Daniel Campos Dec 5th, 2013

RCS: Campod2 RIN: 660996361

Machine Learning CS 4100

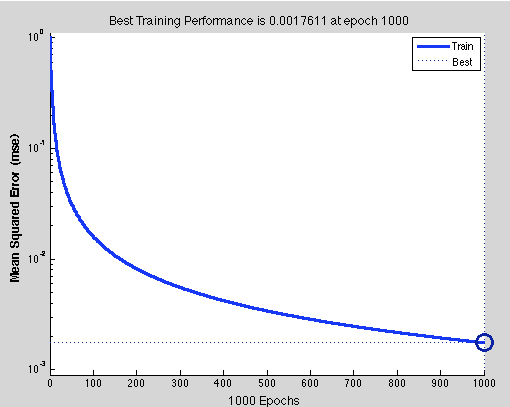
I worked on these problems with Zoe Konrad☺

Problem Set 12

1. (300) Neural Networks and Backpropagation
   1. Tanh

Gradient W1=[-0.0265242, -.00265242,-0.0265242; -0.0265242, -.00265242,-0.0265242]

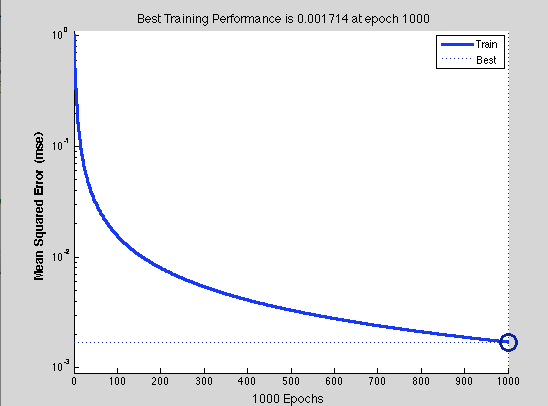
Gradient W2=[-.1781433;-.113221;-.113221]



Identity

Gradient W1=[-0.0462, -.00462,-0.0462; -0.0462, -.00462,-0.0462]

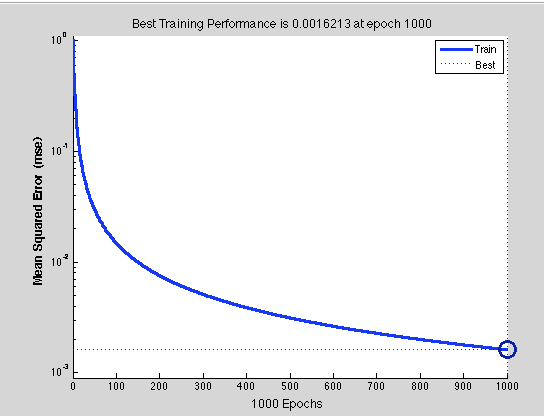
Gradient W2=[-.185456;-.140292;-.140292]



* 1. Tanh

Gradient W1=[-0.0265467, -.00265467,-0.0265467; -0.0265467, -.00265467,-0.0265467]

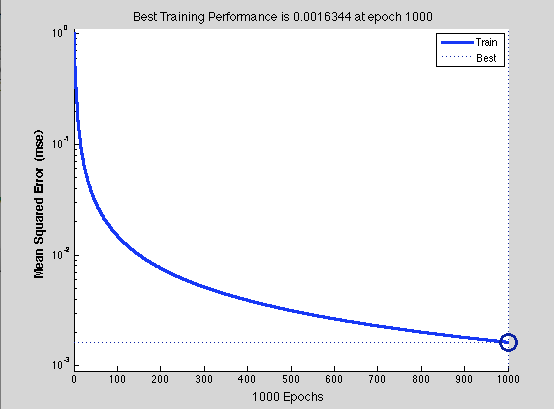
Gradient W2=[-.1781923;-.113521;-.113521]



