CS 6350 - ASSIGNMENT 1b

Please read the instructions below before starting the assignment.

- This assignment consists of two parts. Please create an IntelliJ Idea project with two classes in two different files.

- You should use a cover sheet, which can be downloaded at: http://www.utdallas.edu/~axn112530/cs6350/CS6350_CoverPage.docx

- You are allowed to work in pairs i.e. a group of two students is allowed. Please write the names of the group members on the cover page. Only one submission per team is required.

- You have a total of 4 free late days for the entire semester. You can use at most 2 days for any one assignment. After that, there will be a penalty of 10% for each late day. The submission for this assignment will be closed 2 days after the due date.

- Please ask all questions on Piazza, and not through email to the instructor or TA.

ASSIGNMENT 1b

Part 1 (20 points)

In the first part of assignment 1, you uploaded downloaded and extracted six large text files on HDFS. In this part, you will run a modified version of the WordCount algorithm on all of the files taken together using MapReduce.

Below are the requirements:

1. You will remove stop words during the map phase, and consider only meaningful words i.e. words other than stop words. A list of stop words can be downloaded from https://www.textfixer.com/tutorials/common-english-words-with-contractions.txt

2. You will also remove words that are less than 5 characters in length, special characters (e.g. "," or "."), and convert all words to lowercase.

3. In the reduce phase, you will generate a total count for each key i.e. word and output that to a HDFS file.

4. Finally, you will sort the data by values in a descending way and output the top 20 most frequent words.

Note:

- You have to accomplish all this using MapReduce code written in Java that can be run on the UTD Hadoop cluster as a jar file. Any other solution will not be accepted.
- Please do not use hard-coded path in your code. The program should read arguments for the locations of input and output file from the command line. You can use the project in lab3 as a template.
- **Hint**: The class Path in org.apache.hadoop.fs can represent an entire directory in HDFS
- **UTD cluster does not have DistributedCache. Please do not attempt to put files there, it may cause the cluster to crash.**

Deliverable:

- You should create a Java project and add a class for this part.
- Readme file indicating how to run your code and the output of your program.

Part 2 (20 points)

In the second part of the assignment, you will read in a large file containing movie ratings and use MapReduce to calculate the average rating for each movie.

We will use the academic version of the MovieLens dataset which is available at:
https://grouplens.org/datasets/movielens/latest/

The files are available on UTD Hadoop cluster at this location:
hdfs://cshadoop1/movielens
Please use the files from the above location as input.

Below are the requirements:

1. In the mapper code, you will read in the ***ratings.csv*** file and use the movieId as the key and the rating field as the value.

2. In the reducer code, you will compute the average rating (as a double datatype) for each movieId and store it in the output directory on HDFS.

3. Finally, you will sort the data by values in a descending way, and output the top 20 movies with the highest average ratings.

Note:

- You have to accomplish all this using MapReduce code written in Java that can be run on the UTD Hadoop cluster as a jar file. Any other solution will not be accepted.
- Please do not use hard-coded path for the output in your code. The program should read arguments for the output file from the command line. You can use the project in lab3 as a template.
- **UTD cluster does not have DistributedCache. Please do not attempt to put files there, it may cause the cluster to crash.**

What to submit:
- In the same project that you have part 1, add another class for this program.
Submit the zipped file for the project.
- Readme file indicating how to run your code and the output of your program.