Data Mining

Homework 4

Due: 14/5/2013, 23:59.

You must hand in the homeworks electronically and before the due date and time. Check the web page for instructions about collaboration, about being late, and about handing in the homework.

Hand in:

- The `.py` files with the source code.
- Instruction to execute them.
- The first 10 lines of the file that you created from project Gutenberg.
- The output files generated, and output at the screen.

Problem 1. In this assignment we will implement one round of the k-means algorithm in Hadoop. The specifications of the program are to accept a file where each line corresponds to a document and a value k of the number of clusters, then select k random initial centers, run a round of the k-means algorithm, and return a file with the new centers.

In more detail, the format of the input file is:

$$\mathtt{fileID_i}\backslash\mathtt{t}\ \mathtt{term_{i1}:tfidf_{i1},term_{i2}:tfidf_{i2}}\ldots$$

where $\mathtt{fileID_i}$ is an ID (e.g., the filename) of document i, $\mathtt{term_{ij}}$ the jth term of the ith document, and $\mathtt{tfidf_{ij}}$ the TFIDF value of the jth term in the ith document, after ℓ_2 normalization so that for every i we have that

$$\sum_{j=1}^{n}(\mathtt{tfidf_{ij}})^2 = 1.$$

The main idea of the (full) k-means algorithm in Hadoop should be almost obvious, and it is as follows:

1. First we select the k random centers using MapReduce. To do that we can assign a random number to each document and we select the top k. An alternative, more efficient way (proposed by some of you), is to select the random initial clusters during the preprocessing, and avoid this round of MapReduce.
2. The main part of the algorithm consists of several map–reduce rounds. In the map step, every document must select its closest center. In the reduce rounds, for every cluster we have a list of the documents assigned to it and we recompute the new center.
3. After convergence, for each cluster we report the documents assigned to it. We can also do this with MapReduce.

For this problem, we ask you to implement a single round of k-means for MapReduce. After you select the random centers, execute a single MapReduce phase, and report the updated centers.

After you write and test your program, try to run your algorithm for some of the books of the Gutenberg project (`http://www.gutenberg.org`). Download the August 2003 CD:

`http://www.gutenberg.org/wiki/Gutenberg:The_CD_and_DVD_Project`

The goal is to run you algorithm for all the `.txt` files in the CD. First preprocess the document. Create a single file and for each `.txt` file in the CD:

1. Remove the project Gutenberg header.
2. Remove stopwords, convert to lowercase and stem.
3. Store them in a single file in the format described above.

After you prepare the file, run your MapReduce round with $k = 10$, 20, and 50.

Optionally (extra credit) you can implement the entire k-means algorithm. If interested, read on.

To implement the full MapReduce algorithm, even though the idea of the algorithm is simple it requires quite some programming work. In the following we give you some suggestions for implement it using `dumbo`, but they are only suggestions and you may have some other ideas.

- At every iteration, you will need to pass all the centers to each mapper. You can do this by using an external file.
- You will need three different MapReduce programs (one for choosing the random centers, one to perform an iteration of k-means, and one to report the final clusters) and a main program that controls them. Or you can do it in with two MapReduce programs, if you select the centers avoiding the use of MapReduce.

After you prepare the file, cluster it with $k = 10$, 20, and 50. Try multiple times. Does it end up with the same cluster? Can you suggest a way to combine the different clusterings?