EECS 284 Big Data Systems and Analytics — Lab 01 (50 points)
Due date: February 12

(Mini-)Batch Gradient Descent

Description You are required to implement (mini-)batch gradient descent for logistic regression with two labels (+1/-1). The logistic regression loss function is:

$$f_{LR}(\vec{w}) = \log\left(1 + e^{-y_i \vec{x_i} \cdot \vec{w}}\right)$$

where the d-dimensional vector $\vec{w}$, $d \geq 1$, is the model that has to be found such that the loss function is minimized. The constants $\vec{x_i}$ and y_i, $1 \leq i \leq N$, correspond to the feature vector of the $\mathrm{i^{th}}$ data example and its scalar label, respectively. The gradient of the logistic loss is:

$$\frac{\partial f_{LR}(\vec{w})}{\partial w_j} = x_{ij}\left(-y_i \frac{e^{-y_i \vec{x_i} \cdot \vec{w}}}{1 + e^{-y_i \vec{x_i} \cdot \vec{w}}}\right)$$

and in linear algebra form:

$$\vec{g} = \vec{\vec{X}}_k^T \odot \left(-\vec{Y}_k \cdot \frac{e^{-\vec{Y}_k \cdot \vec{\vec{X}}_k \odot \vec{w}}}{1 + e^{-\vec{Y}_k \cdot \vec{\vec{X}}_k \odot \vec{w}}}\right)$$

You have to implement both a serial and a parallel version of the algorithm. The parallelization strategy is based on parallel kernels for the linear algebra operations. You write the loss and the gradient as linear algebra expressions made of basic operations such as scalar product between two vectors, matrix-vector multiplication, matrix-matrix multiplication, etc. You find a linear algebra library that supports these operations with parallel implementations, e.g., ViennaCL, and call the corresponding functions accordingly.

Input structure You are provided a sparse dataset `w8a` consisting of 59,245 examples. The dimensionality of the dataset is 300. Each example is given on a separate line and follows the following format. First, the label +1 or -1 is given. Then, each non-zero dimension of the example is given as a pair index:value, where index is the dimension with its corresponding feature value. Your program has to take as input several other parameters: the number of iterations; the step size; the decay, which represents the decrease of the step size at each iteration; the batch size; and the number of threads used to execute the program.

Output structure The output prints the value of the loss function at each iteration in the following format:
iteration #: loss_value time_since_start

Submission You have to show how your program executes to the instructor, during the lab. You have complete freedom to choose the programming language for your implementation as long as it satisfies the requirements.