Big Data Science - Spring 2018
Assignment 3

Daniel Rivera Ruiz
Department of Computer Science
New York University
drr342@nyu.edu

1 Word count in Eclipse under Hadoop

1.1 Word count on the *document.txt* file

(a) Figure 1: Snapshot of the *WordCount* project in Eclipse

(b) Figure 2: Snapshot of the *WordCount.java* code in Eclipse

(c) Figure 3: Snapshot of the console output in Eclipse

(d) Figure 4: Snapshot of the output file *WordCountOUTPUT2.txt*

(e) Figure 5: Snapshot of the Hadoop Cluster summary on the web browser [1]

1.2 Word count on the other input files

Using the exact same code from the previous section we can perform word count on different files only by changing the arguments when running the java command. In this case, the first argument corresponds to the *path/to/inputfile* and the second argument corresponds to the *path/to/outputfile*. Therefore, we need only provide the appropriate paths to perform word count on different files.
The output files for all the runs of *WordCount* (*WordCountOUTPUT2.txt*, *WordCountARTICLE1.txt*, *WordCountARTICLE2.txt* and *WordCountARTICLE3.txt*) are provided alongside this document.

2 Targeted advertising using MapReduce

Based on the *WordCount.java* code provided for the previous section, a file *Advertising.java* was generated to execute the *MapReduce* operation specified in the assignment description.
This file, as well as the output generated after running it (*AdvertisingOUTPUT.txt*), are provided alongside this document.

(a) Figure 6: Snapshot of the *Advertising* project in Eclipse

(b) Figure 7: Snapshot of the output file *AdvertisingOUTPUT.txt*

3 Recommender system using Apache Mahout

Following the tutorial for the simple recommender system provided in the assignment, the files *SampleRecommender.java* and *EvaluateRecommender.java* (provided with this document) were generated. The following images show a snapshot of both classes under the Eclipse IDE:

(a) Figure 8: Snapshot of the *SampleRecommender.java* class in Eclipse

(b) Figure 9: Snapshot of the *EvaluateRecommender.java* class in Eclipse

[1] The summary available in *localhost:8088* only provides overall information of the system. I was not able to find information regarding the mappers and reducers specifically, as requested in the assignment.