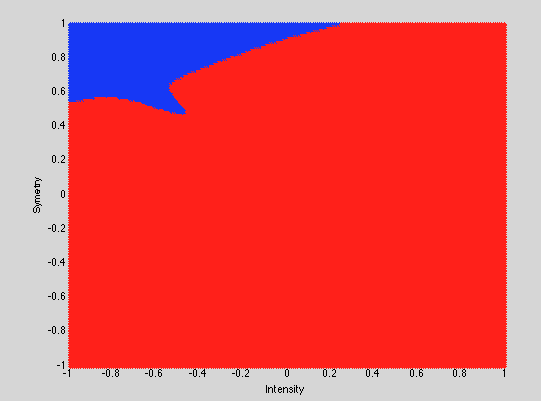
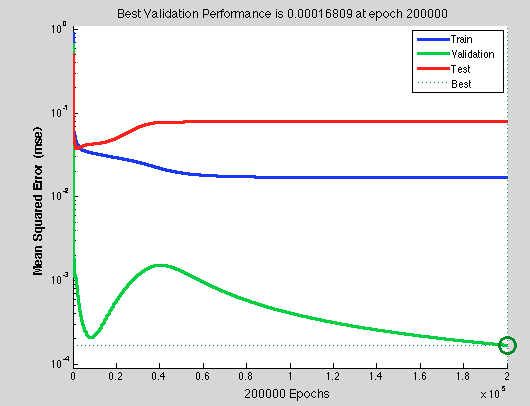
Identity

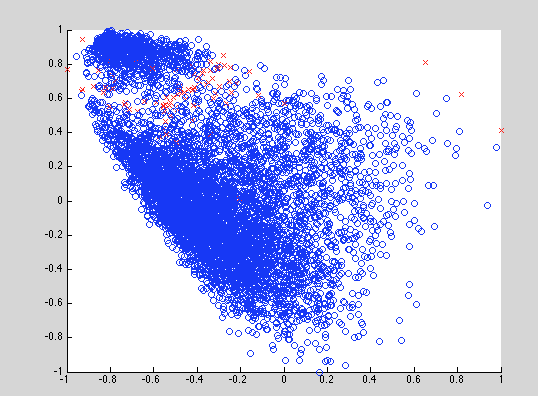
Gradient W1=[-0.0462, -.00462,-0.0462; -0.0462, -.00462,-0.0462]

Gradient W2=[-.1856534;-.140664;-.140664]



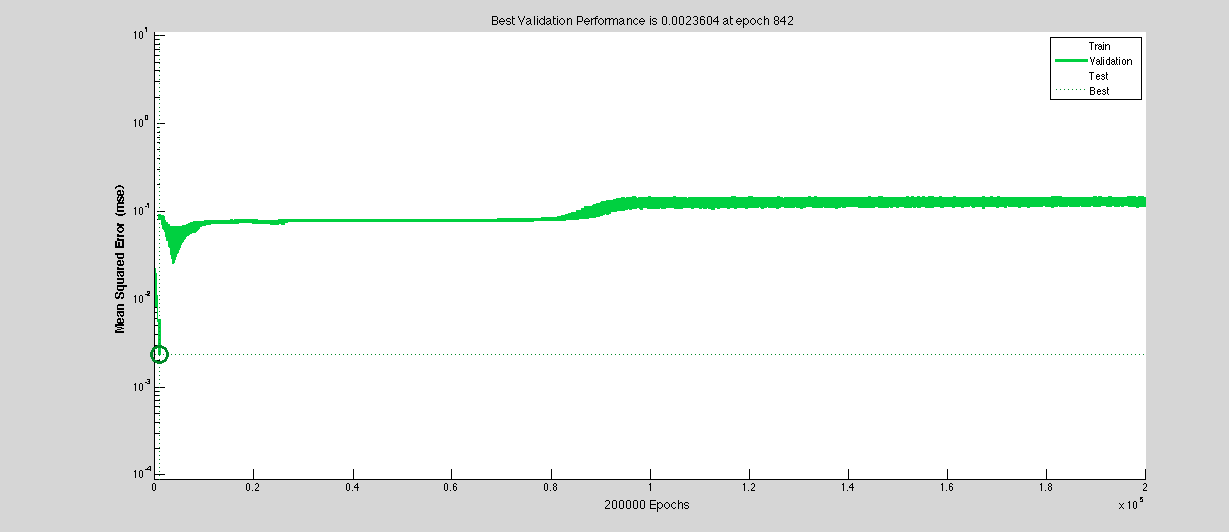
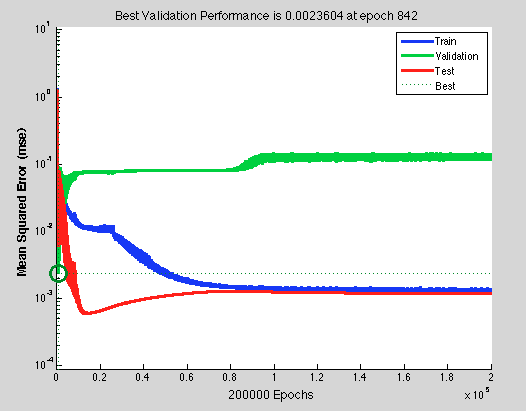
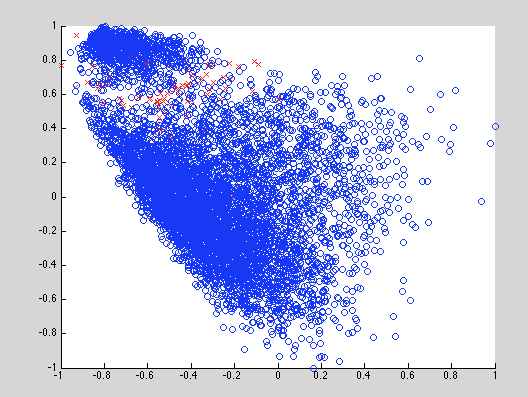
1. 600) Neural Network for Digits
   1. Take only the green line, ignore the other two



