Statistics 151B Homework 4 Due: 2 PM Mon 16 Apr 2012

Bootstrapping a nonparametric logistic regression

The South African heart disease data needs no introduction at this point. In this exercise, you will fit a model similar to (5.6) in Section 5.2.2 of HTF, but using smoothing splines instead of fixed-degrees-of-freedom natural cubic splines. You will study the variability in this fit by using the bootstrap to produce a figure like Figure 5.4 of HTF.

You will need to install the package `gam` (*generalized additive models*) for this exercise.

What to turn in for this problem: email the following three files to the GSI and myself by the above deadline.

1. A file containing the R commands you used. Call it `commands.txt`. This file should have *only* your *input* to R.
2. A file containing a transcript of your R session. Call it `transcript.txt`.
3. A file with the figures you generated, and a discussion of your analysis.

- Load `sa-heart.data` (on the course web site) into R as a data frame `sa.heart`. Do any variable conversions that seem appropriate. Calculate relevant summaries and perform inspections of the data.
- We will focus on the relationship between the predictor variable `age` and the binary response `chd`. Use the function `gam()`, from the `gam` library, to fit a smoothing spline regression of `chd` on `age`. Use 10 degrees of freedom in the smoothing spline. Consult the documentation for `gam()` and `s()` to learn exactly how to run the fit. Call this `orig.ss.fit`.
- Use the built-in `plot()` function to plot the smoothing spline fit. Include pointwise SE's estimated using standard theory (see the documentation of `plot.gam()` for the details).
- Now generate 500 new smoothing spline fits of `chd` versus `age`, each one based on a bootstrap sample of the original data. You will need a `for` loop to do this (see the help page). Save the 500 bootstrapped splines in a list: at each iteration of the for loop, you add one new bootstrapped spline to the (initially empty) list.
- Make a new data frame `newdat`, consisting only of the variable `age`. The values of `newdat$age` should be on a fine grid over the observed range of `age` in the original data.
- Use the function `sapply()` to build a matrix `M` with one column per bootstrapped spline. The contents of each column should be the result of calling `predict()` on that column's spline, with data `newdat`. You will need to spend some time looking at the documentation for `sapply()` and `predict()`.
- Use the function `matplot()` to plot the columns of `M` as gray lines.

- Add a thick black line to this plot which corresponds to `orig.ss.fit` evaluated on `newdat`. (Use `predict()` again.)
- Use the function `apply()` to evaluate the SD of each row of `M` (that is, the pointwise SD's of the bootstrapped splines).
- Add two thick red lines to your plot, corresponding to `orig.ss.fit` $\pm$ 2 bootstrap-estimated SE's.
- Comment on the plot, and compare with the one you obtained using the `plot()` function.