Homework 12
CSCI4100

Han Hai
Rin:661534083
haih2@rpi.edu

December 9th 2018

1. **(300) Neural Networks and Backpropagation** Write a program to implement gradient descent for a 2 input (d(0) = 2), m-hidden unit (d(1) = m), 1 output sigmoidal neural network (L = 2). For the output node transformation, allow for both S(s) = s and $\Theta(s)$ = tanh(s). Implement gradient decent on the squared error Ein , and check your gradient calculation as follows: For this problem, we will use the data set consisting of the 2 points x1 = (1, 0), y1 = +1 and x2 = (-1, 0), y2 = -1.

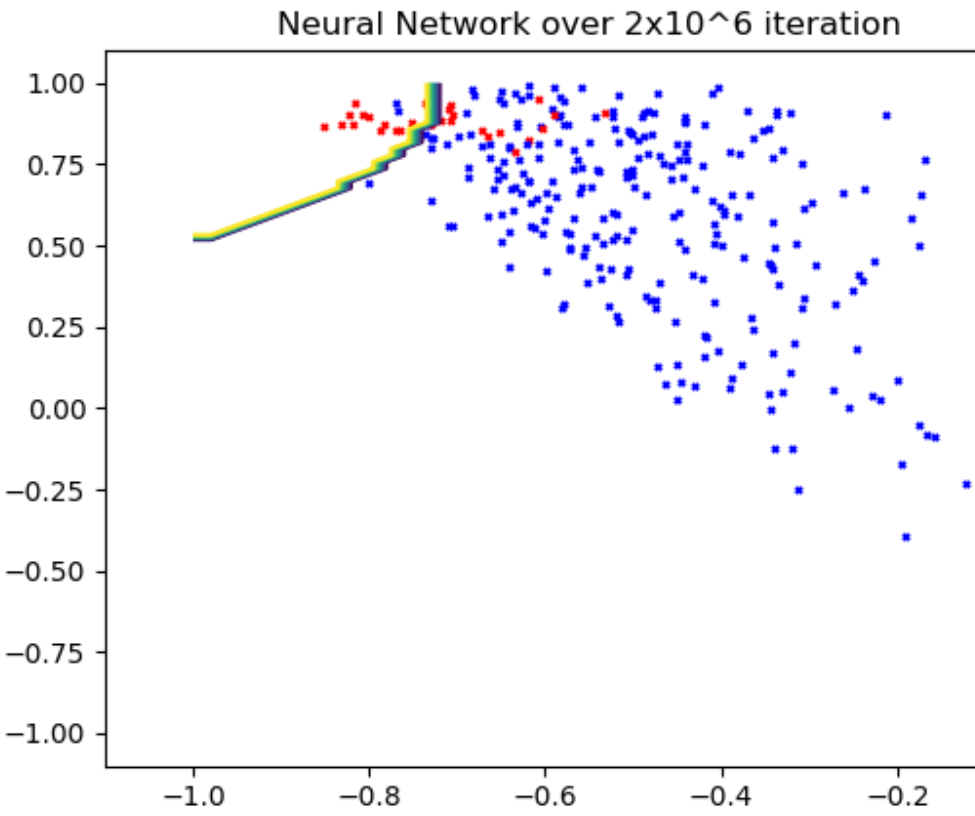
   (a) Use a network with m = 2. Set all the weights to 0.25 and consider a data set with 1 point:x=[1,1] ; y = 1. For both the identity and tanh(·) output node transformation functions, obtain the gradient of Ein(w) using the backpropagation algorithm. Report this result - there should be as many numbers as parameters in this network.

   Solution:

