Homework #6

RELEASE DATE: 12/23/2021

DUE DATE: 01/06/2022, BEFORE 13:00 on Gradescope

QUESTIONS ARE WELCOMED ON THE NTU COOL FORUM.

You will use Gradescope to upload your choices and your scanned/printed solutions. For problems marked with (*), please follow the guidelines on the course website and upload your source code to Gradescope as well. Any programming language/platform is allowed.

Any form of cheating, lying, or plagiarism will not be tolerated. Students can get zero scores and/or fail the class and/or be kicked out of school and/or receive other punishments for those kinds of misconducts.

Discussions on course materials and homework solutions are encouraged. But you should write the final solutions alone and understand them fully. Books, notes, and Internet resources can be consulted, but not copied from.

Since everyone needs to write the final solutions alone, there is absolutely no need to lend your homework solutions and/or source codes to your classmates at any time. In order to maximize the level of fairness in this class, lending and borrowing homework solutions are both regarded as dishonest behaviors and will be punished according to the honesty policy.

You should write your solutions in English or Chinese with the common math notations introduced in class or in the problems. We do not accept solutions written in any other languages.

This homework set comes with 16 problems and a total of 400 points. For each problem, there is one correct choice. If you choose the correct answer, you get 20 points; if you choose an incorrect answer, you get 0 points. For four of the secretly-selected problems, the TAs will grade your detailed solution in terms of the written explanations and/or code based on how logical/clear your solution is. Each of the four problems graded by the TAs counts as additional 20 points (in addition to the correct/incorrect choices you made). In general, each homework (except homework 0) is of a total of 400 points.

Aggregation

1. Consider an aggregation classifier G constructed by uniform blending on 11 classifiers $\{g_t\}_{t=1}^{11}$. That is,

$$G(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{11} g_t(\mathbf{x})\right).$$

Assume that each g_t is of test 0/1 error $E_{\text{out}}(g_t) = e_t$. Which of the following is the tightest upper bound of $E_{\text{out}}(G)$? Choose the correct answer; explain your answer.

[a] $\frac{1}{3}\sum_{t=1}^{11} e_t$

[b] $\frac{1}{4}\sum_{t=1}^{11} e_t$

[c] $\frac{1}{6}\sum_{t=1}^{11} e_t$

[d] $\frac{1}{11}\sum_{t=1}^{11} e_t$

[e] $\frac{1}{12}\sum_{t=1}^{11} e_t$

2. Suppose that each $\mathbf{x} = (x_1, x_2) \in \mathbb{R}^2$ is drawn uniformly from the region

$$\{0 \le x_1 \le 3, 0 \le x_2 \le 3\}$$

and the target function is $f(\mathbf{x}) = \text{sign}(x_2 - x_1)$. Consider blending the following three hypotheses linearly to approximate the target function.

$$\begin{aligned} g_1(\mathbf{x}) &= \text{sign}(x_1 - 2) \\ g_2(\mathbf{x}) &= \text{sign}(x_2 - 1) \\ g_3(\mathbf{x}) &= \text{sign}(x_2 - 2) \end{aligned}$$

That is,

$$G(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{3} \alpha_t \cdot g_t(\mathbf{x})\right)$$

with $\alpha_t \in \mathbb{R}$. What is the smallest possible $E_{\text{out}}(G)$? Choose the correct answer; explain your answer.

(*Hint: The "boundary" of G must be a "combination" of the boundaries of g_t*)

[a] $\frac{6}{18}$
[b] $\frac{5}{18}$
[c] $\frac{4}{18}$
[d] $\frac{3}{18}$
[e] none of the other choices

3. When talking about non-uniform voting in aggregation, we mentioned that $\boldsymbol{\alpha}$ can be viewed as a weight vector learned from any linear algorithm coupled with the following transform:

$$\boldsymbol{\phi}(\mathbf{x}) = \Big(g_1(\mathbf{x}), g_2(\mathbf{x}), \cdots, g_T(\mathbf{x})\Big).$$

