Q1.

(d) 收集網站用戶過往的消費紀錄和點擊、流覽的頻率，通常來說越常消費於此網站的人，在往後也比較容易在此網站重複進行消費，~~並推測其可能的~~消費頻率，便可預測其於接下來7天內進行消費的可能性並進行排序。

Q2.

(e) $\cdot\ y_{n(t)} \cdot W_{t+1}^T \cdot x_{n(t)} > 0$, $y_{n(t)} \in \{+1,-1\}$

let $W_{t+1} = W_t + y_{n(t)} x_{n(t)} \cdot \eta_t$

$\cdot\ y_{n(t)} (W_t^T + y_{n(t)} \cdot \eta_t \cdot x_{n(t)}^T) \cdot x_{n(t)} > 0$

$\cdot\ y_{n(t)} \cdot W_t^T x_{n(t)} + y_{n(t)}^2 \cdot \eta(t) \cdot x_{n(t)}^T \cdot x_{n(t)} > 0$

$\because\ y_{n(t)}^2 = (+1)^2 = (-1)^2 = 1$

$\therefore\ \eta(t) \cdot \|x_{n(t)}\|^2 > -y_{n(t)} \cdot W_t^T \cdot x_{n(t)}$

又 $\because\ y_{n(t)} \cdot W_t^T \cdot x_{n(t)} \le 0$

$\therefore\ \eta(t) \cdot \|x_{n(t)}\|^2 > -y_{n(t)} \cdot W_t^T \cdot x_{n(t)} \ge 0$

$\Rightarrow\ \eta(t) > \left(\frac{-y_{n(t)} \cdot W_t^T \cdot x_{n(t)}}{\|x_{n(t)}\|^2} \right) \ge 0$

而 $\left\lfloor \frac{-y_{n(t)} \cdot W_t^T \cdot x_{n(t)}}{\|x_{n(t)}\|^2} + 1 \right\rfloor \ge \frac{-y_{n(t)} \cdot W_t^T \cdot x_{n(t)}}{\|x_{n(t)}\|^2} \ge 0$

Q3.

(E) $\because$ Dataset is linear separable

$\therefore$ Exists perfect W_f such that $y_n = \mathrm{sign}(W_f^T x_n)$

① inner product grows fast.

$$\frac{W_f^T W_{t+1}}{\|W_f\|} \ge \frac{W_f^T W_t}{\|W_f\|} + \underbrace{\min_n \frac{y_n W_f^T x_n}{\|W_f\|}}_{\rho} \cdot \eta_t$$

② length^2 grows slowly

$$\|W_{t+1}\|^2 \le \|W_t\|^2 + \underbrace{\max_n \left(y_n^2 \|x_n\|^2 \right)}_{R^2} \cdot \eta_t^2$$

starts from $W_0 = 0$, after T mistake corrections

$$1 \ge \frac{W_f^T W_T}{\|W_f\| \|W_T\|} \ge \frac{\sum_{t=0}^{T-1} (\eta_t \cdot \rho)}{\sqrt{\sum_{t=0}^{T-1} (\eta_t)^2} \cdot R} \Rightarrow 1^2 \ge \left(\frac{\sum_{t=0}^{T-1} (\eta_t) \cdot \rho}{\sqrt{\sum_{t=0}^{T-1} (\eta_t)^2} \cdot R} \right)^2 = f(T) \cdot \frac{\rho^2}{R^2}$$

而若 $f(T) = \frac{\left(\sum_{t=0}^{T-1} (\eta_t)\right)^2}{\sum_{t=0}^{T-1} (\eta_t)^2}$ 為 monotonically increasing

而 $f(T) \cdot \frac{\rho^2}{R^2} = 1$ 則在 T 夠大時成立，即 Perfect Halt.

(a) $\lim_{T\to\infty} f(T) = \frac{\left(\sum_{t=0}^{\infty} (2^{-t})\right)^2}{\sum_{t=0}^{\infty} (2^{-2t})} = \frac{2}{\frac{4}{3}} = 1.5$

(b) $\lim_{T\to\infty} f(T) = \frac{\left(\sum_{t=0}^{\infty} 0.6211\right)^2}{\sum_{t=0}^{\infty} (0.6211)^2} = \infty$

(d) $\lim_{T\to\infty} f(T) = \frac{\left(\sum_{t=0}^{\infty} \left(\frac{1}{1+t}\right)\right)^2}{\sum_{t=0}^{\infty} \left(\frac{1}{1+t}\right)^2} = \frac{\infty}{\frac{\pi^2}{6}} = \infty$

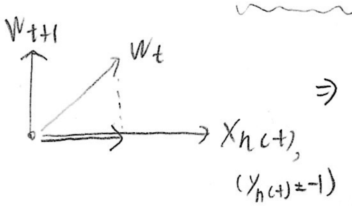
(C)(E)在下一頁

(c)

$$W_{t+1} = W_t + \cancel{y_{n(t)}} X_{n(t)} \cdot \left(\frac{-y_{n(t)} W_t^T X_{n(t)}}{\|X_{n(t)}\|^2} \right)$$

$$= W_t - X_{n(t)} \underbrace{\left(\frac{W_t^T X_{n(t)}}{\|X_{n(t)}\|} \right)}_{}$$

W_t project on $X_{n(t)}$



⇒ 由於 W_{t+1} 永遠是 $X_{n(t)}$ 之法向量，
假若 $\text{sign}(0) = +1$，且 $y_{n(t)} = -1$
而 $X_{n(t+1)} = X_{n(t)}$（即在下一輪中又用
同樣的 X 來更新），此時 $W_{t+1} = W_t$

即 W_t 停止更新，但 $\text{sign}(W_{t+1}^T X_{n(t+1)}) = \text{sign}(0) = 1 \neq -1 = y_{n(t+1)}$
故 (c) 不保證 halt with a perfect line.

(e) $$W_{t+1} = W_t + y_{n(t)} X_{n(t)} \cdot \left[\frac{-y_{n(t)} W_t^T X_{n(t)}}{\|X_{n(t)}\|^2} + 1 \right]$$

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

$$= W_t + y_{n(t)} X_{n(t)} (\alpha + 1)$$，α = 非負整數
$\because -y_{n(t)} W_t^T X_{n(t)} \geq 0$

故 $$\lim_{T \to \infty} f(T) = \frac{\left(\sum_{t=0}^{\infty} (\alpha+1) \right)^2}{\sum_{t=0}^{\infty} (\alpha+1)^2} = \infty$$ ($\because$ $\alpha + 1 \geq 1$) #

故前題之 (b)(d)(e) 之 $f(T)$ 為 monotonically increasing
即有 3 者可 ensure halting with a perfect line.

Q4.

根據題述:

(c) X 為一維度為 $(d+1)\times 1$ 之向量，且

$x_0=1$, $x_{1\sim d}\in\{0,1\}$ 且

同時間 $x_{1\sim d}$ 中最多存在 m 個 1，$m\le d$

另

$$f(x)=\text{sign}(z_+(x)-z_-(x)-0.5)=\text{sign}(w_f^T x)$$

而 w_f^T 為 $\begin{cases} w_{f_0}^T = -0.5 \\ w_{f_{1\sim d}}^T = \{+1,-1\} \end{cases}$，

其中 $w_{f_{1\sim d}}^T$ 中存在 d_+ 個 1 和 d_- 個 -1

此做法是由於 $(z_+(x)-z_-(x))$ 相當於統計算 spam-like word 和 less spam-like 的個數差，因 x 中的值 $\in\{0,1\}$，故 $w_{f_{1\sim d}}=\{+1,-1\}$ 和 $x_{1\sim d}$ 的內積和即等於 $(z_+(x)-z_-(x))$。

而

$$\rho=\min_n\left(\frac{y_n w_f^T x_n}{\|w_f\|}\right)=\frac{0.5}{\sqrt{(0.25+1^2\times d)}}$$

$$R^2=\max_n\left(y_n^2\cdot\|x_n\|^2\right)=1+1^2\times m=1+m$$

$(\because x_{1\sim d}$ 中最多存在 m 個 1$)$

參考 Q3 之說明

故 T 之 upper bound $=\left(\frac{R}{\rho}\right)^2=\frac{R^2}{\rho^2}=\frac{m+1}{\frac{0.25}{0.25+d}}$

$$=4(d+0.25)(m+1)$$

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

#

Q5.