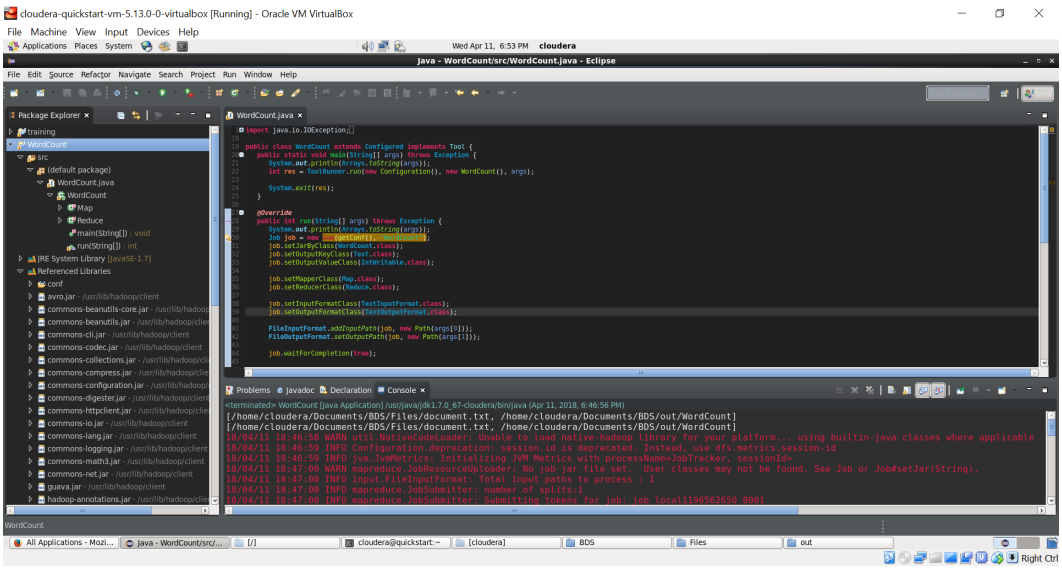

Figure 1

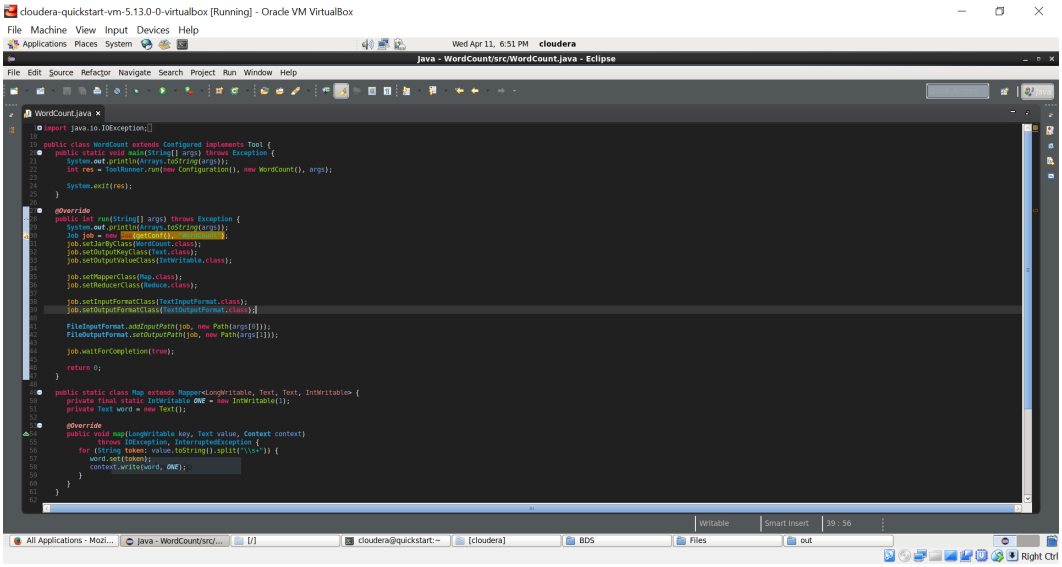

Figure 2

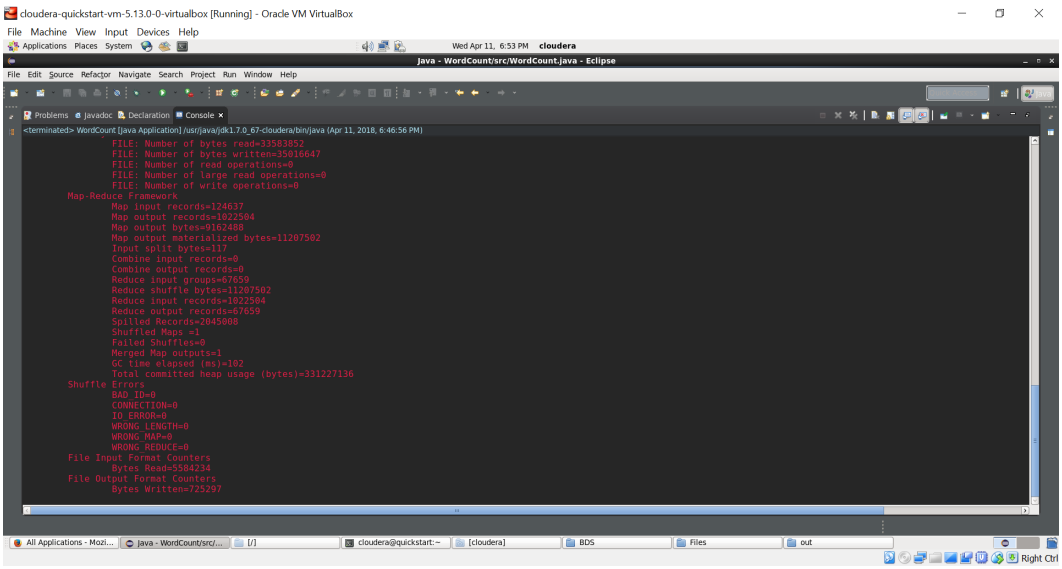

Figure 3

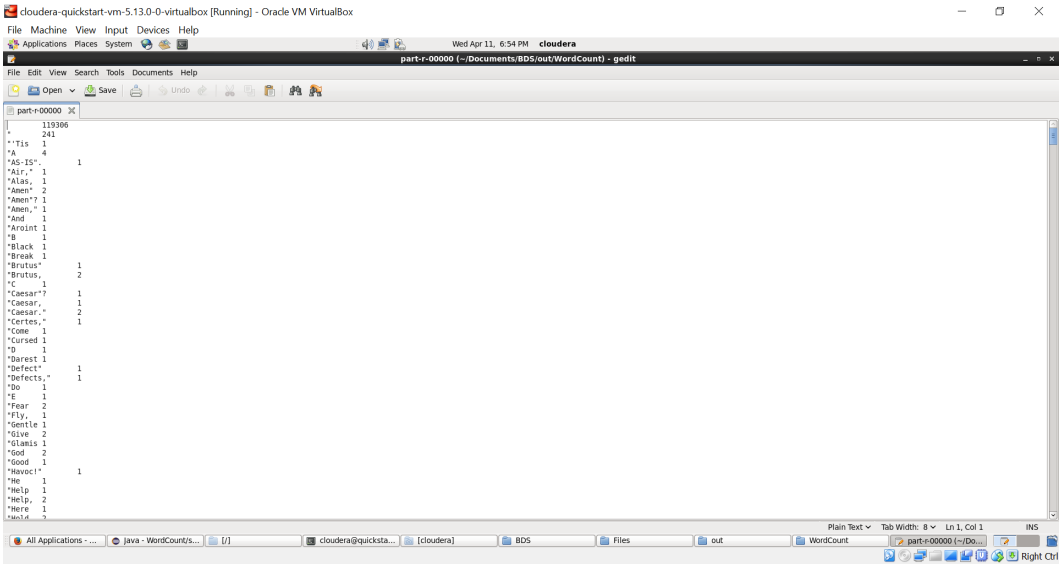

Figure 4

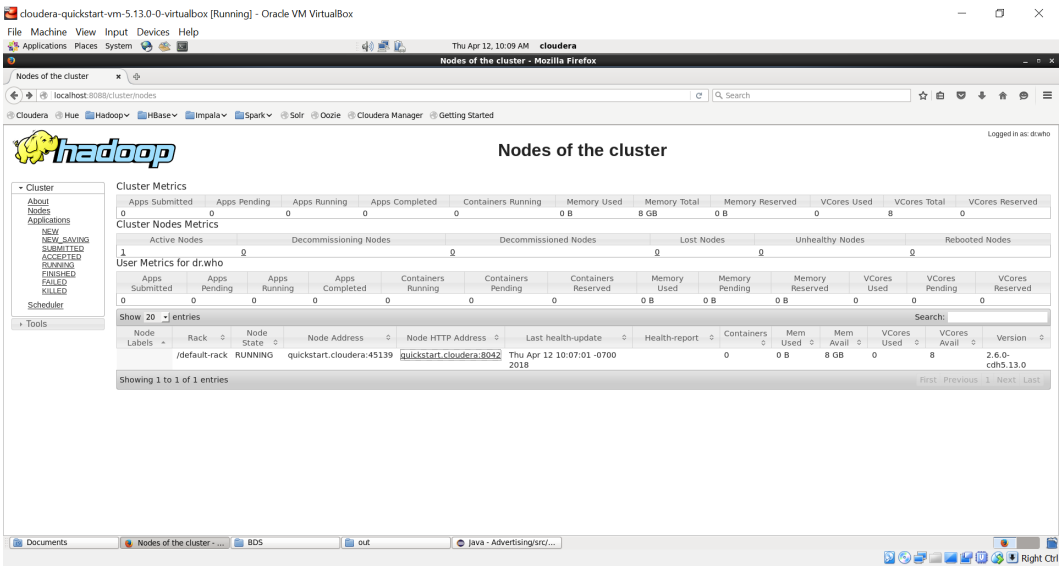

Figure 5

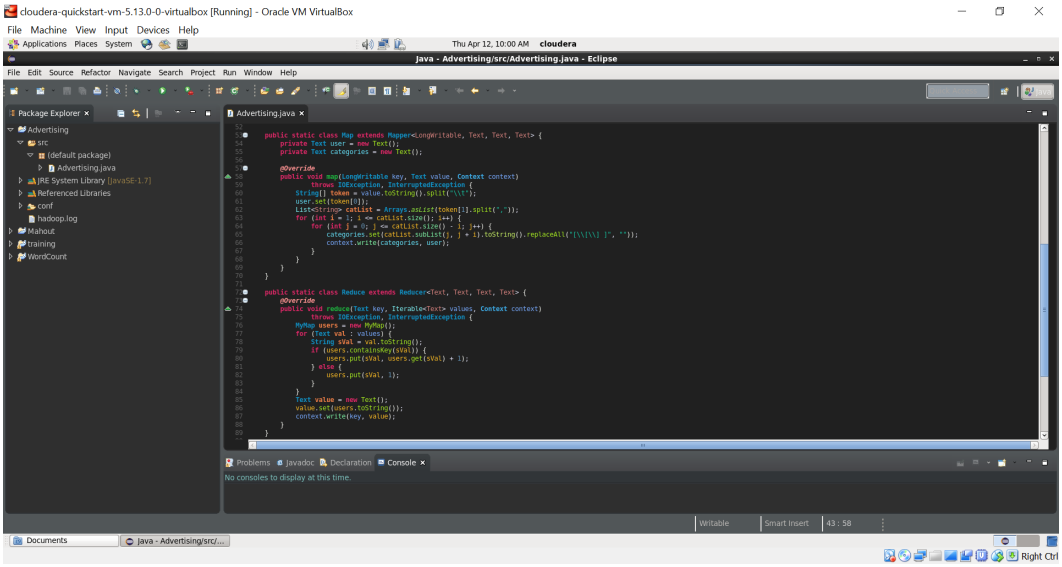

Figure 6

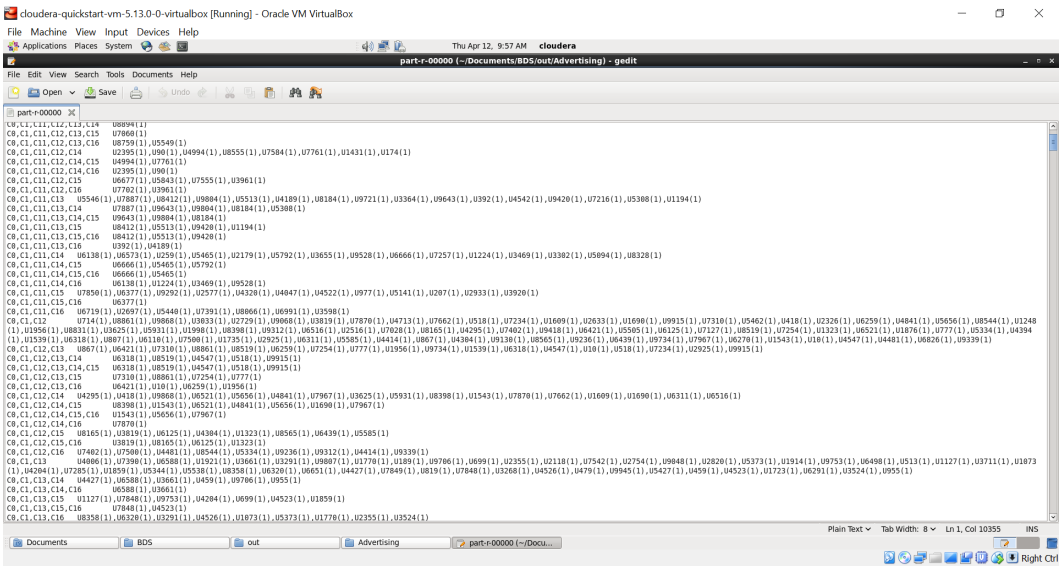

Figure 7

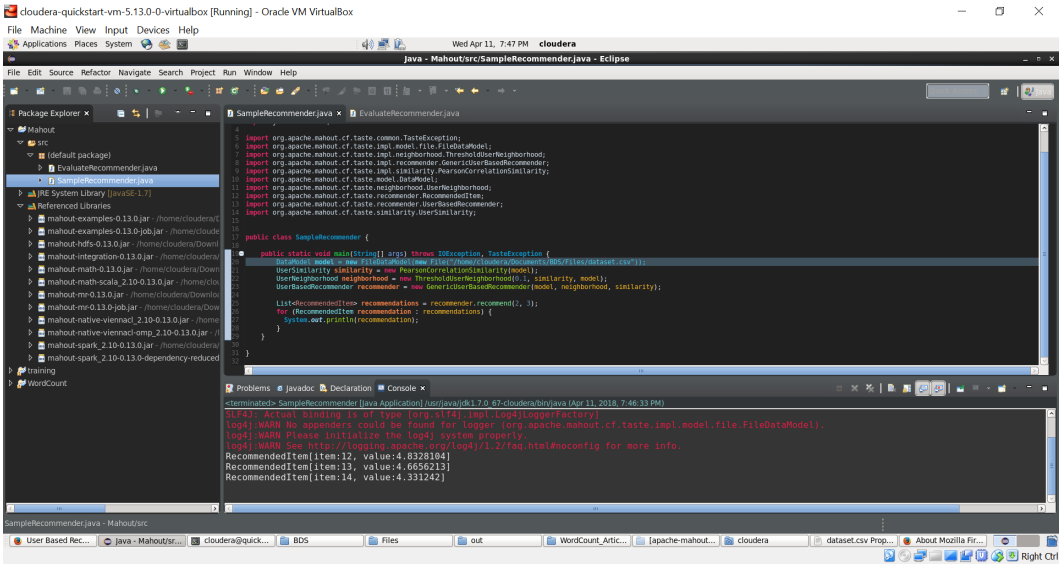

Figure 8

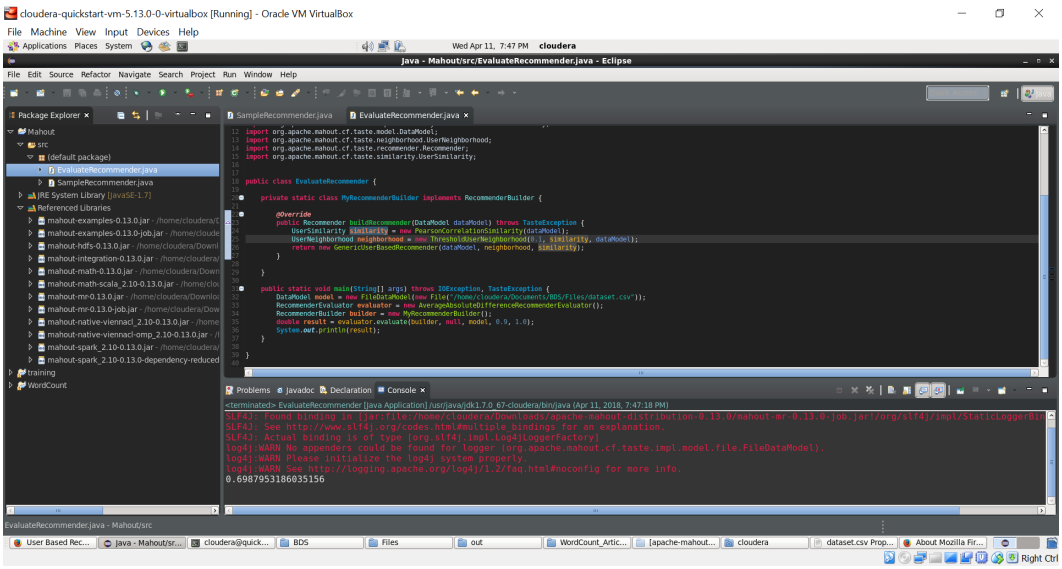
cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
Wed Apr 11, 7:47 PM cloudera
Java - Mahout/src/EvaluateRecommender.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
