Aggregation

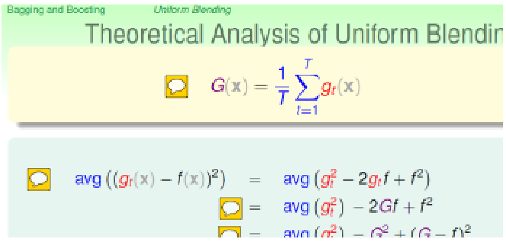
1. ANS [c]

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

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

Assume that each g_t is of test 0/1 error $E_{\mathrm{out}}(g_t) = e_t$. Which of the following is the tightest upper bound of $E_{\mathrm{out}}(G)$? 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] $\frac{1}{12}\sum_{t=1}^{11} e_t$



Bagging and Boosting Uniform Blending

Theoretical Analysis of Uniform Blendir

$$G(\mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T} g_t(\mathbf{x})$$

$$\begin{aligned} \mathrm{avg}\left((g_t(\mathbf{x}) - f(\mathbf{x}))^2\right) &= \mathrm{avg}\left(g_t^2 - 2g_tf + f^2\right) \\ &= \mathrm{avg}\left(g_t^2\right) - 2Gf + f^2 \\ &= \mathrm{avg}\left(g_t^2\right) - G^2 + (G - f)^2 \end{aligned}$$

11 classifiers $\{g_t\}_{t=1}^{11}$

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

$E_{out}(g_t) = e_t = \mathcal{E}[\![g_t \neq f]\!]$

$= \mathcal{E}[\![g_t \cdot f \neq 1]\!]$

求 $E_{out}(G)$ 與 $E_{out}(g_t)$ 的關係

$E_{out}(G) = \mathcal{E}[\![G\cdot f \neq 1]\!]$

$= \mathcal{E}[\![\mathrm{sign}(\sum_{t=1}^{11} g_t)\cdot f \neq 1]\!]$

$= \mathcal{E}[\![\mathrm{sign}(\sum_{t=1}^{11} g_t \cdot f) \neq 1]\!]$

想將 $\underbrace{[\![\mathrm{sign}(\sum_{t=1}^{11} g_t\cdot f)\neq 1]\!]}_{A}$ 換成 $\underbrace{\sum_{t=1}^{11}[\![\mathrm{sign}(g_t\cdot f)\neq 1]\!]}_{B}$

代表算幾個 g_t 出錯

要讓 $A \le \frac{1}{k}B$，此題共 11 個 classifiers

當錯 6 個 or 以上時 $A = 1$，錯 5 個 or 以下時 $A = 0$

	A=0						A=1					
gt錯的個數	0	1	2	3	4	5	6	7	8	9	10	11
B	$\frac{0}{6}$	$\frac{1}{6}$	$\frac{2}{6}$	$\frac{3}{6}$	$\frac{4}{6}$	$\frac{5}{6}$	$\frac{6}{6}$	$\frac{7}{6}$	$\frac{8}{6}$	$\frac{9}{6}$	$\frac{10}{6}$	$\frac{11}{6}$

⇨ k 要取 6 以下才能讓 $A \le \frac{1}{k}B$ 成立

由於求 tightest upper bound 所以取 $k = 6$

$$\leq \mathcal{E}\left(\frac{1}{6}\sum_{t=1}^{11}[\![\operatorname{sign}(gt\cdot f)\neq 1]\!]\right)$$

$$\leq \frac{1}{6}\sum_{t=1}^{11}\mathcal{E}[\![\operatorname{sign}(gt\cdot f)\neq 1]\!]$$

$$\leq \frac{1}{6}\sum_{t=1}^{11} et$$

選 [C]

2. ANS [d]

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.

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

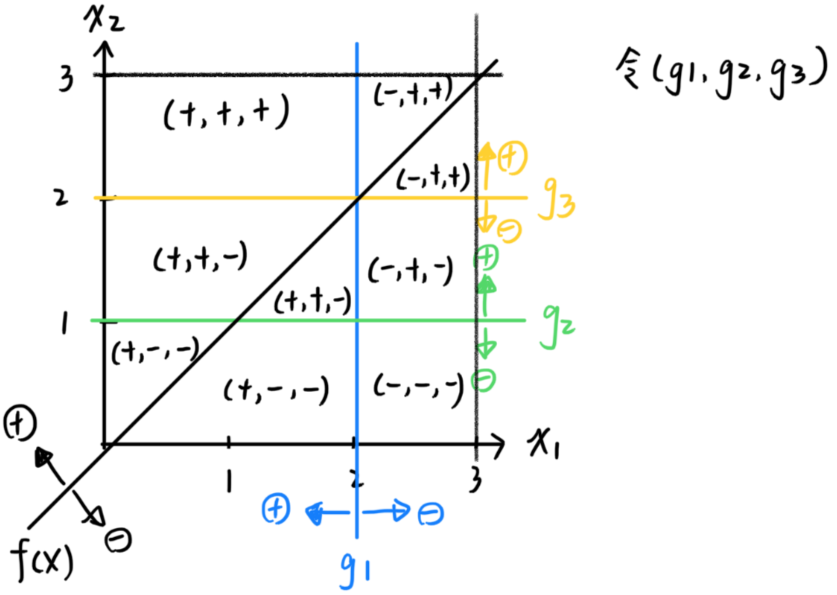
That is,

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

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*)

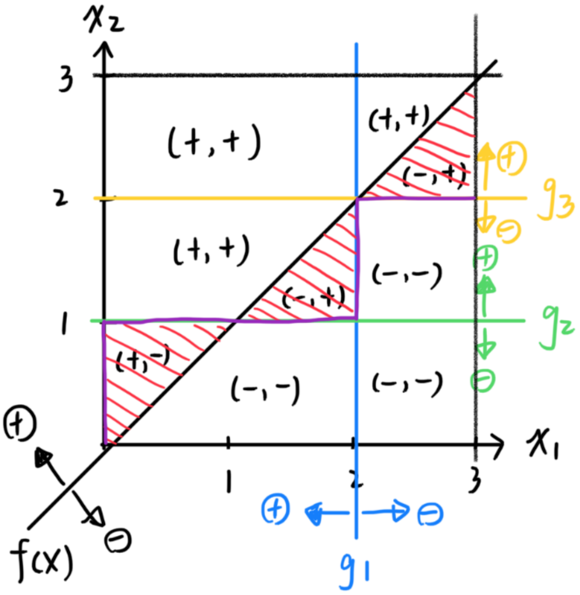
[a] $\frac{6}{18}$
[b] $\frac{5}{18}$
[c] $\frac{4}{18}$
[d] $\frac{3}{18}$
[e] none of the other choices



g_t有α_t, 可選擇哪邊取+/−, 由於要讓E_{out}最小

故要和$f(x)$同號面積越多越好

$\Rightarrow \alpha_1 = -1$, $\alpha_2 = 1 = \alpha_3$



令 (f(x) , 3g)

● 錯的區域

— 為G邊界

$$錯的面積 = \text{smallest } E_{out}(G)$$

$$= \frac{\frac{1\times1}{2}\times 3}{3\times 3} = \frac{\frac{3}{2}}{9} = \frac{3}{18}$$ #

選 [d]

3. ANS [a]

When talking about non-uniform voting in aggregation, we mentioned that α can be viewed as a weight vector learned from any linear algorithm coupled with the following transform:

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

When studying kernel methods, we mentioned that the kernel is simply a computational short-cut for the inner product $(\phi(\mathbf{x}))^T(\phi(\mathbf{x}'))$. 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

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

Define $\phi_{ds}(\mathbf{x}) = \Big(g_{+1,1,2L+1}(\mathbf{x}), g_{+1,1,2L+3}(\mathbf{x}), \ldots, g_{+1,1,2R-1}(\mathbf{x}), \ldots, g_{-1,d,2R-1}(\mathbf{x})\Big)$. What is $K_{ds}(\mathbf{x}, \mathbf{x}') = (\phi_{ds}(\mathbf{x}))^T(\phi_{ds}(\mathbf{x}'))$? 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

$$K_{ds}(x, x') = (\phi_{ds}(x))^T(\phi_{ds}(x'))$$

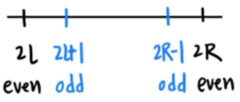
$$= \left(g_{+1,1,2L+1}(x), g_{+1,1,2L+3}(x), \cdots, g_{+1,1,2R-1}(x), \cdots, g_{-1,d,2R-1}(x)\right)$$
$$\cdot \left(g_{+1,1,2L+1}(x'), g_{+1,1,2L+3}(x'), \cdots, g_{+1,1,2R-1}(x'), \cdots, g_{-1,d,2R-1}(x')\right)$$

