA
import sys
import os
import numpy as np
from sklearn import decomposition
from gensim import corpora, models

if len(sys.argv) > 1:
    keyword = sys.argv[1]
else:
    keyword = 'data'

locationList = ["","Dallas","NY","SF","LA","Chicago","Washington","Atlanta"]

#Calculating the highest positive and negative comments for all locations and without any location constraint
for location in locationList:
    resultPolarityTrump, resultSubjectivityTrump = sA.main("trump",location)
    resultPolarityHillary, resultSubjectivityHillary = sA.main("hillary",location)
    resultPolarityObama, resultSubjectivityObama = sA.main("obama",location)
    print("Trump:",resultPolarityTrump, resultSubjectivityTrump)
    print("Hillary:",resultPolarityHillary, resultSubjectivityHillary)
    print("Obama:",resultPolarityObama, resultSubjectivityObama)

    if resultPolarityObama > resultPolarityTrump and resultPolarityObama > resultPolarityHillary:
        highestPol = "Obama"#resultPolarityObama
    elif resultPolarityTrump >  resultPolarityObama and resultPolarityTrump > resultPolarityHillary:
        highestPol = "Trump"#resultPolarityTrump
    else:
        highestPol = "Hillary"#resultPolarityHillary

    if resultSubjectivityObama > resultSubjectivityTrump and resultSubjectivityObama > resultSubjectivityHillary:
        highestSub = "Obama"#resultSubjectivityObama
    elif resultSubjectivityTrump >  resultSubjectivityObama and resultSubjectivityTrump > resultSubjectivityHillary:
        highestSub = "Trump"#resultSubjectivityTrump
    else:
        highestSub = "Hillary"#resultSubjectivityHillary

```

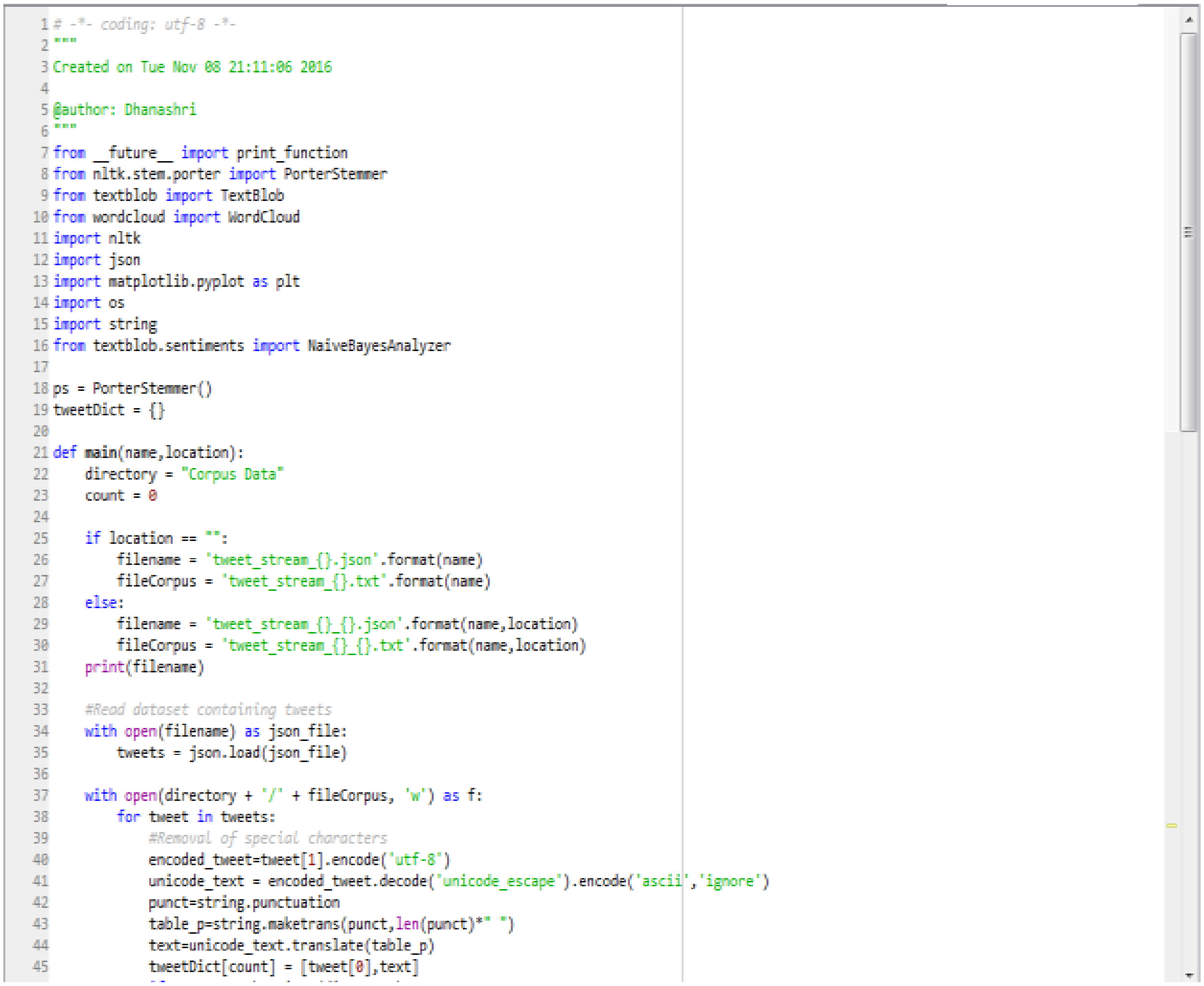