* 1. 

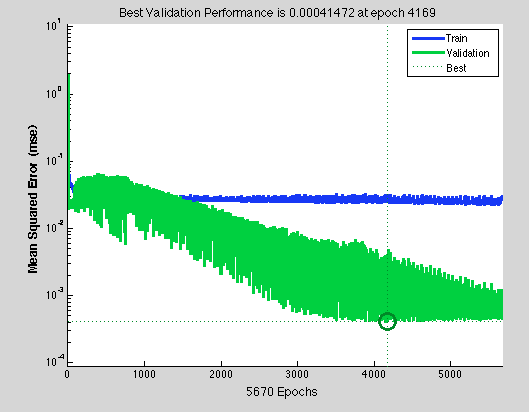
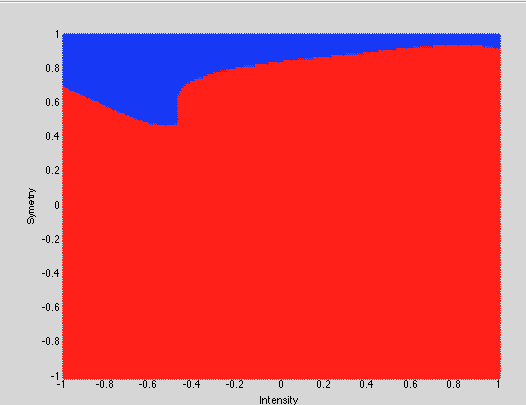
Etest= 0.0542

Ein= 0.0235

* 1. Using weight decay and variable learning rate

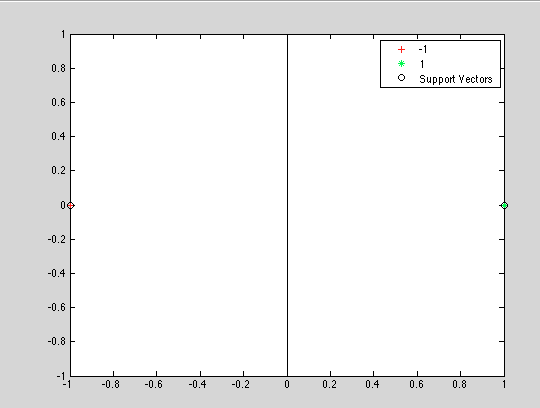
ein = 0.0342

etest= 0.0466

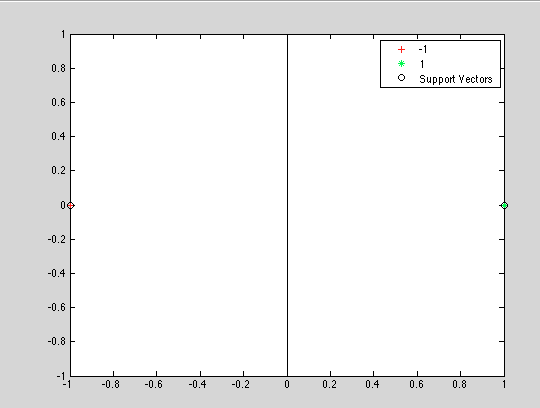
* 1. Using early stopping based on validation set

