In [ ]:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# Columbia EECS E6893 Big Data Analytics
"""
This module is used to pull data from twitter API and send data to
Spark Streaming process using socket. It acts like a client of
twitter API and a server of spark streaming. It opens a listening TCP
server socket, and listens to any connection from TCP client. After
a connection established, it send streaming data to it.


Usage:
  If used with dataproc:
    gcloud dataproc jobs submit pyspark --cluster <Cluster Name> twitterHTTPClient.py

  Make sure that you run this module before you run spark streaming process.
  Please remember stop the job on dataproc if you no longer want to stream data.

"""

import tweepy
from tweepy import OAuthHandler
from tweepy import Stream
import socket
import re

# credentials
ACCESS_TOKEN = "1578524023033667586-nYVmtbLtnhKqa40jEoMirSRmeCJ6vS"
ACCESS_SECRET = "oG98avuyMnjsX8SuVF6cfzY40Pg6AvcwSuDwnIOwaaCLJ"
CONSUMER_KEY = "xIDvSjDs3ZjREUM1WvXlScY0W"
CONSUMER_SECRET = "OV3p8VhuWPtjQnxAo32D0edi35YteQOuNJvljbH6UmHy3WWnG0"
BEARER_TOKEN = "AAAAAAAAAAAAAAAAAAAAAKfdiAEAAAAAZWSgd%2BmDUoyZU07jPDbdgNfTQ6w%3DvbWWUbmbLIsr5yVcWXjo0LKSf7CHh52oEJlgUTFmx0l7ozG4w7"

# the tags to track
tags = ['messi', 'bigdata', 'football', 'ai', 'fcbarcelona']

class MyStream(tweepy.StreamingClient):

    global client_socket
    def on_tweet(self, tweet):
        try:
            msg = tweet
#             print('TEXT:{}\n'.format(msg.text))
            # Remove some non-English tweets, reference only
            temp = re.sub('[^\u0000-\u05C0\u2100-\u214F]+', '', msg.text)
            temp = re.sub(r'http\S+', '', temp)
            client_socket.send(msg.text.encode('utf-8'))
            return True
        except BaseException as e:
            print("Error on_data: %s" % str(e))
            self.disconnect()
            return False

    def on_error(self, status):
        print(status)
        return False

def sendData(c_socket, tags):
    """
    send data to socket
    """
    global client_socket
    client_socket = c_socket
    stream = MyStream(BEARER_TOKEN)

    for tag in tags:
        stream.add_rules(tweepy.StreamRule(tag))

    stream.filter()


class twitter_client:
    def __init__(self, TCP_IP, TCP_PORT):
      self.s = s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
      self.s.bind((TCP_IP, TCP_PORT))

    def run_client(self, tags):
      try:
        self.s.listen(1)
        while True:
          print("Waiting for TCP connection...")
          conn, addr = self.s.accept()
          print("Connected... Starting getting tweets.")
          sendData(conn,tags)
          conn.close()
      except KeyboardInterrupt:
        exit


if __name__ == '__main__':
    client = twitter_client("localhost", 9001)
    client.run_client(tags)
```

```
Waiting for TCP connection...
Connected... Starting getting tweets.

Stream connection closed by Twitter

Error on_data: [Errno 32] Broken pipe
Waiting for TCP connection...
```

In [ ]:

In [ ]: