

# HW3 Solutions

● Graded

Student

Chih-Hao Liao

Total Points

93 / 100 pts

Question 1

Problem 3

20 / 20 pts

✓ - 0 pts Correct

- 20 pts No solution or wrong answer
- 20 pts No page selected or wrong page selected
- 5 pts Minor mistake
- 10 pts Major mistake / Several minor mistakes
- 15 pts Some reasonable effort but fundamentally wrong
- 20 pts Totally wrong solution or almost missing solution
- 20 pts Showed little to no derivation steps
- 10 pts Showed only partial derivation steps

Question 2

Problem 5

18 / 20 pts

- 0 pts Totally Correct, if you have some typos but I can understand what you want to deliver, I will not deduct your points.
- 1 pt Apply the provided fact with the correct condition, without briefly referring to the cases where #inputs < 4, or why can we ignore such cases?

✓ - 2 pts Directly using the provided fact of  $\log N \leq \frac{N}{2}$ , but ignoring the condition of  $N \geq 4$ , raises the problem that the inequality might not hold when  $N < 4$ . One should briefly discuss this.  
(Note: we only assume that  $d \geq 4$ .)

1

- 5 pts Almost correct.
- 10 pts incorrect.
- 10 pts Correct, but don't select page.
- 15 pts Almost incorrect
- 20 pts Wrong answer or no explanation.

### Question 3

#### Problem 9

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

### Question 4

#### Problem 11

15 / 20 pts

- 0 pts Correct

- 1 pt Typo

✓ - 5 pts The answer is almost correct or with minor mistakes or missing some details.

- 10 pts The answer is almost incorrect or missing most of the details.

- 10 pts The answer is obtained by programming.

- 20 pts 1. The answer is incorrect. 2. You chose the wrong page. 3. You didn't provide the solving process.

2

please check the dimension

### Question 5

#### Problem 12

20 / 20 pts

✓ - 0 pts Correct

- 5 pts One minor mistake or steps are a bit unclear for no explanation

- 10 pts One major mistake or no complete math derivation process

- 15 pts Some reasonable effort but fundamentally wrong

- 20 pts Wrong answer or No solution or almost missing solution

## Question 6

Coding Part 0 / 0 pts

6.1 Problem 15 0 / 0 pts

 - 0 pts Correct

- 20 pts Cannot understand the implementatoin

- 160 pts Without source code

No questions assigned to the following page.

# HTML\_2023\_HW3

---

tags: Personal

Discuss with anonymous and TAs.

## Linear Models and More

---

### P1

- CPU time:  $aN$  for training a binary classifier on a size  $N$  binary classification data set
- We have size- $N$   $K$ -class classification data set, which means each class is of size  $\frac{N}{K}$
- By performing one-versus-one(OVO), we need to compute  $\binom{K}{2}$  binary classification on the dataset.
- For each binary classification, we only select 2 classes for computing the result, which means for each binary classification, only  $2 \times \frac{N}{K}$  dataset will be used.
- Therefore, the total CPU time is

$$a \binom{K}{2} \times 2 \times \frac{N}{K} = a \frac{K!}{2!(K-2)!} \frac{2N}{K} = a(K-1)N$$

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

### P2

Since quadratic hypothesis can implement all possible quadratic curve boundaries, such as circle, hyperbola, and parabola, and includes line, and constants. Therefore, we can assumed that  $x_i = (x_{i1}, x_{i2})$ , and transformation function  $\Phi_2(x) = (1, x_1, x_2, x_1^2, x_1x_2, x_2^2) = z$ .

From the transformation, we know that the  $d_{vc}(\mathcal{H}_{\Phi_2(x)}) \leq 6$ , it is trivial that six inputs can be shattered after the transformation. For a more detailed calculation, we have

$$\begin{aligned} x_1 &= (2, 0), & x_2 &= (0, 2), & x_3 &= (-2, 0) \\ x_4 &= (0, -2), & x_5 &= (0, 0), & x_6 &= (1, 1) \end{aligned}$$

Question assigned to the following page: [1](#)

$$\begin{aligned}
\Phi_2(x_1) &= (1, 2, 0, 4, 0, 0) &= z_1 \\
\Phi_2(x_2) &= (1, 0, 2, 0, 0, 4) &= z_2 \\
\Phi_2(x_3) &= (1, -2, 0, 4, 0, 0) &= z_3 \\
\Phi_2(x_4) &= (1, 0, -2, 0, 0, 4) &= z_4 \\
\Phi_2(x_5) &= (1, 0, 0, 0, 0, 0) &= z_5 \\
\Phi_2(x_6) &= (1, 1, 1, 1, 1, 1) &= z_6
\end{aligned}$$

$$z = \begin{bmatrix} z_1^T \\ z_2^T \\ z_3^T \\ z_4^T \\ z_5^T \\ z_6^T \end{bmatrix} = \begin{bmatrix} 1 & 2 & 0 & 4 & 0 & 0 \\ 1 & 0 & 2 & 0 & 0 & 4 \\ 1 & -2 & 0 & 4 & 0 & 0 \\ 1 & 0 & -2 & 0 & 0 & 4 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}, w = \begin{bmatrix} w_1^T \\ w_2^T \\ w_3^T \\ w_4^T \\ w_5^T \\ w_6^T \end{bmatrix}$$

Since  $z$  is invertible, let  $y = zw$ , then for each  $y_i, 0 \leq i \leq 6$ , there exists a  $w = z^{-1}y$ , which means  $z_1, z_2, z_3, z_4, z_5, z_6$  can be shattered by quadratic hypothesis set, therefore,  $x_1, x_2, x_3, x_4, x_5, x_6$  can also be shattered by the union of quadratic, linear, or constant hypothesis sets.

