CS536 Machine Learning
Spring 2007
Assignment 2
Due Date April 16th, 2007

[Q1 15 points] Given a perceptron with a *tanh( )* activation function, i.e., $o=tanh(w^T.x)$. Drive the least-mean-squares weight update rule for such perceptron. (Hint $tanh'(x)=1-tanh^2(x)$)

[Q2 60 points – Using Weka] Download two datasets cancer.arff and canertest.arff from the class web page.

1. Open cancer.arff file as the data file and set cancertest.arff as test file.

2. 2. Use the **SMO,** which is just an implementation of Suppport Vector Machine (SVM) that avoids solving the large QP (ref: http://research.microsoft.com/users/jplatt/smo-book.pdf), from the classifiers of type **functions**.

    a. (12 points) You have to check the performance of SVM with different kernels by tuning the corresponding parameters. The first parameter to check is **C.** This is described as the complexity parameter. Basically, it controls the trade-off between fitting the training data (large values) and maximizing the separating margin (small values). Use **C**= 1, 2, 10, 20, 60, 100. Did the classification error change? What does this say about the training data in terms of generalization?

    b. (16 points) The default Kernel used for SMO is polynomial. The parameter **exponent** defines the degree of the polynomial. Change **C** back to 1. Set **exponent** = 1, 2, 3, 4, 5 . Did the classification error change? Can you comment on the form of the separating hyperplane from the change in the classification error?

    c. (12 points) Change the Kernel to a RBF kernel by setting **useRBF** to true. The parameter **gamma** control the width of the kernel. See the effect of gamma on the classification error by setting **gamma**= 0.25, 0.5, 1, 4, 10, 15. Why do you think the rate is decreasing with the increase of **gamma**?

3. Delete all the classifiers from the **Result list,** otherwise you may run out of memory.

4. Select **Multilayer Perceptron** as the classifier. We will modify three parameters of this classifier, namely **hiddenLayers**, **learningRate** and **momentum**. For details about momentum, read the pdf file in the webpage.
    a. (20 points) Come up with the simplest optimal classifier that takes the minimum amount of time to learn. For both **learningRate** and **momentum,** use the values from 0.1 to 0.6. Read the description of **hiddenLayers** by right clicking on the classifier textbox and try different values.