SampleRecommender.java
EvaluateRecommender.java
Mahout
src
(default package)
EvaluateRecommender.java
SampleRecommender.java
JRE System Library [JavaSE-1.7]
Referenced Libraries
mahout-examples-0.13.0.jar
mahout-examples-0.13.0-job.jar
mahout-hdfs-0.13.0.jar
mahout-integration-0.13.0.jar
mahout-math-0.13.0.jar
mahout-math-scala_2.10-0.13.0.jar
mahout-mr-0.13.0.jar
mahout-mr-0.13.0-job.jar
mahout-native-viennacl_2.10-0.13.0.jar
mahout-native-viennacl-omp_2.10-0.13.0.jar
mahout-spark_2.10-0.13.0.jar
mahout-spark_2.10-0.13.0-dependency-reduced
training
WordCount
import org.apache.mahout.cf.taste.model.DataModel;
import org.apache.mahout.cf.taste.neighborhood.UserNeighborhood;
import org.apache.mahout.cf.taste.recommender.Recommender;
import org.apache.mahout.cf.taste.similarity.UserSimilarity;
public class EvaluateRecommender {
private static class MyRecommenderBuilder implements RecommenderBuilder {
@Override
public Recommender buildRecommender(DataModel dataModel) throws TasteException {
UserSimilarity similarity = new PearsonCorrelationSimilarity(dataModel);
UserNeighborhood neighborhood = new ThresholdUserNeighborhood(0.1, similarity, dataModel);
