

# HW2 Solutions

● Graded

Student

Chih-Hao Liao

Total Points

90 / 100 pts

Question 1

Problem 2

20 / 20 pts

- 0 pts Correct

- 5 pts Minor mistake or no further explanation

- 10 pts Major mistake

- 20 pts No or wrong answer

Question 2

Problem 7

20 / 20 pts

+ 20 pts Correct

+ 5 pts Correct formulation of likelihood function

+ 5 pts Correct formulation of log likelihood function

+ 5 pts Correct derivation of the derivative

+ 5 pts Correct minimizer

+ 0 pts Fundamentally wrong

+ 0 pts Blank

**Question 3**

## Problem 8

Resolved 10 / 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

1 You can always say desired result is one of choice, so only saying the gradient of Ein is equal to correct choice cannot convince TA you really know the solution.

C Regrade Request

Submitted on: May 16

I calculate this equation through Wolfram alpha, and the result comes from the equation, "d/dx-(ln(1+axb+|abx|)-ln(2+2|abx|))", where I replace x,y,w to a,b,x, then I can simply get the result.

I'm not sure where causes the all wrong since the scores should be graded through the rules on the right side of the grade scope, therefore, I need to regrade my scores.

In fact, your steps to get Ein are correct. However, the problem mainly focuses on the answer for the gradient of Ein. In your solution, the steps from Ein to the gradient of Ein are missing and only the final result was presented here. Hence, I cannot know whether you really made a clear and complete calculation. If you have read the grading principle, you may know there are other people applying wrong calculation but still having correct choice, which is another reason why I emphasize on this part. If you used Wolfram alpha to help your calculation, then providing the screenshot or just typing the process will make me understand what happened in your solution. On the other hand, the grading means "Missing (important part of) solution" instead of "Wrong solution". Sorry for any inconvenience. If you still have problems, feel free to discuss with me.

Reviewed on: May 16

C Regrade Request

Submitted on: May 16

Here's the procedure by that I calculate the result from WolframAlpha, as you can see in the link below, it simply shows the result is "-1(ab/(|abx|+1)(|abx|+abx+1))". Therefore, there's no doubt that I really understand the process of why I need to calculate the gradient of E\_in and how it be calculated.

Here's the link for you to check: <https://www.wolframalpha.com/input/?i=d%2Fdx-%28ln%281%2Baxb%2B%7Caxb%7C%29-In%282%2B2%7Caxb%7C%29%29>

In fact, I mean the link should be written in your homework in the begin instead of providing it when regrading. Otherwise, it is unfair to students whose answer was clear, complete and convincing originally. Besides, based on the link, I found that the link said |abx| can be equal to |a||b||x|. However, please seriously think about the validity of it especially whether you can claim |w^Tx| = |w||x|. The mistake may come from that your misuse between scalar and vector. Even though now I can know you understand the calculation of gradient, there are still some errors. I cannot give you full points after regrading. Hopes you can understand the reason. If you still have problems, feel free to discuss with me.

#### Question 4

##### Problem 9

20 / 20 pts

✓ - 0 pts Correct and with clear and complete derivation steps

- 5 pts A minor mistake exists in the answer

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

- 15 pts Some reasonable effort but fundamentally wrong

- 20 pts Wrong answer

- 20 pts No solution / Showed no derivation step

- 20 pts Wrong page

- 20 pts Unclear solution / Almost missing solution / Wrong solution

- 20 pts No page selected

It is better to use X than x to distinguish matrices from scalars.

#### Question 5

##### Problem 12

20 / 20 pts

✓ - 0 pts Correct

- 5 pts Minor mistake

- 0 pts Wrong  $\theta$  range (e.g.  $\theta > 0.5$ )

- 10 pts Major mistake

- 10 pts Laking of considering in some case of  $\theta$  (e.g.  $\theta = -\infty$ )

- 20 pts Wrong answer

- 20 pts No answer

#### Question 6

##### Coding Part

0 / 0 pts

##### 6.1 Problem 17

0 / 0 pts

✓ + 0 pts Correct

- 160 pts Without source code

- 20 pts Cannot understand the implementatoin

- 80 pts Unclear submission.

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

# HTML\_2023\_HW2

---

tags: Personal

Discuss with anonymous and TAs.

## Theory of Generalization

---

### P1

Define a perceptron  $h(x) = \text{sign}(w^T x)$  s.t.  $w_1(x_1 - 11.26) + w_2(x_2 - 62.11) = -w_0, \forall x \in \mathbb{R}^2$ . Consider a two-dimensional plane filled with numerous points. Visualize a line that gradually rotates around the fixed lucky point (11.26, 62.11). Since the line passes over each point on the plane, the classification of that point changes, thereby creating a new state. After the line completes a full rotation around the lucky point, it crosses over a total of  $2N$  points, which creates  $2N$  distinct states.

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

### P2

Consider the hypothesis set  $h_m(x) = \text{sign}(w_m^T x), m = 1, 2, \dots, 1126$  to be 1126 binary classifiers. In order to shatter all possible outputs for any combination of  $n$  different points, we need  $2^n$  kinds of different results. Since we have hypothesis set  $h_m(x) = \text{sign}(w_m^T x), m = 1, 2, \dots, 1126$ , there can exist at most 1126 outputs. Thus, we need to find the largest  $n$  such that  $2^n = m_H(d) \leq 1126$ . Solving for  $n$ , we get  $n = \log_2(1126)$ .

