

Homework #3 Solutions

Professor: Eric Xing

Due Date: October 27, 2008

1 Learning Theory [20 Points, Mark]**1.1 VC-Dimension**

Consider the space of instances X corresponding to all points in the x, y plane. Give the VC dimension of the following hypothesis spaces, and include your proofs:

- (2pts) H_r = the set of all axis aligned rectangles in the x, y plane. Points inside of the target rectangle are classified as positive examples.

Solution: VC-dimension = 4

- (2pts) H_c = circles in the x, y plane. Points inside the circle are classified as positive examples.

Solution: VC-dimension = 3

- (2pts) H_t = triangles in the x, y plane. Points inside the triangle are classified as positive examples.

Solution: VC-dimension = 7

- (2pts) How many training examples suffice to assure with probability .95 that a consistent learner using H_r will learn the target function with accuracy of at least 0.90?

Solution:

The solution is given by:

$$examples \leq \frac{1}{\epsilon} (4 \log_2(2/\delta) + 8VC(H) \log_2(13/\epsilon))$$

For H_r , VC-Dimension is 4, so the number of examples is 2460.

- (2pts) What exactly does it mean in part (4) when we say the learner will succeed with probability 0.95? Answer this question by describing a simple experiment which you could run repeatedly, for which the success rate is expected to be at least 0.95.

Solution:

Choose a distribution $P(X)$ and target function f . Now repeatedly draw a training set $\{x_i; y_i\}$ of size 2460, based on $P(X)$ and f . For each training set, find a hypothesis h in H_r that perfectly classifies all 2460 training examples, and measure the true accuracy of h (i.e., the expected accuracy of h relative to f and $P(X)$). In at least 0.95 of these experiments (ie., with probability 0.95) the true accuracy of h will be at least 0.90.

1.2 Mistake Bounds

Suppose that we want to build a *ensemble* classifier by combining the predictions from K other classifiers. One simple algorithm is as follow:

1. Each of the K classifiers is assigned a weight w_k . At the start, each w_k is initialized to 1.
2. For a given test example, each classifier makes a prediction $y_k \in \{0, 1\}$. Our ensemble classifier makes its prediction according to the following rule: predict 1 if

$$\sum_{i=1}^K w_i I(y_i = 1) \geq \sum_{i=1}^K w_i I(y_i = 0)$$

and predict 0 otherwise. Remember $I(\cdot)$ is the indicator function that returns 1 if its predicate is true.

3. When the correct label l is given for the test example, penalize each incorrect classifier by dividing its weight by 2. For example, if $y_k \neq l$, then $w_k = w_k/2$.
4. Goto step 2

Consider the most accurate of the K individual classifiers. Suppose this classifier makes m_a mistakes and the ensemble classifier makes m_e mistakes. We would like to know how many mistakes our ensemble classifier will make as a function of the number of mistakes of the most accurate individual classifier.

It is possible to prove the following statement:

Theorem 1. *The number of mistakes m_e made by this ensemble classifier is never more than $2.40942(\log_2(K) + m_a)$*

We will now prove this theorem in a series of steps:

1. (2pts) Compute an upper bound on the total weight $W = \sum w_k$ left after m_e mistakes.

Solution:

For each mistake, at least half the weight is incorrect. And we cut this incorrect weight in half. So after m_e mistakes we have:

$$W \leq K \left(\frac{3}{4}\right)^{m_e}$$

2. (2pts) Compute a lower bound on the weight W by finding the weight of the best expert after it has made m_a mistakes.

Solution:

$$\left(\frac{1}{2}\right)^{m_a} \leq W$$

3. (6pts) Given the two bounds, solve for m_e in terms of m_a and K to complete the proof

Solution:

Using the upper and lower bound on W:

$$\begin{aligned} \left(\frac{1}{2}\right)^{m_a} &\leq K\left(\frac{3}{4}\right)^{m_e} \\ m_a \log_2\left(\frac{1}{2}\right) &\leq \log_2(K) + m_e \log_2\left(\frac{3}{4}\right) \\ -m_e \log_2\left(\frac{3}{4}\right) &\leq \log_2(K) + m_a \log_2(2) \\ m_e \log_2\left(\frac{4}{3}\right) &\leq \log_2(K) + m_a(1) \\ m_e &\leq \frac{\log_2(K) + m_a}{\log_2\left(\frac{4}{3}\right)} \\ m_e &\leq 2.40942(\log_2(K) + m_a) \end{aligned}$$

2 Information Theory (Suyash, 20 pts)

2.1 KL-divergence

1. Use Jensen's inequality on $-\text{KL}(p||q)$ to get $-\text{KL}(p||q) \leq 0$. Equality in Jensen's inequality requires $\frac{p_i}{q_i}$ be the same for all i . Since p and q sum to 1, this means $p_i = q_i, \forall i$. Alternative proof- use the property that $\ln x \leq x - 1$ with equality iff $x = 1$.