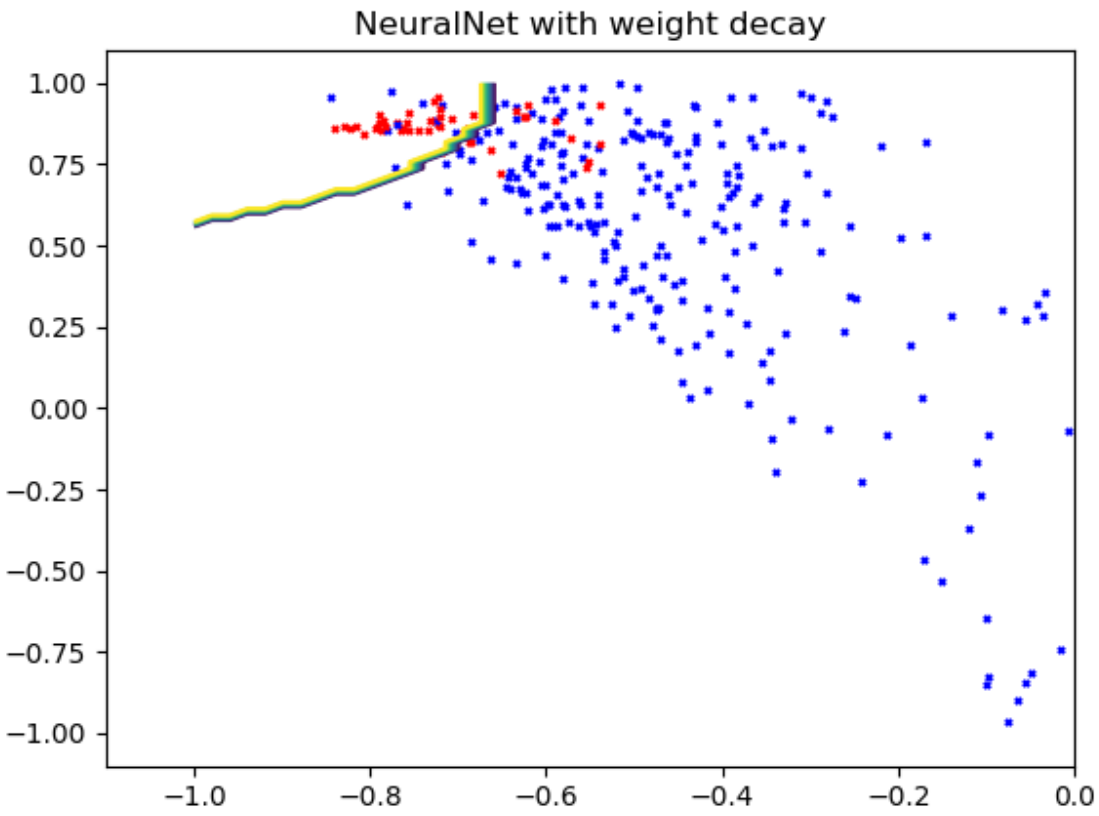
```
tanh:
G1:
[array([[-0.02670654, -0.02670654],
        [-0.02670654, -0.02670654],
        [-0.02670654, -0.02670654]])
G2:
 array([[-0.17906249],
        [-0.11373135],
        [-0.11373135]])
identity:
G1: [array([[-0.046875, -0.046875],
        [-0.046875, -0.046875],
        [-0.046875, -0.046875]])
G2:
array([[-0.1875  ],
        [-0.140625],
        [-0.140625]]
```

   (b) Now, obtain the gradient numerically by peturbing each weight in turn by 0.0001. Report this result. (Hint: If this result is not similar to your previous result then there is something wrong with your backpropagtion gradient calculation.)
   It returns exactly the same results.

