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 [ ]:
```

```