SDS 383D: Homework 3

Giorgio Paulon

February 20, 2017

Problem 1. Basics Concepts

Bias–variance decomposition

Let $\hat{f}(x)$ be a noisy estimate of some function $f(x)$, evaluated at some point x. Define the mean-squared error of the estimate as

$$MSE(\hat{f}, f) = E\{[f(x) - \hat{f}(x)]^2\}\,.$$

Prove that $MSE(f, \hat{f}) = B^2 + V$, where

$$B = E\{\hat{f}(x)\} - f(x) \quad \textit{and} \quad V = Var\{\hat{f}(x)\}\,.$$

We can write

$$\begin{aligned}
\text{MSE}(\hat{f}, f) &= \text{E}\left[f(x)^2 + \hat{f}(x)^2 - 2f(x)\hat{f}(x)\right] \\
&= f(x)^2 + \text{E}\left[\hat{f}(x)^2\right] - 2f(x)\text{E}\left[\hat{f}(x)\right] \\
&= f(x)^2 - 2f(x)\text{E}\left[\hat{f}(x)\right] + \text{E}\left[\hat{f}(x)\right]^2 + \text{E}\left[\hat{f}(x)^2\right] - \text{E}\left[\hat{f}(x)\right]^2 \\
&= \left(f(x) - \text{E}\left[\hat{f}(x)\right]\right)^2 + \text{Var}\left[\hat{f}(x)\right] \\
&= B^2 + V.
\end{aligned}$$

A simple example

Some people refer to the above decomposition as the bias–variance tradeoff. Why a tradeoff? Here's a simple example to convey the intuition.

Suppose we observe $x_1, \ldots, x_n$ from some distribution F, and want to estimate $f(0)$, the value of the probability density function at 0. Let h be a small positive number, called the bandwidth, and define the quantity

$$\pi_h = P\left(-\frac{h}{2} < X < \frac{h}{2}\right) = \int_{-h/2}^{h/2} f(x)dx\,.$$

Clearly $\pi_h \approx hf(0)$.

(A) *Let Y be the number of observations in a sample of size n that fall within the interval $(-h/2, h/2)$. What is the distribution of Y? What are its mean and variance in terms of n and π_h? Propose a simple estimator $\hat{f}(0)$ involving Y.*

The number of observation that fall in the interval $(-h/2, h/2)$ follows a Binomial distribution, i.e.

$$Y \sim \text{Bin}\,(n, \pi_h)\,.$$

Therefore, we get $\mathrm{E}[Y] = n\pi_h$ and $\mathrm{Var}(Y) = n\pi_h(1-\pi_h)$. A simple estimator for $f(0)$, say $\hat{f}(0)$, is

$$\hat{f}(0) = \frac{Y}{nh},$$

so that

$$\mathrm{E}[\hat{f}(0)] = \frac{1}{nh}\mathrm{E}[Y] = \frac{\pi_h}{h} \approx f(0)$$
$$\mathrm{Var}(\hat{f}(0)) = \frac{1}{n^2h^2}\mathrm{Var}(Y) = \frac{\pi_h(1-\pi_h)}{nh^2}$$

(B) *Suppose we expand $f(x)$ in a second-order Taylor series about 0:*

$$f(x) \approx f(0) + xf'(0) + \frac{x^2}{2}f''(0)\,.$$

Use this in the above expression for π_h, together with the bias–variance decomposition, to show that

$$MSE\{\hat{f}(0), f(0)\} \approx Ah^4 + \frac{B}{nh}$$

for constants A and B that you should (approximately) specify. What happens to the bias and variance when you make h small? When you make h big?

We can re-express the probabilities π_h using a second order Taylor approximation for $f(x)$, that is

$$\begin{aligned}
\pi_h &= \int_{-h/2}^{h/2} f(x)dx \\
&\approx \int_{-h/2}^{h/2} f(0)dx + \int_{-h/2}^{h/2} xf'(0)dx + \int_{-h/2}^{h/2} \frac{x^2}{2}f''(0)dx \\
&= f(0)\cdot h + \frac{1}{2}f'(0)\left[\left(\frac{h}{2}\right)^2 - \left(-\frac{h}{2}\right)^2\right] + \frac{1}{2}f''(0)\left[\frac{x^3}{3}\right]_{-h/2}^{h/2} \\
&= h\left(f(0) + \frac{f''(0)}{24}\cdot h^2\right)
\end{aligned}$$

