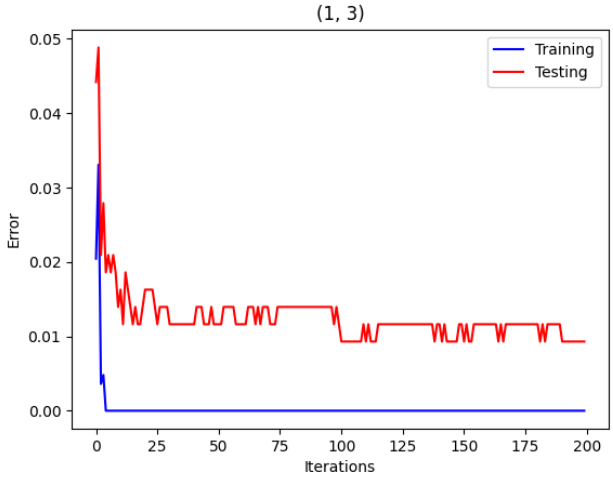
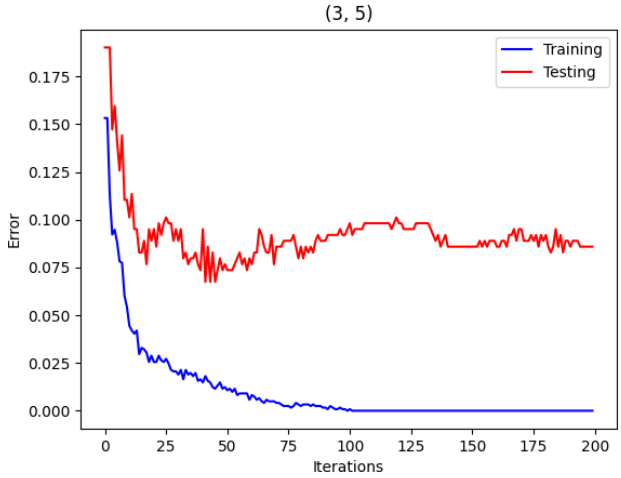




Both testing errors decreased with the addition of more weak learners. The (1, 3) problem arrived at 0 testing error very quickly (after about 5 iterations). Even after this (at 100 iterations), the testing error decreased a little bit more to 0.01. In this problem, AdaBoost did not overfit the problem and consistently kept low testing error. The training error in the (3, 5) problem consistently decreased until hitting 0 at about 100 iterations. This makes sense as the complexity of the (3, 5) problem is higher than (1, 3). The testing error in (3, 5) reached a minimum at about 50 iterations with a value of 0.075. After this, the testing error gradually increased to about 0.1 at 125 iterations indicating AdaBoost started to overfit the data at around 50 iterations. This overfitting did not occur in (1, 3) likely due to the simpler nature of the problem.

2. $X = 3.2$

closest are $(3,5)$; $(2,11)$; $(3,8)$

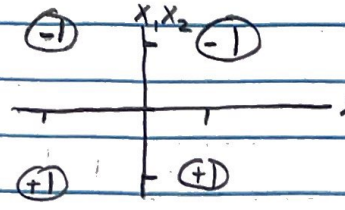
$$g(\vec{x}) = \frac{1}{k}\sum_{i=1}^{k} y_{[i]}(\vec{x})$$
$$= \frac{1}{3}(5+11+8) = \boxed{8}$$

3.

x_1	x_2	XOR
-1	-1	-1
+1	-1	+1
-1	+1	+1
+1	+1	-1

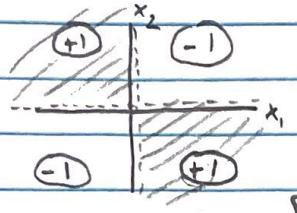
MAP ⟹

x_1	x_1x_2	XOR
-1	+1	-1
+1	-1	+1
-1	-1	+1
+1	+1	-1



- Max margin is at $x_1x_2 = 0$.
- Margin is 1.

Transform space



- Seperator is at $x_1 = 0$, $x_2 = 0$.
- kind of like hyperbola.

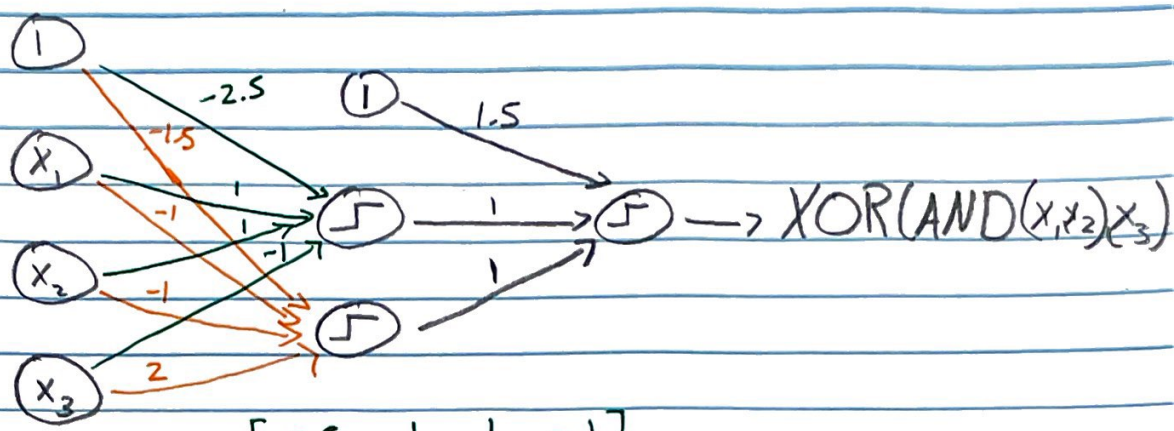
original space

4.
$$dist = \|\phi(\vec{x}_i) - \phi(\vec{x}_j)\|^2$$
$$= \phi(\vec{x}_i)^2 - 2\phi(\vec{x}_i)^T\phi(\vec{x}_j) + \phi(\vec{x}_j)^2$$
$$= \boxed{K(\vec{x}_i, \vec{x}_i) - 2K(\vec{x}_i, \vec{x}_j) + K(\vec{x}_j, \vec{x}_j)}$$

5. $XOR(AND(x_1, x_2), x_3)$

make $AND(x_1, x_2)$ be x_{12}

$XOR(x_1, x_2, x_3) \rightarrow x_1 x_2 \bar{x}_3 + \overline{x_1 x_2} x_3$



$w_1 = [-2.5, 1, 1, -1]$

$w_2 = [-1.5, -1, -1, 2]$

$w_3 = [1.5, 1, 1]$

6a. $E_{in}(\vec{w}) = \frac{1}{N}\sum_{n=1}^{N}\left(\tanh(\vec{w}^T\vec{x}_n) - y_n\right)^2$

$\nabla E_{in}(\vec{w}) = \frac{1}{N}\sum_{n=1}^{N} 2\left(\tanh(\vec{w}^T\vec{x}_n) - y_n\right)\left(\frac{\partial}{\partial w}\left(\tanh(\vec{w}^T x_n) - y_n\right)\right)$

derivative of $\tanh(x)$ is $1-\tanh^2(x)$

$$\boxed{\nabla E_{in}(\vec{w}) = \frac{2}{N}\sum_{n=1}^{N}\left(\tanh(\vec{w}^T\vec{x}_n) - y_n\right)\left(1-\tanh^2(\vec{w}^T x_n)\right)\vec{x}_n}$$

If $\vec{w}$ goes to ∞, then tanh will basically be 1 always, and there will be basically no gradient in any direction. This will make it difficult to improve weights.

6b. If all the weights are 0, then $\tanh(0) = 0$ for the output of all layers. Then, all layers except the input are 0 and the $\vec{x}_n$ in the gradient update will set the gradient to 0!! Thus there will never be a direction for us to update since the grad will always be 0.