P1. D

$$\sigma^2\left(1-\frac{d+1}{N}\right)\geq 0.005,\ 代入\ \sigma=0.1,\ d=19$$

$$\Rightarrow \frac{1}{100}\left(1-\frac{20}{N}\right)\geq 0.005$$

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

$$\Rightarrow \frac{1}{N}\leq \frac{1}{40}$$

$$\Rightarrow \underline{N\geq 40}\ \#$$

P2.C 如果考慮抽樣無限多次的話，效果等同於曲線下面積，故用積分解

$$E_{out}(g) = \int_0^1 \left(x^2 - (w_1 x + w_0)\right)^2 dx$$

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

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

$$\Rightarrow \left[\frac{x^5}{5} + \frac{w_1^2 x^3}{3} + \frac{-2w_1 x^4}{4} + \frac{-2w_0 x^3}{3} + w_0 w_1 x^2 + w_0^2 x\right]_0^1$$

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

希望 E_{out} 愈小愈好，故取微分找極小值

$$\left.\begin{aligned} \frac{df}{dw_0} = 0 &\Rightarrow w_1 + 2w_0 - \frac{2}{3} = 0 \\ \frac{df}{dw_1} = 0 &\Rightarrow \frac{2}{3}w_1 + w_0 - \frac{1}{2} = 0 \end{aligned}\right] \Rightarrow (w_0, w_1) = \left(-\frac{1}{6}, 1\right) \#$$

P3.E 代入$(x_1, x_1^2), (x_2, x_2^2)$進直線方程式: $y = w_1 x + w_0$

$$\begin{cases} x_1^2 = w_1 x_1 + w_0 \\ x_2^2 = w_1 x_2 + w_0 \end{cases} \Rightarrow w_1 = \frac{x_1^2 - x_2^2}{x_1 - x_2} = x_1 + x_2$$

$$\Rightarrow x_1^2 = (x_1 + x_2) x_1 + w_0 \Rightarrow w_0 = -x_1 x_2$$

By P2. $E_{out}(g) = \frac{1}{5} + \frac{1}{3} w_1^2 - \frac{1}{2} w_1 - \frac{2}{3} w_0 + w_0 w_1 + w_0^2$

代入 $(w_0, w_1) = (-x_1 x_2, x_1 + x_2)$

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

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

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

$E_D |E_{in}(g) - E_{out}(g)|$ (因為2點成一直線, 故$E_{in}(g) = 0$)

$$= \int_0^1 \int_0^1 E_{out}(g)\, dx_1\, dx_2$$

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

$$= \frac{1}{30} \#$$

By Wolframe:

Definite integral | More digits | Step-by-step solution

$$\int_0^1 \int_0^1 \left(\frac{x^2}{3} + \frac{y^2}{3} - x^2 y - x y^2 + x^2 y^2 + \frac{4xy}{3} - \frac{x}{2} - \frac{y}{2} + \frac{1}{5} \right) dx\, dy = \frac{1}{30} \approx 0.033333$$

P4. B

選項B: $\frac{1}{N}\sum_{n=1}^{N}\left(y_n'\left(\ln\left(\theta(-w^Tx_n)\right)+(1-y_n')\ln\left(\theta(w^Tx_n)\right)\right)\right)$

考慮 $y_n'=0$ 的情況:

$$\frac{1}{N}\sum_{n=1}^{N}\left(0+(1-0)\ln\left(\theta(w^Tx_n)\right)\right)$$

$$\Rightarrow \frac{1}{N}\sum_{n=1}^{N}\ln\left(\theta(w^Tx_n)\right)$$

$$\Rightarrow \frac{1}{N}\sum_{n=1}^{N}\ln\left(\theta\left(-\left(-w^Tx_n\right)\right)\right)$$

↓因為θ是單調遞增函數,故等價

$$\Rightarrow \frac{1}{N}\sum_{n=1}^{N}\ln\left(-\theta\left(-w^Tx_n\right)\right)$$

↓因為ln是單調遞增函數,故等價

$$\Rightarrow \frac{1}{N}\sum_{n=1}^{N}-\ln\left(\theta\left(-w^Tx_n\right)\right) \quad \cdots ①$$