Recall the formula for the MSE and write

$$\begin{aligned}
\mathrm{MSE}\left[\hat{f}(0), f(0)\right] &= \left[f(0) - \mathrm{E}\left(\hat{f}(0)\right)\right]^2 + \mathrm{Var}\left(\hat{f}(0)\right) \\
&= \left(f(0) - f(0) - \frac{f''(0)}{24}\cdot h^2\right)^2 + \frac{1}{nh}\left(f(0) + \frac{f''(0)}{24}\cdot h^2\right)\left(1 - f(0) - \frac{f''(0)}{24}\cdot h^2\right) \\
&= \left(\frac{f''(0)}{24}\right)^2 h^4 + \frac{1}{nh}\left(f(0) + \frac{f''(0)}{24}\cdot h^2 - f(0)^2 h - \frac{f(0)f''(0)}{12}\cdot h^3 - \frac{f''(0)^2}{24^2}\cdot h^5\right) \\
&\approx \underbrace{\left(\frac{f''(0)}{24}\right)^2\left(1 - \frac{1}{n}\right)}_{\mathrm{A}}\cdot h^4 + \frac{1}{nh}\cdot\underbrace{f(0)}_{\mathrm{B}}.
\end{aligned}$$

When h is small, the bias term gets small and the variance term gets large. The opposite happens when h is large.

(C) *Use this result to derive an expression for the bandwidth that minimizes mean-squared error, as a function of n. You can approximate any constants that appear, but make sure you get the right functional dependence on the sample size.*

One can find the minimum of the MSE with a standard calculus procedure.

$$\begin{aligned}
&\operatorname*{argmin}_h \text{MSE}(n) = \operatorname*{argmin}_h \left(Ah^4 + \frac{B}{nh} \right) \\
&\Rightarrow \frac{d}{dh}\left(Ah^4 + \frac{B}{nh} \right) = 4Ah^3 - \frac{B}{nh^2} = 0 \\
&\Rightarrow h = \left(\frac{B}{4An} \right)^{1/5}
\end{aligned}$$

Problem 2. Curve fitting by linear smoothing

Consider a nonlinear regression problem with one predictor and one response: $y_i = f(x_i) + \epsilon_i$, where the ϵ_i are mean-zero random variables.

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

Return to your least-squares estimator for multiple regression. Show that for the one-predictor case, your prediction $\hat{y}^\star = f(x^\star) = \hat{\beta} x^\star$ may be expressed as a linear smoother of the following form:

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

for any $x^\star$. Inspect the weighting function you derived. Briefly describe your understanding of how the resulting smoother behaves, compared with the smoother that arises from an alternate form of the weight function $w(x_i, x^\star)$:

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

This is referred to as K-nearest-neighbor smoothing.

Let us suppose that $y_i = f(x_i) + \epsilon_i$, where $\epsilon_i \overset{\text{iid}}{\sim} N(0, \sigma^2)$. If the function f is linear, then $\text{E}[y_i] = a\text{E}[x] + b = 0$, which implies $b = 0$. Thus, $f(x) = \beta x$ and $y_i = \beta x + \epsilon_i$.

In the case of Least squares, the estimator is

$$\hat{y}^* = \hat{\beta} x^*,$$

where

$$\hat{\beta} = (\boldsymbol{x}^T \boldsymbol{x})^{-1} \boldsymbol{x}^T \boldsymbol{y} = \frac{\sum_{i=1}^n x_i y_i}{\sum_{i=1}^n x_i^2}.$$

Therefore, we get

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

where the weights are

$$w(x_i, x^*) = \frac{x_i x^*}{\sum_{i=1}^n x_i^2}.$$

(B) *A kernel function $K(x)$ is a smooth function satisyfing*

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

A very simple example is the uniform kernel,

$$K(x) = \frac{1}{2} I(x) \quad \textit{where} \quad I(x) = \begin{cases} 1, & |x| \leq 1 \\ 0, & \textit{otherwise}\,. \end{cases}$$

Another common example is the Gaussian kernel:

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

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

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

where h is the bandwidth. Using this weighting function in a linear smoother is called kernel regression. (The weighting function gives the unnormalized weights; you should normalize the weights so that they sum to 1.)

Write your own R function that will fit a kernel smoother for an arbitrary set of x-y pairs, and arbitrary choice of (positive real) bandwidth h. Set up an R script that will simulate noisy data from some nonlinear function, $y = f(x) + \epsilon$; subtract the sample means from the simulated x and y; and use your function to fit the kernel smoother for some choice of h. Plot the estimated functions for a range of bandwidths large enough to yield noticeable changes in the qualitative behavior of the prediction functions.

In order to do kernel regression, we evaluate the smoothed function on a fine grid of points. For each point, we compute a set of weights that are given to the observations $x_1, \ldots, x_n$ given the distance to that particular point. The value of the bandwidth is crucial, as one can see in Figure 1 and in Figure 2. In particular, higher values for the bandwidth result in a smoother function, whereas smaller values of the bandwidth will tend to yield an interpolating and wiggly curve.

