4.5

(a)

```
[ 0.94520006  0.01974237 -0.01364498  0.04678134]
```

(b)

for 2000: MSE = 13918.63

for 2001: MSE = 2973.02

(c)

```
import numpy as np

def calculate(array):
    A = []
    B = []

    for i in range(4):
        B.append(float(0))
        A.append([])
        for j in range(4):
            A[i].append(float(0))

    for i in range(4):
        for j in range(4):
            cur = 0.0
            for k in range(4, len(array)):
                cur += array[k-j-1]*array[k-i-1]
            A[i][j] = cur

    for i in range(4):
        cur = 0.0
        for k in range(4,len(array)):
            cur += array[k] * array[k-i-1]
        B[i] = cur

    A = np.array(A,float)
    B = np.array(B,float)
    X = np.dot(np.linalg.inv(A) , B)
    return X

def error(array,X):
    Y = []
    for i in range(4, len(array)):
        cur = X[0]*array[i-1] + X[1]*array[i-2] + X[2]*array[i-3] + X[3]*array[i-4]
        Y.append(cur)

    e = 0.0
    for i in range(len(Y)):
        e += (Y[i] - array[i+4]) **2
    e /= len(Y)
    print e



fp = open('nasdaq00.txt', 'r')
#fp = open('nasdaq01.txt', 'r')
array2000 = []
for line in fp.readlines():
    line = line.strip('\n').split(' ')
    array2000.append(float(line[0]))

X = []
X = calculate(array2000)
print X

error(array2000,X)
```

4.6

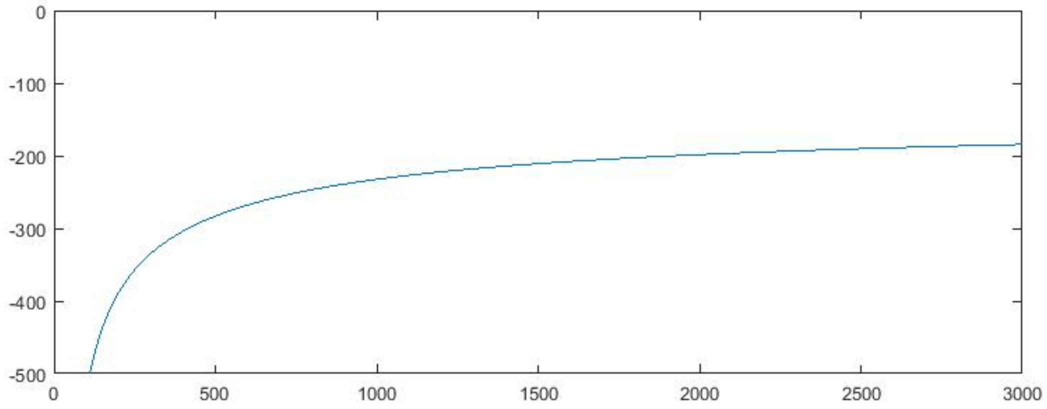
(a)

error rate for train3: 3.57%
error rate for train5: 3.85%
error rate overall: 3.71%

```
w =

    0.6971   -0.1080   -1.0490   -0.7983   -0.4133   -0.7576   -0.4498   -0.0479
    0.8010   -0.0417   -1.1509   -0.9043   -0.3120    0.2457   -0.1742   -0.4225
    0.9076   -0.3412   -0.7329   -0.6148    0.1957    0.1171   -0.3187   -0.3872
    1.0184    0.1839   -0.3500    0.1273    0.2159   -0.3595   -0.3523   -0.6077
    1.0018   -0.0668   -0.1303    0.5902    0.3105   -0.1145   -0.0138   -0.3400
   -0.2049   -0.4751    1.3796    0.7316    0.2290    0.2519    0.1153   -0.5644
   -0.9889    0.6730    2.5369   -0.1523    0.0097   -0.2046    0.0092    0.2550
   -1.4022    0.6319    1.9222    0.2368    0.3302    0.8040    0.7367    0.1802
```

The converge of L using gradient:



(b)

error rate for test3: 5.25%
error rate for test5: 6.75%
error rate overall:6.00%

(c)

```
%0 stands for 5
%1 stands for 3
%sigmf(x,[1 0])
Train3=load('Train3.txt');
Train5=load('Train5.txt');
Test3 =load('Test3.txt');
Test5 =load('Test5.txt');

% initial values of weight.
w= 0.5 * ones(64,1);
L0= sum(log(sigmf(Train3 * w,[1 0])))+sum(log(sigmf(-Train5 * w,[1 0])));
Y1 = 1* ones(700,1);
Y2 = 0* ones(700,1);
Gradient= Train3' *(Y1 - sigmf(Train3*w,[1,0])) + Train5' *(Y2 - sigmf(Train5 *w,[1 0]));
Lr=ones(10000,1);

for i= 1:10000
   w=w+0.02/1400*Gradient;
   Gradient= Train3' *(Y1 - sigmf(Train3*w,[1,0])) + Train5' *(Y2 - sigmf(Train5 *w,[1 0]));
   Lr(i)=sum(log(sigmf(Train3 * w,[1 0])))+sum(log(sigmf(-Train5 * w,[1 0])));
end
sigmf(Train3 *w,[1 0])
L= sum(log(sigmf(Train3 * w,[1 0])))+sum(log(sigmf(-Train5 * w,[1 0])));

T3=sigmf(Train3 * w,[1 0]);
T5=sigmf(Train5 * w,[1 0]);
R3=sigmf(Test3 * w,[1 0]);
R5=sigmf(Test5 * w,[1 0]);
c3_5=0;
t3_5=0;
c5_3=0;
t5_3=0;
for i= 1:400
    if(R3(i)<=0.5)
        c3_5=c3_5+1;
    end
    if(R5(i)>0.5)
        c5_3=c5_3+1;
    end
end
for i= 1:700
    if(T3(i)<=0.5)
        t3_5=t3_5+1;
    end
    if(T5(i)>0.5)
```

```
        t5_3=t5_3+1;
    end
end
fprintf('Error rate of train3: %f\n',t3_5/700);
fprintf('Error rate of train5: %f\n',t5_3/700);
fprintf('Error rate overall:%f\n',t3_5/1400+t5_3/1400);
fprintf('Error rate of test3: %f\n',c3_5/400);
fprintf('Error rate of test5: %f\n',c5_3/400);
fprintf('Error test rate overall:%f\n',c3_5/800+c5_3/800);

x=1:10000;
y=Lr(x);
plot(x,y)

w= reshape(w, [8,8])
```