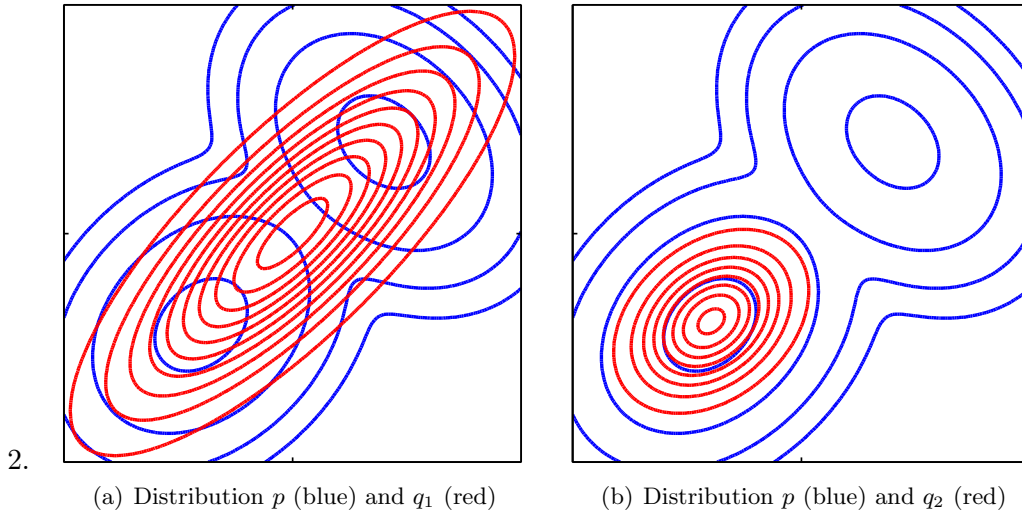


Figure 1: Single gaussian approximations q to a bimodal distribution p . The blue distribution is the bimodal p distribution, and the red distributions are single gaussian approximations q_1 and q_2

$\text{KL}(p||q)$ has high penalty if q is small in a region where p is high. So it is minimized by q distributions that are nonzero where p is nonzero. Fig 1(a) minimizes $\text{KL}(p||q)$.

$\text{KL}(q||p)$ has a large penalty if q is high in regions where p is small. So it is minimized by q distributions that are small where p is small. Fig 1(b) minimizes $\text{KL}(q||p)$.

- In general, the measure is not symmetric. A possible symmetric measure is $D(p||q) = \text{KL}(q||p) + \text{KL}(p||q)$

2.2 Feature selection method

- All X_i are Bernoulli(0.5). Y is Bernoulli(0.75). $X_3 \perp Y$ and $X_4 \perp Y$. Joint probability table for X_1, Y and X_2, Y is the same, given by So $\text{MI}(X_3, Y) = \text{MI}(X_4, Y) = 0$. $\text{MI}(X_1, Y) = \text{MI}(X_2, Y) = 0.5 -$

Table 1: default

X	Y	Prob.
0	0	0
0	1	0.5
1	0	0.25
1	1	0.25

$$0.75 * \log_2(0.75) \approx 0.91$$

- Set $\{X_1, X_2\}$ since the other two features give no information about Y .

2.3 Shannon's entropy

In this problem, we shall show the connection between Shannon's entropy and code lengths.

The entropy for a probability distribution $p = \{p_1, \dots, p_n\}$ is given by $H(p) = -\sum_i p_i \log p_i$, where the logarithm is taken to base 2. Suppose $S = \{s_1, \dots, s_n\}$ is a set of messages, and that $p = \{p_1, \dots, p_n\}$ describes the probability distribution over the messages. A code for S is given by a mapping from each message to a bit string, denoted using $C = \{(s_1, w_1), \dots, (s_n, w_n)\}$. We define the *average length* of a code C as $l_a(C) = \sum_i p(s_i)l(w_i)$. We shall prove that for a special class of codes called *prefix codes*, the entropy is a lower bound on the minimum possible average code length, i.e, $H(p) \leq l_a(C), \forall C$. This result is in fact true for all uniquely decodable codes.