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

### P3

[a]

By sorting the possible result, a key limitation of the problem is that the sequence {o,x,o,x,o} can never occur when the values of the wrong sign {x} are sorted in increasing order, which means  $d_{vc} = 4$ .

[b]

By sorting the possible result, a key limitation of the problem is that the sequence {1, -1, 1, -1, 1} can never occur since the polynomial of degree 3 only pass  $\sum_{i=0}^3 w_i x^i = y = 0$  for three times, which means  $d_{vc} = 4$ .

[c]

For any  $n \in \mathbb{N}$ , consider the set of points  $\{x_1, x_2, \dots, x_n\}$  with arbitrary labels  $\{y_1, y_2, \dots, y_n\} \in \{-1, +1\}^n$ . Then let

$$x_j = 1126^{-j}$$
$$w = \pi(1 + \sum_{i=1}^n 1126^i y'_i), \text{ where } y'_i = \frac{1 - y_i}{2}$$

For any  $j \in [1, n]$  we have

No questions assigned to the following page.

$$\begin{aligned}
h(x_j) &= \sin(1126^{-j} \times \pi(1 + \sum_{i=1}^n 1126^i y'_i)) \\
&= \sin(\pi(1126^{-j} + (\sum_{i=1}^n 1126^{i-j} y'_i))) \\
&\Rightarrow \sin(1126^{-j} \times \pi(1 + \sum_{i=1}^n \frac{1126^i(1-y_i)}{2})) \\
&\Rightarrow \sin(\pi(1126^{-j} + (\sum_{i=1}^n \frac{1126^{i-j}(1-y_i)}{2}))) \\
&\Rightarrow \sin(\pi(1126^{-j} + \sum_{i:i < j, y_i = -1} \frac{1126^{i-j}(1-y_i)}{2})) \\
&\Rightarrow \sin(\pi(1126^{-j} + \frac{(1-y_i)}{2} + \sum_{i:i < j, y_i = -1} 2 \times \\
&= \sin(\pi(1126^{-j} + (\sum_{i=1}^{j-1} 1126^{i-j} y'_i) + y'_j + (\sum_{i=1}^{n-j} 1126^i y'_i))) \Rightarrow \sin(\pi(1126^{-j} + \frac{(1-y_i)}{2} + \sum_{i:i < j, y_i = -1} 112
\end{aligned}$$

For any  $y_i = 1$  and any  $i < j$ , the last term can be dropped from the sum since it only contributes multiples of  $2\pi$ , which causes no change in value. By using the fact that  $\sin(\pi + x) = -\sin(x)$ , it is trivial that the summation in the last term and the second term is always less than 1. Since  $y'_i$  the remaining term, that is

$$\sin(\pi(1126^{-j} + (\sum_{i=1}^{j-1} 1126^{i-j} y'_i) + y'_j)) = \sin(\pi(\sum_{i=1}^{j-1} 1126^{-i} y'_i + 1126^{-j} + y'_j))$$

can be upper and lower bound as follows

$$\begin{aligned}
\pi(\sum_{i=1}^{j-1} 1126^{-i} y'_i) + 1126^{-j} + y'_j &\leq \pi(\sum_{i=1}^{j-1} 1126^{-j} + y'_j) \leq \pi(1 + y'_j) \\
\pi(\sum_{i=1}^{j-1} 1126^{-i} y'_i) + 1126^{-j} + y'_j &> \pi y'_j \\
\therefore \begin{cases} h(x_j) = 1 = y_j, & \text{where } 0 < wx_j < \pi \quad \text{if } y_j = 1 \\ h(x_j) = -1 = y_j, & \text{where } \pi < wx_j < 2\pi \quad \text{if } y_j = -1 \end{cases}
\end{aligned}$$

As a result,  $h(x_j) = y_j$  for all  $j$ , and the set  $\{1126^{-1}, \dots, 1126^n\}$  can be shattered for any value for  $n$ , which means  $d_{vc} = \infty$ .

[d]

It is easy to see that a set of 4 points with coordinate  $(0, 0), (1, -1), (2, -2), (3, -3)$  can be shattered by such triangle. However, it is not possible to completely shatter a set of 5 points using triangles. Consider this situation as a convex shape, and labeling all points positively except for the one within the interior of the convex hull is not possible because a triangle convex shape is a degenerate case where no points are in the interior of the convex hull. This is trivial that  $d_{vc} = 4$ .

[e]

It's trivial that the set of 3 points with coordinates  $(1, 0), (0, 1), (-1, 0)$  can be shattered by axis-aligned squares while there is no set of 4 points that can be fully shattered.

Support that  $P_{top}$  e the highest point,  $P_{bottom}$  the lowest,  $P_{left}$  the leftmost, and  $P_{right}$  the rightmost. Assuming without loss of generality that the difference  $d_{bt}$  of y-coordinates between  $P_{bottom}$  and  $P_{top}$  is greater than the difference  $d_{lf}$  of x-coordinates between  $P_{left}$  and  $P_{right}$ . In such a scenario, it is not possible to label  $P_{bottom}$  and  $P_{top}$  positively while labeling  $P_{left}$  and  $P_{right}$  negatively. Therefore  $d_{vc} = 3$ .

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

Ref: VC Dimension and Model Complexity (<https://www.cs.cmu.edu/~epxing/Class/10701/slides/lecture16-VC.pdf>)

Ref: 10-701 Machine Learning Fall 2011: Homework 2 Solutions

([https://www.cs.cmu.edu/~epxing/Class/10701-11f/HW/HW2\\_solution.pdf](https://www.cs.cmu.edu/~epxing/Class/10701-11f/HW/HW2_solution.pdf))

Ref: Foundations of Machine Learning (<https://cs.nyu.edu/~mohri/ml16/sol2.pdf>)

No questions assigned to the following page.

Ref: [Mehryar Mohri Foundations of Machine Learning 2014](https://cs.nyu.edu/~mohri/ml14/hw2.pdf) (<https://cs.nyu.edu/~mohri/ml14/hw2.pdf>).

Ref: [Advanced Machine Learning: Problem Set II Solutions](https://cse.iitkgp.ac.in/~pabitra/course/aml/hw2_19.pdf)

([https://cse.iitkgp.ac.in/~pabitra/course/aml/hw2\\_19.pdf](https://cse.iitkgp.ac.in/~pabitra/course/aml/hw2_19.pdf)).

Ref: [Question about VC-dimension \[closed\]](https://math.stackexchange.com/questions/3940837/question-about-vc-dimension) (<https://math.stackexchange.com/questions/3940837/question-about-vc-dimension>).

Ref: [CS683 Scribe Notes](https://www.cs.cornell.edu/courses/cs683/2008sp/lecture%20notes/683notes_0428.pdf) ([https://www.cs.cornell.edu/courses/cs683/2008sp/lecture%20notes/683notes\\_0428.pdf](https://www.cs.cornell.edu/courses/cs683/2008sp/lecture%20notes/683notes_0428.pdf)).

## P4

It is possible to demonstrate  $\mathcal{H}$  in two-dimensional case. From lecture slide `class0316.pdf` Page 40, we can simplify  $\mathcal{H}$  to be a many positive intervals question. For one positive interval, we have 2 free parameters, and  $d_{vc} = 2$ , for two positive intervals, we have 4 free parameters, and  $d_{vc} = 4$ , therefore, for  $M$  positive intervals, we have  $2M$  free parameters, which are  $a_1, b_1, a_2, b_2, \dots, a_M, b_M$ , and it is trivial that  $d_{vc} = 2M$ . For more rigorous proof, I will prove it below.

To claim that  $d_{vc}(\mathcal{H}) \geq 2M$ , consider a set  $\{x_1, x_2, \dots, x_{2M}\}$ , where  $x_i = e_i$  if  $i \in [M]$ , and  $x_i = -e_{i-M}$  if  $i > M$ , then it is trivial that all  $x$  can be shattered. Besides, let  $\{y_1, y_2, \dots, y_{2M}\} \subseteq \{-1, 1\}^{2M}$ , and let  $A$  be a set of one-hot vectors in  $\mathbb{R}^d$  and their negations.

$$\begin{aligned} A &= \{l_i \mid 1 \leq i \leq M\} \cup \{-l_i \mid 1 \leq i \leq M\} \\ \text{if } l_i^M &= (0, \dots, 0, 1, 0, \dots, 0) \text{ where only } i\text{th dimension of } l_i = 1 \end{aligned}$$

Given any  $B \subseteq A$ , we can choose

$$\begin{array}{ll} \text{if both } l_i \text{ and } -l_i \in B & a_i = -2 \quad b_i = 2 \\ \text{if } l_i \in B, -l_i \notin B & a_i = 0 \quad b_i = 2 \\ \text{if } l_i \notin B, -l_i \in B & a_i = -2 \quad b_i = 0 \\ \text{otherwise} & a_i = 0 \quad b_i = 0 \end{array}$$

To claim that  $d_{vc}(\mathcal{H}) \leq 2M$ , consider any  $A \subseteq \mathbb{R}^d$  where  $|A| > 2d$ , take subset  $B \subseteq A$  where for every  $1 \leq i \leq M$ , choose points  $ll_i$  and  $rr_i$  with maximum and minimum values respectively in the  $i$ th coordinate. For every  $\mathcal{H}$  that includes all members of  $B$ , we have  $A \subseteq \mathcal{H}$ , thus,  $\mathcal{H}$  cannot cut  $B$  from  $A$ , which means  $\mathcal{H}$  can not shatter  $A$ .

By shifting all parameters to be larger than 0, then we prove the statements.

Moreover, for the proof of a set  $C$  whose size is at least  $2M + 1$ , we can prove it by using pigeonhole principle. If there exists an element  $x \in C$  s.t.  $\forall j \in [M]$ , there exists

$x' \in C$  with  $x'_j \leq x_j$ , and  $x'' \in C$  with  $x''_j \leq x_j$ . With collinearity, labeling in which  $x$  is negative, and the rest of the elements in  $C$  are positive can not be obtained, which means  $\mathcal{H}$  cannot shatter  $C$  with size at least  $2M + 1$ .

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

Ref: [CS 485/685 Course Notes P. 14](https://www.richardwu.ca/notes/cs485-notes.pdf) (<https://www.richardwu.ca/notes/cs485-notes.pdf>).

Ref: [Understanding Machine Learning Solution Manual P. 13](https://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/MLbookSol.pdf)

(<https://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/MLbookSol.pdf>).

Ref: [Understanding Machine Learning: From Theory to Algorithms P.79](https://mcube.lab.nycu.edu.tw/~cfung/docs/books/shalev_shwartz2014ML.pdf)

([https://mcube.lab.nycu.edu.tw/~cfung/docs/books/shalev\\_shwartz2014ML.pdf](https://mcube.lab.nycu.edu.tw/~cfung/docs/books/shalev_shwartz2014ML.pdf)).

## P5

It's trivial that the following is necessary conditions for  $d_{vc} \leq d$

- some set of  $d$  distinct inputs is shattered by  $\mathcal{H}$
- some set of  $d + 1$  distinct inputs is not shattered by  $\mathcal{H}$
- any set of  $d + 1$  distinct inputs is not shattered by  $\mathcal{H}$

Since  $d_{vc}$  is the maximum that  $m_H(N) = 2^N$ , and  $m_H(N) = 2^N$  is the most number of dichotomies of  $N$  inputs. Thus, if we cannot find  $2^{d+1}$  dichotomies on some and any  $d + 1$  inputs,  $m_H(d + 1) < 2^{d+1}$  and hence,  $d_{vc} < d + 1$ , that is  $d_{vc} \leq d$ . On the other hand, if we can find  $2^d$

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

dichotomies on some  $d$  inputs, then  $m_H(d) = 2^d$ , that is  $d_{vc} \geq d$ . Although the statement "any set of  $d$  distinct inputs is shattered by  $\mathcal{H}$ " meets the criteria while it is not a necessary conditions.

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

Ref: Learning from data](<https://work.caltech.edu/telecourse.html>) (<https://work.caltech.edu/telecourse.html>)

## Linear Models

---

### P6

In order to find optimal  $w$ , we can find  $w_{LIN}$  such that  $\nabla E_{in}(w_{LIN}) = 0$

$$\begin{aligned} E_{in}(w) &= \frac{1}{N} \sum_{n=1}^N (h(x_n) - y_n)^2 = \frac{1}{N} \sum_{n=1}^N (w_n x_n - y_n)^2 \\ \nabla E_{in}(w) &= \frac{1}{N} \cdot \frac{\partial}{\partial w} \left( \sum_{n=1}^N (w^T x^T x w - 2w^T x^T y + y^T y) \right) \\ &= \frac{1}{N} \sum_{n=1}^N (2x^T x w - 2x^T y) = \frac{2}{N} \sum_{n=1}^N (x^T x w - x^T y) \\ &= 0 \\ 0 &= \frac{2}{N} \sum_{n=1}^N (x^T x w - x^T y) \\ w_{LIN} &= \sum_{n=1}^N ((x^T x)^{-1} x^T y) = \sum_{n=1}^N \left( \frac{x^T y}{x^T x} \right) \\ &= \frac{\sum_{n=1}^N y_n x_n}{\sum_{n=1}^N x_n^2} \end{aligned}$$

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

### P7

We have

$$\bar{x} = \frac{1}{N} \sum_{n=1}^N x_n$$

[a]

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

$$\begin{aligned}
P(x) &= \frac{e^{-\lambda} \lambda^x}{x!} \Rightarrow P(x) = \prod_{n=1}^N f_\theta(x_n) \\
L(x|\lambda) &= \prod_{n=1}^N \frac{e^{-\lambda} \lambda^{x_n}}{x_n!} \\
\ln(L(x|\lambda)) &= \ln\left(\prod_{n=1}^N \frac{e^{-\lambda} \lambda^{x_n}}{x_n!}\right) = \sum_{n=1}^N \ln\left(\frac{e^{-\lambda} \lambda^{x_n}}{x_n!}\right) \\
&= \sum_{n=1}^N (\ln(e^{-\lambda}) + \ln(\lambda^{x_n}) - \ln(x_n!)) \\
&= \sum_{n=1}^N (-\lambda + x_n \ln(\lambda) - \ln(x_n!)) \\
&= -n\lambda + \ln(\lambda) \sum_{n=1}^N (x_n) - \underbrace{\sum_{n=1}^N (\ln(x_n!))}_{\text{constant}} \\
\frac{\partial \ln(L(x|\lambda))}{\partial \lambda} &= \frac{\partial}{\partial \lambda} (-n\lambda + \ln(\lambda) \sum_{n=1}^N (x_n) - \sum_{n=1}^N (\ln(x_n!))) \\
&= -n + \frac{1}{\lambda} \sum_{n=1}^N x_n = 0 \\
\lambda^* &= \frac{\sum_{n=1}^N x_n}{n} = \bar{x}
\end{aligned}$$

[b]

$$\begin{aligned}
p(x) &= \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(x-\mu)^2} \\
L(x|\mu) &= \prod_{n=1}^N \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(x-\mu)^2} \\
&= \frac{1}{\sqrt{(2\pi)^N}} e^{-\frac{1}{2} \sum_{n=1}^N (x_n - \mu)^2} \\
\ln(L(x|\mu)) &= \ln\left(\frac{1}{\sqrt{(2\pi)^N}} e^{-\frac{1}{2} \sum_{n=1}^N (x_n - \mu)^2}\right) \\
&= -\frac{N}{2} (\ln(2\pi)) - \frac{1}{2} \sum_{n=1}^N (x_n - \mu)^2 \\
\frac{\partial \ln(L(x|\mu))}{\partial \mu} &= \frac{\partial}{\partial \mu} \left(-\frac{N}{2} (\ln(2\pi)) - \frac{1}{2} \sum_{n=1}^N (x_n - \mu)^2\right) \\
&= \sum_{n=1}^N (x_n - \mu) = \sum_{n=1}^N x_n - n\mu = 0 \\
\mu^* &= \frac{1}{n} \sum_{n=1}^N x_n = \bar{x}
\end{aligned}$$

