The Learning Problem

1. ANS [d]

Which of the following problem is suited for machine learning if there is assumed to be enough associated data? Choose the correct answer; explain how you can possibly use machine learning to solve it.

[a] identifying the shortest distance path from Taipei to Kao-Hsiung

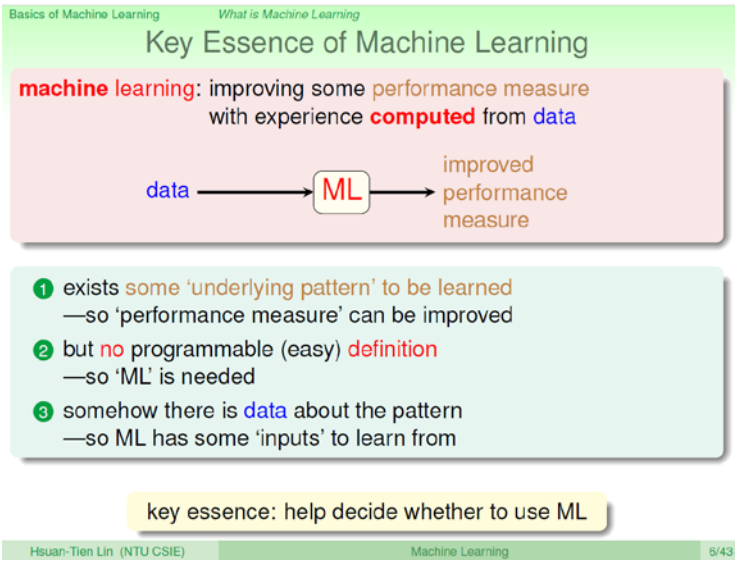
[b] predicting the winning number of the super lotto

[c] calculating the total salary of all research assistants in a lab

[d] sorting your e-commerce website users by their predicted chances of making a purchase in the next 7 days

[e] none of the other choices

根據投影片規範出的 ML 三關鍵如下圖：



[a]為最短路徑問題。不符合第二項，有程式邏輯定義可以計算出來。

[b]預測樂透中獎號碼。不符合第一項，中獎號碼為隨機產生沒有 underlying pattern。

[c]計算 lab 中所有人的薪水。不符合第二項，有程式邏輯定義可以計算出來。

[d]以預測電商平台用戶接下來七天可能消費的機率來將使用者做 sorting。三項都符合，我們可以利用過往使用者消費的資料、日期、節慶假日等等作為 data input，來

去找出消費者消費的 underlying pattern，而沒有簡單的程式邏輯定義可以做到這件事，所以此情況適合運用 ML 來解決。

Modifications of PLA

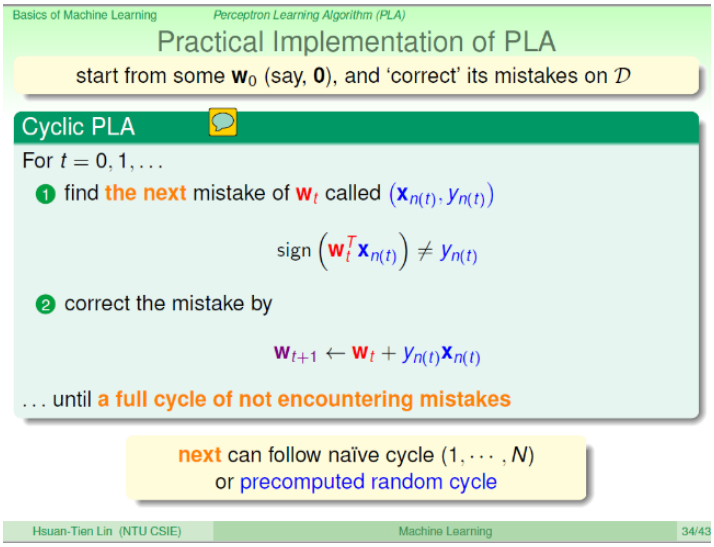
2. ANS [e]

One possible modification of PLA is to insert a per-iteration "learning rate" η_t to the update rule

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_{n(t)}\mathbf{x}_{n(t)} \cdot \eta_t$$

There are many possibilities of setting the learning rate η_t. One is to consider how negative $y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)}$ is, and try to aggressively make $y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{n(t)} > 0$; that is, let $\mathbf{w}_{t+1}$ correctly classify $(\mathbf{x}_{n(t)}, y_{n(t)})$. Another one is to conservatively decrease η_t so there is less oscillation during the final convergence stage. Which of the following update rules make $y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{n(t)} > 0$? Choose the correct answer; explain your answer.

由投影片可知當出現錯誤時，$Sign(\mathbf{w}_t^T\mathbf{x}_{n(t)}) \neq y_{n(t)}$。



$$\text{[a]}\ \ \mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_{n(t)}\mathbf{x}_{n(t)} \cdot (2^{-t})$$

兩邊同承$y_{n(t)}$並同內積$\mathbf{x}_{n(t)}$：

由於內積公式

假設$\mathbf{w}, \mathbf{x}$為兩向量，$\mathbf{w} \cdot \mathbf{x} = \mathbf{w}^\mathbf{T}\mathbf{x} = |\vec{w}||\vec{x}|cos\theta$

可得

$$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{\mathrm{n(t)}} = y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)} + {y_{n(t)}}^2\mathbf{x}_{n(t)}^T\mathbf{x}_{n(t)} \cdot (2^{-t})$$

當$y_{n(t)} = 1$ 且 $Sign(\mathbf{w}_\mathbf{t}^\mathbf{T}\mathbf{x}_{\mathbf{n(t)}})$為負：

$$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{n(t)} = \underbrace{y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)}}_{\text{+ −}} + \underbrace{{y_{n(t)}}^2\mathbf{x}_{n(t)}^T\mathbf{x}_{n(t)} \cdot (2^{-t})}_{\text{+}}$$

無法保證$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{n(t)} > 0$。

當$y_{n(t)} = -1$ 且 $Sign(\mathbf{w_t^T x_{n(t)}})$為正：

$$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{n(t)} = y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)} + y_{n(t)}{}^2\mathbf{x}_{n(t)}^T\mathbf{x}_{n(t)} \cdot (2^{-t})$$

無法保證$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{n(t)} > 0$。

[b] $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_{n(t)}\mathbf{x}_{n(t)} \cdot 0.6211$

兩邊同承$y_{n(t)}$並同內積$\mathbf{x}_{n(t)}$：

由於內積公式

假設$\mathbf{w}, \mathbf{x}$為兩向量，$\mathbf{w} \cdot \mathbf{x} = \mathbf{w^T x} = |\vec{w}||\vec{x}|cos\theta$

可得

$$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{n(t)} = y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)} + y_{n(t)}{}^2\mathbf{x}_{n(t)}^T\mathbf{x}_{n(t)} \cdot 0.6211$$

當$y_{n(t)} = 1$ 且 $Sign(\mathbf{w_t^T x_{n(t)}})$為負：

$$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{n(t)} = y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)} + y_{n(t)}{}^2\mathbf{x}_{n(t)}^T\mathbf{x}_{n(t)} \cdot 0.6211$$

無法保證$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{n(t)} > 0$

當$y_{n(t)} = -1$ 且 $Sign(\mathbf{w_t^T x_{n(t)}})$為正：

$$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{n(t)} = y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)} + y_{n(t)}{}^2\mathbf{x}_{n(t)}^T\mathbf{x}_{n(t)} \cdot 0.6211$$

無法保證$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{n(t)} > 0$。。

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

兩邊同承$y_{n(t)}$並同內積$\mathbf{x}_{n(t)}$：

由於內積公式

假設$\mathbf{w}, \mathbf{x}$為兩向量，$\mathbf{w} \cdot \mathbf{x} = \mathbf{w^T x} = |\vec{w}||\vec{x}|cos\theta$

可得

$$y_{n(t)}\mathbf{w}_{t+1}^{T}\mathbf{x}_{\mathrm{n(t)}} = y_{n(t)}\mathbf{w}_{t}^{T}\mathbf{x}_{n(t)} + {y_{n(t)}}^{2}\mathbf{x}_{n(t)}^{T}\mathbf{x}_{n(t)} \cdot \left(\frac{-y_{n(t)}\mathbf{w}_{t}^{T}\mathbf{x}_{n(t)}}{\left\|\mathbf{x}_{n(t)}\right\|^{2}}\right)$$

簡化式子，且已知$\mathbf{x}_{n(t)}^{T}\mathbf{x}_{n(t)} = \left\|\mathbf{x}_{n(t)}\right\|^{2}$：

$$y_{n(t)}\mathbf{w}_{t+1}^{T}\mathbf{x}_{\mathrm{n(t)}} = y_{n(t)}\mathbf{w}_{t}^{T}\mathbf{x}_{n(t)} - {y_{n(t)}}^{3}\mathbf{w}_{t}^{T}\mathbf{x}_{n(t)}$$

$$y_{n(t)}\mathbf{w}_{t+1}^{T}\mathbf{x}_{\mathrm{n(t)}} = y_{n(t)}\mathbf{w}_{t}^{T}\mathbf{x}_{n(t)}(1 - {y_{n(t)}}^{2})$$

不管是$y_{n(t)} = 1$ *且* $Sign(\mathbf{w}_{t}^{T}\mathbf{x}_{n(t)})$為負，或是$y_{n(t)} = -1$ *且* $Sign(\mathbf{w}_{t}^{T}\mathbf{x}_{n(t)})$為正的情況下，$(1 - {y_{n(t)}}^{2})$*都會等於* 0，故得到$y_{n(t)}\mathbf{w}_{t+1}^{T}\mathbf{x}_{\mathrm{n(t)}} = 0$，非$y_{n(t)}\mathbf{w}_{t+1}^{T}\mathbf{x}_{\mathrm{n(t)}} > 0$。。

$$\mathbf{[d]}\ \ \mathbf{w}_{t+1} \leftarrow \mathbf{w}_{t} + y_{n(t)}\mathbf{x}_{n(t)} \cdot \left(\frac{1}{1+t}\right)$$

兩邊同承$y_{n(t)}$並同內積$\mathbf{x}_{n(t)}$：

由於內積公式

假設$\mathbf{w},\mathbf{x}$為兩向量，$\mathbf{w}\cdot\mathbf{x} = \mathbf{w}^{\mathbf{T}}\mathbf{x} = |\vec{w}||\vec{x}|cos\theta$

可得

$$y_{n(t)}\mathbf{w}_{t+1}^{T}\mathbf{x}_{\mathrm{n(t)}} = y_{n(t)}\mathbf{w}_{t}^{T}\mathbf{x}_{n(t)} + {y_{n(t)}}^{2}\mathbf{x}_{n(t)}^{T}\mathbf{x}_{n(t)} \cdot \left(\frac{1}{1+t}\right)$$

當$y_{n(t)} = 1$ *且* $Sign(\mathbf{w}_{\mathbf{t}}^{\mathbf{T}}\mathbf{x}_{\mathbf{n(t)}})$為負，由定義可知 t>0 故$\left(\frac{1}{1+t}\right) > 0$：

$$y_{n(t)}\mathbf{w}_{t+1}^{T}\mathbf{x}_{n(t)} = \underset{+}{\overset{-}{y_{n(t)}\mathbf{w}_{t}^{T}\mathbf{x}_{n(t)}}} + \underset{+}{\overset{-}{y_{n(t)}^{2}\mathbf{x}_{n(t)}^{T}\mathbf{x}_{n(t)} \cdot \left(\frac{1}{1+t}\right)}}$$

無法保證$y_{n(t)}\mathbf{w}_{t+1}^{T}\mathbf{x}_{n(t)} > 0$。

當$y_{n(t)} = -1$ *且* $Sign(\mathbf{w}_{\mathbf{t}}^{\mathbf{T}}\mathbf{x}_{\mathbf{n(t)}})$為正，由定義可知 t>0 故$\left(\frac{1}{1+t}\right) > 0$：

$$y_{n(t)}\mathbf{w}_{t+1}^{T}\mathbf{x}_{n(t)} = \underset{-}{\overset{+}{y_{n(t)}\mathbf{w}_{t}^{T}\mathbf{x}_{n(t)}}} + \underset{-}{\overset{+}{y_{n(t)}^{2}\mathbf{x}_{n(t)}^{T}\mathbf{x}_{n(t)} \cdot \left(\frac{1}{1+t}\right)}}$$

無法保證$y_{n(t)}\mathbf{w}_{t+1}^{T}\mathbf{x}_{n(t)} > 0$。

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

兩邊同承$y_{n(t)}$並同內積$\mathbf{x}_{n(t)}$：

由於內積公式

假設$\mathbf{w}, \mathbf{x}$為兩向量，$\mathbf{w} \cdot \mathbf{x} = \mathbf{w}^{\mathbf{T}}\mathbf{x} = |\vec{w}||\vec{x}|cos\theta$

可得

$$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{\mathrm{n(t)}} = y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)} + {y_{n(t)}}^2\mathbf{x}_{n(t)}^T\mathbf{x}_{n(t)} \cdot \left[\frac{-y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\left\|\mathbf{x}_{n(t)}\right\|^2} + 1\right]$$

簡化式子，且已知$\mathbf{x}_{n(t)}^T\mathbf{x}_{n(t)} = \left\|\mathbf{x}_{n(t)}\right\|^2$：

$$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{\mathrm{n(t)}} = y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)} - {y_{n(t)}}^3\mathbf{w}_t^T\mathbf{x}_{n(t)} + {y_{n(t)}}^2\mathbf{x}_{n(t)}^T\mathbf{x}_{n(t)}$$

$$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{\mathrm{n(t)}} = y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)}(1 - {y_{n(t)}}^2) + {y_{n(t)}}^2\mathbf{x}_{n(t)}^T\mathbf{x}_{n(t)}$$

不管是$y_{n(t)} = 1$ 且 $Sign(\mathbf{w_t^T x_{n(t)}})$為負，或是$y_{n(t)} = -1$ 且 $Sign(\mathbf{w_t^T x_{n(t)}})$為正的情況下，$y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)}(1 - {y_{n(t)}}^2)$都會等於 0，故剩下：

$$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{\mathrm{n(t)}} = {y_{n(t)}}^2\mathbf{x}_{n(t)}^T\mathbf{x}_{n(t)} > 0$$

由此可知$y_{n(t)}\mathbf{w}_{t+1}^T\mathbf{x}_{n(t)} > 0$。

3. ANS [d]

Dr. Separate decides to use one of the update rules in the previous problem for PLA. When the data set is linear separable, how many choices in the previous problem ensures halting with a "perfect line"? Choose the correct answer; explain the reason behind each perfect halting case.

