CSCE 5063-001: Assignment 2

Due 11:59pm Friday, October 8, 2021

Name: Rashmitha Thoranala
Mail: rthorana@uark.edu
Id: 010962830

Implementation of SVM via Gradient Descent

I used the SVM to predict if a person doesn't have diabetes (negative class) or has diabetes (positive class), where the data is from National Institute of Diabetes and Digestive and Kidney Diseases. A total of 8 features will be used, including Preg (number of times pregnant), Pres (Diastolic blood pressure), Mass (body mass index), etc. There are a total number of 760 patients in the data.

In the given data.txt file the first 8 columns represent the features and last column represent the label and here, we do not need to perform feature normalization and separating test/train data and do not need to shuffle the data as well.

I implemented the soft margin with SVM using Batch, Stochastic and Mini batch gradient descent techniques. The loss function of SVM is given by:

$$J(w, b) = \frac{1}{2} \sum_{j=1}^{n} w_j^2 + C \sum_{i=1}^{m} \max\{0, 1 - y^{(i)}(\sum_{j=1}^{n} w_j x_j^{(i)} + b)\}$$

We have to minimize the loss function and obtained gradient with respect to w_j and the equation for gradient is:

$$\nabla_{w_j} J(w, b) = \frac{\partial J(w, b)}{\partial w_j} = w_j + C \sum_{i=1}^{m} \frac{\partial L(x^{(i)}, y^{(i)})}{\partial w_j}$$

$$\frac{\partial L(x^{(i)}, y^{(i)})}{\partial w_j} = \begin{cases} 0 & \text{if } y^{(i)}(x^{(i)} w + b) >= 1 \\ -y^{(i)} x_j^{(i)} & \text{otherwise} \end{cases}$$

Gradient with respect to b:

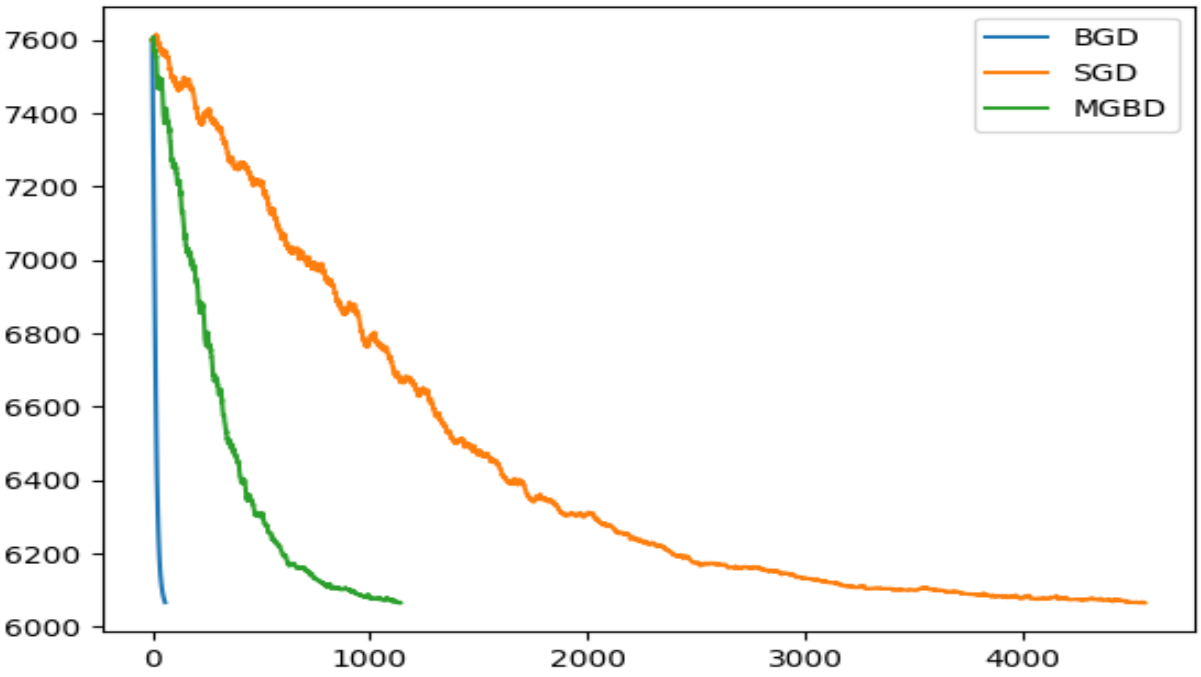
Now, let us compute the gradient with respect to b which is given by:

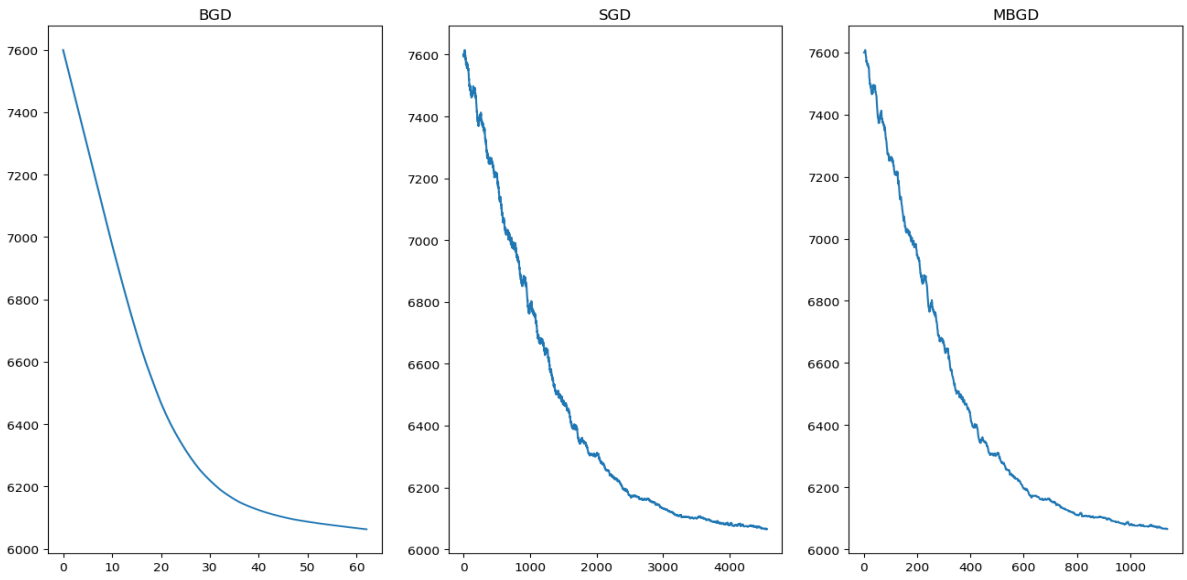
$$\nabla_b J(w, b) = \frac{\partial J(w, b)}{\partial b} = 0 + C \sum_{i=1}^{m} \frac{\partial L(x^{(i)}, y^{(i)})}{\partial b}$$

$$\frac{\partial L\left(x^{(i)}, y^{(i)}\right)}{\partial w_j} = \begin{cases} 0 & \text{if } y^{(i)}(x^{(i)}w + b) >= 1 \\ -y^{(i)} & \text{otherwise} \end{cases}$$

Plotting $J_k(w, b)$ *vs no. of iterations* (k):

I implemented Batch Gradient descent, Stochastic Gradient Descent and Mini-batch Gradient Descent algorithms and found the $J_k(w, b)$ and now plot this values vs number of iterations (k) and all the three graphs looks like:





Convergence Time:

- The total time taken for convergence of BGD is 0.015673398971557617seconds.
- The total time taken for convergence of SGD is 0.2178807258605957seconds.
- The total time taken for convergence of MBGD is 0.10481834411621094 seconds.

If we remove the code for computing the cost function in each iteration for SGD and MBGD then the measure of time for is given by:

- The total time taken for convergence of SGD is 0.1223452091217041seconds.
- The total time taken for convergence of MBGD is 0.060402631759643555seconds.

All these three algorithms converge with the cost values around 6065.