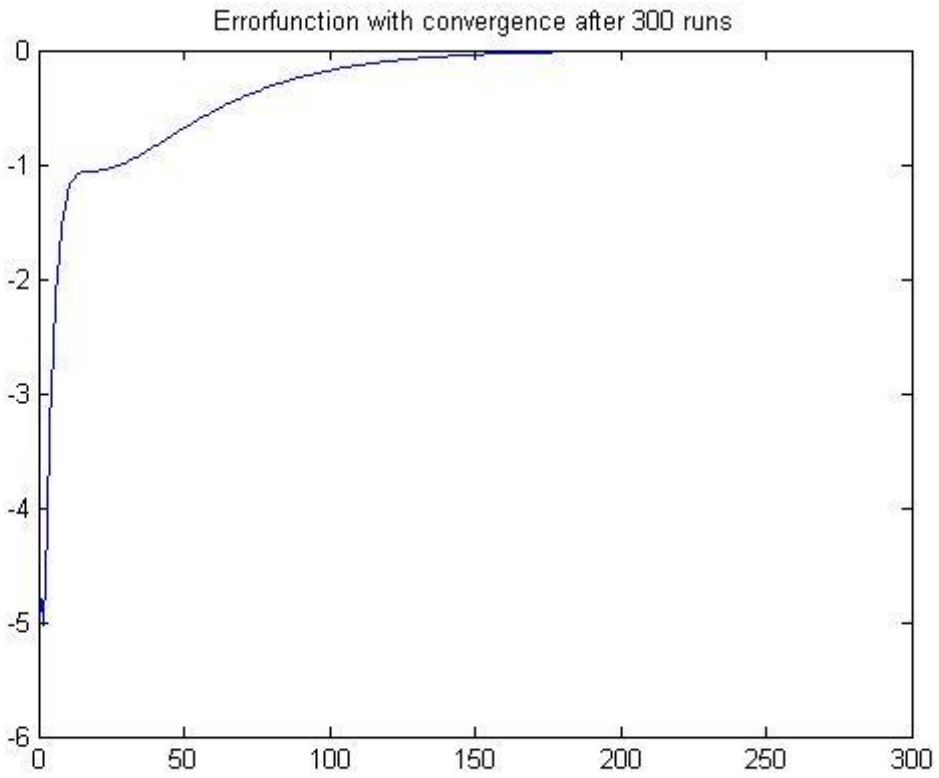
Exercise 3 – Machine Intelligence I

Matalb - Code:

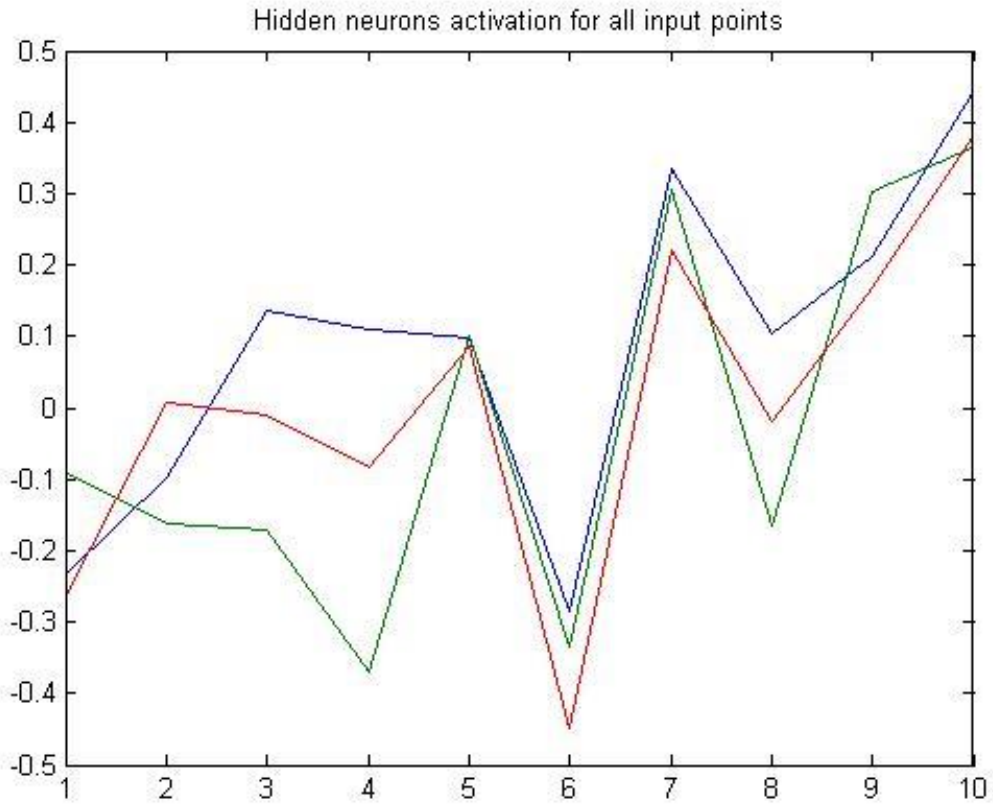
```
clear all
dat = load('RegressionData.txt');
dim = size(dat);
%% Initialization
%weigths and bias
% randomly from intervall [-0.5,0.5]
r = rand(33,1) -0.5;
w = reshape(r(1:(dim(1)+1)*3),dim(1)+1,3);
%learning rate
eta = 0.5;
%gradient descent loop
gdl = 1;
%define input including bias
x = [1;dat(:,1)];
% labels y_orig
y_orig = [1;dat(:,2)];
w = reshape(r(1:(dim(1)+1)*3),dim(1)+1,3);
for gdl = 1:3000
        for j = 1:3                                                     %hidden layers loop
                for i = 1:dim(1)+1                              %input dimension loop
                        n(i,j) = tanh(x(i))*w(i,j);
                        end
                end
y = sum(n');
                y = y';
% quadratic error function
                for j = 1:3                                                             %hidden layers loop
                        for i = 1:dim(1)+1
                        w(i,j) = w(i,j) - eta*(y(i)-y_orig(i))*(1-tanh((x(i)))^2);%*w(i,j);
                        end
                end
% local errors
        for j = 1:3
                for i = 1:dim(1)+1
                E_loc(i,j) = (y(i)-y_orig(i))*(1-tanh((x(i)))^2)*w(i,j);
                end
        end
                E(gdl) = sum(sum(E_loc));
                E(gdl)
                        if abs(E(gdl)) < 0.00001
                                break
                        end
        end
```

```
% a)  visualize errorfunction

figure(1)
plot(E)
title(['Errorfunction with convergence after ',num2str(gdl),' runs'])
```