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

### P3

By hand

$$\begin{aligned}
x_1 &= (0, 0), & y_1 &= -1 \\
x_2 &= (4, 0), & y_2 &= +1 \\
x_3 &= (-4, 0), & y_3 &= +1 \\
x_4 &= (0, 2), & y_4 &= -1 \\
x_5 &= (0, -2), & y_5 &= -1
\end{aligned}$$

$$\begin{aligned}
w_1 &= (-1, 0, 0, 0.5, 0, -0.5) \\
w_2 &= (-1, 0, 0, -0.5, 0, 0.5) \\
w_3 &= (-2, 0, 0, 1, 0, 1) \\
w_4 &= (-1, 0, 0, 0.2, 0, 0.1)
\end{aligned}$$

Question assigned to the following page: [1](#)

$$\begin{aligned}
\Phi(x) &= (1, x_1, x_2, x_1^2, x_1x_2, x_2^2) \\
\Phi(x_1) &= (1, 0, 0, 0, 0, 0) \\
\Phi(x_2) &= (1, 4, 0, 16, 0, 0) \\
\Phi(x_3) &= (1, -4, 0, 16, 0, 0) \\
\Phi(x_4) &= (1, 0, 2, 0, 0, 4) \\
\Phi(x_5) &= (1, 0, -2, 0, 0, 4)
\end{aligned}$$

	$\text{sign}(w_i^T \phi(x_i))$	$y_i$	$\text{sign}(w_i^T \phi(x_i))) = y_i$
$w_1$	[-1,1,1,-1,-1]	[-1,1,1,-1,-1]	[O,O,O,O,O]
$w_2$	[-1,-1,-1,1,1]	[-1,1,1,-1,-1]	[O,X,X,X,X]
$w_3$	[-1,1,1,1,1]	[-1,1,1,-1,-1]	[O,O,O,X,X]
$w_4$	[-1,1,1,-1,-1]	[-1,1,1,-1,-1]	[O,O,O,O,O]

By python

```

1 import numpy as np
2
3 def phi(x):
4     return [1, x[0], x[1], x[0]*x[0], x[0]*x[1], x[1]*x[1]]
5
6 def main(dataset, weight_vector):
7     result = 0
8     for i in range(len(weight_vector)):
9         print("weight_vector: ", weight_vector[i])
10        counter = 0
11        for j in range(len(dataset)):
12            hypothesis = int(np.sign(np.dot(weight_vector[i], phi(dataset[j]))))
13            if hypothesis == dataset[j][2]:
14                print("0:", "(", hypothesis, ",", dataset[j][2], ")")
15                counter += 1
16            else:
17                print("X:", "(", hypothesis, ",", dataset[j][2], ")")
18        if counter == len(dataset):
19            print("==separates the dataset correct==")
20            result += 1
21        else:
22            print("==separates the dataset incorrect==")
23    print("There are", result, "weight vectors that can seperate dataset.")

```

Question assigned to the following page: [1](#)

```
1 if __name__ == '__main__':
2     dataset = np.array([
3         # x_1,x_2, y
4         [0, 0, -1],
5         [4, 0, 1],
6         [-4, 0, 1],
7         [0, 2, -1],
8         [0, -2, -1],
9     ])
10
11    weight_vector = np.array([
12        [-1, 0, 0, 0.5, 0, -0.5],
13        [-1, 0, 0, -0.5, 0, 0.5],
14        [-2, 0, 0, 1, 0, 1],
15        [-1, 0, 0, 0.2, 0, 0.1],
16    ])
17
18    main(dataset, weight_vector)
19
```

Question assigned to the following page: [1](#)

```
1 weight_vector: [-1.  0.  0.  0.5 0. -0.5]
2 O: (-1, -1)
3 O: (1, 1)
4 O: (1, 1)
5 O: (-1, -1)
6 O: (-1, -1)
7 ===separates the dataset correct===
8 weight_vector: [-1.  0.  0. -0.5 0.  0.5]
9 O: (-1, -1)
10 X: (-1, 1)
11 X: (-1, 1)
12 X: (1, -1)
13 X: (1, -1)
14 ===separates the dataset incorrect===
15 weight_vector: [-2.  0.  0.  1.  0.  1.]
16 O: (-1, -1)
17 O: (1, 1)
18 O: (1, 1)
19 X: (1, -1)
20 X: (1, -1)
21 ===separates the dataset incorrect===
22 weight_vector: [-1.  0.  0.  0.2 0.  0.1]
23 O: (-1, -1)
24 O: (1, 1)
25 O: (1, 1)
26 O: (-1, -1)
27 O: (-1, -1)
28 ===separates the dataset correct===
29 There are 2 weight vectors that can separate dataset.
```

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

Question assigned to the following page: [2](#)

