HW1 Solutions

● Graded

2 Hours, 56 Minutes Late

Student

Chih-Hao Liao

Total Points

70 / 100 pts

Question 1

Problem 2

20 / 20 pts

✔ **– 0 pts** Correct

– 5 pts One minor mistake

– 5 pts Steps are a bit unclear.

– 10 pts One major mistake or two minor mistakes

– 15 pts Some reasonable effort but fundamentally wrong

– 20 pts Wrong answer

– 20 pts Choose wrong page

– 20 pts 1.No solution
2.Totally wrong solution
3.Almost missing solution

– 2 pts Penalties

Question 2

Problem 4

20 / 20 pts

✔ **– 0 pts** Correct

– 5 pts Minor errors

– 10 pts Lack of clarity

– 12 pts Major errors

– 20 pts Fundamentally wrong

– 20 pts Blank

– 2 pts Penalty for not selecting the corresponding pages

– 0 pts Click here to replace this description.

Question 3

Problem 9

10 / 20 pts

– **0 pts** Correct

✔ – **5 pts** Without explaining why calculating the proportion of area equal to E_{out} , i.e. you need to mention $\mathbf{x}$ is uniform distribution in your ansewer

– **5 pts** Without explaining why the region you choose can represent E_{out}, i.e. you at least need to point out x in this region has $f(x) = -1,\ h_1(x) = +1$, etc

✔ – **5 pts** A minor mistake exists in the answer

– **10 pts** A major mistake / several minor mistakes exist in the answer

– **15 pts** Did not provide detailed explanation or calculation, including drawing a graph without saying which region you choose to calculate.

– **15 pts** Just providing simulation method and result without writing down derivation steps this question required

– **15 pts** Some reasonable effort

– **20 pts** Wrong answer

– **20 pts** Did not show the steps

– **10 pts** Did not select the corresponding page

– **20 pts** No solution

– **2 pts** Penalty for forgetting to select pages on grade scope.

you should write P[(h_1(x)!=f(x))/area] not P[(h_1(x) \bigcap f(x))/area]

Question 4

Problem 11

20 / 20 pts

✔ – **0 pts** Correct, Steps need clear and nessesary.

– **5 pts** The answer is correct but still a minor mistake, using incorrect notation: E_{out}, E_{in}

– **5 pts** The answer is correct, but you didn't explain clear why $\epsilon^* = \epsilon/4$ or minor mistake during derive ϵ^*.

– **10 pts** Correct answer, mistake during calculation the number of N

– **15 pts** One of 2 following:
1. Correct answer, but use exclusion or unclear solution.
2. Correct answer, but shows a part of the solution because of not choosing entire pages which content solution.

– **20 pts** One of 3 following:
1. The answer is incorrect.
2. You chose the wrong page.
3. You didn't provide the solving process.

– **2 pts** penalties

Question 5

Problem 12 — 0 / 20 pts

+ 20 pts Correct

+ 5 pts Correctly define the probability of the event (not directly say BAD D)

+ 5 pts find that the difference between each |c_i / N - p_i| is \epsilon / 2

+ 5 pts Applying Hoeffding correctly to bound the probability of the estimate value and the expected value.

+ 5 pts Define the worst case to multiply the calculated probability bound with M (not directly apply multiple-bin hoeffding)

− 3 pts without explanation

✔ + 0 pts empty page or wrong answer

Question 6

Coding Part — **0** / 0 pts

6.1 **Problem 13** — **0** / 0 pts

✔ + 0 pts Correct

− 160 pts Without source code

− 10 pts Unclear submission.

− 16 pts penalty for forgetting to select pages on gradescope

Question assigned to the following page: 1

HTML_2023_HW1

tags: Personal

Basics of Machine Learning

P1

[A]

Machine learning is the process of training a model to learn patterns and relationships from data to make predictions or take action on new data. A voice assistant requires understanding natural language and context from various users, recognizing speech patterns in the language, and then generating an appropriate response based on that understanding. Therefore, this is an ideal use case for machine learning algorithms such as NLP and speech recognition.

[B]

Schrödinger's cat is a hypothetical cat, which may be considered simultaneously both alive and dead, while it is unobserved in a closed box, as a result of its fate is linked to a random subatomic event that may or may not occur. It's not a well-defined problem that has a deterministic solution, hence, it's not a suitable task for machine learning.

[C]

Searching for the shortest path to exit a maze is a well-defined algorithmic problem that can be solved using various graph search algorithms such as DFS, BFS, Dijkstra's algorithm, and A star algorithm, which doesn't require machine learning.

[D]

Machine learning is suitable for generating images, but it requires massive amounts of data and training while generating an image of Zeus that matches his actual facial look is not a suitable task for machine learning since Zeus doesn't have an actual facial look and it requires extensive knowledge of Greek mythology and art styles.
As a result, we should choose $[a]$ as our solution.

P2

Let learning rate equals to $\eta_{n(t)}$. In order to find the $\eta_{n(t)}$ that $y_{n(t)}w_{t+1}^{T}x_{n(t)} > 0$, we can first find the $L_{n(t)} = \eta_{n(t)}$ that make the equation equal to zero.

Question assigned to the following page: 1

$$
\begin{aligned}
y_{n(t)} w_{t+1}^T x_{n(t)} &= 0 = w_{t+1}^T \cdot x_{n(t)} \text{ since } y_{n(t)} \in \{-1, 1\} \\
(w_t + y_{n(t)} w_t^T x_{n(t)} L_{n(t)}) \cdot x_{n(t)} &= 0 \\
w_t \cdot x_{n(t)} &= -L_{n(t)} y_{n(t)} w_t^T x_{n(t)} \cdot x_{n(t)} \\
w_t^T x_{n(t)} &= -\frac{L_{n(t)} \|x_{n(t)}\|^2}{y_{n(t)}} \text{ since } y_{n(t)} \in \{-1, 1\} \\
L_{n(t)} &= \frac{-y_{n(t) w_t^T x_{n(t)}}}{\|x_{n(t)}\|^2}
\end{aligned}
$$

Since $L_{n(t)}$ exactly set $y_{n(t)} w_{t+1}^T x_{n(t)} = 0$, the real $\eta_{n(t)}$ must be bigger than $L_{n(t)}$ to strongly ensure that w_{t+1}^T is correct on $(x_{n(t)}, y_{n(t)})$. It is straightforward to see that $\left\lfloor (\frac{-y_{n(t) w_t^T x_{n(t)}}}{\|x_{n(t)}\|^2} + 1) \right\rfloor = \lfloor L_{n(t)} + 1 \rfloor > L_{n(t)}$, therefore, this is the correct answer. Actually, the expression shows the smallest integer that can make $y_{n(t)} w_{t+1}^T x_{n(t)} > 0$.

As a result, we should choose $[d]$ as our solution.

P3

Since the data is linear separable, there exists a perfect w_f such that $y_n = \text{sign}\ (w_f^T z_n), \|w_f\| = 1$

$$
\begin{aligned}
w_f^T w_{t+1} &= w_f^T (w_t + y_{n(t)} z_{n(t)}) \\
&\geq w_f^T w_t + \underbrace{\min_n y_n w_f^T z_n}_{\rho > 0}
\end{aligned}
$$

PLA starts from $w_0 = 0$, it's trivial that $w_f^T w_T \geq \rho$

As for the length $\|w_T\|$

$$
\begin{aligned}
\|w_{t+1}\|^2 &= \|w_t + y_{n(t)} z_{n(t)}\|^2 \\
&= \|w_t\|^2 + 2 y_{n(t)} w_t^T z_{n(t)} + \|y_{n(t)} z_{n(t)}\|^2 \\
&\Rightarrow \text{sign}\ (w_t^T z_{n(t)}) \neq y_{n(t)} \Leftrightarrow y_{n(t)} w_t^T z_{n(t)} \leq 0 \\
&\leq \|w_t\|^2 + 0 + \|y_{n(t)} z_{n(t)}\|^2 \\
&\leq \|w_t\|^2 + \max_n \|y_n z_n\|^2 \\
&\leq \|w_t\|^2 + \underbrace{\max_n \|z_n\|^2}_{R^2}
\end{aligned}
$$

PLA starts from $w_0 = 0$, it's trivial that $\|w_T\|^2 \leq R^2$

Combine the equations above, we have

Question assigned to the following page: 2

$$
\begin{aligned}
1 &\geq \frac{w_f^T}{\|w_f\|}\frac{w_T}{\|w_T\|} \geq \frac{T\rho}{\|w_f\|\sqrt{T}R} \\
T &\leq \left(\frac{\|w_f\|R}{\rho}\right)^2 \\
\because z_n \leftarrow \frac{x_n}{\|x_n\|} &\therefore R^2 = 1 \text{ since } z_n \in \{0,1\} \\
T &\leq \left(\frac{\|w_f\|}{\rho}\right)^2 = \left(\min_n \frac{\|w_f\|}{y_n w_f^T z_n}\right)^2 = \frac{1}{\rho_z^2}
\end{aligned}
$$

As a result, we should choose $[c]$ as our solution.

P4

From the previous question, we have

$$
\begin{aligned}
U = \frac{1}{\rho_z^2} &= \left(\frac{\sqrt{\max_n \|z_n\|^2}\|w_f\|}{\min_n y_n w_f^T z_n}\right)^2 \\
U_{\text{orig}} &= \left(\frac{\sqrt{\max_n \|x_n\|^2}\|w_f\|}{\min_n y_n w_f^T x_n}\right)^2
\end{aligned}
$$

Since $\min_n y_n w_f^T x_n > 0$ and $\min_n y_n w_f^T z_n > 0$, the denominator part can be ignored. Take the numerator part into consideration and take $\|w_f\| = 1$, then we have

$$
\begin{aligned}
U &\Rightarrow \max_n \|z_n\|^2 = \max_n \left\|\frac{x_n}{\|x_n\|}\right\|^2 \in \{0,1\} \\
U_{\text{orig}} &\Rightarrow \max_n \|x_n\|^2 \in \{0,\infty\} \\
&\therefore U \leq U_{\text{orig}}
\end{aligned}
$$

As a result, we should choose $[b]$ as our solution.

P5

- Train examples by PLA
  1. Initialize $w_0 = [0,0,0]$
  2. $x_0 = [1,-2,2], y_0 = -1, \text{sign}(w_0^T x_0) = \text{sign}(0) = 0$
     - $w_1 = w_0 + y_0 x_0 = [0,0,0] + (-1)[1,-2,2] = [-1,2,-2]$
  3. $x_1 = [1,-1,2], y_1 = -1, \text{sign}(w_1^T x_1) = \text{sign}(-7) = -1$
     - $w_2 = w_1 = [-1,2,-2]$ since it is correct classified.
  4. $x_2 = [1,2,0], y_2 = 1, \text{sign}(w_2^T x_2) = \text{sign}(3) = 1$
     - $w_3 = w_2 = [-1,2,-2]$ since it is correct classified.

No questions assigned to the following page.

5. $x_3 = [1, -1, 0], y_3 = -1, \text{sign}(w_3^T x_3) = \text{sign}(-4) = -1$
    - $w_4 = w_3 = [-1, 2, -2]$ since it is correct classified.
6. $x_4 = [1, 1, 1], y_4 = 1, \text{sign}(w_4^T x_4) = \text{sign}(-1) = -1$
    - $w_5 = w_4 + y_4 x_4 = [-1, 2, -2] + (-1)[1, 1, 1] = [2, 1, -3]$

- Train examples by PAM
    1. Initialize $w_0 = [0, 0, 0], \tau = 5$
    2. $x_0 = [1, -2, 2], y_0 = -1, y_0 w_0^T x_0 = 0 < \tau$
        - $w_1 = w_0 + y_0 x_0 = [0, 0, 0] + (-1)[1, -2, 2] = [-1, 2, -2]$
    3. $x_1 = [1, -1, 2], y_1 = -1, y_1 w_1^T x_1 = 7 > \tau$
        - $w_2 = w_1 = [-1, 2, -2]$ since it is correct classified.
    4. $x_2 = [1, 2, 0], y_2 = 1, y_2 w_2^T x_2 = 3 < \tau$
        - $w_3 = w_2 + y_2 x_2 = [-1, 2, -2] + (1)[1, 2, 0] = [0, 4, -2]$
    5. $x_3 = [1, -1, 0], y_3 = -1, y_3 w_3^T x_3 = -5 < \tau$
        - $w_4 = w_3 + y_3 x_3 = [0, 4, -2] + (-1)[1, -1, 0] = [-1, 5, -2]$
    6. $x_4 = [1, 1, 1], y_4 = 1, y_4 w_4^T x_4 = 5 = \tau$
        - $w_5 = w_4 = [-1, 5, -2]$ since it is correct classified.
- Predict examples by PLA
    1. Set $w_{\text{PLA}} = [2, 1, -3]$
    2. $x_0 = [1, \frac{1}{2}, 2], y_0 = 1, \hat{y_0} = \text{sign}([2, 1, -3]^T [1, \frac{1}{2}, 2]) = \text{sign}(-\frac{7}{2}) = -1 \neq y_0$
    3. $x_1 = [1, \frac{1}{4}, 1], y_1 = 1, \hat{y_1} = \text{sign}([2, 1, -3]^T [1, \frac{1}{4}, 1]) = \text{sign}(-\frac{3}{4}) = -1 \neq y_1$
    4. $x_2 = [1, \frac{1}{2}, 0], y_2 = 1, \hat{y_2} = \text{sign}([2, 1, -3]^T [1, \frac{1}{2}, 0]) = \text{sign}(\frac{5}{2}) = 1 = y_2$
    5. $x_3 = [1, -\frac{1}{2}, 1], y_3 = -1, \hat{y_3} = \text{sign}([2, 1, -3]^T [1, -\frac{1}{2}, 1]) = \text{sign}(-\frac{3}{2}) = -1 = y_3$
- Predict examples by PAM
    1. Set $w_{\text{PAM}} = [-1, 5, -2]$
    2. $x_0 = [1, \frac{1}{2}, 2], y_0 = 1, \hat{y_0} = \text{sign}([-1, 5, -2]^T [1, \frac{1}{2}, 2]) = \text{sign}(-\frac{5}{2}) = -1 \neq y_0$
    3. $x_1 = [1, \frac{1}{4}, 1], y_1 = 1, \hat{y_1} = \text{sign}([-1, 5, -2]^T [1, \frac{1}{4}, 1]) = \text{sign}(-\frac{7}{4}) = -1 \neq y_1$
    4. $x_2 = [1, \frac{1}{2}, 0], y_2 = 1, \hat{y_2} = \text{sign}([-1, 5, -2]^T [1, \frac{1}{2}, 0]) = \text{sign}(\frac{3}{2}) = 1 = y_2$
    5. $x_3 = [1, -\frac{1}{2}, 1], y_3 = -1, \hat{y_3} = \text{sign}([-1, 5, -2]^T [1, -\frac{1}{2}, 1]) = \text{sign}(-\frac{11}{2}) = -1 = y$
- Test examples are wrongly predicted by PLA but correctly predicted by PAM
    - Wrong predicted by PLA: x_0, x_1
    - But correctly predicted by PAM: none

As a result, we should choose $[e]$ as our solution.

The Learning Problems

P6

It's trivial that the model learned the pattern based on a set of input features, which are all known ratings (supervised learning: all y_n), and the output value is the viewer's rating of the movie. The output space is a real number within the

Question assigned to the following page: 3

range of $\{y|1 \leq y \leq 5\}$, which is a continuous range, making it a regression problem.
As a result, we should choose $[a]$ as our solution.

P7

From the description, the labeler is always fed with two possible outputs and needs to select the better one, which means the task is to predict one of two possible outcomes given a set of input features that is model-generated output (binary classification: $y = \{-1, +1\}$). The labelers are effectively acting as a binary classifier, choosing between two options presented to them.
As a result, we should choose $[b]$ as our solution.

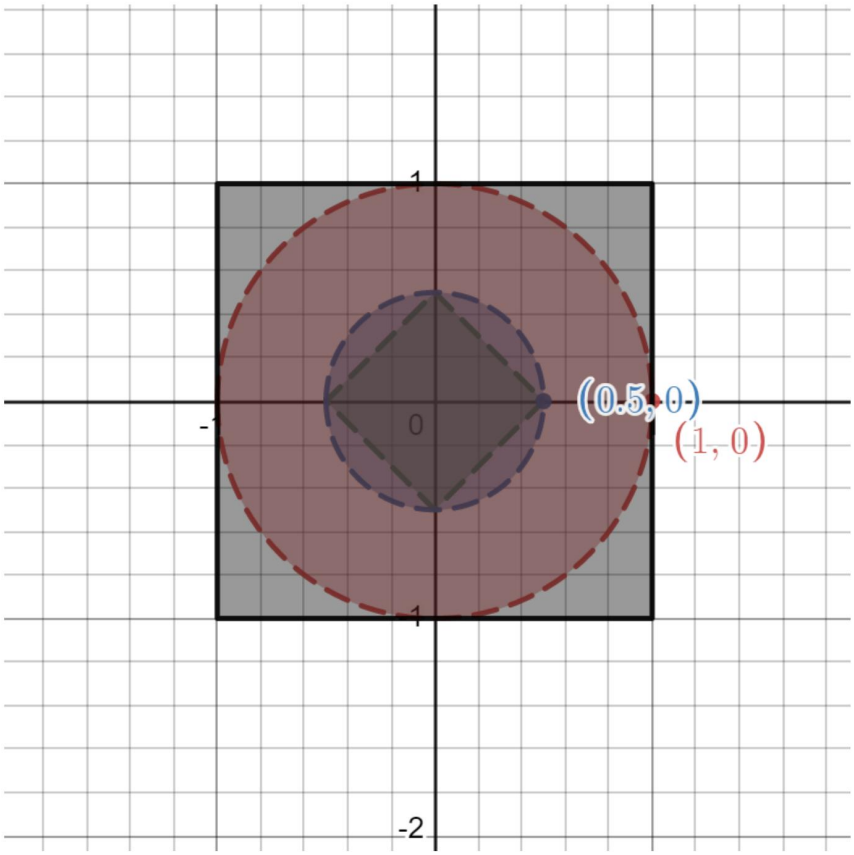
Feasibility of Learning

P8

Set input data x to be $x_i(a_i, b_i), i = \{0, 1, \ldots, 5\}$.
Choose first three examples from $\mathcal{U}$ as $\mathcal{D}$, the perceptron hypothesis $h(x) = \text{sign}(2.5 - b)$ satisfied $y = +1$ for all $y \in \mathcal{D}$. It's trivial that $E_{in}(g) = 0$. and the hypothesis makes the evaluation of g outside $\mathcal{D}$ all correct, which gives the best case $E_{ots} = 0$
Again, choose first three examples from $\mathcal{U}$ as $\mathcal{D}$, the perceptron hypothesis $h(x) = \text{sign}(a + b)$ satisfied $y = +1$ for all $y \in \mathcal{D}$. It's trivial that $E_{in}(g) = 0$. and the hypothesis makes the evaluation of g outside $\mathcal{D}$ all wrong, which gives the worst case $E_{ots} = 1$.
Therefore, the smallest $E_{ots} = 0$, and the biggest $E_{ots} = 1$
As a result, we should choose $[e]$ as our solution.

P9

Question assigned to the following page: 3



Assumed that the area makes $y = +1$, we have

$$\begin{aligned}
\text{Area} &= [+1, -1] \times [+1, -1] && \Rightarrow \text{black} \\
f(x) &= -\text{sign}(x_1^2 + x_2^2 - 0.25) = && \Rightarrow \text{blue} \\
h_1(x) &= -\text{sign}(x_1^2 + x_2^2 - 1) && \Rightarrow \text{red} \\
h_2(x) &= -\text{sign}(|x_1| + |x_2| - 0.5) && \Rightarrow \text{green}
\end{aligned}$$

$$E_{out}(h_1) = \mathbb{P}(\frac{h_1(x) \cap f(x)}{Area}) = \frac{\text{red-blue}}{Area} = \frac{\pi - 0.25\pi}{4} = \frac{3\pi}{16}$$

$$E_{out}(h_2) = \mathbb{P}(\frac{f(x) \cap h_2(x)}{Area}) = \frac{\text{blue-green}}{Area} = \frac{0.25\pi - 0.5}{4} = \frac{\pi - 2}{16}$$

$$\therefore (E_{out}(h_1), E_{out}(h_2)) = (\frac{3\pi}{16}, \frac{\pi - 2}{16})$$

As a result, we should choose $[d]$ as our solution.

P10

$E_{in}(h_1) = E_{in}(h_2)$ means the examples that we draw has the same number of $f(x) \neq h_1(x)$ and $f(x) \neq h_1(x)$. In order to find $E_{in}(h_1) = E_{in}(h_2) = 0$, we have probability $\mathcal{P} = 1 - (\frac{3\pi}{16} + \frac{\pi-2}{16}) = \frac{18-4\pi}{16}$ of each drawing, hence, the probability of drawing 4 examples is $(\frac{18-4\pi}{16})^4 = 0.0133 \approx 0.01$.

Another easier way to solve this problem is to find the probability that both h_1 and h_2 are correct with respect to $f(x)$, hence, we have

Questions assigned to the following page: 6.1, 4, and 5

$$E_{in}(h_1) = E_{in}(h_2) = 0$$
$$\Rightarrow \mathcal{P} = (\frac{\text{black-red+green}}{Area})^4 = (\frac{4-\pi+0.5}{4})^4 = 0.0133 \approx 0.01$$

As a result, we should choose $[b]$ as our solution.

P11

We know that the value $\pi = 4 \cdot \frac{\text{number of darts landed in the circle}}{\text{total number of darts}}$, therefore we can estimate π by $M_n = \frac{4}{n}\sum_{i=1}^{n} x_i$. Then we want $P(|M_n - \pi|) \geq 10^{-2} \leq 0.01$. x_i is a bernolli random variable with $P(x_i = 1) = \frac{\pi}{4}$. The expectation of M_n is

$$\mathbf{E}[M_n] = \mathbf{E}[\frac{4}{n}\sum_{i=1}^{n} x_i] = \frac{4}{n}\sum_{i=1}^{n}\mathbf{E}[x_i] = \frac{4}{n} \cdot n \cdot \frac{\pi}{4} = \pi$$

By Hoeffding's inequality, we have

$$P(|M_n - \pi| \geq 10^{-2}) = P(|\frac{M_n}{4} - \frac{\pi}{4}| \geq 0.0025) \leq 2e^{-2 \cdot 0.0025^2 N}$$
$$2e^{-2 \cdot 0.0025^2 N} \leq 0.01 \Rightarrow N > \ln\frac{0.01}{2} \cdot -\frac{1}{2 \cdot 0.0025^2} \approx 4.23865 \times 10^5$$

As a result, we should choose $[d]$ as our solution.
Ref: **Discrete Mathematics and Probability Theory** (https://hkn.eecs.berkeley.edu/examfiles/cs70_fa14_f_sol.pdf)

P12

With the probability at least $1 - \delta$, we can promise the data for all hypothesis in our hypothesis set is good

$$\delta \leq 2Mexp(-2(\frac{\epsilon}{2})^2 N)$$
$$\ln\frac{\delta}{2M} \leq -\frac{\epsilon^2 N}{2}$$
$$N \geq \frac{1}{2\epsilon^2}\ln\frac{M}{\delta}$$

As a result, we should choose $[c]$ as our solution.

Experiments with Perceptron Learning Algorithm

P13

Question assigned to the following page: 6.1

```python
import random
from tqdm import trange

def average(lst):
    return sum(lst) / len(lst)

def cdot(args, w_t, x_list):
    sum = 0
    for i in range(args['feature']):
        sum += w_t[i]*x_list[i]
    return sum

def E_in(args, w, x_train, y_train):
    sum = 0
    for i in range(args['size']):
        wx = cdot(args, w, x_train[i])
        if((wx <= 0 and y_train[i] > 0) or (wx > 0 and y_train[i] < 0)):
            sum += 1
    return sum/args['size']

def PLA(args, seed, x_train, y_train):
    random.seed(seed)
    time = 0
    M = args['size'] // 2
    w_t = [0]*args['feature']

    while(time < M):
        time += 1
        pick = random.randint(0, args['size']-1)
        wx = cdot(args, w_t, x_train[pick])
        if((wx <= 0 and y_train[pick] > 0) or (wx > 0 and y_train[pick] < 0)):
            time = 0
            w_t = [w_t[i] + x_train[pick][i] * y_train[pick] for i in range(args['feature'])]
    return w_t

def read_file(args):
    x_train = list()
    y_train = list()
    with open (args['filename'], 'r') as f:
        lines = f.readlines()
        for line in lines:
            line_data = line.split()
            line_data = [float(i) for i in line_data]
            x_train.append([1]+line_data[:-1])  # set x_0 = 1 to every X_n
            y_train.append(line_data[-1])
    return x_train, y_train

def main(args):
    x_train, y_train = read_file(args)

    error_list = list()
    for i in trange(args['repeat_time']):
        w_pla = PLA(args, args['seed']+i, x_train, y_train)
        error = E_in(args, w_pla, x_train, y_train)
        error_list.append(error)

    error_avg = average(error_list)
    print("average E_in(w_pla):", error_avg)

if __name__ == '__main__':
    args = {
    'feature': 11,                 # include y
    'filename': 'hw1_train.dat',
    'seed': 1126,
    'size': 256,                    # size = N
    'repeat_time': 1000,
    }

    main(args)
```

Output: $0.01959375 \approx 0.02$

As a result, we should choose $[b]$ as our solution.

P14

We use the same code above, then substitute `M = 4 * args['size']` for `M = args['size'] // 2` in PLA procedure, which is line 24. The updated PLA function shows below.

No questions assigned to the following page.

```python
def PLA(args, seed, x_train, y_train):
    random.seed(seed)
    time = 0
    M = 4 * args['size']
    w_t = [0]*args['feature']

    while(time < M):
        time += 1
        pick = random.randint(0, args['size']-1)
        wx = cdot(args, w_t, x_train[pick])
        if((wx <= 0 and y_train[pick] > 0) or (wx > 0 and y_train[pick] < 0)):
            time = 0
            w_t = [w_t[i] + x_train[pick][i] * y_train[pick] for i in range(args['feature'])]
    return w_t
```

Output: $0.000203125 \approx 0.00020$

As a result, we should choose $[a]$ as our solution.

P15

No questions assigned to the following page.

```python
import random
from tqdm import trange

def average(lst):
    return sum(lst) / len(lst)

def cdot(args, w_t, x_list):
    sum = 0
    for i in range(args['feature']):
        sum += w_t[i]*x_list[i]
    return sum

def E_in(args, w, x_train, y_train):
    sum = 0
    for i in range(args['size']):
        wx = cdot(args, w, x_train[i])
        if((wx <= 0 and y_train[i] > 0) or (wx > 0 and y_train[i] < 0)):
            sum += 1
    return sum/args['size']

def median(lst):
    lst = sorted(lst)
    mid = len(lst) // 2
    res = (lst[mid] + lst[~mid]) / 2
    return res

def PLA(args, seed, x_train, y_train):
    random.seed(seed)
    time = 0
    M = 4 * args['size']
    w_t = [0]*args['feature']
    update = 0

    while(time < M):
        time += 1
        pick = random.randint(0, args['size']-1)
        wx = cdot(args, w_t, x_train[pick])
        if((wx <= 0 and y_train[pick] > 0) or (wx > 0 and y_train[pick] < 0)):
            time = 0
            update += 1
            w_t = [w_t[i] + x_train[pick][i] * y_train[pick] for i in range(args['feature'])]
    return w_t, update

def read_file(args):
    x_train = list()
    y_train = list()
    with open (args['filename'], 'r') as f:
        lines = f.readlines()
        for line in lines:
            line_data = line.split()
            line_data = [float(i) for i in line_data]
            x_train.append([1]+line_data[:-1])  # set x_0 = 1 to every X_n
            y_train.append(line_data[-1])
    return x_train, y_train

def main(args):
    x_train, y_train = read_file(args)

    error_list = list()
    update_list = list()
    for i in trange(args['repeat_time']):
        w_pla, update = PLA(args, args['seed']+i, x_train, y_train)
        error = E_in(args, w_pla, x_train, y_train)
        error_list.append(error)
        update_list.append(update)

    error_avg = average(error_list)
    update_median = median(update_list)
    print("average E_in(w_pla):", error_avg)
    print("median number of updates:", update_median)

if __name__ == '__main__':
    args = {
    'feature': 11,                # include y
    'filename': 'hw1_train.dat',
    'seed': 1126,
    'size': 256,                   # size = N
    'repeat_time': 1000,
    }

    main(args)
```

Output: $454.0 \approx 400$

As a result, we should choose $[d]$ as our solution.

No questions assigned to the following page.

P16

We use the same code above, then record all w_0 of each iteration into a list, and calculate the median value. The updated main function shows below.

```
def main(args):
    x_train, y_train = read_file(args)

    error_list = list()
    update_list = list()
    w_0_list = list()
    for i in trange(args['repeat_time']):
        w_pla, update = PLA(args, args['seed']+i, x_train, y_train)
        error = E_in(args, w_pla, x_train, y_train)
        error_list.append(error)
        update_list.append(update)
        w_0_list.append(w_pla[0])

    error_avg = average(error_list)
    update_median = median(update_list)
    w_0_median = median(w_0_list)
    print("average E_in(w_pla):", error_avg)
    print("median number of updates:", update_median)
    print("median of all w_0:", w_0_median)
```

Output: $35.0 \approx 40$

As a result, we should choose $[e]$ as our solution.

P17

We use the same code above, then update the preprocessing of reading the files. The updated read_file function shows below.

```
def read_file(args):
    x_train = list()
    y_train = list()
    with open (args['filename'], 'r') as f:
        lines = f.readlines()
        for line in lines:
            line_data = line.split()
            line_data = [float(i) for i in line_data]
            x_train.append([0.5]+[line_data[:-1][i]*2 for i in range(args['feature']-1)])  # set x_0 = 1 to every X_
            y_train.append(line_data[-1])
    return x_train, y_train
```

Output: $452.0 \approx 400$

As a result, we should choose $[d]$ as our solution.

P18

We use the same code above, then update the preprocessing of reading the files. The updated read_file function shows below.

```
def read_file(args):
    x_train = list()
    y_train = list()
    with open (args['filename'], 'r') as f:
        lines = f.readlines()
        for line in lines:
            line_data = line.split()
            line_data = [float(i) for i in line_data]
            x_train.append([0]+line_data[:-1])  # set x_0 = 1 to every X_n
            y_train.append(line_data[-1])
    return x_train, y_train
```

Output: $449.0 \approx 400$

As a result, we should choose $[d]$ as our solution.

P19

No questions assigned to the following page.

We use the same code above, then record all w_0x_0 of each iteration into a list, and calculate the median value. The updated main function shows below.

```
def main(args):
    x_train, y_train = read_file(args)

    error_list = list()
    update_list = list()
    w_0_list = list()
    w_0x_0_list = list()
    for i in trange(args['repeat_time']):
        w_pla, update = PLA(args, args['seed']+i, x_train, y_train)
        error = E_in(args, w_pla, x_train, y_train)
        error_list.append(error)
        update_list.append(update)
        w_0_list.append(w_pla[0])
        w_0x_0_list.append(w_pla[0]*x_train[0][0])

    error_avg = average(error_list)
    update_median = median(update_list)
    w_0_median = median(w_0_list)
    w_0x_0_median = median(w_0x_0_list)
    print("average E_in(w_pla):", error_avg)
    print("median number of updates:", update_median)
    print("median of all w_0:", w_0_median)
    print("median of all w_0x_0:", w_0x_0_median)
```

Output: $35.0 \approx 40$

As a result, we should choose $[e]$ as our solution.

P20

We use the same code above, then update the preprocessing of reading the files. The updated read_file function shows below.

No questions assigned to the following page.

```python
import random
from tqdm import trange

def average(lst):
    return sum(lst) / len(lst)

def cdot(args, w_t, x_list):
    sum = 0
    for i in range(args['feature']):
        sum += w_t[i]*x_list[i]
    return sum

def E_in(args, w, x_train, y_train):
    sum = 0
    for i in range(args['size']):
        wx = cdot(args, w, x_train[i])
        if((wx <= 0 and y_train[i] > 0) or (wx > 0 and y_train[i] < 0)):
            sum += 1
    return sum/args['size']

def median(lst):
    lst = sorted(lst)
    mid = len(lst) // 2
    res = (lst[mid] + lst[~mid]) / 2
    return res

def PLA(args, seed, x_train, y_train):
    random.seed(seed)
    time = 0
    M = 4 * args['size']
    w_t = [0]*args['feature']
    update = 0

    while(time < M):
        time += 1
        pick = random.randint(0, args['size']-1)
        wx = cdot(args, w_t, x_train[pick])
        if((wx <= 0 and y_train[pick] > 0) or (wx > 0 and y_train[pick] < 0)):
            time = 0
            update += 1
            w_t = [w_t[i] + x_train[pick][i] * y_train[pick] for i in range(args['feature'])]
    return w_t, update

def read_file(args):
    x_train = list()
    y_train = list()
    with open (args['filename'], 'r') as f:
        lines = f.readlines()
        for line in lines:
            line_data = line.split()
            line_data = [float(i) for i in line_data]
            x_train.append([0.1126]+line_data[:-1])  # set x_0 = 1 to every X_n
            y_train.append(line_data[-1])
    return x_train, y_train

def main(args):
    x_train, y_train = read_file(args)

    error_list = list()
    update_list = list()
    w_0_list = list()
    w_0x_0_list = list()
    for i in trange(args['repeat_time']):
        w_pla, update = PLA(args, args['seed']+i, x_train, y_train)
        error = E_in(args, w_pla, x_train, y_train)
        error_list.append(error)
        update_list.append(update)
        w_0_list.append(w_pla[0])
        w_0x_0_list.append(w_pla[0]*x_train[0][0])

    error_avg = average(error_list)
    update_median = median(update_list)
    w_0_median = median(w_0_list)
    w_0x_0_median = median(w_0x_0_list)
    print("average E_in(w_pla):", error_avg)
    print("median number of updates:", update_median)
    print("median of all w_0:", w_0_median)
    print("median of all w_0x_0:", w_0x_0_median)

if __name__ == '__main__':
    args = {
    'feature': 11,                  # include y
    'filename': 'hw1_train.dat',
    'seed': 1126,
    'size': 256,                    # size = N
    'repeat_time': 1000,
    }

    main(args)
```

No questions assigned to the following page.

Output: $0.4437 \approx 0.4$

As a result, we should choose $[c]$ as our solution.