Assignment 2, Math 3346, 2009

Lecturer: John Maindonald

September 3, 2009

Due date: Sept 21 2009, at 5pm

Please keep code, apart perhaps from short code fragments, in an appendix that is separate from the main body of the report. You will be given marks for presentation and layout.

This exercise will work with experimental control and treatment groups in the data set `nswdemo`, and with non-experimental comparison groups in the data sets `cps1` and `psdi1`. All these datasets are included in the *DAAG* package.

Useful Functions

The following functions will be useful.

Extract a random sample of rows: The following function is designed to extract a random sample of `m` rows from a data frame (or matrix).

```
samprows <- function(df, m)df[sample(1:nrow(df), m), ]
```

Function to evaluate predictive accuracy, given model predictions:

```
confusion <- function(actual, predicted, names=NULL,
                      printit=TRUE, prior=NULL){
   if(is.null(names))names <- levels(actual)
   tab <- table(actual, predicted)
   acctab <- t(apply(tab, 1, function(x)x/sum(x)))
   dimnames(acctab) <- list(Actual=names,
                      "Predicted (cv)"=names)
   if(is.null(prior)){
     relnum <- table(actual)
     prior <- relnum/sum(relnum)
     acc <- sum(tab[row(tab)==col(tab)])/sum(tab)
   } else
   {
     acc <- sum(prior*diag(acctab))
     names(prior) <- names
 }
   if(printit)print(round(c("Overall accuracy"=acc,
                           "Prior frequency"=prior),4))
   if(printit){
     cat("\nConfusion matrix", "\n")
     print(round(acctab,4))
   }
   invisible(list(accuracy=acc, confusion=acctab, prior=prior))
 }
```

Function to fit `lda()` or `qda()` model, and evaluate accuracy: This uses leave-one-out cross-validation, which is not ideal. However it is a good way to get a quick rough initial assessment.

```
check.model <- function(form, df=nswdem, dp=4, prior=NULL, discfun=lda){
     categ <- all.names(form)[2]
     if(is.null(prior)){
         df.disc <- discfun(form, data=df, CV=TRUE)
         acc <- confusion(df[, categ], df.disc$class, printit=FALSE)$accuracy
     } else {
        df.disc <- discfun(form, data=df, CV=TRUE, prior=prior)
        acc <- confusion(df[, categ], df.disc$class, printit=FALSE,
                         prior=prior)$accuracy
    }
     print(paste("Model is", deparse(form)))
     print(c(Accuracy = round(acc, dp)))
     invisible(acc)
 }
```

Datasets

We will want the following datasets

`nswdem` **(Experimental data):** Add the variable `logre75` to the dataframe `nswdemo`

```
library(DAAG); library(DAAGxtras)
library(splines)
dim(nswdemo)
```

```
[1] 722  10
```

```
library(MASS)
unique(sort(nswdemo$re75))[2]/2
```

```
[1] 37.17173
```

```
nswdem <- nswdemo
nswdem$logre75 <- log(nswdemo$re75+37)
```

`nswplus`**: Experimental data, plus non-experimental controls** Create a dataset `nswplus` that combines the experimental data with the two sets of non-experimental controls.

```
library(DAAG); library(DAAGxtras)
nswplus <- rbind(nswdemo, cps1, psid1)
nswplus$gp <- factor(rep(c("nsw","cps","psid"),
                         c(722, nrow(cps1), nrow(psid1))),
                     levels=c("nsw","cps","psid"))
nswplus$logre75 <- log(nswplus$re75+1)
nswplus$gp1 <- 1-as.numeric(nswplus$gp=="nsw")
  # gp1 distinguishes between the experimental and other data
## Now check that the new dataset has been correctly created
all.equal(nswplus[nswplus$gp=="nsw",1:10],nswdemo)
```

```
[1] TRUE
```

```
all.equal(nswplus[nswplus$gp=="cps",1:10],cps1, check.attributes=FALSE)
```

```
[1] TRUE
```

```
all.equal(nswplus[nswplus$gp=="psid",1:10],psid1, check.attributes=FALSE)
```

```
[1] TRUE
```

Questions

Function help page

Question 1: Test the function `confusion()` by using the function `lda()` to do discriminant calculations, and then using this fucntion to determine the predictive accuracy.

By typing

```
prompt(confusion)
```

you can get a skeleton for a help page for the function `confusion()`. Fill in the details for the help page. ***[3 marks]***

Data transformation:

Question 2: Why is it preferable to work with `logre75`, rather than `re75`? ***[1 mark]***

Treated vs controls in `nswdem`, based on pre-treatment variables

Here is code that may be used to compare the control and treatment groups in the experimental `nswdem` dataset.

```
library(lattice)
form1 <- trt ~ age+educ+black+hisp+marr+nodeg+logre75
form2 <- trt ~ ns(age,3)+educ+black+hisp+marr+nodeg+logre75
  # form2 allows for the possibility that the effect of age mayh be nonlinear
form3 <- trt ~ (ns(age,3)+logre75)*(educ+black+hisp+marr)+nodeg
  # form3 allows for interaction effects involving continuous variables
## Try also the equivalent models with form2 and form3.
check.model(form1)
```

