Homework 2

Michael Parker

Oct 19, 2021

1 Paper Problems [40 points + 8 bonus]

1. (a)

   i. We would prefer L_2 because it has a simpler (smaller) hypothesis space which still contains a consistent hypothesis.

   ii. Since $|H_1| > |H_2|$, we know $log(|H_1|) > log(|H_2|)$, so holding ϵ and δ constant, the smaller hypothesis space lets us use a smaller sample size - or to put it another way, we get our guaranteed performance sooner using the smaller hypothesis space.

   (b) So we want:

$$m > \frac{1}{0.1}\big(\log(3^{10}) + \log\frac{1}{0.05}\big).$$

   Using log_2, gives $m > 201$ examples

2. Note that:

$$\sum_{i=1}^{m} D_t(i) = 1$$

and thus:

$$\sum_{i=1}^{m} D_t(i) = \sum_{y_i \neq h_t(x_i)} D_t(i) + \sum_{y_i = h_t(x_i)} D_t(i) \implies \sum_{y_i = h_t(x_i)} D_t(i) = 1 - \sum_{y_i \neq h_t(x_i)} D_t(i)$$

So,

$$\begin{aligned}\epsilon_t &= \frac{1}{2} - \frac{1}{2}\big(\sum_{i=1}^{m} D_t(i) y_i h_t(x_i)\big)\\ &= \frac{1}{2} - \frac{1}{2}\big(\sum_{y_i = h_t(x_i)} D_t(i) - \sum_{y_i \neq h_t(x_i)} D_t(i)\big)\end{aligned}$$

(Since $y_i = h_t(x_i) \implies y_i h_t(x_i) = 1$ and $y_i \neq h_t(x_i) \implies y_i h_t(x_i) = -1$)

$$= \frac{1}{2} - \frac{1}{2}\big(1 - \sum_{y_i \neq h_t(x_i)} D_t(i) - \sum_{y_i \neq h_t(x_i)} D_t(i)\big)$$

$$= \frac{1}{2} - \frac{1}{2}\big(1 - 2\sum_{y_i \neq h_t(x_i)} D_t(i)\big)$$

$$= \frac{1}{2} - \frac{1}{2} + \sum_{y_i \neq h_t(x_i)} D_t(i)$$

$$= \sum_{y_i \neq h_t(x_i)} D_t(i)$$

As desired.

3. (a) We want $x_1 + (1 - x_2) + (1 - x_3) \geq 3$, so $\mathbf{w} = [1, -1, -1]$, $\mathbf{b} = -0.5$, and the hyperplane is $x_1 - x_2 - x_3 - 0.5 = 0$

   (b) We want $(1 - x_1) + (1 - x_2) + (1 - x_3) \geq 1$, so $\mathbf{w} = [-1, -1, -1]$, $\mathbf{b} = 2.5$, and the hyperplane is $-x_1 - x_2 - x_3 + 2.5 = 0$

   (c) We can use the distributive property to show that:
   $(x_1 \vee x_2) \wedge (x_3 \vee x_4) = (x_1 \wedge x_3) \vee (x_1 \wedge x_4) \vee (x_2 \wedge x_3) \vee (x_2 \wedge x_4)$
   Which is equivalent to finding $x_1x_3 + x_1x_4 + x_2x_3 + x_2x_4 \geq 1$, which is clearly not linear but suggests new features $x_5 = x_1x_3$, $x_6 = x_1x_4$, $x_7 = x_2x_3$, $x_8 = x_2x_4$, which can be separated by the hyperplane $x_5 + x_6 + x_7 + x_8 - 0.5 = 0$

   (d) This is the negation of xor, which we showed in class there is no linear classifier. If we define a new feature $x_3 = (2x_1 - 1)(2x_2 - 1)$, then we can separate with the new hyperplane $x_3 = 0$

4. (a) $\phi(\mathbf{x}) = \mathbf{x}^T\mathbf{x}$ and $\phi(\mathbf{y}) = \mathbf{y}^T\mathbf{y}$

   (b) $\phi(\mathbf{x}) = (\mathbf{x}^T\mathbf{x}\mathbf{x}^T)^T$ and $\phi(\mathbf{y}) = \mathbf{y}\mathbf{y}^T\mathbf{y}$

