# Machine Learning Homework 3

資工碩一 r09922055 陳柏妤

1. **(b)**

   For $\sigma = 0.1$ and $d = 11$，

   want $\mathbb{E}_D[E_{in}(w_{lin})] = \sigma^2(1 - \dfrac{d+1}{N}) = 0.01(1 - \dfrac{12}{N}) \geq 0.006$

   $\Rightarrow \dfrac{12}{N} \leq 0.4 \Rightarrow N \geq 30$

2. **(a)**

   如果 $X^T Xw = X^T y$ 有解，只代表此時 $E_{in}(w)$ 在最小值（$\nabla E_{in} = 0$）
   ，不代表 $E_{in}(w) = 0 \Rightarrow$ (b)(c) 錯

   ・如果 $X^T X$ invertible $\Rightarrow w_{lin} = (X^T X)^{-1} X^T y$
   ・如果 $X^T X$ singular $\Rightarrow$ 很多組解，其中一組是 $w_{lin} = X^{\dagger} y$
   因此，$E_{in}(w) = 0$ 可能有 $\geq 1$ 組解，不一定是唯一解 $\Rightarrow$ (d) 錯、(a) 對

3. **(c)**

設 $X$ 有 $N$ 個 data point、$x_n$ 有 $d+1$ 維

$H$：把 $y$ 投影到 $X$ 的 column space

- 對 $X$ 做 column operation 不會改變 column space

  column operation：$X' = XA$

  $$\begin{aligned} H' &= X'((X')^TX')^{-1}(X')^T = XA[(XA)^T(XA)]^{-1}(XA)^T \\ &= XA[A^T(X^TX)A]^{-1}(A^TX^T) = X\cancel{A}\cancel{A}^{-1}(X^TX)^{-1}(\cancel{A}^T)^{-1}\cancel{A}^TX^T \\ &= X(X^TX)^{-1}X^T \\ &= H \end{aligned}$$

- 對 $X$ 做 row operation 會改變 column space

  row operation：$X' = AX$

  $$\begin{aligned} H' &= X'((X')^TX')^{-1}(X')^T = AX[(AX)^T(AX)]^{-1}(AX)^T \\ &= AX[X^TA^TAX]^{-1}(X^TA^T) \\ &\neq H \end{aligned}$$

(a) column operation（$X' = XA$），其中 $A = \begin{bmatrix} 2 & 0 & \dots & 0 \\ 0 & 2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 2 \end{bmatrix}_{(d+1)\times(d+1)}$

(b) column operation（$X' = XA$），其中 $A = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & d \end{bmatrix}_{(d+1)\times(d+1)}$

(c) row operation（$X' = AX$），其中 $A = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & \frac{1}{2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \frac{1}{N} \end{bmatrix}_{N\times N}$

(d) column operation（$X' = XA$），其中 $A$ 為 $(d+1)\times(d+1)$ 的 $I$ 矩陣，且 row i, j, k 的第一個 column = 1（即：$A_{i,1} = A_{j,1} = A_{k,1} = 1$），因為 $A$ 是 linear independent 所以有反矩陣。

4. **(e)**

(a) by Hoeffding's inequality，壞事發生的機率 $P[|\nu-\theta|>\epsilon] \le 2exp(-2\epsilon^2 N)$

(b) $likelihood(\hat{\theta}) = (\hat{\theta})^h \cdot (1-\hat{\theta})^{N-h}$ （$h$ 是 $y_n = 1$ 的個數）

$log\ likelihood(\hat{\theta}) = h \cdot ln(\hat{\theta}) + (N-h) \cdot ln(1-\hat{\theta})$

為了 maximize log likelihood，對 $\hat{\theta}$ 微分算梯度等於零的點

$$\Rightarrow h \cdot \frac{1}{\hat{\theta}} - (N-h) \cdot \frac{1}{1-\hat{\theta}} = 0$$

$\Rightarrow$ 此時 $\hat{\theta} = \frac{h}{N} = \frac{1}{N}\sum_{n=1}^{N} y_n = \nu$ （$\nu$ maximize $likelihood(\hat{\theta})$ over all $\hat{\theta} \in [0,1]$）

(c) $\nabla E_{in}(\hat{y}) = \frac{1}{N}\sum_{n=1}^{N}(2\hat{y} - 2y_n) = 2(\frac{1}{N}\sum_{n=1}^{N}\hat{y} - \frac{1}{N}\sum_{n=1}^{N}y_n) = 2(\hat{y} - \nu)$

$\nabla E_{in}(\hat{y}) = 0$ 時，$\nu = \hat{y} \Rightarrow \nu$ minimize $E_{in}(\hat{y}) = \frac{1}{N}\sum_{n=1}^{N}(\hat{y} - y_n)^2$ over all $\hat{y} \in \mathbb{R}$

