Exercises 2: Fitting curves by smoothing

Linear smoothing: one predictor

Let's consider a problem with one predictor and one response, but a potentially more general regression function: $y_i = f(x_i) + \epsilon_i$.

(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 for $y^\star = \hat{\beta} x^\star$ may be expressed as a *linear smoother*[1] 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 would behave, compared with that arising 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*.

[1] This doesn't mean the weight function $w(\cdot)$ is linear in its argument, but rather than the new prediction is a linear combination of the past outcomes.

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

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

A kernel function is therefore like a probability density function with mean 0. A very simple example is the uniform kernel,

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

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

$$w_b(x_i, x^\star) = \frac{K\left(\frac{x_i - x^\star}{b}\right)}{\sum_{j=1}^{n} K\left(\frac{x_j - x^\star}{b}\right)}\,,$$

where b is called the *bandwidth*. Using this weighting function in a linear smoother is called *kernel regression*.

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 b.[2] Set up an R script that will simulate noisy data from some nonlinear function $y = f(x) + \text{error}$; subtract the sample means from the simulated x and y; and use your function to fit the smoother for some choice of b. Plot the prediction functions over a range of bandwidths large enough to yield noticeable changes in the qualitative behavior of the prediction functions.

[2] Coding tip: write things in a modular way. A kernel function is a function accepting a distance and a bandwidth and returning a nonnegative real. A weighting function is a function that accepts a vector of previous x's, a new x, and a kernel function; and that returns a vector of weights. Et cetera. You will appreciate your foresight in writing modular code later in the course!

Choosing the bandwidth parameter

Cross validation

Again, we're in the one-predictor domain where $y_i = f(x_i) + \epsilon_i$. Left unanswered in the previous problem about linear smoothing was the question: how does one choose the bandwidth b? Assume for now that the goal is to predict well, not necessarily to recover the truth. (These are related but distinct goals.)

(A) Why not just choose b to minimize the sum of squared errors for the observed data set?

(B) Let y be the (unknown) response at some future point x. Let $\hat{f}$ be an estimator of the true function f based on previous data (i.e. not including this x and y). Clearly $y = f(x) + \epsilon$ for some unknown error ϵ, which has mean zero and variance σ^2. Prove the following decomposition for the expected squared error in prediction:

$$E\left\{\left[y - \hat{f}(x)\right]^2\right\} = \sigma^2 + \left[f(x) - E\left\{\hat{f}(x)\right\}\right]^2 + \text{var}\{\hat{f}(x)\}\,.$$

The term inside the expectation on the left-hand side is called the squared error, or risk, of the estimate. All moments are taken under the sampling distribution for the data, given the true function f. Briefly interpret each of the three summands on the right-hand side. You should be able to explain why this is referred to as the *bias–variance decomposition* of an estimator.[3]

[3] Some people call it the bias–variance tradeoff. Why a tradeoff?

(C) "It would be great," you think to yourself, "if I had a fresh data set where I could test my predictions arising from the first, with particular choices of b." Darn right! This is called *out-of-sample predictive validation*.

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 Gaussian kernel smoother to the training data for specified choices of b; and (3) return the estimated functions and the realized prediction error on the testing data for each value of b. Choose the estimate that minimizes the average squared error in prediction:

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

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 a model fit to the training set.

(D) 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.[4] Apply your method from above for each cell in the table, using the testing data to select a bandwidth parameter. Does your out-of-sample predictive validation method lead to reasonable choices of b for each case?

[4] Trigonometric functions, for example, can be pretty wiggly if you make the period small.

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

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

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.[5] 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 b, and for each data point to be held out, fit a whole nonlinear regression model. But you will derive a shortcut!

[5] The intuition here is straightforward: for each possible choice of b, you have to predict each data point using all the others. The bandwidth that with the lowest prediction error is the "best" choice by the LOOCV criterion.

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.[6] 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,

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

[6] Remember that in multiple linear regression this is also true:

$$\hat{y} = X\hat{\beta} = X(X^TX)^{-1}X^Ty = Hy\,.$$

Local polynomial regression

(A) Suppose we observe pairs (x_i, y_i) for $i = 1, \ldots, n$ from our new favorite model, $y_i = f(x_i) + \epsilon_i$ and wish to estimate the value of the underlying function $f(x)$ at some point x by weighted least squares. This is the scalar[7] quantity

$$\hat{f}(x) = a = \arg\min_{\mathbb{R}} \sum_{i=1}^{n} w_i (y_i - a)^2 .$$

[7] Because we are only talking about the value of the function at a specific point x, not the whole function.