   (c) for k even: $\phi(\mathbf{x}) = (\mathbf{x}^T\mathbf{x})^{k/2}$ and $\phi(\mathbf{y}) = (\mathbf{y}^T\mathbf{y})^{k/2}$
   for k odd: $\phi(\mathbf{x}) = ((\mathbf{x}^T\mathbf{x})^{(k-1)/2}\mathbf{x}^T)^T$ and $\phi(\mathbf{y}) = \mathbf{y}(\mathbf{y}^T\mathbf{y})^{(k-1)/2}$

5. (a) Our formula would be:

$$J(\mathbf{w}, b) = \frac{1}{2}\sum_{i=1}^{m}(y_i - (\mathbf{w}^T\mathbf{x}_i + b))^2$$

   (b)

$$\frac{\nabla J(\mathbf{w}, b)}{\nabla \mathbf{w}} = -\sum_{i=1}^{m}(y_i - (\mathbf{w}^T\mathbf{x}_i + b))\mathbf{x}_i$$

$$\frac{\nabla J(\mathbf{w}, b)}{\nabla b} = -\sum_{i=1}^{m}(y_i - (\mathbf{w}^T\mathbf{x}_i + b))$$

Filling in our points:

$$\frac{\nabla J([-1,1,-1]^T,-1)}{\nabla \mathbf{w}} = -\sum_{i=1}^{5}(y_i - [-1,1,-1]\mathbf{x}_i + 1)\mathbf{x}_i$$

$$= -(1 - [-1,1,-1][1,-1,2]^T + 1)[1,-1,2]^T$$
$$-(4 - [-1,1,-1][1,1,3]^T + 1)[1,1,3]^T$$
$$-(-1 - [-1,1,-1][-1,1,0]^T + 1)[-1,1,0]^T$$
$$-(-2 - [-1,1,-1][1,2,-4]^T + 1)[1,2,-4]^T$$
$$-(0 - [-1,1,-1][3,-1,-1]^T + 1)[3,-1,-1]^T$$
$$= -6[1,-1,2]^T - 8[1,1,3]^T + 2[-1,1,0]^T - 6[12,-4]^T - 4[3,-1,-1]^T$$
$$= [-34,-8,-8]^T$$

$$\frac{\nabla J([-1,1,-1]^T,-1)}{\nabla b} = -\sum_{i=1}^{5}(y_i - [-1,1,-1]\mathbf{x}_i + 1)$$
$$= -6 - 8 + 2 - 6 - 4 = -22$$

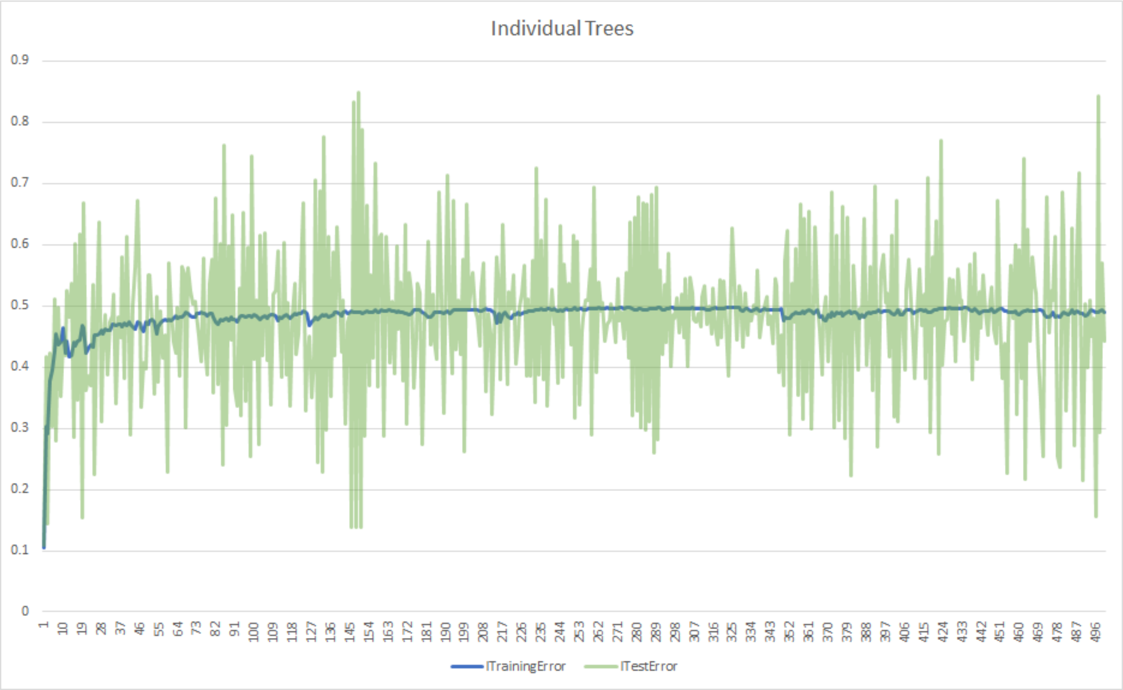
(c)

(d)

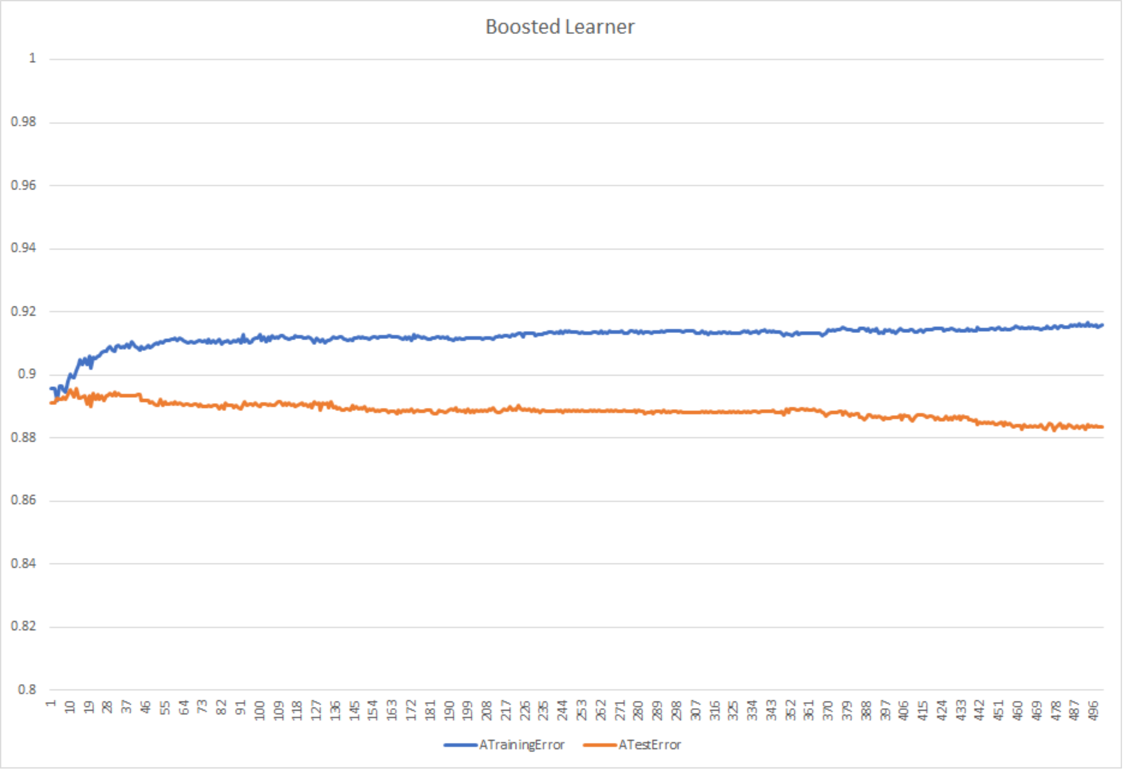
2 Practice [60 points + 10 bonus]

1. Done (branched HW2 off for clarity), https://github.com/FakerMike/MachineLearningLibrary/tree/F

2. (a) Individual learners:



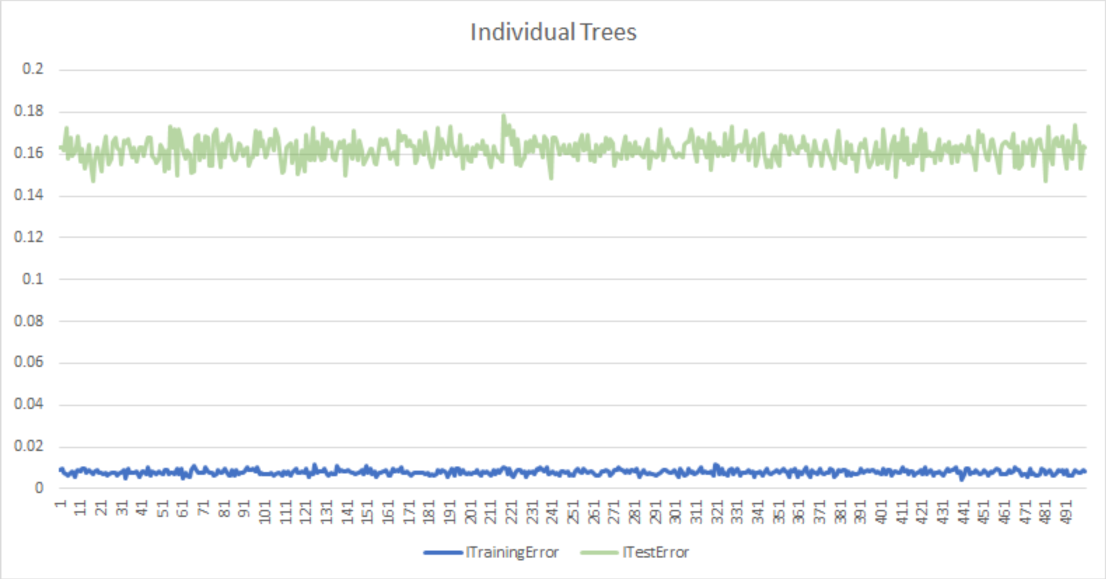
AdaBoost:



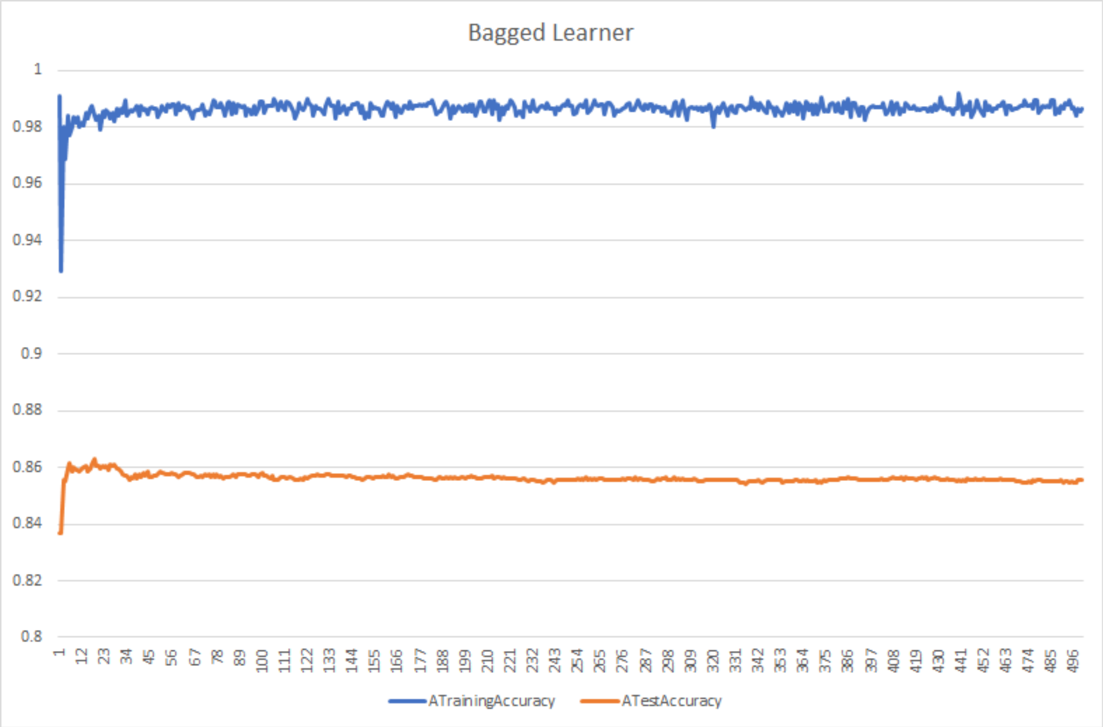
Conclusions: From HW1, the best performing solo tree got to 0.8926 test accuracy with depth 3 using Entropy for information gain. The AdaBoost algorithm got up to a max of 0.8956 test accuracy with 13 classsifiers, but managed to get in the 0.89 - 0.895 range for pretty much any number of classifiers from 4-100. Even when it started trending down towards the end it never fell below 0.8822 - for contrast, the same decision tree from HW1 fell below that at depth 4, just after it hit its peak. So AdaBoost can both reach the same levels of accuracy, as well as hold on to them a lot longer in the face of overfitting.