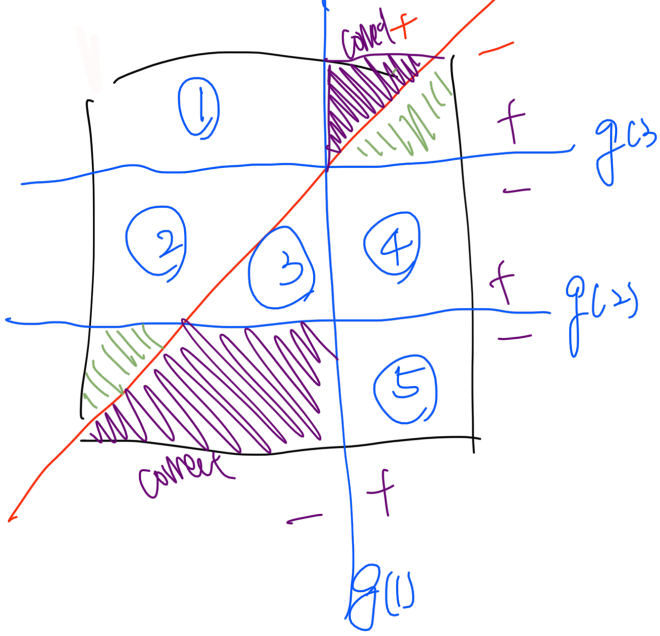
When studying kernel methods, we mentioned that the kernel is simply a computational short-cut for the inner product $(\boldsymbol{\phi}(\mathbf{x}))^T(\boldsymbol{\phi}(\mathbf{x}'))$. In this problem, we mix the two topics together using the decision stumps as our $g_t(\mathbf{x})$.

Assume that the input vectors contain only even integers between (including) $2L$ and $2R$, where $L < R$. Consider the decision stumps $g_{s,i,\theta}(\mathbf{x}) = s \cdot \text{sign}\big(x_i - \theta\big)$, where

$$\begin{aligned} i &\in \{1, 2, \cdots, d\}, \\ d &\text{ is the finite dimensionality of the input space,} \\ s &\in \{-1, +1\}, \\ \theta &\text{ is an odd integer between } (2L, 2R). \end{aligned}$$

Define $\boldsymbol{\phi}_{ds}(\mathbf{x}) = \Big(g_{+1,1,2L+1}(\mathbf{x}), g_{+1,1,2L+3}(\mathbf{x}), \ldots, g_{+1,1,2R-1}(\mathbf{x}), \ldots, g_{-1,d,2R-1}(\mathbf{x})\Big)$. What is $K_{ds}(\mathbf{x}, \mathbf{x}') = (\boldsymbol{\phi}_{ds}(\mathbf{x}))^T(\boldsymbol{\phi}_{ds}(\mathbf{x}'))$? Choose the correct answer; explain your answer.

[a] $2d(R-L) - 2\|\mathbf{x} - \mathbf{x}'\|_1$
[b] $2d(R-L)^2 - 2\|\mathbf{x} - \mathbf{x}'\|_1^2$
[c] $2d(R-L) - 2\|\mathbf{x} - \mathbf{x}'\|_2$
[d] $2d(R-L)^2 - 2\|\mathbf{x} - \mathbf{x}'\|_2^2$
[e] none of the other choices



123 sign(α_1 − + α_2 − + α_3 −)

$G(x) =$

for ▨ $E_{out}(x) = 0$.

for ▨ $E_{out}(x) = ?$

$g(1) = \text{sign}(x_1 - 2)$

$g(2) = \text{sign}(x_2 - 1)$

$g(3) = \text{sign}(x_2 - 2)$

for ① = $(\alpha_2 + \alpha_3) - (\alpha_1)$

for ③ = $(\alpha_2) - (\alpha_1 + \alpha_3)$

for ⑤ = $(\alpha_2) - (\alpha_1 + \alpha_3)$

for ④ = $(\alpha_1 + \alpha_2) - (\alpha_3)$

for ⑤ = $(\alpha_1) - (\alpha_2 + \alpha_3)$

討論 $\alpha_1, \alpha_2, \alpha_3$ 的大小

①、⑤本身互為一個負號轉 target 一樣

令①、⑤沒有 error → $\alpha_2 + \alpha_3 > \alpha_1$ ——— ④

②③相等但跟 target 相反

取最小的 E_{out} → ③ error ② correct

——— ②

$\alpha_2 > \alpha_1 + \alpha_3$

④②可等. $\alpha_2 > \alpha_1$, $\alpha_2 > \alpha_3$?

→ for ④ $\alpha_1 + \alpha_2 - \alpha_3 > 0$

$\alpha_1 + \alpha_2 > \alpha_3$ → error. ⇒ $\frac{1}{18} + \frac{1}{18} + \frac{1}{18} + \frac{1}{9}$

total error = $\frac{5}{18}$.

