Lab 2 - Web Server Log Analysis with Apache Spark

This lab will demonstrate how easy it is to perform web server log analysis with Apache Spark.

Server log analysis is an ideal use case for Spark.  It's a very large, common data source and contains a rich set of information.  Spark allows you to store your logs in files on disk cheaply, while still providing a quick and simple way to perform data analysis on them.  This homework will show you how to use Apache Spark on real-world text-based production logs and fully harness the power of that data.  Log data comes from many sources, such as web, file, and compute servers, application logs, user-generated content,  and can be used for monitoring servers, improving business and customer intelligence, building recommendation systems, fraud detection, and much more.

*The lab is due Jun 19, 2015 at 00:00 UTC. There is a three day grace period for late submissions until Jun 22, 2015 at 00:00 UTC. Submissions after that time will lose 20 points.*

HOW TO COMPLETE THIS ASSIGNMENT

This assignment is broken up into sections with bite-sized examples for demonstrating Spark functionality for log processing. For each problem, you should start by thinking about the algorithm that you will use to*efficiently* process the log in a parallel, distributed manner. This means using the various [RDD](http://spark.apache.org/docs/latest/api/python/pyspark.html#pyspark.RDD) operations along with [lambda functions](https://docs.python.org/2/tutorial/controlflow.html#lambda-expressions) that are applied at each worker.

INSTRUCTIONS

Here are the instructions.

1. Start the VM - To start the VM, from a DOS prompt (Windows) or Terminal (Mac/Linux), issue the command "vagrant up", while in the custom directory [created](https://courses.edx.org/courses/BerkeleyX/CS100.1x/1T2015/courseware/d1f293d0cb53466dbb5c0cd81f55b45b/920d3370060540c8b21d56f05c64bdda/) for this course.
2. Once the Virtual Machine is running, access the Jupyter web UI for running IPython notebooks by navigating your web browser to "<http://localhost:8001/>" (or "<http://127.0.0.1:8001/>").
3. If you have any running notebooks they **SHOULD BE** shutdown.  Only **ONE** notebook should be run at a time.  Running notebooks have a green icon to the left of the notebook name and green text to the right of the screen that says "Running".  Shutdown running notebooks by clicking the checkbox next to the notebook and then clicking the orange "Shutdown" button.
4. Download the Lab 2 IPython notebook.  Make sure that the file extension is .ipynb.  If the download adds an extension (e.g. ".txt"), rename the file so that the extension is just .ipynb.
   * Lab 2 Web Server Log Analysis with Apache Spark: <https://raw.githubusercontent.com/spark-mooc/mooc-setup/master/lab2_apache_log_student.ipynb>. You can view this lab exercise online [here](http://nbviewer.ipython.org/github/spark-mooc/mooc-setup/blob/master/lab2_apache_log_student.ipynb).
5. Upload the IPython notebook.  This process was explained during "[Setting up the Course Software Environment](https://courses.edx.org/courses/BerkeleyX/CS100.1x/1T2015/courseware/d1f293d0cb53466dbb5c0cd81f55b45b/920d3370060540c8b21d56f05c64bdda/)" in week one.
6. In the **Lab 2 Web Server Log Analysis with Apache Spark** notebook, please follow the instructions in the notebook and replace <FILL IN> sections with your solution.  After you confirm that your code passes all of the tests while running in the VM, please export it as a **python file (.py)**and submit it to the autograder server.  The submission process is the same as in the previous week for Lab 1.  In the next module, the instructions are provided again for your convenience.
7. An outline of what will be covered in the notebook is included below.
8. When you have submitted successfully, you can shutdown the VM by issuing the command "vagrant halt".

LOG ANALYSIS EXERCISE

This exercise consists of 4 parts:

* *Part 1*: Apache Web Server Log file format
* *Part 2*: Sample Analyses on the Web Server Log File
* *Part 3*: Analyzing Web Server Log File
* *Part 4*: Exploring 404 Response Codes

**Please do not post your programming exercises in publicly visible repositories, such as GitHub.**

## THE COURSE AUTOGRADER

Assignments in this course are automatically graded using an autograder, which runs in the cloud on Amazon EC2. The autograder uses automatic scaling to handle large increases in the number of submissions.

## SUBMISSION FORMAT

Because we are using an autograder, it is very important that your submissions comply with the following rules:

* Only use the following libraries: standard python libraries, numpy, pyspark, and test\_helper (the autograder library).
* Don't leave in extraneous code, as the autograder will timeout if a submission takes too long.
* Only change sections of code where you see <FILL IN>, as changing other parts of the code, including directory paths, may cause the code to fail the autograder's tests.

## AUTOGRADER RESPONSE TIME

When you submit an assignment to be graded, you can expect to see a graded response within a few minutes. Note that the scale-up process itself can take a few minutes when many users simultaneously submit their assignments, so you may see additional delays in grading. If you do not receive feedback from the autograder server within 1 hour, please resubmit your code.  If you not receive feedback on your resubmitted code within 45 minutes, please use the [Piazza discussion group](https://piazza.com/edx_berkeley/summer2015/cs1001x) to contact the TAs for support.

Note: The autograder server cluster checks the submission volume (numbers of submissions) every five minutes.  If necessary, the autograder launches additional instances to handle the volume.

## AUTOGRADER FEEDBACK

There are 3 types of feedback.

* All tests passed or some tests failed.  The result for each test will be in the feedback.
* Timeout.  The submitted code took too long to execute.  In this case, please optimize your code's running time.
* Abnormal results are detected.  In this case, please review your code carefully according to the guidelines found on this page. If you need help with your code, please use the [Piazza discussion group](https://piazza.com/edx_berkeley/summer2015/cs1001x) to contact the TAs for support.

## SUBMITTING LAB 2 TO THE AUTOGRADER

(100 points possible)

Once you have successfully run the test notebook, you can submit the notebook to the course autograder. First, export the test notebook as an Python (.py) file. Next, use this file chooser to select your file and click**"Check"** to submit your code to the course autograder. The video in the "Running your first notebook" unit shows you this process.

The course autograder automatically scales as the number of pending submissions grows, but the time to receive your feedback will depend on how many other users are submitting at the same time.

Reminder: In the module "Running Your First Notebook" in week 1, we showed you the process for exporting a notebook to a Python file and submitting the file to the autograder.

**Before you submit your lab 2, make sure you follow these steps:**

* Delete any blank cells you added.
* Do not change any of the notebook's code other than the <FILL IN> items.
* Do not import any libraries. The autograder will not import extra libraries.
* Remove any extra print statements that you added.
* **Do not use print statements inside any functions that you use in map() functions.** Using print statements will significantly delay the time for the autograder to complete grading your submission.
* Do not use collect() actions to collect all data from an RDD.
* Replace all the <FILL IN> items with valid Python/pyspark code.
* Starting with the first cell in your iPython notebook, use Shift-ENTER to execute every single cell. Do not submit your notebook if there are any errors or you fail any of the built-in tests.
* Export your iPython notebook as a **Python** file.

Select your lab 2 Python (.py) file: