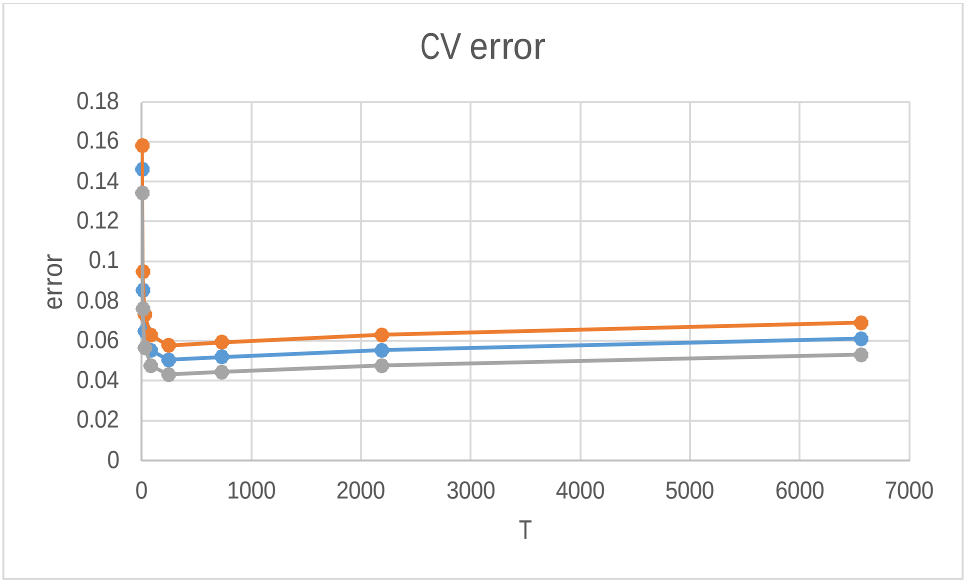
Paul Lintilhac

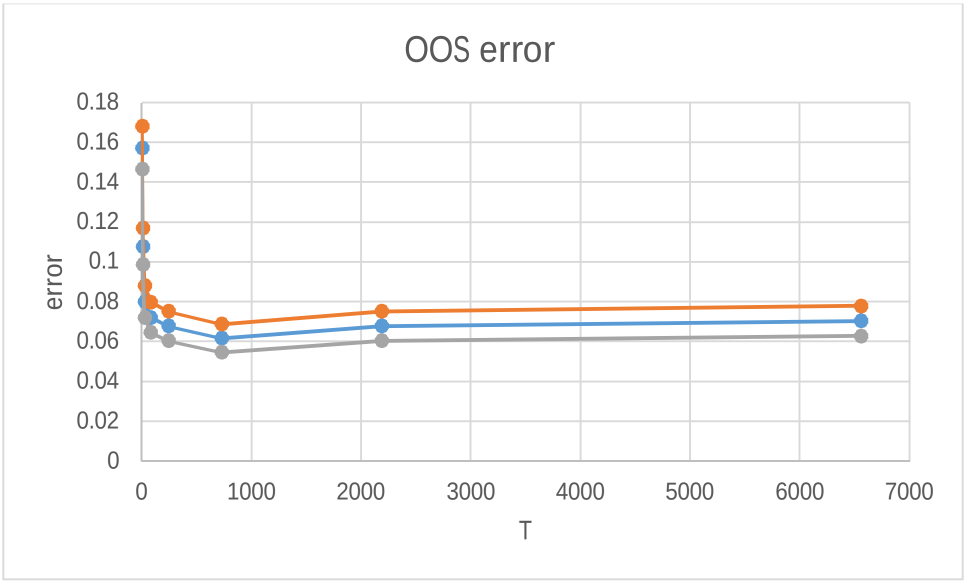
Homework 3 – Part A

Foundations Of Machine Learning

1) I did the problem without using any packages, and did a full implementation in both R and C++. The C++ is much faster to run of course, though the r code is easier to read. For the parameter T, I used powers of 3, i.e. $T_i = \left\{3^i : i \in \{1,2,3,4,5,6,7,8\}\right\}$. I used 4-fold cross-validation. Note that I calculated the error bars by first calculating the point-wise squared error: $\hat{\sigma}^2 = E[R(h)^2] - \hat{R}(h)^2 = \hat{R}(h) - \hat{R}(h)^2$, where I have used the fact that for a binary variable, $\hat{R}(h)^2 = \hat{R}(h)$. Next, since the observations are i.i.d., we can use the law of averages to compute the error over 3540/4 observations (for the cv error)by taking $\hat{\sigma}^2{}_{Tot} = 4 * \hat{\sigma}^2/3450$, or we can take the error over 1170 observations (for the OOS error) by taking $\hat{\sigma}^2{}_{Tot} = 4 * \hat{\sigma}^2/1151$.



Paul Lintilhac



Compared to the previous homework, I was able to achieve a lower error for both cross-validation and out-of-sample testing. The minimum error for cross-validation was 0.050435, achieved at T=243, while the minimum error for out-of-sample testing was 0.061686, achieved at T=729.

Paul Lintilhac

Problem 2)

a.

I will derive the modified boosting algorithm starting from the following assumptions:

1) Our empirical loss function is now $\widehat{R}(h) = \mathbf{1}_{g*y<0}$, i.e. if the guess was not correct but either the prediction or the response is 0 (not sure), then we don't count it towards the error.
2) We maintain the assumption that our final hypothesis is of the form $g = g_T = \sum_{t=1}^{T} \alpha_t h_t$.
3) We maintain the assumption that out distribution evolves as $D_{t+1}(i) = D_t(i) * \frac{e^{-\alpha_t y_i h_t(x_i)}}{Z_t}$

Note that the equations determining α_t and Z_t are not yet known.

Using these assumptions, we can see that the argument for the upper bound on the empirical error is exactly the same, and therefore leads to the same equation:

$$\widehat{R}(h) \leq \prod_{t=1}^{T} Z_t$$

Since Z_t is a normalization factor, we can again use the exact same identity

$$Z_t = \sum_{i=1}^{m} D_t(i)\, e^{-\alpha_t y_i h_t(x_i)}$$

However, this time we decompose the sum into 3 different sums:

$$Z_t = \sum_{i:\, y_i h_t(x_i) = -1}^{m} D_t(i)\, e^{-\alpha_t y_i h_t(x_i)} + \sum_{i:\, y_i h_t(x_i) = 0}^{m} D_t(i)\, e^{-\alpha_t y_i h_t(x_i)} + \sum_{i:\, y_i h_t(x_i) = 1}^{m} D_t(i)\, e^{-\alpha_t y_i h_t(x_i)}$$

$$= \epsilon_t^{-1} e^{\alpha_t} + \epsilon_t^{0} + \epsilon_t^{1} e^{-\alpha_t}$$

Note that $\epsilon_t^0 = 1 - \epsilon_t^{-1} - \epsilon_t^{1}$, and thus we can re-write this as

$$Z_t = \epsilon_t^{-1}(e^{\alpha_t} - 1) + \epsilon_t^{1}(e^{-\alpha_t} - 1)$$

Now, in order to minimize the empirical error, we can note that Z_t is convex and differentiable, and again minimize it with repect to α_t, which yields

$$\alpha_t = \frac{1}{2} \log\left(\frac{\epsilon_t^{-1}}{\epsilon_t^{1}}\right)$$

Paul Lintilhac

Now, substituting this back into the equation for the empirical error, we find that

$$\hat{R}(h) \leq \prod_{t=1}^{T} Z_t = \prod_{t=1}^{T} [\epsilon_t^{-1}(e^{\alpha_t} - 1) + \epsilon_t^{1}(e^{-\alpha_t} - 1) + 1]$$

$$= \prod_{t=1}^{T} \left[\epsilon_t^{-1}\left(\sqrt{\frac{\epsilon_t^{1}}{\epsilon_t^{-1}}} - 1\right) + \epsilon_t^{1}\left(\sqrt{\frac{\epsilon_t^{-1}}{\epsilon_t^{1}}} - 1\right) + 1\right]$$

$$= \prod_{t=1}^{T} \left[2\sqrt{\epsilon_t^{1}\epsilon_t^{-1}} - \epsilon_t^{1} - \epsilon_t^{1} + 1\right]$$

$$= \prod_{t=1}^{T} \left[1 - \left(\sqrt{\epsilon_t^{-1}} - \sqrt{\epsilon_t^{1}}\right)^2\right]$$

$$\leq \prod_{t=1}^{T} e^{-\left(\sqrt{\epsilon_t^{-1}} - \sqrt{\epsilon_t^{1}}\right)^2} \quad (1)$$

(the above answers d)

Paul Lintilhac

b.

We can see from this definition that rather than in the binary example, where the exponent is positive as long as $|\epsilon_t^{-1} - \frac{1}{2}| > 0$, in this case the exponent is positive as long as $|\epsilon_t^1 - \epsilon_t^{-1}| > 0$, i.e. as long as our rate of error is less than our rate of accuracy (excluding any points where $y_i h_t(x_i) = 0$). Thus, our weak learning assumption is now that there exists an algorithm A, $\gamma > 0$, and a polynomial function poly(.,.,.) such that for any $\delta > 0$ and any distribution D on X and for any target concept C, the following holds for any sample size $m \geq poly\left(\frac{1}{\delta}, n, size(C)\right)$:

$$P_{S\sim D^m}[R^{-1}(h) \leq R^1(h) + \gamma] \geq 1 - \delta$$

Where $R^1(h) = E\left[\mathbf{1}_{g*y=1}\right], and\ R^{-`1}(h) = E\left[\mathbf{1}_{g*y=-1}\right]$.

c.

The pseudocode for the algorithm is as follows:

```
For i =1 to m do:
$D_1(i) = \frac{1}{m}$
For t=1 to T do:
    $h_t <-$ base classifier with small *relative* error: $\epsilon_t^{-1} - \epsilon_t^1$
    $\alpha_t = \frac{1}{2}\log(\frac{\epsilon_t^1}{\epsilon_t^{-1}})$
    $Z_t = 2\sqrt{\epsilon_t^1 \epsilon_t^{-1}} + \epsilon_t^0$
    For i in 1 to m do:
        $D_{t+1}(i) = D_t(i)e^{-\alpha_t * h_t(x_i) * y_i} / Z_t$
$g = \sum_{t=1}^{T} \alpha_t h_t$
```