在 时 $\theta_1 \to x_1, x'_1 \Rightarrow ++ \to s(+)\cdot s(+) = 1$

$s=1$ $\quad \theta_2 \to x_1, x'_1 \Rightarrow -+ \to s(-)\, s(+) = -1$

$\theta_3 \to x_1, x'_1 \Rightarrow -- \quad s(-)\, s(-) = 1$

在 $\to ++ \to s(+)\, s(+) = 1$

$s=-1 \to$ $-+$ $=-1$

$--$ $=-1$

s 在正负改变时 → 乘 2 即可.

$g(x)\cdot g(x')$

所有奇数 between 2R, 2L

$$\to \frac{(2R-2L)}{2} = R-L$$

x 与 x' 两者之中的奇数

$$\frac{|x_i - x'_i|}{2}$$

$$\sum\sum_{d\,\theta} g(x)\cdot g(x') = d(R-L) - \frac{|x_i - x'_i|}{2} \times 2$$

$$\to \sum_s \sum_d \sum_\theta = 2\cdot d(R-L) - 2\|x_i - x'_i\|_1$$

Adaptive Boosting

4. Consider applying the AdaBoost algorithm on a binary classification data set where 99% of the examples are positive. Because there are so many positive examples, the base algorithm within AdaBoost returns a constant classifier $g_1(\mathbf{x}) = +1$ in the first iteration. Let $u_n^{(2)}$ be the individual example weight of each example in the second iteration. What is

$$\frac{\sum_{n:\ y_n>0} u_n^{(2)}}{\sum_{n:\ y_n<0} u_n^{(2)}}?$$

Choose the correct answer; explain your answer.

[a] 99
[b] 1/99
[c] 1
[d] 100
[e] 1/100

5. For the AdaBoost algorithm introduced in Lecture 12, let $G_t(\mathbf{x}) = \text{sign}\left(\sum_{\tau=1}^{t} g_\tau(\mathbf{x})\right)$. How many of the following are guaranteed to be non-increasing from the t-th iteration to the $(t+1)$-th iteration? Choose the correct answer; explain each non-increasing case within your answer.

- $E_{\text{in}}(G_t)$ to $E_{\text{in}}(G_{t+1})$
- $E_{\text{out}}(G_t)$ to $E_{\text{out}}(G_{t+1})$
- $\sum_{n=1}^{N} u_n^{(t)}$ to $\sum_{n=1}^{N} u_n^{(t+1)}$
- $u_n^{(t)}$ to $u_n^{(t+1)}$ when g_t is correct on $(\mathbf{x}_n, y_n)$
- $u_n^{(t)}$ to $u_n^{(t+1)}$ when g_t is incorrect on $(\mathbf{x}_n, y_n)$

[a] 1
[b] 2
[c] 3
[d] 4
[e] 5

6. For the AdaBoost algorithm introduced in Lecture 12, let $U_t = \sum_{n=1}^{N} u_n^{(t)}$. Assume that $0 < \epsilon_t < \frac{1}{2}$ for each hypothesis g_t. What is $\frac{U_{t+1}}{U_t}$? Choose the correct answer; explain your answer.

[a] $\sqrt{\epsilon_t(1-\epsilon_t)}$
[b] $2\sqrt{\epsilon_t(1-\epsilon_t)}$
[c] $\sqrt{\frac{\epsilon_t}{(1-\epsilon_t)}}$
[d] $\ln\sqrt{\frac{(1-\epsilon_t)}{\epsilon_t}}$
[e] $\ln\sqrt{\frac{\epsilon_t}{(1-\epsilon_t)}}$

5. ① $E_{in}(G)$, $E_{out}(G)$ 會上下波動 整體趨勢 降低

②  ✗

③ → V by 第6.題 → 會增加 ✗

④ $\frac{t+1}{t} > \varepsilon > 0$ $\blacklozenge = \sqrt{\frac{1-\varepsilon}{\varepsilon}} > 1$ ⇒ ✱ → increase

/ → non-increase.

⑤

6. $$\frac{U_{t+1}}{U_t} = \frac{\sum u_n^{(t-1)} \cdot [1-\varepsilon_t]\exp(-\alpha_t) + [\varepsilon_t]\exp[-\alpha_t]}{\sum u_n^{(t-1)}}$$

$\alpha_t = \ln(\blacklozenge_t) = \ln\sqrt{\frac{1-\varepsilon_t}{\varepsilon_t}}$

$$= \varepsilon_t \cdot \sqrt{\frac{1-\varepsilon_t}{\varepsilon_t}} + (1-\varepsilon_t)\sqrt{\frac{\varepsilon_t}{1-\varepsilon_t}}$$

$$= 2\sqrt{\varepsilon_t(1-\varepsilon_t)} \quad \#$$

7. Following the previous two problems, assume that $\epsilon_t \leq \epsilon < \frac{1}{2}$, which of the following is the tightest upper bound on the number of iterations T required to ensure $E_{\text{in}}(G_T) = 0$? Choose the correct answer; explain your answer.

(*Hint: use the fact that*

$$\sqrt{\epsilon(1-\epsilon)} \leq \frac{1}{2}\exp\left(-2(\frac{1}{2}-\epsilon)^2\right)$$

for all $0 < \epsilon < \frac{1}{2}$).

[a] $\frac{\ln N}{2(\frac{1}{2}-\epsilon)}$

[b] $\frac{\ln N}{2(\frac{1}{2}-\epsilon)^2}$

[c] $\frac{\ln N}{4(\frac{1}{2}-\epsilon)}$

[d] $\frac{\ln N}{4(\frac{1}{2}-\epsilon)^2}$

[e] $\frac{\ln N}{4(\frac{1}{2}-\epsilon)^4}$

Random Forest = Bagging + Decision Tree

8. Suppose we have a data set of size $N = 1126$, and we use bootstrapping to sample N' examples. What is the minimum N' such that the probability of getting at least one duplicated example (with # copies ≥ 2) is larger than 50%? Choose the correct answer; explain your answer.

[a] 25

[b] 30

[c] 35

[d] 40

[e] none of the other choices

9. If bootstrapping is used to sample exactly $2N$ examples out of N, what is the probability that an example is *not* sampled when N is very large? Choose the closest answer; explain your answer.

[a] 77.9%

[b] 60.7%

[c] 36.8%

[d] 13.5%

[e] 1.8%

10. Suppose we have a set of decision trees. Each tree comes with 2 node, each equipped with a fixed branching function. The root node is of two branches, evaluating whether $x_1 \geq 0$. If $x_1 < 0$, the node connects to a leaf with some constant output. Otherwise the node connects to another node of two branches, evaluating whether $x_2 \geq 0$. Each of the branches connects to a constant leaf. Consider three-dimensional input vectors. That is, $\mathbf{x} = (x_1, x_2, x_3)$. Which of the following data set can be shattered by the set of decision trees? Choose the correct answer; explain your answer.

[a] $(1,1,-1),(-1,1,-1),(-1,-1,-1)$

[b] $(1,1,-1),(-1,1,-1),(1,-1,-1)$

[c] $(1,1,-1),(-1,1,-1),(1,-1,-1),(-1,-1,-1)$

[d] $(1,1,-1),(-1,1,-1),(1,-1,-1),(-1,-1,1)$

[e] none of the other choices

7. ∵ $0 \le \epsilon < \frac{1}{2} \rightarrow \sqrt{\epsilon_t(1-\epsilon_t)} \le \sqrt{\epsilon(1-\epsilon)}$

by TA

$$v_{T+1} = \frac{v_{T+1}}{v_T} \cdot \frac{v_T}{v_{T-1}} \cdots \frac{v_2}{v_1} \cdot v_1$$

提到

$$u = \frac{v_2}{v_1} \cdot \frac{v_3}{v_2} \cdots \frac{v_{T+1}}{v_T}$$

$\because v_1 = \sum u_1 = 1$

$u_n = \frac{1}{N}$

$$v_{T+1} = \frac{v_{T+1}}{v_T} \cdot \frac{v_T}{v_{T-1}} \cdots \frac{v_2}{v_1}$$

$$= \prod_1^T \frac{v_{T+1}}{v_T} = \prod_1^T 2 \cdot \sqrt{\epsilon_t(1-\epsilon_t)}$$ ← by 及G

$$v_{T+1} = \prod 2 \cdot \sqrt{\epsilon_t(1-\epsilon_t)} \le \prod 2 \cdot \sqrt{\epsilon(1-\epsilon)}$$ by 題目

$$\le \pi 2 \cdot \frac{1}{2} \exp\left(-2\left(\frac{1}{2}-\epsilon\right)^2\right)$$

$$= \exp\left(-2T\left(\frac{1}{2}-\epsilon\right)^2\right)$$

$$E_{in}(G_T) \le v_{T+1} \le \exp\left(-2T\left(\frac{1}{2}-\epsilon\right)^2\right)$$

→

$$\text{find } T \rightarrow \frac{\ln N}{2\left(\frac{1}{2}-\epsilon\right)^2}$$ #

8.