2.3.1 Prefix codes

Each path from root to leaf is a codeword for a message. Append 0 to the message every time a left edge is traversed, and a 1 if a right edge is traversed.

2.3.2 Kraft's inequality

This can be proved using induction on maximum depth of subtrees. Let tree T have maximum depth d . Then subtrees U and V must have maximum depth $d-1$. In general, define $\text{depth}(l, T)$ to be the depth of node l in tree T . So we have $\text{depth}(l, U) = \text{depth}(l, T) - 1$ and $\text{depth}(l, V) = \text{depth}(l, T) - 1$.

By induction hypothesis, we have that

$$\sum_{l \in \text{leaves}(U)} 2^{-\text{depth}(l,U)} \leq 1$$

$$\sum_{l' \in \text{leaves}(V)} 2^{-\text{depth}(l',V)} \leq 1$$

Substituting for $\text{depth}(l,U)$ and $\text{depth}(l',V)$, and adding the two equations, gives

$$\sum_{l \in \text{leaves}(T)} 2^{-\text{depth}(l,T)+1} \leq 2$$

Divide both sides by 2 to get the result for T. The base case for a single node tree ($\text{depth}=0$) is obviously true. The case for a tree of $\text{depth}=1$ can also be verified to be true.

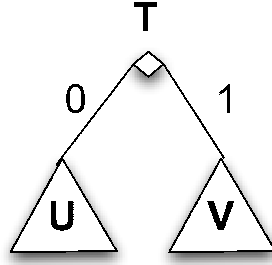


Figure 2: Proof of Kraft's inequality using induction on trees

2.3.3 Entropy lower bound

In the following equations, for a message $s \in S$, $l(s)$ is the associated code length, and $p(s)$ is its probability.

$$\begin{aligned} H(p) - l_a(C) &= \sum_{s \in S} p(s) \log_2 \frac{1}{p(s)} - \sum_{s \in S} p(s) l(s) \\ &= \sum_{s \in S} p(s) \left(\log_2 \frac{1}{p(s)} - l(s) \right) \\ &= \sum_{s \in S} p(s) \left(\log_2 \frac{1}{p(s)} - \log_2 2^{l(s)} \right) \\ &= \sum_{s \in S} p(s) \left(\log_2 \frac{2^{-l(s)}}{p(s)} \right) \\ &= \sum_{s \in S} p(s) \left(\log_2 \frac{2^{-l(s)}}{p(s)} \right) \\ &\leq \log_2 \left(\sum_{s \in S} 2^{-l(s)} \right) \text{ (By Jensen's inequality)} \\ &\leq 0 \text{ (By Kraft's inequality)} \end{aligned}$$

3 Practical Issues [20 Points, Jerry]

1. **Bias-Variance** (14 pts)

We have a set of one dimensional data X_1, \dots, X_n drawn from the positive reals. More specifically, they are drawn from a distribution on the interval $[0, b]$ with probability density function

$$f(x) = \begin{cases} \frac{2x}{b^2} & 0 \leq x \leq b \\ 0 & x < 0 \text{ or } x > b \end{cases}$$

, where b is a positive real parameter, $b > 0$.

- (a) (1 pts) When b is known, what is the mean, μ , of the distribution?

Solution: $2b/3$

$$\begin{aligned} \mathbb{E} \mu &= \int x f(x) dx \\ &= \int_0^b x f(x) dx + 0 \\ &= \int_0^b \frac{2}{b^2} x^2 dx \\ &= \frac{2b}{3} \end{aligned}$$

- (b) (1 pts) What is the Maximum Likelihood estimator, \hat{b} , for b ?

Solution:

$$X_i \leq b \Rightarrow b \geq \max X_i$$

$$\frac{\partial \mathcal{L}}{\partial b} = -\frac{2n}{b} < 0$$

Therefore, \hat{b} is on the lower boundary, $\hat{b} = \max X_i$.

- (c) (2 pts) Now we want to estimate μ . What is the ML estimator $\hat{\mu}$, for μ ? And, what is the mean and variance of $\hat{\mu}$? (Hint: You can take use of \hat{b} when calculating $\hat{\mu}$.)

Solution: Let $y = \max X_i$, F_x be the CDF of X_i , and F_y be the CDF of y .

