# Q1

[a] It's just an optimization problem.

[b] Just probability problem.

[c] Simple math.

✓[d] Is suited for machine learning. If there is enough data of customer's making a purchase in the past, we can use learning problem like PLA to find a pattern, or "g", and predict the chance of of making a purchase for website users in the next 7 days.

For detailed explanation, we can compute a weighted score from features of user like "age", "mean of purchasing amount in the past", "job field", "gender", ... etc. And predict "do purchasing" if $\sum_{i=1}^{d} w_i x_i >$ threshold or predict "do not purchasing" if $\sum_{i=1}^{d} w_i x_i <$ threshold.

$\boxed{Q2}$

[a]

$$y_{n(t)} W_{(t+1)}^T x_{n(t)}$$

$$= y_{n(t)} \left( W_t + y_{n(t)} x_{n(t)} \cdot 2^{-t} \right)^T x_{n(t)}$$

$$= \underbrace{y_{n(t)} W_t^T x_{n(t)}}_{<0} + \underbrace{y_{n(t)}^2 \cdot 2^{-t} \cdot \|x_{n(t)}\|^2}_{>0}$$

$\Rightarrow$ Cannot make sure $y_{n(t)} W_{t+1}^T x_{n(t)}$ is greater or smaller than zero.

[b]

$$y_{n(t)} W_{t+1}^T x_{n(t)}$$

$$= y_{n(t)} \left( W_t + y_{n(t)} x_{n(t)} \cdot 0.6211 \right)^T x_{n(t)}$$

$$= \underbrace{y_{n(t)} W_t^T x_{n(t)}}_{<0} + \underbrace{0.6211\, y_{n(t)}^2 \cdot \|x_{n(t)}\|^2}_{>0}$$

$\Rightarrow$ Cannot make sure $y_{n(t)} W_{t+1}^T x_{n(t)}$ is greater or smaller than zero.

[c]

$$y_{n(t)} W_{t+1}^T x_{n(t)}$$

$$= y_{n(t)} \left( W_t + y_{n(t)} x_{n(t)} \cdot \frac{-y_{n(t)} W_t^T x_{n(t)}}{\|x_{n(t)}\|^2} \right)^T x_{n(t)}$$

$$= y_{n(t)} W_t^T x_{n(t)} + y_{n(t)}^2 \cdot \frac{-y_{n(t)} W_t^T x_{n(t)}}{\cancel{\|x_{n(t)}\|^2}} \cdot \cancel{\|x_{n(t)}\|^2}$$

$$= \underline{y_{n(t)}} W_t^T x_{n(t)} - \underline{y_{n(t)}^3} W_t^T x_{n(t)}$$

| $y_{n(t)}$ | $y_{n(t)}^3$ |
|---|---|
| +1 | +1 |
| −1 | −1 |

$y_{n(t)} = y_{n(t)}^3$

$$= 0$$

$\Rightarrow$ $y_{n(t)} W_{t+1}^T x_{n(t)}$ will be equal to 0.

[d]

$$y_{n(t)} W_{t+1}^T x_{n(t)}$$

$$= y_{n(t)} \left( W_t + y_{n(t)} x_{n(t)} \cdot \frac{1}{1+t} \right)^T x_{n(t)}$$

$$= \underbrace{y_{n(t)} W_t^T x_{n(t)}}_{<0} + \underbrace{\frac{1}{1+t} y_{n(t)}^2 \|x_{n(t)}\|^2}_{>0}$$

$\Rightarrow$ Cannot make sure $y_{n(t)} W_{t+1}^T x_{n(t)}$ is greater or smaller than zero.

[e]

$$w_{t+1} = w_t + y_{n(t)} x_{n(t)} \cdot \eta_t$$

$$[\mathbf{e}] \quad \mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_{n(t)}\mathbf{x}_{n(t)} \cdot \left\lfloor \frac{-y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2} + 1 \right\rfloor$$

$$y_{n(t)} w_{t+1}^T x_{n(t)}$$

$$= y_{n(t)} \left( w_t + y_{n(t)} x_{n(t)} \cdot \eta_t \right) x_{n(t)}$$

$$= \underbrace{y_{n(t)} w_t^T x_{n(t)}}_{\alpha < 0} + y_{n(t)}^2 \| x_{n(t)} \|^2 \cdot \eta_t$$

for option c

$$\eta_t = \frac{-y_{n(t)} w_t^T x_{n(t)}}{\| x_{n(t)} \|^2}$$

$$y_{n(t)}^2 \cancel{\| x_{n(t)} \|^2} \frac{-y_{n(t)} w_t^T x_{n(t)}}{\cancel{\| x_{n(t)} \|^2}} = -y_{n(t)}^3 w_t^T x_{n(t)}$$

$(y_{n(t)} = 1 \text{ or } -1)$

$$= -y_{n(t)} w_t^T x_{n(t)}$$

$$= -\alpha$$

$(\alpha + -\alpha = 0)$

Now $\eta_t' = \lfloor \eta_t + 1 \rfloor$, and if we can prove that $y_{n(t)}^2 \| x_{n(t)} \|^2 \eta_t' > y_{n(t)}^2 \| x_{n(t)} \|^2 \eta_t = -\alpha$, then we can make sure that $\underline{y_{n(t)} w_{t+1}^T x_{n(t)} > 0}$.