return new GenericUserBasedRecommender(dataModel, neighborhood, similarity);
}
}
public static void main(String[] args) throws IOException, TasteException {
DataModel model = new FileDataModel(new File("/home/cloudera/Documents/BDS/Files/dataset.csv"));
RecommenderEvaluator evaluator = new AverageAbsoluteDifferenceRecommenderEvaluator();
RecommenderBuilder builder = new MyRecommenderBuilder();
double result = evaluator.evaluate(builder, null, model, 0.9, 1.0);
System.out.println(result);
}
}
Problems Javadoc Declaration Console
<terminated> EvaluateRecommender [Java Application] /usr/java/jdk1.7.0_67-cloudera/bin/java (Apr 11, 2018, 7:47:18 PM)
SLF4J: Found binding in [jar:file:/home/cloudera/Downloads/apache-mahout-distribution-0.13.0/mahout-mr-0.13.0-job.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
log4j:WARN No appenders could be found for logger (org.apache.mahout.cf.taste.impl.model.file.FileDataModel).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
0.6987953186035156
EvaluateRecommender.java - Mahout/src
User Based Rec...
Java - Mahout/sr...
cloudera@quick...
BDS
Files
out
WordCount_Artic...
[apache-mahout...
cloudera
dataset.csv Prop...
About Mozilla Fir...
Right Ctrl

Figure 9