University of Virginia

DS 5110: Big Data Systems

Assignment: Streaming Analytics with the Twitter API

Last Updated: April 19, 2021

TOTAL POINTS: 20

**INSTRUCTIONS**

You will run two programs in separate terminals to perform the tasks in the **ASSIGNMENT** section. The value *None* is used as a placeholder in the code. The trickiest part may be running the correct code in the correct terminals in the correct order. To simplify things, we refer to the terminals as SERVER and CLIENT.

**SERVER AND CLIENT SETUP**

On the SERVER terminal you will copy/paste/run code from the file *twitter.py*

On the CLIENT terminal you will copy/paste/run code from the file *spark\_streaming\_analytics.py*

The code on the SERVER terminal will call Twitter API and push streaming tweets to Spark Streaming.

The code on CLIENT terminal will consume Twitter tweets from SERVER and return streaming analytics.

FOLLOW THIS SEQUENCE OF STEPS CLOSELY:

**1)** Open two terminals, dedicating one as SERVER and the other as CLIENT (I keep SERVER on top, as that one is started first).

**2)** Open **SERVER** code *twitter.py* and start by working through the SERVER SECTION 1 code block.**a.** In the SET UP TWITTER CREDENTIALS section, enter your credentials for your Twitter API Developer account.  
**b.** Copy/paste/run the LIBRARIES code and run, confirming all libraries are installed.   
If a package is missing, you can open another terminal window and install like this for Python 3.6:  
  
# to install requests, requests\_oauthlib:

$ sudo pip-3.6 install requests requests\_oauthlib

**c.** Copy/paste/run the remainder of code block SERVER SECTION 1 into SERVER.

**3)** Open **CLIENT** code *spark\_streaming\_analytics.py* and copy/paste/run the entire   
CLIENT SECTION 1 code block.

You will see streaming results with a row for timestamp, and no data as expected (we aren’t yet consuming Twitter data).  
  
  
  
  
  
  
  
  
Here is a sample for two time windows with interval size 5 seconds:  
-------------------------------------------

Time: 2019-11-09 20:39:20

-------------------------------------------

-------------------------------------------

Time: 2019-11-09 20:39:25

-------------------------------------------

In your SERVER terminal, you should see that this code has run:

conn, addr = s.accept()  
and you should get your cursor back at the prompt. The socket connection has been made, and you can start delivering data from SERVER to CLIENT.

4) Return to SERVER, where you will copy/paste/run SERVER SECTION 2.

**5)** You should see results printing in the CLIENT terminal. These are the top ten hashtags and counts being printed at 5 second intervals. Your results will vary, as hashtags and volumes are time dependent.

SAMPLE OUTPUT

-------------------------------------------

Time: 2021-04-16 14:46:55

-------------------------------------------

('#beerme', 1)

('#bjj', 1)

('#bjjlifestyle', 1)

('#dohasport', 1)

('#ParishBridge', 1)

('#bjj4life', 1)

('#Pune', 1)

('#Remdesevir', 1)

('#Indore', 1)

('#35mm', 1)

-------------------------------------------

Time: 2021-04-16 14:47:00

-------------------------------------------

('#RT66Run', 1)

('#playingwithfilters', 1)

('#tiktokfilters', 1)

('#euphoricmakeup', 1)

('#filters', 1)

('#FridayLivestream', 1)

('#WomenTakeTheWheel', 1)

('#TopFuel', 1)

('#NHRA', 1)

('#comm\_college', 1)

**TROUBLESHOOTING**

If something breaks and you get a stream of error messages, it is best to close the two terminals and start over.  
To properly end the processes, enter s.close() in the SERVER terminal, and ssc.stop() in the CLIENT terminal.

**ASSIGNMENT**

1. (5 POINTS) Review all of the SERVER and CLIENT code and make sure you understand how it works.   
Depending on your background, this may take some time as there are new concepts to absorb, such as making a socket connection (socket = ip\_address:port). I am assuming you will put in the work, earning 5 points automatically.

2. (9 POINTS TOTAL; 1 for each completed line) In the CLIENT file *spark\_streaming\_analytics.py:*

Replace all placeholder *None* values with the correct code to get it working. Submit the file through Collab.

3. (5 POINTS) Capture consecutive streaming windows of output (2-3 are sufficient), pasting into a document and labeling it OUTPUT\_T5 for time interval 5 seconds. For example, you can show results as I have done above. If possible, let the streaming run for a little while to see if you can capture counts incrementing as I have done.

4. (1 POINT) Change the Spark streaming context interval size from 5 seconds to 10 seconds. Copy/paste the output as in Question 3, labeling it OUTPUT\_T10.