Big Data Analytics
Stony Brook University
CSE545 - Fall 2016

Assignment 2

Assigned: 10/24/2016;

Part I Due: ~~11/10/2017~~11/11/2017, 3:45pm EST

Part II Due: 11/22/2017, 3:45pm EST

Part I. Contextual Word Count over AWS EMR Spark Cluster (50 points)

- **Get an AWS account** *(15 points)*
  - *Step 1*. Apply for AWS educate student credit here:
    https://www.awseducate.com/Application?apptype=student
    https://www.awseducate.com/faqs?app=2#fa0Po000000C0eH1EAJ
    You will have two options for choosing an account:
    - Use an AWS account with credit card on file.
      (don't have one already? click the link "don't have one? Sign up now.")
    - Create an AWS Educate starter account -- doesn't require a credit card.
      (this will give you less credits than the first option; Presumably, Amazon feels that they can give more to those with a credit card as such users are more likely to come back and pay for service in the future.)
  - *Step 2 for the account with credit card:* Redeem your promotion credit code via email.
    Search for "AWS Educate" in your email
    Enter the code here: https://console.aws.amazon.com/billing/home#/credits
  - *Step 2 for starter accounts:* drop a request to QwikLabs support asking for your quota to be extended to at least 40 instances https://qwiklab.zendesk.com/home
  - *Step 3*
    - *If credit card:* Store a screen-shot of the above webpage after entering your code.
      (double-check that you have included anything sensitive in your screen shot)
    - *If using starter account* -- show screenshot of the billing page showing that you have a starter account.

  Useful Tip: If using a regular AWS account (with credit card), enable billing alerts:
  https://console.aws.amazon.com/billing/home#/preferences

- **Warm up: Contextual Word Count in Spark (using AWS EMR)** *(35 points)*
  - **Objective: Create Spark program to count what people "feel"**
    - Input: First ~~5,000~~ **1,000** "WET" files from the Common Crawl September 2016
    - Output: Count of words that appear after the string "I feel"
      - your code should be case insensitive (convert strings to lowercase)

        - "I feel" can appear anywhere and you should only count the one single word appearing after it (this will mean many common words that aren't "feelings" will also be included; i.e. "that", "the", etc…).

- **Create spark code to achieve output from above**
    - The wet-paths.gz file linked to above lists the path to files containing just the text of web-page (i.e. ready for directly reading in for distributed IO).
    - (optional) Test on your own system with a local Spark on approximately 5 downloaded wet files (no need to download more; you will use the remaining files directly through aws).
- **Launch an EMR Spark Instance:**
    - *Step 1.* Make sure you're feeling good about your Spark code because once you launch the EMR instance (as per next steps) you will start losing credits.
    - *Step 2.* Login to: https://console.aws.amazon.com
    - *Step 3.* Navigate to EMR Service Page
      Click on "Services" at the top, then EMR
    - *Step 4.* Configure your cluster
      Use Default settings, except:
        - Under "Applications", choose "Spark: ..."
        - Under "Number of Instances", choose at least 7
          (consult pricing if choosing more)
        - Under EC2 Key pair add a pair a that you plan to use
          (follow "learn how to create EC2 key pair")
    - *Step 5.* Create a screen shot of the EMR console page showing that you have a cluster running.
- **Migrate your code to your EMR Spark Cluster**
    - Connect to your master node via ssh.
    - Test your code in your respective language on limited sample
      (consider the spark or pyspark shell)
    - Access the data through S3
      `(e.g. s3://commoncrawl/crawl-data/CC-MAIN-2016-40/wet.paths.gz)`
    - Run your code on the full sample: first ~~5,000~~ **1,000** wet files
        - Running on the first 5,000 is worth 5 points extra credit.
          (make sure to note this clearly in your output file submission)
    - Filter your results to just those words occurring at least 30 times
      (this can be done inside or outside Spark)
        - Save these as a list of tuples: (word, count)
    - Once completed, make sure you have saved the code to your local machine and terminate the cluster instance from the EMR console page so you stop losing credits.

Deliverables (4 total files):

1. **Image (jpeg or png):** credits screenshot (showing credits redeemed)
   (named `a2_p1_lastname-credits.png`)
2. **Image (jpeg or png):** EMR Console Screen shot (showing running cluster)
   (named `a2_p1_lastname-emr.png`)
3. **Code (python, java, or scala):** contextual word count code in 1 file.
   Include your name at the top of the code as well.
   (named `a2_p1_lastname-code.py|java|scala`)
4. **Output file** (text file): all words that appear after "I feel" >30 times.
   (named `a2_p2_lastname-output.txt`)

   +5 points extra credit for submitted by Thursday 11/10, 3:45pm

Tips:

- Do the first set of tasks ASAP to make sure you're not waiting for your AWS credits
- As inspiration, see: this blog post.
  (note that this used a MapReduce style program instead of Spark).
- Using spot instances may save you quite a bit of credits but isn't a great option if you're working last minute.
- If having trouble sshing in, add your ip address to the inbound security group for master
  (go to security group from the emr page, click on the master group-> edit inbound rules-> add rule-> select "all TCP", "my IP")
- You can read multiple text files into a single RDD with sc.textFile by simply separating each file with a comma (`sc.textFile("file1,file2,file3")`)
- Try running on only ~10 files when first migrating to AWS / EMR to get the AWS-specifics down (i.e. reading from S3). .
- Avoiding use of regular expressions usually speeds things up.

(tips deemed most useful will be added here. For more, check Piazza)

Part II. <TBD, Depending on AWS Educate Grant Coming Through> (50 points; Due 11/23)

Guidelines.

Include your name at the top of the code as well.

Submission. For each part, one single compressed file should be submitted via blackboard containing all deliverables.

Libraries permitted: default file IO, Spark libraries, statistical libraries, but nothing that implements the algorithms you are being asked to implement.

Testing: Our testing will consist of running on data in the same format as the examples provided. You may also be asked by the TA to demonstrate your code running on AWS. Your code should work with Spark >= 2.0.0.

Questions / Clarifications: Please post questions on Piazza, so other classmates may see the answers. Questions posted within 48 hours of the deadline are not guaranteed a response before the deadline.

Academic Integrity: As with all assignments (sans the team project), although you may discuss concepts with others, you must work independently and insure your work and code is not visible to any classmates. You may also not copy any partials solutions from the Web or other resources though you may reference algorithm descriptions and method parameter definitions.