**Homework-3 Report**

**Berke Can Rizai – 69282**

We start with imports and initialization of the class means, deviations and sizes. These are same as the HW-1.

Text, application

Description automatically generated

Y is the y\_truth that holds labels and we plot these points with plt.plot, we have following output.

Chart, scatter chart

Description automatically generated

Our learning rate eta and error tolerance epsilon are 0.01 and 0.001 respectively, with;

Graphical user interface, text

Description automatically generated

K is the number of classes. I could also write 3 instead of K everywhere, but it doesn’t really matter since it already requires some manual work such as initialization etc.

Text

Description automatically generated with low confidence

Here, I added all the points to a single list so that it looks like [x10, x11, x12,.. x20, x21…x2D, … x30, x31, ...] where each point holds two features.

x is initialized to this allP and y\_truth is the y we have declared earlier.

In the gradients, we take the derivative of the

Where y\_pred is the sigmoid result and yic is the y or y\_truth. We change y\_hat with sigmoid and take gradient, as follows:

Text, letter

Description automatically generated

This was gradient\_w, to calculate it we just take the derivative with chain rule. Since inside the sigmoid, we have (wx + w0) and when we take derivative with respect to w we get the x.

For the gradient\_w0, we again do the same iterations however, since we take derivative with respect to w0, wx is a constant and it disappears from the result of gradient.

Than, gradient\_w is 

Which is (y\_truth - y\_predicted) \* y\_predicted \* (y\_predicted -1) \* x. I take the transpose so that dimensions match during next operations. This is done with .T.

Gradient\_w0 is the following;



The addDim function I have used in gradient is basically copying an array and adding it to itself column wise. I have made this to match dimensions on the gradient function.

That is similar to the gradient\_w but, this doesn’t have x and I also simplified it a bit. We take the np.sum since we have sigma in original error function that means summing up.

I initialized Y\_truth, y\_truth here, same as the labs.

Graphical user interface, application, Word

Description automatically generated

Then, we minimize our error by iterating and updating the w and w0 with the gradients. When change is minimal, we stop iterations.

We use the gradients we have derived earlier and with the sigmoid function, we make all the predictions in a single operation.

Text

Description automatically generated

Here, objective\_values is a list. W\_old and w0\_old are stored separately in order to be able to check change in them later.

In each iteration, we update w and w0 with the step size (eta) \* gradient. The np.sqrt is our change.

When the change is lower than the epsilon, we stop the iterations. Then print these. So, our printed w and w0 are,



Chart

Description automatically generated with medium confidence

Here, we can see the change in the error by iterations.

Confusion matrix is as follows; we have 8 errors at total and rest are correctly classified.

Graphical user interface, application, Word

Description automatically generated

Lastly, I created a matplotlib figure that is 10x10. Added all the points to X and created a mesh grid with the x1 and x2 intervals that have 1201x1201 points. Text, letter

Description automatically generated

Then, calculate the discriminant values with the w and w0 we have found. 

Plot the points on the canvas. Chart

Description automatically generated

To mark the wrong predictions, I check all the points and for each of them, first look at my prediction from “a” array and actual label from the y array. Compare them and draw circle around them if they are not the same. Graphical user interface, text, application, email

Description automatically generated

Then, draw the discriminant lines and erase the parts of lines that are not crucial for understanding. For example, the line between green and blue will go over the red however, we don’t need it to be present as the red “dominates” that part. Finally, show the resulting graph.

Text, letter

Description automatically generated

Chart, scatter chart

Description automatically generated