```
print("{} has highest positive comments.".format(highestPol))
print("{} has highest negative comments.".format(highestSub))

#JSON Dataset that has tweets
corpus=['tweet_stream_hillary.json','tweet_stream_obama.json','tweet_stream_trump.json']

#Topic Analysis, LDA
fname=[]
corpus=[]
docs=[]
corpus_root='Corpus Data'
for filename in os.listdir(corpus_root):
    file = open(os.path.join(corpus_root, filename), "r")
    doc = file.read()
    words=doc.split()
    file.close()
    fname.append(filename)
    corpus.append(doc)
    docs.append(words)

vectorizer = TfidfVectorizer(stop_words='english', min_df=2)
dtm = vectorizer.fit_transform(corpus)
vocab = vectorizer.get_feature_names()

num_topics=3
num_top_words=10
clf = decomposition.NMF(n_components=num_topics, random_state=1)
doctopic = clf.fit_transform(dtm)
print num_topics, clf.reconstruction_err_

topic_words = []
for topic in clf.components_:
    word_idx = np.argsort(topic)[::-1][0:num_top_words]
    topic_words.append([vocab[i] for i in word_idx])

for t in range(len(topic_words)):
    print "Topic {}: {}".format(t, ' '.join(topic_words[t][:15]))

dic = corpora.Dictionary(docs)
corp = [dic.doc2bow(text) for text in docs]
tfidf = models.TfidfModel(corp)
corpus_tfidf = tfidf[corp]
model = models.ldamodel.LdaModel(corpus_tfidf, num_topics=num_topics, id2word=dic, update_every=1, passes=100)
```

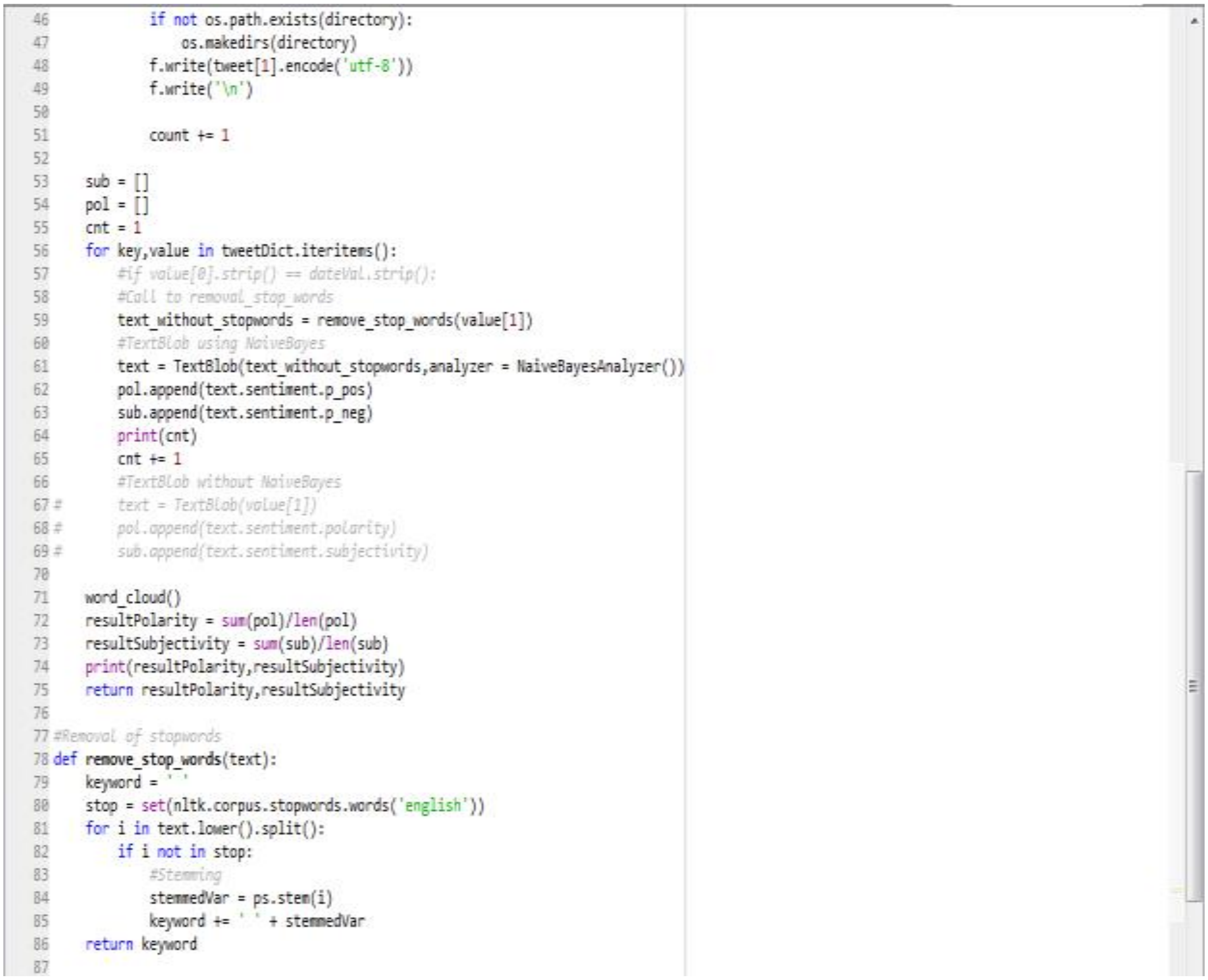


```
print("LDA model")
topics_found = model.print_topics(20)
counter = 1
for t in topics_found:
    print("Topic #{} {}".format(counter, t))
    counter += 1
topics_found2 = model.print_topics(50)
counter2 = 1
for t in topics_found2:
    print("Topic #{} {}".format(counter2, t))
    counter2 += 1
```

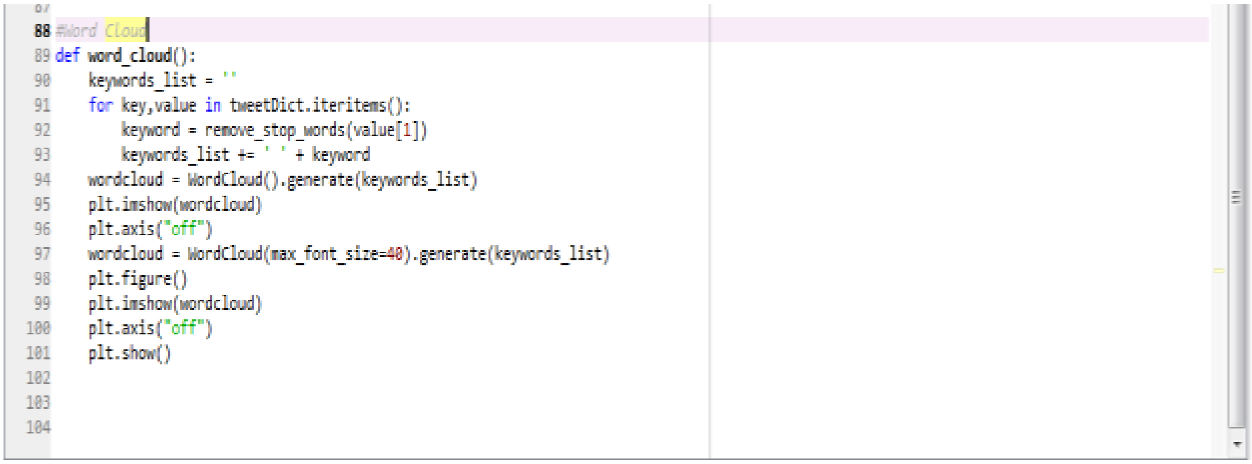
sentimentAnalysis.py



```python
# -*- coding: utf-8 -*-
"""
Created on Tue Nov 08 21:11:06 2016

@author: Dhanashri
"""
from __future__ import print_function
from nltk.stem.porter import PorterStemmer
from textblob import TextBlob
from wordcloud import WordCloud
import nltk
import json
import matplotlib.pyplot as plt
import os
import string
from textblob.sentiments import NaiveBayesAnalyzer

ps = PorterStemmer()
tweetDict = {}

def main(name,location):
    directory = "Corpus Data"
    count = 0

    if location == "":
        filename = 'tweet_stream_{}.json'.format(name)
        fileCorpus = 'tweet_stream_{}.txt'.format(name)
    else:
        filename = 'tweet_stream_{}_{}.json'.format(name,location)
        fileCorpus = 'tweet_stream_{}_{}.txt'.format(name,location)
    print(filename)

    #Read dataset containing tweets
    with open(filename) as json_file:
        tweets = json.load(json_file)

    with open(directory + '/' + fileCorpus, 'w') as f:
        for tweet in tweets:
            #Removal of special characters
            encoded_tweet=tweet[1].encode('utf-8')
            unicode_text = encoded_tweet.decode('unicode_escape').encode('ascii','ignore')
            punct=string.punctuation
            table_p=string.maketrans(punct,len(punct)*" ")
            text=unicode_text.translate(table_p)
            tweetDict[count] = [tweet[0],text]
```

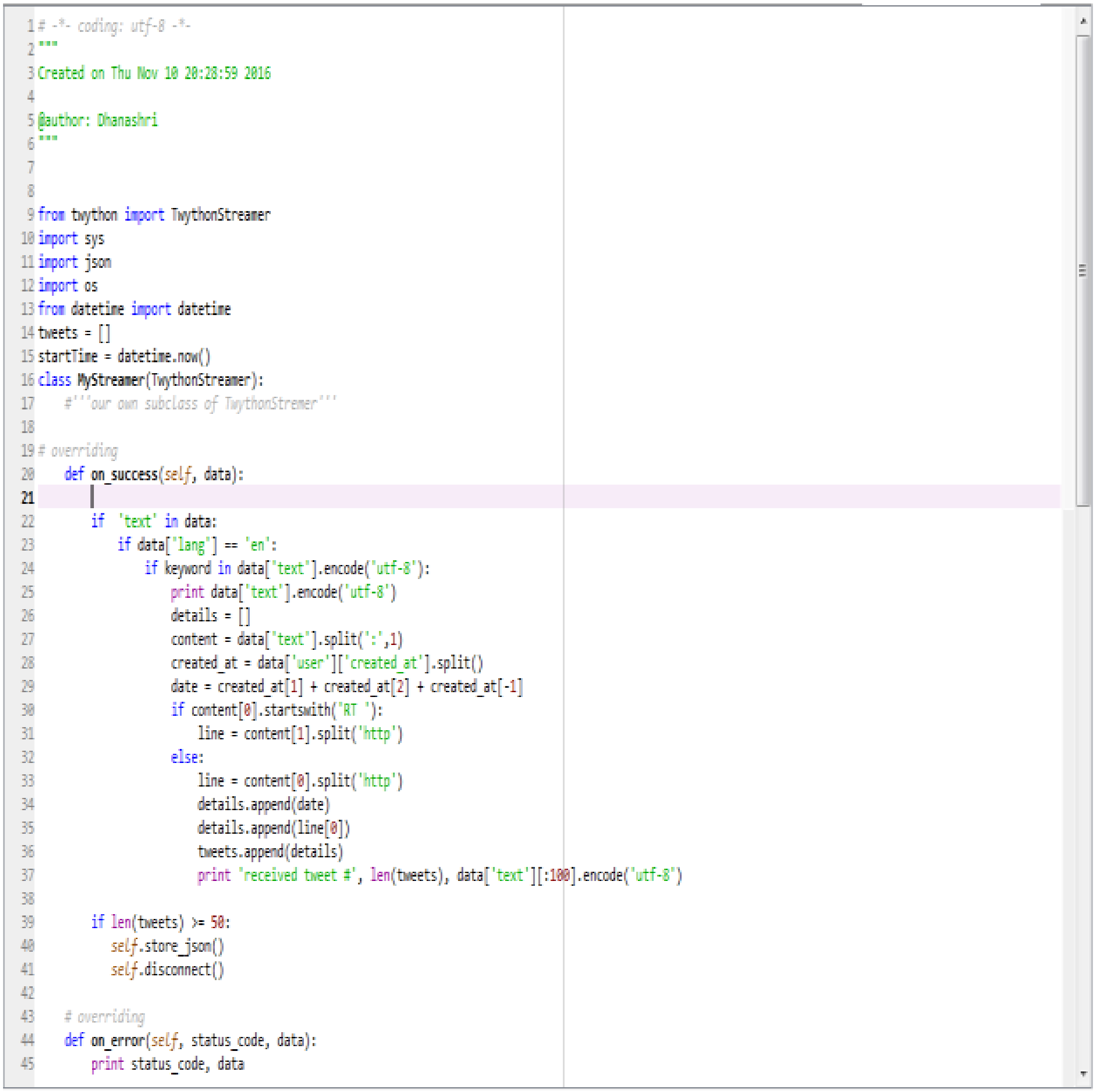