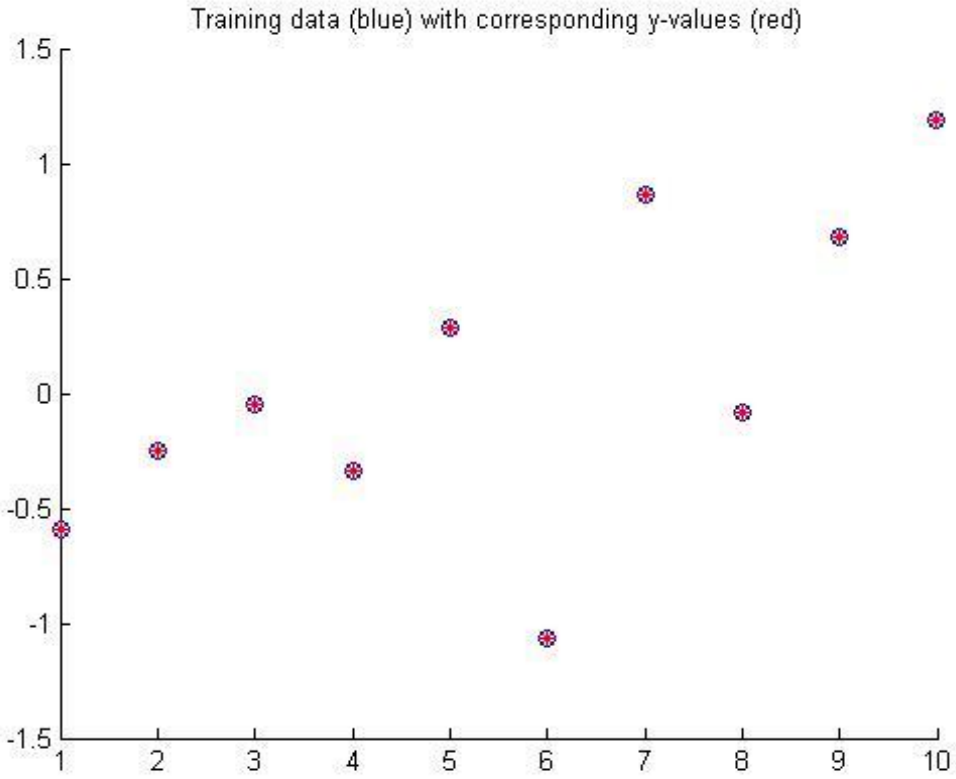


```
% b)
figure(2)
plot(n(2:11,:))
title('Hidden neurons activation for all input points')
```



% c)

```
figure(3)
scatter(1:10,dat(:,2))
hold on
scatter(1:10,y(2:11),'r*')
title('Training data (blue) with corresponding y-values (red)')
hold off
```



% d)

The motivation for the quadratic cost function is given by the linearity of the output neuron.