式①與 $E_{in}(w)$ 在 $y_n=-1$ 時相同,故在 $y_n'=0$ 的情形等價於 $E_{in}(w)$

考慮 $y_n'=1$ 的情況:

$$\frac{1}{N}\sum_{n=1}^{N}\left(\ln\left(\theta\left(-w^Tx_n\right)\right)\right)$$

↓因為 ln, θ 皆是單調遞增函數,故等價

$$\frac{1}{N}\sum_{n=1}^{N}-\ln\left(\theta\left(w^Tx_n\right)\right) \quad \cdots ②$$

式②與 $E_{in}(w)$ 在 $y_n=1$ 時相同,故在 $y_n'=1$ 的情形等價於 $E_{in}(w)$

在所有情形($y_n'\in\{0,1\}$),選項B皆等價於 $E_{in}(w)$

P5.E

選項1：正確

By Silde L3.P15 : $P[|V-\mu|>\varepsilon] \leq 2\exp(-2\varepsilon^2 N)$ ⇒Hoeffeding

$\Rightarrow \underbrace{1-P[|V-\mu|>\varepsilon]}_{1-\delta} \geq 1-\underline{2\exp(-2\varepsilon^2 N)}$

故 $\delta = 2\exp(-2\varepsilon^2 N)$

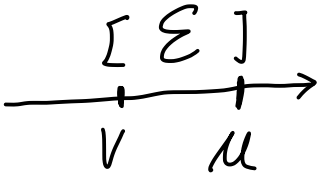
$\Rightarrow \ln(\frac{\delta}{2}) = -2\varepsilon^2 N$

$\Rightarrow \varepsilon = \sqrt{\frac{1}{-2N}(\ln\frac{\delta}{2})}$

$\Rightarrow \underline{\varepsilon = \sqrt{\frac{1}{2N}(\ln\frac{2}{\delta})}}$

故根據 Hoeffding 可得

with probability δ



$\Rightarrow \mu \leq V+\varepsilon = \underline{V+\sqrt{\frac{1}{2N}(\ln\frac{2}{\delta})}}$ #

选项2：正确

$\#\text{ of head} = \sum_{n=1}^{N} y_n = NV$

$\#\text{ of tail} = N - NV$

Find $\frac{\partial}{\partial \hat{\mu}} \text{likelihood}(\hat{\mu}) = 0$

$$\Rightarrow \frac{\partial}{\partial \hat{\mu}} \left(\left[P(\text{head}) \cdot h(\text{head}) \right]^{NV} \cdot \left[P(\text{tail}) \, h(\text{tail}) \right]^{N-NV} \right) = 0$$

$$\Rightarrow \frac{\partial}{\partial \hat{\mu}} \left((\mu \cdot \hat{\mu})^{NV} \cdot \left((1-\mu)(1-\hat{\mu}) \right)^{N-NV} \right) = 0$$

$$\Rightarrow \frac{\partial}{\partial \hat{\mu}} \left[(\mu\hat{\mu})^{NV} \cdot \left[1 - \hat{\mu} - \mu + \mu\hat{\mu} \right]^{N-NV} \right] = 0$$

$$\Rightarrow NV(\mu\hat{\mu})^{NV-1} \cdot \mu \cdot (1 - \hat{\mu} - \mu + \mu\hat{\mu})^{N-NV}$$

$$+ (\mu\hat{\mu})^{NV} \cdot (N - NV) \cdot (1 - \hat{\mu} - \mu + \mu\hat{\mu})^{N-NV-1} \cdot (-1 + \mu) = 0$$

$$\Rightarrow (1 - \hat{\mu} - \mu + \mu\hat{\mu})^{N-NV-1} (\mu\hat{\mu})^{NV-1} \left[NV\mu(1 - \hat{\mu} - \mu + \mu\hat{\mu}) + \mu\hat{\mu}(N - NV)(\mu - 1) \right] = 0$$

$$\Rightarrow V(1 - \hat{\mu} - \mu + \mu\hat{\mu}) + \hat{\mu}(1 - V)(\mu - 1) = 0$$

$$\Rightarrow V - V\hat{\mu} - V\mu + V\mu\hat{\mu} + \hat{\mu}(\mu - 1 - V\mu + V) = 0$$

$$\Rightarrow V - V\hat{\mu} - V\mu + V\mu\hat{\mu} + \hat{\mu}\mu - \hat{\mu} - V\mu\hat{\mu} + V\hat{\mu} = 0$$

