Stats 141C High Performance Statistical Computing Spring 2018

Homework 2

Lecturer: Cho-Jui Hsieh Date Due: May 22, 10:20am, 2018

Keywords: *Multicore Programming*

For this homework, we will use the data and code downloaded from `http://www.stat.ucdavis.edu/~chohsieh/teaching/STA141C_Spring2018/hw2_code.zip`. In this folder, we provide the code for the nearest neighbor classification algorithm in "go_knn.py" (you can directly run this python code, and it will print out the accuracy on the dataset). The main function in this file is "go_nn": for each testing sample, it computes the nearest neighbor in the training set, and checks whether the label predicted by the nearest neighbor matches the testing label.

Problem 1. Multicore Programming [50 pt]

Modify the "go_nn" function to parallelize the computation using multiple cores. We suggest using python "multiprocessing" module. Make sure you get the same accuracy with the original code. Run your new code using 4 cores and report the run time. Compare the run time with the original (single threaded) code.

Problem 2. Parallel Gradient Descent [50 pt]

In previous homework (Hw1, Problem 2) you have implemented the gradient descent solver for logistic regression. Now try to parallelize the program using multicore programming.

Recall that the logistic regression problem is

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \{\frac{1}{n}\sum_{i=1}^{n} \log(1 + e^{-y_i \boldsymbol{w}^T \boldsymbol{x}_i}) + \frac{\lambda}{2}\|\boldsymbol{w}\|^2\} := f(\boldsymbol{w}). \tag{1}$$

Each iteration, the gradient descent method needs to compute

$$\nabla f(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^{n} \frac{-y_i}{1 + e^{y_i \boldsymbol{w}^T \boldsymbol{x}_i}} \boldsymbol{x}_i + \lambda \boldsymbol{w}.$$

Try to parallelize this gradient computation in your code. Run your new code using 4 cores and report the run time. Compare the run time with the original (single threaded) code. [Partial credit will be given even if you do not see any speed up after parallelizing the code]