1 $\quad \sigma = 0.1 \quad d = 19$

(d) $E_D[E_{in}(W_{lin})] = 0.01\left(1 - \frac{20}{N}\right)$

≥ 0.005

$\Rightarrow 1 - \frac{20}{N} \geq 0.5$

$N \geq 40$

25

30

35

40

45

2 $f(x) = x^2$ $h(x) = w_0 + w_1 x$

$$E_{sqr} \text{ in } [0,1] = \int_0^1 (x^2 - w_1 x - w_0)^2 \, dx$$

(c)

$$= \int_0^1 (x^4 + w_1^2 x^2 + w_0^2 - 2w_1 x^3 - 2w_0 x^2 + 2w_1 w_0 x) \, dx$$

$$= \frac{x^5}{5} + \frac{w_1^2 x^3}{3} + w_0^2 x - \frac{w_1 x^4}{2} - \frac{2w_0 x^3}{3} + w_1 w_0 x^2 \Big|_0^1$$

$$= \frac{1}{5} + \frac{w_1^2}{3} + w_0^2 - \frac{w_1}{2} - \frac{2w_0}{3} + w_1 w_0$$

We should min E_{sqr}

$$\min \frac{1}{5} + \frac{w_1^2}{3} + w_0^2 - \frac{w_1}{2} - \frac{2w_0}{3} + w_1 w_0 = f(w_0, w_1)$$

2-variable covex function + 開口向上

$\Rightarrow$ 微分 $= 0$ 是最低點

$$\begin{cases} \dfrac{\nabla f(w_0, w_1)}{\nabla w_0} = 2w_0 - \dfrac{2}{3} + w_1 = 0 \\ \dfrac{\nabla f(w_0, w_1)}{\nabla w_1} = \dfrac{2}{3} w_1 - \dfrac{1}{2} + w_0 = 0 \end{cases}$$

$w_0 = -\frac{1}{6}$ $w_1 = 1$

3

(e)

$f(x) = x^2 \qquad h(x) = w_0 + w_1 x$

交點 $\Rightarrow \quad x^2 - w_1 x - w_0 = 0$

$$x = \frac{w_1 \pm \sqrt{w_1^2 + 4w_0}}{2}$$

$$x_1 = \frac{w_1 - \sqrt{w_1^2 + 4w_0}}{2} \qquad x_2 = \frac{w_1 + \sqrt{w_1^2 + 4w_0}}{2}$$

$$w_1 = x_1 + x_2 \qquad w_0 = -x_1 x_2$$

$$f(w_0, w_1) = \frac{1}{5} + \frac{w_1^2}{3} + w_0^2 - \frac{w_1}{2} - \frac{2w_0}{3} + w_1 w_0$$

$$= \frac{1}{5} + \frac{(x_1+x_2)^2}{3} + x_1^2 x_2^2 - \frac{(x_1+x_2)}{2} + \frac{2}{3} x_1 x_2 - (x_1+x_2) x_1 x_2$$

$$= \frac{1}{5} + x_1^2 x_2^2 - x_1^2 x_2 - x_2^2 x_1 + \frac{x_1^2}{3} + \frac{x_2^2}{3} + \frac{4}{3} x_1 x_2 - \frac{x_1}{2} - \frac{x_2}{2}$$

D 只有2點 所以 $E_{in} = 0$ (2點決定一直線)

E_{out} 對 x_1、x_2 的範圍積分求平均 E_{out}

$$\int_0^1 \int_0^{x_2} \left(\frac{1}{5} + x_1^2 x_2^2 - x_1^2 x_2 - x_2^2 x_1 + \frac{x_1^2}{3} + \frac{x_2^2}{3} + \frac{4}{3} x_1 x_2 - \frac{x_1}{2} - \frac{x_2}{2} \right) dx_1\, dx_2$$

$$= \int_0^1 \left[\frac{x_1}{5} + \frac{x_2^2 x_1^3}{3} - \frac{x_2 x_1^3}{3} - \frac{x_2^2 x_1^2}{2} + \frac{x_1^3}{9} + \frac{x_2^2}{3} x_1 + \frac{2}{3} x_2 x_1^2 - \frac{x_1^2}{4} - \frac{x_2}{2} x_1 \right) \Big|_0^{x_2} dx_2$$

$$= \int_0^1 \left(\frac{x_2^5}{3} - \frac{5}{6} x_2^4 + \frac{10}{9} x_2^3 - \frac{3}{4} x_2^2 + \frac{x_2}{5} \right) dx_2$$

$$= \frac{x_2^6}{18} - \frac{x_2^5}{6} + \frac{5}{18} x_2^4 - \frac{x_2^3}{4} + \frac{x_2^2}{10} \Big|_0^1$$

$$= \frac{1}{18} - \frac{1}{6} + \frac{5}{18} - \frac{1}{4} + \frac{1}{10} = \frac{1}{60}$$

$$\int_0^1 \int_0^{x^2} 1 \, dx_1\, dx_2 = \frac{1}{2}$$

$$\frac{\frac{1}{60}}{\frac{1}{2}} = \frac{1}{30}$$

$$\left| 0 - \frac{1}{30} \right| = \frac{1}{30}$$

4 $y \in \{-1,+1\}$ $y' \in \{0,1\}$ $y'_n = \frac{y_n+1}{2}$

(a)

$$E_{in}(w) = \frac{1}{N}\sum_{n=1}^{N} -\ln\theta(y_n w^T x_n)$$

對每一個 data

if $y'_i = 1$

$-\ln\theta(w^T x_i)$ $w^T x_i$ 越大越好

if $y'_i = 0$

$-\ln\theta(-w^T x_i)$ $-w^T x_i$ 越大越好

⇓ 等價

$$\max\ y'_i\left(\ln\theta(w^T x_i)\right) + (1-y')\left(\ln\theta(-w^T x_i)\right)$$

5 $\quad \nu$: in-sample $\quad \mu$: out-sample

(e) (1) By hoeffding's inequality

$$P(|\nu-\mu|>\epsilon) \le \underbrace{2\exp(-2\epsilon^2 N)}_{\delta}$$

$$\delta = 2\exp(-2\epsilon^2 N)$$

$$\Rightarrow \epsilon = \sqrt{\frac{1}{2N}\ln\frac{2}{\delta}}$$

$$\Rightarrow P(|\nu-\mu| \le \epsilon) > 1-\delta$$

$$|\nu-\mu| \le \sqrt{\frac{1}{2N}\ln\frac{2}{\delta}}$$

$$\Rightarrow -\sqrt{\frac{1}{2N}\ln\frac{2}{\delta}} \le \nu-\mu \le \sqrt{\frac{1}{2N}\ln\frac{2}{\delta}}$$

$$\Rightarrow -\sqrt{\frac{1}{2N}\ln\frac{2}{\delta}}+\mu \le \nu \le \mu+\sqrt{\frac{1}{2N}\ln\frac{2}{\delta}}$$

True

(2) $$E[\nu] = \sum_{i=0}^{N}\binom{N}{i}\mu^i(1-\mu)^{N-i} = \mu$$

True

(3) $N\nu$ head $\quad N(1-\nu)$ unhead

$$E^{sqr}(\hat{y}) = \frac{1}{N}\left(N\nu(1-\hat{y})^2 + N(1-\nu)(\hat{y}-0)^2\right)$$

$$= \nu - 2\hat{y}\nu + \nu\hat{y}^2 + \hat{y}^2 - \nu\hat{y}^2$$

$$= \hat{y}^2 - 2\hat{y}\nu + \nu$$

$$\min \hat{y}^2 - 2\hat{y}\nu + \nu = \min \hat{y}^2 - 2\hat{y}\nu + \nu^2$$

$$= \min(\hat{y}-\nu)^2$$

$\hat{y} = \nu \Rightarrow E^{sqr}(\hat{y})$ is minimum

True

(4) $-\frac{1}{N}\left(NV\ln\hat{y} + N(1-V)\ln(1-\hat{y})\right)$

$= -\left(V\ln\hat{y} + (1-V)\ln(1-\hat{y})\right)$

minimum 在 0、1 或 $\frac{\nabla E^{ce}(\hat{y})}{\nabla\hat{y}} = 0$ 的地方

不可能因為 ln 裡不能是 0

$-\left(\frac{V}{\hat{y}} - \frac{(1-V)}{1-\hat{y}}\right) = 0$

$\Rightarrow \hat{y} = V$ True

6

(a)

