Homework Assignment 6

36-350, Data Mining

Due at the start of class, 7 November 2008

1. *Minimum-Error Classification* In this problem, you will prove that the way to minimize the probability of mis-classification is to always predict the most probable class.

   Let Y be the class, which is binary (0 or 1), and X the input features (generally a vector). Let $\Pr(Y = 1|X = x) = p(x)$. Consider a classifier which makes a *randomized* prediction, predicting 1 with probability $q(x)$. Further, suppose that the actual class and that the prediction are conditionally independent given X.

   (a) For each fixed x, show that the probability of mis-classification, R, is $q + p - 2pq$.

   (b) Plot this error rate as a function of q, in the interval $[0, 1]$ for $p = 0.1$, $p = 0.3$, $p = 0.5$, $p = 0.6$ and $p = 0.9$. Where are the minima?

   (c) Show that the derivative of R with respect to q is never zero, unless $p = 1/2$.

   (d) Show that R is minimized when $q = 1$ if $p > 0.5$, and when $q = 0$ if $p < 0.5$.

2. *Three Classifiers* Download the data set `foobar` from Blackboard. It should have three columns: two real-valued inputs called `x1` and `x2`, and a class called `y`. The two classes are `foo` and `bar`. (You can read it in with `read.table`, among other commands.)

   You can use any previous code you wrote, or code from class, or packages from CRAN, but say where you got your functions.

   (a) Plot the data. Use different colors (via the `col` argument) or point-shapes (via the `pch` argument) for the two classes. If you use different colors, make sure they look distinct when you print them out!

   (b) Divide the data set at random into two equal haves, one for training and one for testing. Include your code. Include a check that the two halves have the right size, and that they do not overlap.

   (c) Fit a prototype classifier to the training data and evaluate it on the test data. Report the error rate.

(d) Do the same with a nearest-neighbor classifier.

(e) Do the same with a classification tree. See below for some R advice. Include a picture of the tree, annotated with the actual splits.

R advice on fitting trees

There are several packages on CRAN for fitting trees on R. The simplest one is called `tree`. It will fit either classification or regression trees. It fits classification trees if the response variable is a factor, otherwise it fits regression trees.

The usual syntax is something like this:

```
my.tree = tree(y ~ x1 + x2, data=my.frame)
```

This tells it that the response variable is `y`, the two input variables are `x1` and `x2`, and the data should come from the data frame `my.frame`. The input data source needs to be a data-frame.

Results can then be plotted:

```
plot(my.tree)
text(my.tree)
```

This plots the tree, and then adds labels. There a number of options for making this look nicer: see `help(plot.tree)` and `help(text.tree)`.

There is also a prediction method:

```
predict(my.tree,newdata=testing.frame)
```

This will return the vector of predicted class probabilities. To get the actual predicted classes, do

```
predict(my.tree,newdata=testing.frame,type="class")
```

Again, see `help(predict.tree)` for more things you can do with the prediction method.

Look at `help(prune.tree)` and `help(cv.tree)` to see how to prune a tree via cross-validation. You do *not* need to do that for this problem.