[c]

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

$$\begin{aligned}
p(x) &= \frac{1}{2} e^{-|x-\mu|} \\
L(x|\mu) &= \prod_{n=1}^N \frac{1}{2} e^{-|x_n-\mu|} \\
&= \frac{1}{2^N} e^{-\sum_{n=1}^N |x_n-\mu|} \\
\ln(L(x|\mu)) &= \ln\left(\frac{1}{2^N} e^{-\sum_{n=1}^N |x_n-\mu|}\right) \\
&= -N \ln(2) - \sum_{n=1}^N |x_n - \mu| \\
\frac{\partial \ln(L(x|\mu))}{\partial \mu} &= \frac{\partial}{\partial \mu} (-N \ln(2) - \sum_{n=1}^N |x_n - \mu|) \\
&= \sum_{n=1}^N |x_n - \mu| \\
\therefore \frac{\partial |x|}{\partial x} &= \frac{\partial \sqrt{x^2}}{\partial x} = x(x^2)^{-\frac{1}{2}} = \frac{x}{|x|} = \text{sign}(x) \\
\therefore \frac{\partial \ln(L(x|\mu))}{\partial \mu} &= \sum_{n=1}^N \text{sign}(x_n - \mu) = 0
\end{aligned}$$

Since  $N$  can be odd or even, we have two cases.

If  $N$  is odd,  $\mu^* = \text{median}(x_1, x_2, \dots, x_N)$ , and there are  $\frac{N-1}{2}$  cases  $x_n < \mu$  and  $\frac{N-1}{2}$  cases  $x_n > \mu$

If  $N$  is even, we can't simply choose one  $x_n$  which will satisfy the condition above. while we can choose either  $x_{\frac{N}{2}}$  or  $x_{\frac{N+1}{2}}$ .

As a result,  $\mu^* = \text{median}(x_1, x_2, \dots, x_N)$

[d]

$$\begin{aligned}
P(x) &= (1-\theta)^{x-1} \theta \\
L(x|\theta) &= \prod_{n=1}^N (1-\theta)^{x_n-1} \theta \\
&= (1-\theta)^{\sum_{n=1}^N x_n - N} \theta^N \\
\ln(L(x|\theta)) &= \ln((1-\theta)^{\sum_{n=1}^N x_n - N} \theta^N) \\
&= (\sum_{n=1}^N x_n - N) \ln(1-\theta) + N \ln \theta \\
\frac{\partial \ln(L(x|\theta))}{\partial \theta} &= -\frac{(\sum_{n=1}^N x_n - N)}{(1-\theta)} + \frac{N}{\theta} = 0 \\
\theta^* &= \frac{N}{\sum_{n=1}^N x_n} = \frac{1}{\bar{x}}
\end{aligned}$$

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

Ref: [MLE for a Poisson Distribution \(Step-by-Step\)](https://www.statology.org/mle-poisson-distribution/) (<https://www.statology.org/mle-poisson-distribution/>)

Ref: [Chapter 8.3. Maximum Likelihood Estimation](https://mathweb.ucsd.edu/~gptesler/283/slides/283_mle_19-handout.pdf) ([https://mathweb.ucsd.edu/~gptesler/283/slides/283\\_mle\\_19-handout.pdf](https://mathweb.ucsd.edu/~gptesler/283/slides/283_mle_19-handout.pdf))

Ref: [Normal distribution - Maximum Likelihood Estimation](https://www.statlect.com/fundamentals-of-statistics/normal-distribution-maximum-likelihood) (<https://www.statlect.com/fundamentals-of-statistics/normal-distribution-maximum-likelihood>)

Ref: [Topic 15: Maximum Likelihood Estimation\\*](https://www.math.arizona.edu/~jwatkins/o-mle.pdf) (<https://www.math.arizona.edu/~jwatkins/o-mle.pdf>)

Ref: [Maximum Likelihood Estimation of Gaussian Parameters](http://jmeyer.github.io/machinelearning/2017/08/18/mle.html) (<http://jmeyer.github.io/machinelearning/2017/08/18/mle.html>)

Ref: [Finding the maximum likelihood estimator](https://math.stackexchange.com/questions/240496/finding-the-maximum-likelihood-estimator) (<https://math.stackexchange.com/questions/240496/finding-the-maximum-likelihood-estimator>)

Ref: [HOMEWORK 7 SOLUTIONS](https://web.stanford.edu/class/archive/stats/stats200/stats200.1172/Solutions07.pdf) (<https://web.stanford.edu/class/archive/stats/stats200.stats200.1172/Solutions07.pdf>)

Questions assigned to the following page: [2](#), [3](#), and [4](#)

Ref: [Examples of Parameter Estimation based on Maximum Likelihood \(MLE\): the exponential distribution and the geometric distribution](#)  
[\(\[https://www.projectrhea.org/rhea/index.php/MLE Examples: Exponential and Geometric Distributions Old Kiwi\]\(https://www.projectrhea.org/rhea/index.php/MLE\_Examples:\_Exponential\_and\_Geometric\_Distributions\_Old\_Kiwi\)\)](https://www.projectrhea.org/rhea/index.php/MLE_Examples:_Exponential_and_Geometric_Distributions_Old_Kiwi)

## P8

$$\begin{aligned}
 \bar{h}(x) &= \frac{1 + w^T x + |w^T x|}{2 + 2|w^T x|} \\
 \theta(s) &= \frac{1 + s + |s|}{2 + 2|s|} \\
 L(\bar{h}(y_n x_n)) &= \prod_{n=1}^N \frac{1 + y_n w^T x_n + |y_n w^T x_n|}{2 + 2|y_n w^T x_n|} \\
 \ln(L(\bar{h}(y_n x_n))) &= \ln\left(\prod_{n=1}^N \frac{1 + y_n w^T x_n + |y_n w^T x_n|}{2 + 2|y_n w^T x_n|}\right) \\
 &= \frac{1}{N} \sum_{n=1}^N -\ln \theta(y_n w^T x_n) \\
 &= \frac{1}{N} \sum_{n=1}^N \underbrace{-(\ln(1 + y_n w^T x_n + |y_n w^T x_n|) - \ln(2 + 2|y_n w^T x_n|))}_{E_{in}(w)} \\
 \frac{\partial \ln(L(\bar{h}(y_n x_n)))}{\partial w} &= \frac{\partial}{\partial w} \left( \frac{1}{N} \sum_{n=1}^N -(\ln(1 + y_n w^T x_n + |y_n w^T x_n|) - \ln(2 + 2|y_n w^T x_n|)) \right) \\
 &= -\frac{1}{N} \sum_{n=1}^N \frac{y_n x_n}{(1 + y_n w^T x_n + |y_n w^T x_n|)(1 + |y_n w^T x_n|)}
 \end{aligned}$$

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

## Beyond Gradient Descent

---

### P9

$$\begin{aligned}
 E_{in}(w) &= \frac{1}{N} \|xw - y\|^2 = \frac{1}{N} (w^T x^T x w - 2w^T x^T y + y^T y) \\
 \nabla E_{in}(w) &= \frac{2}{N} (x^T x w - x^T y) \\
 \nabla^2 E_{in}(w) &= \frac{2}{N} x^T x
 \end{aligned}$$

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

### P10

$$\begin{aligned}
 E_{in}(w) &= \frac{1}{N} \|xw - y\|^2 = \frac{1}{N} (w^T x^T x w - 2w^T x^T y + y^T y) \\
 \nabla E_{in}(w) &= \frac{2}{N} (x^T x w - x^T y) = 0 \\
 w_{LIN} &= \underbrace{(x^T x)^{-1} x^T}_\text{pseudo-inverse } x^\dagger y = x^\dagger y
 \end{aligned}$$

Therefore, when applying Newton method on linear regression, we can simply calculate the pseudo-inverse, then calculate the result  $w_{LIN}$ , which means it only causes 1 iteration.

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

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

## Decision Stumps

---

### P11

The growth function of decision stumps is  $m_H(N) = 2N$ , hence,  $m_H(2N) = 4N$ . Then we have

$$\mathbb{P}\left[\exists h \in \mathcal{H} \text{ s.t. } |E_{in}(h) - E_{out}(h)| > \epsilon\right] \leq 4 \cdot m_H(2N) \cdot \exp\left(-\frac{1}{8}\epsilon^2 N\right)$$

$$\delta \leq 4 \cdot m_H(2N) \cdot \exp\left(-\frac{1}{8}\epsilon^2 N\right)$$

$$\delta \leq 4 \cdot 4N \cdot \exp\left(-\frac{1}{8}\epsilon^2 N\right)$$

$$\epsilon \leq \sqrt{\frac{8}{N} \ln\left(\frac{16N}{\delta}\right)}$$

$$[a] \Rightarrow N = 100, \quad \epsilon = 0.774428 \quad > 0.05$$

$$[b] \Rightarrow N = 1000, \quad \epsilon = 0.0958634 \quad > 0.05$$

$$[c] \Rightarrow N = 10000, \quad \epsilon = 0.0114284 \quad > 0.05$$

$$[d] \Rightarrow N = 100000, \quad \epsilon = 0.00132705 \quad < 0.05$$

$$[e] \Rightarrow N = 1000000, \quad \epsilon = 0.000151125 \quad < 0.05$$

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

### P12

There exists a probability  $\tau$  to flip the output, which causes the noise. On the other hand, there exists a probability  $1 - \tau$  without changing the output. Therefore,  $E_{out}(h, \tau)$  will equal to  $[(\text{noise})x(\text{classified correct in origin}) + (\text{no noise})x(\text{classified incorrect in origin})]$ .

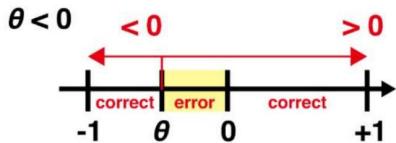
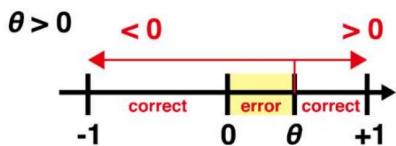
$$\begin{aligned} E_{out}(h, \tau) &= (1 - \tau)E_{out}(h, 0) + \tau(1 - E_{out}(h, 0)) \\ &= E_{out}(h, 0) - \tau E_{out}(h, 0) + \tau - \tau E_{out}(h, 0) \\ &= E_{out}(h, 0)(1 - 2\tau) + \tau \end{aligned}$$

Since  $E_{out}(h, 0)$  means there are no noise and  $h(+1, \theta)$  means the direction indicator  $s = +1$ , we can only consider the relationship between  $x$  and  $\theta$  in the absence of any noise.