Clearly if $w_i = 1/n$, the estimate is simply $\bar{y}$, the sample mean, which is the "best" globally constant estimator. Show (using elementary calculus) that if the weights are

$$w_i \equiv w(x, x_i) = \frac{K\left(\frac{x_i - x}{b}\right)}{\sum_{j=1}^{n} K\left(\frac{x_j - x}{b}\right)} ,$$

then the solution is exactly the kernel-regression estimator. This gives a nice interpretation of kernel regression as locally constant estimator, obtained from locally weighted least squares.

(B) A natural generalization of locally constant regression is local polynomial regression. For points u in a neighborhood of the target point x, define the polynomial

$$g_x(u; a) = a_0 + \sum_{k=1}^{D} a_j (u - x)^k$$

for some vector of coefficients $a = (a_0, \ldots, a_D)$. As above, we will estimate the coefficients a in $g_x(u; a)$ at some target point x using weighted least squares:

$$\hat{a} = \arg\min_{R^{D+1}} \sum_{i=1}^{n} w_i \left\{ y_i - g_x(x_i; a) \right\}^2 ,$$

where $w_i \equiv w(x_i, x)$ are kernel weights defined just above. (Note that $\hat{a}$ obviously depends on the target point x, but we have suppressed this dependence for notational ease.) Derive a concise (matrix) form of the weight vector $\hat{a}$, and by extension, the local function estimate $\hat{f}(x)$ at the target value x.[8] Life will be easier if you define the matrix R_x whose (i, j) entry is $(x_i - x)^{j-1}$, and remember that (weighted) polynomial regression is the same thing as (weighted) linear regression with a polynomial basis.

[8] Observe that at x, $g_x(u; a) = a_0$.

(C) From this, conclude that for the special case of the local linear estimator ($D = 1$), we can write $\hat{f}(x)$ as a linear smoother of the

form

$$\hat{f}(x) = \frac{\sum_{i=1}^{n} w_i y_i}{\sum_{i=1}^{n} w_i},$$

where the unnormalized weights are

$$\begin{aligned} w_i &= K\left(\frac{x - x_i}{h}\right)\{s_2(x) - (x_i - x)s_1(x)\} \\ s_j(x) &= \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)(x_i - x)^j. \end{aligned}$$

(D) What are the mean and variance of the sampling distribution for the local polynomial estimate $\hat{f}(x)$? (Again, this is just a scalar quantity at x, not the whole function.)

(E) Write a new R function that fits the local linear estimator using a Gaussian kernel. Write another wrapper function that uses leave-one-out cross-validation to choose a bandwidth parameter across a grid of possible choices.

Load the data in "utilities.csv" into R. This data set shows the monthly gas bill (in dollars) for a single-family home in Minnesota, along with the average temperature in that month (in degrees F), and the number of billing days in that month. Let y be the average daily gas bill in a given month, and let x be the average temperature. Fit y versus x using both the kernel-regression (locally constant) and local linear methods, in each case using a Gaussian kernel and choosing the bandwidth by LOOCV.

(F) Inspect the results of the models you just fit, paying careful attention to the fit at the far-left and far-right ends of the x axis. Do you see a reason a prefer one method to the other?

(G) Use the results of Part D to construct an approximate point-wise 95% confidence interval for the local linear model for the value of the function at each of the observed points x_i for the utilities data.

A Bayesian version of linear smoothing

A *Gaussian process* is a collection of random variables $\{f(x) : x \in \mathcal{X}\}$ such that, for any finite collection of indices $x_1, \ldots, x_N \in \mathcal{X}$, the random vector $[f(x_1), \ldots, f(x_N)]^T$ has a multivariate normal distribution. It is a generalization of the multivariate normal distribution to infinite-dimensional spaces. The set $\mathcal{X}$ is called the index set or the state space of the process, and need not be countable.