(d) by (c)，$\nabla E_{in}(\hat{y}) = 2(\hat{y} - \nu) \Rightarrow$ 當 $\hat{y} = 0$，$2\nu = -\nabla E_{in}(\hat{y})$

5. **(a)** 參考：https://ocw.mit.edu/courses/mathematics/18-443-statistics-for-applications-fall-2006/lecture-notes/lecture2.pdf

Uniform distribution on the interval $[0,\ \theta]$ has p.d.f：$P(y_n|\theta) = \begin{cases} \frac{1}{\theta},\ 0 \le y_n \le \theta \\ 0,\ otherwise \end{cases}$

$$likelihood(\hat{\theta}) = \prod_{n=1}^{N} P(y_n|\hat{\theta}) = (\frac{1}{\hat{\theta}})^N$$

6. **(b)**

(a) $\nabla err(w,x,y) = \begin{cases} -yx,\ when\ 1 - yw^Tx > 0 \\ yx,\ when\ 1 - yw^Tx < 0 \end{cases}$

(b) $\nabla err(w,x,y) = \begin{cases} -yx,\ when\ \ -yw^Tx > 0 \\ 0,\ when\ \ -yw^Tx < 0 \end{cases}$

(c) $\nabla err(w,x,y) = -yx$

(d) $\nabla err(w,x,y) = \begin{cases} 0,\ when\ \ -yw^Tx > 0 \\ -yx,\ when\ \ -yw^Tx < 0 \end{cases}$

(e) $\nabla err(w,x,y) = \begin{cases} -yx,\ when\ 1 - yw^Tx > 0 \\ 0,\ when\ 1 - yw^Tx < 0 \end{cases}$

fixed learning rate gradient descent：$w_{t+1} \leftarrow w_t - \eta \nabla E_{in}(w_t)$

by 題目 $w_{t+1} \leftarrow w_t - \eta \frac{1}{N} \sum_{n:y_n \ne sign(w_t^T x_n)} -y_n x_n$

因為題目的 Ein 只計算 $y_n \ne sign(w_t^T x_n)$ 的點（即：$-y_n w_t^T x_n > 0$ 的點），其餘的點不計算

$\Rightarrow$ 答案為 (b)

7. **(a)**

$$err_{exp}(w, x_n, y_n) = exp(-y_n w^T x_n)$$

$$-\nabla err_{exp}(w, x_n, y_n) = -exp(-y_n w^T x_n) \cdot (-y_n x_n) = (y_n x_n) exp(-y_n w^T x_n)$$

8. **(b)**

by Taylor's expansion $E(w) = E(u) + b_E(u)^T(w-u) + \frac{1}{2}(w-u)^T A_E(u)(w-u)$

$$\Rightarrow \frac{\partial E(w)}{\partial w} = b_E(u) + A_E(u)w - A_E(u)u = 0$$ （二次式的低點）

$$\Rightarrow A_E(u)(w-u) = -b_E(u)$$

$$\Rightarrow w = u + (-A_E(u)^{-1} b_E(u))$$

$$\Rightarrow v = -A_E(u)^{-1} b_E(u)$$

9. **(b)**

by linear regression，$E_{in}(w_t) = \frac{1}{N}(w_t^T X^T X w_t - 2 w_t^T X^T y + y^T y)$

$$\Rightarrow \nabla E_{in}(w_t) = \frac{2}{N}(X^T X w_t - X^T y)$$

$$\Rightarrow A_E(w_t) = \frac{\partial \nabla E_{in}(w_t)}{\partial w_t} = \frac{2}{N}(X^T X)$$

10. **(b)**

・整理 error function

$$-\sum_{k=1}^{K} [y=k] ln h_k(x) = \sum_{k=1}^{K} [y=k](ln(\sum_{i=1}^{K} exp(w_i^T x)) - ln(exp(w_k^T x)))$$

$$= ln(\sum_{i=1}^{K} exp(w_i^T x)) - \sum_{k=1}^{K} [y=k](ln(exp(w_k^T x)))$$ （前項與 $k$ 無關，能移到 $\sum_{k=1}^{K}$ 左邊）

・error function 微分

$$\frac{\partial[ln(\sum_{i=1}^{K} exp(w_i^T x)) - \sum_{k=1}^{K} [y=k](ln(exp(w_k^T x)))]}{\partial W_{ik}}$$

$$= (\frac{1}{\sum_{i=1}^{K} exp(w_i^T x)} \cdot exp(w_k^T x) \cdot x_i) - [y=k](\frac{1}{exp(w_k^T x)} \cdot exp(w_k^T x) \cdot x_i)$$

$$= (h_k(x) - [y=k]) x_i$$

11. **(e)**

· case1

$$P(y=1|x_n) = \frac{exp(w_1^{*T}x_n)}{exp(w_1^{*T}x_n)+exp(w_2^{*T}x_n)} = \frac{1}{1+exp(w_2^{*T}x_n - w_1^{*T}x_n)}$$

$$= P(y'=-1|x_n) = 1 - \frac{1}{1+exp(-w^Tx_n)} = \frac{1}{1+exp(w^Tx_n)}$$

· case2

$$P(y=2|x_n) = \frac{exp(w_2^{*T}x_n)}{exp(w_1^{*T}x_n)+exp(w_2^{*T}x_n)} = \frac{1}{1+exp(w_1^{*T}x_n - w_2^{*T}x_n)}$$

$$= P(y'=1|x_n) = \frac{1}{1+exp(-w^Tx_n)}$$

by case1 & case2：

$$\frac{1}{1+exp(w_2^{*T}x_n - w_1^{*T}x_n)} = \frac{1}{1+exp(w^Tx_n)}$$

$$\frac{1}{1+exp(w_1^{*T}x_n - w_2^{*T}x_n)} = \frac{1}{1+exp(-w^Tx_n)}$$

$$\Rightarrow w^Tx_n = w_2^{*T}x_n - w_1^{*T}x_n$$

$$\Rightarrow w = w_2^* - w_1^*$$

12. **(e)**

$\phi_2(x) = (1, x_1, x_2, x_1^2, x_1x_2, x_2^2)$

$\phi_2(x_1) = (1,\ 0,\ 1,\ 0,\ 0,\ 1),\ y_1 = -1$

$\phi_2(x_2) = (1,\ 1,\ -0.5,\ 1,\ -0.5,\ 0.25),\ y_2 = -1$

$\phi_2(x_3) = (1,\ -1,\ 0,\ 1,\ 0,\ 0),\ y_3 = -1$

$\phi_2(x_4) = (1,\ -1,\ 2,\ 1,\ -2,\ 4),\ y_4 = +1$

$\phi_2(x_5) = (1,\ 2,\ 0,\ 4,\ 0,\ 0),\ y_5 = +1$

$\phi_2(x_6) = (1,\ 1,\ -1.5,\ 1,\ -1.5,\ 2.25),\ y_6 = +1$

$\phi_2(x_7) = (1,\ 0,\ -2,\ 0,\ 0,\ 4),\ y_7 = +1$

13. **(b)**

已知 d 維的 growth function $\leq d(N-1)\cdot 2+2$（by 作業二）

當 $N \leq d_{vc}$ 時，$2^N = m_H(N) \leq d(N-1)\cdot 2+2$（by VC dimension）

$2^N \leq d(N-1)\cdot 2+2 \Rightarrow 2^N \leq d(N-1)\cdot 2+2d \Rightarrow 2^N \leq 2Nd$

$\Rightarrow N \leq log_2 2 + log_2 N + log_2 d = 1 + log_2 N + log_2 d$

⋆補充：當 $d \geq 4$ 時，

(a) $2((log_2 log_2 d)+1) \geq 4$

(b) $2((log_2 d)+1) \geq 6$

(c) $2((d log_2 d)+1) \geq 18$

(d) $2(d+1) \geq 10$

(e) $2(d^2+1) \geq 34$

by 上述的範圍，$N<4$ 的情況必會被 shatter，只需考慮 $N \geq 4$ 的情況，因此 $N$ 可以套用

$log_2 d \leq \frac{d}{2}\ for\ d \geq 4$ 的式子

$\Rightarrow N \leq 1 + log_2 N + log_2 d \leq 1 + \frac{N}{2} + log_2 d$

$\Rightarrow N \leq 2(1 + log_2 d)$

14. **(d)** 0.60
15. **(c)** 1800
16. **(c)** 0.56
17. **(b)** 0.50
18. **(a)** 0.32
19. **(b)** 0.36
20. **(d)** 0.44

In [1]:
```
import numpy as np
```

In [2]:
```
def read_file(filename):
    X = []
    Y = []
    for lines in open(filename).readlines():
        temp = lines.strip().split()
        x = [1]
        for i in range(10):
            x.append(float(temp[i]))
        X.append(x)
        Y.append(float(temp[-1]))
    X = np.asarray(X)
    Y = np.asarray(Y)
    return X, Y

train_data = "hw3_train.dat.txt"
test_data = "hw3_test.dat.txt"
X, Y = read_file(train_data)
X_test, Y_test = read_file(test_data)

def zero_one_error(data_num, W, X, Y):
    zero_one_error = 0
    for n in range(data_num):
        score = np.dot(W, X[n])
        if np.sign(score) != np.sign(Y[n]):
            zero_one_error += 1
    zero_one_error /= data_num
    return zero_one_error

def transform(data_num, Q, X):
    Q_X = []
    for n in range(data_num):
        temp = [1]
        for q in range(1, Q+1):
            for i in range(1, len(X[n])):
                temp.append(X[n][i] ** q)
        Q_X.append(temp)
    return Q_X
```