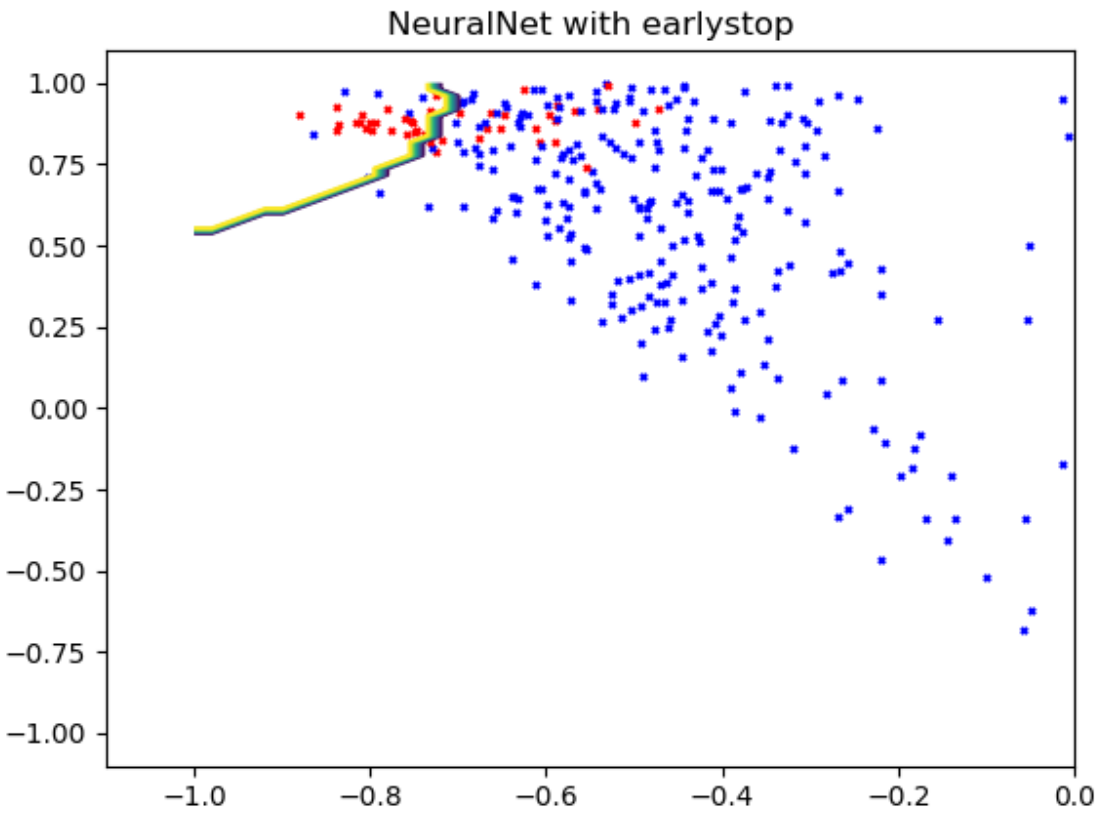
2. **Neural Network for Digits**



Etest:0.1103


NeuralNet with weight decay
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.0
−0.8
−0.6
−0.4
−0.2
0.0

Etest:0.105



Etest:0.1075

3. **(300) Support Vector Machines**

   (a) Show that for these 2 data points, the optimal separating hyperplane (with maximum cushion) is just the 'plane' that is the perpendicular bisector of the line segment joining the two points. In our case, what is the equation of the optimal hyperplane?
   First we calculate the constraint:
   $1 * (1 * w1 + 0 * w2 + b) \geq 1$ for first point
   $-1 * (-1 * w1 + 0 * w2 + b) \geq 1$ for second point
   we get$w1 \geq 1$ and $b \geq 0$
   we can see the optimal solution is $w1* = 1, w2* = 0, b* = 0$
   we get hypthesis$g(x) = sign(x1)$ therefore the optimal plane equation x1=0,
   this is indeed the hyper plane that is the perpendicular bisector of the line segment joining the two points.

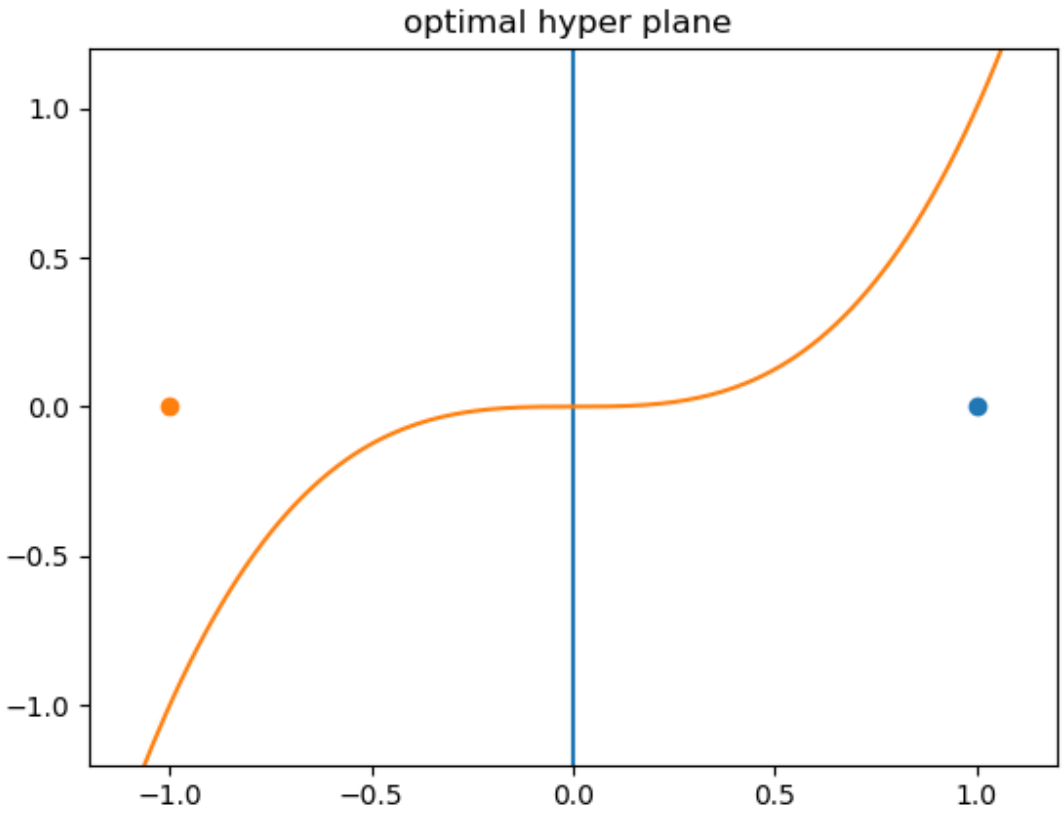
   (b) Now consider a transformation to a more "complicated" Z–space. The transformation is given by $z = [z1, z2], z1 = x1^3 - x2, z2 = x1x2$

