Homework 5

Advanced Methods for Data Analysis (36-402/36-608)

Due Thurs March 20, 2014 at 11:59pm

1 Additive abalones

You'll be looking at (once again!) the abalone data. Refer back to Problem 2 of Homework 3 for a description of the data set. Here we'll be looking to predict the "whole weight" of abalones (column 5 of the abalone data frame), in a nonparametric fashion, from other predictors. Once you've read in the abalone data frame, and saved it as say, `aba`, it will be helpful to name the columns of this data frame:

```
colnames(aba) = c("Sex","Length","Diam","Height","Weight",
                  "Shucked","Viscera","Shell","Age")
```

(Remember that for the last column, we actually have to first add 1.5 to get the age in years.) Most of the tasks in the parts below can be performed using the `gam` package (though the linear models can also be fit with the standard `lm` function, of course). Global hint: make sure you utilize the functionality provided by the `gam` package to answer the questions; this will save you a lot of unnecessary programming. Look back at the code given in the additive models lecture for examples.

(a) Fit an additive model of `Weight` on `Diam`, `Height`, `Age`, and `Sex`. For each of `Diam`, `Height`, `Age`, use a smoothing spline with 4 degrees of freedom. For `Sex`, use a linear smoother. Plot the estimated regression functions (here there should be 4), along with estimates of their (pointwise) standard errors. Describe and interpret the plots.

(b) For each variable in the model from part (a) (`Diam`, `Height`, `Age`, `Sex`), perform two F-tests: one for the presence of a linear effect in this variable versus no effect at all, and one for the presence of a nonlinear effect in this variable versus a linear effect—these are called tests for parametric and nonparametric effects, respectively. (Note that for `Sex`, we would not perform the second test.) Recall the proper interpretations for these tests: if a nonparametric test failed to reject, then we would use a linear effect for the corresponding variable in the model; if a parametric test failed to reject, then we would not include the corresponding variable in the model at all. What are the results of the F-tests, and what effects would you make linear, or drop from the model, if any?

(c) Now you will fit 3 competing additive models to that in part (a), which we will call model 1:

- model 2: `Weight` on `Diam` (smoothing spline, 4 df) `Height` (smoothing spline, 4 df), and `Sex` (linear)
- model 3: `Weight` on `Diam` (linear) `Height` (linear), and `Sex` (linear)

To compare the predictive accuracy of these 3 models, use 5-fold cross-validation. Report the cross-validation estimates of prediction error for each of the 3 models, as well as the standard errors of these estimates.

(d) Compare the cross-validation results for models 1 and 2. Are the estimated prediction errors of the two models comparable (within one standard error)? Hence, is the action you took to drop a variable in part (b) justified, from the perspective of prediction error?

(e) Compare the cross-validation results for models 2 and 3. Again, are their estimated prediction errors close, taking the standard errors into consideration? What do you conclude about the importance of using nonlinear terms in model 2 versus linear ones in model 3?

(f) Model 2 generalizes model 3, in the sense that it replaces the linear terms (for `Diam` and `Height`) by nonlinear ones. Another way to generalize model 3 is to include linear interaction terms. That is, consider fitting yet another model:

- model 4: `Weight` on `Diam` (linear), `Height` (linear), `Sex` (linear), `Diam * Sex` (linear), and `Height * Sex` (linear)

where in the above we have used `a * b` to denote the interaction between two variables `a` and `b` (as per the formula notation used in `gam` or `lm`). Use 5-fold cross-validation again to estimate the prediction error of this model. Compare the estimated prediction errors for models 2 and 4. Is one better than the other, again taking into account standard errors?

(g) Now, we consider the complexities of models 2 and 4. Count the degrees of freedom of model 4—note that you can do this because it is simply a linear model. Count the degrees of freedom of model 2—for an additive model, you can think of the degrees of freedom of the final fit as simply the sum of the degrees of freedom of the individual smoothing operators, *making sure not to double count for the intercept*. As a concrete example, the additive model:

- y on x_1 (smoothing spline 7 degrees of freedom), x_2 (smoothing spline, 4 degrees of freedom), and x_3 (linear)

where x_1, x_2, x_3 are continuous variables, has $(7-1)+(4-1)+2 = 11$ degrees of freedom. Here we have been careful to only include the intercept term once (coupling it with the linear effect). Which fitted model is more complex, according to its degrees of freedom, model 2 or model 4? What can you say now about prediction error estimates that you compared in part (f), and the strength of the additive model?

Bonus: leave-one-out in no time at all

Let $\hat{r}$ denote the smoothing spline fitted regression function (at some fixed value of the smoothing parameter λ), trained on (x_i, y_i), $i = 1, \ldots n$. Let $\hat{r}^{-(i)}$ denote the smoothing spline fit (again, at the same fixed value of λ), but now trained on pairs (x_j, y_j), $j \neq i$.

(a) Consider the leave-one-out cross-validation error estimate

$$A = \frac{1}{n} \sum_{i=1}^{n} \big(y_i - \hat{r}^{-(i)}(x_i)\big)^2,$$

and a weighted sum of the training errors

$$B = \frac{1}{n} \sum_{i=1}^{n} \Big(\frac{y_i - \hat{r}(x_i)}{1 - S_{ii}}\Big)^2.$$

Here S_{ii} is the ith diagonal matrix of the smoothing operator S defined on the full data set (x_i, y_i), $i = 1, \ldots n$, i.e., the matrix S is such that $\hat{y} = (\hat{r}(x_1), \ldots \hat{r}(x_n)) = Sy$. Show empirically that these

two are the same, i.e., $A = B$. You should create a simulated data set with sufficiently arbitrary pairs (x_i, y_i), $i = 1, \ldots n$, evaluate A and B, and show that they are extremely close (any differences are due to rounding inaccuracies). Hint: to access the diagonal elements S_{ii}, $i = 1, \ldots n$, use the `lev` component of the object returned by the `smooth.spline` function.

Note: this provides a huge computational savings for leave-one-out cross-validation with smoothing splines! Why? To compute B, we don't need to refit the smoothing spline operator (i.e., fit $\hat{r}^{-(i)}$, $i = 1, \ldots n$) at all, we can simply use the original fitted regression function $\hat{r}$.

(b) Prove that $A = B$.

(c) Does the analogous result hold for kernel regression? For k-nearest-neighbors?