testError = .056

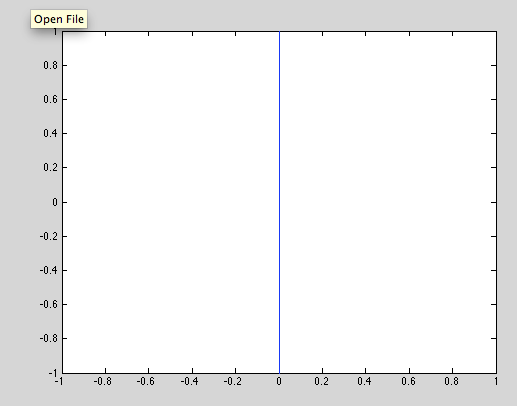
trainError =0.0320

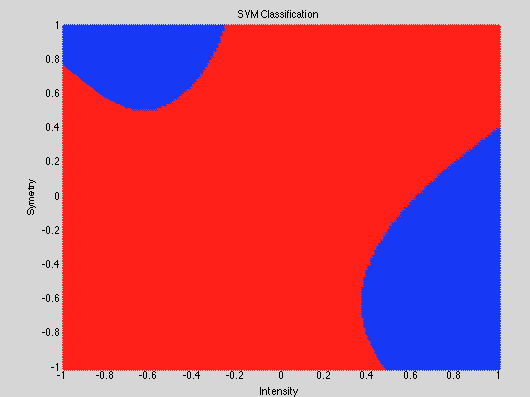
1. (300) Support Vector Machines
   1. 

For a plane to fit between the two points x1, x2, The distance is X1+distance to x2< distance (x1,x2). If we want to minimize the cushion we divide by two. The line would be x1=0. After using the svm function it matlab we had this as the optimal plot. The equation of the optimal hyper plane is x=0. This allows the points to be optimally separated by a hyper plane.

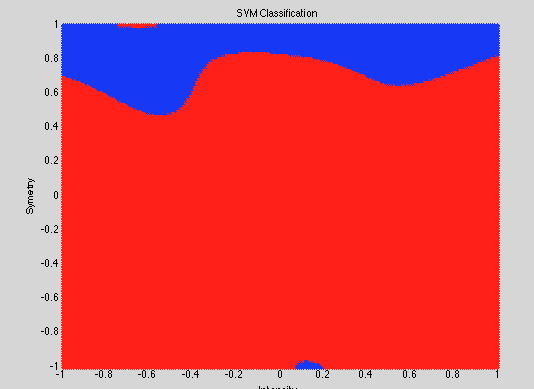
* 1. 1. The data points in this space are actually the same as originally since we had values of 1,0, and -1. X1=(1,0) X2(-1,0)
     2. 

as our z does not change from our x our plane stays the same in the Z dimension thus our line is Z1=0; By this we have that X1^3-X2=0. Slightly different

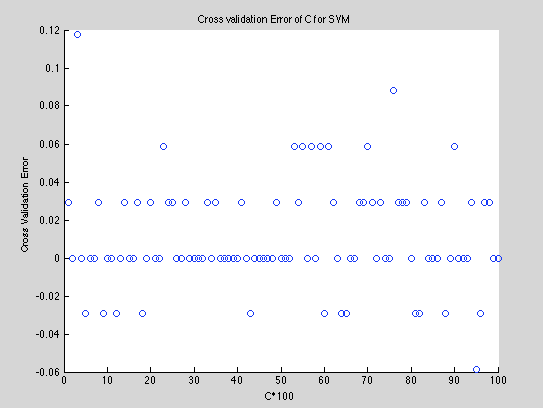
* 1. 
  2. (x1^3-x2)(y1^3-y2)+x1x2y1y2
  3. g(x)=sign(x1^3-x2)

1. (600) SVM with digits data
   1. 

C=10 Small



C=1000Large

* 1. With respect to C we can see some differences in what our models look like. A higher C gives a more complex decision boundary. There is also a higher penalty for violating margins with a high C
  2. C=9500

Lowest cv error at C=950

With cvError= 0.0588

And eTest=.0652

1. (200) Compare Methods: Linear, k-NN, RBF-network, Neural Network, SVM

Following the other homework and adding in both SVM and neural networks I notice that each kind of problem solver has a situation that it is better for, good things, and bad tradeoffs. While the neural network was getting some of the lowest values on average for errors it took much longer than all the others and when it did have high errors it was substantially higher than the others. When we used early stopping with neural networks we got the lowest values. That being said, neural networks aren’t the best tool when dealing with ever changing data that needs to be constantly learned on. For doing that something like a SVM is better. When dealing with multiple types of classification we go back to our old arguments and k-NN really is best if we were trying to classify for all the digits. At the end of the day there are many tools in our toolbox and we can use each one of them in different situations to solve the problem we have. We must look into computing cost and time, complexity of our data, etc and then make a choice based on which method fits us best. Personally I like neural networks most because they are crazy awesome. Though we did get different error values they were almost all equal and very close together. This shows us that when proper Cross Validation is used to choose the best model our erro will be extremely minimal