      i. What are the data points in this space?
      point1 x1=[1,0], point 2 x2=[-1,0]

      ii. Construct the optimal hyperplane in this space
      similarly, we get the final hyptohesis $g(x) = sign(x1^3 - x2)$
      z1=0 is the optimal hyperplane

   (c) Plot (in X–space) the decision boundary for the optimal hyperplane constructed using the data in X–space (from part (a)). On the same plot, plot the decision boundary you would observe in

X-space if you classified X-space points by first transforming to Z-space, and then classifying according to the optimal hyperplane constructed using the data in Z–space (this decision boundary will not be a line!).
Note: the blue curve is decision boundary in X space while the yellow curve is decision boundary in Z space.



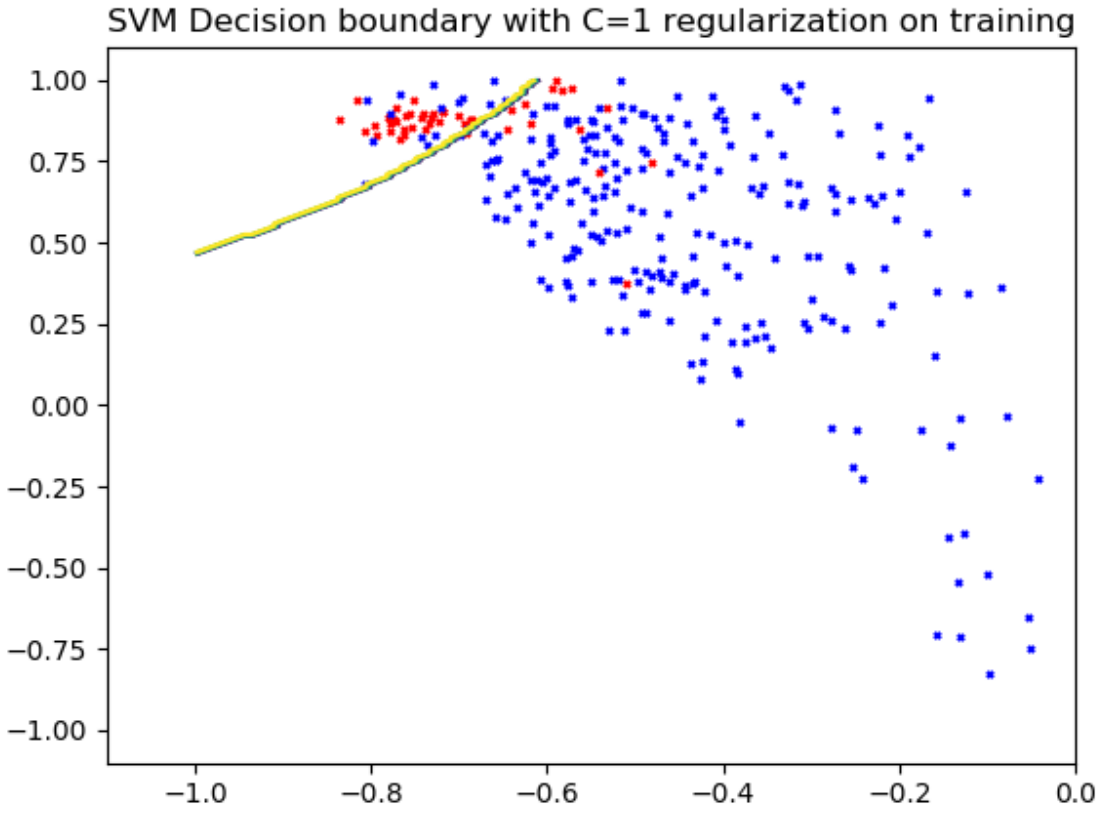
(d) A kernel function, K(x, y), is a function of two vectors in X-space defined by K(x, y) = z(x)· z(y), where z(x) and z(y) are the transformed x and y into Z-space. In other words, the kernel function computes the dot product of the transformed vectors. Give an expression for the kernel function in terms of the components of x and y.

