HW3

(Due December 17, midnight)

Problem 1 Random forests(15 points). Random forests predict with an ensemble of bagged trees each trained on a bootstrap sample randomly drawn from the original training data. Additional random variation among the trees is induced by choosing the variable for each split from a small randomly chosen subset of all of the predictor variables when building each tree. What are the advantages and disadvantages of this random variable selection strategy? How can one introduce additional tree variation in the forest without randomly selecting subsets of variables?

Problem 2 ESL-Ex 10.9 (20 points). Consider a K-class problem where the targets y_{ik} are coded as 1 if observation i is in class k and zero otherwise. Using the multinomial deviance loss function (10.22) and the symmetric logistic transform ($\sum_k \gamma_{jkm} = 0$), use the arguments leading to the gradient boosting Algorithm 10.3 to derive Algorithm 10.4.

Problem 3 ESL-Ex 14.14 (20 points): Generate 200 data points with three features, lying close to a helix. In detail, define $X_1 = cos(s) + 0.1Z_1$, $X_2 = sin(s) + 0.1Z_2$, $X_3 = s + 0.1Z_3$ where s takes on 200 equally spaced values between 0 and 2π, and Z_1, Z_2, Z_3 are independent and have standard Gaussian distributions.

1. Fit a principal curve to the data and plot the estimated coordinate functions. Compare them to the underlying functions $cos(s)$, $sin(s)$ and s.

2. Fit a self-organizing map to the same data, and see if you can discover the helical shape of the original point cloud.

Problem 4. Backward propagation (15 points). Consider a multi-hidden layer neural network trained by sequential steepest-descent using the weight updating formula

$$w_t = w_{t-1} - \eta_t G(w_t)$$

Here t labels the observations presented in sequence (time) and G(w) is the gradient of the squared-error criterion evaluated at w. Derive a recursive "back-propagation" algorithm for updating all of the network weights at each step. With this algorithm the update for an input weight to a particular hidden node is computed using only the value of its corresponding input (that it weights), the value of the output of the hidden node to which it is input, and an "error signal" from each of the nodes in the next higher layer

to which this node is connected. Thus, each node in the network can update its input weights using information provided only by the nodes to which it is connected.

Problem 5. Binary classification of Spam Email (30 points). The data set for this problem is Spam_Data, with documentation files Spam_Info and Spam_Names. The data set is a collection of 4601 emails of which 1813 were considered spam, i.e. unsolicited commercial email. The data set consists of 58 attributes of which 57 are continuous predictors and one is a class label that indicates whether the email was considered spam (1) or not (0). Among the 57 predictor attributes are: percentage of the word "free" in the email, percentage of exclamation marks in the email, etc. See file Spam_Names for the full list of attributes. The goal is, of course, to predict whether or not an email is "spam". This data set is used for illustration in the tutorial Boosting with R Programming. The data set Spam_Train represents a subsample of these emails randomly selected from Spam_Data to be used for training. Spam.Test contains the remaining emails to be used for evaluating results.

1. Based on the training data Spam_Train, fit a gbm model for predicting whether or not an email is"spam", following the example in the tutorial. What is your estimate of the misclassification rate? Of all the spam emails of the test set Spam.Test what percentage was misclassified, and of all the non-spam emails in the test set what percentage was misclassified?

2. Your classifier in part (a) can be used as a spam filter. One of the possible disadvantages of such a spam filter is that it might filter out too many good (non-spam) emails. Therefore, a better spam filter might be the one that penalizes misclassifying non-spam emails more heavily than the spam ones. Suppose that you want to build a spam filter that "throws out" no more that 0.3% of the good (non-spam) emails. You have to find and use a cost matrix that penalizes misclassifying "good" emails as"spam" more than misclassifying "spam" emails as "good" by the method of trial and error. Once you have constructed your final spam filter with the property described above, answer the following questions:

   (a) What is the overall misclassification error of your final filter and what is the percentage of good emails and spam emails that were misclassified respectively?

(b) What are the important variables in discriminating good emails from spam for your spam filter?

(c) Using the interpreting tools provided by gbm, describe the dependence of the response on the most important attributes.

3. Fit on the training set one hidden layer neural networks with 1, 2,..., 10 hidden units and different sets of starting values for the parameters (obtain in this way one model for each number of units). Which structural model performs best at classifying on the test set? (You need first to standardize predictors and choose all the weights starting values at random in the interval [-0.5, 0.5].)

4. Choose the optimal regularization (weight decay for parameters 0,0.1,...,1) for the structural model found above by averaging your estimators of the misclassification error on the test set. The average should be over 10 runs with different starting values. Describe your final best model obtained from the tuning process: number of hidden units and the corresponding value of the regularization parameter. What is an estimation of the misclassification error of your model?