15-381 Fall '09, Homework 5

- Due: Wednesday, November 18th, beginning of the class
- You can work in a group of up to two people. This group does not need to be the same group as for the other homeworks. You only need to turn in one write-up per group, but you must include the andrew ID's of the group members.
- Late homework is due by 3:00 P.M. on the day they are due. Please bring late homeworks to Susan Hrishenko in GHC 7119 (slide it under the door if she is not there; write on the homework the date and time turned in). *Note that GHC 7th floor is now locked after 6pm. If you finish in the evening, you'll probably need to wait until the next morning to turn it in.*
- Email all questions to 15381-instructors@cs.cmu.edu

1 More Decision Trees- 10 pts

1. (5 pts) Describe how one might use search techniques (genetic algorithms, simulated annealing, etc.) to find a good decision tree. Make sure you address the representation problem.
2. (5 pts) What are pros and cons to doing that, compared with using the greedy information gain approach?

2 Artificial Neural Networks- 20 pts

1. (5 pts) In class we discussed that a single perceptron cannot compute XOR. Construct a network of *multiple* perceptrons that will accomplish this (draw a picture, and include weights).
2. (5 pts) Now construct a network that will compute parity of four 0/1 inputs (that is, will output 1 if the sum is odd, or 0 if the sum is even).
3. (5 pts) Suppose instead of using the notion of squared-error, as in lecture, we wanted to use $E = Err = |y - h_W(x)|$. Derive the perceptron learning rule. (Hint, you'll need to assume that error never becomes zero).
4. (5 pts) Why might we want to use a sigmoid function instead of a threshold function for activation?

3 Linear Separators and SVMs- 10 pts

We will construct a support vector machine that computes the XOR function. It will be convenient to use values of 1 and -1 instead of 1 and 0 for the inputs/outputs. So a training example looks like ([-1, 1], 1) or ([-1, -1], -1).

1. (1 point) Draw the four examples using axes x_1, x_2, denoting class by +/-. Label them $(p_1, ..., p_4)$.

2. (6 points) Propose a transformation that will make these points linearly separable (it can be done in 2D). Draw the four input points in this space, and the maximal margin separator. What is the margin?

3. (3 points) Now draw the separating line back in the original Euclidean input space in part 1.

4 Pruning Decision Trees- 35 pts

For this question, if you can find a suitable implementation of the algorithms online, you are welcome to use them.

1. (10 pts) Run ID3 to train a decision tree for the German Credit Approval data set, classifying whether someone is approved for credit. See data available on the webpage. As you are building the tree, record the error rates on both the training set and the test set (similar to Slide 18 in the second Decision Tree lecture). How many nodes are in the resulting tree?

2. (10 pts) Now run reduced-error pruning based on the validation set. Record the error rate for the validation set as the tree is being pruned, with the plot as in the previous part. How many nodes are in the resulting tree?

3. (10 pts) Now use the statistical significance testing method to prune the tree. Plot the accuracy for different thresholds of significance. How many nodes are in the resulting tree?

4. (5 pts) Interpret your plots. How does pruning affect results on the training and test error? What pruning method works best, and why do you think that is? If we picked out a new test set to measure accuracy, which pruning method would have better results?

5 Clustering- 30 pts

For this question, implement the algorithms yourself and submit code in the AFS space.

Sometimes instead of treating data as clumps of points, we may want to assume they have a certain distribution. *Gaussian mixture models* are one example of these.

For example, suppose that for some reason you wanted to classify a population of students as male or female. We know that men are *generally* taller than women (with each population having approximately a Gaussian/normal distribution), but there's a lot of overlap. We also know that the male:female at CMU is 3:2, so if you have a data point of middling height, you're probably safer guessing that the student is male. Gaussian mixture models try to take this into account, by giving a probabilistic assignment to membership, rather than deterministic as in K-means.

The way one can go about solving this is to use what's called the *EM Algorithm*. For Gaussian mixture models, the EM algorithm is similar to K-means, only in addition to calculating means (centers) for each cluster, you also use"mixing" variables a_i to reflect the overall distribution of the classes (a 3:2 ratio would have $(a_0, a_1) = (.6, .4)$. Based on these parameters, each point will be assigned to a cluster with some *probability* of membership to each class. After initialization of μ_i and a_i for $i = (0, 1)$, you iterate through the following two steps until convergence:

First, in the expectation ("E") step, you estimate an expectation value for the "membership" m_{ij} of each point x_j for distribution Y_i.

$$m_{ij} = \frac{a_i f_{Y_i}(x_j; \mu_i, \sigma_i)}{\sum_{k=1}^{K} a_k f_{Y}{}_k(x_j; \mu_k, \sigma_k)}$$

where f is the formula for the normal distribution. Then, based on the membership values assigned, you re-adjust your estimates of the mean and variance of each Gaussian to maximize the likelihood ("M-step"), as well as the mixing coefficients:

Each mixing coefficient is the mean of the membership values for all points.

$$a_i = \frac{1}{N} \sum_{j=1}^{N} m_{ij}$$

The mean of each Gaussian is a *weighted* average of all points with its membership value.

$$\mu_i = \frac{\sum_j m_{ij} x_j}{\sum_j m_{ij}}$$

You can derive an update for variance, but for this problem we'll cheat and assume $\sigma = 1$.

1. (15 points) Implement EM as above for two 1-D Gaussian mixtures, and run on the GMM data set available on the webpage. Do several runs, each time randomizing your initial settings for μ and a (let's say pick some μ between (-10, 10)). Show the distribution of the end estimates for μ and a over all the initial settings. What is the average accuracy? What is the maximum accuracy attained by any single run?

2. (10 points) Implement K-means (one dimensional, for K=2) and repeat, also using several runs with random initialization. Show the distribution for the end estimates of the centers over all initial settings. What is the average accuracy? What is the maximum accuracy attained by any single run?

3. (5 points) How do your results compare? Why do you think one performs better than the other? In which cases would the other perform better?