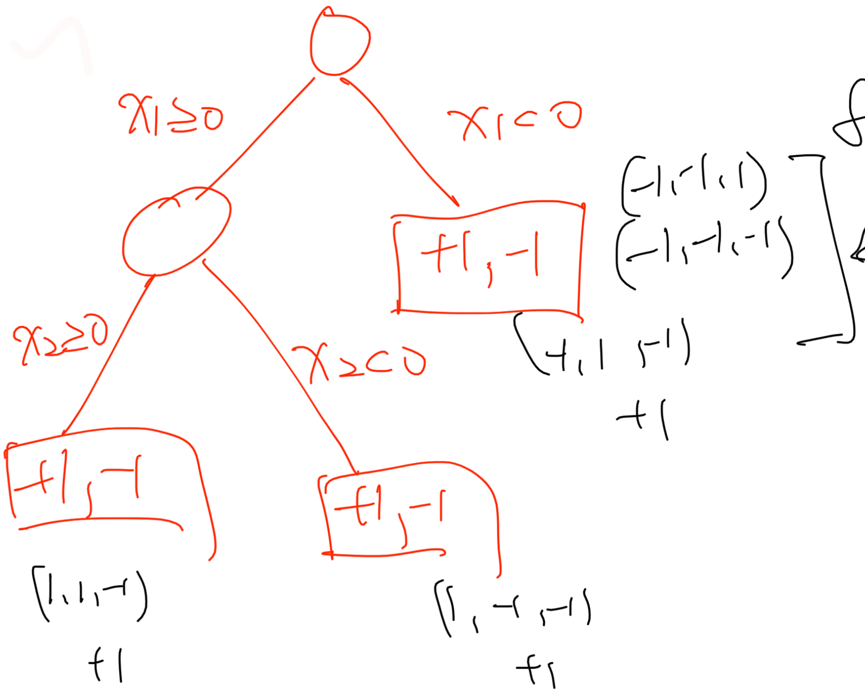
$$\frac{1126}{1126}\times\frac{1125}{1126}\times\cdots\frac{1126-N+1}{1126}$$

$$N = 35 \rightarrow 0.586$$

$$= 40 \rightarrow 0.496 \checkmark$$

$$\frac{1126!}{(1126-N)!\times 1126^N}$$

1-全部不重複≥0.5 全部不重複 ≤0.5



for (a, c, d)

無法 shatter

如果都跑到1個 leaf

2個點 if 1個正，1個負

就不能 shatter

Experiments with Adaptive Boosting

For Problems 11-16, implement the AdaBoost-Stump algorithm as introduced in Classes 12 and 13. Run the algorithm on the following set for training:

`https://www.csie.ntu.edu.tw/~htlin/course/ml21fall/hw6/hw6_train.dat`

and the following set for testing:

`https://www.csie.ntu.edu.tw/~htlin/course/ml21fall/hw6/hw6_test.dat`

Use a total of $T = 500$ iterations (please do not stop earlier than 500), and calculate E_{in} and E_{out} with the 0/1 error.

For the decision stump algorithm, please implement the following steps. Any ties can be arbitrarily broken.

(1) For any feature i, sort all the $x_{n,i}$ values to $x_{[n],i}$ such that $x_{[n],i} \leq x_{[n+1],i}$.

(2) Consider thresholds within $-\infty$ and all the midpoints $\frac{x_{[n],i}+x_{[n+1],i}}{2}$. Test those thresholds with $s \in \{-1, +1\}$ to determine the best (s, θ) combination that minimizes E_{in}^u using feature i.

(3) Pick the best (s, i, θ) combination by enumerating over all possible i.

For those interested in algorithms (who isn't? :-)), step 2 can be carried out in $O(N)$ time only!!

11. (*) What is the value of $E_{\text{in}}(g_1)$? Choose the closest answer; provide your code.

[a] 0.29
[b] 0.33
[c] 0.37
[d] 0.41
[e] 0.45

12. (*) What is the value of $\max_{1 \leq 500 \leq t} E_{\text{in}}(g_t)$? Choose the closest answer; provide your code.

[a] 0.40
[b] 0.45
[c] 0.50
[d] 0.55
[e] 0.60

13. (*) What is the smallest t within the choices below such that $\min_{1 \leq \tau \leq t} E_{\text{in}}(G_\tau) \leq 0.05$? Choose the correct answer; provide your code.