```
            if not os.path.exists(directory):
                os.makedirs(directory)
            f.write(tweet[1].encode('utf-8'))
            f.write('\n')

            count += 1

    sub = []
    pol = []
    cnt = 1
    for key,value in tweetDict.iteritems():
        #if value[0].strip() == dateVal.strip():
        #Call to removal_stop_words
        text_without_stopwords = remove_stop_words(value[1])
        #TextBlob using NaiveBayes
        text = TextBlob(text_without_stopwords,analyzer = NaiveBayesAnalyzer())
        pol.append(text.sentiment.p_pos)
        sub.append(text.sentiment.p_neg)
        print(cnt)
        cnt += 1
        #TextBlob without NaiveBayes
#       text = TextBlob(value[1])
#       pol.append(text.sentiment.polarity)
#       sub.append(text.sentiment.subjectivity)

    word_cloud()
    resultPolarity = sum(pol)/len(pol)
    resultSubjectivity = sum(sub)/len(sub)
    print(resultPolarity,resultSubjectivity)
    return resultPolarity,resultSubjectivity

#Removal of stopwords
def remove_stop_words(text):
    keyword = ' '
    stop = set(nltk.corpus.stopwords.words('english'))
    for i in text.lower().split():
        if i not in stop:
            #Stemming
            stemmedVar = ps.stem(i)
            keyword += ' ' + stemmedVar
    return keyword

```

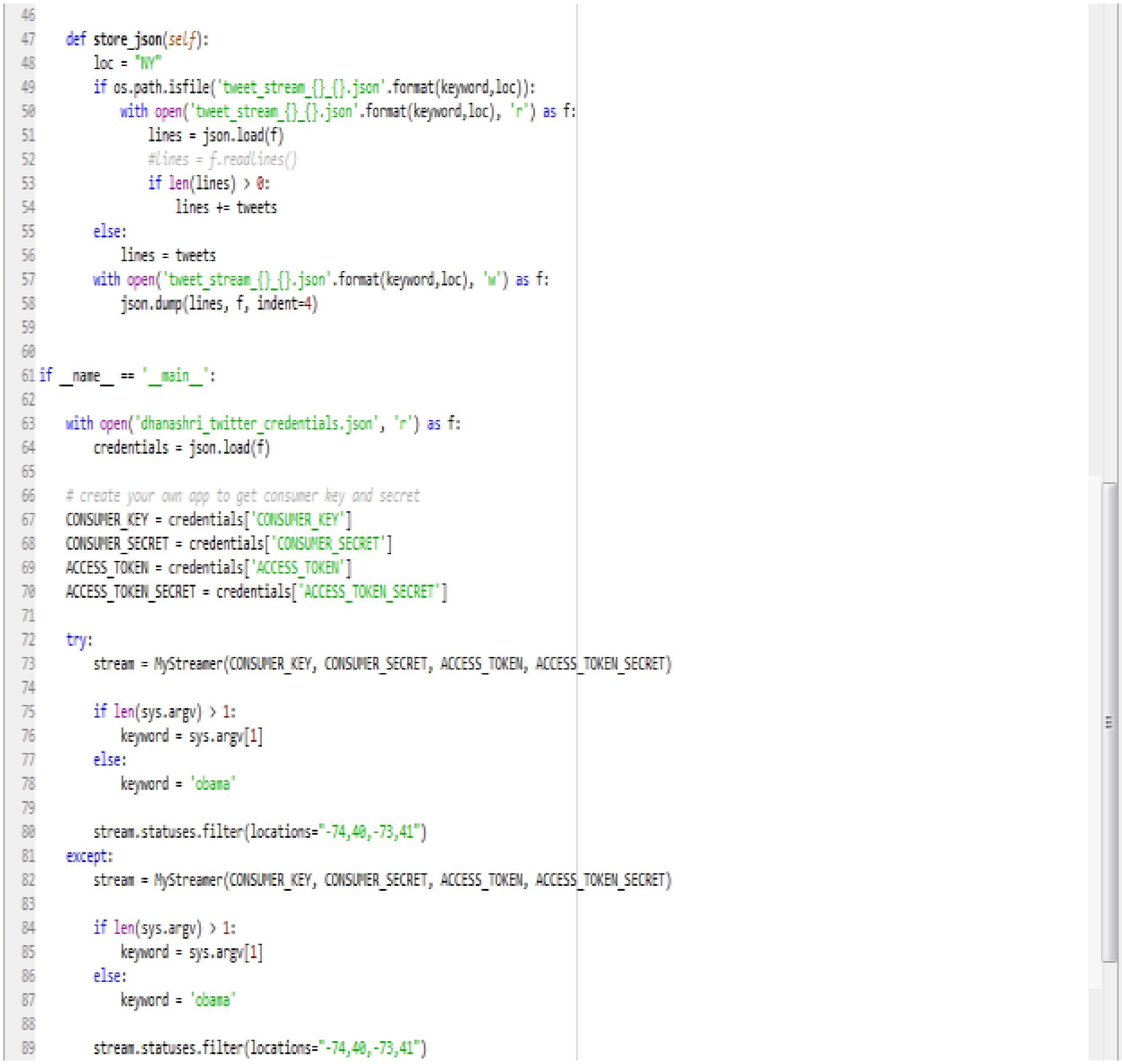


```

#Word Cloud
def word_cloud():
    keywords_list = ''
    for key,value in tweetDict.iteritems():
        keyword = remove_stop_words(value[1])
        keywords_list += ' ' + keyword
    wordcloud = WordCloud().generate(keywords_list)
    plt.imshow(wordcloud)
    plt.axis("off")
    wordcloud = WordCloud(max_font_size=40).generate(keywords_list)
    plt.figure()
    plt.imshow(wordcloud)
    plt.axis("off")
    plt.show()



```

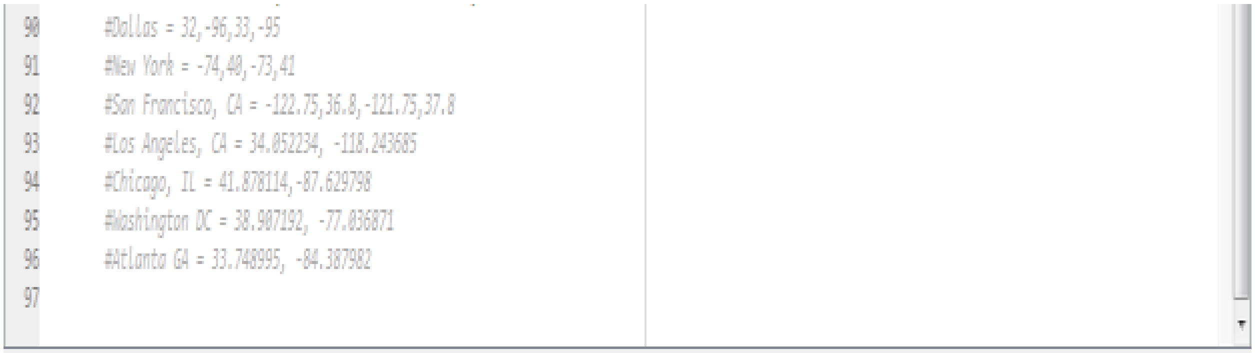
twitterLocation.py



