CSE 417T: Homework 3

Due: October 19 (Wednesday), 2022

Notes:

- Please submit your homework via Gradescope and check the submission instructions.
- There will be two submission links for homework 3: one for report and the other for code. **Your score will be based on the report.** The code you submit is only used for checking the correctness and for running plagiarism checkers.
- Make sure you **specify the pages for each problem correctly**. You **will not get points** for problems that are not correctly connected to the corresponding pages.
- Homework is due **by 11:59 PM on the due date.** Remember that you may not use more than 2 late days on any one homework, and you only have a budget of 5 in total.
- Please keep in mind the collaboration policy. If you discuss questions with others you **must** write their names on your submission, and if you use any outside resources you **must** reference them. **Do not look at each others' writeups, including code.**
- There are 4 problems on 3 pages in this homework.

Problems:

1. (40 points) The weight decay regularizer is also called L_2 regularizer, since $\vec{w}^T\vec{w}$ is the square of the 2-norm of the weight vector $\|\vec{w}\|_2 = \sqrt{\sum_{i=0}^{d} w_i^2}$. Another common regularizer is called L_1 regularizer, since 1-norm ($\|\vec{w}\|_1 = \sum_{i=0}^{d} |w_i|$) is used as the regularizer.

   Below are the definitions of the two regularizations[1]:

   - L_1 regularization: $E_{aug}(\vec{w}) = E_{in}(\vec{w}) + \lambda\|\vec{w}\|_1$
   - L_2 regularization: $E_{aug}(\vec{w}) = E_{in}(\vec{w}) + \lambda\vec{w}^T\vec{w}$

   (a) Answer LFD Problem 4.8.

   (b) Similar to Problem 4.8, derive the update rule of gradient descent for minimizing the augmented error with L_1 regularizer.

   Note that the gradient of 1-norm is not well-defined at 0. To address this issue, we can utilize the *subgradient* idea defined as follows:

$$\frac{\partial}{\partial w_i}\|\vec{w}\|_1 = \begin{cases} +1 & \text{if } w_i > 0 \\ \text{any value in } [-1, 1] & \text{if } w_i = 0 \\ -1 & \text{if } w_i < 0 \end{cases}$$

[1] When applying these regulerizations to linear regression, they are called Ridge Regression (L_2 regularizer) and Lasso Regression (L_1 regularizer) respectively.

To simplify the discussion, we let $\frac{\partial}{\partial w_i}\|\vec{w}\|_1 = 0$ when $w_i = 0$. Please write down the update rule of gradient descent for L_1 regularization. (You can define a $sign()$ function that returns $+1, 0, -1$ when the input is positive, zero, negative).

Truncated gradient (for part (c)): In Lasso regression (linear regression with L_1 regularization), one nice property is that it tends to learn a weight vector with many 0s. However, if we perform gradient descent on the augmented error with L_1 regularization, it won't lead to this nice property partly due to the not-well-defined behavior of subgradient. In this homework, you will implement *truncated* gradient [1], an approach trying to maintain the nice property of L_1 regularizations, as described below.

Let $\vec{w}'(t+1) \leftarrow \vec{w}(t) - \eta \nabla E_{\text{in}}(\vec{w}(t))$ be the update rule of gradient descent without regularization. The update rule for L_1 regularization that you derived should be in the form of

$$\vec{w}(t+1) \leftarrow \vec{w}'(t+1) + \text{additional term}$$

The additional term represents the effect of L_1 regularization compared with no regularization. Truncated gradient works as follows: At each step t, you first perform the update and obtain $\vec{w}(t+1)$. Then for each dimension i, if $w_i(t+1)$ and $w_i'(t+1)$ have different signs and when $w_i'(t+1) \neq 0$, we set the update $w_i(t+1)$ to 0 (i.e., we *truncate* the update if the additional term makes the new weight change signs). [2]

(c) Update your implementation of logistic regression in HW2 to include the L_1 and L_2 regularizers (use truncated gradient for L_1 regularizer and regular gradient descent for L_2 regularizers). Conduct the following experiment and include the results in your report. Also submit the updated python implementation (feel free to update the function headers and/or define new functions).

You will work with digits dataset, classifying whether a digit belongs to $\{1, 6, 9\}$ (labeled as -1) or $\{0, 7, 8\}$ (labeled as $+1$). Below is the link to the pre-processed data.
`http://chienjuho.com/courses/cse417t/hw3/hw3.html`

Examine different $\lambda = 0, 0.0001, 0.001, 0.005, 0.01, 0.05, 0.1$ for both L_1 and L_2 regularizations. Train your models on the training set. For each trained model, report (1) the classification error on the test set and (2) the number of 0s in your learned weight vector. Describe your observations.

For the other parameters, please use the following. Normalize the features. Set learning rate $\eta = 0.01$. The maximum number of iterations is 10^4. Terminate learning if the magnitude of every element of the gradient (of E_{in}) is less than 10^{-6}. When calculating classification error, classify the data using a cutoff probability of 0.5.

Note: While we don't grade on your code efficiency, you are encouraged to check out *vectorization*. The difference in running time is significant.

2. (15 points) LFD **Exercise** (not Problem) 4.5

3. (25 points) LFD Problem 4.25 (a) to (c)

[2] One informal way to think about truncated gradient is that, it splits each update into many tiny steps of updates. When a tiny update makes some dimension of the weight cross 0, you can assign the subgradient value in the additional term appropriately to make it stay at 0. So it is still a valid gradient descent for L_1 regularization, but just "choose" values for subgradients to make the weights stay at 0 as much as possible.

4. (20 points) LFD Problem 5.4. The problem makes a simplifying definition: a stock is called "profitable" if it went up half of the days (and whether a stock goes up or down in a day is a random draw from a distribution that associates with how good that stock is). While this definition of "profitable" might not be accurate in practice, please use it for your discussion.

References

[1] John Langford, Lihong Li, and Tong Zhang. Sparse online learning via truncated gradient. *Journal of Machine Learning Research*, 10:777801, 2009.