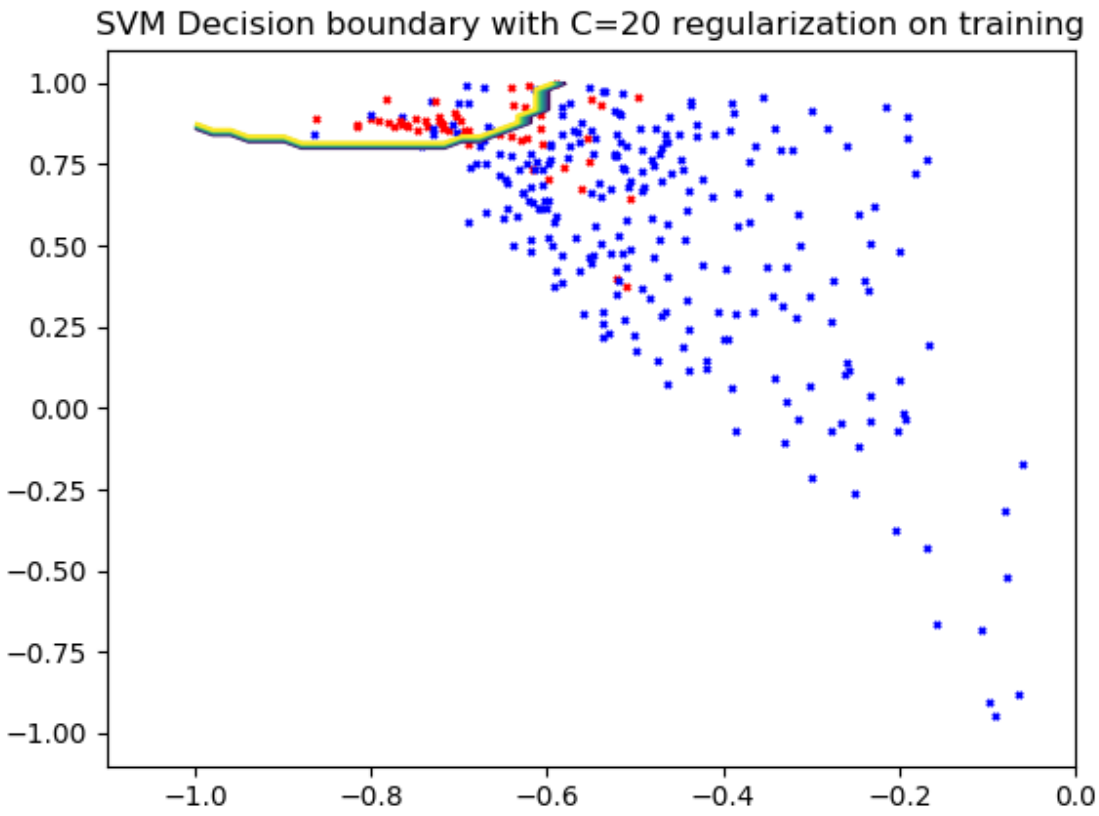
z(x)=$[x1^3 - x2, x1x2]$ z(y)=$[y1^3 - y2, y1y2]$
$z(x) \cdot z(y) = (x1^3 - x2) * (y1^3 - y2) + x1x2y1y2$

(e)) Using this kernel function, give an explicit functional form for the classifier in the X-space g(x)=$sign(x_1^3 y_1^3)$

4. **SVM with digits data**

(a) In the non-separable case, you need to choose the value for C greater than 0 (the 'regularization' parameter). Show the decision boundary for a small and large choice of C. Use your own judgment to determine small and large.