$$\Rightarrow V(1 - \mu) + \hat{\mu}(\mu - 1) = 0$$

$$\Rightarrow V - \hat{\mu} = 0$$

$$\Rightarrow \hat{\mu} = V \quad \#$$

选项3：正确

求 $\frac{\partial E^{sqr}(\hat{y})}{\partial \hat{y}}$

$$\frac{\partial E^{sqr}(\hat{y})}{\partial \hat{y}} = \frac{1}{N}\frac{\partial}{\partial \hat{y}}\left(\sum_{n=1}^{N}(\hat{y}-y_n)^2\right)$$

$$= \frac{1}{N}\left(\sum_{n=1}^{N}2(\hat{y}-y_n)\cdot 1\right)$$

$$= \frac{1}{N}\left(2\sum_{n=1}^{N}\hat{y} - 2\sum_{n=1}^{N}y_n\right)$$

$$= \frac{1}{N}\left(2N\hat{y} - 2\sum_{n=1}^{N}y_n\right)$$

找minimum故求 $\frac{\partial E^{sqr}(\hat{y})}{\partial \hat{y}} = 0$

$$\frac{1}{N}\quad \frac{1}{N}\left(2N\hat{y} - 2\sum_{n=1}^{N}y_n\right) = 0$$

$$\Rightarrow \hat{y} = \frac{1}{N}\sum_{n=1}^{N}y_n = V \quad \#$$

得證V会使 $E^{squ}(\hat{y})$ 最小

选项4：正确

欲求 $\frac{\partial E^{ce}(\hat{y})}{\partial \hat{y}} = 0$

$$\frac{\partial E^{ce}(\hat{y})}{\partial \hat{y}} = \frac{\partial}{\partial \hat{y}}\left[-\frac{1}{N}\sum_{n=1}^{N}(y_n \ln \hat{y} + (1-y_n)\ln(1-\hat{y}))\right]$$

$$= \frac{-1}{N}\sum_{n=1}^{N}\left(\frac{y_n}{\hat{y}} + \frac{1-y_n}{1-\hat{y}}\cdot(-1)\right)$$

$$= \frac{-1}{N}\sum_{n=1}^{N}\left(\frac{y_n(1-\hat{y}) - \hat{y}(1-y_n)}{\hat{y}(1-\hat{y})}\right)$$

$$= \frac{-1}{N\hat{y}(1-\hat{y})}\sum_{n=1}^{N}(y_n - \hat{y})$$

$$\frac{\partial E^{ce}(\hat{y})}{\partial \hat{y}} = 0$$

$$\Rightarrow \frac{-1}{N\hat{y}(1-\hat{y})}\sum_{n=1}^{N}(y_n - \hat{y}) = 0$$

$$\Rightarrow -N\hat{y} + \sum_{n=1}^{N} y_n = 0$$

$$\Rightarrow \hat{y} = \frac{1}{N}\sum_{n=1}^{N} y_n = V \quad \#$$

P6.A 如果迭到 $y_n w_t^T x_n > 0$ 者，不需要更新，故 $err = 0$

如果迭到 $y_n w_t^T x_n < 0$ 者，需要更新：

PLA的式子是：$w_{t+1} = w_t + \underbrace{1 \cdot y_n x_n}_{-\nabla err}$

$-\nabla err = y_n x_n$

$\Rightarrow err = -y_n w^T x_n$

故整合上述情況的 err 應是 $\max(0, -y_n w^T x_n)$ #

P7. A

Want to find $\frac{-\partial}{\partial w_{yn}}\left(err(W, x_n, y_n)\right)$

$$\frac{-\partial}{\partial w_{yn}}\left(err(W, x_n, y_n)\right) = \frac{-\partial}{\partial w_{yn}}\left(-\sum_{k=1}^{K} [\![y_n = k]\!] \ln \hat{y}_k(x_n)\right)$$

$$= \frac{-\partial}{\partial w_{yn}}\left(-\ln(\hat{y}_{yn}(x_n))\right)$$

$$= \frac{1}{\hat{y}_{yn}(x_n)} \cdot \frac{\partial}{\partial w_{yn}}\left(\hat{y}_{yn}(x_n)\right)$$