```
# -*- coding: utf-8 -*-
"""
Created on Thu Nov 10 20:28:59 2016

@author: Dhanashri
"""


from twython import TwythonStreamer
import sys
import json
import os
from datetime import datetime
tweets = []
startTime = datetime.now()
class MyStreamer(TwythonStreamer):
    #'''our own subclass of TwythonStremer'''

# overriding
    def on_success(self, data):
        
        if  'text' in data:
            if data['lang'] == 'en':
                if keyword in data['text'].encode('utf-8'):
                    print data['text'].encode('utf-8')
                    details = []
                    content = data['text'].split(':',1)
                    created_at = data['user']['created_at'].split()
                    date = created_at[1] + created_at[2] + created_at[-1]
                    if content[0].startswith('RT '):
                        line = content[1].split('http')
                    else:
                        line = content[0].split('http')
                        details.append(date)
                        details.append(line[0])
                        tweets.append(details)
                        print 'received tweet #', len(tweets), data['text'][:100].encode('utf-8')

        if len(tweets) >= 50:
           self.store_json()
           self.disconnect()

    # overriding
    def on_error(self, status_code, data):
        print status_code, data
```



```
    def store_json(self):
        loc = "NY"
        if os.path.isfile('tweet_stream_{}_{}.json'.format(keyword,loc)):
            with open('tweet_stream_{}_{}.json'.format(keyword,loc), 'r') as f:
                lines = json.load(f)
                #lines = f.readlines()
                if len(lines) > 0:
                    lines += tweets
        else:
            lines = tweets
        with open('tweet_stream_{}_{}.json'.format(keyword,loc), 'w') as f:
            json.dump(lines, f, indent=4)


if __name__ == '__main__':

    with open('dhanashri_twitter_credentials.json', 'r') as f:
        credentials = json.load(f)

    # create your own app to get consumer key and secret
    CONSUMER_KEY = credentials['CONSUMER_KEY']
    CONSUMER_SECRET = credentials['CONSUMER_SECRET']
    ACCESS_TOKEN = credentials['ACCESS_TOKEN']
    ACCESS_TOKEN_SECRET = credentials['ACCESS_TOKEN_SECRET']

    try:
        stream = MyStreamer(CONSUMER_KEY, CONSUMER_SECRET, ACCESS_TOKEN, ACCESS_TOKEN_SECRET)

        if len(sys.argv) > 1:
            keyword = sys.argv[1]
        else:
            keyword = 'obama'

        stream.statuses.filter(locations="-74,40,-73,41")
    except:
        stream = MyStreamer(CONSUMER_KEY, CONSUMER_SECRET, ACCESS_TOKEN, ACCESS_TOKEN_SECRET)

        if len(sys.argv) > 1:
            keyword = sys.argv[1]
        else:
            keyword = 'obama'

        stream.statuses.filter(locations="-74,40,-73,41")
```



```
        #Dallas = 32,-96,33,-95
        #New York = -74,40,-73,41
        #San Francisco, CA = -122.75,36.8,-121.75,37.8
        #Los Angeles, CA = 34.052234, -118.243685
        #Chicago, IL = 41.878114,-87.629798
        #Washington DC = 38.907192, -77.036871
        #Atlanta GA = 33.748995, -84.387982

```

CONCLUSION

We presented a system for real-time Twitter sentiment analysis of the 2016 U.S. presidential election. We used the Twitter API, Python and Word Clouds and evaluated the public sentiments expressed over Twitter and helped us perform topic modelling in response to the political events.