Question 2:

a)

function[stump\_error, S] = stump(x,y,w)

d = size(x,1);

n = size(x,2);

error\_col = zeros(1,d);

error\_row = zeros(n,1);

yhat = zeros(n,1);

Svector = zeros(1,d);

for j = 1:d

for i = 1:n

indicator = abs(x(j,:)') >= abs(x(j,i));

yhat(indicator) = 1;

yhat(~indicator) = -1;

indicator2 = (y ~= yhat);

error\_row(i) = w\*indicator2/sum(w);

end

[error\_col(j),number] = min(error\_row);

Svector(j) = x(j,number);

end

[stump\_error,num]= min(error\_col);

S = Svector(num);

end

%generate training set

ntrain=2000;

ntest=10000;

Xtrain=zeros(10,ntrain);

ytrain=zeros(ntrain,1);

for i=1:ntrain

Xtrain(:,i)=normrnd(0,1,1,10)';

end

for i=1:ntrain

if sum(Xtrain(:,i).^2)>9.34

ytrain(i)=1;

else

ytrain(i)=-1;

end

end

%generate test set

Xtest=zeros(10,ntest);

ytest=zeros(ntest,1);

for i=1:ntest

Xtest(:,i)=normrnd(0,1,1,10)';

end

for i=1:ntest

if sum(Xtest(:,i).^2)>9.34

ytest(i)=1;

else

ytest(i)=-1;

end

end

numberofiterations=250;

weights= ones(1,ntrain)/ntrain;

alpha=zeros(1,numberofiterations);

final\_train=zeros(numberofiterations,ntrain);

final\_test=zeros(numberofiterations,ntest);

train\_error=zeros(1,numberofiterations);

test\_error=zeros(1,numberofiterations);

for i=1:numberofiterations

%model

[L, S] = searchstump(Xtrain,ytrain,weights);

[row,~] = find(Xtrain == S);

alpha = log((1-L)/(L+0.001));

%train

train\_hat=zeros(ntrain,1);

for j=1:ntrain

if Xtrain(row,j)>=S

train\_hat(j)=1;

else

train\_hat(j)=-1;

end

end

%test

test\_hat=zeros(ntest,1);

for j=1:ntest

if Xtest(row,j)>=S

test\_hat(j)=1;

else

test\_hat(j)=-1;

end

end

weights = weights .\* (exp(alpha \* (train\_hat~=ytrain)))';

weights = weights ./ sum(weights);

final\_train(i,:)=(alpha\*train\_hat)';

final\_test(i,:)=(alpha\*test\_hat)';

H\_train=sum(final\_train,1);

H\_test=sum(final\_test,1);

H\_train(H\_train>=0)=1;

H\_train(H\_train<0)=-1;

H\_test(H\_test>=0)=1;

H\_test(H\_test<0)=-1;

train\_error(i)=sum(H\_train'~=ytrain)/ntrain;

test\_error(i)=sum(H\_test'~=ytest)/ntest;

end

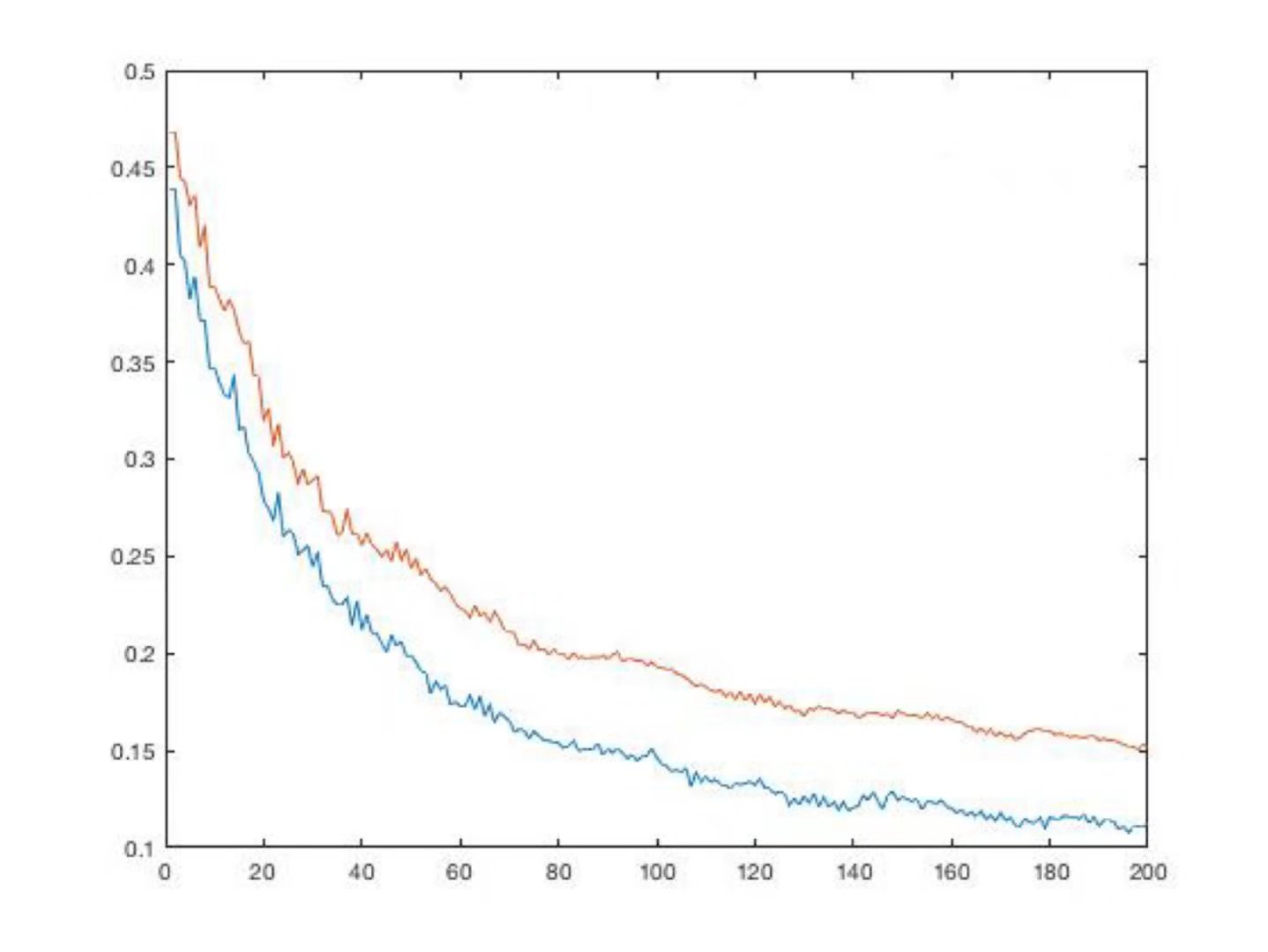
plot(train\_error)

hold on;

plot(test\_error)

hold off;

(b)



wise notice that the training error and test error are all decreasing. And training error is always smaller than test error.

(c)

With the number of iteration increasing, the test error should decrease first and then decrease. But from the above graph, we can conclude in a long round, there’s no rise at all. Adaboost is robust to overfitting. So it hard to decide the number of iterations needed to make the test error finally start to rise.