$g_{s,i,\theta}(x) = s \cdot \text{sign}(x_i - \theta)$

$i \in \{1, 2, \cdots, d\}$

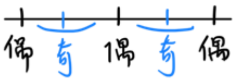
$s \in \{-1, +1\}$

θ is an odd int between $(2L, 2R)$



$\Rightarrow$ $2L+1 \leq \theta \leq 2R-1$

$R_\theta = \{\theta \mid \theta \text{ is odd}, 2L+1 \leq \theta \leq 2R-1\}$



$\Rightarrow$ 奇數個數 = 偶數間隔數

θ 共有 $\frac{2R-2L}{2} = R-L$ 個

$$g_{\substack{-1\\+1},\ \substack{1\\ \vdots\\ d},\ \substack{2L+1\\ \vdots\\ 2R-1}}(x) = s \cdot \text{sign}(x_i - \theta)$$

(above the subscripts: s, i, θ)

$$K_{d_s}(X, X') = (\phi_{d_s}(X))^T(\phi_{d_s}(X'))$$

$$= \left(g_{+1,1,2L+1}(X), g_{+1,1,2L+3}(X), \cdots, g_{+1,1,2R-1}(X), \cdots, g_{-1,d,2R-1}(X)\right)$$

$$\cdot \left(g_{+1,1,2L+1}(X'), g_{+1,1,2L+3}(X'), \cdots, g_{+1,1,2R-1}(X'), \cdots, g_{-1,d,2R-1}(X')\right)$$

$$= \sum_{S=\{-1,+1\}} \sum_{i=1}^{d} \sum_{\theta \in R_\theta} g_{s,i,\theta}(X) \cdot g_{s,i,\theta}(X')$$

$$= \sum_{S=\{-1,+1\}} \sum_{i=1}^{d} \sum_{\theta \in R_\theta} \left[\left(s \cdot \text{sign}(x_i - \theta)\right) \cdot \left(s \cdot \text{sign}(x'_i - \theta)\right)\right]$$

$$= \sum_{i=1}^{d} \sum_{\theta \in R_\theta} \Big[\left(+1 \cdot \text{sign}(x_i - \theta)\right) \cdot \left(+1 \cdot \text{sign}(x'_i - \theta)\right)$$
$$+ \left(-1 \cdot \text{sign}(x_i - \theta)\right) \cdot \left(-1 \cdot \text{sign}(x'_i - \theta)\right)\Big]$$

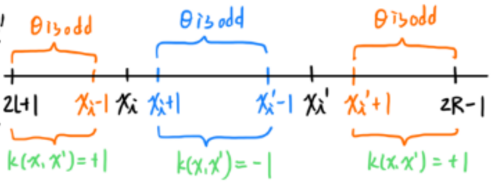
$$= 2 \sum_{i=1}^{d} \sum_{\theta \in R_\theta} \Big[\underbrace{\text{sign}(x_i - \theta) \cdot \text{sign}(x'_i - \theta)}_{\text{令為 } k(x,x')} \Big]$$

$\Big[\ \theta \in R_\theta\ ,\ R_\theta = \{\theta \mid \theta \text{ is odd},\ 2L+1 \le \theta \le 2R-1\}$

假設 $x_i < x_i'$

且input vector

只會是偶數



$$R_{\theta L} = \{\theta \mid \theta \text{ is odd},\ 2L+1 \le \theta \le \min(x_i, x_i') - 1\}$$

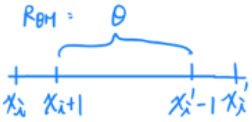
$$R_{\theta M} = \{\theta \mid \theta \text{ is odd},\ \min(x_i, x_i') + 1 \le \theta \le \max(x_i, x_i') - 1\}$$

$$R_{\theta R} = \{\theta \mid \theta \text{ is odd},\ \max(x_i, x_i') + 1 \le \theta \le 2R-1\}\ \Big]$$

$$= 2 \sum_{i=1}^{d} \left[\sum_{\theta \in R_{\theta L}} (+1) + \sum_{\theta \in R_{\theta M}} (-1) + \sum_{\theta \in R_{\theta R}} (+1) \right]$$

$$= 2 \sum_{i=1}^{d} \left[\sum_{\theta \in R_\theta} (+1) - 2 \sum_{\theta \in R_{\theta M}} (-1) \right]$$

$$= 2 \sum_{i=1}^{d} \left[\sum_{\theta \in R_\theta} (+1) + 2 \sum_{\theta \in R_{\theta M}} (+1) \right]$$



θ共有 $\frac{|x_i - x_i'|}{2}$ 個

$$= 2 \sum_{i=1}^{d} \left[(R-L)(+1) + 2 \frac{|x_i - x_i'|}{2} \right]$$

$$= \sum_{i=1}^{d} \left[2(R-L) + 2|x_i - x_i'| \right]$$

$$= 2d(R-L) + 2 \sum_{i=1}^{d} |x_i - x_i'|$$

$\Big[$ norm one : $\|X - X'\|_1 = \sum_{i=1}^{\sim} |x_i - x_i'|$ $\Big]$

$$= 2d(R-L) + 2\|X - X'\|_1$$ ※ 選 [a]

Adaptive Boosting

4. ANS [c]

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

Data : 99% $y_n>0$, 1% $y_n<0$

假設 $N=100$ ⇨ $y_n>0$ 有 99 個 , $y_n<0$ 有 1 個

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

ϵ_t 錯誤率

$$= \frac{\sum_{n:y_n>0} u_n^{(1)} \times \underline{\frac{1}{100}}}{\sum_{n:y_n<0} u_n^{(1)} \times \frac{99}{100}} = \frac{\frac{1}{100} \times 99\, u_n^{(1)}}{\frac{99}{100} \times 1\, u_n^{(1)}}$$

$1-\epsilon_t$

[$u_n^{(1)}$ 可以去掉 ∵ first iteration $u_n^{(1)} = \frac{1}{N}$]

$= 1$ # 選 [c]

5. ANS [c]

For the AdaBoost algorithm introduced in Lecture 12, let $G_t(\mathbf{x}) = \text{sign}\left(\sum_{\tau=1}^{t} \alpha_\tau g_\tau(\mathbf{x})\right)$. 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 $(\mathbf{x}_n, y_n)$

You can assume $0 < \epsilon_t < \frac{1}{2}$ if needed.

[a] 1

[b] 2

[c] 3

[d] 4

[e] 5

- $E_{in}(G_t)$ to $E_{in}(G_{t+1})$

AdaBoost 演算法會讓下一輪的 G_{t+1} 的 E_{in} 表現等於或優於 $E_{in}(G_t)$，因為演算法每一輪都會針對錯誤進行放大專注處理，並用 α_t 來控制新的 g_t，若 g_{t+1} 表現不好在合成 G_{t+1} 時會給較少的權重，故 $E_{in}(G_t) \to E_{in}(G_{t+1})$ 為 non-increasing。

- $E_{out}(G_t)$ to $E_{out}(G_{t+1})$

無法保證運用在 out of sample 表現越來越好，模型過於複雜會有 overfitting 的狀況產生。

⊙ $\sum_{n=1}^{N} u_n^{(t)}$ to $\sum_{n=1}^{N} u_n^{(t+1)}$

根據第6題的結果

$$U_{t+1} = \sum_{n=1}^{N} u_n^{(t+1)} = \sum_{n=1}^{N} u_n^{(t)} \cdot \blacklozenge_t \cdot \epsilon_t$$

$$+ \sum_{n=1}^{N} u_n^{(t)} \cdot \frac{1}{\blacklozenge_t} \cdot (1-\epsilon_t)$$

$$= 2 \sum_{n=1}^{N} u_n^{(t)} \cdot \sqrt{(1-\epsilon_t)\epsilon_t}$$

由於 $0 < \epsilon_t < \frac{1}{2}$ ⇨ $0 < (1-\epsilon_t)\epsilon_t < \frac{1}{4}$

∋ $\frac{1}{2} < 1-\epsilon_t < 1$

⇨ $0 < \sqrt{(1-\epsilon_t)\epsilon_t} < \frac{1}{2}$

⇨ $0 < 2\sqrt{(1-\epsilon_t)\epsilon_t} < 1$

⇨ $\sum_{n=1}^{N} u_n^{(t+1)} = \sum_{n=1}^{N} u_n^{(t)} \cdot \underbrace{2\sqrt{(1-\epsilon_t)\epsilon_t}}_{\text{小於1的數}}$

⇨ $\sum_{n=1}^{N} u_n^{(t+1)} > \sum_{n=1}^{N} u_n^{(t)}$ 為non-increasing。

⊙ $u_n^{(t)}$ to $u_n^{(t+1)}$ when g_t is correct on (x_n, y_n)

• $u_n^{(t)}$ to $u_n^{(t+1)}$ when g_t is incorrect on (x_n, y_n)

$\blacklozenge_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$, $0 < \epsilon_t < \frac{1}{2}$ ⇨ $\blacklozenge_t$ ∋ $\sqrt{\frac{大}{小}}$ ∋ 大於1

incorrect : $u_n^{(t+1)} \leftarrow u_n^{(t)} \cdot \blacklozenge_t$

⇨ $u_n^{(t+1)} > u_n^{(t)}$ ⇨ increase

correct : $u_n^{(t+1)} \leftarrow u_n^{(t)} / \blacklozenge_t$

⇨ $u_n^{(t+1)} < u_n^{(t)}$ ⇨ non-increase

共3個為non-increasing，選[c]

6. ANS [b]

. 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.

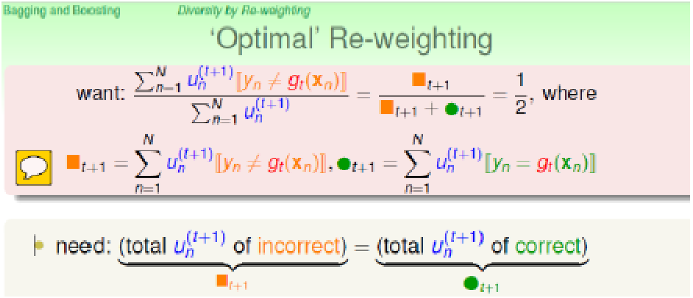
[a] $\sqrt{\epsilon_t(1-\epsilon_t)}$

[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] $\ln\sqrt{\frac{\epsilon_t}{(1-\epsilon_t)}}$



Bagging and Boosting — Diversity by Re-weighting

'Optimal' Re-weighting

want: $\frac{\sum_{n=1}^{N} u_n^{(t+1)} [\![y_n \neq g_t(\mathbf{x}_n)]\!]}{\sum_{n=1}^{N} u_n^{(t+1)}} = \frac{■_{t+1}}{■_{t+1} + ●_{t+1}} = \frac{1}{2}$, where

$■_{t+1} = \sum_{n=1}^{N} u_n^{(t+1)} [\![y_n \neq g_t(\mathbf{x}_n)]\!]$, $●_{t+1} = \sum_{n=1}^{N} u_n^{(t+1)} [\![y_n = g_t(\mathbf{x}_n)]\!]$

- need: $\underbrace{(\text{total } u_n^{(t+1)} \text{ of incorrect})}_{■_{t+1}} = \underbrace{(\text{total } u_n^{(t+1)} \text{ of correct})}_{●_{t+1}}$

$$U_t = \sum_{n=1}^{N} u_n^{(t)} \quad , \quad 0 < \epsilon_t < \frac{1}{2}$$

for each $g_t \Rightarrow \frac{U_{t+1}}{U_t}$

$$\epsilon_t = \frac{\sum_{n=1}^{N} u_n^{(t)} [\![y_n \neq g_t(X_n)]\!]}{\sum_{n=1}^{N} u_n^{(t)}}$$

$$\Rightarrow \sum_{n=1}^{N} u_n^{(t)} \cdot \epsilon_t = \sum_{n=1}^{N} u_n^{(t)} [\![y_n \neq g_t(X_n)]\!]$$

$$U_{t+1} = \sum_{n=1}^{N} u_n^{(t+1)} = \sum_{n=1}^{N} u_n^{(t)} \cdot \blacklozenge_t \overset{\text{incorrect}}{[\![y_n \neq g_t(X_n)]\!]}$$

$$+ \sum_{n=1}^{N} u_n^{(t)} / \blacklozenge_t \overset{\text{correct}}{[\![y_n = g_t(X_n)]\!]}$$

$$= \sum_{n=1}^{N} u_n^{(t)} \cdot \blacklozenge_t \cdot \epsilon_t$$

$$+ \sum_{n=1}^{N} u_n^{(t)} \cdot \frac{1}{\blacklozenge_t} \cdot (1-\epsilon_t)$$

$$= \sum_{n=1}^{N} u_n^{(t)} \cdot \sqrt{(1-\epsilon_t)\epsilon_t}$$

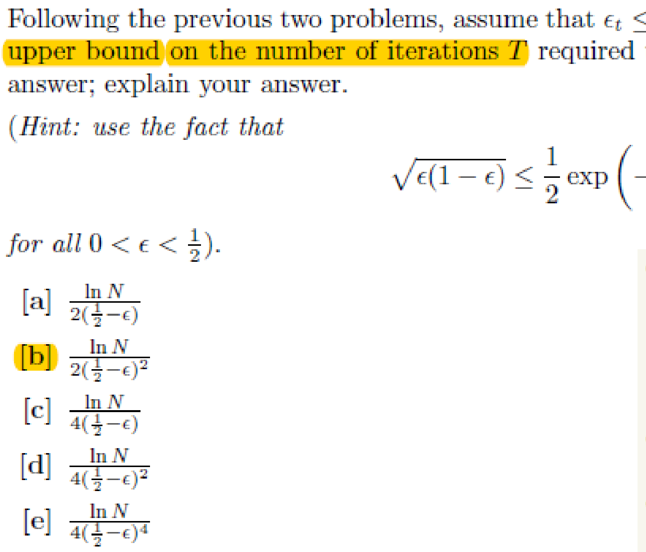
$$+ \sum_{n=1}^{N} u_n^{(t)} \cdot \sqrt{(1-\epsilon_t)\epsilon_t}$$

$$= 2 \sum_{n=1}^{N} u_n^{(t)} \cdot \sqrt{(1-\epsilon_t)\epsilon_t}$$

$$\Rightarrow U_{t+1} = U_t \cdot 2\sqrt{(1-\epsilon_t)\epsilon_t}$$

$$\frac{U_{t+1}}{U_t} = 2\sqrt{(1-\epsilon_t)\epsilon_t}$$ # 選 [b]

7. ANS [b]



Following the previous two problems, assume that $\epsilon_t \le \epsilon < \frac{1}{2}$, 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*

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

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

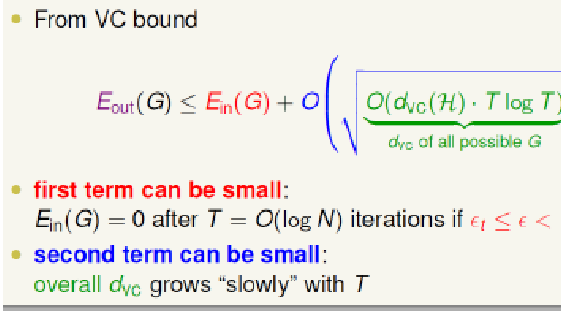
[a] $\frac{\ln N}{2(\frac{1}{2}-\epsilon)}$

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

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

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

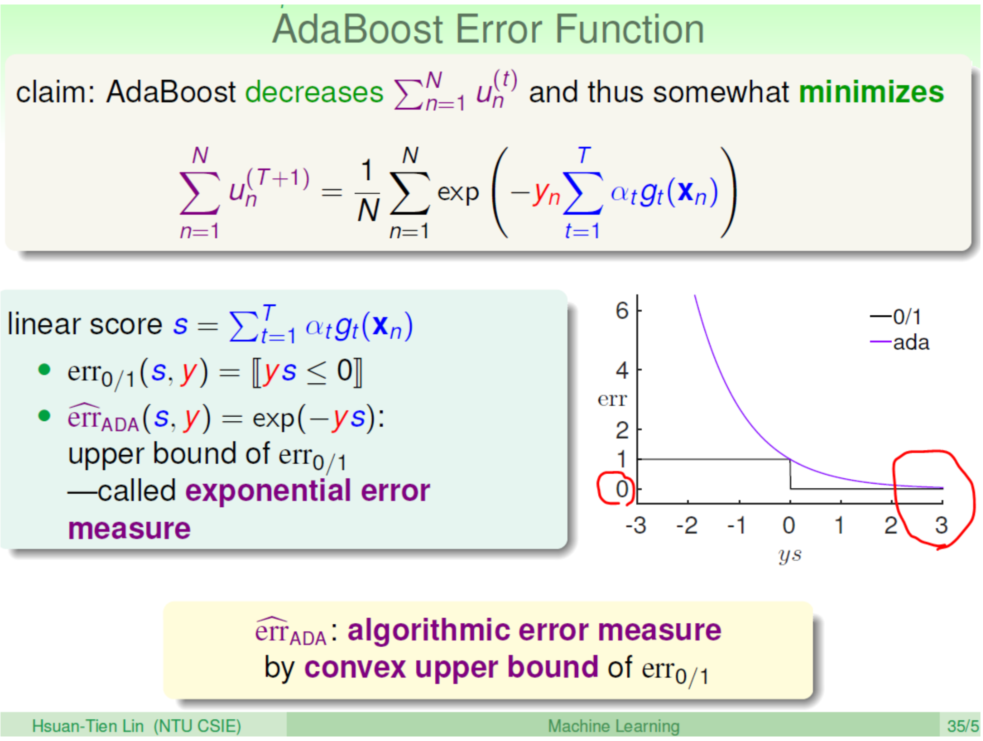
[e] $\frac{\ln N}{4(\frac{1}{2}-\epsilon)^4}$



- From VC bound

$$E_{\text{out}}(G) \le E_{\text{in}}(G) + O\left(\sqrt{\underbrace{O(d_{\text{vc}}(\mathcal{H})\cdot T\log T)}_{d_{\text{vc}}\text{ of all possible } G}}\right.$$

- **first term can be small**:
  $E_{\text{in}}(G) = 0$ after $T = O(\log N)$ iterations if $\epsilon_t \le \epsilon <$
- **second term can be small**:
  overall d_{vc} grows "slowly" with T



AdaBoost Error Function

claim: AdaBoost decreases $\sum_{n=1}^{N} u_n^{(t)}$ and thus somewhat **minimizes**

$$\sum_{n=1}^{N} u_n^{(T+1)} = \frac{1}{N}\sum_{n=1}^{N}\exp\left(-y_n\sum_{t=1}^{T}\alpha_t g_t(\mathbf{x}_n)\right)$$

linear score $s = \sum_{t=1}^{T}\alpha_t g_t(\mathbf{x}_n)$

- $\text{err}_{0/1}(s, y) = [\![ys \le 0]\!]$
- $\widehat{\text{err}}_{\text{ADA}}(s, y) = \exp(-ys)$: upper bound of $\text{err}_{0/1}$ —called **exponential error measure**

$\widehat{\text{err}}_{\text{ADA}}$: **algorithmic error measure** by **convex upper bound** of $\text{err}_{0/1}$

Hsuan-Tien Lin (NTU CSIE) Machine Learning 35/5

由投影片可知

當 0/1 error Ein(G_T)=0，會被E_{ADA} bound 住。

故 Ein(G_T)<= U_{T+1} 。

根據第6題結果得：

$$U_{t+1} = 2\sqrt{(1-\epsilon_t)\epsilon_t} \cdot \sum_{n=1}^{N} u_n^{(t)}$$

由於 $Ein(G_T) \le U_{T+1}$

$$U_{T+1} = U_T \cdot 2\sqrt{(1-\epsilon_T)\epsilon_T}$$

$$= \left(U_{T-1} \cdot 2\sqrt{(1-\epsilon_T)\epsilon_T} \right) \cdot 2\sqrt{(1-\epsilon_T)\epsilon_T}$$

[由於 $\underset{=}{\epsilon_t} \le \epsilon < \frac{1}{2}$]

$$\le U_1 \cdot \left(2\sqrt{(1-\epsilon)\epsilon} \right)^T$$

$$= 1 \cdot \left(2\sqrt{(1-\epsilon)\epsilon} \right)^T$$

[由於 $\sqrt{\epsilon(1-\epsilon)} \le \frac{1}{2} \exp\left(-2\left(\frac{1}{2}-\epsilon\right)^2\right)$]

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

[$(e^2)^T = e^{2T}$]

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

⇨ $U_{T+1} \le \exp\left(-2T\left(\frac{1}{2}-\epsilon\right)^2\right) < \frac{1}{N}$

$$U_{T+1} \le \underline{-2T\left(\frac{1}{2}-\epsilon\right)^2 < -\ln N}$$

$$2T\left(\frac{1}{2}-\epsilon\right)^2 > \ln N$$

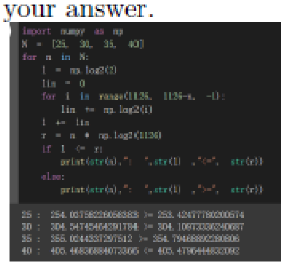
$$T > \frac{\ln N}{2\left(\frac{1}{2}-\epsilon\right)^2}$$ # 選 [b]

Random Forest = Bagging + Decision Tree

8. ANS [d]

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



```
import numpy as np
N = [25, 30, 35, 40]
for n in N:
    l = np.log2(2)
    lin = 0
    for i in range(1126, 1126-n, -1):
        lin += np.log2(i)
    l += lin
    r = n * np.log2(1126)
    if l <= r:
        print(str(n),": ",str(l) ,"<=", str(r))
    else:
        print(str(n),": ",str(l) ,">=", str(r))
```

```
25 :  254.03758226056838 >= 253.42477780200574
30 :  304.54745464291784 >= 304.10973336240687
35 :  355.0244337297512 >= 354.79468892280806
40 :  405.46830884073365 <= 405.47964448330892
```

至少重覆取一次點的机率

$$= 1 - \text{完全沒重覆取點的机率} > \frac{1}{2}$$

$$1 - \frac{N!}{N^{N'}(N-N')!} > \frac{1}{2}$$

$$1 - \frac{N \times (N-1) \times \cdots \times (N-N'+1)}{N^{N'}} > \frac{1}{2}$$

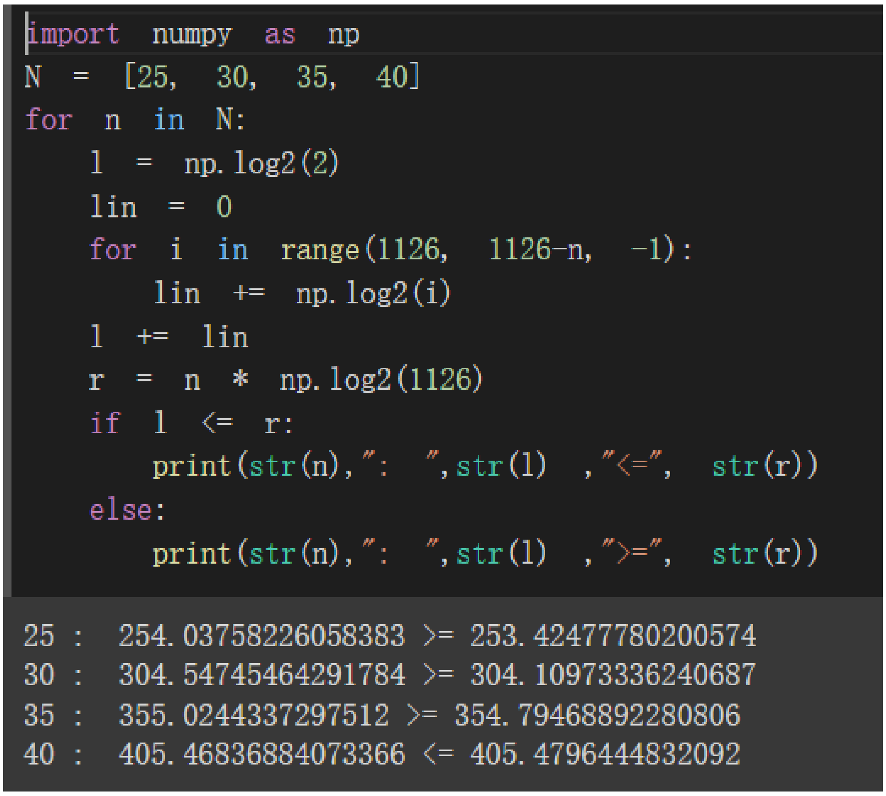
$[N = 1126]$

$$1 - \frac{1126 \times 1125 \times \cdots \times (1126-N'+1)}{1126^{N'}} > \frac{1}{2}$$

$$2 \times 1126 \times 1125 \times \cdots \times (1126-N'+1) < 1126^{N'}$$

$$\Leftrightarrow \log_2 2 + \log_2(1126 \times 1125 \times \cdots \times (1126-N'+1)) < N' \log_2 1126$$

用程式去跑 ↓



```python
import numpy as np
N = [25, 30, 35, 40]
for n in N:
    l = np.log2(2)
    lin = 0
    for i in range(1126, 1126-n, -1):
        lin += np.log2(i)
    l += lin
    r = n * np.log2(1126)
    if l <= r:
        print(str(n),":  ",str(l) ,"<=", str(r))
    else:
        print(str(n),":  ",str(l) ,">=", str(r))
```

```
25 :  254.03758226058383 >= 253.42477780200574
30 :  304.54745464291784 >= 304.10973336240687
35 :  355.0244337297512 >= 354.79468892280806
40 :  405.46836884073366 <= 405.4796444832092
```

選 d。

9. ANS [d]

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%

if N large 根據投影片

$$\left(1-\frac{1}{N}\right)^{N'}=\frac{1}{\left(\frac{N}{N-1}\right)^{N'}}=\frac{1}{\left(1+\frac{1}{N-1}\right)^{N}}\simeq\frac{1}{e}$$

$[N'=2N]$

$$\left(1-\frac{1}{N}\right)^{2N}=\frac{1}{\left(1+\frac{1}{N-1}\right)^{2N}}\approx\frac{1}{e^2}$$

$e\approx 2.71828\ldots$

$\frac{1}{e^2}\approx 0.135335\ldots$

$\approx 13.5\%$ # 選 [d]

10. ANS [b]

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.

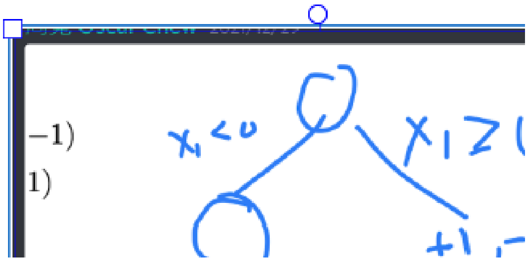
[a] $(1, 1, -1), (-1, 1, -1), (-1, -1, -1)$

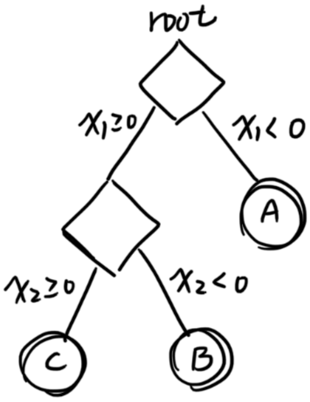
[b] $(1, 1, -1), (-1, 1, -1), (1, -1, -1)$

[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





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

C A A ⇒ No

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

C A B ⇒ Yes #

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

C A B A

⇒ No

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

C A B A

⇒ No

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.

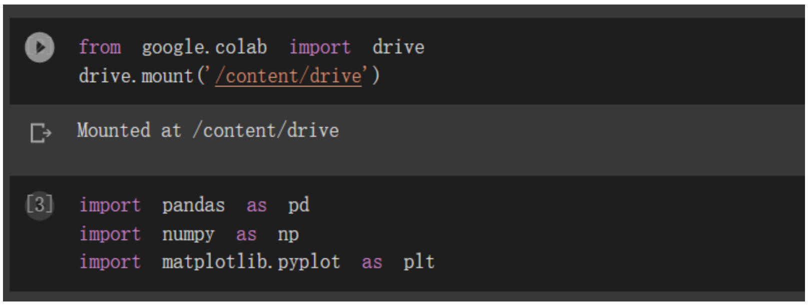
For the decision stump algorithm, please implement the following steps. Any ties can be arbitrarily broken.

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

(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.

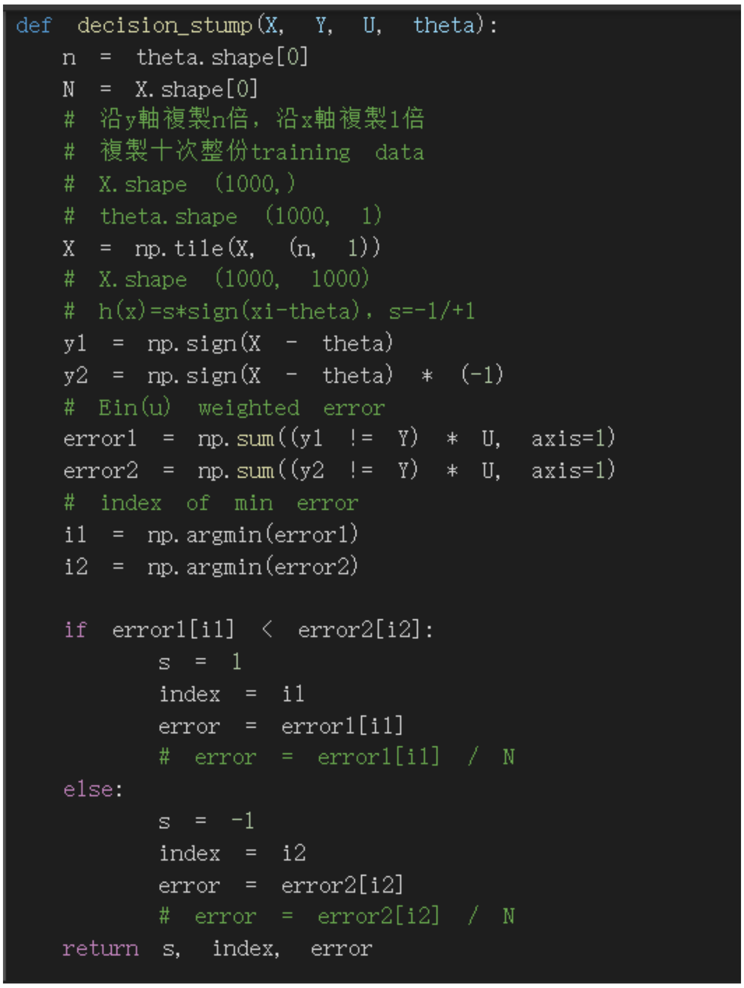
For those interested in algorithms (who isn't? :-)), step 2 can be carried out in $O(N)$ time only!!



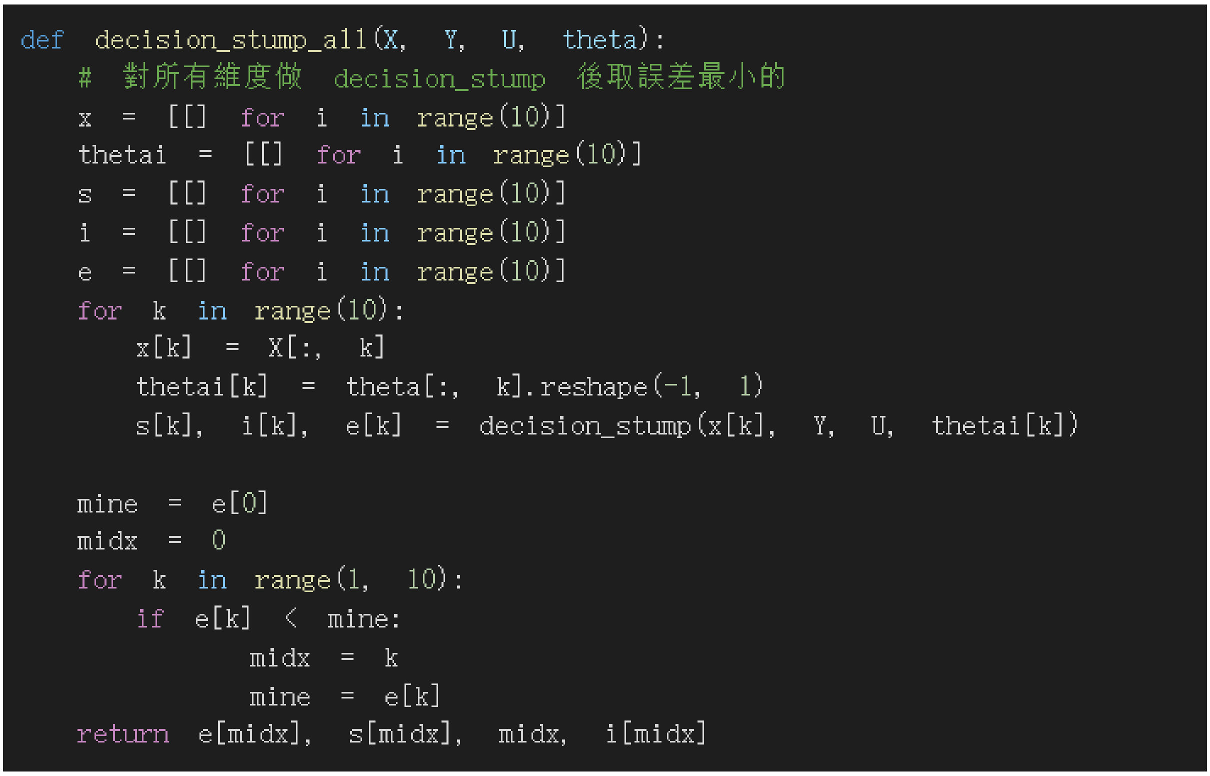
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

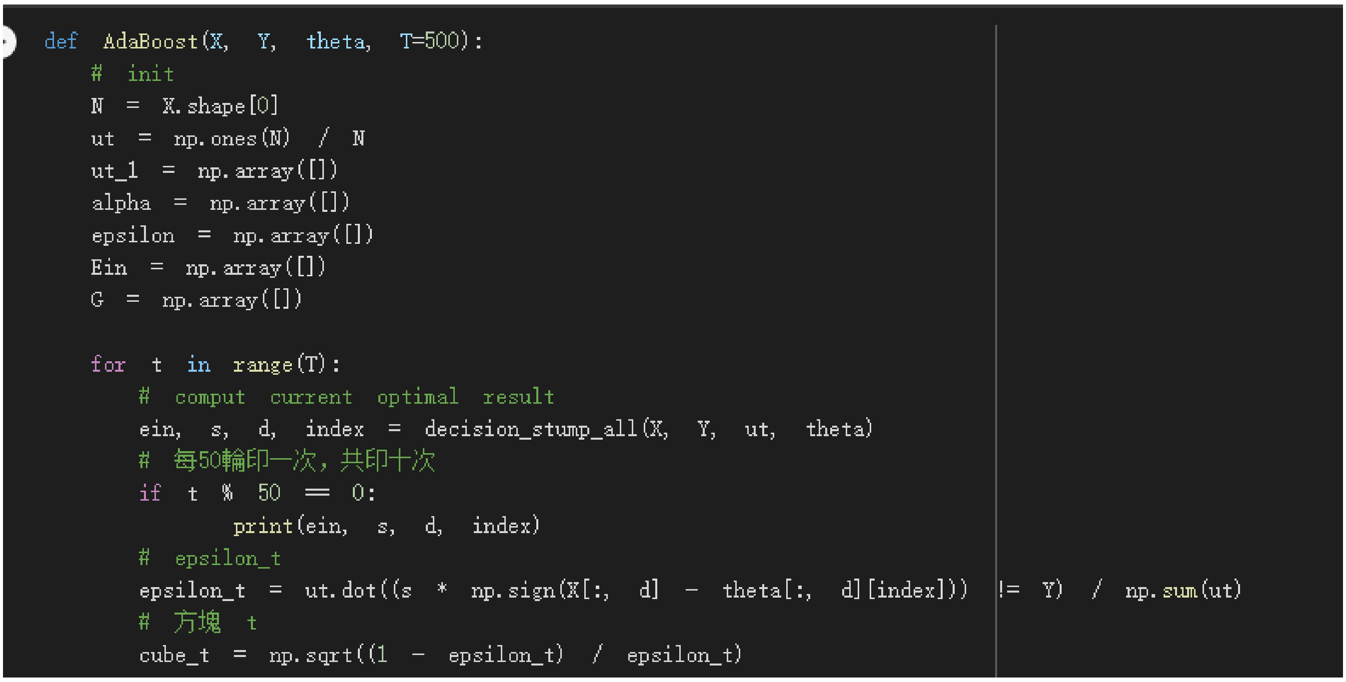
```
[3]
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```



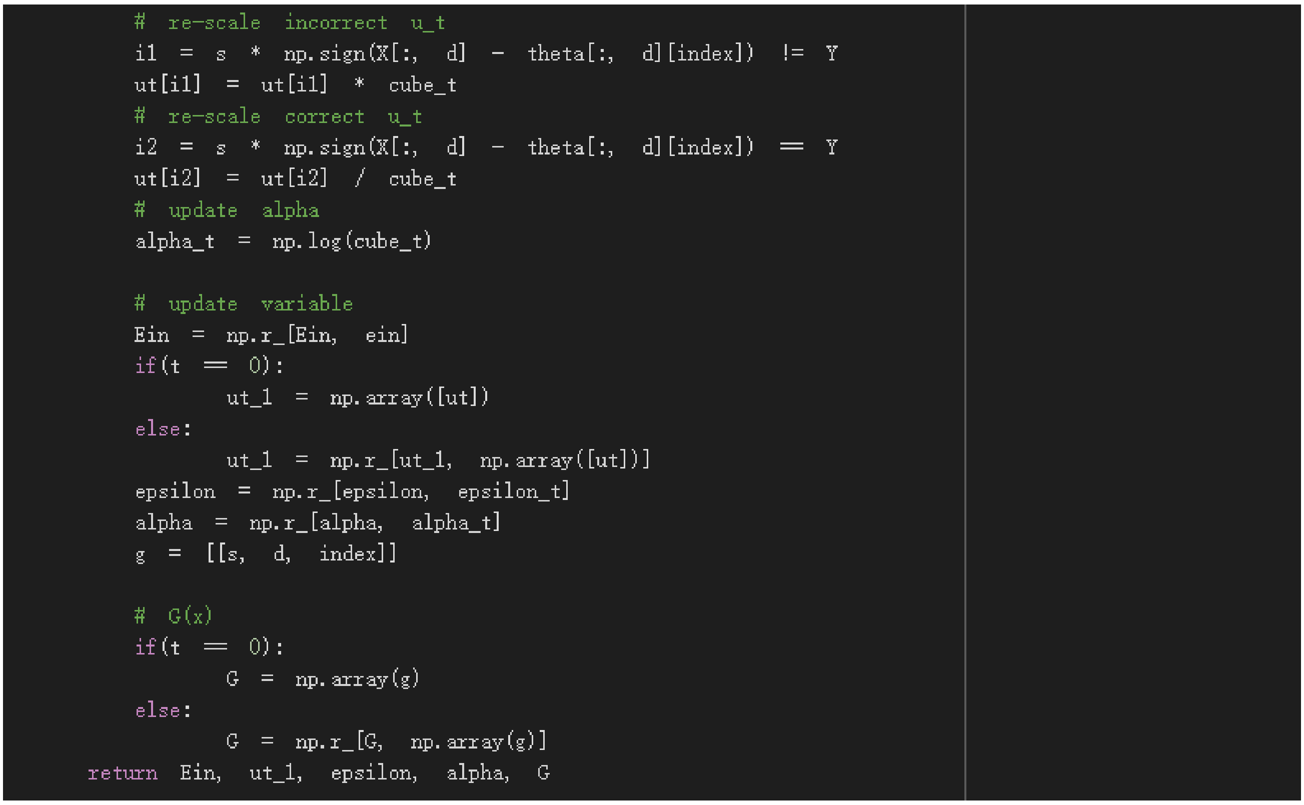
```
def decision_stump(X, Y, U, theta):
    n = theta.shape[0]
    N = X.shape[0]
    # 沿y軸複製n倍，沿x軸複製1倍
    # 複製十次整份training data
    # X.shape (1000,)
    # theta.shape (1000, 1)
    X = np.tile(X, (n, 1))
    # X.shape (1000, 1000)
    # h(x)=s*sign(xi-theta), s=-1/+1
    y1 = np.sign(X - theta)
    y2 = np.sign(X - theta) * (-1)
    # Ein(u) weighted error
    error1 = np.sum((y1 != Y) * U, axis=1)
    error2 = np.sum((y2 != Y) * U, axis=1)
    # index of min error
    i1 = np.argmin(error1)
    i2 = np.argmin(error2)

    if error1[i1] < error2[i2]:
        s = 1
        index = i1
        error = error1[i1]
        # error = error1[i1] / N
    else:
        s = -1
        index = i2
        error = error2[i2]
        # error = error2[i2] / N
    return s, index, error
```



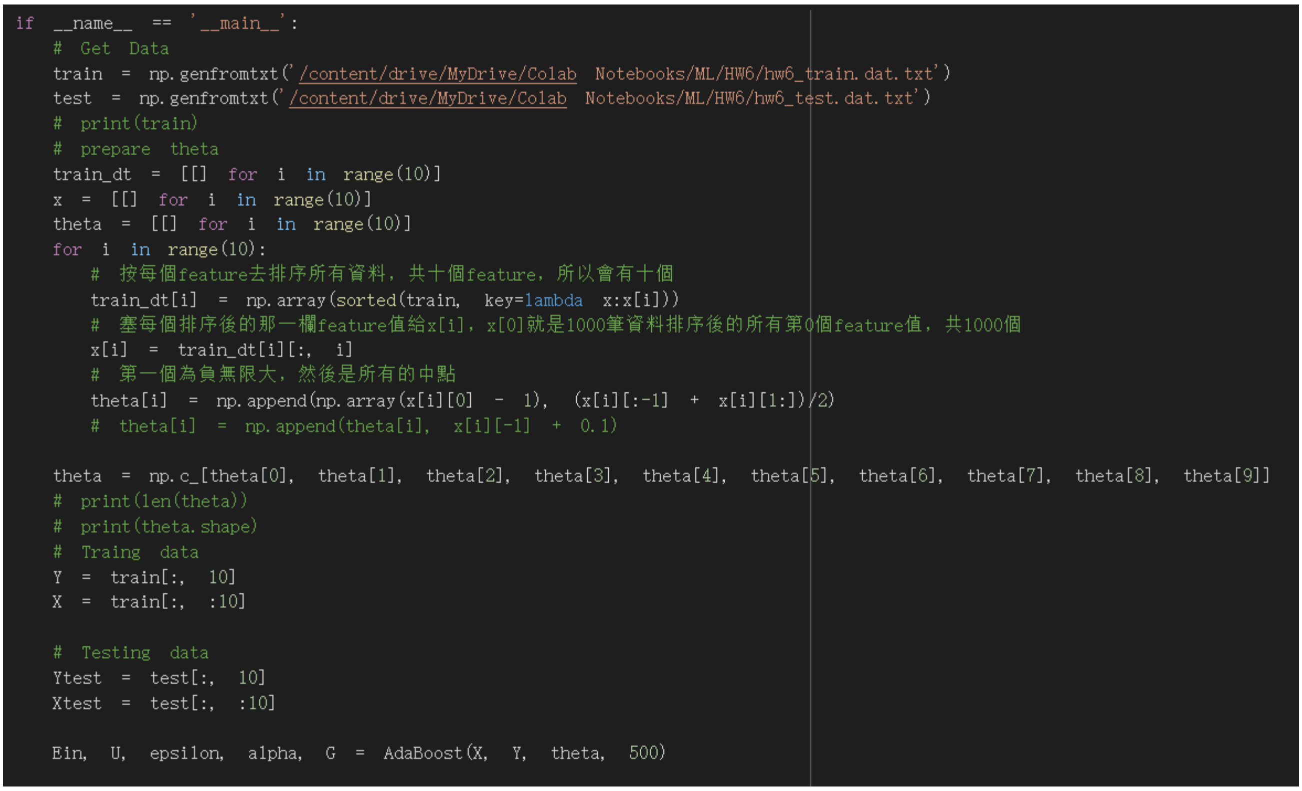
```
def decision_stump_all(X, Y, U, theta):
    # 對所有維度做 decision_stump 後取誤差最小的
    x = [[] for i in range(10)]
    thetai = [[] for i in range(10)]
    s = [[] for i in range(10)]
    i = [[] for i in range(10)]
    e = [[] for i in range(10)]
    for k in range(10):
        x[k] = X[:, k]
        thetai[k] = theta[:, k].reshape(-1, 1)
        s[k], i[k], e[k] = decision_stump(x[k], Y, U, thetai[k])

    mine = e[0]
    midx = 0
    for k in range(1, 10):
        if e[k] < mine:
                midx = k
                mine = e[k]
    return e[midx], s[midx], midx, i[midx]
```



```
def AdaBoost(X, Y, theta, T=500):
    # init
    N = X.shape[0]
    ut = np.ones(N) / N
    ut_1 = np.array([])
    alpha = np.array([])
    epsilon = np.array([])
    Ein = np.array([])
    G = np.array([])

    for t in range(T):
        # comput current optimal result
        ein, s, d, index = decision_stump_all(X, Y, ut, theta)
        # 每50輪印一次，共印十次
        if t % 50 == 0:
                print(ein, s, d, index)
        # epsilon_t
        epsilon_t = ut.dot((s * np.sign(X[:, d] - theta[:, d][index])) != Y) / np.sum(ut)
        # 方塊 t
        cube_t = np.sqrt((1 - epsilon_t) / epsilon_t)
```



```python
        # re-scale incorrect u_t
        i1 = s * np.sign(X[:, d] - theta[:, d][index]) != Y
        ut[i1] = ut[i1] * cube_t
        # re-scale correct u_t
        i2 = s * np.sign(X[:, d] - theta[:, d][index]) == Y
        ut[i2] = ut[i2] / cube_t
        # update alpha
        alpha_t = np.log(cube_t)

        # update variable
        Ein = np.r_[Ein, ein]
        if(t == 0):
            ut_1 = np.array([ut])
        else:
            ut_1 = np.r_[ut_1, np.array([ut])]
        epsilon = np.r_[epsilon, epsilon_t]
        alpha = np.r_[alpha, alpha_t]
        g = [[s, d, index]]

        # G(x)
        if(t == 0):
            G = np.array(g)
        else:
            G = np.r_[G, np.array(g)]
    return Ein, ut_1, epsilon, alpha, G
```

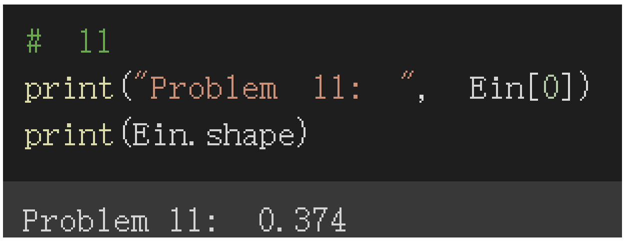


```python
if __name__ == '__main__':
    # Get Data
    train = np.genfromtxt('/content/drive/MyDrive/Colab Notebooks/ML/HW6/hw6_train.dat.txt')
    test = np.genfromtxt('/content/drive/MyDrive/Colab Notebooks/ML/HW6/hw6_test.dat.txt')
    # print(train)
    # prepare theta
    train_dt = [[] for i in range(10)]
    x = [[] for i in range(10)]
    theta = [[] for i in range(10)]
    for i in range(10):
        # 按每個feature去排序所有資料，共十個feature，所以會有十個
        train_dt[i] = np.array(sorted(train, key=lambda x:x[i]))
        # 塞每個排序後的那一欄feature值給x[i]，x[0]就是1000筆資料排序後的所有第0個feature值，共1000個
        x[i] = train_dt[i][:, i]
        # 第一個為負無限大，然後是所有的中點
        theta[i] = np.append(np.array(x[i][0] - 1), (x[i][:-1] + x[i][1:])/2)
        # theta[i] = np.append(theta[i], x[i][-1] + 0.1)

    theta = np.c_[theta[0], theta[1], theta[2], theta[3], theta[4], theta[5], theta[6], theta[7], theta[8], theta[9]]
    # print(len(theta))
    # print(theta.shape)
    # Traing data
    Y = train[:, 10]
    X = train[:, :10]

    # Testing data
    Ytest = test[:, 10]
    Xtest = test[:, :10]

    Ein, U, epsilon, alpha, G = AdaBoost(X, Y, theta, 500)
```

11. ANS [c]

(*) 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

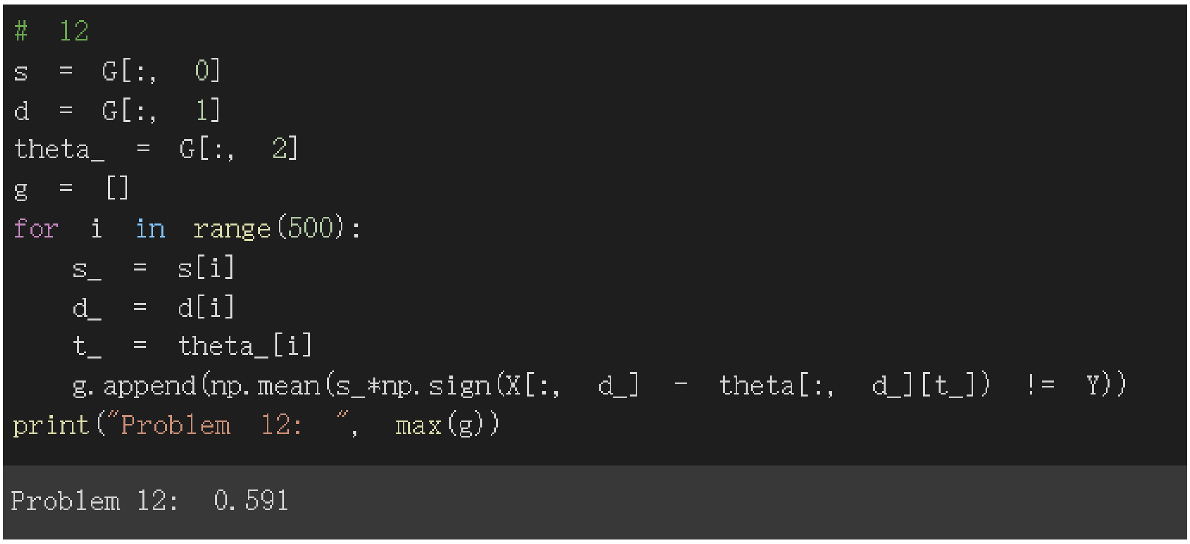


```
# 11
print("Problem 11: ", Ein[0])
print(Ein.shape)

Problem 11:  0.374
```

12. ANS [e]

(*) 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



```
# 12
s = G[:, 0]
d = G[:, 1]
theta_ = G[:, 2]
g = []
for i in range(500):
    s_ = s[i]
    d_ = d[i]
    t_ = theta_[i]
    g.append(np.mean(s_*np.sign(X[:, d_] - theta[:, d_][t_]) != Y))
print("Problem 12: ", max(g))

Problem 12:  0.591
```

13. ANS [d]

(*) 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.

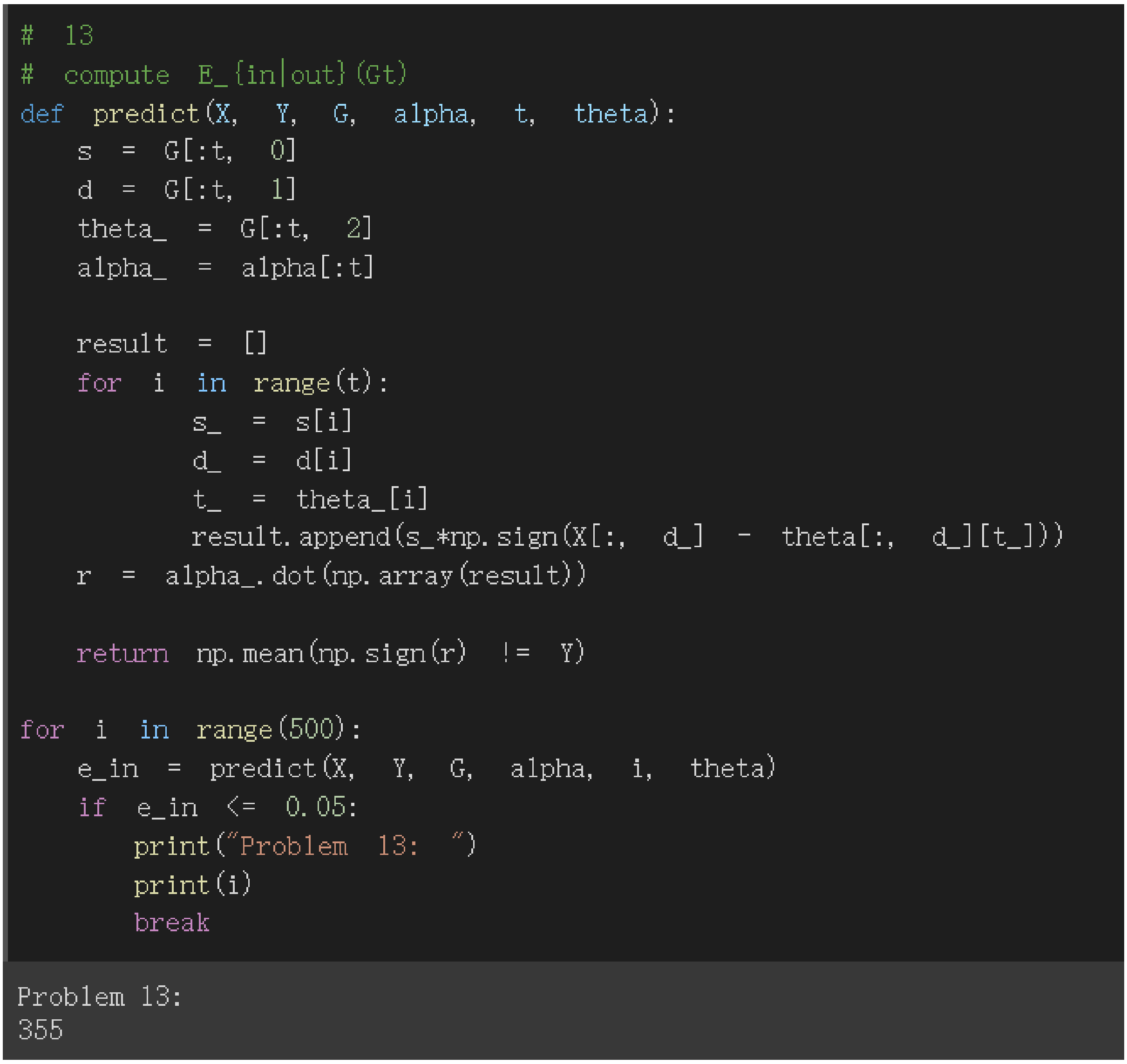
[a] 60

[b] 160

[c] 260

[d] 360

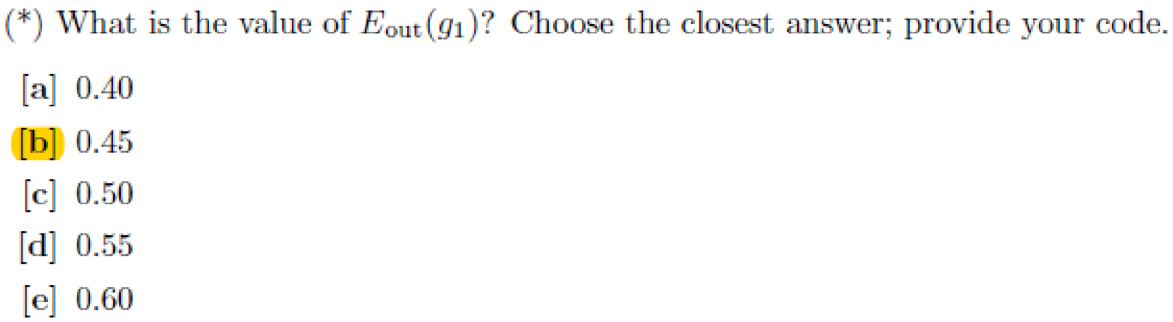
[e] 460



```
# 13
# compute E_{in|out}(Gt)
def predict(X, Y, G, alpha, t, theta):
    s = G[:t, 0]
    d = G[:t, 1]
    theta_ = G[:t, 2]
    alpha_ = alpha[:t]

    result = []
    for i in range(t):
        s_ = s[i]
        d_ = d[i]
        t_ = theta_[i]
        result.append(s_*np.sign(X[:, d_] - theta[:, d_][t_]))
    r = alpha_.dot(np.array(result))

    return np.mean(np.sign(r) != Y)

for i in range(500):
    e_in = predict(X, Y, G, alpha, i, theta)
    if e_in <= 0.05:
        print("Problem 13: ")
        print(i)
        break
```

```
Problem 13:
355
```

14. ANS [b]


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

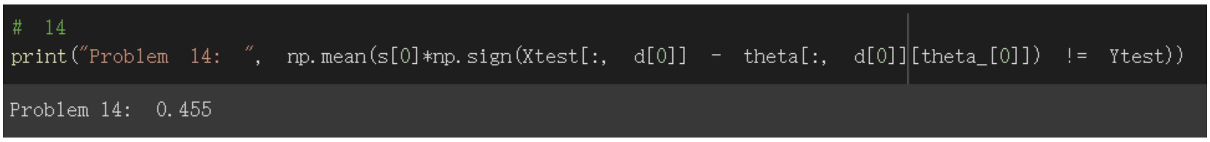
[a] 0.40

[b] 0.45

[c] 0.50

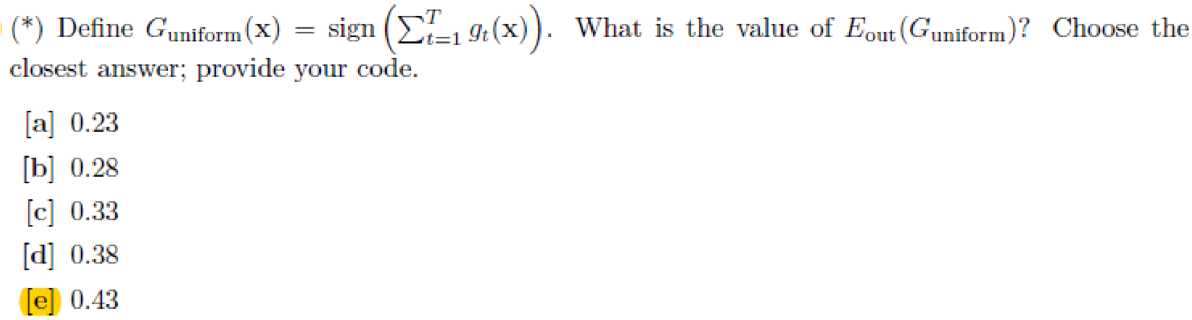
[d] 0.55

[e] 0.60


```
# 14
print("Problem 14: ", np.mean(s[0]*np.sign(Xtest[:, d[0]] - theta[:, d[0]][theta_[0]]) != Ytest))
```

```
Problem 14: 0.455
```

15. ANS [e]


(*) 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.

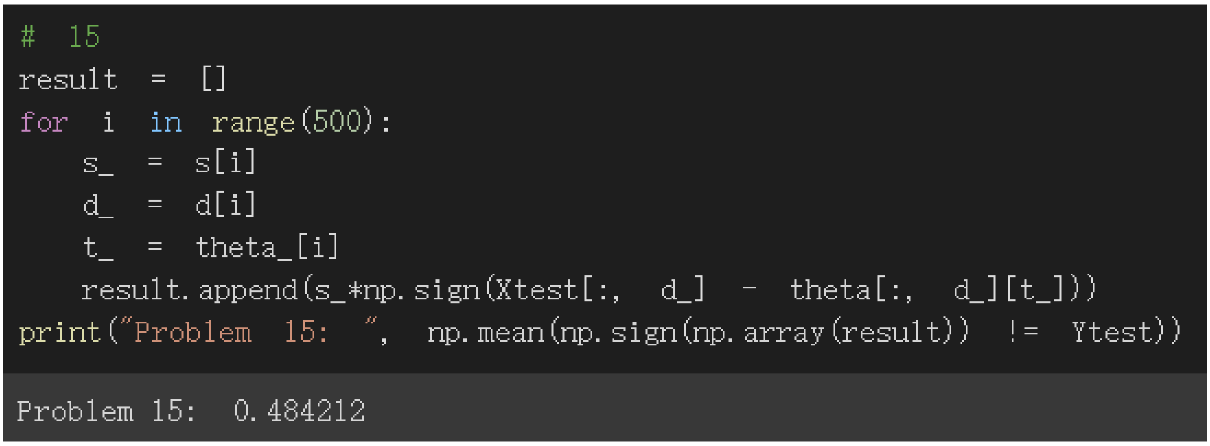
[a] 0.23

[b] 0.28

[c] 0.33

[d] 0.38

[e] 0.43


```
# 15
result = []
for i in range(500):
    s_ = s[i]
    d_ = d[i]
    t_ = theta_[i]
    result.append(s_*np.sign(Xtest[:, d_] - theta[:, d_][t_]))
print("Problem 15: ", np.mean(np.sign(np.array(result)) != Ytest))
```

```
Problem 15: 0.484212
```

16. ANS [b]

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

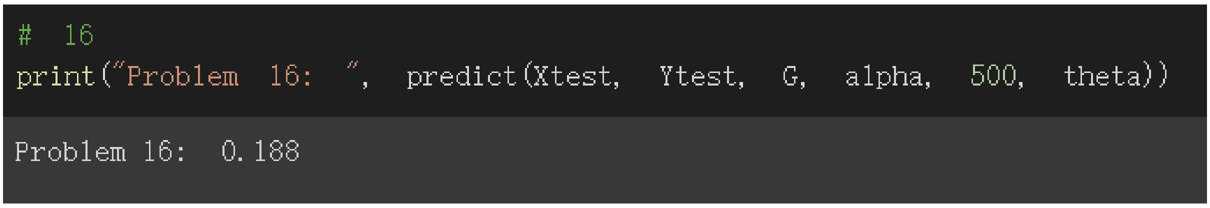
[a] 0.14

[b] 0.18

[c] 0.22

[d] 0.26

[e] 0.30



```
# 16
print("Problem 16: ", predict(Xtest, Ytest, G, alpha, 500, theta))
```

```
Problem 16:  0.188
```