$$= \frac{1}{\hat{y}_{yn}(x_n)} \cdot \frac{\partial}{\partial w_{yn}}\left[\exp(w_y^T x_n)\left[\sum_{i=1}^{K}\exp(w_i^T x_n)\right]^{-1}\right]$$

$$= \frac{1}{\hat{y}_{yn}(x_n)} \cdot \left[\underbrace{\frac{\exp(w_y^T x_n)\cdot x_n}{\sum_{i=1}^{K}\exp(w_i^T x_n)}}_{=\hat{y}_{yn}(x_n)} + \frac{\exp(w_y^T x_n)\cdot(-1)}{\left(\sum_{i=1}^{K}\exp(w_i^T x_n)\right)^2} \cdot \frac{\partial}{\partial w_{yn}}\left(\sum_{i=1}^{K}\exp(w_i^T x_n)\right)\right]$$

$$= \frac{1}{\hat{y}_{yn}(x_n)} \cdot \left[\hat{y}_{yn}(x_n)\cdot x_n - \hat{y}_{yn}(x_n) \cdot \underbrace{\frac{\exp(w_{yn}^T x_n)\cdot x_n}{\sum_{i=1}^{K}\exp(w_i^T x_n)}}_{=\hat{y}_{yn}(x_n)}\right]$$

$$= x_n - \hat{y}_{yn}(x_n) x_n$$

$$= \underline{\left(1 - \hat{y}_{yn}(x_n)\right) x_n \quad \#}$$

P8. E

$$\text{sign}(\tilde{w}\,\Phi(x))$$

$$\Rightarrow \text{sign}\left([0\ 0\ 0\ 0\ 0\ -1]\begin{bmatrix}1\\ x_1\\ x_2\\ x_1^2\\ x_1x_2\\ x_2^2\end{bmatrix}\right)$$

代入選項[e]

$$\Rightarrow \text{sign}(-x_2^2) \quad \text{代入 } x_1 \sim x_4$$

$\vec{x}_1 = (0,1)$: $\text{sign}(-x_2^2) = \text{sign}(-1) = -1 = y_1$ #

$\vec{x}_2 = (0,-1)$: $\text{sign}(-x_2^2) = \text{sign}(-1) = -1 = y_2$ #

$\vec{x}_3 = (-1,0)$: $\text{sign}(-x_2^2) = \text{sign}(0) = 1 = y_3$ #

$\vec{x}_4 = (1,0)$: $\text{sign}(-x_2^2) = \text{sign}(0) = 1 = y_4$ #

所有data皆滿足，故答案選E

P9.C

$$P\vec{x} = \begin{bmatrix} P \end{bmatrix}\begin{bmatrix} x \end{bmatrix} \Rightarrow \begin{bmatrix} P \end{bmatrix}\begin{bmatrix} 1 & 1 & & 1 \\ x_1 & x_2 & \cdots & x_n \\ 1 & 1 & & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & & 1 \\ Px_1 & Px_2 & \cdots & Px_n \\ 1 & 1 & & 1 \end{bmatrix} = \begin{bmatrix} -Px_1- \\ -Px_2- \\ \vdots \\ -Px_n- \end{bmatrix}^T$$

$$\begin{bmatrix} -Px_1- \\ -Px_2- \\ \vdots \\ -Px_n- \end{bmatrix} = (PX^T)^T = \underline{XP^T}.$$

$W_{LIN} = (X^TX)^{-1}X^TY$ (By 投影片 L5, P14)

將 X 代換成 XP^T 即可得 $\tilde{W}$

$$\tilde{W} = ((XP^T)^T(XP^T))^{-1}(XP^T)^TY$$

$$= (PX^TXP^T)^{-1}PX^TY$$

$$= P^{-T}(X^TX)^{-1}\underbrace{P^{-1}P}_{I}X^TY$$

$$= P^{-T}\underbrace{(X^TX)^{-1}X^TY}_{=W_{LIN}}$$

$$= \underline{P^{-T}W_{LIN}} \ \#$$

P10. C

$$\Phi(\vec{x}_1)^T = (1, 0, 0, \cdots 0)$$
$$\Phi(\vec{x}_2)^T = (0, 1, 0, \cdots 0)$$
$$\Phi(\vec{x}_N)^T = (0, 0, 0, \cdots 1)$$

if $\tilde{w} = \vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$,

$$\tilde{w}^T \Phi(\vec{x}_1) = y_1$$
$$\tilde{w}^T \Phi(\vec{x}_2) = y_2$$
$$\vdots$$
$$\tilde{w}^T \Phi(\vec{x}_N) = y_N$$