In [3]:
```
## 14
W_lin = np.dot(np.dot(np.linalg.inv(np.dot(X.T, X)), X.T), Y)
data_num = len(X)
sqr_error = 0
for i in range(data_num):
    score = np.dot(W_lin, X[i])
    sqr_error += (score-Y[i])**2
sqr_error /= data_num
print("14.sqr_error: ", sqr_error)
```

```
14.sqr_error:  0.6053223804672916
```

In [4]:
```
## 15
eta = 0.001
threshold = sqr_error*1.01
total_num_of_iter = 0
for T in range(1000):
    np.random.seed(T)
    num_of_iter = 0
    w = np.zeros([11], dtype=float)
    while(1):
        num_of_iter += 1
        n = np.random.randint(1000)
        v = 2*(Y[n]-np.dot(w, X[n])) * X[n]
        w += (eta * v)
        E_in = 0
        score = np.dot(X, w)
        E_in = np.mean((score-Y)**2)
        if(E_in <= threshold):
            break
    print(T, ":", num_of_iter, end='\r')
    total_num_of_iter += num_of_iter
print("15.aver_iteration: ", total_num_of_iter/1000)
```

```
15.aver_iteration:  1852.502
```

In [5]:
```
## 16
eta = 0.001
aver_ce_error = 0
for T in range(1000):
    print(T, end='\r')
    np.random.seed(T)
    picked_points = np.random.randint(1000, size=500)
    w = np.zeros([11], dtype=float)
    for i in range(500):
        n = picked_points[i]
        s = -Y[n] * (np.dot(w.T, X[n]))
        w += eta * (1/(1+np.exp(-s))) * (Y[n]*X[n])

    ce_error = 0
    for n in range(data_num):
        s = -Y[n] * (np.dot(w.T, X[n]))
        ce_error += np.log(1+np.exp(s))
    ce_error /= data_num
    aver_ce_error += ce_error
aver_ce_error /= 1000
print("16.aver_ce_error: ", aver_ce_error)
```

```
16.aver_ce_error:  0.5691125350817864
```

In [6]:
```
## 17
eta = 0.001
aver_ce_error = 0
for T in range(1000):
    print(T, end='\r')
    np.random.seed(T)
    picked_points = np.random.randint(1000, size=500)
    w = np.copy(W_lin)
    for i in range(500):
        n = picked_points[i]
        s = -Y[n] * (np.dot(w.T, X[n]))
        w += eta * (1/(1+np.exp(-s))) * (Y[n]*X[n])

    ce_error = 0
    for n in range(data_num):
        s = -Y[n] * (np.dot(w.T, X[n]))
        ce_error += np.log(1+np.exp(s))
    ce_error /= data_num
    aver_ce_error += ce_error
aver_ce_error /= 1000
print("17.aver_ce_error: ", aver_ce_error)
```

```
17.aver_ce_error:  0.5028521605674319
```

In [7]:
```
## 18
test_data_num = len(X_test)

train_error = zero_one_error(data_num, W_lin, X, Y)
test_error = zero_one_error(test_data_num, W_lin, X_test, Y_test)

print("18.Ein_Eout:", abs(train_error-test_error))
```

```
18.Ein_Eout: 0.3226666666666666
```

In [8]:
```
## 19
Q = 3
Q_X = transform(data_num, Q, X)
Q_X_test = transform(test_data_num, Q, X_test)
Q_X = np.asarray(Q_X)
Q_X_test = np.asarray(Q_X_test)

Q_W_lin = np.dot(np.dot(np.linalg.inv(np.dot(Q_X.T, Q_X)), Q_X.T),
Y)

train_error = zero_one_error(data_num, Q_W_lin, Q_X, Y)
test_error = zero_one_error(test_data_num, Q_W_lin, Q_X_test, Y_tes
t)

print("19.Ein_Eout:", abs(train_error-test_error))
```

```
19.Ein_Eout: 0.37366666666666665
```

In [9]:
```
## 20
Q = 10
Q_X = transform(data_num, Q, X)
Q_X_test = transform(test_data_num, Q, X_test)
Q_X = np.asarray(Q_X)
Q_X_test = np.asarray(Q_X_test)

Q_W_lin = np.dot(np.dot(np.linalg.inv(np.dot(Q_X.T, Q_X)), Q_X.T),
Y)

train_error = zero_one_error(data_num, Q_W_lin, Q_X, Y)
test_error = zero_one_error(test_data_num, Q_W_lin, Q_X_test, Y_tes
t)

print("20.Ein_Eout:", abs(train_error-test_error))
```

```
20.Ein_Eout: 0.44666666666666666
```

In [ ]: