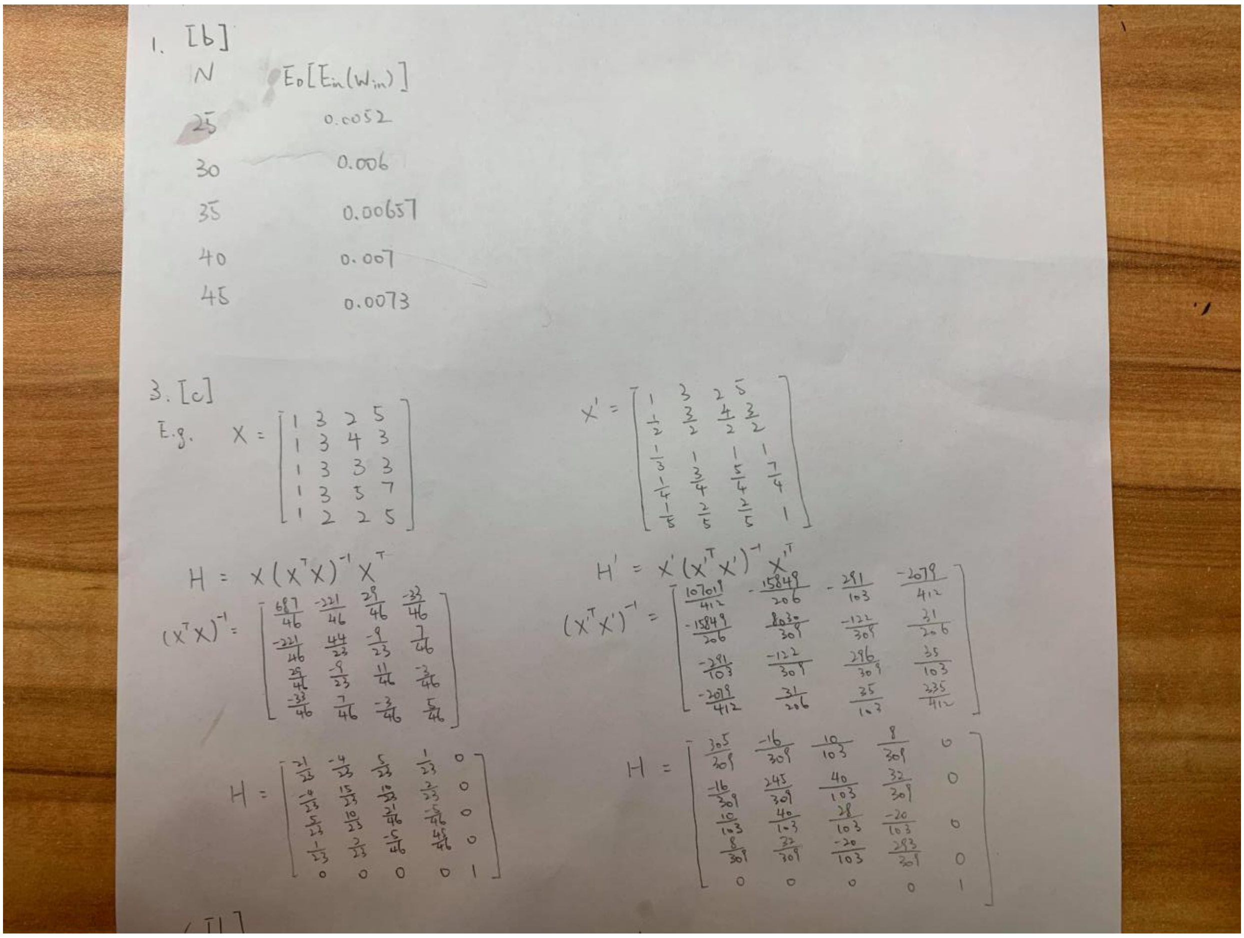
B05902091 黎峻碩



1. [b]

N	$E_D[E_{in}(w_{lin})]$
25	0.0052
30	0.006
35	0.00657
40	0.007
45	0.0073

3. [c]

E.g. $X = \begin{bmatrix} 1 & 3 & 2 & 5 \\ 1 & 3 & 4 & 3 \\ 1 & 3 & 3 & 3 \\ 1 & 3 & 5 & 7 \\ 1 & 2 & 2 & 5 \end{bmatrix}$

$$H = X(X^TX)^{-1}X^T$$

$$(X^TX)^{-1} = \begin{bmatrix} \frac{687}{46} & \frac{-221}{46} & \frac{29}{46} & \frac{-33}{46} \\ \frac{-221}{46} & \frac{44}{23} & \frac{-9}{23} & \frac{7}{46} \\ \frac{29}{46} & \frac{-9}{23} & \frac{11}{46} & \frac{-3}{46} \\ \frac{-33}{46} & \frac{7}{46} & \frac{-3}{46} & \frac{5}{46} \end{bmatrix}$$

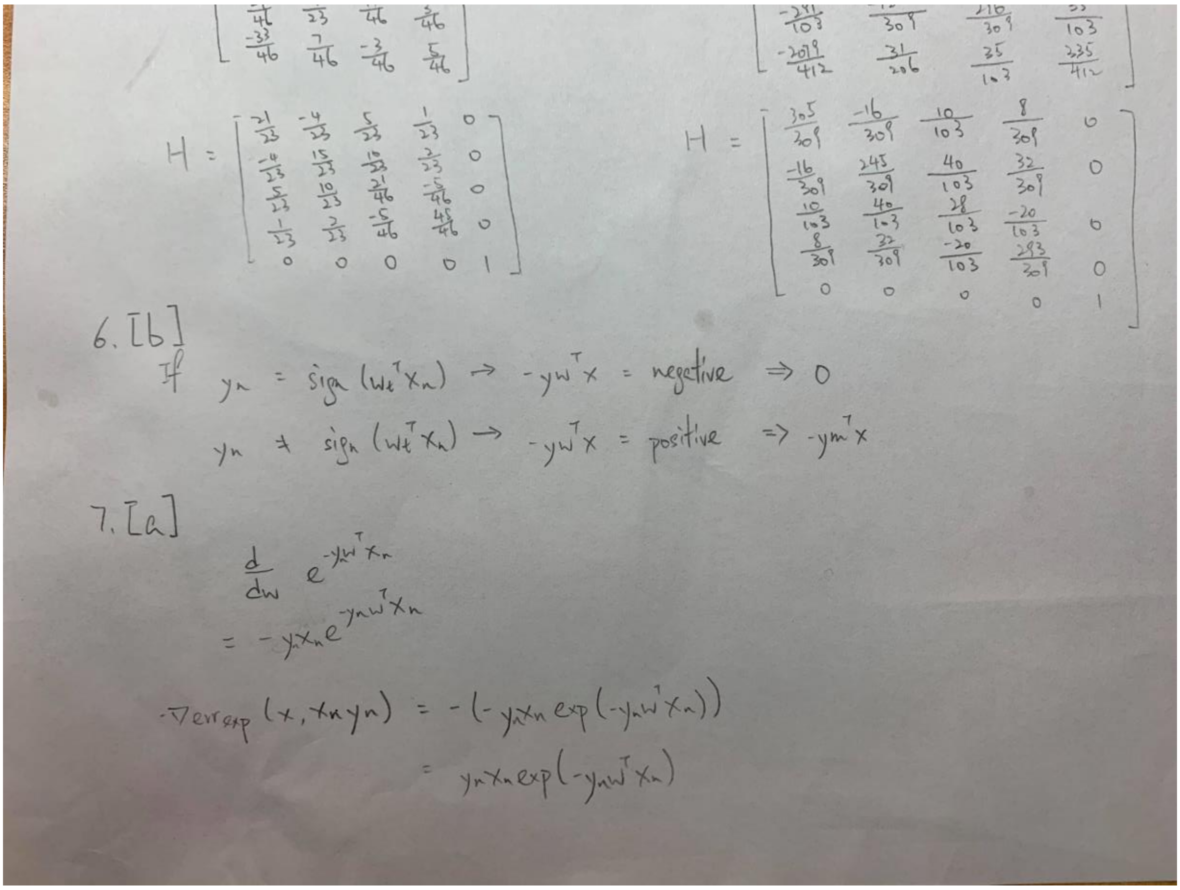
$$H = \begin{bmatrix} \frac{21}{23} & \frac{-4}{23} & \frac{5}{23} & \frac{1}{23} & 0 \\ \frac{-4}{23} & \frac{15}{23} & \frac{10}{23} & \frac{2}{23} & 0 \\ \frac{5}{23} & \frac{10}{23} & \frac{21}{46} & \frac{-5}{46} & 0 \\ \frac{1}{23} & \frac{2}{23} & \frac{-5}{46} & \frac{45}{46} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$X' = \begin{bmatrix} 1 & 3 & 2 & 5 \\ \frac{1}{2} & \frac{3}{2} & \frac{4}{2} & \frac{3}{2} \\ \frac{1}{3} & 1 & 1 & 1 \\ \frac{1}{4} & \frac{3}{4} & \frac{5}{4} & \frac{7}{4} \\ \frac{1}{5} & \frac{2}{5} & \frac{2}{5} & 1 \end{bmatrix}$

$$H' = X'(X'^TX')^{-1}X'^T$$

$$(X'^TX')^{-1} = \begin{bmatrix} \frac{107019}{412} & -\frac{15849}{206} & -\frac{291}{103} & \frac{-2079}{412} \\ -\frac{15849}{206} & \frac{8030}{309} & \frac{-122}{309} & \frac{31}{206} \\ \frac{-291}{103} & \frac{-122}{309} & \frac{296}{309} & \frac{35}{103} \\ -\frac{2079}{412} & \frac{31}{206} & \frac{35}{103} & \frac{335}{412} \end{bmatrix}$$

$$H = \begin{bmatrix} \frac{305}{309} & \frac{-16}{309} & \frac{10}{103} & \frac{8}{309} & 0 \\ \frac{-16}{309} & \frac{245}{309} & \frac{40}{103} & \frac{32}{309} & 0 \\ \frac{10}{103} & \frac{40}{103} & \frac{28}{103} & \frac{-20}{103} & 0 \\ \frac{8}{309} & \frac{32}{309} & \frac{-20}{103} & \frac{293}{309} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$



$$\begin{bmatrix} \frac{}{46} & \frac{}{23} & \frac{}{46} & \frac{}{46} \\ \frac{-33}{46} & \frac{7}{46} & \frac{-3}{46} & \frac{5}{46} \end{bmatrix} \qquad \begin{bmatrix} \frac{-241}{103} & \frac{}{309} & \frac{216}{309} & \frac{}{103} \\ \frac{-2079}{412} & \frac{31}{206} & \frac{35}{103} & \frac{335}{412} \end{bmatrix}$$

$$H = \begin{bmatrix} \frac{21}{23} & \frac{-4}{23} & \frac{5}{23} & \frac{1}{23} & 0 \\ \frac{-4}{23} & \frac{15}{23} & \frac{10}{23} & \frac{2}{23} & 0 \\ \frac{5}{23} & \frac{10}{23} & \frac{21}{46} & \frac{-5}{46} & 0 \\ \frac{1}{23} & \frac{2}{23} & \frac{-5}{46} & \frac{45}{46} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad H = \begin{bmatrix} \frac{305}{309} & \frac{-16}{309} & \frac{10}{103} & \frac{8}{309} & 0 \\ \frac{-16}{309} & \frac{245}{309} & \frac{40}{103} & \frac{32}{309} & 0 \\ \frac{10}{103} & \frac{40}{103} & \frac{28}{103} & \frac{-20}{103} & 0 \\ \frac{8}{309} & \frac{32}{309} & \frac{-20}{103} & \frac{293}{309} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

6. [b]

If $y_n = \text{sign}(w_t^T x_n) \rightarrow -yw^T x = \text{negative} \Rightarrow 0$

$y_n \neq \text{sign}(w_t^T x_n) \rightarrow -yw^T x = \text{positive} \Rightarrow -ym^T x$

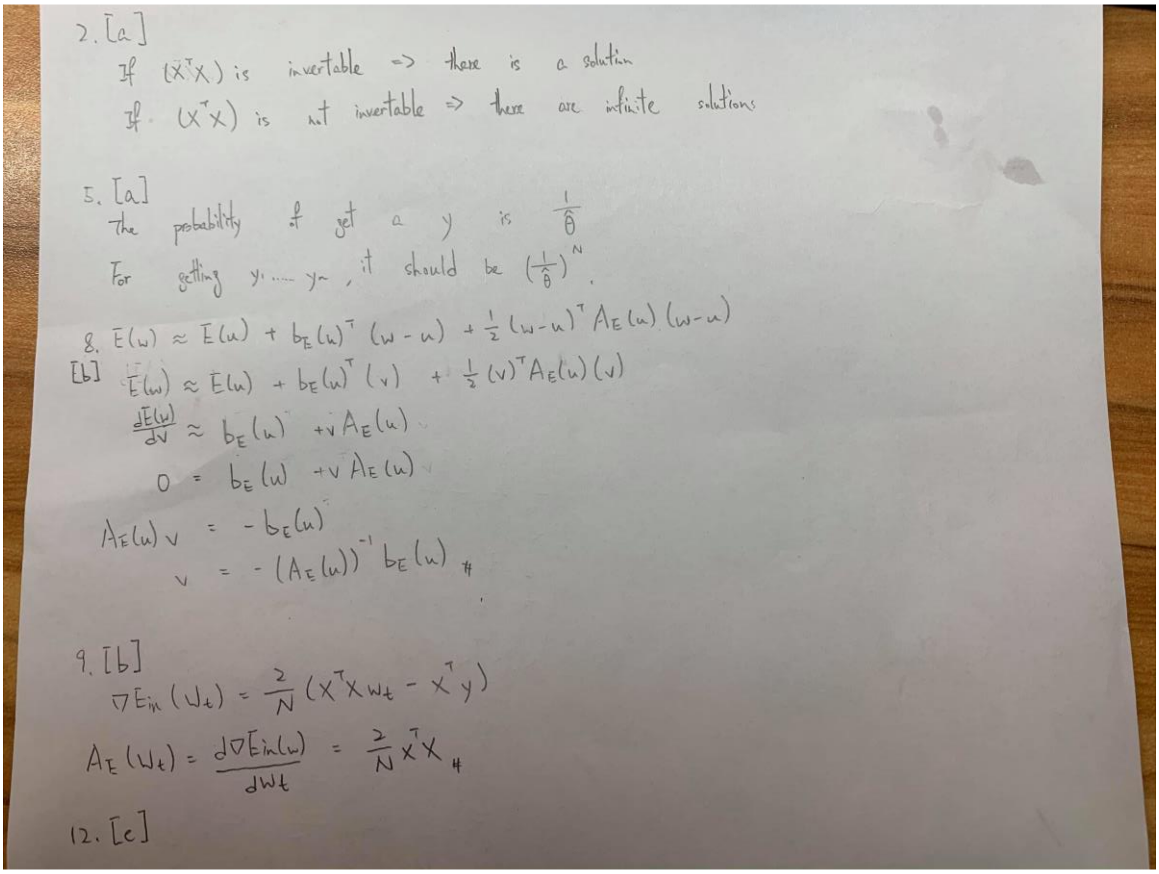
7. [a]

$$\frac{d}{dw} e^{-y_n w^T x_n}$$

$$= -y_n x_n e^{-y_n w^T x_n}$$

$$-\nabla \text{err}_{\exp}(x, x_n y_n) = -(-y_n x_n \exp(-y_n w^T x_n))$$

$$= y_n x_n \exp(-y_n w^T x_n)$$



2. [a]

If $(X^T X)$ is invertable => there is a solution

If $(X^T X)$ is not invertable => there are infinite solutions

5. [a]

The probability of get a y is $\frac{1}{\hat{\theta}}$

For getting $y_1 \ldots y_n$, it should be $\left(\frac{1}{\hat{\theta}}\right)^N$.

8. $E(w) \approx E(u) + b_E(u)^T (w-u) + \frac{1}{2}(w-u)^T A_E(u)(w-u)$

[b] $E(w) \approx E(u) + b_E(u)^T (v) + \frac{1}{2}(v)^T A_E(u)(v)$

$$\frac{dE(w)}{dv} \approx b_E(u) + v A_E(u)$$

$$0 = b_E(w) + v A_E(u)$$

$$A_E(u) v = -b_E(u)$$

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

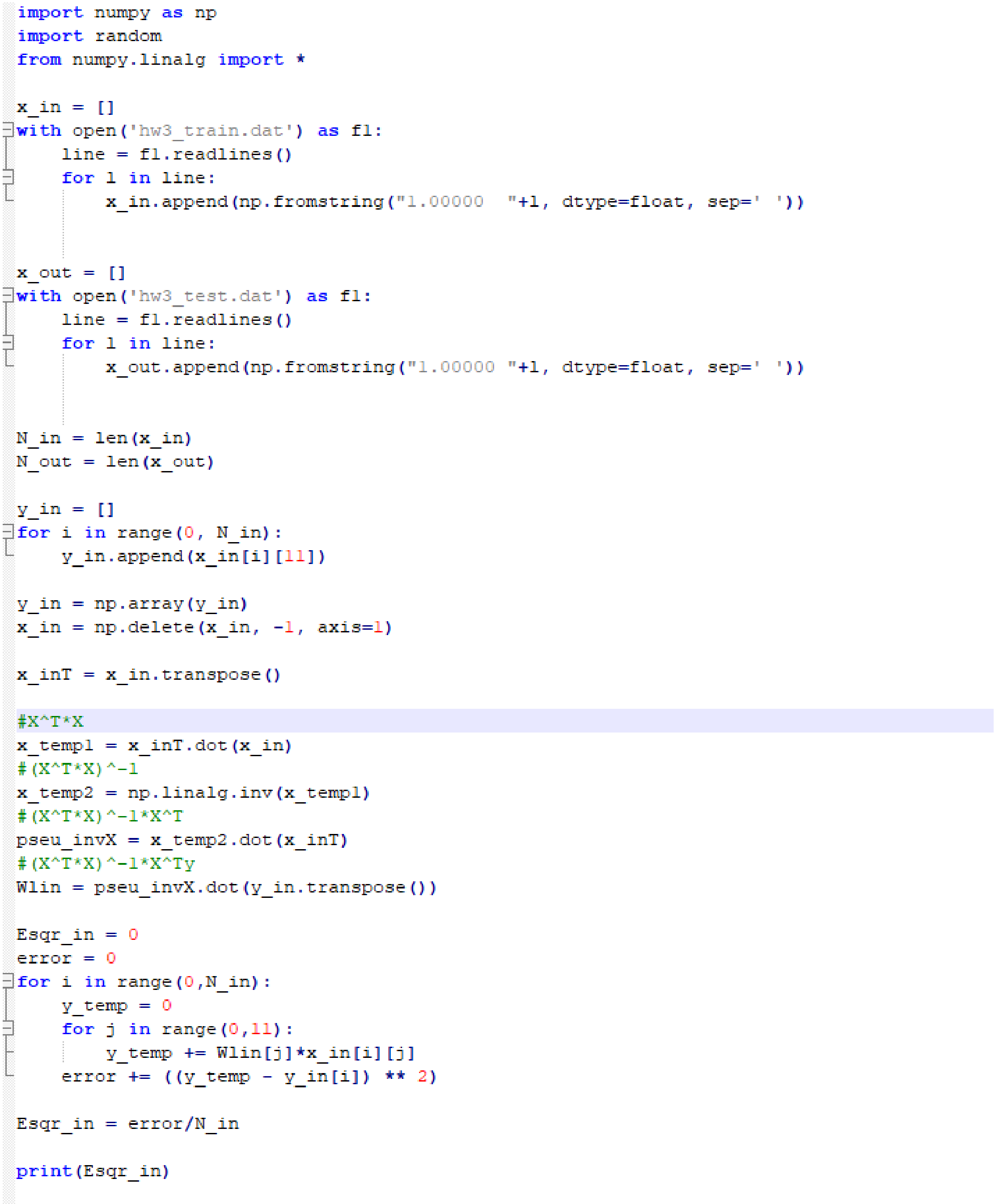
9. [b]

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

$$A_E(w_t) = \frac{d \nabla E_{in}(w)}{d w_t} = \frac{2}{N} X^T X \quad \#$$

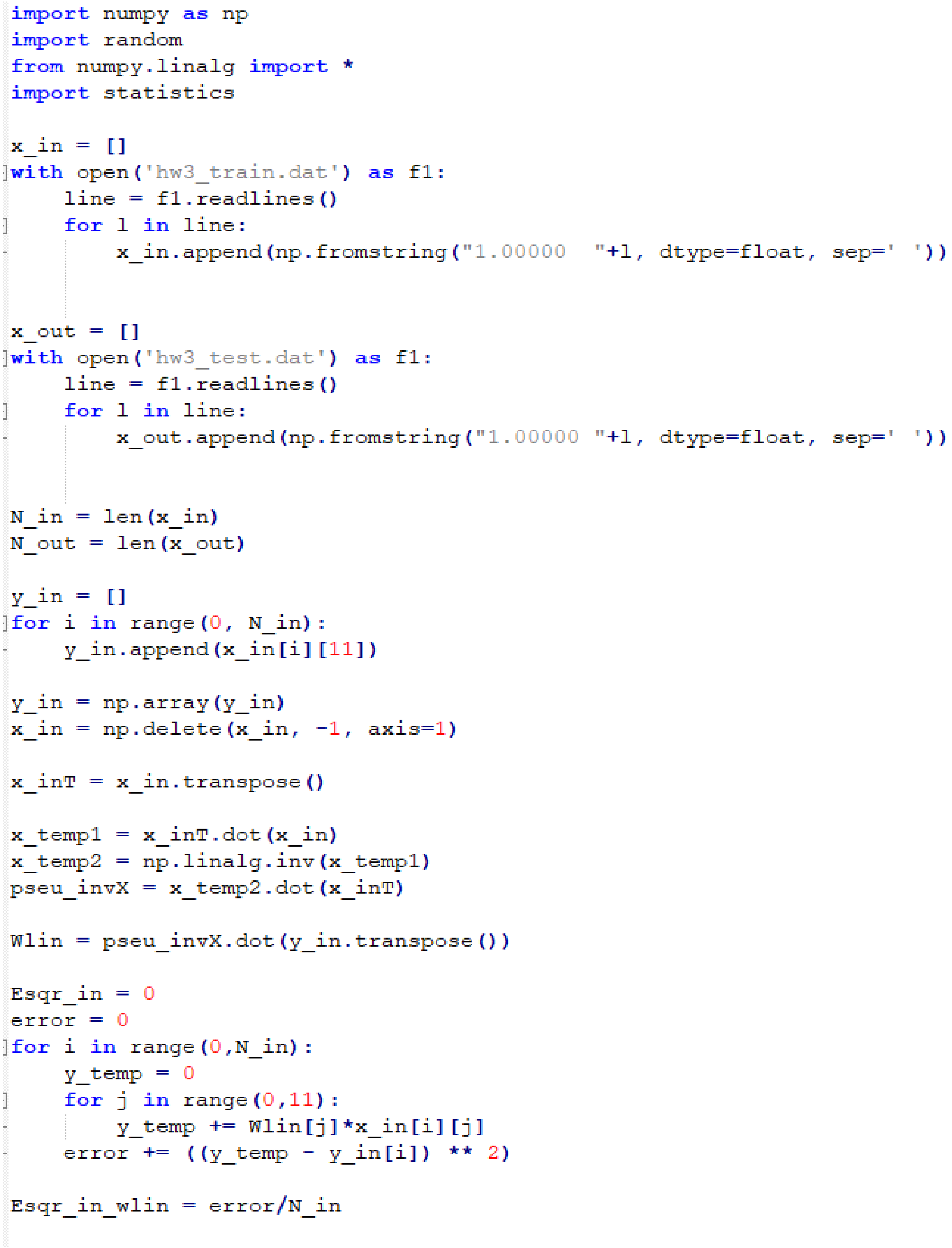
12. [c]

14

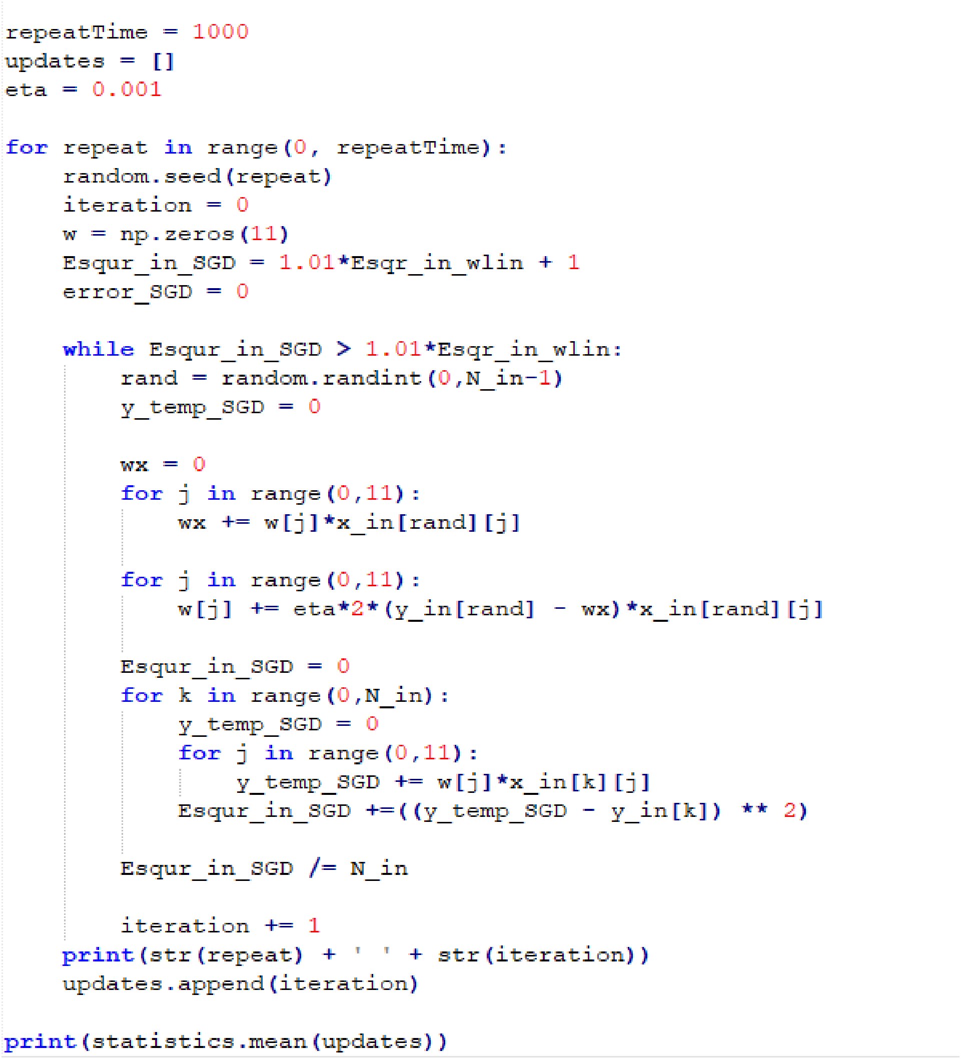


```
import numpy as np
import random
from numpy.linalg import *

x_in = []
with open('hw3_train.dat') as fl:
    line = fl.readlines()
    for l in line:
        x_in.append(np.fromstring("1.00000  "+l, dtype=float, sep=' '))


x_out = []
with open('hw3_test.dat') as fl:
    line = fl.readlines()
    for l in line:
        x_out.append(np.fromstring("1.00000 "+l, dtype=float, sep=' '))


N_in = len(x_in)
N_out = len(x_out)

y_in = []
for i in range(0, N_in):
    y_in.append(x_in[i][11])

y_in = np.array(y_in)
x_in = np.delete(x_in, -1, axis=1)

x_inT = x_in.transpose()

#X^T*X
x_temp1 = x_inT.dot(x_in)
#(X^T*X)^-1
x_temp2 = np.linalg.inv(x_temp1)
#(X^T*X)^-1*X^T
pseu_invX = x_temp2.dot(x_inT)
#(X^T*X)^-1*X^Ty
Wlin = pseu_invX.dot(y_in.transpose())

Esqr_in = 0
error = 0
for i in range(0,N_in):
    y_temp = 0
    for j in range(0,11):
        y_temp += Wlin[j]*x_in[i][j]
    error += ((y_temp - y_in[i]) ** 2)

Esqr_in = error/N_in

print(Esqr_in)
```

15

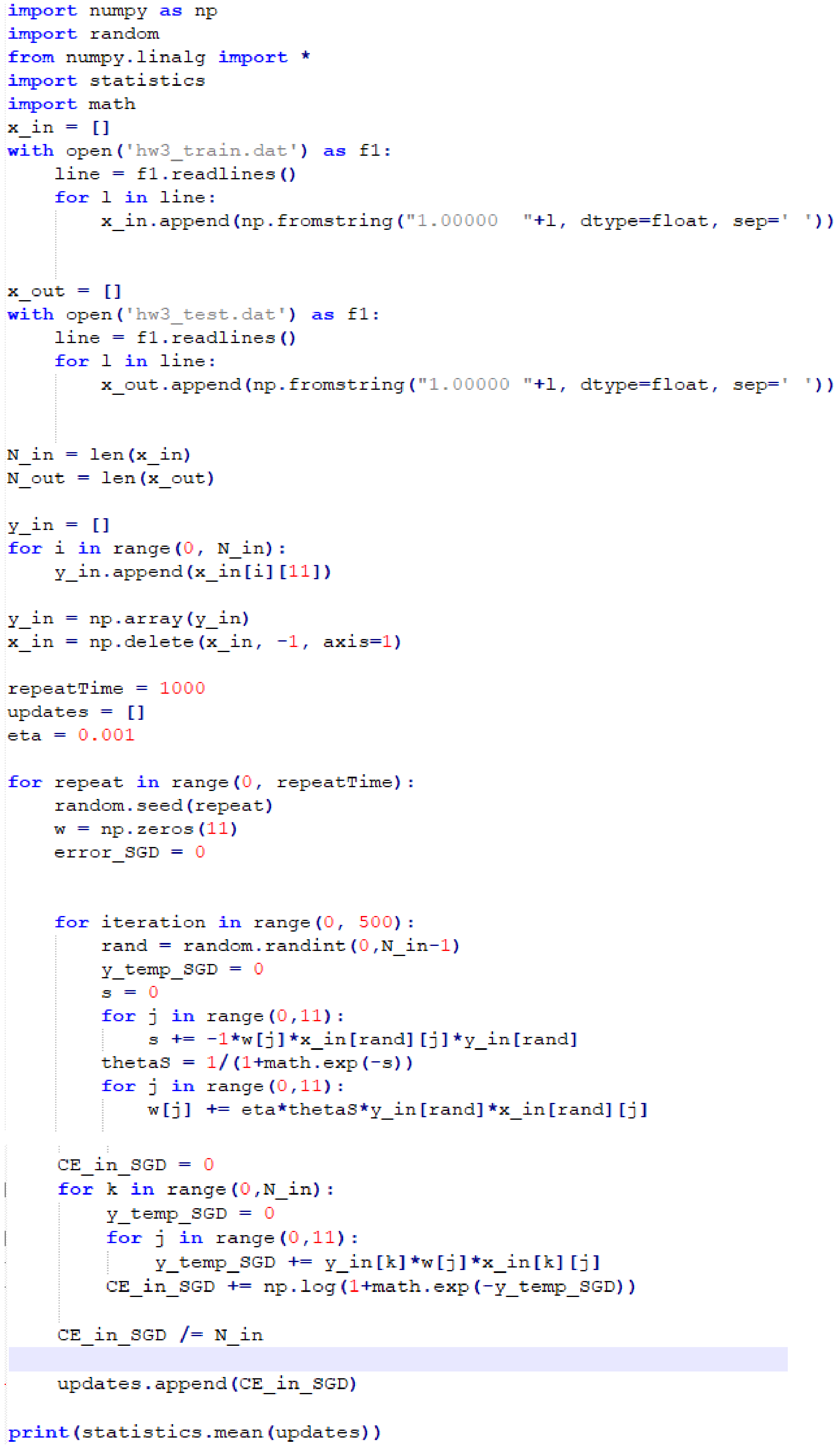


```
import numpy as np
import random
from numpy.linalg import *
import statistics

x_in = []
with open('hw3_train.dat') as f1:
    line = f1.readlines()
    for l in line:
        x_in.append(np.fromstring("1.00000  "+l, dtype=float, sep=' '))


x_out = []
with open('hw3_test.dat') as f1:
    line = f1.readlines()
    for l in line:
        x_out.append(np.fromstring("1.00000 "+l, dtype=float, sep=' '))


N_in = len(x_in)
N_out = len(x_out)

y_in = []
for i in range(0, N_in):
    y_in.append(x_in[i][11])

y_in = np.array(y_in)
x_in = np.delete(x_in, -1, axis=1)

x_inT = x_in.transpose()

x_temp1 = x_inT.dot(x_in)
x_temp2 = np.linalg.inv(x_temp1)
pseu_invX = x_temp2.dot(x_inT)

Wlin = pseu_invX.dot(y_in.transpose())

Esqr_in = 0
error = 0
for i in range(0,N_in):
    y_temp = 0
    for j in range(0,11):
        y_temp += Wlin[j]*x_in[i][j]
    error += ((y_temp - y_in[i]) ** 2)

Esqr_in_wlin = error/N_in
```



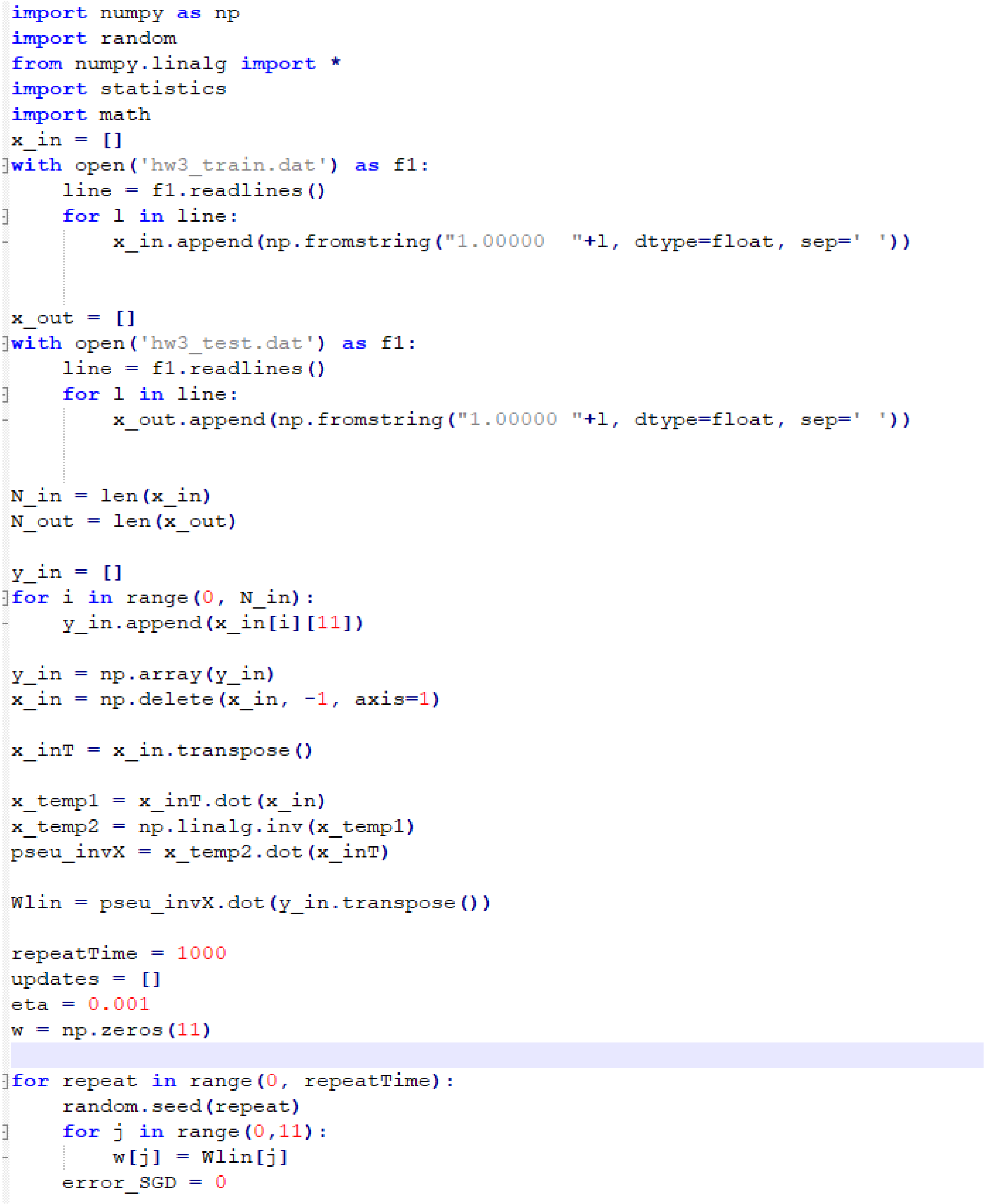
```
repeatTime = 1000
updates = []
eta = 0.001

for repeat in range(0, repeatTime):
    random.seed(repeat)
    iteration = 0
    w = np.zeros(11)
    Esqur_in_SGD = 1.01*Esqr_in_wlin + 1
    error_SGD = 0

    while Esqur_in_SGD > 1.01*Esqr_in_wlin:
        rand = random.randint(0,N_in-1)
        y_temp_SGD = 0

        wx = 0
        for j in range(0,11):
            wx += w[j]*x_in[rand][j]

        for j in range(0,11):
            w[j] += eta*2*(y_in[rand] - wx)*x_in[rand][j]

        Esqur_in_SGD = 0
        for k in range(0,N_in):
            y_temp_SGD = 0
            for j in range(0,11):
                y_temp_SGD += w[j]*x_in[k][j]
            Esqur_in_SGD +=((y_temp_SGD - y_in[k]) ** 2)

        Esqur_in_SGD /= N_in

        iteration += 1
    print(str(repeat) + ' ' + str(iteration))
    updates.append(iteration)

print(statistics.mean(updates))
```

16

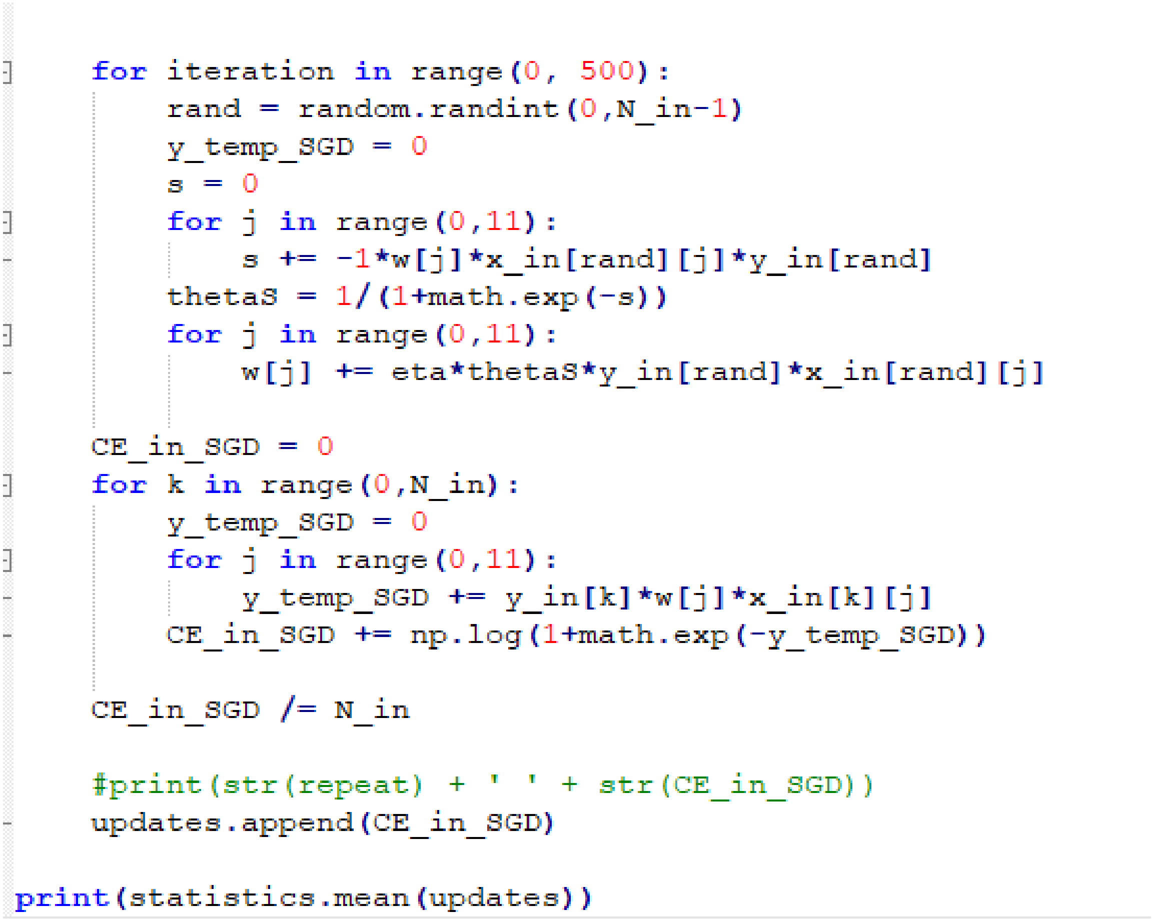


```
import numpy as np
import random
from numpy.linalg import *
import statistics
import math
x_in = []
with open('hw3_train.dat') as f1:
    line = f1.readlines()
    for l in line:
        x_in.append(np.fromstring("1.00000  "+l, dtype=float, sep=' '))


x_out = []
with open('hw3_test.dat') as f1:
    line = f1.readlines()
    for l in line:
        x_out.append(np.fromstring("1.00000 "+l, dtype=float, sep=' '))


N_in = len(x_in)
N_out = len(x_out)

y_in = []
for i in range(0, N_in):
    y_in.append(x_in[i][11])

y_in = np.array(y_in)
x_in = np.delete(x_in, -1, axis=1)

repeatTime = 1000
updates = []
eta = 0.001

for repeat in range(0, repeatTime):
    random.seed(repeat)
    w = np.zeros(11)
    error_SGD = 0


    for iteration in range(0, 500):
        rand = random.randint(0,N_in-1)
        y_temp_SGD = 0
        s = 0
        for j in range(0,11):
            s += -1*w[j]*x_in[rand][j]*y_in[rand]
        thetaS = 1/(1+math.exp(-s))
        for j in range(0,11):
            w[j] += eta*thetaS*y_in[rand]*x_in[rand][j]

    CE_in_SGD = 0
    for k in range(0,N_in):
        y_temp_SGD = 0
        for j in range(0,11):
            y_temp_SGD += y_in[k]*w[j]*x_in[k][j]
        CE_in_SGD += np.log(1+math.exp(-y_temp_SGD))

    CE_in_SGD /= N_in

    updates.append(CE_in_SGD)

print(statistics.mean(updates))
```

17

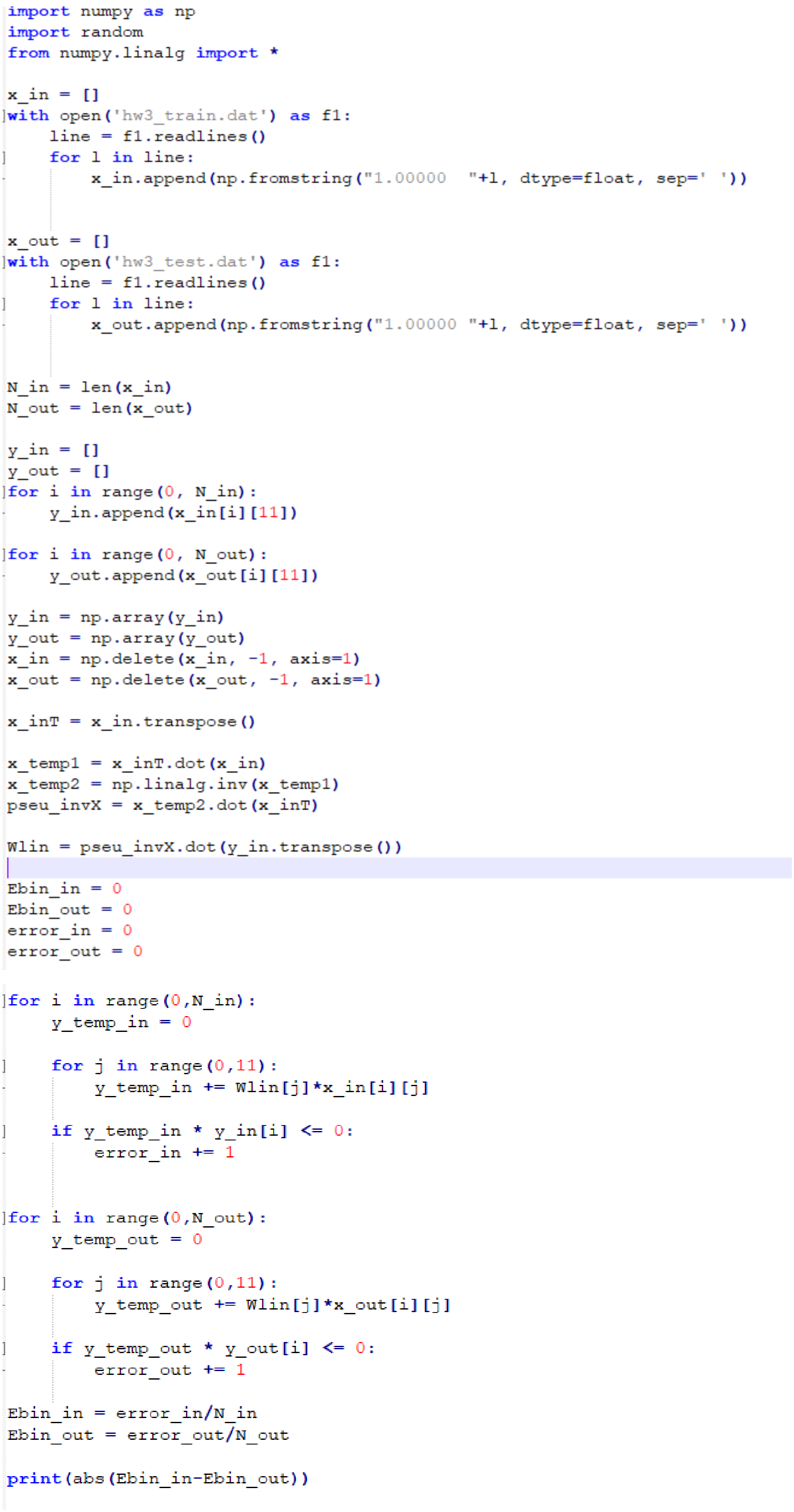


```
import numpy as np
import random
from numpy.linalg import *
import statistics
import math
x_in = []
with open('hw3_train.dat') as f1:
    line = f1.readlines()
    for l in line:
        x_in.append(np.fromstring("1.00000  "+l, dtype=float, sep=' '))


x_out = []
with open('hw3_test.dat') as f1:
    line = f1.readlines()
    for l in line:
        x_out.append(np.fromstring("1.00000 "+l, dtype=float, sep=' '))


N_in = len(x_in)
N_out = len(x_out)

y_in = []
for i in range(0, N_in):
    y_in.append(x_in[i][11])

y_in = np.array(y_in)
x_in = np.delete(x_in, -1, axis=1)

x_inT = x_in.transpose()

x_temp1 = x_inT.dot(x_in)
x_temp2 = np.linalg.inv(x_temp1)
pseu_invX = x_temp2.dot(x_inT)

Wlin = pseu_invX.dot(y_in.transpose())

repeatTime = 1000
updates = []
eta = 0.001
w = np.zeros(11)

for repeat in range(0, repeatTime):
    random.seed(repeat)
    for j in range(0,11):
        w[j] = Wlin[j]
    error_SGD = 0
```



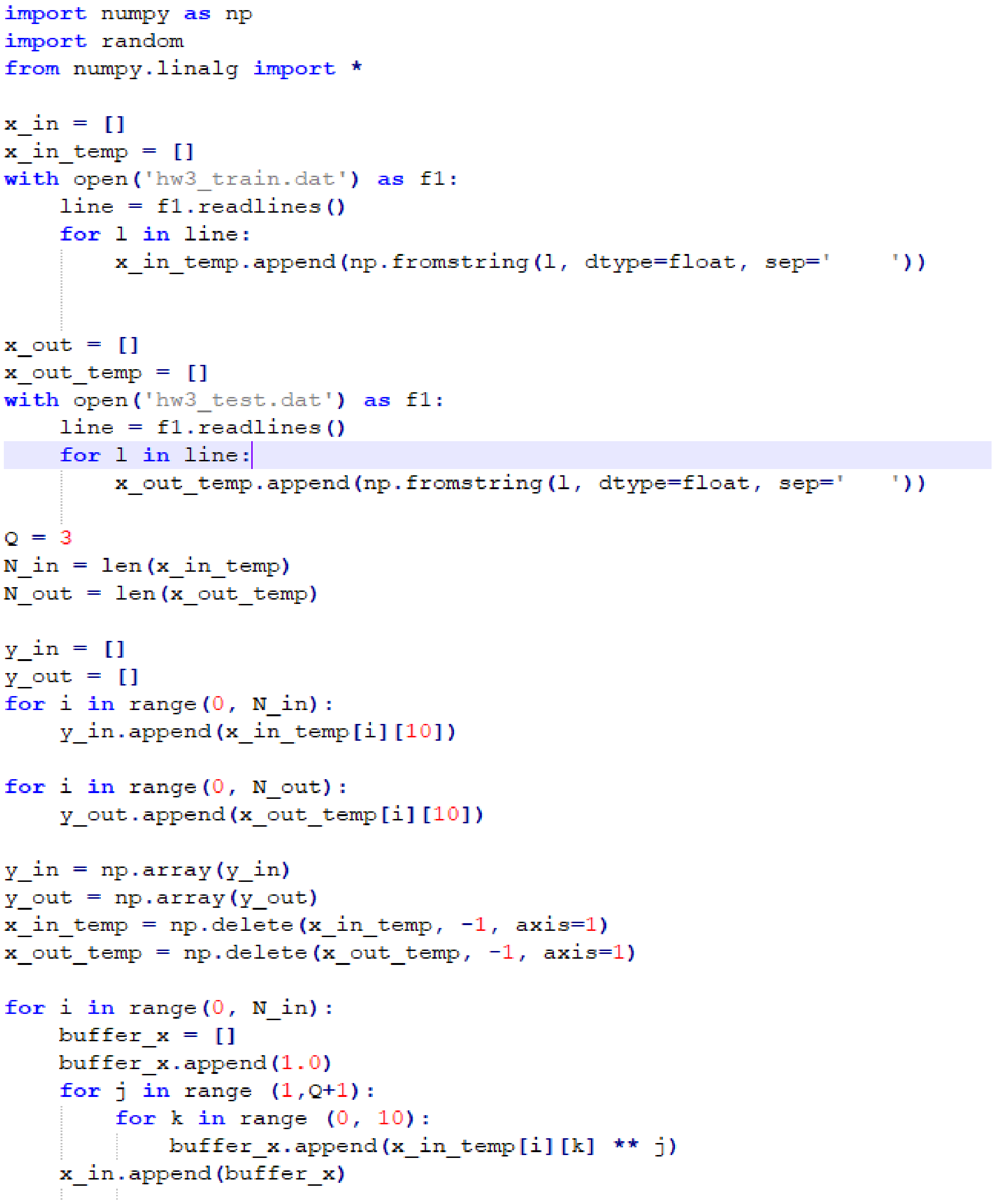
```
    for iteration in range(0, 500):
        rand = random.randint(0,N_in-1)
        y_temp_SGD = 0
        s = 0
        for j in range(0,11):
            s += -1*w[j]*x_in[rand][j]*y_in[rand]
        thetaS = 1/(1+math.exp(-s))
        for j in range(0,11):
            w[j] += eta*thetaS*y_in[rand]*x_in[rand][j]

    CE_in_SGD = 0
    for k in range(0,N_in):
        y_temp_SGD = 0
        for j in range(0,11):
            y_temp_SGD += y_in[k]*w[j]*x_in[k][j]
        CE_in_SGD += np.log(1+math.exp(-y_temp_SGD))

    CE_in_SGD /= N_in

    #print(str(repeat) + ' ' + str(CE_in_SGD))
    updates.append(CE_in_SGD)

print(statistics.mean(updates))
```