From part (a) and (b), we get

$$\hat{\mu} = \frac{2}{3}\hat{b} = \frac{2}{3} \max X_i = \frac{2}{3}y.$$

$$F_x(x) = \int_0^x f(x)dx = \frac{x^2}{b^2}$$

$$F_y(y) = P(\max_i X_i \leq y) = \prod_i P(X_i \leq y) = F_x(y)^n = \frac{y^{2n}}{b^{2n}}$$

so, $f_y(y) = \frac{dF_y(y)}{dy} = \frac{2ny^{2n-1}}{b^{2n}}$

$$\begin{aligned} \mathbb{E} \hat{\mu} &= \frac{2}{3}\mathbb{E} y \\ &= \frac{2}{3} \int y f_y(y) dy \\ &= \frac{2}{3} \frac{2n}{2n+1} b \end{aligned}$$

$$\text{Similarly, } \text{var } \hat{\mu} = \frac{4}{9} \frac{n}{(n+1)(2n+1)^2} b^2$$

- (d) (2 pts) The ML estimator $\hat{\mu}$ is biased. What is the squared bias of $\hat{\mu}$? Is it asymptotically unbiased (i.e. as $n \rightarrow \infty$)?

Solution:

$$\begin{aligned} \text{bias}(\hat{\mu}) &= \mathbb{E} \hat{\mu} - 2b/3 \\ &= \frac{2}{3} \frac{1}{(2n+1)} b \\ \text{bias}(\hat{\mu})^2 &= \frac{4}{9} \frac{1}{(2n+1)^2} b^2 \end{aligned}$$

It is asymptotically unbiased, because

$$\lim_{n \rightarrow \infty} \text{bias}(\hat{\mu}) = 0$$

- (e) (1 pts) What is the mean squared error of $\hat{\mu}$? (Reminder: $MSE(\hat{\theta}) = \text{bias}^2(\hat{\theta}) + \text{var}(\hat{\theta})$)

Solution:

$$MSE(\hat{\mu}) = \frac{4b^2}{9} \left(\frac{1}{(2n+1)^2} + \frac{n}{(n+1)(2n+1)^2} \right) = \frac{4b^2}{9(n+1)(2n+1)}$$

- (f) (4 pts) Based on $\hat{\mu}$, write an unbiased estimator, $\tilde{\mu}$, for μ . Show the squared bias, variance, and mean squared error of $\tilde{\mu}$. (Hint: $\tilde{\mu}$ is a function of $\hat{\mu}$ and n .)

Solution:

$$\begin{aligned}\tilde{\mu} &= \frac{2n+1}{2n} \hat{\mu} = \frac{2(2n+1)}{3 \cdot 2n} \hat{b} \\ \mathbb{E} \tilde{\mu} &= \frac{2n+1}{2n} \hat{\mu} = \frac{2}{3} b \\ \text{bias}(\tilde{\mu})^2 &= 0 \\ \text{var} \tilde{\mu} &= \left(\frac{2n+1}{2n}\right)^2 \text{var} \hat{\mu} = \frac{1}{9n(n+1)} b^2 \\ \text{MSE}(\tilde{\mu}) &= \frac{1}{9n(n+1)} b^2\end{aligned}$$

(g) (3 pts) Finally, suppose our data is actually drawn from the interval [0,1], therefore $b=1$. Let's compare the two estimators for μ . Show the plots that compare the bias, variance and MSE of the estimators for μ as a function of n for $n = 1, \dots, 20$. (Use the formula above, and generate three plots of bias/variance/MSE vs. n , where each plot contains results from the two estimators together). Which estimator would you use? Please explain in one sentence.

Solution: Plots are shown in the figure below. The unbiased estimator is better, because