由投影片可得出以下兩式子：

$$\mathbf{w}_{t+1}^T \mathbf{w}_{t+1} = \mathbf{w}_t^T (\mathbf{w}_{t+1} + \eta_t \cdot y_{n(t)} \mathbf{x}_{n(t)})$$

$$\mathbf{w}_{t+1}^T \mathbf{w}_{t+1} \geq \mathbf{w}_t^T (\mathbf{w}_{t+1} + \eta_t \cdot \min_n y_{n(t)} \mathbf{x}_{n(t)})$$

令$\min_n y_{n(t)} \mathbf{x}_{n(t)} = \rho$：

$$\mathbf{w}_{t+1}^T \mathbf{w}_{t+1} \geq \mathbf{w}_t^T (\mathbf{w}_{t+1} + \eta_t \cdot \rho) \quad \ldots\ldots (1)$$

$$\|\mathbf{w}_{t+1}\|^2 = \left\|\mathbf{w}_t^T + \eta_t \cdot y_{n(t)} \mathbf{x}_{n(t)}\right\|^2$$

$$\|\mathbf{w}_{t+1}\|^2 = \|\mathbf{w}_t^T\|^2 + 2\eta_t \cdot y_{n(t)} \mathbf{w}_t^T \mathbf{x}_{n(t)} + \left\|\eta_t \cdot y_{n(t)} \mathbf{x}_{n(t)}\right\|^2 \quad \ldots\ldots (2)$$

[a] 1
[b] 2
[c] 3
[d] 4
[e] 5

[a] $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_{n(t)} \mathbf{x}_{n(t)} \cdot (2^{-t})$

- 代入 **learning rate**

根據(1)

$$\mathbf{w}_{t+1}^T \mathbf{w}_{t+1} \geq \mathbf{w}_t^T (\mathbf{w}_{t+1} + \eta_t \cdot \rho) \quad \ldots\ldots (1)$$

帶入(2^{-t})：

$$\mathbf{w}_{t+1}^T \mathbf{w}_{t+1} \geq \mathbf{w}_t^T (\mathbf{w}_{t+1} + (2^{-t}) \cdot \rho) \quad \ldots\ldots (A)$$

根據(2)

$$\|\mathbf{w}_{t+1}\|^2 = \|\mathbf{w}_t^T\|^2 + 2\eta_t \cdot y_{n(t)} \mathbf{w}_t^T \mathbf{x}_{n(t)} + \left\|\eta_t \cdot y_{n(t)} \mathbf{x}_{n(t)}\right\|^2 \quad \ldots\ldots (2)$$

帶入(2^{-t})：

$$\|\mathbf{w}_{t+1}\|^2 = \|\mathbf{w}_t^T\|^2 + 2\cdot 2^{-t}\cdot y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)} + \left\|2^{-t}\cdot y_{n(t)}\mathbf{x}_{n(t)}\right\|^2$$

由於$y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)}$<0 且 $\cdot\ {y_{n(t)}}^2 = 1$：

$$\|\mathbf{w}_{t+1}\|^2 \le \|\mathbf{w}_t^T\|^2 + 0 + \left\|2^{-t}\cdot y_{n(t)}\mathbf{x}_{n(t)}\right\|^2$$

$$\|\mathbf{w}_{t+1}\|^2 \le \|\mathbf{w}_t^T\|^2 + 2^{-2t}\cdot \max_n \left\|\mathbf{x}_{n(t)}\right\|^2$$

令$\max_n \left\|\mathbf{x}_{n(t)}\right\|^2 = R^2$：

$$\|\mathbf{w}_{t+1}\|^2 \le \|\mathbf{w}_t^T\|^2 + 2^{-2t}\cdot R^2 \ldots\ldots (B)$$

- **Magic Chain of (A)**

$$\mathbf{w}_{t+1}^T\mathbf{w}_{t+1} \ge \mathbf{w}_t^T(\mathbf{w}_{t+1} + (2^{-t})\cdot\rho) \ldots\ldots (A)$$

根據 magic chain 可得：

$$\mathbf{w}_f^T\mathbf{w}_T \ge \mathbf{w}_f^T\mathbf{w}_0 + (2^0 + 2^1 + 2^2 + \ldots + 2^{-(T-1)})\cdot\rho$$

根據等比級數和公式：

$$= \mathbf{w}_f^T\mathbf{w}_0 + \frac{(1-2^{-T})}{\frac{1}{2}}\cdot\rho$$

$$= \mathbf{w}_f^T\mathbf{w}_0 + 2(1-2^{-T})\cdot\rho$$

由於$\mathbf{w}_0 = 0$：

$$\mathbf{w}_f^T\mathbf{w}_T \ge 2\cdot(1-2^{-T})\cdot\rho \ldots\ldots (C)$$

- **Magic Chain of (B)**

$$\|\mathbf{w}_{t+1}\|^2 \le \|\mathbf{w}_t^T\|^2 + 2^{-2t}\cdot R^2 \ldots\ldots (B)$$

根據 magic chain 且由於$\mathbf{w}_0 = 0$可得：

$$\|\mathbf{w}_T\|^2 \le \frac{4}{3}\cdot(1 - 4^{-T})\cdot R^2 \ldots\ldots\ldots (D)$$

- **統整**

將(C)與(D)合併整理成：

$$1 \ge \frac{\mathbf{w}_f^T\mathbf{w}_T}{\|\mathbf{w}_f\|\|\mathbf{w}_T\|} \ge \frac{\rho\cdot 2\cdot(1-2^{-T})}{\|\mathbf{w}_f\|\sqrt{\frac{4}{3}\cdot(1 - 4^{-T})\cdot R^2}}$$

令$\|\mathbf{w}_f\| = 1$，整理成：

$$1 \geq \frac{\rho \cdot 2 \cdot (1 - 2^{-T})}{\sqrt{\frac{4}{3} \cdot (1 - 4^{-T}) \cdot R^2}}$$

$$1 \geq \frac{\rho^2 \cdot 4 \cdot 1 - 2^{-T}}{\frac{4}{3} \cdot (1 - 4^{-T}) \cdot R^2}$$

$$1 \geq \frac{\rho^2 \cdot 3(1 - 2^{-T})}{(1 + 2^{-T}) \cdot R^2}$$

$$1 \geq \frac{3\rho^2 - 3\rho^2 \cdot 2^{-T}}{R^2 + R^2 \cdot 2^{-T}}$$

$$R^2 + R^2 \cdot 2^{-T} \geq 3\rho^2 - 3\rho^2 \cdot 2^{-T}$$

$$(3\rho^2 + R^2) \cdot 2^{-T} \geq 3\rho^2 - R^2$$

$$2^{-T} \geq \frac{3\rho^2 - R^2}{3\rho^2 + R^2}$$

取 log：

$$\mathrm{T} \geq \log_{\frac{1}{2}} \frac{3\rho^2 - R^2}{3\rho^2 + R^2}$$

由於ρ^2 、R^2皆為常數故取 log 也會是常數，證明出 T 的 upper bond 為常數，會保證 halting。

$$[\mathbf{b}] \quad \mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_{n(t)}\mathbf{x}_{n(t)} \cdot 0.6211$$

- **代入 learning rate**

根據(1)

$$\mathbf{w}_{t+1}^T \mathbf{w}_{\mathrm{t+1}} \geq \mathbf{w}_t^T (\mathbf{w}_{\mathrm{t+1}} + \eta_t \cdot \rho) \ \ldots\ldots (1)$$

帶入 0.6211：

$$\mathbf{w}_{t+1}^T \mathbf{w}_{\mathrm{t+1}} \geq \ \mathbf{w}_t^T (\mathbf{w}_{\mathrm{t+1}} + (0.6211) \ \cdot \rho) \ \ldots\ldots (A)$$

根據(2)

$$\|\mathbf{w}_{\mathrm{t+1}}\|^2 = \ \|\mathbf{w}_t^T\|^2 + 2\eta_t \cdot y_{n(t)} \mathbf{w}_t^T \mathbf{x}_{n(t)} + \left\|\eta_t \ \cdot y_{n(t)} \mathbf{x}_{n(t)}\right\|^2 \ \ldots\ldots (2)$$

帶入 0.6211：

$$\|\mathbf{w}_{t+1}\|^2 = \|\mathbf{w}_t^T\|^2 + 2 \cdot 0.6211 \cdot y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)} + \left\|0.6211 \cdot y_{n(t)}\mathbf{x}_{n(t)}\right\|^2$$

由於$y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)}$<0，且$y_{n(t)}{}^2 = 1$：

$$\|\mathbf{w}_{t+1}\|^2 \leq \|\mathbf{w}_t^T\|^2 + 0 + \left\|0.6211 \cdot y_{n(t)}\mathbf{x}_{n(t)}\right\|^2$$

$$\|\mathbf{w}_{t+1}\|^2 \leq \|\mathbf{w}_t^T\|^2 + 0.6211^2 \cdot \max_n \left\|\mathbf{x}_{n(t)}\right\|^2$$

令$\max_n \left\|\mathbf{x}_{n(t)}\right\|^2 = R^2$：

$$\|\mathbf{w}_{t+1}\|^2 \leq \|\mathbf{w}_t^T\|^2 + 0.6211^2 \cdot R^2 \ldots\ldots (B)$$

- **Magic Chain of (A)**

$$\mathbf{w}_{t+1}^T\mathbf{w}_{t+1} \geq \mathbf{w}_t^T(\mathbf{w}_{t+1} + 0.6211 \cdot \rho) \ldots\ldots (A)$$

根據 magic chain 可得：

$$\mathbf{w}_f^T\mathbf{w}_T \geq \mathbf{w}_f^T\mathbf{w}_0 + 0.6211 \cdot T \cdot \rho$$

由於$\mathbf{w}_0 = 0$：

$$\mathbf{w}_f^T\mathbf{w}_T \geq 0.6211 \cdot T \cdot \rho \ldots\ldots (\mathrm{C})$$

- **Magic Chain of (B)**

$$\|\mathbf{w}_{t+1}\|^2 \leq \|\mathbf{w}_t^T\|^2 + 0.6211^2 \cdot R^2 \ldots\ldots (B)$$

根據 magic chain 且由於$\mathbf{w}_0 = 0$可得：

$$\|\mathbf{w}_\mathrm{T}\|^2 \leq 0.6211^2 \cdot T \cdot R^2 \ldots\ldots\ldots (D)$$

- **統整**

將(C)與(D)合併整理成：

$$1 \geq \frac{\mathbf{w}_f^T\mathbf{w}_T}{\|\mathbf{w}_f\|\|\mathbf{w}_\mathrm{T}\|} \geq \frac{0.6211 \cdot T \cdot \rho}{\|\mathbf{w}_f\|\sqrt{0.6211^2 \cdot T \cdot R^2}}$$

令$\|\mathbf{w}_f\| = 1$，整理成：

$$1 \geq \frac{0.6211 \cdot T \cdot \rho}{\sqrt{0.6211^2 \cdot T \cdot R^2}}$$

$$1 \geq \frac{\rho^2 \cdot 0.6211^2 \cdot T}{0.6211^2 \cdot R^2}$$

$$1 \geq \frac{\rho^2 T}{R^2}$$

$$\mathrm{T} \leq \frac{R^2}{\rho^2}$$

由於ρ^2、R^2皆為常數，證明出 T 的 upper bond 為常數，會保證 halting。

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

這邊舉一反例，假設$\mathbf{w}_t = 0$，這樣後面的$\frac{-y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2}$會因為$\mathbf{w}_t = 0 = \mathbf{w}_f^T$，整項就會是 0。0+0=0，這樣就沒有更新到，所以無法保證 halting。

$$[\mathbf{d}] \quad \mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_{n(t)}\mathbf{x}_{n(t)} \cdot \left(\frac{1}{1+t}\right)$$

- **代入 learning rate**

根據(1)

$$\mathbf{w}_{t+1}^T\mathbf{w}_{t+1} \geq \mathbf{w}_t^T(\mathbf{w}_{t+1} + \eta_t \cdot \rho) \ \ldots\ldots(1)$$

帶入$\frac{1}{1+t}$：

$$\mathbf{w}_{t+1}^T\mathbf{w}_{t+1} \geq \ \mathbf{w}_t^T(\mathbf{w}_{t+1} + (\frac{1}{1+t}) \cdot \rho) \ \ldots\ldots(A)$$

根據(2)

$$\|\mathbf{w}_{t+1}\|^2 = \ \|\mathbf{w}_t^T\|^2 + 2\eta_t \cdot y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)} + \left\|\eta_t \ \cdot y_{n(t)}\mathbf{x}_{n(t)}\right\|^2 \ \ldots\ldots(2)$$

帶入$\frac{1}{1+t}$：

$$\|\mathbf{w}_{t+1}\|^2 = \ \|\mathbf{w}_t^T\|^2 + 2 \cdot (\frac{1}{1+t}) \cdot y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)} + \left\|\frac{1}{1+t} \cdot y_{n(t)}\mathbf{x}_{n(t)}\right\|^2$$

由於$y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)}$<0 且，$y_{n(t)}{}^2 = 1$：

$$\|\mathbf{w}_{t+1}\|^2 \leq \|\mathbf{w}_t^T\|^2 + 0 + \left\|\frac{1}{1+t} \cdot y_{n(t)}\mathbf{x}_{n(t)}\right\|^2$$

$$\|\mathbf{w}_{t+1}\|^2 \leq \|\mathbf{w}_t^T\|^2 + (\frac{1}{1+t})^2 \cdot \max_n \left\|\mathbf{x}_{n(t)}\right\|^2$$

令$\max_n \left\|\mathbf{x}_{n(t)}\right\|^2 = R^2$：

$$\|\mathbf{w}_{t+1}\|^2 \leq \|\mathbf{w}_t^T\|^2 + (\frac{1}{1+t})^2 \cdot R^2 \ \ldots\ldots(B)$$

- **Magic Chain of (A)**

$$\mathbf{w}_{t+1}^{T}\mathbf{w}_{t+1} \geq \mathbf{w}_{t}^{T}(\mathbf{w}_{t+1} + (\frac{1}{1+t}) \cdot \rho) \ \ldots\ldots (A)$$

根據 magic chain 可得：

$$\mathbf{w}_{f}^{T}\mathbf{w}_{T} \geq \mathbf{w}_{f}^{T}\mathbf{w}_{0} + (1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{T}) \cdot \rho$$

由於$\mathbf{w}_0 = 0$：

$$\mathbf{w}_{f}^{T}\mathbf{w}_{T} \geq \sum_{k=1}^{T} \frac{1}{k} \cdot \rho \ \ldots\ldots (\mathrm{C})$$

- **Magic Chain of (B)**

$$\|\mathbf{w}_{t+1}\|^2 \leq \|\mathbf{w}_{t}^{T}\|^2 + (\frac{1}{1+t})^2 \cdot R^2 \ldots\ldots (B)$$

根據 magic chain 且由於$\mathbf{w}_0 = 0$可得：

$$\|\mathbf{w}_{\mathrm{T}}\|^2 \leq (1^2 + (\frac{1}{2})^2 + (\frac{1}{3})^2 + \ldots + (\frac{1}{T})^2) \cdot R^2$$

$$\mathbf{w}_{f}^{T}\mathbf{w}_{T} \geq \sum_{k=1}^{T} \frac{1}{k^2} \cdot R^2 \ \ldots\ldots\ldots (D)$$

- **統整**

將(C)與(D)合併整理成：

$$1 \geq \frac{\mathbf{w}_{f}^{T}\mathbf{w}_{T}}{\|\mathbf{w}_f\|\|\mathbf{w}_{\mathrm{T}}\|} \geq \frac{\sum_{k=1}^{T} \frac{1}{k} \cdot \rho}{\|\mathbf{w}_f\| \sqrt{\sum_{k=1}^{T} \frac{1}{k^2} \cdot R^2}}$$

令$\|\mathbf{w}_f\| = 1$，整理成：

$$1 \geq \frac{(\sum_{k=1}^{T} \frac{1}{k})^2 \cdot \rho^2}{\sum_{k=1}^{T} \frac{1}{k^2} \cdot R^2}$$

由於ρ^2、R^2皆為常數，分子是調和級數的平方$(\sum_{k=1}^{T} \frac{1}{k})^2$，分母為$\sum_{k=1}^{T} \frac{1}{k^2}$，假設 a, b 都大於 0，(a+b)^2 > a^2+b^2，故分子會上升的比分母快。將$\frac{(\sum_{k=1}^{T} \frac{1}{k})^2}{\sum_{k=1}^{T} \frac{1}{k^2}}$視為 F(T)，F(T)會依據 T 變大而升高。而ρ^2、R^2皆為常數，F(T)*常數有小於等於 1 的限制，F(T)又會隨著 T 變大而升高，F(T)有限制，T 就有限制，所以證明出 T 有

upper bond，會保證 halting。

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

- **代入 learning rate**

根據(1)

$$\mathbf{w}_{t+1}^T\mathbf{w}_{t+1} \geq \mathbf{w}_t^T(\mathbf{w}_{t+1} + \eta_t \cdot \rho) \ \ldots\ldots(1)$$

帶入$\left[\frac{-y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2} + 1\right]$：

$$\mathbf{w}_{t+1}^T\mathbf{w}_{t+1} \geq \mathbf{w}_t^T(\mathbf{w}_{t+1} + \left[\frac{-y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2} + 1\right] \cdot \rho)$$

由於$\frac{-y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2} > 0$：

$$\mathbf{w}_{t+1}^T\mathbf{w}_{t+1} \geq \mathbf{w}_t^T(\mathbf{w}_{t+1} + 1 \cdot \rho) \ldots\ldots(A)$$

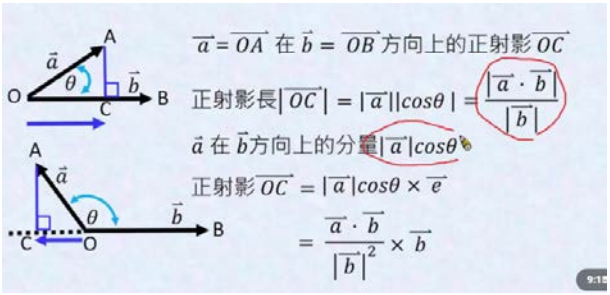
根據投影片的公式：

$$\|\mathbf{w}_{t+1}\|^2 = \|\mathbf{w}_t^T + \eta_t \cdot y_{n(t)}\mathbf{x}_{n(t)}\|^2$$

帶入$\left[\frac{-y_{n(t)}\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2} + 1\right]$：

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

根據正射影公式，且$y_{n(t)} * y_{n(t)} = 1$：



$$\leq \left\|(\mathbf{w}_t^T - \mathbf{x}_{n(t)}\frac{\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2}) + y_{n(t)}\mathbf{x}_{n(t)}\right\|^2$$

$$\leq \left\|(\mathbf{w}_t^T - \mathbf{x}_{n(t)}\frac{\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2})\right\|^2 + 2y_{n(t)}\mathbf{x}_{n(t)}(\mathbf{w}_t^T - \mathbf{x}_{n(t)}\frac{\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2}) + \|y_{n(t)}\mathbf{x}_{n(t)}\|^2$$

由於$2y_{n(t)}\mathbf{x}_{n(t)}\left(\mathbf{w}_t^T - \mathbf{x}_{n(t)}\frac{\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2}\right) < 0$：

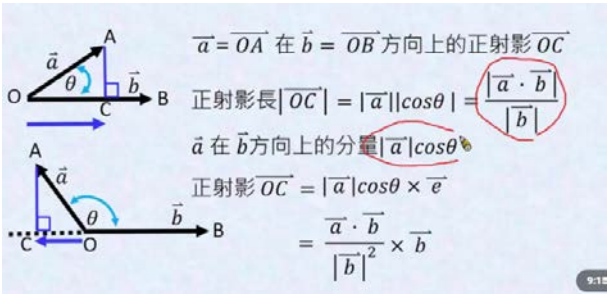
$$\leq \left\|(\mathbf{w}_t^T - \mathbf{x}_{n(t)}\frac{\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2})\right\|^2 + \|y_{n(t)}\mathbf{x}_{n(t)}\|^2$$

$$= \|\mathbf{w}_t^T\|^2 - 2\mathbf{w}_t^T\mathbf{x}_{n(t)}\frac{\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2} + \left\|\mathbf{x}_{n(t)}\frac{\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2}\right\|^2 + \|y_{n(t)}\mathbf{x}_{n(t)}\|^2$$

由於$-2\mathbf{w}_t^T\mathbf{x}_{n(t)}\frac{\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2} < 0$：

$$\leq \|\mathbf{w}_t^T\|^2 + \left\|\mathbf{x}_{n(t)}\frac{\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2}\right\|^2 + \|y_{n(t)}\mathbf{x}_{n(t)}\|^2$$

根據正射影公式，可見$\mathbf{x}_{n(t)} \geq \mathbf{x}_{n(t)}\frac{\mathbf{w}_t^T\mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2}$：



$$\leq \|\mathbf{w}_t^T\|^2 + \|\mathbf{x}_{n(t)}\|^2 + \|\mathbf{x}_{n(t)}\|^2$$

$$\leq \|\mathbf{w}_t^T\|^2 + 2\max_n\|\mathbf{x}_{n(t)}\|^2$$

$$\|\mathbf{w}_{t+1}\|^2 = \|\mathbf{w}_t^T\|^2 + 2\max_n\|\mathbf{x}_{n(t)}\|^2 \ldots\ldots(B)$$

令$\max_n\|\mathbf{x}_{n(t)}\|^2 = R^2$：

$$\|\mathbf{w}_{t+1}\|^2 = \|\mathbf{w}_t^T\|^2 + 2R^2 \ldots\ldots(B)$$

- **Magic Chain of (A)**

$$\mathbf{w}_{t+1}^T\mathbf{w}_{t+1} \geq \mathbf{w}_t^T(\mathbf{w}_{t+1} + 1\cdot\rho) \ldots\ldots(A)$$

根據 magic chain 可得：

$$\mathbf{w}_f^T\mathbf{w}_T \geq \mathbf{w}_f^T\mathbf{w}_0 + \mathrm{T}\cdot\rho\mathbf{w}_f^T$$

由於$\mathbf{w}_0 = 0$：

$$\mathbf{w}_f^T\mathbf{w}_T \geq \mathrm{T}\cdot\rho\mathbf{w}_f^T \ldots\ldots(\mathrm{C})$$

- **Magic Chain of (B)**

$$\|\mathbf{w}_{t+1}\|^2 = \|\mathbf{w}_t^T\|^2 + 2R^2 \ldots\ldots (B)$$

根據 magic chain 且由於$\mathbf{w}_0 = 0$可得：

$$\|\mathbf{w}_T\|^2 \leq 2R^2 \ldots\ldots\ldots (D)$$

- **統整**

將(C)與(D)合併整理成：

$$1 \geq \frac{\mathbf{w}_f^T \mathbf{w}_T}{\|\mathbf{w}_f\| \|\mathbf{w}_T\|} \geq \frac{T \cdot \rho \mathbf{w}_f^T}{\|\mathbf{w}_f\| 2R^2.}$$

令$\|\mathbf{w}_f\| = 1$，整理成：

$$T \leq \frac{2R^2}{\mathbf{w}_f^T \mathbf{w}_f \rho^2}$$

由於ρ^2、R^2皆為常數，且前面另$\|\mathbf{w}_f\| = 1$，所以$\mathbf{w}_f^T \mathbf{w}_f = 1$。證明出 T 的 upper bond 為常數，會保證 halting。

4. ANS[c]

Consider online spam detection with machine learning. We will represent each email $\mathbf{x}$ by the distinct words that it contains. In particular, assume that there are at most m distinct words in each email, and each word belongs to a big dictionary of size $d \geq m$. The i-th component x_i is defined as $[\![\text{word } i \text{ is in email } \mathbf{x}]\!]$ for $i = 1, 2, \ldots, d$, and $x_0 = 1$ as always. We will assume that d_+ of the words in the dictionary are more spam-like, and $d_- = d - d_+$ of the words are less spam-like. A simple function that classifies whether an email is a spam is to count $z_+(\mathbf{x})$, the number of more spam-like words with the email (ignoring duplicates), and $z_-(\mathbf{x})$, the number of less spam-like words in the email, and classify by

$$f(\mathbf{x}) = \text{sign}(z_+(\mathbf{x}) - z_-(\mathbf{x}) - 0.5).$$

That is, an email $\mathbf{x}$ is classified as a spam iff the integer $z_+(\mathbf{x})$ is more than the integer $z_-(\mathbf{x})$.

Assume that f can perfectly classify any email into spam/non-spam, but is unknown to us. We now run an online version of PLA to try to approximate f. That is, we maintain a weight vector $\mathbf{w}_t$ in the online PLA, initialized with $\mathbf{w}_0 = \mathbf{0}$. Then for every email $\mathbf{x}_t$ encountered at time t, the algorithm makes a prediction $\text{sign}(\mathbf{w}_t^T \mathbf{x}_t)$, and receives a true label y_t. If the prediction is not the same as the true label (i.e. a mistake), the algorithm updates $\mathbf{w}_t$ by

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_t \mathbf{x}_t.$$

Otherwise the algorithm keeps $\mathbf{w}_t$ without updating

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t.$$

What is the maximum number of mistakes that the online PLA can make for this spam classification problem? Choose the tightest upper bound; explain your answer.

Note: For those who know the bag-of-words representation for documents, the representation we use is a simplification that ignores duplicates of the same word.

[a] $4(m+1)^2$
[b] $4(d+1)(m+1)$
[c] $(4d+1)(m+1)$
[d] $(4d+1)^2$
[e] $\frac{(d+1)m}{(d_+ - d_-)^2}$

此題目標是要推導出 mistakes 的 tightest upper bound。

由 Lecture 1 投影片中的第 39~41 頁可得以下式子：

- $\mathbf{w}_f{}^T \mathbf{w}_{t+1} \geq \mathbf{w}_f{}^T \mathbf{w}_t + \min_n y_n \mathbf{w}_f{}^T \mathbf{x}_n$

  假設T為 mistake 的次數，由 Magic Chain 得：

  $$\mathbf{w}_f{}^T \mathbf{w}_T \geq \mathbf{w}_f{}^T \mathbf{w}_0 + \min_n y_n \mathbf{w}_f{}^T \mathbf{x}_n * T$$

  由於$\mathbf{w}_0 = 0$：

  $$\mathbf{w}_f{}^T \mathbf{w}_T \geq \min_n y_n \mathbf{w}_f{}^T \mathbf{x}_n * T \quad ---------- (1)$$

- $\|\mathbf{w}_{t+1}\|^2 \leq \|\mathbf{w}_t\|^2 + \max_n \|\mathbf{x}_n\|^2$

  假設T為 mistake 的次數，由 Magic Chain 得：

$$\|\mathbf{w}_T\|^2 \leq \|\mathbf{w}_0\|^2 + \max_n \|\mathbf{x}_n\|^2 * T$$

  由於$\mathbf{w}_0 = 0$：

$$\|\mathbf{w}_T\|^2 \leq \max_n \|\mathbf{x}_n\|^2 * T$$

$$\|\mathbf{w}_T\| \leq \max_n \|\mathbf{x}_n\| * \sqrt{T} \quad ---------- (2)$$

由得出的(1)與(2)可得：

$$1 \geq \frac{\mathbf{w}_f{}^T \mathbf{w}_T}{\|\mathbf{w}_f\| \|\mathbf{w}_T\|} \geq \frac{\min_n y_n \mathbf{w}_f{}^T \mathbf{x}_n * T}{\|\mathbf{w}_f\| * \max_n \|\mathbf{x}_n\| * \sqrt{T}}$$

由於$\frac{\mathbf{w}_f{}^T \mathbf{w}_T}{\|\mathbf{w}_f\| \|\mathbf{w}_T\|}$為兩向量內積除以兩向量長度相承，

由於$\mathbf{w}_f{}^T \mathbf{w}_T = \|\mathbf{w}_f\| \|\mathbf{w}_T\| \cos\theta$，$\cos\theta \leq 1$，所以可以得：

$$1 \geq \frac{\min_n y_n \mathbf{w}_f{}^T \mathbf{x}_n * T}{\|\mathbf{w}_f\| * \max_n \|\mathbf{x}_n\| * \sqrt{T}}$$

$$T \leq \frac{\left(\max_n \|\mathbf{x}_n\| * \|\mathbf{w}_f\|\right)^2}{\min_n y_n \mathbf{w}_f{}^T \mathbf{x}_n{}^2}$$

- $\max_n \|\mathbf{x}_n\|$

  $\max_n \|\mathbf{x}_n\|$代表與 f(x)距離最遠的$\mathbf{x}_n$的距離。

  題目中說每封信最多只會有 m 個 distinct word，word 會來自 dictionary，dictionary 有 d 個 word，m ≤ d。

  設定$x_0 = 1$，所以$\mathbf{x}_n$會有 d+1 個元素，但最多只會有 m+1 個值為 1 的元素，其他為 0。可得：

$$\max_n \|\mathbf{x}_n\| = \sqrt{1^2 * (m+1)} = \sqrt{m+1}$$

- $\min_n y_n \mathbf{w}_f{}^T \mathbf{x}_n{}^2$

  $\min_n y_n \mathbf{w}_f{}^T \mathbf{x}_n{}^2$為$\mathbf{w}_f$與某$\mathbf{x}_n$內積要最小，代表與 f(x)最接近的$\mathbf{x}_n$的情況，也就是$\mathbf{x}_n$中 more spam-like 的字數量要等於 less spam-like 的字數量。

故$f(\mathbf{x}) = \text{sign}(z_+(\mathbf{x}) - z_-(\mathbf{x}) - 0.5)$當中的$z_+(\mathbf{x}) - z_-(\mathbf{x}) = 0$。

且-0.5 為 threshold，就是$\mathbf{w}_f$的w_0。

要讓$\mathbf{w}_f{}^T\mathbf{x}_n$與完美的$z_+(\mathbf{x}) - z_-(\mathbf{x}) - 0.5$結果一樣，

並且是在符合$\min_n y_n \mathbf{w}_f{}^T\mathbf{x}_n{}^2$的情況下，故：

$$z_+(\mathbf{x}) - z_-(\mathbf{x}) = 0$$

$$\mathbf{w}_f{}^T\mathbf{x}_n = -0.5$$

可以假設 d=5, m=4

$$\mathbf{w}_f = [-0.5, 1, 1, -1, -1, 1]$$

且

$$\mathbf{x}_n = [x_0, x_1, x_2, x_3, x_4, x_5]$$

x_1, x_2為 more spam-like 的 word，x_3, x_4, x_5為 less spam-like 的 word。

$$\mathbf{x}_n = [1, 1, 1, 1, 1, 0]$$

這樣符合

$$\mathbf{w}_f{}^T\mathbf{x}_n = -0.5 = z_+(\mathbf{x}) - z_-(\mathbf{x}) - 0.5 = 2 - 2 - 0.5$$

也符合$\min_n y_n \mathbf{w}_f{}^T\mathbf{x}_n{}^2$，因為$\mathbf{w}_0 = 0$，故一定會有錯誤產生進而去更新權重，可知$\mathbf{w}_f$當中不會有元素是等於 0 的情況。

權重更新規則：

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_t\mathbf{x}_t.$$

由於$\mathbf{x}_t$當中的元素一定會是 1 或 0，所以更新後的$\mathbf{w}_f$的元素也會是整數。

由上述可知$\mathbf{w}_f$當中的元素除了$w_0 = -0.5$，其他元素都會是不等於 0 的整數(若為 0 代表該字既不是 spam-like 也不是 unspam-like 就不會出現在 dictionary 中)。這樣就可以知道要符合$\min_n y_n \mathbf{w}_f{}^T\mathbf{x}_n{}^2$，$\mathbf{w}_f$除了$w_0$的其他元素都會是-1 或+1，這樣才會讓$\mathbf{x}_n$最接近 f(x)。而$\mathbf{w}_f$除了$w_0$有 d 個元素，可得：

$$\|\mathbf{w}_f\| = \sqrt{0.5^2 + 1^2 * d} = \sqrt{\frac{1}{4} + d}$$

$y_n = 1 or\ -1$

$$\min_n y_n \mathbf{w}_f{}^T \mathbf{x}_n{}^2 = (y_n * 0.5)^2 = \frac{1}{4}$$

最後可以算出：

$$T \leq \frac{\left(\max_n \|\mathbf{x}_n\| * \|\mathbf{w}_f\|\right)^2}{\min_n y_n \mathbf{w}_f{}^T \mathbf{x}_n{}^2} = \frac{(\sqrt{m+1})^2 * (\sqrt{\frac{1}{4}+d})^2}{\frac{1}{4}} = (\mathbf{1}+\mathbf{4d})(\mathbf{m}+\mathbf{1})$$

選 c。

5. ANS[b]

For multiclass classification with K classes, the multiclass PLA maintains K weight vectors $\mathbf{w}_1^{(t)}$, $\mathbf{w}_2^{(t)}$, ..., $\mathbf{w}_K^{(t)}$, where $^{(t)}$ is used to indicate the vectors in the t-th iteration. The goal is to obtain some hypothesis g represented by the solutions $\mathbf{w}_k^*$ returned from the algorithm, where g makes a prediction by

$$g(\mathbf{x}) = \text{argmax}_{k\in\{1,2,\ldots,K\}} (\mathbf{w}_k^*)^T \mathbf{x} \ .$$

All weight vectors are initialized to $\mathbf{0}$ in the beginning. In iteration t, the multiclass PLA first finds a "mistake example" with index $n(t)$ that

$$y_{n(t)} \neq \text{argmax}_{k\in\{1,2,\ldots,K\}} \left(\mathbf{w}_k^{(t)}\right)^T \mathbf{x}_{n(t)} \ .$$

Abbreviate $y_{n(t)}$ as y and let y' be the argmax result above. The multiclass PLA then updates weight vectors $\mathbf{w}_y^{(t)}$ and $\mathbf{w}_{y'}^{(t)}$ by

$$\begin{aligned} \mathbf{w}_y^{(t+1)} &\leftarrow \mathbf{w}_y^{(t)} + \mathbf{x}_{n(t)} \\ \mathbf{w}_{y'}^{(t+1)} &\leftarrow \mathbf{w}_{y'}^{(t)} - \mathbf{x}_{n(t)} \end{aligned}$$

If there are no more mistakes, return all $\mathbf{w}_k^{(t)}$ as the multiclass PLA solution $\mathbf{w}_k^*$. $w_1^* = w_1^{(t)}, w_2^* = w_2^{(t)}, \cdots$

Assume that $K = 2$. Given a linear separable data set $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^N$, where $y_n \in \{1, 2\}$. Run the binary PLA (initialized with the zero weight vector) taught in class on the data set with transformed labels $\tilde{y}_n = 2y_n - 3$ (so $\tilde{y}_n \in \{-1, +1\}$). Then, run the multiclass PLA above with the same order of examples (i.e. $n(t)$ across the two algorithms are the same). What is the relationship between $\mathbf{w}_{\text{PLA}}$ produced by the binary PLA, and the $\mathbf{w}_1^*$ and $\mathbf{w}_2^*$ produced by the multiclass PLA? Choose the correct answer; explain your answer.

Note: If there is a tie in argmax, we will assume the "worst-case" scenario that argmax returns some k that is not equal to $y_{n(t)}$. Similarly, for the binary PLA, if sign(0) is encountered when checking the mistake on some $(\mathbf{x}_{n(t)}, y_{n(t)})$, we will assume the "worst-case" scenario that sign returns the sign that is not equal to $y_{n(t)}$.

[a] $\mathbf{w}_{\text{PLA}} = \mathbf{w}_1^* = -\mathbf{w}_2^*$

[b] $\mathbf{w}_{\text{PLA}} = -\mathbf{w}_1^* = \mathbf{w}_2^*$

[c] $\mathbf{w}_{\text{PLA}} = \mathbf{w}_2^* - \mathbf{w}_1^*$

[d] $\mathbf{w}_{\text{PLA}} = \mathbf{w}_1^* - \mathbf{w}_2^*$

[e] none of the other choices

根據題目假設$D = (\mathbf{x}_1, \widetilde{y_1}), (\mathbf{x}_2, \widetilde{y_2}) = \{([1,1], 1), ([-1,-1], -1)\}$，

且$\mathbf{w}_1 = [0,0]$，$\mathbf{w}_2 = [0,0]$，以這兩筆資料來跑一次例子。

iteration one

- multiclass PLA

依$y_{\text{n(t)}} \neq argmax_{k\in\{1,2,\ldots,K\}} (\mathbf{w}_k^{(t)})^T \mathbf{x}_{n(t)}$判斷該不該更新：

$$(\mathbf{x}_1, \widetilde{y_1}) = ([1,1], 1)$$

$$\mathbf{w}_1{}^T \mathbf{x}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} [1, \quad 1] = 0$$

$$\mathbf{w}_2{}^T \mathbf{x}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} [1, \quad 1] = 0$$

$$argmax_{k\in\{1,2,\ldots,K\}} \left(\mathbf{w}_k^{(t)}\right)^T \mathbf{x}_{n(t)} = 1$$

$$\widetilde{y_n} = 2y_n - 3 \quad , \widetilde{y_1'} = -1 \neq \widetilde{y_1} = 1$$

要更新：

$$\mathbf{w}_y^{(t+1)} \quad \leftarrow \quad \mathbf{w}_y^{(t)} + \mathbf{x}_{n(t)}$$

更新$\mathbf{w}_2$：$\mathbf{w}_2^{(1)} = \mathbf{w}_2^{(0)} + \mathbf{x}_2$

$$\mathbf{w}_2^{(1)} = [1, \quad 1] = [0, \quad 0] + [1, \quad 1]$$

$$\mathbf{w}_{y'}^{(t+1)} \quad \leftarrow \quad \mathbf{w}_{y'}^{(t)} - \mathbf{x}_{n(t)}$$

更新$\mathbf{w}_1$：$\mathbf{w}_1^{(1)} = \mathbf{w}_1^{(0)} - \mathbf{x}_1$

$$\mathbf{w}_1^{(1)} = [-1, \quad -1] = [0, \quad 0] - [1, \quad 1]$$

- binary PLA

$$\mathbf{w}_0 = [0,0]$$

依 $y_{n(t)} \neq Sign(\mathbf{w}_t^{\mathrm{T}}\mathbf{x}_{\mathrm{n(t)}})$判斷要不要更新：

$$(\mathbf{x}_1, \widetilde{y_1}) = ([1,1], 1)$$

$$\mathbf{w}_0{}^T\mathbf{x}_1 = \begin{bmatrix}0\\0\end{bmatrix}[1, \quad 1] = 0$$

根據題目註釋，$\mathrm{sign}(0) = -\widetilde{y_1} = -1$

要更新：

$$\mathbf{w}_{t+1}^T = \mathbf{w}_t + y_{n(t)}\mathbf{x}_{n(t)}$$

$$\mathbf{w}_1 = \mathbf{w}_0 + (1)\mathbf{x}_1$$

$$\mathbf{w}_1 = [1, \quad 1] = [0, \quad 0] + (1)[1, \quad 1]$$

iteration two

- multiclass PLA

依$y_{\mathrm{n(t)}} \neq argmax_{k\in\{1,2,\ldots,K\}}(\mathbf{w}_k^{(t)})^T\mathbf{x}_{n(t)}$判斷該不該更新：

$$(\mathbf{x}_1, \widetilde{y_1}) = ([1,1], 1)$$

$$\mathbf{w}_1{}^T\mathbf{x}_1 = \begin{bmatrix}-1\\-1\end{bmatrix}[1, \quad 1] = -2$$

$$\mathbf{w}_2{}^T\mathbf{x}_1 = \begin{bmatrix}1\\1\end{bmatrix}[1, \quad 1] = 2$$

$$argmax_{k\in\{1,2,\ldots,K\}}\left(\mathbf{w}_k^{(t)}\right)^T\mathbf{x}_{n(t)} = 2$$

$$\widetilde{y_n} = 2y_n - 3 \quad , \widetilde{y_1}, = 1 = \widetilde{y_1} = 1$$

不用更新。

$$(\mathbf{x}_2, \widetilde{y_2}) = ([-1,-1], -1)$$

$$\mathbf{w}_1{}^T\mathbf{x}_2 = \begin{bmatrix}-1\\-1\end{bmatrix}[-1 \quad -1] = 2$$

$$\mathbf{w}_2{}^T\mathbf{x}_2 = \begin{bmatrix}1\\1\end{bmatrix}[-1 \quad -1] = -2$$

$$argmax_{k\in\{1,2,\ldots,K\}}\left(\mathbf{w}_k^{(t)}\right)^T\mathbf{x}_{n(t)} = 1$$

$$\widetilde{y_n} = 2y_n - 3 \quad , \widetilde{y_1}, = -1 = \widetilde{y_1} = -1$$

不用更新。

$$\mathbf{w}_1^{(2)} = [-1, \quad -1] = \mathbf{w}_1^*$$

$$\mathbf{w}_2^{(2)} = [1, \quad 1] = \mathbf{w}_2^*$$

- binary PLA

$$\mathbf{w}_1 = [1,1]$$

依 $y_{n(t)} \neq Sign(\mathbf{w}_t^{\mathrm{T}}\mathbf{x}_{\mathrm{n(t)}})$判斷要不要更新：

$$(\mathbf{x}_1, \widetilde{y_1}) = ([1,1], 1)$$

$$\mathbf{w}_1{}^T\mathbf{x}_1 = \begin{bmatrix}1\\1\end{bmatrix}[1 \quad 1] = 2$$

$$\mathrm{sign}(2) = 1 = \widetilde{y_1}$$

不用更新。

$$(\mathbf{x}_2, \widetilde{y_2}) = ([-1,-1], -1)$$

$$\mathbf{w}_1{}^T\mathbf{x}_1 = \begin{bmatrix}1\\1\end{bmatrix}[-1 \quad -1] = -2$$

$$\mathrm{sign}(-2) = -1 = \widetilde{y_2}$$

不用更新。

$$\mathbf{w}_2 = [1 \quad 1] = \mathbf{w}_{PLA}$$

經過兩個 iteration，發現$\mathbf{w}_{PLA} = \mathbf{w}_2^* = -\mathbf{w}_1^*$，故答案選 b。

The Learning Problems

6. ANS [d]

Consider the following process: "We set up five cameras and capture video sequences from five different angles. The video sequences are timed but not specifically labeled. After collecting the sequences, we train a learning model that outputs a real-valued vector for each image in the video. The goal is that images that are taken at similar time stamps should be mapped to similar vectors, and images that are at distant time stamps should be mapped to different vectors. The learned model can then be combined with a variety of tasks, such as being mixed with some labeled data for object recognition, or some interactive environment that trains a robot to grab an object from the table." Which of the following best matches the process? Choose the best answer; explain your answer.

[a] active learning
[b] reinforcement learning
[c] multi-class classification
[d] self-supervised learning
[e] online learning

題目描述說設立五台相機將 video sequences 作為 data input，且他們是**沒有 labeled 的**，經過 training 產生一個 learning model 可以把 input 的 video image **output 成 real-valued vector**。目標是時間相近產生的會被 mapped 到相近的 vector。之後利用前面 output 的 vector 再加上其他任務像是**有 label 的** object recognition 進行訓練。

符合[d] self-supervised learning 的情況，因為一開始是利用 unlabeled 的 input 進行訓練，並且有 self-defined goal。之後再利用前面 output 的 vector 加上有 label 的資料進行二度訓練。

self-supervised learning 是將沒有 label 的資料自行訓練出 label，之後再拿訓練好的 output 當作 feature，根據要做的任務去進行 fine tune 或是二次訓練。

The Learning Problems

7. ANS [c]

Consider formulating a learning problem for building a news tagger. First, we gather a training data set that consists of 1126 news articles, stored in utf8 encoding, and ask domain experts to 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 the best answer; explain your answer.

[a] regression, unsupervised learning, active learning, concrete features
[b] text generation, semi-supervised learning, batch learning, concrete features
[c] multilabel classification, semi-supervised learning, batch learning, raw features
[d] regression, reinforcement learning, batch learning, concrete features
[e] multilabel classification, supervised learning, online learning, concrete features

(We are definitely not hinting that you should build a news tagger this way. :-D)

題目描述說要建立一個 news tagger。training dataset 為 1126 news articles，請該領域專家人工將文章做分類 tag。一篇文章可以有好幾個分類 tag。另外再從 database 拿出其他 11265566 news articles，不做人工 labeling。

符合[c]

- multilabel classification：有好幾種文章分類 tag，且一篇文章可以有多個 label。
- semi-supervised learning ：有一部分的訓練資料是沒有 label 的，像是題目說的只有 1126 news articles 是有 labeled 的，剩下 11265566 news articles 沒有，且他們都能作為 input 進行訓練。
- batch learning：一次給完所有 input data。
- raw features：news article 為文章，本身對機器沒有意義需要經過 feature engineering 做處理再進行訓練。

Off-Training-Set Error

8. ANS [b]

As discussed in lecture 3, what we really care about is whether $g \approx f$ *outside* $\mathcal{D}$. For a set of "universe" examples $\mathcal{U}$ with $\mathcal{D} \subset \mathcal{U}$, the error *outside* $\mathcal{D}$ is typically called the Off-Training-Set (OTS) error

$$E_{\text{ots}}(h) = \frac{1}{|\mathcal{U} \setminus \mathcal{D}|} \sum_{(\mathbf{x},y)\in\mathcal{U}\setminus\mathcal{D}} [\![h(\mathbf{x}) \neq y]\!] .$$

Consider $\mathcal{U}$ with 6 examples

x	y
$(-2, 2)$,	$+1$
$(0, 0)$,	$+1$
$(0, 3)$,	$+1$
$(1, 1)$,	-1
$(2, 1)$,	-1
$(3, 0)$,	-1

Run the process of choosing any three examples from $\mathcal{U}$ as $\mathcal{D}$, and learn a perceptron hypothesis (say, with PLA, or any of your "human learning" algorithm) to achieve $E_{\text{in}}(g) = 0$ on $\mathcal{D}$. Then, evaluate g outside $\mathcal{D}$. What is the smallest and largest $E_{\text{ots}}(g)$? Choose the correct answer; explain your answer.

[a] $(0, \frac{1}{3})$
[b] $(0, 1)$
[c] $(0, \frac{2}{3})$
[d] $(\frac{2}{3}, 1)$
[e] $(\frac{1}{3}, 1)$

D 包含於 U，U\D 代表不包含 D 的 U 也就是 U-D，| U\D | " | |"表示不包含 D 的 U 的個數。把不包含 D 的 U 剩下元素的 **x** 帶入 h(**x**)，將 h(**x**) ≠ y 的個數加總起來，然後最後除以不包含 D 的 U 剩下的總元素的個數，所以：

$$E_{ots}(h) = \frac{\text{不包含 D 的 U 錯的總個數}}{\text{不包含 D 的 U 的個數}} = \text{不包含 D 的 U 的錯誤率}$$

題目要求最小與最大的$E_{ots}(h)$。以下是我用 colab 跑的 python 所得到的答案。

```
[1]  import urllib.request
     import numpy as np
     import random
```

```
if __name__ == '__main__':

    X = [['-2', '2'], ['0', '0'], ['0', '3'], ['1', '1'], ['2', '1'], ['3', '0']]
    Y = ['1', '1', '1', '-1', '-1', '-1',]
    X = np.asarray(X, dtype = float)
    Y = np.asarray(Y, dtype = float)
    Eosts = []
    iteration = 1000
    for iter in range(iteration):
        wpla = []
        points = []
        ranlist = []
        #fix seed
        random.seed(iter)
        while len(ranlist) < 3:
            #random index
            rindex = random.randint(0, 5)
            if rindex not in ranlist:
                ranlist.append(rindex)
        print("ranlist: ",ranlist)
        #find Ein(g) = 0 on D
        pla(ranlist)

        #find g outside D
        remainlist = []
        for i in range(6):
            if i not in ranlist:
                remainlist.append(i)
        print("remainlist: ", remainlist)
        pla(remainlist)

        print("wpla: ", wpla)
        print("points: ", points)
```

一開始照題目給的 U 中的六筆資料給 X、Y。每次 iteration 都會算出一個 Eost 的值，

為了準確一點我跑 1000 次去抓出其中的最大最小值。

#fix seed：讓每次 iteration seed 都不一樣。

#random index：一題意隨機抓出三個不重複的 index 放進 ranlist。

#find Ein(g) = 0 on D：這邊將 ranlist 作為 input 給 function pla(後面會詳細講)，pla

最後會將 wpla 的結果與該次抓的點分別存到 wpla 陣列中與 points 陣列中。

#find g outside D：將前面沒被選到的點也就是去掉 D 的 U 存到 remainlist 中，作為 input 給 function pla，pla 最後會將 wpla 的結果與該次抓的點分別存到 wpla 陣列中與 points 陣列中。

以上跑完印出 wpla 與 points，會得到像是以下圖片的輸出：

```
wpla:  [array([  1., -10., -10.]), array([ -5., -14.,   3.])]
points:  [[array([1., 0., 0.]), array([1., 1., 1.]), array([1., 3., 0.])], [array([ 1., -2.,  2.]), array([1., 0., 3.]), array([1., 2., 1.])]]
```

其中 wpla[0]會是 Ein(g) = 0 on D 的 wpla，wpla[1]是 g outside D 的 wpla。

points[0]會是此次 iteration 當作 D 的三個點，points[1]是不包含 D 的 U 的三個點。

```
        #  Eots(g)
        h  =  wpla[0]
        print("h:  ",  h)
        dout  =  points[1]
        print("dout:  ",  dout)
        y  =  []
        for  i  in  range(len(dout)):
            y.append(np.sign(np.dot(h,  dout[0])))
        print("y:  ",  y)
        errorsum  =  0

        for  i,  j  in  zip(remainlist,  range(len(dout))):
            if  y[j]  !=  Y[i]:
                errorsum  =  errorsum  +  1
        print("errorsum:  ",  errorsum)
        Eost  =  errorsum/len(remainlist)
        print("Eost:  ",  Eost)
        Eosts.append(Eost)
    print(Eosts)
    print("Eost  max:  ",max(Eosts))
    print("Eost  min:  ",min(Eosts))
```

