Kadir Pehlivan

IE582 Homework 5

In this homework, I tried to develop my unsuccessful approach from the final exam. First, I checked my loss and gradient functions in python to whether they are correct or not. I realized that I had a mistake in the gradient function. After correcting it, I searched for an optimization tool that I can use. I used scipy.optimize.minimize to minimize my loss function. After several attempt I realize that I need to scale the data to achieve a better result. Then I applied standard scaling and continued with scaled data. There were couple of important parameters that I need to decide. First one was the k parameter of the sigmoid function. In theory it should be infinite to make sigmoid function a step function, but it is not possible in computer. Therefore, I set it to 40 after couple of trials. 40 was small enough for the computer to compute precisely and big enough to make the sigmoid function a good step function. Second parameter to decide was the initial point. The obvious initial point was 0 but when I tried it, I saw that the point 0 was a local optimum and the BFGS algorithm converges to the same point in one step. I cannot figure out if it is a local minimum or maximum or a saddle point. Then I create 10000 random points and find the one that gives the minimum loss value and use it as an initial point, but it did not yield a good result. Lastly, I find a solution that maximizes the accuracy of the estimation using logistic regression and use the solution of it as an initial point of my algorithm. The last parameter to decide was the mu parameter of the regularization. Before finding a good value to it, I set it to 0 and run the algorithm. I tried different solvers such as L-BFGS-B, TNC and BFGS. TNC and L-BFGS-B algorithms fail to converge but they both give good results at where they stop. Only BFGS method converged successfully. Even though, L-BFGS-B algorithm gives a slightly smaller loss value, I choose the result of BFGS method since the difference was not too large and it converges successfully and gives me confidence. I do not include regularization because when I tried to give positive regularization parameter (i.e., mu), all methods failed to converge. In conclusion, I calculate the solution that minimize the loss function and estimate the missing values in the test data using the solution.

There are two thigs that I think can improve this solution, but I cannot achieve. First, finding a good initial point for this problem is significantly hard. Since the problem is in 60 dimensions it is almost impossible to visualize it and it is also hard to understand the nature of the loss function and guess a good starting point. It seems that there are countless local minima of the loss function. The only way to come up with a good initial point is to finding a solution from a different problem (For example, maximizing balanced accuracy for logistic regression problem), and using it as an initial point of the algorithm. I could try different methods to find different initial points and evaluate them separately.

Second, the training data was too small. I think it is not big enough for such a complicated loss function. With a larger training dataset, a better solution can be found. Using a stochastic gradient descent method could be beneficial for a larger training data.

Ps: The relevant code and the estimation of the test data can be found in my GitHub respiratory.