如果沒有錯 $y_n w^T x_n > 0 \Rightarrow -y_n w^T x_n < 0$

同號

$$\Rightarrow \max(0, -y w^T x) = 0$$

如果有錯 $y_n w^T x_n < 0 \Rightarrow -y_n w^T x_n > 0$

異號

$$\Rightarrow \max(0, -y w^T x) = -y w^T x$$

$\eta \qquad W^{(t+1)} \longleftarrow W^{(t)} + \eta V$

(a) 對 error function 進行 W 變數微分

V 會等於 error function 微分的負號

$$W = \begin{bmatrix} | & | & & | \\ w_1 & w_2 & \cdots & w_k \\ | & | & & | \end{bmatrix}_{(d+1)\times k} \quad V = \begin{bmatrix} -\dfrac{\nabla err(W, x_n, y_n)}{\nabla W_{y_n 1}} \\ \vdots \\ -\dfrac{\nabla err(W, x_n, y_n)}{\nabla W_{y_n (d+1)}} \end{bmatrix}$$

$$err(W, x_n, y_n) = -\ln h_{y_n}(x_n)$$

$$-\frac{\nabla err(W, x_n, y_n)}{\nabla W_{y_n 1}} = \frac{1}{h_{y_n}(x_n)}\left[\frac{1}{\sum_{i=1}^{K} \exp(W_i^T x_n)} \cdot \exp(W_{y_n}^T x_n)\, x_{n1} + \right.$$

$$\left. = \exp(W_{y_n}^T x_n) \cdot \left(\frac{-1}{\left(\sum_{i=1}^{K} \exp(W_i^T x_n)\right)^2} \exp(W_{y_n}^T x_n)\, x_{n1}\right)\right]$$

$$= \frac{1}{h_{y_n}(x_n)}\left[h_{y_n}(x_n)\, x_{n1} - h_{y_n}(x_n)^2 x_{n1}\right]$$

$$= (1 - h_{y_n}(x_n))\, x_{n1}$$

$$\Longrightarrow V = (1 - h_{y_n}(x_n))\, x_n$$

8 (a)、(b)、(c)、(d)、(e) 代入

(e) (e) 代入成立

$z_1 = -1^2 \quad y_1 = -1$

$z_2 = -(-1)^2 \quad y_1 = -1$

$z_3 = -(0)^2 \quad y_1 = +1$

$z_4 = -(0)^2 \quad y_1 = +1$

$\mathrm{sign}(0) = 1$

a $\quad w_{lin} = (X^T X)^{-1} X^T y$

(b) $\quad \tilde{X} = X\Gamma^T$

$$\tilde{w} = \left((X\Gamma^T)^T X\Gamma^T\right)^{-1} \Gamma X^T y$$

$$= \left[\Gamma X^T X \Gamma^T\right]^{-1} \Gamma X^T y$$

$$\Gamma X^T X \Gamma^T \tilde{w} = \Gamma X^T y$$

$\downarrow$ Γ invertible

$$X^T X \Gamma^T \tilde{w} = X^T y$$

$$\Gamma^T \tilde{w} = (X^T X)^{-1} X^T y = w_{lin}$$

$$\Downarrow$$

$$\Gamma^T \tilde{w} = w_{lin}$$

10 $\Phi(x)$: 1-hot or all zero

所有 W 都行不用討論

(c) $$\left(\begin{bmatrix} 0 \cdots 1 \cdots 0 \\ 0 \cdots 1 \cdots 0 \\ \\ \\ \end{bmatrix} W - \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \right)^2$$

$z_i = 1$ 代表 $x = x_i$

$z_i \times y_i = y_i$

11　　1　　. . .　　n

e_1　1　$(w_1x_1 - y_1')^2$　　　$(w_1x_n - y_n')^2$

e_2　2　$(w_2x_1 - y_1')^2$　　　$(w_2x_n - y_n')^2$

⋮　　⋮

e_K　K　$(w_Kx_1 - y_1')^2$　　　$(w_Kx_n - y_n')^2$

$y' = +1/-1 \quad y \in \{1, 2, \ldots K\}$

(C)

$$\arg\max(w_Kx_1) = y_1 \Rightarrow E_{in}^{0/1} = 0$$

$$\text{if } E_{in}^{0/1} = 1 \Rightarrow \arg\max(w_Kx_1) \neq y_1$$

算出來最大 $(w_1x_1 + 1)^2$

不是最大 $(w_2x_1 - 1)^2$

⋮

$(w_Kx_1 + 1)^2$

大於 0

$$\Rightarrow \begin{array}{l} w_1^2x_1^2 + 2w_1x_1 + 1 \\ w_2^2x_1^2 - 2w_2x_1 + 1 \end{array}$$

+　+　+　+

大於2

因為 $w_1x_1 > w_2x_1$

所以是正的

0 或 1

大於 2

$$E^{0/1}(x_1) \le \frac{(w_1x_1+1)^2 + \cdots + (w_kx_1+1)^2}{2}$$

$$E^{0/1}(x_2) \le \frac{- \quad - \quad -}{2}$$

$$\vdots$$

$$E^{0/1}(x_n) \le \frac{- \quad -}{2}$$

加起來
等於

$$E_{in}^{0/1}(y) \le \frac{e_1 + \cdots \; e_k}{2}$$

12 (b) 0.3263

13 (d) 0.4576

14 (a) 0.3386

15 (c) (a) 0.1366

(b) 0.1343

(c) 0.1323

(d) 0.2523

(e) 0.2653

16 (b) 0.10527

```python
import numpy as np
import random
from math import pow

def transform_poly(x_data, Q, output):
    for i in range(len(x_data)):
        output.append([])
        data_len = len(x_data[i])
        output[i].append(1.0)

        for j in range(Q):
            for k in range(data_len):
                output[i].append(pow(x_data[i][k],j+1))

def transform_fullorder(x_data, output):
    for i in range(len(x_data)):
        output.append([])
        data_len = len(x_data[i])
        output[i].append(1.0)

        for j in range(data_len):
                output[i].append(x_data[i][j])

        for j in range(data_len):
            for k in range(data_len)[j:]:
                output[i].append(x_data[i][j]*x_data[i][k])


def transform_lower(x_data, n, output):
    for i in range(len(x_data)):
        output.append([])
        output[i].append(1.0)

        for j in range(n):
            output[i].append(x_data[i][j])

def transform_random(x_data, output):
    for i in range(len(x_data)):
        output.append([])
        sample = []
        for j in range(len(x_data[i])):
            sample.append(j)

        data_len = len(x_data[i])
        output[i].append(1.0)
        sample = random.sample(sample, 5)

        for j in range(5):
            output[i].append(x_data[i][sample[j]])

def main():
    iter = 1                                #第十六題改200
    train_x = []
    train_y = []
    test_x = []
    test_y = []
    e_in = 0.0
    e_out = 0.0
    train_size = 0
    test_size = 0
    lower = 8

    with open('hw3_train.dat.txt', 'r') as f:
        while True:
            line = f.readline()
            if line == '\n' or len(line) == 0:
```

```python
                break
            x = [float(i) for i in line.split()[:-1]]
            train_x.append(x)
            train_y.append([float(line.split()[-1])])
            train_size+=1

    with open('hw3_test.dat.txt', 'r') as f:
        while True:
            line = f.readline()
            if line == '\n' or len(line) == 0:
                break
            x = [float(i) for i in line.split()[:-1]]
            test_x.append(x)
            test_y.append([float(line.split()[-1])])
            test_size+=1


    for i in range(iter):
        random.seed(i)
        train_X = []
        train_Y = train_y.copy()
        test_X = []
        test_Y = test_y.copy()

        transform_poly(train_x, 8, train_X)        #12,13題用這個改Q的數值
        transform_fullorder(train_x, train_X)      #14題用這個
        transform_lower(train_x, lower, train_X)    #15題用這個
        transform_random(train_x, train_X)          #16題用這個

        train_X = np.array(train_X)
        train_Y = np.array(train_Y)

        w_op = np.linalg.pinv(train_X).dot(train_Y)
        predict_trainy = train_X.dot(w_op)

        e_in+=((np.sign(predict_trainy)!=train_Y).sum()/train_size)

        transform_poly(test_x, 8, test_X)          #12,13題用這個改Q的數值
        transform_fullorder(test_x, test_X)        #14題用這個
        transform_lower(test_x, lower, test_X)     #15題用這個
        transform_random(test_x, test_X)           #16題用這個

        test_X = np.array(test_X)
        test_Y = np.array(test_Y)

        predict_testy = test_X.dot(w_op)

        e_out+=((np.sign(predict_testy)!=test_Y).sum()/test_size)

    print((e_out-e_in)/iter)


if __name__ == '__main__':
    main()
```