Third Homework Assignment (20% of grade)

Due Date: **5/17/2017** at **11:59pm**

Submission Guidelines:

Please follow carefully the instructions posted on the course's github page when submitting your solutions (https://github.com/MSIA/bigdatacourse/blob/master/README.md). Failure to follow the instructions will result in lost points.

Deliverables:

1. Your java source code file: name the file "Kmeans.java". **NOTE: the class name has to be "Kmeans.java" as well.**
2. Reducer output for each dataset: name the file "KmeansX.txt" where X is either 1 or 2.
3. External script.
4. Short (1/1.5-page) document summarizing your approach and results after running K-means on the medicare dataset.

Notes/tips:

1. All input files are available in /home/public/course. Please copy them directly from /home/public/course to HDFS and not to your home directory on wolf.
2. The directory /home/public/setProject has directions on how to set up a maven project.

K-Means Clustering

Data: /home/public/course/clustering/

In this assignment you will implement the k-means clustering algorithm (https://en.wikipedia.org/wiki/K-means_clustering#Standard_algorithm) in MapReduce. You will find three files in the clustering directory: clustering.txt, and two files about the medicare data set (data file, and pdf file with information). The medicare file is rather large, so you can use the clustering.txt file to test and debug your code.

Coding tips and instructions:

- You can assume that the number of clusters is known in advance. Furthermore, the number of clusters **MUST NOT** be hardcoded in your code. (See RandomSampling.java on the homework 2 discussion board for how to pass command line arguments to your MapReduce program).
- Each record in the input files corresponds to one observation, and each column represents a different variable.
- You can use a termination criterion of your choice for the kmeans algorithm – a maximum number of iterations is the simplest one.
- The k-means clustering algorithm follows an iterative framework (the same steps are repeated multiple times before termination). You can use any external tool (python is a good choice for this) to execute your MapReduce routines multiple times. In a nutshell, you have to:
    - Write a MapReduce program that will execute a single iteration of k-means, and makes use of the **distributed cache** concept.
    - Write an **external script** that will call this MapReduce program multiple times. The script will take the output of the previous iteration, and upload in HDFS so that it can be used in the following iteration.
- Initial centroids: you can initialize them in a number of ways, e.g. uniform distribution over the 0-1 interval.
- **Your code needs to be generic enough to run on both datasets!** This means that (1) your program must be able to handle any whitespace delimited file (both files are whitespace delimited) and (2) the format of the input file to your program must be consistent (i.e. no header, and all columns in the input file must be used in the clustering procedure – this only applies to the medicare data, see below).
- You will need to **standardize** the medicare data (not needed for clustering.txt).
- The output should be the final cluster centroids. Note that you need to submit one output file for clustering.txt (Kmeans1.txt) and another for the medicare dataset (Kmeans2.txt)

Medicare data:

A couple of years ago, the US Government released medicare data. The dataset has 10 million records so don’t forget to copy it directly from the local folder to Hadoop – bypassing copying to your local directory; if you want to peek at the data, use ‘more’ or ‘less’. There is an accompanying pdf file that describes the data. I noticed that there are several numerical fields, so clustering is possible. I challenge you to come up with relevant clustering problems from this data set.

You can read more about the dataset at http://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/Medicare-Provider-Charge-Data/Physician-and-Other-Supplier.html (Don't download the data again, it is pretty large.)

Since you will be using only a subset of the columns from the medicare dataset, you must write another MapReduce program (no need to turn it in) that removes the file's header, and extracts only the relevant columns. Only this way will your Kmeans.java code be generic enough to run on both datasets.

Medicare report:

In addition to your code and output files, you have to submit a short document (not more than 1 page), that outlines (1) the features selected and your reasoning behind it, and (2) insights from the clustering (needless to say that you are welcome to use Tableau, R, d3 to produce breathtaking visualizations).

If someone comes up with interesting results, we'll send them to newspapers (they have been beating to death this data set – I haven't read about clustering studies, but more easy statistics).