$E_{in}(\tilde{w}) = 0$

故 $\tilde{w}$ 為 Optimal, 選 C

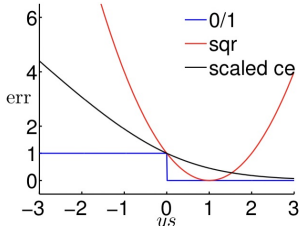
P11. C 假設有N筆 data，則 OVA 犯錯的總次數為：$N \cdot E_{in}^{0/1}(g)$

接下來，把 OVA 的犯錯，拆成各個小分類器的犯錯

其中因為 OVA 犯一次錯，代表最少有2個小分類器犯錯（正確類別預測值太小，錯誤類別值太大），故 tightest upper bound 如下：

$$N \cdot E_{in}^{0/1}(g) \leq N \cdot \sum_{k=1}^{K} E_{in}^{0/1}(W_{[k]}^{*}) \cdot \frac{1}{2}$$

By Slide L5. P.42 的圖可知 $E_{in}^{0/1}(g) \leq E_{in}^{sqr}(g)$



故：$N \cdot E_{in}^{0/1}(g) \leq N \cdot \sum_{k=1}^{K} E_{in}^{0/1}(W_{[k]}^{*}) \cdot \frac{1}{2} \leq N \cdot \sum_{k=1}^{K} E_{in}^{sqr}(W_{[k]}^{*}) \cdot \frac{1}{2}$

$$\Rightarrow N \cdot E_{in}^{0/1}(g) \leq N \cdot \sum_{k=1}^{K} e^{k} \cdot \frac{1}{2}$$

$$\Rightarrow E_{in}^{0/1}(g) \leq \frac{1}{2} \sum_{k=1}^{K} e^{k}$$

故選 C

P12.B

```python
import numpy as np
import random
Q = 2
D = 10 # Dimension of a data point

def sign(x):
    if x >= 0:
        return 1
    else:
        return -1

def data_load(file_name):
    l = []
    with open(file_name, 'r') as f:
        for i in f.readlines():
            x = [float(x_n) for x_n in i.split('\n')[0].split('\t')]
            l.append(x)
    return l

def feature_transfor(data_list):
    l = []
    for i in data_list:
        x_trans = []
        for poly in range(Q+1):
            if poly == 0:
                x_trans += [1]
                continue
            x_trans += [i[j]**poly for j in range(D)]
        l.append(x_trans)
    return l


def eval_error_01(y_pred, y_label):
    error_01 = 0
    for i, score in enumerate(y_pred):
        if sign(score) != y_label[i]: # This is a error prediction
            error_01 += 1/len(y_label)
    return error_01

# Get Training data
train_data = data_load('hw3_train.dat')
test_data = data_load('hw3_test.dat')

# Feature Transformation

train_data_trans = feature_transfor(train_data)
test_data_trans  = feature_transfor(test_data)

# Linear regression
X = []
Y = []
for i in range(len(train_data_trans)):
    X.append(train_data_trans[i])
    Y.append([train_data[i][-1]])
X = np.array(X)
Y = np.array(Y)
X_pesudo = np.linalg.pinv(X, rcond=1e-15, hermitian=False)
W_lin = np.matmul(X_pesudo, Y)
# print(W_lin)

# Evaluate Ein 0/1
y_pred = np.matmul(np.array(train_data_trans), W_lin)
Ein = eval_error_01(y_pred, [i[-1] for i in train_data])
print(f"Ein = {Ein}")

# Evaluate Eout 0/1
y_pred = np.matmul(np.array(test_data_trans), W_lin)
Eout = eval_error_01(y_pred, [i[-1] for i in test_data])
print(f"Eout = {Eout}")

# Print answer
print("|Ein - Eout| = {}".format(abs(Ein - Eout)))
```

