Ch. 3 curve fitting by linear smoothing

Tuesday, February 26, 2019 2:49 PM

Consider a nonlinear regression with one predictor and on response:

$$y_i = f(x_i) + \varepsilon_i$$

where ε_i are mean zero RVs.

A) Suppose we want to estimate the value of the regression function y^* at some new point x^*, denoted $\hat{f}(x^*)$. Assume for the moment that $f(x)$ is linear, and that y and x have already had their means subtracted, in which case

$$y_i = \beta x_i + \varepsilon_i$$

Returning to the OLS estimator for multiple regression, show that for the one predictor case, your prediction $\hat{y}^* = f(x^*) = \hat{\beta} x^*$ may be expressed as a linear smoother of the following form:

$$\hat{f}(x^*) = \sum_{i=1}^{n} w(x_i, x^*) y_i$$

for any x^*. Describe your understanding of how the resulting smoother behaves, compared with the smoother that arises from an alterate form of the weight function $w(x_i, x^*)$:

$$w_K(x_i, x^*) = \begin{cases} \frac{1}{K}, & x_i \text{ one of the } K \text{ closest sample points to } x^* \\ 0, & \text{otherwise.} \end{cases}$$

This is called the K-nearest neighbors smoothing.

For multiple regression, $\hat{\beta} = (X^T X)^{-1} X^T y$ where X is an $n \times p$ matrix of observations and y is an $n \times 1$ vector of response variables. In the case that there is only one predicor, we have

$$\hat{\beta} = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2}$$

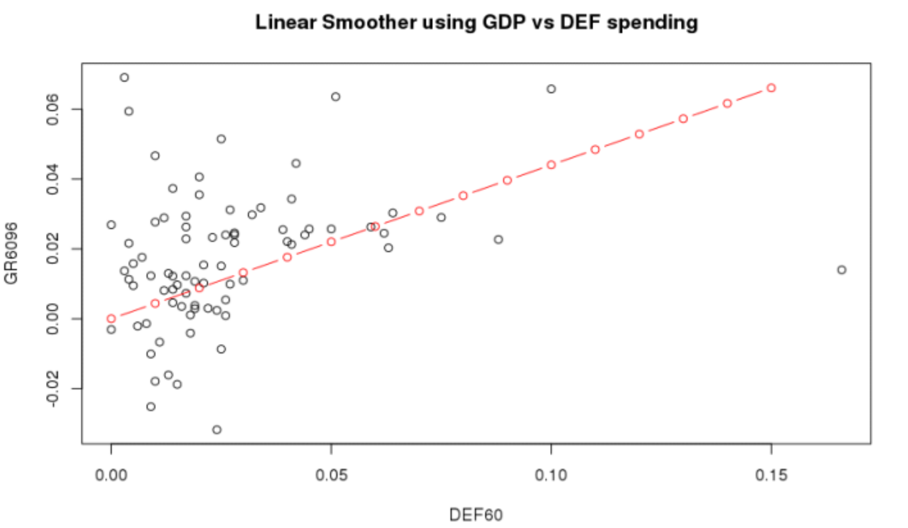
To find a new predicted value $f(x^*) = \hat{\beta} x^*$, we have

$$y^* = f(x^*) = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2} x^*$$

$$\Rightarrow \hat{f}(x^*) = \frac{\sum_{i=1}^{n} x_i x^*}{\sum_{i=1}^{n} x_i^2} y_i = \sum_{i=1}^{n} w(x_i, x^*) y_i \quad \text{where } w(x_i, x^*) = \frac{x_i x^*}{\sum_{i=1}^{n} x_i^2}$$

This weight seems to assign larger weight to the y_i that correspond to X_i that are larger in magnitude, regardless of what x^* is. This in effect smoothes the data into a line. In the case where we use the weight generated by an OLS predictor, we are smoothing the data into the line of best fit. Observe the below figure. The scatterplot is the GDP of various countries against the percentage of the GDP spent on defense. The red line is a set of new predicted values y^*

we are smoothing the data into the line of best fit. Observe the below figure. The scatterplot is the GDP of various countries against the percentage of the GDP spent on defense. The red line is a set of new predicted values y^* given a new set of x^*. We can see that the linear smoother becomes the line of best fit with a zero intercept.



The K nearest neighbors smoother is fundamentally different. Instead of giving linear weight to points, K nearest neighbors acts like a moving average. The new predicted value of x^* is the average of the K closest points to x^*. This moving average will smooth the data into something that retains the shape of the original data but has less noise.

B) A kernel function $K(x)$ is a smoothing function satisfying

$$\int_{\mathbb{R}} K(x)\,dx = 1 \qquad \int_{\mathbb{R}} x\,K(x)\,dx = 0 \qquad \int_{\mathbb{R}} x^2 K(x)\,dx > 0$$

Kernels are used as weighting functions for taking local averages. Specifically define the weighting function

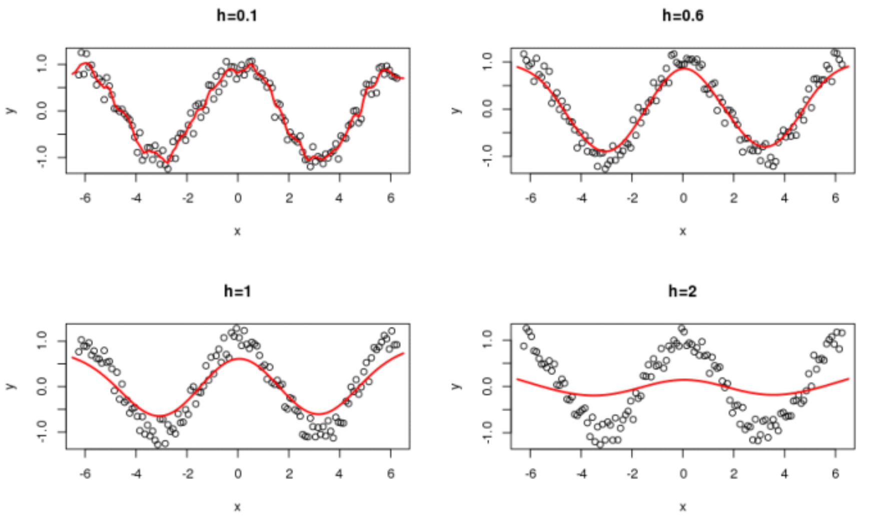
$$w(x_i, x^*) = \frac{1}{h} K\left(\frac{x_i - x^*}{h}\right)$$

where h is the bandwidth. Write an R function that fits a kernel smoother for an arbitrary choice of h. Simulate noisy data from some nonlinear function $y = f(x) + \varepsilon$; subtract the sample means from the simulated x and y; and use your function to fit the kernel smoother for some h. Plot the estimated functions for a range of bandwidths large enough to yield noticable changes in the qualitative behavior of the prediction functions.

The kernel I used is the Gaussian kernel

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{x^2/2}$$

The following figure shows how the smoother works with 4 different values of h. The test function is a cosine with added noise.



Visually, it would appear small values of h do not produce very smooth results as it makes the bin of the smoother too small. Overly large values of h cause the function to flatten because it tries to smooth over too many points.