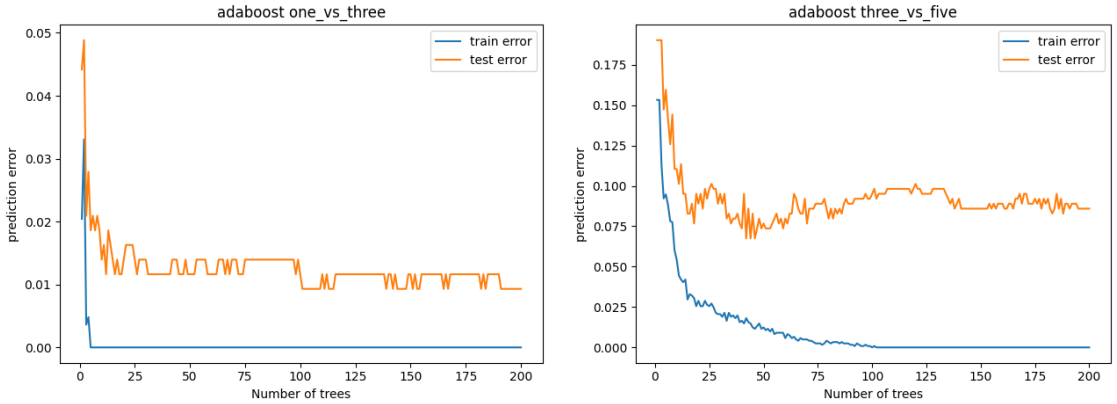
CSE 417T: Homework 5 Solution Sketch

April 1, 2022

Note: These are not intended to be comprehensive, just to help you see what the answers should be.

1. Below are two figures to demonstrate the "shape" of the curves. The idea is that both the train error and test error keep decreasing when the number of weak hypotheses increase. No overfitting is observed.



2. The three closest points are $(3,5),(3,8),(2,11)$. For the 3-NN average, we would predict the average y values of these three points, which would give us 8.

3. The mapping is easy to compute.

   - $[-1,-1]$ maps to $[-1,+1]$;
   - $[-1,+1]$ maps to $[-1,-1]$;
   - $[+1,-1]$ maps to $[+1,-1]$;
   - $[+1,+1]$ maps to $[+1,+1]$.

   The maximal margin separator in the new space is the line $x_1x_2 = 0$, with a margin of 1. In the original space, this is equivalent to $x_1 = 0$ and $x_2 = 0$, which you can think of as the limit of a hyperbolic separator with two branches.

4. The squared Euclidean distance is the dot product of $\Phi(\mathbf{x}_i) - \Phi(\mathbf{x}_j)$ with itself. This can be written as $\Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}_i) + \Phi(\mathbf{x}_j)\cdot\Phi(\mathbf{x}_j) - 2\Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}_j)$, which is just $K(\mathbf{x}_i,\mathbf{x}_i) + K(\mathbf{x}_j,\mathbf{x}_j) - 2K(\mathbf{x}_i,\mathbf{x}_j)$.

5. Adopt the notations from LFD: AND$(x,y) = xy$, OR$(x,y) = x + y$, and NOT$(x) = \bar{x}$. Recall that XOR$(x,y) = \bar{x}y + x\bar{y}$. Apply the boolean algebra, we have

$$\begin{aligned}\text{XOR}(\text{AND}(x_1,x_2),x_3) &= \overline{(x_1x_2)}x_3 + x_1x_2\bar{x_3}\\ &= (\bar{x_1} + \bar{x_2})x_3 + x_1x_2\bar{x_3}\\ &= \bar{x_1}x_3 + \bar{x_2}x_3 + x_1x_2\bar{x_3}\end{aligned}$$

In the resulting neural network, the hidden layer consists of $\bar{x_1}x_3$, $\bar{x_2}x_3$, and $x_1x_2\bar{x_3}$.

Note that this is not the only correct solution. One way to assess correctness (and something you can encourage students to do in order to double check their work) is to run all possible Boolean inputs through their network and make sure each is returning the correct result:

x_1	x_2	x_3	XOR(AND(x_1, x_2), x_3)
-1	-1	-1	-1
+1	-1	-1	-1
-1	+1	-1	-1
-1	-1	+1	+1
+1	+1	-1	+1
+1	-1	+1	+1
-1	+1	+1	+1
+1	+1	+1	-1

6. a The in-sample squared error of the sigmoidal perceptron is:

$$E_{in}(\vec{w}) = \frac{1}{n}\sum_{i=1}^{n}(\tanh(\vec{w}^T\vec{x}_i) - y_i)^2$$

To get the desired result, use the chain rule for partial derivatives and the fact that $^{d\tanh(x)}/_{dx} = 1 - \tanh^2(x)$:

$$\begin{aligned}
\nabla E_{in}(\vec{w}) &= \frac{\partial}{\partial \vec{w}}\left(\frac{1}{n}\sum_{i=1}^{n}(\tanh(\vec{w}^T\vec{x}_i) - y_i)^2\right) \\
&= \frac{1}{n}\sum_{i=1}^{n}\frac{\partial}{\partial \vec{w}}(\tanh(\vec{w}^T\vec{x}_i) - y_i)^2 \\
&= \frac{2}{n}\sum_{i=1}^{n}(\tanh(\vec{w}^T\vec{x}_i) - y_i)\frac{\partial}{\partial \vec{w}}(\tanh(\vec{w}^T\vec{x}_i) - y_i) \\
&= \frac{2}{n}\sum_{i=1}^{n}(\tanh(\vec{w}^T\vec{x}_i) - y_i)(1 - \tanh^2(\vec{w}^T\vec{x}_i))\frac{\partial}{\partial \vec{w}}(\vec{w}^T\vec{x}_i) \\
&= \frac{2}{n}\sum_{i=1}^{n}(\tanh(\vec{w}^T\vec{x}_i) - y_i)(1 - \tanh^2(\vec{w}^T\vec{x}_i))\vec{x}_i
\end{aligned}$$

When $\vec{w}$ is large, $\tanh(\vec{w}^T\vec{x}_i) \approx 1$, which means that the term $(1 - \tanh^2(\vec{w}^T\vec{x}_i))$ will be close to 0 and the entire gradient will be very, very small. Thus, it is not a good idea to initialize the weights to be very, very large because the gradient descent algorithm will take small steps, causing the algorithm to converge slowly.

b If all the weights are 0, then the gradients will be 0 as well. Therefore, it is a bad idea to initialize the weights to be all zeros when optimizing the weights using (stochastic) gradient descent as the weights will never change from iteration to iteration.

To see why the gradients will be 0, check the equations (7.4) and (7.5) in the textbook. In particular, following the lecture note, remember that

$$\frac{\partial e_n(W)}{\partial w_{i,j}^{(\ell)}} = \delta_j^{(\ell)} x_i^{(\ell-1)}$$

$$\delta_j^{(\ell)} = \sum_{k=1}^{d^{(\ell+1)}} \delta_k^{(\ell+1)} w_{j,k}^{(\ell+1)} \theta'(s_j^{(\ell)})$$

Since all weights are 0, all $\delta_j^{(\ell)}$ are 0, and therefore all partial derivative $\frac{\partial e_n(W)}{\partial w_{i,j}^{(\ell)}}$ are 0. This means all gradient are 0.