[a] 60
[b] 160
[c] 260
[d] 360
[e] 460

14. (*) What is the value of $E_{\text{out}}(g_1)$? Choose the closest answer; provide your code.

[a] 0.40
[b] 0.45
[c] 0.50
[d] 0.55
[e] 0.60

15. (*) Define $G_{\text{uniform}}(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{T} g_t(\mathbf{x})\right)$. What is the value of $E_{\text{out}}(G_{\text{uniform}})$? Choose the closest answer; provide your code.

[a] 0.23
[b] 0.28
[c] 0.33
[d] 0.38
[e] 0.43

16. (*) What is the value of $E_{\text{out}}(G_{500})$? Choose the closest answer; provide your code.

[a] 0.14
[b] 0.18
[c] 0.22
[d] 0.26
[e] 0.30

```
            predict += a * s * np.sign(x[d] - theta)

        result.append( np.sign(predict) )

    return np.array(result)
def main():

    (X, Y) = read_data('hw6_train.dat.txt')
    D = X.shape[1]          # dimension of X
    N = X.shape[0]          # data size
    print('X shape:', X.shape)
    print('Y shape:', Y.shape)

    T = 1                       # iteration
    G = []                        # all g
    a = np.zeros(T)               # weight of gt (alpha)
    Ein_g = []                    # 0/1 errors of each iteration
    Ein_G = []                    # 0/1 errors of all iteration
    Epsilon = []                # all epsilon of each iteration
    u = np.array( [ 1/N ] * N )   # weight of each sample
    U = [1]                       # sum of weights u
    sorted_index = []
    unsorted_index = []

    for d in range(D):
        index = np.argsort(X[:, d])
        sorted_index.append( index )
        unsorted_index.append( np.argsort(index) )

    for t in range(T):
        print('\rt = %d' % (t+1), end='', flush=True)

        best_abs_sum, best_s, best_i, best_d = 0, 1, -1, 0
        for d in range(D):

            index = sorted_index[d]
            left, right = 0, np.sum(Y * u)
            abs_sum = abs(right - left)

            if abs_sum > best_abs_sum:
                best_abs_sum = abs_sum
                best_s = 1 if right >= left else -1
                best_i, best_d = -1, d

            Y_tmp, u_tmp = Y[index], u[index]
            for i, y in enumerate(Y_tmp):

                right -= y * u_tmp[i]
                left += y * u_tmp[i]
                abs_sum = abs(right - left)

                if abs_sum > best_abs_sum:
                    best_abs_sum = abs_sum
                    best_s = 1 if right >= left else -1
                    best_i, best_d = i, d

        index = sorted_index[best_d]
        # best division (theta)
        X_tmp = X[index][:, best_d]
        if best_i < 0:
            theta = -np.inf
        elif best_i >= N-1:
            theta = np.inf
        else:
            x1 = X_tmp[best_i]
            x2 = X_tmp[best_i+1]
            theta = (x2 + x1) / 2

        g = (best_s, best_d, theta)

        # predict by small gt
        predict_g = predict([g], [1], X)
        error01_g = abs(predict_g - Y) / 2
        epsilon_g = np.sum(error01_g * u) / u.sum()
        scale = np.sqrt( (1-epsilon_g) / epsilon_g )
        # update u
        incorrect = np.where(error01_g == 1)[0]
        correct = np.where(error01_g == 0)[0]
        u[incorrect] *= scale
        u[correct] /= scale
        U.append( u.sum() )

        a[t] = np.log(scale)
        Ein_g.append( np.sum(error01_g) / N )
        Epsilon.append( epsilon_g )
        G.append(g)

        # predict by big Gt
        predict_G = predict(G, a, X)
        error01_G = np.sum( abs(predict_G - Y) / 2 ) / N
        Ein_G.append(error01_G)

    print('')
    print('Ein(g1):', Ein_g[0])

    print('max Ein(gt):', max(Ein_g))

    for i in range(T):
        if Ein_G[i] <= 0.05:
            print('Ein(G) < 0.05 , t = ' ,i+1)
            break
    (X_test, Y_test) = read_data('hw6_test.dat.txt')
    N_test = X_test.shape[0]
    Eout_g = []
    guni = np.zeros(N_test)
    for i, g in enumerate(G):
            print('\r%d' % (i+1), end='', flush=True)
            predict_g = predict([g], [1], X_test)
            guni +=predict_g
            error01_g = np.sum( abs(predict_g - Y_test) / 2 ) / N_test
            Eout_g.append(error01_g)
```