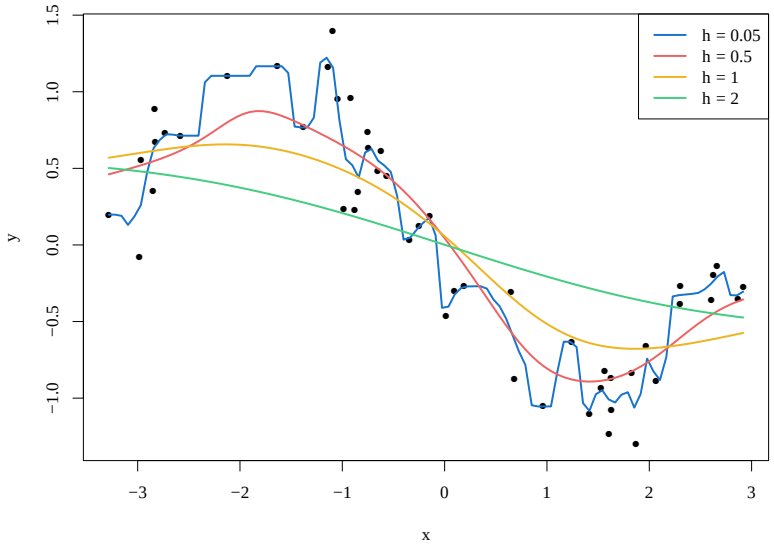

Figure 1: Gaussian kernel regression for the sampled black points. Different values of the bandwidth λ result in different smoothed curves.

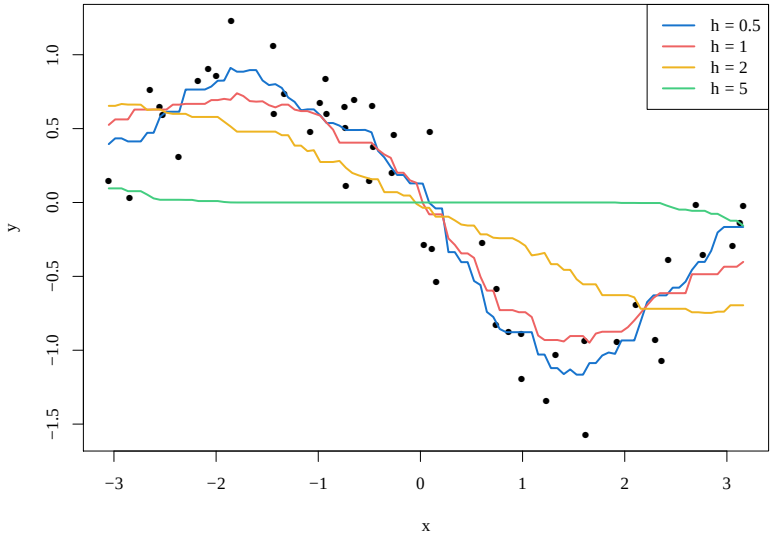

Figure 2: Uniform kernel regression for the sampled black points. Different values of the bandwidth λ result in different smoothed curves.

Problem 3. Cross validation

Left unanswered so far in our previous study of kernel regression is the question: how does one choose the bandwidth h used for the kernel? Assume for now that the goal is to predict well, not necessarily to recover the truth. (These are related but distinct goals.)

(A) *Presumably a good choice of h would be one that led to smaller predictive errors on fresh data. Write a function or script that will: (1) accept an old ("training") data set and a new ("testing") data set*

as inputs; (2) fit the kernel-regression estimator to the training data for specified choices of h; and (3) return the estimated functions and the realized prediction error on the testing data for each value of h. This should involve a fairly straightforward "wrapper" of the function you've already written.

The requested functions are displayed in Listing **??**.

(B) *Imagine a conceptual two-by-two table for the unknown, true state of affairs. The rows of the table are "wiggly function" and "smooth function," and the columns are "highly noisy observations" and "not so noisy observations." Simulate one data set (say, 500 points) for each of the four cells of this table, where the x's take values in the unit interval. Then split each data set into training and testing subsets. You choose the functions. Apply your method to each case, using the testing data to select a bandwidth parameter. Choose the estimate that minimizes the average squared error in prediction, which estimates the mean-squared error:*

$$L_n(\hat{f}) = \frac{1}{n}\sum_{i=1}^{n^\star}(y_i^\star - \hat{y}_i^\star)^2\,,$$

where $(y_i^\star, x_i^\star)$ are the points in the test set, and $\hat{y}_i^\star$ is your predicted value arising from the model you fit using only the training data. Does your out-of-sample predictive validation method lead to reasonable choices of h for each case?

	Wiggly	Smooth
Noisy	0.01556	0.04780
Not noisy	0.00793	0.01947

Table 1: Comparison of the optimal bandwidths for different choices of the underlying function.

(C) *Splitting a data set into two chunks to choose h by out-of-sample validation has some drawbacks. List at least two. Then consider an alternative: leave-one-out cross validation. Define*

$$LOOCV = \sum_{i=1}^{n}\left(y_i - \hat{y}_i^{(-i)}\right)^2\,,$$

where $\hat{y}_i^{(-i)}$ is the predicted value of y_i obtained by omitting the ith pair and fitting the model to the reduced data set. This is contingent upon a particular bandwidth, and is obviously a function of x_i, but these dependencies are suppressed for notational ease. This looks expensive to compute: for each value of h, and for each data point to be held out, fit a whole nonlinear regression model. But you will derive a shortcut!

Observe that for a linear smoother, we can write the whole vector of fitted values as $\hat{y} = Hy$, where H is called the smoothing matrix (or "hat matrix") and y is the vector of observed outcomes. Write $\hat{y}_i$ in terms of H and y, and show that $\hat{y}_i^{(-i)} = \hat{y}_i - H_{ii}y_i + H_{ii}\hat{y}_i^{(-i)}$. Deduce that, for a linear smoother,

$$LOOCV = \sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - H_{ii}}\right)^2\,.$$

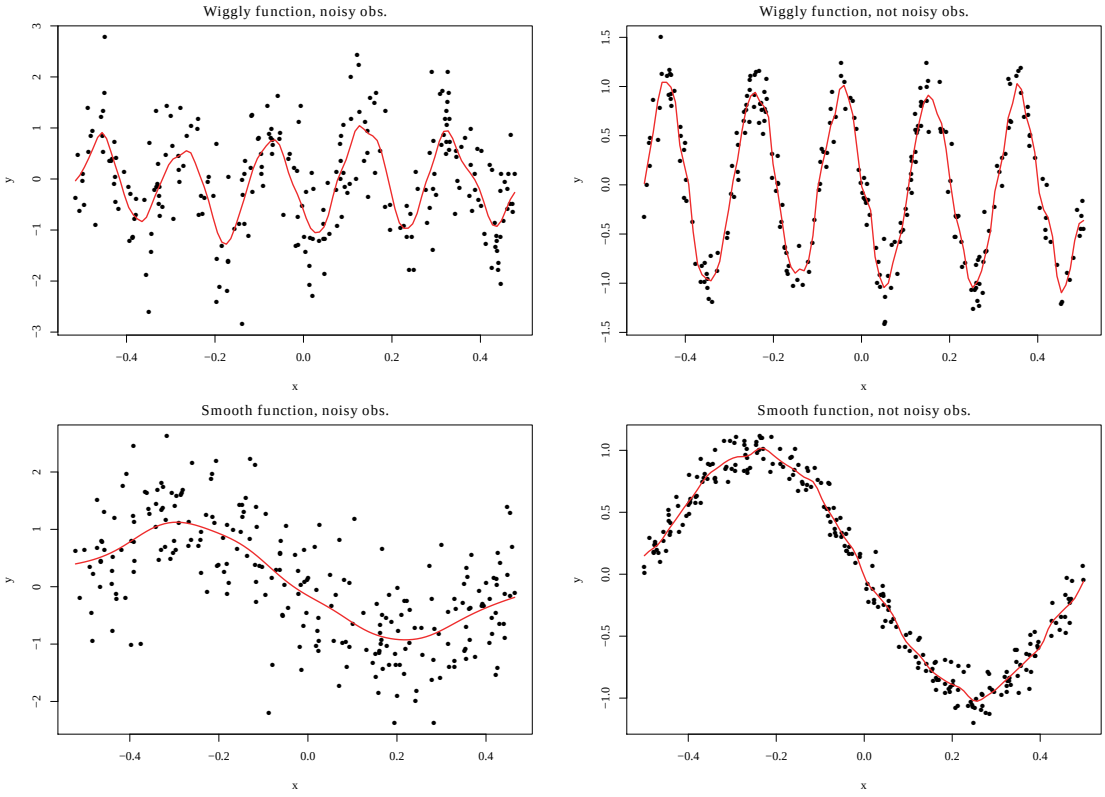
Wiggly function, noisy obs.
Wiggly function, not noisy obs.
Smooth function, noisy obs.
Smooth function, not noisy obs.
x
y

Figure 3

Appendix A

R code