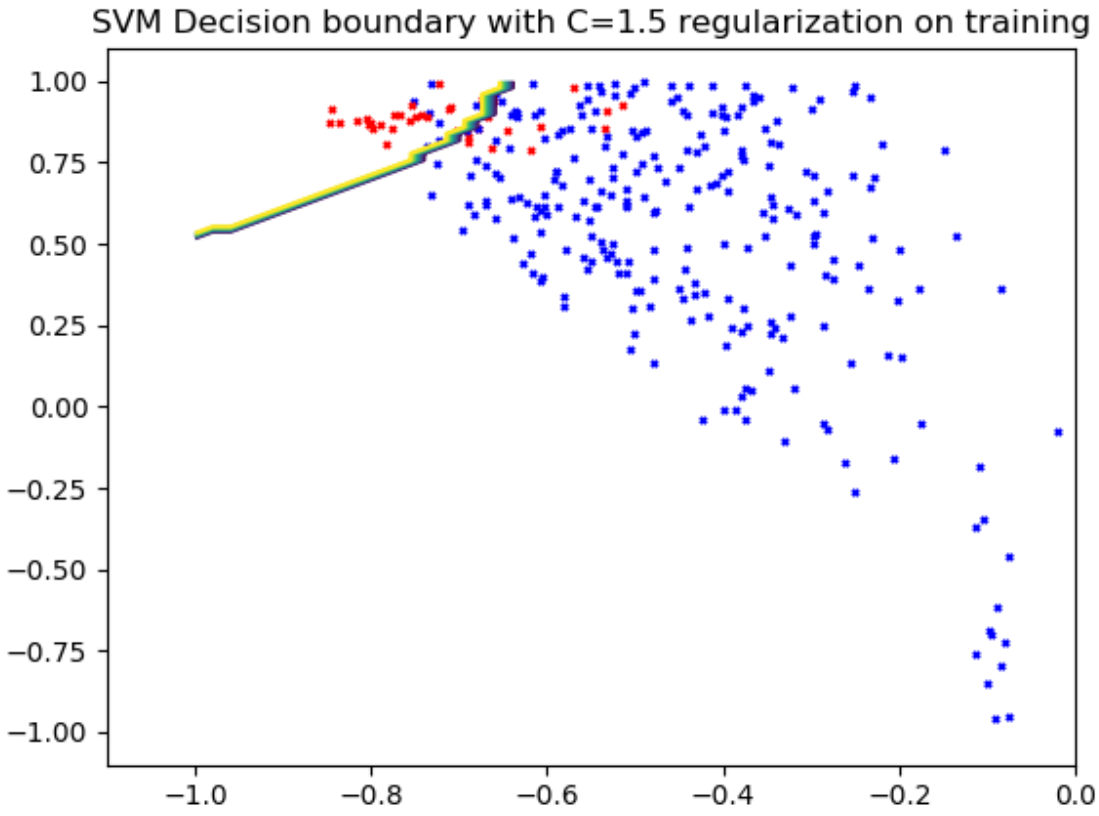




(b) Explain the 'complexity' of the decision boundaries you observe in part (a) with respect to the choice of C

As we can see SVM with C=20 has a lot more extra curves in the boundary, while SVM with C=1 is relatively smoother. This means a higher C C=20 has a much high complexity than lower C, C=1. A higher C means we care more about violating the margin, thus it's stricter, and will be more complex. While lower C allows relatively loose restriction, thus simpler function.

(c)) Using a grid of values for C between your small and large values, use cross validation to pick a good value of C, one that achieves minimum cross validation error. Show the decision boundary for the resulting classifier and give its test error.
Using cross validation, the best C is just C=1.5, with a $E_{test} = 0.081$
Here is the decision boundary



5. **Compare Methods: Linear, k-NN, RBF-network, Neural Network, SVM**
Here are the Etest of these algorithm
Linear with regularization:0.083
KNN: 0.094
RBF: 0.091
SVM:0.081
Neural Network:0.105
Apparently, SVM surpass the linear model and become the best solution so far. The power of SVM over linear might be the SVM is able to maximize the error margin for points in test data set. Neural network didn't do so well, it is probably because some parmeter of NN is not fully optimal. Linear still holds as the second place. It proves that sometimes the simplest algorithm perfrom the best. After hearing other student's result in the last class today. I realize my feature selection is very bad :(+1 and -1 are overlapping a lot. This probably has something to do with my low accuracy.