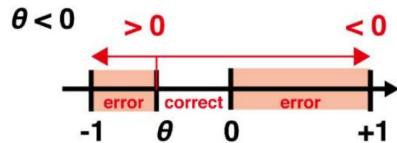
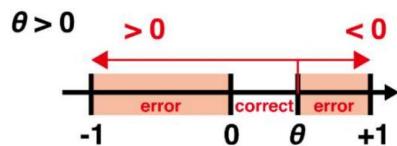
- For  $s = +1$ 
  - $x > \theta \Rightarrow h_{s, \theta}(x) = +1$
  - $x \leq \theta \Rightarrow h_{s, \theta}(x) = -1$
- For  $s = -1$ 
  - $x > \theta \Rightarrow h_{s, \theta}(x) = -1$
  - $x \leq \theta \Rightarrow h_{s, \theta}(x) = +1$

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

when  $S = +1$ , there have 2 conditions:



when  $S = -1$ , there have 2 conditions:



$$\begin{aligned}
 E_{out}(h_{+1}, 0) &= \mathbb{P}[y = f(x) = +1, h_{s,\theta}(x) = -1] + \mathbb{P}[y = f(x) = -1, h_{s,\theta}(x) = +1] \\
 &= \frac{1}{2} \cdot \frac{\theta}{0.5 - (-0.5)} + \frac{1}{2} \cdot \frac{|\theta|}{0.5 - (-0.5)} \text{ where } \theta \in [-\infty, 0.5] \\
 &= \min(|\theta|, 0.5)
 \end{aligned}$$

Therefore, we have

$$\begin{aligned}
 E_{out}(h, \tau) &= E_{out}(h, 0)(1 - 2\tau) + \tau \\
 &= \min(|\theta|, 0.5)(1 - 2\tau) + \tau
 \end{aligned}$$

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

No questions assigned to the following page.

## P13

```
1 import math
2 import numpy as np
3 from tqdm import trange
4
5 def decision_stump(args, x, y):
6     # h(x) = s * sign (x - theta)
7     s = 1; theta = 0
8     theta_list = np.array([float('-inf')] + [((x[i]+x[i+1])/2) for i in range(0, x.shape[0]-1)])
9     error = float('inf')
10    for theta_hypothesis in theta_list:
11        y_pos = np.where(x > theta_hypothesis, 1, -1)
12        y_neg = np.where(x <= theta_hypothesis, 1, -1)
13        error_pos = sum(y_pos != y)
14        error_neg = sum(y_neg != y)
15        if error_pos > error_neg:
16            if error_neg < error:
17                error = error_neg
18                theta = theta_hypothesis
19                s = -1
20        else:
21            if error_pos < error:
22                error = error_pos
23                theta = theta_hypothesis
24                s = 1
25
26    if theta == float('-inf'): theta = -0.5
27    return s, theta, float(error/x.shape[0])
28
29 def generate_data(args):
30     # np.random.seed(args['seed']) # fixed seed may causes error
31     x = np.sort(np.random.uniform(args['range_neg'], args['range_pos'], args['size']))
32     y = np.sign(x)
33     y[y == 0] = -1
34     probability = np.random.uniform(0, 1, args['size'])
35     noise_ratio = 1 - args['tau']
36     y[probability > noise_ratio] *= -1
37     return x, y
38
39 def main(args):
40     total_E_in = 0; total_E_out = 0
41
42     for i in trange(args['repeat_time']):
43         x, y = generate_data(args)
44         s, theta, E_in = decision_stump(args, x, y)
45         E_out = min(math.fabs(theta), args['range_pos'])*(1-2*args['tau'])+args['tau']
46         total_E_in += E_in
47         total_E_out += E_out
48
49     print("Average total_E_in:", total_E_in/args['repeat_time'])
50     print("Average total_E_out:", total_E_out/args['repeat_time'])
51     print("mean of E_out(g,tau)-E_in(g):", (total_E_out-total_E_in)/args['repeat_time'])
52
53 if __name__ == '__main__':
54     args = {
55         'range_neg': -0.5,
56         'range_pos': 0.5,
57         'repeat_time': 100000,
58         'seed': 1126,
59         'size': 2,
60         'tau': 0
61     }
62
63     main(args)
64
```

No questions assigned to the following page.

```
1 | Average total_E_in: 0.0
2 | Average total_E_out: 0.2922032839476481
3 | mean of E_out(g,tau)-E_in(g): 0.2922032839476481
```

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

## P14

We use the same code above, then update the parameter of args.

```
1 | if __name__ == '__main__':
2 |     args = {
3 |         'range_neg': -0.5,
4 |         'range_pos': 0.5,
5 |         'repeat_time': 100000,
6 |         'seed': 1126,
7 |         'size': 128,
8 |         'tau': 0
9 |     }
10 |
11 main(args)
12 |
```

```
1 | Average total_E_in: 0.0
2 | Average total_E_out: 0.0038555886544969927
3 | mean of E_out(g,tau)-E_in(g): 0.0038555886544969927
```

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

## P15

We use the same code above, then update the parameter of args.

```
1 | if __name__ == '__main__':
2 |     args = {
3 |         'range_neg': -0.5,
4 |         'range_pos': 0.5,
5 |         'repeat_time': 100000,
6 |         'seed': 1126,
7 |         'size': 2,
8 |         'tau': 0.2
9 |     }
10 |
11 main(args)
```

```
1 | Average total_E_in: 0.0
2 | Average total_E_out: 0.3913878886120975
3 | mean of E_out(g,tau)-E_in(g): 0.3913878886120975
```

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

## P16

We use the same code above, then update the parameter of args.

