Assignment Map Reduce MPI: k-Nearest Neighbor

The purpose of this assignment is for you to learn more about

- Using Map Reduce to solve a data science problem.

As usual all time measurements are to be performed on the cluster. The application using MapReduce MPI need to be linked against the mrmpi library. In the assignment, the Makefile accounts for it.

1 k-Nearest Neighbor

The k-Nearest Neighbor algorithm is a machine learning algorithm to infer classification of a query based on known classification of a set of observation.

Mathematically, you are given a database of n points located a d-dimensional feature space, and each point is associated with a class (an integer) and queries, vectors in the same d dimensional space. The k-nearest neighbor algorithm will guess a class for each query by: identifying the k points in the database that are the closest to the query (by euclidean distance), computing which class appears the most frequently among the k nearest neighbors.

Question: Go into the `knn/` directory and write the code in `knn/knn_mrmpi.cpp` using MapReduce MPI. There is a sequential implementation of the problem in `knn/knn_seq.cpp`.

The code should take three parameters: the name of the database file, the name of the query file, and k (the number of neighbors to consider).

The database file is a list of points in a high dimensional space and classified. That is to say each line is a point in a high dimensional space. Each line is composed of comma separated values, the last value is an integer class; all d others are coordinates in the high dimension space.

The query file is a similar format. The first entry of the line is a queryID, followed by d values which are coordinates in the high dimension space. And of course, without a class.

The program should:

- Print on `stdout` the queryIDs and estimated class the queryID the k-Nearest Neighboor maps it to.
- Print on `stderr` the time it took for the application to make that computation (IO included).

You can test your code with `make test`. Though you will notice that the test is not as complete as in previous assignments.

Question: Run the code on Centaurus using `make bench`. And once it is over, print a time table with `make plot`.