$\boxed{\text{proof}}$

case 1: $\eta_t$ is integer.

$$\eta_t = \lfloor \eta_t \rfloor < \underbrace{\lfloor \eta_t + 1 \rfloor}_{\eta_t'}$$

$$\Rightarrow \eta_t' > \eta_t$$

$$\underbrace{y_{n(t)}^2 \|X_{n(t)}\|^2}_{>0} \eta_t' > \underbrace{y_{n(t)}^2 \|X_{n(t)}\|^2}_{>0} \eta_t$$

#

case 2: $\eta_t$ is float.

$$\lfloor \eta_t \rfloor < \eta_t < \underbrace{\lfloor \eta_t + 1 \rfloor}_{\eta_t'}$$

$$\Rightarrow \eta_t' > \eta_t$$

$$y_{n(t)}^2 \|X_{n(t)}\|^2 \eta_t' > y_{n(t)}^2 \|X_{n(t)}\|^2 \eta_t$$

#

# Q3

[a] $w_{t+1} \leftarrow w_t + y_{n(t)} x_{n(t)} \cdot 2^{-t}$ ( Revised )

①

$$w_f^T \cdot w_{t+1} = w_f^T ( w_t + y_{n(t)} x_{n(t)} \cdot 2^{-t} )$$

$$= w_f^T w_t + y_{n(t)} w_f^T x_{n(t)} \cdot 2^{-t}$$

$$\geq w_f^T w_t + \underbrace{\min_n y_n w_f^T x_n}_{>0} \cdot \underbrace{2^{-t}}_{>0}$$

$$> w_f^T w_t + 0$$

②

$$\| w_{t+1} \|^2 = \| w_t + y_{n(t)} x_{n(t)} \cdot 2^{-t} \|$$

$$= \| w_t \|^2 + \underbrace{2 \cdot 2^{-t} \cdot y_{n(t)} w_t^T x_{n(t)}}_{\leq 0} + \| 2^{-t} \cdot y_{n(t)} x_{n(t)} \|^2$$

$$\leq \| w_t \|^2 + 0 + \| 2^{-t} \cdot y_{n(t)} x_{n(t)} \|^2$$

$$\leq \| w_t \|^2 + 0 + \max_n \| 2^{-t} \cdot y_n x_n \|^2$$

$\boxed{\text{Combine}}$

$$\min_n y_n w_f^T x_n \cdot 2^{-t} = \rho \cdot 2^{-t}$$

$$\cancel{w_f^T w_1} \geq w_f^T w_0 + \rho \cdot 2^{-0}$$

$$\cancel{w_f^T w_2} \geq \cancel{w_f^T w_1} + \rho \cdot 2^{-1}$$

$$\cancel{w_f^T w_3} \geq \cancel{w_f^T w_2} + \rho \cdot 2^{-2}$$

$$\vdots$$

$$+ \; w_f^T w_T \geq \cancel{w_f^T w_{T-1}} + \rho \cdot 2^{-(T-1)}$$

$$w_f^T w_T \geq w_f^T w_0 + \rho \times \frac{1 \times (1 - 2^{-T})}{1 - 2^{-1}}$$

$$= w_f^T w_0 + 2\rho \times (1 - 2^{-T})$$

$$\max_n \| 2^{-t} \cdot y_n x_n \|^2 = R^2 \cdot 2^{-t}$$

$$\cancel{\| w_1 \|^2} \leq \| w_0 \|^2 + R^2 \cdot 2^{-0}$$

$$\cancel{\| w_2 \|^2} \leq \cancel{\| w_1 \|^2} + R^2 \cdot 2^{-1}$$

$$\cancel{\| w_3 \|^2} \leq \cancel{\| w_2 \|^2} + R^2 \cdot 2^{-2}$$

$$\vdots$$

$$+ \; \| w_T \|^2 \leq \cancel{\| w_{T-1} \|^2} + R^2 \cdot 2^{-(T-1)}$$

$$\| w_T \|^2 \leq \| w_0 \|^2 + R^2 \times \frac{1 \times (1 - 2^{-T})}{1 - 2^{-1}}$$

$$= \| w_0 \|^2 + 2R^2 (1 - 2^{-T})$$

$$\downarrow$$

$$\boxed{w_0 = \vec{0}, \quad \|w_f\| = 1}$$

$$w_f^T w_T \geq 2\rho(1-2^{-T}), \quad \|w_T\|^2 \leq 2R^2(1-2^{-T})$$

$$1 \geq \frac{w_f^T}{\|w_f\|}\frac{w_T}{\|w_T\|} \geq \frac{2\rho(1-2^{-T})}{\|w_T\|} \geq \frac{2\rho(1-2^{-T})}{\sqrt{2}R\sqrt{1-2^{-T}}}$$

$$= \frac{\rho}{R}\sqrt{2} \times (1-2^{-T})$$

$$\lim_{T\to\infty} \frac{\rho}{R}\sqrt{2}(1-2^{-T})$$

$= \frac{\sqrt{2}\rho}{R}$ (變成會因 dataset 而異，如果 $\frac{\rho}{R} < \frac{1}{\sqrt{2}}$，則 PLA 不一定會 halt)

[b]

$$W_{t+1} \leftarrow W_t + y_{n(t)} X_{n(t)} \cdot 0.6211$$

①

$$W_f^T W_{t+1} = W_f^T ( W_t + y_{n(t)} X_{n(t)} \cdot 0.6211 )$$

$$\geq W_f^T W_t + \underbrace{\min_n y_n W_f^T X_n \cdot 0.6211}_{>0}$$

t 項去掉

$$> W_f^T W_t + 0$$

②

$$\| W_{t+1} \|^2 = \| W_t + y_{n(t)} X_{n(t)} \cdot 0.6211 \|^2$$

$$= \|W_t\|^2 + 2 \times 0.6211 \, y_{n(t)} W_t^T X_{n(t)}$$

$$+ \| y_{n(t)} X_{n(t)} \cdot 0.6211 \|^2$$

$$\leq \|W_t\|^2 + 0 + \| y_{n(t)} X_{n(t)} \cdot 0.6211 \|^2 \leq \underline{\|W_t\|^2 + \max_n \| y_n X_n \cdot 0.6211 \|^2}$$

t 項去掉

Combine

$$\underbrace{\min_n y_n W_f^T X_n \cdot 0.6211}_{\rho}$$

$$\cancel{W_f^T W_1} \geq W_f^T W_0 + \rho$$

$$W_f^T \cancel{W_2} \geq \cancel{W_f^T W_1} + \rho$$

$$\vdots$$

$$+ \; W_f^T W_T \geq W_f^T \cancel{W_{T-1}} + \rho$$

$$W_f^T W_T \geq W_f^T W_0 + T\rho$$

$$\underbrace{\max_n \| y_n X_n \cdot 0.6211 \|^2}_{R^2}$$

$$\cancel{\|W_1\|^2} \leq \|W_0\|^2 + R^2$$

$$\cancel{\|W_2\|^2} \leq \cancel{\|W_1\|^2} + R^2$$

$$\vdots$$

$$+ \; \|W_T\|^2 \leq \cancel{\|W_{T-1}\|^2} + R^2$$

$$\|W_T\|^2 \leq \|W_0\|^2 + TR^2$$

Let $W_0 = \vec{0}$, $\|W_f\| = 1$

$$W_f W_T \geq T\rho$$

$$\|W_T\|^2 \leq TR^2$$

$$\|W_T\| \leq \sqrt{T}R$$

$$1 \geq \frac{W_f}{\|W_f\|} \frac{W_T}{\|W_T\|} \geq \frac{T\rho}{\|W_f\| \, \|W_T\|}$$

$$1 \geq \frac{W_f}{\|W_f\|} \frac{W_T}{\|W_T\|} \geq \frac{T\rho}{1 \times \sqrt{T}R}$$

$$\frac{T\rho}{\sqrt{T}R} \leq 1 \rightarrow \sqrt{T} \leq \frac{R}{\rho} \rightarrow T \leq \left(\frac{R}{\rho}\right)^2$$

Ch.al(t) #

[C] $w_{t+1} \leftarrow w_t + y_{n(t)} x_{n(t)} \cdot \left( \frac{-y_{n(t)} w_t^T x_{n(t)}}{\| x_{n(t)} \|^2} \right)$

Based on $Q_2$'s derivation, this option will never halt with a perfect line

That is because after every updating, the new line ($w_{t+1}$) will always overlap with the previous error point ($x_{n(t)}$) that used to update $w_t$.

Hence, there will never be a perfect line that can discriminate all $x_n$.

[d] $w_{t+1} \longleftarrow w_t + y_{n(t)} x_{n(t)} \cdot \left( \frac{1}{1+t} \right)$

① 

$$w_f^T w_{t+1} = w_f^T \left( w_t + y_{n(t)} x_{n(t)} \cdot \frac{1}{1+t} \right)$$

$$= w_f^T w_t + \underbrace{\frac{1}{1+t} y_{n(t)} w_f^T x_{n(t)}}_{>0}$$

( $\rho(t) \to$ function of $t$ )

$$\geq w_f^T w_t + \frac{1}{1+t} \min_n y_n w_f^T x_n \to \frac{1}{1+t} \rho$$

② 

$$\| w_{t+1} \|^2 = \left\| w_t + y_{n(t)} x_{n(t)} \cdot \frac{1}{1+t} \right\|^2$$

$$= \| w_t \|^2 + \underbrace{\frac{2}{1+t} y_{n(t)} w_t^T x_{n(t)}}_{\leq 0} + \left\| \frac{1}{1+t} y_{n(t)} x_{n(t)} \right\|^2$$

$$\leq \| w_t \|^2 + 0 + \left\| \frac{1}{1+t} y_{n(t)} x_{n(t)} \right\|^2$$

$$\leq \| w_t \|^2 + \frac{1}{1+t} \max_n \| y_n x_n \|^2 \to \frac{1}{1+t} R^2$$

Combine

$$\cancel{w_f^T w_1} \geq w_f^T w_0 + \frac{1}{1} \rho$$

$$\cancel{w_f^T w_2} \geq \cancel{w_f^T w_1} + \frac{1}{2} \rho$$

$$\cancel{w_f^T w_3} \geq \cancel{w_f^T w_2} + \frac{1}{3} \rho$$

$$\vdots$$

$$+ \; w_f^T w_T \geq \cancel{w_f^T w_{T-1}} + \frac{1}{T} \rho$$

$$w_f^T w_T \geq w_f^T w_0 + \rho \left( \frac{1}{1} + \frac{1}{2} + \dots + \frac{1}{T} \right)$$

$$\cancel{\| w_1 \|^2} \leq \| w_0 \|^2 + \frac{1}{1} R^2$$

$$\cancel{\| w_2 \|^2} \leq \cancel{\| w_1 \|^2} + \frac{1}{2} R^2$$

$$\cancel{\| w_3 \|^2} \leq \cancel{\| w_2 \|^2} + \frac{1}{3} R^2$$

$$\vdots$$

$$+ \; \| w_T \|^2 \leq \cancel{\| w_{T-1} \|^2} + \frac{1}{T} R^2$$

$$\| w_T \|^2 \leq \| w_0 \|^2 + R^2 \left( \frac{1}{1} + \frac{1}{2} + \dots + \frac{1}{T} \right)$$

$w_0 = \vec{0}$ , $\|w_f\| = 1$

$$1 \ge \frac{w_f^T}{\|w_f\|}\frac{w_T}{\|w_T\|} \ge \frac{\rho\left(\frac{1}{1}+\frac{1}{2}+\dots+\frac{1}{T}\right)}{\|w_T\|}$$

$$= \frac{\rho\left(\frac{1}{1}+\frac{1}{2}+\dots+\frac{1}{T}\right)}{R\sqrt{\frac{1}{1}+\frac{1}{2}+\dots+\frac{1}{T}}}$$

$$= \frac{\rho}{R}\sqrt{\frac{1}{1}+\frac{1}{2}+\dots+\frac{1}{T}}$$

$$\lim_{T\to\infty}\frac{\rho}{R}\sqrt{\left(\frac{1}{1}+\frac{1}{2}+\dots+\frac{1}{T}\right)} = \infty$$

∴ lower bound will be greater that 1 as $T$ increase

∴ PLA will halt with a perfect line.

[e] $w_{t+1} \leftarrow \underbrace{w_t + y_{n(t)} x_{n(t)} \cdot \left\lfloor \frac{-y_{n(t)} w_t^T x_{n(t)}}{\|x_{n(t)}\|^2} + 1 \right\rfloor}_{\eta_t}$

$\boxed{\text{Start from } w_f^T w_{t+1}}$

$$w_f^T w_{t+1} = w_f^T \left( w_t + \left\lfloor - \frac{y_{n(t)} w_t^T x_{n(t)}}{\|x_{n(t)}\|^2} + 1 \right\rfloor y_{n(t)} x_{n(t)} \right)$$

$$= w_f^T w_t + \left\lfloor 1 - \frac{y_{n(t)} w_t^T x_{n(t)}}{\|x_{n(t)}\|^2} \right\rfloor y_{n(t)} w_f^T x_{n(t)}$$

$$\geq w_f^T w_t + y_{n(t)} w_f^T x_{n(t)}$$

$$\geq w_f^T w_t + \min_n y_n w_f^T x_n$$

$$\cancel{w_f^T w_1} \geq w_f^T w_0 + \min_n y_n w_f^T x_n$$

$$w_f^T \cancel{w_2} \geq \cancel{w_f^T w_1} + \min_n y_n w_f^T x_n$$

$$\vdots$$

$$+ \; w_f^T w_T \geq \cancel{w_f^T w_{T-1}} + \min_n y_n w_f^T x_n$$

$$w_f^T w_T \geq \underline{T \times \min_n y_n w_f^T x_n}$$

Bound (half)

# Q4

Give an example for $W_f$ that match the case in problem : $f(X) = \text{sign}\left(z_+(X) - z_-(X) - 0.5\right)$

$$\Rightarrow W_f = (-0.5, \underbrace{-1, -1,}_{d_-} \underbrace{+1, +1, +1}_{d_+})$$

And an example for $X$ which $\|X\|^2$ is largest

$$\Rightarrow X_n = (1, 0, 1, 1, 1, 0)$$

Now according to PLA process referred in the problem, we know $T \le \left(\frac{R}{\rho}\right)^2$.

$$\rho = \min_n y_n \frac{W_f^T}{\|W_f\|} X_n$$

$$R^2 = \max_n \|X_n\|^2$$

For $\min_n y_n w_f^T x_n$ in $\rho$, we can analyze 2 cases.

① $z_+(x_n) - z_-(x_n) = 0$

$w_f^T x_n = -0.5$

$y_n = -1$

$y_n w_f^T x_n = 0.5$

② $z_+(x_n) - z_-(x_n) = 1$

$w_f^T x_n = +0.5$

$y_n = +1$

$y_n w_f^T x_n = 0.5$

$\because$ 在 case ① 與 ② 的 $x_n$ 是離 $w_f$ 最近的2個點，且 $y_n w_f^T x_n$ 皆等於 0.5.

$\therefore \min_n y_n w_f^T x_n = 0.5$

For example of $w_f$ above, $\|w_f\| = \sqrt{\frac{1}{4}+5} = \sqrt{\frac{1}{4}+d}$

Now we can calculate upper bound for $T$.

$$\Rightarrow \rho = \min_n y_n \frac{w_f^T}{\|w_f\|} x_n$$

$$= \frac{1}{\|w_f\|} \min_n y_n w_f^T x_n$$

$$= \frac{1}{2\sqrt{\frac{1}{4}+d}}$$

$$\Rightarrow R^2 = \max_n \|x_n\|^2$$

$$= 3+1$$

$$= m+1$$

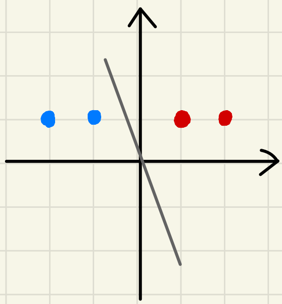
$$\Rightarrow T \le \frac{R^2}{\rho^2} = \frac{m+1}{\frac{1}{4(\frac{1}{4}+d)}}$$

$$= (4d+1)(m+1)$$

#

# Q5

## Illustrate with examples



$X \rightarrow R^2$

$x_1 = (1,1) \quad y_n = 2, \ \tilde{y}_n = 1$

$x_2 = (2,1) \quad y_n = 2 \quad \tilde{y}_n = 1$

$x_3 = (-1,1) \quad y_n = 1 \quad \tilde{y}_n = -1$

$x_4 = (-2,1) \quad y_n = 1 \quad \tilde{y}_n = -1$

$W_{PLA(0)} = (0,0,0), \quad W_{1(0)} = (0,0,0), \quad W_{2(0)} = (0,0,0)$

Insert $x_0 = 0$ to each $X$

1st: for $x_n = x_1$ $(y_n = 2, \tilde{y}_n = 1)$

$W_{PLA(1)} = (0,0,0) + 1 \times (0,1,1) = \underline{(0,1,1)}$

$W_2^{(1)} = (0,0,0) + (0,1,1) = \underline{(0,1,1)}$

$W_1^{(1)} = (0,0,0) - (0,1,1) = \underline{(0,-1,-1)}$

2nd: for $x_n = x_3$ $(y_n = 1, \tilde{y}_n = -1)$

$W_{PLA}^{(2)} = (0,1,1) - (0,-1,1) = \underline{(0,2,0)}$

$W_1^{(2)} = (0,-1,-1) + (0,-1,1) = \underline{(0,-2,0)}$

$W_2^{(2)} = (0,1,1) - (0,-1,1) = \underline{(0,2,0)}$

⋮

We can find in $1^{st}$ updating : $W_{PLA} = -W_1 = W_2$

and in $2^{nd}$ updating : $W_{PLA} = -W_1 = W_2$

it's a pattern , for $n^{th}$ updating,

$$\underline{W_{PLA} = -W_1 = W_2}$$

#

Q6

The feature of self-supervised learning is that it derives label from a co-occuring input to related information, which matches the condition in the problem: "Images that are taken at similar time stamps should be mapped to similar vectors ...".

And the actual task for learned model of self-supervised learning is performed with supervised or unsupervised learning. This feature also matches the condition in the problem: "... combined with a variety of tasks, such as being mixed with some labeled data for object recognition ...".

Based on the two evidence above, self-supervised learning matches the process in the problem.

# Q7

① Evidence for multilabel classification.

c1) Experts tag each article by its categories.

c2) Each artible can belong to several different categories.

c3) The leared "g" can tag future news articles.

② Evidence for semi-supervised learning.

tag each article by its categories. Each article can belong to several different categories. We also gather another 11265566 news articles from our database, without expert labeling. The learning algorithm is expected to learn from all the articles and tags (if any) to obtain a hypothesis that tags future news articles well. What learning problem best matches the description above? Choose

③ Evidence for batch learning

tag each article by its categories. Each article can belong to several different categories. We also gather another 11265566 news articles from our database, without expert labeling. The learning algorithm is expected to learn from all the articles and tags (if any) to obtain a hypothesis that tags future news articles well. What learning problem best matches the description above? Choose

④ Evidence for raw features

Tag of each article → raw features.

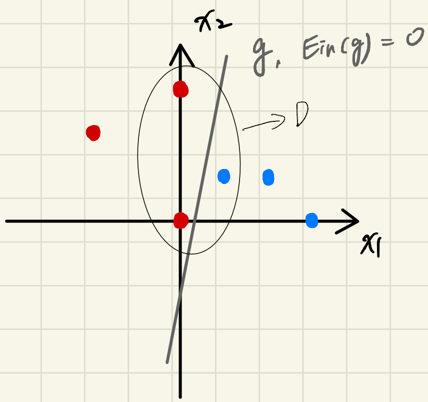
# Q8

$\mathcal{U} \rightarrow 6$ examples, $\quad D \rightarrow 3$ examples from $\mathcal{U}$

We can illustrate some cases to find the smallest and the largest $E_{ots}(g)$.

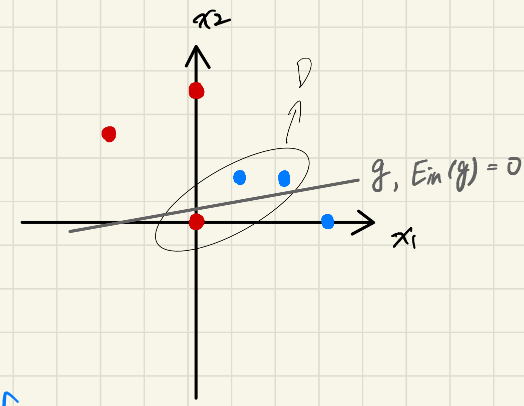
**case 1**

$E_{ots}(g) = 0$




**case 2**

$E_{ots}(g) = 1$




As the possible range of $E_{ots}(g)$ itself is $0 \leq E_{ots}(g) \leq 1$, we have gotten the result of this problem from the 2 cases above.

# Q9

$\hat{\theta}$ is unbiased iff. $E(\hat{\theta}) = \theta$

[a]

$$E(\hat{\theta}) = E\left(\frac{1}{N}\sum_{n=1}^{N}(h(x_n) \ne y_n)\right)$$

$$= \frac{1}{N}\sum_{n=1}^{N} \underline{E[h(x_n) \ne y_n]}$$

$$= \frac{1}{N}\sum_{n=1}^{N} \underline{\theta}$$

$$= \frac{N\theta}{N} = \theta \quad \# \quad \text{(unbiased)}$$

[b]

$$E(\hat{\theta}) = E\left(\frac{1}{N}\sum_{n=1}^{N} x_n\right)$$

$$= \frac{1}{N}\sum_{n=1}^{N} E(x_n)$$

$$= \frac{1}{N}\sum_{n=1}^{N} \theta$$

$$= \frac{N\theta}{N} = \theta \quad \# \quad \text{(unbiased)}$$

$E(x_n)$

$= P(0) \times 0 + P(1) \times 1$

$= (1-\theta) \times 0 + \theta \times 1$

$= \theta$

[C] $\hat{\theta} = \max\{x_1, x_2, \dots, x_N\}$ , $M > N > 0$

p.m.f. $\longrightarrow P(x) = \dfrac{1}{M}$ $\quad \theta = M$

$$\left.\begin{aligned} G_n(y) &= Pr(Y_n \le y) \\ &= Pr(X_1 \le y)\, Pr(X_2 \le y) \cdots Pr(X_n \le y) \\ &= \left(\frac{y}{M}\right)^n \end{aligned}\right\} \text{order statistic}$$

$$E(\hat{\theta}) = E(\max\{x_1, x_2, \dots, x_N\})$$

$$= \sum_{k=1}^{M} k \underbrace{Pr(X_n = k)}_{} \longrightarrow = Pr(X_n \le k) - Pr(X_n \le k-1)$$

$$= \sum_{k=1}^{M} k \times \left[\left(\frac{k}{M}\right)^n - \left(\frac{k-1}{M}\right)^n\right]$$

$$= 1 \times \left(\frac{1}{M}\right)^n + 2 \times \left[\left(\frac{2}{M}\right)^n - \left(\frac{1}{M}\right)^n\right] + 3 \times \left[\left(\frac{3}{M}\right)^n - \left(\frac{2}{M}\right)^n\right]$$

$$+ \cdots + M \times \left[\left(\frac{M}{M}\right)^n - \left(\frac{M-1}{M}\right)^n\right]$$

$$= -1 \times \left(\frac{1}{M}\right)^n - 1 \times \left(\frac{2}{M}\right)^n - 1 \times \left(\frac{3}{M}\right)^n - \cdots - 1 \times \left(\frac{M-1}{M}\right)^n$$

$$+ M \times \left(\frac{M}{M}\right)^n$$

$$= \underbrace{-\left(\frac{1^n}{M^n} + \frac{2^n}{M^n} + \frac{3^n}{M^n} + \cdots + \frac{(M-1)^n}{M^n}\right)}_{\neq 0} + M$$

$$\neq M$$

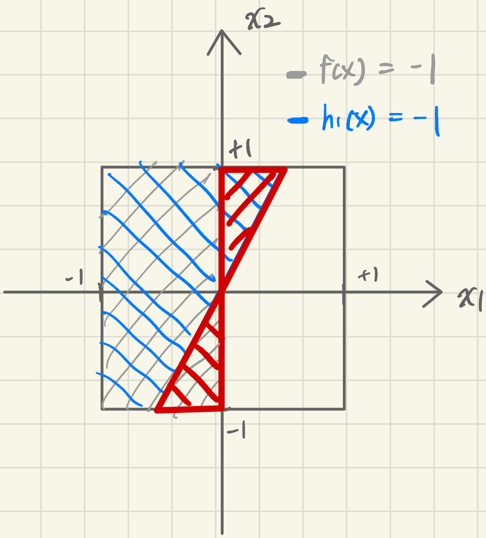
# Q10

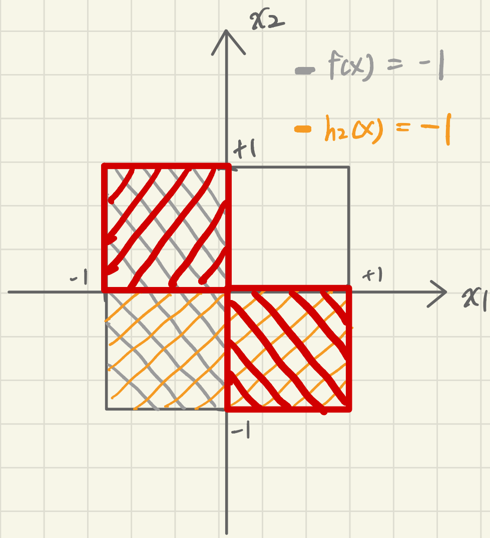
$f(X) = \text{sign}(x_1)$

$h_1(X) = \text{sign}(2x_1 - x_2)$

$h_2(X) = \text{sign}(x_2)$

Consider the $X = [x_1, x_2]^T$ distribution in coordinate plane.




$E_{out}(h_1)$

$= \underset{x \sim P}{\mathcal{E}} [h_1(x) \neq f(x)]$

$= \dfrac{\text{red area}}{\text{whole area}}$

$= \dfrac{1}{8}$ #

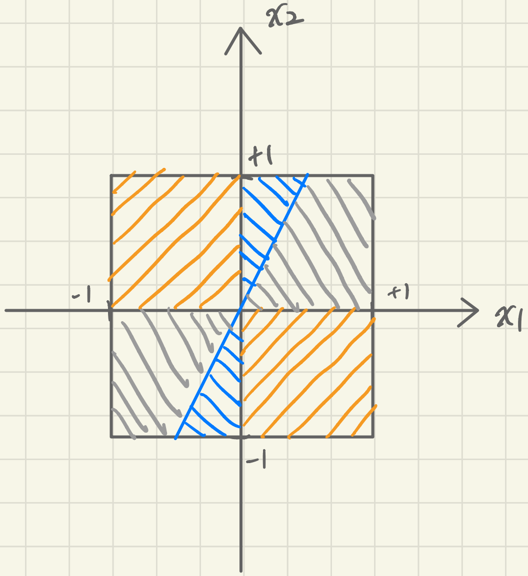
$E_{out}(h_2)$

$= \underset{x \sim P}{\mathcal{E}} [h_2(x) \neq f(x)]$

$= \dfrac{\text{red area}}{\text{whole area}}$

$= \dfrac{1}{2}$ #

# Q11




get 4 examples such that

$E_{in}(h_2) = E_{in}(h_1)$

$E_{out}(h_1) = \frac{1}{8}$, $E_{out}(h_2) = \frac{1}{2}$

● ① $h_1(x) = f(x)$, $h_2(x) = f(x)$, $p = \frac{3}{8}$

● ② $h_1(x) \neq f(x)$, $h_2(x) = f(x)$, $p = \frac{1}{8}$

● ③ $h_1(x) = f(x)$, $h_2(x) \neq f(x)$, $p = \frac{1}{2}$

$\boxed{\text{case 1: } E_{in}(h_1) = E(h_2) = 0}$

$$P_1 = \left(\frac{3}{8}\right)^4 = \frac{81}{4096}$$

$\boxed{\text{case 2: } E_{in}(h_1) = E_{in}(h_2) = \frac{1}{4}}$

$$P_2 = \left(\frac{3}{8}\right)^2 \left(\frac{1}{8}\right)\left(\frac{4}{8}\right) \times \frac{4!}{2!}$$

$$= \frac{432}{4096}$$

$\boxed{\text{case 3: } E_{in}(h_1) = E_{in}(h_2) = \frac{2}{4}}$

$$P_3 = \left(\frac{1}{8}\right)^2 \left(\frac{4}{8}\right)^2 \times \frac{4!}{2!\,2!}$$

$$= \frac{96}{4096}$$

$\Rightarrow P_1 + P_2 + P_3$

$$= \frac{81 + 432 + 96}{4096} = \frac{609}{4096}$$

#

# Q12

| | Green | Orange |
|---|---|---|
| A | 2, 4, 6 | 1, 3, 5 |
| B | 1, 2, 6 | 3, 4, 5 |
| C | 6 | 1, 2, 3, 4, 5 |
| D | 2, 3, 5 | 1, 4, 6 |

Dice for the number that is purely green.

1 : B

2 : A, B, D

3 : D

4 : A

5 : D

6 : ABC

$$\boxed{\begin{array}{l} B + ABD + D + A + ABC \\ - (\text{only } A + \text{only } B + \text{only } D + \text{only } AB) \end{array}}$$

$$\left(\frac{1}{4}\right)^5 + \left(\frac{3}{4}\right)^5 + \left(\frac{1}{4}\right)^5 + \left(\frac{1}{4}\right)^5 + \left(\frac{3}{4}\right)^5$$

$$- 3 \times \left(\frac{1}{4}\right)^5 - \left(\frac{2}{4}\right)^5$$

$$= \underline{1 + 243 + 1 + 1 + 243 - 3 - 32}$$

$$= \frac{489 - 35}{1024} = \frac{454}{1024}$$

#

Q13

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Sat Oct 16 05:37:08 2021

@author: YI SIANG SYU

Description:
    Q13.
    There are four main parts in this .py file. First is to import necessary packages.
    Second is to define function I used in PLA algorithm. Third is to load data and do
    some preprocessing like adding x0 to every xn. Fourth is the key part for doing
    PLA loop (each experiment is done with different random seed).
    For this question, there is no scaling factor.
"""


# %% First: import necessary packages
from IPython import get_ipython
get_ipython().magic('clear')
get_ipython().magic('reset -f')
import numpy as np
from datetime import datetime


# %% Second: define function
def getH(w, x):
    # inner product
    innerProduct = np.matmul(w, x.T)
    # get sign
    if innerProduct > 0:
        h = 1
    elif innerProduct < 0:
        h = -1
    else:  # %% case of innerProduct = 0
        h = -1
    return h


# %% Third: preprocess
# load data
dataSet = np.genfromtxt("hw1_train.dat")

# insert x0, extract x vector set and y set
x0 = 1
dataSet = np.insert(dataSet, 0, x0, axis=1)
xSet = dataSet[:, :-1]
ySet = dataSet[:, -1]

# set other related parameters
cumuCorrectThold = 5*dataSet.shape[0]
repeatTestTimes = 1000
wSet = np.empty((repeatTestTimes, xSet.shape[1]))  # container for storing wPLA


# %% Fourth: main - do experiment
# get current time, used for seed setting.
currentTime = int(datetime.now().strftime("%H:%M:%S").replace(":", ""))
print("Start to do PLA!")
print("\nMonitor progress:")
for testIdx in range(repeatTestTimes):  # Repeat experiment for 1000 times
    # initialization
    t = 0
    cumuCorrect = 0
    w = np.zeros(xSet.shape[1])  # initialize w
    # ensure each experiment is done with different seed.
    np.random.seed(int(currentTime*1e4)+testIdx)
    # do PLA
    while(True):
        # randomly picks an example in every iteration (with replacement)
        idx = np.random.randint(0, xSet.shape[0])
        if getH(w, xSet[idx]) != ySet[idx]:
            cumuCorrect = 0
            # updates w if and only if w is incorrect on the example.
            w = w + ySet[idx]*xSet[idx]
        else:
            cumuCorrect += 1
            if cumuCorrect >= cumuCorrectThold:
                # stop updating and return w as wPLA if w is correct consecutively
                # after checking 5N randomly-picked examples.
                break
        t += 1
    wSet[testIdx] = w
    print("{}, ".format(testIdx), end="")
print("\n\nDone!")
# show the average squared length of wPLA
print("Average squared length of wPLA:", (wSet**2).sum(axis=1).mean())
```

90 lines (78 sloc) | 2.69 KB    Raw  Blame

Q14

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Sat Oct 16 07:19:04 2021

@author: YI SIANG SYU

Description:
    Q14.
    There are four main parts in this .py file. First is to import necessary packages.
    Second is to define function I used in PLA algorithm. Third is to load data and do
    some preprocessing like adding x0 to every xn. Fourth is the key part for doing
    PLA loop (each experiment is done with different random seed).
    For this question, there is a scaling factor which is implemented in
    third part - preprocess.
"""


# %% First: import necessary packages
from IPython import get_ipython
get_ipython().magic('clear')
get_ipython().magic('reset -f')
import numpy as np
from datetime import datetime


# %% Second: define function
def getH(w, x):
    # inner product
    innerProduct = np.matmul(w, x.T)
    # get sign
    if innerProduct > 0:
        h = 1
    elif innerProduct < 0:
        h = -1
    else:  # %% case of innerProduct = 0
        h = -1
    return h


# %% Third: preprocess
# load data
dataSet = np.genfromtxt("hw1_train.dat")

# insert x0, extract x vector set and y set
x0 = 1
dataSet = np.insert(dataSet, 0, x0, axis=1)
xSet = dataSet[:, :-1]
# scaling by factor of 2 for question 14
xSet = xSet*2
ySet = dataSet[:, -1]

# set other related parameters
cumuCorrectThold = 5*dataSet.shape[0]
repeatTestTimes = 1000
wSet = np.empty((repeatTestTimes, xSet.shape[1]))  # container for storing wPLA


# %% Fourth: main - do experiment
# get current time, used for seed setting.
currentTime = int(datetime.now().strftime("%H:%M:%S").replace(":", ""))
print("Start to do PLA!")
print("\nMonitor progress:")
for testIdx in range(repeatTestTimes):  # Repeat experiment for 1000 times
    # initialization
    t = 0
    cumuCorrect = 0
    w = np.zeros(xSet.shape[1])  # initialize w
    # ensure each experiment is done with different seed.
    np.random.seed(int(currentTime*1e4)+testIdx)
    # do PLA
    while(True):
        # randomly picks an example in every iteration (with replacement)
        idx = np.random.randint(0, xSet.shape[0])
        if getH(w, xSet[idx]) != ySet[idx]:
            cumuCorrect = 0
            # updates w if and only if w is incorrect on the example.
            w = w + ySet[idx]*xSet[idx]
        else:
            cumuCorrect += 1
            if cumuCorrect >= cumuCorrectThold:
                # stop updating and return w as wPLA if w is correct consecutively
                # after checking 5N randomly-picked examples.
                break
        t += 1
    wSet[testIdx] = w
    print("{}, ".format(testIdx), end="")
print("\n\nDone!")
# show the average squared length of wPLA
print("Average squared length of wPLA:", (wSet**2).sum(axis=1).mean())
```