P13.D

```python
import numpy as np
import random
Q = 8
D = 10 # Dimension of a data point

def sign(x):
    if x >= 0:
        return 1
    else:
        return -1

def data_load(file_name):
    l = []
    with open(file_name, 'r') as f:
        for i in f.readlines():
            x = [float(x_n) for x_n in i.split('\n')[0].split('\t')]
            l.append(x)
    return l

def feature_transfor(data_list):
    l = []
    for i in data_list:
        x_trans = []
        for poly in range(Q+1):
            if poly == 0:
                x_trans += [1]
                continue
            x_trans += [i[j]**poly for j in range(D)]
        l.append(x_trans)
    return l

def eval_error_01(y_pred, y_label):
    error_01 = 0
    for i, score in enumerate(y_pred):
        if sign(score) != y_label[i]: # This is a error prediction
            error_01 += 1/len(y_label)
    return error_01

# Get Training data
train_data = data_load('hw3_train.dat')
test_data = data_load('hw3_test.dat')

# Feature Transformation
train_data_trans = feature_transfor(train_data)
test_data_trans  = feature_transfor(test_data)

# Linear regression
X = []
Y = []
for i in range(len(train_data_trans)):
    X.append(train_data_trans[i])
    Y.append([train_data[i][-1]])
X = np.array(X)
Y = np.array(Y)
X_pesudo = np.linalg.pinv(X, rcond=1e-15, hermitian=False)
W_lin = np.matmul(X_pesudo, Y)
# print(W_lin)

# Evaluate Ein 0/1
y_pred = np.matmul(np.array(train_data_trans), W_lin)
Ein = eval_error_01(y_pred, [i[-1] for i in train_data])
print(f"Ein = {Ein}")

# Evaluate Eout 0/1
y_pred = np.matmul(np.array(test_data_trans), W_lin)
Eout = eval_error_01(y_pred, [i[-1] for i in test_data])
print(f"Eout = {Eout}")

# Print answer
print("|Ein - Eout| = {}".format(abs(Ein - Eout)))
```

P14. A

```python
import numpy as np

Q = 2
D = 10 # Dimension of a data point
import random

def sign(x):
    if x >= 0:
        return 1
    else:
        return -1

def data_load(file_name):
    l = []
    with open(file_name, 'r') as f:
        for i in f.readlines():
            x = [float(x_n) for x_n in i.split('\n')[0].split('\t')]
            l.append(x)
    return l

def feature_transfor_full_order(data_list):
    l = []
    for i in data_list:
        x_trans = []
        for poly in range(Q+1):
            if poly == 0:
                x_trans += [1]
                continue
            x_trans += [i[j]**poly for j in range(D)]

        # Add C10/2
        for a in range(D-1):
            for b in range(a+1):
                x_trans += [ i[a]*i[b] ]

        # print(len(x_trans))
        l.append(x_trans)
    return l

def eval_error_01(y_pred, y_label):
    error_01 = 0
    for i, score in enumerate(y_pred):
        if sign(score) != y_label[i]: # This is a error prediction
            error_01 += 1/len(y_label)
    return error_01

# Get Training data
train_data = data_load('hw3_train.dat')
test_data = data_load('hw3_test.dat')

# Feature Transformation
# P14
train_data_trans = feature_transfor_full_order(train_data)
test_data_trans  = feature_transfor_full_order(test_data)

# Linear regression
X = []
Y = []
for i in range(len(train_data_trans)):
    X.append(train_data_trans[i])
    Y.append([train_data[i][-1]])
X = np.array(X)
Y = np.array(Y)
X_pesudo = np.linalg.pinv(X, rcond=1e-15, hermitian=False)
W_lin = np.matmul(X_pesudo, Y)
# print(W_lin)

# Evaluate Ein 0/1
y_pred = np.matmul(np.array(train_data_trans), W_lin)
Ein = eval_error_01(y_pred, [i[-1] for i in train_data])
print(f"Ein = {Ein}")

# Evaluate Eout 0/1
y_pred = np.matmul(np.array(test_data_trans), W_lin)
Eout = eval_error_01(y_pred, [i[-1] for i in test_data])
print(f"Eout = {Eout}")

# Print answer
print("|Ein - Eout| = {}".format(abs(Ein - Eout)))
```