## P4

$$\begin{aligned}
\phi(x) &= [(\Gamma x_n)^T] = x \Gamma^T \\
\tilde{w} &= ((x \Gamma^T)^T (x \Gamma^T))^{-1} (x \Gamma^T)^T y \\
&= ((x \Gamma^T)^{-1}) ((x \Gamma^T)^T)^{-1} (x \Gamma^T)^T y \\
&= (x \Gamma^T)^{-1} y \\
&= (\Gamma^T)^{-1} x^{-1} y \\
w_{lin} &= (x^T x^{-1}) x^T y \\
&= x^{-1} (x^T)^{-1} x^T y \\
&= x^{-1} y \\
\tilde{w} &= (\Gamma^T)^{-1} w_{lin} \\
w_{lin} &= \Gamma^T \tilde{w} \\
E_{in}(\tilde{w}) &= \frac{1}{N} \sum_{n=1}^N ((\Gamma x_n)^T \tilde{w} - y_n)^2 \\
&= \frac{1}{N} \sum_{n=1}^N (x_n^T \Gamma^T (\Gamma^T)^{-1} w_{lin} - y_n)^2 \\
&= \frac{1}{N} \sum_{n=1}^N (x_n^T w_{lin} - y_n)^2 \\
&= E_{in}(w_{lin})
\end{aligned}$$

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

## P5

Define  $\Phi_{(k)}(x) = (1, x_k) = (w_0, w_1)$ , then we have  $w_0 + w_1 x_k = 0 \Leftrightarrow x_k = -\frac{w_0}{w_1}$  (threshold), which means if  $x_k$  is larger than threshold, label +1 otherwise, label -1. Hence, we can consider this feature transform as a decision stumps where  $m_H(N) = 2N$  for any  $\Phi_{(k)}(x) = (1, x_k)$ . Except for all positive or all negative conditions, there exists  $N$  inputs and  $N - 1$  intervals between input values can be chosen by  $\mathcal{H}_k$ , this tells us that there are  $2(N - 1)$  possible conditions, and there are 2 conditions should be included, which is all positive or all negatives. Moreover, there are  $d$  transformation functions, which means  $\bigcup_{k=1}^d$  have  $m_H(N) = 2(N - 1)d + 2 = 2Nd$ . From the definition of  $d_{vc}$ , we have

Question assigned to the following page: [2](#)

$$\begin{aligned}
2^N &\leq 2(N-1)d + 2 = 2Nd \\
\Rightarrow \frac{2N}{2} &\leq Nd \Rightarrow 2^{N-1} \leq Nd \\
\Rightarrow N-1 &\leq \log_2 N + \log_2 d \quad \text{(by hint)} \\
\Rightarrow 2N-2 &\leq N + 2\log_2 d \\
\Rightarrow N &\leq 2\log_2 d + 2 = 2(\log_2 d + 1) \\
\therefore d_{vc}(\bigcup_{k=1}^d \mathcal{H}_k) &\leq 2(\log_2 d + 1)
\end{aligned}$$

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

## P6

The feature transform means for each  $x$  in dataset, we have the transformation

$$\Phi(x) = z = \begin{bmatrix} [x = x_1] \\ [x = x_2] \\ \vdots \\ [x = x_N] \end{bmatrix}$$

For each  $[x = x_i], i = 1, 2, \dots, N$  means if  $x = x_i$ , then the value is 1 otherwise 0. From the assumption, it shows that all  $x_n$  are different, which means

$$\Phi(x_1) = z_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \Phi(x_2) = z_2 = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \dots, \Phi(x_N) = z_N = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$

Therefore,  $(\Phi(x))_n$  is a one-hot table. Then let

$$z = \begin{bmatrix} 1 & z_1^T \\ 1 & z_2^T \\ \vdots & \vdots \\ 1 & z_N^T \end{bmatrix} = I_{n \times n} \text{(Identity matrix)}$$

[a]

We know that the rank of  $z$  is  $N$ , which means  $z$  has full column rank. Let  $y = z\tilde{w}$ , then for each  $y_i, 0 \leq i \leq N$ , there exists a  $\tilde{w} = z^{-1}y$ , therefore, we know that

No questions assigned to the following page.

$$\begin{aligned}\therefore \tilde{w} &= (x^T x)^{-1} x^T y \\ &= \begin{bmatrix} 1 & z_1^T \\ 1 & z_2^T \\ \vdots & \vdots \\ 1 & z_N^T \end{bmatrix}^{-1} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \\ \therefore \tilde{w}_n &= y_n\end{aligned}$$

[b]

Since  $E_{in} = (g(x) - y)^2 = (\tilde{w}^T \Phi(x) - y)^2 = \frac{1}{N} \sum_{n=1}^N (y_n - \tilde{w}^T \Phi(x_n))^2 = 0$ , therefore, we have  $E_{in} = 0$

[c]

Since  $g(x_n) = \tilde{w}^T \Phi(x_n) = \tilde{w}^T z_n = y_n$ , therefore we know that  $g(2x_n) = y_n$

[d]

From the definition of  $z$ , we know that for each  $[x = x_i], i = 1, 2, \dots, N$  if  $x = x_i$ , then the value is 1, and for any  $n$ , if  $x \neq x_i$  the value is 0. Hence,  $g(x) = \tilde{w}^T \Phi(x \neq x_n) = 0$

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

No questions assigned to the following page.

## Playing with Regularization

---

P7

$$\begin{aligned} E_{aug}(w) &= E_{in}(w) + \frac{\lambda}{3} \|w\|_1 \\ &= \frac{1}{N} \sum_{n=1}^N (w_0 + w_1 x_n - y_n)^2 + \frac{\lambda}{3} (|w_0| + |w_1|) \\ \frac{\partial E_{aug}(w)}{\partial w_0} &= \frac{1}{N} \sum_{n=1}^N 2(w_0 + w_1 x_n - y_n) + \frac{\lambda}{3} \left( \frac{w_0}{|w_0|} \right) \\ &\Rightarrow \frac{2}{3} (3w_0 + 3w_1 - 3) + \frac{3}{3} \left( \frac{w_0}{|w_0|} \right) \\ &\Rightarrow 2w_0 + 2w_1 - 2 + \left( \frac{w_0}{|w_0|} \right) = 0 \Leftarrow (1) \\ \frac{\partial E_{aug}(w)}{\partial w_1} &= \frac{1}{N} \sum_{n=1}^N 2x_n (w_0 + w_1 x_n - y_n) + \frac{\lambda}{3} \left( \frac{w_1}{|w_1|} \right) \\ &\Rightarrow \frac{2}{3} (3w_0 + 17w_1 + 2) + \frac{3}{3} \left( \frac{w_1}{|w_1|} \right) \\ &\Rightarrow 2w_0 + \frac{34}{3}w_1 + \frac{4}{3} + \left( \frac{w_1}{|w_1|} \right) = 0 \Leftarrow (2) \\ (2) - (1) &\Rightarrow \frac{28}{3}w_1 + \frac{10}{3} + \left( \frac{w_1}{|w_1|} - \frac{w_0}{|w_0|} \right) = 0 \end{aligned}$$

No questions assigned to the following page.

$$\begin{aligned}
&\text{if } \frac{w_1}{|w_1|} = 1, \frac{w_0}{|w_0|} = 1 \Rightarrow w_1 = -\frac{5}{14}, w_0 = \frac{6}{7} \\
&\text{if } \frac{w_1}{|w_1|} = 1, \frac{w_0}{|w_0|} = -1 \Rightarrow w_1 = -\frac{4}{7}, w_0 = \frac{29}{14} \\
&\text{if } \frac{w_1}{|w_1|} = -1, \frac{w_0}{|w_0|} = 1 \Rightarrow w_1 = -\frac{1}{7}, w_0 = \frac{9}{14} \\
&\text{if } \frac{w_1}{|w_1|} = -1, \frac{w_0}{|w_0|} = -1 \Rightarrow w_1 = -\frac{5}{14}, w_0 = \frac{13}{7} \\
&\therefore w_1 = -\frac{1}{7}, w_0 = \frac{9}{14}
\end{aligned}$$

$$\begin{aligned}
E_{aug}(w) &= \frac{1}{N} \sum_{n=1}^N \left( \frac{9}{14} - \frac{1}{7}x_n - y_n \right)^2 + \frac{\lambda}{3} \left( \frac{11}{14} \right) \\
&= \frac{1}{3} \left( \left( -\frac{9}{14} \right)^2 + \left( \frac{3}{14} \right)^2 + \left( -\frac{15}{14} \right)^2 \right) + \frac{11}{14} \\
&= \frac{1}{3} \left( \frac{45}{28} \right) + \frac{11}{14} = \frac{15}{28} + \frac{22}{28} \\
&= \frac{37}{28} \approx 1.321
\end{aligned}$$

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

## P8

$$\begin{aligned}
E_{aug}(w) &= E_{in}(w) + \frac{\lambda}{N} \|w\|_2^2 \\
&= \frac{1}{N} \sum_{n=1}^N (w_0 + w_1 x_n - y_n)^2 + \frac{\lambda}{N} (w_0^2 + w_1^2) \\
\frac{\partial E_{aug}(w)}{\partial w_0} &= \frac{1}{N} \sum_{n=1}^N 2(w_0 + w_1 x_n - y_n) + \frac{\lambda}{N} (2w_0) \\
&\Rightarrow 2w_0 - 8 + \lambda w_0 = 0 \\
w_0 &= \frac{8}{2 + \lambda} \\
\frac{\partial E_{aug}(w)}{\partial w_1} &= \frac{1}{N} \sum_{n=1}^N 2x_1(w_0 + w_1 x_n - y_n) + \frac{\lambda}{N} (2w_1) \\
&\Rightarrow 8w_1 - 20 + \lambda w_1 = 0 \\
w_1 &= \frac{20}{8 + \lambda}
\end{aligned}$$

Therefore, we can take test example back to the equation, then we have

Question assigned to the following page: [3](#)

$$y = w_{reg}^T \begin{bmatrix} 1 \\ x \end{bmatrix} = w_0 + w_1 x$$

$$4 = \frac{8}{2+\lambda} + \frac{20}{8+\lambda}$$

$$\lambda = 2$$

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

P9

$$E_{aug}(w) = E_{in}(w) \frac{\lambda}{N} w^T w$$

$$= E_{in}(w) \frac{\lambda}{N} (w_0^2 + w_1^2 + \dots + w_d^2)$$

$$\frac{\partial E_{aug}(w)}{\partial w_i} = \frac{\partial \nabla E_{in}(w)}{\partial w_i} + \frac{2\lambda}{N} w_i$$

$$\nabla E_{aug}(w) = \nabla E_{in}(w) + \frac{2\lambda}{N} \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix}$$

$$= \nabla E_{in}(w) + \frac{2\lambda}{N} w$$

$$\therefore w_{t+1} \leftarrow w_t - \eta \nabla E_{aug}(w_t)$$

$$w_{t+1} \leftarrow w_t - \eta (\nabla E_{in}(w_t) + \frac{2\lambda}{N} w_t)$$

$$w_{t+1} \leftarrow (1 - \frac{2\eta\lambda}{N}) w_t - \eta \nabla E_{in}(w_t)$$

$$a = 1 - \frac{2\eta\lambda}{N}$$

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

## Virtual Examples and Regularization

---

P10

In order to find the minimum  $w$ , we can set the derivative of the optimal solution to be zero, hence, we have

No questions assigned to the following page.

$$\begin{aligned}
0 &= \frac{\partial}{\partial w} \left( \frac{1}{N+K} \left( \sum_{n=1}^N (y_n - w^T x_n)^2 + \sum_{k=1}^K (\tilde{y}_k - w^T \tilde{x}_k)^2 \right) \right) \\
&= \frac{1}{N+K} \left( \sum_{n=1}^N -2x_n(y_n - w^T x_n) + \sum_{k=1}^K -2\tilde{x}_k(\tilde{y}_k - w^T \tilde{x}_k) \right) \\
&= \frac{2}{N+K} \left( \sum_{n=1}^N w^T x_n^2 - \sum_{n=1}^N x_n y_n + \sum_{k=1}^K w^T \tilde{x}_k^2 - \sum_{k=1}^K \tilde{x}_k \tilde{y}_k \right) \\
\sum_{n=1}^N w^T x_n^2 + \sum_{k=1}^K w^T \tilde{x}_k^2 &= \sum_{n=1}^N x_n y_n + \sum_{k=1}^K \tilde{x}_k \tilde{y}_k \\
w &= (x^T x + \tilde{x}^T \tilde{x})^{-1} (x^T y + \tilde{x}^T \tilde{y})
\end{aligned}$$

Then, we can find the minimum  $w_{reg}$  with the same method, which is setting the derivative of the optimal solution to be zero, and we get

$$\begin{aligned}
0 &= \frac{\partial}{\partial w} \left( \frac{1}{N} \|Xw - y\|^2 + \frac{\lambda}{N} \|w\|^2 \right) \\
&= \frac{\partial}{\partial w} \left( \frac{1}{N} \sum_{n=1}^N (X_n w - y_n)^2 + \frac{\lambda}{N} w^2 \right) \\
&= \frac{1}{N} \sum_{n=1}^N 2X_n(X_n w - y_n) + \frac{2\lambda I}{N} w \\
&= \frac{2}{N} \sum_{n=1}^N w^T X_n^2 - \sum_{n=1}^N X_n y_n + \frac{2\lambda I}{N} w \\
&= \frac{2}{N} \left( \sum_{n=1}^N w^T X_n^2 - \sum_{n=1}^N X_n y_n + \lambda I w \right) \\
\sum_{n=1}^N w^T X_n^2 + \lambda I w &= \sum_{n=1}^N X_n y_n \\
w &= (X^T X + \lambda I)^{-1} X^T y
\end{aligned}$$

Compare with the coefficients, we know that  $w = w_{reg}$ , therefore, we have

$$\begin{aligned}
w &= (x^T x + \tilde{x}^T \tilde{x})^{-1} (x^T y + \tilde{x}^T \tilde{y}) \\
w_{reg} &= (X^T X + \lambda I)^{-1} X^T y \\
w &= w_{reg} \\
(x^T x + \tilde{x}^T \tilde{x})^{-1} (x^T y + \tilde{x}^T \tilde{y}) &= (X^T X + \lambda I)^{-1} X^T y \\
\tilde{X} &= \sqrt{\lambda} I_{d+1} \\
\tilde{y} &= 0
\end{aligned}$$

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

Question assigned to the following page: [4](#)

P11

Question assigned to the following page: [4](#)

$$\begin{aligned}
\mathbb{E}(x_h^T x_h) &= \mathbb{E} \left( \begin{bmatrix} | & \dots & | & \dots & | \\ x_1 & \dots & x_N & \tilde{x}_1 & \dots & \tilde{x}_N \\ | & \dots & | & | & \dots & | \end{bmatrix} \cdot \begin{bmatrix} - & x_1 & - \\ \vdots & \vdots & \vdots \\ - & x_N & - \\ - & \tilde{x}_1 & - \\ \vdots & \vdots & \vdots \\ - & \tilde{x}_N & - \end{bmatrix} \right) \\
&= \mathbb{E} \left( \begin{bmatrix} | & \dots & | & \dots & | \\ x_1 & \dots & x_N & x_1 + \epsilon & \dots & x_N + \epsilon \\ | & \dots & | & | & \dots & | \end{bmatrix} \cdot \begin{bmatrix} - & x_1 & - \\ \vdots & \vdots & \vdots \\ - & x_N & - \\ - & x_1 + \epsilon & - \\ \vdots & \vdots & \vdots \\ - & x_N + \epsilon & - \end{bmatrix} \right) \\
&= \mathbb{E} \left( \begin{bmatrix} | & \dots & | & \dots & | \\ x_1 & \dots & x_N & x_1 + \epsilon & \dots & x_N + \epsilon \\ | & \dots & | & | & \dots & | \end{bmatrix} \cdot \begin{bmatrix} x_{11} \\ \vdots \\ x_{N1} \\ x_{11} + \epsilon \\ \vdots \\ x_{N1} + \epsilon \end{bmatrix} + \dots \right. \\
&\quad \left. + \begin{bmatrix} | & \dots & | & \dots & | \\ x_1 & \dots & x_N & x_1 + \epsilon & \dots & x_N + \epsilon \\ | & \dots & | & | & \dots & | \end{bmatrix} \cdot \begin{bmatrix} x_{1d} \\ \vdots \\ x_{Nd} \\ x_{1d} + \epsilon \\ \vdots \\ x_{Nd} + \epsilon \end{bmatrix} \right) \text{ (2)} \\
&= \mathbb{E}(x^T x + x^T x) + \mathbb{P} \left( \begin{bmatrix} \epsilon^2 & 0 & \dots & 0 & 0 \\ 0 & \epsilon^2 & \dots & 0 & 0 \\ \vdots & 0 & \ddots & 0 & \vdots \\ 0 & 0 & \dots & \epsilon^2 & 0 \\ 0 & 0 & \dots & 0 & \epsilon^2 \end{bmatrix} \right) \\
&= \mathbb{E}(x^T x + x^T x) + N \cdot \mathbb{P}(\epsilon^2) I_{d+1} \\
&= \mathbb{E}(x^T x + x^T x) + N \cdot \frac{1}{2r} \int_{-r}^r \epsilon^2 I_{d+1} \\
&= x^T x + x^T x + N \cdot \frac{1}{2r} \frac{2r^3}{3} I_{d+1} \\
&= 2x^T x + \frac{N}{3} r^2 I_{d+1}
\end{aligned}$$