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

```

1 if __name__ == '__main__':
2     args = {
3         'range_neg': -0.5,
4         'range_pos': 0.5,
5         'repeat_time': 100000,
6         'seed': 1126,
7         'size': 128,
8         'tau': 0.2
9     }
10    main(args)
11
12 Average total_E_in: 0.195386171875
13 Average total_E_out: 0.20937300431054243
14 mean of E_out(g,tau)-E_in(g):  0.01398683243554242

```

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

## P17

```

1 import numpy as np
2
3 def decision_stump(args, x, y):
4     #  $h(x) = s * \text{sign}(x - \theta)$ 
5     s = 1; theta = 0
6     theta_list = np.array([float('-inf')] + [((x[i]+x[i+1])/2) for i in range(0, x.shape[0]-1)])
7     error = float('inf')
8     for theta_hypothesis in theta_list:
9         y_pos = np.where(x > theta_hypothesis, 1, -1)
10        y_neg = np.where(x <= theta_hypothesis, 1, -1)
11        error_pos = sum(y_pos != y)
12        error_neg = sum(y_neg != y)
13        if error_pos > error_neg:
14            if error_neg < error:
15                error = error_neg
16                theta = theta_hypothesis
17                s = -1
18            else:
19                if error_pos < error:
20                    error = error_pos
21                    theta = theta_hypothesis
22                    s = 1
23
24        if theta == float('-inf'): theta = -0.5
25    return s, theta, float(error/x.shape[0])
26
27 def decision_stump_multi(args, x, y):
28     s = np.zeros((args['size'],))
29     theta = np.zeros((args['size'],))
30     error = np.zeros((args['size'],))
31
32     for i in range(args['size']):
33         s[i], theta[i], error[i] = decision_stump(args, x[:, i], y)
34     dimension = np.argmin(error)
35     if args['type'] == 0: dimension = np.argmax(error)
36
37     return dimension, s[dimension], theta[dimension], error[dimension]

```

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

```

1 def predict(dimension, s, theta, x_test):
2     predict = s * np.sign(x_test[:, dimension]-theta)
3     predict[predict == 0] = -1
4     return predict
5
6 def read_file(filename):
7     data = np.loadtxt(filename, dtype=float)
8     x = data[:, :-1]
9     y = data[:, -1]
10    return x, y
11
12 def main(args):
13     x_train, y_train = read_file(args['filename_train'])
14     x_test, y_test = read_file(args['filename_test'])
15
16     # change args parameter
17     args['size'] = x_train.shape[1]
18
19     dimension_best, s_best, theta_best, E_in_best = decision_stump_multi(args, x_train)
20     args['type'] = 0
21     dimension_worst, s_worst, theta_worst, E_in_worst = decision_stump_multi(args, x_train)
22
23     # predict
24     y_pred_best = predict(dimension_best, s_best, theta_best, x_test)
25     y_pred_worst = predict(dimension_worst, s_worst, theta_worst, x_test)
26
27     # E_out
28     E_out_best = np.sum(y_pred_best != y_test) / len(y_test)
29     E_out_worst = np.sum(y_pred_worst != y_test) / len(y_test)
30
31     print("E_in_best: ", E_in_best)
32     print("E_out_best: ", E_out_best)
33     print("delta E_in: ", E_in_worst - E_in_best)
34     print("delta E_out: ", E_out_worst - E_out_best)
35
36 if __name__ == '__main__':
37     args = {
38         'filename_test': "hw2_test.dat",
39         'filename_train': "hw2_train.dat",
40         'size': 1126,
41         'type': 1, # 1: best, 0: worst
42         'tau': 0
43     }
44
45     main(args)

```

```

1 E_in_best:  0.026041666666666668
2 E_out_best:  0.078125
3 delta E_in:  0.3020833333333333
4 delta E_out:  0.34375

```

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

## P18

Since the result is written in the last question at the same time, therefore, we use the same code above without updating anything.

```

1 E_in_best:  0.026041666666666668
2 E_out_best:  0.078125
3 delta E_in:  0.3020833333333333
4 delta E_out:  0.34375

```

No questions assigned to the following page.

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

## P19

Since the result is written in the last question at the same time, therefore, we use the same code above without updating anything.

```
1 | E_in_best:  0.02604166666666668
2 | E_out_best: 0.078125
3 | delta E_in:  0.3020833333333333
4 | delta E_out: 0.34375
```

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

## P20

Since the result is written in the last question at the same time, therefore, we use the same code above without updating anything.

```
1 | E_in_best:  0.02604166666666668
2 | E_out_best: 0.078125
3 | delta E_in:  0.3020833333333333
4 | delta E_out: 0.34375
```

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