CSE 251A: Machine learning Winter 2021

Programming project 2 — Coordinate Descent

Name: Xuanyu Chen

PID: A53306915

1. **Description**

Coordinate descent is a simple and useful approach to solve an unconstrained optimization problem. Coordinate descent algorithms solve optimization problems by successively performing approximate minimization along coordinate directions. If

is a k-dimensional function, we can minimize by successively minimizing each of the individual dimensions of while holding the values of in the other dimensions fixed. The primary advantage of this approach is that it takes an arbitrarily complex k-dimensional problem and reduces it to a collection of k one-dimensional problems.

1. **Pseudocode**

**Algorithm 1** Coordinate Descent

1. **Input**:

: a positive constant, the step-size for every update in one coordinate

: threshold

N: maximum number of iterations

F: bool, if False means the loss value is not optimal, True means optimal

1. **Output**: vector and optimal loss value
2. **Procedure** COORDINATEDESCENT
3. set and choose
5. **while** :
6. choose index which is
8. ;
10. **while** *:*
11. choose index which is
13. ;
15. **if** :
17. **break**
19. *;*
20. **Experimental results**

The optimal log-loss calculated based on the LogisticRegression function (with parameter C = 50) from scikit-learn package is 0.02102.

Here are the results calculated by my Coordinate Descent algorithm:

Table 1.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Max-Iteration | 500 | 1500 | 5000 | 7000 | 9000 |
| alpha | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| threshold | 1e-3 | 1e-4 | 1e-5 | 1e-6 | 1e-6 |
| Log-loss | 0.475 | 0.158 | 0.0589 | 0.048 | 0.0386 |
| iterations | 403 | 1500 | 4622 | 7000 | 9000 |

Figure 1. (First 100 iterations)

