36-350: Data Mining

Lab 12
Date: November 14, 2003 **Due: end of lab**

Interspersed throughout this lab are questions that you will have to answer at check-off.

1. Download the files for this lab from the course web page to the desktop:

   `http://www.stat.cmu.edu/~minka/courses/36-350/lab/`

2. Open a Word or Notepad document to record your work.

Start R

3. `Start -> All Programs -> Class software -> R 1.7.0`

4. Load the special functions for this lab:

   `File -> Source R code...`

   Browse to the desktop and pick `lab12.r` (it may have been renamed to `lab12.r.txt` when you downloaded it). Another window will immediately pop up for you to pick the `mining.zip` file you downloaded.

The dataset

5. The dataset describes 1000 individuals who acquired loans from a bank. Each individual is described by 39 variables, the most important being `Class` which is `Good` if the person repaid the loan. The bank would like to use this data to decide which customers in the future are likely to repay a loan. There is a matrix of training data called `x.tr` and a matrix of test data called `x.te`.

Classification trees

6. Using the commands below, make a classification tree from the training data to predict `Class`. *What is its performance on the test data, according to misclassification rate and deviance?*

7. Use cross-validation on deviance to estimate the best tree size, and prune your tree to that size. The resulting classifier should be pretty simple. *Does the pruned tree perform better on the test set? According to which measure?*

8. Run cross-validation with 2 blocks multiple times, and compare to using 10 blocks multiple times. *Which number of blocks is more stable? When there are 2 blocks, how much of the training data is used to build each of the 2 trees? When there are 10 blocks, how much of the training data is used to build each of the 10 trees?*

9. Plot the test set performance of each pruning of the original tree. *Is your pruning above close to the best pruning for the test set?*

K-nearest neighbor

10. Construct a 1-nearest-neighbor classifier from the training set. *What is its performance on the test set, according to misclassification rate and deviance?*
11. Use cross-validation to choose a better k, and evaluate this k on the test set. *It is better than $k = 1$?*
12. Plot the performance of each k on the test set. *Is your cross-validated k close to the best k for the test set?*
13. You can now get checked off.

Constructing a tree The `tree` function is similar to `lm` and `smooth`:

```
fit = tree(<formula>,<data>)
plot.graph.tree(fit,cex=.8)
```

K-nearest neighbor

```
fit = knn.model(<formula>,<data>,k=<number of neighbors>)
```

The knn classifier automatically converts the predictors to numeric, standardizes them, and applies Euclidean distance.

Measuring performance The predictions of a tree or knn `fit` on a dataset `data` can be evaluated in two ways:

```
misclass(fit,data,rate=T)
deviance(fit,data,rate=T)
```

Cross-validation pruning If `fit` is a tree:

```
fit2 = best.size.tree(fit,10)
```

plots the cross-validated deviance of various prunings of `fit`, and returns the best pruning. 10 is the number of blocks to use.

Cross-validation for k If `fit` is a knn object:

```
fit2 = best.k.knn(fit)
```

plots the cross-validated misclassification rate of various k, and returns a knn object with the best k. (This can take a while.)

Holdout pruning As a tree `fit` is pruned, the performance on a dataset `data` is plotted:

```
plot(prune.tree(fit,newdata=data),type="o")
```

Holdout for k The knn `fit` is modified to use various values of k, with misclassification rate on `data` plotted:

```
test.k.knn(fit,data)
```