1.

$$E_D[E_{in}(w_{lin})] = \sigma^2\left(1 - \frac{d+1}{N}\right)$$

$\sigma = 0.1$, $d = 19$

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

$$\Rightarrow 0.005 \geq 0.01\left(1 - \frac{20}{N}\right)$$

$$40 \geq N$$

ans : d

2.

(a) $\int_0^1 (x^2-x)^2\,dx = 0.33$

(b) $\int_0^1 (x^2-\frac{1}{2}x-\frac{1}{2})\,dx = 0.2$

(c) $\int_0^1 (x^2-x+\frac{1}{6})^2\,dx = 0.005$, squared error最小

(d) $\int_0^1 (x^2-\frac{1}{4}x+\frac{1}{4})^2\,dx = 0.2625$

(e) $\int_0^1 (x^2-\frac{1}{3})^2\,dx = 0.08$

ans: C

3. $g(x)=(x_2+x_1)(x-x_1)+x_1^2$

$E\,|E_{in}(g)-E_{out}(g)|^2=$

$$\int_0^1\int_0^1\int_0^1 [x^2-(x_1x+x_2x-x_1^2-x_1x_2+x_2^2)]^2\,dx_1\,dx_2\,dx$$

$$=\int_0^1\int_0^1 \frac{1}{5}-\frac{x_1}{2}-\frac{x_2}{2}+\frac{x_1^2}{3}+\frac{x_2^2}{3}+\frac{4x_1x_2}{3}-x_1x_2^2$$

$$-x_1^2x_2+(x_1x_2)^2\,dx_1x_2$$

$$=\int_0^1\left(\frac{1}{5}-\frac{x_1}{2}-\frac{x_2}{2}+\frac{x_1^2}{3}+\frac{x_2^2}{3}+\frac{4x_1x_2}{3}-x_1x_2^2\right.$$

$$\left.-x_1^2x_2+(x_1x_2)^2\right)dx_1$$

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

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

3.

4.

$$\theta(s) = \frac{1}{1+\exp(-s)} = \frac{\exp(s)}{1+\exp(s)}$$

$$\Rightarrow \frac{1+\exp(s)}{1} = \frac{1+\exp(s)}{\exp(s)}$$

$$\Rightarrow \ln(1+\exp(s)) = \ln(1+\exp(s)) - s \quad -(1)$$

$$\Rightarrow s = \ln\left(\frac{1+\exp(s)}{1+\exp(-s)}\right) \quad -(2)$$

当 $y_n \in \{0,1\}$ 时，$P(y=1|s) = \theta(s) = \frac{1}{1+e^{-s}}$，$s = w^T x$

- $P(y=0|s) = 1-\theta(s) = \frac{1}{1+e^{s}}$

- $P(y|s) = \theta(s)^y (1-\theta(s))^{1-y}$

$$J(s) = -\ln\left(\prod_n^N P(y_n|s_n)\right) = -\sum_n^N \ln\left(P(y_n|s_n)\right)$$

$$= -\sum_n^N \ln\left(\left(\frac{1}{1+e^{-s_n}}\right)^{y_n}\left(\frac{1}{1+e^{s_n}}\right)^{1-y_n}\right)$$

$$= -\sum_n^N \ln\left(\left(\frac{1}{1+e^{-s_n}}\right)^{y_n}\right) + \ln\left(\left(\frac{1}{1+e^{s_n}}\right)^{1-y_n}\right)$$

$$= -\sum_n^N y_n \ln\left(\theta(s_n)\right) + (1-y_n)\ln\left(\theta(-s_n)\right) \quad (3)$$

$$= -\sum_n^N -y_n \ln\left(1+e^{-s_n}\right) - (1-y_n)\ln\left(1+e^{s_n}\right)$$

$$= \sum_n^N y_n \ln\left(1+e^{-s}\right) + (1-y_n)\ln\left(1+e^{s_n}\right)$$

$$= \sum_n^N y_n \ln\left(\frac{1+e^{-s_n}}{1+e^{s_n}}\right) + \ln\left(1+e^{s_n}\right)$$

$$= \sum_n^N -s_n \cdot y_n + \ln\left(1+e^{s_n}\right)$$

$$= \sum_n^N -y_n s_n + \ln\left(1+e^{s_n}\right)$$

當 $y_n \in \{-1, 1\}$ 時，$P(y_n|x_n) = \theta(y \cdot s)$

$$L(s) = -\ln\left(\prod_n^N P(y_n|s_n)\right)$$

$$= -\sum_n^N \ln\left(\frac{1}{1+e^{-y s_n}}\right)$$

$$= -\sum_n^N -\ln(1+e^{-y s_n})$$

$$= \sum_n^N \ln(1+e^{-y s})$$

假設 $\ln(1+e^{-y s}) = -y s_n + \ln(1+e^{s})$

當 $y=1$，$\ln(1+e^{-y s}) = -y s + \ln(1+e^{s})$

$$\Rightarrow \ln(1+e^{-s}) = -s + \ln(1+e^{s})$$

如(1)