18

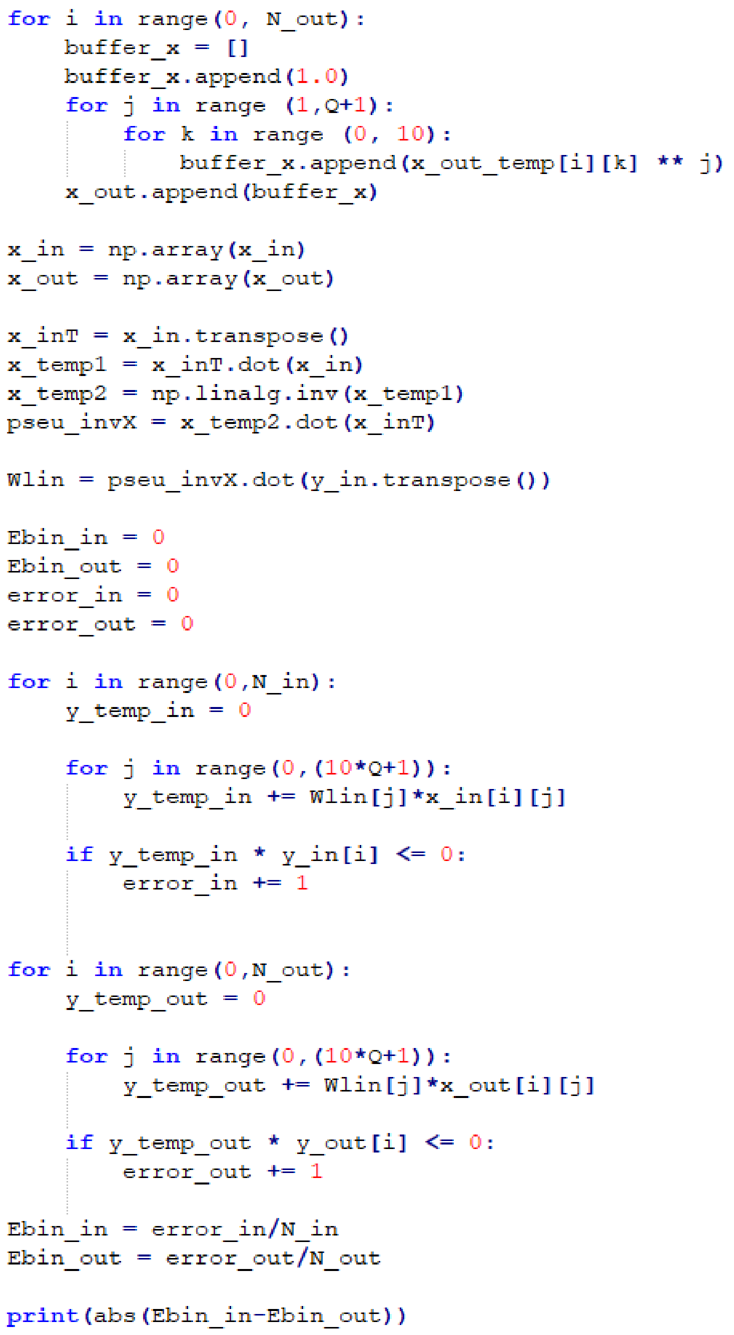


```python
import numpy as np
import random
from numpy.linalg import *

x_in = []
with open('hw3_train.dat') as f1:
    line = f1.readlines()
    for l in line:
        x_in.append(np.fromstring("1.00000  "+l, dtype=float, sep=' '))


x_out = []
with open('hw3_test.dat') as f1:
    line = f1.readlines()
    for l in line:
        x_out.append(np.fromstring("1.00000 "+l, dtype=float, sep=' '))


N_in = len(x_in)
N_out = len(x_out)

y_in = []
y_out = []
for i in range(0, N_in):
    y_in.append(x_in[i][11])

for i in range(0, N_out):
    y_out.append(x_out[i][11])

y_in = np.array(y_in)
y_out = np.array(y_out)
x_in = np.delete(x_in, -1, axis=1)
x_out = np.delete(x_out, -1, axis=1)

x_inT = x_in.transpose()

x_temp1 = x_inT.dot(x_in)
x_temp2 = np.linalg.inv(x_temp1)
pseu_invX = x_temp2.dot(x_inT)

Wlin = pseu_invX.dot(y_in.transpose())

Ebin_in = 0
Ebin_out = 0
error_in = 0
error_out = 0

for i in range(0,N_in):
    y_temp_in = 0

    for j in range(0,11):
        y_temp_in += Wlin[j]*x_in[i][j]

    if y_temp_in * y_in[i] <= 0:
        error_in += 1


for i in range(0,N_out):
    y_temp_out = 0

    for j in range(0,11):
        y_temp_out += Wlin[j]*x_out[i][j]

    if y_temp_out * y_out[i] <= 0:
        error_out += 1

Ebin_in = error_in/N_in
Ebin_out = error_out/N_out

print(abs(Ebin_in-Ebin_out))
```

19

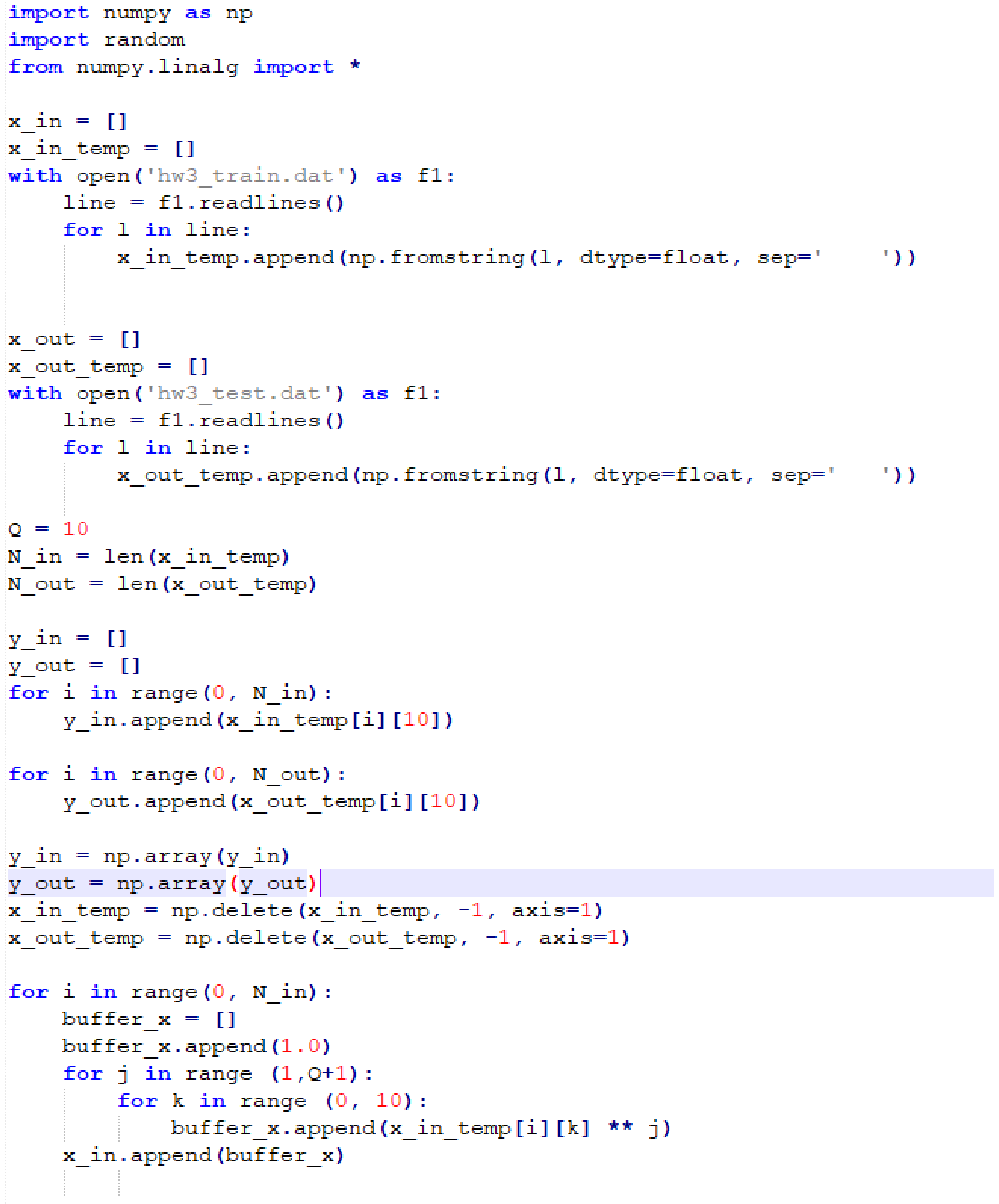


```
import numpy as np
import random
from numpy.linalg import *

x_in = []
x_in_temp = []
with open('hw3_train.dat') as f1:
    line = f1.readlines()
    for l in line:
        x_in_temp.append(np.fromstring(l, dtype=float, sep='    '))


x_out = []
x_out_temp = []
with open('hw3_test.dat') as f1:
    line = f1.readlines()
    for l in line:
        x_out_temp.append(np.fromstring(l, dtype=float, sep='   '))

Q = 3
N_in = len(x_in_temp)
N_out = len(x_out_temp)

y_in = []
y_out = []
for i in range(0, N_in):
    y_in.append(x_in_temp[i][10])

for i in range(0, N_out):
    y_out.append(x_out_temp[i][10])

y_in = np.array(y_in)
y_out = np.array(y_out)
x_in_temp = np.delete(x_in_temp, -1, axis=1)
x_out_temp = np.delete(x_out_temp, -1, axis=1)

for i in range(0, N_in):
    buffer_x = []
    buffer_x.append(1.0)
    for j in range (1,Q+1):
        for k in range (0, 10):
            buffer_x.append(x_in_temp[i][k] ** j)
    x_in.append(buffer_x)
```

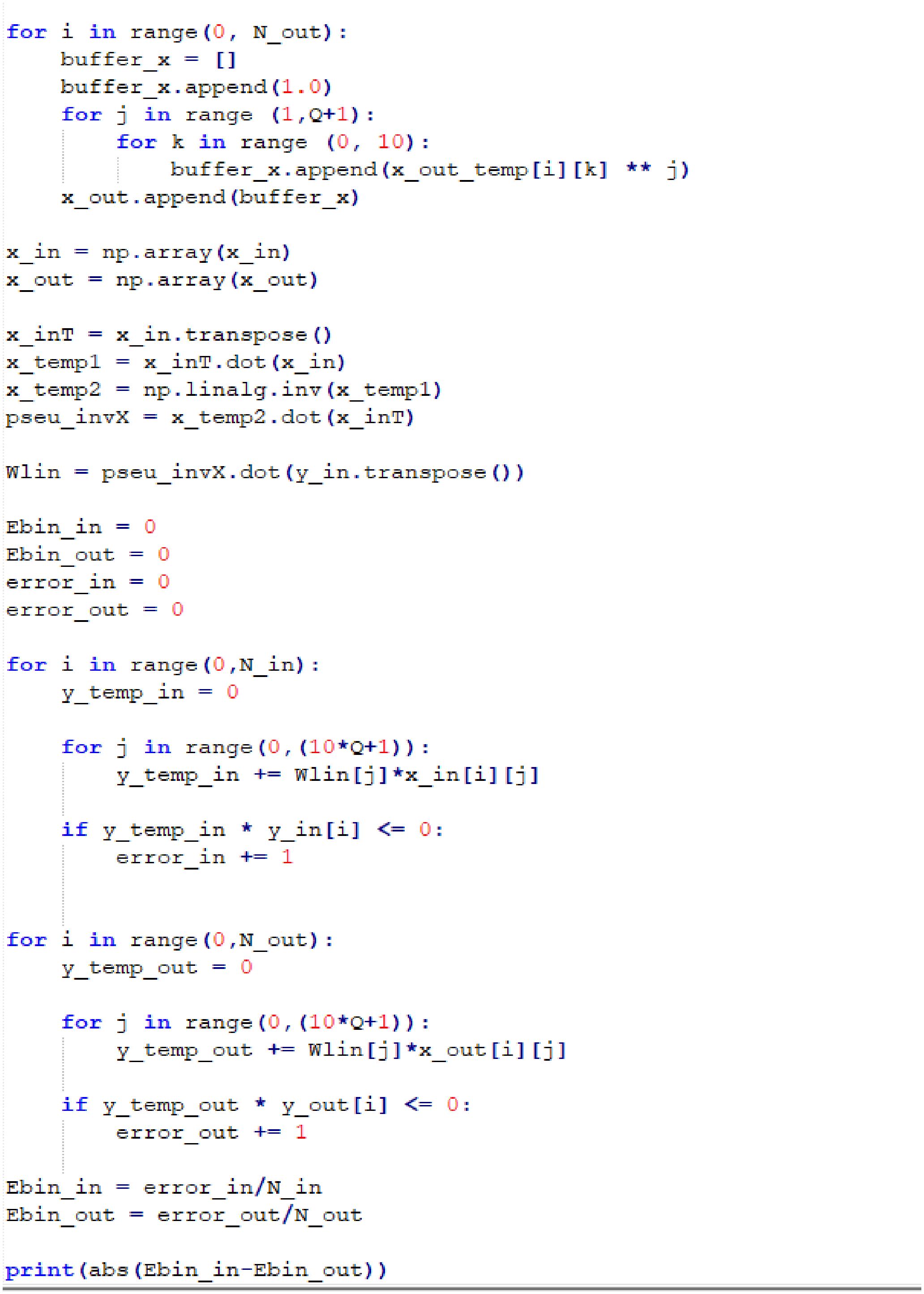


```
for i in range(0, N_out):
    buffer_x = []
    buffer_x.append(1.0)
    for j in range (1,Q+1):
        for k in range (0, 10):
            buffer_x.append(x_out_temp[i][k] ** j)
    x_out.append(buffer_x)

x_in = np.array(x_in)
x_out = np.array(x_out)

x_inT = x_in.transpose()
x_temp1 = x_inT.dot(x_in)
x_temp2 = np.linalg.inv(x_temp1)
pseu_invX = x_temp2.dot(x_inT)

Wlin = pseu_invX.dot(y_in.transpose())

Ebin_in = 0
Ebin_out = 0
error_in = 0
error_out = 0

for i in range(0,N_in):
    y_temp_in = 0

    for j in range(0,(10*Q+1)):
        y_temp_in += Wlin[j]*x_in[i][j]

    if y_temp_in * y_in[i] <= 0:
        error_in += 1


for i in range(0,N_out):
    y_temp_out = 0

    for j in range(0,(10*Q+1)):
        y_temp_out += Wlin[j]*x_out[i][j]

    if y_temp_out * y_out[i] <= 0:
        error_out += 1

Ebin_in = error_in/N_in
Ebin_out = error_out/N_out

print(abs(Ebin_in-Ebin_out))
```

20



```
import numpy as np
import random
from numpy.linalg import *

x_in = []
x_in_temp = []
with open('hw3_train.dat') as f1:
    line = f1.readlines()
    for l in line:
        x_in_temp.append(np.fromstring(l, dtype=float, sep='    '))


x_out = []
x_out_temp = []
with open('hw3_test.dat') as f1:
    line = f1.readlines()
    for l in line:
        x_out_temp.append(np.fromstring(l, dtype=float, sep='   '))

Q = 10
N_in = len(x_in_temp)
N_out = len(x_out_temp)

y_in = []
y_out = []
for i in range(0, N_in):
    y_in.append(x_in_temp[i][10])

for i in range(0, N_out):
    y_out.append(x_out_temp[i][10])

y_in = np.array(y_in)
y_out = np.array(y_out)
x_in_temp = np.delete(x_in_temp, -1, axis=1)
x_out_temp = np.delete(x_out_temp, -1, axis=1)

for i in range(0, N_in):
    buffer_x = []
    buffer_x.append(1.0)
    for j in range (1,Q+1):
        for k in range (0, 10):
            buffer_x.append(x_in_temp[i][k] ** j)
    x_in.append(buffer_x)
```



```
for i in range(0, N_out):
    buffer_x = []
    buffer_x.append(1.0)
    for j in range (1,Q+1):
        for k in range (0, 10):
            buffer_x.append(x_out_temp[i][k] ** j)
    x_out.append(buffer_x)

x_in = np.array(x_in)
x_out = np.array(x_out)

x_inT = x_in.transpose()
x_temp1 = x_inT.dot(x_in)
x_temp2 = np.linalg.inv(x_temp1)
pseu_invX = x_temp2.dot(x_inT)

Wlin = pseu_invX.dot(y_in.transpose())

Ebin_in = 0
Ebin_out = 0
error_in = 0
error_out = 0

for i in range(0,N_in):
    y_temp_in = 0

    for j in range(0,(10*Q+1)):
        y_temp_in += Wlin[j]*x_in[i][j]

    if y_temp_in * y_in[i] <= 0:
        error_in += 1


for i in range(0,N_out):
    y_temp_out = 0

    for j in range(0,(10*Q+1)):
        y_temp_out += Wlin[j]*x_out[i][j]

    if y_temp_out * y_out[i] <= 0:
        error_out += 1

Ebin_in = error_in/N_in
Ebin_out = error_out/N_out

print(abs(Ebin_in-Ebin_out))
```