Hard Activation Function

A Comparison:

Given the large separation between the two categories our perceptron was able to correctly categorize each point based on the error value after each iteration. Both the training and testing datasets had 100% accuracy and a total error value of 0.00000.

B Comparison:

The accuracy for both test sets was radically different, the second group had a near 95% accuracy compared to the first group’s 71%. The total error values were similar, 8:6, but the interesting point for me would be the classification of the falseFemale datapoints. For the first group all female points were correctly classified while the males were split, however in the second group while the classification of the males got much better, we also ended up with two females falsely classified. This could be due to additional test values or simply a different decision line that ended up cutting into the female category.

C Comparison:

The hard activation function for dataset C did very poorly, each one completely failing to correctly classify the males in the dataset. That said both groups did completely classify the females correctly, so the accuracy for each group was roughly 47%, given that there were less females in the overall dataset.

Soft Activation Function

A Comparison:

Our soft activation function had a similar result to our hard activation function for A, in that both perfectly separated and correctly categorized the dataset. The only real difference between the two was the total error, which was higher for our first dataset.

B Comparison:

The soft activation function for the B dataset was very good, both having accuracies in the high 90’s with the total error being much smaller in the second group. The errors can easily be explained by the overlap between the two categories as any decision line we setup will not be perfect. Still this is an excellent example of our perceptron training in action.

C Comparison:

The results for the soft activation function for dataset C were almost identical to the hard activation function of C, the only difference being the total error values for each group were higher. I understand that would be that given how much overlap there were between the two categories, there was an upward limit to the accuracy we could achieve for our perceptron training models.

I would generally want to use a smaller portion of the overall dataset for training the perceptron, so we can avoid overfitting our model and so we can have a better set of judging criteria to point out errors and statistical outliers. The exception for that would be if there are many classes to be labelled or if I need enough training data to cover every use case. While adding more training data will generally improve the model, it doesn’t mean the resulting model will be exponentially more accurate than if you used less training data.