P15.C

```python
import numpy as np

Q = 2
D = 10 # Dimension of a data point
import random

def sign(x):
    if x >= 0:
        return 1
    else:
        return -1

def data_load(file_name):
    l = []
    with open(file_name, 'r') as f:
        for i in f.readlines():
            x = [float(x_n) for x_n in i.split('\n')[0].split('\t')]
            l.append(x)
    return l

def feature_transfor_lower_dim(data_list, dim):
    l = []
    for i in data_list:
        x_trans = [1]
        for d in dim:
            x_trans += [ i[d] ]
        l.append(x_trans)
    return l


def eval_error_01(y_pred, y_label):
    error_01 = 0
    for i, score in enumerate(y_pred):
        if sign(score) != y_label[i]: # This is a error prediction
            error_01 += 1/len(y_label)
    return error_01

# Get Training data
train_data = data_load('hw3_train.dat')
test_data = data_load('hw3_test.dat')

# P15
N_DIM_USE = 3 # From 1~10
print(f"N_DIM_USE = {N_DIM_USE}")
train_data_trans = feature_transfor_lower_dim(train_data, [d for d in range(N_DIM_USE)]) # dim =
[0,1,2,3,4,5,6,7,8,9] means use all dimension
test_data_trans  = feature_transfor_lower_dim(test_data, [d for d in range(N_DIM_USE)])

# Linear regression
X = []
Y = []
for i in range(len(train_data_trans)):
    X.append(train_data_trans[i])
    Y.append([train_data[i][-1]])
X = np.array(X)
Y = np.array(Y)
X_pesudo = np.linalg.pinv(X, rcond=1e-15, hermitian=False)
W_lin = np.matmul(X_pesudo, Y)
# print(W_lin)

# Evaluate Ein 0/1
y_pred = np.matmul(np.array(train_data_trans), W_lin)
Ein = eval_error_01(y_pred, [i[-1] for i in train_data])
print(f"Ein = {Ein}")

# Evaluate Eout 0/1
y_pred = np.matmul(np.array(test_data_trans), W_lin)
Eout = eval_error_01(y_pred, [i[-1] for i in test_data])
print(f"Eout = {Eout}")

# Print answer
print("|Ein - Eout| = {}".format(abs(Ein - Eout)))
```

P16.D

```python
import numpy as np

Q = 2
D = 10 # Dimension of a data point
import random

def sign(x):
    if x >= 0:
        return 1
    else:
        return -1

def data_load(file_name):
    l = []
    with open(file_name, 'r') as f:
        for i in f.readlines():
            x = [float(x_n) for x_n in i.split('\n')[0].split('\t')]
            l.append(x)
    return l

def feature_transfor_lower_dim(data_list, dim):
    l = []
    for i in data_list:
        x_trans = [1]
        for d in dim:
            x_trans += [ i[d] ]
        l.append(x_trans)
    return l


def eval_error_01(y_pred, y_label):
    error_01 = 0
    for i, score in enumerate(y_pred):
        if sign(score) != y_label[i]: # This is a error prediction
            error_01 += 1/len(y_label)
    return error_01

# Get Training data
train_data = data_load('hw3_train.dat')
test_data = data_load('hw3_test.dat')

# Feature Transformation
ans = 0
for _ in range(200):
    dim = sorted(random.sample(range(D), 5))
    print(f"dim = {dim}")
    train_data_trans = feature_transfor_lower_dim(train_data, dim) # dim = [0,1,2,3,4,5,6,7,8,9] means
use all dimension
    test_data_trans  = feature_transfor_lower_dim(test_data, dim)

    # Linear regression
    X = []
    Y = []
    for i in range(len(train_data_trans)):
        X.append(train_data_trans[i])
        Y.append([train_data[i][-1]])
    X = np.array(X)
    Y = np.array(Y)
    X_pesudo = np.linalg.pinv(X, rcond=1e-15, hermitian=False)
    W_lin = np.matmul(X_pesudo, Y)
    # print(W_lin)

    # Evaluate Ein 0/1
    y_pred = np.matmul(np.array(train_data_trans), W_lin)
    Ein = eval_error_01(y_pred, [i[-1] for i in train_data])
    print(f"Ein = {Ein}")

    # Evaluate Eout 0/1
    y_pred = np.matmul(np.array(test_data_trans), W_lin)
    Eout = eval_error_01(y_pred, [i[-1] for i in test_data])
    print(f"Eout = {Eout}")

    # Print answer
    print("|Ein - Eout| = {}".format(abs(Ein - Eout)))


    ans += abs(Ein - Eout)/200
print(f"ans = {ans}")
```