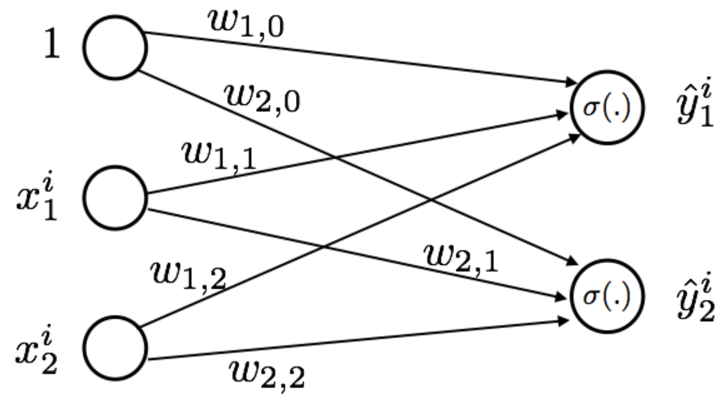
CS/ECE/ME532 Activity 24

Estimated Time: 25 minutes for P1, 25 minutes for P2

1. A script is available to train two neurons using stochastic gradient descent to solve two different classification problems. The two classifier structures are shown below. Here we use a logistic activation function $\sigma(z) = (1 + e^{-z})^{-1}$. The code generates



training data and labels corresponding to two decision boundaries: $x_2^i = -2x_1^i + 0.2$, and $x_2^i = 5(x_1^i)^3$.

a) Do you expect that a single neuron will be able to accurately classify data from case 1? Why or why not? Explain the impact of the bias term associated with $w_{1,0}$. Yes, one neuron will likely approximate the linear boundary well. The bias term allows for movement of the boundary instead of being stuck at (0,0).

b) Do you expect that a single neuron will be able to accurately classify data from case 2? Why or why not? Explain the impact of the bias term associated with $w_{2,0}$. No, more neurons are needed so that the boundary can be polynomial instead of just linear.

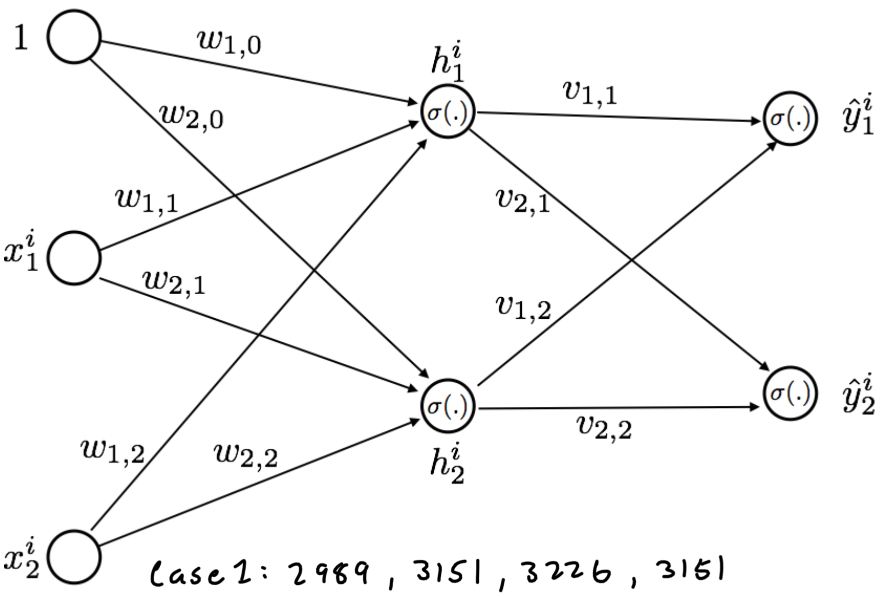
c) Run SGD for one epoch. This means you cycle through all the training data one time, in random order. Repeat this five times and find the average number of errors in cases 1 and 2. Case 1: 144 + 183 + 175 + 143 + 150 = 795 / 5 = $\underline{159}$
Case 2: 780 + 719 + 735 + 742 + 739 = 3715 / 5 = $\underline{743}$

d) Run SGD over twenty epochs. This means you cycle through all the training data twenty times, in random order. Repeat this five times and find the average number of errors in cases 1 and 2. Case 1: 31 + 31 + 29 + 26 + 25 = 142 / 5 = $\underline{28.4}$
Case 2: 730 + 750 + 740 + 740 + 764 = 3724/5 = $\underline{744.8}$

e) Explain the differences in classification performance for the two cases that result with both one and twenty epochs. From the graph of predicted labels we can see that the gradient area is much narrower for 20 epoches, so more data is classified correctly. This likely results from each epoch run being able to classify more data correctly, so more data is being taken from the middle area and placed on the correct side. Each epoch finetunes the weights to make them more specific and accurate.

For case 2, since there aren't even enough neurons to represent the boundary, there will always be around the same amount of errors no matter where the boundary lies or how it looks.

2. This remainder of this activity uses a three-layer neural network with three input nodes and two output nodes to solve two classification problems. We will vary the number of hidden nodes. The figure below depicts the structure when there are two hidden nodes.



Case 1: 2989 , 3151 , 3226 , 3151
Case 2: 723 , 722 , 740 , 721

A second script is available that generates training data and trains the network using SGD assuming a logistic activation function $\sigma(z) = (1 + e^{-z})^{-1}$.

The performance is about the same, around 750 errors, as case 2 from the problem above

a) Use $M = 2$ hidden nodes and ten epochs in SGD. Run this four or five times and comment on the performance of the two classifiers and whether it varies from run to run. The performance is very bad on case 1 with about 3000 errors per run. While the data forms a circle in the training data graph, the predicted values form two linear boundories.

b) Repeat $M = 2$ but use 100 epochs in SGD. (You may use fewer epochs if it takes more than a minute or two per run.) Run this several times and comment on the performance of the classifiers and whether it varies from run to run.
Performance is very similar to part a), but case 1 has around 2700 errors instead.

c) Recall the two-layer network results from the previous problem. How do the possible decision boundaries change when you add a hidden layer?
hidden layers add more terms to the decision boundary, enabling non linear boundaries.

d) Now use $M = 3$ hidden nodes and run 100 epochs of SGD (or as many as you can compute). Does going from two to three hidden nodes affect classifier performance? This change greatly decreased case 1 errors to around 250 and decreased case 2 errors to around 500. From the predicted value graphs we can also see how the classifier is starting to become nonlinear and follow the curves of the actual boundary.

e) Repeat the previous part for $M = 4$ hidden nodes and comment on classifier performance. Now case 1 and case 2 errors are very similar with both ranging between 50 and 200 errors each run. Case 2 errors are usually less than Case 1 now when in part d) they were more
From the predicted value graphs, they show a very similar shape to the training data graphs now. However, variation between each run is also much greater, with over 100 errors difference between runs.