接下來要算出這次的 Eost(g)

#Eost(g)：將 Ein(g) = 0 on D 的 wpla 設為 h，dout 為 points[1]就是不包含 D 的 U 的三個點。for 迴圈計算出每個 dout 中的點內積 h 的值的 sign，將結果存給 y。

errorsum 初始為 0。第二個 for 迴圈，若 y[i]≠Y[i]，就是我裡用 D 算出的權重內積

dout 的 X 去跟正確的 label Y 不一樣，代表發生錯誤，將發生錯誤的個數紀錄在 errorsum。最後按題目公式算出 Eost，將所有 Eost 結果存給 Eosts 陣列。

當跑完 1000 次就會得到 1000 個 Eost 的結果，從中抓出 max 跟 min。

結果如下圖：

```
Eost max:  1.0
Eost min:  0.0
```

```
def  pla(indexlist):
    #  W  =  np.zeros((len(indexlist)))
    W  =  [-10]*len(indexlist)
    point  =  []
    cnt  =  0
    print(len(indexlist))
    while  cnt  <  len(indexlist):
        Xi  =  X[indexlist[cnt]]
        Xi  =  np.concatenate((np.array([1.]),  np.array(Xi)))
        Yi  =  Y[indexlist[cnt]]
        WXi  =  np.sign(np.dot(W,  Xi))
        #sign(0)  =  -1
        if  WXi  ==  0:
            WXi  =  -1
        #mistake  so  update
        if  WXi  !=  np.sign(Yi):
            #W  =  W  +  np.dot(Yi,  Xi)
            print(W)
            W  =  W  +  Yi  *  Xi
            print(W)
            cnt  =  0
            point  =  []
        else:  #right
            cnt  =  cnt  +  1
            point.append(Xi)

    print("cnt  ",  cnt)
    print("point:  ",  point,  '\n')
    wpla.append(W)
    points.append(point)
```

這一段是我 defined 的 pla 的 function，input 參數是 indexlist。這個 function 主要是按講義上的邏輯跑 pla，由於題目沒有設定好 **w**0 與 x0 應該為多少所以我將 x0 設為 1 並將 sign(0)=-1，另外分別用 **w**0=0 與 **w**0=-10 跑了兩次。

兩次有出現不同的結果：

1. **w**0=0

```
def  pla(indexlist):
    #  W  =  np.zeros((len(indexlist)))
    W  =  [0]*len(indexlist)
    point  =  []
```

```
Eost max:  1.0
Eost min:  0.3333333333333333
```

下面是某一次 iteration **w** 更新的狀況，可以看到因為 **w**0=0，讓他第一次更新後的權重就能快速抓到一個點，進而快速抓到第二個點，沒有偏移的限制，可以較好的去抓到對於題目設定的資料較好的 wpla。

Source code on colab：

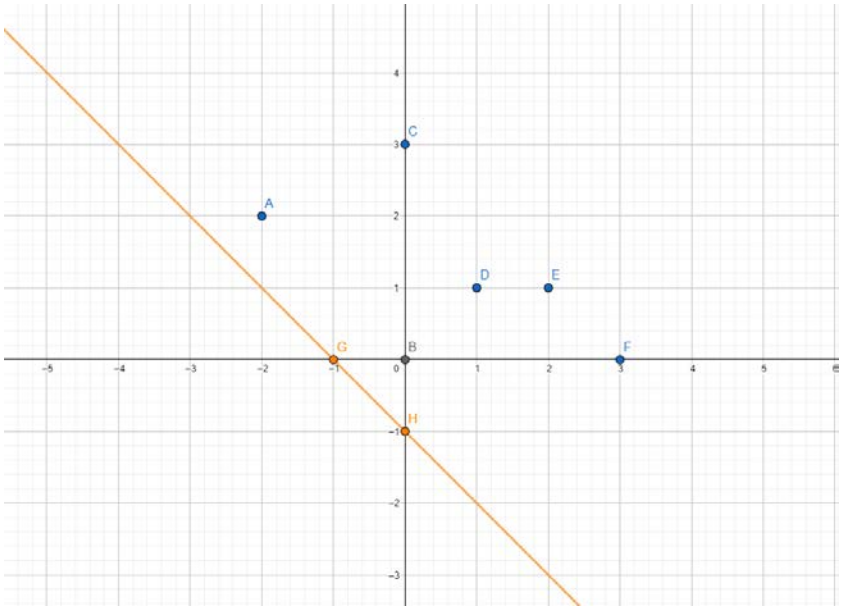
https://colab.research.google.com/drive/1ruBZ9WQOlXA4ECjUcJ-5e2aj4gB6C2L0?usp=sharing

2. **w**0=-10

```
def  pla(indexlist):
    #  W  =  np.zeros((len(indexlist)))
    W  =  [-10]*len(indexlist)
    point  =  []
    cnt  =  0
```

```
Eost max:  1.0
Eost min:  0.0
```

橘色線代表：-10 - 10x1 - 10x2 =0 就是 **w**0=-10 的情況，所有點(A-F)都被分在同一邊。這樣的初始情況比 **w**0=0 的情況還糟糕(會往左偏移)，所以出現的 max Eost 就會出現 1。



Source code on colab：

https://colab.research.google.com/drive/130wZrz6cCr3908yt7xWMvpAHjSeVLZKt?usp=sharing

由於題目中沒有設定 **w**0 為多少，故答案選 b。

Point Estimation

9. ANS[c]

In lecture 3, we try to infer the unknown out-of-sample error $E_{\text{out}}(h)$ by the in-sample error $E_{\text{in}}(h)$ based on the observed data. This is an example of so-called *point estimation*. Formally, a *point estimator* is defined as a function of the data:

$$\hat{\theta} = g(x_1, \ldots, x_N),$$

where $\{x_1, \ldots, x_N\}$ is a set of N examples independent and identically (i.i.d.) generated from a distribution P. That is, $\hat{\theta}$ is also a random variable. Assume that the quantity that we try to infer is θ (determined by the distribution P), we attempt to use $\hat{\theta}$ as a "guess" of θ. For instance, $\theta = E_{\text{out}}(h)$ and $\hat{\theta} = E_{\text{in}}(h)$ in our lecture. One criteria to judge the goodness of $\hat{\theta}$ is whether it is *unbiased*. We call $\hat{\theta}$ unbiased iff

$$\mathbb{E}[\hat{\theta}] = \theta.$$

Consider the following point estimators. Which of them is *not* unbiased? Choose the correct answer; explain your answer.

[a] $\hat{\theta} = \frac{1}{N}\sum_{n=1}^{N} [\![h(\mathbf{x}_n) \neq y_n]\!]$ and $\theta = E_{\text{out}}(h) = \mathbb{E}_{\mathbf{x}\sim P}[\![h(\mathbf{x}) \neq f(\mathbf{x})]\!]$, where h is a fixed hypothesis, $f(\mathbf{x}) = y$ is the target function, and P is some fixed probability distribution that generates the data.

$$E[\hat{\theta}] = E[\frac{1}{N}\sum_{n=1}^{N}[\![h(\mathbf{x}_n) \neq y_n]\!]]$$

由於$f(\mathbf{x})$是完美的，故：

$$= E[\frac{1}{N}\sum_{n=1}^{N}[\![h(\mathbf{x}_n) \neq f(\mathbf{x}_n)]\!]]$$

當 N 趨近於無限大時，根據弱大數法則

弱大數法則 [編輯]

弱大數法則(WLLN) 也稱為辛欽定理，陳述為：樣本均值依機率收斂於期望值。[1]

$\overline{X}_n \xrightarrow{P} \mu \quad \text{as} \quad n \to \infty$

也就是說對於任意正數 ε,

$$\lim_{n\to\infty} P\left(|\overline{X}_n - \mu| > \varepsilon\right) = 0$$

可得：

$$= E[E[\![h(\mathbf{x}_n) \neq f(\mathbf{x}_n)]\!]]$$

期望值取期望值等於取一次期望值，故：

$$= E[\![h(\mathbf{x}_n) \neq f(\mathbf{x}_n)]\!]$$

得證為 unbiased：

$$E[\hat{\theta}] = E[\![h(\mathbf{x}_n) \neq f(\mathbf{x}_n)]\!] = \theta$$

[b] $\hat{\theta} = \frac{1}{N}\sum_{n=1}^{N} x_n$. P is a Bernoulli distribution defined by $P(x) = \theta^x(1-\theta)^{(1-x)}$, where $x \in \{0, 1\}$.

按柏努利分布的定義，θ代表的是 x=1 發生的機率 p。

$$E[\hat{\theta}] = E[\frac{1}{N}\sum_{n=1}^{N} x_n]$$

$$= E[\frac{1}{N}(x_1 + \ldots + x_N)]$$

$$= \frac{1}{N}(E(x_1) + \ldots + E(x_N))$$

$$= \frac{Np}{N} = p = \theta$$

[c] $\hat{\theta} = \max\{x_1, \ldots, x_N\}$. P is a discrete uniform distribution whose probability mass function is defined as $P(x) = \frac{1}{M}$ for $x \in \{1, \ldots, M\}$ $(M > N > 0)$. $\theta = M$.

$$E[\hat{\theta}] = E[max\{x_1 + \ldots + x_N\}]$$

假設$max\{x_1 + \ldots + x_N\}$中得出最大的為$x_n$：

$$= E[x_n]$$

其中x_n來自 P(x)=$\frac{1}{M}$，所以：

$$= \frac{1}{M} * 1 + \frac{1}{M} * 2 + \ldots + \frac{1}{M} * M$$

$$= \frac{\frac{(1+M)M}{2}}{M} = \frac{1+M}{2} \neq \theta = M$$

得證為 not unbiased，選 c。

[d] $\hat{\theta} = \frac{1}{N}\sum_{n=1}^{N} x_n^2$. P is a zero-mean *Gaussian distribution* and θ is its variance.

$$E[\hat{\theta}] = E[\frac{1}{N}\sum_{n=1}^{N} {x_n}^2]$$

由變異數公式：

$$\theta = \sigma^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu)^2$$

而題目中提到是 zero-mean 高斯分布，故$\mu = 0$：

$$E[\hat{\theta}] = E[\frac{1}{N}\sum_{n=1}^{N} {x_n}^2]$$

$$= E[\frac{1}{N}({x_1}^2 + \ldots + {x_N}^2)]$$

$$= \frac{1}{N}(E({x_1}^2) + \ldots + E({x_N}^2))$$

期望值也可以通過變異數計算公式來計算變異數：

$$\mathrm{Var}(X) = \mathrm{E}(X^2) - \mathrm{E}(X)^2$$

由於（平方期望值減的期望值平方）　　且此題的 E(x)就是 mean 為 0，所以：

$$\theta = E({x_n}^2) - E(x_n)^2 = E({x_n}^2)$$

代入前面的式子：

$$E[\hat{\theta}] = \frac{1}{N}(E({x_1}^2) + \ldots + E({x_N}^2))$$

$$= \frac{1}{N}(N * \theta) = \theta$$

得證為 unbiased。

[e] none of the other choices (i.e. all of the other choices are unbiased)

選出 C 了不考慮 e。

Bad Data

10. ANS[a]

Consider $\mathbf{x} = [x_1, x_2]^T \in \mathbb{R}^2$, a target function $f(\mathbf{x}) = \text{sign}(x_1)$, a hypothesis $h_1(\mathbf{x}) = \text{sign}(2x_1 - x_2)$, and another hypothesis $h_2(\mathbf{x}) = \text{sign}(x_2)$. What is $(E_{\text{out}}(h_1), E_{\text{out}}(h_2))$ subject to the uniform distribution in $[-1, +1] \times [-1, +1]$ that generates $\mathbf{x}$? Choose the correct answer; explain your answer.

