Class 9: Homework

Realtime and Big Data Analytics

Summer 2017

Homework

Class 9

Analytics Project

1. Paper Drop #2

   Create a second draft of your paper, expanding existing sections with more detail/discussion. You can start writing the remaining sections (Future Work, Results, Conclusion). Final paper length should be 5-6 pages for a 3 person team, for example.

   Each team member should upload the paper to NYU Classes. (All team members upload the same thing.)

1. Code Drop #2

   Continue developing your code using only **Big Data tools** for the analytics part of your project.

   Please upload your in-progress code. You should be running / assessing / improving your analytic at this time.

   Each team member should upload a zip of the team's analytics code to NYU Classes. (All team members upload the same thing.)

   Do not include ETL code you previously submitted.

Homework

Class 9

See NYU Classes for due dates - they are different than usual.

Spark Homework

1. Complete the introductory Spark exercise that appears on the following pages.

Readings

1. TDG book: How MapReduce Works- pages 185-200

Optional Reading

1. TDG book: Spark - pp 549-558
2. TDG book: Oozie - pp 179-184
3. TDG book: Sqoop - pp 401-415

Homework
Class 9

1. Move the file weblogs.zip in the NYU Classes Resources tab into your VM (or into Dumbo) and unzip it.

2. Change to the directory that contains the weblogs directory created when you unzipped weblogs.zip. Use the cat command to view one of the files in the weblogs directory, then put the file into HDFS:

```
$ cat weblogs/2013-09-15.log
$ hdfs dfs -put weblogs
$ hdfs dfs -ls weblogs
```

3. In a terminal window, start the Scala Spark Shell:

```
$ spark-shell
```

4. Spark creates a SparkContext object for you called sc. Make sure the object exists:

```
scala> sc
```

5. Using command completion, you can see all the available SparkContext methods:

   type sc. (sc followed by a dot) and then the [TAB] key.

6. Read the test data into a Spark RDD. (Note: Adjust the directory name as needed for Dumbo/VM.)

```
scala> val alldata = sc.textFile("/user/cloudera/weblogs")
```

Homework

Class 9

7. Using command completion, you can see all the available transformations and operations you can perform on an RDD.

```
Type alldata. and then the [TAB] key
```

8. Spark has not yet read the file at this point - it will not do so until you perform an action on the RDD (this is the lazy execution model, just as we saw with Pig). Try counting the number of lines in the dataset. The count operation causes the RDD to be materialized (created and populated). The number of lines should be displayed. Enter the number of lines counted in the **NYU Classes** assignment and enter the amount of time required to count the lines.

```
scala> alldata.count()
```

9. Try executing the take command to display the data in the RDD. Note that this returns and displays just the specified number of lines.

```
scala> alldata.take(4)
```

10. When you issued the take command, the lines did not print out neatly. Use the following command to print each line in the file on its own line:

```
scala> alldata.take(4).foreach(println)
```

Homework

Class 9

11. Read the log data from one log file (/user/cloudera/weblogs/2014-03-15.log) into a Spark RDD called oneLog and enter into **NYU Classes** the number of lines in the file.

12. Execute the take operation to display one line of data in the RDD. Enter the command you used in **NYU Classes**.

13. Create an RDD, jpgLines, containing only those lines that contain '.jpg'. This time, use all of the log data files as input:

```
scala> var jpglines = alldata.filter(line => line.contains(".jpg"))
```

14. Count the number of lines in the RDD jpgLines. Enter the number of lines counted in **NYU Classes**.

15. You can exit the shell at any time by typing exit.

16. Start the beeline shell and count the number of lines in all of the files in the weblogs directory. Notice how long it takes to run this command in Hive compared to Spark.

```
$ beeline -u jdbc:hive2://quickstart:10000/default -n cloudera -d org.apache.hive.jdbc.HiveDriver

hive> create external table webLogsData (wholeLine string)
      row format delimited fields terminated by ','
      location '/user/cloudera/weblogs/';

hive> select count(*) from webLogsData;
```