ENGR 421 - Machine Learning
Mehmet Üstek
HW3 Report

Completeness of the project:
All requirements are done with correct outputs regarding the homework description.

In the process of doing this project, I used class and lab materials, specifically based on lab03 and lab04.

First, I assigned the given parameters of Gaussian density. Then, I plotted the points as we did in hw1. Then using the lab04 materials, I created my gradient descent functions with learning parameters, eta and epsilon. I generated W and w0 weights to be very low, since initially we want these parameters to be slowly affected by the change. Then, again using the lab 04, I created my while loop where we learn W and w0 using gradient descent.

The difference from lab04 as stated in the homework description is to change softmax function to k-sigmoid function. Thus, using the lecture notes and lab03 material, I used a sigmoid function to predict y.

The y values are in the form of [a b 1-(a+b)], with a and b are the probabilities of the first two classes, and the 1-(a+b) function is the probability of the reference class K. Thus, the predicted y value matrix consists of 3 predictions for each data point. In our case we have a total of 300 data points, and 3 predictions for each. Thus yielding the Y_predicted shape of (300,3).

The second difference from lab04 was the change of error function from negative log-likelihood to the sum of squared errors. Thus, now we are going to minimize the error of this function. I changed the objective_values variable to accept this change.

The gradient descent functions change from derivative of negative log-likelihood function below:

$$\frac{\partial \text{Error}}{\partial \boldsymbol{w}_c} = -\sum_{i=1}^{N} (y_{ic} - \widehat{y}_{ic})\boldsymbol{x}_i$$

$$\frac{\partial \text{Error}}{\partial w_{c0}} = -\sum_{i=1}^{N} (y_{ic} - \widehat{y}_{ic})$$

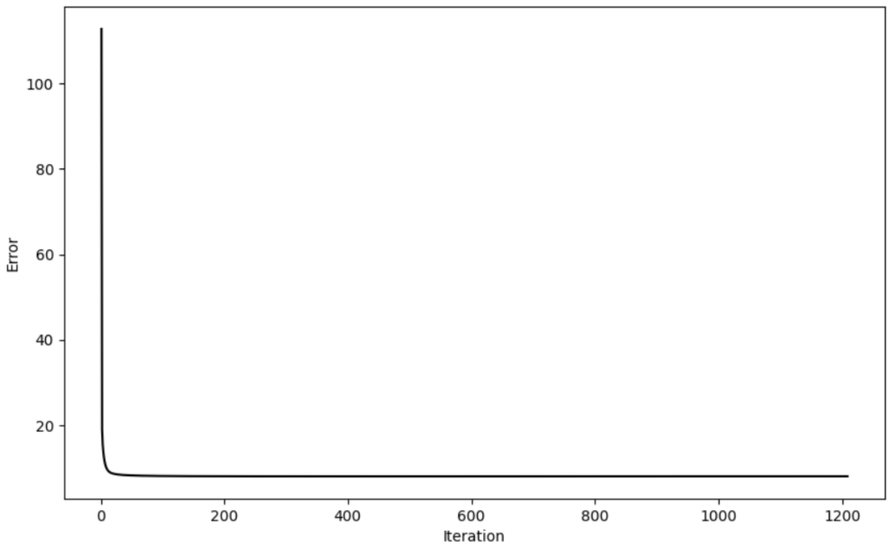
To the following scheme:

Y_predicted = sigmoid function
Derivative of sigmoid(w) = sigmoid(w) * (1-sigmoid(w))
Thus the derivative of y_predicted = (y_predicted) * (1- y_predicted)

To get gradient descent of w:
$(W^T x + w_0)$ / d(W)
By chain rule:

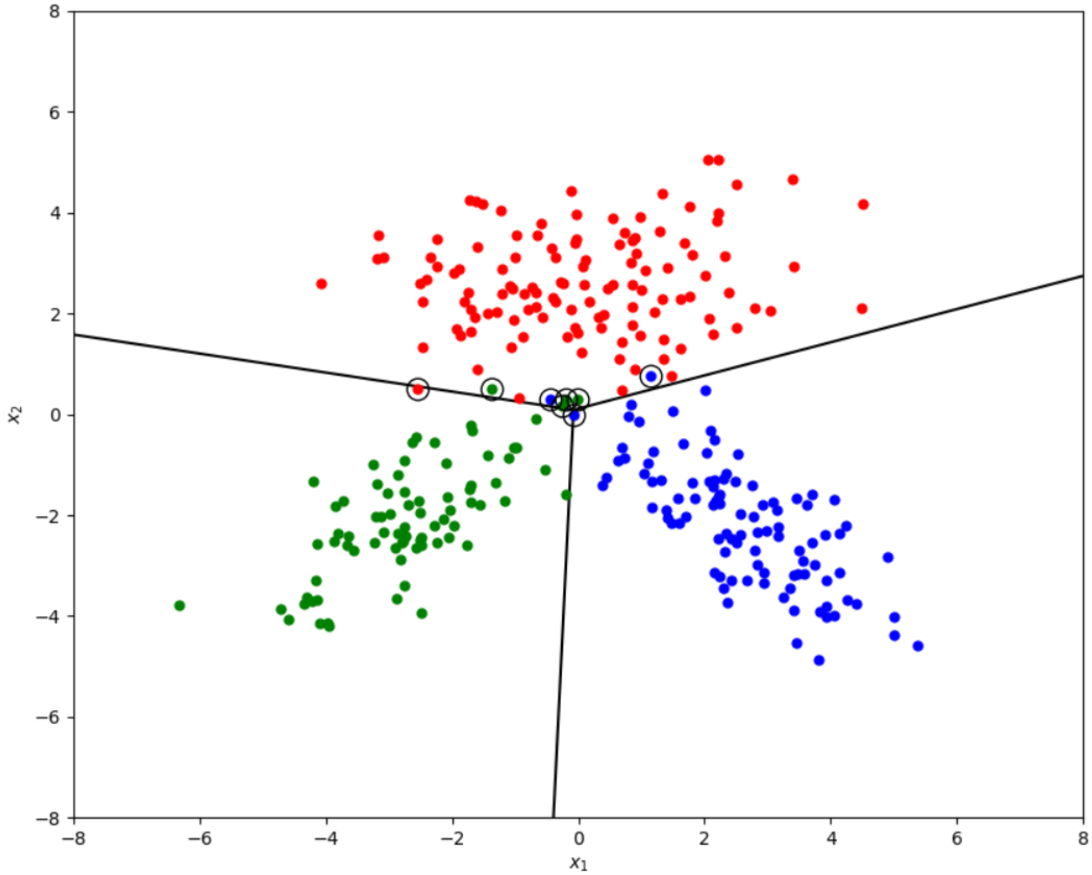
Error /dW = [Error / d(Y_predicted)] * [d(Y_predicted) / d(W)]
Error function is 0.5 ∑∑ (y_truth - y_predicted)^2
Derivative with respect to W:
[0.5 * 2 * (y_truth - y_predicted) * - y_predicted * (1 - y_predicted)] * X
Derivative with respect to w0:
0.5 * 2 * (y_truth - y_predicted) * - y_predicted * (1 - y_predicted)

Moreover, I followed the steps of lab04 to plot the data and display the error-iteration graph, which yield the result with random seed 421 as follows:



This graph was expected, as it starts with a quite high error value and then suddenly drops to more moderate values. The starting error value is 112, and then it drops monotonically and drastically to 19, 13 and then the values that are below 10, where it slows down its drastic drop This is exactly what is expected from gradient descent. It should drop drastically when the error is larger and slowly when the error is smaller. It stops when a local minimum is reached, or namely when gradient descent weights W and w0 does not change as in the bound of epsilon.

Finally, I have displayed the confusion matrix and plotted the decision boundaries using discriminant values for these 3 classes. It is seen below that the graph yielded the correct results by discriminating between the incorrect predictions. I used the exact same code from lab 4 for plotting these graphs.



Overall, the homework was easy to understand and implement using the lab and lecture materials. I had the chance of understanding the concepts of gradient descent and sigmoid functions in the process of making this project.
Note: The data and graphs indicated above were given with random seed 421.

Acknowledgements:
I understand the university rules for plagiarism and I have never shared or used any code or slice of code while doing this project. Thus, the effort belongs only to me.
Additionally, I push all my progress to a private github repository, in case need of any proof to display my effort.