Question 3a: With `form` taken to be whichever of `form1`, `form2` and `form3` gives the best accuracy, run the following code:

```
nswdem1.lda <- lda(form, data=nswdem)
score <- predict(nswdem.lda)$x
plot(densityplot(~score, groups=nswdem$trt, auto.key=list(columns=2)))
```

Given that subjects had been randomly divided between treated and control (though with a greater number in control), is the graph much what might be expected? Explain.
[2 marks]

3b: Compare the accuracies with the accuracy achieved by assigning all observations to the most frequent category. (In output from `rpart()`, this is the *root node accuracy*.) Does the best accuracy that was achieved give any reason to suspect the randomization? Explain.
[2 marks]

Pre-treatment comparison between nswdem, cps1 and psdi1

Check the number of rows in each of `nswdem`, `cps1` and `psid1`.

The following will use the smoothing spline methodology of the function `gam()` in the *mgcv* package to check possible transformation of the variables `age`, `educ`, and `logre75`. The software automatically makes what should be a sensible choice of the amount of smoothing, corresponding to each of the terms `s(age)`, `s(educ)`, and `s(logre75)`.

```
library(mgcv)
form <- gp1 ~ s(age)+s(educ)+s(logre75)+black+hisp+marr+nodeg
nswcps.gam <- gam(form, data=subset(nswplus, gp!="psid"))
nswpsid.gam <- gam(form, data=subset(nswplus, gp!="cps"))
```

Question 4: Would the same transformations that are effective for the comparison between the experimental data and the `cps1` data be useful for the comparison between the experimental data and the `psid1` data. Justify your answer by reference to the graphs given by `plot(nswcps.gam)` and `plot(nswpsid.gam)`. You might want to do:

```
opar <- par(mfrow=c(2,3), mar=c(3.6,3.6,0.6,0.6), mgp=c(2.25,.5,0))
plot(nswcps.gam)
plot(nswpsid.gam)
par(opar)
```

[2 marks]

Comparison between the three groups, using lda()

Question 5: Use `lda()` to obtain a graphical comparison between the three groups. It may be desirable to account for nonlinear effects from any or all of `age`, `educ` and `logre75`. The graphs from Question 4 suggest that spline curves would be worth trying when we use `lda()`. (The function textttlda() does not have the functionality, implemeted in `gam()`, for automatic choice of the degree of spline curve.) In what follows, we try normal splines of degree 4.

```
library(splines)
fm1 <- gp ~ age+educ+black+hisp+marr+nodeg+logre75
fm2 <- gp ~ ns(age,4)+ns(educ,4)+ns(logre75,4)+black+hisp+marr+nodeg
```

Using whichever of `fm1` and `fm2` gives the best cross-validation predictive accuracy, fit the model, and graph the results. For example:

```
nswplus.lda <- lda(fm1, data=nswplus, prior=rep(1,3)/3)
scores <- predict(nswplus.lda)$x
library(lattice)
xyplot(scores[,1] ~ scores[,2], groups=nswplus$gp,
       auto.key=list(columns=3), par.settings=simpleTheme(cex=0.5, alpha=0.2))
```

The graph has a very striking feature. Can you shed any light on it? Hint: Consider looking at ranges of variables, within suitable splits of the data. ***[2 marks]***

Comparison between the three groups, using randomForest()

Proximities from `randomForest` calculations can be used to get a plot that reflects the way that the data have been classified. Points are close together or widely separated according as they have, or have not, followed the same path down the trees to the terminal node. The proximity is, for each pair of points, the proportion of trees in which they have come together at the same terminal node. With more than several thousand points, the number gets so large that it creates problems for the computations.

```
## Extract subset of data for plotting
m <- cumsum(c(nrow(nswdem), nrow(cps1), nrow(psid1)))
n1 <- sample(1:m[1], 500)
n2 <- sample((m[1]+1):m[2], 500)
n3 <- sample((m[2]+1):m[3], 500)
take <- c(n1,n2,n3)
nswplus.rf <- randomForest(fm1, data=nswplus, sampsize=rep(722,3))
nswplus.pred <- predict(nswplus.rf, newdata=nswplus[take, ], proximity=TRUE)
sim <- 1-nswplus.pred$proximity
scores.rf <- cmdscale(sim)
xyplot(scores.rf[,1] ~ scores.rf[,2], groups=nswplus$gp[take],
       auto.key=list(columns=3), par.settings=simpleTheme(cex=0.5, alpha=0.5))
```

Question 6a: Annotate the above code, explaining what each step does, and the choice of function arguments. ***[3 marks]***

Question 6b: Comment on the comparison between the `randomForest()` analysis and the `lda()` analysis. ***[2 marks]***

Overview

Question 7: Write brief notes on what can be deduced from the several steps of the analysis. NB: Keep in mind the reasons for collecting and comparing these various sets of data. ***[4 marks]***

TOTAL MARKS = 20