ASSIGNMENT 12

LFD is the class textbook

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

   Write a program to implement neural networks and gradient descent using backpropagation for a neural network architecture specified by a vector $[d^{(0)}, d^{(1)}, \ldots, d^{(L)}]$, where $d^{(L)} = 1$. All the hidden node transformations are tanh, $\theta(s) = \tanh(s)$ for hidden-layer nodes. For the output node transformation in the last layer, you should allow the user to pick between identity, $\theta(s) = s$, tanh, $\theta(s) = \tanh(s)$ and sign, $\theta(s) = \text{sign}(x)$. Set the architecture to $[2, m, 1]$: 2 inputs ($d^{(0)} = 2$), m-hidden units ($d^{(1)} = m$), and 1 output node ($L = 2$). Implement gradient decent on the squared error $E_{\text{in}}(\mathbf{w}) = \frac{1}{4N} \sum_{n=1}^{N} \left(h(\mathbf{x}_n, \mathbf{w}) - y_n\right)^2$, and check your gradient calculation as follows:

   (a) Use a network with $m = 2$. Set all the weights to 0.25 and consider a data set with 1 point: $\mathbf{x}_1 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$; $y = 1$. For the output node using the identity transformation and also the $\tanh(\cdot)$ transformation, obtain the gradient of $E_{\text{in}}(\mathbf{w})$ using the backpropagation algorithm. Report this result - there should be as many numbers as parameters in this network.

   (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.)

2. **(600) Neural Network for Digits**

   Use your neural network implementation with $m = 10$ hidden units to build a classifier for separating "digit 1" from "not digit 1". Use the two features and data you developed in a previous assignment and the 300 training data points you selected in assignment #9.

   Randomly initialize the weights to small values. Use the tanh() transformation function for the hidden units. When you train your network, use the linear transformation function for the output unit in the last layer (regression for classification). Note, only the last layer transformation function is linear for training. At the end, when you are done training, you now change the transformation function for the output unit in the last layer to take the sign of the output to produce a classification.

   (a) Plot $E_{\text{in}}(\mathbf{w})$ versus iteration for the variable learning rate gradient descent heuristic and 2×10^6 iterations. Show the decision boundary for the resulting classifier.

   (b) Now use weight decay with $\lambda = 0.01/N$ and use variable learning rate gradient descent to minimize the augmented error. Show the decision boundary for the resulting classifier.

   (c) Now use early stopping with a validation set of size 50 and training set of size 250, and show the decision boundary for the resulting classifier that had minimum validation error.

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

   For this problem, we will use the data set consisting of the 2 points $\mathbf{x}_1 = (1, 0),\ y_1 = +1$ and $\mathbf{x}_2 = (-1, 0),\ y_2 = -1$.

   (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?

(b) Now consider a transformation to a more "complicated" Z–space. The transformation is given by

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} x_1^3 - x_2 \\ x_1 x_2 \end{bmatrix} \tag{1}$$

i. What are the data points in this space?

ii. Construct the optimal hyperplane in this space (i.e., what is the equation for the hyperplane in terms of the Z–coordinates).

To classify a point in X-space, one first transforms the point into Z-space. One then classifies the point in Z-space using the optimal hyperplane in Z-space.

(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!).

(d) A kernel function, $K(\mathbf{x}, \mathbf{y})$, is a function of two *vectors* in X-space defined by $K(\mathbf{x}, \mathbf{y}) = \mathbf{z}(\mathbf{x}) \cdot \mathbf{z}(\mathbf{y})$, where $\mathbf{z}(\mathbf{x})$ and $\mathbf{z}(\mathbf{y})$ are the transformed $\mathbf{x}$ and $\mathbf{y}$ 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 $\mathbf{x}$ and $\mathbf{y}$.

(e) Using this kernel function, give an explicit functional form for the classifier in the X-space.

4. (**600) SVM with digits data**

Implement the non-separable SVM using your two features for the 300 training points you selected in assignment #9. Use the 8th order polynomial kernel,

$$K(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^{\mathrm{T}}\mathbf{x}')^8.$$

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

(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.

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

Compare the final test error from your attempts to solve the digits problem:

(i) Regularized linear model with 8th order polynomial transform and λ selected by CV.
(ii) k-NN rule with k selected by CV.
(iii) RBF-network with number of centers selected by CV.
(iv) Neural network with early stopping.
(v) SVM with 8th order polynomial kernel and C selected by CV.

Make some intelligent comments.