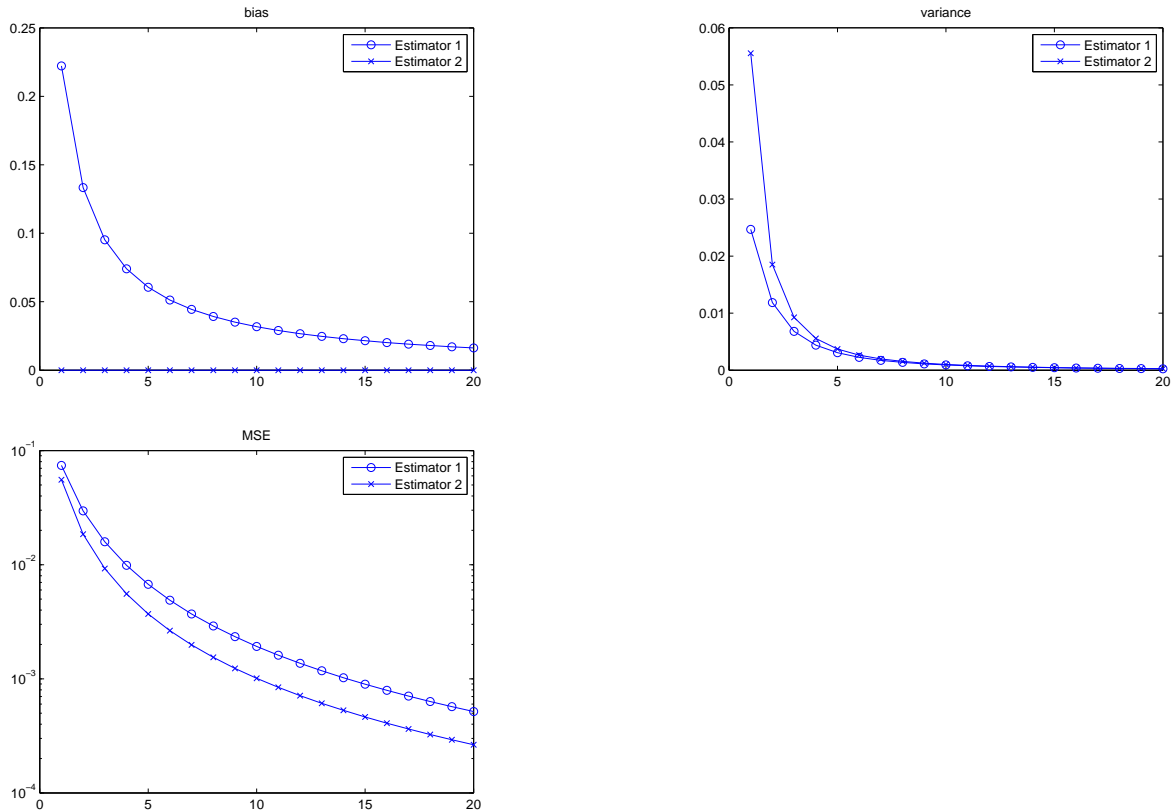


Figure 3: Bias, variance and MSE of the two estimators when b is 1.

$$\text{MSE}(\hat{\mu}) > \text{MSE}(\tilde{\mu}).$$

2. (3 pts) Consider two variables x and y with joint distribution $p(x, y)$. Show that

$$\begin{aligned}\mathbb{E}[x] &= \mathbb{E}_y[\mathbb{E}_x[x | y]] \\ \text{var}[x] &= \mathbb{E}_y[\text{var}_x[x | y]] + \text{var}_y[\mathbb{E}_x[x | y]]\end{aligned}$$

Here $\mathbb{E}_x[x | y]$ denotes the expectation of x under the conditional distribution $p(x | y)$, with a similar notation for the conditional variance.

Solution:

$$\begin{aligned}\mathbb{E}_y[\mathbb{E}_x[x | y]] &= \sum_y p(y) \mathbb{E}_x[x | y] \\ &= \sum_y p(y) \sum_x x p(x | y) \\ &= \sum_y p(y) \sum_x x \frac{p(x, y)}{p(y)} \\ &= \sum_y \sum_x p(y) x \frac{p(x, y)}{p(y)} \\ &= \sum_y \sum_x x p(x, y) \\ &= \sum_x x \sum_y p(x, y) \\ &= \sum_x x p(x) = \mathbb{E}[x]\end{aligned}$$

$$\begin{aligned}\text{var } z &= \mathbb{E} z^2 - (\mathbb{E} z)^2 \\ \text{var}_x[x | y] &= \mathbb{E}_x[x^2 | y] - (\mathbb{E}_x[x | y])^2 \\ \text{var}_y[\mathbb{E}_x[x | y]] &= \mathbb{E}_y[(\mathbb{E}_x[x | y])^2] - (\mathbb{E}_y[\mathbb{E}_x[x | y]])^2 \\ \text{so, } \mathbb{E}_y[\text{var}_x[x | y]] + \text{var}_y[\mathbb{E}_x[x | y]] & \\ &= \mathbb{E}_y[\mathbb{E}_x[x^2 | y]] - \mathbb{E}_y[(\mathbb{E}_x[x | y])^2] + \mathbb{E}_y[(\mathbb{E}_x[x | y])^2] - (\mathbb{E}_y[\mathbb{E}_x[x | y]])^2 \\ &= \mathbb{E}_y[\mathbb{E}_x[x^2 | y]] - (\mathbb{E}_y[\mathbb{E}_x[x | y]])^2 \\ &= \mathbb{E}_x[x^2] - (\mathbb{E}_x[x])^2 \\ &= \text{var}[x]\end{aligned}$$

3. (3 pts) Chris recently adopts a new (binary) classifier to filter email spams. He wants to quantitatively evaluate how good the classifier is. He has a small dataset of 100 emails on hand which, you can assume, are randomly drawn from all emails. He tests the classifier on the 100 emails and gets 83 classified correctly, so the error rate on the small dataset is 17%. However, the number on 100 samples could be either higher or lower than the real error rate just by chance. With a confidence level of 95%, what is likely to be the range of the real error rate? Please write down all important steps. (Hint: You need some approximation in this problem.)

Solution:

$$e_{sample} = 0.17$$

$$Var_{sample} = 0.17 * (1 - 0.17)/100 = 0.001411$$

By Central Limit Theorem, $e_{sample} \sim N(e_{real}, Var_{real})$. With a confidence level of 95%,

$$|e_{real} - e_{sample}| \leq 1.96 * \sqrt{Var_{real}} \approx 1.96 * \sqrt{Var_{sample}} = 0.07$$

$$0.10 \leq e_{real} \leq 0.24$$

4 AdaBoost [Hanghang, 40 points]

Given N examples (x_i, y_i) , where y_i is the label and $y_i = +1$ or $y_i = -1$. Let $I(\cdot)$ be the indicator function, which is 1 if the condition in (\cdot) is true and 0 otherwise. In this exercise, we use the following version for AdaBoost algorithm:

1. Initialize $w_i^1 = 1/N$ ($i = 1, \dots, N$)
2. For $t = 1, \dots, M$,
 - a. Learn a weak classifier $h_t(x)$ by minimizing the weighed error function J_t , where $J_t = \sum_{i=1}^N w_i^t I(h_t(x_i) \neq y_i)$;
 - b. Compute the error rate for the learnt weak classifier $h_t(x)$: $\epsilon_t = \sum_{i=1}^N w_i^t I(h_t(x_i) \neq y_i)$;
 - c. Compute the weight for $h_t(x)$: $\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}$;
 - d. Update the weight for each example: $w_i^{t+1} = \frac{w_i^t \exp\{-\alpha_t y_i h_t(x_i)\}}{Z_t}$, where Z_t is the normalization factor for w_i^{t+1} : $Z_t = \sum_{i=1}^N w_i^t \exp\{-\alpha_t y_i h_t(x_i)\}$.
3. Output the final classifier: $H(x) = \text{sign}(f_M(x))$, where $f_M(x)$ is a linear combination of the weak classifiers, i.e., $f_M(x) = \sum_{t=1}^M \alpha_t h_t(x)$.

1.1 Sequential Optimization [10 pts]

- (1) [5 pts] Remember in AdaBoost, we try to minimize the negative exponential loss: $E = \sum_{i=1}^N \exp\{-y_i f_M(x_i)\}$ sequentially. That is to say, at the m^{th} ($1 \leq m \leq M$) iteration, we want to choose appropriate weight α_m and the corresponding weak classifier $h_m(x)$ so that the overall loss E (accumulated up to m^{th} iteration) is minimized.

Prove that the above strategy will lead to the following update rule: in the m^{th} iteration, $\alpha_m = \frac{1}{2} \ln \frac{1-\epsilon_m}{\epsilon_m}$, where ϵ_m is the error rate for the weak classifier $h_m(x)$.

hints: you can use the fact that $w_i^m \propto \exp\{-y_i f_{m-1}(x_i)\}$, which can be viewed as constant when we try to optimize α_m in the m^{th} iteration.

Solutions: At the m^{th} iteration, we have

$$\begin{aligned}
E &= \sum_{i=1}^N \exp\{-y_i f_m(x_i)\} \\
&= \sum_{i=1}^N \exp\{-y_i (\sum_{t=1}^{m-1} \alpha_t h_t(x_i)) - y_i \alpha_m h_m(x_i)\} \\
&= \sum_{i=1}^N \exp\{-y_i f_{m-1}(x_i)\} \exp\{-y_i \alpha_m h_m(x_i)\} \\
&\propto \sum_{i=1}^N w_i^m \exp\{-y_i \alpha_m h_m(x_i)\} = E'
\end{aligned} \tag{1}$$

For E' defined in eq. (1), we can re-write it as

$$\begin{aligned}
E' &= \sum_{i=1}^N w_i^m \exp\{-y_i \alpha_m h_m(x_i)\} \\
&= \sum_{i \in \mathcal{T}_1} w_i^m \exp\{-\alpha_m\} + \sum_{i \in \mathcal{T}_2} w_i^m \exp\{\alpha_m\} \\
&= \exp\{-\alpha_m\} (1 - \epsilon_m) + \exp\{\alpha_m\} \epsilon_m
\end{aligned} \tag{2}$$

where \mathcal{T}_1 is the set of examples which are correctly classified by $h_m(x)$ and \mathcal{T}_2 is the set of examples which are mis-classified by $h_m(x)$.

Based on eq. (2), we have

$$\frac{\partial E'}{\partial \alpha_m} = -\exp\{-\alpha_m\} (1 - \epsilon_m) + \exp\{\alpha_m\} \epsilon_m \tag{3}$$

Setting $\partial E' / \partial \alpha_m = 0$, we have $\alpha_m = 1/2 \ln \frac{1 - \epsilon_m}{\epsilon_m}$.

- (2) [5 pts] Now, if we change the objective function E to $E = \sum_{i=1}^N (y_i - f_M(x_i))^2$ and we still want to optimize it sequentially. What is the new update rule for α_m ?

Solutions: $\alpha_m = \frac{\sum_{i=1}^N r_i h_m(x_i)}{\sum_{i=1}^N h_m^2(x_i)} = \frac{\sum_{i=1}^N r_i h_m(x_i)}{N}$, where $r_i = y_i - \sum_{t=1}^{m-1} \alpha_t h_t(x_i)$.

1.2 Training Error [10 pts]

- (1) [5 pts] Prove that the training error $\epsilon_{\text{training}}$ of AdaBoost is upper bounded by $\prod_{t=1}^M Z_t$, where the training error $\epsilon_{\text{training}} = \frac{1}{N} \sum_{i=1}^N I(H(x_i) \neq y_i)$.

hints: First try to show that $\prod_{t=1}^M Z_t = \frac{1}{N} \sum_{i=1}^N \exp\{-y_i f_M(x_i)\}$

Solutions: First, we prove that $\prod_{t=1}^M Z_t = \frac{1}{N} \sum_{i=1}^N \exp\{-y_i f_M(x_i)\}$.

We have

$$w_i^{M+1} = \frac{w_i^M \exp\{-\alpha_M y_i h_M(x_i)\}}{Z_M} \tag{4}$$

Now, recursively replacing $w_i^t (t = M, \dots, 1)$ in eq. (4), we have

$$\begin{aligned}
w_i^{M+1} &= \frac{w_i^M \exp\{-\alpha_M y_i h_M(x_i)\}}{Z_M} \\
&= \frac{w_i^{M-1} \exp\{-\alpha_{M-1} y_i h_{M-1}(x_i)\} \exp\{-\alpha_M y_i h_M(x_i)\}}{Z_M Z_{M-1}} \\
&\dots \\
&= \frac{1/N \exp\{-y_i \sum_{t=1}^M \alpha_t h_t(x_i)\}}{\prod_{t=1}^M Z_t} \\
&= \frac{1/N \exp\{-y_i f_M(x_i)\}}{\prod_{t=1}^M Z_t} \tag{5}
\end{aligned}$$

Now, taking the summation of i from 1 to N for both sides of eq. (5), we have,

$$1 = \sum_{i=1}^N w_i^{M+1} = \frac{1}{N} \frac{\sum_{i=1}^N \exp\{-y_i f_M(x_i)\}}{\prod_{t=1}^M Z_t} \tag{6}$$

Therefore, we have,

$$\prod_{t=1}^M Z_t = \frac{1}{N} \sum_{i=1}^N \exp\{-y_i f_M(x_i)\} \tag{7}$$

Now, for each i , if $y_i f_M(x_i) > 0$, we have $I(H(x_i) \neq y_i) = 0 \leq \exp\{-y_i f_M(x_i)\}$; on the other hand, if $y_i f_M(x_i) < 0$, we have $I(H(x_i) \neq y_i) = 1 \leq \exp\{-y_i f_M(x_i)\}$. So, we always have $I(H(x_i) \neq y_i) \leq \exp\{-y_i f_M(x_i)\}$ for each example x_i . Therefore,

$$\begin{aligned}
\epsilon_{training} &= \frac{1}{N} \sum_{i=1}^N I(H(x_i) \neq y_i) \\
&\leq \frac{1}{N} \sum_{i=1}^N \exp\{-y_i f_M(x_i)\} \\
&= \prod_{t=1}^M Z_t \tag{8}
\end{aligned}$$