[a] $(\frac{1}{8}, \frac{1}{2})$
[b] $(\frac{7}{8}, \frac{1}{2})$
[c] $(\frac{1}{2}, \frac{1}{8})$
[d] $(\frac{1}{2}, \frac{7}{8})$
[e] $(\frac{1}{2}, \frac{1}{2})$

$$\mathrm{f}(\mathbf{x}) = \text{sign}(x_1)$$

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

$$h_2(\mathbf{x}) = \text{sign}(x_2)$$

- $E_{out}(h_1)$

$$\mathrm{f}(\mathbf{x}) \neq h_1(\mathbf{x})$$

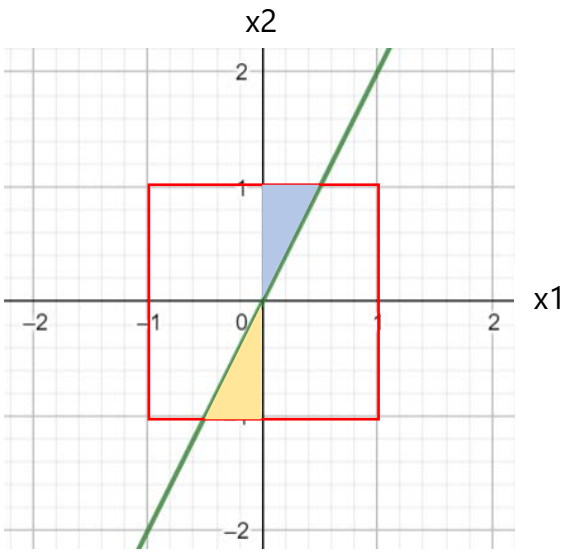
$$\text{sign}(x_1) \neq \text{sign}(2x_1 - x_2)$$

會有下列兩種情況造成錯誤發生：

1. $x_1 > 0 \cdot 2x_1 - x_2 < 0$

2. $x_1 < 0 \cdot 2x_1 - x_2 > 0$

下圖藍色三角形表示可造成 1. 狀況的點出現的範圍，黃色三角形為造成 2. 狀況的點出現的範圍。綠線為$2x_1 - x_2 = 0$，紅框是規定的點產生的範圍。



所以可得：

$$E_{out}(h_1) = \frac{\frac{1}{2} * 1 * \frac{1}{2} * 2}{2 * 2} = \frac{\frac{1}{2}}{4} = \frac{1}{8}$$

- $E_{out}(h_2)$

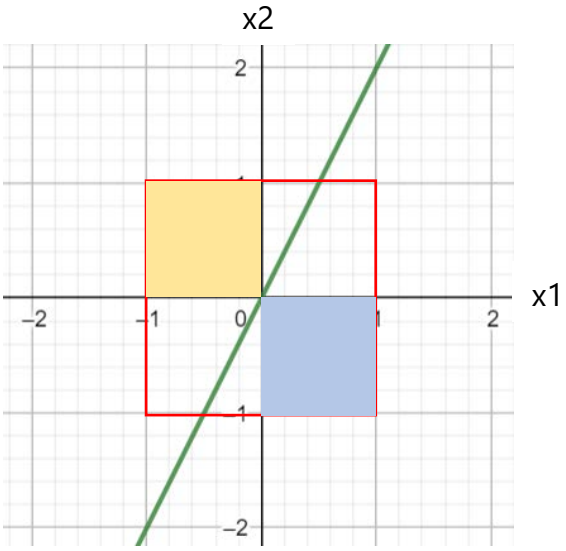
$$\mathrm{f}(\mathbf{x}) \neq h_2(\mathbf{x})$$

$$\mathrm{sign}(x_1) \neq \mathrm{sign}(x_2)$$

會有下列兩種情況造成錯誤發生：

1. $x_1 > 0 \cdot x_2 < 0$
2. $x_1 < 0 \cdot x_2 > 0$

下圖藍色正方形表示可造成 1. 狀況的點出現的範圍，黃色正方形為造成 2. 狀況的點出現的範圍。紅框是規定的點產生的範圍。



所以可得：

$$E_{out}(h_2) = \frac{1 * 1 * 2}{2 * 2} = \frac{2}{4} = \frac{1}{2}$$

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

11. ANS[b]

Following the previous problem, when drawing 4 examples independently and uniformly within $[-1,+1]\times[-1,+1]$ as $\mathcal{D}$, what is the probability that we get 4 examples such that $E_{\text{in}}(h_2) = E_{\text{in}}(h_1)$, including both the zero and non-zero E_{in} cases? Choose the correct answer; explain your answer.

Note: This is one of the BAD-data cases where we cannot distinguish the better-E_{out} hypothesis h_1 from the worse hypothesis h_2.

[a] $\frac{514}{4096}$
[b] $\frac{609}{4096}$
[c] $\frac{784}{4096}$
[d] $\frac{1126}{4096}$
[e] $\frac{1243}{4096}$

因為取四個 example，所以$E_{in}(h_1) = E_{in}(h_2)$可能的情況有五種：

- 都沒有出錯$E_{in}(h_1) = E_{in}(h_2) = 0$

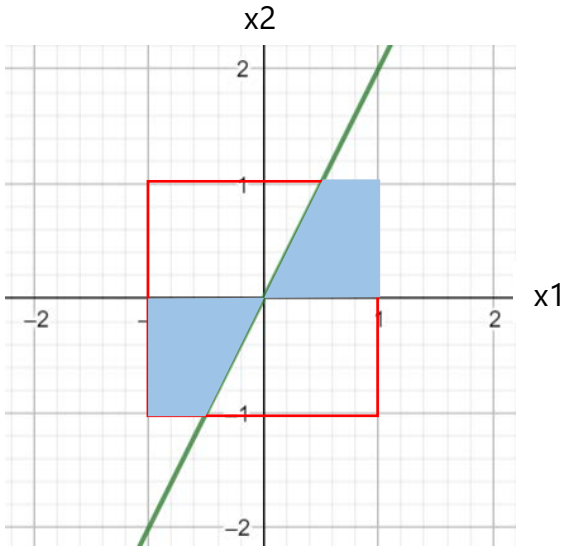
  代表

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

  且

  $$\text{sign}(x_1) = \text{sign}(x_2)$$

  根據第十題的結果可知，都沒出錯的四個 example 產生的範圍如下圖：

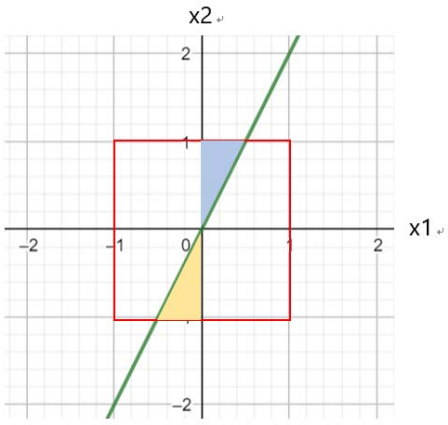


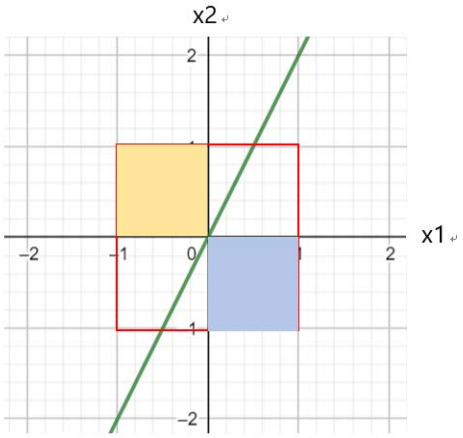
  故取四次都沒錯的機率是：

  $$(\frac{\frac{3}{4} * 2}{4}) = (\frac{3}{8})^4 = \frac{81}{4096}$$

- 都全錯$E_{in}(h_1) = E_{in}(h_2) = 1$

  都全錯代表，抽取某次的 data 同時讓$\mathbf{f(x)} \neq h_1(\mathbf{x})$ (下左圖)與$\mathbf{f(x)} \neq h_2(\mathbf{x})$ (下右圖)成立，根據第十題的結果，可以看到讓他們同時成立的區塊並不存在。所以都全錯的情況發生的機率為 0。





- 都錯一個$E_{in}(h_1) = E_{in}(h_2) = \frac{1}{4}$

  由於某一個 data 同時讓$\mathbf{f(x)} \neq h_1(\mathbf{x})$與$\mathbf{f(x)} \neq h_2(\mathbf{x})$成立的情況不存在。

  我們剩下要考慮的情況是，

  四次中某一次的 data 讓$\mathbf{f(x)} \neq h_1(\mathbf{x})$且$\mathbf{f(x)} = h_2(\mathbf{x})$成立，

  剩下三次的某一次 data 讓$\mathbf{f(x)} = h_1(\mathbf{x})$且$\mathbf{f(x)} \neq h_2(\mathbf{x})$成立，

  其餘次數讓$\mathbf{f(x)} = h_1(\mathbf{x})$且$\mathbf{f(x)} = h_2(\mathbf{x})$成立。

  故取四次都錯一個的機率是：

  (第十題時已知$\mathbf{f(x)} \neq h_1(\mathbf{x})$)發生機率為$\frac{1}{8}$，$\mathbf{f(x)} \neq h_2(\mathbf{x})$發生機率為$\frac{1}{2}$，由都沒有出錯的區段得出$\mathbf{f(x)} = h_1(\mathbf{x})$且$\mathbf{f(x)} = h_2(\mathbf{x})$)的機率為$\frac{3}{8}$。)

$$C_1^4 * \frac{1}{8} * C_1^3 * \frac{1}{2} * (\frac{3}{8})^2 = \frac{432}{4096}$$

- 都錯二個

  由於某一個 data 同時讓$\mathbf{f(x)} \neq h_1(\mathbf{x})$與$\mathbf{f(x)} \neq h_2(\mathbf{x})$成立的情況不存在。

  我們剩下要考慮的情況是，

  四次中某兩次的 data 讓$\mathbf{f(x)} \neq h_1(\mathbf{x})$且$\mathbf{f(x)} = h_2(\mathbf{x})$成立，

剩下兩次的 data 讓$\mathrm{f}(\mathbf{x}) = h_1(\mathbf{x})$且$\mathrm{f}(\mathbf{x}) \neq h_2(\mathbf{x})$成立。

故取四次都錯兩個的機率是：

$$\mathrm{C}_2^4 * (\frac{1}{8})^2 * \mathrm{C}_2^2 * (\frac{1}{2})^2 = \frac{96}{4096}$$

- 都錯三個

  由於某一個 data 同時讓$\mathrm{f}(\mathbf{x}) \neq h_1(\mathbf{x})$與$\mathrm{f}(\mathbf{x}) \neq h_2(\mathbf{x})$成立的情況不存在且只取四次 data，都錯三個至少要取六次 data 才夠，故此題都錯三個的機率為 0。

最後將結果相加得到答案 b：

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

Multiple-Bin Sampling

12. ANS[b]

We illustrate what happens with multiple-bin sampling with an experiment that use a dice (instead of a marble) to bind the six faces together. Please note that the dice is not meant to be thrown for random experiments. The probability below only refers to drawing the dices from the bag. Try to view each number as a *hypothesis*, and each dice as an *example* in our multiple-bin scenario. You can see that no single number is always green—that is, E_{out} of each hypothesis is always non-zero. In the following problem, we are essentially asking you to calculate the probability of the minimum $E_{\text{in}}(h_i) = 0$.

Consider four kinds of dice in a bag, with the same (super large) quantity for each kind.

- A: all even numbers are colored green, all odd numbers are colored orange
- B: $(1, 2, 6)$ are colored green, others are colored orange
- C: the number 6 is colored green, all other numbers are colored orange
- D: all primes are colored green, others are colored orange

If we draw 5 dices independently from the bag, what is the probability that we get *some number* that is purely green? Choose the correct answer; explain your answer.

[a] $\frac{512}{1024}$
[b] $\frac{454}{1024}$
[c] $\frac{333}{1024}$
[d] $\frac{243}{1024}$
[e] $\frac{32}{1024}$

根據題意可製成下表：

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

題目目標是要隨機且獨立的抽出五個骰子，其中有某些數字在這五個骰子上都是綠色的。

由上表可發現要讓數字 1 都是綠色的抽出的五個骰子要是 BBBBB，而當抽出 BBBBB 的時候也會讓數字 2 與 6 都是綠色的，以此類推抽出 5D 時會讓數字 2、3、5 都是綠色，抽出 5A 讓數字 2、4、6 都是綠色，所以我們可以將數字 1、3、4、5 都是綠色的情形不考慮，因為他們發生的情形包含在數字 2、6 都是綠色的情形中。

故將數字 2 與 6 的情況拉出來看：

	2	6
A	Green	Green
B	Green	Green
C	Orange	Green
D	Green	Orange

會發現他們差在 C 與 D 的骰子，只要抽出的五個骰子中，C 與 D 的骰子不同時出現，就會讓至少一個數字都是綠色的。

故讓至少一個數字在五個 i.i.d 下抽出的骰子都是綠色的機率為：

$$1-\left[\text{同時出現 C 與 D 的情形}\right]$$

同時出現 C 與 D 的情形所呈現的 C、D 的型態有十種，與他們發生的機率如下表：(補充說明：像是 2C2D 形態包含 2C2D1A、2C2D1B)

1C1D	1C2D	1C3D	1C4D	2C1D
$C_1^5\frac{1}{4}C_1^4\frac{1}{4}(\frac{2}{4})^3=\frac{160}{1024}$	$C_1^5\frac{1}{4}C_2^4(\frac{1}{4})^2(\frac{2}{4})^2=\frac{120}{1024}$	$C_1^5\frac{1}{4}C_3^4(\frac{1}{4})^3(\frac{2}{4})^1=\frac{40}{1024}$	$C_1^5\frac{1}{4}C_4^4(\frac{1}{4})^4=\frac{5}{1024}$	$C_2^5(\frac{1}{4})^2C_1^3(\frac{1}{4})^1(\frac{2}{4})^2=\frac{120}{1024}$
2C2D	2C3D	3C1D	3C2D	4C1D
$C_2^5(\frac{1}{4})^2C_2^3(\frac{1}{4})^2(\frac{2}{4})^1=\frac{60}{1024}$	$C_2^5(\frac{1}{4})^2C_3^3(\frac{1}{4})^3=\frac{10}{1024}$	$C_3^5(\frac{1}{4})^3C_1^2(\frac{1}{4})^1(\frac{2}{4})^1=\frac{40}{1024}$	$C_3^5(\frac{1}{4})^3C_2^2(\frac{1}{4})^2=\frac{10}{1024}$	$C_4^5(\frac{1}{4})^4(\frac{1}{4})^1=\frac{5}{1024}$

故：

$$\begin{aligned}1-\left[\text{同時出現 C 與 D 的情形}\right]&=1-\frac{160+120+40+5+120+60+10+40+10+5}{1024}\\&=1-\frac{570}{1024}=\frac{454}{1024}\end{aligned}$$

答案為 b。

Experiments with Perceptron Learning Algorithm

13. ANS [b]

Next, we use an artificial data set to study PLA. The data set with $N = 100$ examples is in

`http://www.csie.ntu.edu.tw/~htlin/course/ml21fall/hw1/hw1_train.dat`

Each line of the data set contains one $(\mathbf{x}_n, y_n)$ with $\mathbf{x}_n \in \mathbb{R}^{10}$. The first 10 numbers of the line contains the components of $\mathbf{x}_n$ orderly, the last number is y_n. Please initialize your algorithm with $\mathbf{w} = \mathbf{0}$ and take sign(0) as -1.