Questions assigned to the following page: [4](#) and [5](#)

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

## P12

In order to find the minimum  $y \in \mathbb{R}$ , we can set the derivative of the optimal solution to be zero, which means

$$\begin{aligned}
0 &= \frac{\partial}{\partial y} \left( \frac{1}{N} \sum_{n=1}^N (y - y_n)^2 + \frac{\lambda}{N} \Omega(y) \right) \\
&= \frac{1}{N} \sum_{n=1}^N 2(y - y_n) + \frac{\lambda}{N} \Omega'(y) \\
&= \frac{2}{N} \left( \sum_{n=1}^N y - \sum_{n=1}^N y_n \right) + \frac{\lambda}{N} \Omega'(y) \\
&= \frac{2}{N} (Ny - \sum_{n=1}^N y_n) + \frac{\lambda}{N} \Omega'(y) \\
&= 2y - \frac{2}{N} \sum_{n=1}^N y_n + \frac{\lambda}{N} \Omega'(y) \\
\frac{\lambda}{N} \Omega'(y) &= \frac{2}{N} \sum_{n=1}^N y_n - 2y
\end{aligned}$$

Question assigned to the following page: [5](#)

$$\begin{aligned}
\therefore \text{optimal } y^* &= \frac{(\sum_{n=1}^N y_n) + K}{N + 2K} \\
\therefore \frac{\lambda}{N} \Omega'(y) &= \frac{2}{N} \sum_{n=1}^N y_n - 2 \left( \frac{(\sum_{n=1}^N y_n) + K}{N + 2K} \right) \\
&= \frac{1}{N + 2K} \left( \frac{2 \cdot (N + 2K) \cdot (\sum_{n=1}^N y_n)}{N} - 2 \cdot \left( \sum_{n=1}^N y_n \right) - 2K \right) \\
&= \frac{1}{N + 2K} \left( 2 \cdot \left( \sum_{n=1}^N y_n \right) + 4 \cdot \frac{K}{N} \left( \sum_{n=1}^N y_n \right) - 2 \cdot \left( \sum_{n=1}^N y_n \right) - 2K \right) \\
&= \frac{1}{N + 2K} \left( \frac{4K}{N} \left( \sum_{n=1}^N y_n \right) - 2K \right) \\
&= \frac{2K}{N + 2K} \left( \frac{2}{N} \left( \sum_{n=1}^N y_n \right) - 1 \right) \\
\Omega'(y) &= \frac{N}{\lambda} \cdot \frac{2K}{N + 2K} \cdot \left( \frac{2 \cdot (\sum_{n=1}^N y_n) - N}{N} \right) \\
&= \frac{2K}{\lambda} \cdot \frac{1}{N + 2K} \cdot \left( 2 \cdot \left( \sum_{n=1}^N y_n \right) - N \right) \\
\therefore \text{optimal } y^* &= \frac{(\sum_{n=1}^N y_n) + K}{N + 2K} \\
\therefore \Omega'(y) &= \frac{2K}{\lambda} \cdot (2y - 1) \\
&= \frac{2K}{\lambda} \cdot 2 \cdot \left( y - \frac{1}{2} \right) \\
&= \frac{2K}{\lambda} \cdot 2 \cdot (y - 0.5) \\
\Omega(y) &= \frac{2K}{\lambda} (y - 0.5)^2
\end{aligned}$$

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

No questions assigned to the following page.

# Experiments with Linear and Nonlinear Models

---

## P13

```
1 import numpy as np
2
3 def calculate_E_in_sqr(x, y, w):
4     return np.mean([(np.dot(w, x[i]) - y[i])**2 for i in range(len(x))])
5
6 def calculate_w_lin(x, y):
7     # pseudo inverse: np.linalg.pinv(np.array([values]))
8     return np.dot(np.linalg.pinv(x), y)
9
10 def read_file(filename):
11     data = np.loadtxt(filename, dtype=float)
12     x = data[:, :-1]
13     y = data[:, -1]
14     return x, y
15
16 def main(args):
17     x_train, y_train = read_file(args['filename_train'])
18     x_train = np.c_[np.ones(len(x_train)), x_train] # x_{n, 0}=1
19
20     weight = calculate_w_lin(x_train, y_train)
21     E_in = calculate_E_in_sqr(x_train, y_train, weight)
22     print("E_in: ", E_in)
23
24 if __name__ == '__main__':
25     args = {
26         'filename_test': "hw3_test.dat",
27         'filename_train': "hw3_train.dat"
28     }
29
30     main(args)
```

```
1 | E_in:  0.792234776110557
```

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

No questions assigned to the following page.