當 $y \neq 1$，則左式的 $y=-1$，右式的 $y=0$

$$\Rightarrow \ln(1+e^{s}) = \ln(1+e^{s})$$

上面推导过程在 $y\in(0,1)$, $y\{+1,-1\}$ 时，
均加上 $\frac{1}{N}$，其等式不变，

故(3)

$$\Rightarrow \frac{1}{N}\sum_{n}^{N} y_n \ln(1+e^{-s}) + (1-y_n)\ln(1+e^{s})$$

$$\Rightarrow ans = [a]$$

6. [a]

when $\eta = 1$, $w^T x$ is large, PLA can viewed as SGD

SGD: $w_{t+1} \leftarrow w_t + \eta \theta(-y_n w_t^T x_n)(y_n x_n)$

if $\eta = 1$, $w^T x$ is large

當判斷正確: $-y_n w_t^T x_n \simeq -\infty$, $\theta(-ys) \simeq 0$

w不更新

當判斷錯誤: $-y_n w_t^T x_n \simeq \infty$, $\theta(-ys) \simeq 1$

[a] $err(w, x, y) = \max(0, -yw^T x)$

$err_{CE}(s, y)$

$= \ln(1 + \exp(-ys))$

if predict correct, ys is large $\Rightarrow err_{CE} \simeq \ln(1) = 0$

if predict incorrect, $ys \simeq -\infty$ $\Rightarrow err_{CE} \simeq \ln(\exp(-ys))$

$= -ys = -yw^T x$

故(a)為 ans.

7

8.

$\Phi_2(X)$: $X_1 \to X_1' = [1\ 0\ 1\ 0\ 0\ 1]$

$X_2 \to X_2' = [1\ 0\ -1\ 0\ 0\ 1]$

$X_3 \to X_3' = [1\ -1\ 0\ 1\ 0\ 0]$

$X_4 \to X_4' = [1\ 1\ 0\ 1\ 0\ 0]$

(e) when weight is $(0\ 0\ 0\ 0\ 0\ -1)$

then $w^T X_1' = 1$

$w^T X_2' = 1$

$w^T X_3' = 0$

$w^T X_4' = 0$

故 (e) is ans.

9.

$$w_{lin} = X^{\dagger} y$$

$$= (X^T X)^{-1} X^T y$$

$X' = \Phi x = DX$ _ when D is $d+1 \times d+1$

$$\tilde{w} = (X'^T X')^{-1} X'^T y$$

$$= ((DX)^T DX)^{-1} (DX)^T y$$

$$= (X^T D^T DX)^{-1} X^T D^T \cdot K = \tilde{w} \cdot K = w_{lin}$$

$$(X^T X)^{-1} X^T \cdot = w_{lin}$$

$$X^T D^T \cdot K = X^T D^T DX (X^T X)^{-1} X^T$$

$$K = DX (X^T X)^{-1} X^T$$

$$K = DX X^{\dagger}$$

$$KX = DXX^tX$$

$$K.X = DX$$

$$K = D \Rightarrow \text{cons} : (a)$$

10. 假設 $\Phi(x)$, $x = x_i$, 假設 $x_i = 1$, 其它皆不是1

則 $\Phi(x) = \left(\begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix}, \begin{bmatrix}0\\0\\\vdots\\0\end{bmatrix}, \cdots \begin{bmatrix}0\\0\\\vdots\\0\end{bmatrix}\right)$

$x_1 \quad x_2 \quad x_n$

(a) $w = [1, 1, \cdots 1]$

$w \cdot x = [n, 0, 0 \cdots 0]$

$\Rightarrow [+1, -1, -1 \cdots -1]$, $E_{in}(\tilde{w}) = 0$

```python
import numpy as np
from numpy.linalg import multi_dot



# 1.取出所有(training data,training labels) 、(testing data,testing labels)
# 2.將所有資料(both test and train)分別轉換至Q12~Q16指定的domain
# 3.計算 M_plus = np.linalg.pinv
# 4.W_lin = np.dot(M_plus,labels)
# 5.|計算err_in(g) - err_out(g)|

# M_plus = np.linalg.pinv(DataSET[0][0])
# W_lin = np.dot(M_plus, DataSET[0][1])

# Q12 Q13
def Q_order(data,order):

    len_data = len(data)
    ans = np.zeros(1+len_data*order)
    temp_ans = np.zeros((order,len_data))
    for _ in range(1,order+1):
        temp_ans[_-1] = data ** _
    temp_ans = temp_ans.reshape(-1)
    ans[0] = 1
    ans[1:] = temp_ans
    return ans
# Q14
def full_order_2_polynomial(data):
    len_data = len(data)

    second_order = []
    counter = 0
    for i in range(len_data):
        x1 = data[i]
        for j in range(counter,len_data):
            second_order.append(x1*data[j])
        counter += 1
    data = np.insert(data,0,1,axis=0)
    data =list(data)
    t_data = np.array(data+second_order)
    return t_data
# Q15
def trans_lower_dim(data,split_num  = 1):
    data = data[:split_num]
    data = np.insert(data,0,1,axis=0)
    return data
def random_Dimension(data):
    data = np.array(data)
    index = np.random.choice(np.arange(10),size = 5,replace=False)
    data = data[index]
    data = list(data)
    x_0 = [1]
    data = np.array(x_0+data)
    return data
def read_data_and_trans(data_path,transTo,Q15_dim = 1)->np.ndarray:
    All_data = []
    with open(data_path) as file:
        for data in file:
            data = [float(d) for d in data.split()]
            All_data.append(data)
    All_data = np.array(All_data)
    data,labels = All_data[:,:-1],All_data[:,-1:]
    labels = labels.reshape(-1)
    h,w = data.shape
    transData = []
    if transTo == "Q12":
        # trans_data = np.zeros((h,w),dtype=np.float32)
        for _ in range(h):
            t_data = Q_order(data[_],2)
            transData.append(t_data)
        transData = np.array(transData)
    if transTo == "Q13":
        for _ in range(h):
            t_data = Q_order(data[_],8)
            transData.append(t_data)
        transData = np.array(transData)
    if transTo == "Q14":
        for _ in range(h):
            t_data = full_order_2_polynomial(data[_])
            transData.append(t_data)
        transData = np.array(transData)
    if transTo == "Q15":
        for _ in range(h):
            t_data = trans_lower_dim(data[_],Q15_dim)
            transData.append(t_data)
        transData = np.array(transData)
    if transTo == "Q16":
        for _ in range(h):
            t_data = random_Dimension(data[_])
            transData.append(t_data)
        transData = np.array(transData)
    return transData,labels
def Q12(training_data_path,testing_data_path):
    t_traning_data ,training_labels = read_data_and_trans(training_data_path,"Q12",Q15_dim=0)
    t_testinging_data ,testing_labels = read_data_and_trans(testing_data_path,"Q12",Q15_dim=0)

    M_plus = np.linalg.pinv(t_traning_data)
    W_lin = np.dot(M_plus, training_labels).reshape(21,1)

    tran_pred = np.dot(t_traning_data,W_lin)
    tran_pred[tran_pred>=0] = 1
    tran_pred[tran_pred<0] = -1
    tran_pred = tran_pred.reshape(-1)

    pred = np.dot(t_testinging_data,W_lin)
    pred[pred>=0] = 1
    pred[pred<0] = -1
    pred = pred.reshape(-1)

    Ein = sum(tran_pred==training_labels) / len(tran_pred)
    Eout = sum(pred == testing_labels) / len(pred)

    return abs(Ein- Eout)
def Q13(training_data_path,testing_data_path):
    t_traning_data, training_labels = read_data_and_trans(training_data_path, "Q13", Q15_dim=0)
    t_testinging_data, testing_labels = read_data_and_trans(testing_data_path, "Q13", Q15_dim=0)

    M_plus = np.linalg.pinv(t_traning_data)
    W_lin = np.dot(M_plus, training_labels).reshape(81, 1)

    tran_pred = np.dot(t_traning_data, W_lin)
    tran_pred[tran_pred >= 0] = 1