Also interesting to note is just how bad the individual classifiers eventually got - once we went past 100 classifiers, most of them had an error rate of about 0.49 (with 0.5 being the worst possible). So no surprise really that they didn't help much.

(b) Individual learners:



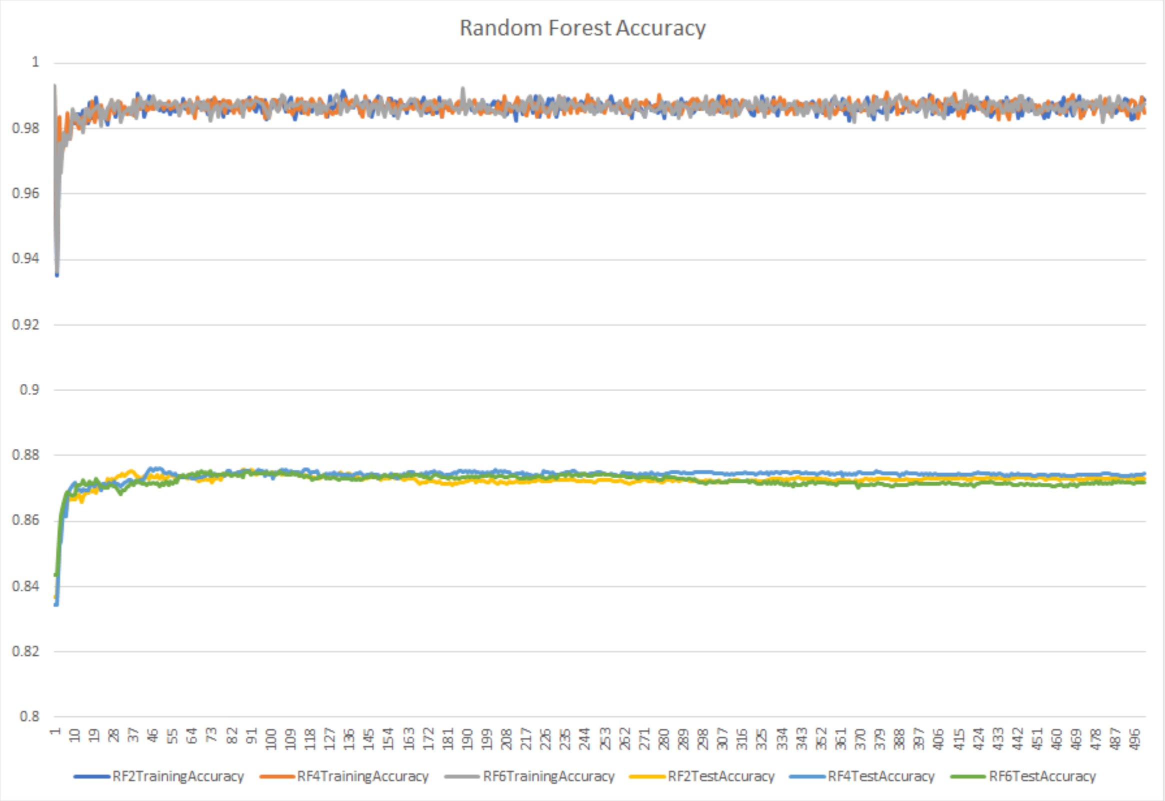
Bagged:



Conclusions: Test accuracy jumped up rapidly after only a few trees were added to the bag, then levelled out and never changed much. The bagged learner was much better than the boosted one on the training data, but performance on test data was significantly worse - it never got much higher than 0.86, while Adaboost never fell below 0.88. The bagged trees did perform better than a single tree (see the left of the graph), and managed to get rid of some of the damage done by overfitting the individual learners.

(c)

(d) Random Forests:



Conclusions: Random forests managed to beat the bagged trees, with a test accuracy of around .875 at the highest (done by the tree with 4 options per level). This means it still lost to the boosted trees from part a, but performance was more reliable for the random forest (though there was an optimal number of trees in the forest, about 60ish, it never fell much from there as the number increased). Once again it had very good training accuracy, but worse performance on the test data, suggesting it still overfit the model compared to the algorithm for Adaboost.

(e)

3.

4. (a)

(b)

(c)