## P14

```
1 import numpy as np
2 import random
3 from tqdm import trange
4
5 def calculate_E_in_sqr(x, y, w):
6     return np.mean([(np.dot(w, x[i]) - y[i])**2 for i in range(len(x))])
7
8 def read_file(args, filename):
9     data = np.loadtxt(filename, dtype=float)
10    x = data[:, :-1]
11    y = data[:, -1]
12    args['features'] = len(x[0])
13    args['size'] = len(x)
14    return x, y
15
16 def SGD(args, x, y):
17     error_list = list()
18     w_list = list()
19     for i in trange(args['repeat_time']):
20         random.seed()
21         w = np.zeros(args['feature'])
22         for j in range(args['iteration']):
23             pick = random.randint(0, args['size']-1)
24             w = w + args['learning_rate'] * 2 * (y[pick] - np.dot(w, x[pick])) * x[
25             pick]
26             error_list.append(calculate_E_in_sqr(x, y, w))
27             w_list.append(w)
28     return w_list, error_list
```

Question assigned to the following page: [6.1](#)

```

1 def main(args):
2     x_train, y_train = read_file(args, args['filename_train'])
3     x_train = np.c_[np.ones(len(x_train)), x_train] # x_{n, 0}=1
4     args['feature'] = len(x_train[0])
5
6     weight_list, E_in_list = SGD(args, x_train, y_train)
7     E_in_average = np.mean(E_in_list)
8     print("E_in_average: ", E_in_average)
9
10 if __name__ == '__main__':
11     args = {
12         'filename_test': "hw3_test.dat",
13         'filename_train': "hw3_train.dat",
14         'feature': 11, # without include y
15         'iteration': 800,
16         'learning_rate': 0.001,
17         'repeat_time': 1000,
18         'size': 0
19     }
20
21     main(args)

```

```
1 | E_in_average: 0.8225428632462761
```

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

## P15

We use the same code above, then update the SGD steps and the error function. The updated functions show below.

Question assigned to the following page: [6.1](#)

```
1 def calculate_E_in_ce(x, y, w):
2     return np.mean(np.log(1 + np.exp(-y * np.dot(x, w))))
3
4 def SGD(args, x, y):
5     error_list = list()
6     w_list = list()
7     for i in range(args['repeat_time']):
8         random.seed()
9         w = np.zeros(args['feature'])
10        for j in range(args['iteration']):
11            pick = random.randint(0, args['size']-1)
12            # w = w + args['learning_rate'] * 2 * (y[pick] - np.dot(w, x[pick])) *
13            w = w + args['learning_rate'] * sigmoid(-y[pick] * np.dot(w, x[pick]))
14            error_list.append(calculate_E_in_ce(x, y, w))
15            w_list.append(w)
16    return w_list, error_list
```

```
1 E_in_average:  0.6571562557258926
```

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

## P16

We use the same code above, then update the initialization of  $w$  from  $w = 0$  to  $w = w_{lin}$ . The updated functions show below.

No questions assigned to the following page.

```

1 def calculate_w_lin(x, y):
2     # pseudo inverse: np.linalg.pinv(np.array([values]))
3     return np.dot(np.linalg.pinv(x), y)
4
5 def SGD(args, x, y):
6     error_list = list()
7     w_list = list()
8     for i in range(args['repeat_time']):
9         random.seed()
10        w = args['w_lin']
11        for j in range(args['iteration']):
12            pick = random.randint(0, args['size']-1)
13            # w = w + args['learning_rate'] * 2 * (y[pick] - np.dot(w, x[pick])) *
14            w = w + args['learning_rate'] * sigmoid(-y[pick] * np.dot(w, x[pick]))
15            error_list.append(calculate_E_in_ce(x, y, w))
16            w_list.append(w)
17    return w_list, error_list
18
19 def main(args):
20     x_train, y_train = read_file(args, args['filename_train'])
21     x_train = np.c_[np.ones(len(x_train)), x_train] # x_{n, 0}=1
22     args['feature'] = len(x_train[0])
23     args['w_lin'] = calculate_w_lin(x_train, y_train)
24
25     weight_list, E_in_list = SGD(args, x_train, y_train)
26     E_in_average = np.mean(E_in_list)
27     print("E_in_average: ", E_in_average)
28
29 if __name__ == '__main__':
30     args = {
31         'filename_test': "hw3_test.dat",
32         'filename_train': "hw3_train.dat",
33         'feature': 11, # without include y
34         'iteration': 800,
35         'learning_rate': 0.001,
36         'repeat_time': 1000,
37         'size': 0,
38         'w_lin': 0, # weight_vector is a list
39     }
40
41     main(args)

```

1 | E\_in\_average: 0.6051732441261142

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

No questions assigned to the following page.