90 lines (78 sloc) | 2.75 KB

Raw Blame

Q15

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Sat Oct 16 07:24:23 2021

@author: YI SIANG SYU

Description:
    Q15.
    There are four main parts in this .py file. First is to import necessary packages.
    Second is to define function I used in PLA algorithm. Third is to load data and do
    some preprocessing like adding x0 to every xn. Fourth is the key part for doing
    PLA loop (each experiment is done with different random seed).
    For this question, there a scaling factor which is implemented in
    third part - preprocess.
"""


# %% First: import necessary packages
from IPython import get_ipython
get_ipython().magic('clear')
get_ipython().magic('reset -f')
import numpy as np
from datetime import datetime


# %% Second: define function
def getH(w, x):
    # inner product
    innerProduct = np.matmul(w, x.T)
    # get sign
    if innerProduct > 0:
        h = 1
    elif innerProduct < 0:
        h = -1
    else:  # %% case of innerProduct = 0
        h = -1
    return h


# %% Third: preprocess
# load data
dataSet = np.genfromtxt("hw1_train.dat")

# insert x0, extract x vector set and y set
x0 = 1
dataSet = np.insert(dataSet, 0, x0, axis=1)
xSet = dataSet[:, :-1]
# scaling down each xn to length 1 for question 15
xSet = xSet / np.sqrt((xSet**2).sum(axis=1)).reshape(-1, 1)
ySet = dataSet[:, -1]

# set other related parameters
cumuCorrectThold = 5*dataSet.shape[0]
repeatTestTimes = 1000
wSet = np.empty((repeatTestTimes, xSet.shape[1]))  # container for storing wPLA


# %% Fourth: main - do experiment
# get current time, used for seed setting.
currentTime = int(datetime.now().strftime("%H:%M:%S").replace(":", ""))
print("Start to do PLA!")
print("\nMonitor progress:")
for testIdx in range(repeatTestTimes):  # Repeat experiment for 1000 times
    # initialization
    t = 0
    cumuCorrect = 0
    w = np.zeros(xSet.shape[1])  # initialize w
    # ensure each experiment is done with different seed.
    np.random.seed(int(currentTime*1e4)+testIdx)
    # do PLA
    while(True):
        # randomly picks an example in every iteration (with replacement)
        idx = np.random.randint(0, xSet.shape[0])
        if getH(w, xSet[idx]) != ySet[idx]:
            cumuCorrect = 0
            # updates w if and only if w is incorrect on the example.
            w = w + ySet[idx]*xSet[idx]
        else:
            cumuCorrect += 1
            if cumuCorrect >= cumuCorrectThold:
                # stop updating and return w as wPLA if w is correct consecutively
                # after checking 5N randomly-picked examples.
                break
        t += 1
    wSet[testIdx] = w
    print("{}, ".format(testIdx), end="")
print("\n\nDone!")
# show the average squared length of wPLA
print("Average squared length of wPLA:", (wSet**2).sum(axis=1).mean())
```

87 lines (75 sloc) | 2.66 KB

Raw Blame

Q16

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Sat Oct 16 07:36:11 2021

@author: YI SIANG SYU

Description:
    Q16.
    There are four main parts in this .py file. First is to import necessary packages.
    Second is to define function I used in PLA algorithm. Third is to load data and do
    some preprocessing like adding x0 to every xn. Fourth is the key part for doing
    PLA loop (each experiment is done with different random seed).
    For this question, the x0 added to every xn is set to 0.
"""


# %% First: import necessary packages
from IPython import get_ipython
get_ipython().magic('clear')
get_ipython().magic('reset -f')
import numpy as np
from datetime import datetime


# %% Second: define function
def getH(w, x):
    # inner product
    innerProduct = np.matmul(w, x.T)
    # get sign
    if innerProduct > 0:
        h = 1
    elif innerProduct < 0:
        h = -1
    else:  # %% case of innerProduct = 0
        h = -1
    return h


# %% Third: preprocess
# load data
dataSet = np.genfromtxt("hw1_train.dat")

# insert x0, extract x vector set and y set
x0 = 0  # set x0 = 0 which will be added to each xn for question 16
dataSet = np.insert(dataSet, 0, x0, axis=1)
xSet = dataSet[:, :-1]
ySet = dataSet[:, -1]

# set other related parameters
cumuCorrectThold = 5*dataSet.shape[0]
repeatTestTimes = 1000
wSet = np.empty((repeatTestTimes, xSet.shape[1]))  # container for storing wPLA


# %% Fourth: main - do experiment
# get current time, used for seed setting.
currentTime = int(datetime.now().strftime("%H:%M:%S").replace(":", ""))
print("Start to do PLA!")
print("\nMonitor progress:")
for testIdx in range(repeatTestTimes):  # Repeat experiment for 1000 times
    # initialization
    t = 0
    cumuCorrect = 0
    w = np.zeros(xSet.shape[1])  # initialize w
    # ensure each experiment is done with different seed.
    np.random.seed(int(currentTime*1e4)+testIdx)
    # do PLA
    while(True):
        # randomly picks an example in every iteration (with replacement)
        idx = np.random.randint(0, xSet.shape[0])
        if getH(w, xSet[idx]) != ySet[idx]:
            cumuCorrect = 0
            # updates w if and only if w is incorrect on the example.
            w = w + ySet[idx]*xSet[idx]
        else:
            cumuCorrect += 1
            if cumuCorrect >= cumuCorrectThold:
                # stop updating and return w as wPLA if w is correct consecutively
                # after checking 5N randomly-picked examples.
                break
        t += 1
    wSet[testIdx] = w
    print("{}, ".format(testIdx), end="")
print("\n\nDone!")
# show the average squared length of wPLA
print("Average squared length of wPLA:", (wSet**2).sum(axis=1).mean())
```