- (2) [5 pts] Another way to explain AdaBoost is by sequentially minimizing the training error $\epsilon_{training}$. One way to achieve this is to minimize its upper bound $\prod_{t=1}^m Z_t$ sequentially at each iteration m ($1 \leq m \leq M$).

Prove that this strategy (i.e., to minimize $\prod_{t=1}^m Z_t$ at each iteration m) will lead to the same update rule for α_m used in AdaBoost, i.e., $\alpha_m = \frac{1}{2} \ln \frac{1-\epsilon_m}{\epsilon_m}$.

hints: at iteration m , minimizing $\prod_{t=1}^m Z_t$ is equivalent to minimizing Z_m since $\prod_{t=1}^{m-1} Z_t$ is fixed.

Solutions: At iteration m , minimizing $\prod_{t=1}^m Z_t$ is equivalent to minimizing Z_m since $\prod_{t=1}^{m-1} Z_t$ is fixed. Now notice that $Z_m = \sum_{i=1}^N w_i^m \exp\{-y_i h_m(x_i) \alpha_i\}$ is exactly the same as E' defined in eq. (1). Therefore, by the same procedure, we can prove that minimizing Z_m will lead to updating α_m by $\alpha_m = 1/2 \ln \frac{1-\epsilon_m}{\epsilon_m}$.

1.3 **Implementation** [20 pts] Implement AdaBoost on your own. Use decision stumps as the weak classifier. Remember that a decision stump is a decision tree with one single node. It corresponds to a single threshold on one of the features, and predicts the class label for examples falling above and below the threshold respectively (i.e., “ $x^j < c$, predict 1, otherwise predict 0”; or “ $x^j < c$, predict 0, otherwise predict 1”. Here x^j is the j^{th} dimension of the features.) At each iteration, we will greedily choose the feature which minimizes the weighted training error ϵ_t .

- (1) [10 pts] Evaluate your implementation on this dataset; do 10-fold cross-validation and report the average training error vs. the iteration number, and the average test error vs. the iteration number on the same figure. (The maximum iteration number is 100)

Solutions: See figure 4

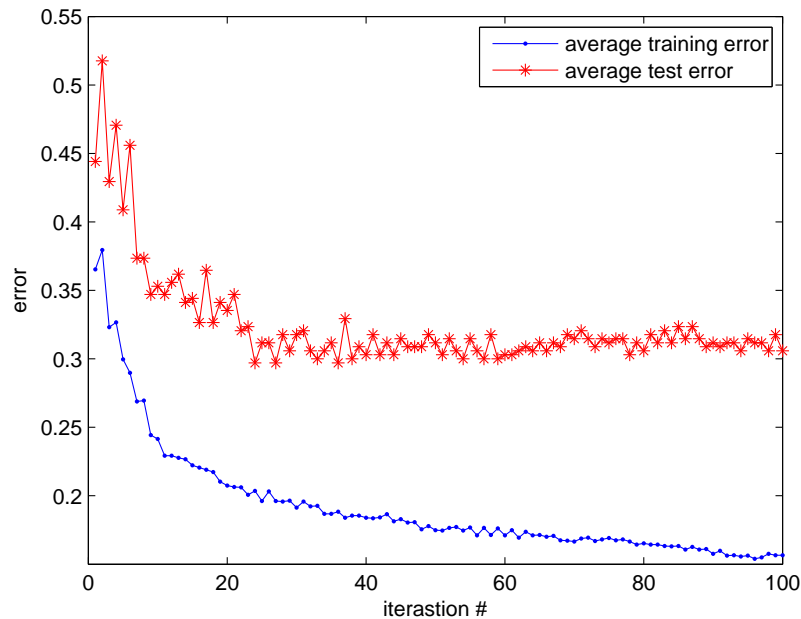


Figure 4: 10-fold cross validation

- (2) [2 pts] Based on the figure you got, what do you think is the optimal iteration number for this dataset?

Solutions: (Depending on the actual curve you got), 30 in our case.

- (3) [8 pts] Submit a hardcopy of your code