## P17

```
1 import numpy as np
2 import random
3 from tqdm import trange
4
5 def calculate_E_in_01(x, y, w):
6     return np.mean(np.dot(x, w) * y <= 0)
7
8 def calculate_w_lin(x, y):
9     # pseudo inverse: np.linalg.pinv(np.array([values]))
10    return np.dot(np.linalg.pinv(x), y)
11
12 def read_file(args, filename):
13    data = np.loadtxt(filename, dtype=float)
14    x = data[:, :-1]
15    y = data[:, -1]
16    args['features'] = len(x[0])
17    args['size'] = len(x)
18    return x, y
19
20 def SGD(args, x, y):
21    error_list = list()
22    w_list = list()
23    for i in trange(args['repeat_time']):
24        random.seed()
25        w = args['w_lin']
26        for j in range(args['iteration']):
27            pick = random.randint(0, args['size']-1)
28            # w = w + args['learning_rate'] * 2 * (y[pick] - np.dot(w, x[pick])) *
29            w = w + args['learning_rate'] * sigmoid(-y[pick] * np.dot(w, x[pick]))
30            error_list.append(calculate_E_in_01(x, y, w))
31            w_list.append(w)
32    return w_list, error_list
33
34 def sigmoid(s):
35     return (1 / (1 + np.exp(-s)))
```



No questions assigned to the following page.

```
1 def main(args):
2     x_train, y_train = read_file(args, args['filename_train'])
3     x_train = np.c_[np.ones(len(x_train)), x_train] # x_{n, 0}=1
4     args['feature'] = len(x_train[0])
5     args['w_lin'] = calculate_w_lin(x_train, y_train)
6     weight_list, E_in_list = SGD(args, x_train, y_train)
7
8     x_test, y_test = read_file(args, args['filename_test'])
9     x_test = np.c_[np.ones(len(x_test)), x_test] # x_{n, 0}=1
10    args['feature'] = len(x_train[0])
11    E_out_list = np.array([calculate_E_in_01(x_test, y_test, weight_list[i]) for i
12                           print("|\u0395_in-\u0395_out|: ", np.mean(np.abs(E_in_list - E_out_list)))
13
14 if __name__ == '__main__':
15     args = {
16         'filename_test': "hw3_test.dat",
17         'filename_train': "hw3_train.dat",
18         'feature': 11, # without include y
19         'iteration': 800,
20         'learning_rate': 0.001,
21         'repeat_time': 1000,
22         'size': 0,
23         'w_lin': 0, # weight_vector is a list
24     }
25
26     main(args)
```

```
1 |\u0395_in-\u0395_out|:  0.030122500000000017
```

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

No questions assigned to the following page.

## P18

```
1 import numpy as np
2 from tqdm import trange
3
4 def calculate_E_in_01(x, y, w):
5     return np.mean(np.dot(x, w) * y <= 0)
6
7 def calculate_w_lin(x, y):
8     # pseudo inverse: np.linalg.pinv(np.array([values]))
9     return np.dot(np.linalg.pinv(x), y)
10
11 def read_file(args, filename):
12     data = np.loadtxt(filename, dtype=float)
13     x = data[:, :-1]
14     y = data[:, -1]
15     return x, y
16
17 def main(args):
18     x_train, y_train = read_file(args, args['filename_train'])
19     x_train = np.c_[np.ones(len(x_train)), x_train] # x_{n, 0}=1
20     x_test, y_test = read_file(args, args['filename_test'])
21     x_test = np.c_[np.ones(len(x_test)), x_test] # x_{n, 0}=1
22
23     weight = calculate_w_lin(x_train, y_train)
24     E_in = calculate_E_in_01(x_train, y_train, weight)
25     E_out = calculate_E_in_01(x_test, y_test, weight)
26
27     print("E_in: ", E_in)
28     print("E_out: ", E_out)
29     print("|E_in-E_out|: ", np.abs(E_in - E_out))
30
31 if __name__ == '__main__':
32     args = {
33         'filename_test': "hw3_test.dat",
34         'filename_train': "hw3_train.dat",
35     }
36
37     main(args)
```

```
1 E_in:  0.3
2 E_out:  0.34
3 |E_in-E_out|:  0.04000000000000036
```

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

No questions assigned to the following page.

## P19

We use the same code above, then update the transformation function and apply it to the training and testing dataset. The updated functions show below.

```
1 def transformation(x, Q):
2     # x_0 is included
3     x_origin = x[:, 1:]
4     transformed_x = x.copy()
5     for q in range(2, Q+1):
6         transformed_x = np.hstack((transformed_x, x_origin**q))
7     return transformed_x
8
9 def main(args):
10    x_train, y_train = read_file(args, args['filename_train'])
11    x_train = np.c_[np.ones(len(x_train)), x_train] # x_{n, 0}=1
12    x_test, y_test = read_file(args, args['filename_test'])
13    x_test = np.c_[np.ones(len(x_test)), x_test] # x_{n, 0}=1
14    x_train = transformation(x_train, args['Q'])
15    x_test = transformation(x_test, args['Q'])
16
17    weight = calculate_w_lin(x_train, y_train)
18    E_in = calculate_E_in_01(x_train, y_train, weight)
19    E_out = calculate_E_in_01(x_test, y_test, weight)
20
21    print("E_in: ", E_in)
22    print("E_out: ", E_out)
23    print("|E_in-E_out|: ", np.abs(E_in - E_out))
24
25 if __name__ == '__main__':
26    args = {
27        'filename_test': "hw3_test.dat",
28        'filename_train': "hw3_train.dat",
29        'Q': 2,
30    }
31
32    main(args)
```

```
1 E_in:  0.23
2 E_out:  0.3125
3 |E_in-E_out|:  0.0824999999999999
```

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

No questions assigned to the following page.

## P20

We use the same code above, then update the parameter of args. The updated functions show below.

```
1 if __name__ == '__main__':
2     args = {
3         'filename_test': "hw3_test.dat",
4         'filename_train': "hw3_train.dat",
5         'Q': 8,
6     }
7
8     main(args)
```

```
1 E_in:  0.0
2 E_out: 0.415
3 |E_in-E_out|: 0.415
```

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