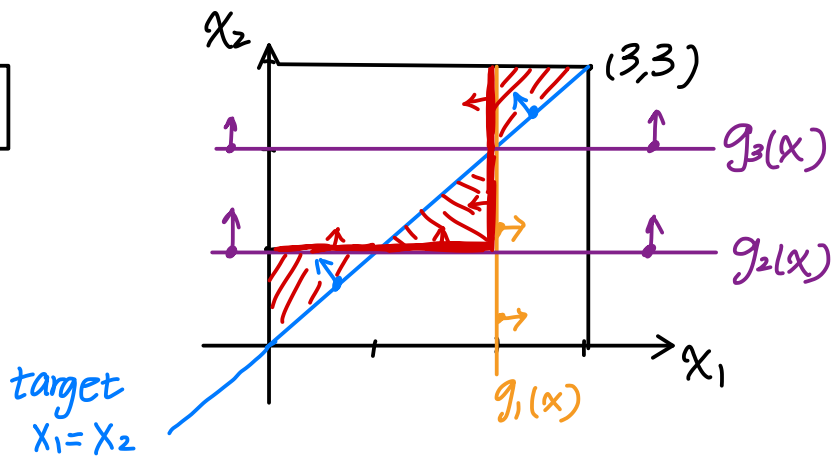
P1.C 只有當1半以上的 $g_t(x)$ 犯錯時, $G(x)$ 才會一起犯錯

而要求 tightest bound 所以考慮 6個 $g_t(x)$ 犯錯的情形

$$\Rightarrow \frac{\overbrace{e_1+e_2+e_3+e_4+e_5+e_6}^{\text{make mistake}}+\overbrace{\boxed{\cdots\cdots+e_{11}}}^{\text{don't make mistake}}}{6} = E_{out}(G) \quad (=0)$$

$$\Rightarrow \frac{\sum_{n=1}^{11} e_i}{6} = E_{out}(G) \quad \text{故選[C]}$$

P2. D



紅色是optimal的decision boundary，紅色斜線是做錯的部份

$$\Rightarrow E_{out} = \frac{\frac{1}{2} \times 3}{9} = \frac{3}{18}$$ 故選D

P3, A 先只看 $S=1$, ith feature 的情況, WLOG 設 $x_i=8$, $x_i'=4$

2L ⋯ 1	2	3	4	5	6	7	8	9	10 ⋯ 2R	
⋯ θ_j	θ_{j+1}	x_i'	θ_{j+2}		θ_{j+3}	x_i	θ_{j+4}		⋯	
⋯ 1	1		1	1			-1		⋯	$=\text{sign}(x_i-\theta_j)$
⋯ 1	1		-1	-1			-1		⋯	$=\text{sign}(x_i'-\theta_j)$
⋯ 1	1		-1	-1			1		⋯	$=\text{sign}(x_i-\theta_j)\cdot\text{sign}(x_i'-\theta_j)$

異號時為負, 同號時為正

根據上述情況: Number of threshold $=\frac{2R-2L}{2}=R-L$

Number of different sign $=\frac{|x_i-x_i'|}{2}$

因此 $\sum_{j=2L+1}^{2R-1}\text{sign}(x_i-\theta_j)\cdot\text{sign}(x_i'-\theta_j)=R-L-2\cdot\frac{|x_i-x_i'|}{2}$

$=R-L-|x_i-x_i'|$

現在推廣到 N 個 feature

$\sum_{k=1}^{d}\sum_{j=2L+1}^{2R-1}\text{sign}(x_i-\theta_j)\cdot\text{sign}(x_i'-\theta_j)=d(R-L)-\|\vec{x}-\vec{x}'\|_1$

再推廣到 $S=\{-1,1\}$ 的情況

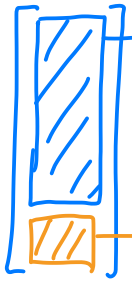
$\sum_{S=-1,1}\sum_{k=1}^{d}\sum_{j=2L+1}^{2R-1}\text{sign}(x_i-\theta_j)\cdot\text{sign}(x_i'-\theta_j)\cdot S^2=2(d(R-L)-\|\vec{x}-\vec{x}'\|_1)$

$\Rightarrow \phi_{ds}(\vec{x})\cdot\phi_{ds}(\vec{x}')=2d(R-L)-2\|\vec{x}-\vec{x}'\|_1$ #

故選 A

P4.C By Silde 212. P34

$$\varepsilon_1 = \frac{\sum_{n=1}^{N} u_n^{(1)} [\![y_n \neq g_t(x_n)]\!]}{\sum_{n=1}^{N} u_n^{(1)}} = \frac{1}{100}, \quad \blacklozenge_1 = \sqrt{\frac{1-\varepsilon_t}{\varepsilon_t}} = \sqrt{\frac{1-\frac{1}{100}}{\frac{1}{100}}} = \sqrt{99}$$

$\vec{u}^{(2)} =$ 

correct $\Rightarrow u_2 = \frac{u_1}{\blacklozenge_1} = \frac{\frac{1}{N}}{\sqrt{99}} = \frac{1}{\sqrt{99}N}$

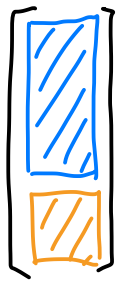
incorrect $\Rightarrow u_2 = u_1 \cdot \blacklozenge_1 = \frac{\sqrt{99}}{N}$

$$\frac{\sum_{n: y_n > 0} u_n^{(2)}}{\sum_{n: y_n < 0} u_n^{(2)}} = \frac{\frac{1}{\sqrt{99}N} \cdot N \cdot \frac{99}{100}}{\frac{\sqrt{99}}{N} \cdot N \cdot \frac{1}{100}} = \frac{\frac{\sqrt{99}}{100}}{\frac{\sqrt{99}}{100}} = 1$$

故選C

P5, B 證明 $\sum_{n=1}^{N} u_n^{(t)}$ to $\sum_{n=1}^{N} u_n^{(t+1)}$ 是 non-increasing

$u_{all}^{(t)} = \sum_{n=1}^{N} u_n^{(t)}$, where $\varepsilon_t = \frac{u_{BAD}^{(t)}}{u_{all}^{(t)}}$, where $\blacklozenge_t = \sqrt{\frac{1-\varepsilon_t}{\varepsilon_t}}$



$u_{good}^{(t)} = \sum_{n=1}^{N} u_n^{(t)} \cdot (1-\varepsilon_t)$

$u_{BAD}^{(t)} = \sum_{n=1}^{N} u_n^{(t)} \cdot \varepsilon_t$

$$\sum_{n=1}^{N} u_n^{(t+1)} = u_{good}^{(t)} \cdot \frac{1}{\blacklozenge_t} + u_{BAD}^{(t)} \cdot \blacklozenge_t$$

$$= u_{all}^{(t)} \cdot (1-\varepsilon_t) \cdot \sqrt{\frac{\varepsilon_t}{1-\varepsilon_t}} + u_{all}^{(t)} \cdot \varepsilon_t \cdot \sqrt{\frac{1-\varepsilon_t}{\varepsilon_t}}$$

$$= u_{all}^{(t)} \left[\sqrt{\varepsilon_t (1-\varepsilon_t)} + \sqrt{\varepsilon_t (1-\varepsilon_t)} \right]$$

$$= u_{all}^{(t)} \cdot \left(2\sqrt{\varepsilon_t (1-\varepsilon_t)} \right)$$

因此，如果能 proof $2\sqrt{\varepsilon_t(1-\varepsilon_t)} \leq 1$，$\sum_{n=1}^{N} u_n^{(t)}$ 到 $\sum_{n=1}^{N} u_n^{(t+1)}$ 就是 non-increasing.

Proof ⇒ 找 $\max_{\varepsilon_t} 2\sqrt{\varepsilon_t(1-\varepsilon_t)}$

$$\frac{\partial 2\sqrt{\varepsilon_t(1-\varepsilon_t)}}{\partial \varepsilon_t} = 0$$

$$\Rightarrow 2 \cdot \frac{1}{2} (\varepsilon_t (1-\varepsilon_t))^{-\frac{1}{2}} \cdot (1-\varepsilon_t - \varepsilon_t) = 0$$

$$= \frac{1-2\varepsilon_t}{\sqrt{\varepsilon_t - \varepsilon_t^2}} = 0$$，當 $\varepsilon_t = \frac{1}{2}$，發生最大值，而 $2\sqrt{\frac{1}{2} \cdot \frac{1}{2}} = 1$

最大值也不超過 1，故得証 non-increasing.

後頁續

證明若 g_t correct 則 $u_n^{(t)}$ to $u_n^{(t+1)}$ non-increasing

By 213 p33:

$$u_n^{(t+1)} = u_n^{(t)} \cdot \frac{1}{\blacklozenge_t}, \text{ where } \blacklozenge_t = \sqrt{\frac{1-\varepsilon_t}{\varepsilon_t}}$$

假設 $\varepsilon_t \leq \varepsilon < \frac{1}{2}$，$\varepsilon_t$ 的 upper bound 小於 $\frac{1}{2}$

則 $\blacklozenge_t > 1$

⇒顯然 $u_n^{(t+1)}$ 比 $u_n^{(t)}$ 小，因為 $u_n^{(t)}$ 會乘上一個永遠小於 1 的數字

P6.B 結果已在P5推導过

$$U_t = \sum_{n=1}^{N} u_n^{(t)} \quad , \text{where } \varepsilon_t = \frac{u_{BAD}^{(t)}}{u_{all}^{(t)}} \quad , \text{where } \blacklozenge_t = \sqrt{\frac{1-\varepsilon_t}{\varepsilon_t}}$$

$$u_{good}^{(t)} = \sum_{n=1}^{N} u_n^{(t)} \cdot (1-\varepsilon_t)$$

$$u_{BAD}^{(t)} = \sum_{n=1}^{N} u_n^{(t)} \cdot \varepsilon_t$$

$$U_{t+1} = u_{good}^{(t)} \cdot \frac{1}{\blacklozenge_t} + u_{BAD}^{(t)} \cdot \blacklozenge_t$$

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

$$= U_t \left[\sqrt{\varepsilon_t (1-\varepsilon_t)} + \sqrt{\varepsilon_t (1-\varepsilon_t)} \right]$$

$$= U_t \cdot \left(2\sqrt{\varepsilon_t (1-\varepsilon_t)} \right)$$

$$\Rightarrow \frac{U_{t+1}}{U_t} = 2\sqrt{\varepsilon_t (1-\varepsilon_t)} \quad \#$$

P7.B By Silde 43 P35 $\widehat{Err}_{ADA}(y,s)$ 是 $\widehat{Err}_{0/1}(y,s)$ 的 upper bound.

已知 $E_{in}(G_T) \le U_{T+1} = \widehat{Err}_{ADA}(y,s) \times \frac{1}{N}$

By 第六題結論 $U_{T+1} = U_T \cdot 2 \cdot \sqrt{\varepsilon_T(1-\varepsilon_T)}$

$= U_1 \cdot 2^T \cdot \sqrt{\varepsilon_1\varepsilon_2\cdots\varepsilon_T(1-\varepsilon_1)(1-\varepsilon_2)\cdots(1-\varepsilon_T)}$

因為 $\vec{u}_1 = [\frac{1}{N}, \frac{1}{N}, \cdots \frac{1}{N}]$，故 $U_1 = 1$
（共 N 項）

$\Rightarrow E_{in}(G_T) \le 2^T \cdot \sqrt{\varepsilon_1\varepsilon_2\cdots\varepsilon_T(1-\varepsilon_1)(1-\varepsilon_2)\cdots(1-\varepsilon_T)}$

因為要求 tightest upper bound，所以要考慮最糟的情況

也就是 $\varepsilon_1 = \varepsilon_2 = \cdots = \varepsilon_T = \varepsilon$，where ε 是每輪 ε_t 的 upper bound.

$\Rightarrow E_{in}(G_T) \le 2^T \sqrt{\varepsilon^T(1-\varepsilon)^T}$

$\Rightarrow E_{in}(G_T) \le (4\varepsilon(1-\varepsilon))^{\frac{T}{2}}$

代入題目的 hint:

$\sqrt{\varepsilon(1-\varepsilon)} \le \frac{1}{2}\exp(-2(\frac{1}{2}-\varepsilon)^2)$

$\Rightarrow E_{in}(G_T) \le (4\varepsilon(1-\varepsilon))^{\frac{T}{2}} \le \exp(-2T(\frac{1}{2}-\varepsilon)^2)$

由 E_{in} 是離散的值，最小的 E_{in} 就是在整个 data 只犯一個錯誤 i.e. $E_{in} = \frac{1}{N}$

因此可知：$E_{in}(G_T) < \frac{1}{N} \Leftrightarrow E_{in}(G_T) \le 0$，因此：

$\Rightarrow \exp(-2T(\frac{1}{2}-\varepsilon)^2) = \frac{1}{N}$（當滿足此條件時會發生 tightest upper bound）

$\Rightarrow 2T(\frac{1}{2}-\varepsilon)^2 = \ln N$

$\Rightarrow T = \frac{\ln N}{2(\frac{1}{2}-\varepsilon)^2}$ ※

P8.D

$N' = 2$

$$1 - \left(\frac{1126}{1126} \times \frac{1125}{1126} \times \frac{1124}{1126} \times \cdots \times \frac{1126-N'+1}{1126}\right)$$

$$\Rightarrow 1 - \frac{\frac{1126!}{(1126-N')!}}{1126^{N'}}$$

用程式找最小 N' 使得上述機率為 0.5 以上

```python
N_p=0
while True:
    p = 1
    for i in range(N_p):
        p *= ((1126-i)/1126)

    print(N_p)
    print( f"Propability = {1-p}" )

    if 1-p > 0.5:
        break
    N_p+=1
```

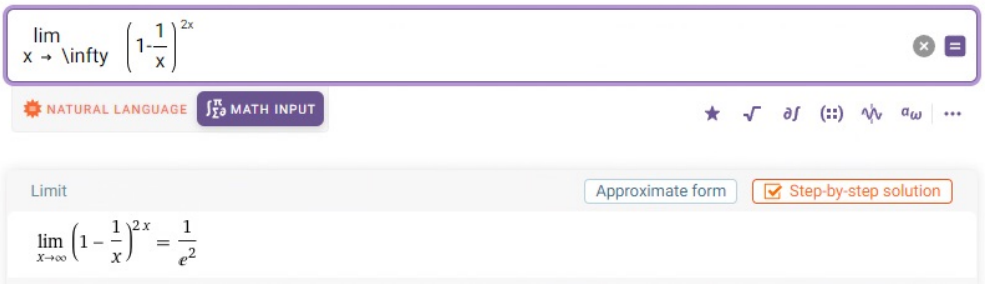
```
29
Propability = 0.3048652453841566
30
Propability = 0.322768360733943
31
Propability = 0.34081183247282554
32
Propability = 0.35895999694293423
33
Propability = 0.3771778300671137
34
Propability = 0.3954310552960526
35
Propability = 0.4136862454558521
36
Propability = 0.4319109181104216
37
Propability = 0.45007362410333884
38
Propability = 0.4681440289951474
39
Propability = 0.48609298716405
40
Propability = 0.503892608390162
(base)spiderkiller@LAPTOP-1EP27GVK:~
 $
```

$N' = 40$ 時，propability 為 0.5038，故選 D

P9. D

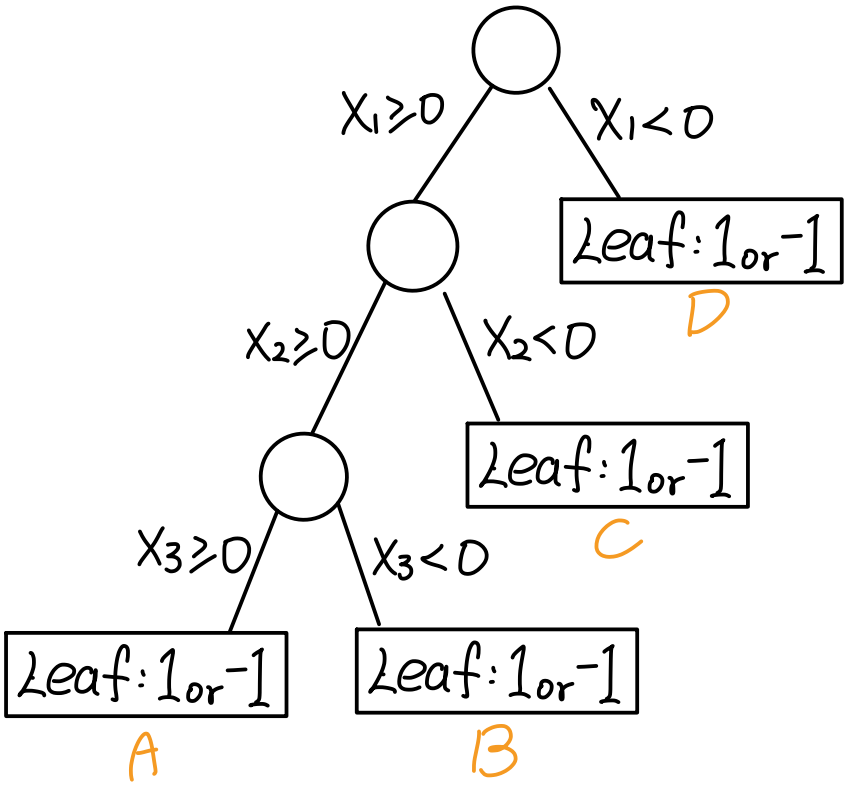
$$\lim_{N\to\infty}\left(1-\frac{1}{N}\right)^{2N}=\frac{1}{e^2}=0.13533\ \#$$ 故选D

By Wolframe:



P10. B Decision Tree 如下圖，一共有4種 leaf，ABCD，能決定 output是 $1_{or}-1$

要看能不能 shatter input，要確保每個 input 都落在不同的 Leaf



選項B中 $(1, 1, -1) \Rightarrow B$

$(-1, 1, -1) \Rightarrow D$

$(1, -1, -1) \Rightarrow C$

3個 input 的 Leaf 都不同

選項B能被 shatted. #

P11.C

Adaboot train

```
import math
from numpy import log # this is ln
import pickle
INT_MAX = 99999999999999999999
INT_MIN = -99999999999999999999

train_data_x = []
train_data_y = []
with open("hw6_train.dat", 'r') as f:
    for line in f.readlines():
        train_data_x.append([float(ele) for ele in line.split("\n")[0].split(" ")[:-1] ])
        train_data_y.append(int(float(line.split("\n")[0].split(" ")[-1])))
print(f"train_data_x[0] = {train_data_x[0]}")
print(f"train_data_y[0] = {train_data_y[0]}")


test_data_x = []
test_data_y = []
with open("hw6_test.dat", 'r') as f:
    for line in f.readlines():
        test_data_x.append([ float(ele) for ele in line.split("\n")[0].split(" ")[:-1] ])
        test_data_y.append( int(float(line.split("\n")[0].split(" ")[-1])) )
print(f"test_data_x[0] = {test_data_x[0]}")
print(f"test_data_y[0] = {test_data_y[0]}")


N_FEATURE = len(train_data_x[0])
N_TRAIN_DATA = len(train_data_x)
N_TEST_DATA = len(test_data_x)
print(f"Number fo features = {N_FEATURE}")
print(f"Number of train data = {N_TRAIN_DATA}")
print(f"Number of test data = {N_TEST_DATA}")

u = [] # importance of example
for i in range(N_TRAIN_DATA):
    u.append(1/N_TRAIN_DATA) # TODO not very sure

# Pre-process training data
sort_feature = []
for f_idx in range(N_FEATURE):
    feature = []
    for n_idx in range(N_TRAIN_DATA):
        feature.append(train_data_x[n_idx][f_idx])
    sort_feature.append(sorted(feature))
```

後頁續

```python
theta_list = [] # theta_list[feature_idx][theta_idx]
for f_idx in range(N_FEATURE):
    t_list = []
    for n_idx in range(N_TRAIN_DATA):
        if n_idx == 0:
            left = INT_MIN
            right = sort_feature[f_idx][n_idx]
        else:
            left = sort_feature[f_idx][n_idx-1]
            right = sort_feature[f_idx][n_idx]

        # Calcuate midpoint
        t_list.append( (left + right)/2 )

    theta_list.append(t_list)
# print(theta_list)
# print(sort_feature[])

def sign(x):
    if x >= 0:
        return 1
    else:
        return -1

def decision_stump():
    best = (INT_MAX, None, None, None) # [w_e, s, i, theta]
    for f_idx in range(N_FEATURE):
        for theta in theta_list[f_idx]:
            for s in (-1, 1):
                # Calcuate E_in # Weight Error
                w_e = 0 # Weight Error
                for data_i in range(N_TRAIN_DATA):
                    pred = s*sign( train_data_x[data_i][f_idx] - theta )
                    # If make mistake, update w_e
                    if pred != train_data_y[data_i]:
                        w_e += u[data_i] # /N_TRAIN_DATA # TODO I think this is optional
                # print(f"w_e = {w_e}")
                if w_e < best[0]: # This is current best
                    best = (w_e, s, f_idx, theta)
    return best
```

後頁續

```
T = 500
GT_LIST = []
for t in range(T):
    # Get gt by decision stump
    w_e, s, i , theta = decision_stump()

    # calculate et
    e_t = w_e/(sum(u))
    print(f"e_t = {e_t}")
    # caculate dia_t(diamond_t)
    dia_t = math.sqrt((1-e_t)/e_t)
    # Update u
    for data_i in range(N_TRAIN_DATA):
        pred = s*sign( train_data_x[data_i][i] - theta )

        if pred == train_data_y[data_i]: # If this is correct exmaple
            u[data_i] /= dia_t
        else:
            u[data_i] *= dia_t
    # print(f"sum(u) = {sum(u)}")

    # Calcuate alpha_t
    alpha_t = log(dia_t)
    #
    GT_LIST.append( (s, i, theta, alpha_t) )
    print(f"(s, i, theta, alpha_t) = {(s, i, theta, alpha_t)}")

    # Calculate Ein
    E_in = 0
    for data_i in range(N_TRAIN_DATA):
        accumulate_vote = 0
        for s, i, theta, alpha_t in GT_LIST:
            accumulate_vote += alpha_t*s*sign( train_data_x[data_i][f_idx] - theta )
        pred = sign(accumulate_vote)
        if pred != train_data_y[data_i]:
            E_in += 1/N_TRAIN_DATA
    print(f"E_in = {E_in}")

    # Calculate E_out
    E_out = 0
    for data_i in range(N_TEST_DATA):
        accumulate_vote = 0
        for s, i, theta, alpha_t in GT_LIST:
            accumulate_vote += alpha_t*s*sign( test_data_x[data_i][f_idx] - theta )
        pred = sign(accumulate_vote)
        if pred != test_data_y[data_i]:
            E_out += 1/N_TEST_DATA
    print(f"E_out = {E_out}")

print(f"GT_LIST = {GT_LIST}")

with open('adaboost.model', 'wb') as f:
    pickle.dump(GT_LIST, f)
```

後頁續

Adaboost Predict

```python
import pickle
INT_MAX = 99999999999999999999
INT_MIN = -99999999999999999999

# Load GT_LIST
with open('adaboost.model', 'rb') as f:
    GT_LIST = pickle.load(f)
# print(GT_LIST)

# Load training data
train_data_x = []
train_data_y = []
with open("hw6_train.dat", 'r') as f:
    for line in f.readlines():
        train_data_x.append([float(ele) for ele in line.split("\n")[0].split(" ")[:-1] ])
        train_data_y.append(int(float(line.split("\n")[0].split(" ")[-1])))
print(f"train_data_x[0] = {train_data_x[0]}")
print(f"train_data_y[0] = {train_data_y[0]}")

# Load testing data
test_data_x = []
test_data_y = []
with open("hw6_test.dat", 'r') as f:
    for line in f.readlines():
        test_data_x.append([ float(ele) for ele in line.split("\n")[0].split(" ")[:-1] ])
        test_data_y.append( int(float(line.split("\n")[0].split(" ")[-1])) )
print(f"test_data_x[0] = {test_data_x[0]}")
print(f"test_data_y[0] = {test_data_y[0]}")

#
N_FEATURE = len(train_data_x[0])
N_TRAIN_DATA = len(train_data_x)
N_TEST_DATA = len(test_data_x)
print(f"Number fo features = {N_FEATURE}")
print(f"Number of train data = {N_TRAIN_DATA}")
print(f"Number of test data = {N_TEST_DATA}")

def sign(x):
    if x >= 0:
        return 1
    else:
        return -1

# p11
# Calcuate E_in (0/1 error)
E_in = 0
s, f_idx, theta, alpha_t = GT_LIST[0] # (-1, 9, 0.44824087255)
for data_i in range(N_TRAIN_DATA):
    pred = s*sign( train_data_x[data_i][f_idx] - theta )
    if pred != train_data_y[data_i]:
        E_in += 1/N_TRAIN_DATA
print(f"p11_ans = {E_in}")
```

P12,E

```python
# p12
E_in_max = INT_MIN
for gt in GT_LIST:
    E_in = 0
    s, f_idx, theta, alpha_t = gt
    for data_i in range(N_TRAIN_DATA):
        pred = s*sign( train_data_x[data_i][f_idx] - theta )
        if pred != train_data_y[data_i]:
            E_in += 1/N_TRAIN_DATA
    if E_in > E_in_max:
        E_in_max = E_in
print(f"p12_ans = {E_in_max}")
```

P13, D

```python
# p13
for t in range(len(GT_LIST)):
    E_in = 0
    for data_i in range(N_TRAIN_DATA):
        # Start Adaboost prediction
        accumulate_vote = 0
        for gt_idx in range(t+1):
            s, f_idx, theta, alpha_t = GT_LIST[gt_idx]
            accumulate_vote += alpha_t*s*sign( train_data_x[data_i][f_idx] - theta )
        pred = sign(accumulate_vote)
        if pred != train_data_y[data_i]:
            E_in += 1/N_TRAIN_DATA
    print(f"t = {t+1}, Ein = {E_in}")

    #
    if E_in <= 0.05:
        print(f"p13_ans = {t+1}")
        break
```

P14, B

```python
# P14
E_out = 0
s, f_idx, theta, alpha_t = GT_LIST[0] # (-1, 9, 0.44824087255)
for data_i in range(N_TEST_DATA):
    pred = s*sign( test_data_x[data_i][f_idx] - theta )
    if pred != test_data_y[data_i]:
        E_out += 1/N_TEST_DATA
print(f"p14_ans = {E_out}")
```

P15, A

```python
# p15
E_out = 0
for data_i in range(N_TEST_DATA):
    # Start Adaboost prediction
    accumulate_vote = 0
    for s, f_idx, theta, alpha_t in GT_LIST:
        accumulate_vote += s*sign( test_data_x[data_i][f_idx] - theta ) # uniform
    pred = sign(accumulate_vote)
    if pred != test_data_y[data_i]:
        E_out += 1/N_TEST_DATA
print(f"p15_ans = {E_out}")
```

P16_B

```python
# p16
E_out = 0
for data_i in range(N_TEST_DATA):
    # Start Adaboost prediction
    accumulate_vote = 0
    for s, f_idx, theta, alpha_t in GT_LIST:
        accumulate_vote += alpha_t*s*sign( test_data_x[data_i][f_idx] - theta ) # uniform
    pred = sign(accumulate_vote)
    if pred != test_data_y[data_i]:
        E_out += 1/N_TEST_DATA
print(f"p15_ans = {E_out}")
```