    tran_pred[tran_pred < 0] = -1
    tran_pred = tran_pred.reshape(-1)

    pred = np.dot(t_testinging_data, W_lin)
    pred[pred >= 0] = 1
    pred[pred < 0] = -1
    pred = pred.reshape(-1)

    Ein = sum(tran_pred == training_labels) / len(tran_pred)
    Eout = sum(pred == testing_labels) / len(pred)

    return abs(Ein - Eout)
def Q14(training_data_path,testing_data_path):
    t_traning_data, training_labels = read_data_and_trans(training_data_path, "Q14", Q15_dim=0)
    t_testinging_data, testing_labels = read_data_and_trans(testing_data_path, "Q14", Q15_dim=0)

    t_testinging_data_len = len(t_testinging_data[0])

    M_plus = np.linalg.pinv(t_traning_data)
    W_lin = np.dot(M_plus, training_labels).reshape(t_testinging_data_len, 1)

    tran_pred = np.dot(t_traning_data, W_lin)
    tran_pred[tran_pred >= 0] = 1
    tran_pred[tran_pred < 0] = -1
    tran_pred = tran_pred.reshape(-1)

    pred = np.dot(t_testinging_data, W_lin)
    pred[pred >= 0] = 1
    pred[pred < 0] = -1
    pred = pred.reshape(-1)

    Ein = sum(tran_pred == training_labels) / len(tran_pred)
    Eout = sum(pred == testing_labels) / len(pred)

    return abs(Ein - Eout)
def Q15(training_data_path,testing_data_path):
    # 假設一個全錯，一個全對，取abs最大也對2
    find_min_err_L = 2
    Err_bag = []
    ans = 0
    for i in range(1,10+1):
        t_traning_data, training_labels = read_data_and_trans(training_data_path, "Q15", Q15_dim=i)
        t_testinging_data, testing_labels = read_data_and_trans(testing_data_path, "Q15", Q15_dim=i)

        t_testinging_data_len = len(t_testinging_data[0])

        M_plus = np.linalg.pinv(t_traning_data)
        W_lin = np.dot(M_plus, training_labels).reshape(t_testinging_data_len, 1)

        tran_pred = np.dot(t_traning_data, W_lin)
        tran_pred[tran_pred >= 0] = 1
        tran_pred[tran_pred < 0] = -1
        tran_pred = tran_pred.reshape(-1)

        pred = np.dot(t_testinging_data, W_lin)
        pred[pred >= 0] = 1
        pred[pred < 0] = -1
        pred = pred.reshape(-1)

        Ein = sum(tran_pred == training_labels) / len(tran_pred)
        Eout = sum(pred == testing_labels) / len(pred)
        Err_bag.append(abs(Ein - Eout))
        if abs(Ein - Eout) < find_min_err_L:
            find_min_err_L = abs(Ein - Eout)
            ans = i


    return Err_bag,ans
def Q16(training_data_path,testing_data_path):

    average_err = 0
    for _ in range(200):
        t_traning_data, training_labels = read_data_and_trans(training_data_path, "Q16", Q15_dim=0)
        t_testinging_data, testing_labels = read_data_and_trans(testing_data_path, "Q16", Q15_dim=0)
        t_testinging_data_len = len(t_testinging_data[0])

        M_plus = np.linalg.pinv(t_traning_data)
        W_lin = np.dot(M_plus, training_labels).reshape(t_testinging_data_len, 1)

        tran_pred = np.dot(t_traning_data, W_lin)
        tran_pred[tran_pred >= 0] = 1
        tran_pred[tran_pred < 0] = -1
        tran_pred = tran_pred.reshape(-1)

        pred = np.dot(t_testinging_data, W_lin)
        pred[pred >= 0] = 1
        pred[pred < 0] = -1
        pred = pred.reshape(-1)

        Ein = sum(tran_pred == training_labels) / len(tran_pred)
        Eout = sum(pred == testing_labels) / len(pred)
        average_err += abs(Ein - Eout)
    average_err = average_err/200
    return average_err


if __name__ == '__main__':

    training_data_path = "./hw3_train.txt"
    testing_data_path = "./hw3_test.txt"
    Q12_ans = Q12(training_data_path, testing_data_path)
    print(f"Q12 : {Q12_ans}")

    Q13_ans = Q13(training_data_path, testing_data_path)
    print(f"Q13 : {Q13_ans}")

    Q14_ans = Q14(training_data_path, testing_data_path)
    print(f"Q14 : {Q14_ans}")

    all_err, min_err_index = Q15(training_data_path, testing_data_path)
    # print(all_err)
    print(f"Q15 : {min_err_index}")

    Q16_ans = Q16(training_data_path, testing_data_path)
    print(f"Q16 : ", Q16_ans)
```

```
Q12 : 0.32633333333333333
Q13 : 0.4576666666666666
Q14 : 0.3386666666666666
Q15 : 3
Q16 : 0.1064866666666666
```