A Gaussian process can be thought of as a random function defined over $\mathcal{X}$, often the real line or $\mathbb{R}^p$. We write $f \sim \mathrm{GP}(m, C)$ for some mean

function $m : \mathcal{X} \to \mathbb{R}$ and a covariance function $C : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$. These functions define the moments of finite-dimensional marginals of the process, in the sense that

$$E\{f(x_1)\} = m(x_1) \quad \text{and} \quad \text{cov}\{f(x_1), f(x_2)\} = C(x_1, x_2)$$

for all $x_1, x_2 \in \mathcal{X}$. More generally, the random vector $[f(x_1), \ldots, f(x_N)]^T$ has covariance matrix whose (i, j) element is $C(x_i, x_j)$. Typical covariance functions are those that decay as a function of increasing distance between points x_1 and x_2. The notion is that $f(x_1)$ and $f(x_2)$ will have high covariance when x_1 and x_2 are close to each other.

(A) Define the *squared exponential* covariance function as

$$C_{SE}(x_1, x_2) = \tau_1^2 \exp\left\{-\frac{1}{2}\left(\frac{x_1 - x_2}{b}\right)^2\right\} + \tau_2^2 \delta(x_1, x_2)\,,$$

where (b, τ_1^2, τ_2^2) are constants (often called *hyperparameters*), and where $\delta(a, b)$ is the Kronecker delta function that takes the value 1 if $a = b$, and 0 otherwise.

Let's start with the simple case where $\mathcal{X} = [0, 1]$, the unit interval. Write an R function that simulates a mean-zero Gaussian process on $[0, 1]$ under the squared-exponential covariance function. The function will accept as arguments: (1) finite set of points $x_1, \ldots, x_N$ on the unit interval; and (2) a triplet (b, τ_1^2, τ_2^2). It will return the value of the random process at each point: $f(x_1), \ldots, f(x_N)$.

Use your function to simulate (and plot) Gaussian processes across a range of values for b, τ_1^2, and τ_2^2. Try starting with a very small value of τ_2^2 (say, 10^{-6}) and playing around with the other two first. On the basis of your experiments, describe the role of these three hyperparameters in controlling the overall behavior of the random functions that result. What happens when you try $\tau_2^2 = 0$? Why?

(B) Suppose you observe the value of a Gaussian process $f \sim \text{GP}(m, C)$ at points $x_1, \ldots, x_N$. What is the conditional distribution of the value of the process at some new point $x^\star$? For the sake of notational ease simply write the value of the (i, j) element of the covariance matrix as $C_{i,j}$, rather than expanding it in terms of a specific covariance function.

(C) *A preliminary lemma.* Suppose that the joint distribution of two vectors y and θ has the following properties: (1) the conditional distribution for y given θ is multivariate normal, $(y \mid \theta) \sim N(R\theta, \Sigma)$;

and (2) the marginal distribution of θ is multivariate normal, $\theta \sim N(m, V)$. Assume that R, Σ, m, and V are all constants. Prove that the joint distribution of y and θ is multivariate normal. What are its mean and covariance matrix?[9]

[9] Remember your result about linear combinations, and remember that if $a \sim N(b, C)$, then $a = b + e$ where $e \sim N(0, C)$.

(D) Suppose we observe data $y_i = f(x_i) + \epsilon_i$, $\epsilon_i \sim N(0, \sigma^2)$, for some unknown function f. Suppose that the prior distribution for the unknown function is a mean-zero Gaussian process: $f \sim \mathrm{GP}(0, C)$ for some covariance function C. Let $x_1, \ldots, x_N$ denote the previously observed x points. Derive the posterior distribution for the random vector $[f(x_1), \ldots, f(x_N)]^T$, given the corresponding outcomes $y_1, \ldots, y_N$, assuming that you know σ^2. (Use your lemma!)

(E) As before, suppose we observe data $y_i = f(x_i) + \epsilon_i$, $\epsilon_i \sim N(0, \sigma^2)$, for $i = 1, \ldots, N$. Now we wish to predict the value of the function $f(x^\star)$ at some new point $x^\star$ where we haven't seen previous data. Suppose that f has a mean-zero Gaussian process prior, $f \sim GP(0, C)$. Show that the posterior mean $E\{f(x^\star) \mid y_1, \ldots, y_N\}$ is a linear smoother, and derive expressions both for the smoothing weights and the posterior variance of $f(x^\star)$.