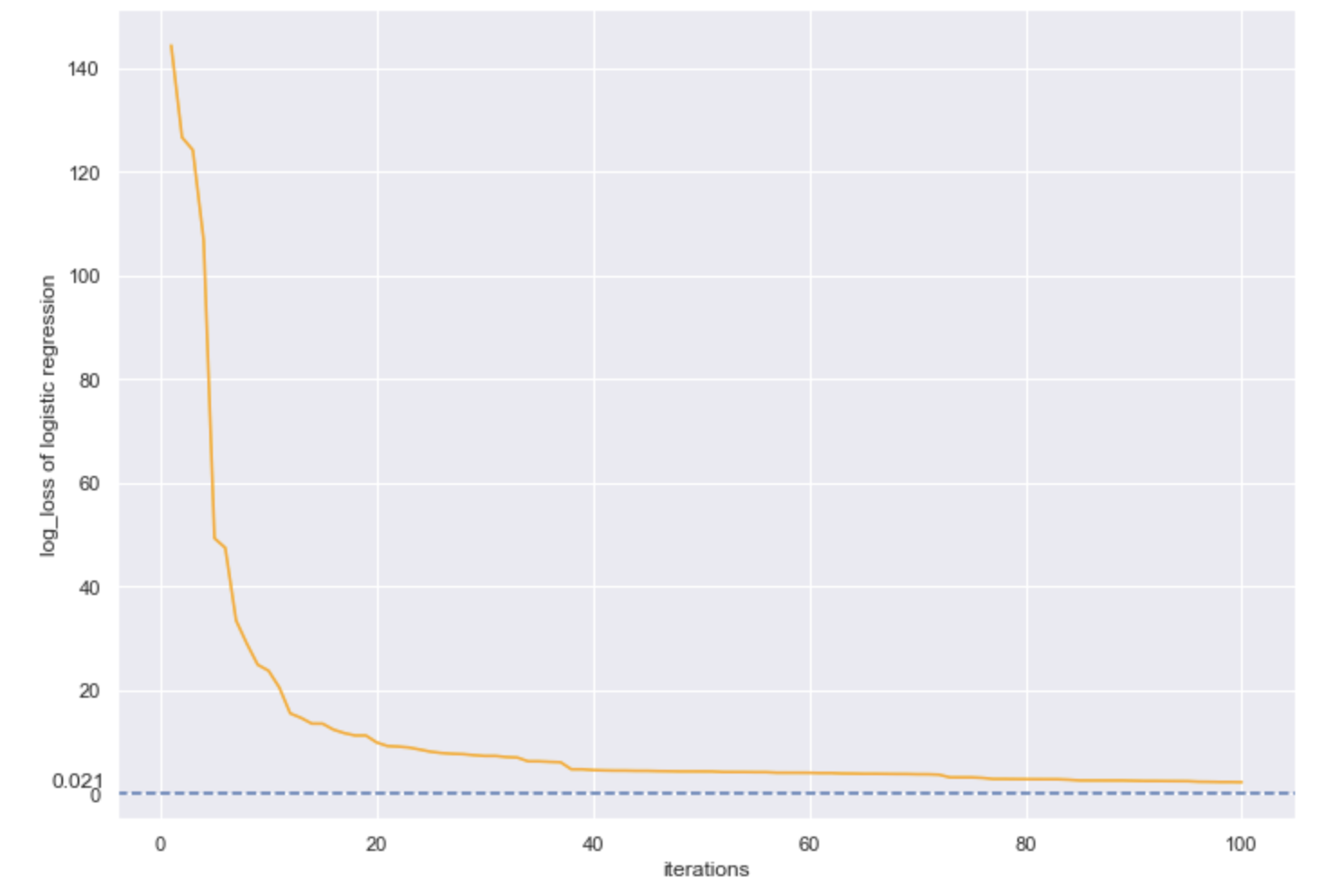
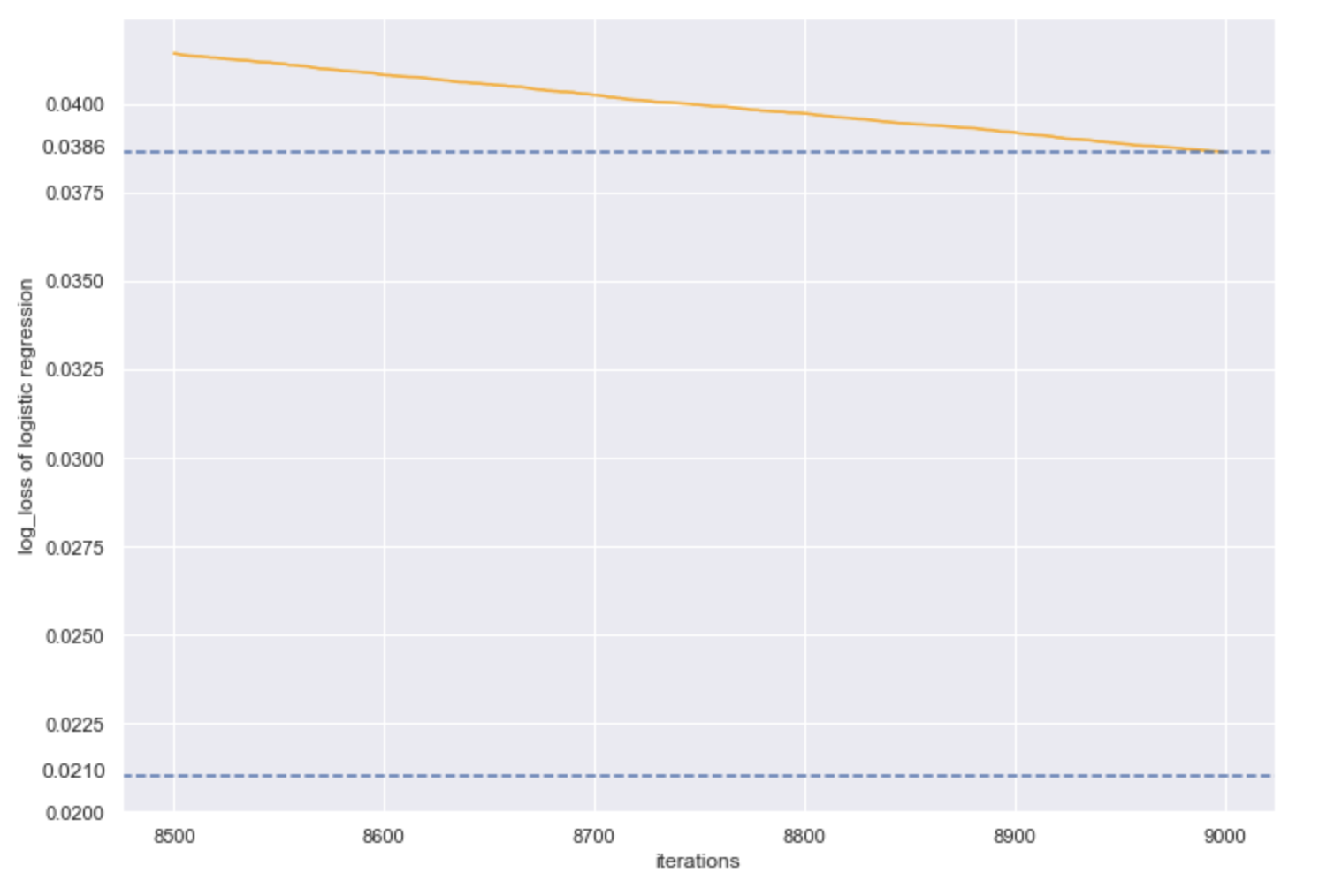


Figure 2. (Last 500 iterations)



1. **Critical evaluation**
2. In my Coordinate Descent algorithm, I choose the index of coordinate which maximizes the first derivative of function. I also tried to use the method of choosing the index of coordinate randomly. It turns out that the difference between these two methods is not such big like I imagined before. In the future, I may try to use other smarter method to select the coordinate index so that can approach to more quickly and better.
3. Through my experiment, the “initial starting point” and the original data influence my algorithm a lot. If I use a vector containing all zeros as my initial weight, then it turns out that my final outcome is pretty bad. Moreover, I use MinMaxScaler() function to standardize the original data so that can approach to better.
4. For every coordinate-dimension update, my step size is constant. Intuitively, the update step size will decrease as iteration goes on. I have read a paper about *Adam Optimization Algorithm* before and it inspired me that my future improvements could include updating step size in real time.