13. (*) Please first follow page 29 of lecture 1, and add $x_0 = 1$ to every $\mathbf{x}_n$. Implement a version of PLA that randomly picks an example $(\mathbf{x}_n, y_n)$ in every iteration, and updates $\mathbf{w}_t$ if and only if $\mathbf{w}_t$ is incorrect on the example. Note that the random picking can be simply implemented *with replacement*—that is, the same example can be picked multiple times, even consecutively. Stop updating and return $\mathbf{w}_t$ as $\mathbf{w}_{\text{PLA}}$ if $\mathbf{w}_t$ is correct consecutively after checking $5N$ randomly-picked examples.

Hint: (1) The update procedure described above is equivalent to the procedure of gathering all the incorrect examples first and then randomly picking an example among the incorrect ones. But the description above is usually much easier to implement. (2) The stopping criterion above is a randomized, more efficient implementation of checking whether $\mathbf{w}_t$ makes no mistakes on the data set.

Repeat your experiment for 1000 times, each with a different random seed. What is the average squared length of $\mathbf{w}_{\text{PLA}}$? That is, $\|\mathbf{w}_{\text{PLA}}\|^2$? Choose the closest value.

[a] 360
[b] 390
[c] 420
[d] 450
[e] 480

題目要求：

- 重複跑 1000
- 每次 iteration 中隨機 pick (x,y)
- 若連續 5N 次都沒有錯就 return wt 當作 wpla
- 求平均 Wpla 的 len (平方的平均)

Source Code on Colab：

https://colab.research.google.com/drive/12pSXCIzUA3jpBgtFYFe7IFsWrRD08Axs?usp=sharing

```
[ ]  import urllib.request
     import numpy as np
     import random
```

```
if __name__ == '__main__':
    #get data
    contents = urllib.request.urlopen("http://www.csie.ntu.edu.tw/~htlin/course/ml21fall/hw1/hw1_train.dat").read()
    text = str(contents,'utf-8')
    # data preprocessing
    sptext = text.split('\n')

    X = []
    Y = []

    for i in range(len(sptext)-1):
        data = sptext[i].split('\t')
        # print(data)
        label = data[-1]
        # print(type(data))
        data.pop()
        # print(len(data))
        X += [data]
        Y += [label]

    X = np.asarray(X, dtype = float)
    Y = np.asarray(Y, dtype = float)
    print(X)
    print(Y)
    iteration = 1000
    wpla = [0]*iteration
    #run 1000 times
    for iter in range(iteration):
        pla(iter)

    print(wpla)
    average = np.mean(wpla)
    print(average)
```

#get data：利用 urllib 套件抓網頁上的資料下來。

#data preprocessing：將資料做前處理最後餵給 X 跟 Y。X 到這邊會是一個包含一百個裡面有 10 個元素的一維陣列的二維陣列，Y 會是有一百個元素的一維陣列。

設定 iteration=1000。初始化紀錄每次 iteration 產生 wpla 的內積值的陣列 wpla。開始進 for 迴圈跑 1000 次 function pla。最後算出 wpla 陣列的平均就是答案。**最後跑出的結果是** -383.1132757495113 **，故答案為 b。**

(下一頁有 function pla 的詳細說明)

```
def  pla(iter):
    print("iteration  ",  iter,  '\n')
    #different  random  seed  in  every  iteration
    random.seed(iter)

    W  =  np.zeros((11))
    cnt  =  0

    while  cnt  <  5*X.shape[0]:
        #random  index
        rindex  =  random.randint(0,  99)

        Xi  =  X[rindex]
        Xi  =  np.concatenate((np.array([1.]),  np.array(Xi)))
        Yi  =  Y[rindex]
        WXi  =  np.sign(np.dot(W,  Xi))
        #sign(0)  =  -1
        if  WXi  ==  0:
            WXi  =  -1
        #mistake  so  update
        if  WXi  !=  np.sign(Yi):
            #W  =  W  +  np.dot(Yi,  Xi)
            print(W)
            W  =  W  +  Yi  *  Xi
            print(W)
            cnt  =  0
        else:  #right  add  cnt
            cnt  =  cnt  +  1
    print("cnt  ",cnt,'\n')
    print("wpla  ",np.dot(W,  W))
    wpla[iter]  =  np.dot(W,  W)
```

#different random seed in every iteration：利用傳進來的 iter 來產生 seed。

初始化 W 為有 11 個元素的 nparray，因為要加上 w0 所以 11 個。

cnt 是用來記錄連續對幾筆 x 的變數。

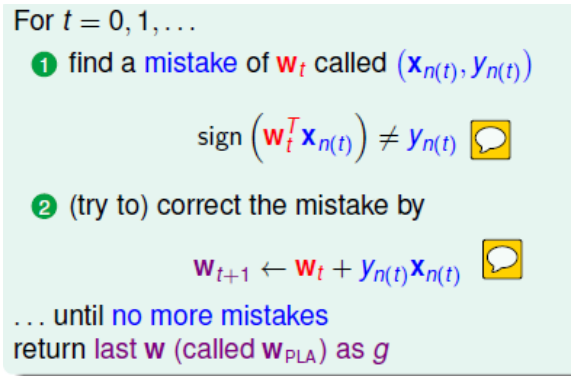
`while cnt < 5*X.shape[0]`：

若連續 5N 次都沒有錯就 return wt 當作 wpla，用 cnt 有沒有超過 5N 這邊會是 500，來當作結束迴圈的條件。

#random index：隨機產生一個範圍在 0-99 的 index，因為只有 100 筆資料。然後抓出隨機的 X 跟 Y 分別給 Xi、Yi。

```
WXi = np.sign(np.dot(W, Xi))
```

這裡後面要拿來判斷所以先將 W 與 Xi 內積且取 sign 好的結果給 WXi。根據 pla 的定義去寫。




```
#sign(0) = -1
if WXi == 0:
   WXi = -1
```

根據題目定義所設定。

```
#mistake so update
if WXi != np.sign(Yi):
  #W = W + np.dot(Yi, Xi)
  # print(W)
  W = W + Yi * Xi
  # print(W)
  cnt = 0
else: #right add cnt
  cnt = cnt + 1
```

根據 pla 定義去寫。若沒有錯就 cnt+=1。

```
wpla[iter] = np.dot(W, W)
```

最後跳出迴圈代表連續成功 500 次，將最後的 W 視為 wpla，這邊我先算出$||W||^2$再存進 wpla 陣列中。

The Learning Problems

14. ANS [c]

(*) Scale up each $\mathbf{x}_n$ by 2, including scaling each x_0 from 1 to 2. Then, run PLA on the scaled examples for 1000 experiments. What is the average squared length of $\mathbf{w}_{\text{PLA}}$? That is, $\|\mathbf{w}_{\text{PLA}}\|^2$? Choose the closest value.

[a] 1260
[b] 1410
[c] 1560
[d] 1710
[e] 1860

根據 13 題的 code，14 題要求將 **x**n 變兩倍大，x0 也從 1 變成 2。

```
def pla(iter):
    print("iteration ", iter, '\n')
    #different random seed in every iteration
    random.seed(iter)

    W = np.zeros((11))
    cnt = 0

    while cnt < 5*X.shape[0]:
        #random index
        rindex = random.randint(0, 99)

        Xi = X[rindex]
        Xi = Xi * 2
        Xi = np.concatenate((np.array([2.]), np.array(Xi)))
        Yi = Y[rindex]
        WXi = np.sign(np.dot(W, Xi))
        #sign(0) = -1
        if WXi == 0:
            WXi = -1
        #mistake so update
        if WXi != np.sign(Yi):
            #W = W + np.dot(Yi, Xi)
            print(W)
            W = W + Yi * Xi
            print(W)
            cnt = 0
        else: #right add cnt
            cnt = cnt + 1
    print("cnt ",cnt,'\n')
    print("wpla ",np.dot(W, W))
    wpla[iter] = np.dot(W, W)
```

其他 code 都跟 13 題一樣，
只差在黃色畫起來的地方。

結果為 1532.453102998045 ，選 c。

Source Code on Colab：

https://colab.research.google.com/drive/1fMK7gWsdzOtHWkR26XZa3lo4wsB7QvRO?usp=sharing

15. ANS [e]

(*) Scale down each $\mathbf{x}_n$ (including x_0) by $\|\mathbf{x}_n\|$, which makes each scaled example of length 1. Then, run PLA on the scaled examples for 1000 experiments. What is the average squared length of $\mathbf{w}_{\text{PLA}}$? That is, $\|\mathbf{w}_{\text{PLA}}\|^2$? Choose the closest value.

[a] 3.1
[b] 4.1
[c] 5.1
[d] 6.1
[e] 7.1

根據 13 題的 code，15 題要求將 **x**n 除以||**x**n||。

```
def pla(iter):
    print("iteration ", iter, '\n')
    #different random seed in every iteration
    random.seed(iter)

    W = np.zeros((11))
    cnt = 0

    while cnt < 5*X.shape[0]:
        #random index
        rindex = random.randint(0, 99)

        Xi = X[rindex]
        scale = np.dot(Xi, Xi) ** 0.5
        Xi = Xi/scale
        Xi = np.concatenate((np.array([1./scale]), np.array(Xi)))
        Yi = Y[rindex]
        WXi = np.sign(np.dot(W, Xi))
        #sign(0) = -1
        if WXi == 0:
            WXi = -1
        #mistake so update
        if WXi != np.sign(Yi):
            #W = W + np.dot(Yi, Xi)
            print(W)
            W = W + Yi * Xi
            print(W)
            cnt = 0
        else: #right add cnt
            cnt = cnt + 1
    print("cnt ",cnt,'\n')
    print("wpla ",np.dot(W, W))
    wpla[iter] = np.dot(W, W)
```

其他 code 都跟 13 題一樣，
只差在黃色畫起來的地方。

結果為 7.2154096020697285 ，選 e。

Source Code on Colab：

https://colab.research.google.com/drive/1SWZiWLUpq_NzB806uiuTpqqO5VXeQzjz?usp=sharing

16. ANS [a]

(*) Set $x_0 = 0$ to every $\mathbf{x}_n$ instead of $x_0 = 1$, and do not do any scaling. This equivalently means not adding any x_0, and you will get a separating hyperplane that passes the origin. Repeat the 1000 experiments above. What is the average squared length of $\mathbf{w}_{\text{PLA}}$? That is, $\|\mathbf{w}_{\text{PLA}}\|^2$? Choose the closest value.

[a] 530
[b] 580
[c] 630
[d] 680
[e] 730

根據 13 題的 code，16 題要求將 x0 設成 0。

```
def pla(iter):
    print("iteration ", iter, '\n')
    #different random seed in every iteration
    random.seed(iter)

    W = np.zeros((11))
    cnt = 0

    while cnt < 5*X.shape[0]:
        #random index
        rindex = random.randint(0, 99)

        Xi = X[rindex]
        Xi = np.concatenate((np.array([0.]), np.array(Xi)))
        Yi = Y[rindex]
        WXi = np.sign(np.dot(W, Xi))
        #sign(0) = -1
        if WXi == 0:
            WXi = -1
        #mistake so update
        if WXi != np.sign(Yi):
            #W = W + np.dot(Yi, Xi)
            print(W)
            W = W + Yi * Xi
            print(W)
            cnt = 0
        else: #right add cnt
            cnt = cnt + 1
    print("cnt ",cnt,'\n')
    print("wpla ",np.dot(W, W))
    wpla[iter] = np.dot(W, W)
```

其他 code 都跟 13 題一樣，
只差在黃色畫起來的地方。

結果為 531.5961884047607 ，選 a。

Source Code on Colab：

https://colab.research.google.com/drive/18OF8nNCDmrThrXB-BBxTCSA09HYSjurd?usp=sharing