(b) Consider binary PLA

$$w_{t+1}\leftarrow w_t+y_{n(t)}x_{n(t)}$$

考慮兩種 mistake

① 若 $y_{n(t)}=-1\ne\text{sign}(w_t^T x_{n(t)})$，則 $w_{t+1}\leftarrow w_t-x_{n(t)}$

而 Multi-Class PLA 會 $\begin{cases} w_1^{(t+1)}\leftarrow w_1^{(t)}+x_{n(t)} \\ w_2^{(t+1)}\leftarrow w_2^{(t)}-x_{n(t)} \end{cases}$

② 若 $y_{n(t)}=1\ne\text{sign}(w_t^T x_{n(t)})$，則 $w_{t+1}\leftarrow w_t+x_{n(t)}$

而 Multi-Class PLA 會 $\begin{cases} w_1^{(t+1)}\leftarrow w_1^{(t)}-x_{n(t)} \\ w_2^{(t+1)}\leftarrow w_2^{(t)}+x_{n(t)} \end{cases}$

即 $w_T=\sum_{t=0}^{T-1}y_{n(t)}x_{n(t)}$

$w_1^{(T)}=\sum_{t=0}^{T-1}(-y_{n(t)}x_{n(t)})$

$w_2^{(T)}=\sum_{t=0}^{T-1}(y_{n(t)}x_{n(t)})$

故 $w_{PLA}=-w_1^*=w_2^*$

Q6.

(d) (a) Train 的過程並無任何向外界的主動探詢，不符 X

(b) 訓練過程中並無明顯的 feedback of goodness 來調整訓練參數. X

(c) 分類結果不為有限數量類別，不符 X

(e) 訓練之輸入為預先錄製的完整資料集，不符 X

(d) 訓練之產出為自我定義的 label，且可結合後續工程，產生更具物理/實際意義之資料，符合 ✓

Q7.

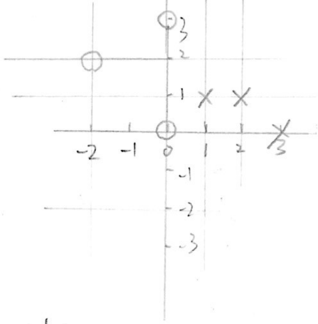
(c) 由於訓練時的 input，總數 (1126556+1126) 筆的資料中，僅有 1126 筆有 target multi-label（事先由專家上好的，label 有總數上限，每一筆 data 有不等數量的 label）而分類目標也為 multi-label
每個Articile 會被分類到複數個tag

故屬於 semi-superised learning 和 multi-label classification。

透過刪去法，僅有 (c) 符合。

Q8.

(b) 採用 Human-Learning：



○ = +1

× = −1

若solo以 (0,3)，(0,0)，(1,1) 三點進行 PLA，則可能得出使兩 label 完美分割的 w_f

例 $w_f^T = [-1, 0, 0.5]$，其中 $w_{f_2}^T$ 是對應常數項

以 (0,3) 為例 $\text{sign}(w_f^T \cdot \begin{bmatrix} 0 \\ 3 \\ 1 \end{bmatrix}) = \text{sign}(0.5) = +1$

而若抽選到都是相同 label（此處以 +1 為例），則可能得到如 $W^T = [1, 0, -3]$，其中 w_2^T 對應常數項

對 +1 label 會全為正確，但對 Out-of-test 的 −1 例而言則是全錯

故 $(\min E_{ots}, \max E_{ots}) = (\frac{1}{3} \times 0, \frac{1}{3} \times 3) = (0, 1)$

Q9.

(c)

(a) $E[\hat{\theta}] = E\left[\frac{1}{N} \sum_{n=1}^{N} [\![h(x_n) \neq y_n]\!]\right]$

$= \frac{1}{N} \sum_{n=1}^{N} E\left([\![h(x_n) \neq y_n]\!]\right) = E\left([\![h(x_n) \neq y_n]\!]\right) = \theta$

$(\because P \text{ is fixed})$

(b) $E[\hat{\theta}] = \frac{1}{N} \sum_{n=1}^{N} E[x_n] = \frac{1}{N} \sum_{n=1}^{N} \left(0 \times (1-\theta) + 1 \times (\theta)\right)$

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

(d) $E[\hat{\theta}] = \frac{1}{N} \sum_{n=1}^{N} E[x_n^2]$ $\quad (\because Var[x] = E[x_n^2] - (E[x_n])^2)$

$= \frac{1}{N} \sum_{n=1}^{n} \left(Var[x] + (E[x_n])^2\right)$

$= \frac{1}{N} \sum_{n=1}^{n} \left(\theta + (0^2)\right)$ $\quad (\because \text{zero-mean})$

$= \theta$

(c) $P(\hat{\theta} \leq x) = P(X_1 \leq x, X_2 \leq x, \ldots X_N \leq x) = \prod_{i=1}^{n} P(X_i \leq x) = \left(\frac{x}{m}\right)^n$

$PDF(x) = P'(\hat{\theta} \leq x) = \frac{n x^{n-1}}{m^n}$

$E(\hat{\theta}) = \frac{1}{m^n} \int_0^m x \cdot n x^{n-1} \, dx$

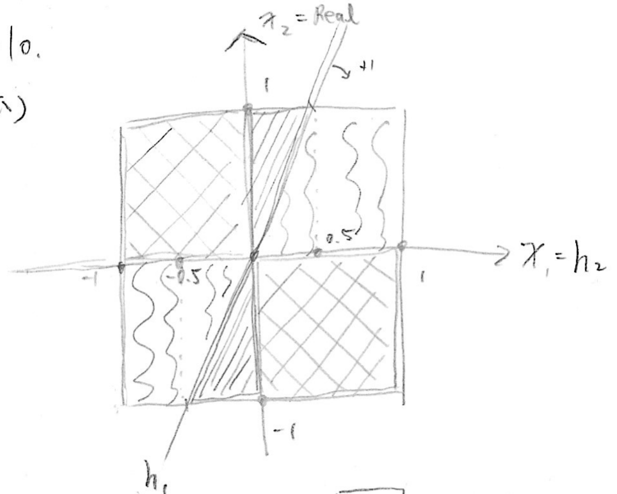
$= \frac{1}{m^n} \int_0^m n x^n \, dx = \frac{1}{m^n} \cdot \frac{n}{n+1} \cdot m^{n+1}$

$= \frac{n}{n+1} \cdot m \neq m = \theta$

#

Q10.

(a)



$\boxed{//}$: $\begin{cases} h_1(x) \neq f(x) \\ h_2(x) = f(x) \end{cases}$

$\boxed{SSS}$: $h_1(x) = h_2(x) = f(x)$

$\boxed{\times\times}$: $\begin{cases} h_1(x) = f(x) \\ h_2(x) \neq f(x) \end{cases}$

$\boxed{//} + \boxed{\times\times} + \boxed{SSS} = [-1,1] \times [-1,1]$

故 $E_{out}(h_1) = \dfrac{\boxed{//}}{2\times2} = \dfrac{(1\times0.5/2)\times2}{2\times2} = \dfrac{1}{8}$

$E_{out}(h_2) = \dfrac{\boxed{\times\times}}{2\times2} = \dfrac{(1\times1)\times2}{2\times2} = \dfrac{1}{2}$

Q11 根據 Q10 圖之結果得

(b)

$h_1(x) = f(x)$	$h_2(x) = f(x)$	
T	T	$\dfrac{\boxed{SSS}}{2\times2} = \dfrac{1.5}{4} = \dfrac{3}{8}$
T	F	$\dfrac{\boxed{\times\times}}{2\times2} = \dfrac{2}{4} = \dfrac{4}{8}$
F	T	$\dfrac{\boxed{//}}{2\times2} = \dfrac{0.5}{4} = \dfrac{1}{8}$
F	F	$\dfrac{0}{2\times2} = 0$

若 $E_{in}(h_2) = E_{in}(h_1)$，則僅有以下三種組合(單組合內順序不拘)

h_1: T T T T, h_2: T T T T ; $E_{in}(h_1) = E_{in}(h_2) = 1$

h_1: T T T F, h_2: F T T T ; $E_{in}(h_1) = E_{in}(h_2) = \frac{3}{4}$

h_1: T T F F, h_2: F F T T ; $E_{in}(h_1) = E_{in}(h_2) = \frac{1}{2}$

故 $\left(\frac{3}{8}\right)^4 \times \frac{4!}{4!} + \frac{4}{8} \times \left(\frac{3}{8}\right)^2 \times \frac{1}{8} \times \frac{4!}{2!} + \left(\frac{4}{8}\right)^2 \times \left(\frac{1}{8}\right)^2 \times \frac{4!}{2!\times2!} = \frac{81}{4096} + \frac{36\times12}{4096} + \frac{16\times6}{4096} = \frac{609}{4096}$ ※

Q12.

(b)

	1	2	3	4	5	6
A		G		G		G
B	G	G				G
C						G
D		G	G		G	

1 types only : $(5) \times (A), (B), (C), (D)$ $\quad \left(\frac{5!}{5!}\right) \times 4 = 4$

2 types : $\begin{matrix}(4,1) \\ (3,2)\end{matrix} \times \begin{matrix}(A,B), (A,C), (A,D), \\ (B,C), (B,D)\end{matrix}$ $\quad \left(\frac{5!}{4!} \times 2 + \frac{5!}{3!2!} \times 2\right) \times 5 = 150$

3 types : $\begin{matrix}(3,1,1) \\ (2,2,1)\end{matrix} \times \begin{matrix}(A,B,D), \\ (A,B,C)\end{matrix}$ $\quad \left(\frac{5!}{3!} \times \frac{3!}{2!} + \frac{5!}{2!2!} \times \frac{3!}{2!}\right) \times 2 = 300$

∴ The probability that we got some number that is purely green

is : $\left(\frac{1}{4}\right)^5 \times (4 + 150 + 300) = \frac{454}{1024}$ ※

In [ ]:
```
import pandas as pd
import numpy as np
import copy
```

In [ ]:
```
data_url="https://www.csie.ntu.edu.tw/~htlin/course/ml21fall/hw1/hw1_train.dat"
col_names = [i for i in range(1,12)]
data = pd.read_csv(data_url, header=None, names=col_names, delimiter='\t')
data
```

Out[ ]:

	1	2	3	4	5	6	7	8	9	10	1
0	1.56186	-2.54905	-1.98638	-0.30684	-1.00062	3.52667	2.62601	-0.30951	1.32496	-2.26376	-1
1	3.89045	-0.02852	2.20650	1.20511	0.12892	1.16363	1.41855	-1.30547	-2.31854	-1.40395	-1
2	-1.85626	-1.38071	-0.07550	-2.90992	-2.75206	-0.23195	-1.10457	-1.11643	-2.35446	-1.43411	1
3	-1.58778	-2.07548	0.00738	-4.24154	0.91851	-5.61822	0.97835	0.06143	4.33760	-1.50129	1
4	2.15052	5.26269	-1.26788	-2.68478	-1.15078	-0.36360	0.74234	1.31526	-2.77029	-0.14857	1
...	...	...	...	...	...	...	...	...	...	...	
95	-3.08999	-3.59724	5.58267	-0.73572	5.16949	3.41667	-0.97299	-0.67884	0.98183	-0.39283	-1
96	-0.17985	1.50163	0.66528	1.89992	0.79647	-1.60727	0.31752	-0.06467	-0.51961	3.71141	-1
97	-3.81450	0.89167	-2.15984	-3.80682	-4.75878	-0.78957	-0.28329	0.45259	-1.57172	0.15997	1
98	-2.11276	-1.91391	-0.63889	-3.53088	-2.24357	-1.22243	0.65278	2.75600	-3.45234	-1.29036	1
99	-2.50787	-1.02966	0.52740	3.15535	-3.28735	1.44250	1.93997	-0.31516	1.14198	0.64107	-1

100 rows × 11 columns

In [ ]:
```
target = data[11]
target
```

Out[ ]:
```
0    -1.0
1    -1.0
2     1.0
3     1.0
4     1.0
     ...
95   -1.0
96   -1.0
97    1.0
98    1.0
99   -1.0
Name: 11, Length: 100, dtype: float64
```

In [ ]:
```
data = data.drop(11, axis=1)
data
```

Out[ ]:

	1	2	3	4	5	6	7	8	9	10
0	1.56186	-2.54905	-1.98638	-0.30684	-1.00062	3.52667	2.62601	-0.30951	1.32496	-2.26376
1	3.89045	-0.02852	2.20650	1.20511	0.12892	1.16363	1.41855	-1.30547	-2.31854	-1.40395

	1	2	3	4	5	6	7	8	9	10
2	-1.85626	-1.38071	-0.07550	-2.90992	-2.75206	-0.23195	-1.10457	-1.11643	-2.35446	-1.43411
3	-1.58778	-2.07548	0.00738	-4.24154	0.91851	-5.61822	0.97835	0.06143	4.33760	-1.50129
4	2.15052	5.26269	-1.26788	-2.68478	-1.15078	-0.36360	0.74234	1.31526	-2.77029	-0.14857
...	...	...	...	...	...	...	...	...	...	...
95	-3.08999	-3.59724	5.58267	-0.73572	5.16949	3.41667	-0.97299	-0.67884	0.98183	-0.39283
96	-0.17985	1.50163	0.66528	1.89992	0.79647	-1.60727	0.31752	-0.06467	-0.51961	3.71141
97	-3.81450	0.89167	-2.15984	-3.80682	-4.75878	-0.78957	-0.28329	0.45259	-1.57172	0.15997
98	-2.11276	-1.91391	-0.63889	-3.53088	-2.24357	-1.22243	0.65278	2.75600	-3.45234	-1.29036
99	-2.50787	-1.02966	0.52740	3.15535	-3.28735	1.44250	1.93997	-0.31516	1.14198	0.64107

100 rows × 10 columns

Q13 ~ 16

PLA with different data preprocessing

In [ ]:
```
def sign(val):
    if (val <=0): return -1.0
    else: return 1.0

def PLA(data, target, random_seed, preprocess):
    data_in = copy.deepcopy(data)
    data_in = preprocess(data_in)

    rng = np.random.RandomState(random_seed)

    N = data_in.shape[0] # Number of Samples
    w = np.zeros(data_in.shape[1]) # Init to zeros

    while(True):
        success=True
        for i in range(5*N):
            idx = rng.randint(0, N)
            tmp = np.dot(w, data_in[idx])
            if (sign(tmp) != target[idx]):
                w = w + target[idx] * data_in[idx]
                success = False
                break
        if (success):
            break

    return w
```

Q13

In [ ]:
```
def preprocess_0(data_in):
    data_in.insert(0, 0, np.ones(100))
    data_in = data_in.to_numpy()
```

```
    return data_in

w_pla_length=[]
for i in range(1000):
    wt = PLA(data, target, i, preprocess_0)
    w_pla_length.append(np.linalg.norm(wt)**2)
print(np.average(w_pla_length))
```

```
388.0550458239121
```

Q14

In [ ]:
```
def preprocess_1(data_in):
    data_in.insert(0, 0, np.ones(100))
    data_in = data_in.to_numpy()
    return data_in * 2

w_pla_length=[]
for i in range(1000):
    wt = PLA(data, target, i, preprocess_1)
    w_pla_length.append(np.linalg.norm(wt)**2)
print(np.average(w_pla_length))
```

```
1552.2201832956484
```

Q15

In [ ]:
```
def preprocess_2(data_in):
    data_in.insert(0, 0, np.ones(100))
    data_in = data_in.to_numpy()
    for i in data_in:
        i /= np.linalg.norm(i)
    return data_in

w_pla_length=[]
for i in range(1000):
    wt = PLA(data, target, i, preprocess_2)
    w_pla_length.append(np.linalg.norm(wt)**2)
print(np.average(w_pla_length))
```

```
7.0702934116262535
```

Q16

In [ ]:
```
def preprocess_3(data_in):
    data_in.insert(0, 0, np.zeros(100))
    data_in = data_in.to_numpy()
    return data_in

w_pla_length=[]
for i in range(1000):
    wt = PLA(data, target, i, preprocess_3)
    w_pla_length.append(np.linalg.norm(wt)**2)
print(np.average(w_pla_length))
```

```
541.4407857585554
```